行政院國家科學委員會專題研究計畫　成果報告

不確定數據下新排程理論研究專書

計畫類別：個別型計畫
計畫編號：NSC93-2416-H-002-011-
執行期間：93 年 08 月 01 日至 94 年 07 月 31 日
執行單位：國立臺灣大學工商管理學系

計畫主持人：賴聰乾

計畫參與人員：Yuri N. Sotskov

報告類型：精簡報告

處理方式：本計畫可公開查詢

中　華　民　國 94 年 9 月 30 日

. pages pages
segments.
period.
count,
between.
fractions.
11794.
font
of .16666 entries).

STABILITY AND UNCERTAINTY IN SEQUENCING AND SCHEDULING

Yuri SOTSKOV
Nadezhda SOTSKOVA
Tsung-Chyan LAI

To Maya Sotskova and Nazar Sotskov

PREFACE

The idea of our approach to scheduling problem with uncertain numerical data arose in Minsk (Belarus) at the Institute of Engineering Cybernetics of the National Academy of Sciences of Belarus when the first author investigated stability of an optimal schedule. This approach was developed further in Magdeburg (Germany) at the Institute of Mathematical Optimization of the Magdeburg University of Otto fon Guericke where the second author wrote her PhD thesis "Optimal scheduling with uncertainty in the numerical data on the basis of stability analysis".

Of course, the ideas and even scientific results based on the common ideas are only necessary but not sufficient for writing a book. What one needs else in order to write a book is *time*. And such a time (and not only) was ensured by the National Scientific Council of Taiwan (Project NSC). Due to this project, the book was completed in 2005 in Taipei (Taiwan) when and where the first author worked as a visiting Professor at the Department of Industrial and Business Management of the National Taiwan University. It was the second stay of the first author at the National Taiwan University as a visiting Professor. The first one was in 1997 also under support of the National Scientific Council of Taiwan (Project NSC). Both NSC projects were headed by the third author obtaining a lot of theoretical results on the topic of the book together with the first and second authors in the time period from 1997 till 2005.

What is this book about? It is about *sequencing* and *scheduling* (as it follows from the book title). In contrast to many other books and surveys devoted to sequencing and scheduling (see [Bak74, BESW93, Bru95, GLLRK79, CMM67, LLRKS93, LLP97, Pin95a, SH99, TSS94] among others), we focus on *stability* and *uncertainty* of an optimal schedule and scheduling.

One can say that schedules arise everywhere if only people want to organize their activities currently or in the future time. In this book, *schedule* is considered as a mathematical (abstract) object and so it does not matter where the schedule is used. *Scheduling* is considered as optimal *sequencing* of the given activities over *time*.

We can also say that this book is about *time* in the following senses. First, a schedule defines a *time* interval when each activity has to be realized. Second, input data for a scheduling problem may include processing *times*, release *time*, due dates, and other *time* parameters which a decision maker has to take into account while constructing a schedule. Third, scheduling algorithm needs *time* to be realized, and realization *time* is one of the most important attribute of a scheduling algorithm. Now it is clear, that to construct efficient (polynomial-*time*) algorithms is unlikely for the most scheduling problems since they are NP-hard with only a few exceptions. Forth, decision-making in production scheduling has to be realized within a *time*.

In order to outline the approach considered in this book, we discuss a connection between *time* necessary for solving scheduling problem, and *time* used for the practical realization of a schedule. What is the role of *time* in a practical scheduling process? Real *time* is not a resource which we can manage. Moreover, we are forced to consider *time* as independent of us and that having only one direction for changing. So, we can use all the available information about the activities, which are already realized, along with results obtained for the scheduling process, but we cannot look into the future of the realization of the process under consideration.

In this book, we propose to take into account that unit of *time* may have different "price" while solving a scheduling problem. More exactly, we propose to distinguish several *time* intervals in a scheduling process. At the *preliminary stage*, there more *time* for decision-making (and so the "price" of *time* unit may be low), but often there is no enough information to get right (*optimal*) solution. In other words, at the *preliminary stage*, it is necessary to find a solution of the scheduling problem provided that input data are *uncertain*. Later, at the *on-line stage*, there are often available essentially more information for a decision-maker (situation under consideration becomes more *certain*), but there is often no enough *time* for using this information in order to construct an optimal schedule. The main idea of the approach, that we propose in this book, is to be better prepared for the *on-line stage* of scheduling (when the "price" of *time* unit will be high) due to using partial solution obtained at the *preliminary stage* of scheduling (when the "price" of *time* unit was low).

This book describes a method how to deal with *uncertainty* arising in practical scheduling. In contrast to *stochastic model* ([Pin95a]) and *fuzzy model* ([SH99]), we assume that in spite of the *uncertain input data*, the desired *schedule* has to fix the unique place in the desired *sequence* for each activity.

The scheme of the method proposed in this book is as follows. It is proposed to decompose original scheduling problem into two (or more) sequential scheduling problems. At the *preliminary stage*, a set of *potentially* optimal schedules has to be constructed under conditions of the *uncertain* numerical input data. More exactly, only lower and upper bounds for the activity durations are assumed to be known at the *preliminary stage*.

In contrast to stochastic model, the probability distribution of the random activity duration is unknown. For solving scheduling problem with *uncertain* numerical data we propose to use a *stability analysis* of an optimal schedule to variations of numerical input data. Since the "price" of *time* is not high at the *preliminary stage*, one can use even time-consuming algorithms for solving scheduling problem with uncertain numerical input data. At the subsequent stages (such as *on-line stage* or stage of *schedule realization*), it is necessary to choose a schedule which has to be realized in an optimal way, i.e., such a schedule which is optimal for the *actual processing times* of the given activities. Note that to solve scheduling problem at the *on-line stage*, one need only efficient (polynomial-*time*) algorithms.

It should be noted that the method and model described in this book aims to complement but not replace other methods and models to deal with *uncertainty* arising in sequencing and scheduling. E.g., a *stochastic model* is very useful when one has enough information to characterize the probability distributions of random activity durations and there is rather large number of realizations of similar processes, but it may have a limited

significance for a small number of realizations of the process.

We have also to emphasize that in this book random activity duration is due to external forces in contrast to scheduling problems with *controllable durations* (see [CJK98, DK95, IMN87, IN86, Str95]), when the objective is to determine optimally the durations (which are under control of a decision maker) and the schedule at the same time.

Another related yet different model is used for the *hoist scheduling* (see articles [CCP98, LW91] among others). Hoist scheduling problems arise in chemical, electroplating and medical industries, where the objective is to minimize the cycle time of a repetitive process. Due to the nature of the chemical process, the activity durations have to be strictly *controlled* by a decision maker within given lower and upper bounds.

The focus of this book has to be mostly theory oriented. The book is written in the theorem-proof format. Claims are formulated as theorems, lemmas and corollaries. Some claims are printed in italics but have no titles. Each claim is given with complete proof with only a few exceptions for some well-known results which proofs can be found in the most books on scheduling.

The main chapter of this book is Chapter 3 which contains the description od the method for solving scheduling problem with uncertain numerical data. Chapters 1 and 2 are auxiliary. Chapter 1 deals with *stability radius* of an optimal schedule which means the largest independent variations of the activity durations such that this schedule remains optimal. Within the book, we widely use mixed graphs for modeling scheduling environment. Chapter 2 contains results on stability analysis of the optimal and approximate solution of the Boolean programming problem. The results on *stability radius* are used in Chapter 3 for solving scheduling problem with uncertain numerical input data. Chapter 3 present some complexity results for problems considered in Chapters 1, 2 and 3, and includes computational results for those problems from Chapters 1 and 3 that are NP-hard.

Four chapters are written mainly as independent one of another. To this end, a short abstract and the main notations are given at the beginning of each chapter, and a summary, remarks and bibliographic notes are given at the end of each chapter. The independence of the chapters implies some repetitions in definitions, notations and argumentation. In each chapter, we use the notations which are suitable for the problem under consideration. We try to save the most notations (for criterion, stability radius and variable data) to be similar for the whole book. We hope that this will not imply any ambiguity for reader, since cross-references between different chapters of the book are used mainly on the level of final results (theorems, lemmas and corollaries) presented in a chapter.

We would not have been able to write this book in the present form and *in the present time* without many people helped us. During the last more than ten years that we worked on the results presented in this book, we have received support of many people from the United Institute of Informatics Problems of the National Academy of Sciences of Belarus (Minsk), the Magdeburg University of Otto fon Guericke (Magdeburg), and the National Taiwan University (Taipei).

The second author wrote her PhD thesis under the supervision of Prof. Dr. Frank Werner, and it is a great pleasure for us to express our gratitude to him for his very fruitful participating in the scientific research which results are presented in this book. We acknowledge Georgii Andreev from the United Institute of Informatics Problems for

his qualified help in programming and testing our algorithms. We would like to thank Natalja M. Leshchenko from the same institute for her new results on flow shop problem with uncertain numerical data included in Chapter 3 of this book before defending her PhD thesis.

Special thanks go to Nina Sotskova and Ivan Mihov for their enthusiasm, faith and love which were of great help for us during the period of writing this book. We have benefitted in many ways from their advices.

We would like to gratefully acknowledge the collaborators of the Department of Industrial and Business Management of the National Taiwan University for their good attitude and for creating the *optimal* atmosphere for our research and writing this book.

As it was already mentioned, the first and third authors obtained financial support from the National Scientific Council of Taiwan. Along with this support, the first author was partially supported by INTAS (project 03-51-5501) and by ISTC (project B-986).

Minsk, Magdeburg, Taipei
November 2005

Yuri Sotskov
Nadezhda Sotskova
Tsung-Chyan Lai

Contents

Introduction

Sequencing and scheduling problems arise in the most decision-making processes in manufacturing, transport, logistics, computer technology, service industries, education, sport, TV, broadcasting, information-processing, etc. Scheduling concerns the allocation of the given tasks (*jobs*) to the given resources (*machines*) over time. In real life, the machines and jobs may take different forms, e.g., machines in a workshop and operations in a production process, runways at an airport and take-offs and landings, crews at the constructions site and stages in the construction project, processing units in a computing environment and executions of computer programs, teachers at the university and student groups, etc. Jobs may differ one from another by priorities, release times, due dates, processing times and type of machine which can process it. Practical scheduling is a decision-making process that has a goal to reach specified objective via better sequencing jobs, decreasing machine idle times, completing jobs just-in-time, reducing machine setup times, and so on.

Mathematical problems arising in sequencing and scheduling are studied in the framework of *scheduling theory*. Modern scheduling theory contains two main parts, based on deterministic or stochastic models. *Deterministic models* are introduced for scheduling environments (see [All97, BDP96, LLRKS93, TSS94] among many other papers and books) in which the processing time (duration) of each job processed by a machine is supposed to be given in advance (i.e., before applying a scheduling procedure) and assumed to be a *constant* during realization of a schedule. Often in real life however, exact numerical data are not known in advance, and difficulties arise when some job processing times (which were assumed in advance) will vary due to a change in a dynamic environment. Even if all the job processing times are fixed in advance, one is forced to take into account errors within the practical realization of a schedule, the precision of the equipment for calculating the processing times, round-off errors in the calculation of a schedule on the computer, machine breakdowns, additionally arriving jobs with high priority and so on. The inadequacy of a deterministic scheduling problem in modeling real-world situations was emphasized in several publications, e.g., in [ML93, PL94, Pin95a].

More general scheduling settings have been considered using a *stochastic model* (see [CCLe95, pp. 33-59], [Pin95a]), where job processing time is assumed to be a *random variable* with a known probability distribution. However in practice, difficulties may still arise in some scenarios. First, one may not have enough prior information to characterize the probability distribution of a random processing time. Second, even if the probability distributions of all the random processing times are known a priori, these distributions are really useful for a large number of realizations of similar scheduling environments but can be of little practical sense for a unique realization or for a small number of similar

realizations.

In this book, a model of one of the more realistic scheduling scenarios is considered: It is assumed that in a realization of a schedule the job processing time may take any real value between the *lower* and *upper bounds*, which are given before applying a scheduling procedure. Obviously, a deterministic model is a special case of the model under consideration (namely, if lower and upper bounds are given equal for a processing time). The model considered can also be interpreted as a stochastic one under 'strict uncertainty', when there is no sufficient a priori information about the probability distribution of a random processing time (or more precisely, it is only known that the random processing time will fall between the given lower and upper bounds with probability one). In spite of obvious practical importance, the model under such a strict uncertainty attracts a very limited attention in the OR literature so far. Almost all the known results for scheduling problem with uncertain numerical data are presented in this book.

Next, we introduce this model more formally. We consider a multi-stage processing system (for brevity, a shop), which consists of a set of machines $M = \{M_1, M_2, \ldots, M_m\}$ that have to process a set of the given jobs $J = \{J_1, J_2, \ldots, J_n\}$. For a shop under consideration, there are assumed five conditions which follow.

Condition 1: *At any time, each machine $M_k \in M$ either processes one job from set J or is idle.*

Condition 2: *At any time, each job $J_i \in J$ either is processed by one machine from set M, waits for processing or is completed.*

Condition 3: *Machine order $(M_{i_1}, M_{i_2}, \ldots, M_{i_{n_i}})$ for processing each job $J_i \in J$, called (technological) route of job J_i, is fixed before scheduling.*

The processing of a job $J_i \in J$ by a machine $M_{i_k} \in M$ at the *stage* $k \in \{1, 2, \ldots, n_i\}$ of the technological route is called an *operation* denoted as O_{ik}. Let Q^J denote the set of all operations for processing all the jobs from set J: $Q^J = \{Q_{ik} \ : \ J_i \in J, k = 1, 2, \ldots, n_i\}$. Condition 3 means that processing of a job from set J includes processing of the given set of operations in the fixed order, provided that distribution of operations Q^J to machines from set M is fixed via technological routes of jobs J.

If the technological routes are given *the same* for all the jobs from set J, e.g., $(M_1, M_2, \ldots, M_m)$, then we have a **flow shop**, otherwise (if the technological routes may be given *differently* for different jobs), we have a **job shop**. In the former case, each job has to be processed once by each machine, while in the latter case, both repetitions and absence of a machine in the technological route of a job are allowed. In both cases each operation is assigned to a certain machine, and the technological route $(M_{i_1}, M_{i_2}, \ldots, M_{i_{n_i}})$ of job $J_i \in J$ defines linearly ordered operations (a sequence) $(O_{i1}, O_{i2}, \ldots, O_{in_i})$ such that operation O_{ik} has to be processed by machine $M_{i_k} \in M$. For a flow shop, the equality $n_i = m$ holds for each job $J_i \in J$, while in the general case of a job shop, the value n_i may be smaller or larger than m or equal to m for a job J_i from set J. Since a flow shop is special case of a job shop, all the results obtained for a job shop hold for a flow shop as well.

The following condition is also holds for the shop under consideration.

Condition 4: *Preemptions of an operation are forbidden.*

Condition 4 means that in any (feasible) schedule, operation $O_{ij} \in Q^J$ being started at time s_{ij} has to be processed up to its completion time $c_{ij} = s_{ij} + p_{ij}$, where p_{ij} denotes the processing time (duration) of operation O_{ij}.

Let Q_k^J denote the set of all operations from set Q^J, $Q_k^J \subseteq Q^J$, which have to be processed by machine $M_k \in M$. In a *deterministic model*, the processing times p_{ij} are known in advance for all operations O_{ij}, $J_i \in J$, $j = 1, 2, \ldots, n_i$. Therefore, a *schedule* may be defined as a set of starting times s_{ij} (or completion times c_{ij}) of all operations Q^J provided that Conditions 1 - 4 hold. Such a set of starting (or completion) times of operations Q^J defines a unique sequence for processing operations Q_k^J by each machine M_k, $k = 1, 2, \ldots, m$. Thus, a schedule uniquely defines m sequences (unique sequence of operations Q_k^J for each machine $M_k \in M$).

The objective of a scheduling problem is to find such a *schedule* (i.e., to find such m sequences of operations Q_k^J on machines M_k, $k = 1, 2, \ldots, m$) for which the value of the given objective function $\Phi(C_1, C_2, \ldots, C_n)$ is minimal. Hereafter, the equality $C_i = c_{in_i}$ holds, and so C_i is equal to the completion time of job $J_i \in J$.

If function $\Phi(C_1, C_2, \ldots, C_n)$ is a non-decreasing one, such a criterion is called *regular* [LLRKS93]. In scheduling theory, the most popular regular criteria are minimization of maximum flow time (makespan)

$$\Phi(C_1, C_2, \ldots, C_n) = \max\{C_i : J_i \in J\} = \mathcal{C}_{max},$$

and minimization of mean flow time

$$\Phi(C_1, C_2, \ldots, C_n) = \sum_{i=1}^{n} C_i = \sum \mathcal{C}_i.$$

Scheduling problems are usually classified by a triplet $\alpha/\beta/\gamma$ (see [LLRKS93]). The α field describes the machine environment and usually contains a single entry (type of processing system) or double entries (type of system and number of machines). The β field provides details of the processing characteristics and may contain no entries, a single entry, or multiple entries (number of jobs, restriction on the processing times, etc.). The γ field contains the objective function to be minimized and it usually contains a single entry. Using such a three-field notation, the deterministic job shop problems considered in Chapter 1 are denoted by $\mathcal{J}//\mathcal{C}_{max}$ and $\mathcal{J}//\sum \mathcal{C}_i$ for the job shop and by $\mathcal{F}//\mathcal{C}_{max}$ and $\mathcal{F}//\sum \mathcal{C}_i$ for the flow shop. Hereafter, symbols $\mathcal{J}$ and $\mathcal{F}$ are used to indicate job shop and flow shop, respectively.

Job shop problem is NP-hard for all criteria considered in scheduling theory [LLRKS93] even for small number of machines and jobs. For example, the following problems are unary NP-hard: $\mathcal{F}3//\mathcal{C}_{max}$, $\mathcal{F}3//\sum \mathcal{C}_i$, $\mathcal{J}3/p_{ij} = 1/\mathcal{C}_{max}$ and $\mathcal{J}2/p_{ij} \in \{1, 2\}/\mathcal{C}_{max}$ (see [GJS76, LRKB77, LRK79]). Problems $\mathcal{F}/n = 3/\mathcal{C}_{max}$, $\mathcal{F}/n = 3/\sum \mathcal{C}_i$, $\mathcal{J}3/n = 2/\mathcal{C}_{max}$ and $\mathcal{J}3/n = 2/\sum \mathcal{C}_i$ are binary NP-hard (see [BKS99, Sot91a, SS95]). To solve practically a job shop problem even with moderate numbers of machines and jobs one needs to develop very complicated algorithms. In 1963, job shop problem $\mathcal{J}10/n = 10/\mathcal{C}_{max}$ with 10 machines and 10 jobs was published in [FT63] (see input data presented in Table 1). In

spite of the large efforts to solve this test problem optimally, it remains unsolved almost 25 years. The first brunch and bound algorithm capable to solve this problem was published only in 1989 (see [CP89]). The optimal schedule with $\mathcal{C}_{max} = 930$ for this famous problem is given in Table 2.

Table 1: *Job routes and operation durations for test problem* $\mathcal{J}10/n = 10/\mathcal{C}_{max}$

Route of job J_1	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
Operation durations	29	78	9	36	49	11	62	56	44	21
Route of job J_2	M_1	M_3	M_5	M_{10}	M_4	M_2	M_7	M_6	M_8	M_9
Operation durations	43	90	75	11	69	28	46	46	72	30
Route of job J_3	M_2	M_1	M_4	M_3	M_9	M_6	M_8	M_7	M_{10}	M_5
Operation durations	91	85	39	74	90	10	12	89	45	33
Route of job J_4	M_2	M_3	M_1	M_5	M_7	M_9	M_8	M_4	M_{10}	M_6
Operation durations	81	95	71	99	9	52	85	98	22	43
Route of job J_5	M_3	M_1	M_2	M_6	M_4	M_5	M_9	M_8	M_{10}	M_7
Operation durations	14	6	22	61	26	69	21	49	72	53
Route of job J_6	M_3	M_2	M_6	M_4	M_9	M_{10}	M_1	M_7	M_5	M_8
Operation durations	84	2	52	95	48	72	47	65	6	25
Route of job J_7	M_2	M_1	M_4	M_3	M_7	M_6	M_{10}	M_9	M_8	M_5
Operation durations	46	37	61	13	32	21	32	89	30	55
Route of job J_8	M_3	M_1	M_2	M_6	M_5	M_7	M_9	M_{10}	M_8	M_4
Operation durations	31	86	46	74	32	88	19	48	36	79
Route of job J_9	M_1	M_2	M_4	M_6	M_3	M_{10}	M_7	M_8	M_5	M_9
Operation durations	76	69	76	51	85	11	40	89	26	74
Route of job J_{10}	M_2	M_1	M_3	M_7	M_9	M_{10}	M_6	M_4	M_5	M_8
Operation durations	85	13	61	7	64	76	47	52	90	45

In turn, the job shop problem $\mathcal{J}//\Phi$ is a special case of a **general shop** problem $\mathcal{G}//\Phi$, in which *arbitrary* precedence constraints may be given on the set of operations. Hereafter, $\mathcal{G}$ indicates general shop, and Φ denotes any regular criterion: $\Phi = \Phi(C_1, C_2, \ldots, C_n)$. So, a general shop is defined via partially ordered set of the given operations. For a general shop, notion of a job may lose its sense, e.g., for $\mathcal{C}_{max}$ criterion, and one can use more simple notation of operations as follows.

Let $Q = \{1, 2, \ldots, q\}$ denote the set of all operations which have to be processed in a general shop, and Q_k denote the set of all operations from set Q, $Q_k \subseteq Q$, which have to be processed by machine $M_k \in M$. If $i \in Q_k$, then the non-negative real value p_i denotes the processing time of operation i by machine $M_k \in M$.

Since a job shop is a special case of a general shop, we can use the above general shop notations for job shop and flow shop as follows. Let Q^{J_i} denote all the operations belonging to job $J_i \in J$. Using general shop notations, we assume that job J_1 has the operations $\{1, 2, \ldots, n_1\} = Q^{J_1}$, job J_2 has the operations $\{n_1 + 1, n_1 + 2, \ldots, n_1 + n_2\} = Q^{J_2}$, and so on, job J_n has the operations

$$\{\sum_{j=1}^{n-1} n_j + 1, \sum_{j=1}^{n-1} n_j + 2, \ldots, \sum_{j=1}^{n} n_j = q\} = Q^{J_n}.$$

Table 2: *Optimal schedule for test problem* $\mathcal{J}10/n = 10/\mathcal{C}_{max}$

Job sequence on machine M_1	J_9	J_2	J_1	J_7	J_4	J_5	J_{10}	J_8	J_6	J_3
Operation completion time	76	119	148	185	256	262	275	361	408	493
Job sequence on machine M_2	J_4	J_6	J_7	J_{10}	J_9	J_5	J_3	J_8	J_1	J_2
Operation completion time	81	86	132	217	286	308	399	445	523	665
Job sequence on machine M_3	J_6	J_4	J_5	J_8	J_2	J_7	J_{10}	J_9	J_1	J_3
Operation completion time	84	179	193	224	314	327	388	506	532	606
Job sequence on machine M_4	J_6	J_7	J_9	J_5	J_3	J_1	J_2	J_4	J_{10}	J_8
Operation completion time	233	294	370	396	532	568	637	735	787	892
Job sequence on machine M_5	J_4	J_2	J_5	J_6	J_8	J_1	J_9	J_7	J_{10}	J_3
Operation completion time	355	430	499	505	551	617	694	753	877	920
Job sequence on machine M_6	J_6	J_5	J_9	J_7	J_8	J_1	J_{10}	J_3	J_2	J_4
Operation completion time	138	369	421	442	519	628	675	709	799	842
Job sequence on machine M_7	J_7	J_4	J_{10}	J_6	J_9	J_8	J_1	J_2	J_3	J_5
Operation completion time	359	368	395	473	557	645	707	753	842	895
Job sequence on machine M_8	J_4	J_6	J_5	J_9	J_7	J_3	J_1	J_8	J_2	J_{10}
Operation completion time	505	530	579	668	698	721	777	813	885	930
Job sequence on machine M_9	J_6	J_4	J_5	J_{10}	J_7	J_3	J_8	J_9	J_1	J_2
Operation completion time	281	420	441	520	609	699	718	792	836	915
Job sequence on machine M_{10}	J_6	J_2	J_7	J_9	J_{10}	J_5	J_8	J_4	J_3	J_1
Operation completion time	353	441	474	517	593	669	766	788	887	908

After such an enumeration of operations

$$Q^J = \{O_{ij},\ J_i \in J,\ j = 1, 2, \ldots, n_i\}$$

we obtain one-to-one correspondence between operations of sets Q^J and Q (operations of sets Q_k^J and Q_k). E.g., linearly ordered set of operations of job J_1 is denoted as $(O_{1,1}, O_{1,2}, \ldots, O_{1n_1})$ using job shop notations, and that is denoted as $(1, 2, \ldots, n_1)$ using general shop notations. Where it is possible, we use general shop notations (since it is more simple), otherwise, we use job shop notations. Such a principe of 'rational sufficiency' is used within the whole book. Some adjectives which may be omitted are given in parenthesis, e.g., (undirected) edge and (directed) arc.

Assumption, which is usually used in scheduling theory, that durations of all the operations are given in advance (before scheduling) and cannot change during realization of a schedule often restricts usefulness of scheduling theory for practise. In Chapter 1, we present known results on the calculation of the stability radius of an optimal schedule for general and job shops. The *stability radius* denotes the largest quantity of independent variations of the operation durations such that the given schedule remains optimal. The main attention is paid to the results on a stability analysis which are used further in Chapter 3 and Chapter 5. Some other approaches to stability analysis and related results are briefly discussed in the last section of Chapter 1.

Chapter 3 deals with a mathematical model for scheduling scenarios in which the processing time of each operation $i \in Q$ is *uncertain* before applying a scheduling procedure and may take any value between a given *lower bound* $a_i \geq 0$ and an *upper*

bound $b_i \geq a_i$. More precisely, in Chapter 3 we consider the general shop problem when the *structural input data* are fixed, while only a lower bound $a_i \geq 0$ and an upper bound $b_i \geq a_i$ for the processing time of operation $i \in Q$, are given as *numerical input data* before applying a scheduling procedure, i.e., the following condition holds.

Condition 5: *The actual processing time* p_i *of operation* $i \in Q$ *may take any real value between given lower and upper bounds, i.e.*

$$a_i \leq p_i \leq b_i, \ i \in Q. \tag{1}$$

It should be noted that while Conditions 1, 2, 3 and 4 are commonly used in the modern scheduling theory, Condition 5 is rather new for the OR literature. The main aim of this book is to introduce Condition 5 into some settings of scheduling problems.

A general shop problem which satisfies Conditions 1 - 5 is denoted by $\mathcal{G}/a_i \leq p_i \leq b_i/\Phi$. On the one hand, problem $\mathcal{G}/a_i \leq p_i \leq b_i/\Phi$ can be considered as a *stochastic* general shop problem under 'strict uncertainty', when there is no prior information about probability distributions of the random processing times. On the other hand, if $a_i = b_i$ for each operation $i \in Q$, problem $\mathcal{G}/a_i \leq p_i \leq b_i/\Phi$ turns out to be a *deterministic* general shop problem $\mathcal{G}//\Phi$.

Problem $\mathcal{G}/a_i \leq p_i \leq b_i/\Phi$ seems to be rather realistic, at least, it is not restrictive: Even if there is no prior information on the possible perturbations of the processing times p_i, one can consider 0 as lower bound of p_i and a sufficiently large number (e.g., the *planning horizon*) as upper bound for p_i. It should be noted that for a flow shop problem fixing the structural input data means only to fix the number n of jobs and the number m of machines. Consequently, any two flow shop problems with the same number n of jobs and the same number m of machines, i.e., problems $\mathcal{F}m/n=k/\Phi$ may differ one from another only in their processing times.

Since problem $\mathcal{G}//\Phi$ is a special case of problem $\mathcal{G}/a_i \leq p_i \leq b_i/\Phi$, then NP-hardness of problem $\mathcal{G}//\Phi$ implies NP-hardness of problem $\mathcal{G}/a_i \leq p_i \leq b_i/\Phi$. Some non-trivial results concerning complexity of problem $\mathcal{G}/a_i \leq p_i \leq b_i/\Phi$ are presented in Chapter 5. The last chapter includes computational results of the calculation of the stability radii of optimal schedules for randomly generated job shop problems, when the objective is to minimize mean or maximum flow times. We test algorithms coded in Fortran-77 for a stability analysis, in which an optimal schedule has already been constructed and the question is to determine such maximal variations of the operation processing times, which do not destroy schedule optimality. We present also computational results for solving randomly generated problems $\mathcal{J}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ and $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$.

For the convenience of a reader, common notations are summarized in Table 5.11 at the end of Chapter 5 (page 286). Along with common notations, we summarize general shop notations in Table 1.3 at the end of Chapter 1 (page 65). Notations for simple assembly line balancing problem are also given in Table 1.3. Job shop notations are given in Table 4.34 at the end of Chapter 3 (page 256). Notations used for Boolean programming are summarized in Table 2.1 at the end of Chapter 2 (page 105). In Conclusions, we summarize the known results and outline some topics for future research.

Chapter 1

Stability Radius of an Optimal Schedule

The usual assumption that the processing times of all the operations are exactly known before scheduling restricts practical aspects of the modern scheduling theory since it is often not valid for real-world processes. The main part of this chapter (i.e., Sections 1.1 - 1.4) is devoted to the results obtained for the stability analysis of an optimal schedule which may help to extend the significance of scheduling theory for some production scheduling problems.

The terms 'stability analysis', 'sensitivity analysis' or 'postoptimal analysis' are used for the phase of an algorithm at which a solution of an optimization problem has already been found, and additional calculations are performed in order to investigate how this solution depends on the numerical input data. In this chapter, we prove some results on job shop and general shop scheduling problems for the calculation of the stability radius of an optimal schedule, when the objective is to minimize mean or maximum flow time. Stability radius of an optimal schedule denotes the largest quantity of independent variations of the processing times of the operations such that this schedule remains optimal. The extreme values of the stability radius are considered in detail.

The results presented in this chapter on the stability analysis may be considered as an investigation of scheduling problems *with uncertain numerical data*, when the aim is to study the influence of round-off errors or changes of the processing times on the property of a schedule to be optimal. The main reason for performing a stability analysis is that in most practical cases the processing times of the operations are inexact or uncertain before applying a scheduling procedure. In such cases a stability analysis is necessary to investigate the credibility of an optimal schedule at hand.

On the one hand, if possible errors of the processing times are larger than the stability radius of an optimal schedule, this schedule may not be the best in a practical realization and there is not much sense in large efforts to construct an optimal schedule: It may be more advisable to restrict the scheduling procedure to the construction of an approximate or heuristic solution. On the other hand, this is not the case when each possible change of the processing time is less than or equal to the stability radius of an optimal schedule: An a priori constructed optimal schedule will remain optimal (the best) in the practical realization as well.

Another reason for calculating the stability radius is connected with the need to solve a set of similar scheduling problems. In reality the main characteristics of a shop (such as the number of machines, the technological routes, the range of variations of the processing times and so on) do not change quickly, and it may be possible to use previous computations of an optimal schedule for solving a new similar scheduling problem.

Since the majority of scheduling problems is NP-hard, enumeration schemes such as branch-and-bound are often used for finding an optimal schedule. To this end, it is necessary to construct a *solution tree*, which is often huge. Unfortunately, most of the information contained in the solution tree, is lost after having solved the problem. In such a situation the stability radius of the optimal schedule constructed gives the possibility to use a part of this information for solving further similar scheduling problems.

It is well-known that different scheduling problems may be represented as extremal problems on mixed (disjunctive) graphs. The only requirement for such a representation is the prohibition of operation preemptions (see Condition 4 at page 3 in Introduction). In Section 1.1, we describe the mixed (disjunctive) graphs to represent the input data of the general shop problem $\mathcal{G}//\Phi$. A small example of a job shop problem illustrates the disjunctive graph approach.

In Section 1.2, we describe some results concerning the calculation of the stability radius of an optimal schedule for problem $\mathcal{G}//\Phi$ via the reduction to a non-linear mathematical programming problem. The calculation of the stability radius along with characterizations of its extreme values for problems $\mathcal{G}//\mathcal{C}_{max}$ and $\mathcal{G}//\sum\mathcal{C}_i$ are considered in Sections 1.3 and 1.4, respectively. In Section 1.5 we derived bounds and exact value of the stability radii of an optimal line balance for simple assembly line balancing problem. In Section 1.6, we survey related approaches to the stability analysis in combinatorial optimization.

1.1 Mixed Graphs for Modeling General Shop

We consider a **general shop** in which the given partially ordered set of operations $Q = \{1, 2, \ldots, q\}$ has to be processed by the set of machines $M = \{M_1, M_2, \ldots, M_m\}$. It is assumed that each operation $j \in Q$ is assigned to one machine from set M, and at any time each machine can process at most one operation (see Condition 1 at page 2). Let p_j denote the processing time (duration) of operation $j \in Q$, and c_j denote the completion time of operation j. Operation preemptions are not allowed (Condition 4 at page 3): If an operation j starts at time s_j, its processing is not interrupted until operation j is completed (up to time $c_j = s_j + p_j$). The problem of finding an optimal schedule minimizing the given objective function Φ of job completion times is denoted as $\mathcal{G}//\Phi$. In this book, only regular criteria are considered, i.e., it is assumed that function $\Phi(C_1, C_2, \ldots, C_n)$ is non-decreasing where C_i is the completion time of job J_i.

Set of operations Q is partially ordered by *precedence constraints* $\rightarrow$. Given two operations $i \in Q$ and $j \in Q$, notation $i \rightarrow j$ means that operation i is a predecessor of operation j. In other words, if $i \rightarrow j$, then inequality

$$c_i + p_j \leq c_j \tag{1.1}$$

must hold for any (feasible) schedule.

Let Q_k be a set of operations processed by machine $M_k \in M$, and $\{Q_k : k = 1, 2, \ldots, m\}$ be a partition of set Q, i.e.

$$Q = \bigcup_{k=1}^{m} Q_k,\ Q_k \neq \emptyset,\ \text{ and }\ Q_k \bigcap Q_l = \emptyset,\ \text{ if }\ k \neq l,\ k = 1, 2, \ldots, m,\ l = 1, 2, \ldots, m.$$

We can say that the above partition defines *capacity constraints*. Since at any time each machine $M_k \in M$ can process at most one operation (Condition 1) and operation preemptions are not allowed (Condition 4), the two inclusions $i \in Q_k$ and $j \in Q_k$ imply one of the following inequalities:

$$c_i + p_j \leq c_j \quad \text{or} \quad c_j + p_i \leq c_i. \tag{1.2}$$

For the case of a **job shop** problem $\mathcal{J}//\Phi$, along with the above partition, set of operations Q is also partitioned into n chains (linearly ordered sets)

$$Q = \bigcup_{i=1}^{n} Q^{J_i},\ Q^{J_i} \neq \emptyset,\ \text{ and }\ Q^{J_i} \bigcap Q^{J_j} = \emptyset,\ \text{ if }\ i \neq j,\ i = 1, 2, \ldots, n,\ j = 1, 2, \ldots, n \tag{1.3}$$

where each chain includes set Q^{J_i} of operations for processing job J_i, $1 \leq i \leq n$. This chain represents the technological route of job J_i. For a job shop all sets Q^{J_i} are a priori known.

For problem $\mathcal{G}//\Phi$, the processing time p_i of each operation $i \in Q$ is fixed before scheduling, and therefore a *schedule* of operations Q on machines M may be defined by the completion times c_i or by the starting times $s_i = c_i - p_i$ of all the operations $i \in Q$.

If the operation processing times are not fixed before scheduling (see Condition 5 at page 6), it is not possible to define s_i and c_i for all operations $i \in Q$. Therefore, in the general case of problem $\mathcal{G}/a_i \leq p_i \leq b_i/\Phi$, the goal is to determine a processing sequence of the set of operations Q_k on each machine $M_k \in M = \{M_1, M_2, \ldots, M_m\}$. Such a set of m sequences satisfying both the given precedence constraints (1.1) and capacity constraints (1.2) may be considered as a *schedule* for problem $\mathcal{G}/a_i \leq p_i \leq b_i/\Phi$. The general shop problem is to find such a schedule, which minimizes the value of the given non-decreasing objective function $\Phi(C_1, C_2, \ldots, C_n)$.

A mixed (or *disjunctive*) graph is often introduced to model a deterministic scheduling problem. We follow this approach and represent the structural input data for a **general shop** problem by means of a mixed graph $G = (Q, A, E)$, where

- set Q of operations is the set of vertices;

- precedence constraints (1.1) are represented by the set of non-transitive (directed, *conjunctive*) arcs A: If operation i has to be processed before operation j starts, i. e., $i \to k$ holds and there is no other operation k that $i \to k$ and $k \to j$, then arc (i, j) has to belong to set A:

$$A = \{(i, j)\ :\ i \to j,\ i \in Q, j \in Q,$$

there is no operation $k \in Q$ such that both $i \to k$ and $k \to j$ hold$\}$;

- capacity constraints (1.2) are represented by the set E of (undirected) edges $[i,j]$ connecting unordered operations i and j, which have to be processed by the same machine:

$$E = \{[i,j] \ : \ i \in Q_k, \ j \in Q_k, \quad k = 1,2,\ldots,m, \quad i \not\to j, \ j \not\to i\}.$$

For a deterministic problem $\mathcal{G}//\Phi$, the processing times p_i of all operations $i \in Q$ are known before scheduling, and we can associate a non-negative weight p_i with each vertex $i \in Q$ in the mixed graph $G = (Q, A, E)$ constructed for problem $\mathcal{G}//\Phi$. As a result we obtain the *weighted* mixed graph $G(p) = (Q(p), A, E)$, which represents both the structural and numerical input data for general shop scheduling problem $\mathcal{G}//\Phi$.

For solving problem $\mathcal{G}//\Phi$ using mixed graph G, it is necessary to replace edge $[i,j] \in E$ by an arc incident to the same vertices i and j. Indeed due to Condition 1, if edge $[i,j]$ belongs to set E, then for pair of operations i and j there exist two possibilities: To complete operation $i \in Q_k$ before operation $j \in Q_k$ starts on their common machine $M_k \in M$ and to provide the first inequality from (1.2) (in this case edge $[i,j]$ has to be replaced by arc (i,j)), or to complete operation j before operation i starts and to provide the second inequality from (1.2) (in this case edge $[i,j]$ has to be replaced by arc (j,i)).

In scheduling theory, for the same purposes as a mixed graph $G = (Q, A, E)$, so-called *disjunctive graph* is often used. We denote

$$E^* = \bigcup_{[i,j] \in E} \{(i,j),(j,i)\}.$$

The term 'disjunctive graph' is associated with the choice of one of the above two possibilities for each pair of arcs $\{(i,j),(j,i)\} \subseteq E^*$ called *disjunctive arcs*. It means that for solving problem $\mathcal{G}//\Phi$ using disjunctive graph (Q, A, E^*) one of these arcs must be added to a subset $E_s \subset E^*$ of *chosen arcs* and the other one must be rejected from disjunctive graph (Q, A, E^*):

(*) Arc (i,j) belongs to set E_s if and only if $(j,i) \in E^* \backslash E_s$.

In terms of a mixed graph $G = (Q, A, E)$, the above *choice* of arc (i,j) from set E^* is equivalent to *replacement* edge $[i,j] \in E$ by arc (i,j). If such a *replacement* (*choice*, respectively) is done for each edge $[i,j] \in E$ (for each pair of disjunctive arcs $\{(i,j),(j,i)\} \subset E^*$), then we obtain the same set of arcs E_s. As a result, mixed graph $G = (Q, A, E)$ (disjunctive graph (Q, A, E^*)) is transformed to the same resulting digraph $G_s = (Q, A \cup E_s, \emptyset)$.

Not each of such subsets E_s may be feasible for constructing a schedule for problem $\mathcal{G}//\Phi$ since set of chosen arcs E_s may cause a contradiction, i.e., arc set $A \cup E_s$ may implies that 'some operations have to be started before they are completed'. Every time an arc (i,j) is chosen, all disjunctive arcs that are made redundant due to the transitive and antireflexive nature of the precedence relations, should be excluded. It is easy to convince that a *feasible* schedule s is defined by a subset $E_s \subset E^*$ such that along with the above condition (*) the following condition (**) is satisfied:

(**) Digraph $G_s = (Q, A \cup E_s, \emptyset)$ has no circuit.

In what follows, we use mainly terminology based on mixed graph modeling. The adjective 'feasible' is usually omitted before 'schedule'. Digraph $G_s = (Q, A \cup E_s, \emptyset)$ generated from mixed graph $G = (Q, A, E)$ via orienting all the edges of set E is called *feasible* if and only if G_s contains no circuit. Let $\Lambda(G) = \{G_1, G_2, \ldots, G_\lambda\}$ be set of all feasible digraphs $G_s = (Q, A \cup E_s, \emptyset)$. In other words, digraphs $G_s = (Q, A \cup E_s, \emptyset)$ is generated from disjunctive graph $G = (Q, A, E^*)$ and satisfies both conditions (*) and (**). We need the following definition.

Definition 1.1 *A schedule is called semiactive if no operation $i \in Q$ can start earlier without delaying the processing of some other operation from set Q and/or without altering the processing sequence of operations on any of the machines M.*

For solving problem $\mathcal{G}//\Phi$, we can restrict ourselves by considering only set S of all the semiactive schedules since it is easy to prove the following claim.

Lemma 1.1 *If objective function $\Phi(C_1, C_2, \ldots, C_n)$ is non-decreasing one, then it is sufficient to consider only semiactive schedules while solving problem $\mathcal{G}//\Phi$.*

Proof. Let there exist schedule s which is optimal for problem $\mathcal{G}//\Phi$. If schedule s belongs to set S, it is done. Otherwise, we can construct semiactive schedule $s' \in S$ from schedule s by starting each operation as early as possible without delaying the processing of other operations from set Q and without altering the processing sequence of operations on any of the machines M. Since operation preemptions are forbidden, completion time of each operation $i \in Q$ at schedule s' is less or equal to that at schedule s. Since objective function $\Phi(C_1, C_2, \ldots, C_n)$ is non-decreasing, optimality of schedule s implies optimality of semiactive schedule s'.

$\diamond$

Next, we show that there exists a one-to-one correspondence between all semiactive schedules $S = \{1, 2, \ldots, \lambda\}$ constructed for problem $\mathcal{G}//\Phi$ and all circuit-free digraphs $\Lambda(G) = \{G_1, G_2, \ldots, G_\lambda\}$ generated from the mixed graph G by orienting all the edges from set E. In other words, we prove the following claim.

Lemma 1.2 *Each feasible digraph $G_s = (Q, A \cup E_s, \emptyset)$, $G_s \in \Lambda(G)$, uniquely defines a semiactive schedule $s \in S$ for problem $\mathcal{G}//\Phi$, and vice versa.*

Proof. On the one hand, given a vector $p = (p_1, p_2, \ldots, p_q)$ of the operation processing times, a feasible digraph $G_s = (Q, A \cup E_s, \emptyset)$, $G_s \in \Lambda(G)$, defines the weighted digraph $G_s(p) = (Q(p), A \cup E_s, \emptyset)$ which uniquely defines the *earliest completion time* $c_i(s)$ of each operation $i \in Q$. Consequently, weighted digraph $G_s(p)$ defines the unique semiactive schedule

$$s = (c_1(s), c_2(s), \ldots, c_q(s)).$$

On the other hand, each semiactive schedule $s \in S$ defines m sequences of operations Q_k on machines M_k, $k = 1, 2, \ldots, m$. These m sequences define unique digraph $G_s(p) \in \Lambda(G)$.

$\diamond$

In what follows, we call digraph G_s and weighted digraph $G_s(p)$ *optimal* if and only if schedule $s \in S$ is *optimal*. Due to Lemma 1.2, we can use a digraph $G_s \in \Lambda(G)$ and an optimal digraph G_s instead of a schedule $s \in S$ and an optimal schedule s. Digraph $G_s \in \Lambda(G)$ uniquely defines a set of m sequences for processing operations Q_k by machine $M_k \in M = \{M_1, M_2, \ldots, M_m\}$, and vice versa. Note that digraph G_s is more appropriate in stability analysis and in solving problem $\mathcal{G}/a_i \leq p_i \leq b_i/\Phi$ since it is 'more stable' than schedule s with respect to variations of operation durations. Note also that the starting and completion times of the operations Q, the value of the objective function and other characteristics of a semiactive schedule s, corresponding to an acyclic weighted digraph $G_s(p)$, can be easily determined using *longest path calculations*.

Given a fixed vector $p = (p_1, p_2, \ldots, p_q)$ of the operation processing times, in order to construct an optimal schedule for problem $\mathcal{G}//\Phi$ using mixed graph model, one may enumerate (explicitly or implicitly) feasible digraphs $G_1(p), G_2(p), \ldots, G_\lambda(p)$ generated by orienting all edges of the mixed graph G and selecting an optimal digraph, i.e., a feasible digraph with minimal value of the objective function. Unfortunately, the number λ of such feasible digraphs (i.e., the number of semiactive schedules) grows exponentially in the edge number $|E|$, and an overall enumeration of feasible digraphs is practically impossible for large numbers of jobs and machines. For example, mixed graph $G = (Q, A, E)$ constructed for test job shop problem $\mathcal{J}10/n = 10/\mathcal{C}_{max}$ presented in Table 1 at page 4 has the following parameters: $|Q| = 100$, $|A| = 90$ and $|E| = 450$. (Hereafter $|Q|$ denotes the number of elements in the set Q.) Therefore, the whole number of digraphs (with and without circuits) generated by this mixed graph is equal to 2^{450}. Of course, the cardinality of set $\Lambda(G)$ of circuit-free digraphs is less than $2^{|E|}$, but a procedure for testing whether a digraph is circuit-free or not takes also running time.

Nevertheless, for our computational experiments presented in Chapter 5, we use an explicit enumeration of feasible digraphs for rather small job shop problems in order to calculate the stability radii for all the optimal schedules.

Although problem $\mathcal{G}//\Phi$ is unary NP-hard for any given regular criterion Φ considered in scheduling theory, the running time of calculating an optimal schedule $s = (c_1(s), c_2(s), \ldots, c_q(s))$ may be restricted by an $O(q^2)$-algorithm after having constructed an optimal digraph $G_s(p)$. Thus, the main difficulty of problem $\mathcal{G}//\Phi$ (in terms of the disjunctive graph approach) is to construct an optimal digraph $G_s = (Q, A \cup E_s, \emptyset)$, i.e., to define the best set E_s of arcs generated by orienting the edges of set E. Due to the particular importance of set E_s, it is called a *signature* of a schedule s. Each feasible digraph $G_s = (Q, A \cup E_s, \emptyset)$ is uniquely defined by its signature, i.e., by set of arcs E_s which replace set of edges E.

As it was noted in [BDP96], the disjunctive graph model "has mostly replaced the solution representation by Gantt charts as described in [Gan19]". We can give the following comments to elaborate this kind of preference. First, while a Gantt chart is useful for the graphical presentation of a particular solution, the mixed graph model is suitable for the whole scheduling process from the initial mixed graph $G(p)$ (representing the input data) until a final digraph G_s (representing a solution $s \in S$) has been found. Second, a Gantt chart is a representation of one particular situation when there are no changes both in the a priori known processing times and in the calculated starting times. However, such a situation is 'ideal' (at least, it occurs rather seldom in real life). Thus, a Gantt chart seems to be more appropriate 'after realization' of the process (when all the processing

times, starting times and completion times are known) while 'before realization' a mixed graph $G(p)$ and a digraph G_s seem to be more useful, since they are stable with respect to possible changes of the above 'times'. Third, while a Gantt chart is simply a picture in the plane, a digraph is a mathematical (i.e., abstract) object and can assume different graphical presentations. In particular, one can view a Gantt chart as a diagram of the weighted digraph $G_s(p)$ in the plane.

Mixed Graph for a Job Shop Problem

Next, we show how a mixed graph model may be introduced in the case of a **job shop** problem, when job operation is presented as O_{ij}.

To present the structural input data for job shop problem $\mathcal{J}//\Phi$, one can also use the following mixed graph (Q^J, A^J, E^J), where

- $Q^J = \{O_{ij} \ : \ J_i \in J, \ j = 1, 2, \ldots, n_i\}$,
- $A^J = \{(O_{ij}, O_{i,j+1}) \ : \ J_i \in J, \ j = 1, 2, \ldots, n_i - 1\}$,
- $E^J = \{[O_{ij}, O_{uv}] \ : \ O_{ij} \in Q_k, \ O_{uv} \in Q_k, \ J_i \neq J_u\}$.

The set of arcs A^J defines *precedence constraints* (i.e., technological routes) as follows. Since each job $J_i \in J$ may be processed by at most one machine from set M at a time (Condition 2 at page 2) and technological route is fixed for each job (Condition 3 at page 2), operation O_{ij} has to be completed before operation $O_{i,j+1}$ $(1 \leq j \leq n_i - 1)$ starts: $c_{ij} \leq s_{i,j+1}$. The route of job $J_i \in J$ defines linearly ordered operations (a sequence) $(O_{i1}, O_{i2}, \ldots, O_{in_i})$. At the stage $k \in \{1, 2, \ldots, n_i\}$ of the technological route of job J_i, operation O_{ik} has to be processed by machine $M_{i_k} \in M$.

The set of edges E^J defines *capacity constraints* as follows. Set Q_k^J is the set of operations which has to be processed by machine $M_k \in M$. Since any machine $M_k \in M$ can process at most one operation at a time (Condition 1 at page 2) and operation preemptions are not allowed (Condition 4 at page 3), operation $O_{ij} \in Q_k^J$ has to precede operation $O_{uv} \in Q_k^J$ or vice versa: $c_{ij} \leq s_{uv}$ or $c_{uv} \leq s_{ij}$.

Since a job shop is a special case of a general shop, one can also use the notations of a general shop for the job shop as well, assuming that the first job J_1 consists of the set of operations $\{1, 2, \ldots, n_1\} = Q^{J_1}$, the second job J_2 of the operations $\{n_1 + 1, n_1 + 2, \ldots, n_1 + n_2\} = Q^{J_2}$, and so on, the last job J_n of the operations $\{\sum_{j=1}^{n-1} n_j + 1, \sum_{j=1}^{n-1} n_j + 2, \ldots, \sum_{j=1}^{n} n_j = q\} = Q^{J_n}$. It is often more convenient to use the following notation for the above operation enumeration. Let $w(i) = \sum_{k=0}^{i-1} n_k$ and $n_0 = 0$. Then job J_1 consists of the ordered operations $(w(1) + 1, w(1) + 2, \ldots, w(1) + n_1)$, job J_2 of the ordered operations $(w(2) + 1, w(2) + 2, \ldots, w(2) + n_2)$, and so on, job J_n of the ordered operations $(w(n) + 1, w(n) + 2, \ldots, w(n) + n_n)$.

Due to this enumeration of operation for problem $\mathcal{J}//\Phi$ one can use the mixed graph $G = (Q, A, E)$ with

$$Q = \bigcup_{i=1}^{n} Q^{J_i}$$

for modeling a job shop similarly as for a general shop. We consider a small example to demonstrate the above notations.

Example 1.1 *Figure 1.1 shows an example of a weighted mixed graph (Q^J, A^J, E^J) constructed for a job shop problem $\mathcal{J}4/n = 3/\Phi$ with three jobs $J = \{J_1, J_2, J_3\}$ and four machines $M = \{M_1, M_2, M_3, M_4\}$. For the example of a job shop, one can use O_{ij} as operation notation. Operation processing times are presented near corresponding vertices in Figure 1.1.*

For this example of problem $\mathcal{J}4/n = 3/\Phi$, set of operations Q is $\{O_{1,1}, O_{1,2}, \ldots, O_{3,3}\}$, job J_1 consists of the ordered set of operations $\{O_{1,1}, O_{1,2}, O_{1,3}\}$, job J_2 of operations $\{O_{2,1}, O_{2,2}\}$, job J_3 of operations $\{O_{3,1}, O_{3,2}, O_{3,3}\}$. Machine M_1 has to process operations $O_{1,1}$ and $O_{3,3}$, machine M_2 operations $O_{1,2}$ and $O_{3,2}$, machine M_3 operations $O_{1,3}$ and $O_{2,2}$, and machine M_4 operations $O_{2,1}$ and $O_{3,1}$.

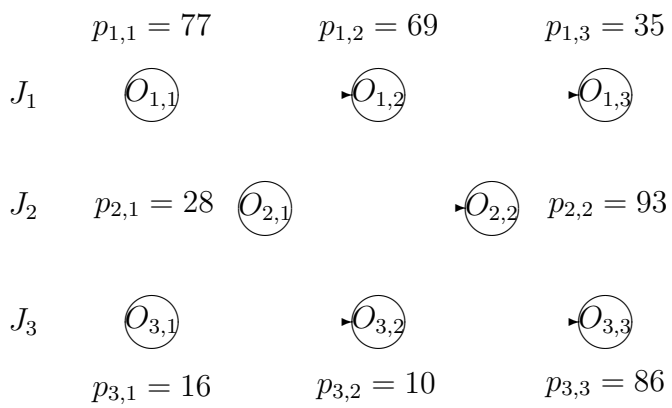

Figure 1.1: *Mixed graph (Q^J, A^J, E^J) and processing times for the problem $\mathcal{J}4/n = 3/\Phi$*

The objective in problem $\mathcal{J}4/n = 3/\Phi$ is to sequence optimally the four sets of operations $Q_1 = \{O_{1,1}, O_{3,3}\}$, $Q_2 = \{O_{1,2}, O_{3,2}\}$, $Q_3 = \{O_{1,3}, O_{2,2}\}$ and $Q_4 = \{O_{2,1}, O_{3,1}\}$.

Since a job shop is a special case of a general shop, one can use general shop notation for this example as well. Thus, set of operations Q^J may be denoted as $Q = \{1, 2, \ldots, 8\}$ using notations for a general shop. Set $\{1, 2, 3\}$ denotes set of operations Q^{J_1} for processing job J_1, $\{4, 5\} = Q^{J_2}$ and $\{6, 7, 8\} = Q^{J_3}$.

There are $2^4 = 16$ possible digraphs which can be generated from the mixed graph $G = (Q, A, E)$ constructed for this example $\mathcal{J}4/n = 3/\Phi$, and 12 of them are feasible since they have no circuits. The maximal weight of a path in the weighted digraph $G_s(p)$ (called critical weight) defines the makespan $\mathcal{C}_{max} = \max\{c_{n_i}(s) : J_i \in J\}$ of schedule $s \in S$. The path in $G_s(p)$ with a critical weight is called a critical path.

It is easy to see that there are two optimal digraphs for makespan criterion $\mathcal{C}_{max}$ with the following length of a critical path (length of an optimal schedule): $\mathcal{C}_{max} = 181$. Thus, this example of problem $\mathcal{J}4/n = 3/\Phi$ has two optimal semiactive schedules for makespan criterion $\Phi = \mathcal{C}_{max}$.

For the mean flow time criterion, this example of problem $\mathcal{J}4/n = 3/\sum \mathcal{C}_i$ has one optimal digraph with the optimal value $\sum \mathcal{C}_i = \sum_{J_i \in J} c_{n_i}(s) = 465$ of the objective function $\Phi = \sum \mathcal{C}_i$.

The digraph $G_1 \in \Lambda(G) = \{G_1, G_2, \ldots, G_{12}\}$, represented in Figure 1.2, is optimal for both criteria $\mathcal{C}_{max}$ and $\sum \mathcal{C}_i$, while digraph $G_2 = (Q, A \cup E_2, \emptyset)$ with the signature

$E_2 = \{(1,8),(5,3),(6,4),(7,2)\}$ is optimal for criterion $\mathcal{C}_{max}$, but it is not optimal for criterion $\sum \mathcal{C}_i$.

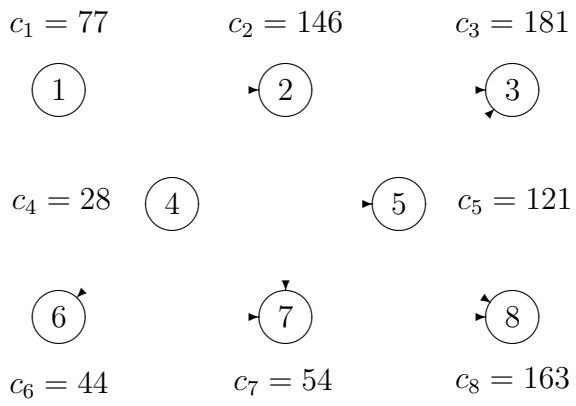

Figure 1.2: *Digraph $G_1 = (Q, A \cup E_1, \emptyset)$ which is optimal for both criteria $\mathcal{C}_{max}$ and $\sum \mathcal{C}_i$*

1.2 Regular Criterion

If operation durations, which are given before applying a scheduling procedure, may vary in the realization of a schedule, it is not enough to construct only an optimal digraph $G_s \in \Lambda(G)$ for solving problem $\mathcal{G}//\Phi$. It is also important to analyze the question of how much the durations of the operations may vary so that the digraph G_s remains optimal. In the following sections of this chapter, we present some results for the stability ball of an optimal digraph $G_s(p)$, i.e., a closed ball in the space of the numerical input data such that within this ball a schedule s remains optimal. Section **??** contains a formal definition of the stability radius, which is the maximal value of the radius of such a stability ball. Example 1.1 is used to demonstrate the notations and results.

In the rest of this chapter the main question is as follows. How can one vary the processing times $p_i, i \in Q$, simultaneously, such that a given schedule $s \in S$, which is optimal for problem $\mathcal{G}//\Phi$ with processing times $p_i, i \in Q$, remains optimal for the new processing times, and how can one calculate the largest quantity of such simultaneous and independent variations of the processing times?

Note that any variation $p_i \pm \epsilon,\ \epsilon > 0$, of a processing time p_i in the problem $\mathcal{G}//\Phi$ implies change of at least one completion time $c_i(s)$ of operation $i \in Q$ in any feasible semiactive schedule $s = (c_1(s), \ldots, c_i(s), \ldots, c_q(s)) \in S$ and, as a result, schedule s has to be transformed to another schedule: $(\ldots, c_i(s) + \epsilon, \ldots)$ or $(\ldots, c_i(s) - \epsilon, \ldots)$ in order to be feasible. However, the optimal digraph $G_s(p) = (Q(p), A \cup E_s, \emptyset)$ for the new problem $\mathcal{G}//\Phi$ obtained due to such a variation of the processing time p_i may remain the same if ϵ is sufficiently small. A signature E_s of an optimal schedule s is essentially more stable. It should be noted that it is often more important to keep in mind not the calendar times when the operations have to be started and have to be completed, but only m optimal sequences in which the operations $Q_k,\ k = 1, 2, \ldots, m$, have to be processed on machine $M_k \in M$.

This chapter is devoted to the stability of an optimal digraph $G_s(p)$ which represents an optimal solution to problem $\mathcal{G}//\Phi$. The above question may be concretized as follows. Under which largest simultaneous and independent changes in the components of the vector $p = (p_1, p_2, \ldots, p_q)$ of operation processing times, digraph $G_s(p)$ remains optimal? Next, we introduce these notions in a formal way.

Let R^q be the space of all q-dimensional real vectors p with the maximum (Chebyshev) metric: The distance $d(p, p')$ between the vectors $p \in R^q$ and $p' = (p'_1, p'_2, \ldots, p'_q) \in R^q$ is defined as follows:

$$d(p, p') = \max_{i \in Q} |p_i - p'_i|, \tag{1.4}$$

where $|p_i - p'_i|$ denotes the absolute value of the difference $p_i - p'_i$.

Let R^q_+ be the space of all q-dimensional non-negative real vectors:

$$R^q_+ = \{x = (x_1, x_2, \ldots, x_q) : \ x_i \geq 0, \ i \in Q\}.$$

Let schedule $s \in S$ be optimal for the problem $\mathcal{G}//\Phi$ with the non-negative real vector $p \in R^q_+ \subset R^q$ of the processing times.

Definition 1.2 *The closed ball $O_\varrho(p)$ with radius $\varrho \in R^1_+$ and center $p \in R^q_+$ in the space of q-dimensional real vectors R^q is called a stability ball of schedule $s \in S$ (of digraph $G_s \in \Lambda(G)$) if for any vector $p' \in O_\varrho(p) \cap R^q_+$ of the processing times schedule s (digraph $G_s(p')$) remains optimal. The maximum value $\varrho_s(p)$ of radius ϱ of a stability ball $O_\varrho(p)$ of schedule s (of digraph G_s) is called the stability radius of schedule s (of digraph G_s):*

$$\varrho_s(p) = \max\{\varrho \in R^1_+ : \textit{If } p' \in O_\varrho(p) \cap R^q_+, \textit{ digraph } G_s \textit{ is optimal}\}.$$

We denote the stability radius by $\varrho_s(p)$ for an arbitrarily given regular criterion. For criterion $\mathcal{C}_{max}$, the stability radius is denoted by $\widehat{\varrho}_s(p)$, and for criterion $\sum \mathcal{C}_i$ by $\overline{\varrho}_s(p)$. In what follows, we use whenever appropriate the notion "stability radius of optimal digraph $G_s \in \Lambda(G)$" instead of "stability radius of optimal schedule $s \in S$".

Due to the maximum metric, the set $O_\varrho(p) \cap R^q_+$ is a polytope for any positive $\varrho \in R^1_+$.

Definition 1.2 implies a general approach for calculating $\varrho_s(p)$, which is discussed in the second part of this section for any regular criterion Φ, and which is concretized for $\Phi = \mathcal{C}_{max}$ and for $\Phi = \sum \mathcal{C}_i$ in Section 1.3 and in Section 1.4, respectively. Formulas for calculating the stability radius for the makespan criterion and the characterization of the extreme values of $\widehat{\varrho}_s(p)$ are proven in Section 1.3. The same questions for the mean flow time criterion are considered in Section 1.4.

Next, calculation of the stability radius $\varrho_s(p)$ is reduced to solving a non-linear programming problem. We give this reduction for the general shop problem $\mathcal{G}//\Phi$ provided that set of all operations $Q = \{1, 2, \ldots, q\}$ is partitioned into n linearly ordered subsets of operations Q^{J_i} defining the technological routes of jobs $J_i \in J = \{J_1, J_2, \ldots, J_n\}$ (see (1.3)). It should be noted that partition (1.3) does not course any restriction on the generality of problem $\mathcal{G}//\Phi$, since one can assume that $|Q^{J_i}| = 1$ for any job $J_i \in J$, i.e., each job may consist of only one operation, and as a result any precedence constraints may be given on the set of operations Q.

We denote by $\{\mu\}$ the set of vertices which form a path μ in the digraph G_k and by $l^p(\mu)$ the weight of this path:

$$l^p(\mu) = \sum_{i \in [\mu]} p_i.$$

Let operation $j_i \in Q^{J_i} \subseteq Q$ be the last operation of job J_i, $1 \leq i \leq n$, in the technological route, and $\tilde{H}_k^i$ denote the set of all paths in the digraph $G_k = (Q, A \cup E_k, \emptyset) \in \Lambda(G)$ ending in vertex $j_i \in Q^{J_i} \subseteq Q$. Obviously, the completion time $C_i(k) = c_{j_i}(k)$ of job J_i at the schedule $k \in S$ is equal to value $\max_{\mu \in \tilde{H}_k^i} l^p(\mu)$ of the largest weight of a path in the set $\tilde{H}_k^i$. While calculating $c_{j_i}(k)$, $J_i \in J$, it is sufficient to consider only a subset of set $\tilde{H}_k^i$ due to the following dominance relation defined on the set of paths.

Definition 1.3 *The path $\mu \in \tilde{H}_s^i$ is called dominant if there is no path $\nu \in \tilde{H}_s^i$ such that $\{\mu\}$ is a proper subset of set $\{\nu\}$: $\{\mu\} \subset \{\nu\}$. Otherwise, if set $\{\mu\}$ is a proper subset of set $\{\nu\}$, path μ is dominated by path ν.*

The dominance relation given in Definition 1.3 is a *strict order* binary relation since *transitivity* and *antireflexivity* hold. Let H_k^i denote the set of all dominant paths in set $\tilde{H}_k^i$. Since $p_i \geq 0$ for all $i \in Q$, we obtain

$$C_i(k) = c_{j_i}(k) = \max_{\mu \in H_k^i} l^p(\mu).$$

Thus, the value of the objective function $\Phi(C_1, C_2, \ldots, C_n)$ for the schedule $s = (c_1(s), c_2(s), \ldots, c_q(s)) \in S$ may be calculated as follows:

$$\Phi(\max_{\mu \in H_s^1} l^p(\mu), \max_{\mu \in H_s^2} l^p(\mu), \ldots, \max_{\mu \in H_s^n} l^p(\mu)).$$

Therefore, a schedule $s = (c_1(s), c_2(s), \ldots, c_q(s)) \in S$ is optimal for problem $\mathcal{G}//\Phi$ with regular criterion Φ if and only if

$$\Phi(\max_{\mu \in H_s^1} l^p(\mu), \max_{\mu \in H_s^2} l^p(\mu), \ldots, \max_{\mu \in H_s^n} l^p(\mu)) = \min_{k=1,2,\ldots,\lambda} \Phi(\max_{\nu \in H_k^1} l^p(\nu), \max_{\nu \in H_k^2} l^p(\nu), \ldots, \max_{\nu \in H_k^n} l^p(\nu)) \tag{1.5}$$

For brevity, we denote

$$\Phi_s^p = \Phi(\max_{\mu \in H_s^1} l^p(\mu), \max_{\mu \in H_s^2} l^p(\mu), \ldots, \max_{\mu \in H_s^n} l^p(\mu)).$$

Let $S^\Phi(p) \subseteq S$ denote the set of all optimal semiactive schedules for problem $\mathcal{G}//\Phi$ with vector $p = (p_1, p_2, \ldots, p_q) \in R_+^q$ of the operation processing times and let inclusion $s \in S^\Phi(p)$ hold. From Definition 1.2 it follows that

$$\varrho_s(p) = \inf\{d(p, x) : x \in R_+^q, s \notin S^\Phi(x)\}. \tag{1.6}$$

From (4.17) and (1.6) it follows that, in order to calculate $\varrho_s(p)$, it is sufficient to calculate the optimal value of the objective function $f(x_1, x_2, \ldots, x_q)$ of the following non-linear programming problem:

$$\text{Minimize } f(x_1, x_2, \ldots, x_q) = \max_{i=1,2,\ldots,q} |x_i - p_i| \tag{1.7}$$

subject to

$$\Phi_s^x > \min\{\Phi_k^x \ : \ k = 1, 2, \ldots, \lambda; \ k \neq s\}, \tag{1.8}$$

$$x_i \geq 0, \quad i = 1, 2, \ldots, q. \tag{1.9}$$

If condition (1.8) is not satisfied for any vector $x \in R^q_+$, then digraph $G_s(p)$ is optimal for all vectors $x \in R^q_+$ of the processing times: $s \in S^\Phi(x)$, $x \in R^q$, and we obtain

$$\begin{cases} \Phi^x_s \leq \min\{\Phi^x_k \ : \ k = 1, 2, \ldots, \lambda; \ k \neq s\}, \\ x_i \geq 0, i = 1, 2, \ldots, q. \end{cases}$$

In this case, we say that the stability radius is *infinitely large*. To indicate that stability radius is infinitely large we write:

$$\varrho_s(p) = \infty.$$

In all other cases, there exists an optimal value f^* of the objective function of problem (1.7)-(1.9):

$$f^* = \inf \max_{i=1,2,\ldots,q} |x_i - p_i|,$$

where the infimum is taken over all vectors x satisfying conditions (1.8) and (1.9). To find the value f^*, it is sufficient to calculate solution $x^0 = (x^0_1, x^0_2, \ldots, x^0_q)$ of the following problem (1.10)-(1.12) which is obtained from problem (1.7)-(1.9) due to replacement the sign $>$ in inequality (1.8) by the sign $\geq$:

$$\text{Minimize } f(x_1, x_2, \ldots, x_q) = \max_{i=1,2,\ldots,q} |x_i - p_i| \tag{1.10}$$

subject to

$$\Phi^x_s \geq \min\{\Phi^x_k \ : \ k = 1, 2, \ldots, \lambda; \ k \neq s\}, \tag{1.11}$$

$$x_i \geq 0, \quad i = 1, 2, \ldots, q. \tag{1.12}$$

It is clear that the following equalities hold

$$f^* = \max_{i=1,2,\ldots,q} |x^0_i - p_i| = d(x^0, p) = \varrho_s(p)$$

and for any small $\epsilon > 0$, there exists a vector $x^\epsilon = (x^\epsilon_1, x^\epsilon_2, \ldots, x^\epsilon_q) \in R^q_+$ such that $d(x^\epsilon, p) = \varrho_s(p) + \epsilon$ and $s \notin S^\Phi(x^\epsilon)$.

It may occur that vector p is itself a solution to the non-linear programming problem (1.10)-(1.12). In the latter case, equalities

$$\varrho_s(p) = d(p, p) = 0$$

hold and it means that the optimal digraph $G_s(p)$ is *unstable*: For any small real $\epsilon > 0$, there exists a vector $p' \in R^q_+$ such that $s \notin S^\Phi(p')$ and $d(p, p') = \epsilon$.

If vector p is not a solution to the problem (1.10)-(1.12), we have $\varrho_s(p) > 0$ and the optimal digraph $G_s(p)$ is *stable*.

1.3 Maximum Flow Time (Makespan)

The best studied case of a general shop problem $\mathcal{G}//\Phi$ (and that of job shop and flow shop problems) is the one with $\Phi = \mathcal{C}_{max}$, where the objective is to find an *optimal* schedule for makespan criterion, i.e., to find a feasible schedule $s = (c_1(s), c_2(s), \ldots, c_q(s))$ with

a minimum value of the maximum flow time $\max\{c_i(s) : i \in Q\}$ among all semiactive schedules S. In this section, the stability radius of a solution to problem $\mathcal{G}//\Phi$ with $\Phi = \mathcal{C}_{max}$ is considered.

Let $\tilde{H}_k$ ($\tilde{H}$, respectively) be the set of all paths in digraph $G_k(p) \in \Lambda(G)$ (in digraph $(Q, A, \emptyset)$) constructed for the general shop problem $\mathcal{G}//\mathcal{C}_{max}$. Let H and H_k denote the set of all dominant paths in digraph $(Q, A, \emptyset)$ and that in digraph $G_k \in \Lambda(G)$, respectively (see Definition 1.3). Thus, we can write

$$H_k = \{\nu \in \tilde{H}_k \ : \ \text{Inclusion } \{\nu\} \subset \{\mu\} \text{ does not hold for any path } \mu \in \tilde{H}_k\}.$$

The set $H \subseteq \tilde{H}$ is defined similarly. The value of $\max_{i=1}^{n} C_i$ of a schedule s is given by the weight of the maximum-weight path (called *critical path*) in the weighted digraph $G_s(p)$. Obviously, at least one critical path in $G_s(p)$ is dominant and for any path $\mu \in H$, there exists a path $\nu \in H_s$ that dominates path μ or inclusion $\mu \in H_s$ holds. Thus, equality (4.17) for problem $\mathcal{G}//\mathcal{C}_{max}$ is converted to the following one:

$$\max_{\mu \in H_s} l^p(\mu) = \min_{k=1,2,\ldots,\lambda} \max_{\nu \in H_k} l^p(\nu). \tag{1.13}$$

Therefore, a schedule $s = (c_1(s), c_2(s), \ldots, c_q(s)) \in S$ is optimal for problem $\mathcal{G}//\mathcal{C}_{max}$ with makespan criterion if and only if equality (1.13) holds.

Zero Stability Radius

Let $H_k(p)$ denote the set of all critical dominant paths in the weighted digraph $G_k(p)$, $G_k \in \Lambda(G)$. Obviously, we have $H_k(p) \subseteq H_k$. To prove the necessary and sufficient conditions for equality $\widehat{\varrho}_s(p) = 0$ we need the following auxiliary claim.

Lemma 1.3 *There exists a real $\epsilon > 0$ such that the set $H_k \backslash H_k(p)$ contains no critical path of digraph $G_k \in P(G)$ for any vector of processing times $p^\epsilon = (p_1^\epsilon, p_2^\epsilon, \ldots, p_q^\epsilon) \in O_\epsilon(p) \cap R_+^q$, i.e.*

$$H_k(p^\epsilon) \subseteq H_k(p).$$

Proof. We calculate the positive real number

$$\epsilon_k = \min_{\nu \in H_k \backslash H_k(p)} \frac{l_k^p - l^p(\nu)}{2q}. \tag{1.14}$$

Hereafter l_k^p denotes *critical weight* of digraph $G_k \in P(G)$ which defines the value of the objective function $\mathcal{C}_{max}$ for schedule k with the vector p of the processing times:

$$l_k^p = \max_{\mu \in H_k} l^p(\mu) \tag{1.15}$$

For any real ϵ, which satisfies the inequalities $0 < \epsilon < \epsilon_k$, the difference in the right side of equality (1.14) remains positive when vector p is replaced by any vector $p^\epsilon \in O_\epsilon(p) \cap R_+^q$. Indeed, the number of vertices in any path ν in digraph G_k is at most equal to q and, therefore, the difference $l_k^p - l^p(\nu)$ may not be 'overcome' by a vector p^ϵ if $d(p, p^\epsilon) < \epsilon_k$.

$\diamond$

Theorem 1.1 *For an optimal schedule $s \in S^{\Phi}(p)$, $\Phi = \mathcal{C}_{max}$, of the problem $\mathcal{G}//\mathcal{C}_{max}$, equality $\widehat{\varrho}_s(p) = 0$ holds if and only if there exists another optimal schedule $k \in S^{\Phi}(p)$, $k \neq s$, and there exists a path $\mu^* \in H_s(p)$ such that there does not exist a path $\nu^* \in H_k(p)$ with $\{\mu^*\} \subseteq \{\nu^*\}$.*

PROOF. *Sufficiency.* We have to show that if the conditions of the theorem are satisfied, then we have $\widehat{\varrho}_s(p) < \epsilon$ for any given $\epsilon > 0$.

To this end, we construct a vector $p^* = (p_1^*, p_2^*, \ldots, p_q^*)$ with the components

$$p_i^* = \begin{cases} p_i + \epsilon^*, & \text{if } i \in \{\mu^*\} \\ p_i & \text{otherwise,} \end{cases}$$

where $\epsilon^* = \min\{\epsilon_k, \epsilon_s, \epsilon\}$ with ϵ_s and ϵ_k defined as in (1.14). If $\max_{\nu \in H_k} l^{p^*}(\nu) = l^{p^*}(\nu^0)$, then due to Lemma 1.3 and the inequalities $p_i^* \geq p_i$, $i \in Q$, we obtain

$$\begin{aligned} l^{p^*}(\nu^0) &= \max_{\nu \in H_k(p)} l^{p^*}(\nu) \\ &= l^p(\nu^*) + \epsilon^* |\{\mu^*\} \cap \{\nu^*\}| = l^p(\mu^*) + \epsilon^* |\{\mu^*\} \cap \{\nu^*\}|. \end{aligned} \tag{1.16}$$

Since $\{\mu^*\} \subseteq \{\nu^*\}$ does not hold for any $\nu^* \in H_k(p)$, the inequality

$$|\{\mu^*\} \cap \{\nu^*\}| < |\{\mu^*\}|$$

holds and we can continue (1.16) in the following way:

$$l^p(\mu^*) + \epsilon^* |\{\mu^*\} \cap \{\nu^*\}| < l^p(\mu^*) + \epsilon^* |\{\mu^*\}| = l^{p^*}(\mu^*) = \max_{\mu \in H_s} l^{p^*}(\mu).$$

Thus, we obtain $l_k^{p^*} < l_s^{p^*}$ and $s \notin S^{\Phi}(p^*)$, which imply $\widehat{\varrho}_s(p) < \epsilon$ because of $d(p, p^*) = \epsilon^* \leq \epsilon$.

Necessity. We prove necessity by contradiction. Suppose that $\widehat{\varrho}_s(p) = 0$ but the conditions of the theorem are not satisfied. We consider two cases i) and ii) of violating these conditions.

i) Assume that there does not exist another optimal makespan schedule: $S^{\Phi}(p) = \{s\}$. Then we calculate the real number

$$\epsilon^0 = \frac{1}{2q} \min\{l_t^p - l_s^p : t = 1, 2, \ldots, \lambda;\ t \neq s\}.$$

Since s is the only optimal makespan schedule, we have $\epsilon^0 > 0$. For any positive real $\epsilon < \epsilon^0$, the difference $l_t^p - l_s^p$ remains strictly positive when vector p is replaced by an arbitrary vector $p^0 \in O_\epsilon(p) \cap R_+^q$. So we can conclude that digraph G_s remains optimal for any such vector p^0 of the processing times. Therefore, we have $\widehat{\varrho}_s(p) \geq \epsilon > 0$ which contradicts the assumption $\widehat{\varrho}_s(p) = 0$.

ii) Assume that $|S^{\Phi}(p)| > 1$ and for any schedule $k \in S^{\Phi}(p)$, $k \neq s$, and for any path $\mu^* \in H_s(p)$, there exists a path $\nu_k^* \in H_k(p)$ such that $\{\mu^*\} \subseteq \{\nu_k^*\}$. In this case we can take any ϵ that satisfies the inequalities

$$0 < \epsilon < \min\Big\{ \min_{k \in \phi^{max}(p)} \epsilon_k,\ \frac{1}{2q} \min\{l_t^p - l_s^p : t = 1, 2, \ldots, \lambda;\ t \notin S^{\Phi}(p)\} \Big\}.$$

From Lemma 1.3, due to inequality $\epsilon < \epsilon_s$, we get equality

$$l_s^{p^0} = \max_{\mu \in H_s(p^0)} l^{p^0}(\mu) = \max_{\mu \in H_s(p)} l^{p^0}(\mu) \tag{1.17}$$

for any vector $p^0 \in O_\epsilon(p) \cap R_+^q$. Since $\epsilon < \epsilon_s$ and $\epsilon < \epsilon_k$, and there exists a path $\nu_k^* \in H_k(p),\ k \in S^\Phi(p),\ k \neq s$, for any path $\mu^* \in H_s(p)$ such that $\{\mu^*\} \subseteq \{\nu_k^*\}$, we obtain the inequality

$$\max_{\mu \in H_s(p)} l^{p^0}(\mu) \leq \max_{\nu \in H_k(p)} l^{p^0}(\nu). \tag{1.18}$$

Thus, due to (1.17) and (1.18), we have

$$l_s^{p^0} \leq \max_{\nu \in H_k(p)} l^{p^0}(\nu) \tag{1.19}$$

for any optimal schedule $k \in S^\Phi(p),\ k \neq s$. Since

$$\epsilon < \frac{1}{2q} \min\{l_t^p - l_s^p : t = 1, 2, \ldots, \lambda; t \notin S^\Phi(p)\},$$

condition $t \notin S^\Phi(p)$ implies $t \notin S^\Phi(p^0)$. Taking into account (1.19) and the latter implication, we can conclude that $s \in S^\Phi(p^0)$ for any vector $p^0 \in R_+^q$ with $d(p, p^0) \leq \epsilon$. Consequently, we have $\widehat{\varrho}_s(p) \geq \epsilon > 0$, which contradicts the assumption $\widehat{\varrho}_s(p) = 0$.

◇

Obviously, the conditions of Theorem 1.1 are violated if $H_s(p) \subseteq H$. Therefore, the following corollary holds.

Corollary 1.1 *If s is an optimal schedule for problem $\mathcal{G}//\mathcal{C}_{max}$ and $H_s(p) \subseteq H$, then $\widehat{\varrho}_s(p) > 0$.*

For the following corollary it is not necessary to know the set $H_s(p)$.

Corollary 1.2 *If s is a unique optimal schedule for problem $\mathcal{G}//\mathcal{C}_{max}$, then $\widehat{\varrho}_s(p) > 0$.*

Infinite Stability Radius

Next, we prove the following characterization of an infinitely large stability radius.

Theorem 1.2 *For an optimal schedule $s \in S^\Phi(p),\ \Phi = \mathcal{C}_{max}$, of the problem $\mathcal{G}//\mathcal{C}_{max}$, the stability radius $\widehat{\varrho}_s(p)$ is infinitely large if and only if for any path $\mu \in H_s \backslash H$ and for any digraph $G_t(p) \in \Lambda(G)$, there exists a path $\nu \in H_t$ such that $\{\mu\} \subseteq \{\nu\}$.*

Proof. *Necessity.* Following the contradiction method, we suppose that $\widehat{\varrho}_s(p) = \infty$ but there exist a path $\mu \in H_s \backslash H$ and a digraph $G_t \in P(G)$ such that for any $\nu \in H_t$ the relation $\{\mu\} \subseteq \{\nu\}$ does not hold. We set $\epsilon' = \max_{i=1}^q p_i$ and consider the vector $p' = (p'_1, p'_2, \ldots, p'_q) \in R_+^q$, where

$$p'_i = \begin{cases} \epsilon', & \text{if } i \in \{\mu\} \\ 0 & \text{otherwise.} \end{cases}$$

For any path $\nu \in H_t$, we have

$$l^{p'}(\nu) = \epsilon' |\{\mu\} \cap \{\nu\}|.$$

Since relation $\{\mu\} \subseteq \{\nu\}$ does not hold, we have $l^{p'}(\nu) < l^{p'}(\mu)$. Therefore, we obtain $l_t^{p'} < l^{p'}(\mu) = l_s^{p'}$ and hence $s \notin S^{\Phi}(p')$. We get a contradiction:

$$\widehat{\varrho}_s(p) < d(p', p) \leq \max_{i=1}^{q} p_i < \infty.$$

Sufficiency. Let ϵ be a positive number as large as desired. We take any vector $p^0 \in O_\epsilon(p) \cap R_+^q$ and suppose that

$$l_s^{p^0} = l^{p^0}(\mu).$$

If $\mu \in H$, then $l^{p^0}(\mu) \leq l_t^{p^0}$ holds for every $t = 1, 2, \ldots, \lambda$. If $\mu \in H_s \backslash H$, then, due to the conditions of the theorem, for any feasible schedule t, there exists a path $\nu \in H_t$ such that $\{\mu\} \subseteq \{\nu\}$. Therefore, we get $l^{p^0}(\mu) \leq l^{p^0}(\nu) \leq l_t^{p^0}$. Thus in both cases we have $s \in S^{\Phi}(p^0)$.

$\diamond$

Directly from the above proof of the necessity we obtain the following corollary which gives a simple upper bound for the stability radius $\widehat{\varrho}_s(p)$.

Corollary 1.3 *If $\widehat{\varrho}_s(p) < \infty$, then $\widehat{\varrho}_s(p) \leq \max_{i=1}^{q} p_i$.*

Due to Theorem 1.2, one can identify a general shop problem $\mathcal{G}//\mathcal{C}_{max}$ whose optimal schedule is implied only by the precedence constraints given on the set of operations Q and by the given distribution of the operations Q to the machines M, but independent from the processing times $p \in R_+^q$ of the operations Q. However, because of the generality of problem $\mathcal{G}//\mathcal{C}_{max}$, it is difficult to check the conditions of Theorem 1.2.

Next, it is shown that for a job shop problem $\mathcal{J}//\mathcal{C}_{max}$, there are necessary and sufficient conditions for $\widehat{\varrho}_s(p) = \infty$ which can be verified in $O(q^2)$ time. To present the latter conditions, we need the following notations.

Let A_k (B_k, respectively) be the set of all operations $i \in Q$ such that $i \to j$ ($j \to i$) and $j \in Q_k$, $i \notin Q_k$:

$$A_k = \{i : i \to j, i \in Q \cap Q_k, j \in Q_k\}$$
$$B_k = \{j : i \to j, j \in Q \cap Q_k, i \in Q_k\}.$$

For a set B of operations, let $n(B)$ denote the number of jobs in the set J having at least one operation in the set B.

Theorem 1.3 *For the problem $\mathcal{J}//\mathcal{C}_{max}$, there exists an optimal digraph $G_s(p)$ with an infinitely large stability radius if and only if the following two conditions hold:*
1) inequality $\max\{|A_k|, |B_k|\} \leq 1$ holds for any machine M_k with $n(Q_k) > 1$;
2) if there exist two operations $g \in A_k$ and $f \in B_k$ of job J_l, then there exists a path from f to g in the digraph $(Q, A, \emptyset)$ (possibly $f = g$).

PROOF. *Necessity.* Let $\widehat{\varrho}_s(p) = \infty$, but $|A_k| > 1$. If $n(A_k) > 1$, then there exist at least two jobs J_u and J_v, $u \neq v$, which have to be processed by different machines and after that both jobs have to be processed by the same machine M_k. We can choose operation $i \in Q_k \cap Q^{(u)}$, $f \in A_k \cap Q^{(u)}$, $j \in Q_k \cap Q^{(v)}$, $g \in A_k \cap Q^v$ such that $f \to i$ and $g \to j$ hold. Since $\widehat{\varrho}_s(p) = \infty$, digraph G_s has to be optimal for any vector $p \in R^q_+$.

In particular, digraph G_s has to be optimal for vector p with $p_f = p_j = 1$ and $p_h = 0$ for operations $h \in \{1, 2, ..., q\} \setminus \{f, j\}$. Since operations f and j are processed by different machines, we obtain $\mathcal{C}_{max}(s) = 1$. Hence, arc (j, i) has to belong to set E_s: $(j, i) \in E_s$.

On the other hand, if we consider vector p' with $p'_g = p'_i = 1$ and $p'_h = 1$ for operations $h \in \{1, 2, ..., q\} \setminus \{g, i\}$, then similarly we obtain that arc (i, j) has to belong to set E_s. Thus, we obtain a contradiction, which implies that there is no digraph $G_s \in \Lambda(G)$ which remains optimal for any vector $p \in R^q_+$ of operation processing times. In this case, inequality $\widehat{\varrho}_s(p) < \infty$ must hold for any digraph $G_s \in \Lambda(G)$.

Next, we have to consider the case when $|A_k| > 1$, but $n(A_k) = 1$. In is clear that set of all operations A_k has to belong to the same job, say job J_w.

(a) Since $|Q_k/J| > 1$, there exist operation $j \in Q_k \setminus Q^{J_w}$ and operation $b \in Q^{J_w} \cap Q_k$ such that in digraph $(Q, A, \emptyset)$ there exists a path of the form $(g, ..., b, ..., f)$ with operations $\{g, f\}$, which do not belongs to set Q_k and inclusion $[j, b] \in E$ holds.

Let $G_s \in \Lambda(G)$ be optimal digraph with infinitely large stability radius: $\widehat{\varrho}_s(p) = \infty$. If we set $p_g = p_j = 1$ and $p_h = 0$ for any other $h \in \{1, 2, ..., q\} \setminus \{g, j\}$, then for such vector $p \in R^q_+$ of the operation durations arc (j, b) has to belong to set E_s.

On the other hand, if we consider vector p' with $p'_j = p'_f = 1$ and $p'_h = 0$ for any other $h \in \{1, 2, ..., q\} \setminus \{j, f\}$, then arc (b, j) has to belong to set E_s. Hence, it is impossible to construct digraph $G_s \in \Lambda(G)$ which remains optimal for any vector $p \in R^q_+$ of operation processing times: $\widehat{\varrho}_s(p) < \infty$.

Thus, equality $\widehat{\varrho}_s(p) = \infty$, implies $|A_k| \leq 1$. Similarly we can prove that $\widehat{\varrho}_s(p) = \infty$ implies $|B_k| \leq 1$.

(b) If there exists job $J_l \in J$ such that $A_k \cap Q^{J_l} = g$ and $B_k \cap Q^{J_l} = f$ and there exists path from vertex g to vertex f in the graph $(Q, A, \emptyset)$, then similarly as in the case (a), one can prove that $\widehat{\varrho}_s(p) < \infty$.

Sufficiency. Obviously, the set of machines satisfying the conditions of Theorem 1.3 can be partitioned into the following five subsets:

1) $|A_k| = |B_k| = 0$;
2) $|A_k| = 1$, $|B_k| = 0$;
3) $|A_k| = 0$, $|B_k| = 1$;
4) $|A_k| = |B_k| = 1$ and there exists job J_l such that there exists a path from vertex $f \in B_k \cap Q^{J_l}$ to vertex $g \in A_k \cap Q^{J_l}$ in the digraph $(Q, A, \emptyset)$ (possibly $f = g$);
5) $|A_k| = |B_k| = 1$ and there is no set Q^{J_l}, containing both sets A_k and B_k.

Now we can consider any mixed graph G in which for any machine $M_k \in M$ with $n(Q_k) > 1$ one of the conditions 1) – 5) holds. Then we construct digraph $G_s \in \Lambda(G)$ using the following algorithm.

Algorithm 1.1

Input: Mixed graph $G = (Q, A, E)$ constructed for problem $\mathcal{J}//\mathcal{C}_{max}$.
Output: Digraph G_s with $\widehat{\varrho}_s(p) = \infty$ (if such a digraph exists).

Step 1: **IF** for machine M_k condition 1) holds, **THEN** we orient all edges incident with operations Q_k arbitrarily without circuit appearance.

Step 2: **IF** for machine M_k condition 4) holds, **THEN** we partition set $Q^{J_l} \cap Q^k$ into two subsets R and L, where set R (set L) contains each vertex h, if there exists path from vertex h (vertex g) to vertex g (vertex h) in digraph $(Q, A, \emptyset)$.

Step 3: We orient all edges of set E^k incident to operations Q_k as follows: If edge is incident with vertices from set R, (set L, respectively), then generated arcs are leaving (entering) the vertices from set R (from set L); the remaining edges are oriented arbitrarily without circuit appearance.

Step 4: **IF** for machine M_k one of the conditions 2), 3) or 5) holds, **THEN** all edges incident with Q_k have to be oriented as follows: all edges incident with vertex Q^L, $Q^L \cap A_k \neq \emptyset$, have to be replaced by arcs entering the operations Q^{J_l}; all edges incident with vertex Q^L, $Q^L \cap A_k \neq \emptyset$, have to be replaced by arcs leaving the operations Q^{J_l}; the remaining edges are oriented arbitrarily without circuit appearance. **STOP**.

Obviously, $G_s \in \Lambda(G)$. It is easy to convince that for any path μ in digraph G_s either there exists machine $M_k \in M$ that $\{\mu\} \subseteq Q_k$ or there exists job $J_l \in J$ that $\{\mu\} \subseteq Q^{J_l}$. Therefore, for digraph G_s constructed via Algorithm (1.1) the following equality holds:

$$C_{max} = \max\{\max_k\{\sum_{i \in Q_k} p_i\}, \max_l\{\sum_{i \in Q^{J_l}} p_i\}\}.$$

Consequently, digraph G_s is optimal. This digraph remains optimal for any vector $p \in R^q_+$ of operation durations, and so $\widehat{\varrho}_s(p) = \infty$.

$\diamond$

From Theorem 1.3 it follows that there are job shop problems $\mathcal{J}//\mathcal{C}_{max}$ with an optimal schedule with an infinitely large stability radius for any given number of jobs n and number of machines m. Next, we show that testing conditions of Theorem 1.3 takes $O(q^2)$ time. Let input data given for problem $\mathcal{J}//\mathcal{C}_{max}$ includes set of jobs $J = J_1, J_2, ..., J_n$, sets $Q_1, Q_2, ..., Q_m$ and technological routes of jobs J. For testing inequality $|A_k| > 2$ (inequality $|B_k| > 2$) for each $k = 1, 2, ..., m$ it is sufficient for each operation $i \in Q_k$ to find operation which follows after i. It is easy to see that such testing needs $O(\sum_{k=1}^{n}(q|Q_k|+q))$ time, i.e. $O(q^2)$ time. If $|A_k| \leq 2$ and $|B_k| \leq 2$ for each k, then it is sufficient to consider machine M_k such that equalities $|A_k| = |B_k| = 1$ hold and to test condition of Theorem 1.3 which need $O(q)$ time. Hence, $O(q^2)$ time is sufficient to test whether a problem $\mathcal{J}//\mathcal{C}_{max}$ has a solution with infinitely large stability radius.

Note that for a flow shop problem such a schedule can exist only for trivial case if n or m is equal to 1. Indeed, for the problem $\mathcal{F}//\mathcal{C}_{max}$ we have $n(A_k) > 1$ for any machine M_k with $k \geq 2$ provided that $n \geq 2$ and $m \geq 2$. Thus, Theorem 1.3 implies the following corollary.

Corollary 1.4 *For the problem $\mathcal{F}//\mathcal{C}_{max}$ with $n \geq 2$ and $m \geq 2$, inequality $\widehat{\varrho}_s(p) > 0$ holds.*

In the next section, it is shown that there does not exist an optimal schedule s with $\varrho_s(p) = \infty$ for a problem $\mathcal{J}//\Phi$ with all other regular criteria Φ which are considered in classical scheduling theory.

The above results for infinite stability radius seem to have only theoretical significance. Next, we try to overcome such an opinion using the following two examples.

Example 1.2 *We consider the problem of scheduling traffic-light. First, we want to find an optimal (makespan) schedule for the traffic-light at the intersection of two roads. As usual, a schedule for traffic-light is assumed periodic, and so it is sufficient to consider only one period of such a schedule. For traffic-light scheduling, one can construct the mixed graph $G = (Q, A, E)$ shown at Figure 1.3. Job J_1 consists of a sequence of red, yellow and green signals for the first road (operations 1, 2 and 3, respectively). Job J_2 consists of a sequence of red, yellow and green signals for the second road (operations 4, 5 and 6, respectively). The operations 3 and 4 have to be connected by edge, since in order to cross intersection of two roads the cars (and people) are allowed to move along no more than one road at any time. The operations of the same job have to be connected by arcs consecutively. In order to find a traffic-light schedule of minimal length, we can to solve the following job shop problem $\mathcal{J}3/n = 2/\mathcal{C}_{max}$ with three machines and two jobs. It is assumed that operations 1 and 2 have to be processed by machine M_1, operations 3 and 4 by machine M_2, and operations 5 and 6 by machine M_3.*

It is easy to see that mixed graph $G = (Q, A, E)$ shown at Figure 1.3 satisfies the conditions of Theorem 1.2. Due to Theorem 1.2 there exists optimal schedule s with infinitely large stability radius. Using algorithm 1.1 from the proof of Theorem 1.2, one can construct optimal digraph $G_s = (Q, A \cup (4, 3), \emptyset)$ with $\widehat{\varrho}_s(p) = \infty$.

Hence, it is not necessary to take the operation durations into account while looking for a traffic-light schedule of minimal length for intersection of two roads. However, this is not the case for the intersection of three roads as follows from Example 1.3.

Example 1.3 *Figure 1.4 represents mixed graph $G = (Q, A, E)$ constructed for the traffic-light scheduling problem for the intersection of three roads. In this case, job J_1 and job J_2 are defined similarly as in Example 1.2. Job J_3 consists of a sequence of red, yellow and green signals for the third road (operations 7, 8 and 9, respectively). Operations 3, 4 and 9 have to be pairwise connected by edges in the mixed graph G in order to ensure safe crossing the intersection of three roads. The operations of the same job have to be connected by arcs consecutively. As a result we obtain job shop problem $\mathcal{J}4/n = 3/\mathcal{C}_{max}$ with four machines and three jobs. It is assumed that operations 1 and 2 have to be processed by machine M_1, operations 3, 4 and 9 by machine M_2, operations 5 and 6 by machine M_3, and operations 7 and 8 by machine M_4. It is easy to see that for mixed*

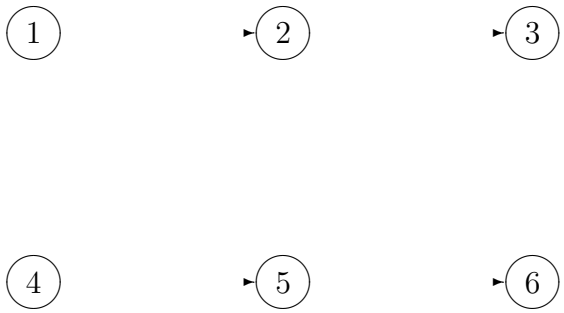

Figure 1.3: *Mixed graph $G = (Q, A, E)$ for Example 1.2*

$G = (Q, A, E)$ conditions of Theorem 1.2 do not hold. Hence for this case, there no exists an optimal schedule with infinitely large stability radius. Inequality $\widehat{\varrho}_s(p) < \infty$ holds for any optimal schedule s constructed for this problem $\mathcal{J}4/n = 3/\mathcal{C}_{max}$.

Thus, for an intersection of three roads, different schedules can be optimal (with makespan criterion) depending on the durations of the red, yellow and green signals of the traffic-light.

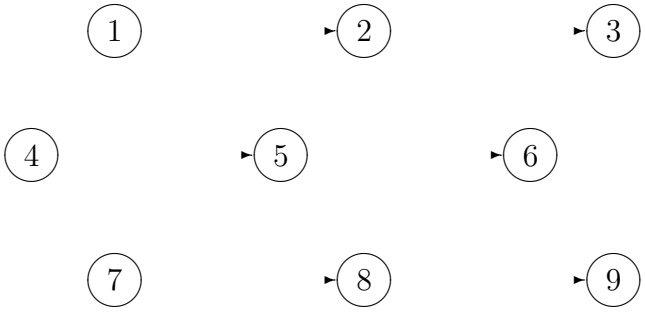

Figure 1.4: *Mixed graph $G = (Q, A, E)$ for Example 1.3*

Formula for Calculating Stability Radius

Next, we derive a formulas for calculating stability radius $\widehat{\varrho}_s(p)$. This calculation is reduced to solving an extremal problem on the set of digraphs $\Lambda(G) = \{G_1, G_2, \ldots, G_\lambda\}$ with a variable vector of weights assigned to the vertices of digraph $G_k \in \Lambda(G)$. The main objects for the calculation are the sets of dominant paths $H_k,\ k = 1, 2, \ldots, \lambda$.

Assume that inequality $\widehat{\varrho}_s(p) < \infty$ holds for a given optimal schedule $s \in S^\Phi(p),\ \Phi = \mathcal{C}_{max}$, of the problem $\mathcal{G}//\mathcal{C}_{max}$. Using equality (1.13), we can conclude that equality (1.6)

for criterion $\Phi = \mathcal{C}_{max}$ is converted to the following one:

$$\widehat{\varrho}_s(p) = \inf\Big\{d(p,x) : x \in R^q_+,\ \max_{\mu\in H_s} l^x(\mu) > \min_{k=1,2,\ldots,\lambda;\ k\neq s}\ \max_{\nu\in H_k} l^x(\nu)\Big\}.$$

Therefore, to find the stability radius $\widehat{\varrho}_s(p)$ it is sufficient to construct a vector $x \in R^q_+$ that satisfies the following three conditions.
(a) There exists a digraph $G_k \in \Lambda(G),\ k \neq s$, such that $l^x_s = l^x_k$, i.e.

$$\max_{\mu\in H_s} l^x(\mu) = \max_{\nu\in H_k} l^x(\nu). \tag{1.20}$$

(b) For any given real $\epsilon > 0$, which may be as small as desired, there exists a vector $p^\epsilon \in R^q_+$ such that $d(x,p^\epsilon) = \epsilon$ and inequality

$$\max_{\mu\in H_s} l^{p^\epsilon}(\mu) > \max_{\nu\in H_k} l^{p^\epsilon}(\nu) \tag{1.21}$$

is satisfied for at least one digraph $G_k \in \Lambda(G)$.
(c) The distance $d(p,x)$ achieves its minimal value among the distances between vector p and the other vectors in the space R^q_+ which satisfy both the above conditions (a) and (b).

After having constructed such a vector $x \in R^q_+$ one can define the stability radius of digraph G_s:

$$\widehat{\varrho}_s(p) = d(p,x),$$

since the critical path in digraph G_s becomes larger than that of digraph G_k for any vector $p^\epsilon \in R^q_+$ with positive real ϵ, which may be as small as possible (see condition (b)), and so digraph G_s is no longer optimal, while for any vector from the intersection of the ball $O_{d(p,x)}(p) \subset R^q$ with the set R^q_+ digraph G_s remains optimal (see condition (c)). Digraph G_k, which satisfies conditions (a), (b) and (c) is called a *competitive* digraph for the optimal digraph G_s.

Thus, the calculation of the stability radius may be reduced to a rather sophisticated extremal problem on the given set of weighted digraphs $\Lambda(G) = \{G_1, G_2, \ldots, G_\lambda\}$ with a variable vector of weights assigned to the vertices of each digraph $G_k \in \Lambda(G)$. As it follows from (1.20) and (1.21), the main objects for such a calculation are the sets of dominant paths $H_k,\ k = 1,2,\ldots,\lambda$.

To satisfy conditions (a), (b) and (c), we look next for a vector $x = p(r) = (p_1(r), p_2(r), \ldots, p_q(r)) \in R^q_+$ with the components $p_i(r) \in \{p_i, p_i + r, p_i - r\}$ on the basis of a direct comparison of the paths from the set H_s and the paths from the sets H_k, where $k = 1,2,\ldots,\lambda$ and $k \neq s$.

Let the value $l^p(\nu)$ be greater than the length of a critical path in an optimal digraph G_s. To satisfy equality (1.20), the length of a path $\nu \in H_k$ has to be not greater than that of at least one path $\mu \in H_s$ and there must be a path $\nu \in H_k$ with a length equal to the length of a critical path in G_s. Thus, if we have calculated

$$r_\nu = \min_{\mu\in H_s} \frac{l^p(\nu) - l^p(\mu)}{|\{\mu\} \cup \{\nu\}| - |\{\mu\} \cap \{\nu\}|}, \tag{1.22}$$

we obtain equality

$$\max_{\mu\in H_s} l^{p(r)}(\mu) = l^{p(r)}(\nu) \tag{1.23}$$

for the vector $p(r) = p(r_\nu)$ with the components

$$p_i(r) = p_i(r_\nu) = \begin{cases} p_i + r_\nu, & \text{if } i \in \{\mu\}, \\ p_i - r_\nu, & \text{if } i \in \{\nu\}\backslash\{\mu\}, \\ p_i, & \text{if } i \notin \{\mu\} \cup \{\nu\}. \end{cases} \tag{1.24}$$

Here the following remark can be given: Due to (1.22) vector $p(r)$ calculated in (1.24) is the closest one to the given vector p among all vectors for which equality (1.23) holds.

To reach equality (1.20) for the whole digraph G_k, we have to repeat calculation (1.22) for each path $\nu \in H_k$ with $l^p(\nu) > l^p_s$. Thus, instead of vector $p(r_\nu)$ we have to consider the vector $p(r) = p(r_{G_k})$ calculated according to formula (1.24), where

$$r_{G_k} = \min_{\mu \in H_s} \max_{\nu \in H_k;\, l^p(\nu) > l^p_s} \frac{l^p(\nu) - l^p(\mu)}{|\{\mu\} \cup \{\nu\}| - |\{\mu\} \cap \{\nu\}|}. \tag{1.25}$$

Next, we consider inequality (1.21). Since the processing times are non-negative, this inequality may not be valid for a vector $p^\epsilon \in R^q_+$ if path μ is dominated by path ν. Thus, we can restrict our consideration to the subset H_{sk} of the set H_s of all paths, which are not dominated by paths from the set H_k:

$$H_{sk} = \Big\{\mu \in H_s : \text{ There is no path } \nu \in H_k \text{ such that } \{\mu\} \subseteq \{\nu\}\Big\}.$$

Hence, we can replace H_s in equality (1.25) by H_{sk}.

To obtain the desired vector $x \in R^q_+$, we have to use equality (1.25) for each digraph $G_k \in \Lambda(G), k \neq s$. Let r denote the minimum of such a value r_{G_k}:

$$r = r_{G_{k^*}} = \min\{r_{G_k} : G_k \in \Lambda(G), k \neq s\}$$

and let $\nu^* \in H_{k^*}$ and $\mu^* \in H_{sk^*}$ be paths at which value $r_{G_{k^*}}$ has been reached:

$$r_{G_{k^*}} = r_{\nu^*} = \frac{l^p(\nu^*) - l^p(\mu^*)}{|\{\mu^*\} \cup \{\nu^*\}| - |\{\mu^*\} \cap \{\nu^*\}|}.$$

Taking into account (1.24), we note that, if $r_{\nu^*} \leq p_i$ for each $i \in \{\nu^*\}\backslash\{\mu^*\}$, vector $p(r) = p(r_{\nu^*})$ does not contain negative components, i.e., $p(r) \in R^q_+$. Due to the remark given after formula (1.24), we have obtained a lower bound of the stability radius:

$$\widehat{\varrho}_s(p) \geq r = \min_{k=1,2,\ldots,\lambda;\, k\neq s} \min_{\mu \in H_{sk}} \max_{\nu \in H_k;\, l^p(\nu) > l^p_s} \frac{l^p(\nu) - l^p(\mu)}{|\{\mu\} \cup \{\nu\}| - |\{\mu\} \cap \{\nu\}|}. \tag{1.26}$$

The bound (1.26) is tight: If $\widehat{\varrho}_s(p) \leq p_i$ for each $i \in \{\nu^*\} \setminus \{\mu^*\}$, then $\widehat{\varrho}_s(p) = r$. For example, we have $\widehat{\varrho}_s(p) = r$ in (1.26) if $\widehat{\varrho}_s(p) \leq \min\{p_i : i \in Q\}$.

To obtain the exact value of $\widehat{\varrho}_s(p)$ in the general case, we construct vector $x = p^*(r) = (p^*_1(r), p^*_2(r), ..., p^*_q(r))$ with the components

$$p^*_i(r) = \begin{cases} p_i + r, & \text{if } i \in \{\mu\}, \\ \max\{0, p_i - r\}, & \text{if } i \in \{\nu\} \setminus \{\mu\}, \\ p_i, & \text{if } i \notin \{\mu\} \cup \{\nu\} \end{cases}$$

instead of vector $p(r)$ defined in (1.24). As it follows from the remark given after formula (1.24), such a vector $p^*(r)$ is the closest one to vector p among all vectors $x \in R^q_+$ which satisfy both conditions (a) and (b).

For calculating the maximal value r for vector $p^*(r)$ we can consider the operations of the set $\{\nu\} \setminus \{\mu\}$ in non-decreasing order of their processing times. Let $p^{\nu\mu}_{(0)}$ be equal to zero and let

$$(p^{\nu\mu}_{(0)}, p^{\nu\mu}_{(1)}, \ldots, p^{\nu\mu}_{(\omega_{\nu\mu})}) \tag{1.27}$$

denote a non-decreasing sequence of processing times of operations from the set $\{\nu\} \setminus \{\mu\}$, where $w_{\nu\mu} = |\{\nu\} \setminus \{\mu\}|$. We obtain the following assertion.

Theorem 1.4 *If G_s is optimal digraph for problem $\mathcal{G}//\mathcal{C}_{max}$, $s \in S^{\Phi}(p)$, $\Phi = \mathcal{C}_{max}$, and inequality $\widehat{\varrho}_s(p) < \infty$ holds, then*

$$\widehat{\varrho}_s(p) = \min_{k=1,2,\ldots,\lambda; k\neq s} \widehat{r}_{ks}, \tag{1.28}$$

where

$$\widehat{r}_{ks} = \min_{\mu\in H_{sk}} \quad \max_{\nu\in H_k,\ l^p(\nu)\geq l^p_s} \quad \max_{\beta=0,1,\ldots,w_{\nu\mu}} \frac{l^p(\nu) - l^p(\mu) - \sum_{\alpha=0}^{\beta} p^{\nu\mu}_{(\alpha)}}{|\{\mu\} \cup \{\nu\}| - |\{\mu\} \cap \{\nu\}| - \beta}. \tag{1.29}$$

Equality (1.28) means that one has to compare an optimal digraph $G_s(p)$ with all other feasible digraphs $G_k(p)$. Note that the formulas in Theorem 1.4 turn into $\widehat{\varrho}_s(p) = \infty$ if $H_{sk} = \emptyset$ for any $k = 1, 2, \ldots, \lambda;\ k \neq s$ (see Theorem 1.2). Moreover, if only a subset of the processing times (say, $P \subseteq \{p_1, p_2, \ldots, p_q\}$) can be changed but the other ones cannot be changed, formula (1.26) and Theorem 1.4 remain valid provided that the difference $|\{\mu\} \cup \{\nu\}| - |\{\mu\} \cap \{\nu\}|$ is replaced by $|\{\{\mu\} \cup \{\nu\}\} \cap P| - |\{\{\mu\} \cap \{\nu\}\} \cap P|$.

In Section 3.3, we show how it is possible to restrict this enumeration and the comparisons (see Corollary 3.5 at page 138). We code these formulas in Fortran-77 (see Chapter 5).

Example 1.4 *Let a job shop problem $\mathcal{J}//\mathcal{C}_{max}$ be specified by the mixed graph $G = (Q, A, E)$ given in Figure. 1.5. Job J_1 consists of operations 1 and 2, and Job J_2 of operations 3, 4 and 5. We have the precedence constraints $1 \to 2$, $3 \to 4$ and $4 \to 5$. The assignment of the operations to the machines is as follows: $Q_1 = \{1, 4\}$, $Q_2 = \{2, 3, 5\}$. The vector $p = (10, 20, 30, 40, 20)$ defines the processing times of the operations $Q = \{1, 2, 3, 4, 5\}$.*

For Example 1.4 we get $\Lambda(G) = \{G_1, G_2, G_3, G_4, G_5\}$ with the following signatures of the semiactive schedules: $E_1 = \{(1, 4), (3, 2), (2, 5)\}$, $E_2 = \{(1, 4), (3, 2), (5, 2)\}$, $E_3 = \{(1, 4), (2, 3), (2, 5)\}$, $E_4 = \{(4, 1), (3, 2), (2, 5)\}$ and $E_5 = \{(4, 1), (3, 2), (5, 2)\}$. The corresponding sets of dominant paths are as follows: $H_1 = \{(1, 2, 5), (3, 2, 5), (1, 4, 5), (3, 4, 5)\}$, $H_2 = \{(1, 4, 5, 2), (3, 4, 5, 2)\}$, $H_3 = \{(1, 2, 3, 4, 5)\}$, $H_4 = \{(3, 4, 1, 2, 5)\}$, $H_5 = \{(3, 4, 1, 2), (3, 4, 5, 2)\}$. The optimal digraph $G_1 = (Q, A \cup E_1, \emptyset)$ is represented in Figure 1.6. It defines the unique optimal semiactive schedule $(10, 50, 30, 70, 90)$.

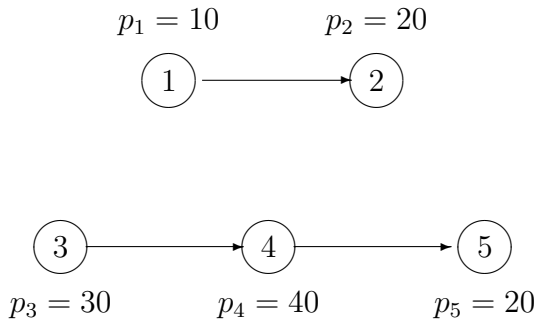

Figure 1.5: *Mixed graph $G = (Q, A, E)$ for Example 1.4*

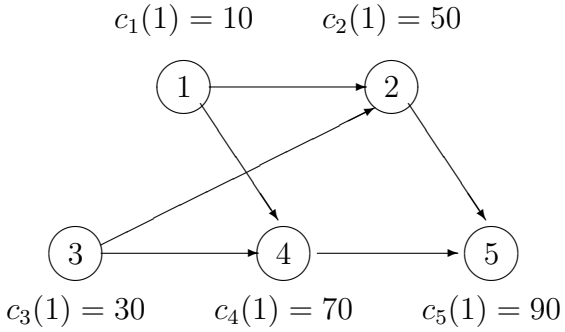

Figure 1.6: *The optimal digraph $G_1 = (Q, A \cup E_1, \emptyset$ for Example 1.4*

Since there exists only one optimal digraph G_1, we conclude that $\widehat{\varrho}_1(p) > 0$ due to Theorem 1.1 or Corollary 1.4. On the other hand, the value $\widehat{\varrho}_1(p)$ cannot be infinitely large. Indeed, there exists the digraph $G_5 \in \Lambda(G)$ and the path $\nu = (3, 4, 1, 2)$ in the set H_5 such that for the path $\mu = (1, 2, 5)$ from the set H_1 the relation $\{\mu\} \subseteq \{\nu\}$ does not hold. Due to Theorem 1.2, we have $\widehat{\varrho}_1(p) < \infty$ and, as a result, the inequalities $0 < \widehat{\varrho}_1(p) \leq \max_{i=1}^{q} p_i$ hold (see Corollary 1.3)).

Next, we calculate exact value of $\widehat{\varrho}_s(p)$ for Example 1.4 using Theorem 1.4. First, we compare the sets H_1 and H_2. Obviously, the paths $(1, 2, 5) \in H_1$ and $(1, 4, 5) \in H_1$ are dominated by the path $(1, 4, 5, 2) \in H_2$. The paths $(3, 2, 5) \in H_1$ and $(3, 4, 5) \in H_1$ are dominated by the path $(3, 4, 5, 2) \in H_2$. Thus, we have $H_{1,2} = \emptyset$. Similarly, one can verify that $H_{1,3} = H_{1,4} = \emptyset$. So for Example 1.4 only the set $H_{1,5}$ is nonempty: $H_{1,5} = \{(1, 2, 5), (1, 4, 5)\}$, and we have to compare the lengths of four pairs of paths. For path $\nu_1 = (3, 4, 1, 2) \in H_5$ and for the two paths from $H_{1,5}$, we have

$$r_{\nu_1} = \min\Big\{\frac{100 - 50}{3}, \frac{100 - 70}{3}\Big\} = 10.$$

For path $\nu_2 = (3, 4, 5, 2) \in H_5$ and for the two paths from $H_{1,5}$, we have

$$r_{\nu_2} = \min\Big\{\frac{110 - 50}{3}, \frac{110 - 70}{3}\Big\} = 13\frac{1}{3}.$$

Thus, we can calculate $r = r_{G_5} = \max\{10, 13\frac{1}{3}\} = 13\frac{1}{3}$, $\nu^* = \nu_2 = (3, 4, 5, 2)$ and $\mu^* = (1, 4, 5)$. Since $r \leq p_i$ holds for each $i \in \{2, 3\} = \{\nu^*\}\backslash\{\mu^*\}$, we conclude that $\widehat{\varrho}_1(p) = r = 13\frac{1}{3}$. In fact, we use here the simpler formula (1.26) as equality $\widehat{\varrho}_s(p) = r$ on

the basis of the above remark about the tightness of the bound (1.26). According to (1.24), we can calculate vector $p' = p(r)$*:*

$$p' = (10 + 13\frac{1}{3}, 20 - 13\frac{1}{3}, 30 - 13\frac{1}{3}, 40 + 13\frac{1}{3}, 20 + 13\frac{1}{3}) = (23\frac{1}{3}, 6\frac{2}{3}, 16\frac{2}{3}, 53\frac{1}{3}, 33\frac{1}{3}),$$

for which we have $l_1^{p'} = l_5^{p'} = 110,\ l_2^{p'} = 116\frac{2}{3},\ l_3^{p'} = l_4^{p'} = 133\frac{1}{3}$*. Thus,* $S^{\Phi}(p') = \{1, 5\}$ *holds (see condition (a)) and for any given small* $\epsilon > 0$ *we can construct the vector* $p^{\epsilon} = (23\frac{1}{3} + \epsilon, 6\frac{2}{3}, 16\frac{2}{3}, 53\frac{1}{3}, 33\frac{1}{3})$ *for which* $S^{\Phi}(p^{\epsilon}) = \{5\}$*. Due to* $H_{1,2} = H_{1,3} = H_{1,4} = \emptyset$*, it is easy to see that condition (c) is satisfied, too.*

Example 1.1 (continued). *Returning to the Example 1.1 presented in Figure 1.1, one can calculate the stability radii* $\widehat{\varrho}_1(p) = 5.75,\ \widehat{\varrho}_2(p) = 1.8$ *for digraph* G_1 *and digraph* G_2 *using formulas 1.28 and 1.29 of Theorem 1.4.*

The equality $\widehat{\varrho}_1(p) = 5.75$ *means that digraph* G_1 *remains optimal for criterion* $\mathcal{C}_{max}$ *if no processing time changes its value by more than* 5.75*. On the other hand, there exist such changes* $p_i \pm (5.75 + \epsilon)$ *of the processing times* $p_i, i \in Q$*, for which* $G_1(p)$ *is no longer optimal and this statement is valid for any small positive real* ϵ*. Obviously, if we have both optimal schedules (the first defined by digraph* G_1 *and the second defined by digraph* G_2*), the first one is preferable for practical use since its stability radius is essentially larger.*

1.4 Mean Flow Time

In this section, we consider stability radius $\overline{\varrho}_s(p)$ of an optimal schedule for problem $\mathcal{G}//\sum \mathcal{C}_i$ with criterion $\sum \mathcal{C}_i$. If $\Phi = \sum \mathcal{C}_i$, conditions (4.17) and (1.6) for the general shop problem $\mathcal{G}//\Phi$ are converted to the following conditions (1.30) and (1.31), respectively.

$$\sum_{i=1}^{n} \max_{\mu \in H_s^i} l^p(\mu) = \min_{k=1,2,\ldots,\lambda} \sum_{i=1}^{n} \max_{\nu \in H_k^i} l^p(\nu), \tag{1.30}$$

$$\overline{\varrho}_s(p) = \inf \Big\{ d(p, x) \ :\ x \in R_+^q,\ \sum_{i=1}^{n} \max_{\mu \in H_s^i} l^x(\mu) > \min_{k=1,2,\ldots,\lambda;\ k \neq s} \sum_{i=1}^{n} \max_{\nu \in H_k^i} l^x(\nu) \Big\}, \tag{1.31}$$

where $H_k^i \subseteq \tilde{H}_k^i$ is the set of all dominant paths in the digraph G_k ending in the *fixed* vertex $u \in Q^{J_i}$ (where $u = w(i) + n_i$ is the last operation of job J_i), and starting from *different* vertices $v \in Q^{J_r}, r = 1, 2, \ldots, n$ (where $v = w(r) + 1$ is the first operation of job J_r).

Obviously, the value C_i for a digraph $G_s(p)$ is equal to the largest weight of a path from the set H_s^i, and hence, to solve problem $\mathcal{G}//\sum \mathcal{C}_i$, it is sufficient to find a digraph $G_s(p)$ such that equality (1.30) holds.

Due to equality (1.31) to find the stability radius $\overline{\varrho}_s(p)$, it is sufficient to construct a vector $x \in R_+^q$ that satisfies the following three conditions.
(a') There exists a digraph $G_k(p) \in \Lambda(G), k \neq s$, such that

$$\sum_{i=1}^{n} \max_{\mu \in H_s^i} l^x(\mu) = \sum_{i=1}^{n} \max_{\nu \in H_k^i} l^x(\nu). \tag{1.32}$$

(b') For any given real $\epsilon > 0$, which may be as small as desired, there exists a vector $p^\epsilon \in R^q_+$ such that $d(x, p^\epsilon) = \epsilon$ and inequality

$$\sum_{i=1}^{n} \max_{\mu \in H_s^i} l^{p^\epsilon}(\mu) > \sum_{i=1}^{n} \max_{\nu \in H_k^i} l^{p^\epsilon}(\nu) \tag{1.33}$$

is satisfied for at least one digraph $G_k(p) \in \Lambda(G)$.
(c') The distance $d(p, x)$ achieves its minimal value among the distances between the vector p and the other vectors in the space R^q_+ which satisfy both above conditions (a') and (b').

Similarly as in the previous section (see conditions (a), (b) and (c)), after having constructed such a vector $x \in R^q_+$ one can define the stability radius of digraph $G_s(p)$: $\overline{\varrho}_s(p) = d(p, x)$. Thus, due to (1.32) and (1.33) the calculation of the stability radius may again be reduced to an extremal problem on the set of weighted digraphs $\Lambda(G)$. However, in this case we are forced to consider sets of representatives of the family of sets $H_k^i,\ 1 \le i \le n$, which may be defined as follows.

Let Ω_k^u be a set of representatives of the family of sets $(H_k^i)_{1 \le i \le n}$. More precisely, the set Ω_k^u includes exactly one path from each set $H_k^i, 1 \le i \le n$. Since $H_k^i \cap H_k^j = \emptyset$ for each pair of different jobs J_i and J_j, we have $|\Omega_k^u| = n$ and there exist $\omega_k = \prod_{i=1}^{n} |H_k^i|$ different sets of representatives for each digraph G_k, namely: $\Omega_k^1, \Omega_k^2, \ldots, \Omega_k^{\omega_k}$. For each set Ω_k^u we can calculate the integer vector $n(\Omega_k^u) = (n_1(\Omega_k^u), n_2(\Omega_k^u), \ldots, n_q(\Omega_k^u))$, where $n_j(\Omega_k^u), j \in Q = \{1, 2, ..., q\}$, is equal to the number of paths in Ω_k^u which includes vertex j. Since a path $\nu \in H_k^i$ includes vertex $j \in Q$ at most once, the value $n_j(\Omega_k^u)$ is equal to the number of copies of vertex j containing in the multiset $\{\{\nu\} : \nu \in \Omega_k^u\}$.

Similarly to the proof of Theorem 1.4, one can find vector x satisfying conditions (a') and (b') in the form $x(r) = (x_1(r), x_2(r), \ldots, x_q(r))$ with the components $x_i(r)$ from the set $\{p_i, p_i + r, p_i - r\}$ on the basis of a direct comparison of the set $\Omega_s^u \in \Omega_{s,k}$ of representatives of the family of sets $(H_s^i)_{1 \le i \le n}$, and the set Ω_k^u of representatives of the family of sets $(H_k^i)_{1 \le i \le n}$, where $k = 1, 2, \ldots, \lambda,\ k \ne s$, and

$$\Omega_{s,k} = \{\Omega_s^v : \text{ There does not exist a set } \Omega_k^u \text{ such that } n_i(\Omega_s^v) \le n_i(\Omega_k^u) \text{ for each } i = 1, 2, \ldots, q\}.$$

As a result, the following lower bound of the stability radius has been obtained:

$$\overline{\varrho}_s(p) \ge r = \min_{k=1,2,\ldots,\lambda;\ k \ne s} \ \min_{\Omega_s^v \in \Omega_{s,k}} \ \max_{u \in \{1,2,\ldots,\omega_k\}} r_{\Omega_k^u, \Omega_s^v}, \tag{1.34}$$

where

$$r_{\Omega_k^u, \Omega_s^v} = \frac{\sum_{\nu \in \Omega_k^u} l^p(\nu) - \sum_{\mu \in \Omega_s^v} l^p(\mu)}{\sum_{i=1}^{q} |n_i(\Omega_k^u) - n_i(\Omega_s^v)|}.$$

This bound is tight. Indeed, we obtain $\overline{\varrho}_s(p) = r$ in (1.34) if $r \le \min\{p_i : i \in Q\}$.

It is easy to see that it may happen that the above vector $x(r) \in R^q$ does not belong to set R^q_+ since substraction value r from some component of vector p may result a negative number. To obtain the exact value of $\overline{\varrho}_s(p)$, one can use the vector $x^*(r) = (x_1^*(r), x_2^*(r), \ldots, x_q^*(r))$ with the components

$$x_i^*(r) = \begin{cases} p_i + r, & \text{if } n_i(\Omega_k^u) < n_i(\Omega_s^v), \\ \max\{0, p_i - r\}, & \text{if } n_i(\Omega_k^u) > n_i(\Omega_s^v), \\ p_i, & \text{if } n_i(\Omega_k^u) = n_i(\Omega_s^v) \end{cases}$$

In contrast to vector $x(r)$, vector $x^*(r)$ necessary has no negative components.

Let the set of operations Q be ordered in the following way:

$$i_1, i_2, \ldots, i_m, i_{m+1}, \ldots, i_q, \tag{1.35}$$

where $n_{i_\alpha}(\Omega_k^u) \leq n_{i_\alpha}(\Omega_s^v)$ for each $\alpha = 1, 2, \ldots, m$ and $n_{i_\alpha}(\Omega_k^u) > n_{i_\alpha}(\Omega_s^v)$ for each $\alpha = m+1, m+2, \ldots, q$. Moreover, for the sequence (1.35) the inequalities

$$p_{i_{m+1}} \leq p_{i_{m+2}} \leq \ldots \leq p_{i_q}$$

have to be satisfied. Using sequence of operations 1.35 it is easy to derive the following formula for calculating $\overline{\varrho}_s(p)$.

Theorem 1.5 *If G_s is an optimal digraph for problem $\mathcal{G}//\sum \mathcal{C}_i$, then*

$$\overline{\varrho}_s(p) = \min_{k=1,2,\ldots,\lambda; k\neq s} \overline{r}_{ks}, \tag{1.36}$$

where

$$\overline{r}_{ks} = \min_{\Omega_s^v \in \Omega_{s,k}} \max_{u=1,2,\ldots,\omega_k} \max_{\beta=0,1,\ldots,q-m} \frac{\sum_{\alpha=1}^{m+\beta} p_{i_\alpha} |n_{i_\alpha}(\Omega_k^u) - n_{i_\alpha}(\Omega_s^v)|}{\sum_{\alpha=1}^{m+\beta} |n_{i_\alpha}(\Omega_k^u) - n_{i_\alpha}(\Omega_s^v)|}. \tag{1.37}$$

If only a subset of the processing times can be changed but the other ones cannot be changed, a formulas similar to (1.36) and (1.37) can be derived (see the remark after Theorem 1.4).

Zero Stability Radius

We consider the case of $\overline{\varrho}_s(p) = 0$. Similarly to the notion of a critical path and the critical weight, which is important for problem $\mathcal{G}//\mathcal{C}_{max}$ (see Section 1.3), we introduce the notion of a critical set of paths $\Omega_k^{u^*}$ and the critical sum of weights of digraph $G_k(p) \in \Lambda(G)$ for problem $\mathcal{G}//\sum \mathcal{C}_i$.

The set $\Omega_k^{u^*}, u^* \in \{1, 2, \ldots, \omega_k\}$, is called *critical set* if the value of the objective function

$$L_k^p = \max_{u\in\{1,2,\ldots,\omega_k\}} \sum_{\nu\in\Omega_k^u} l^p(\nu) \tag{1.38}$$

for the weighted digraph $G_k(p)$ is reached on this set:

$$\sum_{\nu\in\Omega_k^{u^*}} l^p(\nu) = \max_{u\in\{1,2,\ldots,\omega_k\}} \sum_{\nu\in\Omega_k^u} l^p(\nu) = L_k^p.$$

The value L_k^p defined in (1.38) is called *critical sum of weights* for digraph $G_k(p)$.

Obviously, a critical set $\Omega_k^{u^*}$ may include a path $\nu \in H_k^i$, $i = 1, 2, \ldots, n$, if and only if

$$l^p(\nu) = \max_{\mu\in H_k^i} l^p(\mu)$$

and so for different vectors $p \in R_+^q$ of the processing times, different sets Ω_k^u, $u \in \{1, 2, \ldots, \omega_k\}$, may be critical.

Let $\Omega_k(p)$ denote the set of all critical sets $\Omega_k^{u^*}$ of digraph $G_k(p)$ at the vector $p = (p_1, p_2, \ldots, p_q) \in R_+^q$ of the processing times and let Ω_k denote the set $\{\Omega_k^u : u = 1, 2, \ldots, \omega_k\}$. To prove the necessary and sufficient conditions for $\overline{\varrho}_s(p) = 0$, we need the following auxiliary claim.

Lemma 1.4 *There exists a real $\epsilon > 0$ such that the set $\Omega_k \backslash \Omega_k(p)$ contains no critical set of digraph $G_k(p) \in \Lambda(G)$ for any vector $p' \in O_\epsilon(p) \cap R^q_+$ of the processing times, i.e.*

$$\Omega_k(p') \subseteq \Omega_k(p).$$

PROOF. After having calculated the value

$$\epsilon_k = \frac{1}{2qn} \min \Big\{ L^p_k - \sum_{\nu \in \Omega^u_k} l^p(\nu) : \Omega^u_k \in \Omega_k \backslash \Omega_k(p) \Big\}, \tag{1.39}$$

one can verify that for any real ϵ, which satisfies the inequalities $0 < \epsilon < \epsilon_k$, the difference in the right side of equality (1.39) remains positive when vector p is replaced by any vector $p' \in O_\epsilon(p) \cap R^q_+$. Indeed, for any $u \in \{1, 2, \ldots, \omega_k\}$, the cardinality of set Ω^u_k may be at most equal to qn. Thus, the difference $L^p_k - \sum_{\nu \in \Omega^u_k} l^p(\nu)$ may not be 'overcome' by a vector p' if $d(p, p') < \epsilon_k$.

$\diamond$

Next, we prove the following necessary and sufficient conditions for equality $\overline{\varrho}_s(p) = 0$.

Theorem 1.6 *Let G_s be an optimal digraph for problem $\mathcal{G}//\sum \mathcal{C}_i$ with positive processing times $p_i > 0$ of all operations $i \in Q$. The equality $\overline{\varrho}_s(p) = 0$ holds if and only if the following three conditions hold:*
1) there exists another optimal schedule $k \in S^\Phi(p)$, $\Phi = \sum \mathcal{C}_i$, $k \neq s$,
2) there exists a set $\Omega^{v^*}_s \in \Omega_s(p)$ such that for any set $\Omega^{u^*}_k \in \Omega_k(p)$, there exists an operation $i \in Q$ for which the condition

$$n_i(\Omega^{v^*}_s) \geq n_i(\Omega^u_k), \ \Omega^u_k \in \Omega_k(p), \tag{1.40}$$

holds (or the condition

$$n_i(\Omega^{v^*}_s) \leq n_i(\Omega^u_k), \ \Omega^u_k \in \Omega_k(p), \tag{1.41}$$

holds) and
3) inequality (1.40) (or inequality (1.41), respectively) is satisfied as a strict one for the set $\Omega^{u^*}_k$.

PROOF. We prove *necessity* by contradiction. Assume that $\overline{\varrho}_s(p) = 0$ but the conditions of the theorem are not satisfied. We consider three cases (j), (jj) and (jjj) of violating these conditions.

(j) Assume that there does not exist another optimal schedule, i.e., we have $S^\Phi(p) = \{s\}$ with $\Phi = \sum \mathcal{C}_i$. Then we consider a real ϵ such that

$$0 < \epsilon < \frac{1}{2qn} \min_{t \neq s} (L^p_t - L^p_s)$$

holds. Similarly to the proof of Lemma 1.4, we can show that digraph G_s remains optimal for any vector $p^0 = (p^0_1, p^0_2, \ldots, p^0_q) \in R^q_+$ of the processing times provided that $d(p, p^0) \leq \epsilon$. Therefore, we have $\overline{\varrho}_s(p) \geq \epsilon > 0$ which contradicts the assumption $\overline{\varrho}_s(p) = 0$.

(jj) Assume that $|S^\Phi(p)| > 1$ and for any optimal schedule $k \in S^\Phi(p)$ with $k \neq s$, and for any set $\Omega^v_s \in \Omega_s(p)$, there exists a set $\Omega^{u^*}_k \in \Omega_k(p)$ such that $n_i(\Omega^v_s) = n_i(\Omega^{u^*}_k)$ for any operation $i \in Q$.

In this case, we can take any ϵ that satisfies the inequalities

$$0 < \epsilon < \min\Big\{\epsilon_s, \epsilon_k, \frac{1}{2qn} \min_{t \notin \phi^{\Sigma}(p)} (L_t^p - L_s^p)\Big\}. \tag{1.42}$$

From Lemma 1.4, due to inequality $\epsilon < \epsilon_s$, we get that equality

$$L_s^{p^0} = \max_{\Omega_s^v \in \Omega_s(p)} \sum_{\mu \in \Omega_s^v} l^{p^0}(\mu) \tag{1.43}$$

holds for any vector $p^0 \in O_\epsilon(p) \cap R_+^q$. Since there exists a set $\Omega_k^{u^*} \in \Omega_k(p)$ for any set $\Omega_s^v \in \Omega_s(p)$ and any $k \in S^{\Phi}(p),\ k \neq s$, such that $n_i(\Omega_s^v) = n_i(\Omega_k^{u^*}),\ i \in Q$, we obtain the inequality

$$\max_{\Omega_s^v \in \Omega_s(p)} \sum_{\mu \in \Omega_s^v} l^{p^0}(\mu) \leq \max_{\Omega_k^u \in \Omega_k(p)} \sum_{\nu \in \Omega_k^{u^*}} l^{p^0}(\nu),$$

because of $\epsilon < \epsilon_s$ and $\epsilon < \epsilon_k$. Therefore, due to (1.43) we have

$$L_s^{p^0} \leq \max_{\Omega_k^u \in \Omega_k(p)} \sum_{\nu \in \Omega_k^{u^*}} l^{p^0}(\nu) \tag{1.44}$$

for any optimal schedule $k \in S^{\Phi}(p),\ k \neq s$. Since $\epsilon < \frac{1}{2qn} \min_{t \notin S^{\Phi}(p)}\{L_t^p - L_s^p\}$, condition $t \notin S^{\Phi}(p)$ implies $t \notin S^{\Phi}(p^0)$. So taking into account (1.42) and the latter implication, we conclude that $s \in S^{\Phi}(p^0)$ for any vector $p^0 \in R_+^q$ provided that $d(p, p^0) \leq \epsilon$. Consequently, we have $\overline{\varrho}_s(p) \geq \epsilon > 0$, which contradicts the assumption $\overline{\varrho}_s(p) = 0$.

(jjj) Assume that $|S^{\Phi}(p)| > 1$ and for any optimal schedule $k \in S^{\Phi}(p), k \neq s$, and for any set $\Omega_s^v \in \Omega_s(p)$, there exists a set $\Omega_k^{u^*} \in \Omega_k(p)$ such that for any operation $i \in Q$ with $n_i(\Omega_s^v) > n_i(\Omega_k^{u^*})$ there exists a set $\Omega_k^{u^0} \in \Omega_k(p)$ such that $n_i(\Omega_s^v) < n_i(\Omega_k^{u^0})$. Arguing in the same way as in case (jj), we can show that $\overline{\varrho}_s(p) \geq \epsilon > 0$, where ϵ is as in (1.42), since for any vector $p^0 \in O_\epsilon(p) \cap R_+^q$, the value $\sum_{\mu \in \Omega_s^{v^*}} l^{p^0}(\mu)$ is less than or equal to value $\sum_{\nu \in \Omega_k^{u^*}} l^{p^0}(\nu)$ or value $\sum_{\nu \in \Omega_k^{u^0}} l^{p^0}(\nu)$.

Sufficiency. We show that, if the conditions of Theorem 1.6 are satisfied, then $\overline{\varrho}_s(p) < \epsilon$ for any given $\epsilon > 0$.

We construct a vector $p^* = (p_1^*, p_2^*, \ldots, p_q^*) \in R_+^q$ with components $p_i^* \in \{p_i, p_i + \epsilon^*, p_i - \epsilon^*\}$, where $\epsilon^* = \min\{\epsilon_k, \epsilon, \min_{i \in Q} p_i\}$, using the following rule: For each $\Omega_k^{u^*} \in \Omega_k(p)$, mentioned in Theorem 1.6, we set $p_i^* = p_i + \epsilon^*$, if inequalities (1.40) hold, or we set $p_i^* = p_i - \epsilon^*$, if inequalities (1.41) hold. Note that $\epsilon^* > 0$ since $p_i > 0, i \in Q$.

After changing at most $|\Omega_k(p)|$ components of vector p according to this rule, we obtain a vector p^* of processing times for which inequality

$$\sum_{\mu \in \Omega_s^{v^*}} l^{p^*}(\mu) > \sum_{\nu \in \Omega_k^{u^*}} l^{p^*}(\nu)$$

holds for each set $\Omega_k^{u^*} \in \Omega_k(p)$. Due to $\epsilon^* \leq \min_{i \in Q} p_i$, we have $p^* \in R_+^q$. Since $\epsilon^* \leq \epsilon_k$, we have

$$L_k^{p^*} = \max_{u \in \{1,2,\ldots,\omega_k\}} \sum_{\nu \in \Omega_k^u} l^{p^*}(\nu) =$$

$$\max_{\Omega_k^u \in \Omega_k(p)} \sum_{\nu \in \Omega_k^u} l^{p^*}(\nu) = \sum_{\nu \in \Omega_k^{u^*}} l^{p^*}(\nu) < \sum_{\mu \in \Omega_s^{v^*}} l^{p^*}(\mu) \leq L_s^{p^*}.$$

Thus, we conclude that $s \notin S^{\Phi}(p^*)$ with $d(p, p^*) = \epsilon^*$ which implies $\overline{\varrho}_s(p) < \epsilon^* \leq \epsilon$.

$\diamond$

Theorem 1.6 directly implies the following assertion.

Corollary 1.5 *If $s \in S$ is a unique optimal schedule for problem $\mathcal{G}//\sum \mathcal{C}_i$, then $\overline{\varrho}_s(p) > 0$.*

It is easy to prove the following upper bound for the stability radius of an optimal schedule for problem $\mathcal{G}//\sum \mathcal{C}_i$.

Theorem 1.7 *If $s \in S$ is an optimal schedule for problem $\mathcal{G}//\sum \mathcal{C}_i$ with $\lambda > 1$ and $p_i > 0$ for at least one operation $i \in Q$, then*

$$\overline{\varrho}_s(p) \leq \max_{i \in Q} p_i.$$

PROOF. We consider vector $p^0 \in R^q_+$ with zero components: $p^0_i = 0$ for each $i \in Q$. For this vector of processing times, each feasible digraph $G_t \in \Lambda(G)$ is optimal and each set of representatives Ω^u_t is critical. We can take a schedule $k \in S^{\Phi}(p^0)$ which has only one arc $(j, i) \in E_k$ different from the arcs in E_s, i.e. $(i, j) \in E_s$ and $E_s \backslash \{(i, j)\} = E_k \backslash \{(j, i)\}$. It is easy to see that there exist sets $\Omega^v_s \in \Omega_s(p)$ and $\Omega^u_k \in \Omega_k(p)$ such that

$$n_i(\Omega^v_s) > n_i(\Omega^u_k).$$

Setting $p^{\epsilon}_i = \epsilon > 0$ and $p^{\epsilon}_l = 0$ for each $l \in Q \backslash \{i\}$, we obtain $s \notin S^{\Phi}(p^{\epsilon})$ and $d(p^{\epsilon}, p) < \max\{p_i : i \in Q\}$.

$\diamond$

Remark 1.1 As it follows from Theorem 1.7, problem $\mathcal{J}//\sum \mathcal{C}_i$ with $\lambda > 1$ cannot have an optimal schedule with an infinitely large stability radius in contrast to problem $\mathcal{J}//\mathcal{C}_{max}$ and problem $\mathcal{J}//\mathcal{L}_{max}$ (see Section 1.3).

Note that all of the results in this section and in Section **??** are valid for any general shop scheduling problem. However, we use the partition of the set of operations Q into n chains $Q^{(i)}, i = 1, 2, \ldots, n$, (which is necessary for the job shop and flow shop but is not necessary for the general shop) for a better presentation of the results. Table 1.1 collects special cases of the shop scheduling problem which are characterized by the machine service and the technological routes of the jobs. In scheduling theory often the classical job shop is considered for which each job has to be processed exactly once on each machine (see problem in the third row in Table 1.1). For our consideration this restriction is not important. We will consider the job shop problem $\mathcal{J}m//\Phi$ with *recirculation* (see [Pin95a]), which may occur when a job may visit a machine more than once.

To illustrate the above notations and some results, we consider in Section 6 an example of a job shop scheduling problem with two jobs and two machines.

Example 1.5 *The job shop problem is specified by the mixed graph $G = (Q, A, E)$ given in Figure 1.5. The first job consists of operations 1 and 2, and the second job consists of operations 3 and 4. So we have the precedence constraints $1 \to 2$ and $3 \to 4$. The*

assignment of the operations to the machines is as follows: $Q_1 = \{1, 4\}$, $Q_2 = \{2, 3\}$. *The vector* $p = (10, 30, 20, 40)$ *defines the processing times of the operations* $Q = \{1, 2, 3, 4\}$.

For this problem we get $\Lambda(G) = \{G_1, G_2, G_3\}$ *with the following signatures of semiactive schedules:* $E_1 = \{(1, 4), (3, 2)\}$, $E_2 = \{(1, 4), (2, 3)\}$ *and* $E_3 = \{(4, 1), (3, 2)\}$. *The corresponding sets of dominant paths are as follows:* $H_1^1 = \{(1, 2), (3, 2)\}$, $H_1^2 = \{(1, 4), (3, 4)\}$, $H_2^1 = \{(1, 2)\}$, $H_2^2 = \{(1, 2, 3, 4)\}$, $H_3^1 = \{(3, 4, 1, 2)\}$ *and* $H_3^2 = \{(3, 4)\}$. *The optimal digraph* $G_1 = (Q, A \cup E_1, \emptyset)$ *is represented in Figure 1.5 and it defines the unique optimal semiactive schedule* $(10, 50, 20, 60)$ *for both criteria* $\mathcal{C}_{max}$ *and* $\sum \mathcal{C}_i$. *Therefore, due to Corollary 1.4 and Corollary 1.5 we have* $\widehat{\varrho}_s(p) > 0$ *and* $\overline{\varrho}_s(p) > 0$.

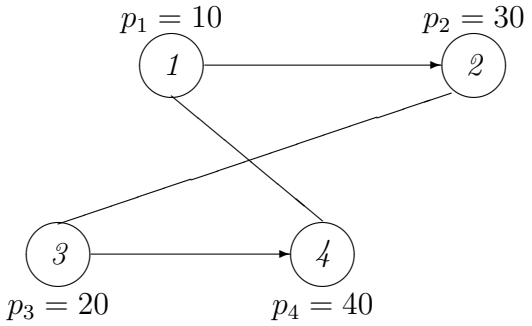

Figure 1: *Mixed graph* $G = (Q, A, E)$ *for example 1.5*

We can calculate the exact value of $\widehat{\varrho}_1(p)$ *on the basis of Theorem 1.5. First, we compare the digraphs* G_1 *and* G_2. *We have four sets of representatives for digraph* G_1, *namely:* $\Omega_1^1 = \{(1, 2), (1, 4)\}$, $\Omega_1^2 = \{(1, 2), (3, 4)\}$, $\Omega_1^3 = \{(3, 2), (1, 4)\}$ *and* $\Omega_1^4 = \{(3, 2), (3, 4)\}$. *We can calculate vectors* $n(\Omega_1^1) = (2, 1, 0, 1)$, $n(\Omega_1^2) = (1, 1, 1, 1)$, $n(\Omega_1^3) = (1, 1, 1, 1)$ *and* $n(\Omega_1^4) = (0, 1, 2, 1)$. *Digraph* G_2 *has only one set of representatives:* $\Omega_2^1 = \{(1, 2), (1, 2, 3, 4)\}$ *with* $n(\Omega_2^1) = (2, 2, 1, 1)$. *Obviously, we have* $n_i(\Omega_1^u) \leq n_i(\Omega_2^1)$ *for* $u \in \{1, 2, 3\}$ *and for all* $i \in \{1, 2, 3, 4\}$. *Thus, we have to calculate only*

$$r_{\Omega_2^1, \Omega_1^4} = \frac{(40 + 100) - (50 + 60)}{\mid 2 - 0 \mid + \mid 2 - 1 \mid + \mid 1 - 2 \mid + \mid 1 - 1 \mid} = \frac{30}{4} = 7.5.$$

Next, we compare the digraphs G_1 *and* G_3. *Digraph* G_3 *has only one set of representatives:* $\Omega_3^1 = \{(3, 4, 1, 2), (3, 4)\}$ *with* $n(\Omega_3^1) = (1, 1, 2, 2)$. *Obviously, we have* $n_i(\Omega_1^u) \leq n(\Omega_3^1)$ *for* $u \in \{2, 3, 4\}$ *and for all* $i \in \{1, 2, 3, 4\}$. *Thus, we have to calculate only*

$$r_{\Omega_3^1, \Omega_1^1} = \frac{(100 + 60) - (40 + 50)}{\mid 1 - 2 \mid + \mid 1 - 1 \mid + \mid 2 - 0 \mid + \mid 2 - 1 \mid} = \frac{70}{4} = 17.5.$$

Since $7.5 < 10 = \min\{p_1, p_2, p_3, p_4\}$, *the bound (1.34) is reached and we obtain* $\varrho_1^{\Sigma}(p) = \min\{7.5; 17.5\} = 7.5$ *and* G_2 *is a competitive digraph for digraph* G_1, *which is optimal for* $p = (10, 30, 20, 40)$.

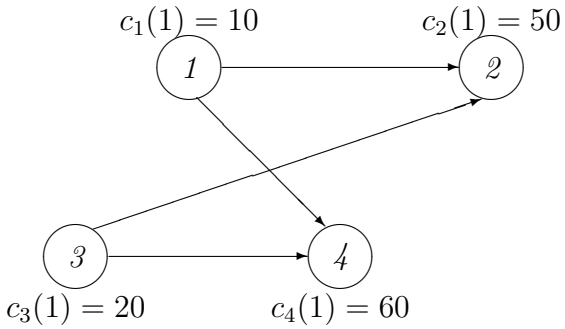

Figure 2: *The optimal digraph* $G_1 = (Q, A \cup E_1, \emptyset)$

Next we consider the stability radius of the optimal digraph G_1 *with respect to criterion* $\mathcal{C}_{max}$. *We have the following sets of dominant paths:* $H = \{(1,2),(3,4)\}$, $H_1 = \{(1,2),(1,4),(3,2),(3,4)\}$, $H_2 = \{(1,2,3,4)\}$ *and* $H_3 = \{(3,4,1,2)\}$. *It is clear that any path* $\mu \in H_1 \backslash H$ *is dominated by the path* $(1,2,3,4) \in H_2$ *and by the path* $(3,4,1,2) \in H_3$. *Thus, due to Theorem 1.1 we have* $\widehat{\varrho}_1(p) = \infty$.

In scheduling theory **open shop** problem is also considered when technological routes for processing jobs are not fixed before scheduling, i.e., for each job $J_i \in J$ only set of operations Q^{J_i} is given but order of these operations is not fixed. While solving open shop problem $\mathcal{O}//\Phi$ it is necessary to find optimal orders of operations Q^{J_i} for each job $J_i \in J$ along with optimal orders of operations on each machine $M_k \in M$. Similarly to flow shop problem, each job $J_i \in J$ has to be processed by each machine exactly once. The same property of the technological routes are assumed for so-called **classical job shop**. Classification of shop scheduling problems is given in Table 1.1.

Table 1.1: *Different shop scheduling problems*

Characterization of machine service	Shop scheduling problem	Technological routes of the jobs
Each job $J_i \in J$ has to be processed by each machine $M_k \in M$ exactly once	Open shop $\mathcal{O}//\Phi$	Different jobs may have different routes, which are not fixed before scheduling
	Flow shop $\mathcal{F}//\Phi$	Jobs have the same route, which is fixed before scheduling
	Classical job shop	Different jobs may have different routes, which are fixed before scheduling
A job may be processed by a machine more than once	Job shop $\mathcal{J}//\Phi$	

1.5 Stability of an Optimal Line Balance

This section is addressed the simple assembly line balancing problem (abbreviation: SALBP), in which it is necessary to minimize number of stations m for processing n partially ordered operations $V = \{1, 2, \ldots, n\}$ within the cycle time c.

Let the processing time p_i of operation $i \in V$ and cycle time c be given. However, during the life cycle of the assembly line the values p_i are definitely fixed only for the subset of automated operations $V \setminus \widetilde{V}$. Another subset $\widetilde{V}$ includes manual operations, for which it is impossible to fix exact processing times during the whole life cycle of the assembly line. If $j \in V \setminus \widetilde{V}$, then operation time p_j can be different for different cycles of production process.

In this section, for the optimal line balance $\mathbf{b}$ of paced assembly line with vector $p = (p_1, p_2, \ldots, p_n)$ of the operation times, we investigate stability of its optimality with respect to possible variations of the processing times p_j of the manual operations $j \in \widetilde{V}$. In particular, we derive necessary and sufficient conditions when optimality of the line balance $\mathbf{b}$ is stable with respect to sufficiently small variations of the operation times p_j, $j \in \widetilde{V}$. We show how to calculate the maximal value of independent variations of the processing times of all the manual operations, which definitely keep the feasibility and optimality of the line balance $\mathbf{b}$.

Simple Assembly Line Balancing Problem

We consider a single-model paced assembly line, which continuously manufactures homogeneous product in large quantities (mass production). In this section, we use terminology and main notations given in survey [Bay86] and monograph [Sch99].

The assembly line under consideration is a sequence of m linearly ordered stations, which are linked by conveyor belt (or other material handling equipment). All stations of the assembly line have to perform the same set of operations repeatedly during the life cycle of the assembly line. Set of operations V, which have to be processed on the assembly line within one cycle time c, is fixed. Cycle time c is fixed as well. Each operation $j \in V$ is considered indivisible: An operation j has to be completely processed on one station within one cycle time c. All the m stations start simultaneously the sequences of their operations and buffers between stations are absent.

We assume that set V includes operations of two types. On the one hand, subset $\widetilde{V}$ of set V includes all the operations, for which it is impossible to fix exact processing times for the whole life cycle of the assembly line (e.g., manual operations). On the other hand, each operation $i \in V \setminus \widetilde{V}$ is one with operation time p_i being fixed during the life cycle of the assembly line (e.g., in the case of automated operations).

The known technological factors define a partial order on the set of operations V. Let digraph $D = (V, A)$ with vertices V and arcs A define partially ordered set of operations $V = \{1, 2, \ldots, n\}$, which have to be processed on the assembly line within cycle time c. Without loss of generality, we assume that

$$\widetilde{V} = \{1, 2, \ldots, \widetilde{n}\} \text{ and } V \setminus \widetilde{V} = \{\widetilde{n} + 1, \widetilde{n} + 2, \ldots, n\},$$

where $1 \leq \widetilde{n} \leq n$. If $\widetilde{n} = 0$, then $\widetilde{V} = \emptyset$. We will use the following notations for the

vectors of the operation times:

$$\widetilde{p} = (p_1, p_2, ..., p_{\widetilde{n}}),\ p = (\widetilde{p}, \overline{p}) = (p_1, p_2, ..., p_n).$$

Thus, we have $V = \{1, 2, \ldots, \widetilde{n}, \widetilde{n}+1, \ldots, n\}$. If $\widetilde{n} = n$, then $\widetilde{V} = V$.

Simple Assembly Line Balancing Problem (SALBP) is to find an optimal balance of the assembly line for the given cycle time c, i.e., to find a feasible assignment of all the operations V into the minimal possible number m of stations. (Note that in [Bay86] and [Sch99], abbreviation SALBP-1 is used for such a problem.)

Assignment $V = V_1 \cup V_2 \cup \ldots \cup V_m$ of operations V into m linearly ordered stations

$$\mathbf{S} = (S_1, S_2, ..., S_m)$$

(i.e., partition of set V into m mutually disjoint non-empty subsets V_k, $k = 1, 2, ..., m$) is feasible operation assignment (also called *line balance*) if the following two conditions hold.

Condition (I): Feasible operation assignment does not violate the precedence constraints given by digraph $D = (V, A)$, i.e., inclusion $(i, j) \in A$ implies that operation i is assigned to station S_k: $i \in V_k$, and operation j is assigned to station S_t: $j \in V_t$, such that $1 \le k \le t \le m$.

Condition (II): Cycle time c is not violated for each station S_k, $k \in \{1, 2, \ldots, m\}$, i.e., sum of the processing times of all the operations assigned to station S_k (also called *station time*), has to be not greater than cycle time c:

$$\sum_{i \in V_k} p_i \le c. \tag{1.45}$$

For SALBP, line balance $\mathbf{b}$ is optimal when it uses the minimal number of m stations (optimality) and when the Condition (I) and Condition (II) are satisfied for $\mathbf{b}$ (feasibility). The following claim may be easily proven by polynomial reduction of NP-complete partition problem to the decision version of SALBP with two stations and with $A = \emptyset$ (see, e.g., [Sch99]).

Constructing an optimal line balance for SALBP is binary NP-hard problem even for the case of two stations used in the optimal line balance, $\mathbf{S} = (S_1, S_2)$*, empty precedence constraints,* $A = \emptyset$*, and fixed processing times of all the operations* V *processed on the assembly line,* $\widetilde{V} = \emptyset$*.*

For the sake of simplicity, we will use the following notation

$$p(V_k) = \sum_{i \in V_k} p_i,$$

but only for the original vector $p = (p_1, p_2, ..., p_n)$ of the operation times (in order to avoid misunderstanding). As it was already mentioned, set V includes operations of two types: Subset $\widetilde{V}$ of operations with variable processing time (manual operations) and

subset $V \setminus \widetilde{V}$ of operations with fixed processing time (automated operations). We assume that, if $j \in \widetilde{V}$, then operation time p_j is given non-negative real number: $p_j \geq 0$, however the value of this operation time can vary during life cycle of the assembly line and can even be equal to zero. Zero operation time p'_j will mean that operation $j \in V_k \bigcap \widetilde{V}$ will be processed (e.g., by an additional worker) in such a way that processing operation j will do not increase station time for S_k for the new vector $p = (\tilde{p}', \bar{p}) = (p'_1, p'_2, ..., p'_{\tilde{n}}, p_{\tilde{n}+1}, ..., p_n)$ of the operation times:

$$\sum_{i \in V_k} p'_i = \sum_{i \in V_k \setminus \{j\}} p'_i.$$

Obviously, the latter equality is only possible if $p'_j = 0$. If $i \in V \bigcap \widetilde{V}$, then operation time p_i is given real number fixed during the life cycle of the assembly line. We assume that $p_i > 0$ for each operation $i \in V \bigcap \widetilde{V}$. As far as the processing time of the automated operation is fixed, one can consider only automated operations, which have strictly positive processing times. Indeed, an operation with fixed zero processing time (e.g., due to using an additional machine for processing this operation) has no influence on the solution of SALBP under consideration.

In contrast to usual stochastic settings of SALBP, we do not assume any probability distribution known in advance for the random processing times of the manual operations. Moreover, this section does not deal with algorithms for constructing optimal line balance in this type of uncertainty. It is assumed that optimal line balance $\mathbf{b}$ is already constructed for the fixed vector $p = (p_1, p_2, ..., p_n)$ of the operation times. Our aim is to investigate stability of the concrete line balance $\mathbf{b}$, which is optimal for the given operation times $p_i, i \in V$, with respect to independent variations of the processing times of all the manual operations or a portion of the manual operations.

More precisely, we investigate stability radius of an optimal line balance. Stability radius may be interpreted as a maximal possible error of the processing times p_j of all the manual operations $j \in \widetilde{V}$, which definitely keep the optimality of the line balance $\mathbf{b}$. Next, we give some motivation for stability analysis of the optimal line balance and then formal definition of stability radius.

Motivation for Stability Analysis

Assembly line balancing problem arises when a new assembly line must be installed, and the internal demands and properties of the assembly line have to be well estimated since assembly line has to be used for a long time (long life cycle). Cycle time c may be defined on the basis of customer demands in the finished products. More precisely, the value of c may be calculated as the ratio of available operating time of the assembly line and production volume for the same calendar interval. One of the common mathematical problems at the stage of assembly line design is SALBP. This problem may arise also when cycle time c of acting assembly line has to be changed because of changing customer demands in the finished product.

In the real-world assemble lines, processing times of some operations may be known exactly and fixed for a long time (e.g., if operation has to be done by fully-automated machine or by semi-automated machine). Modern machines and robots are able to work permanently at a constant speed for a long time. However, in some cases it is not realistic

to assume constant operation times, e.g., if an operation has to be done by a human operator with rather simple tools. In the case of a human work, operation time is subject to physical, psychological and other factors. Moreover, due to the learning of operators, the operation times during the first days (weeks, months) of a life cycle of the assembly line may differ considerably from the processing times of the same operations during the later days (weeks, months). Also, some workers can leave the plant, and new workers with lower or higher skills have to replace them.

In the case of changeable operation times, it is important to know the credibility of the optimal line balance at hand with respect to possible independent variations of all or a portion of the operation times. Line balance $\mathbf{b}$, which is optimal for the original vector $p = (\tilde{p}, \bar{p})$ of the operation times, may lose its optimality (and even feasibility) for some new vector of the operation times. For example, due to increasing of some operation times, line balance $\mathbf{b}$ may become infeasible for cycle time c since inequality (1.45) may be violated. In such a case, it is necessary to look for another line balance and to use it, if possible, for a suitable modification of production process on the assembly line.

Also, line balance $\mathbf{b}$ may lose its optimality with saving its feasibility. It may occur if another operation assignment, say $\mathbf{b}_s$, become feasible for the modified vector $p' = (\tilde{p}', \bar{p})$ of the operation times, and $\mathbf{b}_s$ uses less stations than line balance $\mathbf{b}$ uses. Of course, each re-engineering and modification of the assembly line being in process take an additional time and other expenditure. So, assembly line modification has to be started when it is really necessary: When the income from it will be larger than the total expenditure caused by the re-engineering. Thus, an evaluation of expenditures and benefits should be conducted before deciding whether re-engineering of assembly line is necessary. However, these expenditures and benefits are difficult to evaluate before the end of the re-engineering process. In what follows, we derive some sufficient conditions for keeping the optimality of the line balance being in process. E.g., re-engineering is not necessary if the currently applied line balance remains optimal in spite of the changes of the operation times $\tilde{p}$.

To test whether line balance $\mathbf{b}$ remains feasible for the new vector $p' = (\tilde{p}', \bar{p})$ of the operation times takes $O(\tilde{n})$ time (if station times are included in the input data) or $O(n)$ time (otherwise). Indeed, for the new operation times we have to verify inequality (1.45) for each subset $V_k, k = 1, 2, ..., m$, which includes at least one manual operation with changed processing time in the new vector p'. On the other hand, in the case of feasibility of the line balance $\mathbf{b}$ for the new vector p', in order to test its optimality for p' we have again to solve NP-hard SALBP.

Intuitively, it is clear that sufficiently small changes of the operation times $p_1, p_2, ..., p_{\tilde{n}}$ may keep line balance $\mathbf{b}$ feasible and optimal for the new vector $p' = (\tilde{p}', \bar{p})$ of the operation times. In what follows, our aim is to estimate or (what is better) to calculate the largest independent variations of the operation times $p_i, i \in \widetilde{V}$, which do not violate the feasibility and optimality of the line balance $\mathbf{b}$ at hand. Also note that at the stage of the design of the assembly line, there may exist a lot of optimal line balances. Using stability analysis, we can select such an optimal line balance, which feasibility and optimality are more stable with respect to possible variations of the operation times $p_i, i \in \widetilde{V}$.

Further, stability analysis of an optimal line balance will be developed. To this end, we will use the notion of stability radius, which is similar to stability radius of an optimal schedule introduced in Section 1.2 for general shop scheduling problems. If stability radius of line balance $\mathbf{b}$ is strictly positive, then any independent changes of the operation times

$p_j, j \in \widetilde{V}$, within the ball with this radius, definitely keep the optimality of line balance **b**. However, if stability radius of **b** is equal to zero (i.e., if the optimality of line balance **b** is unstable), then some even small changes of the processing times of all or a portion of the manual operations may deprive the optimality of line balance **b**.

Let B denote the set of all assignments of operations V into stations $S_1, S_2, ..., S_m$ (for all possible numbers m of stations: $1 \leq m \leq n$), which satisfy Condition (I). Subset of set B of all operation assignments (line balances) which also satisfy Condition (II) for the given vector $p = (p_1, p_2, ..., p_n)$ of the operation times is denoted by $B(p) = \{\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_h\}$. Let subset of set $B(p)$ of all the optimal line balances be denoted by $B_{opt}(p)$.

Thus, inclusion $\mathbf{b} \in B_{opt}(p)$ implies that line balance $\mathbf{b}$:

$$V = V_1^b \cup V_2^b \cup ... \cup V_{m_\mathbf{b}}^b,$$

satisfies Condition (I), Condition (II), and the following optimality condition for the vector $p = (p_1, p_2, ..., p_n)$ of the operation times.

Condition (III). $m_b = \min\{m_{b_k} : \mathbf{b}_k \in B(p)\}$.

Hereafter $\mathbf{b}_k, k \in \{1, 2, ..., h\}$, means the following line balance:

$$V = V_1^{b_k} \cup V_2^{b_k} \cup ... \cup V_{m_{b_k}}^{b_k}.$$

Since line balance **b** is contained in the set $B(p)$, we have $\mathbf{b} = \mathbf{b}_r \in B(p)$ for some index $r \in \{1, 2, ..., h\}$. However, as a matter of convenience, index r will be omitted for the optimal line balance **b**, which stability will be investigated.

Note that in both definitions of set B and set $B(p)$, number m of stations is not fixed. Namely, for each line balance $\mathbf{b}_k$ from the set $B(p)$, inequalities $m_b \leq m_{b_k} \leq n$ must hold. And number of stations in an operation assignment from set B has to belong to set $\{1, 2, \ldots, n\}$.

The main questions under consideration may be formulated as follows. How much can be modified the components of the vector $\tilde{p}$ simultaneously and independently from each other that the given line balance **b** remains feasible and optimal?

Let line balance **b** be optimal for the given non-negative real vector $p = (\tilde{p}, \bar{p}) = (p_1, p_2, ..., p_n) \in R_+^n$ of the operation times, i.e., $\mathbf{b} \in B_{opt}(p)$. The formal definition of stability radius of an optimal line balance **b** may be introduced as follows.

Definition 1.4 *Open ball $O_\rho(\tilde{p})$ with radius $\rho \in R_+^1$ and center $\tilde{p} \in R_+^{\tilde{n}}$ in the space $R^{\tilde{n}}$ with the maximum metric is called a stability ball of the line balance $\mathbf{b} \in B_{opt}(p)$, if for each vector $p^* = (\tilde{p^*}, \bar{p})$ of the operation times with $\tilde{p}^* \in O_\rho(\tilde{p}) \cap R_+^{\tilde{n}}$ operation assignment* **b** *remains feasible and optimal. The maximal value $\rho_\mathbf{b}(p)$ of the radius ρ of stability ball $O_\rho(\tilde{p})$ of the optimal line balance* **b** *is called stability radius denoted by $\rho_\mathbf{b}(p)$.*

It should be noted that in Definition 1.4 vector $\tilde{p} = (p_{\tilde{n}+1}, p_{\tilde{n}+2}, ..., p_n)$ of the processing times of the automated operations and the complete vector $p = (\tilde{p}, \bar{p}) = (p_1, p_2, ..., p_n)$ of the operation times are fixed, while vector $p^* = (p_1^*, p_2^*, ..., p_{\tilde{n}}^*)$ may vary within the intersection of the open ball $O_\rho(\tilde{p}) \subset R^n$ with the space $R_+^{\tilde{n}}$ of non-negative real vectors. Stability radius $\rho_\mathbf{b}(p)$ is equal to the minimal upper bound of independent variations ε_i

of the processing times p_i of all the manual operations $i \in \widetilde{V}$, which definitely keep the optimality of the line balance $\mathbf{b}$. In other words, inclusion $\mathbf{b} \in B_{opt}(p^*)$ must hold, if inequalities $\max\{0, p_i - \varepsilon_i\} \leq p_i^* \leq p_i + \varepsilon_i$ hold for each operation $i \in V$.

To illustrate the above notations, we use the following example of SALBP.

Example 1.6 *We suppose that* $c = 10$, $\tilde{n} = 4$, $n = 8$ *and* $\underline{p} = (2, 3, 3, 9, 5, 3, 7, 2)$. *Thus,* $\widetilde{V} = \{1, 2, 3, 4\}$ *is set of the manual operations, and* $V \backslash \widetilde{V} = \{5, 6, 7, 8\}$ *is set of the automated operations. Digraph* $D = (V, A)$ *is represented in Figure 1.7 where the manual operation times (which can be changed during the life cycle of the assembly line) are written in the usual form over the vertices, while the fixed operation times* p_i *for automated operations* $i \in V \backslash \widetilde{V}$ *are written in the bold.*

The following line balance $\mathbf{b} = \mathbf{b}_r \in B(p)$:

$$V_1^{\mathbf{b}} = \{1, 3, 5\}, V_2^{\mathbf{b}} = \{2, 6\}, V_3^{\mathbf{b}} = \{4\}, V_4^{\mathbf{b}} = \{7, 8\}$$

is optimal since $\mathbf{b}$ *uses minimal possible number* $m_{\mathbf{b}} = 4$ *of stations. Indeed, the sum of all operation times* $p_i, i \in V$, *is equal to* $p(V) = 34$, *and number of stations in any line balance cannot be less than*

$$\lceil \frac{p(V)}{c} \rceil = \lceil \frac{34}{10} \rceil = 4,$$

where $\lceil a \rceil$ *denotes the smallest integer greater than or equal to* a.

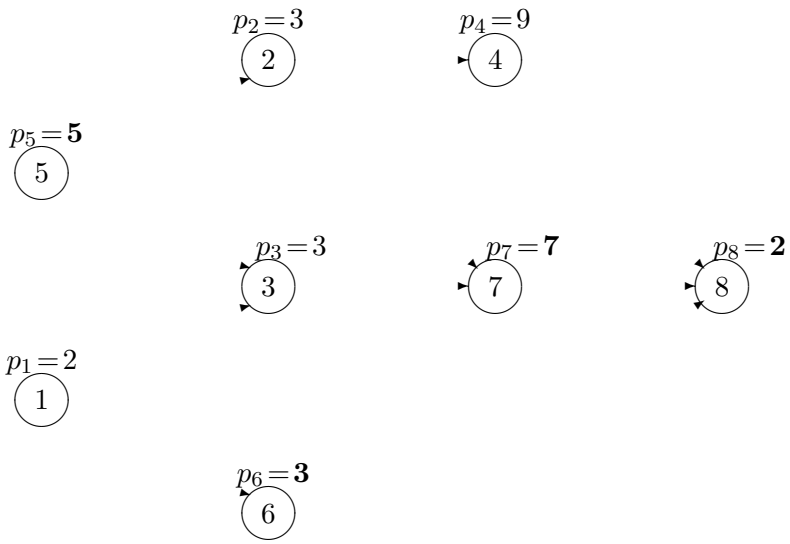

Figure 1.7: *Digraph* $D = (V, A)$ *and operation times*

Let $\widetilde{V}_k^{\mathbf{b}}$ denote the subset of manual operations of set $V_k^{\mathbf{b}}$, and $\bar{V}_k^{\mathbf{b}}$ denote the subset of automated operations of set $V_k^{\mathbf{b}}$. Thus, for each index $k \in \{1, 2, ..., m_{\mathbf{b}}\}$, we have $V_k^{\mathbf{b}} = \widetilde{V}_k^{\mathbf{b}} \cup \bar{V}_k^{\mathbf{b}}$. The following remark will be used in the proofs of the main results in the rest of this section.

Remark 1.2 Let us consider the line balance $\mathbf{b} \in B_{opt}(p)$ being in process and the modified vector $p' = (\tilde{p}', \bar{p})$ of the operation times. If there exists subset $V_k^{\mathbf{b}}, k \in \{1, 2, ..., m_{\mathbf{b}}\}$, in the line balance $\mathbf{b}$ such that

$$\sum_{i \in V_k^b} p'_i = 0, \tag{1.46}$$

we continue to affirm that the line balance $\mathbf{b}$ uses $m_{\mathbf{b}}$ stations for the modified vector $p' = (\tilde{p}', \bar{p})$ as well. We can argue this as follows. In spite of the equality (1.46) valid for the vector $p' = (\tilde{p}', \bar{p})$, station S_k is still exists in the assembly line with line balance $\mathbf{b}$. At least to delete station S_k may cause some additional cost and time for re-engineering the assembly line. Moreover, after deleting station S_k we will obtain another line balance, say $\mathbf{b}^* \in B$:

$$V = \bigcup_{i \in \{1,2,...,m_b\}, i \neq k} V_i^{\mathbf{b}} = V_1^{\mathbf{b}^*} \cup V_2^{\mathbf{b}^*} \cup ... \cup V_{m_{\mathbf{b}^*}}^{\mathbf{b}^*},$$

where $m_{\mathbf{b}^*} = m_{\mathbf{b}} - 1$.

Note that because of inequality $p_i > 0$ is valid for each automated operation $i \in V \backslash \widetilde{V}$, equality (1.46) is only possible if $V_k^{\mathbf{b}} = \widetilde{V}_k^{\mathbf{b}}$.

Example 1.6 (continued). *For Example 1.6, we have $V_3^{\mathbf{b}} = \widetilde{V}_3^{\mathbf{b}} = \{4\}$, and equality (1.46) holds for station S_3 and the modified vector $p' = (\tilde{p}', \bar{p})$ with $\tilde{p}' = (2, 3, 3, 0)$. Hence, for the vector $p' = (\tilde{p}', \bar{p}) = (2, 3, 3, 0, 5, 3, 7, 2)$ of the operation times there exists another line balance which is better than line balance $\mathbf{b}$ since it uses less stations than $\mathbf{b}$.*

Indeed, manual operation 4 (with zero operation time: $p'_4 = 0$) may be assigned to another station without increasing station time. For such a moving operation 4, we need only to guarantee feasibility of a new operation assignment with respect to Condition (I). E.g., we can use the following operation assignment $\mathbf{b}^{(s)}$:

$$V_1^{\mathbf{b}^{(s)}} = \{1, 3, 5\}, V_2^{\mathbf{b}^{(s)}} = \{2, 4, 6\}, V_3^{\mathbf{b}^{(s)}} = \{7, 8\},$$

obtained from the line balance $\mathbf{b}$ via moving operation 4 from station S_4 to station S_2.

It is easy to see that operation assignment $\mathbf{b}^{(s)}$ satisfies both Conditions (I) and (II) for the modified vector $p' = (\tilde{p}', \bar{p})$ and $\mathbf{b}^{(s)}$ uses fewer stations than line balance $\mathbf{b}$ uses:

$$m_{\mathbf{b}^{(s)}} = 3 < 4 = m_{\mathbf{b}}.$$

It should be noted that line balance $\mathbf{b}^{(s)}$ is not feasible for the original vector p of the operation times ($\mathbf{b}^{(s)} \notin B(p)$), since inequality (1.45) from Condition (II) is violated for station S_2: $p(V_2^{\mathbf{b}^{(s)}}) = 3 + 3 + 9 = 15 > 10 = c$. Note that $d(\tilde{p}, \tilde{p}') =\mid p_4 - p'_4 \mid= 9$, and thus we can conclude that $\rho_{\mathbf{b}}(p) \leq 9$.

Moreover, we can consider another vector $p'' = (2, 3, 3, 4, 5, 3, 7, 2)$ of the operation times for which $\mathbf{b} \notin B_{opt}(p'')$ and $\mathbf{b}^{(s)} \in B_{opt}(p'')$. The vector p'' is closer to the original vector p than vector p' : $d(\tilde{p}, \tilde{p}) =\mid p_4 - p''_4 \mid= 9 - 4 = 5$. Therefore, we conclude that $\rho_{\mathbf{b}}(p) \leq 5$.

Farther, we show how to modify the original vector p of the operation times when there may be no subset V_k^b with valid equality (1.46) in the line balance $\mathbf{b}$.

Zero Stability Radius

We derive the necessary and sufficient conditions for the case when optimality (or feasibility) of the line balance $\mathbf{b} \in B_{opt}(p)$ is unstable.

Theorem 1.8 *For the line balance* $\mathbf{b} \in B_{opt}(p)$ *equality* $\rho_{\mathbf{b}}(p) = 0$ *holds if and only if there exists subset* $V_k^b, k \in \{1, 2, ..., m_{\mathbf{b}}\}$, *such that* $\widetilde{V}_k^b \neq \emptyset$ *and* $p(V_k^b) = c$.

Proof. *Sufficiency (if).* We make the following assumption (called assumption IF). Let there exist subset V_k^b with $\widetilde{V}_k^b \neq \emptyset$ and $p(V_k^b) = c$.

Since $\widetilde{V}_k^b \neq \emptyset$, there exists at least one manual operation $j \in \widetilde{V}_k^b$. For any small positive real ε, we set $p_j^* = p_j + \varepsilon$. And for all other manual operations $i \in \widetilde{V} \backslash \{j\}$, we set $p_i^* = p_i$. As a result, we obtain vector $\tilde{p}^* = (p_1^*, p_2^*, ..., p_{\tilde{n}}^*)$. For the complete vector $p^* = (\tilde{p}^*, \bar{p})$ of the operation times, from equality $p(V_k^b) = c$ (see assumption IF), it follows:

$$\sum_{i \in V_k^b} p_i^* = \sum_{j \in \widetilde{V}_k^b} p_j^* + \sum_{i \in \bar{V}_k^b} p_i = \sum_{i \in V_k^b} p_i + \varepsilon = c + \varepsilon > c.$$

Therefore, operation assignment $\mathbf{b}$ does not satisfy Condition (II) for the new vector $p^* = (\tilde{p}^*, \bar{p})$. As a consequence, $\mathbf{b} \notin B(p^*)$, and hence $\mathbf{b} \notin B_{opt}(p^*)$. Since distance defined by formula (1.4) with $Q = V$ between the original vector $\tilde{p}$ and the new vector $\tilde{p}^*$ may be as small as desired: $d(\tilde{p}, \tilde{p}^*) = \varepsilon$, we conclude that $\rho_{\mathbf{b}}(p) = 0$. In such a case, stability ball $O_\rho(\tilde{p})$ may be interpreted as a single point: $O_\rho(\tilde{p}) = \{\tilde{p}\}$.

Necessity (only if). We make the opposite assumption (called assumption ONLY IF). There is no subset $V_k^b, k \in \{1, 2, \ldots, m_{\mathbf{b}}\}$, such that both conditions $\widetilde{V}_k^b \neq \emptyset$ and $p(V_k^b) = c$ hold. Hence, if $\widetilde{V}_k^b \neq \emptyset$, then $p(V_k^b) < c$.

In order to get zero stability radius for the line balance $\mathbf{b}$ we have to show that for any arbitrarily small real $\varepsilon > 0$ there exists vector $\tilde{p} \in R_+^{\tilde{n}}$ that $d(\tilde{p}, \tilde{p}') = \varepsilon$ and $\mathbf{b} \in B_{opt}(p')$ with $p = (\tilde{p}, \bar{p})$ and $p' = (\tilde{p}', \bar{p})$. In principle, for realization equality $\rho_{\mathbf{b}}(p) = 0$ there are three possibilities (we denote them as case (i), case (ii) and case (iii)), each of which corresponds to violation Condition (I), Condition (II) or Condition (III) for the operation assignment $\mathbf{b}$ with modified vector p' of the operation times. Next, we consider these possibilities in details and prove that they are impossible for sufficiently small real $\varepsilon > 0$.

(i) To make operation assignment $\mathbf{b}$ inadmissible due to violation Condition (I) for the modified vector p' of the operation times, i.e., $\mathbf{b} \notin B$.

However, the latter contradicts to the inclusion $\mathbf{b} \in B_{opt}(p)$ being assumed in Theorem 1.8.

(ii) To make operation assignment $\mathbf{b} \in B$ inadmissible for the modified vector p' due to violation Condition (II), i.e., $\mathbf{b} \notin B(p')$.

Since $\mathbf{b} \in B$, cycle time c may be exceeded for the vector $p = (\tilde{p}, \bar{p}) = (p_1', p_2', ..., p_{\tilde{n}}', p_{\tilde{n}+1}', \ldots, p_n) \in R_+^n$ (i.e., inequality (1.45) may be violated) only for some station S_k with $\widetilde{V}_k^{\mathbf{b}} \neq \emptyset$. Due to assumption ONLY IF for each such station S_k inequality $p(V_k^{\mathbf{b}}) < c$ must hold.

Let us consider successively all such stations $S_k, k \in \{1, 2, ..., m_{\mathbf{b}}\}$, with $\widetilde{V}_k^{\mathbf{b}} \neq \emptyset$ and $p(V_k^{\mathbf{b}}) < c$. To violate inequality (1.45) for subset $p(V_k^{\mathbf{b}})$ (and so to violate Condition (II) for $\mathbf{b}$)), the closest to $\tilde{p}$ vector $\tilde{p}'$ may be obtained by increasing all the components

$p_i, i \in \widetilde{V}_k^{\mathbf{b}}$, of the vector $\tilde{p} = (p_1, p_2, ..., p_{\tilde{n}})$ by the same value, which has to be more than the following fraction:

$$\frac{c - p(V_k^{\mathbf{b}})}{|\widetilde{V}_k^{\mathbf{b}}|} = \delta_k^{\mathbf{b}}. \tag{1.47}$$

Due to inequalities $p(V_k^{\mathbf{b}}) < c$ and $|\widetilde{V}_k^{\mathbf{b}}| \geq 1$ the value $\delta_k^{\mathbf{b}}$ is strictly positive: $\delta_i^{\mathbf{b}} > 0$. If we add $\delta_i^{\mathbf{b}}$ to the processing time p_i of each operation $i \in \widetilde{V}_k^{\mathbf{b}} : p_i^O = p_i + \delta_k^{\mathbf{b}}$, and if we set $p_j^O = p_j$ for all other manual operations $j \in \widetilde{V} \backslash \widetilde{V}_k^{\mathbf{b}}$, we obtain equality $\sum_{i \in V_k^{\mathbf{b}}} p_i^O = c$ for the modified vector $p^O = (\tilde{p}^O, \bar{p})$ with $p^O = (p_1^O, p_2^O, ..., p_{\tilde{n}}^O)$.

In order to violate Condition (II), we can continue similarly to the above proof of sufficiency. Note only that in this case, distance $d(\tilde{p}^O, \tilde{p})$ is equal to $\delta_k^{\mathbf{b}}$ and therefore such a distance cannot be arbitrarily small for the fixed line balance $\mathbf{b}$.

Also note that to realize case (ii) it is sufficient to violate inequality (1.45) for at least one station (and as a result Condition (II) will be violated for the line balance $\mathbf{b}$). Therefore, we can calculate the following lower bound of distance $d(\tilde{p}', \tilde{p})$ for the desired vector p:

$$d(\tilde{p}^O, \tilde{p}) > \delta^{\mathbf{b}} = \min\{\delta_k^{\mathbf{b}} : \widetilde{V}_k^b \neq \emptyset, k \in \{1, 2, \ldots, m_{\mathbf{b}}\}\}. \tag{1.48}$$

Remind that inequality $p(V_k^{\mathbf{b}}) < c$ is certainly implied by condition $\widetilde{V}_k^b \neq \emptyset$ (see assumption ONLY IF). Since $\delta_k^{\mathbf{b}} > 0$ for each subset V_k^b with $\widetilde{V}_k^b \neq \emptyset$, it follows that $\delta^{\mathbf{b}} > 0$. Thus, we conclude that Condition (II) may be violated for operation assignment $\mathbf{b}$ with some modified vector $\tilde{p}' \in R_+^{\tilde{n}}$ only if for the complete vector $p' = (\tilde{p}', \bar{p})$ of the operation times the following inequalities hold:

$$d(\tilde{p}, \tilde{p}') > \delta^{\mathbf{b}} > 0. \tag{1.49}$$

What is important, the above distance $d(\tilde{p}, \tilde{p}')$ cannot become arbitrarily small for the fixed line balance $\mathbf{b}$.

(iii) To find the modified vector $p'' = (\tilde{p}'', \bar{p})$ of the operation times, for which line balance $\mathbf{b}$ is feasible: $(\mathbf{b} \in B(p''))$, but not optimal: $(\mathbf{b} \notin B_{opt}(p''))$.

As we assume (see Remark 1.2), it is impossible to change the number of stations used in any operation assignment from set B. Therefore, to realize case (iii) we have to find another operation assignment, say $\mathbf{b}^* \in B$, which is feasible for a suitably modified vector $p'' = (\tilde{p}'', \bar{p})$ of the operation times $(\mathbf{b}^* \in B(p''))$, and which is better than line balance $\mathbf{b} \in B(p'')$ due to inequality $m_{\mathbf{b}^*} < m_{\mathbf{b}}$. In other words, we have to find the closest to $\tilde{p}$ vector $\tilde{p}^*$ such that $\mathbf{b}^* \in B(p'')$ for the complete vector $p'' = (\tilde{p}'', \bar{p})$ and $m_{\mathbf{b}^*} < m_{\mathbf{b}}$. As a result, we will be able to affirm that $\mathbf{b} \notin (B_{opt}(p^*)$.

Let us investigate mandatory properties of the desired operation assignment $\mathbf{b}^*$:

$$V = V_1^{\mathbf{b}^*} \cup V_2^{\mathbf{b}^*} \cup ... \cup V_{m_{\mathbf{b}^*}}^{\mathbf{b}^*}.$$

The following two properties have been already proven:

(a): $m_{\mathbf{b}^*} < m_{\mathbf{b}}$.

(b): $\mathbf{b}^* \in B(p'')$.

Note that if $p(\bar{V}_k^{\mathbf{b}^*}) > c$ for at least one index $k \in \{1, 2, ..., m_{\mathbf{b}^*}\}$, then operation assignment $\mathbf{b}^*$ cannot become feasible with respect to Condition (II) (see inequality (1.45)) for any modified vector $p'' = (\tilde{p}'', \bar{p})$. Therefore, we obtain the third property of the operation assignment $\mathbf{b}^*$:

(c): $p(\bar{V}_k^{\mathbf{b}^*}) \le c, k = 1, 2, ..., m_{\mathbf{b}^*}$.

From the feasibility (if any) of the operation assignment $\mathbf{b}^*$ for the original vector p of the processing times it would follow that $\mathbf{b} \notin B_{opt}(p)$, since $\mathbf{b}^*$ uses fewer stations than $\mathbf{b}$ uses (see property (a)). Therefore, operation assignment $\mathbf{b}^*$ cannot be feasible for the original vector p of the processing times, and we can denote this as property

(d): $\mathbf{b}^* \notin B(p)$.

It is easy to see that both properties (b) and (d) are only possible if in the operation assignment $\mathbf{b}^*$ there exists at least one subset $V_p^{\mathbf{b}^*}$ that inequality $p(V_p^{\mathbf{b}^*}) > c$ holds. We present this as property (e) of operation assignment $\mathbf{b}^*$.

(e): $p(V_p^{\mathbf{b}^*}) > c$ for at least one station $S_p, p \in \{1, 2, ..., m_{\mathbf{b}^*}\}$.

Let us consider any operation assignment $\mathbf{b} \in B$ that has all the above properties $(a), (b), (c), (d)$ and (e). Obviously, property (e) and property (c) for station S_p may simultaneously hold only if $\widetilde{V}_p^{\mathbf{b}^*} \neq \emptyset$. Therefore, for one subset $V_p^{\mathbf{b}^*}$ we can calculate the following strictly positive fraction:

$$\frac{t(V_p^{\mathbf{b}^*}) - c}{|\widetilde{V}_p^{\mathbf{b}^*}|} = \triangle(V_p^{\mathbf{b}^*}). \tag{1.50}$$

Numerator in the left-hand side of (1.50) shows how much station time of S_p exceeds cycle time c. Denominator shows in how much parts the numerator have to be divided, taking into account that all the processing times of operations $\widetilde{V}_p^{\mathbf{b}^*}$ may be simultaneously changed. In order to minimize distance $d(\tilde{p}'', \tilde{p})$ for desired vector $\tilde{p}''$, we can decrease each processing time $p_j, j \in \widetilde{V}_p^{\mathbf{b}^*}$, by the same value $\triangle(V_p^{\mathbf{b}^*})$ defined in (1.50). Thus, $\triangle(V_p^{\mathbf{b}^*})$ is equal to the minimal possible real number which has to be subtracted from each processing time p_j of the manual operation $j \in V_p^{\mathbf{b}^*}$ in order to obtain the closest to $\tilde{p}$ vector $\tilde{p}''$ of the processing times, for which the considered subset $V_p^{\mathbf{b}^*}$ becomes feasible with respect to inequality (1.45). Obviously, to define vector p'' we can set

$$p_j'' = p_j - \triangle(V_p^{\mathbf{b}^*}) \tag{1.51}$$

for each manual operation $j \in \widetilde{V}_p^{\mathbf{b}^*}$. For all other manual operations $i \in \widetilde{V} \setminus \widetilde{V}_p^{\mathbf{b}^*}$, we set $p_i'' = p_i$. As a result, we obtain vector $\tilde{p}_i'' = (p_1'', p_2'', ..., p_{\tilde{n}}'')$ with distance $d(\tilde{p}'', \tilde{p}) = \triangle(V_p^{\mathbf{b}^*}) > 0$ from vector $\tilde{p}$. It is clear that there is no vector $\tilde{p}^{**} \in R_+^{\tilde{n}}$, which is closer to $\tilde{p}$ than vector $\tilde{p}''$ provided that inclusion $\mathbf{b} \in B((\tilde{p}^{**}, \bar{p}))$ holds. (Taking into account Remark 1.2, substituting the original vector p by the modified vector $p'' = (\tilde{p}'', \bar{p})$ cannot decrease the number $m_{\mathbf{b}}$ of stations used in the line balance $\mathbf{b}$.)

Similarly, we can consider all the operation assignments $\mathbf{b}^* \in B$, which have properties $(a), (b), (c), (d)$ and (e). Then we calculate the minimum among values $\triangle(V_p^{\mathbf{b}^*})$. We denote this minimum by

$$\triangle^{\mathbf{b}} = \min\{\triangle(V_p^{\mathbf{b}^*}) : \mathbf{b}^* \in B \text{ possessed properties } (a), (b), (c), (d), (e)\}.$$

Since $\triangle(V_p^{\mathbf{b}^*}) > 0$ for each such $\mathbf{b}^* \in B$, we conclude that $\triangle^{\mathbf{b}} > 0$. Thus, for the closest to $\tilde{p}$ vector $\tilde{p}''$, for which case (iii) may be realized, the following inequalities must hold

$$d(\tilde{p}'', \tilde{p}) \ge \triangle^{\mathbf{b}} > 0. \tag{1.52}$$

Therefore, for case (iii) distance $d(\tilde{p}'', \tilde{p})$ cannot be arbitrarily small for the fixed line balance $\mathbf{b}$.

Summarizing, we conclude, first, that case (i) is not possible. Second, due to inequalities (1.49) and (1.52) either case (ii) or case (iii) may be realized for some modified vector $p' = (\tilde{p}', \bar{p})$ only if

$$d(\tilde{p}, \tilde{p}') \geq \min\{\delta^{\mathbf{b}}, \triangle^{\mathbf{b}}\}.$$

Thus, distance $d(\tilde{p}, \tilde{p}')$ cannot be arbitrarily small for the fixed line balance $\mathbf{b}$. Hence, $\rho_{\mathbf{b}}(p) > 0$, and Theorem 1.8 is proven.

$\diamond$

It is easy to see that testing criterion given in Theorem 1.8 takes $O(n)$ time. The latter bound is defined due to calculating station times $p(V_k^{\mathbf{b}}), k = 1, 2, ..., m_{\mathbf{b}}$. Therefore, if stations times are assumed to be included in the input data of the algorithm, then it takes $O(m_{\mathbf{b}})$ time. The following lower bound on stability radius has been obtained within the above proof of necessity of Theorem 1.8.

Corollary 1.6 *If optimality of line balance $\mathbf{b} \in B_{opt}(p)$ is stable, then*

$$\rho_{\mathbf{b}}(p) \geq \min\{\delta^{\mathbf{b}}, \triangle^{\mathbf{b}}\}. \tag{1.53}$$

Example 1.6 (continued). *Returning to Example 1.6, it is easy to convince that in the line balance $\mathbf{b}$ there exists subset $V_1^{\mathbf{b}} = \{1, 2, 5\}$, for which $p(V_1^{\mathbf{b}}) = 10 = c$ and $\widetilde{V}_1^{\mathbf{b}} = \{1, 3\} \neq \oslash$. Thus, Theorem 1.8 implies equality $\rho_{\mathbf{b}}(p) = 0$. Indeed, for the line balance $\mathbf{b}$ each of the two manual operations 1 and 3 are critical. If processing time of a critical operation will be increased (even by a small real number ε), then sum of the processing times of the operations assigned to station S_1 will exceed the cycle time $c = 10$ in the operation assignment $\mathbf{b}$. Namely, if we set either $p_1' = p_1 + \varepsilon = 2 + \varepsilon$ or $p_3' = p_3 + \varepsilon = 3 + \varepsilon$, then operation assignment $\mathbf{b}$ becomes unfeasible since Condition (II) is violated.*

In Table 1.5, other optimal line balances are represented for Example 1.6. Note that for the above optimal line balance $\mathbf{b}$ we use also notation $\mathbf{b}_r$. For each line balance in the second row of Table 1.5, we represent the station times as a sum of two items: automated operation times and manual operation times.

Let us consider optimal line balance $\mathbf{b}_s \in B(p)$:

$$V_1^{\mathbf{b}_s} = \{2, 5\}, V_2^{\mathbf{b}_s} = \{1, 3, 6\}, V_3^{\mathbf{b}_s} = \{4\}, V_4^{\mathbf{b}_s} = \{7, 8\},$$

which is also optimal for the original vector $p = (\tilde{p}, \bar{p}) = (2, 3, 3, 9, 5, 3, 7, 2)$, since

$$m_{\mathbf{b}_s} = 4 = \lceil \frac{p(V)}{c} \rceil.$$

For the line balance $\mathbf{b}_s \in B_{opt}(p)$ there does not exist subset $V_k^{\mathbf{b}_s}, k \in \{1, 2, 3, 4\}$, such that both conditions $\widetilde{V}_k^{\mathbf{b}_s} \neq \oslash$ and $p(V_k^{\mathbf{b}_s}) = c$ hold. Therefore, due to Theorem 1.8 the optimality of line balance $\mathbf{b}_s$ is stable: $\rho_{\mathbf{b}_s}(p) > 0$. The latter inequality means that if variations of all the processing times of the manual operations $\widetilde{V} = \{1, 2, 3, 4\}$ will be sufficiently small, then line balance $\mathbf{b}_s$ will remain optimal. Due to Theorem 1.8, line

Table 1.2: *Optimal line balances for vector* $p = (2, 3, 3, 9, 5, 3, 7, 2)$ *of the operation times*

$\mathbf{b}_r$	$V_1^{\mathbf{b}_r} = \{1, 3, 5\}$	$V_2^{\mathbf{b}_r} = \{2, 6\}$	$V_3^{\mathbf{b}_r} = \{4\}$	$V_4^{\mathbf{b}_r} = \{7, 8\}$
$p(V_k^{\mathbf{b}_r}) =$ $p(\widetilde{V}_k^{\mathbf{b}_r}) + p(\bar{V}_k^{\mathbf{b}_r})$	$10 =$ $(2 + 3) + 5$	$6 = 3 + 3$	$9 = 9 + 0$	$9 = 0 +$ $(7 + 2)$
$\mathbf{b}_s$	$V_1^{\mathbf{b}_s} = \{2, 5\}$	$V_2^{\mathbf{b}_s} = \{1, 3, 6\}$	$V_3^{\mathbf{b}_s} = \{4\}$	$V_4^{\mathbf{b}_s} = \{7, 8\}$
$p(V_k^{\mathbf{b}_s}) =$ $p(\widetilde{V}_k^{\mathbf{b}_s}) + p(\bar{V}_k^{\mathbf{b}_s})$	$8 = 3 + 5$	$8 =$ $(2 + 3) + 3$	$9 = 9 + 0$	$9 = 0 +$ $(7 + 2)$
$\mathbf{b}_u$	$V_1^{\mathbf{b}_u} = \{1, 5\}$	$V_2^{\mathbf{b}_u} = \{2, 3, 6\}$	$V_3^{\mathbf{b}_u} = \{4\}$	$V_4^{\mathbf{b}_u} = \{7, 8\}$
$p(V_k^{\mathbf{b}_u}) =$ $p(\widetilde{V}_k^{\mathbf{b}_u}) + p(\bar{V}_k^{\mathbf{b}_u})$	$7 = 2 + 5$	$9 =$ $(3 + 3) + 3$	$9 = 9 + 0$	$9 = 0 +$ $(7 + 2)$
$\mathbf{b}_w$	$V_1^{\mathbf{b}_w} = \{1, 2, 5\}$	$V_2^{\mathbf{b}_w} = \{3, 7\}$	$V_3^{\mathbf{b}_w} = \{4\}$	$V_4^{\mathbf{b}_w} = \{6, 8\}$
$p(V_k^{\mathbf{b}_w}) =$ $p(\widetilde{V}_k^{\mathbf{b}_w}) + p(\bar{V}_k^{\mathbf{b}_w})$	$10 =$ $(2 + 3) + 5$	$10 = 3 + 7$	$9 = 9 + 0$	$5 = 0 +$ $(3 + 2)$
$\mathbf{b}_z$	$V_1^{\mathbf{b}_z} = \{1, 3, 5\}$	$V_2^{\mathbf{b}_z} = \{2, 7\}$	$V_3^{\mathbf{b}_z} = \{4\}$	$V_4^{\mathbf{b}_z} = \{6, 8\}$
$p(V_k^{\mathbf{b}_z}) =$ $p(\widetilde{V}_k^{\mathbf{b}_z}) + p(\bar{V}_k^{\mathbf{b}_z})$	$10 =$ $(2 + 3) + 5$	$10 = 3 + 7$	$9 = 9 + 0$	$5 = 0 +$ $(3 + 2)$

balance $\mathbf{b}_u$ *has strictly positive stability radius:* $\rho_{\mathbf{b}_u}(p) > 0$*, while line balances* $\mathbf{b}_w$ *and* $\mathbf{b}_z$ *have zero stability radii:*

$$\rho_{\mathbf{b}_w}(p) = \rho_{\mathbf{b}_z}(p) = 0.$$

Optimal line balances $\mathbf{b}_s$ *and* $\mathbf{b}_u$ *with positive stability radii are better than optimal line balances* $\mathbf{b}_r$*,* $\mathbf{b}_w$ *and* $\mathbf{b}_z$ *with zero stability radii. To choose better line balance among* $\mathbf{b}_s$ *and* $\mathbf{b}_u$ *we have to calculate the exact values of their stability radii. In the following subsection, it will be shown how to do such a calculation.*

Exact Value of Stability Radius

We derive a formula for calculating the exact value of stability radius $\rho_{\mathbf{b}}(p)$. We will prove it on the basis of more precise calculation than it was used within the proof of Theorem 1.8 (see case (ii) and case (iii)). Before formulating Theorem 1.9 about exact value of stability radius, we present its proof.

As it was already mentioned in the proof of Theorem 1.8, there are only two possibilities to violate the optimality of line balance $\mathbf{b} \in B_{opt}(p)$: Either in case (ii) to violate feasibility of operation assignment $\mathbf{b}$ via increasing some operation times in order to violate inequality (1.45) for at least one subset $V_k^{\mathbf{b}}, k \in \{1, 2, ..., m_{\mathbf{b}}\}$, or in case (iii) to violate optimality (with saving feasibility) of operation assignment $\mathbf{b}$ via decreasing some operation times in such a way that another operation assignment (with number of used stations less than $m_{\mathbf{b}}$) becomes feasible for the modified vector of the processing times.

Obviously, any increase of an operation time in case (ii) keeps the obtained vector of operation times within the space R_+^n, and as follows from the proof of Theorem 1.8 (see case (ii)) the value $\delta^{\mathbf{b}}$ defined in (1.48) gives the minimal possible distance $d(\tilde{p}', \tilde{p})$

(a lower bound) for any vector $(\tilde{p}', \bar{p}) \in R^n_+$, which may be suitable for realization of case (ii).

Next, we show that lower bound $\delta^{\mathbf{b}}$ is maximal. To this end, we show that desired vector $\tilde{p}'$ may be chosen to be arbitrarily close to the ball $O_{\delta^{\mathfrak{b}}}(p)$, i.e., for any small real $\varepsilon > 0$ there exists vector $\tilde{p}' = \tilde{p}^{\varepsilon} \in R^{\tilde{n}}_+$ such that $d(\tilde{p}^{\varepsilon}, \tilde{p}) = \delta^{\mathbf{b}} + \varepsilon$ and $\mathbf{b} \notin B((\tilde{p}^{\varepsilon}, \bar{p}))$.

Let the value $\delta^{\mathbf{b}}$ in (1.48) is reached for the station S_g (with $\widetilde{V}^{\mathbf{b}}_g \neq \oslash, p(V^{\mathbf{b}}_g) < c, g \in \{1, 2, ..., m_{\mathbf{b}}\}$), i.e., $\delta^{\mathbf{b}} = \delta^{\mathbf{b}}_g$ with $\delta^{\mathbf{b}}_g$ being defined in (1.47). For any small positive real ε, we construct vector $\tilde{p} = (p^{\varepsilon}_1, p^{\varepsilon}_2, ...p^{\varepsilon}_{\tilde{n}})$, where $p^{\varepsilon}_j = p_j + \delta^{\mathbf{b}} + \varepsilon$, if $j \in \widetilde{V}^{\mathbf{b}}_g$, and $p^{\varepsilon}_i = p_i$ if $i \in \widetilde{V} \setminus \widetilde{V}^{\mathbf{b}}_g$. For the complete vector $(\tilde{p}', \tilde{p})$, feasibility of the operation assignment $\mathbf{b}$ is violated. Indeed,

$$\sum_{i \in V^{\mathbf{b}}_g} p^{\varepsilon}_i = \sum_{i \in V^{\mathbf{b}}_g} p_i + \mid \widetilde{V}^{\mathbf{b}}_g \mid \delta^{\mathbf{b}} + \mid \widetilde{V}^{\mathbf{b}}_g \mid \varepsilon = c + \varepsilon \mid \widetilde{V}^{\mathbf{b}}_g \mid > c,$$

and therefore $\mathbf{b} \notin B((\tilde{p}^{\varepsilon}, \bar{p}))$. Since the distance $d(\tilde{p}^{\varepsilon}, \tilde{p}) = \delta^{\mathbf{b}} + \varepsilon$ between vector $\tilde{p}^{\varepsilon}$ and the sphere of the ball $O_{\delta^{\mathfrak{b}}}(\tilde{p})$ is equal to ε which may be arbitrarily small.

On the other hand, feasibility of the operation assignment $\mathbf{b}$ will be definitely kept for any vector $\tilde{p}^* \in O_{\delta^{\mathfrak{b}}}(\tilde{p}) \bigcap R^{\tilde{n}}_+$, i.e., $\mathbf{b} \in B((\tilde{p}^{\varepsilon}, \bar{p}))$. (It follows directly from the proof of necessity of Theorem 1.8, see inequalities (1.49)). Hence, for calculating the exact value of stability radius we can use value $\delta^{\mathbf{b}}$ just as it is defined in (1.48). In other words if $\delta^{\mathbf{b}} < \triangle^{\mathbf{b}}$, then inequality (1.53) turns out into equality $\rho_{\mathbf{b}}(p) = \delta^{\mathbf{b}}$.

Unfortunately, it is not the case for the value $\Delta^{\mathbf{b}}$ used in (1.52) and (1.53). And there are two reasons for this, which will be considered in details as case (j) and case (jj).

(j) After decreasing of the operation times $p_j, j \in (\widetilde{V}^{\mathbf{b}^*}_p)$, by the value $\Delta(V^{\mathbf{b}^*}_p)$ (see (1.51)) we can obtain vector $\tilde{p}''$, which may have one or even more negative components (namely, if $p_j < \Delta(\widetilde{V}^{\mathbf{b}^*}_p)$). Thus, it may happen that $\tilde{p}'' \notin R^{\tilde{n}}_+$ and so $(\tilde{p}'', \bar{p}) \notin R^n_+$. Therefore, in the desired formula for the exact value of stability radius $\rho_{\mathbf{b}}(p)$, values $\Delta(V^{\mathbf{b}^*}_p)$ and $\Delta^{\mathbf{b}}$ have to be changed in order to guarantee that the desired vector of the operation times contains only non-negative components. Such a change may be done as follows.

Let $B^{(\mathbf{b})}$ denote the set of all operation assignments from set B such that each $\mathbf{b}^{(r)} \in B^{(\mathbf{b})}$ has properties $(a), (c), (d)$ and (e):

$$B^{(\mathbf{b})} = \{\mathbf{b}^{(1)}, \mathbf{b}^{(2)}, \ldots, \mathbf{b}^{(q)}\} = \{\mathbf{b}^{(r)} \in B \ : \ \mathbf{b}^{(r)} \text{ possessed properties } (a), (c), (d), (e)\}.$$

In the definition of set $B^{(\mathbf{b})}$ we do not indicate property (b) since it uses the vector $p'' \in R^n_+$, which may be deferent for different lines balances $\mathbf{b}^{(r)} \in B^{(\mathbf{b})}$. And our aim is just to find vector $p'' = (\tilde{p}'', \bar{p}) \in R^n_+$ of the operation times which is the closest to vector p such that for at least one operation assignment $\mathbf{b}^{(d)} \in B^{(\mathbf{b})}$ inclusion $\mathbf{b}^{(d)} \in B(p'')$ holds, and therefore $\mathbf{b} \notin B_{opt}(p'')$. In any operation assignment $\mathbf{b}^{(d)} \in B(p'')$ there exists subset $V^{\mathbf{b}^{(d)}}_p$ that inequality $p(V^{\mathbf{b}^{(d)}}_p) > c$ holds (see property (e)). In order to insure inclusion $p'' \in R^n_+$, we need the following additional calculation.

Let equality $\widetilde{V}^{\mathbf{b}^{(d)}}_p = \{i_1, i_2, \ldots, i_u\}$ hold, where $u = |\widetilde{V}^{\mathbf{b}^{(d)}}_p|$ and indices v of operations i_v are assigned in such a way that the following inequalities hold:

$$p_{i_1} \leq p_{i_2} \leq \ldots \leq p_{i_u}. \tag{1.54}$$

We also set $p_{i_u} = 0$. Then the closest to p vector $p'' = (\tilde{p}'', \bar{p}) \in R^n_+$ of the operation times, for which subset $V_p^{\mathbf{b}^{(d)}}$ is feasible (i.e., inequality (1.45) holds for subset $V_p^{\mathbf{b}^{(d)}}$ with vector $p'' = (\tilde{p}'', \bar{p})$ of the operation times), can be obtained if for each operation $i_q \in \widetilde{V}_p^{\mathbf{b}^{(d)}}$ we set

$$p_{i_q} = \max\{0, p_j - \hat{\Delta}(V_p^{\mathbf{b}^{(d)}})\}, \tag{1.55}$$

where j and i_q denote the same manual operation ($j = i_q$), and value $\hat{\Delta}(V_p^{\mathbf{b}^{(d)}})$ is calculated as follows:

$$\hat{\Delta}(V_p^{\mathbf{b}^{(d)}}) = \max\{\frac{\sum_{i \in V_p^{\mathbf{b}^{(d)}}} p_i - c - \sum_{\alpha=0}^{\beta} p_{i_\alpha}}{|\widetilde{V}_p^{\mathbf{b}^{(d)}}| - \beta} : \beta = 0, 1, \ldots, |\widetilde{V}_p^{\mathbf{b}^{(d)}}|\}, \tag{1.56}$$

where maximum is taken among right-hand fractions calculated for all $\beta = 0, 1, \ldots, |\widetilde{V}_p^{\mathbf{b}^{(d)}}| - 1$. Thus, in order to get the modified vector p'' with only non-negative components, we have to use (1.55) instead of (1.51) with $\hat{\Delta}(V_p^{\mathbf{b}^{(d)}})$ being used instead of $\Delta(V_p^{\mathbf{b}^{(d)}})$.

Next, we consider the second reason for the need to modify $\Delta^{\mathbf{b}}$.

(jj) Note that in the operation assignment $\mathbf{b}^{(d)} \in B^{(\mathbf{b})}$ there may exist several subsets $V_p^{\mathbf{b}^{(d)}}$, that inequality $p(V_p^{\mathbf{b}^{(d)}}) > c$ (see property (e)) and inequality $p(\bar{V}_p^{\mathbf{b}^{(d)}}) \leq c$ (see property (c)) hold. Therefore, the desired modification of the vector p will be obtained if we will use setting (1.55) for each manual operation $j \in \widetilde{V}_p^{\mathbf{b}^{(d)}}$ and for each subset $V_p^{\mathbf{b}^{(d)}}$ that inequalities $p(V_p^{\mathbf{b}^{(d)}}) > c$ and $p(\bar{V}_p^{\mathbf{b}^{(d)}}) \leq c$ hold.

Recall that $\Delta(V_p^{\mathbf{b}^{(d)}})$ defined in (1.50) with $\mathbf{b}^* = \mathbf{b}^{(d)}$ is equal to the minimal possible real number, which has to be subtracted from each processing time p_j of the manual operation $j \in \widetilde{V}_p^{\mathbf{b}^{(d)}}$ in order to obtain the closest to $\tilde{p}$ vector $\tilde{p}'' \in R^{\tilde{n}}$ of the processing times such that subset $\widetilde{V}_p^{\mathbf{b}^{(d)}}$ becomes feasible (with respect to inequality (1.45)). Just the same claim is valid for $\hat{\Delta}(V_p^{\mathbf{b}^{(d)}})$ with only one exception that $\tilde{p}'' \in R^{\tilde{n}}_+$ (in contrast to inclusion $\tilde{p}'' \in R^{\tilde{n}}$ in the case (iii) of the proof of Theorem 1.8). For all other manual operations (i.e., operations $i \in V_k^{\mathbf{b}^{(d)}}$ with $p(V_k^{\mathbf{b}^{(d)}}) \leq c$) we set $p''_i = p_i$. Obviously, to make operation assignment $\mathbf{b}^{(d)}$ feasible we have to find the closest to p vector p'' of the processing times for which all subsets $V_p^{\mathbf{b}^{(d)}}$ with $p(V_p^{\mathbf{b}^{(d)}}) > c$ will satisfy inequalities (1.45) with vector p'' (only in this case Condition (II) will hold). Therefore, we have to choose the maximal value $\Delta(\mathbf{b}^{(d)})$ among all values $\hat{\Delta}(V_p^{\mathbf{b}^{(d)}})$:

$$\Delta(\mathbf{b}^{(d)}) = \max\{\hat{\Delta}(V_p^{\mathbf{b}^{(d)}}) : p(V_p^{\mathbf{b}^{(d)}}) > c\}. \tag{1.57}$$

Similarly, we can consider all the operation assignments $\mathbf{b}^{(d)} \in B^{(\mathbf{b})}$, and calculate the minimal value $\hat{\Delta}^{\mathbf{b}}$ among all values $\Delta(\mathbf{b}^{(d)})$:

$$\hat{\Delta}^{\mathbf{b}} = \min\{\Delta(\mathbf{b}^{(d)}) : \mathbf{b}^{(d)} \in B^{(\mathbf{b})}\}. \tag{1.58}$$

As a result, we obtain the closest to p vector $p'' = (\tilde{p}'', \bar{p}) \in R^n_+$ for which there exists operation assignment $\mathbf{b}^{(d)} \in B^{(\mathbf{b})}$ that $\mathbf{b}^{(d)} \in B(p'')$, and therefore $\mathbf{b} \notin B_{opt}(p'')$ since $m_{\mathbf{b}^{(d)}} < m_{\mathbf{b}}$.

Note that if $m_{\mathbf{b}} \leq 3$, then instead of the set $B^{(\mathbf{b})}$ it is sufficient to consider subset $B^{(m_{\mathbf{b}}-1)} \subset B^{(\mathbf{b})}$ defined as follows:

$$B^{(m_{\mathbf{b}}-1)} = \{\mathbf{b}^{(r)} \in B^{(\mathbf{b})} : m_{\mathbf{b}^r} = m_{\mathbf{b}} - 1\}.$$

Indeed, let minimum in (1.58) be reached for some operation assignment $\mathbf{b}^{(r)} \in B^{(\mathbf{b})} \setminus B^{(m_{\mathbf{b}}-1)}$:

$$V = V_1^{\mathbf{b}^{(r)}} \cup V_2^{\mathbf{b}^{(r)}} \cup \ldots \cup V_{m_{\mathbf{b}^{(r)}}}^{\mathbf{b}^{(r)}},$$

with $1 \leq m_{\mathbf{b}^{(r)}} \leq m_{\mathbf{b}} - 2$. Via increasing the number of stations we can obtain another operation assignment, say $\mathbf{b}^{(e)}$, without increasing value $\Delta(\mathbf{b}^{(r)})$, i.e., $\Delta(\mathbf{b}^{(r)}) = \Delta(\mathbf{b}^{(e)})$. To this end, we can partition some set $V_k^{\mathbf{b}^{(r)}}$ with $|V_k^{\mathbf{b}^{(r)}}| \geq 2$ into two parts and add these parts to two stations with saving the value $\Delta(\mathbf{b}^{(r)}) = \Delta(\mathbf{b}^{(e)})$. We have only to save the precedence constraints given by digraph $D = (V, A)$. By repeating this procedure $(m_{\mathbf{b}^{(r)}} - m_{\mathbf{b}} + 1)$ times we obtain operation assignment $\mathbf{b}^{(f)} \in B^{(m_{\mathbf{b}}-1)}$ with $m_{\mathbf{b}^{(f)}} = m_{\mathbf{b}} - 1$ and $\Delta(\hat{\mathbf{b}}^{(f)}) = \Delta(\hat{\mathbf{b}}^{(r)})$. Thus, instead of (1.58) we can use (1.59):

$$\hat{\Delta}^{\mathbf{b}} = \min\{\Delta(\mathbf{b}^{(d)}) : \mathbf{b}^{(d)} \in B^{(m_{\mathbf{b}}-1)}\} \tag{1.59}$$

with value $\hat{\Delta}^{\mathbf{b}}$ being the same in (1.58) and (1.59).

Obviously, using (1.59) takes less calculation. Summarizing, we conclude that if $\hat{\Delta}^{\mathbf{b}} \leq \delta^{\mathbf{b}}$, then $\rho_{\mathbf{b}}(p) = \hat{\Delta}^{\mathbf{b}}$, where $\delta^{\mathbf{b}}$ is defined in (1.48) and $\hat{\Delta}^{\mathbf{b}}$ is defined in (1.59). Thus, the following theorem is proven.

Theorem 1.9 *If optimality of a line balance $\mathbf{b} \in B_{opt}(p)$ is stable, then $\rho_{\mathbf{b}}(p) = \min\{\delta^{\mathbf{b}}, \hat{\Delta}^{\mathbf{b}}\}$ with $\delta^{\mathbf{b}}$ being defined in (1.48) and $\hat{\Delta}^{\mathbf{b}}$ in (1.59).*

Example 1.6 (continued). *We illustrate equality (1.56), which looks rather complicated, by example. In order to calculate value $\hat{\Delta}^{\mathbf{b}_s}$ for line balance $\mathbf{b}_s \in B_{opt}(p)$ in Example 1.6 (see Table 1.5) we have to compare line balance $\mathbf{b}_s$ with, e.g., operation assignment $\mathbf{b}^{(d)}$ defined as follows: $V_1^{\mathbf{b}^{(d)}} = \{2, 5\}$, $V_2^{\mathbf{b}^{(d)}} = \{1, 3, 6, 7\}$ and $V_3^{\mathbf{b}^{(d)}} = \{4, 8\}$. Operation assignment $\mathbf{b}^{(d)}$ possessed properties $(a), (c), (d), (e)$, and equality $m_{\mathbf{b}^{(d)}} = m_{\mathbf{b}} - 1$ holds. Therefore, $\mathbf{b}^{(d)} \in B^{(m_{\mathbf{b}}-1)}$. To find the closest to p vector p'' such that $\mathbf{b}^{(d)} \in B(p'')$ (see property (b)), first, we try to use (1.50) for $\Delta(V_p^{\mathbf{b}^{(d)}})$ with $p = 2$:*

$$\Delta(V_2^{\mathbf{b}^{(d)}}) = \frac{(2 + 3 + 3 + 7) - 10}{2} = \frac{15 - 10}{2} = 2, 5.$$

Using (1.51) gives $p_1'' = 2 - 2.5 = -0.5$ and $p_3'' = 3 - 2.5 = 0.5$. Thus, component p_1'' of the desired vector p'' becomes negative.

Therefore, we are forced to use (1.55) instead of (1.51), and (1.56) instead of (1.50). On the basis of (1.56) we calculate $\hat{\Delta}(V_2^{\mathbf{b}^{(d)}})$ as follows (note that for the order (p_1, p_3) of the components of vector p inequality from (1.54) holds: $p_1 < p_3$):

$$\hat{\Delta}(V_2^{\mathbf{b}^{(d)}}) = \max\{\frac{15 - 10 - 0}{2 - 0}; \frac{15 - 10 - 2}{2 - 1}\} = \max\{2.5;\ 3\} = 3.$$

Using (1.55) gives $p_1'' = \max\{0;\ 2 - 3\} = 0$ and $p_3'' = \max\{0;\ 3 - 3\} = 0$. We obtain both non-negative components p_1'' and p_3''. Continuing in a similar way for subset $V_3^{\mathbf{b}^{(d)}}$ we

obtain $\hat{\Delta}(V_3^{\mathbf{b}^{(d)}}) = 1$ *and* $p_4'' = 8$. *Using (1.57) gives* $\Delta(\mathbf{b}^{(d)}) = \max\{3;\ 1\} = 3$. *Thus, we obtain the closest to* p *vector* $p'' = (0, 3, 0, 8, 5, 3, 7, 2) \in R_+^8$ *that inclusion* $\mathbf{b}^{(d)} \in B(p'')$ *and property* (b) *hold.*

Remark 1.3 In Definition 1.4 of the stability radius we use open stability ball $O_\rho(\tilde{p})$. Indeed, if $\hat{\Delta}^{\mathbf{b}} \leq \delta^{\mathbf{b}}$, then $\rho_{\mathbf{b}}(p) = \hat{\Delta}^{\mathbf{b}}$, and at the boundary (sphere) of the ball $O_{\rho_{\mathbf{b}}(p)}(\tilde{p})$, line balance $\mathbf{b}$ may lose its optimality (see case (iii) in Theorem 1.9). However, if $\hat{\Delta}^{\mathbf{b}} > \delta^{\mathbf{b}}$, then $\rho_{\mathbf{b}}(p) = \delta^{\mathbf{b}}$, and so at the boundary of the ball $O_{\rho_{\mathbf{b}}(p)}(\tilde{p})$ line balance $\mathbf{b}$ is definitely optimal. Thus, if $\hat{\Delta}^{\mathbf{b}} > \delta^{\mathbf{b}}$, we can consider closed stability ball with maximal radius $\rho_{\mathbf{b}}(p)$. Since a priory we do not know either $\hat{\Delta}^{\mathbf{b}} > \delta^{\mathbf{b}}$ or $\hat{\Delta}^{\mathbf{b}} \leq \delta^{\mathbf{b}}$, we are forced to use open stability ball $O_{\rho_{\mathbf{b}}(p)}(\tilde{p})$ in Definition 1.4 of stability radius $\rho_{\mathbf{b}}(p)$. Obviously, for any $\rho < \rho_{\mathbf{b}}(p)$ one can consider closed stability ball $O_\rho(\tilde{p})$ of the line balance $\mathbf{b}$.

Bounds on Stability Radius

In Corollary 1.6, we already present a lower bound of stability radius $\rho_{\mathbf{b}}(p)$. To calculate value $\delta^{\mathbf{b}}$ we only need line balance $\mathbf{b}$, cycle time c and operation times p. Then calculation $\delta^{\mathbf{b}}$ can be done as follows.

Algorithm 1.2

Input: Line balance $\mathbf{b}$, cycle time c, processing times p.
Output: Lower bound $\delta^{\mathbf{b}}$ on stability radius $\rho_{\mathbf{b}}(p)$.

Step 1: Calculate station time for each $S_k, k = 1, 2, ..., m$.

Step 2: Using equality (1.47), calculate $\delta_k^{\mathbf{b}}$ for each station S_k.

Step 3: Choose minimal value $\delta^{\mathbf{b}}$ among values $\delta_k^{\mathbf{b}}, k = 1, 2, ..., m$. **STOP**.

Obviously, steps 2 and 3 of Algorithm 1.2 take $O(m)$ time. So, if station times are assumed to be given within input data of the algorithm, then calculation $\delta^{\mathbf{b}}$ takes $O(m)$ time, otherwise, realization of Algorithm 1.2 takes $O(n)$ time. Unfortunately, calculation value $\Delta^{\mathbf{b}}$ used in the lower bound (1.53) may be essentially harder. In the worst case, we have to know all the operation assignments $B^{(m_{\mathbf{b}}-1)}$. Fortunately, if we want to calculate stability radii for several line balances from set $B_{opt}(p)$, then set $B^{(m_{\mathbf{b}}-1)}$ remains the same for all $\mathbf{b}_r \in B_{opt}(p)$. So, we can use notation $B^{(m-1)}$ instead of $B^{(m_{\mathbf{b}}-1)}$ since only number of stations in $\mathbf{b}^{(d)} \in B^{(m-1)}$ is of importance.

If operation assignment $\mathbf{b}^{(d)} \in B^{(m-1)}$ is given, then calculation $\Delta(V_p^{\mathbf{b}^{(d)}})$ takes $O(n)$ time (steps 1, 2 and 3 are similar to those used for calculating $\delta^{\mathbf{b}}$ in Algorithm 1.2, with only exception that equality (1.50) is used instead of (1.47)) at step 2. Thus, to calculate lower bound (1.53) takes $O(n|B^{(m-1)}|)$ time.

Complexity of calculation of the exact value of stability radius via Theorem 1.9 takes $O(n^2 m_{\mathbf{b}}|B^{(m-1)}|)$ time, since in addition we have to calculate value $\hat{\Delta}(V_k^{\mathbf{b}^{(d)}})$ for each index $k = 1, 2, \ldots, m$. In both cases, to enumerate operation assignments from set $B^{(m-1)}$ is hard problem, and we need simpler bounds on stability radius $\rho_{\mathbf{b}}(p)$. The following lower bound may be calculated in $O(n)$ time, but it is valid only for the case when

$$m_{\mathbf{b}} = \lceil \frac{p(V)}{c} \rceil.$$

Corollary 1.7 *If* $m_{\mathbf{b}} = \lceil \frac{p(V)}{c} \rceil$, *then*

$$\rho_{\mathbf{b}}(p) \geq \min\{\delta^{\mathbf{b}}; \frac{p(V) - c(m_{\mathbf{b}} - 1)}{\tilde{n}}\}.$$

PROOF. In order to make operation assignment $\mathbf{b}^{(d)} \in B^{(\mathbf{b})}$ with $m_{\mathbf{b}^{(d)}} = m_{\mathbf{b}} - 1$ feasible for the modified vector of the processing times we have to decrease the sum of all station times from $p(V)$ to $c(m_{\mathbf{b}} - 1)$. Therefore, to obtain the closest to p vector p^* of operation times we have to subtract from each processing time p_i of a manual operation $i \in \widetilde{V}$ at least value

$$\frac{p(V) - c(m_{\mathbf{b}} - 1)}{\tilde{n}}.$$

Thus,

$$d(p, p^*) \geq \frac{p(V) - c(m_{\mathbf{b}} - 1)}{\tilde{n}}$$

and Corollary 1.7 follows from Theorem 1.9.

◇

Due to Corollary 1.7, we obtain the following special case of SALBP when stability radius can be calculated in $O(n)$ time.

Corollary 1.8 *If*

$$m_{\mathbf{b}} = \lceil \frac{p(V)}{c} \rceil \quad \textit{and} \quad \delta^{\mathbf{b}} \leq \frac{p(V) - c(m_{\mathbf{b}} - 1)}{\tilde{n}},$$

then $\rho_{\mathbf{b}}(p) = \delta^{\mathbf{b}}$.

In Section 1.3, for infinitely large stability radius of an optimal schedule with makespan criterion for a general shop scheduling problem the necessary and sufficient conditions have been proven. For the optimal line balance in SALBP, infinitely large stability radius is impossible, since the following upper bound of stability radius holds.

Corollary 1.9 *If* $\mathbf{b} \in B_{opt}(p)$, *then* $\rho_{\mathbf{b}}(p) \leq \min\{\delta^{\mathbf{b}}, \max_{i \in \widetilde{V}} p_i\}$.

PROOF. This bound of stability radius is based on the fact that $\delta^{\mathbf{b}}$ is exact value of stability radius in the case of $\delta^{\mathbf{b}} < \hat{\Delta}^{\mathbf{b}}$ and that $\hat{\Delta}^{\mathbf{b}} \leq \max_{i \in \widetilde{V}} p_i$ due to Definition 1.4

◇

As it was shown, $\delta^{\mathbf{b}}$ may be easy calculated for each fixed optimal line balance $\mathbf{b} \in B_{opt}(p)$. Nevertheless, we can ask: How much the largest value $\delta^{\mathbf{b}} = \delta^*$ can be for arbitrary optimal line balance $\mathbf{b} \in B_{opt}(p)$? Without considering digraph $D = (V, A)$ we can claim that δ^* may be equal at most to $c - \max_{i \in \widetilde{V}} p_i = c - p_{j^*}$. Indeed, in the worst case operation $j^* \in \widetilde{V}$ may be a single one in some subset $V_k^{\mathbf{b}} : V_k^{\mathbf{b}} = \{j^*\}$ (and the same for other manual operations). In such a case $\delta^* = c - p_{j^*}$. Thus, Corollary 1.9 may be represented in the following form.

Corollary 1.10 *If* $\mathbf{b} \in B_{opt}(p)$*, then*

$$\rho_{\mathbf{b}}(p) \leq \min\{c - \max_{i \in \widetilde{V}} p_i, \ \max_{i \in \widetilde{V}} p_i\}.$$

Obviously, the largest value of the upper bound from Corollary 1.10 is obtained when $\max_{i \in \widetilde{V}} p_i = c/2$. Thus, we conclude that in any case:

$$\rho_{\mathbf{b}}(p) \leq c/2. \tag{1.60}$$

Calculating upper bound from Corollary 1.9 takes $O(n)$ times, and that from Corollary 1.10 takes $O(\log \tilde{n})$ time. If we know at least one operation assignment $\mathbf{b} \in B^{m_{\mathbf{b}}-1}$, then we can calculate the following upper bound.

Corollary 1.11 *If* $\mathbf{b} \in B_{opt}(p)$ *and* $\mathbf{b}^{(d)} \in B^{m_{\mathbf{b}}-1}$*, then*

$$\rho_{\mathbf{b}}(p) \leq \min\{\delta^{\mathbf{b}}, \Delta(\mathbf{b}^{(d)})\}.$$

Calculating the latter upper bound from Corollary 1.11 takes $O(n)$ time, but quality of this bound essentially depends on the used operation assignment $\mathbf{b}^{(d)} \in B^{m_{\mathbf{b}}-1}$.

Example 1.6 (continued). *Returning to the Example 1.6, we can calculate lower bounds for optimal line balances* $\mathbf{b}_s$ *and* $\mathbf{b}_u$*. Due to Corollary 1.7, we obtain*

$$\rho_{\mathbf{b}_s}(p) \geq \min\{\min\{\frac{10-8}{1}, \frac{10-2}{2}, \frac{10-9}{1}\}; \frac{34-10\cdot(4-1)}{4}\} = \min\{1; 1\} = 1;$$

$$\rho_{\mathbf{b}_u}(p) \geq \min\{\min\{\frac{10-7}{1}, \frac{10-9}{2}, \frac{10-9}{1}\}; \frac{34-10\cdot(4-1)}{4}\} = \min\{\frac{1}{2}; 1\} = \frac{1}{2}.$$

It is easy to convince that for both line balances $\mathbf{b}_s$ *and* $\mathbf{b}_u$ *conditions of Corollary 1.8 hold. Thus,* $\rho_{\mathbf{b}_s}(p) = \delta^{\mathbf{b}_s} = 1$ *and* $\rho_{\mathbf{b}}(p) = \delta^{\mathbf{b}_u} = \frac{1}{2}$.

It is worth noting that all conditions obtained in this section (except Theorem 1.9 and Corollary 1.6) may be tested in polynomial time, which is important for real-world assembly lines with large numbers of operations and stations. Moreover, for exact value of stability radius, feasibility of the line balance $\mathbf{b}$, which is defined by the value $\delta^{\mathbf{b}}$, may be tested in polynomial time even in Theorem 1.9.

In practice, the tendency at the design stage must be to find an optimal line balance which stability is as much as possible. Of course, the common objective is to assign to each station a set of operations with roughly the same total operation time. However, due to the above results, we have to defer stations with manual operations and stations without manual operations. Theorem 1.9 shows that for the station with manual operations it is

desirable to have some slack between cycle time and station time. The larger this slack is, the larger stability radius of the line balance may be. On the other hand, for the stations with only automated operations, such a slack may be as small as possible, which gives the possibility to increase slacks for stations loaded by manual operations. Since stability radius cannot be larger than $c/2$ (see (1.60)), one has to pay special attention to the manual operations with possible variations of processing times more than $c/2$ (such operations may cause instability of optimality of the line balance at hand). If it is possible at the design stage, such an operation has to be divided into shorter manual operations.

If line balance will be used for a long time for assembling the same finished product, it is desirable at the design stage, to construct several optimal line balances, and select among them that which has better stability characteristics. So, it is useful to develop algorithms, which construct a set of optimal line balances (instead of only one optimal line balance), in order to carry out stability analysis for them. Or, better yet, it is useful to include in the branch-and-bound or other algorithms used for SALBP specific rules in order to construct an optimal line balance with large stability radius.

In a concrete study, the set of manual operations can be reduced (e.g., only critical manual operations may be considered) or, on the contrary, set may be completed by some unstable automated operations. By changing the set of operations with variable times, the designer can study the influence of different operations on stability of optimality and feasibility of line balances.

1.6 Comments and References

It is well-known that different scheduling problems may be represented as extremal problems on mixed (disjunctive) graphs.

As it was mentioned in [CCLe95, pp. 277-293], the disjunctive graph approach is the most suitable one for traditionally difficult scheduling problems.

A mixed (or *disjunctive*) graph is often introduced to model a deterministic scheduling problem (see [BDP96, Pin95b, RS64, TSS94]).

Problem $\mathcal{G}//\Phi$ is unary NP-hard for any given regular criterion Φ considered in scheduling theory, [LLRKS93, TSS94], the running time of calculating an optimal schedule $s = (c_1(s), c_2(s), \ldots, c_q(s))$ may be restricted by an $O(q^2)$-algorithm (see [[TSS94], p. 285]).

Due to the particular importance of set E_s, it is called a *signature* of a schedule s [BSW96, Sot91b, SLG95, STW98, Sus72].

The results of [BSW96, KSW95, Sot91b, SSW97] were devoted to the stability of an optimal digraph $G_s(p)$ which represents an optimal solution to problem $\mathcal{G}//\Phi$.

Next, we introduce these notions in a formal way (see [Sot91b]).

In [Sot91b], the calculation of $\varrho_s(p)$ has been reduced to a non-linear programming problem. Next, we give the presentation from [Sot91b, STW98] for the case of the general shop problem $\mathcal{G}//\Phi$ when the set of all operations Q is partitioned into n technological routes $Q^{(i)}$ of a job $J_i, i \in \{1, 2, \ldots, n\}$.

Formulas for calculating the stability radius for the makespan criterion and the characterization of the extreme values of $\widehat{\varrho}_s(p)$ have been proven in [Sot91b, 84]. The same questions for the mean flow time criterion have been considered in [BSW96].

a formal definition of the stability radius, which is the maximal value of the radius of such a stability ball (see [Sot91b]).

In order to calculate $\varrho_s(p)$, it is sufficient to know the optimal value of the objective function $f(x_1, x_2, \ldots, x_q)$ of the following non-linear programming problem (see [Sot91b]):

In [Sot91b, 84], the stability radius for criterion $\mathcal{C}_{max}$ has been considered and here we survey these results.

Next, we present necessary and sufficient conditions for equality $\widehat{\varrho}_s(p) = 0$ proven in [Sot91b].

In [Sot91b], the following characterization of an infinitely large stability radius was proven.

In [KSW95], it has been shown that for problem $\mathcal{J}//\mathcal{C}_{max}$, there are necessary and sufficient conditions for $\widehat{\varrho}_s(p) = \infty$ which can be verified in $O(q^2)$ time.

Moreover, In [KSW95], the analogies to Theorems 1.2 and 1.3 for the job shop problem $\mathcal{J}//\mathcal{L}_{max}$ with minimizing maximum lateness (see [LLRKS93]) have been proven and it has been shown that there does not exist an optimal schedule s with $\varrho_s(p) = \infty$ for all other regular criteria (see [LLRKS93]), which are considered in classical scheduling theory.

Formulas for calculating $\widehat{\varrho}_s(p)$ have been derived in [Sot91b].

Formulas for calculating the stability radius for criterion $\mathcal{C}_{max}$ have been derived in [Sot91b, 84].

In this section, we survey results from [BSW96, STW98], where the stability radius $\overline{\varrho}_s(p)$ for criterion $\sum \mathcal{C}_i$ has been studied.

Continuing similarly as in the above proof of Theorem 1.4 the following claim may be proven in [BSW96].

The extreme values of $\overline{\varrho}_s(p)$ were considered in [BSW96, STW98]. The following necessary and sufficient conditions for equality $\overline{\varrho}_s(p) = 0$ have been derived in [BSW96].

Section 1.5 was based on the results proven in [SD01, SDP05].

Assembly line balancing problem with variable operation times was considered, e.g., in [GTL96, LJ91, SEDE99, TGK95]. In [GTL96, TGK95], fuzzy set theory was used to represent uncertainty of the operation times. Genetic algorithms were used either to minimize the total operation time for each station [GTL96] or to minimize the efficiency of the fuzzy line balance [TGK95]. In [LJ91], the entire decision process has been decomposed into two parts: Deterministic problem and stochastic problem. For the former problem, integer programming is used to minimize number of stations. For the latter problem, which takes into account variations of the operation times over different products, queuing network analysis is used to determine the necessary capacity of the material-handling system. In [SEDE99], dynamic programming and branch-and-bound methods have been used to minimize the total labor cost and the expected incompletion cost arising from operations not completed within cycle time c.

(e.g., via branch-and-bound algorithm [BT00, PDK83, SEDE99, SK98], or via integer programming [DGHL02, LJ91, PDK83], etc.).

Some preliminary results for such a stability analysis have been obtained in [SD01]. In [SD01], slightly different definitions of stability radius and stability ball of an optimal line balance have been used. Namely, it was assumed that . Therefore in that paper generally smaller stability radii were obtained (in particular, it cannot be greater than). The above Definition 1 from Section 2 seems to be more appropriate for practical assembly lines.

The scheduling theory has received a lot of attention among OR practitioners, management scientists, production and operations research workers and mathematicians since the early 1950s. However, the utilization of classical scheduling theory in most production environments is minimal (see [ML93, PL94, Pin95a]). MacCarthy and Liu [ML93] aim the gap between scheduling theory and scheduling practice. They also discuss some research issues which attempt to make scheduling theory more useful in practice. Next, we describe some recent trends in scheduling research which try to make it more relevant and applicable.

For an uncertain scheduling environment *stochastic* models are introduced, where the processing times (and some other parameters) are assumed to be random variables with known probability distributions. For example, such stochastic models for a single machine with the minimization of mean flow time are considered by Chand et al. [CTU96], by Li and Cao [LC95], and with the minimization of earliness-tardiness penalties by Cai and Tu [CT96] as well as by Robb and Rohleder [RR96]. Since it is possible for a company to estimate the times at which jobs are expected to arrive, Chand et al. [CTU96] develop a decomposition approach such that a large problem can be solved by combining optimal solutions of several smaller problems. The model of Robb and Rohleder [RR96] consists of a *probabilistic dynamic scheduling problem* with non-regular performance measures. Using simulation, they explore the robustness of the heuristics with respect to uncertainty in the durations of the operations.

Schmidt [Sch00] reviews some results related to *deterministic* scheduling problems where the machines are not continuously available for processing. The complexity of single and multi-machine problems is analyzed considering criteria depending on the completion times and the due dates. Chu and Gordon [CG00] consider a single machine problem including both due date assignment and the scheduling decision. It is assumed that the due dates are proportional to the job processing times. The objective is to minimize the weighted earliness-tardiness and the penalty related to the size of the dates with respect to the processing times. Jain and Meeran [JM99] present a concise overview of job shop scheduling techniques and the best computational results obtained.

The scheduling problem with an *availability constraint* is very important, as it happens often in the industry. For example, a machine may not be available during the scheduling horizon due to a breakdown (stochastic) or preventive maintenance (deterministic). In an *on-line* setting, machine availabilities are not known in advance. Unexpected machine breakdowns are a typical example of events that arise on-line. Sometimes schedulers have partial knowledge of the availabilities, i.e. they have some 'look-ahead' information. They might know the next time interval where a machine requires maintenance or they might know when a broken machine will be available again [San95]. In an *off-line* setting, one assumes complete information, i.e. all machine availabilities are known prior to the schedule generation [Sch00].

Several *on-line models* have been proposed, and the main difference between these models are the assumptions on the information that becomes available to the scheduler. For a description of these on-line models, we refer to the survey by Sgall [Sga98]. According to [CV97], *on-line* means that jobs arrive over time, and all job characteristics become known at their arrival time [CV97]. Jobs do not have to be scheduled immediately upon arrival. At each time a machine is idle and a job is available, the algorithm decides which one of the available jobs is scheduled, if any. An on-line algorithm for the problem

of scheduling jobs on identical parallel machines with the objective of minimizing the makespan is proposed and analyzed by Chen and Vestjens [CV97]. This problem is NP-hard when the *off-line* version is considered, although it can be solved in polynomial time by an on-line algorithm if preemption is allowed [CV97].

Seiden [Sei98] studies on-line scheduling of jobs with fixed start and completion times. Jobs must be scheduled on a single machine which runs at most one job at a given time. The problem is on-line since jobs are unknown until their start times. Each job must be started or rejected immediately when it becomes known. The goal is to maximize the sum of the value the *payoff* (the sum of the values of those jobs which run to completion).

Scheduling problems with *controllable processing times* have received an increasing attention during the last decade. It is often assumed that the actual possible processing time of a job can be continuously controlled, i.e. it can be any number in a given interval. Recent results are presented in [DHM96, KDV00, Str95, Tri94].

Traditional scheduling procedures consider static and deterministic future conditions even though this may not be the case in actual scheduling problems. After a description, the preplanned schedule can become inapplicable to the new conditions. As Graves [Gra81] stated, there is no scheduling problem but rather a *rescheduling problem*. Responding to such dynamic factors immediately as they occur is called *real-time scheduling*. An on-line simulation methodology is proposed by Davis and Jones [DJ88] to analyze several scheduling rules in a stochastic job shop. The job shop rescheduling problem is considered as a particularly hard combinatorial optimization problem (Parunak and van Dyke [PD91]). The production rescheduling problem deals with uncertainty caused by the exterior business environment and interior production conditions. Since it has practical applications, the rescheduling problem is studied by many authors (see e.g. [LLLH00, PD91, SK94]).

A *reactive scheduling approach* is developed by Smith et al. [SOM$^+$90], which uses different knowledge sources and aims to make decisions faster with less emphasis on optimality. For the knowledge-based systems, the most difficult operation is to decide which knowledge source has to be activated. A discussion of knowledge-based reactive scheduling systems can be found in Blazewicz et al. [BESW93] as well as Szelke and Kerr [SK94]. Bean et al. [BBMN91] propose a 'match-up' heuristic method for scheduling problems with disruptions. They show that assuming enough idle time is present in the original schedule and disruptions are sufficiently spaced over time, the optimal rescheduling strategy is to match-up with the preschedule at some time in the future. The objective in [AG99] is to create a new schedule that is consistent with the order production planning decisions like material flow, tooling and purchasing. When a machine breakdown forces a modified flow shop out of the prescribed state, the proposed strategy reschedules a part of the initial schedule to match-up with the preschedule at some point.

Fuzzy scheduling techniques proposed in the literature either fuzzify directly the existing scheduling rules, or solve mathematical programming problems to determine the optimal schedules. The optimality of a fuzzy logic alternative to the usual treatment of uncertainties in a scheduling system using probability theory was examined by Ozelkan and Duckstein [OD99]. The purpose of the latter paper was to investigate necessary optimality conditions of fuzzy counterparts of 'classical' dispatching rules, such as the shortest processing time (SPT) and the earliest due date (EDD). Essentially, *any element of a scheduling problem may be uncertain.*

Dumitru and Lubau [DL82] propose fuzzy mathematical models to solve the job shop problem. Grabot and Geneste [GG94] use a fuzzy rule-based approach to find a compromise between different job shop dispatching rules. Kuroda and Wang [KW96] also analyze fuzzy job shop problems using a branch-and-bound algorithm to obtain results for lateness related criteria. A mathematical programming approach to a single machine scheduling problem with *fuzzy precedence relation* is given in [IT95]. Job shop scheduling with both fuzzy processing times and fuzzy due dates are proposed in [SK00]. Sakawa and Kubota [SK00] formulate a multiobjective fuzzy job shop problem as three-objective ones which not only maximizes the minimum agreement index but also maximizes the average agreement index and minimizes the maximum fuzzy completion time. Generally, the topic of fuzzy scheduling has received much attention during the last decade. Slowinski and Hapke [SH99] collect the main works.

In most of the classical shop scheduling models, it is assumed that an individual processing time incorporates all other time parameters (lags) attached to a job or to an operation. In practice, however, such parameters often have to be viewed separately from the actual processing times. For example, if for an operation some pre-processing and/or post-processing is required, then we obtain a scheduling model with *set-up* and/or removal times separated. Strusevich [Str99] considers a two-machine open shop problem with involved interstage transportation times. He assumes that there is a known *time lag* (transportation time) between the completion of an operation and the beginning of the next operation of the same job.

The majority of scheduling research assumes *set-up* as negligible or as a part of the processing time. While this assumption simplifies the analysis, it adversely affects the solution quality for many applications which require an explicit treatment of set-up times. Such applications, coupled with the emergence of production concepts like time-based competition and group technology, have motivated an increasing interest to include set-up considerations in scheduling problems. The paper [AGA98] provides a comprehensive review of the literature on scheduling problems involving set-up times (set-up costs). In [All97], Allahverdi considers a two-machine flow shop problem with the objective to minimize the expected makespan where machines suffer breakdowns and the job set-up and removal times are separated from the processing times. The same author [All95] proposes a dominance relation where no assumption about the breakdown processes is made. In general, such a dominance relation does not yield optimal schedules. However, if certain assumptions about the breakdowns distributions and counting processes hold, it is possible to obtain an optimal schedule.

Decision-makers often consider multiple objectives when making scheduling decisions. However, very little research has been done in multiple machine environments with *multiple objectives*. Allahverdi and Mittenthal [AM98] consider a two-machine flow shop scheduling problem, where machines suffer random breakdowns and processing times are constant, with respect to both the makespan and the maximum lateness objective functions. Kyparisis and Koulamas [KK00] study the two-machine open shop problem with a hierarchical objective: Minimize the total completion time subject to minimum makespan $\mathcal{O}2//\sum \mathcal{C}_i|\mathcal{C}_{max}$.

Cheng and Shakhlevich [CS98a] consider a special class of flow shop problems, known as the *proportionate flow shop*. In such a shop, each job flows through the machines in the same order and has equal processing times on the machines. It is assumed that

all operations of a job may be compressed by the same amount which will incur an additional cost. The objective is to minimize the makespan of the schedule together with a compression cost function which is nondecreasing with respect to the amount of compression. A *bicriterion approach* to solve the single machine scheduling problem in which the job release dates can be compressed while incurring additional costs, is considered in [CS98b].

Stein and Wein [SJ97] give a proof that, for any instance of a rather general class of scheduling problems, there exists a schedule with a makespan at most twice that of the optimal value and of a total weighted completion time at most twice that of the optimal value.

Brucker and Krämer [BK96b] derive complexity results for *resource-constrained scheduling problems* with a fixed number of operation types in which either the processing times are bounded or the number of processors is fixed. They consider shop problems with multiprocessor operations, in which either the number of jobs or the number of stages is fixed. They present polynomial time algorithms for these problems with makespan, mean flow time, weighted number of tardy operations, and sum of tardiness as objective functions.

The papers above address problems of practical importance in planning, scheduling, and control. It is therefore important to produce schedules that are both stable (robust) and adaptable to system disturbances. More importantly, it offers unique properties that lead to a more effective planning and control methods for systems under uncertainty.

An extensive survey of the obtained results within such an a posteriori analysis is given in [SLG95]. Greenberg [Gre97] categorizes types of postoptimal sensitivity analyses and gives a survey of the literature started in the late 1970's. A primary concern of sensitivity analysis is how the optimal solution values change when the data changes. The subject of post-solution analysis includes debugging a scenario, such as when it is anomalous, unbounded or infeasible.

In spite of obvious practical importance, the literature on stability analysis in scheduling is rather small. Outside the considered approach, one can mention [KRKvHW94, Mel78, PQ78]: In [KRKvHW94], the sensitivity of a heuristic algorithm with respect to the variation of the processing time of one job is investigated, in [Mel78] the stability of an optimal schedule for the flow shop problem $\mathcal{F}//\mathcal{C}_{max}$ is considered, and in [PQ78] the results for the traveling salesman problem are used for a one machine scheduling problem with minimizing tardiness (see [LLRKS93]).

In general, studying a scheduling problem with *uncertain processing times* and *its sensitivity analysis* is of importance. The reasons can be illustrated by giving references to practical applications. In many cases the data used are imprecise due to uncertainty with respect to the exact parameter values or due to errors in the measurement. In industrial applications of mathematical programming models, there are almost always uncertain elements that are assumed away or suppressed in the formal description of the model (see [Wag95]).

We have to emphasize that the random processing times $p_i, i \in Q$, in problem $\mathcal{G}/a_i \leq p_i \leq b_i/\Phi$ are due to external forces in contrast to scheduling problems with *controllable* processing times, see e.g. [DHM96, IMN87, IN86, Jan88, Str95, Tri94], where the objective is to choose both the optimal processing times (which are under the control of a decision-maker) and an optimal schedule for the chosen processing times. Both of the above parts of

a solution are supposed to be arguments in the objective function which is non-decreasing in the job completion times and non-increasing in the operation processing times.

To model scheduling in an uncertain environment, a *two-person non-zero sum game* is introduced by Chrysslouris et al. [CDL94], where the decision-maker was considered as player 1 and the 'nature' as player 2.

Next, we observe known results for *makespan* minimization under 'strict uncertainty' of the numerical input data. Lai and Sotskov [LS99] use a weighted mixed graph G for representing the input data of a job shop problem which implies a one-to-one correspondence between the set of semiactive schedules S and circuit-free digraphs $\Lambda(G)$. Since the optimality of a schedule $s \in S$ for the makespan criterion depends on the critical path in the corresponding digraph G_s, the analysis in [LS99] is focused on the set of paths in $G_s \in \Lambda(G)$ which may be critical.

In [LS99], the critical path method [Dij59] is modified for constructing a minimal digraph containing only possible candidates of critical paths. A minimal set of makespan optimal schedules for uncertain numerical input data is characterized in [LS99], where an exact and a heuristic algorithm are developed for problem $\mathcal{J}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$. Note that the approach developed in [LS99] is based on the stability property of a makespan optimal schedule, which is theoretically investigated in [KSW95, Sot91b, SWW98] and in some other papers (see [SLG95, STW98] for surveys of stability analysis for scheduling problems).

Briefly, the main issue of the research presented in [LS99] is to simplify the digraph G_s due to the existence of two types of dominance relations between its paths (see Section **??** and 3.2). In this dissertation, we perform a further step in this direction by focusing on two types of dominance relations between feasible digraphs (schedules) (see Section 3.4 below). This step is useful for shop scheduling problems under 'strict uncertainty' with both C_{max} and $\sum C_i$ criteria since it allows to reduce significantly the number of schedules which are sufficient to consider as candidates for a solution.

As follows from [BHTW99a, BHTW99b, BK96a], a reduction of the digraphs may be essential even for all non-negative perturbations of the processing times: $0 \leq p_i < \infty$. Bräsel et al. [BHTW99a], Bräsel et al. [BHTW99b] and Bräsel and Kleinau [BK96a] introduce the set of so-called *'irreducible' schedules* for a classical job shop problem $\mathcal{J}//\mathcal{C}_{max}$ and for an open shop problem $\mathcal{O}//\mathcal{C}_{max}$: For any non-negative processing times, this set contains at least one optimal schedule. On the basis of computations with $n \leq 3$ and $m \leq 7$, it is shown that only a relatively small part of semiactive schedules is irreducible for an open shop and this part becomes even relatively smaller when the size of the problem grows. By computational experiments [BHTW99a], it is demonstrated that the hardness of a classical job shop problem essentially depends on the cardinality of the set of irreducible schedules. Using the above extension of the three-field notation, we can say that the classical job shop problem $\mathcal{J}/0 \leq p_i < \infty/\mathcal{C}_{max}$ is a subject of [BHTW99a] and the open shop problem $\mathcal{O}/0 \leq p_i < \infty/\mathcal{C}_{max}$ is a subject of [BHTW99a, BK96a].

Kouvelis et al. [KDV00] focus on manufacturing environments where job processing times are uncertain. In these settings, scheduling decision-makers are exposed to the risk that an optimal schedule with respect to a deterministic or stochastic model will perform poorly when evaluated relative to *actual* processing times. *Robust scheduling*, i.e. determining a schedule whose performance (compared to the associated optimal schedule) is relatively insensitive to the potential realizations of job processing times. The paper

[KDV00] focuses on a two-machine flow shop problem and the performance measure of interest is the makespan criterion. A similar robust scheduling approach is developed for a single-machine problem by Daniels and Kouvelis [DK95]. Other robust decision-making formulations are presented by Rosenblatt and Lee [RL87], Kouvelis et al. [KKG92] and Mulvey et al. [MVZ95].

Leon et al. [LWS94] consider robustness measures and robust scheduling methods that generate job shop schedules that maintain high performance over a range of system disturbances. Wu et al. [WBS99] study the weighted tardiness job shop problem. A basic thesis of the latter paper is that "global scheduling performance is determined primarily by a subset of the scheduling decisions to be made". Wu et al. [WBS99] propose to identify a critical subset of the scheduling decisions at the beginning of the planning horizon and relegate the rest of the scheduling decisions to future points in time. Our approach considered in Chapters 2, 3 and 4 uses a similar idea.

Table 1.3: *Notations for the general shop scheduling and assembly line balancing problems*

Symbols	Description
	Notations for the general shop scheduling problem $\mathcal{G}//\Phi$
Q	Set of operations: $Q = \{1, 2, \ldots, q\}$
Q_k	Set of operations which has to be processed by machine $M_k \in M$: $Q = \bigcup_{k=1}^{m} Q_k$ and $Q_k \bigcap Q_l = \emptyset$, if $k \neq l$
Q^{J_i}	Set of operations for processing job $J_i \in J$: $Q = \bigcup_{i=1}^{n} Q^{J_i}$, $Q^{J_i} \neq \emptyset$ and $Q^{J_i} \bigcap Q^{J_j} = \emptyset$, if $i \neq j$
$w(i)+1$	First operation of job J_i, $1 \leq i \leq n$, where $w(i) = \sum_{k=0}^{i-1} n_k$ and $n_0 = 0$
$w(i)+n_i$	Last operation of job J_i, $1 \leq i \leq n$
s_i	Starting time of operation i
c_i	Completion time of operation i
p_i	Processing time of operation i
a_i	Lower bound for the processing time of operation i
b_i	Upper bound for the processing time of operation i
$c_i(s)$	Earliest completion time of operation $i \in Q$ in digraph $G_s(p)$
$s \in S$	Semiactive schedule $s = (c_1(s), c_2(s), \ldots, c_q(s))$ defined by digraph G_s
$\tilde{H}$	Set of paths in digraph $(Q, A, \emptyset)$
$\tilde{H}_s$	Set of paths in digraph $G_s \in \Lambda(G)$
H	Set of dominant paths in digraph $(Q, A, \emptyset)$
H_s	Set of dominant paths in digraph $G_s \in \Lambda(G)$
$H_k(p)$	Set of critical dominant paths in digraph $G_k \in \Lambda(G)$ (with respect to vector p) $H_k(p) \subseteq H_k$
l_k^p	Critical weight of digraph $G_k \in \Lambda(G)$ with vector p of the processing times: $\Phi_k^p = l_k^p = \max_{\mu \in H_s} l^p(\mu) = l^p(\mu^*)$, where $\mu^* \in H_s(p)$
$\tilde{H}_k^i$	Set of paths in digraph G_k ending in vertex $r \in Q^{J_i}$, where vertex r denotes the last operation of job J_i, i.e., $r = w(i) + n_i$
H_k^i	Subset of dominant paths in set $\tilde{H}_k^i$
H_{sk}	Subset of H_s of paths, which are not dominated by paths from H_k: $H_{sk} = \{\mu \in H_s :$ There is no path $\nu \in H_k$ such that $\{\mu\} \subseteq \{\nu\}\}$
$H_{sk}(T)$	$H_{sk}(T) = \{\mu \in H_s :$ There is no path $\nu \in H_k$ which dominates path μ in polytope $T\}$
	Notations for simple assembly line balancing problem SALBP
$D = (V, A)$	Digraph defining partially ordered operations $V = \{1, 2, \ldots, n\}$
$\widetilde{V}$	Set of manual operations $\widetilde{V} \in V$: $\widetilde{V} = \{1, 2, \ldots, \widetilde{n}\}$
$V \setminus \widetilde{V}$	Set of automated operations $V \setminus \widetilde{V} = \{\widetilde{n}+1, \widetilde{n}+2, \ldots, n\}$
c	Cycle time
$\widetilde{p}$	Vector of the operation times of the manual operations: $\widetilde{p} = (p_1, p_2, ..., p_{\widetilde{n}})$, $p = (\widetilde{p}, \overline{p}) = (p_1, p_2, ..., p_n)$
$\mathbf{S}$	Linearly ordered stations: $\mathbf{S} = (S_1, S_2, ..., S_m)$
$\mathbf{b}$	Line balance: Feasible assignment $V = V_1 \cup V_2 \cup \ldots \cup V_m$ into stations $\mathbf{S}$
B	Set of assignments of operations V into stations $S_1, S_2, ..., S_m$, $1 \leq m \leq n$, which do not violate precedence constraints defined by digraph $D = (V, A)$
$B(p)$	Set of line balances: $B(p) = \{\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_h\}$
$B_{opt}(p)$	Set of optimal line balances: $B_{opt}(p) \subseteq B(p)$
$O_\rho(\tilde{p})$	Stability ball of the optimal line balance $\mathbf{b} \in B_{opt}(p)$
$\rho_{\mathbf{b}}(p)$	Stability radius of the optimal line balance $\mathbf{b} \in B_{opt}(p)$

Chapter 2

Stability Analysis in Boolean Programming

Using the above results presented in Sections 1.2, 1.3 and 1.4, it is possible to find the exact value of stability radius of optimal schedule for general shop problems only with very small dimensions since formula (1.29) in Theorem 1.4 and formula (1.37) in Theorem 1.5 are based on the direct comparison of dominant paths in the set (or in the subset) of feasible digraphs $\Lambda(G)$. To exclude such an enormous enumeration, we consider now rather simple scheduling problems (essentially simpler than $\mathcal{G}//\Phi$ but still NP-hard in the general case), for which the set of dominant paths H_s of each solution s consists of a unique path. First, we consider of linear ('min-sum') criterion and then bottleneck ('min-max') criterion.

More precisely, in this section we restrict our attention to those scheduling problems which may be represented in terms of Boolean programming. We prove stability results for an ϵ-approximate solution to Boolean programming problem. For simplicity, we use here the notations which are more suitable for Boolean programming. However, we try to save the most notations (for criterion, stability radius and variable data) to be closed to that, used in the other chapters of the book. We hope that this will not imply any ambiguity for reader, since we do not use cross-references between this chapter and other chapters of the book.

Let N be the following subset of natural numbers: $N = \{1, 2, \ldots, n\}$, and $X \subseteq \{0, 1\}^n$ be set of all feasible vectors (feasible solutions). For a given objective vector p from the space R^n_+ with maximum metric and feasible vectors $x \in X$, let

$$F(p, x) = \sum_{i \in N} p_i x_i \tag{2.1}$$

be linear objective function of n variables given as components of vector $x \in X$. We assume that the set X of feasible solutions is finite and does not depend on objective vector $p = (p_1, p_2, \ldots, p_n)$. We call the components p_i of the objective vector p the objective coefficients.

The Boolean linear programming problem (being under consideration in Sections 2.1-2.4) is to find an optimal vector (optimal solution) $x^p = (x^p_1, x^p_2, \ldots, x^p_n) \in X$ with

$$F(p, x^p) = \min\{F(p, x) \ : \ x \in X\}. \tag{2.2}$$

We investigate problem (2.2) under the assumption that all or a subset of the objective coefficients $p_1, p_2, \ldots, p_n$ can change their values in comparison with the given initial ones. We first consider the stability region and the stability ball of an ϵ-approximate solution of problem (2.2) with $\epsilon \in R^1_+$.

Let $x \in X$ be an ϵ-approximate solution of the problem (2.2), i.e., the condition

$$F(p, x) \leq (1 + \epsilon) \cdot F(p, x^p) \tag{2.3}$$

holds. We investigate the situation when w given components of the objective vector p, $1 \leq w \leq n$, can be changed after solving problem (2.2) but the remaining $n - w$ components of p cannot be changed (they are stable objective coefficients). Without loss of generality, we assume that the first w components $p_1, p_2, \ldots, p_w$ of the objective vector p can be changed (unstable objective coefficients).

Hence, the values of the objective coefficients $p_{w+1}, p_{w+2}, \ldots, p_n$ are reliable, but the values of the objective coefficients $p_1, p_2, \ldots, p_w$ can be changed after solving problem (2.2).

2.1 Stability Region

We consider objective vectors $p \in R^n_+$, where the last $n-w$ components are equal to given values, say, $p_{w+1} = \bar{p}_{w+1},\ p_{w+2} = \bar{p}_{w+2}, \ldots,\ p_n = \bar{p}_n$. These $n - w$ components are stable objective coefficients. For such objective vectors we define $\tilde{p} = (p_1, p_2, \ldots, p_w) \in R^w_+$ as the vector of its first w components (unstable objective coefficients).

Definition 2.1 *The set of all objective vectors $\tilde{p} \in R^w_+$, for which $x \in X$ is an ϵ-approximate solution, is called the stability region of x and it is denoted by $K^w_\epsilon(x)$:*

$$K^w_\epsilon(x) = \{\tilde{p} \in R^w_+ \ :\ F(p, x) \leq (1 + \epsilon) \cdot F(p, x^p)\}. \tag{2.4}$$

This chapter focusses on the calculation of the largest real number $\varrho \in R^1_+$ for which vector $x \in X$ remains ϵ-approximate solution of the problem (2.2) if w unstable objective coefficients change simultaneously and independently, but each one not more than ϱ. More formally we are looking for the following stability ball $O^w_\varrho(p)$ and its largest possible radius ϱ.

Definition 2.2 *The closed ball $O^w_\varrho(p)$ in the space R^w with radius $\varrho \in R^1_+$ and vector $\tilde{p} \in R^w_+$ as center is called a stability ball of ϵ-approximate solution $x \in X$ if $O^w_\varrho(p) \cap R^w_+ \subseteq K^w_\epsilon(x)$. The radius ϱ of the largest stability ball $O^w_\varrho(p)$ of x is called the stability radius of x and is denoted by $\varrho^w_\epsilon(x, p)$. If $O^w_\varrho(p)$ is stability ball of x for any $\varrho \in R^1_+$, then $\varrho^w_\epsilon(x, p) = \infty$.*

Note that we can view optimal solution $x^p = (x^p_1, x^p_2, \ldots, x^p_n) \in X$ as a special case of ϵ-approximate solution $x \in X$ when $\epsilon = 0$. In particular, $\varrho^w_0(x, p)$ denotes stability radius of the optimal schedule $x = x^p$.

Next, we prove some properties of the set $K^w_\epsilon(x)$. For every $x \in X$, we define $\tilde{x} = (x_1, x_2, \ldots, x_w) \in \{0, 1\}^w$ as the vector of its first w components. Furthermore, we define $\tilde{X} = \{\tilde{x} \in R^w_+ \ :\ x \in X\}$.

Let U be the reflexive binary relation on the set $\tilde{X}$ with maximal cardinality such that $(\tilde{x}, \tilde{x}') \in U$ if and only if for all $i \in \{1, 2, \ldots, w\}$ equality $x_i = 1$ implies $x'_i = 1$. In other words, we have $(\tilde{x}, \tilde{x}') \in U$ if and only if the set of indices of the variables having the value 1 in x is a subset of those in x'. Due to the maximal cardinality, the relation U is uniquely determined for each $\tilde{X} \subseteq \{0, 1\}^w$.

In the following, we use vectors $\tilde{x}' = (x'_1, x'_2, \ldots, x'_w)$ and $x' = (x'_1, x'_2, \ldots, x'_w, x'_{w+1}, \ldots, x'_n)$. We define

$$K^w_\epsilon = \bigcap_{x \in X} K^w_\epsilon(x).$$

Obviously, K^w_ϵ contains all objective vectors p that are contained in the stability region of every feasible solution $x \in X$. Using the above notations, we can formulate and then prove the following assertions about the stability region $K^w_\epsilon(x)$ with $x \in X, w \leq n$ and $\epsilon \geq 0$.

Property 1: *The stability region $K^w_\epsilon(x)$ and K^w_ϵ are polyhedra.*

Property 2: *If $\epsilon_1 < \epsilon_2$, then $K^w_{\epsilon_1}(x) \subseteq K^w_{\epsilon_2}(x)$.*

Property 3: *If the inclusion $(\tilde{x}, \tilde{x}') \in U$ and the inequality*

$$\sum_{i=w+1}^{n} t_i x_i \leq \sum_{i=w+1}^{n} t_i x'_i \tag{2.5}$$

hold, then $K^w_\epsilon(x') \subseteq K^w_\epsilon(x)$.

Property 4: *We have*

$$\bigcup_{x \in X} K^w_\epsilon(x) = K^w.$$

Property 5: *At least one of the regions $K^w_\epsilon(x), x \in X$, is unbounded.*

Proof. First we show that $K^w_\epsilon(x)$ can be represented as the set of solutions of a system of linear inequalities. From (2.2) and (2.4) it follows that the set $K^w_\epsilon(x)$ contains all objective vectors p which satisfy the inequality

$$\sum_{i \in N} p_i x_i \leq (1 + \epsilon) \cdot \min \left\{ \sum_{i \in N} p_i x'_i \ : \ x' \in X \right\}. \tag{2.6}$$

Consequently, the inequality (2.3) can be written as

$$\sum_{i=1}^{w} p_i [x_i - (1 + \epsilon) x'_i] \leq b', \quad x' \in X, \tag{2.7}$$

$$p_i \geq 0, \quad i = 1, 2, \ldots, w, \tag{2.8}$$

where

$$b' = (1+\epsilon)\cdot\sum_{i=w+1}^{n}\bar{p}_i x'_i - \sum_{i=w+1}^{n}\bar{p}_i x_i.$$

Hence, the stability region $K_\epsilon^w(x)$ is given as the set of solutions of the system (2.7)-(2.8) of linear inequalities with the variables objective coefficients $p_1, p_2, \ldots, p_w$. Thus, $K_\epsilon^w(x)$ is a polyhedron (note that $K_\epsilon^w = \emptyset$ is possible). K_ϵ^w is also a polyhedron because it is the intersection of polyhedra. Thus, Property 1 holds.

The validity of Property 2 immediately follows from the definition of $K_\epsilon^w(x)$. Indeed, if the inequality $F(p,x) \le (1+\epsilon)\cdot F(p,x^p)$ holds for $\epsilon = \epsilon_1$, then it is also satisfied for each $\epsilon = \epsilon_2$ with $\epsilon_1 < \epsilon_2$.

Let the conditions of Property 3 be satisfied, objective vector $\tilde{p}^0 \in K_\epsilon^w(x')$ be arbitrary and p^0 be equal to $(p_1^0, p_2^0, \ldots, p_w^0, \bar{p}_{w+1}, \ldots, \bar{p}_n)$. We consider the value

$$F(p^0,x) = \sum_{i=1}^{w} p_i^0 x_i + \sum_{i=w+1}^{n}\bar{p}_i x_i.$$

Because of $(\tilde{x}, \tilde{x}') \in U$, the inequality

$$\sum_{i=1}^{w} p_i^0 x_i \le \sum_{i=1}^{w} p_i^0 x'_i$$

holds. Considering (2.5), we obtain $F(p^0,x) \le F(p^0,x') \le (1+\epsilon)\cdot F(p^0,x^{p^0})$, i.e., the vector x is an ϵ-approximate solution of the problem (2.2) for any objective vector $\tilde{p}^0$ from the region $K_\epsilon^w(x')$. Hence, $K_\epsilon^w(x') \subseteq K_\epsilon^w(x)$ holds which proves Property 3.

For the proof of Property 4, we note that, because the set X is finite, there exists an ϵ-approximate solution of problem (2.2) for any given $p \in R_+^w$ and any given $\epsilon \ge 0$. In particular, we can take the vector x^p, automatically satisfying inequality (2.3), as ϵ-approximate solution of problem (2.2).

Property 4 means that for any $\epsilon \ge 0$, there exists a finite covering of the space R_+^w by stability regions $K_\epsilon^w(x)$. Because we have a finite number of regions in

$$\bigcup_{x\in X}\{K_\epsilon^w(x) \;:\; x \in X\}$$

but R_+^w is unbounded, at least one stability region must be unbounded, i.e., Property 5 holds.

◇

Property 1 can be strengthened for the important special case when $w = n$.

Property 6: *The set $K_\epsilon^n(x)$ is a closed convex cone with the origin in $(p_1 = 0, p_2 = 0, \ldots, p_n = 0) = \bar{0} \in R_+^n$.*

PROOF. For any given $\lambda \ge 0$, from 2.3 it follows

$$F(\lambda p, x) = \lambda F(p,x) \le (1+\epsilon)\cdot\lambda F(p,x^p) \le (1+\epsilon)\cdot F(\lambda p, x^{\lambda p}).$$

Hence, we have $\lambda p \in K_\epsilon^n(x)$ for any $\lambda \ge 0$ which proves Property 6.

◇

As we illustrate in the following example, the set $K_\epsilon^w(x)$ (and, consequently, in general the set K_ϵ^w) can be empty. However, in the case $w = n$, due to Property 6 any set $K_\epsilon^w(x)$ is non-empty. In fact, $K_\epsilon^n \neq \emptyset$ because the n-dimensional vector $(p_1 = 0, p_2 = 0, \ldots, p_n = 0)$ belongs to K_ϵ^n for any $\epsilon \geq 0$.

Example 2.1 *A wide class of discrete optimization problems can be formulated as Boolean programming, e.g., a one-machine scheduling problem with sequence dependent setup times between the processing of the jobs which is equivalent to the* traveling salesman *problem. Here, we consider as an illustrative example the scheduling problem of minimizing the sum of processing times of the jobs on parallel (but not identical) machines.*

We have m machines that have to process l jobs, where the processing times do not depend on the jobs but on the machine and on the processing order. The matrix P gives the processing times $p_{ij} \geq 0$ if a job is the j-th job in the job order of machine i. The problem is to assign all l jobs to the machines such that the sum of processing times becomes minimal.

Let $m = 2, l = 3$ and the matrix $P = [p_{ij}]_{(m,l)}$ of the initial data be as follows:

$$P = \begin{bmatrix} 10 & 5 & 5 \\ 7 & 7 & 7 \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 \\ p_4 & p_5 & p_6 \end{bmatrix}.$$

We set $p = (p_1, p_2, \ldots, p_6)$ and $n = m \cdot l$. Let $x_{ij} = 1$ if at least j jobs are processed on machine i and $x_{ij} = 0$ otherwise. Analogously, we denote

$$\begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \end{bmatrix}.$$

The set X of feasible vectors for this problem can be described as follows:

$$\begin{aligned} X \;=\; \Big\{ x \in X^n \;:\; & \sum_{i=1}^{n} x_i = l;\; [x_i = 1, lu < i \leq l(u+1), u \geq 0] \\ & \Rightarrow [x_{lu+1} = x_{lu+2} = \ldots = x_{i-1} = 1] \Big\}. \end{aligned}$$

For the given values of m and l, we have four feasible vectors:

$$\begin{aligned} X \;=\; & \{x^{(1)} = (1,1,1,0,0,0), x^{(2)} = (1,1,0,1,0,0), \\ & x^{(3)} = (1,0,0,1,1,0), x^{(4)} = (0,0,0,1,1,1)\}. \end{aligned}$$

First, we determine the stability regions $K_\epsilon^w(x^{(i)})$ for $i \in \{1,2,3,4\}$ with $p = (p_1, p_2, 5, 7, 7, 7)$ (i.e., $w = 2$) and $\epsilon = 0$. For $x = x^{(1)}$, system (2.7)-(2.8) of linear inequalities is as follows:

$$\begin{aligned} p_1 + p_2 + 5 &\leq p_1 + p_2 + 7 \\ p_1 + p_2 + 5 &\leq p_1 + 7 + 7 \\ p_1 + p_2 + 5 &\leq 7 + 7 + 7 \\ p_1 &\geq 0 \\ p_2 &\geq 0 \end{aligned}$$

The above system is equivalent to the following system:

$$\begin{aligned} p_2 &\leq 9 \\ p_1 + p_2 &\leq 16 \\ p_1 &\geq 0 \\ p_2 &\geq 0 \end{aligned}$$

For $x = x^{(2)}$, system (2.7)-(2.8) of linear inequalities is inconsistent:

$$\begin{aligned} p_1 + p_2 + 7 &\leq p_1 + p_2 + 5 \\ p_1 + p_2 + 7 &\leq p_1 + 7 + 7 \\ p_1 + p_2 + 7 &\leq 7 + 7 + 7 \\ p_1 &\geq 0 \\ p_2 &\geq 0 \end{aligned}$$

For $x = x^{(3)}$, system (2.7)-(2.8) of linear inequalities is equivalent to the following system:

$$\begin{aligned} p_2 &\geq 9 \\ p_1 &\leq 7 \\ p_1 &\geq 0 \\ p_2 &\geq 0 \end{aligned}$$

For $x = x^{(4)}$, system (2.7)-(2.8) of linear inequalities is equivalent to the following system:

$$\begin{aligned} p_1 + p_2 &\geq 16 \\ p_1 &\leq 7 \\ p_1 &\geq 0 \\ p_2 &\geq 0 \end{aligned}$$

The above stability regions form a covering of the space R_+^w with $w = 2$ (see Figure 2.1). Note that $K_0^2(x^{(2)}) = \emptyset$ and, consequently, $K_0^2 = \emptyset$. The regions $K_0^2(x^{(3)})$ and $K_0^2(x^{(4)})$ are unbounded. Analogously, we determine the stability regions $K_\epsilon^2(x^{(i)})$ for $1 \leq i \leq 4$ with $p = (10, 5, 5, 7, 7, 7)$, $w = 2$ and $\epsilon = 0.5$. It is easy to see that the regions $K_{0.5}^2(x^{(1)}), K_{0.5}^2(x^{(2)}), K_{0.5}^2(x^{(3)})$ and $K_{0.5}^2(x^{(4)})$, respectively, are polyhedra given by

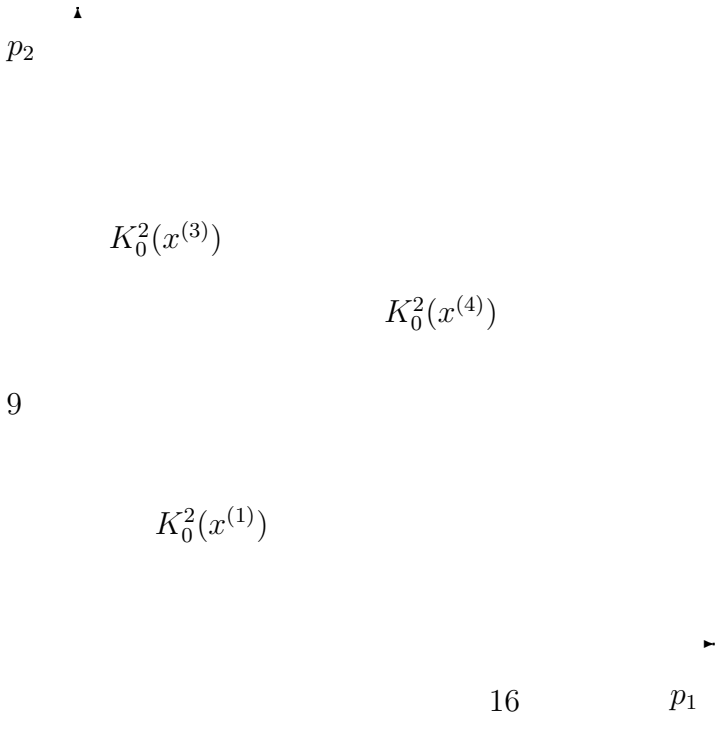

Figure 2.1: *Stability regions $K_0^2(x^{(1)}), K_0^2(x^{(3)})$ and $K_0^2(x^{(4)})$*

the solutions of the following systems of linear inequalities:

$$
\begin{array}{lrcl}
1) & p_1 - 0.5p_2 & \leq & 16 \\
 & p_1 + p_2 & \leq & 26.5 \\
 & p_1 & \geq & 0 \\
 & p_2 & \geq & 0 \\
2) & -0.5p_1 + p_2 & \leq & 14 \\
 & p_1 + p_2 & \leq & 24.5 \\
 & p_1 & \geq & 0 \\
 & p_2 & \geq & 0 \\
3) & p_1 + 3p_2 & \geq & 13 \\
 & p_1 & \leq & 17.5 \\
 & p_1 & \geq & 0 \\
 & p_2 & \geq & 0 \\
4) & p_1 + p_2 & \geq & 9 \\
 & p_1 & \geq & 0 \\
 & p_2 & \geq & 0
\end{array}
$$

These stability regions are shown in Figure 2.2. The bounds of these regions are shaded in the same way as in Figure 2.1, only the stability region of feasible solution $x^{(2)}$ additionally occurs in a new form of shading. For the regions $K_{0.5}^2(x')) = K_{0.5}^2(x^{(2)})$ and $K_{0.5}^2(x) = K_{0.5}^2(x^{(1)})$ the conditions of Property 3 are satisfied. Hence, $K_{0.5}^2(x') \subseteq K_{0.5}^2(x)$ holds. We

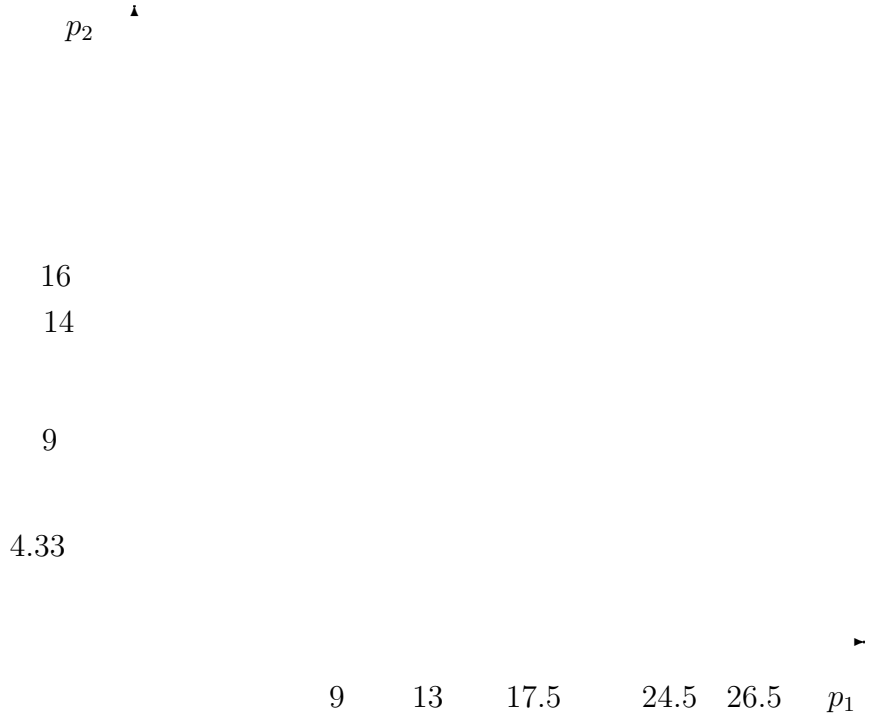

Figure 2.2: *Stability regions* $K^2_{0.5}(x^{(1)}), K^2_{0.5}(x^{(2)}), K^2_{0.5}(x^{(3)})$ *and* $K^2_{0.5}(x^{(4)})$

note that the polyhedron $K^2_{0.5}$ *is given by the set of solutions of the following system:*

$$\begin{aligned} p_1 - 0.5p_2 &\leq 14 \\ p_1 + p_2 &\leq 24.5 \\ p_1 + p_2 &\geq 9 \\ p_1 + 3p_2 &\geq 13 \\ p_1 &\geq 0 \\ p_2 &\geq 0 \end{aligned}$$

Figure 2.3 illustrates Property 2 if $p = (10, 5, 5, 7, 7, 7), w = 2, x = x^{(1)}$ *and* $\epsilon \in \{0, 0.5, 1\}$.

Obviously, the determination of the stability regions is possible only for special problems of the type (2.2) with a very small number of variables. Moreover, for $w > 3$ we need a special form even to draw the stability region in the plane.

2.2 Extreme Values and Bounds of Stability Radius

If the ball $O^w_\varrho(p)$ is a stability ball of an ϵ-approximate solution x for any positive ϱ, then we write $\varrho^w_\epsilon(x, p) = \infty$. We prove the following criterion for the existence of an infinitely large stability radius of an ϵ-approximate solution.

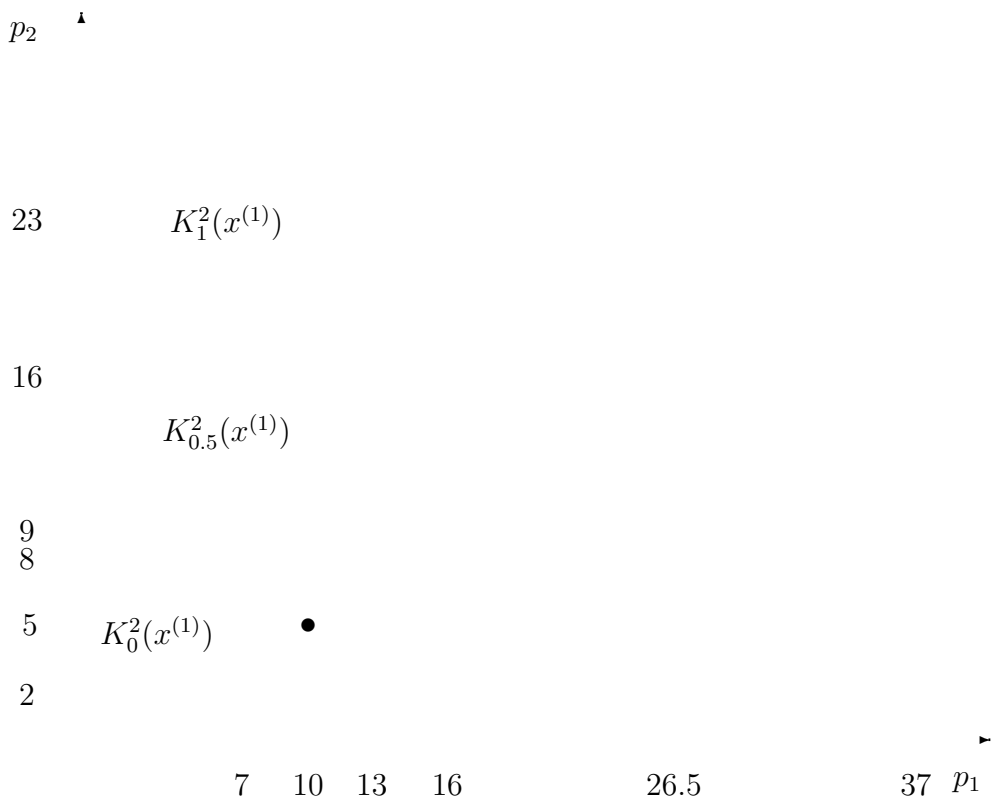

Figure 2.3: *Stability regions $K_0^2(x^{(1)}), K_{0.5}^2(x^{(1)}), K_1^2(x^{(1)})$ and the stability ball $O_3^2(p)$ of the ϵ-approximate solution $x^{(1)}$ with $p = (10, 5, 5, 7, 7, 7)$ and $\epsilon = 0.5$*

Theorem 2.1 *The stability radius $\varrho_\epsilon^w(x, p)$ equals ∞ if and only if inclusion $(\tilde{x}, \tilde{x}') \in U$ and inequality*

$$\sum_{i=w+1}^{n} p_i x_i \leq (1+\epsilon) \cdot \sum_{i=w+1}^{n} p_i x'_i \tag{2.9}$$

hold for any feasible solution $x' \in X$.

PROOF. *Necessity:* We consider two possible cases when either inclusion $(\tilde{x}, \tilde{x}') \in U$ or inequality (2.9) is violated.

(a) Let $\varrho_\epsilon^w(x, p) = \infty$ with $x \in X$. We assume that a feasible vector $x' \in X$ exists such that $(\tilde{x}, \tilde{x}') \notin U$.

Then there exists an index j with $1 \leq j \leq w$ such that $x_j = 1$ and $x'_j = 0$. We choose a real number p'_ϵ which satisfies the inequality $p'_\epsilon > p_\epsilon$ where p_ϵ is defined as follows:

$$p_\epsilon = \max\left\{ \max\{p_i \ : \ 1 \leq i \leq w\}, \ (1+\epsilon) \cdot \sum_{i=w+1}^{n} p_i - \sum_{i=w+1}^{n} p_i x_i \right\}. \tag{2.10}$$

We consider the following vector $p' = (p'_1, p'_2, \ldots, p'_n) \in R_+^n$. We set $p'_j = p'_\epsilon > p_\epsilon > 0$, where p_ϵ is given by (2.10). For each i with $1 \leq i \leq w$ and $i \neq j$, we set $p'_i = 0$. For the stable components of the objective vector p' we have, of course, $p'_i = p_i$ $(w+1 \leq i \leq n)$.

For the vector p', we evaluate $F(p', x)$ with respect to the value $(1+\epsilon)\cdot F(p', x)$:

$$\begin{aligned} F(p', x) &= \sum_{i\in N} p'_i x_i = \sum_{i=1}^{w} p'_i x_i + \sum_{i=w+1}^{n} p'_i x_i \\ &= p'_\epsilon + \sum_{i=w+1}^{n} p'_i x_i = p'_\epsilon + \sum_{i=w+1}^{n} p_i x_i \\ &> (1+\epsilon)\cdot \sum_{i=w+1}^{n} p_i = (1+\epsilon)\cdot \sum_{i=w+1}^{n} p'_i \\ &\geq (1+\epsilon)\cdot \sum_{i=1}^{w} p'_i x'_i + (1+\epsilon)\cdot \sum_{i=w+1}^{n} p'_i x'_i \\ &= (1+\epsilon)\cdot \sum_{i=1}^{n} p'_i x'_i \geq (1+\epsilon)\cdot F(p', x^{p'}). \end{aligned}$$

The first of the above inequalities holds because of $p'_\epsilon > p_\epsilon$ and the second inequality follows from

$$\sum_{i=1}^{w} p'_i x'_i = 0 \quad \text{and} \quad \sum_{i=w+1}^{n} p_i \geq \sum_{i=w+1}^{n} p_i x'_i.$$

Consequently, the vector x is not an ϵ-approximate solution to problem (2.2) for objective vector $p = p'$. Therefore, we have

$$\varrho_\epsilon^w(x, p) \leq d(p, p') \leq p'_\epsilon < \infty.$$

We obtain a contradiction to assumption $\varrho_\epsilon^w(x, p) = \infty$.

(b) Now we consider another possible case. Let $\varrho_\epsilon^w(x, p) = \infty$ with $x \in X$. And we assume that there exists a vector $x' \in X$ for which the inequality (2.9) does not hold, i.e., we have

$$\sum_{i=w+1}^{n} p_i x_i > (1+\epsilon)\cdot \sum_{i=w+1}^{n} p_i x'_i.$$

Then we consider the vector $p'' = (p''_1, p''_2, \ldots, p''_n)$ with the components

$$p''_i = \begin{cases} 0 & \text{for } i = 1, 2, \ldots, w, \\ p_i & \text{for } i = w+1, w+2, \ldots, n. \end{cases}$$

We evaluate $F(p'', x)$:

$$\begin{aligned} F(p'', x) &= \sum_{i\in N} p''_i x_i = \sum_{i=w+1}^{n} p_i x_i \\ &> (1+\epsilon)\cdot \sum_{i=w+1}^{n} p_i x'_i = (1+\epsilon)\cdot \sum_{i=1}^{n} p''_i x'_i \\ &\geq (1+\epsilon)\cdot \min\left\{ \sum_{i=1}^{n} p''_i x''_i \ : \ (x''_1, x''_2, \ldots, x''_w, x''_{w+1}, \ldots, x''_n) \in X \right\} \\ &= (1+\epsilon)\cdot F(p'', x^{p''}). \end{aligned}$$

Hence, the vector $x \in X$ is not an ϵ-approximate solution to problem (2.2) for objective vector $p = p''$. As a result we obtain a contradiction to assumption $\varrho_\epsilon^w(x,p) = \infty$:

$$\varrho_\epsilon^w(x,p) \le d(p,p'') \le \max\{p_i \ : \ 1 \le i \le w\} < \infty.$$

Thus, the necessity is proven.

Sufficiency: Let condition $(\tilde{x}, \tilde{x}') \in U$ and inequality (2.9) hold for any vector $x' \in X$.

We choose an arbitrary vector $\overline{p}^0 = (\overline{p}_1^0, \overline{p}_2^0, \ldots, \overline{p}_w^0) \in R_+^w$ and the corresponding vector $p^0 = (\overline{p}_1^0, \overline{p}_2^0, \ldots, \overline{p}_w^0, p_{w+1}, p_{w+2}, \ldots, p_n) \in R_+^n$. We consider the value

$$F(p^0, x) = \sum_{i=1}^{w} \overline{p}_i^0 x_i + \sum_{i=w+1}^{n} p_i x_i.$$

Since inclusion $(\tilde{x}, \tilde{x}') \in U$ holds for any vector $x' \in X$ we obtain

$$\sum_{i=1}^{w} \overline{p}_i^0 x_i \le \sum_{i=1}^{w} \overline{p}_i^0 x_i'.$$

Therefore, we have

$$\sum_{i=1}^{w} \overline{p}_i^0 x_i \le \min\{\sum_{i=1}^{w} \overline{p}_i^0 x_i' \ : \ x' \in X\}.$$

Since inequality (2.9) holds for any vector $x' \in X$, we get

$$\sum_{i=w+1}^{n} p_i x_i \le (1+\epsilon) \cdot \min\{\sum_{i=w+1}^{n} p_i x_i' \ : \ x' \in X\}.$$

Thus, we obtain

$$\begin{aligned} F(p^0, x) &\le \min\{\sum_{i=1}^{w} \overline{p}_i^0 x_i' \ : \ x' \in X\} + (1+\epsilon) \cdot \min\{\sum_{i=w+1}^{n} p_i x_i' \ : \ x' \in X\} \\ &\le (1+\epsilon) \cdot (\min\{\sum_{i=1}^{w} \overline{p}_i^0 x_i' \ : \ x' \in X\} + \min\{\sum_{i=w+1}^{n} p_i x_i' \ : \ x' \in X\}) \\ &= (1+\epsilon) \cdot F(p^0, x^{p^0}). \end{aligned}$$

Hence, the vector x is an ϵ-approximate solution to the problem (2.2) for any objective vector $\overline{p}^0 \in R_+^w$. In other words, stability region $K_\epsilon^w(x)$ coincides with the whole space R_+^w and $\varrho_\epsilon^w(x,p) = \infty$.

$\diamond$

For the case $w = n$ when all the objective coefficients are unstable we obtain the following corollary from Theorem 2.1.

Corollary 2.1 *The stability radius $\varrho_\epsilon^n(x,p)$ equals ∞ if and only if inclusion $(\tilde{x}, \tilde{x}') \in U$ holds for any feasible solution $x' \in X$.*

Zero Value of Stability Radius

Let $\overline{0}$ denote the zero vector in the space R_+^w. Necessary and sufficient conditions for $\varrho_\epsilon^w(x,p)=0$ are proven separately for the case $\tilde{x}\neq\overline{0}$ as well as for the case $\tilde{x}=\overline{0}$.

Theorem 2.2 *Let $\tilde{p}\in K_\epsilon^w(x)$, $\tilde{x}\neq\overline{0}$, $w\leq n$, $\epsilon>0$ and $p_i>0$ for each $i=1,2,\ldots,w$. Then we have $\varrho_\epsilon^w(x,p)=0$ if and only if condition (2.3) is satisfied as an equality.*

Proof. *Sufficiency:* Let the condition (2.3) be satisfied as an equality, i.e.,

$$F(p,x)=(1+\epsilon)\cdot F(p,x^p). \tag{2.11}$$

Since $\tilde{x}\neq\overline{0}$ and $p_i>0$ for each $i=1,2,\ldots,w$, the value $F(p,x)$ is not equal to zero. Because we have $\epsilon>0$ in (2.3), we conclude that vector x is not a solution to the problem (2.2).

We take any nonzero component of vector $\tilde{x}$ (say $\tilde{x}_i=1$, $i\in\{1,2,\ldots,w\}$), and consider the following two possible cases.

Case (i): $x_i^p=0$. For any given arbitrarily small real number $\gamma>0$, we can construct the following vector $p^\gamma=(p_1,p_2,\ldots,p_{i-1},p_i+\gamma,p_{i+1},\ldots,p_w,p_{w+1},\ldots,p_n)\in R_+^n$. This vector p^γ satisfies the following conditions:

$$\begin{aligned} F(p^\gamma,x) &= \gamma+F(p,x)=\gamma+(1+\epsilon)\cdot F(p,x^p)\\ &= \gamma+(1+\epsilon)\cdot F(p^\gamma,x^p)\geq\gamma+(1+\epsilon)\cdot F(p^\gamma,x^{p^\gamma})\\ &> (1+\epsilon)\cdot F(p^\gamma,x^{p^\gamma}). \end{aligned}$$

Thus, we have $\tilde{p}^\gamma\notin K_\epsilon^w(x)$ and, because of $d(\tilde{p},\tilde{p}^\gamma)=\gamma$, the set $O_\gamma^w(p)\cap R_+^w$ is not a subset of stability region $K_\epsilon^w(x)$. Consequently, the ball $O_\gamma^w(p)$ is not a stability ball of the ϵ-approximate solution x. Since this holds for any arbitrarily small $\gamma>0$, we have $\varrho_\epsilon^w(x,p)=0$.

Case (ii): $x_i^p=1$. We consider the objective vector $p^{(\gamma)}=(p_1,\ldots,p_{i-1},p_i-\gamma,p_{i+1},\ldots,p_w,p_{w+1},\ldots,p_n)$. If $0<\gamma\leq p_i$, then $p^{(\gamma)}\in R_+^w$ due to $p_i>0$. Since $\epsilon>0$, we obtain for the vector $p^{(\gamma)}$:

$$\begin{aligned} F(p^{(\gamma)},x) &= F(p,x)-\gamma=(1+\epsilon)\cdot F(p,x^p)-\gamma\\ &> (1+\epsilon)\cdot F(p,x^p)-(1+\epsilon)\cdot\gamma=(1+\epsilon)\cdot F(p^{(\gamma)},x^p)\\ &\geq (1+\epsilon)\cdot F(p^{(\gamma)},x^{p^{(\gamma)}}). \end{aligned}$$

Thus, we have $p^{(\gamma)}\notin K_\epsilon^w(x)$ and the ball $O_\gamma^w(p)$ is not a stability ball of the ϵ-approximate solution x. Because this holds for any arbitrarily small $\gamma>0$, we conclude that $\varrho_\epsilon^w(x,p)=0$.

Necessity: We prove necessity by contradiction. Assume that $\varrho_\epsilon^w(x,p)=0$ and condition (2.3) is satisfied as a strict inequality, i.e., we have

$$(1+\epsilon)\cdot F(p,x^p)-F(p,x)=\Delta_p>0. \tag{2.12}$$

We show that there exists a real number $\varrho > 0$ such that inequality (2.12) remains valid for any objective vector $p' \in R^n_+$ with $\tilde{p}' \in O^w_\varrho(p) \cap R^w_+$. We set $\varrho = \beta = \Delta_p/(3+\epsilon)w > 0$. Because of $d(\tilde{p}, \tilde{p}') \leq \beta$, we obtain

$$F(p', x) - F(p, x) \leq w \cdot \beta = \Delta_p/(3+\epsilon) \tag{2.13}$$

and

$$\begin{aligned}(1+\epsilon) \cdot F(p, x^p) - (1+\epsilon) \cdot F(p', x^{p'}) &\leq (1+\epsilon) \cdot F(p, x^{p'}) - (1+\epsilon) \cdot F(p', x^{p'}) \\ &\leq w \cdot \beta \cdot (1+\epsilon) = \Delta_p \cdot (1+\epsilon)/(3+\epsilon).\end{aligned}$$

We summarize the left and the right terms of inequality (2.13) with the latter one:

$$\begin{aligned}& F(p', x) - F(p, x) + (1+\epsilon) \cdot F(p, x^p) - (1+\epsilon) \cdot F(p', x^{p'}) \\ \leq\ & \Delta_p/(3+\epsilon) + \Delta_p \cdot (1+\epsilon)/(3+\epsilon) \\ =\ & \Delta_p \cdot (2+\epsilon)/(3+\epsilon).\end{aligned}$$

From this we obtain

$$\begin{aligned}(1+\epsilon) \cdot F(p', x^{p'}) - F(p', x) &\geq (1+\epsilon) \cdot F(p, x^p) - F(p, x) - \Delta_p \cdot (2+\epsilon)/(3+\epsilon) \\ &= \Delta_p - \Delta_p \cdot (2+\epsilon)/(3+\epsilon) \\ &= \Delta_p/(3+\epsilon) > 0.\end{aligned}$$

Thus, $\tilde{p}' \in O^w_\beta(p) \cap R^w_+$ implies $\tilde{p}' \in K^w_\epsilon(x)$, i.e., the ball $O^w_\beta(p)$ is a stability ball of the ϵ-approximate solution x. We obtain a contradiction to the assumption: $\varrho^w_\epsilon(x, p) \geq \beta = \Delta_p/(3+\epsilon)w > 0$.

◇

Next, we prove the analogy to Theorem 2.2 for the case $\tilde{x} = \overline{0}$.

Theorem 2.3 *Let $\tilde{p} \in K^w_\epsilon(x)$, $\tilde{x} = \overline{0}$, $w \leq n$, $\epsilon > 0$ and $\tilde{p}_i > 0$ for each $i = 1, 2, \ldots, w$. Then we have $\varrho^w_\epsilon(x, p) = 0$ if and only if condition (2.3) is satisfied as an equality and there exists an optimal solution x^p to problem (2.2) for which $\tilde{x}^p \neq 0$.*

PROOF. *Sufficiency:* Since $\tilde{x}^p \neq 0$, there exists a nonzero component $x^p_i = 1$, $i \in \{1, 2, \ldots, w\}$. As in case (ii) in the proof of Theorem 2.2, we consider the vector $p^{(\gamma)}$ and by using arguments quite similar as before we obtain $\varrho^w_\epsilon(x, p) = 0$.

Necessity: Let $\varrho^w_\epsilon(x, p) = 0$. The case when condition (2.3) is satisfied as a strict inequality can be considered similarly as in the proof of the necessity of Theorem 2.2. To complete the proof of Theorem 2.3, we still have to consider the situation when $\Delta_p = 0$ holds (where Δ_p is as defined in (2.12)) but for any solution x^p of problem (2.2) we have $\tilde{x}^p = 0$. Since $x_i = x^p_i = 0$ for any $i \in \{1, 2, \ldots, w\}$, due to the equality

$$F(p, x) = (1+\epsilon) \cdot F(p, x^p), \tag{2.14}$$

we obtain

$$F(p, x) = \sum_{i=w+1}^{n} p_i x_i = (1+\epsilon) \sum_{i=w+1}^{n} p_i x^p_i < (1+\epsilon) \cdot \sum_{i=1}^{n} p_i x'_i = (1+\epsilon) \cdot F(p, x')$$

for any vector $x' \in X$ which is not a solution to problem (2.2). Now we calculate the value

$$\Delta_0 = \min\{(1+\epsilon)\cdot F(p,x') - F(p,x) \ : \ x' \in X\backslash\{x\},\ F(p,x') > F(p,x^p)\} > 0.$$

Setting $\beta = \beta_0 = \Delta_0/(3+\epsilon)w > 0$ and repeating the steps performed for the case $\Delta_p > 0$ in the proof of the necessity of Theorem 2.2, we obtain the inequality

$$(1+\epsilon)\cdot F(p',x') - F(p',x) > 0 \tag{2.15}$$

which holds for any vector $\tilde{p}' \in O^w_\beta(p) \cap R^w_+$.

Due to $\tilde{x} = \overline{0}$ and $x^p = 0$ for any solution x^p to problem (2.2), any variation of the objective vector $\tilde{p}$ does not affect the validity of equation (2.14). Thus, for any objective vector $\tilde{p}'$ both inequality (2.15) for any vector $x' \in X$, which is not a solution to problem (2.2), and inequality (2.14) (with p replaced by p') for any solution x^p to the problem (2.2) are valid. We can conclude that $O^w_\beta(p)$ is a stability ball of the ϵ-approximate solution x which contradicts to the assumption that the stability radius of x is equal to zero.

$\diamond$

The above proofs give lower bound on stability radius of ϵ-approximate solution. In the next section, we present it along with greater lower bound on stability radius of optimal solution.

Bounds on Stability Radius

The proof of necessity of Theorem 2.1 implies the following upper bound on $\varrho^w_\epsilon(x,p)$ for arbitrary number w, $1 \le w \le n$, of unstable objective coefficients.

Corollary 2.2 *If $\varrho^w_\epsilon(x,p) < \infty$, then we have $\varrho^w_\epsilon(x,p) \le p_\epsilon$, where p_ϵ is defined in (2.10).*

For the case $w = n$, when all n objective coefficients are unstable, Corollary 2.2 is transformed as follows.

Corollary 2.3 *If $\varrho^n_\epsilon(x,p) < \infty$, then $\varrho^n_\epsilon(x,p) \le p_*$ where*

$$p_* = \max\{p_i \ : \ 1 \le i \le n\}. \tag{2.16}$$

It is easy to see that the upper bound $\varrho^n_\epsilon(x,p) \le p_*$ is tight. The upper bound

$$\varrho^w_\epsilon(x,p) \le (1+\epsilon)\cdot \sum_{i=w+1}^{n} p_i - \sum_{i=w+1}^{n} p_i x_i$$

is tight under the condition that the vector $(x_1, x_2, \ldots, x_w, 1, 1, \ldots, 1)$ is a feasible solution.

From Theorem 2.1 and Corollary 2.2 we can conclude that either stability radius $\varrho^w_\epsilon(x,p)$ is arbitrary large or we have the following bounds for stability radius $\varrho^w_\epsilon(x,p)$:

$$0 \le \varrho^w_\epsilon(x,p) \le p_\epsilon. \tag{2.17}$$

In the case $\varrho_\epsilon^w(x,p)=\infty$, for any non-negative real number ϱ and any objective vector $\overline{p}\in R_+^w$, the ball $O_\varrho^w(t)$ is a stability ball of the ϵ-approximate solution $x\in X$. In the case $\varrho_\epsilon^w(x,p)<\infty$, for any ϱ with $0\le\varrho\le p_\epsilon$, the ball $O_\varrho^w(t)$ is a stability ball of an ϵ-approximate solution $x\in X$ to the problem (2.13) for a given vector $\overline{p}\in R_+^w$.

While proving the necessity of Theorem 2.2 and the necessity of Theorem 2.3 for the case $\Delta_p>0$, we obtained the following lower bound for the value $\varrho_\epsilon^w(x,p)$.

Corollary 2.4 *Let $\tilde{p}\in K_\epsilon^w(x)$, $w\le n$, $\epsilon\ge 0$. If condition (2.3) is satisfied as a strict inequality, then*

$$\varrho_\epsilon^w(x,p)\ge\frac{\Delta_p}{(3+\epsilon)w}>0.$$

Note that any strictly positive lower bound can be used as conservative estimate of the stability radius, i.e., $O_\varrho^w(p)$ is a stability ball if $\varrho>0$ is equal to a lower bound on $\varrho_\epsilon^w(x,p)$.

One may be interested in lower bounds on the stability radius of an optimal solution x^p (i.e., when $\epsilon=0$). For optimal solution x^p we derive the following lower bound, which is often better than the one given in Corollary 2.4.

Theorem 2.4 *Let x^p be an optimal solution to the problem (2.2) and let $v(x^p)$ denote the number of 1's in vector $\tilde{x}^p$. Then*

$$\varrho_\epsilon^w(x^p,p)\ge\min\{\frac{\Delta_p}{(1+\epsilon)\cdot w-\epsilon\cdot v(x^p)},\frac{\Delta_p}{(1+\epsilon)\cdot w-v(x^p)}\}.$$

PROOF. For any pair of feasible solutions $y\in X$ and $z\in X$ to problem (2.2), let $D(y,z)$ denote the set of components which are equal to 1 in vector $\tilde{y}$ and 0 in vector $\tilde{z}$, and let $E(y,z)$ denote the set of components which are equal to 1 in both vectors $\tilde{y}$ and $\tilde{z}$. We can calculate real number

$$L=\min\{\frac{\Delta_p}{(1+\epsilon)\cdot w-\epsilon v(x^p)},\frac{\Delta_p}{(1+\epsilon)\cdot w-v(x^p)}\}.$$

We can define objective vector $p'=(p'_1,p'_2,\ldots,p'_n)\in R_+^n$ such that $d(p,p')\le L$. Then for any feasible solution $x'=(x'_1,x'_2,\ldots,x'_n)\in X$ we obtain

$$\begin{aligned}
&\sum_{i=1}^n p'_ix_i^p-(1+\epsilon)\cdot\sum_{i=1}^n p'_ix'_i\\
=\;&\sum_{i=1}^n p_ix_i^p+\sum_{i=1}^w(p'_i-p_i)x_i^p-(1+\epsilon)\cdot\sum_{i=1}^n p_ix'_i-(1+\epsilon)\cdot\sum_{i=1}^w(p'_i-p_i)x'_i\\
=\;&\sum_{i=1}^n p_ix_i^p-(1+\epsilon)\cdot\sum_{i=1}^n p_ix'_i+\sum_{i\in D(x^p,x')}(p'_i-p_i)\\
-\;&\epsilon\cdot\sum_{i\in E(x^p,x')}(p'_i-p_i)-(1+\epsilon)\cdot\sum_{i\in D(x',x^p)}(p'_i-p_i)\\
\le\;&\sum_{i=1}^n p_ix_i^p-(1+\epsilon)\cdot\sum_{i=1}^n p_ix_i^p+|D(x^p,x')|\cdot L\\
-\;&\epsilon\cdot|E(x^p,x')|\cdot(-L)-(1+\epsilon)\cdot|D(x',x^p)|\cdot(-L)\\
=\;&-\Delta_p+|D(x^p,x')|\cdot L+\epsilon\cdot|E(x^p,x')|\cdot L+(1+\epsilon)\cdot|D(x',x^p)|\cdot L.
\end{aligned}$$

If $\epsilon \leq 1$, then the latter expression is at most

$$\begin{aligned} & -\Delta_p + |D(x^p, x')| \cdot L + |E(x^p, x')| \cdot L + (1+\epsilon) \cdot |D(x', x^p)| \cdot L \\ \leq\ & -\Delta_p + v(x^p) \cdot L + (1+\epsilon) \cdot (w - v(x^p)) \cdot L \\ =\ & -\Delta_p + [(1+\epsilon) \cdot w - \epsilon \cdot v(x^p)] \cdot L = 0. \end{aligned}$$

Hence, x^p is an ϵ-approximate solution to problem (2.2) for the objective vector p'. For the case $\epsilon > 1$ the proof is similar.

◇

It is easy to see that all lower and upper bounds on finite stability radius $\varrho^w_\epsilon(x^p, p)$ can be calculated in polynomial time. In the next two sections, we show how to calculate approximate or exact value of stability radius.

2.3 Approximate and Exact Value of Stability Radius

The following claim gives characterization of a stability ball, i.e., it defines the necessary and sufficient condition when a closed ball in the space R^w is the stability ball of an ϵ-approximate solution x of the problem (2.2).

Lemma 2.1 *The closed ball $O^w_\varrho(p)$ with center $\varrho > 0$ is a stability ball of an ϵ-approximate solution x of the problem (2.2) if and only if inclusion $(\tilde{p} + \delta) \in K^w_\epsilon(x)$ holds for all 2^w vectors $\delta = (\delta_1, \delta_2, \ldots, \delta_w) \in R^w_+$ with the components $\delta_i \in \{\varrho, \max\{-p_i, -\varrho\}\}$, $i = 1, 2, \ldots, w$.*

PROOF. *Necessity:* We prove necessity by contradiction. Let $O^w_\varrho(p)$ be the stability ball of an ϵ-approximate solution x. We assume that there exists a vector $\delta' = (\delta'_1, \delta'_2, \ldots, \delta'_w)$ with the components $\delta'_i \in \{\varrho, \max\{-p_i, -\varrho\}\}$, $1 \leq i \leq w$, such that $(\tilde{p} + \delta') \notin K^w_\epsilon(x)$.

Because of $d(p, p + \delta') \leq \varrho$ and $p_i + \delta'_i \geq p_i + \max\{-p_i, -\varrho\} \geq 0$, we obtain $(\tilde{p} + \delta') \in O^w_\varrho(p) \cap R^w_+$. Thus, the inclusion $O^w_\varrho(p) \cap R^w_+ \subseteq K^w_\epsilon(x)$ does not hold. Thus, we obtain a contradiction to the assumption that ball $O^w_\varrho(p)$ is a stability ball of ϵ-approximate solution x.

Sufficiency: We assume that inclusion $(\tilde{p} + \delta) \in K^w_\epsilon(x)$ holds for all 2^w vectors $\delta = (\delta_1, \delta_2, \ldots, \delta_w) \in R^w_+$ with $\delta_i \in \{\varrho, \max\{-p_i, -\varrho\}\}$, $i = 1, 2, \ldots, w$.

Due to using a Chebyshev metric, all elements $(\tilde{p} + r), r = (r_1, r_2, \ldots, r_w)$, of the set $O^w_\varrho(p) \cap R^w_+$ can be represented as a convex polyhedron K obtained as the set of all solutions of the following system of linear inequalities:

$$\begin{aligned} p_i - \varrho \leq p_i + r_i &\leq p_i + \varrho, \quad i = 1, 2, \ldots, w, \\ p_i + r_i &\geq 0, \quad i = 1, 2, \ldots, w. \end{aligned}$$

Because the stability region $K^w_\epsilon(x)$ is convex set (see Property 1), we conclude that all faces of polyhedra K belong to set $K^w_\epsilon(x)$, since all vertices $\tilde{p} + \delta$ of the polyhedra K belong to set $K^w_\epsilon(x)$. From this it follows that all inner points of polyhedra K belong to set $K^w_\epsilon(x)$ as well. In other words, inclusion $K \subseteq K^w_\epsilon(x)$ must hold, and therefore ball $O^w_\varrho(p)$ is a stability ball of ϵ-approximate solution $x \in X$.

◇

Using Lemma 2.1 we can prove the following theorem which defines stronger characterization of a stability ball, namely, we can fix every component δ_i of the vector $\delta = (\delta_1, \delta_2, \ldots, \delta_w)$, if equality $x_i = 0$ holds. Due to this, the number of vectors δ for testing inclusion $(\tilde{p} + \delta) \in K^w_\epsilon(x)$ is less than 2^w (if there exist zero components in the vector $\tilde{x}$).

Let $V(x)$ denote the set of indices corresponding to 1's in the part $\tilde{x} \in \{0,1\}^w$ of the vector $x \in X \subseteq \{0,1\}^n$ which is a feasible solution of the problem (2.2):

$$V(x) = \{i \ : \ x_i = 1, \ i = 1, 2, \ldots, w\}. \tag{2.18}$$

Theorem 2.5 *The closed ball $O^w_\varrho(p)$ with center $\varrho > 0$ is a stability ball of an ϵ-approximate solution x of the problem (2.2) if and only if inclusion $(\tilde{p}+\delta) \in K^w_\epsilon(x)$ holds for all vectors $\delta = (\delta_1, \delta_2, \ldots, \delta_w) \in R^w_+$ which have components $\delta_i \in \{\varrho, \max\{-p_i, -\varrho\}\}$, if $i \in V(x)$, and components $\delta_j = \max\{-p_j, -\varrho\}$, if $j \in \{1, 2, \ldots, w\} \setminus V(x)$.*

Proof. It suffices to show that the condition of Lemma 2.1 implies the seemingly stronger condition of Theorem 2.5.

We assume that the condition of Theorem 2.5 is satisfied, i.e., $(\tilde{p} + \delta) \in K^w_\epsilon(x)$ for all vectors $\delta = (\delta_1, \delta_2, \ldots, \delta_w) \in R^w_+$ with $\delta_i \in \{\varrho, \max\{-p_i, -\varrho\}\}$, if $x_i = 1$, and with $\delta_i = \max\{-p_i, -\varrho\}$, if $x_i = 0$.

Let δ' be arbitrary vector with components $\delta'_i \in \{\varrho, \max\{-p_i, -\varrho\}\}$, $i = 1, 2, \ldots, w$. Then we define the vector δ'' by setting $\delta''_i = \delta'_i$, if $x_i = 1$, and $\delta''_i = \max\{-p_i, -\varrho\}$, if $x_i = 0$, $i = 1, 2, \ldots, w$.

Let x' be arbitrary feasible solution to problem (2.2), i.e., $x' \in X$. Then we can obtain the following relations:

$$\sum_{i=1}^{w}(p_i + \delta'_i)x_i + \sum_{i=w+1}^{n} p_i x_i = \sum_{i=1}^{w}(p_i + \delta''_i)x_i + \sum_{i=w+1}^{n} p_i x_i$$

$$\leq (1+\epsilon) \cdot [\sum_{i=1}^{w}(p_i + \delta''_i)x'_i + \sum_{i=w+1}^{n} p_i x'_i] \leq (1+\epsilon) \cdot [\sum_{i=1}^{n}(p_i + \delta'_i)x'_i + \sum_{i=w+1}^{n} p_i x'_i].$$

Indeed, the above equality holds because δ'' and δ' differ only in components for which the corresponding component of the vector x is equal to 0. The first inequality is true because the condition of Theorem 2.5 is satisfied. The last inequality follows from the fact that inequality $\delta'' \leq \delta'$ holds and all the components of vector x' are non-negative.

Since x' is an arbitrary feasible solution to problem (2.2), it now follows that inclusion $(\tilde{p} + \delta') \in K^w_\epsilon(x)$ holds. Thus, inclusion $(\tilde{p} + \delta'') \in K^w_\epsilon(x)$ implies inclusion $(\tilde{p} + \delta') \in K^w_\epsilon(x)$. In other words, condition of Theorem 2.5 implies condition of Lemma 2.1. Thus, Theorem 2.5 follows from Lemma 2.1.

◇

Next, we demonstrate Theorem 2.5 on the Example 2.1.

Example 2.1 (continued). *We can show that for objective vector $p = (10, 5, 5, 7, 7, 7)$ the ball $O^2_3(p)$ with the center $\tilde{p} = (10, 5)$ is not a stability ball for the exact solution $x^{(1)}$ (i.e.,*

$\epsilon = 0$*). Indeed, the point* $(10+3, 5+3) = (13, 8)$ *does not belong to the set* $K_0^2(x^{(1)})$ *since inequality (2.3) is not satisfied with* $\epsilon = 0$:

$$13 + 8 + 5 > (1 + 0) \cdot \min\{13 + 8 + 7, 13 + 7 + 7, 7 + 7 + 7\}.$$

On the other hand, it is easy to show that the ball $O_3^2(p)$ *is a stability ball of the* ϵ*-approximate solution* $x^{(1)}$ *with* $p = (10, 5, 5, 7, 7, 7)$ *and* $\epsilon = 0.5$*. Indeed, checking inequality (2.3), we obtain that all the vectors* $(13, 8), (13, 2), (7, 8)$ *and* $(7, 2)$ *belong to the set* $K_{0.5}^2(x^{(1)})$*. Thus, due to Theorem 2.5, we conclude that ball* $O_3^2(p)$ *is a stability ball of the* ϵ*-approximate solution* $x^{(1)}$ *with* $\epsilon = 0.5$*. In Figure 2.3 this ball is shaded.*

Further, let us consider feasible solution $x^{(4)}$*. None of the components of* $\tilde{x}^{(4)}$ *is equal to* 1*. Therefore, to check whether* $O_3^2(p)$ *is a stability ball of the* ϵ*-approximate solution* $x^{(4)}$ *with* $\epsilon = 0.5$*, it suffices to check only whether objective vector* $(7, 2)$ *belongs to the set* $K_{0.5}^2(x^{(4)})$*. It is left to the reader to verify that this is indeed the case. Therefore, equality* $\varrho_\epsilon^w(x^{(4)}, p) = 3$ *with* $\epsilon = 0.5$ *holds.*

Due to Theorem 2.5 we can calculate stability radius $\varrho_\epsilon^w(x, p)$ with the given accuracy $\Delta > 0$ using any algorithm for calculating optimal objective value $F(p', x^{p'})$ of the problem (2.2).

Corollary 2.5 *If there exists an* $O(g(n))$ *algorithm for calculating optimal objective value* $F(p', x^{p'})$*, then there exists* $O(2^{|V(x)|} \cdot g(n) \cdot \log(p_\epsilon/\Delta))$ *algorithm for calculating stability radius* $\varrho_\epsilon^w(x, p)$ *with the given accuracy* $\Delta > 0$*.*

Proof. Due to inequalities (2.17) (see page 80), it suffices to look for the stability radius on the closed interval $[0, p_\epsilon]$. To obtain the desired complexity of the algorithm for calculating approximate value of stability radius $\varrho_\epsilon^w(x, p)$ we can use the binary search.

At the first iteration of the algorithm, we test condition of Theorem 2.5 for the ball $O_\varrho^w(p) \subseteq R^w$ with radius $\varrho = p_\epsilon/2$.

In the case if condition of Theorem 2.5 holds for $\varrho = p_\epsilon/2$, then ball $O_{p_\epsilon/2}^w(p)$ is a stability ball of an ϵ-approximate solution x and so the inequalities

$$p_\epsilon/2 \le \varrho_\epsilon^w(x, p) \le p_\epsilon$$

hold, otherwise ball $O_{p_\epsilon/2}^w(p)$ is not a stability ball of an ϵ-approximate solution x and so the inequalities

$$0 \le \varrho_\epsilon^w(x, p) < p_\epsilon/2$$

hold.

At the second iteration of the algorithm, we test condition of Theorem 2.5 either for value $\varrho = p_\epsilon/4$ in the former case, or for value $\varrho = (3p_\epsilon)/4$ in the latter case.

Continuing in a similar way after $\log(p_\epsilon/\Delta)$ iterations, we obtain approximate value of stability radius $\varrho_\epsilon^w(x, p)$ with accuracy Δ.

At each iteration of such an algorithm, for testing condition of Theorem 2.5 it suffers to calculate optimal objective value $F(p', x^{p'})$ and test inequality (2.3) with $p = p' = (\tilde{p} + \delta, p_{w+1}, p_{w+2}, \ldots, p_n)$ for all $2^{|V(x)|}$ vectors $\delta \in R_+^w$ defined in Theorem 2.5. Due to condition of the corollary, it takes $O(2^{|V(x)|} \cdot g(n))$ time. Thus, asymptotic complexity of the algorithm for calculating stability radius $\varrho_\epsilon^w(x, p)$ with the given accuracy $\Delta > 0$ is $O(2^{|V(x)|} \cdot g(n) \cdot \log(p_\epsilon/\Delta))$.

$\diamond$

Next, we show how to calculate exact value of stability radius $\varrho_\epsilon^w(x,p)$ using Theorem 2.5. Set $V(x)$ used in Theorem 2.6 is defined by (2.18).

Theorem 2.6 *Let $x \in X$ be an ϵ-approximate solution to the problem (2.2). If calculation of the optimal objective value $F(p', x^{p'})$ can be done in $O(g(n))$ time for any objective vector $p' \in R_+^n$, then stability radius $\varrho_\epsilon^w(x,p)$ can be calculated in $O(2^{|V(x)|} \cdot w \cdot g(n))$ time.*

PROOF. The proof is constructive. For every subset I of the set $V(x)$, $I \subseteq V(x)$, and for every real number $\varrho \geq 0$, we define variation vector $\delta^{[I,\varrho]} = (\delta_1^{[I,\varrho]}, \delta_2^{[I,\varrho]}, \ldots, \delta_w^{[I,\varrho]}) \in R_+^w$ of the given vector $\tilde{p} \in R_+^w$ by setting $\delta_i^{[I,\varrho]} = \varrho$, if $i \in I$, and $\delta_j^{[I,\varrho]} = \max\{-p_j, -\varrho\}$, if $j \in \{1,2,\ldots,w\} \setminus I$. It is easy to see that absolute value $|\delta_i^{[I,\varrho]}|$ of each component $\delta_i^{[I,\varrho]}$ (i.e., variation of the objective coefficient p_i) is as large as possible provided that inequality

$$d(\tilde{p} + \delta^{[I,\varrho]}, p) = \max\{|\delta_i^{[I,\varrho]}| \ : \ i = 1,2,\ldots,w\} \leq \varrho$$

and inclusion $(\tilde{p} + \delta^{[I,\varrho]}) \in R_+^w$ hold.

Let ϱ_I denote the largest value of ϱ for which vector $\tilde{p} + \delta^{[I,\varrho]}$ belongs to stability region $K_\epsilon^w(x)$. We will show how to calculate exact values ϱ_I for all $2^{|V(x)|}$ subsets $I \subseteq V(x)$. Then due to Definition 2.2 and Theorem 2.5, the stability radius $\varrho_\epsilon^w(x,p)$ will be equal to the minimum of these $2^{|V(x)|}$ values ϱ_I:

$$\varrho_\epsilon^w(x,p) = \min\{\varrho_I \ : \ I \subseteq V(x)\}. \tag{2.19}$$

First, we consider a fixed subset $I \subseteq V(x)$. For any real number $\varrho \geq 0$, we define objective vector

$$p^{[I,\varrho]} = (\tilde{p}^{[I,\varrho]}, p_{w+1}^{[I,\varrho]}, p_{w+2}^{[I,\varrho]}, \ldots, p_n^{[I,\varrho]}) \in R_+^n$$

by setting $\tilde{p}^{[I,\varrho]} = \tilde{p} + \delta^{[I,\varrho]}$ and $p_i^{[I,\varrho]} = p_i$ for each index $i = w+1, w+2, \ldots, n$.

Let us consider the objective value $F(p^{[I,\varrho]}, x)$ of the ϵ-approximate solution $x \in X$ to the problem (2.2) as a function of single variable ϱ with range $0 \leq \varrho < \infty$. If variable ϱ increases from 0, then initially $F(p^{[I,\varrho]}, x)$ is a linear function of ϱ. In Cartesian coordinate system, the *slope* of the graph of this linear piece of function $F(p^{[I,\varrho]}, x)$ (i.e., the tangent of the angle between horizontal axis of argument ϱ and the corresponding straight segment presented this linear piece of function $F(p^{[I,\varrho]}, x)$) is equal to

$$|I| - |V(x) \setminus I|.$$

When variable ϱ becomes equal to $\min\{p_i \ : \ i \in V(x) \setminus I\}$, then the slope of the linear piece of function $F(p^{[I,\varrho]}, x)$ changes from value $(|I| - |V(x) \setminus I|)$ into value

$$|I| - |\{i \ : \ i \in V(x) \setminus I, \varrho < p_i\}|,$$

and so on. Thus, it is clear that $F(p^{[I,\varrho]}, x)$ is a continuous and piecewise linear function of argument ϱ, with breakpoints occurring exactly when equalities $\varrho = p_i$ hold for indices $i \in V(x) \setminus I$. (Breakpoint is such a point of the argument where derivative of the function does not exist. Note that derivative of piecewise linear function coincides with the function where derivative exists.)

Arguing just in the same way, we conclude that for every feasible solution $x' \in X$, the corresponding function $F(p^{[I,\varrho]}, x')$ is continuous and piecewise linear function of single argument ϱ, with breakpoints occurring at some subset of the following set of points:

$$\{p_i \ : \ i \in \{1, 2, \ldots, w\} \setminus I\}.$$

Therefore, we will refer to the points $p_i, i \in \{1, 2, \ldots, w\} \setminus I$, as the *critical points*. For convenience, we also define 0 and ∞ to be critical points p_{i_0} and $p_{i_{w+1}}$, respectively. Thus, the positive horizontal axis $[0, \infty)$ of argument ϱ may be partitioned into $w+1$ intervals: $[p_{i_0}, p_{i_1}), [p_{i_1}, p_{i_2}), \ldots, [p_{i_w}, p_{i_{w+1}})$ where the following relations hold:

$$0 = p_{i_0} \leq p_{i_1} \leq p_{i_2} \leq \ldots \leq p_{i_w} < p_{i_{w+1}} = \infty. \tag{2.20}$$

Note that in (2.20), equality $p_{i_c} = p_{i_{c+1}}$ is possible for some $c \in \{0, 1, \ldots, w-1\}$, and as a result, some of the intervals $[p_{i_c}, p_{i_{c+1}})$ may be empty. Obviously, the above partition of the positive horizontal axis $[0, \infty)$ includes at most $w+1$ non-empty intervals.

It is clear that between two consecutive different critical points in (2.20), the functions $F(p^{[I,\varrho]}, x')$ defined for different feasible solutions $x' = (x'_1, x'_2, \ldots, x'_n) \in X$ are all linear. Moreover, the slopes of each of these linear functions are integers in the range from $|I| - w$ to $|I|$, with the extreme values occurring when $x'_i = 0$ if and only if $i \notin I$, and when $x'_i = 1$ if and only if $i \in I$, respectively.

Let us define the following function $H_I(\varrho)$ of single variable ϱ with range $0 \leq \varrho < \infty$:

$$H_I(\varrho) = (1+\epsilon) \cdot \min_{x' \in X} \{F(p^{[I,\varrho]}, x')\}. \tag{2.21}$$

Since between two consecutive different critical points $\varrho = p_{i_c}$ and $\varrho = p_{i_{c+1}}, p_{i_c} \neq p_{i_{c+1}}$, functions $(1+\epsilon) \cdot F(p^{[I,\varrho]}, x')$ defined for all feasible solutions $x' \in X$ are all linear, it follows that on the segment $[p_{i_c}, p_{i_{c+1}}]$ function $H_I(\varrho)$ is the minimum of a finite number of linear functions (see definition (2.21)). It is known (see [ES76]) that this implies that $H_I(\varrho)$ is continuous, piecewise linear and concave function on such a segment $[p_{i_c}, p_{i_{c+1}}]$.

Since between two consecutive different critical points each of the functions $F(p^{[I,\varrho]}, x')$, $x' \in X$, has an integer slope in the range from $|I| - w$ to $|I|$, it follows from definition (2.21) that each slope of the function $H_I(\varrho)$ belongs always to the following set:

$$\{(1+\epsilon)(|I| - w), (1+\epsilon)(|I| - w + 1), \ldots, (1+\epsilon)(|I|)\}.$$

Because of concavity of function $H_I(\varrho)$ on each non-empty segment $[p_{i_c}, p_{i_{c+1}}]$, the sequence of the consecutive slopes of the function $H_I(\varrho)$ on the segment $[p_{i_c}, p_{i_{c+1}}]$ is non-increasing, which implies that function $H_I(\varrho)$ has at most w breakpoints on each segment $[p_{i_c}, p_{i_{c+1}}]$ between two different consecutive critical points p_{i_c} and $p_{i_{c+1}}$. Moreover, the continuity of the functions $(1+\epsilon) \cdot F(p^{[I,\varrho]}, x')$, $x \in X$, implies that function $H_I(\varrho)$ is also continuous on the whole range of argument: $0 \leq \varrho < \infty$. Since there are at most $w+1$ non-empty intervals $[p_{i_c}, p_{i_{c+1}})$, $c \in \{0, 1, \ldots, w\}$, in the whole range of argument ϱ and at most w breakpoints on each segment $[p_{i_c}, p_{i_{c+1}}]$, we obtain the following claim.

The number of linear pieces of function $H_I(\varrho)$ grows as $O(w^2)$.

Since function $F(p^{[I,\varrho]}, x)$ is linear on the segment $[p_{i_c}, p_{i_{c+1}}]$, $p_{i_c} \neq p_{i_{c+1}}$, and function $H_I(\varrho)$ is concave on $[p_{i_c}, p_{i_{c+1}}]$, we obtain the following claim.

If inequality

$$F(p^{[I,\varrho]}, x) \leq H_I(\varrho) \tag{2.22}$$

holds for both different values $\varrho = p_{i_c}$ *and* $\varrho = p_{i_{c+1}}$, , $c \in \{0, 1, \ldots, w\}$, *then inequality (2.22) holds for all values* $\varrho \in [p_{i_c}, p_{i_{c+1}}]$.

Combining inequality (2.22) with definition (2.21) of function $H_I(\varrho)$ we obtain

$$F(p^{[I,\varrho]}, x) \leq H_I(\varrho) = (1+\epsilon) \cdot \min_{x' \in X}\{F(p^{[I,\varrho]}, x')\}.$$

Therefore, the latter claim may be rewritten as follows.

If x *is an* ϵ*-approximate solution of the problem (2.2) with objective vector* $p^{[I,\varrho]}$ *for both different values* $\varrho = p_{i_c}$ *and* $\varrho = p_{i_{c+1}}$, , $c \in \{0, 1, \ldots, w\}$, *then* x *is an* ϵ*-approximate solution of the problem (2.2) for all values* $\varrho \in [p_{i_c}, p_{i_{c+1}}]$.

Thus, by evaluating functions $F(p^{[I,\varrho]}, x)$ and $H_I(\varrho)$ only at the different critical points $\varrho = p_{i_c}$, $c = 0, 1, \ldots, w$, we find the largest finite critical point, say $\varrho^* = p_{i_{c^*}}$, $c^* \in \{0, 1, \ldots, w\}$, for which x is an ϵ-approximate solution of the problem (2.2). As a result, we obtain the following inequalities for ϱ_I:

$$p_{i_{c^*}} \leq \varrho_I < p_{i_{c^*+1}}.$$

Note that the above calculation of lower bound ϱ^* of the value ϱ_I takes $O(w \cdot g(n))$ time. Indeed, there are at most $w+1$ finite critical points and due to definition (2.21) calculating $H_I(\varrho)$ at each of them boils down to calculating objective value $F(p', x^{p'})$ for a fixed objective vector $p' \in R^n_+$. Due to condition of the theorem, the latter calculation requires $O(g(n))$ time.

Thus, the desired value ϱ_I belongs to the interval $[\varrho^*, p_{i_{c^*+1}}]$. If $\varrho^* = p_{i_{c^*+1}}$, then we obtain exact value ϱ_I. Next, we show how to find exact value ϱ_I, if $\varrho^* < p_{i_{c^*+1}}$.

We have already observed that on the segment $[\varrho^*, p_{i_{c^*+1}}]$ between critical point ϱ^* and different consequent critical point $p_{i_{c^*+1}}$, function $F(p^{[I,\varrho]}, x)$ is linear, while $H_I(\varrho)$ is concave and piecewise linear function with at most w breakpoints. There exists a method (see [ES76]) which determines all the linear pieces of function $H_I(\varrho)$ on such an interval in $O(B \cdot g(n))$ time, where B is the number of breakpoints of function $H_I(\varrho)$ on this interval. Hence, it takes $O(w \cdot g(n))$ time to determine the linear pieces of function $H_I(\varrho)$ on the segment $[\varrho^*, p_{i_{c^*+1}}]$. Once this has been done, the value ϱ_I is found by calculating the largest intersection point, say ϱ^{**}, of function $F(p^{[I,\varrho]}, x)$ and function $H_I(\varrho)$ on the segment $[\varrho^*, p_{i_{c^*+1}}]$.

It should be noted that there always exists an intersection point of functions $F(p^{[I,\varrho]}, x)$ and $H_I(\varrho)$ on $[\varrho^*, p_{i_{c^*+1}}]$, if critical point $p_{i_{c^*+1}}$ is finite: $p_{i_{c^*+1}} < \infty$. Hence, if function $F(p^{[I,\varrho]}, x)$ and function $H_I(\varrho)$ do not intersect, then we certainly conclude that $c^* = w$ and so $p_{i_{c^*+1}} = \infty$ and also $\varrho_I = \infty$.

It is easy to convince that if there exist more than one intersection point of function $F(p^{[I,\varrho]}, x)$ and function $H_I(\varrho)$ on segment $[\varrho^*, p_{i_{c^*+1}}]$, then the set of all intersection points fills out a linear piece of function $H_I(\varrho)$ and function $F(p^{[I,\varrho]}, x)$. In such a case, ϱ_I is equal to the right boundary of argument ϱ for this linear piece of function $H_I(\varrho)$ (since

this boundary is inner point of segment $[\varrho^*, p_{i_{c^*+1}}]$, then only function $H_I(\varrho)$ changes the slope in this point).

In any case, the largest intersection point ϱ^{**} of function $F(p^{[I,\varrho]}, x)$ and function $H_I(\varrho)$ on the segment $[\varrho^*, p_{i_{c^*+1}}]$ either can be found in time bounded by the number w of linear pieces of the function $H_I(\varrho)$ on $[\varrho^*, p_{i_{c^*+1}}]$ or it can be proven that such an intersection does not exist (and it takes the same time $O(w)$).

Summarizing we conclude that for any subset $I \subseteq V(x)$, it is possible to calculate exact value ϱ_I in $O(w \cdot g(n))$ time. Thus, using equality (2.19) we can calculate exact value of stability radius $\varrho_\epsilon^w(x,p)$ after considering all $2^{|V(x)|}$ subsets I of set $V(x)$. The whole complexity of this calculation of stability radius is $O(2^{|V(x)|} \cdot w \cdot g(n))$. Conclusion of the theorem now follows.

$\diamond$

From the proof of Theorem 2.6 we obtain the following useful corollaries.

Corollary 2.6 *The number of linear pieces of function $H_I(\varrho)$ defined by (2.21) grows as $O(w^2)$.*

Corollary 2.7 *If x is an ϵ-approximate solution of the problem (2.2) with objective vector $p^{[I,\varrho]}$ for both different values $\varrho = p_{i_c}$ and $\varrho = p_{i_{c+1}}$, , $c \in \{0, 1, \ldots, w\}$, then x is an ϵ-approximate solution of the problem (2.2) for all values $\varrho \in [p_{i_c}, p_{i_{c+1}}]$.*

From the constructive proof of Theorem 2.6 we obtain the following algorithm for calculating exact value of stability radius $\varrho_\epsilon^w(x,p)$.

Algorithm 2.1

Input: ϵ-approximate solution x of the problem (2.2).
Output: Stability radius $\varrho_\epsilon^w(x,p)$.

Step 1: Define set $V(x) = \{i \ : \ x_i = 1, i = 1, 2, \ldots, w\}$;
assume that $\varrho_\epsilon^w(x,p) = \infty$.
Step 2: **FOR** all subset $I \subseteq V(x)$ **DO**
BEGIN
Step 3: Calculate the largest finite critical point $\varrho^* = p_{i_{c^*}}$
for which inequality $F(p^{[I,\varrho^*]}, x) \leq H_I(\varrho')$ holds.
Point $p_{i_{c^*+1}}$ is the next critical point.
Step 4: Determine the linear pieces of function $H_I(\varrho)$ on segment $[\varrho^*, p_{i_{c^*+1}}]$.
Step 5: **IF** functions $F(p^{[I,\varrho]}, x)$ and $H_I(\varrho)$ intersect on the segment $[\varrho^*, p_{i_{c^*+1}}]$
THEN value ϱ_I is defined as the largest intersection point ϱ^{**}
of functions $F(p^{[I,\varrho]}, x)$ and $H_I(\varrho)$
ELSE assume $\varrho_I = \infty$.
Step 6: **IF** $\varrho_I < \varrho_\epsilon^w(x,p)$ **THEN** set $\varrho_\epsilon^w(x,p) = \varrho_I$
END STOP.

We illustrate Algorithm 2.1 by calculating exact value of stability radius $\varrho_\epsilon^w(x,p)$ for Example 2.1.

Example 2.1 (continued). *We calculate stability radius $\varrho_\epsilon^w(x,p)$ with $w=2,\ \epsilon=0.5,\ x=x^{(1)}$ and $p=(10,5,5,7,7,7)$.*

Since $\tilde{x}^{(1)}=(1,1)$, we obtain set $V(x^{(1)})=\{1,2\}$, which means that subsets I of the set $V(x^{(1)})$ are as follows: $\emptyset$, $\{1\}$, $\{2\}$, $\{1,2\}$. For each of these subsets $I\subseteq V(x^{(1)})$ we have to calculate value ϱ_I. Next, we consider the calculation of the value $\varrho_{\{2\}}$ in detail.

The vector $p^{[\{2\},\varrho]}$ is defined as follows: $(\max\{0,10-\varrho\},5+\varrho,5,7,7,7)$. Therefore, we obtain four functions

$$F(p^{[\{2\},\varrho]},x^{(i)})=\sum_{j=1}^{6}p_j^{[\{2\},\varrho]}x_j^{(i)}$$

for $i\in\{1,2,3,4\}$, which graphs are shown in Figure 2.4. Note that three of these functions have a breakpoint at $\varrho=10$, which corresponds to critical point p_1. There are no other breakpoints, since 1 is the only element of the set $V(x)\setminus I$. Hence, there are three critical points: 0, 10 and ∞. From Figure 2.4 we see that function

$$H_{\{2\}}(\varrho)=1.5\cdot\min\{F(p^{[\{2\},\varrho]},x^{(i)})\ :\ i\in\{1,2,3,4\}\}$$

is defined as follows:

$$H_{\{2\}}(\varrho)=\begin{cases}1.5\cdot(20), & 0\le\varrho\le 4,\\ 1.5\cdot(24-\varrho), & 4\le\varrho\le 10,\\ 1.5\cdot(14), & 10\le\varrho.\end{cases}$$

Note that piecewise linear function $H_{\{2\}}(\varrho)$ is indeed concave between consecutive critical points. In Figure 2.5, the function $H_{\{2\}}(\varrho)$ is drawn together with function $F(p^{[\{2\},\varrho]},x^{(1)})$. The largest value of ϱ for which inequality $F(p^{[\{2\},\varrho]},x^{(1)})\le H_{\{2\}}(\varrho)$ holds is equal to 11. Hence, this is the value of $\varrho_{\{2\}}$: $\varrho_{\{2\}}=11$.

Algorithm 2.1, however, does not construct the two functions $F(p^{[\{2\},\varrho]},x^{(1)})$ and $H_{\{2\}}(\varrho)$ completely, before it determines value $\varrho_{\{2\}}$. What the algorithm does, is the following. For the smallest strictly positive critical point $\varrho=10$, the values $F(p^{[\{2\},10]},x^{(1)})$ and $H_{\{2\}}(10)$ are calculated. The first of the above calculations is trivial, since $x^{(1)}$ is a fixed feasible solution. For the second calculation, we need to solve an instance of the problem (2.2) (see the proof of Theorem 2.6).

Since $F(p^{[\{2\},10]},x^{(1)})\le H_{\{2\}}(10)$, Algorithm 2.1 proceeds with the next critical point. This point is ∞, and therefore the functions $F(p^{[\{2\},\varrho]},x^{(1)})$ and $H_{\{2\}}(\varrho)$ are considered for arbitrarily large values of ϱ. Because inequality $F(p^{[\{2\},\varrho]},x^{(1)})>H_{\{2\}}(\varrho)$ holds for such values, we conclude that $\varrho_{\{2\}}\in[10,\infty)$.

To find the value of $\varrho_{\{2\}}$, we first construct function $H_{\{2\}}(\varrho)$ on the interval, which is easy in this case since the function is linear on the interval. (How the construction may be done in general will be described below.) Subsequently $\varrho_{\{2\}}$ is calculated as the largest intersection point of functions $H_{\{2\}}(\varrho)$ and $F(p^{[\{2\},\varrho]},x^{(1)})$.

In Figure 2.6, we have illustrated the calculation of $\varrho_{\{1,2\}}$. It is left to the reader to verify that the functions have indeed the shown shape. In this case, the only critical points are 0 and ∞. Note again that function $H_{\{1,2\}}(\varrho)$ is indeed concave between these two values of argument ϱ.

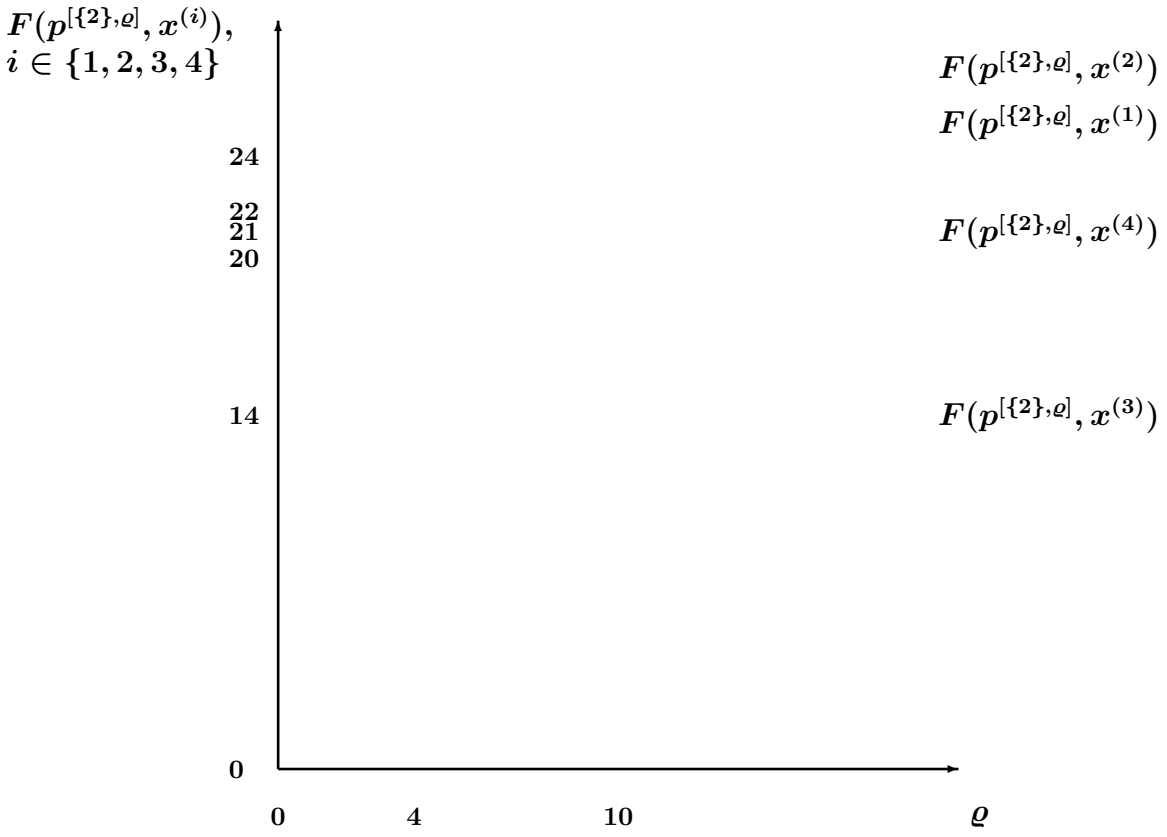

Figure 2.4: *Functions* $F(p^{[\{2\},\varrho]}, x^{(i)}), i \in \{1,2,3,4\}$, *for subset* $I = \{2\}$

The method to construct this function $H_{\{1,2\}}(\varrho)$ *can briefly be described as follows. First, we determine optimal solutions corresponding to the endpoints of the interval. These solutions define linear functions, which are possible pieces of the function to be constructed. Next, the intersection point of the two linear functions is calculated. In our case, this is* $\varrho = 0.5$*, and therefore* $H_{\{1,2\}}(0.5)$ *is calculated. Since this does not yield a new solution (function is linear), it can be concluded that all the linear pieces of* $H_{\{1,2\}}(\varrho)$ *have been found. In general, we continue this procedure by calculating the intersection points of the newly found linear function with the concave lower envelope of the linear functions found earlier. The procedure stops if no new intersection points are generated. We refer to [ES76] for a proof of the correctness of this procedure, which requires solving* $O(b)$ *instances of the problem (2.2), where* b *is the number of breakpoints of the function to be constructed.*

From Figure 2.6, we see that $\varrho_{\{1,2\}} = 5.75$*. It is now left to the reader to verify that* $\varrho_{\emptyset} = \infty$ *and* $\varrho_{\{1\}} = 16.5$*. Hence, we can calculate the exact value of stability radius* $\varrho^2_{0.5}(x^{(1)}, p)$ *using formula (2.19):*

$$\varrho^2_{0.5}(x^{(1)}, p) = \min\{11,\ 5.75,\ 16.5,\ \infty\} = 5.75.$$

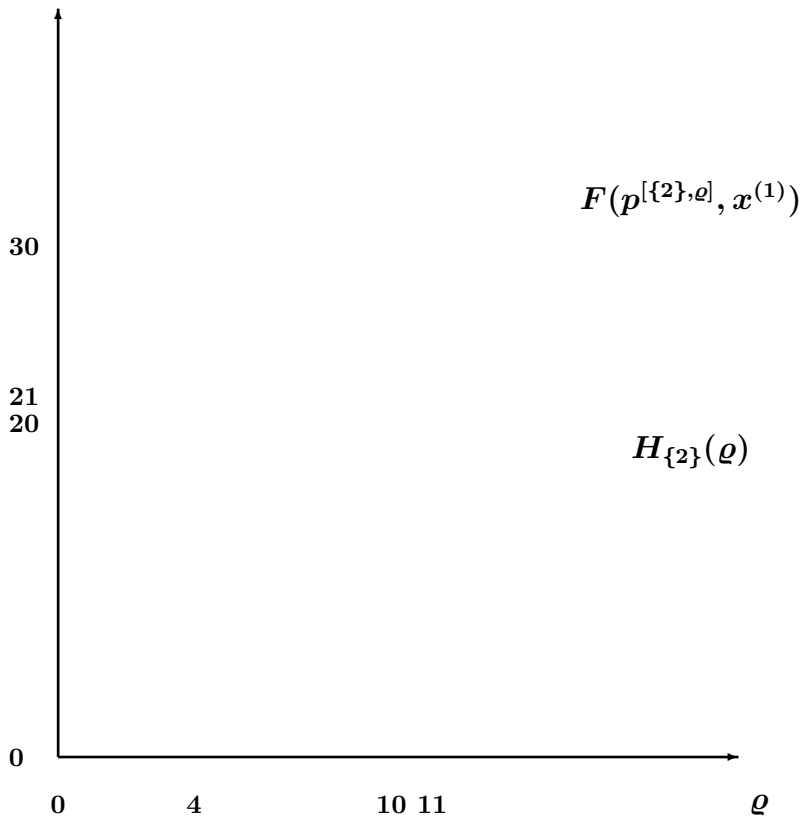

Figure 2.5: *Calculation of* $\varrho_{\{2\}} = 11$

2.4 Polynomial Algorithms for Calculating Stability Radius

It is worse noting that cardinality $|V(x)|$ of the set $V(x)$ may be significantly less than number w of unstable objective coefficients. In particular, if the number of unstable objective coefficients increases when n grows, the problem structure may prevent $|V(x)|$ from growing as fast as w. For instance, in the traveling salesman problem, if the unstable objective coefficients represent the distances with respect to one specific city, then w is of the same order as the number of cities, but equality $|V(x)| = 2$ holds for any feasible solution $x \in X$. We can use Algorithm 2.1 based on the proof of Theorem 2.6 for calculating stability radius $\varrho_\epsilon^w(x, p)$ for such a traveling salesman problem. If the unstable objective coefficients represent the distances with respect to one specific city, then the complexity of Algorithm 2.1 turns out to be $O(w \cdot g(n))$.

Note however that even for the fixed number w of unstable coefficients, asymptotic bound $O(2^{|V(x)|} \cdot w \cdot g(n))$ of Algorithm 2.1 based on Theorem 2.6 remains exponential, if $g(n)$ defined in Theorem 2.6 is an exponential function. This is no surprise, since as it will be shown in Chapter 4, even for one unstable objective coefficient ($w = 1$) computing the stability radius exactly for any $\epsilon \geq 0$ is NP-hard problem, if the associated Boolean programming problem (2.2) is NP-hard itself.

In this section, we consider special cases of the Boolean programming problem (2.2) when stability radius $\varrho_\epsilon^w(x, p)$ can be calculated in polynomial time.

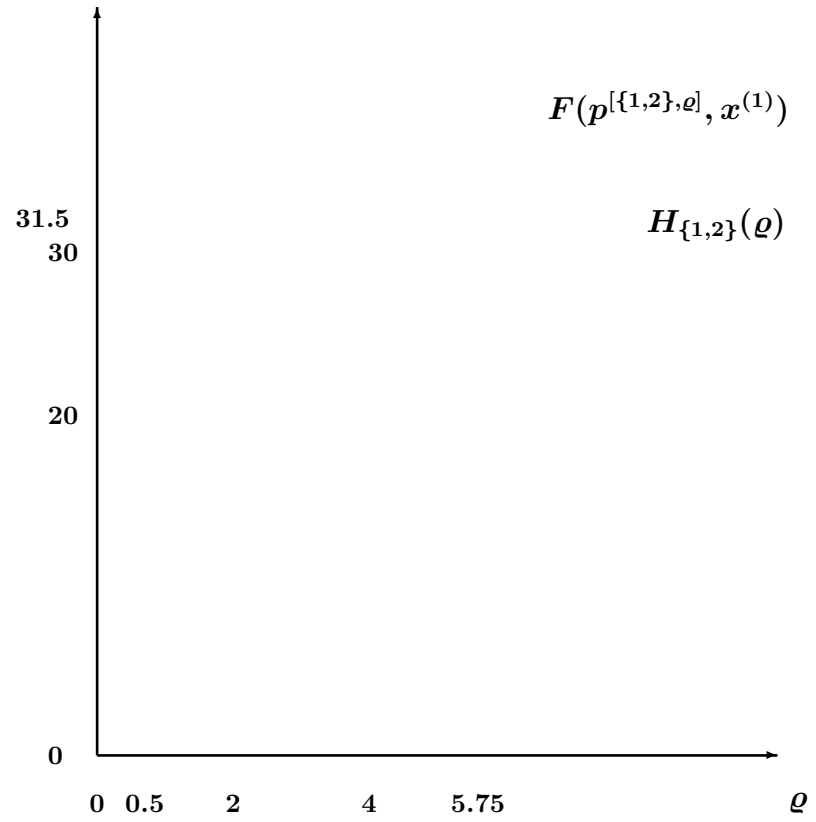

Figure 2.6: *Calculation of* $\varrho_{\{1,2\}} = 5.75$

From Corollary 2.5 we can obtain efficient algorithm for calculating approximate value of stability radius, if problem (2.2) is polynomially solvable and $|V(x)|$ grows rather slowly with n.

Corollary 2.8 *Suppose that problem (2.2) is solvable in polynomial time $g(n)$. If $|V(x)|$ grows as $O(\log\ n)$, then the stability radius $\varrho_\epsilon^w(x,p)$ can be calculated in polynomial time $O(n \cdot g(n) \cdot \log(p_\epsilon/\Delta))$ with the given accuracy $\Delta > 0$.*

Proof. The function $g(n)$ in Corollary 2.5 is now polynomial in n, and asymptotic bound of cardinality $|V(x)|$ of the set $V(x)$ is now $O(\log\ n)$. Hence, the running time $O(2^{|V(x)|} \cdot g(n) \cdot \log(p_\epsilon/\Delta))$ of the algorithm described in Corollary 2.5 becomes $O(n \cdot g(n) \cdot \log(p_\epsilon/\Delta))$, which is clearly polynomial in n.

$\diamond$

Similarly from Theorem 2.6 one can obtain the following claim.

Corollary 2.9 *Suppose that problem (2.2) is solvable in polynomial time $g(n)$. If $|V(x)|$ grows as $O(\log\ n)$, then the stability radius $\varrho_\epsilon^w(x,p)$ can be calculated in polynomial time $O(n \cdot w \cdot g(n))$.*

Further, we describe an algorithm for calculating stability radius $\varrho_\epsilon^w(x,p)$ which requires the solution of a polynomial number of instances of the Boolean linear programming problem (2.2). In particular, this means that, if problem (2.2) is polynomially solvable,

then the stability radius of an ϵ-approximate solution of the problem (2.2) can be calculated in polynomial time as well. This algorithm is similar to Algorithm 2.1 based on the proof of Theorem 2.6. The main difference between two algorithms is due to enumeration schemes used in them. While Algorithm 2.1 is based on enumeration of all subsets I of set $V(x)$, the algorithm described in the rest of this section is based on enumeration of some feasible solutions $x \in X$.

First, we consider the case of the problem (2.2) when objective coefficients $p_i, i \in N$, are unrestricted in sign, i.e., now it is assumed that objective vector p may belong to the whole space R^n (recall that in Sections 2.1 - 2.3 it was assumed that $p \in R^n_+$). Of course, the negative processing times have no sense in a usual scheduling problem, but there are some other important interpretations of a Boolean linear programming problem (2.2) when both negative and positive objective coefficients have to be allowed.

Unrestricted Objective Coefficients

It should be noted that for objective vector $p \in R^n$ with objective coefficients, which are unrestricted in sign, the concept of ϵ-approximate solution only makes sense if

$$F(p, x') \geq 0 \tag{2.23}$$

for all feasible solutions $x' = (x'_1, x'_2, \ldots, x'_n) \in X$. Therefore, in this subsection, we assume that inequality (2.23) holds for all $x' \in X$.

Let $x \in X$ be an ϵ-approximate solution to the problem instance (2.2) with objective coefficients which are unrestricted in sign, i.e., $p \in R^n$. (Definition 2.1 and Definition 2.2 may be simply rewritten using space R^w instead of space R^w_+ in order to define stability radius $\varrho^w_\epsilon(x, p)$ for the case $p \in R^n$.) Thus, we want to determine the largest real number $\varrho = \varrho^w_\epsilon(x, p)$ such that if inequalities $|\delta_i| \leq \varrho$, $i = 1, 2, \ldots, w$, hold for all vectors $\delta = (\delta_1, \delta_2, \ldots, \delta_w) \in R^w$, then inequalities

$$\sum_{i=1}^{w}(p_i + \delta_i)x_i + \sum_{i=w+1}^{n} p_i x_i \leq (1+\epsilon) \cdot [\sum_{i=1}^{w}(p_i + \delta_i)x'_i + \sum_{i=w+1}^{n} p_i x'_i] \tag{2.24}$$

hold for all feasible solutions $x' \in X$. Similarly as Theorem 2.1 we can prove the following claim.

Theorem 2.7 *Let $x \in X$ be an ϵ-approximate solution to the problem instance (2.2) with objective coefficients which are unrestricted in sign: $p \in R^n$. Stability radius $\varrho^w_\epsilon(x, p)$ is infinite if and only if all feasible solutions $x' \in X$ have components $x'_i = x_i$ for all indices $i = 1, 2, \ldots, w$. If $\varrho^w_\epsilon(x, p)$ is finite, then inequality $\varrho^w_\epsilon(x, p) \leq \varrho^\epsilon$ holds where*

$$p^\epsilon = \max\{p_i \ : \ 1 \leq i \leq w\} + (1+\epsilon) \cdot [\sum_{i=w+1}^{n} \max\{p_i, 0\} - \sum_{i=w+1}^{n} \min\{p_i, 0\}]. \tag{2.25}$$

Due to Theorem 2.7 it is suffices to look for the stability radius on the interval $[0, p^\epsilon]$. We present inequality (2.24) as

$$\sum_{i=1}^{w} p_i x_i + \sum_{i=1}^{w} \delta_i x_i + \sum_{i=w+1}^{n} p_i x_i \leq (1+\epsilon) \cdot \sum_{i=1}^{w} p_i x'_i + (1+\epsilon) \cdot \sum_{i=1}^{w} \delta_i x'_i + (1+\epsilon) \cdot \sum_{i=w+1}^{n} p_i x'_i$$

and move the first and third terms of the left-hand side to the right-hand side, and the second term of the right-hand side to the left-hand side. As a result, we obtain inequality (2.26) which is equivalent to inequality (2.24):

$$\sum_{i=1}^{w} \delta_i[x_i - (1+\epsilon)x'_i] \leq \sum_{i=1}^{n} p_i[(1+\epsilon)x'_i - x_i]. \tag{2.26}$$

Let us consider inequality (2.26) for a fixed, but unknown feasible solution $x \in X$, and a fixed real number $\varrho \geq 0$. Then the right-hand of (2.26) is a constant and so inequality (2.26) holds if and only if it holds for those values of δ_i with $|\delta_i| \leq \varrho$, $i = 1, 2, \ldots, w$, which maximize the left-hand side of inequality (2.26).

We consider an index $i \in \{1, 2, \ldots, w\}$ and suppose $x_i = 0$, then the term $\delta_i[x_i - (1+\epsilon)x'_i]$ in the left-hand side of inequality (2.26) is equal to $-\delta_i(1+\epsilon)x'_i$, which is maximized at $\delta_i = -\varrho$, irrespective of the value of x'_i. In this case, we define $d_i = 1 + \epsilon$. Hence, the maximum value is equal to $\varrho d_i x_i$.

We suppose that $x_i = 1$ for some index $i \in \{1, 2, \ldots, w\}$. If $x'_i = 0$, then the term $\delta_i[x_i - (1+\epsilon)x'_i]$ is equal to δ_i, which is maximized at $\delta_i = \varrho$. If $x'_i = 1$, then the term $\delta_i[x_i - (1+\epsilon)x'_i]$ is equal to $\delta_i(-\epsilon)$, which is maximized at $\delta_i = -\varrho$. Therefore, we define $d_i = -1 + \epsilon$ in this case. Thus, the maximum value is always equal to $\varrho + \varrho d_i x_i$.

For convenience, we also define $d_i = 0$ for all indices $i = w+1, w+2, \ldots, n$. Then we have derived that inequality (2.26) holds if and only if

$$\varrho \sum_{i=1}^{w} x_i + \sum_{i=1}^{n} \varrho d_i x'_i \leq \sum_{i=1}^{n} p_i[(1+\epsilon)x'_i - x_i].$$

This immediately implies the following result.

Theorem 2.8 *Stability radius $\varrho_\epsilon^w(x, p)$ of an ϵ-approximate solution to the problem (2.2) with unrestricted coefficients of objective vector $p \in R^n$ is equal to the largest real number ϱ for which the following inequality holds:*

$$\min_{x' \in X} \{\sum_{i=1}^{n} [p_i(1+\epsilon) - \varrho d_i]x'_i\} \geq \sum_{i=1}^{n} p_i x_i + \varrho \sum_{i=1}^{w} x_i. \tag{2.27}$$

The right-hand side of inequality (2.27) is a linear function of argument ϱ. The left-hand side is the objective value function of a *parametric* version of problem (2.2) with $p \in R^n$ where the objective coefficients are linear functions of argument ϱ. We denote this value function by $v(\varrho)$:

$$v(\varrho) = \min_{x' \in X} \{\sum_{i=1}^{n} [p_i(1+\epsilon) - \varrho d_i]x'_i\}. \tag{2.28}$$

It is well known (see, e.g., [ES76] or [Gus83]) that $v(\varrho)$ is continuous, piecewise linear and concave function of ϱ. Next, we prove the following claim.

Lemma 2.2 *The number of linear pieces of function $v(\varrho)$ defined by (2.28) grows as $O(w^2)$.*

PROOF. Since the slope of function $v(\varrho)$ is always equal to $\sum_{i=1}^{n} d_i x'_i$ for some feasible solution $x' \in X$, it follows from the definition of values d_i, $i = 1, 2, \ldots, n$, that this slope takes on values in the set $\{(m - 2k) - m\epsilon \ : \ k \in \{0, 1, \ldots, m\}, m \in \{0, 1, \ldots, w\}\}$. Moreover, because of concavity, the sequence of slopes is non-increasing. The asymptotic bound on the number of linear pieces of function $v(\varrho)$ now follows.

$\diamond$

There exists a quite simple procedure (see [ES76]) to determine the function $v(\varrho)$ on the closed interval $[0, \varrho^{\epsilon}]$. It starts with computing values $v(0)$ and $v(\varrho^{\epsilon})$. The optimal solutions associated with these values each defines a linear function on the closed interval $[0, \varrho^{\epsilon}]$. If these linear functions are identical, then $v(\varrho)$ is simply this linear function. Otherwise, we obtain two different linear functions which intersect at a unique argument value $\varrho' \in [0, \varrho^{\epsilon}]$. If point $(\varrho', v(\varrho'))$ coincides with the intersection point, then $v(\varrho)$ is the concave lower envelope of these two linear functions. Otherwise, the optimal solutions associated with ϱ' defines a third linear function which intersects each of the other linear functions constructed on $[0, \varrho^{\epsilon}]$ at argument values $\varrho'' \in [0, \varrho^{\epsilon}]$ and $\varrho''' \in [0, \varrho^{\epsilon}]$. These two new intersection points define new values of ϱ for which function $v(\varrho)$ is to be computed, and so on.

For the correctness of this procedure we refer to [ES76], where the running time is proven to be $O(b\, r(n^0, \epsilon))$ time, where b is the number of linear pieces of function $v(\varrho)$, n^0 is the size of the problem instance with objective vector $p \in R^n$, and $r(n^0, \epsilon)$ is the asymptotic complexity of solving Boolean linear programming problem (2.2) corresponding to any value $\varrho \in [0, \varrho^{\epsilon}]$.

Once function $v(\varrho)$ has been computed, it is trivial to find the largest value ϱ^0 of argument ϱ for which function $v(\varrho)$ is greater than or equal to the linear function $\sum_{i=1}^{n} p_i x_i + \varrho \sum_{i=1}^{w} x_i$:

$$v(\varrho) \geq \sum_{i=1}^{n} p_i x_i + \varrho \sum_{i=1}^{w} x_i.$$

From definition (2.28) it follows that the largest value ϱ^0 of variable ϱ is obtained for which inequality (2.27) holds. Thus, due to Theorem 2.8 ϱ^0 is equal to stability radius $\varrho_{\epsilon}^{w}(x, p)$ of an ϵ-approximate solution $x \in X$. Due to Lemma 2.2 the number b of linear pieces of function $v(\varrho)$ is restricted by $O(w^2)$. Therefore, stability radius $\varrho_{\epsilon}^{w}(x, p)$ is calculated in $O(w^2 r(n^0, \epsilon))$ time. Thus, the following claim is proven.

Theorem 2.9 *Let $x \in X$ be an ϵ-approximate solution to the problem (2.2) with objective coefficients which are unrestricted in sign. If $r(n^0, \epsilon)$ is complexity of solving problem (2.2) corresponding to any value $\varrho \in [0, \varrho^{\epsilon}]$, then stability radius $\varrho_{\epsilon}^{w}(x, p)$ can be calculated in $O(w^2 r(n^0, \epsilon))$ time.*

Theorem 2.9 has the following important implication.

Theorem 2.10 *If Boolean linear programming problem (2.2) is polynomially solvable for any objective vector $p \in R^n$, then stability radius $\varrho_{\epsilon}^{w}(x, p)$ of an ϵ-approximate solution $x \in X$ can be calculated in polynomial time.*

PROOF. Note that on the closed interval $[0, \varrho^\epsilon]$, the objective coefficients $p_i(1+\epsilon)-\varrho d_i,\ i = 1, 2, \ldots, n$, are polynomial in p and ϵ. Hence, if problem (2.2) is polynomially solvable for any objective vector $p \in R^n$ with objective coefficients unrestricted in sign, then $r(n^0, \epsilon)$ is polynomial in the size of the problem instances, which are in turn polynomial in n^0 and ϵ.

$\diamond$

Non-Negative Objective Coefficients

The above approach can easily be extended to problem (2.2) in which the objective coefficients are required to be non-negative: $p \in R^n_+$. We assume, without loss of generality, that $p_1 \leq p_2 \leq \ldots \leq p_w$.

Let us consider only values of ϱ in the segment $[p_j, p_{j+1}]$ for some index $j \in \{1, 2, \ldots, w-1\}$. Then, for every index $i \leq j$, value δ_i may be chosen smaller than $-p_i$. Therefore, for these indices i, $i \leq j$, if equality $x_i = 0$ holds, the maximum value of the term $\delta_i[x_i - (1+\epsilon)x'_i]$ is now equal to $(1+\epsilon)x'_i$.

Let equality $x_i = 1$ hold. Then the maximum value of the term $\delta_i[x_i-(1+\epsilon)x'_i]$ is equal to ϱ if $x'_i = 0$. If equality $x'_i = 1$ holds, the maximum value of the term $\delta_i[x_i - (1+\epsilon)x'_i]$ is equal to $p_i\epsilon$.

Hence, in this case, the maximum value of the term $\delta_i[x_i-(1+\epsilon)x'_i]$ is always equal to $\varrho - \varrho x_i + p_i\epsilon x_i$. This means that if the stability radius belongs to the segment $[p_j, p_{j+1}]$, then it is the largest value of ϱ in this segment for which the following inequality holds:

$$\min_{x' \in X}\{\sum_{1 \leq i \leq j,\ x_i=1} (p_i + \varrho)x'_i + \sum_{i=j+1}^{n} [p_i(1+\epsilon) - \varrho d_i]x'_i\} \geq \sum_{i=1}^{n} p_i x_i + \varrho \sum_{i=1}^{w} x_i. \qquad (2.29)$$

As in the previous subsection, the value function on the left-hand side of inequality (2.29) is piecewise linear and concave function on segment $[p_j, p_{j+1}]$. Note that for any fixed value of ϱ, indeed a problem instance with non-negative objective coefficients results.

To find the stability radius $\varrho^w_\epsilon(x, p)$, it is not necessary to construct the value function of every segment $[p_j, p_{j+1}]$, $j = 1, 2, \ldots, w-1$. Note that if inequality (2.29) holds in the endpoints p_j and p_{j+1} of the segment $[p_j, p_{j+1}]$, then because of concavity of left-hand side function of (2.29), inequality (2.29) has to hold on the complete segment $[p_j, p_{j+1}]$. Therefore, the segment which contains the stability radius $\varrho^w_\epsilon(x, p)$ can be easily found by checking only endpoints $p_1, p_2, \ldots, p_w, p^\epsilon$ of the segments $[p_j, p_{j+1}]$, $j \in \{1, 2, \ldots, w-1\}$, and segment $[p_w, p^\epsilon]$.

This means that the correct segment including stability radius can be found in $O(w\ r(n^0, \epsilon))$ time. Once that segment is known, the stability radius $\varrho^w_\epsilon(x, p)$ is calculated in $O(w^2 r(n^0, \epsilon))$ time. Thus, the complexity of this approach is the same as before for the case of unrestricted objective coefficients and the following result is proven.

Theorem 2.11 *If Boolean linear programming problem (2.2) is polynomially solvable for any objective vector $p \in R^n_+$, then stability radius $\varrho^w_\epsilon(x, p)$ of an ϵ-approximate solution $x \in X$ can be calculated in polynomial time.*

In the next section, we prove similar result for the case of bottleneck criterion.

2.5 Min-Max Criterion

At the end of this chapter, we consider maximum objective function $F_{max}(p,x)$ instead of linear objective function (2.1) considered in Sections 2.1 - 2.4:

$$F_{max}(p,x)=\max_{i\in N}\{p_ix_i\}. \tag{2.30}$$

The corresponding min-max (bottleneck) Boolean programming problem is to find an optimal vector (optimal solution) $x^p=(x_1^p,x_2^p,\ldots,x_n^p)\in X$ with

$$F_{max}(p,x^p)=\min\{F_{max}(p,x)\ :\ x\in X\}. \tag{2.31}$$

In this section, we derive an explicit expression for the stability radius of optimal solution x^p and show that it may be calculated by solving at most polynomially many instances of problem (2.31) of about the same size as the given instance. Both Definition 2.1 and Definition 2.2 may be similarly rewritten using maximum function (2.30) instead of linear function (2.1). Let $\widehat{\varrho}_0^w(x,p)$ denote stability radius of optimal solution $x=x^p\in X$ of the problem (2.31) with min-max criterion. All the other notations used for problem (2.31) are assumed to be the same as that used for the problem (2.2) with min-sum criterion.

Let Boolean bottleneck programming problem (2.31) with objective vector $p\in R_+^n$ be given and its optimal solution x^p be available. First, we consider the case of calculating stability radius $\widehat{\varrho}_0^w(x^p,p)$ when all the objective coefficients of the problem (2.31) are unstable, i.e., $w=n$. In this case, equality $x^p=\tilde{x}^p$ holds.

Recall that $V(x^p)$ denotes the set of indices of 1's in the vector x^p (see definition (2.18) at page 83). For each index $j\in V(x^p)$ we can solve the modified instance of the given problem (2.31) in which component x_j of each feasible solution $x\in X$ is required to be 0. Let $x^{[j]}=(x_1^{[j]},x_2^{[j]},\ldots,x_n^{[j]})$ denote an optimal solution of such modified instance of the problem (2.31) with set $\{x\ :\ x\in X,x_j=0\}$ of the feasible solutions:

$$F_{max}(p,x^{[j]})=\min\{F_{max}(p,x)\ :\ x\in X,x_j=0\}.$$

Let real number $b^{[j]}$ be the corresponding optimal objective value:

$$b^{[j]}=\max_{i\in N}\{p_ix_i^{[j]}\}.$$

We denote

$$\varrho_j=\frac{b^{[j]}-p_j}{2}. \tag{2.32}$$

If equality $x_j=1$ holds for all feasible solutions $x\in X$ of the given instance of the problem (2.31) (i.e., if set of the feasible solutions of the modified instance of the problem (2.31) is empty), then we define $b^{[j]}=\varrho_j=\infty$.

Theorem 2.12 *If all the n objective coefficients of the problem (2.31) are unstable ($w=n$), then*

$$\widehat{\varrho}_0^w(x^p,p)=\min\{\varrho_j\ :\ j\in V(x^p)\}.$$

PROOF. First, we prove inequality

$$\widehat{\varrho}_0^w(x^p, p) \geq \min\{\varrho_j \ : \ j \in V(x^p)\}. \tag{2.33}$$

To this end, we consider a vector $\delta = (\delta_1, \delta_2, \ldots, \delta_n) \in R^n$ with $|\delta_i| \leq \min\{\varrho_j \ : \ j \in V(x^p)\}$ for all indices $i = 1, 2, \ldots, n$.

It is clear that any feasible solution $x \in X$ with $x_j = 1$ for all indices $j \in V(x^p)$ has always a value of the objective function greater than or equal to $F_{max}(p + \delta, x^p)$.

Therefore, it is suffices to consider only feasible solutions $x \in X$ which have equality $x_j = 0$ for some indices $j \in V(x^p)$. For such a feasible solution, let $k \in V(x^p)$ be such an index that $x_k = 0$ and $p_k \geq p_j$ for all indices $j \in V(x^p)$ with $x_j = 0$. Note that $x_j = 1$ for all indices $j \in V(x^p)$ with $p_k < p_j$. For any feasible solution $x \in X$, this implies inequality:

$$F_{max}(p + \delta, x) \geq \max\{p_i + \delta_i \ : \ i \in V(x^p), \ p_i > p_k\}. \tag{2.34}$$

Furthermore, it follows from inequality $F_{max}(p, x) \geq b^{[k]}$ and equality (2.32) that

$$F_{max}(p + \delta, x) \geq b^{[k]} - \min\{\varrho_j \ : \ j \in V(x^p)\} \geq b^{[k]} - \varrho_k = (b^{[k]} + p_k)/2. \tag{2.35}$$

For any index $i \in V(x^p)$ with $p_k \geq p_i$ we obtain

$$p_i + \delta_i \leq p_k + \min\{\varrho_j \ : \ j \in V(x^p)\} \leq p_k + \varrho_k \leq (b^{[k]} + p_k)/2. \tag{2.36}$$

Using the lower bounds on $F_{max}(p + \delta, x)$ defined in (2.34) and (2.35), as well as inequalities (2.36), we obtain

$$F_{max}(p + \delta, x) \geq \max\{(b^{[k]} + p_k)/2, \ \max\{p_i + \delta_i \ : \ i \in V(x^p), \ p_i > p_k\}\}$$

$$\geq \max\{p_i + \delta_i \ : \ i \in V(x^p)\} = F_{max}(p + \delta, x^p).$$

This establishes the desired inequality (2.33).

Next, we prove opposite inequality

$$\widehat{\varrho}_0^w(x^p, p) \leq \min\{\varrho_j \ : \ j \in V(x^p)\}. \tag{2.37}$$

Obviously, if $\min\{\varrho_j \ : \ j \in V(x^p)\} = \infty$, we are already done with the proof.

So, we have to prove inequality (2.37) for the case when $\min\{\varrho_j \ : \ j \in V(x^p)\} < \infty$. To this end, we will show that for any real number ϱ strictly greater than $\min\{\varrho_j \ : \ j \in V(x^p)\}$, there exists a vector $\delta \in R^n$ such that $|\delta_i| \leq \varrho$ for each index $i \in N$, while $F_{max}(p + \delta, x) < F_{max}(p + \delta, x^p)$ for some feasible solution $x \in X$.

To be more specific, we assume that $\varrho > \varrho_k$ for some index k with $x_k^p = 1$ and let $x = x^{[k]}$. We consider the vector δ where $\delta_k = \varrho$. For $i \neq k$, we assume that $\delta_i = -\varrho_k$, if $x_i^{[k]} = 1$ and $p_i \geq (b^{[k]} + p_k)/2$, and we assume $\delta_i = 0$ otherwise.

On the one hand, we obtain

$$F_{max}(p + \delta, x^p) \geq p_k + \delta_k > p_k + \varrho_k = (b^{[k]} + p_k)/2.$$

The last equality follows from (2.32). On the other hand, we have

$$F_{max}(p + \delta, x^{[k]}) = \max\{p_i + \delta_i \ : \ i \in N, x_i^{[k]} = 1\}.$$

Due to the above definition of δ, if $x_i^{[k]} = 1$ and $p_i \geq (b^{[k]} + p_k)/2$, then

$$p_i + \delta_i = p_i - \varrho_k \leq F_{max}(p, x^{[k]}) - (b^{[k]} - p_k)/2 = (b^{[k]} + p_k)/2.$$

As a result we obtain $F_{max}(p+\delta, x^{[k]}) < F_{max}(p+\delta, x^p)$. Hence, inequality (2.37) holds, which completes the proof.

$\diamond$

To compute value $b^{[j]}$ for index $j \in V(x^p)$, we just need to solve the instance of problem (2.31) with objective vector $\tilde{p} = (\tilde{p}_1, \tilde{p}_2, \ldots, \tilde{p}_n) \in R^n_+$, where $\tilde{p}_j$ is equal to a value g which is strictly greater than the largest value of $p_1, p_2, \ldots, p_n$ and $\tilde{p}_i = p_i$ for all other indices $i \in N$, $i \neq j$. If the optimal objective value of this problem instance turns out to be g, then it follows that $x_j = 1$ for each feasible solution $x \in X$ and so $b^{[j]} = \infty$. Otherwise, the optimal objective value is exactly $b^{[j]}$. It therefore follows from Theorem 2.12 that when all the objective coefficients are unstable, stability radius $\widehat{\varrho}_0^w(x^p, p)$ can be calculated by solving $|V(x^p)|$ instances of problem (2.31). It is clear that $|V(x^p)| = \sum_{i=1}^n x_i^p$. Thus, the following corollary is proven.

Corollary 2.10 *If $w = n$, then stability radius $\widehat{\varrho}_0^w(x^p, p)$ can be calculated by solving $\sum_{i=1}^n x_i^p$ instances of problem (2.31).*

Stable and Unstable Objective Coefficients

Let w be any arbitrary positive integer less than or equal to n. Without loss of generality, we assume that $p_{w+1} \leq p_{w+2} \leq \ldots \leq p_n$. Next, we show how to compute stability radius $\widehat{\varrho}_0^w(x^p, p)$ as the minimum of certain value $\widehat{\varrho}_j$, $j \in V(x^p)$, which are defined below.

Instead of the value $b^{[j]}$ used in Theorem 2.12 for calculating stability radius $\widehat{\varrho}_0^w(x^p, p)$ with $w = n$, we use here the value $b^{[j,m]}$ defined for each index $j \in V(x^p)$, $j \leq w$, and index m as follows.

Let $b^{[j,m]}$ denote the optimal objective value of the modified instance of the given problem (2.31) with the same objective vector p and the additional restrictions on the set of feasible solutions: $x_j = 0$ and $x_i = 0$ for all indices i with $m < i \leq n$. If this modified instance does not have a feasible solution, then we define $b^{[j,m]} = \infty$.

Let $x^{[j,m]}$ be an optimal solution of the modified problem instance, and let $d^{[j,m]}$ denote the value of the largest p_i, $i > w$, for which $x^{[j,m]} = 1$. We define $d^{[j,m]}$ to be $-\infty$ if $x_i^{[j,m]} = 0$ for all indices $i > w$. For index $j \in V(x^p)$, $j \leq w$, we define

$$\widehat{\varrho}_j = \min_{m \geq w} \max\{\frac{b^{[j,m]} - p_j}{2},\ d^{[j,m]} - p_j\}. \tag{2.38}$$

First, we prove the following auxiliary claim.

Lemma 2.3 *Let $x \in X$ be an optimal solution of the modified instance of the given problem (2.31) with $x_j = 0$ for some index $j \in V(x^p)$, $j \leq w$. Let l be the largest index such that $l > w$ with $x_l = 1$ (we assume $p_l = -\infty$, if no such index exists). Then*

$$p_i + \widehat{\varrho}_j \leq \max\{(F_{max}(p, x) + p_j)/2, p_l\}.$$

PROOF. We suppose that $x_i = 0$ for all indices $i > w$. Then $F_{max}(p, x) \geq b^{[j,w]}$. Since $d^{[j,w]} = -\infty$, it follows that

$$p_j + \widehat{\varrho}_j \leq \max\{(b^{[j,w]} + p_j)/2, d^{[j,w]}\} \leq (F_{max}(p, x) + p_j)/2.$$

If $x_i = 1$ for some index $i > w$, then $F_{max}(p, x) \geq b^{[j,l]}$ and $p_l \geq d^{[j,l]}$. Therefore, $p_j + \widehat{\varrho}_j \leq \max\{(b^{[j,l]} + p_j)/2, d^{[j,l]}\} \leq \max\{(F_{max}(p, x) + p_j)/2, p_l\}$.

◇

Now, we can formulate and prove the following theorem.

Theorem 2.13 *If there are both stable and unstable objective coefficients in the problem (2.31), then*

$$\widehat{\varrho}_0^w(x^p, p) = \min\{\widehat{\varrho}_j \ : \ j \in V(x^p)\}.$$

PROOF. First, we show that the stability radius is at least equal to $\min\{\widehat{\varrho}_j \ : \ j \in V(x^p)\}$. To this end, we consider a vector $\delta \in R^n$ such that $\delta_i = 0$ for all indices $i > w$, and $|\delta_i| \leq \min\{\widehat{\varrho}_j \ : \ j \in V(x^p)\}$ for all indices $i \leq w$. Thus, we obtain

$$F_{max}(p + \delta, x^p) = \max \begin{cases} \max\{p_i + \delta_i \ : \ i \in V(x^p), i \leq w, p_i \leq p_k\}, \\ \max\{p_i + \delta_i \ : \ i \in V(x^p), i \leq w, p_i > p_k\}, \\ \max\{p_i \ : \ i \in V(x^p), i > w, x_i^p = 1\}, \\ \max\{p_i \ : \ i \in V(x^p), i > w, x_i^p = 0\}. \end{cases} \tag{2.39}$$

For any feasible solution $x \in X$, we will show that the four expressions on the right-hand side of (2.39) are all lower bounds on $F_{max}(p + \delta, x)$.

Let expression $\{p_i \ : \ i \in V(x^p), i \leq w, x_i^p = 0\}$ take the maximum value for index $i = k$. Then the first expression on the right-hand side of (2.39), $\max\{p_i + \delta_i \ : \ i \in V(x^p), i \leq w, p_i \leq p_k\}$, is less than or equal to $p_k + \min\{\widehat{\varrho}_j \ : \ j \in V(x^p)\} \leq p_k + \widehat{\varrho}_k$. Because of Lemma 2.3 this is at most $\max\{(F_{max}(p, x) + p_k)/2, p_l\}$, where p_l is defined as in Lemma 2.3. Clearly, $p_l \leq F_{max}(p + \delta, x)$. Furthermore, if $(F_{max}(p, x) + p_k)/2 > p_l$, then $F_{max}(p + \delta, x) \geq F_{max}(p, x) - \widehat{\varrho}_k = (F_{max}(p, x) + p_k)/2$.

To prove that the second expression on the right-hand side of (2.39) is a lower bound on $F_{max}(p + \delta, x)$, it suffices to observe that if $i \in V(x^p)$ and $p_i > p_k$, then $x_i^p = 1$.

The third expression on the right-hand side of (2.39) is obvious lower bound on $F_{max}(p + \delta, x)$.

Let expression $\{p_i \ : \ i \in V(x^p), i > w, x_i^p = 0\}$ take the maximum value for index $i = r$. Then the forth expression on the right-hand side of (2.39) is equal to p_r. To show that this is a lower bound on $F_{max}(p + \delta, x)$, we first note that this is certainly true if it is not greater than p_l. Now we suppose that $p_r > p_l$, i.e., $x_i^p = 0$ for all $i > w$ with $p_i \geq p_r$. Then $F_{max}(p, x) \geq g^r$, and we have $F_{max}(p + \delta, x) \geq F_{max}(p, x) - \widehat{\varrho}_r \geq g^r - \widehat{\varrho}_r = p_r$.

This establishes the desired inequality.

If we are given any real number $\widehat{\varrho}$ strictly greater than $\min\{\widehat{\varrho}_j \ : \ j \in V(x^p)\}$, then we can find a vector $\delta \in R^n$ such that $|\delta_i| \leq \widehat{\varrho}$ for each index $i \leq w$, $\delta_i = 0$ for each index $i > w$, while $F_{max}(p + \delta, x) < F_{max}(p + \delta, x^p)$ for some feasible solution $x \in X$. The argument is quite similar to that used in the proof of Theorem 2.12, and therefore we omit the details. So, this completes the proof.

◇

Similarly as Corollary 2.10 was proven on the basis of Theorem 2.12, next we prove analogous Corollary 2.11 on the basis of Theorem 2.13.

To compute value $b^{[j,m]}$, $j \in V(x^p), j \leq w, m \geq w$, we need to solve the instance of problem (2.31) with objective vector $\tilde{p}^* = (\tilde{p}_1^*, \tilde{p}_2^*, \ldots, \tilde{p}_n^*) \in R_+^n$, where $\tilde{p}_i^*$ is equal to a value g for $i = j$ and all $i > m$, and $\tilde{p}_k^* = p_k$ for all other indices $k \in N$. If the optimal objective value of this problem instance turns out to be g, then $b^{[j,m]} = \infty$. Otherwise, the optimal objective value is exactly $b^{[j,m]}$ and we obtain an optimal solution $x^{[j,m]}$ and the corresponding value $d^{[j,m]}$. In fact, to compute $\widehat{\varrho}_j$ it actually suffices to calculate $b^{[j,m]}$ in order of decreasing m until a value of m is reached for which inequality

$$\frac{b^{[j,m]} - p_j}{2} \geq d^{[j,m]}$$

holds, because $b^{[j,m]}$ is non-increasing in m.

For $j \in V(x^p), j > w$, let g^j denote the optimal value of the problem instance with the objective vector p and the additional restrictions $x_j = 0$ and $x_i = 0$ for all indices $i > w$ with $p_i \geq p_j$. If this problem instance does not have a feasible solution, then we obtain $g^j = \infty$. For $j \in V(x^p), j > w$, we obtain

$$\widehat{\varrho}_j = g^j - p_j. \tag{2.40}$$

The calculation of g^j is obvious. Thus, we proved the following claim.

Corollary 2.11 *If inequalities $1 \leq w < n$ hold, then stability radius $\widehat{\varrho}_0^w(x^p, p)$ can be calculated by solving $\sum_{i=1}^n x_i^p$ instances of problem (2.31).*

Corollaries 2.10 and 2.11 imply the following claim.

Theorem 2.14 *If Boolean bottleneck programming problem (2.31) is polynomially solvable for any objective vector $p \in R_+^n$, then stability radius $\widehat{\varrho}_0^w(x^p, p)$ of an optimal solution x^p can be calculated in polynomial time.*

2.6 Comments and References

This is no surprise, since it has been shown in [RC95] and [vHW99] that, even if $w = 1$, computing the stability radius exactly for any $\epsilon \geq 0$, is NP-hard if the associated optimization problem (in our case: the scheduling problem) is NP-hard.

The traveling salesman problem was the first linear trajectory problem for which a formula for calculating the stability radius of the whole set of optimal trajectories has been derived [Leo75]. The proof of Theorem 4 can be found in [Sot93]. In the following we strengthen an upper bound for the stability radius given in [Sot93] by using the following result about the existence of an infinitely large stability radius of an ϵ-approximate solution.

Necessary and sufficient conditions for $\varrho_\epsilon^w(x, p) = 0$ have been given in [Sot93], which are valid only for $\tilde{x} \neq \tilde{0}$, where $\tilde{0}$ is the zero vector in R_+^w.

A concept of stability analysis for the latter problem has been developed in [GL80, GL85, Leo75, Leo76, Lib91, Tar82] and in some other papers (see [SLG95] for the ixtensive

survey). It should be noted that most results have been obtained for the stability radius of the whole set of optimal trajectories, i.e., for the largest radius of an open ball in the space of the numerical input data such that a new optimal trajectory does not arise. Unfortunately, the set of all optimal trajectories is often unknown since its cardinality may be large. Even if the optimal trajectory is unique for the problem, this information usually is inaccessible for OR workers. On the other hand, the investigation of the stability radius of one optimal trajectory of such a problem has the following drawback: The stability radius of an optimal solution of a linear trajectory problem is equal to zero, if at least one alternative optimal solution exists. Therefore, in [KSL98, Sot93] the stability of an ϵ-approximate solution has been investigated.

In [Sot93] the following theorem has been proved.

There exist a lot of papers presenting different approaches to stability analysis of *discrete optimization problems*, and in the last part of this section, we provide a sketch of some approaches to stability analysis, which are close to the subject of this dissertation.

A related approach to stability analysis for *linear trajectory problems* (such as the *traveling salesman problem*, the *assignment problem*, the *shortest path problem*) and some other discrete optimization problems has been initiated in [GL80, Leo75, Leo76, Lib91, SW80, Tar82] and developed in some other papers (see Sotskov et al. [SLG95] for an extensive survey). Most results have been obtained for the stability radius of the whole set of solutions (optimal trajectories), i.e. for the largest radius $\varrho(p)$ of an open ball in the space of the numerical input data p such that a new optimal trajectory does not arise. A formula for calculating the stability radius $\varrho(p)$ of the set of all solutions of the traveling salesman problem is obtained by Leontev [Leo75, Leo76] and the extreme values of $\varrho(p)$ are determined. Gordeev and Leontev [GL80] derive analogous results for a similar problem with a *bottleneck objective function*. A specific transformation of a branch-and-bound algorithm for the traveling salesman problem for calculating $\varrho(p)$ is suggested by Gordeev et al. [GLS83]. Gordeev [Gor89] proposes a polynomial algorithm for calculating the stability radius of the whole set of solutions of *extremal problems on matroids* and on the intersection of two matroids.

It should be noted that related approaches to stability analysis for the traveling salesman problem, the shortest path problem, and some others, which can be represented as a *binary optimization problem* with a linear objective function, are developed in [GL80, Lib99, Lib91, LvdPSvdV96, Tar82].

The complexity of calculating the stability radius $\varrho(p)$ of a solution of a discrete optimization problem is studied in [GL85, RC95]. Ramaswamy and Chakravarti [RC95] show that the problem of determining the *arc tolerance* for a discrete optimization problem is as hard as the problem itself (the arc tolerance is the maximum change, i.e. increase or decrease, of a single weight, which does not destroy the optimality of a solution). This means that in the case of the traveling salesman problem, the arc tolerance problem is NP-hard even if an optimal tour is given. Gordeev [Gor89] proved the NP-hardness of the problem of calculating $\varrho(p)$ for the polynomially solvable shortest path problem in a digraph without negative circuits. Sotskov et al. [SWW98] show that the stability radius of an approximate solution may be calculated in polynomial time if the number of unstable components grows rather slowly, namely as $O(\log N)$, where N is the number of cities in the traveling salesman problem. Libura et al. [LvdPSvdV96] argue that it is rather convenient from a computational point of view to use the set of k shortest tours

when applying a stability analysis to the *symmetric traveling salesman problem.*

In deterministic scheduling theory the processing times are supposed to be given in advance, i.e., before applying a scheduling procedure. More general cases have been considered in stochastic scheduling (see [Pin95a]), where p_i is a random variable with a known distribution of probabilities. However, in practice such functions may also be unknown. The results surveyed and developed in this paper may be considered as an attempt to initialize further investigations of scheduling problems under conditions of uncertainty.

We applied the same stability analysis for a large scale of scheduling problems: That which may be represented as linear binary programming problem and more general scheduling problems which may be represented as extremal problem on a disjunctive (mixed) graph. Of cause, complexity of the problem has to be taken into account: The stability results which seems to be appropriate for the general shop problem (see Sections 2 – 6) are rather rough for the linear binary programming problem which allows to derive more deep mathematical results and more efficient algorithms.

In turn, stability properties of an optimization problem may be used to characterize its complexity. We can illustrate it on a job shop problem from the first part, and on a traveling salesman problem and an assignment problem from the second part. (The letter means optimal distribution of n jobs on m parallel non-identical machines in a single-stage system). The stability radius of optimal schedule for problem $J//C_{max}$ have usually strictly positive stability radius, even if it is not unique optimal schedule (see Theorem 1 and computational results in Section 6). On the other hand, it is easy to show that $\varrho_0^{\omega}(x,p) = 0$ if there exist at least two optimal solutions for a traveling salesman problem (or for an assignment problem). Because of this reason, the main focus in the second part of the paper was given to stability analysis of ϵ-approximate solution (see Theorems 6 and 7). As follows from [BSW96] a general shop problem with mean flow time criterion (i.e. of type 'min-sum') becomes even more difficult for stability analysis than problem G
C_{max}.

So possible trends for future research may be to investigate connections between the complexity of scheduling problems and the complexity of calculating the stability radius of an optimal schedule (see [RC95, SLG95, vHW99]). Recall that in [RC95, vHW99] it was shown that the existence of a polynomial algorithm for calculating $\varrho_0^1(x,p)$ implies a polynomial algorithm for problem (6.1). In [vHW99] similar implication was proven also for the case $\epsilon > 0$. Moreover, in [RC95] it was shown that if problem (6.1) is polinomially solvable, then $\varrho_0^1(x,p)$ may be calculated in polynomial time. Thus the value $\varrho_0^1(x,p)$ may be calculated in polynomial time for the assignment problem, if optimal solution x is given, while the calculating the same value for traveling salesman problem is NP-hard. Interesting subject for research may be connected with generalization of the result from [RC95]: Is it possible to find the stability radius $\varrho_s(p)$ of an optimal schedule s in polynomial time, if s may be constructed in polynomial time?

The setting of scheduling problem in the first part of the paper is so general that it is unlikely to find simple answers to those questions, which are usual for deterministic scheduling problems. So, possible stability analysis may consist in determining classes of rather simple scheduling problems for which it is possible to find the stability radius of an optimal or an approximate solution in a suitable time, e.g., if the stability region for such

a class will be convex, an algorithm similar to that developed in the proof of Theorem 11 may be applied. Therefore also interesting topic of research is to establish that for some kinds of scheduling problems the stability region of an optimal schedule is a convex set.

Important question is connected with the determination of simple conditions (preferably conditions which can be verified in polynomial time) for the validity of $\varrho_s(p) = 0$ similar to those derived in [KSW95] for $\varrho_s(p) = \infty$ for the problems $J//C_{max}$ and $J//L_{max}$. It is of interest also to develop simple bounds for $\varrho_s(p)$ and $\varrho_\epsilon^w(x,p)$ (see e.g. (2.2), Theorem 5, Corollary 2 and Theorem 2.4). An interesting question is how a branch and bound algorithm, which is often used for NP-hard scheduling problems, can be combined with calculating (bounds on) the stability radius of an optimal or ϵ-approximate schedule (see [LvdPSvdV96, SSW97]).

Concluding we can note that the above approach to stability analysis is not the only possible one (see survey [SLG95]). For example, in [KRKvHW94] the sensitivity of a heuristic calculating algorithm with respect to the variation of the processing time of one job has been investigated.

Table 2.1: *Notations for Boolean programming*

Symbols	Description
X	Set of feasible solutions x (n-dimensional Boolean vectors): $x \in X \subseteq \{0,1\}^n$
N	Subset of natural numbers: $N = \{1, 2, \ldots, n\}$
$F(p,x)$	Linear objective function: $F(p,x) = \sum_{i \in N} p_i x_i$
p	Objective vector $p = (p_1, p_2, \ldots, p_n) \in R^n_+$, where p_i is objective coefficient
x^p	Optimal solution (optimal vector) $x^p = (x^p_1, x^p_2, \ldots, x^p_n) \in X \subseteq \{0,1\}^n$: $F(p,x^p) = \min\{F(p,x) \ : \ x \in X\}$
x	ϵ-approximate solution $x = (x_1, x_2, \ldots, x_n) \in X$, if $F(p,x) \leq (1+\epsilon) \cdot F(p,x^p)$
w	Number of unstable objective coefficients
$\tilde{x}$	Vector $\tilde{x} = (x_1, x_2, \ldots, x_w)$ of the first w components of n-dimensional Boolean vector $x \in \{0,1\}^n$, $1 \leq w \leq n$
$\tilde{X}$	Set of w-dimensional Boolean vectors of the first w components of feasible solutions x: $\tilde{X} = \{\tilde{x} \in \{0,1\}^w \ : \ x \in X \subseteq \{0,1\}^n\}$
$\tilde{p}$	w-dimensional vector of unstable objective coefficients $\tilde{p} = (p_1, p_2, \ldots, p_w)$ of the objective vector p with $n-w$ stable objective coefficients: $p_{w+1} = \bar{p}_{w+1}, \ p_{w+2} = \bar{p}_{w+2}, \ldots, \ p_n = \bar{p}_n$
$K^w_\epsilon(x)$	Stability region of an ϵ-approximate solution $x \in X$: $K^w_\epsilon(x) = \{\tilde{p} \in R^w_+ \ : \ F(p,x) \leq (1+\epsilon) \cdot F(p,x^p)\}$
$O^w_\varrho(p)$	Stability ball of ϵ-approximate solution $x \in X$, if $O^w_\varrho(p) \cap R^w_+ \subseteq K^w_\epsilon(x)$
$\varrho^w_\epsilon(x,p)$	Stability radius of the ϵ-approximate solution $x \in X$ of the Boolean programming problem with linear objective function
$\varrho^w_0(x,p)$	Stability radius of the optimal solution $x = x^p \in X$ of the Boolean programming problem with linear objective function
U	Binary relation on set $\tilde{X}$ with maximal cardinality: If $(\tilde{x}, \tilde{x}') \in U$ and $i \in \{1, 2, \ldots, w\}$, then $x_i = 1$ implies $x'_i = 1$
$\overline{0}$	Zero vector in the space R^w_+: $\overline{0} = (0, 0, \ldots, 0) \in R^w_+$
p_*	Non-negative real number defined as follows: $p_* = \max\{p_i \ : \ 1 \leq i \leq n\}$
p_ϵ	Non-negative real number defined as follows: $p_\epsilon = \max\left\{\max\{p_i \ : \ 1 \leq i \leq w\}, \ (1+\epsilon) \cdot \sum_{i=w+1}^n p_i - \sum_{i=w+1}^n p_i x_i\right\}$
Δ_p	Non-negative real number defined as follows: $\Delta_p = (1+\epsilon) \cdot F(p,x^p) - F(p,x)$
$V(x)$	Set of indices of 1's in the first w components of n-dimensional Boolean vector $x \in X \subseteq \{0,1\}^n$, $1 \leq w \leq n$
$v(x)$	Cardinality of set $V(x)$: $v(x) = \lvert V(x) \rvert$
$F_{max}(p,x)$	Maximum objective function: $F_{max}(p,x) = \max_{i \in N}\{p_i x_i\}$
$\widehat{\varrho}^w_0(x,p)$	Stability radius of the optimal solution $x = x^p \in X$ of the Boolean programming problem with min-max criterion
$x^{[j]}$	Optimal solution of the modified instance of the given problem with $w = n$: $F_{max}(p,x^{[j]}) = \min\{F_{max}(p,x) \ : \ x \in X, x_j = 0\}$
$b^{[j]}$	Optimal objective value for the modified instance of the given problem with $w = n$: $b^{[j]} = \max_{i \in N} p_i x^{[j]}_i$
ϱ_j	Non-negative real number defined as follows: $\varrho_j = (b^{[j]} - p_j)/2$
$x^{[j,m]}$	Optimal solution of the modified instance of the given problem: $F_{max}(p,x^{[j,m]}) = \min\{F_{max}(p,x) \ : \ x \in X, x_j = 0, x_i = 0, i > m\}$
$b^{[j,m]}$	Optimal objective value for the modified instance of the given problem: $b^{[j,m]} = \max_{i \in N} p_i x^{[j,m]}_i$

Chapter 3

General and Job Shops with Uncertain Processing Times

This chapter deals with the general shop and job shop scheduling problems with the objective to minimize the makespan or mean flow time provided that the numerical input data are uncertain.

In [Pin95a], it was noted that one "source of uncertainty is processing times, which, typically, are not known in advance. Thus, a good model of a scheduling problem would need to address these forms of uncertainty." In stochastic settings of scheduling problems, the random processing time of an operation is assumed to take a known probability distribution. The scheduling environments that we consider in this chapter are so uncertain that all information available about the processing time of an operation is its upper and lower bounds.

To be more specific, we consider problem $\mathcal{G}/a_i \leq p_i \leq b_i/\Phi$ and problem $\mathcal{J}/a_i \leq p_i \leq b_i/\Phi$ for dealing with uncertain scheduling environments in which only lower bound a_i and upper bound b_i for the processing time p_i of operation $i \in Q$ are known before scheduling. Such problems may arise in many practical situations since, even if no specific bounds for an uncertain processing time p_i are known in advance, we can set $a_i = 0$ and b_i equal to the planning horizon. In spite of practical interest, such a type of scheduling problems was considered in a limited OR literature so far.

In this chapter, we use a mixed graph model for representing the input data, the scheduling process and the final solution. Our 'strategy' is to separate the 'structural' input data from the 'numerical' input data as much as possible. The precedence and capacity constraints (i.e., the structural input data) are given by the mixed graph G, which completely defines the set of semiactive schedules. The set of optimal semiactive schedules is defined by the weighted mixed graph $G(p)$ which presents both the structural and numerical input data.

In Section 3.1, we define a solution of problem $\mathcal{G}/a_i \leq p_i \leq b_i/\Phi$ as a set $\Lambda^*(G)$ of digraphs (semiactive schedules) with the following property: For any fixed vector p of the processing times with components p_i in the closed intervals $[a_i, b_i],\ i \in Q$, there exists at least one digraph in set $\Lambda^*(G)$ which is optimal.

This chapter is organized as follows. In section 3.1, we demonstrate some preliminary ideas of our approach using results from Chapter 1 for solving a small example of a job shop problem $\mathcal{J}/a_i \leq p_i \leq b_i/\Phi$ with uncertain processing times. Section 3.2 deals with the

required mathematical background for later presentations. In Section 3.3, we present the main formulas and an algorithm for solving problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$ and show how to restrict the number of digraphs which have to be considered for calculating stability radius. In sections 3.1 - 3.3, we present an approach to deal with problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$ based on an improved stability analysis of an optimal schedule and demonstrate this approach on an example of the job shop problem $\mathcal{J}/a_i \le p_i \le b_i/\mathcal{C}_{max}$. A summary and some remarks to this chapter are given in Section 3.7.

In the course of this chapter, an *optimal* schedule (digraph), a *better* and a *best* schedule (digraph) are considered with respect to criterion $\mathcal{C}_{max}$ (Sections 3.1 - 3.6) or criterion $\sum \mathcal{C}_i$ (Sections 3.4 - 3.6). All necessary notions from Chapter 1 are generalized for problem $\mathcal{G}/a_i \le p_i \le b_i/\Phi$ for a more effective use.

3.1 Minimal Solution

Let us consider a general shop scheduling problem as described in Section 1.1. In a *deterministic* setting of scheduling problem, the processing times p_i are assumed to be known exactly for all operations $i \in Q$, and as it was mentioned in Chapter 1, a schedule may be defined by the start times s_i or completion times c_i of all operations $i \in Q$. Unfortunately, in many practical cases, the operation processing times are unknown before scheduling or only known with some error. In such an uncertain environment, it is not possible to determine a priori the starting times or completion times of all operations.

We represent the input data of a general shop scheduling problem by a mixed graph $G = (Q, A, E)$ introduced in Section 1.1. If the processing times of all operations Q are known exactly before scheduling, we associate a non-negative weight p_i (operation duration) with each vertex $i \in Q$ in the mixed graph $G = (Q, A, E)$ to obtain the weighted mixed graph denoted by $G(p) = (Q(p), A, E)$.

As in Chapter 1, $\Lambda(G) = \{G_1, G_2, \ldots, G_\lambda\}$ denotes the set of all feasible (i.e., circuit-free) digraphs $G_s = (Q, A \cup E_s, \emptyset) \in \Lambda(G)$ generated from the mixed graph $G = (Q, A, E)$ via substituting its edges E by arcs E_s. Given a vector $p = (p_1, p_2, \ldots, p_q)$ of the processing times, weighted digraph $G_s(p) = (Q(p), A \cup E_s, \emptyset)$ corresponding to feasible digraph $G_s = (Q, A \cup E_s, \emptyset) \in \Lambda(G)$ uniquely defines the *earliest completion time* $c_i(s)$ of each operation $i \in Q$ along with the makespan value $\max\{c_i(s) : i \in Q\}$ of schedule s. Since p_i is non-negative for each $i \in Q$, the running time of calculating $c_1(s), c_2(s), \ldots, c_q(s)$ for the given weighted digraph $G_s(p)$ may be restricted by $O(q^2)$ where q is the number of operations: $q = |Q|$. The maximal weight of a path in the weighted digraph $G_s(p)$ (called *critical weight*) defines the makespan, $\max\{c_i(s) : i \in Q\}$, of schedule s. The path in $G_s(p)$ with a critical weight is called a *critical path*.

As already mentioned in Section 1.1, given a fixed vector $p = (p_1, p_2, \ldots, p_q)$ of the processing times, in order to construct an optimal schedule for the general shop problem $\mathcal{G}//\mathcal{C}_{max}$, one may first enumerate (explicitly or implicitly) all *feasible* digraphs $G_1(p)$, $G_2(p), \ldots, G_\lambda(p)$ and then select an *optimal* digraph, i.e., one with a minimal value of the critical weight among all λ feasible digraphs.

It is worthwhile to note that the *feasibility* of a schedule s (and the *feasibility* of a weighted digraph $G_s(p)$) is independent of the vector $p = (p_1, p_2, \ldots, p_q)$ of the processing times, while the *optimality* of a weighted digraph $G_s(p)$ depends on the vector p. In

other words, the set $\Lambda(G) = \{G_1, G_2, \ldots, G_\lambda\}$ of feasible digraphs is completely defined by the mixed graph $G = (Q, A, E)$ (without weights p) while the information on the vector p of the processing times is needed to determine whether a schedule $k \in S$ is optimal or not, i.e., the optimality of a schedule is defined by the weighted mixed graph $G(p) = (Q(p), A, E)$. As in Chapter 1, $S = \{1, 2, \ldots, \lambda\}$ denotes the set of all semiactive schedules (see Definition 1.1 at page 11).

If vector p of the processing times is not known exactly before applying a scheduling procedure (e.g., the processing times may *vary* in a practical realization of a schedule), different realizations may result in different critical paths in the weighted digraph $G_s(p)$.

For practical problems, the cardinality λ of the set $\Lambda(G)$ may be huge (obvious upper bound $\lambda \leq 2^{|E|}$ could be tight). However, as we will show, we often need only to consider some subset B of the set $\Lambda(G) : B \subseteq \Lambda(G)$. Since $p_i \geq 0$ for all $i \in Q$, we obtain the equality

$$\max_{i \in Q} c_i(s) = \max_{\mu \in H_s} l^p(\mu),$$

and from equality (1.13) (see page 19) it follows that digraph $G_s(p)$ has the minimal critical weight within the set $B \subseteq \Lambda(G)$ if and only if

$$\max_{\mu \in H_s} l^p(\mu) = \min_{G_k \in B} \max_{\nu \in H_k} l^p(\nu). \tag{3.1}$$

For the case $B = \Lambda(G)$, the equality (3.1) provides an *optimality* criterion of a schedule $s \in S$ (if vector p is fixed).

In this chapter, we allow the duration p_i of an operation $i \in Q$ to assume any value in the fixed closed interval $[a_i, b_i]$, $0 \leq a_i \leq b_i$, defined by inequalities (1) (see Condition 5 at page 6). Such a general shop scheduling problem with *uncertain* processing times is denoted by $\mathcal{G}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$. As it was already mentioned, the deterministic problem $\mathcal{G}//\mathcal{C}_{max}$ is a special case of a general shop problem $\mathcal{G}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ with *uncertain* processing times (namely, when $a_i = b_i$ for each operation $i \in Q$). Also, one can interpret p_i in the problem $\mathcal{G}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ as a random variable x_i with the following cumulative probability distribution function:

$$F_i(t) = P(x_i < t) = \begin{cases} 0, & \text{if } t < a_i, \\ 1, & \text{if } t = b_i. \end{cases}$$

The density function $f_i(t)$ of a cumulative probability distribution is uncertain in the closed interval $[a_i, b_i]$ for operation $i \in Q$:

$$f_i(t) = \frac{dF_i(t)}{dt} = \begin{cases} 0, & \text{if } t < a_i, \\ ? & \text{if } a_i \leq t \leq b_i, \\ 0, & \text{if } t > b_i. \end{cases}$$

In the framework of stochastic scheduling ([Pin95a], pp. 167 – 252), each random variable x_i associated with the processing time of operation $i \in Q$ (and perhaps similar random variables associated with release times, due dates, setup times, etc.) is assumed to have a *known* probability distribution. For example, a stochastic version of the problem $\mathcal{G}//\mathcal{C}_{max}$ with exponential continuous time distributions with rates α_i, $i \in Q$, is denoted by $\mathcal{G}/p_i \sim exp(\alpha_i)/E\mathcal{C}_{max}$, where density function of an exponentially distributed random

variable x_i is $f_i(t) = \alpha_i e^{-\alpha t}$, the corresponding probability distribution function is $F_i(t) = P(x_i < t) = 1 - e^{-\alpha t}$, and

$$E_i(x_i) = \int_0^\infty t f_i(t) dt = \int_0^\infty t dF_i(t) = \frac{1}{\alpha_i}$$

is the expected (mean) value of random processing time x_i. The objective of problem $\mathcal{G}/p_i \sim exp(\alpha_i)/E\mathcal{C}_{max}$ is to minimize the expected value of makespan $E\mathcal{C}_{max}$ for a schedule which may be constructed using an appropriate *scheduling policy*.

The approach we present in this chapter for solving problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$ is based on an improved stability analysis of an optimal digraph (see Sections 1.2 - 1.4). As follows from Chapter 1, an optimal digraph $G_s \in \Lambda(G)$ provides a solution of problem $\mathcal{G}//\mathcal{C}_{max}$. In other words, an optimal digraph defines a set of m sequences of the operations Q_k processed by machine $M_k, k = 1, 2, \ldots, m$, with a minimal value of the makespan among all feasible schedules when the vector $p = (p_1, p_2, \ldots, p_q)$ of the processing times is given.

Next, we define a *solution* of problem $\mathcal{G}/a_i \le p_i \le b_i/\Phi$ for the general case $a_i \le b_i, i \in Q$. Let T denote the polytope in the space R^q_+ (with maximum metric) defined by inequalities (1) (see page 6):

$$T = \{x = (x_1, x_2, \ldots, x_q) : \ a_i \le x_i \le b_i, \ i \in Q\}.$$

A set $\Lambda^*(G) \subseteq \Lambda(G)$ of feasible digraphs is called a *solution* of the problem $\mathcal{G}/a_i \le p_i \le b_i/\Phi$ if this set contains at least one optimal digraph for each fixed vector $x \in T$ of the processing times. Note again that instead of considering a set of digraphs $\Lambda^*(G)$, we can consider directly a set of schedules $S^* \subseteq S = \{1, 2, \ldots, \lambda\}$ which can be induced by the set $\Lambda^*(G)$:

$$S^* = \{k \ : \ G_k \in \Lambda^*(G)\}.$$

Obviously, the whole set $\Lambda(G)$ may be considered as a solution of problem $\mathcal{G}/a_i \le p_i \le b_i/\Phi$ for any given polytope $T \subseteq R^q_+$, i.e., for each pair of vectors $a = (a_1, a_2, \ldots, a_q) \in R^q_+$ and $b = (b_1, b_2, \ldots, b_q) \in R^q_+$ with $a_i \le b_i, \ i = 1, 2, \ldots, q$. However, such solution $\Lambda(G)$ is usually redundant: Polytope T may contain no point p, where some digraph from set $\Lambda(G)$ is optimal. Moreover, the construction of the whole set $\Lambda(G)$ is only possible for a very small size of a scheduling problem since the cardinality λ of the set $\Lambda(G)$ could be equal to $2^{|E|}$. Note also that during the realization of a schedule, a decision-maker may have difficulties while dealing with such a huge set of possible candidates for schedule realization.

Therefore, it is practically important to look for a *'minimal solution'* $\Lambda^*(G) \subseteq \Lambda(G)$ for problem $\mathcal{G}/a_i \le p_i \le b_i/\Phi$. A set $\Lambda^*(G)$ is called a *minimal solution* if any proper subset of set $\Lambda^*(G)$ is not a solution. Note that set $\Lambda^*(G)$ may be not unique, e.g., since there may exist two or more optimal digraphs for some vector $p \in T$ of the processing times. We combine these arguments as follows.

Definition 3.1 *A set of digraphs $\Lambda^*(G) \subseteq \Lambda(G)$ is called a solution of problem $\mathcal{G}/a_i \le p_i \le b_i/\Phi$ if for each fixed vector $p \in T$ of the processing times the set $\Lambda^*(G)$ contains at least one optimal digraph. If any proper subset of the set $\Lambda^*(G)$ is no longer a solution of problem $\mathcal{G}/a_i \le p_i \le b_i/\Phi$, it is called a minimal solution denoted as $\Lambda^T(G)$.*

Table 3.1 summarizes different settings of a general shop scheduling problem with criterion $\mathcal{C}_{max}$ in accordance with information which is available on a vector p of the processing times. These problems are classified on the basis of a mixed graph approach.

Row 1 in Table 3.1 refers to the *mass* general shop scheduling problem, where the only information requirement on the processing times p is that vector p belongs to the space R^q_+. Thus, individual problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$ turns out into mass general shop scheduling problem if b_i is assumed to be ∞ and $a_i = 0$ for each operation $i \in Q$.

Table 3.1: *Scheduling with different requirements on the numerical data*

	Scheduling problem	Input data	Semiactive schedules	Solution
1	Mass general shop scheduling problem	$G(p) = (Q(p), A, E)$; $0 \le p_i < \infty,\ i \in Q$	$\Lambda(G)$	$\Lambda(G)$
2	Individual problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$	$G(p) = (Q(p), A, E)$; $a_i \le p_i \le b_i,\ i \in Q$	$\Lambda(G)$	$\Lambda^*(G) \subseteq \Lambda(G)$
3	Individual problem $\mathcal{G}/p_i \sim F_i(t)/E\mathcal{C}_{max}$	$G(x) = (Q(x), A, E)$; $F_i(t) = P(x_i < t)$	$\Lambda(G)$	$\{G_s\} \subseteq \Lambda^*(G) \subseteq \Lambda(G)$
4	Individual problem $\mathcal{G}//\mathcal{C}_{max}$	$G(p) = (Q(p), A, E)$; $a_i = p_i = b_i,\ i \in Q$	$\Lambda(G)$	$G_s \in \Lambda(G)$

For any digraph $G_s \in \Lambda(G)$ it is possible to construct problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$ with appropriate values a_i and b_i such that digraph G_s will be optimal in some realization of vector p, $a_i \le p_i \le b_i,\ i \in Q$, of the processing times.

This statement is true since for each critical path $\mu \in H_s$ we can set p_i equal to a sufficiently small real $\epsilon \ge 0$ for each operation

$$i \in Q^0 = [\mu] \setminus \bigcup_{k \ne s} \bigcup_{\nu \in H_k(p)} [\nu],$$

where $\mu \in H_s$ is a critical path in digraph G_s. For such a setting of the processing times, equality (3.1) is satisfied with $B = \Lambda(G)$. In particular, if $Q^0 = Q$, we get a trivial individual problem $\mathcal{G}//\mathcal{C}_{max}$ with $p_i = \epsilon = 0,\ i \in Q$, where any digraph G_s from set $\Lambda(G)$ is optimal.

In this chapter, we consider an individual problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$ which is very general (see row 2 in Table 3.1). In one extreme case when $a_i = 0$ and $b_i = \infty$ for each operation $i \in Q$, problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$ coincides with the whole mass problem presented in row 1. In the other extreme case when $a_i = b_i$ for each $i \in Q$, problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$ is reduced to problem $\mathcal{G}//\mathcal{C}_{max}$ (see row 4), which is one of the basic problems studied in deterministic scheduling.

The more information about the processing times p is available before applying a scheduling procedure, the 'better' solution may be obtained: The cardinality of the minimal solution $\Lambda^T(G)$ is smaller if the polytope T is defined by smaller closed intervals $[a_i, b_i]$. For example, a minimal solution (see Definition 3.1) is reduced to a single optimal digraph $G_s \in \Lambda(G)$ in the case of deterministic problem $\mathcal{G}//\mathcal{C}_{max}$ (see row 4).

Row 3 refers to the individual problem $\mathcal{G}/p_i \sim F_i(t)/E\mathcal{C}_{max}$, one of the basic problems studied in stochastic scheduling, where each operation $i \in Q$ is assumed to be a random variable with a probability distribution $F_i(t)$ known before applying a scheduling procedure. For problem $\mathcal{G}/p_i \sim F_i(t)/E\mathcal{C}_{max}$, the optimal solution may be a single digraph G_s when one adopts a static scheduling policy ([Pin95a], p.178) or a subset of feasible digraphs $\Lambda^*(G)$ when one adopts a dynamic scheduling policy ([Pin95a], p.179).

When a static scheduling policy is adopted, a decision-maker constructs and then uses an optimal schedule $s \in S$ which minimizes the expected makespan $E\mathcal{C}_{max}$ and thus schedule s remains unchanged during the entire process. In the case of a dynamic scheduling policy, an initial schedule s may be constantly revised during the solution process based on the updated information available (see [Pin95a]).

We can note that the minimal solution $\Lambda^T(G)$ for problem $\mathcal{G}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$, may be calculated exactly before realization of the process, while for problem $\mathcal{G}/p_i \sim F_i(t)/E\mathcal{C}_{max}$ the solution may vary and may even be the whole set $\Lambda(G)$ for some of probability distributions $F_i(t)$.

It is worth to note that for all four formulations presented in Table 3.1, set of feasible solutions (set of semiactive schedules) remains the same and therefore the properties of feasible digraphs $\Lambda(G)$ are of particular importance for all problem settings presented in Table 3.1.

Our approach for solving problem $\mathcal{G}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ is based on the stability of an optimal digraph which guarantees that a feasible digraph remains optimal after feasible variations of the processing times. To facilitate the presentation of the main ideas of our approach for solving general shop scheduling problem $\mathcal{G}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$, we demonstrate it using the following small example of job shop problem with uncertain numerical data.

Example 3.1 *We consider the job shop problem $\mathcal{J}3/n = 2, a_i \leq p_i \leq b_i/\mathcal{C}_{max}$, with two jobs $J = \{J_1, J_2\}$ and three machines $M = \{M_1, M_2, M_3\}$. Job J_1 (job J_2, respectively) consists of the following set of ordered operations $\{O_{1,1}, O_{1,2}, O_{1,3}\} = Q^{J_1}$ (operations $\{O_{2,1}, O_{2,2}, O_{2,3}\} = Q^{J_2}$). The assignment of operations $\{O_{1,1}, O_{1,2}, O_{1,3}, O_{2,1}, O_{2,2}, O_{2,3}\}$ to the set of machines M is as follows: $Q_1^J = \{O_{1,1},\ O_{2,2}\}$, $Q_2^J = \{O_{1,2}, O_{2,1}, O_{2,3}\}$, $Q_3^J = \{O_{1,3}\}$.*

For criterion $\mathcal{C}_{max}$, it is useful to consider dummy operation O_{ij} provided that its processing time is equal to zero: $p_{ij} = 0$. To accommodate dummy operations in the framework of the mixed graph $G = (Q^J, A^J, E^J)$, we assume that each dummy operation 'has to be processed' by a special dummy machine with a zero processing time, and we assume that the number of dummy machines is equal to the number of dummy operations. Therefore, each dummy operation is an isolated vertex in the subgraph $(Q^J, \emptyset, E^J)$ of mixed graph G.

In Example 3.1, we consider operations $O_{0,1}$ and $O_{0,2}$ as dummy operations, and machines M_4 and M_5 as dummy machines. Let operation $O_{0,1}$ (operation $O_{0,2}$) denote the start (the end) of a schedule, and so operation $O_{0,1}$ precedes all other operations (all other operations precede operation $O_{0,2}$). We assume that $Q_4^J = \{O_{0,1}\}$ and $Q_5^J = \{O_{0,2}\}$.

The input data of Example 3.1 is represented by the weighted mixed graph $G(p) = (Q^J(p), A^J, E^J)$ in Figure 3.1, where each processing time p_{ij} is presented near the vertex $O_{ij} \in Q^J$, and the vector p of the processing times is defined as follows:

$$p = (p_{1,1}, p_{1,2}, p_{1,3}, p_{2,1}, p_{2,2}, p_{2,3}) = (75, 50, 40, 60, 55, 30).$$

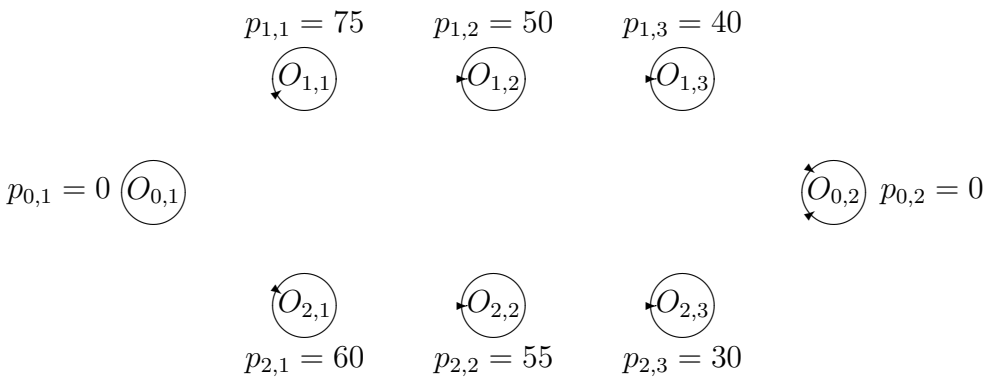

Figure 3.1: *Weighted mixed graph* $G(p) = (Q^J(p), A^J, E^J)$

Vector $p \in R^6_+$ *does not include processing times of dummy operations* $O_{0,1}$ *and* $O_{0,2}$. *For this small example, we can explicitly enumerate all feasible digraphs* $\Lambda(G) = \{G_1, G_2, \ldots, G_5\}$ *with the signatures as follow:*

$$\begin{aligned}
E_1^J &= \{(O_{1,1}, O_{2,2}), (O_{2,1}, O_{1,2}), (O_{1,2}, O_{2,3})\},\\
E_2^J &= \{(O_{1,1}, O_{2,2}), (O_{2,1}, O_{1,2}), (O_{2,3}, O_{1,2})\},\\
E_3^J &= \{(O_{1,1}, O_{2,2}), (O_{1,2}, O_{2,1}), (O_{1,2}, O_{2,3})\},\\
E_4^J &= \{(O_{2,2}, O_{1,1}), (O_{2,1}, O_{1,2}), (O_{1,2}, O_{2,3})\},\\
E_5^J &= \{(O_{2,2}, O_{1,1}), (O_{2,1}, O_{1,2}), (O_{2,3}, O_{1,2})\}.
\end{aligned}$$

We calculate the makespans for all feasible digraphs from set $\Lambda(G)$*:*

$$\begin{aligned}
l_1^p &= \max\{c_{in_i}(1) : J_i \in J\} = 165,\\
l_2^p &= \max\{c_{in_i}(2) : J_i \in J\} = 250,\\
l_3^p &= \max\{c_{in_i}(3) : J_i \in J\} = 270,\\
l_4^p &= \max\{c_{in_i}(4) : J_i \in J\} = 280,\\
l_5^p &= \max\{c_{in_i}(5) : J_i \in J\} = 280,
\end{aligned}$$

and select an optimal weighted digraph $G_1(p) = (Q^J(p), A^J \cup E_1^J, \emptyset)$ *with the signature* $E_1^J = \{(O_{1,1}, O_{2,2}), (O_{2,1}, O_{1,2}), (O_{1,2}, O_{2,3})\}$. *Digraph* $G_1(p)$ *has a minimal critical weight equal to 165 (see Figure 3.2). Using formulas (1.28) and (1.29) (see page 29), we can calculate the stability radius of the optimal digraph* $G_1(p)$. *To this end, we compare digraph* $G_1(p)$ *with each digraph* $G_k \in \Lambda(G) \setminus \{G_1\}$.

In Table 3.2, we present calculations with respect to formulas (1.28) and (1.29) in detail. Each path $\mu \in H_{1k} \subseteq H_1$ *in* $G_1(p)$ *presented in column 3 is compared with each path* $\nu \in H_k$ *presented in column 4 provided that* $l^p(\nu) \geq l_1^p = 165$. *The cardinalities of the sets* $H_{1k}, k = 2, 3, \ldots, \lambda$, *are given in column 2. (Since* $H_{1,2} = \emptyset$ *and* $H_{1,4} = \emptyset$, *digraphs* G_2 *and* G_4 *are not involved in the computations.) The non-decreasing sequence (1.27) of the processing times* $(p_{(0)}^{\nu\mu}, p_{(1)}^{\nu\mu}, \ldots, p_{(\omega_{\nu\mu})}^{\nu\mu})$ *defined at page 29, is given in column 5. In column 6, we present the calculations according to the fraction in the formula (1.28) consecutively for each* $\beta = 0, 1, \ldots, \omega_{\nu\mu}$. *In columns 7, 8 and 9, respectively, we extract the maximum for* $\beta = 0, 1, \ldots, w_{\nu\mu}$, *the maximum for* $\nu \in H_k, l^p(\nu) \geq l_1^p$, *and the minimum for* $\mu \in H_{1k}$ *from the values obtained in column 6. As a result, column 9 presents the*

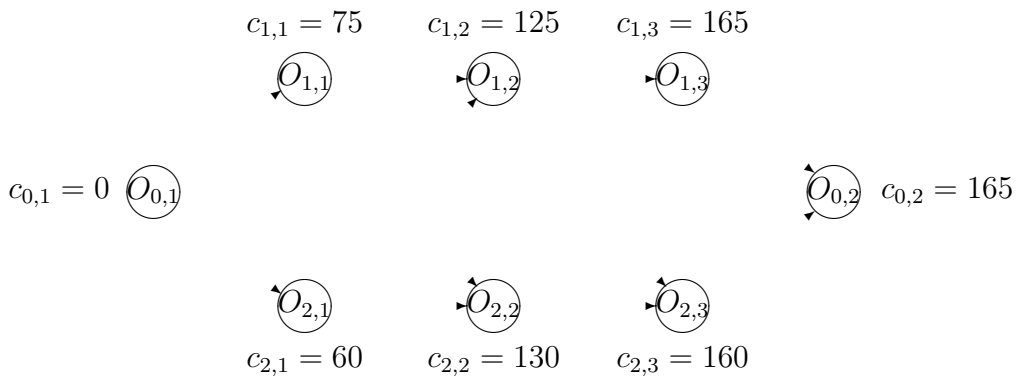

Figure 3.2: *Optimal digraph $G_1 = (Q^J, A^J \cup E_1^J, \emptyset)$ with the completion times c_{ij} presented near the vertices $O_{ij} \in Q^J$*

values of r_{k1} for the digraphs G_k. The last step is to adapt the formula (1.29) from Theorem 1.4 (see page 29). The minimum value $\widehat{r}_{ks}$ is given in column 9.

Due to formula (1.28) we calculate stability radius $\widehat{\varrho}_1(p) = \min\{30, 30\} = 30$. Thus, digraph G_1 remains optimal for any vector $x = (x_{1,1}, x_{1,2}, x_{1,3}, x_{2,1}, x_{2,2}, x_{2,3})$ of the processing times if it distance from the given vector $p = (75, 50, 40, 60, 55, 30)$ is no more than 30 :

$$p_{ij} - 30 \le x_{ij} \le p_{ij} + 30.$$

Therefore, while solving the instance of problem $\mathcal{J}3/n = 2, a_i \le p_i \le b_i/\mathcal{C}_{max}$, digraph $G_1 = (Q^J, A^J \cup E_1^J, \emptyset)$ remains optimal if all possible variations of the processing times $x = (x_{1,1}, x_{1,2}, x_{1,3}, x_{2,1}, x_{2,2}, x_{2,3})$ belongs to stability ball $O_{\widehat{\varrho}_1(p)}(p) = O_{30}(p)$, i.e., if the following inequalities hold:

$$\max_{O_{ij} \in Q^J} \{x_{ij} - a_{ij}, b_{ij} - x_{ij}\} \le 30. \tag{3.2}$$

In such a case, polytope T defined by inequalities (1) in the space R^6_+ has to be completely contained in the stability ball $O_{30}(p)$ of the optimal digraph G_1 : $T \subseteq O_{30}(p)$. In other words, digraph G_1 provides a solution of the problem $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\mathcal{C}_{max}$ as long as inequality (3.2) is satisfied:

$$\Lambda^*(G) = \{G_1\}.$$

It is clear that this solution $\Lambda^*(G)$ is also minimal solution of the problem $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\mathcal{C}_{max}$ due to Definition 3.1.

Thus in the case when inequality (3.2) holds, a decision-maker needs to use only digraph G_1 from the set $\Lambda(G) = \{G_1, G_2, G_3, G_4, G_5\}$ as optimal one in any possible realization of the processing times. In such a case, solution of the problem $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\mathcal{C}_{max}$ turns out to be the same as for the deterministic problem $\mathcal{J}3/n=2/\mathcal{C}_{max}$ with the fixed vector p of the processing times from the polytope T. In other words, the minimal solution of the problem $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\mathcal{C}_{max}$ consists of a single digraph:

$$\{G_1\} = \Lambda^*(G) \subseteq \Lambda^T(G).$$

Let polytope T for the instance $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\mathcal{C}_{max}$ under consideration be given such that inequality (3.2) does not hold. Then optimality of digraph G_1 is not guaranteed

Table 3.2: *Calculation of the stability radius $\widehat{\varrho}_1(p)$ for problem $\mathcal{J}3/n=2/\mathcal{C}_{max}$*

G_k	$\lvert H_{1k}\rvert$	$\mu\in H_{1k}$ $l^p(\mu)$	$\nu\in H_k$ $l^p(\nu)\geq l_1^p$	$p_{(\beta)}^{\nu\mu}$ $0\leq\beta\leq w_{\nu\mu}$	$\dfrac{l^p(\nu)-l^p(\mu)-\sum_{\alpha=0}^{\beta}p_{(\alpha)}^{\nu\mu}}{\lvert[\mu]\cup[\nu]\rvert-\lvert[\mu]\cap[\nu]\rvert-\beta}$	$\max\limits_{\beta}$	$\max\limits_{\nu}$	$\min\limits_{\mu}$
1	2	3	4	5	6	7	8	9
G_2	0							
G_3	1	$(O_{2,1},O_{1,2},O_{1,3})$ $l^p(\mu)=150$	$(O_{1,1},O_{1,2},O_{1,3})$ $l^p(\nu)=165=l_1^p$	$p_{(0)}^{\nu\mu}=0$ $p_{(1)}^{\nu\mu}=75$	$\frac{165-150-0}{2-0}=7.5$ $\frac{165-150-75}{2-1}<0$	7.5	30	30
			$(O_{1,1},O_{1,2},O_{2,1},O_{2,2},O_{2,3})$ $l^p(\nu)=270$ >165	$p_{(0)}^{\nu\mu}=0$ $p_{(1)}^{\nu\mu}=30$ $p_{(2)}^{\nu\mu}=55$ $p_{(3)}^{\nu\mu}=75$	$\frac{270-150-0}{4-0}=30$ $\frac{270-150-30}{4-1}=30$ $\frac{270-150-(30+55)}{4-2}=17.5$ $\frac{270-150-(30+55+75)}{4-3}<0$	30		
G_4	0							
G_5	2	$(O_{1,1},O_{1,2},O_{2,3})$ $l^p(\mu)=155$	$(O_{2,1},O_{2,2},O_{1,1},O_{1,2},O_{1,3})$ $l^p(\nu)=280$ >165	$p_{(0)}^{\nu\mu}=0$ $p_{(1)}^{\nu\mu}=40$ $p_{(2)}^{\nu\mu}=50$ $p_{(3)}^{\nu\mu}=60$	$\frac{280-155-0}{4-0}=31.25$ $\frac{280-155-40}{4-1}=28\frac{1}{3}$ $\frac{280-155-(40+55)}{4-2}=15$ $\frac{280-155-(40+55+60)}{4-3}<0$	31.25	31.25	30
			$(O_{2,1},O_{2,2},O_{2,3},O_{1,2},O_{1,3})$ $l^p(\nu)=235$ >165	$p_{(0)}^{\nu\mu}=0$ $p_{(1)}^{\nu\mu}=40$ $p_{(2)}^{\nu\mu}=55$ $p_{(3)}^{\nu\mu}=60$	$\frac{235-155-0}{4-0}=20$ $\frac{235-155-40}{4-1}=13\frac{1}{3}$ $\frac{235-155-(40+55)}{4-2}<0$ $\frac{235-155-(40+55+60)}{4-3}<0$	20		
		$(O_{1,1},O_{2,2},O_{2,3})$ $l^p(\mu)=160$	$(O_{2,1},O_{2,2},O_{1,1},O_{1,2},O_{1,3})$ $l^p(\nu)=280$ >165	$p_{(0)}^{\nu\mu}=0$ $p_{(1)}^{\nu\mu}=40$ $p_{(2)}^{\nu\mu}=50$ $p_{(3)}^{\nu\mu}=60$	$\frac{280-160-0}{4-0}=30$ $\frac{280-160-40}{4-1}=26\frac{2}{3}$ $\frac{280-160-(40+50)}{4-2}=15$ $\frac{280-160-(40+50+60)}{4-3}<0$	30	30	
			$(O_{2,1},O_{2,2},O_{2,3},O_{1,2},O_{1,3})$ $l^p(\nu)=235$ >165	$p_{(0)}^{\nu\mu}=0$ $p_{(1)}^{\nu\mu}=40$ $p_{(2)}^{\nu\mu}=50$ $p_{(3)}^{\nu\mu}=60$	$\frac{235-160-0}{4-0}=18.75$ $\frac{235-160-40}{4-1}=11\frac{2}{3}$ $\frac{235-160-(40+50)}{4-2}=7.5$ $\frac{235-160-(40+50+60)}{4-3}<0$	18.75		

within the given polytope T: There exists another feasible digraph $G_k\in\Lambda(G)$, $k\neq 1$, (we call it a competitive digraph for G_1) with a critical weight being smaller than critical weight of digraph G_1 in some realization of the processing times $x\in R_+^6$. If such a 'superiority' of the competitive digraph G_k occurs when the processing times are defined by the vector $p^*=(p_{1,1}^*,p_{1,2}^*,\ldots,p_{2,3}^*)\in T$ (i.e., digraph G_k instead of G_1 is optimal for new vector p^* of the processing times), we can calculate stability radius $\widehat{\varrho}_k(p^*)$ of the digraph G_k for this

new vector p^*.

In the case when stability radius $\widehat{\varrho}_k(p^*)$ is strictly positive, we can consider the union $O_{30}(p) \cup O_{\widehat{\varrho}_k(p^*)}(p^*)$ of the two balls (instead of one ball $O_{30}(p)$). If inclusion $T \subseteq O_{30}(p) \cup O_{\widehat{\varrho}_k(p^*)}(p^*)$ holds, then problem $\mathcal{J}2/n = 3, a_i \le p_i \le b_i/\mathcal{C}_{max}$ is solved. In such a case, a decision-maker needs to use either digraph G_1 or digraph G_k for realization of an optimal schedule:

$$\Lambda^*(G) = \{G_1, G_k\}.$$

Otherwise (if inclusion $T \subseteq O_{30}(p) \cup O_{\widehat{\varrho}_k(p^*)}(p^*)$ does not hold), we have to calculate stability radius $\widehat{\varrho}_r(p^{**})$ of a competitive digraph G_r of digraph G_k for such vector p^{**} of the processing times, for which digraph G_r is optimal.

Continuing in this manner, we may cover the given polytope T by the union of the stability balls of some feasible digraphs from set $\Lambda(G)$. As a result, for any vector of the processing times from the polytope T (i.e., whenever inequalities (1) hold), we may obtain at least one optimal schedule.

For Example 3.1 with the original vector $p = (75, 50, 40, 60, 55, 30)$, competitive digraphs for the optimal digraph G_1 are the digraphs $G_3 = (Q^J, A^J \cup E_3^J, \emptyset)$ and $G_5 = (Q^J, A^J \cup E_5^J, \emptyset)$, where $E_3^J = \{(O_{1,1}, O_{2,2}), (O_{1,2}, O_{2,1}), (O_{1,2}, O_{2,3})\}$ and $E_5^J = \{(O_{2,2}, O_{1,1}), (O_{2,1}, O_{1,2}), (O_{2,3}, O_{1,2})\}$. Digraph G_3 along with the completion times of the operations Q^J are presented in Figure 3.3.

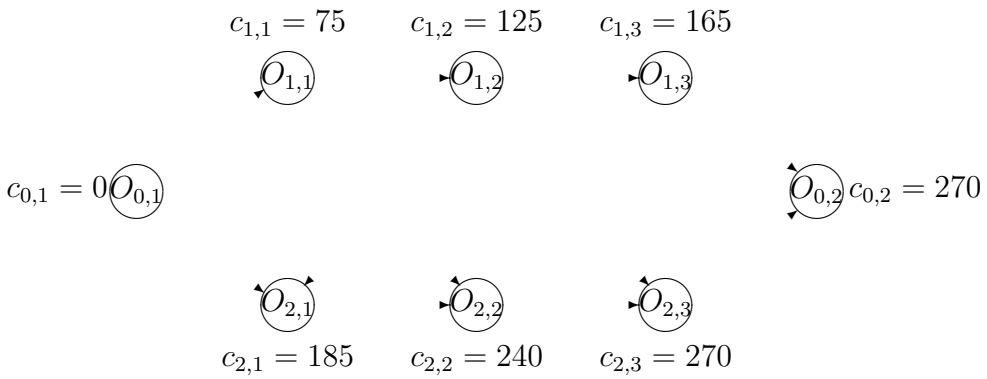

Figure 3.3: *Competitive digraph $G_3 = (Q^J, A^J \cup E_3^J, \emptyset)$ for $G_1 = (Q^J, A^J \cup E_1^J, \emptyset)$ which is optimal for vector $p = (75, 50, 40, 60, 55, 30)$ of the processing times*

As the calculation of the stability radius shows, at the boundary of the ball $O_{30}(p)$ (namely, at the point $p^* = (p_{1,1}^*, p_{1,2}^*, p_{1,3}^*, p_{2,1}^*, p_{2,2}^*, p_{2,3}^*) = (45, 80, 70, 90, 25, 0) \in R_+^6$) both digraphs G_1 and G_3 are optimal. Note that vector p^* is already determined during our calculation of the stability radius on the basis of formulas (1.28) and (1.29). Specifically, vector p^* is obtained from vector p by decreasing the processing times of the operations $O_{1,1}, O_{2,2}, O_{2,3}$ by the value $\widehat{\varrho}_1(p)$:

$$p_{1,1}^* = 75 - 30 = 45,\ p_{2,2}^* = 55 - 30 = 25,\ p_{2,3}^* = 30 - 30 = 0,$$

and by increasing the processing times of the operations $O_{1,2}, O_{1,3}, O_{2,1}$ by the same value $\widehat{\varrho}_1(p)$:

$$p_{1,2}^* = 50 + 30 = 80,\ p_{1,3}^* = 40 + 30 = 70,\ p_{2,1}^* = 60 + 30 = 90.$$

In other words, to obtain vector p^* we can use the following formula:

$$p_i^* = \begin{cases} p_i + r, & \text{if } i \in [\mu], \\ \max\{0, p_i - r\}, & \text{if } i \in [\nu] \setminus [\mu], \\ p_i, & \text{if } i \notin [\mu] \cup [\nu], \mu \in H_{sk}, \nu \in H_k, \end{cases} \tag{3.3}$$

where $[\mu] = \{O_{1,2}, O_{1,3}, O_{2,1}\}, \mu \in H_{1,3}$, $[\nu] = \{O_{1,1}, O_{1,2}, O_{2,1}, O_{2,2}, O_{2,3}\}, \nu \in H_3$, and $r = \widehat{\varrho}_1(p) = 30$. Due to such changes in the processing times, the critical weight of digraph G_1 is increased from 165 to 240, while the critical weight of digraph G_3 is decreased from 270 to 240.

Note that the existence of two or more optimal digraphs is a necessary condition (but not a sufficient one) for the stability radius to be equal to zero (see Theorem 1.1 at page 20). Nevertheless, the 'nonstability' of an optimal digraph may happen at the boundary of a stability region (the stability region of the digraph G_s is the whole set of the vectors $p \in R_+^q$ with the schedule s being optimal), where at least two optimal digraphs exist. Such a situation occurs for Example 3.1 under consideration, namely: $\widehat{\varrho}_1(p^*) = \widehat{\varrho}_3(p^*) = 0$. Indeed, according to Theorem 1.1, there exists a path $\mu^* \in H_1(p)$, $[\mu^*] = \{O_{1,2}, O_{1,3}, O_{2,1}\}$, such that there does not exist any path $\nu \in H_3(p)$ with $[\mu^*] \subseteq [\nu]$. On the other hand, there exists a path $\nu^* \in H_3(p)$, $[\nu^*] = \{O_{1,1}, O_{1,2}, O_{2,1}, O_{2,2}, O_{2,3}\}$, such that there does not exist any path $\mu \in H_1(p)$ with $[\nu^*] \subseteq [\mu]$.

Note also that for point p^*, the only competitive digraph for digraph G_3 is digraph G_1 (and vice versa), where the stability radius of G_1 for the original point $p \in R_+^q$ has been already calculated.

Considering the competitive digraph G_5 instead of the competitive digraph G_3 gives also zero stability radii for both digraphs G_1 and G_5 with the corresponding vector $p' = (105, 20, 10, 30, 85, 60)$ of the processing times, constructed due to (1.24) with $r = \widehat{\varrho}_1(p) = 30$ for the paths $[\mu'] = \{O_{1,1}, O_{2,2}, O_{2,3}\}, \mu' \in H_{1,5}$, and $[\nu'] = \{O_{1,1}, O_{1,2}, O_{1,3}, O_{2,1}, O_{2,2}\}, \nu' \in H_5$.

Thus, we can conclude that using notion of a stability radius given in Definition 1.2 is not sufficient for solving Example 3.1 when inequality (3.2) does not hold for the given set T of feasible vectors of the processing times.

From the above discussion it follows that another type of stability radius is required for solving problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$. While $\widehat{\varrho}_s(p)$ denotes the largest radius of a ball $O_{\widehat{\varrho}}(p)$ within which digraph G_s is 'the best' for the whole set $\Lambda(G)$ (see Definition 1.2 at page 16), we need to determine the largest ball within which digraph G_s is 'the best' for some subset B of the set $\Lambda(G)$ of feasible digraphs.

For Example 3.1 we need to calculate the largest radius of the ball within which digraph G_3 has the minimal critical weight among the feasible digraphs $\Lambda(G)$ except digraph G_1. Indeed, digraph G_1 is optimal within the ball $O_{\widehat{\varrho}_1(p)}(p)$ and so digraph G_1 has already to be contained in the set of candidates for optimal realization: $G_1 \in \Lambda^*(G)$. Thus, in Example 3.1, we need to consider the set

$$B = \Lambda(G) \setminus \{G_1\}$$

instead of the whole set $\Lambda(G)$ of feasible digraphs while calculating stability radius of digraph G_3. It is clear that considering set B instead of set $\Lambda(G)$ may increase such a stability radius in an appropriate way.

In Section 3.2, we propose another definition (more general than Definition 1.2) of stability radius which is suitable for solving problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$. Note that the given bounds a_i and b_i for possible variations of the processing time $x_i,\ i \in Q$, may also enlarge the stability ball of the optimal digraph G_s. E.g., this is true for Example 3.1 since inequality (3.2) becomes only a sufficient condition for the optimality of digraph G_1 (but not a necessary one).

In Section 3.2, we provide both necessary and sufficient conditions for a zero (and for an infinitely large) stability radius. In Section 3.3, the formulas (1.28) and (1.29) proven for the case of calculating the stability radius with $0 \le p_i < \infty,\ i \in Q$, are generalized to the case when the variations of the processing times are given by inequalities (1) and some feasible digraphs have to be excluded from the comparisons with 'the best' one.

3.2 Relative Stability Radius

In Chapter 1, stability radius $\widehat{\varrho}_s(p)$ of an optimal digraph G_s has been investigated which denotes the largest quantity of independent variations of the processing times p_i of operations $i \in Q$ within the interval $[0, \infty)$ such that digraph G_s remains 'the best' (i.e., the weighted digraph $G_s(p)$ has the minimal critical weight) among all feasible digraphs $\Lambda(G)$ (see Definition 1.2 at page 16).

From Example 3.1 it follows that for solving problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$, we need a more general notion of a stability radius since the processing time of operation $i \in Q$ falls within the given closed interval $[a_i, b_i],\ 0 \le a_i \le b_i$, and competitive digraphs have to belong to a subset B of set $\Lambda(G)$. The following generalization of stability radius $\widehat{\varrho}_s(p)$ (we call it *relative* stability radius) is defined by considering the closed interval $[a_i, b_i]$ instead of $[0, \infty)$ and considering set $B \subseteq \Lambda(G)$ instead of the whole set $\Lambda(G)$. In Definition 3.2, l_s^p denotes the critical weight of the weighted digraph $G_s(p)$, $p \in T$, see equality (1.15) at page 19.

Definition 3.2 *Let digraph $G_s \in B \subseteq \Lambda(G)$ have the minimal critical weight $l_s^{p'}$ for each vector $p' \in O_\varrho(p) \cap T$ among all digraphs from the given set B*

$$l_s^{p'} = \min\{l_k^{p'} \ :\ G_k \in B\}. \tag{3.4}$$

The maximal value of radius ϱ of such a ball $O_\varrho(p)$ is denoted by $\widehat{\varrho}_s^B(p \in T)$ and is called relative stability radius of digraph G_s with respect to polytope T (for criterion $\mathcal{C}_{max}$).

Note that relativity of $\widehat{\varrho}_s^B(p \in T)$ is defined not only by polytope T of feasible vectors, but also by set B of feasible digraphs. From Definition 1.2 and Definition 3.2, if follows:

$$\widehat{\varrho}_s(p) = \widehat{\varrho}_s^{\Lambda(G)}(p \in R_+^q).$$

Thus, relative stability radius is equal to the *maximal error* of the given processing times $p_i\ \ (a_i \le p_i \le b_i,\ i \in Q)$ within which the 'superiority' of digraph G_s is still preserved over the given subset B of feasible digraphs.

The following two extreme cases of relative stability radius are of particular importance for solving problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$. On the one hand, if for any positive real number

$\epsilon > 0$ which may be as small as desired, there exist vector $p' \in O_\epsilon(p) \cap T$ and digraph $G_k \in B$ such that $l_s^{p'} > l_k^{p'}$, we obtain zero relative stability radius:

$$\widehat{\varrho}_s^B(p \in T) = 0.$$

On the other hand, if $l_s^{p'} \leq l_k^{p'}$ for any vector $p' \in T$ and for any digraph $G_k \in B$, we obtain infinitely large relative stability radius:

$$\widehat{\varrho}_s^B(p \in T) = \infty.$$

Note that even in the case of finite upper bonds ($b_i < \infty, i \in Q$), i.e., when the maximal error of processing time p_i for each operation $i \in Q$ is restricted by

$$\epsilon_{max} = \max\{\{p_i - a_i, b_i - p_i\} \ : \ i \in Q\}, \tag{3.5}$$

value of $\widehat{\varrho}_s^B(p \in T)$ may be infinitely large as it follows from Definition 3.2. Deterministic problem $\mathcal{G}//\mathcal{C}_{max}$ with vector p of the processing times and optimal digraph G_s provides such a trivial example with an infinitely large relative stability radius $\widehat{\varrho}_s^B(p \in T)$. Indeed, if $a_i = p_i = b_i$ for each operation $i \in Q$, then polytope T degenerates into a single point:

$$T = \{p\}$$

and therefore from inclusion $p' \in O_\varrho(p) \cap T$ it follows that vector p' mentioned in Definition 3.2 is definitely equal to vector p, for which digraph G_s is optimal.

To characterize the extreme values of $\widehat{\varrho}_s^B(p \in T)$, we define the following binary relation which generalizes the dominance relation used in Chapter 1.

Definition 3.3 *Path ν dominates path μ in set T if and only if for any vector $x = (x_1, x_2, \ldots, x_q) \in T$ the following inequality holds:*

$$l^x(\mu) \leq l^x(\nu). \tag{3.6}$$

Note that binary relation introduced in Definition 3.3 is an extension of the dominance relation introduced in Definition 1.3 (see page 17) in the sense that path ν dominates path μ in any set $T \subseteq R_+^q$ (i.e., due to Definition 3.3) if path ν dominates path μ (due to Definition 1.3). Indeed, if $[\mu] \subset [\nu]$, then inequality $l^x(\mu) \leq l^x(\nu)$ holds for any vector $x \in R_+^q$. Note also that both dominance relations coincide at least when $a_i = 0$ and $b_i = \infty$ for each operation $i \in Q$ (it is easy to see that inclusion $[\mu] \subset [\nu]$ holds if and only if inequality (3.6) holds for $a_i = 0$ and $b_i = \infty,\ i \in Q$). Moreover, in this case equality $l^x(\mu) = l^x(\nu)$ is achieved only if $x_i = a_i = 0$ for any operation $i \in [\nu] \setminus [\mu]$.

Thus, we conclude that the dominance relation introduced in Definition 1.3 is a special case of the dominance relation defined by inequality (3.6) when T is equal to the whole space R_+^q (i.e., $a_i = 0$ and $b_i = \infty$ for each operation $i \in Q$). Hence, the phrase "path ν dominates path μ" is identical to the phrase "path ν dominates path μ in R_+^q".

The following lemma gives a simple criterion for the dominance relation defined by inequality (3.6) in Definition 3.3.

Lemma 3.1 *Path ν dominates path μ in set T if and only if the following inequality holds:*

$$\sum_{i\in[\mu]\setminus[\nu]} b_i \le \sum_{j\in[\nu]\setminus[\mu]} a_j. \tag{3.7}$$

PROOF. By subtracting all common variables from the left-hand side and the right-hand side of inequality (3.6) and taking into account that $a_i \le b_i$ for each operation $i \in Q$, we obtain that inequality (3.6) is equivalent to the following ones:

$$\sum_{i\in[\mu]\setminus[\nu]} x_i \le \sum_{j\in[\nu]\setminus[\mu]} x_j \quad \text{for any } x_i \quad \text{with} \quad a_i \le x_i \le b_i,\ i \in [\nu]\cup[\mu]. \tag{3.8}$$

Vector $x \in T$ satisfies inequalities (3.8) if and only if inequality (3.7) holds since we have:

$$\sum_{i\in[\mu]\setminus[\nu]} a_i \le \sum_{i\in[\mu]\setminus[\nu]} b_i \le \sum_{j\in[\nu]\setminus[\mu]} a_j \le \sum_{j\in[\nu]\setminus[\mu]} b_j.$$

◇

On the basis of the above path domination, we can introduce the following dominance relation of path sets.

Definition 3.4 *Set of paths H_k dominates set of paths H_s in T if and only if for any path $\mu \in H_s$, there exists a path $\nu \in H_k$, which dominates path μ in set T.*

The following statement gives a simple sufficient condition, when domination of sets of paths does not hold.

Lemma 3.2 *Set of paths H_k does not dominate set of path H_s in T if there exists a path $\mu \in H_s$ such that system*

$$\begin{cases} \sum_{i\in[\nu]\setminus[\mu]} a_i < \sum_{j\in[\mu]\setminus[\nu]} b_j, \\ a_i \le x_i \le b_i,\ i \in Q, \end{cases} \tag{3.9}$$

has a solution for any path $\nu \in H_k$.

PROOF. From Definition 3.4 it follows that set of paths H_k does not dominate set of paths H_s in T if there exists a path $\mu^* \in H_s$ such that there is no path $\nu \in H_k$ which dominates path μ^* in set T. This means that inequality (3.6) is violated for path $\mu^* \in H_s$ for some vector $x^0 \in T$, i.e., system

$$\begin{cases} l^x(\nu) < l^x(\mu), \\ a_i \le x_i \le b_i,\ i \in Q, \end{cases} \tag{3.10}$$

has a solution for any path $\nu \in H_k$. Furthermore, system (3.10) is consistent if and only if it has the following solution:

$$x_i = x_i^0 = \begin{cases} a_i, & \text{if } i \in [\mu^*] \setminus [\nu], \\ b_i, & \text{if } i \in [\nu] \setminus [\mu^*]. \end{cases} \tag{3.11}$$

It is easy to see that vector x according to (3.11) is a solution of system (3.10) if and only if condition (3.7) does not hold for any vertex $i \in [\nu]\cup[\mu^*]$. In other words, vector $x^0 = (x_1^0, x_2^0, \ldots, x_q^0) \in T$ and path $\mu^* \in T$ provide a solution of the equivalent system (3.9).

◇

Obviously, if $H_k = H_k(p)$, we have $H_k(p') \subseteq H_k = H_k(p)$ for any vector $p' \in R^q_+$ of the processing times. The following lemma shows that set of the critical paths is not expanded for small variations of the processing times.

Lemma 3.3 *If $H_k \neq H_k(p)$, the inclusion $H_k(p') \subseteq H_k(p)$ holds for any vector $p' \in O_\epsilon(p) \cap R^q_+$ with real number $\epsilon_k > \epsilon > 0$ defined as follows:*

$$\epsilon_k = \frac{1}{q}\Big[l^p_k - \max\{l^p(\nu) \ : \ \nu \in H_k \setminus H_k(p)\}\Big]. \tag{3.12}$$

PROOF. Since $H_k \setminus H_k(p) \neq \emptyset$, we can consider any path $\nu^\star \in H_k$ with

$$l^p(\nu^\star) = \max\{l^p(\nu) \ : \ \nu \in H_k \setminus H_k(p)\}.$$

From (3.12) it follows that $l^p_k - l^p(\nu^\star) = q \cdot \epsilon_k$, and therefore, to make the difference $l^p_k - l^p(\nu^\star)$ equal to zero, we need a vector p' with a distance from the vector p greater than or equal to ϵ_k : $d(p, p') \geq \epsilon_k$. But due to condition of Lemma 3.3, we have $d(p, p') \leq \epsilon < \epsilon_k$. Consequently, $\nu^\star \notin H_k(p')$.

Since for any path $\nu \in H_k \setminus H_k(p)$ with $l^p(\nu) < l^p(\nu^\star)$ the difference $l^p_k - l^p(\nu)$ is still greater than the product $q \cdot \epsilon_k$, such a path ν cannot belong to set $H_k(p')$.

$\diamond$

Next, we present a generalization of Theorem 1.1 with the necessary and sufficient conditions for a zero stability radius to the case of a zero relative stability radius.

Theorem 3.1 *Let digraph G_s have the minimal critical weight l^p_s, $p \in T$, within the given subset $B \subseteq \Lambda(G)$ of feasible digraphs. Then equality $\widehat{\varrho}^B_s(p \in T) = 0$ holds if and only if there exists a digraph $G_k \in B$ such that $l^p_s = l^p_k$, $k \neq s$, and set of paths $H_k(p)$ does not dominate set of paths $H_s(p)$ in T.*

PROOF. *Sufficiency (if).* Let conditions of Theorem 3.1 hold: There exists a digraph $G_k \in B$ such that $l^p_s = l^p_k$, $k \neq s$, and $H_k(p)$ does not dominate set $H_s(p)$ in T. We show that $\widehat{\varrho}^B_s(p \in T) < \epsilon$ for any given $\epsilon > 0$ which may be as small as desired.

Since set $H_k(p)$ does not dominate set $H_s(p)$ in T, there exists a path $\mu^* \in H_s(p)$ such that no path $\nu \in H_k(p)$ dominates path μ^* in set T, i.e., system (3.10) has a solution for any path $\nu \in H_k(p)$. First, we make the following remark.

Remark 3.1 From the consistency of system (3.10), it follows that for the considered problem $\mathcal{G}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$, the trivial case with $a_i = b_i$ for each $i \in Q$ does not hold. Indeed, in this case the first inequality in (3.10) is transformed into inequality $l^p(\nu) < l^p(\mu^*)$ which is certainly wrong: $l^p(\nu) = l^p_k = l^p_s = l^p(\mu^*)$.

We construct a vector $p' = (p'_1, p'_2, \ldots, p'_q)$ with the following components:

$$p'_i = \begin{cases} p_i + \epsilon', & \text{if } i \in [\mu^*],\ p_i \neq b_i, \\ p_i - \epsilon', & \text{if } i \in \{\cup_{\nu \in H_k(p)}[\nu]\} \setminus [\mu^*],\ p_i \neq a_i, \\ p_i, & \text{otherwise,} \end{cases} \tag{3.13}$$

where ϵ' is chosen as a strictly positive real number less than both value ϵ and value

$$\epsilon_{min} = \max\{0,\ \min\{\min\{p_i - a_i :\ p_i > a_i, i \in Q\},\ \min\{b_i - p_i :\ b_i > p_i, i \in Q\}\}\}.$$

We can also choose ϵ' less than $\epsilon_k > 0$ defined in (3.12). More precisely, if $H_k \neq H_k(p)$, then $\epsilon_k > 0$, and we can choose ϵ' such that $0 < \epsilon' < \min\{\epsilon, \epsilon_k, \epsilon_{min}\}$. Otherwise, if $H_k = H_k(p)$, we choose ϵ' such that $0 < \epsilon' < \min\{\epsilon, \epsilon_{min}\}$. Such choices are possible since in both cases, inequality $\epsilon_{min} > 0$ holds due to Remark 3.1.

The following arguments are the same for both cases of the choice of ϵ' except the 'last step' since $H_k \setminus H_k(p) = \emptyset$ in the latter case.

Since system (3.10) has a solution for each path $\nu \in H_k$, the first inequality in (3.10)

$$l^x(\nu) < l^x(\mu^*)$$

has a solution for $x \in T$ which implies that inclusion $[\mu^*] \subset [\nu]$ does not hold for any path $\nu \in H_k(p)$. Therefore, from the equalities $l^p(\nu) = l_k^p = l_s^p = l^p(\mu^*)$ and (3.13), we can conclude that vector p' is a solution of system (3.10) for each path $\nu \in H_k(p)$. In other words, vector p' is a solution of the following system of inequalities:

$$\begin{cases} l^x(\nu) < l^x(\mu^*),\ \nu \in H_k(p), \\ a_i \leq x_i \leq b_i,\ i \in Q. \end{cases}$$

Thus, we obtain inequality $l^{p'}(\nu) < l^{p'}(\mu^*)$ for each path $\nu \in H_k(p)$, and therefore

$$\max\{l^{p'}(\nu)\ :\ \nu \in H_k(p)\} < l^{p'}(\mu^*). \tag{3.14}$$

The 'last step' in the proof of sufficiency is as follows. Since $p' \in O_{\epsilon'}(p) \cap R_+^q$ with $0 < \epsilon' < \epsilon_k$, due to Lemma 3.3 we obtain $H_k(p') \subseteq H_k(p)$ and, as a result,

$$l^{p'}(\tau) < l_k^{p'} = \max\{l^{p'}(\nu)\ :\ \nu \in H_k(p)\} \tag{3.15}$$

for each path $\tau \in H_k \setminus H_k(p)$.

From inequalities (3.14) and (3.15), it follows that $l_k^{p'} < l_s^{p'}$. Taking into account that $d(p', p) = \epsilon' < \epsilon$, we conclude that $\widehat{\varrho}_s^B(p \in T) < \epsilon$.

Necessary (only if). We prove necessity by contradiction. Let us suppose that $\widehat{\varrho}_s^B(p \in T) = 0$ but condition of Theorem 3.1 does not hold. The following two cases *(i)* and *(ii)* of violating condition of Theorem 3.1 may hold.

(i) There does not exist a digraph $G_k \in B$ such that $l_s^p = l_k^p,\ k \neq s$.

In the trivial case when $B = \{G_s\}$, we have $\widehat{\varrho}_s^B(p \in T) = \infty$ due to Definition 3.2.

Let $B \setminus \{G_s\} \neq \emptyset$. Then we calculate the following real number:

$$\epsilon^* = \frac{1}{q} \min\{l_t^p - l_s^p\ :\ G_t \in B,\ t \neq s\} \tag{3.16}$$

which is strictly positive since inequality $l_s^p < l_t^p$ holds for each digraph $G_t \in B,\ t \neq s$. Arguing in a similar way as in the proof of Lemma 3.3, we can show that the difference $l_t^p - l_s^p$ cannot become negative when vector p is replaced by an arbitrary vector $p^0 \in O_{\epsilon^*}(p) \cap R_+^q$. Next, we show that the difference $l_t^p - l_s^p$ cannot become negative when vector p is replaced by an arbitrary vector $p^0 \in O_{\epsilon^*}(p) \cap T \subseteq R_+^q$ with $0 < \epsilon^* < \epsilon_k$.

Since $H_k \backslash H_k(p) \neq \emptyset$, we can consider any path $\nu^\star \in H_k$ with

$$l^p(\nu^\star) = \max\{l^p(\nu) \ : \ \nu \in H_k \backslash H_k(p)\}.$$

From (3.16) it follows that $l_k^{p^0} - l_s^{p^0} \geq q \cdot \epsilon^*$, and therefore, to make the difference $l_k^{p^0} - l_s^{p^0}$ equal to zero, we need a vector p' with a distance from the vector p greater than or equal to $\epsilon_k : \ d(p^0, p') \geq \epsilon_k$. However, due to conditions of Lemma 3.3, we have $d(p, p') \leq \epsilon^* < \epsilon_k$. Consequently, $\nu^\star \notin H_k(p')$.

Since for any digraph $G_t \in B$, with $l^{p^0}(\nu) < l^{p^0}(\nu^\star)$ the difference $l_k^{p^0} - l_s^{p^0}$ is still greater than the product $q \cdot \epsilon^*$, such a path ν cannot belong to the set $H_k(p')$. So we conclude that digraph G_s remains 'the best' (perhaps one of the 'best') within the set B for any vector p^0 of the processing times. Due to Definition 3.2, we have $\widehat{\varrho}_s^B(p \in T) \geq \epsilon^* > 0$ which contradicts the above assumption of $\widehat{\varrho}_s^B(p \in T) = 0$.

(ii) There exists a digraph $G_k \in B$ such that $l_s^p = l_k^p, \ k \neq s$, and for any such digraph G_k, set of paths $H_k(p)$ dominates set of paths $H_s(p)$ in T.

In this case, we can take any ϵ that satisfies the following inequalities:

$$0 < \epsilon < \min\Big\{ \min\{\epsilon_k \ : \ l_k^p = l_s^p, \ G_k \in B\}, \ \frac{1}{q} \min\{l_t^p - l_s^p \ : \ l_t^p > l_s^p, \ G_t \in B\}\Big\}.$$

Due to inequality $\epsilon > \epsilon_s$, we get from Lemma 3.3 that equalities

$$l_s^{p^0} = \max_{\mu \in H_s(p^0)} l^{p^0}(\mu) = \max_{\mu \in H_s(p)} l^{p^0}(\mu) \tag{3.17}$$

hold for any vector $p^0 \in O_\epsilon(p) \cap R_+^q$. The statement that for any digraph $G_k \in B, k \neq s$, with $l_s^p = l_k^p$ set of paths $H_k(p)$ dominates set of paths $H_s(p)$ in T means that for any path $\mu \in H_s(p)$, there exists a path $\nu^* \in H_k(p)$ such that system

$$\begin{cases} l^x(\nu^*) < l^x(\mu), \\ a_i \leq x_i \leq b_i, \ i \in Q, \end{cases}$$

has no solution. Therefore, inequality

$$l^x(\mu) \leq l^x(\nu^*) \tag{3.18}$$

holds for any vector $x \in T$. Due to inequality (3.18) and taking into account that $\epsilon < \epsilon_k$ and $\epsilon < \epsilon_s$, we obtain the following inequality using Lemma 3.3:

$$\max_{\mu \in H_s(p)} l^{p^0}(\mu) \leq \max_{\nu \in H_k(p)} l^{p^0}(\nu). \tag{3.19}$$

Thus, due to (3.17) and (3.19), we obtain inequality

$$l_s^{p^0} \leq \max_{\nu \in H_k(p)} l^{p^0}(\nu) \tag{3.20}$$

for any digraph $G_k \in B, \ l_s^p = l_k^p, \ k \neq s$. Since

$$\epsilon < \frac{1}{q} \min\{l_t^p - l_s^p : \ l_t^p > l_s^p, \ G_t \in B\},$$

inequality $l_t^p > l_s^p$ implies inequality $l_t^{p^0} > l_s^{p^0}$. Taking into account (3.20), we conclude that $l_s^{p^0} \leq l_k^{p^0}$ for any digraph $G_k \in B$ and for any vector $p^0 \in T$ with $d(p, p^0) \leq \epsilon$. Consequently, $\widehat{\varrho}_s^B(p \in T) \geq \epsilon > 0$, which contradicts the assumption of $\widehat{\varrho}_s^B(p \in T) = 0$.

$\diamond$

Theorem 3.1 directly implies the following statement.

Corollary 3.1 *If $G_s \in B$ is unique optimal digraph for vector $p \in T$ of the processing times, then $\widehat{\varrho}_s^B(p \in T) > 0$.*

From Theorem 3.1 we obtain the following lower bound for relative stability radius.

Corollary 3.2 *If $G_s \in B$ and $l_s^p = \min\{l_k^p \ : \ G_k \in B\}$, then $\widehat{\varrho}_s^B(p \in T) \geq \epsilon^*$ with ϵ^* calculated according to (3.16).*

PROOF. If there exists a digraph $G_k \in B$ such that $l_s^p = l_k^p,\ k \neq s$, the equality $\widehat{\varrho}_s^B(p \in T) \geq \epsilon^* = 0$ holds due to Definition 3.2. Otherwise, inequality $\widehat{\varrho}_s^B(p \in T) \geq \epsilon^*$ follows from the above proof of necessity of Theorem 3.1 (see case *(i)*).

$\diamond$

Theorem 3.1 identifies a digraph $G_s \in \Lambda(G)$ whose 'superiority' within the set B is *unstable*: Even a very small change in the processing times can make another digraph from the set B to be 'better' than G_s. The following theorem identifies a digraph G_s whose 'superiority' within the set B in the polytope T is 'absolute': Any changes of the processing times within the polytope T cannot make another digraph from the set B to be 'better' than digraph G_s.

Theorem 3.2 *For digraph $G_s \in B$, we have $\widehat{\varrho}_s^B(p \in T) = \infty$ if and only if for any digraph $G_t \in B,\ t \neq s$, set of paths H_t dominates set of paths $H_s \setminus H$ in T.*

PROOF. *Sufficiency.* Let ϱ be any positive number (as large as desired). We take any vector $p \in O_\varrho(p) \cap T \subseteq R_+^q$ and consider a path $\mu \in H_s$ such that $l_s^p = l^p(\mu)$.

(j) If $\mu \in H$, then inequality $l_s^p = l^p(\mu) \leq l_t^p$ holds for any digraph $G_t \in \Lambda(G)$.

(jj) If $\mu \in H_s \setminus H$, then due to condition of Theorem 3.2, it follows that for any digraph $G_t \in B, t \neq s$, there exists a path $\nu^* \in H_t$ such that inequality

$$l^x(\mu) \leq l^x(\nu^*)$$

holds for any vector $x \in T$ (and for the vector p as well). Therefore, we have $l_s^p = l^p(\mu) < l^p(\nu^*) \leq l_t^p$. Thus, in both above cases *(j)* and *(jj)* we have $l_s^p = \min\{l_t^p \ : \ G_t \in B\}$.

Necessity. We prove necessity by contradiction. Let us suppose that $\widehat{\varrho}_s^B(p \in T) = \infty$, but there exists a digraph $G_t \in B, t \neq s$, such that set of paths H_t does not dominate set of paths $H_s \setminus H$ in T. Thus, there exists a path $\mu^0 \in H_s \setminus\ H$ such that for any path $\nu \in H_t$, system of inequalities

$$\begin{cases} l^x(\nu) < l^x(\mu^0), \\ a_i \leq x_i \leq b_i,\ i \in Q, \end{cases} \tag{3.21}$$

has a solution. Therefore, due to Lemma 3.2, inequality

$$\sum_{i \in [\nu] \setminus [\mu^0]} a_i < \sum_{j \in [\mu^0] \setminus [\nu]} b_j \tag{3.22}$$

holds. We consider the vector $p^* = (p_1^*, p_2^*, \ldots, p_q^*) \in T$ with

$$p_i^* = \begin{cases} a_i, & \text{if } i \in \{\bigcup_{[\nu]\in H_t}[\nu]\} \ \setminus \ [\mu^0], \\ b_i, & \text{if } i \in [\mu^0], \\ p_i & \text{otherwise.} \end{cases}$$

Adding to the left-hand side and to the right-hand side of (3.22) the value $\sum_{j\in[\nu]\cap[\mu^0]} b_j$, we obtain that inequality

$$\sum_{i\in[\nu]\setminus[\mu^0]} a_i + \sum_{j\in[\nu]\cap[\mu^0]} b_j < \sum_{j\in[\mu^0]} b_j$$

holds. Thus, we can conclude that vector p^* is a solution of the system of linear inequalities obtained by joining systems (3.21) for all paths $\nu \in H_t$, i.e., we obtain

$$\begin{cases} l^{p^*}(\nu) < l^{p^*}(\mu^0), \ \nu \in H_t, \\ a_i \le x_i \le b_i, \ i \in Q. \end{cases}$$

Therefore, $l_t^{p^*} < l^{p^*}(\mu^0) \le l_s^{p^*}$, and hence, we get a contradiction to the above assumption: $\widehat{\varrho}_s^B(p \in T) < d(p^*, p) \le \epsilon_{max} < \infty$.

◇

From Theorem 3.2 we obtain the following upper bound for the relative stability radius.

Corollary 3.3 *If $\widehat{\varrho}_s^B(p \in T) < \infty$, then $\widehat{\varrho}_s^B(p \in T) \le \epsilon_{max}$, where value ϵ_{max} is calculated according to (3.5).*

Proof. The desired bound immediately follows from the proof of necessity in Theorem 3.2.

◇

In the following section, we use Theorem 3.2 as a stopping rule in the algorithm developed for solving problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$ since the optimality of digraph $G_s \in B$ with $\widehat{\varrho}_s^B(p \in T) = \infty$ does not depend on the vector $p \in T$ of the processing times.

3.3 Algorithms for Problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$

From Sections 3.1 and 3.2, it follows that problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$ may be solved on the basis of a repeated calculation of the relative stability radii $\widehat{\varrho}_s^B(p \in T)$.

In Section 1.3, the formulas (1.28) and (1.29) were proven for calculating stability radius $\widehat{\varrho}_s(p) = \widehat{\varrho}_s^{\Lambda(G)}(p \in R_+^q)$ (see Theorem 1.4 at page 29). Theorem 3.3, which follows, generalizes these formulas for any given subset $B \subseteq \Lambda(G)$ of feasible digraphs and for any given polytope $T \subseteq R_+^q$ of feasible vectors of the processing times. To present the new formulas, we need the following notations.

Let μ and ν be paths in the digraphs from the set $\Lambda(G)$. Then $[\mu] + [\nu]$ denotes the 'symmetric difference' $[\mu] \cup [\nu] \ \setminus \ [\mu] \cap [\nu]$ of the sets $[\mu]$ and $[\nu]$.

We calculate the following values:

$$\Delta^i(\mu,\nu) = \begin{cases} b_i - p_i, & \text{if } \ i \in [\mu] \setminus [\nu], \\ p_i - a_i, & \text{if } \ i \in [\nu] \setminus [\mu]. \end{cases} \tag{3.23}$$

Let $\Delta_0^i(\mu,\nu)$ be equal to zero. We order the set of values $\Delta^i(\mu,\nu)$ for all operations i from the set $[\mu]+[\nu]$ in the following way:

$$\Delta_1^{i_1}(\mu,\nu) \le \Delta_2^{i_2}(\mu,\nu) \le \ldots \le \Delta_{|[\mu]+[\nu]|}^{i_{|[\mu]+[\nu]|}}(\mu,\nu), \tag{3.24}$$

where the subscript $j \in \{1,2,\ldots,[\mu]+[\nu]\}$ indicates the location of $\Delta^i(\mu,\nu)$ in the above sequence (3.24), and the superscript i_j denotes operation $i_j \in [\mu]+[\nu]$ for which the value $\Delta_j^{i_j}(\mu,\nu)$ was calculated. For simplicity, we will substitute superscript i_j by one index i (which means operation $i = i_j$). We hope that it will not cause a misunderstanding.

For any two feasible digraphs G_s and G_k, we denote

$$H_{sk}(T) = \Big\{\mu \in H_s : \text{ There is no path } \nu \in H_k \text{ which dominates path } \mu \text{ in set } T\Big\}.$$

Theorem 3.3 *If digraph G_s has the minimal critical weight l_s^p, $p \in T$, in the given set $B \subseteq \Lambda(G)$ of feasible digraphs, then*

$$\widehat{\varrho}_s^B(p \in T) = \min_{G_k \in B} \widehat{r}_{ks}^B, \tag{3.25}$$

where

$$\widehat{r}_{ks}^B = \min_{\mu \in H_{sk}(T)} \ \max_{\nu \in H_k,\ l^p(\nu) \ge l_s^p} \ \max_{\beta=0,1,\ldots,|[\mu]+[\nu]|-1} \frac{l^p(\nu) - l^p(\mu) - \sum_{\alpha=0}^{\beta} \Delta_\alpha^i(\mu,\nu)}{|[\mu]+[\nu]| - \beta}. \tag{3.26}$$

Proof. From Definition 3.2 it follows:

$$\widehat{\varrho}_s^B(p \in T) = \inf\{d(p,x) : \ x \in T,\ l_s^x > \min\{l_k^x : \ G_k \in B\}\}.$$

Therefore, to find the relative stability radius $\widehat{\varrho}_s^B(p \in T)$, it is sufficient to construct a vector $x \in T$ which satisfies the following three conditions.
(1) There exists a digraph $G_k(p) \in B$, $k \ne s$, such that $l_s^x = l_k^x$, i.e.,

$$\max_{\mu \in H_s} l^x(\mu) = \max_{\nu \in H_k} l^x(\nu). \tag{3.27}$$

(2) For any given real number $\epsilon > 0$, which may be as small as desired, there exists a vector $p^\epsilon \in T$ such that $d(x,p^\epsilon) = \epsilon$ and $l_s^{p^\epsilon} > l_k^{p^\epsilon}$, i.e., inequality

$$\max_{\mu \in H_s} l^{p^\epsilon}(\mu) > \max_{\nu \in H_k} l^{p^\epsilon}(\nu) \tag{3.28}$$

is satisfied for at least one digraph $G_k(p) \in B$.
(3) The distance $d(p,x)$ achieves the minimal value among the distances between the vector p and the other vectors in the given polytope T which satisfy both above conditions (1) and (2).

After having constructed such a vector $x \in T$, one can define the relative stability radius of the digraph G_s as follows:

$$\widehat{\varrho}_s^B(p \in T) = d(p, x).$$

Indeed, the critical path of digraph G_s becomes larger than that of digraph G_k for any vector $p^\epsilon \in T$ with positive real ϵ, which may be as small as desired (see condition (2)). Therefore, digraph G_s has no longer the minimal critical weight among all other feasible digraphs, while in the ball $O_{d(p,x)}(p \in T)$ digraph G_s has the minimal critical weight (see condition (3)). Digraph G_k satisfying conditions (1) and (2) is called a *competitive* digraph for digraph G_s.

To satisfy conditions (1), (2) and (3) (except the inclusion $x \in T$), we first search for a vector $x = p(r) = (p_1(r), p_2(r), \ldots, p_q(r)) \in R^q$ with the components $p_i(r) \in \{p_i, p_i + r, p_i - r\}$ on the basis of a direct comparison of the paths from set H_s and the paths from sets H_k, where $G_k \in B$.

Let the value $l^p(\nu)$ be greater than the weight of a critical path in optimal digraph G_s. To satisfy equality (3.27), the weight of a path $\nu \in H_k$ must be smaller than or equal to the weight of at least one path $\mu \in H_s$, and there must exist a path $\nu \in H_k$ with a weight equal to the weight of a critical path of G_s. Thus, if we have calculated

$$r_\nu = \min_{\mu \in H_s} \frac{l^p(\nu) - l^p(\mu)}{|[\mu] + [\nu]|}, \tag{3.29}$$

we obtain equality

$$\max_{\mu \in H_s} l^{p(r)}(\mu) = l^{p(r)}(\nu) \tag{3.30}$$

for vector $p(r) = p(r_\nu)$ with components

$$p_i(r) = p_i(r_\nu) = \begin{cases} p_i + r_\nu, & \text{if } i \in [\mu], \\ p_i - r_\nu, & \text{if } i \in [\nu] \backslash [\mu], \\ p_i, & \text{if } i \notin [\mu] + [\nu]. \end{cases} \tag{3.31}$$

We can make the following remark.

Remark 3.2 Due to (3.29), the vector $p(r)$ calculated in (3.31) is the closest one to the given vector p among all vectors x for which equality (3.30) with $p(r) = x$ holds. Indeed, to make the difference $l^x(\nu) - \max_{\mu \in H_s} l^x(\mu)$ equal to zero, one needs a q-dimensional vector x with a distance from the vector p greater than or equal to r_ν : $d(p, x) \ge r_\nu$.

To reach equality (3.27) for digraph G_k, we have to repeat calculation (3.29) for each path $\nu \in H_k$ with $l^p(\nu) \ge l_s^p$. Thus, instead of vector $p(r_\nu)$, we have to consider vector $p(r) = p(r_{G_k})$ calculated according to formula (3.31), where

$$r_{G_k} = \min_{\mu \in H_s} \max_{\nu \in H_k;\ l^p(\nu) \ge l_s^p} \frac{l^p(\nu) - l^p(\mu)}{|[\mu] + [\nu]|}. \tag{3.32}$$

Next, we consider inequality (3.28). Since vectors of the processing times have to belong to polytope $T \subseteq R_+^q$, this inequality may not be valid for a vector $p^\epsilon \in T$ if path ν dominates path μ in set T. Thus, we can restrict our consideration to the subset $H_{sk}(T)$

of the set H_s of all paths, which are not dominated in T by paths from set H_k and for which there does not exist a path $\nu \in H_k$ such that $[\nu] = [\mu]$. Hence, we can replace H_s in equality (3.32) by $H_{sk}(T)$.

To obtain the desired vector $x \in R^q$, we have to use equality (3.32) for each digraph $G_k \in \Lambda(G), k \neq s$. Let r denote the minimum of such a value r_{G_k}:

$$r = r_{G_{k^*}} = \min\{r_{G_k} : G_k,\ k \neq s\}$$

and let $\nu^* \in H_{k^*}$ and $\mu^* \in H_{sk^*}$ be paths at which value $r_{G_{k^*}}$ has been reached:

$$r_{G_{k^*}} = r_{\nu^*} = \frac{l^p(\nu^*) - l^p(\mu^*)}{|[\mu] + [\nu]|}.$$

Taking into account (3.32), we note that, if

$$r_{\nu^*} \leq \Delta = \min\{b_i - p_i :\ i \in [\nu^*] \setminus [\mu^*]\}$$

and

$$r_{\nu^*} \leq \Delta' = \min\{p_i - a_i :\ i \in [\nu^*] \setminus [\mu^*]\},$$

then vector $p(r)$ belongs to polytope T.

Due to Remark 3.2, we have obtained a lower bound for the stability radius:

$$\widehat{\varrho}_s^B(p \in T) \geq r \ = \ \min_{G_k \in B} \ \min_{\mu \in H_{sk}(T)} \ \max_{\nu \in H_k;\ l^p(\nu) \geq l_s^p} \ \frac{l^p(\nu) - l^p(\mu)}{|[\mu] + [\nu]|}. \tag{3.33}$$

The bound (3.33) is tight: If $\widehat{\varrho}_s^B(p \in T) \leq \min\{\Delta^i(\mu^*, \nu^*) \ : \ i \in [\mu^*] \cup [\nu^*]\}$, then $\widehat{\varrho}_s^B(p \in T) = r$. In particular, we have $\widehat{\varrho}_s^B(p \in T) = r$ in (3.33) if $\widehat{\varrho}_s^B(p \in T) \leq \epsilon_{min}$.

To obtain the exact value of relative stability radius $\widehat{\varrho}_s^B(p \in T)$ in the general case, we can use vector $x = p^*(r) = (p_1^*(r), p_2^*(r), ..., p_q^*(r))$ with components

$$p_i^*(r) = \begin{cases} p_i + \min\{r, b_i - p_i\}, & \text{if } i \in [\mu], \\ p_i - \min\{r, p_i - a_i\}, & \text{if } i \in [\nu] \setminus [\mu], \\ p_i, & \text{if } i \notin [\mu] + [\nu], \end{cases} \tag{3.34}$$

instead of the vector $p(r)$ defined in (3.31). As it follows from Remark 3.2, such a vector $p^*(r) \in T$ is the closest one to the vector p among all vectors $x \in T$ which satisfy both conditions (1) and (2).

For calculating the maximal value r for the vector $p^*(r)$, we can consider each operation i from the set $[\mu] \cup [\nu]$ one by one in non-decreasing order (3.24) of the values $\Delta^i(\mu, \nu)$ defined in (3.23). As a result, formula (3.33) will be transformed into the formulas given in Theorem 3.3.

$\diamond$

Remark 3.3 Note that the formulas (3.25) and (3.26) defined in Theorem 3.3 turn into $\widehat{\varrho}_s^B(p \in T) = \infty$ if $H_{sk}(T) = \emptyset$ for each digraph $G_k \in B$.

Table 3.3: *Numerical data for problem* $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\mathcal{C}_{max}$

i	1	1	1	2	2	2
j	1	2	3	1	2	3
a_{ij}	35	40	20	50	45	20
b_{ij}	100	90	110	80	80	40

Example 3.1 (continued). *Returning to the Example 3.1 given in Section 3.1, let us consider problem* $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\mathcal{C}_{max}$ *whose input data are given by the weighted mixed graph* $G(p)$ *in Figure 3.1 together with vector* $a=(a_{1,1}, a_{1,2}, \ldots, a_{2,3})$ *and vector* $b=(b_{1,1}, b_{1,2}, \ldots, b_{2,3})$ *of lower and upper bounds for the possible variations of the processing times* p*, where* $a=(35,40,20,50,45,20)$ *and* $b=(100,90,110,80,80,40)$*. The numerical input data for this instance of problem* $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\mathcal{C}_{max}$ *are given in Table 3.3.*

Since the mixed graph G *is the same for the above problem* $\mathcal{J}3/n=2/\mathcal{C}_{max}$ *considered in Section 3.1 and for the new problem* $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\mathcal{C}_{max}$*, we have the same set* $\Lambda(G)$ *of feasible digraphs. Moreover, if we start calculation with the same initial vector* $p=(75,50,40,60,55,30)$ *of the processing times, we obtain the same optimal digraph* G_1*, presented in Figure 3.2 with the dummy operations* $O_{0,1}$ *and* $O_{0,2}$*. Using Theorem 3.3, we can calculate the relative stability radius of this digraph:* $\widehat{\varrho}_1^{\Lambda(G)}(p \in T) = 60$*, where the polytope* $T \in R_+^6$ *is defined by the above vectors* a *and* b *(see Table 3.3). Note that, due to these bounds* a_{ij} *and* b_{ij} *for the possible variations of the processing times* p_{ij}*,* $O_{ij} \in Q^J = \{O_{1,1}, O_{1,2}, \ldots, O_{2,3}\}$*, the stability radius of the digraph* G_1 *increased from 30 to 60 (remind that in Section 3.1, we calculated* $\widehat{\varrho}_1^{\Lambda(G)}(p \in R_+^6) = \varrho_1(p) = 30$*).*

In Table 3.4, one can observe the calculation of $\widehat{\varrho}_1^{\Lambda(G)}(p \in T)$ *in detail. The set* $H_{1k}(T)$ *is empty for each digraph* $G_k, k \in \{2,4,5\}$*. Note that* $H_{sk}(T) \subseteq H_{sk}$*, therefore we have* $H_{1,2}(T) = \emptyset$ *and* $H_{1,4}(T) = \emptyset$*. Moreover, for both paths* $\mu_1 = (O_{1,1}, O_{1,2}, O_{2,3}) \in H_{1,5}$ *and* $\mu_2 = (O_{1,1}, O_{2,2}, O_{2,3}) \in H_{1,5}$*, there exists a path* $\nu_2 = (O_{2,1}, O_{2,2}, O_{1,1}, O_{1,2}, O_{1,3}) \in H_5$ *which dominates both paths* μ_1 *and* μ_2 *in* T*, i.e., inequality (3.7) holds.*

Table 3.4 has an analogous design as Table 3.2 from Section 3.1 with the exception of column 5, which contains the values $\Delta_\beta^{ij}(\mu,\nu)$*,* $\beta = 0,1,\ldots,|[\mu]+[\nu]|-1$*, defined by formula (3.23) at page 126 in non-decreasing order defined by sequence (3.24). Let us consider path* $\mu = (O_{2,1}, O_{1,2}, O_{1,3}) \in H_{1,3}(T)$ *and path* $\nu_1 = (O_{1,1}, O_{1,2}, O_{1,3}) \in H_3$*. For each vertex from the set* $[\mu]+[\nu_1]$ *(set* $[\mu]+[\nu_1]$ *means the symmetric difference of sets* $[\mu]$ *and* $[\nu_1]$*),* $|[\mu]+[\nu_1]| = 2$*, we calculate the values* $\Delta^{1,1}(\mu,\nu_1) = p_{1,1} - a_{1,1} = 75-35 = 40$*,* $\Delta^{2,1}(\mu,\nu_1) = b_{2,1} - p_{2,1} = 80-60 = 20$*. By a comparison of the path* μ *with the path* $\nu_2 = (O_{1,1}, O_{1,2}, O_{2,1}, O_{2,2}, O_{2,3}) \in H_3$*, we find the values* $\Delta^{1,1}(\mu,\nu_2) = p_{1,1} - a_{1,1} = 75-35 = 40$*,* $\Delta^{1,3}(\mu,\nu_2) = b_{1,3} - p_{1,3} = 110-40 = 70$*,* $\Delta^{2,2}(\mu,\nu_2) = p_{2,2} - a_{2,2} = 55-45 = 10$*,* $\Delta^{2,3}(\mu,\nu_2) = p_{2,3} - a_{2,3} = 30-20 = 10$*. The sequential calculations of the fraction from the formula (3.26) are represented in column 6 of Table 3.4. Column 9 (see Table 3.2) is redundant for this small example.*

So, one of the two competitive digraphs, namely digraph G_3 *(see Figure 3.3 at page 116), remains also a competitive digraph of* G_1 *for problem* $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\mathcal{C}_{max}$*. However, the new vector of the processing times* $p^* = p^{(2)}$ *has to be calculated due*

Table 3.4: *Calculation of the relative stability radius* $\widehat{\varrho}_1^{\Lambda(G)}(p \in T)$ *for problem* $\mathcal{J}3/n = 2, a_i \le p_i \le b_i/\mathcal{C}_{max}$

G_k	$\lvert H_{1k}(T)\rvert$	$\mu \in H_{1k}(T)$ $l^p(\mu)$	$\nu \in H_k$ $l^p(\nu) \ge l_1^p$	$\Delta_\beta^{ij}(\mu,\nu)$ $0 \le \beta \le \lvert[\mu]+[\nu]\rvert - 1$	$\frac{l^p(\nu) - l^p(\mu) - \sum_{\alpha=0}^{\beta} \Delta_\alpha^{ij}(\mu,\nu)}{\lvert[\mu]+[\nu]\rvert - \beta}$	$\max\limits_{\beta}$	$\max\limits_{\nu}$
1	2	3	4	5	6	7	8
G_2	0						
G_3	1	$(O_{2,1}, O_{1,2}, O_{1,3})$	$(O_{1,1}, O_{1,2}, O_{1,3})$ $l^p(\nu_1) = 165 = l_1^p$	$\Delta_0^{ij}(\mu,\nu_1) = 0$ $\Delta_1^{2,1}(\mu,\nu_1) = 20$	$\frac{165-150-0}{2-0} = 7.5$ $\frac{165-150-20}{2-1} < 0$	7.5	60
		$l^p(\mu) = 150$	$(O_{1,1}, O_{1,2}, O_{2,1}, O_{2,2}, O_{2,3})$ $l^p(\nu_2) = 270 > 165$	$\Delta_0^{ij}(\mu,\nu_2) = 0$ $\Delta_1^{2,2}(\mu,\nu_2) = 10$ $\Delta_2^{2,3}(\mu,\nu_2) = 10$ $\Delta_3^{1,1}(\mu,\nu_2) = 40$	$\frac{270-150-0}{4-0} = 30$ $\frac{270-150-10}{4-1} = 36\frac{2}{3}$ $\frac{270-150-(10+10)}{4-2} = 50$ $\frac{270-150-(10+10+40)}{4-3} = 60$	60	
G_4	0						
G_5	0						

to formula (3.34) with $r = \widehat{\varrho}_1^{\Lambda(G)}(p \in T) = 60$, $\mu = (O_{2,1}, O_{1,2}, O_{1,3}) \in H_1$ *and* $\nu = (O_{1,1}, O_{1,2}, O_{2,1}, O_{2,2}, O_{2,3}) \in H_{1,3}(T) \subseteq H_3$. *Thus, vector* p^* *is as follows:* $p^* = p^{(2)} = (35, 90, 100, 80, 45, 20)$.

Next, we follow the scheme proposed at pages 114–117 for obtaining a solution of problem $\mathcal{J}3/n = 2, a_i \le p_i \le b_i/\mathcal{C}_{max}$. *We calculate* $\widehat{\varrho}_3^{\Lambda(G)\setminus\{G_1\}}(p^{(2)} \in T) = 32.5$ *on the basis of Theorem 3.3 and obtain the competitive digraph* G_2 *of digraph* G_3. *For digraph* G_2, *the minimum in (3.25) is reached on the set* $B = \Lambda(G) \setminus \{G_1\}$, *and thus digraph* G_2 *becomes optimal at least for one point* $p^{(3)}$ *of the stability sphere (the boundary of the stability ball* $O_{32.5}(p^{(2)})$*). Then we calculate stability radius* $\widehat{\varrho}_2^{\Lambda(G)\setminus\{G_1,G_3\}}(p^{(3)} \in T) = 27.5$ *for the new optimal digraph* G_2 *and for the new set* $B := B \setminus \{G_3\} = \Lambda(G) \setminus \{G_1, G_3\}$. *Then on the basis of Theorem 3.2 (or Theorem 3.3), we obtain* $\widehat{\varrho}_4^{\Lambda(G)\setminus\{G_1,G_2,G_3\}}(p^{(4)} \in T) = \infty$.

Thus, solving problem $\mathcal{J}3/n = 2, a_i \le p_i \le b_i/\mathcal{C}_{max}$ *takes four iterations presented in Table 3.5. We obtain that set of digraphs* $\{G_1, G_2, G_3, G_4\}$ *is a solution of the problem* $\mathcal{J}3/n = 2, a_i \le p_i \le b_i/\mathcal{C}_{max}$ *under consideration:* $\Lambda^*(G) = \{G_1, G_2, G_3, G_4\}$. *In other words, the given polytope* T *is covered by the union of the stability balls of four digraphs from set* $\Lambda^*(G)$.

The projections of these stability balls on the plane for the component $p_{1,3}$ *of the vector* p *given at the axis of x-coordinates and for the component* $p_{2,2}$ *of the vector* p *given at the axis of y-coordinates are drawn in Figure 3.4. The last stability ball has infinite radius* $\widehat{\varrho}_4(p^{(4)} \in T) = \infty$, *i.e., it coincides with the whole space* R^6: $O_\infty(p^{(4)}) = R^6$. *Thus, stability ball* $O_\infty(p^{(4)})$ *covers the given polytope* T *and all other stability balls. (The stability ball* $O_\infty(p^{(4)})$ *is not shown in Figure 3.4.) The competitive digraph, for which the minimum in (1.28) is reached, becomes optimal at least for one point of the stability sphere*

Table 3.5: *Solution of the problem* $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\mathcal{C}_{max}$ *by Algorithm* $SOL\_\mathcal{C}_{max}(1)$

i	Center $p^{(i)} \in T$ of the stability ball	Set B of feasible digraphs	Optimal digraph G_s	$\widehat{\varrho}_s^B(p^{(i)} \in T)$	Competitive digraph of G_s
1	$(75, 50, 40, 60, 55, 30)$	$\Lambda(G)$	G_1	60	G_3
2	$(35, 90, 100, 80, 45, 20)$	$\Lambda(G) \setminus \{G_1\}$	G_3	32.5	G_2
3	(67.5, 90, 67.5, 80, 77.5, 40)	$\Lambda(G) \setminus \{G_1, G_3\}$	G_2	27.5	G_4
4	$(40, 90, 95, 80, 80, 40)$	$\Lambda(G) \setminus \{G_1, G_2, G_3\}$	G_4	∞	$-$

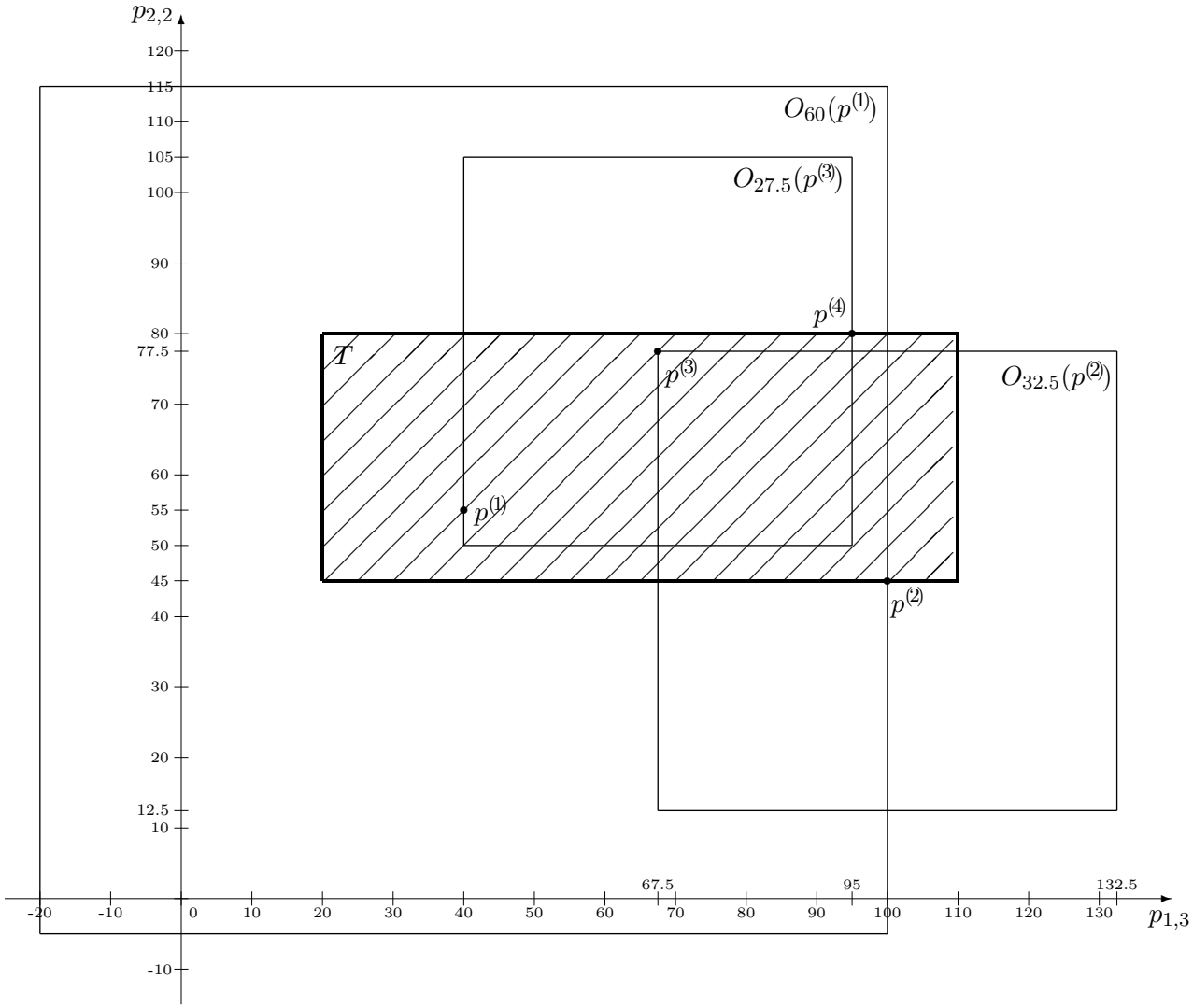

Figure 3.4: *Projections of the stability balls on the plane for problem* $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\mathcal{C}_{max}$ *constructed by Algorithm* $SOL\_\mathcal{C}_{max}(1)$

(the boundary of the stability ball $O_{\widehat{\varrho}_s(p)}(p)$*) and it becomes 'better' than digraph* $G_s(p)$*. For some suitable changes of the processing times* $p_i \pm (\widehat{\varrho}_s(p) + \epsilon)$ *(where* ϵ *is positive real, and*

it may be as small as desired) at least one of the four digraphs $\{G_1, G_2, G_3, G_4\}$ *becomes optimal. Therefore, a decision-maker can use one of the schedules from set* $\Lambda^*(G) = \{G_1, G_2, G_3, G_4\}$ *for the possible realization of the processing times.*

As it will be shown at the end of this section, set $\{G_1, G_2, G_3, G_4\}$ *is not minimal solution of the considered instance of problem* $\mathcal{J}3/n = 2, a_i \le p_i \le b_i/\mathcal{C}_{max}$ *since at least digraph* G_2 *is redundant in this set.*

In general, problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$ may be solved as follows. Let B denote a subset of feasible digraphs which contains solution $\Lambda^*(G)$ for the problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$. On the basis of the algorithm, which follows, we can expand the set $\Lambda' \subseteq \Lambda^*(G)$ starting with $\Lambda' = \emptyset$ and finishing with $\Lambda' = \Lambda^*(G)$.

Algorithm $SOL\_\mathcal{C}_{max}(1)$

Input: Set $\Lambda(G)$ of feasible digraphs, polytope T of feasible vectors of the processing times.

Output: Solution $\Lambda^*(G)$ of the problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$.

Step 1: Find set $B \subseteq \Lambda(G)$ such that $\Lambda^*(G) \subseteq B$.

Step 2: Set $\Lambda' = \emptyset$.

Step 3: Fix vector p of the processing times, $p \in T$.

Step 4: Find an optimal digraph $G_s(p) \in B$ for the problem $\mathcal{G}//\mathcal{C}_{max}$ with vector p of the processing times.

Step 5: Calculate relative stability radius $\widehat{\varrho}_s^B(p \in T)$.

Step 6: **IF** $\widehat{\varrho}_s^B(p \in T) < \infty$ and $B \setminus \{G_s\} \neq \emptyset$ **THEN**
BEGIN

Step 7: Select digraph $G_k(p) \in B$ which is a competitive digraph for $G_s(p)$.

Step 8: Find a vector $p^* \in T$ of the processing times closest to p such that $l_s^{p^*} = l_k^{p^*}$ and for any small $\epsilon > 0$, there exists a vector p^ϵ with $l_s^{p^\epsilon} > l_k^\epsilon$ and $d(p^*, p^\epsilon) \le \epsilon$.

Step 9: Set $\Lambda' := \Lambda' \cup \{G_s\}$.

Step 10: Set $B := B \setminus \{G_s\}$.

Step 11: Set $s = k$; $p = p^*$ **GOTO** *Step 5*
END

Step 12: **ELSE** $\Lambda^*(G) = \Lambda' \cup \{G_s\}$ **STOP**.

Now we concretize some steps of Algorithm $SOL\_\mathcal{C}_{max}(1)$. In Step 1, the determination of the set $B = \Lambda(G)$ of all feasible digraphs by an explicit enumeration is possible only

for a small number of edges in the mixed graph G. In the computational experiments discussed in Section 5.1, such a direct enumeration of feasible digraphs $\Lambda(G)$ has been used for small number $|E|$ of edges of the mixed graph $G = (Q, A, E)$, namely for $|E| \le 30$. These experiments have shown that a competitive digraph has a critical weight that is usually very close to that of an optimal digraph.

Moreover, using the simple bound proven at the end of this section, one can considerably restrict the number of feasible digraphs, with which a comparison of an optimal digraph G_s has to be done while calculating relative stability radius $\widehat{\varrho}_s^B(p \in T)$. For a larger cardinality of the set E, we can use a branch-and-bound algorithm for the construction of the k best digraphs (see Section 5.1). As it was shown for the traveling salesman problem [Lib99, LvdPSvdV96] and for linear binary programming [WJ88], the running time of such a branch-and-bound algorithm grows rather slowly with k.

In Step 3 we have to fix the processing times as any vector from set T. For example, we can use a 'historical' vector p of the processing times which helps to simplify the Steps 3, 4 or 5 (as it was done in Example 3.1). If the input data of the problem are new, we can set

$$p_i = \frac{1}{2}(b_i - a_i),\ i \in Q.$$

Step 4 may be realized by an explicit enumeration or by an implicit enumeration (e.g., by a branch-and-bound method) of the feasible digraphs B. In Step 4 we can apply Theorem 3.1 to guarantee that the selected optimal digraph G_s is stable. If $\widehat{\varrho}_s^B(p \in T) = 0$, we can take another optimal digraph (the latter exists due to Theorem 3.1) which is stable, or we can change the initial vector p of the processing times.

Steps 5, 7, and 8 may be done on the basis of Theorem 3.2 or Theorem 3.3. If $\widehat{\varrho}_s^B(p \in T) = \infty$, Theorem 3.2 can be used as a 'stopping rule' of the algorithm. Otherwise, we are forced to use Theorem 3.3 which is more time-consuming. A competitive digraph and a new vector p^* of the processing times are calculated in Algorithm $SOL\_\mathcal{C}_{max}(1)$ in parallel with the calculation of relative stability radius $\widehat{\varrho}_s^B(p \in T)$. Note that a competitive digraph is not necessarily uniquely determined, and so we can take one of them.

Steps 5 and 7 are rather complicated. In Algorithm $SOL\_\mathcal{C}_{max}(1)$ we must anew construct a set $H_{sk}(T)$ in each iteration based on a direct comparison of the paths in a new optimal digraph G_s and in each other digraph G_k from the set B, so it is very time-consuming.

Next, we propose a more efficient Algorithm $SOL\_\mathcal{C}_{max}(2)$, which focuses on one of the optimal digraphs G_1 and on one vector p of the processing times from set T. Let $\Gamma_i, i = 1, 2, \ldots, I$, be a set of competitive digraphs of digraph G_1 with respect to the set B, where i is a counter of the current iteration and I is the number of iterations.

Algorithm $SOL\_\mathcal{C}_{max}(2)$

Input: Set $\Lambda(G)$ of feasible digraphs, polytope T of feasible vectors of the processing times.

Output: Solution $\Lambda^*(G)$ of the problem $\mathcal{G}/a_i \le p_i \le b_i/\mathcal{C}_{max}$.

Step 1: Find set $B \subseteq \Lambda(G)$ such that $\Lambda^*(G) \subseteq B$.

Step 2: Set $\Lambda' = \emptyset$; $i = 1$ and $\Gamma_i = \emptyset$.

Step 3: Fix vector p of the processing times, $p \in T$.

Step 4: Find optimal digraph $G_1(p) := G_s(p) \in B$ for the problem $\mathcal{G}//\mathcal{C}_{max}$ with vector p of the processing times.

Step 5: Calculate $\widehat{\varrho}_1^B(p \in T)$.

Step 6: **IF** $\widehat{\varrho}_1^B(p \in T) < \infty$ **THEN**
BEGIN

Step 7: Select set of competitive digraphs Γ_i of digraph $G_1(p)$ with respect to set B.

Step 8: Set $\Lambda' := \Lambda' \cup \Gamma_i$.

Step 9: Set $B := B \setminus \Gamma_i$ and $i := i + 1$. **GOTO** *Step 5*
END

Step 10: **ELSE** $\Lambda^*(G) := \Lambda' \cup \{G_1\}$ **STOP**.

Using Algorithm $SOL\_\mathcal{C}_{max}(2)$, we construct an increasing sequence of relative stability radii $\widehat{\varrho}_1 < \widehat{\varrho}_2 < \ldots < \widehat{\varrho}_I$ of the stability balls $O_{\widehat{\varrho}_i}(p)$, $i \in \{1, 2, \ldots, I\}$, with the same center $p \in T$ and different sets of feasible digraphs

$$B = \Lambda(G) \ \setminus \ \bigcup_{j=1}^{i} \Gamma_j.$$

Moreover, we construct a sequence of 'nested sets' of the competitive digraphs

$$\Gamma_1, \ \Gamma_1 \bigcup \Gamma_2, \ \ldots, \ \bigcup_{i=1}^{I} \Gamma_i$$

of digraph G_1, where the set

$$\{G_1\} \bigcup \{\bigcup_{i=1}^{I} \Gamma_i\}$$

is a solution $\Lambda^*(G)$ of the scheduling problem for the mixed graph $(Q, A \cup E_1, \emptyset)$, and G_1 is one of the optimal digraphs in the set $\Lambda(G)$ for vector $p \in T$ of the processing times.

The most difficult part of Algorithm $SOL\_\mathcal{C}_{max}(2)$ is to find the stability radius $\widehat{\varrho}_1^B(p \in T)$ (Step 5 and Step 6) and to find the sets of competitive digraphs (Step 7). However, we can make the following remark.

Remark 3.4 It is not necessary to perform Steps 1 - 11 since we can construct a solution $\Lambda^*(G)$ in one scan. Namely, from Remark 3.3 it follows that all digraphs $G_k, k \neq 1$, for which a set $H_{1k}(T) \neq \emptyset$ was constructed in Step 5 are united with the optimal digraph G_1 and compose a solution:

$$\Lambda^*(G) = \{G_1\} \bigcup \{\bigcup_{i=1}^{I} \Gamma_i\} = \{G_1\} \bigcup \{G_k : \ H_{1k}(T) \neq \emptyset\}.$$

We can use the software developed for the problems discussed in Chapter 1 with the following modification: We add the loop of Steps 6 - 9. An increasing sequence of the relative stability radii of the stability balls with the same center $p \in T$ corresponds to an increasing sequence of the values $\widehat{r}_{k1}^{B}$ calculated by (3.26) for the optimal digraph $G_1(p)$ in Step 5. A competitive digraph (or set of competitive digraphs Γ_i) of digraph $G_1(p)$ is constructed in one scan, as well.

A more effective strategy (without adding the above loop) is described in Chapter 5.

Example 3.1 (continued). *Solving the above problem takes only two iterations by Algorithm SOL_$\mathcal{C}_{max}(2)$ (see Table 3.6). Thus, the set of digraphs*

$$\Lambda^*(G) = \{G_1, G_3\}$$

is also a solution of the problem $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\mathcal{C}_{max}$ under consideration. Thus, using Algorithm SOL_$\mathcal{C}_{max}(2)$, one can construct two stability balls $O_{60}(p)$ and $O_{\infty}(p)$, which cover the polytope T (see Figure 3.5). (Again, the stability ball with an infinite radius cannot be shown.)

So, we convince that Algorithm SOL_$\mathcal{C}_{max}(1)$ did not construct a minimal solution. Indeed in the solution $\{G_1, G_2, G_3, G_4\}$ constructed by Algorithm SOL_$\mathcal{C}_{max}(1)$ both digraphs G_2 and G_4 are redundant.

In general case, we do not know whether Algorithm SOL_$\mathcal{C}_{max}(2)$ constructs a minimal solution as well. However for Example 3.1, it is easy to see that there is no one-element solution for this problem. Hence solution $\Lambda^*(G) = \{G_1, G_3\}$ presented in Table 3.6 is minimal for Example 3.1 (with respect to cardinality of set $\Lambda^*(G)$).

Remark 3.5 For both algorithms, fixing the initial vector p in Step 3 and the choice of an optimal digraph $G_s(p)$ in Step 4 (and also in Step 7 for Algorithm SOL_$\mathcal{C}_{max}(1)$) have a large influence on the further calculations and the resulting solution.

Next, we show how to restrict the number of digraphs G_k (the cardinality of the set B) with which an optimal digraph has to be compared in the process of the calculation of the relative stability radius $\widehat{\varrho}_s^B(p \in T)$.

Redundant Digraphs for Calculating $\widehat{\varrho}_s^B(p \in T)$

Due to formulas (3.25) at page 126, the calculation of the relative stability radius is reduced to a complicated calculation on the set of digraphs $B \subseteq \Lambda(G)$. The main objects

Table 3.6: *Solution of problem $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\mathcal{C}_{max}$ by Algorithm SOL_$\mathcal{C}_{max}(2)$*

i	Set B	$\widehat{\varrho}_1^B(p \in T)$	Set Γ_i of competitive digraphs of the optimal digraph G_1
1	$\Lambda(G)$	60	$\{G_3\}$
2	$\Lambda(G) \setminus \{G_3\}$	∞	$\emptyset$

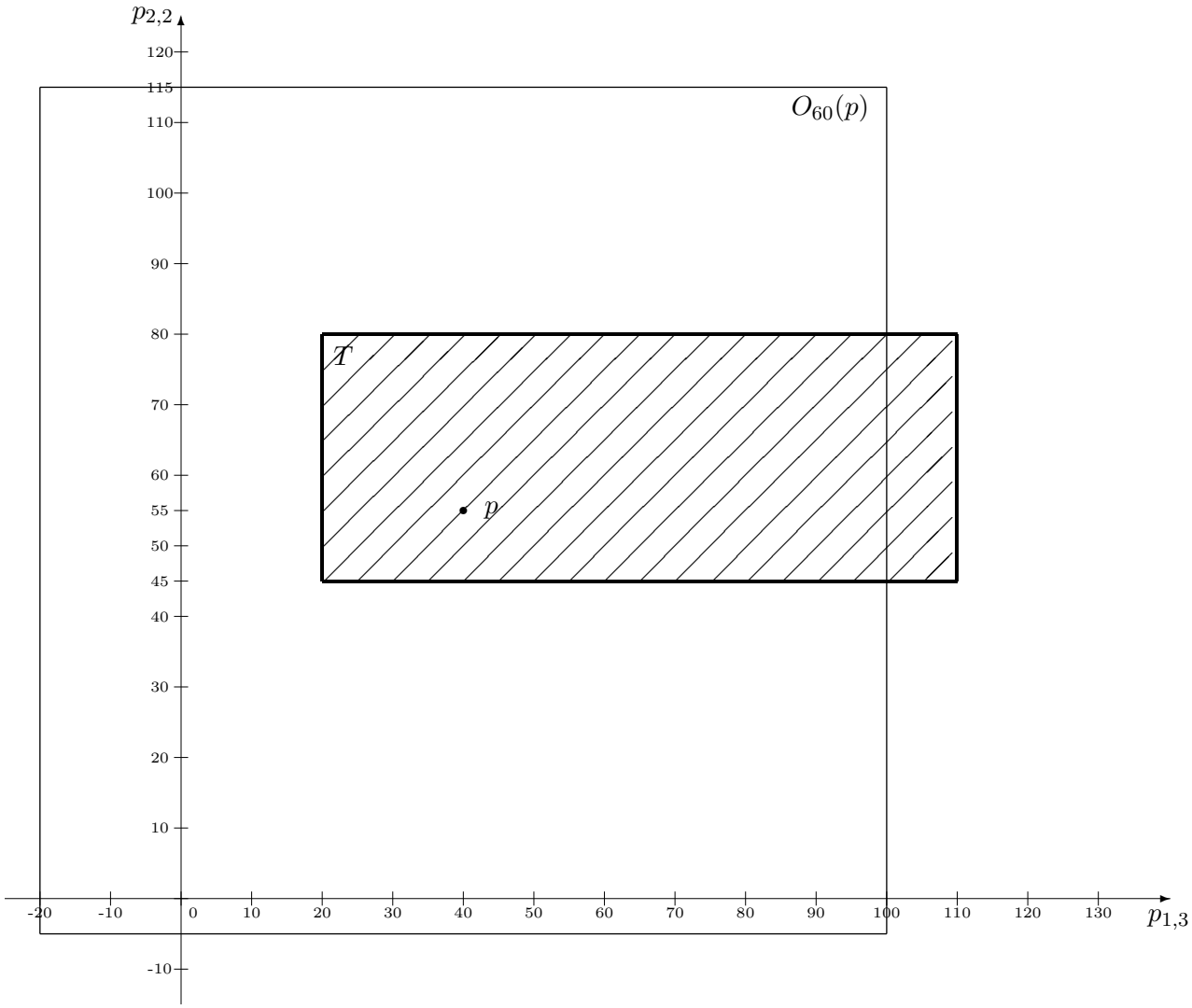

Figure 3.5: *Projections of the stability balls on the plane for problem* $\mathcal{J}3/n=2, a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ *constructed by Algorithm* $SOL\_\mathcal{C}_{max}(2)$

for the calculation of $\widehat{\varrho}_s^B(p \in T)$ are the sets of paths in the digraphs $G_k \in B$. At the worst case, the calculation of $\widehat{\varrho}_s^B(p \in T)$ implies to have an optimal digraph G_s and to construct all digraphs $B \subseteq \{G_1, G_2, \ldots, G_\lambda\}$. In order to restrict the number of digraphs G_k with which a comparison of the optimal digraph G_s has to be done during the calculation of $\widehat{\varrho}_s^B(p \in T)$, we can use the upper bound of the relative stability radius

$$\widehat{\varrho}_s^B(p \in T) \leq \widehat{r}_{ks}^B,$$

where $\widehat{r}_{ks}^B$ is defined according to formula (3.26) at page 126.

Lemma 3.4 *If* $\widehat{\varrho}_s^B(p \in T) < \infty$ *and there exists a digraph* $G_k \in B$ *such that*

$$\widehat{r}_{ks}^B \leq \frac{l_t^p - l_s^p}{q} \tag{3.35}$$

for some digraph $G_t \in B$, then it is not necessary to consider digraph G_t during the calculation of relative stability radius $\widehat{\varrho}_s^B(p \in T)$.

PROOF. To calculate relative stability radius $\widehat{\varrho}_s^B(p \in T)$, one can compare the optimal digraph G_s consecutively with each feasible digraph $G_i, i \neq s$, from the set B. The value $\widehat{r}_{ks}^B$ calculated according to (3.26) (see page 126) shows that there exists a feasible digraph G_k, which becomes better than digraph G_s for some vector $p' \in T$ of the processing times if

$$d(p, p') = \widehat{r}_{ks}^B + \epsilon,$$

where ϵ is a positive real number which may be as small as desired (see condition (2) introduced at page 126).

Next, we show that, if the condition of Lemma 3.4 is satisfied, i.e., inequality (3.35) holds, then the value $\widehat{r}_{ts}^B$ calculated for the digraph G_t does not improve the minimum in formula (3.25) (since inequalities $\widehat{\varrho}_s^B(p \in T) \le \widehat{r}_{ks}^B \le \widehat{r}_{ts}^B$ hold).

Let us compare digraph G_s, which is optimal for vector p of the processing times, with feasible digraph $G_t,\ t \neq k$. From condition (1) at page 126 (condition (2), respectively) it follows that digraph G_t is a competitive digraph for G_s if the weight of each path $\nu \in H_t$ of digraph G_t becomes equal to (smaller than) the weight of at least one path $\mu^* \in H_s$ of digraph G_s for some new vector $\widehat{x} \in T$ (for some new vector $\widehat{p}^{\epsilon} = \widehat{x} \pm \epsilon \in T$, where real number $\epsilon = d(\widehat{x}, \widehat{p}^{\epsilon}) > 0$ may be as small as desired). Hence, the inequality

$$\max_{\mu^* \in H_s} l^{\widehat{x}}(\mu^*) > \max_{\nu \in H_t} l^{\widehat{x}}(\nu)$$

$$\left(\max_{\mu^* \in H_s} l^{\widehat{p}^{\epsilon}}(\mu^*) > \max_{\nu \in H_t} l^{\widehat{p}^{\epsilon}}(\nu) \right)$$

holds. It means that the critical weight of digraph G_t becomes smaller than that of digraph G_s for some feasible vector of the processing times. Such a 'superiority' of the competitive digraph G_t occurs for some suitable changes of the processing times $\widehat{p}_i^{\epsilon} = p_i \pm (\widehat{r}_{ts}^B + \epsilon) = \widehat{x}_i \pm \epsilon$, when the value $\widehat{r}_{ts}^B = d(p, \widehat{x})$ calculated in (3.26) reaches the minimum value in (3.25) (see condition 3). To this end, one must increase the weights of the vertices, which form a path $\mu^* \in H_s$, by the minimal value $\widehat{r}_{ts}^B$ and decrease the weights of vertices from the set $[\nu^*] \setminus [\mu^*],\ \nu^* \in H_t$, by the same value $\widehat{r}_{ts}^B$ (according to formula (3.34)). Note that we must take such a path $\nu^* \in H_t$ for which the maximum in (3.26) is reached.

So, for the competitive digraph G_t, the distance $d(p, \widehat{x}) = \widehat{r}_{ts}^B$ must achieve its minimal value in formula (3.25) among the distances between the vector p and the other vectors in polytope T (i.e., the non-strict inequality $\widehat{r}_{ts}^B \le \widehat{r}_{ks}^B$ is also satisfied). However, we show further that, due to (3.35), value $\widehat{r}_{ts}^B$ cannot be smaller than $\widehat{r}_{ks}^B$ during the calculation of the relative stability radius $\widehat{\varrho}_s^B(p \in T)$ (see Theorem 1.4):

$$\begin{aligned} \widehat{r}_{ks}^B &\le \frac{l_t^p - l_s^p}{q} \le \frac{l_t^p - l^p(\mu^*)}{q} \le \frac{l_t^p - l^p(\mu^*)}{|[\nu^*] \setminus [\mu^*] + [\mu^*] \setminus [\nu^*]|} \\ &\le \max_{\nu \in H_t} \frac{l^p(\nu) - l^p(\mu^*)}{|[\mu^*] \cup [\nu]| - |[\mu^*] \cap [\nu]|} \le \min_{\mu \in H_s} \max_{\nu \in H_t} \frac{l^p(\nu) - l^p(\mu)}{|[\mu] + [\nu]|} \le \widehat{r}_{ts}^B. \end{aligned}$$

Since $\widehat{\varrho}_s^B(p \in T) \le \widehat{r}_{ks}^B \le \widehat{r}_{ts}^B$, the value $\widehat{r}_{ts}^B$ cannot decrease the value $\widehat{r}_{ks}^B$ in (3.25) and therefore digraph G_t need not to be considered during calculation of the relative stability radius using formulas (3.25) and (3.26).

$\diamond$

Note that bound (3.35) is tight. Lemma 3.4 directly implies the following corollary.

Corollary 3.4 *Let set* $B = \{G_s{=}G_{i_1}, G_{i_2}, \ldots, G_{i_{|B|}}\}$ *be sorted in non-decreasing order of the objective function values:* $l^p_{i_1} \le l^p_{i_2} \le \ldots \le l^p_{i_{|B|}}$. *If for the currently compared digraph* G_{i_k} *from set* $B \subseteq \Lambda(G)$ *inequality*

$$\widehat{r}^{\,B}_{i_k s} \le \frac{l^p_{i_t} - l^p_{i_1}}{q} \tag{3.36}$$

holds for digraph $G_{i_t} \in B$ *with* $l^p_{i_k} \le l^p_{i_t}$, *then it is possible to exclude digraphs* $G_{i_t}, G_{i_{t+1}}, \ldots, G_{i_{|B|}}$ *from further considerations during the calculation of relative stability radius* $\widehat{\varrho}^{\,B}_s(p \in T)$.

Proof. Since the digraphs in the set $B \subseteq \Lambda(G)$ are sorted in non-decreasing order of the objective function values and inequality (3.36) holds for digraph G_{i_t}, inequality

$$\widehat{r}^{\,B}_{i_k s} \le \frac{l^p_{i_j} - l^p_{i_1}}{q}$$

holds for each digraph $G_{i_j}, j = t+1, t+2, \ldots, |B|$. Therefore, due to Lemma 3.4, these digraphs need not to be considered during the calculation of the relative stability radius $\widehat{\varrho}^{\,B}_s(p \in T)$ (since we have the upper bound $\widehat{\varrho}^{\,B}_s(p \in T) \le \widehat{r}^{\,B}_{i_k s} \le \frac{l^p_{i_j} - l^p_{i_1}}{q}$).

◇

Using Corollary 3.4, we can compare the optimal digraph $G_s = G_{i_1}$ consecutively with the digraphs $G_{i_2}, G_{i_3}, \ldots, G_{i_{|B|}}$ from set B in non-decreasing order of the objective function values: $l^p_{i_1} \le l^p_{i_2} \le \ldots \le l^p_{i_{|B|}}$. If for the currently compared digraph $G_k = G_{i_r}$ inequality (3.35) holds, we can exclude the digraphs $G_{i_r}, G_{i_{r+1}}, \ldots, G_{i_{|B|}}$ from further considerations. Note that bound (3.35) is tight.

Since $\widehat{\varrho}_s(p) = \widehat{\varrho}^{\,\Lambda(G)}_s(p \in R^q_+)$, Corollary 3.4 implies Corollary 3.5 which allows to restrict the number of feasible digraphs while calculating stability radius $\widehat{\varrho}_s(p)$ (see Definition 1.2 at page 16).

Corollary 3.5 *Let set* $\Lambda(G) = \{G_s{=}G_{i_1}, G_{i_2}, \ldots, G_{i_\lambda}\}$ *be sorted in non-decreasing order of the objective function values:* $l^p_{i_1} \le l^p_{i_2} \le \ldots \le l^p_{i_\lambda}$. *If for the currently compared digraph* G_{i_k} *from set* $\Lambda(G) = \{G_s{=}G_{i_1}, G_{i_2}, \ldots, G_{i_k}, \ldots, G_{i_t}, \ldots, G_{i_\lambda}\}$ *inequality*

$$\widehat{r}^{\,\Lambda(G)}_{i_k s} \le \frac{l^p_{i_t} - l^p_{i_1}}{q} \tag{3.37}$$

holds for digraph $G_{i_t} \in \Lambda(G)$ *with* $l^p_{i_k} \le l^p_{i_t}$, *then it is possible to exclude digraphs* $G_{i_t}, G_{i_{t+1}}, \ldots, G_{i_\lambda}$ *from further considerations during the calculation of stability radius* $\widehat{\varrho}_s(p)$.

3.4 Dominance Relations

In this section, the job shop problem with the objective of minimizing the sum of job completion times under uncertain numerical input data is modeled in terms of a mixed graph. As far as scheduling is concerned, the mean flow time seems to be more important than makespan. While $\mathcal{C}_{max}$ aims for minimizing the schedule duration, in industry this duration is often defined by the periodicity of the process, say, a working day or a working week. So, the criterion $\sum \mathcal{C}_i$ is more suitable for a periodical scheduling environment. Theoretically speaking, the problem with $\sum \mathcal{C}_i$ is more difficult to solve than that with $\mathcal{C}_{max}$ as evidenced by the facts that the two-machine open shop problem $\mathcal{O}2//\mathcal{C}_{max}$ and the flow shop problem $\mathcal{F}2//\mathcal{C}_{max}$ are polynomially solvable while problems $\mathcal{O}2//\sum \mathcal{C}_i$ and $\mathcal{F}2//\sum \mathcal{C}_i$ are NP-hard (see [BEP+96, LLRKS93]). (Obviously, the NP-hardness of the latter problem implies the NP-hardness of problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$ under consideration). In academic research, criterion $\mathcal{C}_{max}$ has mainly been considered for multi-stage systems (open, flow and job shops) while criterion $\sum \mathcal{C}_i$ for single-stage systems, i.e., systems with one machine or with a set of *parallel* (usually identical) machines. One explanation could be that makespan is easier to attack than mean flow time.

In the problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$, it is assumed that only the structural input data (i.e., precedence and capacity constraints) are fixed while for the operation processing times only their lower and upper bounds are known before scheduling and the probability distribution functions of the random processing times are unknown. The structural input data are defined by the technological routes of the jobs, e.g., for a flow or open shop fixing the structural input data simply means to fix the number of jobs and the number of machines.

In this and next sections, two variants of a branch-and-bound method are developed. The first one constructs a set of k schedules which are the best with respect to the mean flow time criterion for a fixed feasible vector of the processing times. The second variant constructs a set of potentially optimal schedules for all perturbations of the processing times within the given lower and upper bounds. To exclude redundant schedules, we use a stability analysis based on the pairwise comparison of schedules. Along with implicit enumerations based on a branch-and-bound method, we realize an explicit enumeration of all feasible schedules.

We assume that n jobs $J = \{J_1, J_2, \ldots, J_n\}$ have to be processed on m machines $M = \{M_1, M_2, \ldots, M_m\}$ when only the technological routes of the jobs are given before scheduling. At the stage $j \in \{1, 2, \ldots, n_i\}$ of processing job J_i, operation $O_{ij} \in Q_k^J$ has to be processed by machine $M_k \in M$. The distribution of the operations to the machines M is fixed via the given technological routes of the jobs J. Each machine can process at most one operation at a time (see Condition 1 at page 2) and preemptions of an operation are forbidden (see Condition 4 at page 3).

We consider the job shop problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$ with fixed technological routes and uncertain processing times p_{ij}, which have to satisfy only the inequalities $a_{ij} \le p_{ij} \le b_{ij},\ J_i \in J;\ j = 1, 2, \ldots, n_i$ (see Condition 5 at page 6). The sum of the job completion times (mean flow time) is considered as the objective function $\Phi = \Phi(C_1, C_2, \ldots, C_n) = \sum_{i=1}^{n} C_i = \sum \mathcal{C}_i$, where $C_i = c_{in_i}$ is the completion time of job $J_i \in J$.

To present the structural input data for problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$, we use the mixed graph (Q^J, A^J, E^J) introduced at page 13. Such a mixed graph $G = (Q^J, A^J, E^J)$

defines the structural input data (precedence and capacity constraints) which are supposed to be known before scheduling. A schedule is defined as a circuit-free digraph $G_s = (Q^J, A^J \cup E_s^J, \emptyset)$ generated from the mixed graph (Q^J, A^J, E^J) by replacing each edge $[O_{ij}, O_{uv}] \in E^J$ by one of the arcs (O_{ij}, O_{uv}) or (O_{uv}, O_{ij}). In the rest of this chapter, we use the terms of an *optimal schedule (digraph)*, a *better* and a *best schedule (digraph)* with respect to the mean flow time criterion $\sum \mathcal{C}_i$. However, the makespan criterion $\mathcal{C}_{max}$ and a regular criterion Φ are considered in the rest of this chapter as well.

Due to Definition 3.1 given at page 110, a *solution* $\Lambda^*(G)$ of the problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$ is a set of digraphs containing at least one optimal digraph for each feasible vector $p = (p_{1,1}, p_{1,2}, \ldots, p_{nn_n})$ of the processing times, i.e., for each vector $p \in T$, where

$$T = \{x = (x_{1,1}, x_{1,2}, \ldots, x_{nn_n}) : \ a_{ij} \le x_{ij} \le b_{ij};\ i = 1, 2, \ldots, n;\ j = 1, 2, \ldots, n_i\}$$

is the polytope of all feasible vectors of the processing times in the space R_+^q, with $q = |Q^J| = \sum_{i=1}^n n_i = \sum_{k=1}^m |Q_k^J|$. It is practically important to look for a *minimal solution* $\Lambda^T(G)$ of problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$, i.e., for a minimal subset of the set $\Lambda(G)$ containing at least one optimal digraph for each fixed vector $p \in T$ of the processing times such that any proper subset of the set $\Lambda^T(G)$ is not a solution (see Definition 3.1).

If the processing times p_{ij} of all operations $O_{ij} \in Q^J$ are fixed, one can calculate the value of the objective function for a digraph $G_s \in \Lambda(G)$ using the critical path method. As it follows from Section 1.2, to solve problem $\mathcal{J}//\Phi$ we must find a digraph G_s such that $\Phi_s^p = \min\{\Phi_k^p : \ k = 1, 2, \ldots, \lambda\}$ (see formula (4.17) at page 196), where

$$\Phi_k^p = \Phi(\max_{\nu \in H_k^1} l^p(\nu), \max_{\nu \in H_k^2} l^p(\nu), \ldots, \max_{\nu \in H_k^n} l^p(\nu))$$

is the value of the objective function of the job completion times for the digraph $G_k \in \Lambda(G)$ with fixed processing times $p \in R_+^q$, and $l^p(\mu)$ is the weight of path μ: $l^p(\mu) = \sum_{O_{ij} \in [\mu]} p_{ij}$. Remind that $\Phi_s^p = l_s^p$ for criterion $\mathcal{C}_{max}$ and $\Phi_s^p = L_s^p$ for criterion $\sum \mathcal{C}_i$.

As it has been proven in Chapter 1 (see Theorem 1.3 at page 22), there exists problem $\mathcal{J}m/n = n^0/\mathcal{C}_{\max}$ with any given number of machines m and number of jobs $n = n^0$, for which the optimality of digraph $G_s \in \Lambda(G)$ does not depend on the numerical input data. In other words, $\widehat{\varrho}_s(p) = \infty$ which means that schedule s minimizes the makespan for all non-negative processing times. However, such a schedule cannot exist for criterion $\sum \mathcal{C}_i$. In other words, each optimal digraph (for mean flow time criterion) loses its optimality for some vectors $p \in R_+^q$ of the processing times, i.e., $\overline{\varrho}_s(p) < \infty$ (see Theorem 1.7 at page 36 and Remark 1.1). As it will be shown in the proof of Theorem 3.7 in the case of the relative stability radius $\overline{\varrho}_s^B(p \in T)$ (see Definition 3.6 below) when $T \subset R_+^q$ and $B \subset \Lambda(G)$, an unrestricted value of $\overline{\varrho}_s^B(p \in T)$ is still possible.

For the problem $\mathcal{J}/a_i \le p_i \le b_i/\Phi$, we introduce the following two transitive *dominance relations* which define partial orderings on the set of feasible digraphs $\Lambda(G)$.

Definition 3.5 *Feasible digraph G_s (strongly) dominates feasible digraph G_k in set $D \subseteq R_+^q$ if inequality $\Phi_s^p \le \Phi_k^p$ (inequality $\Phi_s^p < \Phi_k^p$, respectively) holds for any vector $p \in D$ of the processing times. We denote the dominance relation by $G_s \preceq_D G_k$, and the strong dominance relation by $G_s \prec_D G_k$.*

Let equality $a_{ij} = b_{ij}$ hold for each operation $O_{ij} \in Q^J$. In other words, set T turns into one point: $T = \{a\}$ where $a = (a_{1,1}, a_{1,2}, \ldots, a_{nn_n})$. Therefore, problem $\mathcal{J}/a_i \le p_i \le b_i/\Phi$ becomes a deterministic problem $\mathcal{J}//\Phi$, and the dominance relation $\preceq_T$ defines a total ordering on the set of digraphs $\Lambda(G)$. Consequently, minimal solution $\Lambda^T(G)$ consists of a single digraph: $\Lambda^T(G) = \{G_s\}$, where G_s is any optimal digraph for problem $\mathcal{J}//\Phi$ with processing times p_{ij} being equal to $a_{ij} = b_{ij}$ for each operation $O_{ij} \in Q^J$. In other words, digraph G_s dominates all digraphs $G_k \in \Lambda(G)$ at the point $a \in R^q_+$: $G_s \preceq_a G_k$. Moreover, if the strong dominance relation holds for each digraph $G_k \in \Lambda(G)$ at the point a, i.e., if $G_s \prec_a G_k$, then digraph G_s is the unique optimal digraph for the processing times p_{ij} equal to $a_{ij} = b_{ij}$. As it was shown in the computational results (see Section 5.2 below), an optimal digraph for problem $\mathcal{J}//\sum \mathcal{C}_i$ was mainly uniquely determined. In such cases, if the dominance relation $G_s \preceq_a G_k$ is valid for each digraph $G_k \in \Lambda(G)$, then generally the strong dominance relation $G_s \prec_a G_k$ is valid for each digraph $G_k \in \Lambda(G)$ with $k \ne s$. (Note that this is not the case for the makespan criterion: For the most job shop instances which have been randomly generated makespan optimal digraphs were not uniquely determined.)

In the general case of the problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$, the operation processing times may vary between given lower and upper bounds and therefore it is a priori unknown which path from the set H^i_k will have the largest weight in a practical realization of schedule k corresponding to digraph G_k. Thus, we have to consider the whole set Ω^u_k of representatives of the family of sets $(H^i_k)_{J_i \in J}$ in a similar way to the approach considered for the problem $\mathcal{J}//\sum \mathcal{C}_i$ (see Section 1.4).

Each of these sets Ω^u_k includes exactly one path from each set H^i_k, $J_i \in J$. Since $H^i_k \cap H^j_k = \emptyset$ for any pair of different jobs J_i and J_j, we have the equality $|\Omega^u_k| = n$ and so there exist $\omega_k = \prod_{i=1}^n |H^i_k|$ different sets of representatives for digraph G_k, namely: $\Omega^1_k, \Omega^2_k, \ldots, \Omega^{\omega_k}_k$. Next, we show how to restrict the number of sets of representatives which have to be considered while solving problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$.

Similarly to the notion of a *critical path* which is particularly important for criterion C_{max}, we use the notion of a *critical set* $\Omega^{u^*}_k$ for criterion $\sum C_i$. The set $\Omega^{u^*}_k$, $u^* \in \{1, 2, \ldots, \omega_k\}$, is a critical set in $G_k \in \Lambda(G)$ if the objective function value L^p_k is reached on this set of paths, i.e., if equality $\sum_{\nu \in \Omega^{u^*}_k} l^p(\nu) = L^p_k$ holds.

For different vectors $p \in R^q_+$ of the processing times, different sets Ω^u_k, $u \in \{1, 2, \ldots, \omega_k\}$, may be *critical*, however a path $\nu \in H^i_k$, $J_i \in J$, may belong to a critical set only if $l^p(\nu) = \max_{\mu \in H^i_k} l^p(\mu)$. Therefore, while solving problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$, it is sufficient to consider only paths from the set H^i_k which may have the largest weight for at least one vector $p \in T$ of the processing times. Moreover, if there are two or more paths in H^i_k which have the largest weight at the same vector $p \in T$, it is sufficient to consider only one of them. Thus, it is sufficient to consider only *dominant* paths which were defined in Section 3.2 (see Definition 3.6 at page 119).

Using Corollary 3.2, one can simplify digraph G_s while solving problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$ or problem $\mathcal{J}/a_i \le p_i \le b_i/\mathcal{C}_{max}$. First, we delete all transitive arcs, then we delete some arcs on the basic of domination of path sets (see Definition 3.4).

Let $H^i_s(T)$ denote the set of all dominant paths in H^i_s with respect to the polytope T. Since $H_s \subseteq \cup_{i=1}^n H^i_s$ for problem $\mathcal{J}/a_i \le p_i \le b_i/\mathcal{C}_{max}$, one can construct a set $H^i_s(T)$ as a subset of the set of all dominant paths H_s by selecting all paths ending in vertex O_{in_i}

(if they exist). Let $G_s^T = (Q_s^T, E_s^T, \emptyset)$ be a *minimal subgraph* of digraph G_s such that, if $\mu \in \bigcup_{i=1}^n H_s^i(T)$, then digraph G_s^T contains path μ. To construct the digraph G_s^T, one can use the following straightforward modification of the critical path method [Dij59].

Assume that the path μ has the maximal weight among all paths in digraph G_s ending in vertex O_{ij} when the processing times are defined by the vector $p \in R_+^q$. As usual, the weight of path μ minus p_{ij} is called *earliest start time* of operation O_{ij} and we denote it by $l_s^p(O_{ij})$:

$$l_s^p(O_{ij}) = \sum_{O_{uv} \in [\mu] \setminus \{O_{ij}\}} p_{uv}.$$

The following recursive relations are obvious:

$$l_s^a(O_{ij}) = \max\{l_s^a(O_{uv}) + a_{uv} \ : \ (O_{uv}, O_{ij}) \in A^J \cup E_s^J\},$$

$$l_s^b(O_{ij}) = \max\{l_s^b(O_{uv}) + b_{uv} \ : \ (O_{uv}, O_{ij}) \in A^J \cup E_s^J\}.$$

Starting with a vertex in digraph G_s which has a zero in-degree and following the critical path method, we define values $l_s^a(O_{ij})$ and $l_s^b(O_{ij})$ for each vertex $O_{ij} \in Q^J$. Then, using backtracking, we define vertices Q_s^T and arcs E_s^T of digraph G_s^T as follows. Initially, we set $Q_s^T = \{O_{in_i} : \ J_i \in J\}$ for problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$ (if $H_s^i(T) \neq \emptyset$, $H_s^i(T) \subseteq H_s$, for problem $\mathcal{J}/a_i \le p_i \le b_i/\mathcal{C}_{max}$) and we set $E_s^T = \emptyset$. Then we add vertex O_{uv} to the set Q_s^T, and we add arc (O_{uv}, O_{in_i}) to the set E_s^T:

$$Q_s^T := Q_s^T \cup \{O_{uv}\}, \ E_s^T := E_s^T \cup \{(O_{uv}, O_{in_i})\}$$

if and only if the following two conditions hold:
1) there is no arc $(O_{u_1v_1}, O_{in_i})$ such that $l_s^b(O_{u_1v_1}) < l_s^a(O_{uv})$, and
2) inequality $l_s^b(O_{uv}) + b_{in_i} \ge l_s^a(O_{in_i})$ holds.
Continuing in a similar way for each vertex which is already included in the set Q_s^T, we construct the digraph G_s^T (see Lemma 3.1).

Thus, instead of digraphs $G_k, \ k = 1, 2, \ldots, \lambda$, one can consider digraphs G_k^T which contain all dominant paths $\bigcup_{i=1}^n H_k^i(T)$ and which are often essentially simpler than the corresponding digraphs G_k. The transformation of digraph G_k into digraph G_k^T by testing inequality (3.7)

$$\sum_{O_{ij} \in [\mu] \setminus [\nu]} b_{ij} \le \sum_{O_{uv} \in [\nu] \setminus [\mu]} a_{uv}$$

(see page 119) takes $O(q^2)$ elementary steps (q is the number of operations).

Let for criterion $\sum \mathcal{C}_i$ the superscripts of the sets $\Omega_k^1, \Omega_k^2, \ldots, \Omega_k^{\omega_k^T}, \ldots, \Omega_k^{\omega_k}$ be such that for a path μ the inclusion

$$\mu \in \bigcup_{i=1}^n H_k^i(T)$$

holds if and only if

$$\mu \in \bigcup_{i=1}^{\omega_k^T} \Omega_k^i, \ \omega_k^T = \prod_{i=1}^n |H_k^i(T)|.$$

Example 3.2 *To illustrate the above notions and definitions, we introduce a job shop problem* $\mathcal{J}3/n = 3, a_i \le p_i \le b_i/\sum \mathcal{C}_i$ *with* $Q_1^J = \{O_{1,1}, O_{1,3}, O_{3,2}\}, Q_2^J = \{O_{1,2}, O_{2,1}, O_{3,3}\}$

and $Q_3^J = \{O_{2,2}, O_{3,1}\}$. *The mixed graph* $G = (Q^J, A^J, E^J)$ *represented in Figure 3.6 defines the structural input data. The numerical input data are defined by polytope* $T \in R_+^8$ *(see Table 3.7) within which the actual vector* p *of the processing times has to be contained.*

For this small example, one can explicitly enumerate all feasible digraphs of set $\Lambda(G)$. *(the cardinality* λ *of the set* $\Lambda(G)$ *is equal to 22). Since not all digraphs may be optimal for the given segments* $[a_{ij}, b_{ij}]$ *of possible variations of the processing times* p_{ij}, *we construct a subset* B *of the set* $\Lambda(G)$ *of possible candidates of competitive digraphs (optimal digraphs) using the algorithms from Section 3.6 below. The cardinality of the set* B *is equal to 12, while* $\lambda = 22$.

Table 3.7: *Numerical data for problem* $\mathcal{J}3/n=3, a_i \le p_i \le b_i/\sum \mathcal{C}_i$

i	1	1	1	2	2	3	3	3
j	1	2	3	1	2	1	2	3
a_{ij}	60	20	45	10	50	60	30	30
b_{ij}	80	40	60	30	70	80	50	40

Before finding a minimal solution $\Lambda^T(G)$ *of this problem* $\mathcal{J}3/n=3, a_i \le p_i \le b_i/\sum \mathcal{C}_i$, *we consider its deterministic version* $\mathcal{J}3/n=3/\sum \mathcal{C}_i$ *by setting the vector of the processing times to be equal to* $p^0 = (p^0_{1,1}, p^0_{1,2}, \ldots, p^0_{3,3}) \in T$ *with* $p^0_{1,1} = 70, p^0_{1,2} = 30, p^0_{1,3} = 60, p^0_{2,1} = 20, p^0_{2,2} = 60, p^0_{3,1} = 70, p^0_{3,2} = 40$, *and* $p^0_{3,3} = 30$. *(Note that this vector can be chosen arbitrarily from the polytope* T.*) We number the digraphs* $G_1, G_2, \ldots, G_{12}$ *in accordance with non-decreasing values of the function* $\sum \mathcal{C}_i$ *calculated for the vector* p^0 *of the processing times:* $L_1^{p^0} = 440, L_2^{p^0} = 470, L_3^{p^0} = 500, L_4^{p^0} = 500, L_5^{p^0} = 520, L_6^{p^0} = 530, L_7^{p^0} = 540, L_8^{p^0} = 550, L_9^{p^0} = 570, L_{10}^{p^0} = 610, L_{11}^{p^0} = 610, L_{12}^{p^0} = 620$.

For the vector $p^0 \in T$, *the digraph* $G_1 = (Q^J, A^J \cup E_1^J, \emptyset)$ *with the signature* $E_1^J = \{(O_{1,1}, O_{3,2}), (O_{3,2}, O_{1,3}), (O_{2,1}, O_{1,2}), (O_{1,2}, O_{3,3}), (O_{2,1}, O_{3,3}), (O_{3,1}, O_{2,2})\}$ *is the only optimal digraph. Therefore, for the initial problem* $\mathcal{J}3/n=3, a_i \le p_i \le b_i/\sum \mathcal{C}_i$, *we have to include the digraph* G_1 *in the desired minimal solution* $\Lambda^T(G)$. *Using the above modification of critical path method, we simplify the digraphs* G_1, $G_2, \ldots, G_{12}$. *Then we compare*

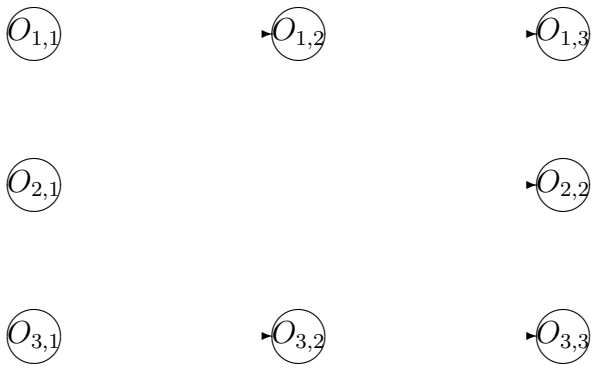

Figure 3.6: *Mixed graph* $G = (Q^J, A^J, E^J)$ *for problem* $\mathcal{J}3/n=3, a_i \le p_i \le b_i/\sum \mathcal{C}_i$

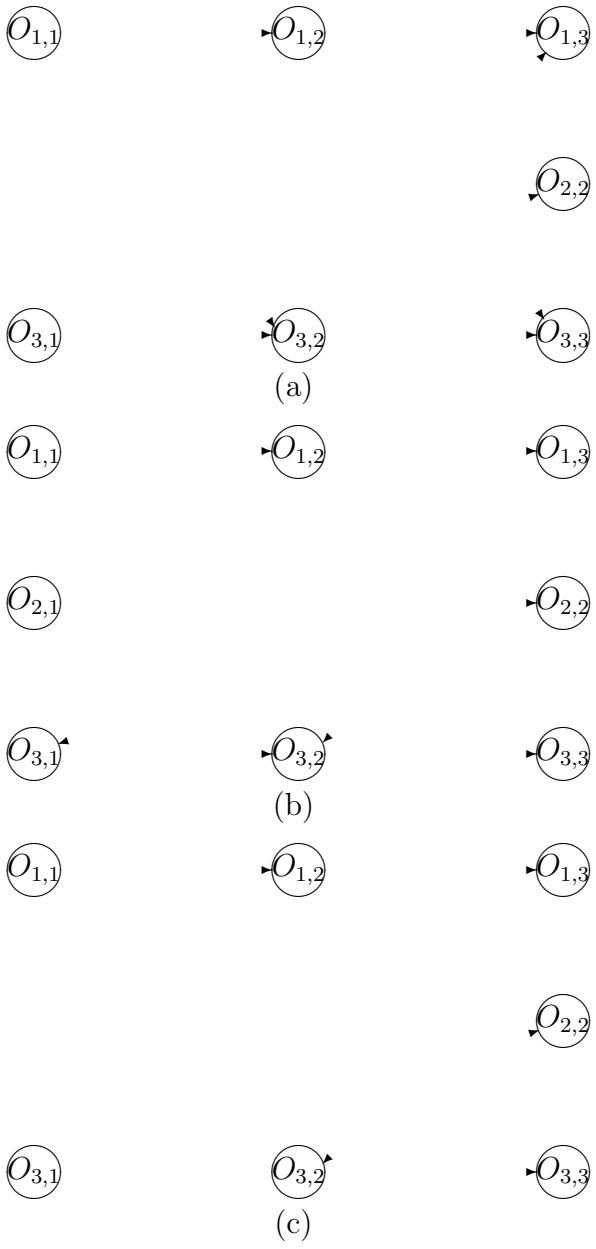

Figure 3.7: *Digraphs G_1^T, G_2^T and G_5^T which define a minimal solution $\Lambda^T(G)$ for Example 3.2*

the sets of representatives $\Omega_1^1, \Omega_1^2, \ldots, \Omega_1^{\omega_1^T}$ for the digraph G_1 with the sets of representatives $\Omega_k^1, \Omega_k^2, \ldots, \Omega_k^{\omega_k^T}$ for the other digraphs G_k, $k = 2, 3, \ldots, 12$. Due to a pairwise comparison of these sets, we find that only two digraphs may be better than the digraph G_1 (provided that the vector of the processing times belongs to the polytope T defined in Table 3.7). These two digraphs are as follows: Digraph $G_2 = (Q^J, A^J \cup E_2^J, \emptyset)$ with the signature $E_2^J = \{(O_{1,1}, O_{3,2}), (O_{1,3}, O_{3,2}), (O_{2,1}, O_{1,2}), (O_{1,2}, O_{3,3}), (O_{2,1}, O_{3,3}), (O_{2,2}, O_{3,1})\}$ and digraph $G_5 = (Q^J, A^J \cup E_5^J, \emptyset)$ with the signature $E_5^J = \{(O_{1,1}, O_{3,2}), (O_{1,3}, O_{3,2}),$

$(O_{2,1}, O_{1,2}), (O_{1,2}, O_{3,3}), (O_{2,1}, O_{3,3}), (O_{3,1}, O_{2,2})\}$. *Moreover, the digraph* G_2 *is the only optimal one, for example, for the vector* $p' = (60, 20, 60, 10, 60, 80, 40, 30) \in T$, *and digraph* G_5 *is the only optimal one for the vector* $p'' = (60, 20, 45, 30, 70, 80, 50, 30) \in T$. *Consequently, a minimal solution of problem* $\mathcal{J}3/n=3, a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$ *consists of three digraphs, namely:* $\Lambda^T(G) = \{G_1, G_2, G_5\}$. *The corresponding digraphs* G_1^T, G_2^T *and* G_5^T *are represented in Figure 3.7 a), b) and c), respectively. Note that while digraph* G_1 *has* $\omega_1 = 4 \cdot 2 \cdot 5 = 40$ *sets of representatives, digraph* G_1^T *has only* $\omega_1^T = 3 \cdot 1 \cdot 3 = 9$ *sets of representatives. For the digraphs* G_2 *and* G_2^T, *these numbers are* $\omega_2 = 16$ *and* $\omega_2^T = 2$, *and for the digraphs* G_5 *and* G_5^T, *these numbers are* $\omega_5 = 28$ *and* $\omega_5^T = 1$.

The above full enumeration of digraphs $\Lambda(G)$ is only possible for a small number of edges in the mixed graph G, and for a practical use one has to reduce the number of digraphs which have to be constructed. E.g., for the example under consideration, it is sufficient to construct only $k = 5$ digraphs, which are the best for the initial vector p^0 of the processing times. Further, in Section 3.6 such a calculation will be developed on the basis of a branch-and-bound method for constructing k best digraphs. Moreover, the digraphs G_3 and G_4 in the set of the $k = 5$ best digraphs are also redundant. In Section 3.6, we present a branch-and-bound method for constructing all digraphs which are the only ones that may be optimal for feasible vectors of the processing times. We also show how to calculate the stability radius of an optimal digraph on the basis of an explicit enumeration of the digraphs $\Lambda(G)$. The calculation of the stability radius will be used in Sections 3.6 and 5.4 as the main procedure for finding a minimal solution of problem $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$.

The remainder of this chapter is organized as follows. Necessary and sufficient conditions for a set of digraphs to be a solution respectively a minimal solution of the job shop problem with uncertain processing times are proven in Section 3.5. Three exact and four heuristic algorithms for problems $\mathcal{J}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ and $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$ are given in Section 3.6. Section 3.6 contains a branch-and-bound method ($B\&B1$) for constructing k schedules which are the best for problem $\mathcal{J}/a_i \leq p_i \leq b_i/\Phi$ with some fixed processing times. As it has been shown for the traveling salesman problem [Lib99, LvdPSvdV96, vdP97] and for linear binary programming [PZ76, WJ88], the running time of such a branch-and-bound variant grows relatively slowly with k. We develop also a branch-and-bound method ($B\&B2$) for constructing all schedules which may be optimal if the processing times vary between given lower and upper bounds. Unfortunately, both algorithms $B\&B1$ and $B\&B2$ may construct some redundant schedules, which are not necessarily contained in a minimal solution of problem $\mathcal{J}/a_i \leq p_i \leq b_i/\Phi$. To reject such redundant schedules, we use a deeper (but more time-consuming) stability analysis of an optimal schedule.

3.5 Characterization of a Solution

A characterization of a solution Λ of problem $\mathcal{J}/a_i \leq p_i \leq b_i/\Phi$ which is a proper subset of the set $\Lambda(G), \Lambda \subset \Lambda(G)$, may be obtained on the basis of the dominance relation $\preceq_D$ introduced in Section 3.4. Next, we prove necessary and sufficient conditions for a set of feasible digraphs to be a solution of the problem $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$.

Theorem 3.4 *The set $\Lambda \subset \Lambda(G)$ is a solution of problem $\mathcal{J}/a_i \le p_i \le b_i/\Phi$ if and only if there exists a finite covering of the polytope T by convex closed sets $D_j \subset R^q_+$:*

$$T \subseteq \bigcup_{j=1}^{d} D_j,\ d \le |\Lambda|,$$

such that for any digraph $G_k \in \Lambda(G)$ and for any set $D_j,\ j = 1, 2, \ldots, d$, there exists a digraph $G_s \in \Lambda$ for which dominance relation $G_s \preceq_{D_j} G_k$ holds.

Proof. *Sufficiency.* For any fixed vector $p \in T$, one can find a set $D_j, 1 \le j \le d$, such that $p \in D_j$. From the condition of Theorem 3.4, it follows that for any digraph $G_k \in \Lambda(G)$, there exists a digraph G_s such that dominance relation $G_s \preceq_{D_j} G_k$ holds. Hence, we have $\Phi^p_s \le \Phi^p_k$ and so inequality $\min\{\Phi^p_s\ :\ G_s \in \Lambda \subseteq \Lambda(G)\} \le \Phi^p_k$ holds for each $k = 1, 2, \ldots, \lambda$. Consequently, for any vector $p \in T$ of the processing times, set Λ contains an optimal digraph.

Necessity. Let set $\Lambda \subseteq \Lambda(G)$ be a solution of problem $\mathcal{J}/a_i \le p_i \le b_i/\Phi$. We define a subset Λ' of the set Λ such that each digraph $G_s \in \Lambda'$ is optimal for at least one vector $p \in T$ of the processing times.

For each digraph $G_s \in \Lambda$, one can define its *stability region*, i.e., the set of all vectors $p \in T \subseteq R^q_+$ for which digraph G_s is optimal. Let D_s be the intersection of the stability region of the digraph G_s with the polytope T:

$$D_s = \{p \in R^q_+\ :\ \Phi^p_s \le \Phi^p_k,\ k = 1, 2, \ldots, \lambda\} \cap T. \tag{3.38}$$

Since Λ' is a solution, we have

$$T \subseteq \bigcup_{j=1}^{|\Lambda'|} D_j \subset R^q_+$$

and for each digraph $G_k \in \Lambda(G)$ and each set D_s, the dominance relation $G_s \preceq_{D_s} G_k$ holds. The inclusion $G_s \in \Lambda'$ implies $D_s \neq \emptyset$. From inequality (3.38) it follows that D_s is a closed set.

Note that, if digraph G_s is optimal for the vector p, it remains optimal for a feasible vector αp with any positive real number $\alpha > 0$. Consequently, the stability region is convex set and so set D_s is convex as the intersection of convex sets.

$\diamond$

Theorem 3.4 implies the following corollary which characterizes a single-element solution of problem $\mathcal{J}/a_i \le p_i \le b_i/\Phi$, which is necessarily a minimal solution.

Corollary 3.6 *The equality $\Lambda^T(G) = \{G_s\}$ holds if and only if the dominance relation $G_s \preceq_T G_k$ holds for any digraph $G_k \in \Lambda(G)$.*

A minimal solution which includes more than one digraph may be characterized on the basis of the strong dominance relation $\prec_D$ as follows.

Theorem 3.5 *Let set $\Lambda^*(G)$ be a solution to problem $\mathcal{J}/a_i \le p_i \le b_i/\Phi$ with $|\Lambda^*(G)| \ge 2$. This solution is minimal if and only if for each digraph $G_s \in \Lambda^*(G)$, there exists a vector $p^{(s)} \in T$ such that the strong dominance relation $G_s \prec_{p^{(s)}} G_k$ holds for each digraph $G_k \in \Lambda^*(G) \setminus \{G_s\}$.*

PROOF. *Sufficiency.* If the condition of Theorem 3.5 holds, then for any digraph $G_s \in \Lambda^*(G)$, the set $\Lambda^*(G) \setminus \{G_s\}$ is no longer a solution to problem $\mathcal{J}/a_i \le p_i \le b_i/\Phi$.

Indeed, for the above vector $p^{(s)} \in T$, inequality $\Phi_s^{p^{(s)}} < \Phi_k^{p^{(s)}}$ holds for each digraph $G_k \in \Lambda^*(G) \setminus \{G_s\}$. It follows that G_s is the unique optimal digraph to problem $\mathcal{J}//\Phi$ with vector $p^{(s)}$ of the processing times. Therefore, solution $\Lambda^*(G)$ is minimal.

Necessity. We assume that $\Lambda^*(G)$ is a minimal solution but the condition of Theorem 3.5 does not hold, i.e., there exists a digraph $G_s \in \Lambda^*(G)$ such that for each vector $p^{(s)} \in T$, there exists a digraph $G_k \in \Lambda^*(G) \setminus \{G_s\}$ for which the strong dominance relation $G_s \prec_{p^{(s)}} G_k$ does not hold, i.e., we have $\Phi_s^p \ge \Phi_k^p$. It follows that the set $\Lambda^*(G) \setminus \{G_s\}$ is also a solution of problem $\mathcal{J}/a_i \le p_i \le b_i/\Phi$ (since the set $\Lambda^*(G)$ is supposed to be a solution). Thus, we get a contradiction to the assumption that solution $\Lambda^*(G)$ is minimal. ◇

Section 3.6 deals with different algorithms for finding a solution and a minimal solution on the basis of an explicit or an implicit enumeration of feasible digraphs. All algorithms developed are based on the fact that a digraph $G_s \in \Lambda(G)$ being optimal for the fixed vector $p \in R_+^q$ of the processing times, generally remains optimal within some neighborhood of the point p in the space R_+^q. In other words, digraph G_s dominates all digraphs in a neighborhood of the point p. We consider the closed ball $O_r(p) \subset R^q$ with the center $p \in T$ and the radius $r > 0$ as the neighborhood of the point $p \in T \subset R_+^q$ in the space R^q. Next, we rewrite some basis notions using dominance relation $\preceq_D$.

The closed ball $O_r(p)$ is called a *stability ball* of the digraph G_s if this digraph dominates all digraphs $G_k \in \Lambda(G)$ in the polytope $T^* = O_r(p) \cap T$, i.e., if $G_s \preceq_{T^*} G_k$ for each digraph $G_k \in \Lambda(G)$ (in this case, from Corollary 3.6 it follows that $\Lambda^{T^*}(G) = \{G_s\}$). As it was noted in Section 3.2, the radius r of a stability ball may be interpreted as the *error* of the given processing times $p = (p_{1,1}, p_{1,2}, \ldots, p_{nn_n}) \in R_+^q$ such that for all variable processing times $x = (x_{1,1}, x_{1,2}, \ldots, x_{nn_n}) \in R_+^q$ with $p_{ij} - r \le x_{ij} \le p_{ij} + r$ digraph G_s remains the best. The maximal value of such a radius is of particular importance for finding a minimal solution $\Lambda^T(G)$. Similarly to Definition 3.2 of the relative stability radius for the makespan criterion, we give the definition of the relative stability radius for the mean flow time criterion.

Definition 3.6 *Assume that for each vector $p' \in O_\varrho(p) \cap T$ digraph $G_s \in B \subseteq \Lambda(G)$ with the vector p' of weights has the minimal critical sum of weights $L_s^{p'}$ among all digraphs of the set B. The maximal value of the radius ϱ of such a ball $O_\varrho(p)$ is denoted by $\overline{\varrho}_s^B(p \in T)$ and is called the relative stability radius of the digraph G_s with respect to the polytope T for criterion $\sum \mathcal{C}_i$.*

Remark 3.6 From Definition 3.5 and Definition 3.6, it follows that the *relative stability radius* $\overline{\varrho}_s^B(p \in T)$ of the digraph $G_s \in B$ is equal to the maximal value of the radius ϱ of a ball $O_\varrho(p)$ such that for each digraph $G_k \in B \subseteq \Lambda(G)$ dominance relation $G_s \preceq_{T^*} G_k$ holds where $T^* = O_\varrho(p) \cap T$.

Similarly to Section 1.4 (which deals with the stability radius $\overline{\varrho}_s(p)$ (see conditions (1.30) and (1.31) at page 31), to find the relative stability radius $\overline{\varrho}_s^B(p \in T)$ for problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$ it is sufficient to construct a vector $\overline{x} = (\overline{x}_{1,1}, \overline{x}_{1,2}, \ldots, \overline{x}_{nn_n}) \in T \subseteq R_+^q$

which satisfies the following three conditions.
(a^*) There exists a digraph $G_k(p) \in B, k \neq s$, such that $L_s^{\overline{x}} = L_k^{\overline{x}}$, i.e.,

$$\sum_{i=1}^{n} \max_{\mu \in H_s^i} l^{\overline{x}}(\mu) = \sum_{i=1}^{n} \max_{\nu \in H_k^i} l^{\overline{x}}(\nu). \tag{3.39}$$

(b^*) For any given real $\epsilon > 0$, which may be as small as desired, there exists a vector $\overline{p}^{\epsilon} \in T$ such that $d(\overline{x}, \overline{p}^{\epsilon}) = \epsilon$ and $L_s^{\overline{p}^{\epsilon}} > L_k^{\overline{p}^{\epsilon}}$, i.e., inequality

$$\sum_{i=1}^{n} \max_{\mu \in H_s^i} l^{\overline{p}^{\epsilon}}(\mu) > \sum_{i=1}^{n} \max_{\nu \in H_k^i} l^{\overline{p}^{\epsilon}}(\nu) \tag{3.40}$$

is satisfied for at least one digraph $G_k(p) \in B$.
(c^*) The distance $d(p, \overline{x})$ achieves its minimal value among the distances between the vector p and the other vectors in the polytope T which satisfy both above conditions (a^*) and (b^*).

Next, we describe the calculation of the relative stability radius $\overline{\varrho}_s^B(p \in T)$ using the above notation of the dominance relation. To this end, we prove Lemma 3.5 below about the dominance relation $\preceq_T$, and then we derive a formula for the calculation of the relative stability radius $\overline{\varrho}_s^B(p \in T)$ which is presented in Theorem 3.6.

If $\Lambda^T(G) = \{G_s\}$, then digraph G_s dominates all digraphs in the polytope T (see Corollary 3.6). In such a case, we assume that $\overline{\varrho}_s^{\Lambda(G)}(p \in T) = \infty$, since digraph G_s remains the best for all variable feasible vectors $x \in T$ of the processing times. Otherwise, there exists a digraph $G_k \in \Lambda(G)$ such that dominance relation $G_s \preceq_T G_k$ does not hold, and from Corollary 3.6 and Remark 3.6, it follows that the stability radius $\overline{\varrho}_s^{\Lambda(G)}(p \in T)$ has to be finite, i.e., there exists a vector $\overline{p}^{\epsilon} \in T$ such that inequality (3.41)

$$L_s^{\overline{p}^{\epsilon}} > L_k^{\overline{p}^{\epsilon}} \tag{3.41}$$

holds. To calculate the stability radius $\overline{\varrho}_s^B(p \in T),\ B \subseteq \Lambda(G)$, we will consider digraphs $G_k \in B$ such that dominance relation $G_s \preceq_T G_k$ does not hold, and for each of these digraphs G_k, we will look for the vector $\overline{p}^{\epsilon} \in T$ which is the closest to p, among all vectors for which inequality (3.41) holds (see condition (c^*)). The following lemma allows to restrict the set of digraphs $G_k \in B$ which have to be considered for any regular criterion.

Lemma 3.5 *Digraph $G_s \in B$ dominates digraph $G_k \in B$ in the polytope T if (only if) the following inequality (3.42) holds (inequalities (3.43) hold, respectively):*

$$L_s^b \leq L_k^a \tag{3.42}$$

$$(L_s^a \leq L_k^a,\ L_s^b \leq L_k^b). \tag{3.43}$$

PROOF. *Sufficiency.* Since the objective function is non-decreasing, it follows from inequality (3.42) that

$$L_s^x \leq \sum_{i=1}^{n} l_s^b(O_{in_i}) = L_s^b \leq L_k^a = \sum_{i=1}^{n} l_k^a(O_{in_i}) \leq L_k^x$$

for any vector $x \in T$. Therefore, dominance relation $G_s \preceq_T G_k$ holds.

Necessity. Dominance relation $G_s \preceq_T G_k$ means that inequality $L_s^x \le L_k^x$ holds for any vector $x \in T$ and thus for both vectors $a \in T$ and $b \in T$, i.e., inequalities (3.43) hold.

$\diamond$

The test of inequalities (3.42) and (3.43) takes $O(q^2)$ elementary steps, however, there is a 'gap' between the necessary and sufficient conditions of Lemma 3.5, if $L_s^a \ne L_s^b$. To overcome this gap for problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$, we are forced to compare the sets $\Omega_s^v,\ v = 1, 2, \ldots, \omega_s^T$, with the sets $\Omega_k^u,\ u = 1, 2, \ldots, \omega_k^T$, since we do not know a priori which set will be critical.

First, we will find a vector $\overline{x} = (\overline{x}_{1,1}, \overline{x}_{1,2}, \ldots, \overline{x}_{nn_n}) \in T$, which is the closest to the vector $p \in T$ such that $L_s^{\overline{x}} = L_k^{\overline{x}}$ (see condition (a^*)). For the desired vector $\overline{x}$, the value $\sum_{\nu \in \Omega_k^u} l^{\overline{x}}(\nu)$ for each set $\Omega_k^u,\ u = 1, 2, \ldots, \omega_k^T$, has to be not greater than the value $\sum_{\mu \in \Omega_s^v} l^{\overline{x}}(\mu)$ for at least one set $\Omega_s^v,\ v = 1, 2, \ldots, \omega_s^T$. If the opposite inequality holds for the given vector $p \in T$, i.e., if $\sum_{\mu \in \Omega_s^v} l^p(\mu) < \sum_{\nu \in \Omega_k^u} l^p(\nu)$, we can calculate the value

$$r = \frac{\sum_{\nu \in \Omega_k^u} l^p(\nu) - \sum_{\mu \in \Omega_s^v} l^p(\mu)}{\sum_{O_{ij} \in Q} |n_{ij}(\Omega_k^u) - n_{ij}(\Omega_s^v)|} \tag{3.44}$$

(where $n_{ij}(\Omega_k^u)$ is the number of copies of operation O_{ij} in the multiset $\{[\nu] :\ \nu \in \Omega_k^u\}$) in order to obtain vector $\overline{x}$ with

$$\sum_{\mu \in \Omega_s^v} l^{\overline{x}}(\mu) = \sum_{\nu \in \Omega_k^u} l^{\overline{x}}(\nu). \tag{3.45}$$

It is easy to convince that equality (3.45) holds for the vector $\overline{x}$ obtained from the vector p by adding the value r calculated in (3.44) to all components p_{ij} with $n_{ij}(\Omega_k^u) < n_{ij}(\Omega_s^v)$ and by subtracting the same value r from all components p_{ij} with $n_{ij}(\Omega_k^u) > n_{ij}(\Omega_s^v)$. Note that for the above vector $\overline{x}$, the inclusion $\overline{x} \in T$ may not hold. To guarantee this inclusion, we have to look for a vector $\overline{x}$ in the form $\overline{x} = p(r) = (p_{1,1}(r), p_{1,2}(r), \ldots, p_{nn_n}(r))$, where

$$\overline{x}_{ij} = p_{ij}(r) = \begin{cases} p_{ij} + \min\{r, b_{ij} - p_{ij}\}, & \text{if } n_{ij}(\Omega_k^u) < n_{ij}(\Omega_s^v), \\ p_{ij} - \min\{r, p_{ij} - a_{ij}\}, & \text{if } n_{ij}(\Omega_k^u) > n_{ij}(\Omega_s^v), \\ p_{ij}, & \text{if } n_{ij}(\Omega_k^u) = n_{ij}(\Omega_s^v). \end{cases} \tag{3.46}$$

Let $r_{\Omega_k^u, \Omega_s^v}$ denote the minimal distance between the given vector $p \in T$ and the desired vector $\overline{x} = p(r) \in T$ for which equality (3.45) holds: $r_{\Omega_k^u, \Omega_s^v} = d(p, p(r))$. Next, we show how to calculate this value $r_{\Omega_k^u, \Omega_s^v}$. To this end, we define the value

$$\Delta^{ij}(\Omega_s^v, \Omega_k^u) = \begin{cases} b_{ij} - p_{ij}, & \text{if } n_{ij}(\Omega_k^u) < n_{ij}(\Omega_s^v), \\ p_{ij} - a_{ij}, & \text{if } n_{ij}(\Omega_k^u) > n_{ij}(\Omega_s^v), \end{cases} \tag{3.47}$$

for each operation

$$O_{ij} \in N(\Omega_k^u, \Omega_s^v) = \{ \bigcup_{\mu \in \Omega_k^u \cup \Omega_s^v} [\mu] :\ n_{ij}(\Omega_k^u) \ne n_{ij}(\Omega_s^v)\},$$

which we put in non-decreasing order:

$$\Delta_1^{ij}(\Omega_s^v, \Omega_k^u) \le \Delta_2^{ij}(\Omega_s^v, \Omega_k^u) \le \ldots \le \Delta_{|N(\Omega_s^v, \Omega_k^u)|}^{ij}(\Omega_s^v, \Omega_k^u). \tag{3.48}$$

Note that each value $\Delta_\alpha^{ij}(\Omega_s^v, \Omega_k^u)$ is calculated according to (3.47) for all different operations O_{ij}, and the subscript $\alpha = 1, 2, \ldots, |N(\Omega_s^v, \Omega_k^u)|$ indicates the location of value (3.47) in the sequence (3.48). We define value

$$N_\alpha(\Delta) = |n_{ij}(\Omega_k^u) - n_{ij}(\Omega_s^v)|$$

for each $\Delta_\alpha^{ij}(\Omega_s^v, \Omega_k^u), \alpha = 1, 2, \ldots, |N(\Omega_s^v, \Omega_k^u)|$, and we assume that $\Delta_0^{ij}(\Omega_s^v, \Omega_k^u) = 0$ and $N_0(\Delta) = 0$.

From (3.46) and (3.48), it follows that equality (3.49) holds:

$$r_{\Omega_s^v, \Omega_k^u} = \max_{\beta=0,1,\ldots,|N(\Omega_s^v,\Omega_k^u)|-1} \frac{\sum_{\nu\in\Omega_k^u} l^p(\nu) - \sum_{\mu\in\Omega_s^v} l^p(\mu) - \sum_{\alpha=0}^{\beta} \Delta_\alpha^{ij}(\Omega_s^v, \Omega_k^u)\ N_\alpha(\Delta)}{\sum_{O_{ij}\in Q^J} |n_{ij}(\Omega_k^u) - n_{ij}(\Omega_s^v)| - \sum_{\alpha=0}^{\beta} N_\alpha(\Delta)}. \quad (3.49)$$

To ensure equality $L_s^{\overline{x}} = L_k^{\overline{x}}$ for the digraph G_k and the vector $\overline{x} = p(r) \in T$, we have to repeat the calculations (3.46) - (3.49) for each set $\Omega_k^u,\ u \in \{1, 2, \ldots, \omega_k^T\}$, with

$$\sum_{\nu\in\Omega_k^u} l^p(\nu) \geq L_s^p.$$

Then we have to take the maximum of $r_{\Omega_s^v,\Omega_k^u}$, for each set $\Omega_k^u,\ u \in \{1, 2, \ldots, \omega_k^T\}$, and to take the minimum of the maximum obtained:

$$\overline{r}_{ks}^B = \min_{v\in\{1,2,\ldots,\omega_s^T\}} \max_{\substack{u\in\{1,2,\ldots,\omega_k^T\},\\ \sum_{\nu\in\Omega_k^u} l^p(\nu)\geq L_s^p}} r_{\Omega_s^v,\Omega_k^u}. \quad (3.50)$$

Note that, if there exists a vector $\overline{x} \in T$ such that equality $L_s^{\overline{x}} = L_k^{\overline{x}}$ holds (see condition (a^*)), nevertheless it may be that there exists no vector $\overline{p}^\epsilon \in T$ defined in condition (b^*) such that $L_s^{\overline{p}^\epsilon} > L_k^{\overline{p}^\epsilon}$. However, as follows from Definition 3.5, only inequality (3.40) guaranties that digraph G_s does not dominate digraph G_k in the polytope T. Therefore, we look for a vector $\overline{p}^\epsilon \in T$ such that inequality (3.40) holds which may be rewritten in the following equivalent form:

$$\max_{v\in\{1,2,\ldots,\omega_s^T\}} \sum_{\mu\in\Omega_s^v} l^{\overline{p}^\epsilon}(\mu) > \max_{u\in\{1,2,\ldots,\omega_k^T\}} \sum_{\nu\in\Omega_k^u} l^{\overline{p}^\epsilon}(\nu). \quad (3.51)$$

Remark 3.7 It is easy to see that there exists a vector $\overline{p}^\epsilon \in T$ such that

$$\sum_{\mu\in\Omega_s^v} l^{\overline{p}^\epsilon}(\mu) > \sum_{\nu\in\Omega_k^u} l^{\overline{p}^\epsilon}(\nu) \quad (3.52)$$

if and only if inequality (3.52) holds for the vector $\overline{p}^\epsilon = p^* = (p_{1,1}^*, p_{1,2}^*, \ldots, p_{nn_n}^*) \in T$, where

$$p_{ij}^* = \begin{cases} b_{ij}, & \text{if } n_{ij}(\Omega_k^u) < n_{ij}(\Omega_s^v), \\ a_{ij}, & \text{if } n_{ij}(\Omega_k^u) > n_{ij}(\Omega_s^v), \\ p_{ij}, & \text{if } n_{ij}(\Omega_k^u) = n_{ij}(\Omega_s^v). \end{cases} \quad (3.53)$$

Indeed, all components of the vector $p^* \in T$ with $n_{ij}(\Omega_k^u) < n_{ij}(\Omega_s^v)$ are as large as possible in the polytope T and all components with $n_{ij}(\Omega_k^u) > n_{ij}(\Omega_s^v)$ are as small as possible in the polytope T (obviously, changing components with $n_{ij}(\Omega_k^u) = n_{ij}(\Omega_s^v)$

does not influence the difference $\sum_{\nu\in\Omega_k^u} l^x(\nu) - \sum_{\mu\in\Omega_s^v} l^x(\mu)$). Thus, we have to restrict the consideration of the sets Ω_s^v in inequality (3.51) to the subset Ω_{sk}^* of the set $\{\Omega_s^v : v = 1, 2, \ldots, \omega_s^T\}$ defined as follows: Ω_{sk}^* is the set of all sets of representatives Ω_s^v, $v \in \{1, 2, \ldots, \omega_s^T\}$, for which inequality

$$\sum_{\mu\in\Omega_s^v} l^{p^*}(\mu) > \sum_{\nu\in\Omega_k^u} l^{p^*}(\nu) \tag{3.54}$$

holds for each set of representatives Ω_k^u, $u \in \{1, 2, \ldots, \omega_k^T\}$. Thus, sufficiency in Lemma 3.5 may be generalized as follows.

Lemma 3.6 *Digraph $G_s \in B$ dominates digraph $G_k \in B$ in the polytope T if $\Omega_{sk}^* = \emptyset$.*

Proof. Equality $\Omega_{sk}^* = \emptyset$ implies that for each set of representatives Ω_s^v, $v \in \{1, 2, \ldots, \omega_s^T\}$, inequality

$$\sum_{\mu\in\Omega_s^v} l^{p^*}(\mu) \le \sum_{\nu\in\Omega_k^u} l^{p^*}(\nu)$$

holds for each set of representatives Ω_k^u, $u \in \{1, 2, \ldots, \omega_k^T\}$. Thus, due to Remark 3.7, inequality

$$\sum_{\mu\in\Omega_s^v} l^{\overline{p}^\epsilon}(\mu) \le \sum_{\nu\in\Omega_k^u} l^{\overline{p}^\epsilon}(\nu)$$

holds for each vector $\overline{p}^\epsilon \in T$. Hence, $L_s^{\overline{p}^\epsilon} \le L_k^{\overline{p}^\epsilon}$ and so dominance relation $G_s \preceq_T G_k$ holds.

◇

Due to Lemma 3.6, we can rewrite equality (3.50) as follows:

$$\overline{r}_{ks}^B = \min_{\Omega_s^v\in\Omega_{sk}^*} \max_{\substack{u\in\{1,2,\ldots,\omega_k^T\},\\ \sum_{\nu\in\Omega_k^u} l^p(\nu)\ge L_s^p}} r_{\Omega_s^v,\Omega_k^u}. \tag{3.55}$$

To obtain the desired vector $\overline{p}^\epsilon \in T$, we have to calculate $\overline{r}_{ks}^B$ according to (3.55) for each digraph $G_k \in B$ which is not dominated by digraph G_s (if $G_s \not\preceq_T G_k$) and to take the minimum over all such digraphs G_k. We summarize the above arguments in the following claim.

Theorem 3.6 *If for digraph $G_s \in B \subseteq \Lambda(G)$ dominance relation $G_s \preceq_p G_k$ holds for each digraph $G_k \in B$ and fixed vector $p \in T$ of the processing times, then equalities*

$$\begin{aligned} \overline{\varrho}_s^B(p \in T) &= \min\{ \min_{\Omega_s^v\in\Omega_{sk}^*} \max_{\substack{u\in\{1,2,\ldots,\omega_k^T\},\\ \sum_{\nu\in\Omega_k^u} l^p(\nu)\ge L_s^p}} r_{\Omega_s^v,\Omega_k^u} \ :\ G_s \not\preceq_T G_k\} \\ &= \min\{\overline{r}_{ks}^B :\ G_s \not\preceq_T G_k\} \end{aligned} \tag{3.56}$$

hold, where value $r_{\Omega_s^v,\Omega_k^u}$ is calculated according to (3.49).

The following corollary is used in the proof of Theorem 3.8.

Corollary 3.7 *The value $r_{\Omega_s^{v^0},\Omega_k^u}$ calculated according to (3.49) for the set $\Omega_s^{v^0} \in \Omega_{sk}^* \setminus \Omega_s(p)$ is strongly positive.*

PROOF. Due to formula (3.56), we have to repeat the calculation (3.49) for each set $\Omega_s^v \in \Omega_{sk}^*$ and each set $\Omega_k^{u^0}, u^0 \in \{1,2,\ldots,\omega_k^T\}$, such that $\sum_{\nu\in\Omega_k^{u^0}} l^p(\nu) \geq L_s^p$. Since there exists a set $\Omega_s^{v^0} \in \Omega_{sk}^* \setminus \Omega_s(p)$, i.e., $\sum_{\mu\in\Omega_s^{v^0}} l^p(\mu) < L_s^p$, the inequalities

$$\begin{aligned}
&\sum_{\nu\in\Omega_k^{u^0}} l^p(\nu) - \sum_{\mu\in\Omega_s^{v^0}} l^p(\mu) \\
\geq\ & \min_{\substack{u^0\in\{1,2,\ldots,\omega_k^T\},\\ \sum_{\nu\in\Omega_k^{u^0}} l^p(\nu)\geq L_s^p}} \sum_{\nu\in\Omega_k^{u^0}} l^p(\nu) - \max_{\substack{v^0\in\{1,2,\ldots,\omega_s^T\},\\ \sum_{\nu\in\Omega_s^{v^0}} l^p(\mu)< L_s^p}} \sum_{\mu\in\Omega_s^{v^0}} l^p(\mu) \\
\geq\ & L_s^p - \max_{\substack{v^0\in\{1,2,\ldots,\omega_s^T\},\\ \sum_{\nu\in\Omega_s^{v^0}} l^p(\mu)< L_s^p}} \sum_{\mu\in\Omega_s^{v^0}} l^p(\mu) > 0
\end{aligned}$$

hold. Therefore, due to the calculation of the value $r_{\Omega_s^{v^0},\Omega_k^{u^0}}$, the numerator in (3.49) is strongly positive at least for $\beta = 0$. Since we have to take the maximum value among all values calculated for each $\beta = 0,1,\ldots,|N(\Omega_s^{v^0},\Omega_k^{u^0})| - 1$ (see formula (3.49)), we get $r_{\Omega_s^{v^0},\Omega_k^{u^0}} > 0$.

◇

Next, we present necessary and sufficient conditions for an infinitely large relative stability radius $\overline{\varrho}_s^B(p \in T)$ for problem $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$ if $B \subset \Lambda(G)$ and $T \subset R_+^q$. Note that deterministic problem $\mathcal{J}//\sum \mathcal{C}_i$ with $\lambda > 1$ cannot have an optimal digraph with an infinitely large stability radius $\overline{\varrho}_s(p)$ (see Remark 1.1). Recall that $\overline{\varrho}_s(p) = \overline{\varrho}_s^{\Lambda(G)}(p \in R_+^q)$.

Theorem 3.7 *For digraph $G_s \in B \subseteq \Lambda(G)$, we have $\overline{\varrho}_s^B(p \in T) = \infty$ if and only if $\Omega_{sk}^* = \emptyset$ for each digraph $G_k \in B$.*

PROOF. *Necessity.* Following the contradiction method, we suppose that $\overline{\varrho}_s^B(p \in T) = \infty$ but there exists a digraph $G_k \in B$ such that the set of representatives $\Omega_s^{v^0}, v^0 \in \{1,2,\ldots,\omega_s^T\}$, belongs to the set Ω_{sk}^*. It follows that the inequality

$$\sum_{\mu\in\Omega_s^{v^0}} l^{p^*}(\mu) > \sum_{\nu\in\Omega_k^u} l^{p^*}(\nu)$$

holds for the vector p^* calculated according to formula (3.53) for the set of representatives $\Omega_k^u, u \in \{1,2,\ldots,\omega_k^T\}$. Thus, due to Remark 3.7 there exists a vector $p' \in T$ such that inequality

$$\sum_{\mu\in\Omega_s^{v^0}} l^{p'}(\mu) > \sum_{\nu\in\Omega_k^u} l^{p'}(\nu)$$

holds. Since this inequality holds for all sets Ω_k^u, $u \in \{1,2,\ldots,\omega_k^T\}$, this inequality holds for a critical set $\Omega_k^{u^*} \in \Omega_k(p)$ as well. Therefore, we obtain

$$\sum_{\mu\in\Omega_s^{v^0}} l^{p'}(\mu) > \sum_{\nu\in\Omega_k^{u^*}} l^{p'}(\nu) = L_k^{p'}$$

and hence digraph G_s cannot be optimal for the processing times given by vector $p' \in T$. We get a contradiction:

$$\overline{\varrho}_s^B(p \in T) < d(p, p') \leq \max_{O_{ij} \in Q^J} \{b_{ij} - p_{ij}, p_{ij} - a_{ij}\} < \infty.$$

Sufficiency. Due to Lemma 3.6, equality $\Omega_{sk}^* = \emptyset$ (valid for each digraph $G_k \in B$) implies that digraph $G_s \in B$ dominates all digraphs $G_k \in B$ in polytope T. Hence, inequality $L_s^{p'} \leq L_k^{p'}$ holds for each vector $p' \in T$ and so $\overline{\varrho}_s^B(p \in T) = \infty$.

$\diamond$

From the above proof of the necessity, we obtain an upper bound for the relative stability radius $\overline{\varrho}_s^B(p \in T)$.

Corollary 3.8 *If $\overline{\varrho}_s^B(p \in T) < \infty$, then $\overline{\varrho}_s^B(p \in T) \leq \max\{\{b_{ij} - p_{ij}, p_{ij} - a_{ij}\} : \ O_{ij} \in Q^J\}$.*

Moreover, we can strengthen Corollary 3.6 as follows.

Corollary 3.9 *The following propositions are equivalent:*

1) $\Lambda^T(G) = \{G_s\}$,

2) $\overline{\varrho}_s^{\Lambda(G)}(p \in T) = \infty$,

3) $G_k \in \Lambda(G) \ \Rightarrow \ G_s \preceq_T G_k$,

4) $G_k \in \Lambda(G) \ \Rightarrow \ \Omega_{sk}^* = \emptyset$.

To present necessary and sufficient conditions for $\overline{\varrho}_s^B(p \in T) = 0$, we need the following auxiliary lemma. Let Ω_k denote the set $\{\Omega_k^u : \ u = 1, 2, \ldots, \omega_k\}$.

Lemma 3.7 *If $\Omega_k \neq \Omega_k(p)$, the inclusion $\Omega_k(p') \subseteq \Omega_k(p)$ holds for any vector $p' \in O_\epsilon(p) \cap R_+^q$ with $\overline{\epsilon}_k > \epsilon > 0$ defined as follows:*

$$\overline{\epsilon}_k = \frac{1}{qn} \min\Big\{L_k^p - \sum_{\nu \in \Omega_k^u} l^p(\nu) : \ \Omega_k^u \in \Omega_k \backslash \Omega_k(p)\Big\}. \tag{3.57}$$

The following theorem is a generalization of Theorem 1.6.

Theorem 3.8 *Let G_s be an optimal digraph of problem $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$ with the minimal objective function value L_s^p, $p \in T$, within the set $B \subseteq \Lambda(G)$ of feasible digraphs. The equality $\overline{\varrho}_s^B(p \in T) = 0$ holds if and only if the following three conditions hold:*
1) there exists a digraph $G_k \in B$ such that $L_s^p = L_k^p$, $k \neq s$,
2) the set $\Omega_s^{v^*} \in \Omega_{sk}^* \cap \Omega_s(p)$ is such that for any set $\Omega_k^{u^*} \in \Omega_k(p)$, there exists an operation $O_{ij} \in Q^J$ for which condition

$$n_{ij}(\Omega_s^{v^*}) \geq n_{ij}(\Omega_k^{u^*}) \tag{3.58}$$

holds (or condition

$$n_{ij}(\Omega_s^{v^*}) \leq n_{ij}(\Omega_k^{u^*}) \tag{3.59}$$

holds) and
3) inequality (3.58) (or inequality (3.59), respectively) is satisfied as a strict one for at least one set $\Omega_k^{u^0} \in \Omega_k(p)$: $n_{ij}(\Omega_s^{v^*}) > n_{ij}(\Omega_k^{u^0})$ (or $n_{ij}(\Omega_s^{v^*}) < n_{ij}(\Omega_k^{u^0})$).

PROOF. *Necessity.* We prove necessity by contradiction. Assume that $\overline{\varrho}_s^B(p \in T) = 0$ but the conditions of the theorem are not satisfied. We consider four cases *i*, *ii*, *iii* and *iv* of violating these conditions.

i) Assume that there does not exist another optimal digraph $G_k \in B$ such that $L_s^p = L_k^p, k \neq s$. If $B \setminus \{G_s\} \neq \emptyset$, we can calculate the value

$$\overline{\epsilon}^* = \frac{1}{qn} \min_{t \neq s}(L_t^p - L_s^p), \tag{3.60}$$

which is strictly positive since $L_s^p < L_t^p$ for each $G_t \in B, t \neq s$. Using Lemma 3.7, one can verify that for any real ϵ, which satisfies the inequalities $0 < \epsilon < \overline{\epsilon}^*$, the difference in the right-hand side of equality (3.60) remains positive when vector p is replaced by any vector $p^0 \in O_\epsilon(p) \cap T$. Indeed, for any $v \in \{1, 2, \ldots, \omega_s^T\}$, the cardinality of the set Ω_s^v may be at most equal to qn: $|\Omega_s^v| \leq qn$. Thus, the difference $L_t^p - L_s^p = L_t^p - \max_{v \in \{1,2,\ldots,\omega_s^T\}} \sum_{\nu \in \Omega_s^v} l^p(\nu)$ may not be 'overcome' by a vector p^0 if $d(p, p^0) < \overline{\epsilon}^*$. Hence, we conclude that digraph G_s remains optimal for any vector $p^0 = (p_{11}^0, p_{12}^0, \ldots, p_{nn_n}^0) \in T$ of the processing times provided that $d(p, p^0) \leq \epsilon < \overline{\epsilon}^*$. Therefore, we have $\overline{\varrho}_s^B(p \in T) \geq \overline{\epsilon}^* > \epsilon > 0$ which contradicts the assumption $\overline{\varrho}_s^B(p \in T) = 0$.

ii) Assume that there exists a digraph $G_k \in B$ such that $L_s^p = L_k^p, k \neq s$, and $\Omega_{sk}^* \cap \Omega_s(p) = \emptyset$. Note that $\Omega_{st}^* \neq \emptyset$ for all digraphs $G_t \in B, t \neq s$. Otherwise, we get $\overline{\varrho}_s^B(p \in T) = \infty$ due to Theorem 3.7.

Assume that there exists a set $\Omega_{st}^* \neq \emptyset$ for the digraphs G_s and G_t with $L_t^p > L_s^p$, i.e. there exists a set $\Omega_s^{v'} \in \Omega_{st}^*$. Similarly to the proof of Corollary 3.7, we can show that all values $r_{\Omega_s^{v'}, \Omega_t^u}$ calculated for each digraph G_t with $L_t^p > L_s^p$ cannot be equal to zero: We obtain a strongly positive numerator in formula (3.49) at least for $\beta = 0$:

$$\sum_{\nu \in \Omega_t^u} l^p(\nu) - \sum_{\mu \in \Omega_s^{v'}} l^p(\mu) > 0.$$

Therefore, the maximum taken according to (3.49) is also strongly positive, i.e. $r_{\Omega_s^{v'}, \Omega_t^u} > \epsilon > 0$, where we can choose any ϵ such that the inequality

$$\epsilon < \min\Big\{\overline{\epsilon}_s, \overline{\epsilon}_k, \frac{1}{qn} \min_{\substack{G_t \in B, \\ L_t^p > L_s^p}} (L_t^p - L_s^p)\Big\} \tag{3.61}$$

is satisfied. This means that only in the case of the calculation of the value $r_{\Omega_s^v, \Omega_k^u}$ for the optimal digraphs $G_k \in B, L_k^p = L_s^p$, with $\Omega_{sk}^* \neq \emptyset$ we can obtain $r_{\Omega_s^v, \Omega_k^u} = 0$.

Assume that there exists a set $\Omega_{sk}^* \neq \emptyset$ for the digraphs G_s and G_k with $L_k^p = L_s^p$, i.e. there exists a set $\Omega_s^{v''} \in \Omega_{sk}^*$. In this case, we can set

$$\epsilon' = \min\Big\{\overline{\epsilon}_s, \overline{\epsilon}_k, \frac{1}{qn} \min\Big\{L_s^p - \max \sum_{\mu \in \Omega_s^v} l^p(\mu) : \sum_{\nu \in \Omega_s^v} l^p(\mu) < L_s^p\Big\}\Big\}.$$

Taking into account our assumption that for each digraph $G_k \in B, L_s^p = L_k^p, k \neq s$, the set $\Omega_{sk}^* \cap \Omega_s(p)$ is empty, it follows from the proof of Corollary 3.7 that $r_{\Omega_s^{v''}, \Omega_k^u} > \epsilon' > 0$.

Hence, for all digraphs $G_t, L_t^p \geq L_s^p$, inequality $\overline{r}_{ts}^B > \min\{\epsilon, \epsilon'\}$ holds, where the value $\overline{r}_{ts}^B$ is calculated due to formula (3.55) using the value $r_{\Omega_s^v, \Omega_t^u} > 0$. Therefore, the relative

stability radius satisfies the following inequalities: $\overline{\varrho}_s^B(p \in T) > \min\{\epsilon, \epsilon'\} > 0$, which contradicts the above assumption $\overline{\varrho}_s^B(p \in T) = 0$.

iii) Assume that there exists a digraph $G_k \in B$ such that $L_s^p = L_k^p, k \neq s$, and for any set $\Omega_s^{v^*} \in \Omega_{sk}^* \cap \Omega_s(p)$ there exists a set $\Omega_k^{u^*} \in \Omega_k(p)$ such that $n_{ij}(\Omega_s^{v^*}) = n_{ij}(\Omega_k^{u^*})$ for any operation $O_{ij} \in Q^J$.

In this case, we can take any ϵ that satisfies the inequality (3.61). Due to $\epsilon < \overline{\epsilon}_s$, we get from Lemma 3.7 that equality

$$L_s^{p^0} = \max_{\Omega_s^{v^*} \in \Omega_s(p^0)} \sum_{\mu \in \Omega_s^{v^*}} l^{p^0}(\mu) = \max_{\Omega_s^{v^*} \in \Omega_s(p)} \sum_{\mu \in \Omega_s^{v^*}} l^{p^0}(\mu) \tag{3.62}$$

holds for any vector $p^0 \in O_\epsilon(p) \cap T$.

On the other hand, since there exists a set $\Omega_k^{u^*} \in \Omega_k(p)$ such that $n_{ij}(\Omega_s^{v^*}) = n_{ij}(\Omega_k^{u^*})$, $O_{ij} \in Q$, for any set $\Omega_s^{v^*} \in \Omega_{sk}^* \cap \Omega_s(p)$ and for any digraph $G_k, L_s^p = L_k^p$, we obtain the inequality

$$\max_{\Omega_s^{v^*} \in \Omega_s(p)} \sum_{\mu \in \Omega_s^{v^*}} l^{p^0}(\mu) \leq \max_{\Omega_k^u \in \Omega_k(p)} \sum_{\nu \in \Omega_k^{u^*}} l^{p^0}(\nu),$$

because of $\epsilon < \overline{\epsilon}_s$ and $\epsilon < \overline{\epsilon}_k$. Therefore, due to (3.62) we have

$$L_s^{p^0} \leq \max_{\Omega_k^u \in \Omega_k(p)} \sum_{\nu \in \Omega_k^{u^*}} l^{p^0}(\nu) \tag{3.63}$$

for any optimal digraph $G_k,\ k \neq s$. Since

$$\epsilon < \frac{1}{qn} \min_{L_t^p \neq L_s^p} \{L_t^p - L_s^p\},$$

the condition $L_t^p \neq L_s^p$ implies $L_t^{p^0} \neq L_s^{p^0}$. So taking into account (3.61) and the latter implication, we conclude that the digraph G_s becomes an optimal digraph for any vector $p^0 \in T$, provided that $d(p, p^0) \leq \epsilon$. Consequently, we have $\overline{\varrho}_s^B(p \in T) \geq \epsilon > 0$, which contradicts the assumption $\overline{\varrho}_s^B(p \in T) = 0$.

iv) Assume that conditions 1 and 2 of Theorem 3.8 hold. More exactly, there exists a digraph $G_k \in B$ such that $L_s^p = L_k^p, k \neq s$, and one of the two cases of condition 2 and one of the two cases of condition 3 hold. Assume that for any set $\Omega_s^{v^*} \in \Omega_{sk}^* \cap \Omega_s(p)$, there exists a set $\Omega_k^{u^*} \in \Omega_k(p)$ such that for any operation $O_{ij} \in Q^J$ with $n_{ij}(\Omega_s^{v^*}) > n_{ij}(\Omega_k^{u^*})$, there exists a set $\Omega_k^{u^0} \in \Omega_k(p)$ with $n_{ij}(\Omega_s^{v^*}) < n_{ij}(\Omega_k^{u^0})$.

Arguing in the same way as in case *iii*, we can show that $\overline{\varrho}_s^B(p \in T) \geq \epsilon > 0$, where ϵ is as in (3.61), since for any vector $p^0 \in O_\epsilon(p) \cap T$, the value $\sum_{\mu \in \Omega_s^{v^*}} l^{p^0}(\mu)$ is less than or equal to the value $\sum_{\nu \in \Omega_k^{u^*}} l^{p^0}(\nu)$ or to the value $\sum_{\nu \in \Omega_k^{u^0}} l^{p^0}(\nu)$.

Sufficiency. We show that, if the conditions of Theorem 3.8 are satisfied, then $\overline{\varrho}_s^B(p \in T) < \epsilon$ for any given $\epsilon > 0$. First, we make the following remark.

Remark 3.8 In the trivial case of $a_{ij} = b_{ij}$ for each operation $O_{ij} \in Q^J$, the set $\Omega_{sk}^* \cap \Omega_s(p)$ is empty, since in this case the vector p is equal to the vector p^* constructed according to (3.53), and the strong inequality (3.54) does not hold.

We construct a vector $p' = (p'_{1,1}, p'_{1,2}, \ldots, p'_{nn_n}) \in T$ with components $p'_{ij} \in \{p_{ij}, p_{ij} + \epsilon', p_{ij} - \epsilon'\}$, where $\epsilon' = \min\{\epsilon, \overline{\epsilon}_k, \overline{\epsilon}_{min}\}$ with the value $\overline{\epsilon}_k > 0$ defined in (3.57), and

$$\overline{\epsilon}_{min} = \max\{0,\ \min\{\min\{p_{ij} - a_{ij} :\ p_{ij} > a_{ij}, O_{ij} \in Q^J\},\ \min\{b_{ij} - p_{ij} :\ b_{ij} > p_{ij}, O_{ij} \in Q^J\}\}\},$$

using the following rule: For each $\Omega_k^{u^*} \in \Omega_k(p)$, mentioned in Theorem 3.8, we set $p'_{ij} = p_{ij} + \epsilon'$, if inequalities (3.58) hold, or we set $p'_{ij} = p_{ij} - \epsilon'$, if inequalities (3.59) hold.

More precisely, we can choose ϵ' as follows: If $\Omega_k \neq \Omega_k(p)$, then $\overline{\epsilon}_k > 0$, and we can choose ϵ' such that $0 < \epsilon' < \min\{\epsilon, \overline{\epsilon}_k, \overline{\epsilon}_{min}\}$. Otherwise, if $\Omega_k = \Omega_k(p)$, we choose ϵ' such that $0 < \epsilon' < \min\{\overline{\epsilon}, \overline{\epsilon}_{min}\}$. Such choices are possible since in both above cases, inequality $\overline{\epsilon}_{min} > 0$ holds due to Remark 3.8. Note that $\epsilon' > 0$ since $p_{ij} > 0, O_{ij} \in Q^J$. The following arguments are the same for both cases of the choice of ϵ'.

After changing at most $|\Omega_k(p)|$ components of the vector p according to this rule, we obtain a vector p' of the processing times for which inequality

$$\sum_{\mu \in \Omega_s^{v^*}} l^{p'}(\mu) > \sum_{\nu \in \Omega_k^{u^*}} l^{p'}(\nu)$$

holds for each set $\Omega_k^{u^*} \in \Omega_k(p)$. Due to $\epsilon' \leq \overline{\epsilon}_{min}$, we have $p' \in T$. Since $\epsilon' \leq \overline{\epsilon}_k$, we have

$$\begin{aligned} L_k^{p'} &= \max_{u \in \{1,2,\ldots,\omega_k^T\}} \sum_{\nu \in \Omega_k^u} l^{p'}(\nu) = \max_{\Omega_k^u \in \Omega_k(p)} \sum_{\nu \in \Omega_k^u} l^{p'}(\nu) \\ &= \sum_{\nu \in \Omega_k^{u^*}} l^{p'}(\nu) \ < \ \sum_{\mu \in \Omega_s^{v^*}} l^{p'}(\mu) \ \leq \ L_s^{p'}. \end{aligned}$$

Thus, we conclude that digraph G_s is not optimal for the vector $p' \in T$ with $d(p, p') = \epsilon'$ which implies $\overline{\varrho}_s^B(p \in T) < \epsilon' \leq \epsilon$.

$\diamond$

Corollary 3.10 *If $G_s \in B$ is a unique optimal digraph for the vector $p \in T$, then $\overline{\varrho}_s^B(p \in T) > 0$.*

From Theorem 3.8 we obtain the following lower bound for the relative stability radius $\overline{\varrho}_s^B(p \in T)$.

Corollary 3.11 *If $G_s \in B$ is an optimal digraph, then $\overline{\varrho}_s^B(p \in T) \geq \overline{\epsilon}^*$, where $\overline{\epsilon}^*$ is calculated according to (3.60).*

PROOF. If there exists a digraph $G_k \in B$ such that $L_s^p = L_k^p$, $k \neq s$, the equality $\overline{\varrho}_s^B(p \in T) \geq \overline{\epsilon}^* = 0$ holds due to Definition 3.6. Otherwise, inequality $\overline{\varrho}_s^B(p \in T) \geq \overline{\epsilon}^*$ follows from the above proof of necessity (see case *i*).

$\diamond$

Example 3.2 (continued). *Returning to the Example 3.2 and using Theorem 3.6, we can calculate the relative stability radius of the digraph $G_1 \in B \subseteq \Lambda(G), |B| = 12$, for the vector $p = p^0 = (70, 30, 60, 20, 60, 70, 40, 30)$ of the processing times according to formula (3.56). After a pairwise comparison of the sets of representatives for digraph G_1^T with*

those for digraphs $G_2^T, G_3^T, \ldots, G_{12}^T$, we obtain equality $\overline{\varrho}_1^B(p^0 \in T) = 3$, which means that digraph G_1 remains optimal at least for all vectors $p \in O_3(p^0) \cap T$ of the processing times. Due to calculation of the relative stability radius, we show that only digraphs G_2 and G_5 may be better than digraph G_1 provided that vector p of the processing times belongs to polytope T, and for each digraph $G_k \in \Lambda(G)$ with $k \neq 2$ and $k \neq 5$, dominance relation $G_1 \preceq_T G_k$ holds. We also obtain the following equalities: $\overline{\varrho}_1^B(p^0 \in T) = \overline{r}_{2,1}^B = 3$, $\overline{\varrho}_1^{B\setminus\{G_2\}}(p^0 \in T) = \overline{r}_{5,1}^{B\setminus\{G_2\}} = 10$, where the values $\overline{r}_{k1}^B$ are calculated according to (3.55), and they give the minimum over all digraphs $G_k \in B$ which are not dominated by digraph G_1. Next, it follows from Theorem 3.7 that $\overline{\varrho}_1^{B\setminus\{G_2,G_5\}}(p^0 \in T) = \infty$.

Due to Theorem 3.4, set $\Lambda^*(G) = \{G_1, G_2, G_5\}$ is a solution to problem $\mathcal{J}3/n = 3, a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$, since there exists a covering of the polytope T by sets $D_s = \{p \in R_+^8 : L_s^p \leq L_k^p, k = 1, 2 \ldots, \lambda\} \cap T$ with $s \in \{1, 2, 5\}$. More exactly, for any digraph $G_k \in \Lambda(G)$ and for any set D_s, $s \in \{1, 2, 5\}$, there exists a digraph $G_s \in \Lambda^*(G)$ for which dominance relation $G_s \preceq_{D_s} G_k$ holds (since the dominance relation $G_1 \preceq_T G_k$ holds for each digraph $G_k \in \Lambda(G), k \neq 2, k \neq 5$, it follows that set $\{D_1, D_2, D_5\}$ is indeed a covering of the polytope T). Moreover, since for each digraph $G_s \in \Lambda^*(G)$ there exists a point (see vectors p^0, $\overline{p}^\epsilon$ and $\overline{x}$, given in Section 3.4), for which this digraph is the unique optimal one, it follows from Theorem 3.5 that solution $\Lambda^*(G) = \{G_1, G_2, G_5\}$ is minimal.

Note that from a practical point of view, it is more useful to consider a covering of the polytope T by nested balls $O_3(p^0), O_{10}(p^0)$ and $O_{r^*}(p^0)$, where r^* may be any real number no less than $\max\{b_{ij} - p_{ij}^0, p_{ij}^0 - a_{ij} : i = 1, 2, \ldots, n;\ j = 1, 2, \ldots, n_i\}$. Indeed, due to the calculation of the stability radius $\overline{\varrho}_1^B(p^0 \in T)$, we know that for each vector $p \in O_3(p^0)$ digraph G_1 is optimal. Moreover, for each vector $p \in O_{10}(p^0)$ at least one digraph G_1 or G_2 is optimal since $\overline{\varrho}_1^{B\setminus\{G_2\}}(p^0 \in T) = 10$. Finally, for each vector $p \in O_{r^*}(p^0)$ at least one digraph G_1, G_2 or G_5 is optimal since $\overline{\varrho}_1^{B\setminus\{G_2,G_5\}}(p^0 \in T) = \infty$.

Table 3.8: *Solution of problem $\mathcal{J}3/n=3, a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$ with the initial vector $p^0 \in T$*

i	Set B	$\overline{\varrho}_1^B(p^0 \in T)$	Set Γ_i of competitive digraphs of digraph G_1
1	$B = \{G_1, G_2, \ldots, G_{12}\}$	3	$\{G_2\}$
2	$B \setminus \{G_2\}$	10	$\{G_5\}$
3	$B \setminus \{G_2, G_5\}$	∞	$\emptyset$

Remark 3.9 Solving problem $\mathcal{J}3/n = 3, a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$ takes three iterations by the above algorithm (see Table 3.8). But similarly to the calculation of the relative stability radius and the construction of a solution of the scheduling problem with the makespan criterion (see Remark 3.4), we can construct a solution $\Lambda^*(G)$ for the mean flow time criterion in one scan as follows.

We union one of the optimal digraphs G_s with all digraphs $G_k, k \neq s$, for which a nonempty set $\Omega_{sk}^* \neq \emptyset$ exists, i.e., for which the dominance relation $G_s \preceq_T G_k$ does not hold, and the union of these digraphs composes such a solution $\Lambda^*(G)$. In other words,

a solution of problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$ *is the union of an optimal digraph and of all its competitive digraphs* $\Lambda^*(G) = \{G_s\} \cup \{G_k : \ G_s \not\preceq_T G_k\} = \{G_s\} \cup \{\cup_{i=1}^{I} \Gamma_i\}$, *where* Γ_i *is the set of competitive digraphs of digraph* G_s *with respect to the set* B *in the iteration* $i = 1, 2, \ldots, I$.

Next, we consider a small problem $\mathcal{J}3/n{=}2/\sum \mathcal{C}_i$ to illustrate the calculation of $\overline{\varrho}_1(p)$ by formulas (1.36) and (1.37). Then we calculate the relative stability radius for problem $\mathcal{J}3/n{=}2, a_i \le p_i \le b_i/\sum \mathcal{C}_i$. Notice that we use the same notations for different examples: Example 3.1 and Example 3.2.

Example 3.1 (continued). *Returning to the Example 3.1 from Section 3.1, we consider the job shop problem with the mean flow time criterion* $\mathcal{J}3/n{=}2/\sum \mathcal{C}_i$, *whose input data are given by the weighted mixed graph* $G(p)$ *with* $p = (75, 50, 40, 60, 55, 30)$, *presented in Figure 3.1. Obviously, the set of all feasible digraphs* $\Lambda(G)$ *is the same, but we number these digraphs in non-decreasing order of the objective function values:* $L_1^p \le L_2^p \le \ldots \le L_5^p$ *(see Figure 3.8). Note that for criterion* $\sum \mathcal{C}_i$ *we do not need to use dummy operations. As we can see, digraph* $G_1(p)$ *is optimal for both criteria* $\mathcal{C}_{max}$ *and* $\sum \mathcal{C}_i$. *Next, we calculate stability radius* $\overline{\varrho}_1(p)$ *for digraph* $G_1(p)$.

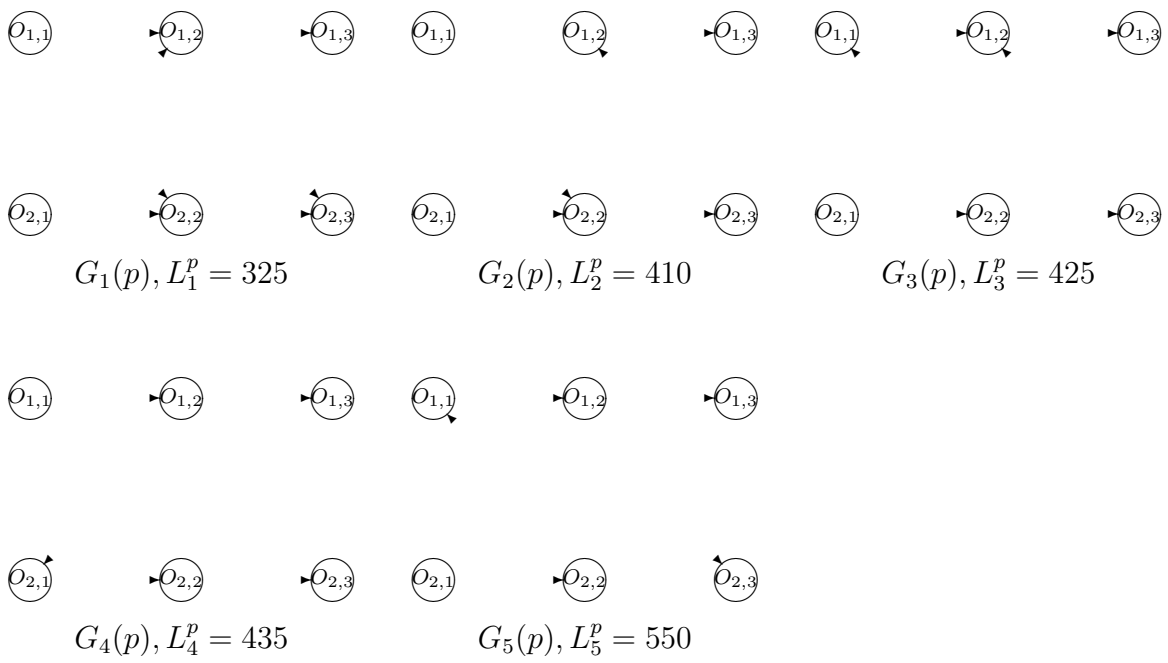

Figure 3.8: *Digraphs* $\Lambda(G) = \{G_1, G_2, \ldots, G_5\}$ *numbered in non-decreasing order of the objective function values* $\sum \mathcal{C}_i$

To this end, we construct an auxiliary Table 3.9, where for each feasible digraph $G_k, k = 1, 2, \ldots, 5$, *column 2 presents the sets* Ω_k^u *of representatives of the family of sets* $(H_k^i)_{J_i \in J}$, *column 3 presents the integer vector* $n(\Omega_k^u) = (n_{1,1}(\Omega_k^u), n_{1,2}(\Omega_k^u), \ldots, n_{2,3}(\Omega_k^u))$, *where the value* $n_{ij}(\Omega_k^u)$ *is equal to the number of vertices* O_{ij} *in the multiset* $\{[\nu] : \nu \in \Omega_k^u\}$

Table 3.9: *Auxiliary information for problem* $\mathcal{J}3/n=2/\sum \mathcal{C}_i$

G_k	$\Omega_k^u, u = 1, 2, \ldots, \omega_k$	$n_{1,1}$	$n_{1,2}$	$n_{1,3}$	$n_{2,1}$	$n_{2,2}$	$n_{2,3}$	$\sum_{\nu\in\Omega_k^u} l^p(\nu)$
1	2	3						4
G_1	$\Omega_1^1=\{O_{1,1}, O_{1,2}, O_{1,3}; O_{1,1}, O_{1,2}, O_{2,3}\}$	2	2	1	0	0	1	320
	$\Omega_1^2=\{O_{1,1}, O_{1,2}, O_{1,3}; O_{2,1}, O_{1,2}, O_{2,3}\}$	1	2	1	1	0	1	305
	$\Omega_1^3=\{O_{1,1}, O_{1,2}, O_{1,3}; O_{1,1}, O_{2,2}, O_{2,3}\}$	2	1	1	0	1	1	325
	$\Omega_1^4=\{O_{1,1}, O_{1,2}, O_{1,3}; O_{2,1}, O_{2,2}, O_{2,3}\}$	1	1	1	1	1	1	310
	$\Omega_1^5=\{O_{2,1}, O_{1,2}, O_{1,3}; O_{1,1}, O_{1,2}, O_{2,3}\}$	1	2	1	1	0	1	305
	$\Omega_1^6=\{O_{2,1}, O_{1,2}, O_{1,3}; O_{2,1}, O_{1,2}, O_{2,3}\}$	0	2	1	2	0	1	290
	$\Omega_1^7=\{O_{2,1}, O_{1,2}, O_{1,3}; O_{1,1}, O_{2,2}, O_{2,3}\}$	1	1	1	1	1	1	310
	$\Omega_1^8=\{O_{2,1}, O_{1,2}, O_{1,3}; O_{2,1}, O_{2,2}, O_{2,3}\}$	0	1	1	2	1	1	295
G_2	$\Omega_2^1=\{O_{1,1}, O_{2,2}, O_{2,3}, O_{1,2}, O_{1,3}; O_{1,1}, O_{2,2}, O_{2,3}\}$	2	1	1	0	2	2	410
	$\Omega_2^2=\{O_{1,1}, O_{2,2}, O_{2,3}, O_{1,2}, O_{1,3}; O_{2,1}, O_{2,2}, O_{2,3}\}$	1	1	1	1	2	2	395
	$\Omega_2^3=\{O_{2,1}, O_{2,2}, O_{2,3}, O_{1,2}, O_{1,3}; O_{1,1}, O_{2,2}, O_{2,3}\}$	1	1	1	1	2	2	395
	$\Omega_2^4=\{O_{2,1}, O_{2,2}, O_{2,3}, O_{1,2}, O_{1,3}; O_{2,1}, O_{2,2}, O_{2,3}\}$	0	1	1	2	2	2	380
G_3	$\Omega_3^1=\{O_{2,1}, O_{2,2}, O_{1,1}, O_{1,2}, O_{1,3}; O_{2,1}, O_{2,2}, O_{2,3}\}$	1	1	1	2	2	1	425
	$\Omega_3^2=\{O_{2,1}, O_{2,2}, O_{2,3}, O_{1,2}, O_{1,3}; O_{2,1}, O_{2,2}, O_{2,3}\}$	0	1	1	2	2	2	380
G_4	$\Omega_4^1=\{O_{1,1}, O_{1,2}, O_{1,3}; O_{1,1}, O_{1,2}, O_{2,1}, O_{2,2}, O_{2,3}\}$	2	2	1	1	1	1	435
G_5	$\Omega_5^1=\{O_{2,1}, O_{2,2}, O_{1,1}, O_{1,2}, O_{1,3}; O_{2,1}, O_{2,2}, O_{1,1}, O_{1,2}, O_{2,3}\}$	2	2	1	2	2	1	550

(for simplicity we use notation n_{ij} *instead of* $n_{ij}(\Omega_k^u)$*), and column 4 presents the value*

$$\sum_{\nu\in\Omega_k^u} l^p(\nu) = \sum_{O_{ij}\in[\nu],\ \nu\in\Omega_k^u} p_{ij} \ \cdot\ n_{ij}(\Omega_k^u).$$

The calculation of $\overline{\varrho}_1(p)$ *by formula (1.37) is given in Table 3.10, which presents the results of the computations for each* $\beta = 1, 2, \ldots, q-m$*, where* m *is the number of operations* $O_{ij} \in \Omega_1^v \cup \Omega_k^u$*,* $\Omega_1^v \in \Omega_{1k}$*, for which* $n_{ij}(\Omega_1^v) < n_{ij}(\Omega_k^u)$*. The cardinality of the set* $\Omega_{1k}, k = 1, 2, \ldots, 5$*, and the elements* Ω_1^ν *of this set are presented in column 2 and column 3, respectively. The elements of the set* $\Omega_k^u, u = 1, 2, \ldots, \omega_k$*, for which* $\sum_{\nu\in\Omega_k^u} l^p(\nu) \geq L_1^p = 325$ *are presented in column 4.*

Since the vector $n(\Omega_k^u) = (n_{1,1}(\Omega_k^u), n_{1,2}(\Omega_k^u), \ldots, n_{2,3}(\Omega_k^u))$ *is the same for both sets* Ω_1^2 *and* Ω_1^5*, for both sets* Ω_1^4 *and* Ω_1^7*, and for both sets* Ω_2^2 *and* Ω_2^3 *(see Table 3.9), the results calculated by formula (1.37) are the same for these pairs of sets, too. Therefore, we combine these calculations in column 7 in Table 3.10. In column 6 we give the sequence of processing times of the operations* $O_{ij} \in \Omega_1^v \cup \Omega_k^u$ *with* $n_{ij}(\Omega_1^v) < n_{ij}(\Omega_k^u)$ *ordered in the following way:* $p_{ij_{(m+1)}} \geq p_{ij_{(m+2)}} \geq \ldots \geq p_{ij_{(q)}}$*. Note that in column 7 we do not write components with* $n_{ij}(\Omega_1^v) = n_{ij}(\Omega_k^u)$ *in the fraction from formula (1.37). For the sets* Ω_1^1 *and* Ω_2^1*, we give a more detailed computation and for each other pair of the sets* Ω_1^v *and* Ω_k^u *at each following iteration, we use the value of the fraction obtained at the previous iteration. From the derived values in column 7, we write their maximum for* $\beta = 1, 2, \ldots, q-m$*, the maximum for* $\Omega_k^u, u = 1, 2, \ldots, \omega_k$*, and the minimum for* $\Omega_1^v \in \Omega_{1k}$*, respectively, in columns 8, 9 and 10. Using formula (1.36), we take the minimum value from column 10. Therefore, we obtain* $\overline{\varrho}_1(p) = 17.5$*.*

Table 3.10: *Calculation of the stability radius $\overline{\varrho}_1(p)$ for problem $\mathcal{J}3/n=2/\sum \mathcal{C}_i$*

G_k	$\lvert\Omega_{1k}\rvert$	$\Omega_1^v \in \Omega_{1k}$	Ω_k^u, $1 \le u \le \omega_k$	β	$p_{ij_{(m+\beta)}}$, $1 \le \beta \le q-m$	$\frac{\sum_{\alpha=1}^{m+\beta} p_{ij_{(\alpha)}}(n_{ij_{(\alpha)}}(\Omega_k^u) - n_{ij_{(\alpha)}}(\Omega_1^v))}{\sum_{\alpha=1}^{m+\beta} \lvert n_{ij_{(\alpha)}}(\Omega_k^u) - n_{ij_{(\alpha)}}(\Omega_1^v)\rvert}$	$\max_{\beta}$	$\max_{\Omega_k^u}$	$\min_{\Omega_1^v}$
1	2	3	4	5	6	7	8	9	10
G_2	4	Ω_1^1	Ω_2^1	1	$p_{ij_{(5)}} = 55$	$\frac{50(1-2)+55(2-0)}{\lvert 1-2\rvert+\lvert 2-0\rvert} = 20$	22.5	22.5	22.5
				2	$p_{ij_{(6)}} = 30$	$\frac{50(1-2)+55(2-0)+30(2-1)}{\lvert 1-2\rvert+\lvert 2-0\rvert+\lvert 2-1\rvert} = 22.5$			
			Ω_2^2, Ω_2^3	1	$p_{ij_{(4)}} = 60$	$\frac{75(1-2)+50(1-2)+60(1-0)}{1+1+1} = \frac{-65}{3}$	12.5		
				2	$p_{ij_{(5)}} = 55$	$\frac{-65+55(2-0)}{3+2} = \frac{45}{5} = 9$			
				3	$p_{ij_{(6)}} = 30$	$\frac{45+30(2-1)}{5+1} = 12.5$			
			Ω_2^4	1	$p_{ij_{(4)}} = 60$	$\frac{75(0-2)+50(1-2)+60(2-0)}{2+1+2} = \frac{-80}{5}$	7.5		
				2	$p_{ij_{(5)}} = 60$	$\frac{-80+55(2-0)}{5+2} = \frac{30}{7} = 4\frac{2}{3}$			
				3	$p_{ij_{(6)}} = 30$	$\frac{30+30(2-1)}{7+1} = 7.5$			
		Ω_1^2, Ω_1^5	Ω_2^1	1	$p_{ij_{(4)}} = 75$	$\frac{50(1-2)+60(0-1)+75(2-1)}{1+1+1} = \frac{-35}{3}$	17.5	22.5	
				2	$p_{ij_{(5)}} = 55$	$\frac{-35+55(2-0)}{3+2} = \frac{75}{5} = 15$			
				3	$p_{ij_{(6)}} = 30$	$\frac{75+30(2-1)}{5+1} = 17.5$			
			Ω_2^2, Ω_2^3	1	$p_{ij_{(5)}} = 55$	$\frac{50(1-2)+55(2-0)}{1+2} = \frac{60}{3} = 20$	22.5		
				2	$p_{ij_{(6)}} = 30$	$\frac{60+30(2-1)}{3+1} = 22.5$			
			Ω_2^4	1	$p_{ij_{(4)}} = 60$	$\frac{75(0-1)+50(1-2)+60(2-1)}{1+1+1} = \frac{-65}{3}$	12.5		
				2	$p_{ij_{(5)}} = 55$	$\frac{-65+55(2-0)}{3+2} = \frac{45}{5} = 9$			
				3	$p_{ij_{(6)}} = 30$	$\frac{45+30(2-1)}{5+1} = 12.5$			
		Ω_1^6	Ω_2^1	1	$p_{ij_{(4)}} = 75$	$\frac{50(1-2)+60(0-2)+75(2-0)}{1+2+2} = \frac{-20}{5}$	15	22.5	
				2	$p_{ij_{(5)}} = 55$	$\frac{-20+55(2-0)}{5+2} = \frac{90}{7} = 12\frac{6}{7}$			
				3	$p_{ij_{(6)}} = 30$	$\frac{90+30(2-1)}{7+1} = 15$			
			Ω_2^2, Ω_2^3	1	$p_{ij_{(4)}} = 75$	$\frac{50(1-2)+60(1-2)+75(1-0)}{1+1+1} = \frac{-35}{3}$	17.5		
				2	$p_{ij_{(5)}} = 55$	$\frac{-35+55(2-0)}{3+2} = \frac{75}{5} = 15$			
				3	$p_{ij_{(6)}} = 30$	$\frac{75+30(2-1)}{5+1} = 17.5$			
			Ω_2^4	1	$p_{ij_{(5)}} = 55$	$\frac{50(1-2)+55(2-0)}{1+2} = \frac{60}{3} = 20$	22.5		
				2	$p_{ij_{(6)}} = 30$	$\frac{60+30(2-1)}{3+1} = 22.5$			
G_3	5	Ω_1^1	Ω_3^1	1	$p_{ij_{(5)}} = 60$	$\frac{75(1-2)+50(1-2)+60(2-0)}{1+1+2} = \frac{-5}{4}$	17.5	17.5	17.5
				2	$p_{ij_{(6)}} = 55$	$\frac{-5+55(2-0)}{4+2} = 17.5$			
			Ω_3^2	1	$p_{ij_{(4)}} = 60$	$\frac{75(0-2)+50(1-2)+60(2-0)}{2+1+2} = \frac{-80}{5}$	7.5		
				2	$p_{ij_{(5)}} = 55$	$\frac{-80+55(2-0)}{5+2} = \frac{30}{7} = 4\frac{2}{7}$			
				3	$p_{ij_{(6)}} = 30$	$\frac{30+30(2-1)}{7+1} = 7.5$			
		Ω_1^2, Ω_1^5	Ω_3^1	1	$p_{ij_{(5)}} = 60$	$\frac{50(1-2)+60(2-1)}{1+1} = \frac{10}{2} = 5$	30	30	
				2	$p_{ij_{(6)}} = 55$	$\frac{10+55(2-0)}{2+2} = 30$			
			Ω_3^2	1	$p_{ij_{(4)}} = 60$	$\frac{75(0-1)+50(1-2)+60(1+1-1)}{1+1+1} = \frac{-65}{3}$	12.5		
				2	$p_{ij_{(5)}} = 55$	$\frac{-65+55(2-0)}{3+2} = \frac{45}{5} = 9$			
				3	$p_{ij_{(6)}} = 30$	$\frac{45+30(2-1)}{5+1} = 12.5$			

Table 3.10 (continuation)**:** *Calculation of the stability radius* $\overline{\varrho}_1(p)$ *for problem* $\mathcal{J}3/n=2/\sum \mathcal{C}_i$

G_k	$\lvert\Omega_{1k}\rvert$	$\Omega_1^v \in \Omega_{1k}$	Ω_k^u, $1 \le u \le \omega_k$	β	$p_{ij_{(m+\beta)}}$, $1 \le \beta \le q-m$	$\frac{\sum_{\alpha=1}^{m+\beta} p_{ij_{(\alpha)}}(n_{ij_{(\alpha)}}(\Omega_k^u) - n_{ij_{(\alpha)}}(\Omega_1^v))}{\sum_{\alpha=1}^{m+\beta} \lvert n_{ij_{(\alpha)}}(\Omega_k^u) - n_{ij_{(\alpha)}}(\Omega_1^v)\rvert}$	$\max_{\beta}$	$\max_{\Omega_k^u}$	$\min_{\Omega_1^v}$
1	2	3	4	5	6	7	8	9	10
		Ω_1^3	Ω_3^1	1	$p_{ij_{(5)}} = 60$	$\frac{75(1-2)+60(2-0)}{1+2} = \frac{45}{3} = 15$	25	25	
				2	$p_{ij_{(6)}} = 55$	$\frac{45+55(2-1)}{3+1} = 25$			
			Ω_3^2	1	$p_{ij_{(4)}} = 60$	$\frac{75(0-2)+60(2-0)}{2+2} = \frac{-30}{4}$	$9\frac{1}{6}$		
				2	$p_{ij_{(5)}} = 55$	$\frac{-30+55(2-1)}{4+1} = \frac{25}{5} = 5$			
				3	$p_{ij_{(6)}} = 30$	$\frac{25+30(2-1)}{5+1} = 9\frac{1}{6}$			
		Ω_1^6	Ω_3^1	1	$p_{ij_{(5)}} = 75$	$\frac{50(1-2)+75(1-0)}{1+1} = \frac{25}{2} = 12.5$	33.75	33.75	
				2	$p_{ij_{(6)}} = 55$	$\frac{25+55(2-0)}{2+2} = 33.75$			
			Ω_3^2	1	$p_{ij_{(5)}} = 55$	$\frac{50(1-2)+55(2-0)}{1+2} = \frac{60}{3} = 20$	22.5		
				2	$p_{ij_{(6)}} = 30$	$\frac{60+30(2-1)}{3+1} = 22.5$			
G_4	2	Ω_1^6	Ω_4^1	1	$p_{ij_{(5)}} = 75$	$\frac{60(1-2)+75(2-0)}{1+2} = \frac{90}{3} = 30$	36.25	36.25	35
				2	$p_{ij_{(6)}} = 55$	$\frac{90+55(1-0)}{3+1} = 36.25$			
		Ω_1^8	Ω_4^1	1	$p_{ij_{(5)}} = 75$	$\frac{60(1-2)+75(2-0)}{1+2} = \frac{90}{3} = 30$	35	35	
				2	$p_{ij_{(6)}} = 55$	$\frac{90+55(2-1)}{3+1} = 35$			
G_5	0								

We consider job shop problem $\mathcal{J}3/n = 2, a_i \le p_i \le b_i/\sum \mathcal{C}_i$ *(see Example 3.1) with the aim to illustrate the idea of constructing a solution to this problem mentioned in Remark 3.9. The structural input data are given by mixed graph* G *in Figure 3.1. The numerical input data are given in Table 3.3. Obviously, the set of all feasible digraphs* $\Lambda(G)$ *is identical for* $\mathcal{C}_{max}$ *and* $\sum \mathcal{C}_i$*, and here we number these digraphs in non-decreasing order of the values* $\sum \mathcal{C}_i$ *with the same initial vector* $p = (75, 50, 40, 60, 55, 30)$ *as for problem* $\mathcal{J}3/n = 2, a_i \le p_i \le b_i/\mathcal{C}_{max}$ *considered in Chapter 3:* $L_1^p \le L_2^p \le \ldots \le L_5^p$ *(see Figure 3.8). Using the modification of critical path method described at page 142, we can simplify digraphs* $G_1, G_2, \ldots, G_5$*, but for these numerical input data (see Table 3.3) the corresponding digraphs* $G_1^T, G_2^T, \ldots, G_5^T$ *remain the same. It means that the number of sets of representatives* ω_k^T *is equal to the number* ω_k *for each digraph* $G_k, k = 1, 2, \ldots, 5$*, (see Table 3.9). We calculate the relative stability radius* $\overline{\varrho}_1^{\Lambda(G)}(p \in T)$ *for the optimal digraph* $G_1(p)$ *presented in Figure 3.8.*

First, due to Remark 3.7 we have to construct the set Ω_{1k}^* *for each digraph* $G_k, k = 2, 3, 4, 5$*. To this end, we construct an auxiliary Table 3.11, where for each combination of the sets* $\Omega_1^v, v = 1, 2, \ldots, \omega_1^T$*, and* $\Omega_k^u, u = 1, 2, \ldots, \omega_k^T, k = 2, 3, 4, 5$*, we obtain the vector* p^* *according to formula (3.53) (see column 4) and check inequality (3.54) (see column 5). As we made in Table 3.10, we combine the same calculations for each pair of sets* Ω_1^2 *and* Ω_1^5*,* Ω_1^4 *and* Ω_1^7*,* Ω_2^2 *and* Ω_2^3*. Since* $\Omega_{sk}^* \subseteq \Omega_{sk}$*, we do not perform such a calculation for the*

Table 3.11: *Auxiliary information for construction sets* $\Omega^*_{1k}, k \in \{2,3,4,5\}$, *for problem* $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\sum \mathcal{C}_i$

G_k	Ω^v_1	Ω^u_k	$p^* \in T$	$\sum_{\mu\in\Omega^v_1} l^{p^*}(\mu) > \sum_{\mu\in\Omega^v_1} l^{p^*}(\mu)$	Ω^*_{1k}
1	2	3	4	5	6
G_2	Ω^1_1	Ω^1_2	$(75,90,40,60,45,20)$	$390 \not> 410$	$\Omega^1_1 \notin \Omega^*_{1,2}$
		Ω^2_2, Ω^3_2	$(100,90,40,50,45,20)$	$440 > 410$	
		Ω^4_2	$(100,90,40,50,45,20)$	$440 > 410$	
	Ω^2_1, Ω^5_1	Ω^1_2	$(35,90,40,80,45,20)$	$400 > 330$	$\{\Omega^2_1, \Omega^5_1\} \not\subseteq \Omega^*_{1,2}$
		Ω^2_2, Ω^3_2	$(75,90,40,60,45,20)$	$375 \not> 395$	
		Ω^4_2	$(100,90,40,50,45,20)$	$390 > 360$	
	Ω^3_1	Ω^1_2	$(75,50,40,60,45,20)$	$305 \not> 370$	$\Omega^3_1 \notin \Omega^*_{1,2}$
		Ω^2_2, Ω^3_2	$(100,50,40,50,45,20)$	$355 \not> 370$	
		Ω^4_2	$(100,50,40,50,45,20)$	$355 > 320$	
	$\{\Omega^4_1, \Omega^7_1\} \not\subseteq \Omega_{1,2}$				$\{\Omega^4_1, \Omega^7_1\} \not\subseteq \Omega^*_{1,2}$
	Ω^6_1	Ω^1_2	$(35,90,40,80,45,20)$	$400 > 330$	$\Omega^6_1 \notin \Omega^*_{1,2}$
		Ω^2_2, Ω^3_2	$(35,90,40,80,45,20)$	$400 > 375$	
		Ω^4_2	$(75,90,40,60,45,20)$	$360 \not> 380$	
	$\Omega^8_1 \notin \Omega_{1,2}$				$\Omega^8_1 \notin \Omega^*_{1,2}$
G_3	Ω^1_1	Ω^1_3	$(100,90,40,50,45,30)$	$450 \not> 450$	$\Omega^1_1 \notin \Omega^*_{1,3}$
		Ω^2_3	$(100,90,40,50,45,20)$	$440 > 360$	
	Ω^2_1, Ω^5_1	Ω^1_3	$(75,90,40,50,45,30)$	$375 \not> 425$	$\{\Omega^2_1, \Omega^5_1\} \not\subseteq \Omega^*_{1,3}$
		Ω^2_3	$(100,90,40,50,45,20)$	$390 > 360$	
	Ω^3_1	Ω^1_3	$(100,50,40,50,45,30)$	$365 \not> 410$	$\Omega^3_1 \notin \Omega^*_{1,3}$
		Ω^2_3	$(100,50,40,50,45,20)$	$355 > 320$	
	$\{\Omega^4_1, \Omega^7_1\} \not\subseteq \Omega_{1,3}$				$\{\Omega^4_1, \Omega^7_1\} \not\subseteq \Omega^*_{1,3}$
	Ω^6_1	Ω^1_3	$(35,90,40,60,45,30)$	$370 \not> 405$	$\Omega^6_1 \notin \Omega^*_{1,3}$
		Ω^2_3	$(75,90,40,60,45,20)$	$360 \not> 380$	
	$\Omega^8_1 \notin \Omega_{1,3}$				$\Omega^8_1 \notin \Omega^*_{1,3}$
G_4	$\Omega^1_1 \notin \Omega_{1,4}$				$\Omega^1_1 \notin \Omega^*_{1,4}$
	$\{\Omega^2_1, \Omega^5_1\} \not\subseteq \Omega_{1,4}$				$\{\Omega^2_1, \Omega^5_1\} \not\subseteq \Omega^*_{1,4}$
	$\Omega^3_1 \notin \Omega_{1,4}$				$\Omega^3_1 \notin \Omega^*_{1,4}$
	$\{\Omega^4_1, \Omega^7_1\} \not\subseteq \Omega_{1,4}$				$\{\Omega^4_1, \Omega^7_1\} \not\subseteq \Omega^*_{1,4}$
	Ω^6_1	Ω^1_4	$(35,50,40,80,45,30)$	$330 \not> 365$	$\Omega^6_1 \notin \Omega^*_{1,4}$
	Ω^8_1	Ω^1_4	$(35,40,40,80,55,30)$	$325 \not> 355$	$\Omega^8_1 \notin \Omega^*_{1,4}$
G_5	$\{\Omega_{1,5}\} = \emptyset$				$\{\Omega^*_{1,5}\} = \emptyset$

sets Ω^u_k, *which do not belong to the sets* $\Omega_{1k}, k = 2,3,4,5$, *(see Table 3.10). So, it follows from column 5 that there is no set of representatives* Ω^v_1, $v \in \{1,2,\ldots,\omega^T_1\}$, *such that the inequality (3.54) holds for each set of representatives* Ω^u_k, $u \in \{1,2,\ldots,\omega^T_k\}$. *Thus,* $\Omega^*_{1k} = \emptyset$ *for each digraph* $G_k \in B = \Lambda(G) \setminus \{G_1\}$. *Therefore, from Theorem 3.6 and Theorem 3.7, it follows that* $\overline{\varrho}^{\Lambda(G)}_1(p \in T) = \infty$. *(For the numerical input data presented in Table 3.3, digraph* G_1 *dominates all digraphs* $G_k \in \Lambda(G)$ *in the polytope* T *and remains the best for all feasible vectors* $x \in T$ *of the processing times.) In such a case, we obtain a single-element minimal solution* $\Lambda^T(G) = \{G_1\}$.

To illustrate the case of formula (3.56) from Theorem 3.6, we give the following example.

Example 3.3 *We consider a similar job shop problem* $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\sum \mathcal{C}_i$ *with the same structural input data (see Figure 3.1), but with different numerical input data (see Table 3.12). We do not simplify the digraphs* $G_1, G_2, \ldots, G_5$ *for the new numerical input data, i.e., the corresponding digraphs* $G_1^T, G_2^T, \ldots, G_5^T$ *have the same sets of representatives* $\{\Omega_k^u : u = 1, 2, \ldots, \omega_k^T,\ \omega_k^T = \omega_k,\ k = 1, 2, \ldots, 5\}$. *For the same initial vector* $p = (75, 50, 40, 60, 55, 30)$, *we have the same optimal digraph* $G_1(p)$ *and all feasible digraphs* $\Lambda(G)$ *are numbered as for the above problem (see Figure 3.8). We calculate the relative stability radius* $\overline{\varrho}_1^{\Lambda(G)}(p \in T)$ *on the basis of Theorem 3.6.*

Table 3.12: *Numerical data for problem* $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\sum \mathcal{C}_i$

i	1	1	1	2	2	2
j	1	2	3	1	2	3
a_{ij}	35	30	40	45	10	15
b_{ij}	105	100	75	85	65	50

For the combination of the numerical input data given in Table 3.12 we construct the following sets:

$$\Omega_{1,2}^* = \{\Omega_1^1, \Omega_1^2, \Omega_1^5, \Omega_1^6\},$$
$$\Omega_{1,3}^* = \{\Omega_1^1, \Omega_1^2, \Omega_1^5, \Omega_1^6\},$$
$$\Omega_{1,4}^* = \{\Omega_1^6\},$$
$$\Omega_{1,5}^* = \emptyset.$$

It means that digraph G_1 *does not dominate digraphs* G_2, G_3, G_4 *in the polytope* T *and due to Theorem 3.7, we have* $\overline{\varrho}_1^{\Lambda(G)}(p \in T) \neq \infty$.

In Table 3.13 one can observe the calculation of the relative stability radius $\overline{\varrho}_1^{\Lambda(G)}(p \in T)$ *for the vector* $p = (75, 50, 40, 60, 55, 30)$. *Following Theorem 3.6, we must compare digraph* G_1 *with each digraph* $G_k, k = 2, 3, 4$, *for which* $\Omega_{1k}^* \neq \emptyset$. *Thus, we perform the calculations due to formulas (3.49) and (3.56) for each set* $\Omega_1^v \in \Omega_{1k}^*$ *(see column 2) and each set* Ω_k^u *(see column 3). For the sets* Ω_k^u *in column 3, inequality* $\sum_{\nu \in \Omega_k^u} l^p(\nu) \ge L_1^p = 325$ *holds (see Table 3.9). Column 5 contains the values* $\Delta_\beta^{ij}(\Omega_1^v, \Omega_k^u), \beta = 0, 1, \ldots, |N(\Omega_1^v, \Omega_k^u)| - 1$, *defined by formula (3.47) (see page 149) for each operation* $O_{ij} \in N(\Omega_k^u, \Omega_1^v) = \{\cup_{\mu \in \Omega_k^u \cup \Omega_1^v}[\mu] : n_{ij}(\Omega_k^u) \neq n_{ij}(\Omega_1^v)\}$. *The order of these values is defined by (3.48). The corresponding values* $N_\beta(\Delta)$ *are given in column 6. Column 8 contains the value* $r_{\Omega_1^v, \Omega_k^u}$ *which is equal to the maximum of the values given in column 7 for* $\beta = 0, 1, \ldots, |N(\Omega_1^v, \Omega_k^u)| - 1$ *(see formula (3.49)). The values*

$$\overline{r}_{k1}^B = \min_{\Omega_1^v \in \Omega_{1k}^*} \max_{\substack{u \in \{1,2,\ldots,\omega_k^T\}, \\ \sum_{\nu \in \Omega_k^u} l^p(\nu) \ge L_1^p}} r_{\Omega_1^v, \Omega_k^u}$$

Table 3.13: *Calculation relative stability radius* $\overline{\varrho}_1^{\Lambda(G)}(p \in T)$ *for problem* $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\sum \mathcal{C}_i$

G_k	$\Omega_1^v \in \Omega_{1k}^*$	Ω_k^u	β	$\Delta_\beta^{ij}(\Omega_1^v, \Omega_k^u)$	$N_\beta(\Delta)$	$\frac{\sum_{\nu\in\Omega_k^u} l^p(\nu) - \sum_{\mu\in\Omega_1^v} l^p(\mu) - \sum_{\alpha=0}^{\beta} \Delta_\alpha^{ij}(\Omega_1^v, \Omega_k^u)\cdot N_\alpha(\Delta)}{\sum_{O_{ij}\in Q^J} \lvert n_{ij}(\Omega_k^u) - n_{ij}(\Omega_1^v)\rvert - \sum_{\alpha=0}^{\beta} \lvert N_\alpha(\Delta)\rvert}$	$\max\limits_{\beta}$	$\max\limits_{\Omega_k^u}$	$\min\limits_{\Omega_1^v}$
1	2	3	4	5	6	7	8	9	10
G_2	Ω_1^1	Ω_2^1	0	$\Delta_0^{ij}(\Omega_1^1, \Omega_2^1)=0$	0	$\frac{410-320-0}{4-0} = \frac{90}{4} = 22.5$	25	25	25
			1	$\Delta_1^{2,3}(\Omega_1^1, \Omega_2^1)=15$	1	$\frac{90-15}{4-1} = \frac{75}{3} = 25$			
			2	$\Delta_2^{2,2}(\Omega_1^1, \Omega_2^1)=45$	2	$\frac{75-45\cdot 2}{3-2} < 0$			
		Ω_2^2, Ω_2^3	0	$\Delta_0^{ij}(\Omega_1^1, \Omega_2^2)=0$	0	$\frac{395-320-0}{6-0} = \frac{75}{6} = 12.5$	12.5		
			1	$\Delta_1^{2,1}(\Omega_1^1, \Omega_2^2)=15$	1	$\frac{75-15}{6-1} = \frac{60}{5} = 12$			
			2	$\Delta_2^{2,3}(\Omega_1^1, \Omega_2^2)=15$	1	$\frac{60-15}{5-1} = \frac{45}{4} = 11.25$			
			3	$\Delta_3^{1,1}(\Omega_1^1, \Omega_2^2)=30$	1	$\frac{45-30}{4-1} = \frac{15}{3} = 5$			
			4	$\Delta_4^{2,2}(\Omega_1^1, \Omega_2^2)=45$	2	$\frac{15-45\cdot 2}{3-2} < 0$			
		Ω_2^4	0	$\Delta_0^{ij}(\Omega_1^1, \Omega_2^4)=0$	0	$\frac{380-320-0}{8-0} = \frac{60}{8} = 7.5$	7.5		
			1	$\Delta_1^{2,3}(\Omega_1^1, \Omega_2^4)=15$	1	$\frac{60-15}{8-1} = \frac{45}{7} = 6\frac{3}{7}$			
			2	$\Delta_2^{2,1}(\Omega_1^1, \Omega_2^4)=15$	2	$\frac{45-15\cdot 2}{7-2} = \frac{15}{5} = 3$			
			3	$\Delta_3^{1,1}(\Omega_1^1, \Omega_2^4)=30$	2	$\frac{15-30\cdot 2}{5-2} < 0$			
	Ω_1^2, Ω_1^5	Ω_2^1	0	$\Delta_0^{ij}(\Omega_1^2, \Omega_2^1)=0$	0	$\frac{410-305-0}{6-0} = \frac{105}{6} = 17.5$	18	25	
			1	$\Delta_1^{2,3}(\Omega_1^2, \Omega_2^1)=15$	1	$\frac{105-15}{6-1} = \frac{90}{5} = 18$			
			2	$\Delta_2^{2,1}(\Omega_1^2, \Omega_2^1)=25$	1	$\frac{90-25}{5-1} = \frac{65}{4} = 16.25$			
			3	$\Delta_3^{1,1}(\Omega_1^2, \Omega_2^1)=40$	1	$\frac{65-40}{4-1} = \frac{25}{3} = 8\frac{1}{3}$			
			4	$\Delta_4^{2,2}(\Omega_1^2, \Omega_2^1)=45$	2	$\frac{25-44\cdot 2}{3-2} < 0$			
		Ω_2^2, Ω_2^3	0	$\Delta_0^{ij}(\Omega_1^2, \Omega_2^2)=0$	0	$\frac{395-305-0}{4-0} = \frac{90}{4} = 22.5$	25		
			1	$\Delta_1^{2,3}(\Omega_1^2, \Omega_2^2)=15$	1	$\frac{90-15}{4-1} = \frac{75}{3} = 25$			
			2	$\Delta_2^{2,2}(\Omega_1^2, \Omega_2^2)=45$	2	$\frac{75-45\cdot 2}{3-2} < 0$			
		Ω_2^4	0	$\Delta_0^{ij}(\Omega_1^2, \Omega_2^4)=0$	0	$\frac{380-305-0}{6-0} = \frac{75}{6} = 12.5$	12.5		
			1	$\Delta_1^{2,1}(\Omega_1^2, \Omega_2^4)=15$	1	$\frac{75-15}{6-1} = \frac{60}{5} = 12$			
			2	$\Delta_2^{2,3}(\Omega_1^2, \Omega_2^4)=15$	1	$\frac{60-15}{5-1} = \frac{45}{4} = 11.25$			
			3	$\Delta_3^{1,1}(\Omega_1^2, \Omega_2^4)=30$	1	$\frac{45-30}{4-1} = \frac{15}{3} = 5$			
			4	$\Delta_4^{2,2}(\Omega_1^2, \Omega_2^4)=45$	2	$\frac{15-45\cdot 2}{3-2} < 0$			
	Ω_1^6	Ω_2^1	0	$\Delta_0^{ij}(\Omega_1^6, \Omega_2^1)=0$	0	$\frac{410-290-0}{8-0} = \frac{120}{8} = 15$	15	25	
			1	$\Delta_1^{2,3}(\Omega_1^6, \Omega_2^1)=15$	1	$\frac{120-15}{8-1} = \frac{105}{7} = 15$			
			2	$\Delta_2^{2,1}(\Omega_1^6, \Omega_2^1)=25$	2	$\frac{105-25\cdot 2}{7-2} = \frac{55}{5} = 11$			
			3	$\Delta_3^{1,1}(\Omega_1^6, \Omega_2^1)=40$	2	$\frac{55-40\cdot 2}{5-2} < 0$			
		Ω_2^2, Ω_2^3	0	$\Delta_0^{ij}(\Omega_1^6, \Omega_2^2)=0$	0	$\frac{395-290-0}{6-0} = \frac{105}{6} = 17.5$	18		
			1	$\Delta_1^{2,3}(\Omega_1^6, \Omega_2^2)=15$	1	$\frac{105-15}{6-1} = \frac{90}{5} = 18$			
			2	$\Delta_2^{2,1}(\Omega_1^6, \Omega_2^2)=25$	1	$\frac{90-25}{5-1} = \frac{65}{4} = 16.25$			
			3	$\Delta_3^{1,1}(\Omega_1^6, \Omega_2^2)=40$	1	$\frac{65-40}{4-1} = \frac{25}{3} = 8\frac{1}{3}$			
			4	$\Delta_4^{2,2}(\Omega_1^6, \Omega_2^2)=45$	2	$\frac{25-45\cdot 2}{3-2} < 0$			
		Ω_2^4	0	$\Delta_0^{ij}(\Omega_1^6, \Omega_2^4)=0$	0	$\frac{380-290-0}{4-0} = \frac{90}{4} = 22.5$	25		
			1	$\Delta_1^{2,3}(\Omega_1^6, \Omega_2^4)=15$	1	$\frac{90-15}{4-1} = \frac{75}{3} = 25$			
			2	$\Delta_2^{2,2}(\Omega_1^6, \Omega_2^4)=45$	2	$\frac{75-45\cdot 2}{3-2} < 0$			

Table 3.13 (continuation)**:** *Calculation relative stability radius* $\overline{\varrho}_1^{\Lambda(G)}(p \in T)$ *for problem* $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\sum \mathcal{C}_i$

G_k	$\Omega_1^v \in \Omega_{1k}^*$	Ω_k^u	β	$\Delta_\beta^{ij}(\Omega_1^v, \Omega_k^u)$	$N_\beta(\Delta)$	$\dfrac{\sum\limits_{\nu\in\Omega_k^u} l^p(\nu) - \sum\limits_{\mu\in\Omega_1^v} l^p(\mu) - \sum\limits_{\alpha=0}^{\beta} \Delta_\alpha^{ij}(\Omega_1^v, \Omega_k^u)\cdot N_\alpha(\Delta)}{\sum\limits_{O_{ij}\in Q^J} \lvert n_{ij}(\Omega_k^u) - n_{ij}(\Omega_1^v)\rvert - \sum\limits_{\alpha=0}^{\beta} \lvert N_\alpha(\Delta)\rvert}$	$\max\limits_{\beta}$	$\max\limits_{\Omega_k^u}$	$\min\limits_{\Omega_1^v}$
1	2	3	4	5	6	7	8	9	10
G_3	Ω_1^1	Ω_3^1	0	$\Delta_0^{ij}(\Omega_1^1, \Omega_3^1)=0$	0	$\frac{425-320-0}{6-0} = \frac{105}{6} = 17.5$	18.75	18.75	18.75
			1	$\Delta_1^{2,1}(\Omega_1^1, \Omega_3^1)=15$	2	$\frac{105-15\cdot 2}{6-2} = \frac{75}{4} = 18.75$			
			2	$\Delta_2^{1,1}(\Omega_1^1, \Omega_3^1)=30$	1	$\frac{75-30}{4-1} = \frac{45}{3} = 15$			
			3	$\Delta_3^{2,2}(\Omega_1^1, \Omega_3^1)=45$	2	$\frac{45-45\cdot 2}{3-2} < 0$			
		Ω_3^2	0	$\Delta_0^{ij}(\Omega_1^1, \Omega_3^2)=0$	0	$\frac{380-290-0}{8-0} = \frac{60}{8} = 7.5$	7.5		
			1	$\Delta_1^{2,1}(\Omega_1^1, \Omega_3^2)=15$	1	$\frac{60-15}{8-1} = \frac{45}{7} = 6\frac{3}{7}$			
			2	$\Delta_2^{2,3}(\Omega_1^1, \Omega_3^2)=15$	2	$\frac{45-15\cdot 2}{7-2} = \frac{15}{5} = 3$			
			3	$\Delta_3^{1,1}(\Omega_1^1, \Omega_3^2)=30$	2	$\frac{15-30\cdot 2}{5-2} < 0$			
	Ω_1^2, Ω_1^5	Ω_3^1	0	$\Delta_0^{ij}(\Omega_1^2, \Omega_3^1)=0$	0	$\frac{425-305-0}{4-0} = \frac{120}{4} = 30$	35	35	
			1	$\Delta_1^{2,1}(\Omega_1^2, \Omega_3^1)=15$	1	$\frac{120-15}{4-1} = \frac{105}{3} = 35$			
			2	$\Delta_2^{2,2}(\Omega_1^2, \Omega_3^1)=45$	2	$\frac{105-45\cdot 2}{3-2} = 15$			
		Ω_3^2	0	$\Delta_0^{ij}(\Omega_1^2, \Omega_3^2)=0$	0	$\frac{380-305-0}{6-0} = \frac{75}{6} = 12.5$	12.5		
			1	$\Delta_1^{2,1}(\Omega_1^2, \Omega_3^2)=15$	1	$\frac{75-15}{6-1} = \frac{60}{5} = 12$			
			2	$\Delta_2^{2,2}(\Omega_1^2, \Omega_3^2)=45$	1	$\frac{60-15}{5-1} = \frac{45}{4} = 11.25$			
			3	$\Delta_3^{1,1}(\Omega_1^2, \Omega_3^2)=30$	1	$\frac{45-30}{4-1} = \frac{15}{3} = 5$			
			4	$\Delta_4^{2,2}(\Omega_1^2, \Omega_3^2)=45$	2	$\frac{15-45\cdot 2}{3-2} < 0$			
	Ω_1^6	Ω_3^1	0	$\Delta_0^{ij}(\Omega_1^6, \Omega_3^1)=0$	0	$\frac{425-290-0}{4-0} = \frac{135}{4} = 33.75$	33.75	33.75	
			1	$\Delta_1^{1,1}(\Omega_1^6, \Omega_3^1)=40$	1	$\frac{135-40}{4-1} = \frac{95}{3} = 31\frac{2}{3}$			
			2	$\Delta_2^{2,2}(\Omega_1^6, \Omega_3^1)=45$	2	$\frac{95-45\cdot 2}{3-2} = 5$			
		Ω_3^2	0	$\Delta_0^{ij}(\Omega_1^6, \Omega_3^2)=0$	0	$\frac{380-290-0}{4-0} = \frac{90}{4} = 22.5$	25		
			1	$\Delta_1^{2,3}(\Omega_1^6, \Omega_3^2)=15$	1	$\frac{90-15}{4-1} = \frac{75}{3} = 25$			
			2	$\Delta_2^{2,2}(\Omega_1^6, \Omega_3^2)=45$	2	$\frac{75-45\cdot 2}{3-2} < 0$			
G_4	Ω_1^6	Ω_4^1	0	$\Delta_0^{ij}(\Omega_1^6, \Omega_4^1)=0$	0	$\frac{435-290-0}{4-0} = \frac{145}{4} = 36.25$	40	40	40
			1	$\Delta_1^{2,1}(\Omega_1^6, \Omega_4^1)=25$	1	$\frac{145-25}{4-1} = \frac{120}{3} = 40$			
			2	$\Delta_2^{1,1}(\Omega_1^6, \Omega_4^1)=40$	2	$\frac{120-40\cdot 2}{3-2} = 40$			

calculated according to (3.55) are given in column 10. As follows from Theorem 3.6, the last step is to take the minimum value in column 10: $\overline{\varrho}_1^{\Lambda(G)}(p \in T) = \min\{\overline{r}_{k1}^B,\ k = 2,3,4\} = \overline{r}_{3,1} = 18.75$.

As follows from Remark 3.9, we also construct an increasing sequence of relative stability radii $\overline{\varrho}_1^{\Lambda(G)}(p \in T) = \overline{r}_{3,1} = 18.75$, $\overline{\varrho}_1^{\Lambda(G)\setminus\{G_3\}}(p \in T) = \overline{r}_{2,1} = 25$, $\overline{\varrho}_1^{\Lambda(G)\setminus\{G_3,G_2\}}(p \in T) = \overline{r}_{4,1} = 40$, $\overline{\varrho}_1^{\Lambda(G)\setminus\{G_3,G_2,G_4\}}(p \in T) = \infty$ *(see Figure 3.9) and a sequence of nested sets of competitive digraphs* G_k *of digraph* G_1*:* $\Gamma_1 = \{G_3\}$, $\Gamma_2 = \{G_2\}$, $\Gamma_3 = \{G_4\}$, *for which dominance relation* $G_1 \preceq_T G_k$ *does not hold. We draw the projections of the stability balls in Figure 3.9 for the same components* $p_{1,3}$ *and* $p_{2,2}$ *of the vector* p

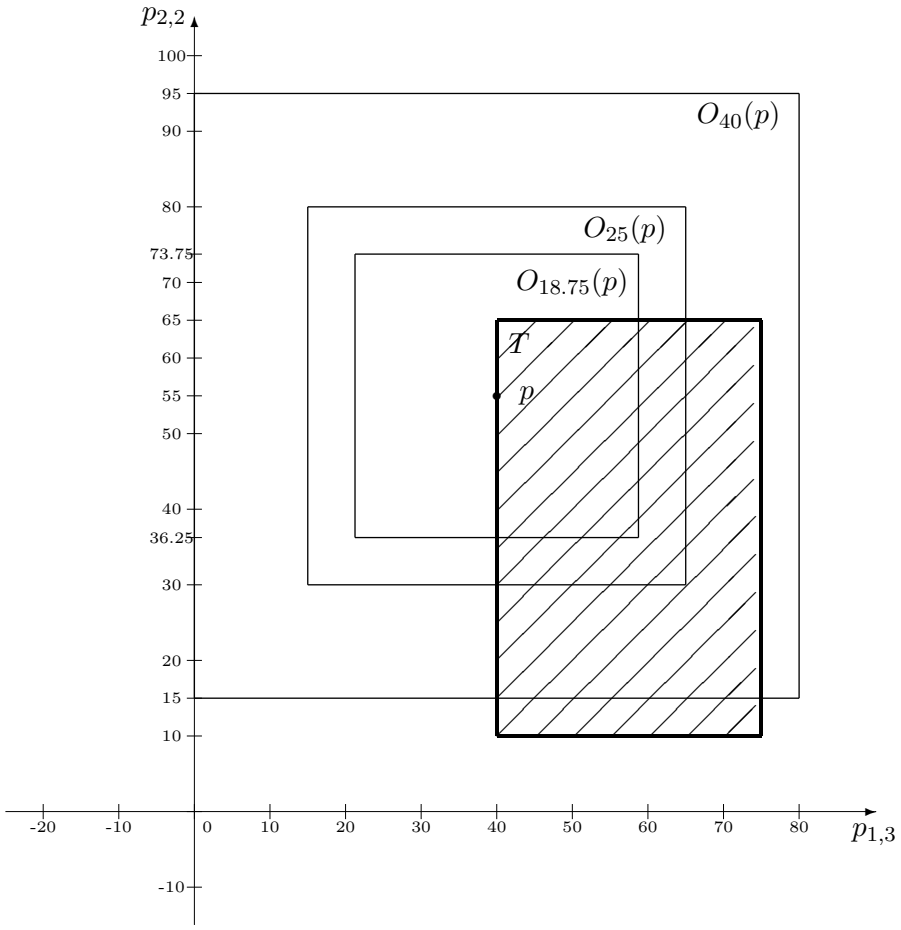

Figure 3.9: *Projections of the stability balls with the center* $p = (75, 50, 40, 60, 55, 30)$ *on the plane for problem* $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\sum \mathcal{C}_i$

Table 3.14: *Optimal digraphs for problem* $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\sum \mathcal{C}_i$ *with different initial vectors* $p \in T$

Initial vector $p^j \in T$	Objective function values $\sum \mathcal{C}_i$	Optimal digraph G_s	$\overline{\varrho}_s^{\Lambda(G)}(p^j \in T)$	Competitive digraph of G_s
1	2	3	4	5
$p^1=(75,95,40,60,10,30)$	$L_2^{p^1}=365, L_3^{p^1}=380,$ $L_1^{p^1}=410, L_4^{p^1}=480,$ $L_5^{p^1}=550$	G_2	$\overline{r}_{3,2}^{\Lambda(G)}=3.75$	G_3
$p^2=(80,95,40,55,10,35)$	$L_3^{p^2}=380, L_2^{p^2}=385,$ $L_1^{p^2}=425, L_4^{p^2}=490,$ $L_5^{p^2}=555$	G_3	$\overline{r}_{2,3}^{\Lambda(G)}=1.25$	G_2
$p^3=(35,35,50,85,10,30)$	$L_4^{p^3}=315, L_2^{p^3}=320,$ $L_1^{p^3}=335, L_3^{p^3}=340,$ $L_5^{p^3}=410$	G_4	$\overline{r}_{1,4}^{\Lambda(G)}=1.25$	G_1

as for problem $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\mathcal{C}_{max}$ from Example 3.1 in Section 3.1 (see Figure 3.4 and Figure 3.5). From Theorem 3.4 and Remark 3.9, it follows that the set $\Lambda^*(G) = \{G_1\} \cup \{\cup_{i=1}^{3}\Gamma_i\} = \{G_1\} \cup \{G_k : G_1 \not\preceq_T G_k\} = \{G_1, G_2, G_3, G_4\}$ is a solution of problem $\mathcal{J}3/n=2, a_i \le p_i \le b_i/\sum \mathcal{C}_i$. Moreover, this solution is minimal since for each digraph $G_k \in \Lambda^*(G)$, there exists a feasible vector for which this digraph is the unique optimal one (see Table 3.14).

In the rest of this section, we prove a bound for relative stability radii $\overline{\varrho}_s^B(p \in T)$, which is analogous to the bound proven for relative stability radius $\widehat{\varrho}_s^B(p \in T)$ for makespan criterion (see Lemma 3.4 at page 136). This bound can considerably restrict the number of feasible digraphs from set B which have to be considered while calculating of the relative stability radius.

Redundant Digraphs for Calculating $\overline{\varrho}_s^B(p \in T)$

To calculate the relative stability radius $\overline{\varrho}_s^B(p \in T)$ of the optimal digraph G_s, we can use the formulas (3.49) and (3.56) from Theorem 3.6. More exactly, one must compare each set $\Omega_s^v, v = 1, 2, \ldots, \omega_s^T$, of representatives of the family of sets $(H_s^i)_{J_i \in J}$, with the sets $\Omega_k^u, u = 1, 2, \ldots, \omega_k^T$, of representatives of the family of sets $(H_k^i)_{J_i \in J}$ of each digraph $G_k \in B \subseteq \Lambda(G),\ k = 1, 2, \ldots, |B|,\ k \ne s$. The following bound, in which $\overline{r}_{ks}^B$ is defined by formula (3.55) at page 151, restricts the number of feasible digraphs G_k with which a comparison of the optimal digraph G_s has to be done during the calculation of the relative stability radius $\overline{\varrho}_s^B(p \in T)$.

Lemma 3.8 *If $\overline{\varrho}_s^B(p \in T) < \infty$ and there exists a digraph $G_k \in B$ such that*

$$\overline{r}_{ks}^B \le \frac{L_t^p - L_s^p}{nq - n} \tag{3.64}$$

for some digraph $G_t \in B$, then it is not necessary to consider digraph G_t during the calculation of relative stability radius $\overline{\varrho}_s^B(p \in T)$.

Proof. To calculate relative stability radius $\overline{\varrho}_s^B(p \in T)$ using Theorem 3.6, we have to compare the optimal digraph G_s consecutively with each feasible digraph $G_i, i \ne s$, from the set B. Let us compare the optimal digraph G_s with a feasible digraph $G_t,\ t \ne k$.

Digraph $G_t, t \ne s$, is a competitive digraph for G_s if we can construct a vector $\overline{x} \in T$ that satisfies the condition (a^*) at page 148, i.e., equality (3.39) holds: $L_s^{\overline{x}} = L_t^{\overline{x}}$. Moreover, for any given real $\epsilon > 0$, which may be as small as desired, there must exist a vector $\overline{p}^\epsilon \in T$ such that $d(\overline{x}, \overline{p}^\epsilon) = \epsilon$ and inequality (3.40) $L_s^{\overline{p}^\epsilon} > L_t^{\overline{p}^\epsilon}$ is satisfied for digraph G_t (see condition (b^*)). More precisely, we must construct a vector $\overline{x}$ of the form $\overline{x} = p(\overline{r}_{ts}^B) = (p_{1,1}(\overline{r}_{ts}^B), p_{1,2}(\overline{r}_{ts}^B), \ldots, p_{nn_n}(\overline{r}_{ts}^B))$ with the components $p_{ij}(\overline{r}_{ts}^B)$ from the set $\{p_{ij}, p_{ij} + \min\{\overline{r}_{ts}^B, b_{ij} - p_{ij}\}, p_{ij} - \min\{\overline{r}_{ts}^B, p_{ij} - a_{ij}\}\}$ according to formula (3.46). Due to condition (c^*) (see page 148), the distance $d(p, \overline{x}) = d(p, p(\overline{r}_{ts}^B)) = \overline{r}_{ts}^B$ must achieve minimal value among the distances between the vector p and the other vectors in the polytope T which satisfy both conditions (a^*) and (b^*).

We suppose that the conditions of Lemma 3.8 are satisfied, i.e., inequality (3.64) holds, and the vector $\overline{x} = p(\overline{r}_{ks}^B)$ satisfies both conditions (a^*) and (b^*). Then we show that the

distance $d(p, p(\overline{r}^B_{ts}))$ cannot become less than the distance $d(p, p(\overline{r}^B_{ks}))$. Next, we show that inequality $\overline{r}^B_{ks} \leq \overline{r}^B_{ts}$ follows from condition (3.64). We have:

$$\begin{aligned} \overline{r}^B_{ks} &\leq \frac{L^p_t - L^p_s}{nq - n} \\ &= \frac{\sum_{\nu \in \Omega^{u^*}_t} l^p(\nu) - \sum_{\mu \in \Omega^{v^*}_s} l^p(\mu)}{n(q-1)} = \overline{r}', \end{aligned}$$

where $\Omega^{u^*}_t, u^* \in \{1, 2, \ldots, \omega^T_t\}$, and $\Omega^{v^*}_s, v^* \in \{1, 2, \ldots, \omega^T_s\}$, are critical sets for the digraphs G_t and G_s, respectively. Since $\sum_{O_{ij} \in Q^J} |n_{ij}(\Omega^{u^*}_k) - n_{ij}(\Omega^{v^*}_s)| < n$, we get the following inequalities:

$$\begin{aligned} \overline{r}' &< \frac{\sum_{\nu \in \Omega^{u^*}_t} l^p(\nu) - \sum_{\mu \in \Omega^{v^*}_s} l^p(\mu)}{(\sum_{O_{ij} \in Q^J} |n_{ij}(\Omega^{u^*}_k) - n_{ij}(\Omega^{v^*}_s)|)(q-1)} \\ &\leq \max_{u \in \{1,2,\ldots,\omega^T_t\}} \frac{\sum_{\nu \in \Omega^u_t} l^p(\nu) - \sum_{\mu \in \Omega^{v^*}_s} l^p(\mu)}{\sum_{O_{ij} \in Q^J} |n_{ij}(\Omega^u_t) - n_{ij}(\Omega^{v^*}_s)|} \\ &\leq \min_{\Omega^v_s \in \Omega_{s,t}} \max_{u \in \{1,2,\ldots,\omega^T_t\}} \frac{\sum_{\nu \in \Omega^u_t} l^p(\nu) - \sum_{\mu \in \Omega^v_s} l^p(\mu)}{\sum_{O_{ij} \in Q^J} |n_{ij}(\Omega^u_t) - n_{ij}(\Omega^v_s)|} \leq \overline{r}^B_{ts}. \end{aligned}$$

Thus, the value $\overline{r}^B_{ts}$ cannot become less than $\overline{r}^B_{ks}$ and therefore digraph G_t need not to be considered during the calculation of the relative stability radius $\overline{\varrho}^B_s(p \in T)$.

◇

Note that bound (3.64) is tight. The above lemma directly implies the following assertion.

Corollary 3.12 *Let set $B = \{G_s = G_{i_1}, G_{i_2}, \ldots, G_{i_{|B|}}\}$ be sorted in non-decreasing order of the objective function values $L^p_{i_1} \leq L^p_{i_2} \leq \ldots \leq L^p_{i_{|B|}}$. If for the currently compared digraph G_{i_k} from the set $B \subseteq \Lambda(G)$ inequality*

$$\overline{r}^B_{i_k s} \leq \frac{L^p_{i_t} - L^p_{i_1}}{nq - n} \tag{3.65}$$

holds for digraph $G_{i_t} \in B$ with $L^p_{i_k} \leq L^p_{i_t}$, then it is possible to exclude the digraphs $G_{i_t}, G_{i_{t+1}}, \ldots, G_{i_{|B|}}$ from further considerations during the calculation of relative stability radius $\overline{\varrho}^B_s(p \in T)$.

Proof. Since the set $B \subseteq \Lambda(G)$ is sorted in non-decreasing order of the objective function values and inequality (3.65) holds for digraph G_{i_t}, inequality

$$\overline{r}^B_{i_k s} \leq \frac{L^p_{i_j} - L^p_{i_1}}{nq - n}$$

holds for each digraph $G_{i_j}, j = t+1, t+2, \ldots, |B|$, and due to Lemma 3.8, these digraphs need not to be considered during the calculation of the relative stability radius.

◇

Using Corollary 3.12 we can compare the optimal digraph $G_s = G_{i_1}$ consecutively with the digraphs $G_{i_2}, G_{i_3}, \ldots, G_{i_{|B|}}$ from set B in non-decreasing order of the objective function values: $L^p_{i_1} \le L^p_{i_2} \le \ldots \le L^p_{i_{|B|}}$. If for the currently compared digraph $G_k = G_{i_r}$ inequality (3.64) holds, we can exclude digraphs $G_{i_r}, G_{i_{r+1}}, \ldots, G_{i_{|B|}}$ from further considerations.

Since $\overline{\varrho}_s(p) = \overline{\varrho}^{\Lambda(G)}_s(p \in R^q_+)$, Corollary 3.12 implies Corollary 3.13 which allows to restrict the number of feasible digraphs while calculating stability radius $\overline{\varrho}_s(p)$ (see Definition 1.2 at page 16).

Corollary 3.13 *Let set $\Lambda(G) = \{G_s = G_{i_1}, G_{i_2}, \ldots, G_{i_\lambda}\}$ be sorted in non-decreasing order of the objective function values: $L^p_{i_1} \le L^p_{i_2} \le \ldots \le L^p_{i_\lambda}$. If for the currently compared digraph G_{i_k} from set $\Lambda(G) = \{G_s = G_{i_1}, G_{i_2}, \ldots, G_{i_k}, \ldots, G_{i_t}, \ldots, G_{i_\lambda}\}$ inequality*

$$\overline{r}^{\Lambda(G)}_{i_k s} \le \frac{L^p_{i_t} - L^p_{i_1}}{q} \tag{3.66}$$

holds for digraph $G_{i_t} \in \Lambda(G)$ with $L^p_{i_k} \le L^p_{i_t}$, then it is possible to exclude digraphs $G_{i_t}, G_{i_{t+1}}, \ldots, G_{i_\lambda}$ from further considerations during the calculation of stability radius $\overline{\varrho}_s(p)$.

3.6 Algorithms for Problems $\mathcal{J}/a_i \le p_i \le b_i/\Phi$

In this section, we focus on criterion $\sum \mathcal{C}_i$ but we indicate also the necessary changes for criterion $\mathcal{C}_{max}$ as well. Using the above mathematical background, we propose first Algorithm $SOL\_\sum \mathcal{C}_i$ for finding a solution $\Lambda^*(G) \subseteq \Lambda(G)$ with 'relatively small' cardinality. As the input data for Algorithm $SOL\_\sum \mathcal{C}_i$, a set of schedules $B \subseteq \Lambda(G)$, which is a solution of problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$, and a vector $p \in T$ of the processing times are used. This algorithm generates a covering of the polytope T (see Theorem 3.4) by nested closed balls $O_r(p)$ with the common center $p \in T$ and different radii r which are relative stability radii $\overline{\varrho}^B_s(p \in T)$ of the same digraph G_s but for different nested sets B.

Let set $B \subseteq \Lambda(G)$ be a given solution to problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$ (in the worst case, the whole set $\Lambda(G)$ of digraphs may be used as such an *input set* B). We also fix a vector $p \in T$ of the processing times and number the digraphs of the set $B = \{G_1, G_2, \ldots, G_{|B|}\}$ in non-decreasing order of the values of the objective function. An 'expected' vector of the processing times (or a vector which has been considered in a previous calculation or some other suitable vector from the polytope T) may be used as the *input vector* p in the following algorithm (in square brackets we give the changes of this algorithm in the case of criterion $\mathcal{C}_{max}$).

Algorithm $SOL\_\sum \mathcal{C}_i$ [**Algorithm** $SOL\_\mathcal{C}_{max}$]

Input: Fixed vector $p = (p_{1,1}, p_{1,2}, \ldots, p_{nn_n}) \in T$, set $B = \{G_1, G_2, \ldots, G_{|B|}\}$ such that $L^p_1 \le L^p_2 \le \ldots \le L^p_{|B|}$ for criterion $\sum \mathcal{C}_i$ [such that $l^p_1 \le l^p_2 \le \ldots \le l^p_{|B|}$ for criterion $\mathcal{C}_{max}$].

Output: Relative stability radius $\overline{\varrho}^B_1(p \in T)$ [$\widehat{\varrho}^B_1(p \in T)$] of the optimal digraph G_1 and a solution $\Lambda^*(G)$.

Step 1: Set $k = 2$ and $\Lambda = \emptyset$.

Step 2: For digraph $G_k \in B$, test dominance relation $G_1 \preceq_T G_k$ using Lemma 3.5 with the objective function $\Phi_k^p = L_k^p$ $[\Phi_k^p = l_k^p]$.

Step 3: **IF** $G_1 \not\preceq_T G_k$ **THEN** calculate $\overline{r}_{k1}^B$ $[\widehat{r}_{k1}^B]$ using formulas (3.49) and (3.55) [formula (3.26)] for the input vector p **ELSE GOTO** *Step 5*.

Step 4: Set $\Lambda := \Lambda \cup \{G_k\}$.

Step 5: Set $k := k + 1$.
IF $k \leq |B|$ **THEN GOTO** *Step 2*.
ELSE using Theorem 3.6 [Theorem 3.3 and Remark 3.6] calculate $\overline{\varrho}_1^B(p \in T) = \min\{\overline{r}_{k1}^B : \ G_1 \not\preceq_T G_k\}$ $[\widehat{\varrho}_1^B(p \in T) = \min\{\widehat{r}_{k1}^B : \ G_1 \not\preceq_T G_k\}]$ and set $\Lambda^*(G) = \Lambda \cup \{G_1\}$ **STOP**.

It is easy to see that the set $\Lambda^*(G) = \{G_{i_1=1}, G_{i_2}, \ldots, G_{i_{|\Lambda^*(G)|}}\}$, $i_1 < i_2 < \ldots < i_{|\Lambda^*(G)|}$, generated by Algorithm $SOL\_\sum \mathcal{C}_i$ is a solution of problem $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$. Indeed, the set $\Lambda^*(G)$ is a subset of the set B which is assumed to be a solution of problem $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$ and the set $\Lambda^*(G)$ includes digraph G_1 and also each digraph G_k, $2 \leq k \leq |B|$, provided that dominance relation $G_1 \preceq_T G_k$ does not hold.

Along with a solution $\Lambda^*(G)$, Algorithm $SOL\_\sum \mathcal{C}_i$ calculates the value $\overline{r}_{k1}^B$ for each digraph $G_k \in B$ such that dominance relation $G_1 \preceq_T G_k$ does not hold (see Step 3). The value $\overline{r}_{k1}^B$ denotes the largest distance $d(p, p')$ such that inequality $L_1^{p'} > L_k^{p'}$ is guaranteed for each vector $p' \in T$ of the processing times. Therefore, dominance relation $G_1 \preceq_{T^*} G_k$ holds for each polytope $T^* = T \cap O_r(p)$ if $r \leq \overline{r}_{k1}^B$, and we have $G_1 \not\preceq_{T^*} G_k$ for the polytope $T^* = T \cap O_r(p)$ if $r > \overline{r}_{k1}^B$. Let us put the digraphs in the set $\Lambda^*(G)$ in non-decreasing order of the values $\overline{r}_{k1}^B$: $\Lambda^*(G) = \{G_{j_1=1}, G_{j_2}, \ldots, G_{j_{|\Lambda^*(G)|}}\}$, where $\overline{r}_{j_2 1}^B \leq \overline{r}_{j_3 1}^B \leq \ldots \leq \overline{r}_{j_{|\Lambda^*(G)|}1}^B$. Due to Theorem 3.6, it follows that $\overline{\varrho}_1^B(p \in T) = \overline{r}_{j_2 1}^B$. Similarly, for the set $B \setminus \{G_{j_2}\}$ we have the equality $\overline{\varrho}_1^{B\setminus\{G_{j_2}\}}(p \in T) = \overline{r}_{j_3 1}^B$, and in general, we have the equality $\overline{\varrho}_1^{B\setminus\{\cup_{k=2}^{l} G_{j_k}\}}(p \in T) = \overline{r}_{j_{l+1}1}^B$, where $1 < l < |\Lambda^*(G)|$. These values $\overline{r}_{k1}^B$ will be used in Algorithm $MINSOL\_\sum \mathcal{C}_i$ which follows. Moreover, they may be used in a realization of the best schedule. Indeed, to realize a solution $\Lambda^*(G)$ (when values $\overline{r}_{k1}^B$ are known), we can start with digraph G_1 which is optimal (or one of the optimal digraphs) for the 'expected' vector $p \in T$ of the processing times. If we will get additional information about the error r of the processing times p_{ij}, we can use r for a suitable modification of the schedule which is currently realized. To this end, we select $\overline{r}_{j_l 1}^B$ such that inequalities $\overline{r}_{j_l 1}^B < r \leq \overline{r}_{j_{l+1}1}^B$ hold, and we can find a better digraph in the set $\cup_{u=1}^{l} G_{i_u}$ which may be realized further instead of the initial digraph G_1. It is practically important that, if the possible error of the given processing times is no more than r, we have the guarantee that the set $\cup_{u=1}^{l} G_{i_u}$ contains at least one optimal digraph.

Note that solution $\Lambda^*(G)$ generated by Algorithm $SOL\_\sum \mathcal{C}_i$ may be not minimal. To

exclude redundant digraphs, we can test the dominance relation $\preceq_T$ between the digraphs from the set $\Lambda^*(G) \setminus \{G_{i_1=1}\}$ which may be done as follows. First, we exclude all digraphs $G_{i_k},\ 2 < k \le |\Lambda^*(G)|$, from the set $\Lambda^*(G)$ for which dominance relation $G_{i_2} \preceq_T G_{i_k}$ holds. To this end, we repeat Algorithm $SOL\_\sum \mathcal{C}_i$ with the set $\Lambda^*(G) \setminus \{G_{i_1=1}\}$ being used instead of set B. Then, similarly, we can exclude all digraphs from the solution which are dominated by digraph G_{i_3} and so on. After no more than $|\Lambda^*(G)| - 2$ repetitions of Algorithm $SOL\_\sum \mathcal{C}_i$ we can remove all redundant digraphs (or an essential part of the redundant digraphs) from the set $\Lambda^*(G)$. As a result we often get minimal solution $\Lambda^T(G)$. Next, we give a formal algorithm for finding a minimal solution on the basis of the above repetitions of Algorithm $SOL\_\sum \mathcal{C}_i$ (see Step 3) and the verification of the strong dominance relation (see Step 5). We set $\Lambda' = \Lambda^*(G) \setminus \{G_{i_1=1}\} = \{G_{i_2}, G_{i_3}, \ldots, G_{i_{|\Lambda^*(G)|}}\}$, where $\Lambda^*(G)$ is obtained by Algorithm $SOL\_\sum \mathcal{C}_i$ provided that inequalities $L_2^p \le L_3^p \le \ldots \le L_{|\Lambda^*(G)|}^p$ hold.

Algorithm $MINSOL\_\sum \mathcal{C}_i$ [**Algorithm** $MINSOL\_\mathcal{C}_{max}$]

Input: Set $\Lambda' = \Lambda^*(G) \setminus \{G_{i_1=1}\} = \{G_{i_2}, G_{i_3}, \ldots, G_{i_{|\Lambda^*(G)|}}\}$.
Output: Minimal solution $\Lambda^T(G)$.

Step 1: Set $\Lambda^T(G) = \{G_{i_1=1}\}$.

Step 2: Set $B = \Lambda'$ and change the subscripts of the digraphs as follows: $G_u := G_{i_{u+1}},\ 1 \le u < |\Lambda'| - 1$, i.e., in the following *Steps 3* and *4* the ordered set $(G_{i_2}, G_{i_3}, \ldots, G_{|\Lambda'|+1})$ will be referred to as the ordered set $(G_1, G_2, \ldots, G_{|\Lambda'|})$.

Step 3: Implement Algorithm $SOL\_\sum \mathcal{C}_i$ [Algorithm $SOL\_\mathcal{C}_{max}$] with input set $B = \{G_1, G_2, \ldots, G_{|B|}\}$ defined in *Step 2* and with the same vector p.

Step 4: Set $\Lambda' := \Lambda' \setminus \{G_1\}$ and $\Lambda^T(G) := \Lambda^T(G) \cup \{G_1\}$.
IF $|\Lambda'| \ge 2$ **THEN GOTO** *Step 2* **ELSE GOTO** *Step 5*.

Step 5: **FOR** each digraph $G_s \in \Lambda^T(G)$ **DO**
BEGIN
Calculate vector $p^{(s)} \in T$ such that strong dominance relation $G_s \prec_{p^{(s)}} G_k$ holds for each digraph $G_k \in \Lambda^T(G) \setminus \{G_s\}$.
IF there does not exist such a vector $p^{(s)} \in T$
THEN set $\Lambda^T(G) := \Lambda^T(G) \setminus \{G_s\}$
END STOP.

Obviously, solution $\Lambda^T(G)$ generated by Algorithm $MINSOL\_\sum \mathcal{C}_i$ satisfies the conditions of Theorem 3.5 and hence this solution is minimal. However, Step 5 may be rather complicated, at least it needs to be discussed in more detail. As the desired vector $p^{(s)}$ for digraph G_s, we can test the vector $p_{ij}(r)$ calculated by formula

(3.46) in Algorithm $SOL\_\sum\mathcal{C}_i$, where $r = \overline{r}^B_{ks} + \epsilon$ with ϵ being a small positive real number. This vector will be either sufficient for Step 5 or not. In the latter case, i.e., when for the vector $p^{(s)}$ the strong dominance relation $G_s \prec_{p^{(s)}} G_k$ does not hold for at least one digraph $G_k \in \Lambda^T(G) \setminus \{G_s\}$, the realization of Step 5 in Algorithm $MINSOL\_\sum\mathcal{C}_i$ may be more sophisticated. In our experiments (see Chapter 5) we test Algorithm $MINSOL^*\_\sum\mathcal{C}_i$ (Algorithm $MINSOL^*\_\mathcal{C}_{max}$) which consists of Steps 1 – 4 of the above Algorithm $MINSOL\_\sum\mathcal{C}_i$ (Algorithm $MINSOL\_\mathcal{C}_{max}$). If for the solution $\Lambda^T(G) = \Lambda^*(G)$ generated by Algorithm $MINSOL^*\_\sum\mathcal{C}_i$ the inequality $|\Lambda^*(G)| \leq 2$ holds, then set $\Lambda^*(G)$ obviously satisfies the conditions of Theorem 3.5 and therefore this solution is minimal. If $|\Lambda^*(G)| > 2$, solution $\Lambda^*(G)$ may be not minimal. Indeed, even if $\Lambda^*(G) = \{G_1, G_2, G_3\}$, Algorithm $MINSOL^*\_\sum\mathcal{C}_i$ only guarantees that no digraph from the set $\Lambda^*(G)$ dominates another digraph from the set $\Lambda^*(G)$. However, it might be that two digraphs 'jointly dominate' the remaining one which is not recognized by Algorithm $MINSOL^*\_\sum\mathcal{C}_i$. Nevertheless, Algorithm $MINSOL^*\_\sum\mathcal{C}_i$ often constructs a minimal solution even if $|\Lambda^T(G)| > 2$. Indeed, it is easy to see that, if a schedule is the unique optimal schedule in the interior of its stability region, then dominance relation $\preceq_D$ implies the strong dominance relation $\prec_D$ (except points at the boundary of the stability region, where an optimal schedule usually is not unique). Fortunately, as it was shown in [SSW97] by computational experiments, a mean flow time optimal schedule is uniquely determined for most job shop problems provided that the processing times are non-negative real numbers (not necessarily integers as it is often assumed in classical scheduling theory), and thus, due to the test of the dominance relation $\preceq_D$, Algorithm $MINSOL^*\_\sum\mathcal{C}_i$ usually constructs a minimal solution.

Next, we present three algorithms for constructing a solution B (for any regular criterion Φ) used as input set in Algorithm $SOL\_\sum\mathcal{C}_i$ (Algorithm $SOL\_\mathcal{C}_{max}$). The first one (called Algorithm $EXPL$) is based on an explicit enumeration of all semiactive schedules for the case of a classical job shop problem. The other two algorithms (called $B\&B1$ and $B\&B2$) are of the branch-and-bound type and may be used for the job shop problem $\mathcal{J}/a_i \leq p_i \leq b_i/\Phi$ with uncertain numerical input data and any regular criterion.

Algorithm $EXPL$

Input: Polytope T, a weighted mixed graph $G(p) = (Q(p), A, E), p \in T$.
Output: Optimal digraph $G_s(p)$, a set $B = \{G^T_1, G^T_2, \ldots, G^T_{|B|}\} \subseteq \Lambda(G)$.

Step 1: Generate all feasible digraphs $\Lambda(G) = \{G_1, G_2, \ldots, G_\lambda\}$ by an explicit enumeration of the permutations of the operations Q_k for $k = 1, 2, \ldots, m$, and by testing whether the generated digraph has a circuit.

Step 2: Calculate the values Φ^a_k, Φ^b_k and Φ^p_k for each digraph G_k, $k = 1, 2, \ldots, \lambda$, and transform each digraph G_k into digraph G^T_k.

Step 3: Find a digraph G^T_s such that $\Phi^b_s = \min\{\Phi^b_k \ : \ G_k \in \Lambda(G)\}$.

Step 4: Set $B = \{G^T_k \ : \ G_k \in \Lambda(G),\ \Phi^b_s > \Phi^a_k\}$ **STOP**.

As follows from Lemma 3.5, the set $\Lambda(G) \setminus B$ contains only digraphs G_k such that dominance relation $G_s^T \preceq_T G_k^T$ holds (see Step 4 of Algorithm $EXPL$), and Algorithm $EXPL$ excludes only such digraphs from the set $\Lambda(G)$. To present branch-and-bound algorithms, we need the following preliminary arguments.

Both branch-and-bound algorithms realize an implicit enumeration scheme which may be represented by a branching tree. Each vertex of this tree is a mixed graph $G_{(s)} = (Q, A_{(s)}, E_{(s)})$ with $A \subseteq A_{(s)}$ and $E_{(s)} \subseteq E$. The root of the tree is a mixed graph $G = G_{(1)}$, and a pair $G_{(s)}$ and $G_{(k)}$ is connected by the arc $(G_{(s)}, G_{(k)})$ if and only if the mixed graph $G_{(k)}$ is obtained directly from the mixed graph $G_{(s)}$ by orientating one edge. In both branch-and-bound algorithms under consideration, an edge is oriented only if it is a conflict one, i.e., when both orientations of this edge imply a conflict with previously calculated earliest start times. Next, we give a formal definition of a conflict edge. For a mixed graph $G_{(s)} = (Q, A_{(s)}, E_{(s)})$ with $[O_{ij}, O_{uv}] \in E_{(s)}$, let us define the following three digraphs:

$$G_{s^0} = (Q, A_{(s)}, \emptyset),$$
$$G_{s'} = (Q, A_{(s)} \cup \{(O_{ij}, O_{uv})\}, \emptyset) \text{ and}$$
$$G_{s''} = (Q, A_{(s)} \cup \{(O_{uv}, O_{ij})\}, \emptyset).$$

Definition 3.7 *An edge $[O_{ij}, O_{uv}] \in E_{(s)}$ of the mixed graph $G_{(s)}$ is called a conflict edge, if there exists a vector $p \in T$ such that the following inequalities (3.67) and (3.68) hold:*

$$l^p_{s^0}(O_{uv}) < l^p_{s'}(O_{uv}), \tag{3.67}$$

$$l^p_{s^0}(O_{ij}) < l^p_{s''}(O_{ij}). \tag{3.68}$$

Obviously, if inequalities (3.67) and (3.68) hold, then each orientation of the edge $[O_{ij}, O_{uv}]$ implies an increase of the value $l^p_{s'}(O_{uv})$ or the value $l^p_{s''}(O_{ij})$. To verify whether an edge is a conflict one, we can use the following necessary conditions.

Lemma 3.9 *An edge $[O_{ij}, O_{uv}] \in E_{(s)}$ is not a conflict edge if one of the following inequalities (3.69) or (3.70) holds:*

$$l^a_{s^0}(O_{uv}) \ge l^b_{s^0}(O_{ij}) + b_{ij}, \tag{3.69}$$

$$l^a_{s^0}(O_{ij}) \ge l^b_{s^0}(O_{uv}) + b_{uv}. \tag{3.70}$$

Proof. It is easy to see that inequality (3.67) may hold only if the maximal path ending in vertex O_{uv} includes the arc (O_{ij}, O_{uv}), i.e., if

$$l^p_{s'}(O_{uv}) = l^p_{s^0}(O_{ij}) + p_{ij}. \tag{3.71}$$

Similarly, inequality (3.68) may hold only if

$$l^p_{s''}(O_{ij}) = l^p_{s^0}(O_{uv}) + p_{uv}. \tag{3.72}$$

First, suppose that inequality (3.69) holds. For any vector $p \in T$, we have $l^p_{s^0}(O_{uv}) \geq l^a_{s^0}(O_{uv}) \geq l^b_{s^0}(O_{ij}) + b_{ij} \geq l^p_{s^0}(O_{ij}) + p_{ij}$. Taking into account (3.71), we conclude that inequality $l^p_{s^0}(O_{uv}) \geq l^p_{s'}(O_{uv})$ holds which means that edge $[O_{ij}, O_{uv}]$ is not a conflict one.

Now, suppose that inequality (3.70) holds. For any vector $p \in T$, we have $l^p_{s^0}(O_{ij}) \geq l^a_{s^0}(O_{ij}) \geq l^b_{s^0}(O_{uv}) + b_{uv} \geq l^p_{s^0}(O_{uv}) + p_{uv}$. Taking into account (3.72), we conclude that inequality $l^p_{s^0}(O_{ij}) \geq l^p_{s''}(O_{ij})$ holds which means that edge $[O_{ij}, O_{uv}]$ is not a conflict one. $\diamond$

For each edge $[O_{ij}, O_{uv}] \in E_{(s)}$, one can calculate a *conflict measure* as follows:

$$\min\{\max\{0,\ l^p_{s^0}(O_{ij}) + p_{ij} - \bar{l}^p_{s^0}(O_{uv})\},\ \max\{0,\ l^p_{s^0}(O_{uv}) + p_{uv} - \bar{l}^p_{s^0}(O_{ij})\}\},$$

where $\bar{l}^p_{s^0}(O_{ij})$ denotes the *latest start time* of operation O_{ij}, i.e., the difference between the weight of the critical path μ in digraph G_{s^0} and the maximal weight of the path in G_{s^0} starting from vertex O_{ij}:

$$\bar{l}^p_s(O_{ij}) = l^p(\mu) - \sum_{O_{uv} \in [\nu]} p_{uv},$$

where path ν has the maximal weight among all paths in digraph G_{s^0} starting from O_{ij} and ending in vertex $O_{ln_l}, J_l \in J$. The conflict measure gives the smallest possible increase of the earliest start time of the operation due to the orientation of this edge (e.g., for a non-conflict edge this measure is equal to zero). So, in order to branch a set $\Lambda(G_{(s)})$ into two subsets $\Lambda(G_{(s')})$ and $\Lambda(G_{(s'')})$, where

$$G_{(s')} = (Q,\ A_{(s)} \cup \{(O_{ij}, O_{uv})\},\ E_{(s)} \setminus \{[O_{ij}, O_{uv}]\}) \text{ and}$$
$$G_{(s'')} = (Q,\ A_{(s)} \cup \{[O_{uv}, O_{ij}]\},\ E_{(s)} \setminus \{[O_{ij}, O_{uv}]\}),$$

we select the edge $[O_{ij}, O_{uv}]$ which has the largest value of the conflict measure. We use the following lower bound in both branch-and-bound algorithms. For any digraph $G_t = (Q, A_{(s)} \cup A_t, \emptyset) \in \Lambda(G_{(s)})$, the bound

$$\sum_{i=1}^{n} l^p_t(O_{in_i}) \geq \sum_{i=1}^{n} l^p_{s^0}(O_{in_i}) \tag{3.73}$$

is valid since the set of arcs in the digraph $G_{s^0} = (Q, A_{(s)}, \emptyset)$ is a subset of the arcs in digraph G_t. Note that, if digraph $G_{(s)}$ has no conflict edge, there exists a digraph $G_t \in \Lambda(G_{(s)})$ such that condition (3.73) is realized as equality. To construct such a digraph, we have to replace each remaining edge $[O_{ij}, O_{uv}] \in E_s$ by the arc (O_{ij}, O_{uv}) if inequality (3.69) holds, or by the arc (O_{uv}, O_{ij}) if inequality (3.70) holds. Obviously, for each p_{ij} and p_{uv} with $a_{ij} \leq p_{ij} \leq b_{ij}$ and $a_{uv} \leq p_{uv} \leq b_{uv}$, all operations in the resulting digraph will have the same earliest start times as in the digraph G_{s^0}. We use the latter as a stopping rule for branching the set $\Lambda(G_{(s)})$. Next, we present an algorithm for constructing a set of k schedules which are the best for the input vector $p \in T$ of the processing times and which will be used as the input set B in Algorithm $SOL\_\sum \mathcal{C}_i$ or Algorithm $SOL\_\mathcal{C}_{max}$ depending on the chosen objective function values $\Phi^p_s = L^p_s$ and $\Phi^p_s = l^p_s$, respectively.

Algorithm $B\&B1$

Input: Polytope T, weighted mixed graph $G(p) = (Q(p), A, E), p \in T$, number k of the best generated digraphs.
Output: Optimal digraph $G_s(p)$, set $B = \{G_1, G_2, \ldots, G_k\} \subseteq \Lambda(G)$.

Step 1: Set $X = \{G\} := \{G_{(1)}\}$, $Y = \emptyset$ and $\Phi = \infty$.

Step 2: **IF** $X = \emptyset$ **THEN GOTO** *Step 8*;
ELSE select a mixed graph $G_{(s)} \in X$ with the smallest value $\Phi^p_{s^0}$ and set $X := X \setminus \{G_{(s)}\}$.

Step 3: **IF** mixed graph $G_{(s)}$ has no conflict edge **THEN GOTO** *Step 6*.

Step 4: Select conflict edge $[O_{ij}, O_{uv}] \in E_{(s)}$ with the largest conflict measure.

Step 5: **IF** $\Phi^p_{s'} < \Phi$ **THEN** set $X := X \cup \{G_{(s')}\}$;
IF $\Phi^p_{s''} < \Phi$ **THEN** set $X := X \cup \{G_{(s'')}\}$ **GOTO** *Step 2*.

Step 6: **IF** $|Y| < k$ **THEN** set $Y := Y \cup \{G_{(s)}\}$ **GOTO** *Step 2*;
ELSE IF $\Phi^p_{s^0} < \Phi$ (where $\Phi = \Phi^p_t$) **THEN** set $Y := Y \cup \{G_{(s)}\} \setminus \{G_{(t)}\}$.

Step 7: Calculate $\Phi = \max\{\Phi^p_t \ : \ G_{(t)} \in Y\}$ **GOTO** *Step 2*.

Step 8: Construct set $\Lambda(G_{(t)})$ for each mixed graph $G_{(t)} \in Y$.

Step 9: Select subset B of k best digraphs from set $\cup_{G_{(t)} \in Y} \Lambda(G_{(t)})$.

Step 10: Calculate $\Phi^* = \min\{\Phi^b_s \ : \ G_s \in B\}$ and set $B := B \setminus \{G_t \ : \ \Phi^a_t \ge \Phi^*\}$
STOP.

In Algorithm $B\&B1$, the lower bound for the objective function is calculated in Step 7, branching is realized in Step 5, and the stopping rule of branching is realized in Step 3. Step 6 has a special form in order to construct the k best schedules (instead of only one optimal schedule). Steps 8 and 9 are also necessary only if $k > 1$. Indeed, if $k = 1$, then it is sufficient to consider only one best schedule from the set $\Lambda(G_{(s)})$, and for any mixed graph $G_{(s)} = (Q, A_{(s)}, E_{(s)})$, the set $\Lambda(G_{(s)})$ has at least one best schedule $G_u \in \Lambda(G_{(s)})$ for which Φ^p_u reaches the minimal possible value Φ^p_s, where $G_{s^0} = (Q, A_{(s)}, \emptyset)$ (condition (3.73) turns into an equality). But, if $k > 1$, we have to generate also other schedules from the set $\Lambda(G_{(s)})$. Unfortunately, we cannot use Algorithm $EXPL$ for a fast generation of the set $\Lambda(G_{(s)})$ because the edges of the set $E \setminus E_{(s)}$ are already oriented. Step 8 realizes a procedure based on the sequential orientation of non-conflict edges, which is essentially slower than the permutation enumeration used in Algorithm $EXPL$.

Using sufficiency of Lemma 3.5, Algorithm $B\&B2$ aims to construct a set of schedules which necessarily dominate all other schedules from the set $\Lambda(G)$ in the polytope T. Steps

1-5 and Steps 8-10 in Algorithm $B\&B2$ are similar to those in Algorithm $B\&B1$. So, in the following we describe only Steps 6 and 7 of Algorithm $B\&B2$, which are different from those in Algorithm $B\&B1$.

Algorithm $B\&B2$ **(specific part)**

Step 6: **IF** $\Phi^a_{s^0} \leq \Phi$ **THEN** set $Y := Y \cup \{G_{(s)}\}$.

Step 7: Calculate $\Phi = \min\{\Phi^b_{(t)} \ : \ G_{(t)} \in Y\}$ **GOTO** *Step 2.*

In Section 5.4, we present computational results for randomly generated classical job shop problems solved by the above algorithms coded in Fortran-77.

Example 3.2 (continued). *As it was noted, the solution $\Lambda^*(G)$ and the minimal solution $\Lambda^T(G)$ of the scheduling problem with uncertain processing times may be not unique. From Remark 3.5 it follows that fixing the vector $p \in T$ and the choice of an optimal digraph $G_s(p)$ have a large influence on the resulting solution for criterion $\mathcal{C}_{max}$. For the job shop problem $\mathcal{J}3/n=3, a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$ from Example 3.2, we find a solution set $\Lambda^*(G)$ with different initial real vectors $p \in T$ (see column 1 in Table 3.15) the components of which are taken from the closed intervals $[a_{ij}, b_{ij}]$ (vectors a and b are given in Table 3.7). The three algorithms EXPL, B&B1 and B&B2 construct the same set $B = \{G_1, G_2, \ldots, G_{12}\}$ for Example 3.2 as it was constructed with the initial vector p^0 (see page 143), but the digraphs from the set B form another order according to non-decreasing mean flow time objective functions values with different feasible vectors (see column 3 in Table 3.15). We calculate the following sums of job completion times with the initial vectors $p^1, p^2, \ldots, p^9$ from Table 3.15 (to avoid a confusion, we leave the same subscript of the digraphs indicating the location according to non-decreasing values of the function $\sum \mathcal{C}_i$ calculated with the vector p^0 (see page 143)):*
$L_5^{p^1} = 482, L_2^{p^1} = 486, L_1^{p^1} = 486, L_9^{p^1} = 512, L_3^{p^1} = 516, L_4^{p^1} = 536, L_7^{p^1} = 566, L_6^{p^1} = 596, L_8^{p^1} = 636, L_{12}^{p^1} = 666, L_{11}^{p^1} = 676, L_{10}^{p^1} = 686;$
$L_2^{p^2} = 450, L_1^{p^2} = 470, L_5^{p^2} = 470, L_9^{p^2} = 500, L_3^{p^2} = 500, L_4^{p^2} = 520, L_7^{p^2} = 550, L_6^{p^2} = 580, L_8^{p^2} = 580, L_{10}^{p^2} = 630, L_{12}^{p^2} = 650, L_{11}^{p^2} = 660;$
$L_2^{p^3} = 455, L_1^{p^3} = 460, L_5^{p^3} = 505, L_3^{p^3} = 510, L_4^{p^3} = 515, L_7^{p^3} = 520, L_8^{p^3} = 520, L_9^{p^3} = 555, L_6^{p^3} = 570, L_{10}^{p^3} = 575, L_{11}^{p^3} = 635, L_{12}^{p^3} = 645;$
$L_1^{p^4} = 365, L_2^{p^4} = 370, L_3^{p^4} = 395, L_4^{p^4} = 415, L_5^{p^4} = 420, L_6^{p^4} = 435, L_8^{p^4} = 435, L_7^{p^4} = 445, L_9^{p^4} = 450, L_{10}^{p^4} = 485, L_{11}^{p^4} = 495, L_{12}^{p^4} = 505;$
$L_1^{p^5} = 438, L_2^{p^5} = 460, L_3^{p^5} = 497.5, L_4^{p^5} = 502.5, L_5^{p^5} = 510, L_6^{p^5} = 532.5, L_7^{p^5} = 538.5, L_8^{p^5} = 547.5, L_9^{p^5} = 560, L_{10}^{p^5} = 612.5, L_{11}^{p^5} = 617.5, L_{12}^{p^5} = 627.5;$
$L_1^{p^6} = 510, L_2^{p^6} = 550, L_4^{p^6} = 590, L_5^{p^6} = 600, L_3^{p^6} = 600, L_6^{p^6} = 630, L_7^{p^6} = 630, L_8^{p^6} = 660, L_9^{p^6} = 670, L_{11}^{p^6} = 740, L_{10}^{p^6} = 740, L_{12}^{p^6} = 750;$
$L_1^{p^7} = 460, L_4^{p^7} = 515, L_7^{p^7} = 520, L_2^{p^7} = 520, L_5^{p^7} = 550, L_8^{p^7} = 560, L_3^{p^7} = 580, L_6^{p^7} = 585, L_{10}^{p^7} = 635, L_9^{p^7} = 640, L_{11}^{p^7} = 660, L_{12}^{p^7} = 670;$
$L_2^{p^8} = 497, L_5^{p^8} = 497, L_1^{p^8} = 498, L_9^{p^8} = 527, L_3^{p^8} = 528, L_4^{p^8} = 558, L_7^{p^8} = 578, L_6^{p^8} =$

Table 3.15: *Solution of problem* $\mathcal{J}3/n=3, a_i \le p_i \le b_i/\sum \mathcal{C}_i$ *for different initial vectors* $p \in T$

Initial vector $p^j \in T$	G_s	Set B	$G_k,\ G_s \not\preceq_T G_k$	$\overline{r}^B_{ks}$
1	2	3	4	5
$p^1=(60,20,46,$ $30,70,80,50,30)$	G_5	$B=\{G_5,G_2,G_1,G_9,G_3,G_4,$ $G_7,G_6,G_8,G_{12},G_{11},G_{10}\}$	G_1	$\overline{\varrho}^B_5(p^1\in T)=\overline{r}^B_{1,5}=0.5$
			G_2	$\overline{r}^B_{2,5}=0.6667$
			G_3	$\overline{r}^B_{3,5}=5.6667$
			G_4	$\overline{r}^B_{4,5}=7.7143$
			G_6	$\overline{r}^B_{6,5}=12.3333$
			G_8	$\overline{r}^B_{8,5}=12.8333$
			G_7	$\overline{r}^B_{7,5}=14$
			G_{10}	$\overline{r}^B_{10,5}=19$
	Solution: $\Lambda^*(G)=\{G_1,G_2,G_3,G_4,G_5,G_6,G_8,G_7,G_{10}\}$ Minimal solution: $\Lambda^T(G)=\{G_1,G_2,G_5\}$			
$p^2=(60,20,50,$ $30,50,80,50,30)$	G_2	$B=\{G_2,G_1,G_5,G_9,G_3,G_4,$ $G_7,G_6,G_8,G_{10},G_{12},G_{11}\}$	G_1	$\overline{\varrho}^B_2(p^2\in T)=\overline{r}^B_{1,2}=3.3333$
			G_5	$\overline{r}^B_{5,2}=15$
			G_4	$\overline{r}^B_{4,2}=16.6667$
			G_8	$\overline{r}^B_{8,2}=16.6667$
			G_7	$\overline{r}^B_{7,2}=20$
	Solution: $\Lambda^*(G)=\{G_1,G_2,G_4,G_5,G_7,G_8\}$ Minimal solution: $\Lambda^T(G)=\{G_1,G_2,G_5\}$			
$p^3=(80,20,50,$ $10,65,60,45,35)$	G_2	$B=\{G_2,G_1,G_5,G_3,G_4,G_7,$ $G_8,G_9,G_6,G_{10},G_{11},G_{12}\}$	G_1	$\overline{\varrho}^B_2(p^3\in T)=\overline{r}^B_{1,2}=1$
			G_8	$\overline{r}^B_{8,2}=11$
			G_4	$\overline{r}^B_{4,2}=15$
			G_5	$\overline{r}^B_{5,2}=18.3333$
			G_7	$\overline{r}^B_{7,2}=20$
	Solution: $\Lambda^*(G)=\{G_1,G_2,G_4,G_5,G_7,G_8\}$ Minimal solution: $\Lambda^T(G)=\{G_1,G_2,G_5\}$			
$p^4=(60,20,45,$ $10,50,60,30,30)$	G_1	$B=\{G_1,G_2,G_3,G_4,G_5,G_6,$ $G_8,G_7,G_9,G_{10},G_{11},G_{12}\}$	G_2	$\overline{\varrho}^B_1(p^4\in T)=\overline{r}^B_{2,1}=1.25$
			G_5	$\overline{r}^B_{5,1}=18.3333$
	Solution: $\Lambda^*(G)=\{G_1,G_2,G_5\}$ Minimal solution: $\Lambda^T(G)=\{G_1,G_2,G_5\}$			
$p^5=(70,30,52.5,$ $20,60,70,40,35)$	G_1	$B=\{G_1,G_2,G_3,G_4,G_5,G_6,$ $G_7,G_8,G_9,G_{10},G_{11},G_{12}\}$	G_2	$\overline{\varrho}^B_1(p^5\in T)=\overline{r}^B_{2,1}=2.5$
			G_5	$\overline{r}^B_{5,1}=9.2857$
	Solution: $\Lambda^*(G)=\{G_1,G_2,G_5\}$ Minimal solution: $\Lambda^T(G)=\{G_1,G_2,G_5\}$			
$p^6=(80,40,60,$ $30,70,80,50,40)$	G_1	$B=\{G_1,G_2,G_4,G_5,G_3,G_6,$ $G_7,G_8,G_9,G_{11},G_{10},G_{12}\}$	G_2	$\overline{\varrho}^B_1(p^6\in T)=\overline{r}^B_{2,1}=8$
			G_5	$\overline{r}^B_{5,1}=18.75$
	Solution: $\Lambda^*(G)=\{G_1,G_2,G_5\}$ Minimal solution: $\Lambda^T(G)=\{G_1,G_2,G_5\}$			
$p^7=(80,40,60,$ $30,65,60,30,35)$	G_1	$B=\{G_1,G_4,G_7,G_2,G_5,G_8,$ $G_3,G_6,G_{10},G_9,G_{11},G_{12}\}$	G_2	$\overline{\varrho}^B_1(p^7\in T)=\overline{r}^B_{2,1}=12$
			G_5	$\overline{r}^B_{5,1}=19.2857$
	Solution: $\Lambda^*(G)=\{G_1,G_2,G_5\}$ Minimal solution: $\Lambda^T(G)=\{G_1,G_2,G_5\}$			

Table 3.15 (continuation)**:** *Solution of problem* $\mathcal{J}3/n=3, a_i \le p_i \le b_i/\sum \mathcal{C}_i$ *for different initial vectors* $p \in T$

Initial vector $p^j \in T$	G_s	Set B	$G_k,\ G_s \not\preceq_T G_k$	$\overline{r}^B_{ks}$
1	2	3	4	5
$p^8=(60,20,49,$ $30,69,80,50,40)$	G_2	$B=\{G_2,G_5,G_1,G_9,G_3,G_4,$ $G_7,G_6,G_8,G_{12},G_{10},G_{11}\}$	G_5	$\overline{\varrho}^B_2(p^8\in T)=\overline{r}^B_{5,2}=0$
			G_1	$\overline{r}^B_{1,2}=0.1667$
			G_4	$\overline{r}^B_{4,2}=15$
			G_8	$\overline{r}^B_{8,2}=17.0909$
			G_7	$\overline{r}^B_{7,2}=20$
	Solution: $\Lambda^*(G)=\{G_1,G_2,G_4,G_5,G_7,G_8\}$ Minimal solution: $\Lambda^T(G)=\{G_1,G_2,G_5\}$			
	G_5	$B=\{G_5,G_2,G_1,G_9,G_3,G_4,$ $G_7,G_6,G_8,G_{12},G_{10},G_{11}\}$	G_2	$\overline{\varrho}^B_5(p^8\in T)=\overline{r}^B_{2,5}=0$
			G_1	$\overline{r}^B_{1,5}=0.125$
			G_3	$\overline{r}^B_{3,5}=5.1667$
			G_4	$\overline{r}^B_{4,5}=7.6250$
			G_6	$\overline{r}^B_{6,5}=12$
			G_8	$\overline{r}^B_{8,5}=12.5455$
			G_7	$\overline{r}^B_{7,5}=14$
			G_{10}	$\overline{r}^B_{10,5}=18.8$
	Solution: $\Lambda^*(G)=\{G_1,G_2,G_3,G_4,G_5,G_6,G_7,G_8,G_{10}\}$ Minimal solution: $\Lambda^T(G)=\{G_1,G_2,G_5\}$			
$p^9=(60,20,50,$ $30,70,80,50,30)$	G_1	$B=\{G_1,G_2,G_5,G_9,G_3,G_4,$ $G_7,G_6,G_8,G_{12},G_{11},G_{10}\}$	G_2	$\overline{\varrho}^B_1(p^9\in T)=\overline{r}^B_{2,1}=0$
			G_5	$\overline{r}^B_{5,2}=0$
	Solution: $\Lambda^*(G)=\{G_1,G_2,G_5\}$ Minimal solution: $\Lambda^T(G)=\{G_1,G_2,G_5\}$			
	G_2	$B=\{G_2,G_1,G_5,G_9,G_3,G_4,$ $G_7,G_6,G_8,G_{12},G_{11},G_{10}\}$	G_1	$\overline{\varrho}^B_2(p^9\in T)=\overline{r}^B_{1,2}=0$
			G_5	$\overline{r}^B_{5,2}=0$
			G_4	$\overline{r}^B_{4,2}=15$
			G_8	$\overline{r}^B_{8,2}=17.2727$
			G_7	$\overline{r}^B_{7,2}=20$
	Solution: $\Lambda^*(G)=\{G_1,G_2,G_4,G_5,G_7,G_8\}$ Minimal solution: $\Lambda^T(G)=\{G_1,G_2,G_5\}$			
	G_5	$B=\{G_5,G_1,G_2,G_9,G_3,G_4,$ $G_7,G_6,G_8,G_{12},G_{11},G_{10}\}$	G_1	$\overline{\varrho}^B_5(p^9\in T)=\overline{r}^B_{1,5}=0$
			G_2	$\overline{r}^B_{2,5}=0$
			G_9	$\overline{r}^B_{9,5}=5$
			G_4	$\overline{r}^B_{4,5}=7.1429$
			G_6	$\overline{r}^B_{6,5}=12$
			G_8	$\overline{r}^B_{8,5}=12.7273$
			G_7	$\overline{r}^B_{7,5}=14$
			G_{10}	$\overline{r}^B_{10,5}=19$
	Solution: $\Lambda^*(G)=\{G_1,G_2,G_4,G_5,G_6,G_7,G_8,G_9,G_{10}\}$ Minimal solution: $\Lambda^T(G)=\{G_1,G_2,G_5\}$			

$618, L_8^{p^8}=646, L_{12}^{p^8}=698, L_{10}^{p^8}=706, L_{11}^{p^8}=708;$
$L_1^{p^9}=490, L_2^{p^9}=490, L_5^{p^9}=490, L_9^{p^9}=520, L_3^{p^9}=520, L_4^{p^9}=540, L_7^{p^9}=570, L_6^{p^9}=$

$600, L_8^{p^9} = 640, L_{12}^{p^9} = 670, L_{11}^{p^9} = 680, L_{10}^{p^9} = 690$.

First, we construct a solution $\Lambda^*(G)$ by Algorithm $SOL\_\sum \mathcal{C}_i$, and then a minimal solution $\Lambda^T(G)$ by Algorithm $MINSOL^*\_\sum \mathcal{C}_i$. In column 2 of Table 3.15, we give the chosen optimal digraph $G_s(p^j)$ for the fixed vector p^j. Set $B := B \setminus \{G_s\}$ ordered according to non-decreasing values $L_u^{p^j}, u \in \{1, 2, \ldots, |B|\}$, is presented in column 4. For digraph $G_k \in B$, we test the dominance relation $G_s \preceq_T G_k$ using Lemma 3.5 with the objective function values $\Phi_s^p = L_s^p$ (see Step 2 of Algorithm $SOL\_\sum \mathcal{C}_i$). For all digraphs G_k with $G_s \not\preceq_T G_k$ (see column 4), we calculate the value $\overline{r}_{ks}^B$ using formulas (3.49) and (3.55) from Theorem 3.6 for the input vector p^j. Column 5 presents a non-decreasing order of the values $\overline{r}_{ks}^B$ calculated according to (3.55).

Due to Theorem 3.6, it follows that the minimal value of $\overline{r}_{ks}^B$ is equal to the relative stability radius $\overline{\varrho}_s^B(p^j \in T)$. An optimal digraph G_s and all digraphs G_k, for which the dominance relation $G_s \preceq_T G_k$ does not hold, form the solution $\Lambda^*(G)$ of the scheduling problem $\mathcal{J}3/n=3, a_i \le p_i \le b_i/\sum \mathcal{C}_i$.

As we see, a choice of the initial vector $p \in T$ gives different solution sets. The best choice of such a feasible vector is still an open question. We fix, for example, the vector p^4 (p^6) equal to the given lower bound a (upper bound b, respectively) of the feasible polytope T (see Table 3.7), and the vector p^5 with components $p_{ij} = \frac{1}{2}(b_{ij} - a_{ij})$. Such a choice gives the following solution set $\Lambda^*(G) = \Lambda^T(G) = \{G_1, G_2, G_5\}$. Note that there is no minimal solution set with a smaller cardinality than $\Lambda^T(G) = \{G_1, G_2, G_5\}$. Moreover, each of the digraphs G_1, G_2 and G_5 is the unique optimal one for some vector $p \in T$, i.e., for example, the following strong dominance relations hold:

$$
\begin{array}{ll}
G_1 \prec_{p^0} G_k, & G_k \in \Lambda(G) \setminus \{G_1\}, \\
G_2 \prec_{p^2} G_k, & G_k \in \Lambda(G) \setminus \{G_2\}, \text{ and} \\
G_5 \prec_{p^1} G_k, & G_k \in \Lambda(G) \setminus \{G_5\} \text{ (see column 3 of Table 3.15).}
\end{array}
$$

It means that there is no proper subset of the set $\{G_1, G_2, G_5\}$ which is a solution of problem $\mathcal{J}3/n=3, a_i \le p_i \le b_i/\sum \mathcal{C}_i$ and so the solution $\Lambda^*(G) = \{G_1, G_2, G_5\}$ is minimal in the sense of inclusion and in the sense of cardinality equal to 3. As we see, our developed Algorithm $SOL\_\sum \mathcal{C}_i$ may construct some redundant schedules, which are not necessarily in a minimal solution set $\Lambda^T(G)$ of problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$.

As it was noted, for the scheduling problem with the makespan criterion (see Remark 3.5) not only fixing the initial vector $p \in T$ has a large influence on the resulting solution, but also the choice of an optimal digraph G_s for the further calculations, if it is not uniquely determined. For the vectors p^8 and p^9, the optimal schedule is not unique. For example, there are two optimal digraphs $G_2(p^8)$ and $G_5(p^8)$ for the vector p^8, therefore we run Algorithm $SOL\_\sum \mathcal{C}_i$ twice. First, we order digraphs in the set B as follows $\{G_2, G_5, G_1, G_9, G_3, G_4, G_7, G_6, G_8, G_{12}, G_{10}, G_{11}\}$ and we make all calculations according to Algorithm $SOL\_\sum \mathcal{C}_i$ for the first digraph G_2 in the set B. Secondly, we order digraphs in the set B as follows $\{G_5, G_2, G_1, G_9, G_3, G_4, G_7, G_6, G_8, G_{12}, G_{10}, G_{11}\}$ and we make all calculations for the first digraph G_5. Thus in the first case, solution $\Lambda^*(G)$ consists of six schedules since there are five digraphs G_k for which the dominance relation $G_2 \preceq_T G_k$ does not hold. In the second case, the solution $\Lambda^*(G)$ consists of nine schedules since there are eight digraphs G_k, $G_5 \not\preceq_T G_k$. Since there are three optimal digraphs for the vector p^9 at all, the corresponding cardinalities of the obtained solutions $\Lambda^*(G)$ are 3, 6 and 9, respectively (see Table 3.15).

As we see from Table 3.15, the covering of the polytope T by the minimal number of stability balls (cardinality-minimal covering) is an interesting question. The cardinality-minimal covering seems to be a more difficult problem than an inclusion-minimal covering. However, this dissertation deals only with the investigation of inclusion-minimal coverings. At least we do not know a practicable algorithm for constructing a cardinality-minimal covering of polytope T.

3.7 Comments and References

The following lemma which was proven in [STW98, LSSW97]. The proof of such a lower bound for $\widehat{\varrho}_s^B(p \in T)$ can be found in [LSSW97]. for calculating $\widehat{\varrho}_s(p) = \widehat{\varrho}_s^{\Lambda(G)}(p \in R_+^q)$ were given in [Sot91b]. In the computational experiments discussed in [SSW97] (see Section 5.1), a direct enumeration has been used for $|E| \leq 30$.

In [SSW97], a bound of the stability radius was used to restrict the number of digraphs considered for calculating the stability radius (these results are given in Section 2.4). The main results presented in this chapter have been published in [LSSW97, SSW97].

The results which are proven in this chapter have been published in [LSSW98, Sot99b, Sot99c, SW00].

In [Sot99b, SW00], an approach for dealing with 'strict uncertainty' based on a stability analysis of an optimal semiactive schedule was generalized for an uncertain job shop problem with any given regular criterion Φ.

Necessary and sufficient conditions for a solution of problem $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$ have been derived in [LSSW98].

Theorem 3.4 implies the following corollary from [LSSW98] which characterizes a single-element solution of problem $\mathcal{J}/a_i \leq p_i \leq b_i/\Phi$, which is necessarily a minimal solution.

A similar theorem formulated for problem $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$ has been proven in [LSSW98].

Similar theorems and the above lemma formulated for the special case of problem $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$ have been proven in [LSSW98].

To present necessary and sufficient conditions for $\overline{\varrho}_s^B(p \in T) = 0$, we need the following auxiliary lemma proven in [BSW96]. Let Ω_k denote the set $\{\Omega_k^u : \ u = 1, 2, \ldots, \omega_k\}$.

Since the optimality of a schedule s depends on the critical path in the digraph G_s, we focused on the set of paths in digraph G_s which may be critical (see Lemma 3.3 and Theorem 3.1). To restrict the set of paths which may be critical, one can use a dominance relation for the set of paths reduced in Section **??** (see Definition 1.3). Although this relation is based only on the structural input data, its use may considerably reduce the set of paths which may be critical. To deal with problem $\mathcal{G}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ in Section 3.2, we generalized the dominance relation (see Definition 3.3) due to the numerical input data as well. On the basis of this dominance relation, we presented a characterization of a zero relative stability radius (Theorem 3.1) and an infinite relative stability radius (Theorem 3.2). In Section 3.3, we have given a formula for calculating the exact value of the relative stability radius (Theorem 3.3). These results may be considered as a mathematical background for developing algorithms for solving problem $\mathcal{G}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$.

This approach seems to be particularly useful when the structural input data are fixed before applying a scheduling algorithm but the numerical input data are uncertain, especially when a lot of scheduling problems with the same (or close) structural input data have to be solved.

In Section 3.4, we have defined a solution of job shop problems with uncertain processing times. The network presentation of the structural input data (precedence and capacity constraints) and a minimal solution have been discussed in Section 3.4, where the decision process is presented as the construction of a set of schedules (digraphs) which dominate other schedules. To solve problem $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$, we developed an approach for calculating the relative stability radius. Theorem 3.6 generalized the results from [BSW96], where the stability radius $\overline{\varrho}_s^B(p \in T)$ was investigated for the special case when $B = \Lambda(G)$ and the whole space R_+^q being used instead of the polytope T. Theorem 3.8 (Theorem 3.7) provides necessary and sufficient conditions for a zero (for an infinitely large, respectively) relative stability radius.

In Sections **??** and **??**, upper bounds for the relative stability radii $\widehat{\varrho}_s^B(p \in T)$ and $\overline{\varrho}_s^B(p \in T)$ have been used to restrict the number of digraphs compared with an optimal digraph for calculating the relative stability radius. These bounds have been derived for $\widehat{\varrho}_s(p)$ and $\overline{\varrho}_s(p)$ in [SSW97] and will be used in Chapter 5.

Note that in this dissertation the term 'time' is used in three different senses: namely, as the time for processing an operation, as the time for decision-making, and as the time for running an algorithm. We hope that these different uses of the same word do not cause any confusion.

Table **??** combines the main notations used in this chapter for the general shop problem. The common notations for job shop and general shop problems are given in Table 5.11 at page 286.

Table 3.16: *Notations for the job shop problem with uncertain processing times*

Symbols	Description
O_{ik}	Operation of job $J_i \in J$ processed by machine $M_{i_k} \in M$ at stage $k \in \{1,2,\ldots,n_i\}$ of the technological route $(M_{i_1}, M_{i_2}, \ldots, M_{i_{n_i}})$
O_{in_i}	Last operation of job $J_i \in J$, $1 \le i \le n$
Q^J	Set of all operations for processing jobs $J_i \in J$: $Q^J = \cup_{i=1}^{n} Q^{J_i} = \{Q_{ij} \ : \ J_i \in J, j = 1,2,\ldots,n_i\}$
Q_k^J	Set of all operations which has to be processed by machine $M_k \in M$: $Q^J = \cup_{k=1}^{m} Q_k^J$ and $Q_k^J \cap Q_l^J = \emptyset$ if $k \neq l$
$(M_{i_1}, M_{i_2}, \ldots, M_{i_{n_i}})$	Technological route for processing job $J_i \in J$ in a job shop
$(M_1, M_2, \ldots, M_m)$	Technological route for processing job $J_i \in J$ in a flow shop
$(O_{i1}, O_{i2}, \ldots, O_{in_i})$	Ordered set of operations for processing job $J_i \in J$
M_{i_k}	Machine from set M which has to process operation O_{ik}
s_{ij}	Start time of operation O_{ij}
c_{ij}	Completion time of operation O_{ij}
p_{ij}	Processing time of operation O_{ij}
a_{ij}	Lower bound for the processing time of operation O_{ij}
b_{ij}	Upper bound for the processing time of operation O_{ij}
$l_k^p(O_{ij})$	Earliest start time of operation O_{ij} in digraph G_k
L_k^p	Critical sum of weights for digraph $G_k(p)$: $L_k^p = \max_{u \in \{1,\ldots,\omega_k\}} \sum_{\nu \in \Omega_k^u} l^p(\nu)$
(Q^J, A^J, E^J)	Mixed graph defining the structural input data
Ω_k^u	Set of representatives of the family of sets $(H_k^i)_{J_i \in J}$
ω_k	Number of different sets of representatives for digraph G_k : $\omega_k = \prod_{i=1}^{n} \lvert H_k^i \rvert$
Ω_k	Set of all sets of representatives for digraph G_k : $\{\Omega_k^u : \ u = 1,2,\ldots,\omega_k\}$
ω_k^T	Number of different sets of representatives for G_k^T : $\omega_k^T = \prod_{i=1}^{n} \lvert H_k^i(T) \rvert$
$n_{ij}(\Omega_k^u)$	Number of copies of operation O_{ij} contained in multiset $\{[\nu] : \nu \in \Omega_k^u\}$
$\Omega_k^{u^*}$	Critical set of digraph $G_k(p) \in \Lambda(G)$, $u^* \in \{1,2,\ldots,\omega_k\}$
$\Omega_k(p)$	Set of all critical sets $\Omega_k^{u^*}$ of digraph $G_k \in \Lambda(G)$ with $p \in R_+^q$
Ω_{sk}	Family of sets of representatives $\Omega_s^v, v \in \{1,2,\ldots,\omega_k\}$, such that there does not exist a set Ω_k^u with $n_{ij}(\Omega_s^v) \le n_{ij}(\Omega_k^u)$ for each $i = 1,2,\ldots,n$ and each $j = 1,2,\ldots,n_i$
Ω_{sk}^*	Set of all sets of representatives Ω_s^v, $v \in \{1,2,\ldots,\omega_s^T\}$, such that for the vector $p^* \in T$ defined by formula (3.53) inequality $\sum_{\mu \in \Omega_s^v} l^{p^*}(\mu) > \sum_{\nu \in \Omega_k^u} l^{p^*}(\nu)$ holds for each set Ω_k^u, $u \in \{1,2,\ldots,\omega_k^T\}$
T	Polytope in the space R_+^q of feasible vectors of job processing times
$\Lambda^*(G) \subseteq \Lambda(G)$	Solution of a scheduling problem with uncertain processing times
$\Lambda^T(G) \subseteq \Lambda^*(G)$	Minimal solution of a problem with uncertain processing times
Γ	Set of competitive digraphs
γ	Number of competitive digraphs
$\widehat{\varrho}_s^B(p \in T)$	Relative stability radius of digraph G_s with respect to polytope T for makespan criterion
$\overline{\varrho}_s^B(p \in T)$	Relative stability radius of digraph G_s with respect to polytope T for mean flow time criterion
$H_k^i(T)$	Subset of dominant paths of set H_k^i with respect to polytope T
$G_s \preceq_D G_k$	Dominance relation implying $\Phi_s^p \le \Phi_k^p$ for each vector $p \in D$
$G_s \prec_D G_k$	Strong dominance relation implying $\Phi_s^p < \Phi_k^p$ for each vector $p \in D$

Chapter 4

Two-Machine Scheduling with Uncertain Processing Intervals

In this chapter, the flow shop and job shop scheduling problems with n jobs processed on two machines are addressed where jobs have random processing times, set-up times are random or machines may have intervals of non-availability.

In Sections 4.1 and 4.2, it is assumed that the probability distributions of random processing times are unknown, and only their lower and upper bounds are given before scheduling. In such a case, there may not exist a unique schedule that remains optimal for all feasible realizations of job processing times, and we look for a minimal set of schedules which dominates all feasible schedules for the given objective function. We obtain necessary and sufficient conditions for the case when it is possible to fix order of two jobs for makespan criterion (in spite of uncertainty of numerical input data). We characterize the easiest case of a two-machine flow shop problem with uncertain job processing times, i.e., we prove necessary and sufficient conditions for existence of single schedule which is dominant for makespan criterion. On the other hand, we describe the set of the hardest problems for which any semiactive schedule may be unique optimal one for some feasible realizations of job processing times. Along with off-line scheduling problem, we consider on-line problem when a part of the schedule is already realized. We show how to use additional information available for on-line problem to obtain better solution than that constructed for off-line version of scheduling problem. All the conditions proven may be tested in polynomial time of n. We show how to use these results for two-stage supply chain scheduling.

4.1 Flow Shop with Uncertain Processing Times

Two types of stochastic flow shop scheduling problems have been addressed in the OR literature. In a stochastic *job* problem, processing times are assumed to be random variables following certain probability distributions (mainly the normal distribution has been considered). In a stochastic scheduling environment, the objective is to find such a *scheduling policy* that minimizes expected value of the given objective function. For stochastically minimizing makespan (this criterion is denoted as $E\mathcal{C}_{max}$), flow shop problem was considered in [ET00, Kam99], among others.

In a stochastic *machine* problem, processing times are deterministic (i.e., fixed before scheduling), while job completion times are random variables as a result of machine breakdowns or other reasons of machine non-availability. In [All99, AM94], a flow shop problem with stochastically minimizing total flow time $E\sum \mathcal{C}_i$ was considered.

In this section, we address the stochastic job problem and consider frequent situation arising in supply chain scheduling when it is hard to obtain reliable probability distributions for random processing times and assuming a specific probability distribution is not realistic. As such, a schedule obtained by assuming a certain probability distribution (e.g., normal) may not be even close to the optimal schedule for the practical realization of the process. More precisely, we consider the following non-preemptive two-machine flow shop scheduling problem with bounded random processing times and makespan criterion.

Two machines $M = \{M_1, M_2\}$ have to process n jobs $J = \{J_1, J_2, ..., J_n\}$ with the same machine (technological) route: (M_1, M_2). At first, each job $J_i \in J$ has to be processed by machine M_1 without preemption, and then by machine M_2 without preemption as well. All the n jobs are available to be processed from time $t = 0$.

In contrast to well-studied *deterministic* flow shop problem with two machines [Joh54], it is assumed that processing time p_{ij} of job $J_i \in J$ by machine $M_j \in M$ is not fixed before scheduling. In the realization of the process, p_{ij} may take any real value between lower bound a_{ij} and upper bound b_{ij} being given before scheduling. Moreover, the probability distribution of random processing time is unknown before scheduling. In such a case, there may not exist a unique schedule that remains optimal for all possible realizations of job processing times. And as a *solution* to this problem, we consider a minimal set of feasible schedules that dominates all the other ones.

Let $C_i(\pi_k)$ denote the completion time of job $J_i \in J$ in the schedule π_k, and criterion $\mathcal{C}_{max}$ denote minimization of schedule length $C_{max}(\pi_k)$ (makespan):

$$\mathcal{C}_{max} = \min_{\pi_k \in S^\pi} C_{max}(\pi) = \min_{\pi_k \in S^\pi} \{\max\{C_i(\pi) \ : \ J_i \in J\}\}$$

where S^π denotes the set of all feasible permutations (permutation schedules) including at least one optimal permutation for makespan criterion. This problem is denoted as $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$.

We have to emphasize that random processing times in the problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ are due to *external forces* in contrast to scheduling problems with *controllable* processing times. In the latter problem, the objective is to choose optimal processing times (which are under the control of the decision-maker) and optimal schedule with chosen processing times. The objective function depends on both job processing times and job completion times (see [Str95], among others).

In this section, we find a combination of *sufficient conditions* which put together *necessary conditions* for fixing the order of two jobs in a solution to problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$. For a two-machine flow shop scheduling problem with uncertain processing times, we prove necessary and sufficient conditions for the easiest case (when one schedule dominates all the others), and for the hardest case (when there does not exist any non-trivial schedule domination at all). In contrast to Chapter 3, where exponential algorithms based on exhausting enumeration of the feasible schedules have been derived for a general shop job shop problems, in this section, we develop polynomial procedures for solving special case of the problem when $m = 2$ and all the jobs have the same technological route.

The remainder of this section is organized as follows. First, we give main notations and formal definition for solution to a flow shop problem with uncertain processing times. Then extreme cases of the problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ are characterized. We show how to use Johnson's permutations for solving general case of a two-machine flow shop problem with uncertain processing times. At the end of this section, we present sufficient conditions for existence of dominant schedule when its part is already realized (in the case of on-line scheduling problem), and demonstrate how it is possible to use solution to problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ for supply chain scheduling.

Motivation and Definition

We address the special case of a stochastic two-machine flow shop scheduling problem. Namely, we assume that before scheduling, it is impossible to obtain reliable probability distribution for random processing time, and prior assuming a specific probability distribution is not realistic. Commonly, schedules obtained after assuming a certain probability distribution may be not even close to the optimal schedule that would be constructed on the basis of the exact knowledge of the actual job processing times. It has been observed that, although the probability distribution of job processing times may be unknown before scheduling, their upper and lower bounds are easy to obtain in many practical situations. As an example, we consider the following two-stage supply chain.

Every item (job) has to go through the processing by the supplier's machine M_1, followed by the manufacturer's machine M_2. In such a case, processing time p_{i1} has to include time for choosing a suitable supply for buying raw material, and transportation time from chosen raw material supply to the location of manufacturer's machine M_2. Thus, processing time p_{i1} may depend on a distance between chosen raw material supply and machine M_2, type of transport, traffic conditions, etc. Processing time p_{i2} may be uncertain before scheduling as well. It may depend of the intervals of availability of machine M_2, possible machine breakdown, raw material quality, appearance new unexpected item with high priority and close due date, personal qualifications and experience, etc. However, in such a supply chain environment we can estimate processing times p_{i1} and p_{i2} for the best possible case and for the worst possible case, and as a result we obtain reliable values a_{ij} and b_{ij}, $M_j \in M$, of lower and upper bounds of the processing times of all the items $J_i \in J$. Next, we show that information on the bounds of item processing times may be important and should be utilized in finding a solution to the supply chain scheduling.

In what follows, a set of all feasible vectors $p = (p_{1,1}, p_{1,2}, \ldots, p_{n1}, p_{n2})$ of job processing times will be denoted by T:

$$T = \{p \ : \ a_{ij} \leq p_{ij} \leq b_{ij}, \ J_i \in J, \ M_j \in M\}.$$

Let $S^\pi = \{\pi_1, \pi_2, \ldots, \pi_{n!}\}$ be a set of all permutations (sequences) of n jobs from set J:

$$\pi_k = \{J_{k_1}, J_{k_2}, \ldots, J_{k_n}\}, \ k \in \{1, 2, \ldots, n!\},$$

$$\{k_1, k_2, \ldots, k_n\} = \{1, 2, \ldots, n\}.$$

This set S^π defines *permutation schedules* that are *dominant* for the problem of two-machine flow shop with criterion $\mathcal{C}_{max}$ and random processing times (see [KN86]). In other

words, there exists at least one schedule with the same job sequence on both machines M_1 and M_2 which is optimal (i.e., this schedule *dominates* all the other feasible schedules) for the problem of two-machine flow shop scheduling with makespan criterion and random processing times.

Since the processing times are also random variables in the problem under consideration, set of permutation schedules is dominant for the problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$ as well. Set of permutations (schedules), which is sufficient to consider while solving problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$, has cardinality $n! = |S^\pi|$ since each of the two machines from set M may process set J of jobs in the same order. Next, we restrict further the set of feasible permutations sufficient to consider for solving problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$.

If equality $a_{ij} = b_{ij}$ holds for each job $J_i \in J$ and machine $M_j \in M$, then problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$ turns to usual flow shop problem $\mathcal{F}2//\mathcal{C}_{max}$ that is polynomially solvable due to Johnson (1954). We call problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$ as *uncertain* flow shop problem in contrast to problem $\mathcal{F}2//\mathcal{C}_{max}$ called *deterministic* flow shop problem.

Since preemption in the processing of each job $J_i \in J$ by each machine $M_j \in M$ is not allowed in the problem $\mathcal{F}2//\mathcal{C}_{max}$, each permutation $\pi_k \in S^\pi$ defines a unique set of the earliest job completion times $C_1(\pi_k), C_2(\pi_k), \ldots, C_n(\pi_k)$ which in turn defines a unique *semiactive* schedule (see Definition 1.1 at page 11). So in what follows, we will not distinguish a permutation $\pi_k \in S^\pi$ and a semiactive schedule defined by this permutation. At least, we certainly need such an agreement for the case of an uncertain problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$ in which job processing times are not fixed, and as a result it is impossible to calculate all the job completion times for the permutation $\pi_k \in S^\pi$. Therefore, for uncertain problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$ we will use notation $C_i(\pi_k, p)$ for the completion time of job $J_i \in J$, and notation $C_{max}(\pi_k, p) = \max\{C_i(\pi_k, p) \ : \ J_i \in J\}$ for the makespan if concrete vector $p \in T$ of job processing times will be treated.

In the famous paper by Johnson (1954), it was proven that set of permutation schedules is dominant for deterministic problem $\mathcal{F}2//\mathcal{C}_{max}$, and it takes $O(n \log n)$ time to construct permutation $\pi_i = (J_{i_1}, J_{i_2}, \ldots, J_{i_n})$ satisfying condition

$$\min\{p_{i_k 1}, p_{i_{k+1} 2}\} \le \min\{p_{i_{k+1} 1}, p_{i_k 2}\} \tag{4.1}$$

for each index $k = 1, 2, \ldots, n-1$. The latter permutation gives an optimal schedule to problem $\mathcal{F}2//\mathcal{C}_{max}$. In this schedule, each of the two machines from set M processes set J of jobs in the same order defined by the above permutation π_i. Algorithm for constructing optimal permutation $\pi_i \in S^\pi$ (called Johnson's permutation) for the deterministic problem $\mathcal{F}2//\mathcal{C}_{max}$ is based on the following rule.

Johnson's rule: *Partition set J into two (disjoint) subsets N_1 and N_2 with N_1 containing the jobs with $p_{i1} \le p_{i2}$ and N_2 the jobs with $p_{i1} \ge p_{i2}$. (The jobs with equality $p_{i1} = p_{i2}$ may be either in set N_1 or in set N_2.) In the optimal schedule, the jobs from set N_1 are processed first, and they are processed in non-decreasing order of p_{i1}. The jobs from set N_2 follow after jobs N_1 in non-increasing order of p_{i2}. (Ties may be broken arbitrary.)*

It should be noted that for the problem $\mathcal{F}2//\mathcal{C}_{max}$ some optimal schedule may be defined by a permutation from set S^π that does not satisfy condition (4.1). In other words, inequalities (4.1) are *sufficient* for optimality of permutation $\pi_i \in S^\pi$ but they are not *necessary* for permutation optimality. Nevertheless in what follows, the set of optimal permutations concerned for problem $\mathcal{F}2//\mathcal{C}_{max}$ will be restricted by Johnson's ones.

Our main aim is to construct a minimal set of dominant schedules for the problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$, and so we can remove all the redundant permutations of set S^π. In particular, if there is no feasible vector $p \in T$ such that permutation $\pi_k \in S^\pi$ is a Johnson's one for the deterministic problem $\mathcal{F}2//\mathcal{C}_{max}$ with vector p of job processing times, then we remove permutation π_k from further consideration. Summarizing, we introduce the following definition of a *solution* to uncertain problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$.

Definition 4.1 *Set of permutations $S^\pi(T) \subseteq S^\pi$ is called a solution to the uncertain problem $\mathcal{F}2//\mathcal{C}_{max}$, if for any feasible vector $p \in T$ set $S^\pi(T)$ contains at least one permutation, which is Johnson's one (and so it is optimal) for the deterministic problem $\mathcal{F}2//\mathcal{C}_{max}$ with vector p of job processing times, any proper subset of set $S^\pi(T)$ being not a solution to the problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$.*

Next, we prove necessary and sufficient conditions for the extreme cases of problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$ (namely, for the easiest case when $|S^\pi(T)| = 1$ and for the hardest case when $|S^\pi(T)| = n!$).

Extreme Cases of an Uncertain Flow shop Problem

Let us consider the easiest case of the problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$ with respect to its solution cardinality, i.e., when there exists a permutation $\pi_i \in S^\pi$ constituting a single-element set (singleton) that is solution $S^\pi(T) = \{\pi_i\}$ to the problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$. In other words, this permutation π_i has to be optimal for deterministic problem $\mathcal{F}2//\mathcal{C}_{max}$ with any feasible vector $p \in T$ of job processing times, and so singleton $\{\pi_i\}$ has to be a solution to the uncertain problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$ (due to Definition 4.1). Such a permutation π_i dominates all the permutations $\pi_k \in S^\pi$ with respect to set T in the following sense.

Definition 4.2 *Permutation $\pi_i \in S^\pi$ dominates permutation $\pi_k \in S^\pi$ with respect to T if inequality $C_{max}(\pi_i, p) \le C_{max}(\pi_k, p)$ holds for any feasible vector $p \in T$ of job processing times. Permutation π_i is called dominant if it dominates each permutation $\pi_k \in S$ with respect to T.*

Let closed intervals $[a_{i1}, b_{i1}]$ and $[a_{i2}, b_{i2}]$ have at most one common point. Then we call *left interval* of job $J_i \in J$ be the closed interval $[a_{ij}, b_{ij}]$ that is allocated to the left-hand side of the closed interval $[a_{ik}, b_{ik}]$ at the real number axis provided that $\{j, k\} = \{1, 2\}$ (in this case, inequality $a_{ij} \le b_{ik}$ must hold). Let N_{M_1} (N_{M_2}, respectively) denote a set of jobs with left intervals being intervals of job processing times on machine M_1 (machine M_2).

Theorem 4.1 *Solution $S^\pi(T) \subset S^\pi$ to the problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$ is a singleton, $|S^\pi(T)| = 1$, if and only if the following conditions hold:*

a) For each job $J_i \in J$ (possibly except one job $J_{i^*} \in J$), its intervals of the processing times on machines M_1 and M_2 have at most one common point;

b) For any pair of jobs with left intervals on the same machine, their left intervals have at most one common point;

c) If there exists job $J_{i^*} \in J$, the following two inequalities hold

$$a_{i^*1} \geq \max\{b_{i1} \ : \ J_i \in N_{M_1}\}, \tag{4.2}$$

$$a_{i^*2} \geq \max\{b_{j2} \ : \ J_j \in N_{M_2}\}. \tag{4.3}$$

PROOF. *Sufficiency.* Let conditions *a*), *b*), and *c*) hold in common. We have to construct optimal permutation $\pi_i \in S^\pi$ with $\{\pi_i\} = S^\pi(T)$ using Johnson's rule (i.e., desired permutation π_i has to dominate each permutation $\pi_k \in S^\pi$ with respect to T).

Due to condition *a*) sets N_{M_1} and N_{M_2} can be defined in such a way that one can partition job set J into three pairwise disjoint subsets: $J = N_{M_1} \cup N^* \cup N_{M_2}$ where set N^* consists of a single job J_{i^*} (if this job exists within set J, i.e., we have either $N^* = \{J_{i^*}\}$ or $N^* = \emptyset$). It is easy to see that both inclusions $N_{M_1} \subseteq N_1$ and $N_{M_2} \subseteq N_2$ may hold for any feasible vector $p \in T$ of job processing times.

From condition *b*) it follows that using Johnson's rule we can order all the jobs of set N_{M_1} (set N_{M_2}, respectively) in the same way for all feasible vectors $p \in T$ of job processing times. As a result, we obtain unique permutation π^a (permutation π^b) of all the jobs of set N_{M_1} (set N_{M_2}) disposed in non-decreasing (non-increasing) order of job processing times on machine M_1 (machine M_2). Thus if $N^* = \emptyset$, permutation (π^a, π^b) (obtained via concatenation of permutations π^a and π^b) provides the single-element solution $S^\pi(T) = \{(\pi^a, \pi^b)\}$ to the problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$.

Let there exist the above job $J_{i^*} \in J$. Then there may exist two possibilities depending on the realization of the processing times of job J_{i^*}, i.e., the following two cases *(i)* and *(ii)* have to be tested.

(i): $p_{i^*1} < p_{i^*2}$

In case *(i)*, job J_{i^*} has to belong to set N_1 and due to condition *c*) in the Johnson's permutation, job J_{i^*} has to be processed after all the other jobs of set N_1, i.e., Johnson's permutation has to be as follows: (π^a, J_{i^*}, π^b).

(ii): $p_{i^*1} \geq p_{i^*2}$

If $p_{i^*1} > p_{i^*2}$ (if $p_{i^*1} = p_{i^*2}$) job J_{i^*} has to belong (may belong) to set N_2 and due to condition *c*) in the Johnson's permutation, job J_{i^*} has to be processed (may be processed) before all the other jobs of set N_2. Therefore in case *(ii)*, Johnson's permutation may be just the same as in the case *(i)*, namely: (π^a, J_{i^*}, π^b).

Thus in both cases *(i)* and *(ii)*, we can obtain the same Johnson's permutation $\pi_i = (\pi^a, J_{i^*}, \pi^b)$ which is optimal for any feasible vector $p \in T$ of job processing times. Therefore, solution $S^\pi(T)$ to the problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ consists of a permutation: $S^\pi(T) = \{\pi_i\}$ which completes sufficiency sufficiency proof.

Necessity. We prove necessity of each condition *a*), *b*) and *c*) by contradiction method. Assume that $|S^\pi(T)| = 1$ but at least one of these conditions does not hold. Next, we examine violation either condition *c*) or *b*) or *a*) (in this order).

1. Let conditions *a*) and *b*) hold, while condition *c*) do not hold. Then for the job $J_{i^*} \in J$ there may exist two possible cases: *(c-1)* or *(c-2)*.

(c-1): Let inequality (4.38) from condition *c*) be violated, i.e., there exists such a job $J_{i^0} \in N_{M_1}$ that opposite inequality $a_{i^*1} < b_{i^0 1}$ holds. (Here, we assume that only job J_{i^0} violates inequality (4.38). The case when two or more jobs from set N_{M_1} violate inequality (4.38) may be considered similarly.)

Then we consider vector $p = (p_{1,1}, p_{1,2}, \ldots, p_{n1}, p_{n2}) \in T$ where $p_{i^0 1} = a_{i^0 1}$, $p_{i^* 1} = a_{i^* 1}$, $p_{i^* 2} = b_{i^* 2}$ and other components p_{ij} of the vector p take any feasible values: $a_{ij} \leq p_{ij} \leq b_{ij}$.

Since intervals of the processing times of job J_{i^*} on machines M_1 and M_2 have more than one common point, then we have inequality $a_{i^* 1} < b_{i^* 2}$ that implies inequality

$$p_{i^* 1} < p_{i^* 2}. \tag{4.4}$$

Therefore, inclusion $J_{i^*} \in N_1$ must hold for the considered vector $p \in T$ of job processing times. The above inequality $a_{i^* 1} < b_{i^0 1}$ implies inequality

$$p_{i^* 1} < p_{i^0 1}. \tag{4.5}$$

Thus, for the feasible vector p of job processing times Johnson's permutation has to be as follows: $\pi' = (\pi_a \setminus \{J_{i^0}\}, J_{i^*}, J_{i^0}, \pi_b)$, and due to strict inequalities (4.4) and (4.5) this order of jobs J_{i^*} and J_{i^0} has to be fixed in any Johnson's permutation which may be constructed for the above vector $p \in T$.

On the other hand, we can consider vector $p^0 = (p^0_{1,1}, p^0_{1,2}, \ldots, p^0_{n1}, p^0_{n2}) \in T$ with the same components as vector p has except the following three ones: $p^0_{i^0 1} = a_{i^0 1}$, $p^0_{i^* 1} = b_{i^* 1}$, $p^0_{i^* 2} = a_{i^* 2}$. Then we obtain optimal Johnson's permutation $\pi_i = (\pi^a, J_{i^*}, \pi^b)$ for the vector p^0 of job processing times (note that π' cannot be a Johnson's permutation for the vector p^0).

So due to Definition 1, any solution $S^\pi(T)$ to the problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ under consideration must include at least two permutations, namely: $\{\pi_i, \pi'\} \subseteq S^\pi(T)$. We get a contradiction with the assumption $|S^\pi(T)| = 1$.

(c-2) Let inequality (4.39) from condition *c*) be violated, i.e., there exists such a job $J_{j^0} \in N_{M_2}$, that opposite inequality $a_{i^* 2} < b_{j^0 2}$ holds. (Again, we assume that only job J_{j^0} violates inequality (4.39). The case when two or more jobs from set N_{M_2} violate inequality (4.39) may be considered similarly.)

Then we consider vector $p^* = (p^*_{1,1}, p^*_{1,2}, \ldots, p^*_{n1}, p^*_{n2}) \in T$ where $p^*_{j^0 2} = b_{j^0 2}$, $p^*_{i^* 1} = b_{i^* 1}$, $p^*_{i^* 2} = a_{i^* 2}$ and other components p^*_{ij} of the vector p^* take any feasible values: $a_{ij} \leq p^*_{ij} \leq b_{ij}$.

Since intervals of the processing times of job J_{i^*} on machines M_1 and M_2 have more than one common point, then we have inequality $a_{i^* 2} < b_{i^* 1}$ that implies inequality

$$p^*_{i^* 2} < p^*_{i^* 1}. \tag{4.6}$$

Therefore, inclusion $J_{i^*} \in N_2$ must hold for the vector p^* of job processing times. The above inequality $a_{i^* 2} < b_{j^0 2}$ implies inequality

$$p^*_{i^* 2} < p^*_{j^0 2}. \tag{4.7}$$

Thus for the vector p^*, Johnson's permutation has to be as follows: $\pi'' = (\pi_a, J_{j^0}, J_{i^*}, \pi_b \setminus \{J_{j^0}\})$ and due to strict inequalities (4.6) and (4.7) this order of jobs J_{j^0} and J_{i^*} has to be fixed in any Johnson's permutation which may be constructed for the vector p^* of job processing times.

Continuing in a similar way as in the case *(c-1)* we can conclude that any solution $S^\pi(T)$ to the problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ must include at least two permutations: $\{\pi_i, \pi''\} \subseteq S^\pi(T)$ that contradicts to assumption $|S^\pi(T)| = 1$.

2. Let condition *a*) hold, while condition *b*) does not hold. Then the following two cases *(b-1)* and *(b-2)* have to be tested.

(b-1) There exist two jobs $J_i \in N_{M_1}$, $J_j \in N_{M_1}$ with the following inequality:

$$\max\{a_{i1}, a_{j1}\} < \min\{b_{i1}, b_{j1}\}.$$

Then we can construct two feasible vectors $p' \in T$ and $p'' \in T$, $p' = (p'_{1,1}, p'_{1,2}, \ldots, p'_{n1}, p'_{n2})$, $p'' = (p''_{1,1}, p''_{1,2}, \ldots, p''_{n1}, p''_{n2})$, such that for the vector p' inequalities

$$\max\{a_{i1}, a_{j1}\} \le p'_{i1} < p'_{j1} \le \min\{b_{i1}, b_{j1}\} \tag{4.8}$$

hold, and for the vector p'' inequalities

$$\max\{a_{i1}, a_{j1}\} \le p''_{j1} < p''_{i1} \le \min\{b_{i1}, b_{j1}\} \tag{4.9}$$

hold. Other components p'_{uv} and p''_{uv} of the vectors p' and p'' take arbitrary but the same feasible values: $a_{uv} \le p'_{uv} = p''_{uv} \le b_{uv}$. Obviously, for both vectors p' and p'' jobs J_i and J_j have to belong to set N_{M_1}.

It is clear that for the vector p', Johnson's permutation has to look as $\pi_1 = (\ldots, J_i, \ldots, J_j, \ldots)$, while for the vector p'' Johnson's permutation has to look as $\pi_2 = (\ldots, J_j, \ldots, J_i, \ldots)$. Due to inequalities (4.8) (inequalities (4.9), respectively) this order of jobs J_i and J_j has to be the same in all the Johnson's permutations constructed for vector p' (vector p'') of job processing times. Due to Definition 4.1, the inclusion $\{\pi_1, \pi_2\} \subseteq S^{\pi}(T)$ has to hold for any solution $S^{\pi}(T)$ contradicting to assumption $|S^{\pi}(T)| = 1$.

(b-2) There exist two jobs $J_u \in N_{M_2}$ and $J_v \in N_{M_2}$ with the following inequality:

$$\max\{a_{u2}, a_{v2}\} < \min\{b_{u2}, b_{v2}\}.$$

Similar to the case *(b-1)* we can show that there are two feasible vectors $p^* \in T$ and $p^{**} \in T$ such that for vector p^* any Johnson's permutation looks as $\pi_3 = (\ldots, J_u, \ldots, J_v, \ldots)$, while for vector p^{**} any Johnson's permutation looks as $\pi_4 = (\ldots, J_v, \ldots, J_u, \ldots)$. Thus, due to Definition 4.1 we obtain inclusion $\{\pi_3, \pi_4\} \subseteq S^{\pi}(T)$ for any solution $S^{\pi}(T)$ contradicting to assumption $|S^{\pi}(T)| = 1$.

3. Let condition *a*) do not hold, i.e., there exist at least two jobs J_{i^*} and $J_{i^{**}}$ with the following inequalities:

$$\max\{a_{i^*1}, a_{i^*2}\} < \min\{b_{i^*1}, b_{i^*2}\},$$

$$\max\{a_{i^{**}1}, a_{i^{**}2}\} < \min\{b_{i^{**}1}, b_{i^{**}2}\}.$$

We can construct two vectors $p' \in T$ and $p'' \in T$ that for vector p' inequalities

$$\max\{a_{i^*1}, a_{i^*2}\} \le p'_{i^*1} < p'_{i^*2} \le \min\{b_{i^*1}, b_{i^*2}\},$$

$$\max\{a_{i^{**}1}, a_{i^{**}2}\} \le p'_{i^{**}2} < p'_{i^{**}1} \le \min\{b_{i^{**}1}, b_{i^{**}2}\}$$

hold, and for vector p'' inequalities

$$\max\{a_{i^*1}, a_{i^*2}\} \le p''_{i^*2} < p''_{i^*1} \le \min\{b_{i^*1}, b_{i^*2}\},$$

$$\max\{a_{i^{**}1}, a_{i^{**}2}\} \le p''_{i^{**}1} < p''_{i^{**}2} \le \min\{b_{i^{**}1}, b_{i^{**}2}\}$$

hold. Other components p'_{uv} and p''_{uv} of the vectors p' and p'' take arbitrary but the same feasible values: $a_{uv} \le p'_{uv} = p''_{uv} \le b_{uv}$.

In only two jobs J_{i^*} and $J_{i^{**}}$ violate condition a), then for the vector p', Johnson's permutation is $\pi_5 = (\pi_a, J_{i^*}, J_{i^{**}}, \pi_b)$, while for the vector p'', Johnson's permutation is $\pi_6 = (\pi_a, J_{i^{**}}, J_{i^*}, \pi_b)$. Due to Definition 4.1, inclusion $\{\pi_5, \pi_6\} \subseteq S^\pi(T)$ holds contradicting to assumption $|S^\pi(T)| = 1$. The case when three or more jobs violate condition a) may be considered similarly.

Thus, equality $|S^\pi(T)| = 1$ implies conditions a), b) and c), and this completes the proof.

◇

Example 4.1 *We consider example of the problem $\mathcal{F}2/a_i \le p_i \le b_i, n = 5/\mathcal{C}_{max}$ with feasible intervals of job processing times given in Table 4.1. These intervals are presented also in Figure 4.1. It is easy to convince that conditions a), b) and c) from Theorem 4.1 are satisfied. Therefore, there exists minimal solution of Example 4.1 which is singleton.*

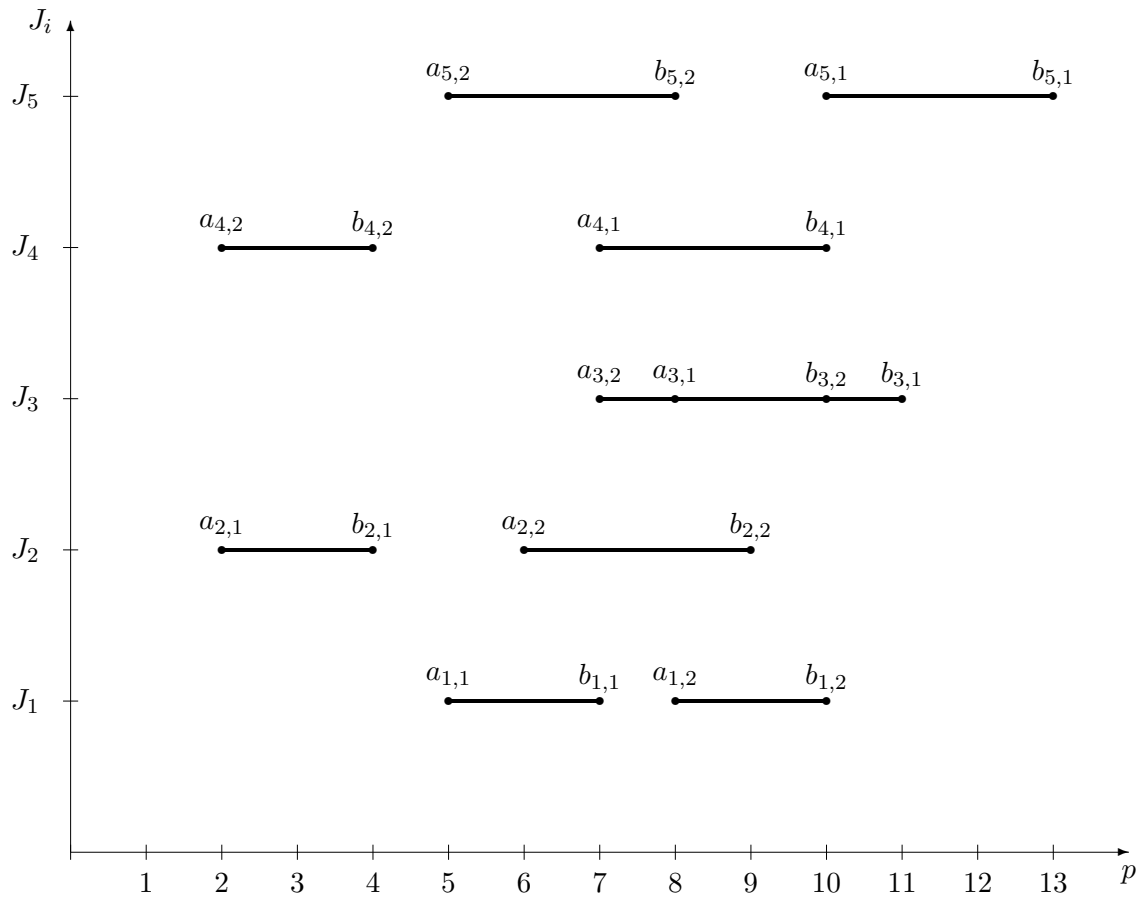

Figure 4.1: *Feasible intervals of job processing times in Example 4.1*

The above constructive proof of sufficiency of Theorem 4.1 allows to find this minimal solution: $S^\pi(T) = \{\pi\} = \{(J_2, J_1, J_3, J_4, J_5)\}$. Thus, permutation $\pi = (J_2, J_1, J_3, J_4, J_5)$ of five jobs is optimal (for criterion makespan) for any vector p of job processing times belonging to polytope T defined by Table 4.1.

Table 4.1: *Lower and upper bounds of job processing times in Example 4.1*

i	a_{i1}	b_{i1}	a_{i2}	b_{i2}
1	5	7	8	10
2	2	4	6	9
3	8	11	7	10
4	7	10	2	4
5	10	13	5	8

Next, we consider the worst case of the problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ with respect to its solution cardinality. Namely, we will analyze the case when solution to the uncertain scheduling problem has maximal cardinality: $|S^\pi(T)| = n!$. We will use the following notations: $a_{max} = \max\{a_{ij} \ : \ J_i \in J, M_j \in M\}$ and $b_{min} = \min\{b_{ij} \ : \ J_i \in J, M_j \in M\}$.

Theorem 4.2 *If for the problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ inequality*

$$a_{max} < b_{min} \tag{4.10}$$

holds, then $S^\pi(T) = S^\pi$.

PROOF. Let $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n})$ be any permutation from set S^π. In order to show that this belongs to solution $S^\pi(T)$, it is sufficient to construct vector construct vector $p^k = (p^k_{1,1}, p^k_{1,2}, \ldots, p^k_{n1}, p^k_{n2}) \in T$ with the following condition:

$$\begin{aligned} a_{max} \leq p^k_{k_1 1} < p^k_{k_1 2} = p^k_{k_2 1} < p^k_{k_2 2} = p^k_{k_3 1} < \ldots \\ < p^k_{k_{n-1} 2} = p^k_{k_n 1} < p^k_{k_n 2} \leq b_{min}. \end{aligned} \tag{4.11}$$

Due to inequality (4.40), there exist $2n$ real numbers p^k_{ij}, $J_i \in J$, $M_j \in M$, that condition (4.41) holds (since the set of real numbers is compact). Thus, we can construct the above vector $p^k \in T$ for the permutation $\pi_k \in S^\pi$.

For such a vector $p^k \in T$ all the jobs from set J belong to set N_1 (i.e., set N_2 is empty). It is easy to convince that within permutation π_k all n jobs are ordered with respect to Johnson's rule. Therefore, permutation π_k is optimal Johnson's permutation for vector p^k. Since $p^k_{i1} \neq p^k_{j1}$ for each pair of jobs J_i and J_j, $i \neq j$, Johnson's permutation π_k is unique for the vector $p^k \in T$ of job processing times. Therefore, permutation π_k has to belong to set $S^\pi(T)$.

Since permutation π_k is assumed to be arbitrary in set S^π, the last claim remains correct for any permutation $\pi_i \in S^\pi$ with corresponding vector $p^i \in T$ of job processing times. Thus, Definition 1 implies that all permutations from set S^π have to belong to set $S^\pi(T)$: $S^\pi(T) = S^\pi$. This completes the proof.

$\diamond$

Obviously, to test conditions *a*) and *c*) takes $O(n)$ time and to test condition *b*) takes $O(n \log n)$. Thus, conditions of Theorem 4.1 may be tested in $O(n \log n)$ time. Complexity of calculating a_{max} and b_{min} defines complexity of testing condition (4.40) of Theorem 4.2. Thus, to test condition of Theorem 4.2 takes $O(n)$ time.

Example 4.2 *Let us consider example* $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{\max}$ *with numerical input date given both in Figure 4.2 and in Table 4.2. From Figure 4.2 it follows that segment* $[6, 7]$ *belongs to each feasible interval of job processing times. Thus, for this example condition 4.40 of Theorem 4.2 holds:* $p^a_{\max} = 6 < 7 = p^b_{\min}$. *Due to Theorem 4.2 minimal solution* $S^\pi(T)$ *for Example 4.2 with set* T *of feasible vectors of job processing times given by Table 4.2 has maximal possible cardinality equal to* $n! = 5! = 120$ *since* $S^\pi(T) = S^\pi$.

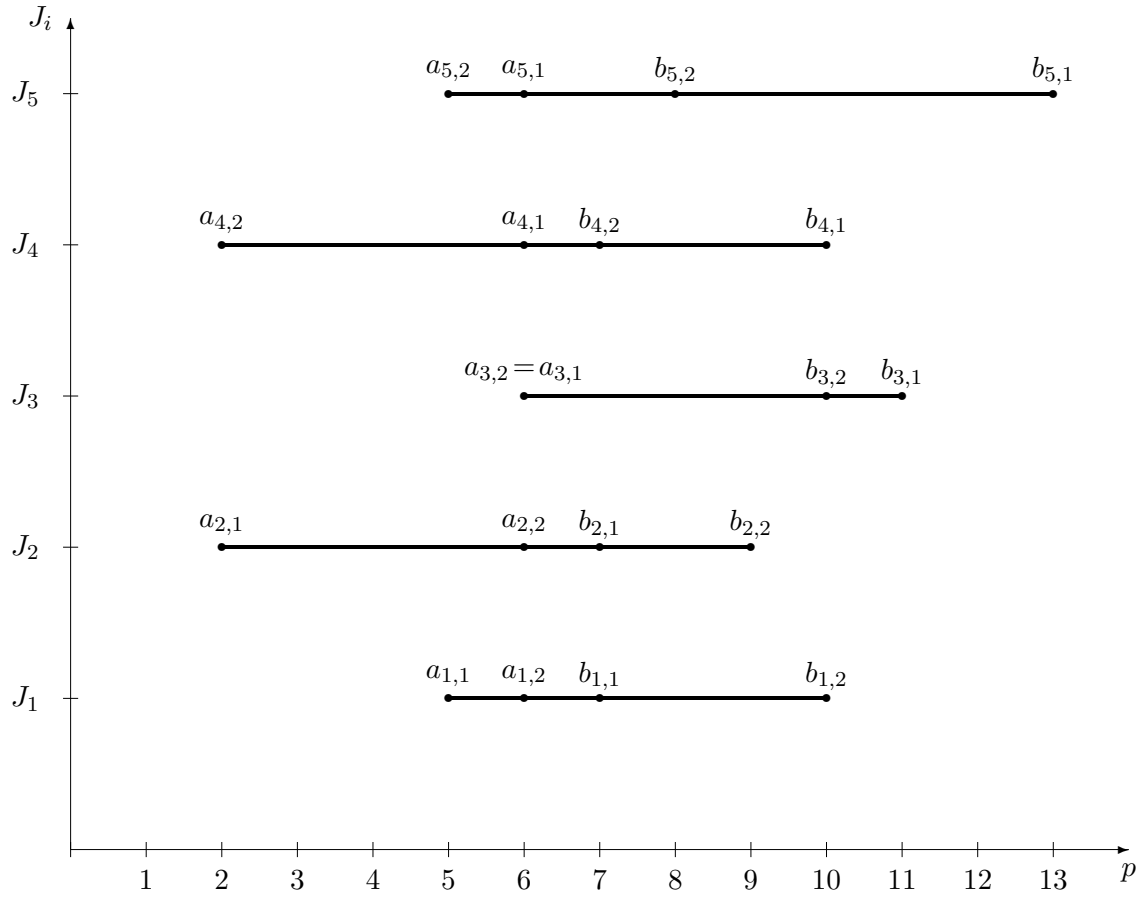

Figure 4.2: *Intervals of job processing times satisfying Theorem 4.2*

Theorems 4.1 and 4.2 characterize the extreme cases of a solution to the problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$, i.e., when $|S^\pi(T)| = 1$ and $S^\pi(T) = S^\pi$. Next, we consider general case of the problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$.

General Case of an Uncertain Two-Machine Flow Shop Problem

We show how to delete all redundant permutations from set S^π while solving uncertain flow shop problem. To this end, we will fix the order of two jobs $J_w \in J$ and $J_i \in J$ in the desired solution $S^\pi(T)$ if there exists at least one Johnson's permutation of the form $\pi_k = (s_1, J_w, s_2, J_i, s_3) \in S^\pi$ for any feasible vector $p \in T$ of job processing times.

Table 4.2: *Lower and upper bounds of job processing times in Example 4.2*

i	a_{i1}	b_{i1}	a_{i2}	b_{i2}
1	5	7	6	10
2	2	7	6	9
3	6	11	6	10
4	6	10	2	7
5	6	13	5	8

Theorem 4.3 *If the following condition (j) holds:*

$$b_{w2} \leq a_{w1} \text{ and } b_{i1} \leq a_{i2}, \tag{4.12}$$

then for each vector $p \in T$ of job processing times there exists permutation $\pi_k = (s_1, J_i, s_2, J_w, s_3) \in S^\pi$ that is a Johnson's permutation for the problem $\mathcal{F}2//\mathcal{C}_{max}$ with vector $p \in T$ of job processing times.

PROOF. Let polytope T satisfy inequalities (4.12). We consider any fixed vector $p \in T$ of the processing times. Let in the vector p, jobs J_w and J_i have processing times p_{w1}, p_{w2}, p_{i1} and p_{i2}. From the first inequality in (4.12) it follows that $p_{w2} \leq p_{w1}$, and therefore there exists a Johnson's permutation for vector p of the processing times in which job J_w is included in the set N_2. From the second inequality in (4.12) it follows that $p_{i1} \leq p_{i2}$ and therefore there exists a Johnson's permutation for vector p of the processing times in which job J_i is included in the set N_1.

Let π_1 denote such a Johnson's permutation in which inclusions $J_j \in N_1$ and $J_w \in N_2$ simultaneously hold. Due to Johnson's rule, job J_i has to precede job J_w in permutation π_1, i.e., permutation π_1 may be represented in the following form $\pi = \pi_1 = (s_1, J_i, s_2, J_w, s_3)$. For this fixed vector p of the processing times permutation π_1 dominates any other permutation from the set S^π with respect to $\{p\}$.

To complete the proof we note that vector p was taken from set T arbitrarily.

◇

Theorem 4.4 *If the following condition (jj) holds:*

$$b_{i1} \leq a_{w1} \text{ and } b_{i1} \leq a_{i2}, \tag{4.13}$$

then for each vector $p \in T$ of job processing times there exists permutation $\pi_k = (s_1, J_i, s_2, J_w, s_3) \in S^\pi$ that is a Johnson's permutation for the problem $\mathcal{F}2//\mathcal{C}_{max}$ with vector $p \in T$ of job processing times.

PROOF. Let polytope T satisfy inequalities (4.13). We consider any fixed vector $p \in T$ of job processing times. Let in the vector $p \in T$ job J_i and J_w have processing times p_{i1}, p_{i2}, p_{w1} and p_{w2}. From the second inequality in (4.13) it follows $p_{i1} \leq p_{i2}$. Therefore, there exists a Johnson's permutation π_2 for vector p of the processing times in which job J_i is included in the set N_1. In permutation π_2, job J_w may be either in the set N_1 or in the set N_2. Next, we consider both cases.

Case 1. Let job J_w belong to set N_1 in Johnson's permutation π_2. Then from the first inequality in (4.13) it follows $p_{i1} \leq b_{i1} \leq a_{w1} \leq p_{w1}$. Since $p_{i1} \leq p_{w1}$ for jobs J_i and J_w in the set N_1, then job J_i has to precede job J_w in the Johnson's permutation π_2, i.e., permutation π_2 may be represented in the form $\pi_2 = (s_1, J_i, s_2, J_w, s_3)$. For this fixed vector p of the processing times permutation π_2 dominates any other permutation from set S^π with respect to $\{p\}$.

Case 2. Let job J_i belong to set N_2 in the Johnson's permutation π_2. By an argument similar to the proof of Theorem 4.3, it is easy to show that job J_w precedes job J_i in the Johnson's permutation π_2, and so permutation π_2 dominates any permutation from set S^π.

Thus, we can conclude that for each vector $p \in T$ of the processing times in both cases there exists permutation $\pi_2 = \pi = (s_1, J_j, s_2, J_w, s_3) \in S^\pi$ which dominates each permutation from set S^π.

◇

Theorem 4.5 *If the following condition (jjj) holds:*

$$b_{w2} \leq a_{w1} \text{ and } b_{w2} \leq a_{i2}, \tag{4.14}$$

then for each vector $p \in T$ of job processing times there exists permutation $\pi_k = (s_1, J_i, s_2, J_w, s_3) \in S^\pi$ that is a Johnson's permutation for the problem $\mathcal{F}2//\mathcal{C}_{max}$ with vector $p \in T$ of job processing times.

Proof. It is easy to see that inequalities (4.14) in Theorem 4.5 coincide with inequalities (4.13) in Theorem 4.4 if interchanging job J_i and job J_w. Thus, proof of Theorem 4.5 is analogous to proof of Theorem 4.4.

◇

It should be noted that in all Theorems 4.3, 4.4 and 4.5 subsequence s_2 may be empty, i.e., jobs J_i and J_w may be adjacent in the above Johnson's permutation: $\pi_k = (s_1, J_i, J_w, s_3)$.

If for a pair of jobs $J_i \in J$ and $J_w \in J$ at least one condition *(j), (jj)* or *(jjj)* holds, then in a solution $S^\pi(T)$ to the problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ one can fix the order of these two jobs as follows: $J_i \rightarrow J_w$, i.e., job J_i has to be processed before job J_w. After testing pairs of inequalities (4.12), (4.13) and (4.14) we can define (partial) strong order in set J of jobs. Of course, the more jobs will be involved in a strong order, the more redundant permutations will be deleted.

Let $G = (J, A)$ denote a digraph with vertex set J and arc set A. It is assumed that arc (J_i, J_w) belongs to set A if and only if for each feasible vector $p \in T$ there exist permutation $\pi_k \in S^\pi$ of the form $(s_1, J_i, s_2, J_w, s_3)$ that is a Johnson's permutation for the deterministic problem $\mathcal{F}2//\mathcal{C}_{max}$ with vector p of job processing times. In other words, vertices J_i and J_w are non-adjacent in digraph G (i.e., neither arc (J_i, J_w) nor arc (J_w, J_i) belongs to set A) if and only if order of jobs J_i and J_w cannot be fixed in a solution $S^\pi(T)$ to the problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$.

Next, we show that sufficient conditions *(j), (jj)* and *(jjj)* put together necessary condition for fixing the order of two jobs in a solution to problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$. More precisely, we prove that if no condition of Theorems 4.3, 4.4 or 4.5 holds for two

jobs J_i and J_w, then the order $J_i \to J_w$ of these jobs cannot be fixed in a solution $S^\pi(T)$. Moreover, if in addition no analogous condition holds for the opposite order $J_w \to J_i$, then due to Definition 4.1 at least one permutation with order $J_i \to J_w$ and at least one permutation with order $J_w \to J_i$ have to be included in any solution $S^\pi(T)$ to the problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$. In such a case, vertices J_i and J_w have to be non-adjacent in the above digraph $G = (J, A)$. The following theorem gives the possibility to construct digraph G which represent a solution to uncertain flow shop problem in the condense form.

Theorem 4.6 *The order of two jobs $J_i \in J$ and $J_w \in J$ can be fixed in a solution $S^\pi(T)$ to the problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$ if and only if at least one condition (j), (jj) or (jjj) holds.*

PROOF. *Sufficiency* of each condition *(j)*, *(jj)* and *(jjj)* follows immediately from Theorems 4.3, 4.4 and 4.5.

Necessity. For any job $J_j \in J$, closed intervals $[a_{j1},\ b_{j1}]$ and $[a_{j2}, b_{j2}]$ may have either no more than one common point (and there are two such cases):

$$b_{j1} \le a_{j2}, \tag{4.15}$$

$$b_{j2} \le a_{j1} \tag{4.16}$$

or more than one common point:

$$\max\{a_{j1}, a_{j2}\} < \min\{b_{j1}, b_{j2}\}. \tag{4.17}$$

Hence for any pair of jobs $J_i \in J$ and $J_w \in J$, there exist nine combinations of dispositions of their processing time intervals on machine M_1 and machine M_2. Taking into account *symmetry* of jobs J_i and J_w, it is sufficient to check the following six combinations of inequalities (4.15), (4.16) and (4.17) for jobs J_i and J_w: (4.15)-(4.15), (4.15)-(4.16), (4.15)-(4.17), (4.16)-(4.16), (4.16)-(4.17) and (4.17)-(4.17) where the first inequality (in each pair) is indicated for job $J_j = J_i$, and the second inequality for job $J_j = J_w$. Next, we consider these six cases in details.

Case (4.17)-(4.17):

$$\max\{a_{i1}, a_{i2}\} < \min\{b_{i1}, b_{i2}\}, \tag{4.18}$$

$$\max\{a_{w1}, a_{w2}\} < \min\{b_{w1}, b_{w2}\}. \tag{4.19}$$

In this case, we can show that vertices J_i and J_w are non-adjacent in digraph G. To this end, it is sufficient to find two vectors $p' \in T$ and $p'' \in T$ such that in all Johnson's permutations constructed with vector p' of job processing times, the order of these jobs is $J_i \to J_w$, while in those constructed with vector p'' the order is $J_w \to J_i$.

Let us construct vector $p' = (p'_{1,1}, p'_{1,2}, \ldots, p'_{n1}, p'_{n2})$ that both sets of the following inequalities hold:

$$\max\{a_{i1}, a_{i2}\} \le p'_{i1} < p'_{i2} \le \min\{b_{i1}, b_{i2}\}, \tag{4.20}$$

$$\max\{a_{w1}, a_{w2}\} \le p'_{w2} < p'_{w1} \le \min\{b_{w1}, b_{w2}\}. \tag{4.21}$$

Obviously, inequalities (4.18) and (4.19) implies that there exist components p'_{i1}, p'_{i2}, p'_{w1}, p'_{w2} of the vector $p' \in T$ that inequalities (4.20) and (4.21) hold. Thus, in any Johnson's

permutation constructed with vector p' jobs J_i and J_w have to be arranged in the order $J_i \to J_w$ (since Johnson's rule implies inclusions $J_i \in N_1$ and $J_w \in N_2$). Due to strict inequalities in (4.20) and (4.21) such order of two jobs has to be the same in all Johnson's permutations constructed with vector p' of job processing times.

One can similarly show that there exist vector $p'' = (p''_{1,1}, p''_{1,2}, \ldots, p''_{n2}, p''_{n2}) \in T$ that sets of inequalities

$$\max\{a_{i1}, a_{i2}\} \le p''_{i2} < p''_{i1} \le \min\{b_{i1}, b_{i2}\},$$

$$\max\{a_{w1}, a_{w2}\} \le p''_{w1} < p''_{w2} \le \min\{b_{w1}, b_{w2}\}$$

hold. And in all Johnson's permutations constructed for the vector p'' jobs J_i and J_w have to be arranged in the opposite order $J_w \to J_i$.

Thus, the order of jobs J_i and J_w in a Johnson's permutation is determined by a vector of job processing times chosen from set T, and cannot be fixed in any solution $S^\pi(T)$. Moreover, it is easy to check that no condition *(j)*, *(jj)* or *(jjj)* holds in case (4.17)-(4.17).

Case (4.15)-(4.15):

$$b_{i1} \le a_{i2}, \tag{4.22}$$

$$b_{w1} \le a_{w2}. \tag{4.23}$$

In this case, both jobs J_i and J_w have to belong to set N_1. We study when it is possible to fix their order in Johnson's permutations constructed for a vector $p \in T$ of job processing times. Obviously, it is sufficient to compare processing time intervals of jobs J_i and J_w on machine M_1 only. Next, we check three possible variants.

1) If inequality $b_{i1} \le a_{w1}$ holds, then it together with inequality (4.22) define condition *(jj)*. And due to Theorem 4.4, for any vector $p \in T$ there exists Johnson's permutation with order $J_i \to J_w$.

2) If inequality $b_{w1} \le a_{i1}$ holds, then it together with inequality (4.23) define condition *(jj)* (provided that alternative notations of job J_i and job J_w being used). Due to Theorem 4.4, for any vector $p \in T$ there exists Johnson's permutation with order $J_w \to J_i$.

Thus, in both cases 1) and 2) order of jobs J_i and J_w can be fixed due to condition *(jj)*.

3) Let inequality

$$\max\{a_{i1}, a_{w1}\} < \min\{b_{i1}, b_{w1}\} \tag{4.24}$$

hold. Due to (4.24), we can construct feasible vectors p' and p'' that the following inequalities hold:

$$\max\{a_{i1}, a_{w1}\} \le p'_{i1} < p'_{w1} \le \min\{b_{i1}, b_{w1}\},$$

$$\max\{a_{i1}, a_{w1}\} \le p''_{w1} < p''_{i1} \le \min\{b_{i1}, b_{w1}\}.$$

It is clear that in all Johnson's permutations constructed for vector p' (vector p'') jobs J_i and J_w have to be arranged in the order $J_i \to J_w$ ($J_w \to J_i$, respectively).

Thus, if inequalities (4.22), (4.23) and (4.24) hold in common, then order of jobs J_i and J_w in a Johnson's permutations depends on vector $p \in T$ of job processing times and cannot be fixed in any solution $S^\pi(T)$. It is easy to check that no condition *(j)*, *(jj)* or *(jjj)* holds in this case.

Case (4.16)-(4.16):

$$b_{i2} \le a_{i1}, \tag{4.25}$$

$$b_{w2} \leq a_{w1} \tag{4.26}$$

can be tested similarly as case (4.15)-(4.15), and conclusion would be as follows. If $b_{w2} \leq a_{i2}$, then in all Johnson's permutations jobs J_i and J_w have to be arranged in the order $J_i \rightarrow J_w$ (since inequalities $b_{w2} \leq a_{i2}$ and (4.26) give condition *(jjj)*). If $b_{i2} < a_{w2}$, then in all Johnson's permutations jobs J_i and J_w have to be arranged in the order $J_w \rightarrow J_i$ (since inequalities $b_{i2} < a_{w2}$ and (4.25) define condition *(jjj)* with alternative notations of jobs J_i and J_w). If inequality $\max\{a_{i2}, a_{w2}\} < \min\{b_{i2}, b_{w2}\}$ holds, jobs J_i and J_w are non-adjacent in digraph G. And in the latter case, no condition *(j)*, *(jj)* or *(jjj)* holds.

Case (4.15)-(4.16):

$$b_{i1} \leq a_{i2}, \tag{4.27}$$

$$b_{w2} \leq a_{w1}. \tag{4.28}$$

Inequalities (4.27) and (4.28) give condition *(j)* from Theorem 4.3. Therefore, for all vectors $p \in T$ there exists Johnson's permutation with jobs J_i and J_w disposed in the order $J_i \rightarrow J_w$.

Case (4.15)-(4.17):

$$b_{i1} \leq a_{i2}, \tag{4.29}$$

$$\max\{a_{w1}, a_{w2}\} < \min\{b_{w1}, b_{w2}\}. \tag{4.30}$$

Since job J_i belongs to set N_1, we need to consider only dispositions of the intervals of job processing times on machine M_1. Let us consider two possible cases. Let inequality $b_{i1} \leq a_{w1}$ hold. The latter inequality together with inequality (4.29) give condition *(jj)*. Therefore, due to Theorem 4.4 for any vector $p \in T$ there exists Johnson's permutation with job order $J_i \rightarrow J_w$.

Let now inequality

$$b_{i1} > a_{w1} \tag{4.31}$$

hold. We construct two vectors $p' \in T$ and $p'' \in T$ that for the vector p' inequalities $a_{w1} \leq p'_{w1} < p'_{i1} \leq b_{i1}$ and $\max\{a_{w1}, a_{w2}\} \leq p'_{w1} < p'_{w2} \leq \min\{b_{w1}, b_{w2}\}$ hold, and for vector p'' inequalities $\max\{a_{w1}, a_{w2}\} \leq p''_{w2} < p''_{w1} \leq \min\{b_{w1}, b_{w2}\}$ hold (obviously, one can find such values of the vector components due to inequalities (4.30) and (4.31)). It is easy to see, that in all Johnson's permutations for the vector p', the order of two jobs is $J_w \rightarrow J_i$, and in all Johnson's permutations for the vector p'', the order of two jobs is $J_i \rightarrow J_w$. Therefore, in the case (4.15)-(4.17) with inequality (4.31), jobs J_i and J_w are non-adjacent in digraph G. In the latter case, no condition *(j)*, *(jj)* or *(jjj)* holds.

Case (4.16)-(4.17):

$$b_{i2} \leq a_{i1},$$

$$\max\{a_{w1}, a_{w2}\} < \min\{b_{w1}, b_{w2}\}.$$

For this case, it is sufficient to repeat arguments used for the case (4.15)-(4.17) provided that Theorem 4.4 is replaced by Theorem 4.5 and condition *(jj)* is replaced by condition *(jjj)*. As a result, we obtain the following statements. If $b_{i2} \leq a_{w2}$, then for any vector $p \in T$ there exists Johnson's permutation with order $J_w \rightarrow J_i$. If $a_{w2} < b_{i2}$, then vertices J_i and J_w are non-adjacent in digraph G.

It is proven that in any possible case, we can either fix the order of jobs J_i and J_w in a solution $S^{\pi}(T)$ to problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$, if at least one condition *(j)*, *(jj)* or *(jjj)*

holds, or vertices J_i and J_w are non-adjacent in digraph G (if neither condition *(j)*, *(jj)* nor *(jjj)* holds).

$\diamond$

Obviously, using Theorem 4.6 one can construct digraph G in $O(n^2)$ time, and this digraph represents solution $S^\pi(T)$ in condense form.

Dominant Schedule

In practice, in spite of uncertainty of job processing times it is necessary to choose only one schedule for realization. That is why, if no additional agreements or calculations will be used, then only Theorem 4.1 from the above results will allow to obtain (without fail) a permutation π_i which will dominate all the other permutations in set S^π with respect to T. And such a schedule (if any) is the best for any possible realization of the process under consideration (provided that conditions *a)*, *b)* and *c)* of Theorem 4.1 hold in common).

In the rest of this section, we find another sufficient conditions for existence a single *dominant* permutation π_i (see Definition 4.2). To this end, we will use agreement to relax the above demand for the desired permutation π_i to be a Johnson's one. In other words, we will use weaker Definition 4.2 instead of Definition 4.1 for solving problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$. Moreover, along with off-line scheduling problem, we will consider on-line scheduling problem when initial part of the schedule is already realized and as a result uncertain (in the original scheduling problem) processing times of completed jobs have received the concrete fixed values. For simplicity, we will consider mainly uncertain flow shop problem with only two elements in solution $S^\pi(T)$ (some extensions to the case $|S^\pi(T)| > 2$ are given at the end of this section).

Let problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{max}$ have solution $S^\pi(T)$ consisting of two $S^\pi(T) = \{\pi_u, \pi_v\}$. It is clear that there exist only two non-adjacent vertices in the digraph $G = (J, A)$ constructed for set $S^\pi(T) = \{\pi_u, \pi_v\}$. Due to Definition 4.1 permutation π_u (permutation π_v) is optimal Johnson's permutation for at least one feasible vector of job processing times but surely it is not Johnson's permutation for all feasible vectors $p \in T$.

When will one of these two permutations remain optimal for all feasible vectors $p \in T$ of job processing times provided that we will relax the demand for such a permutation to be Johnson's one? To answer this question we have to consider possible orders of non-adjacent vertices in the digraph G.

Let only vertices J_k and J_{k+1} be non-adjacent in the digraph G. We will say that such jobs J_k and J_{k+1} are in conflict. Without loss of generality, we can assume that $\pi_u = (J_1, J_2, \ldots, J_{k-1}, J_k, J_{k+1}, \ldots, J_n)$ and $\pi_v = (J_1, J_2, \ldots, J_{k-1}, J_{k+1}, J_k, \ldots, J_n)$ with $1 < k < n$. Since there are two permutations in the solution $S^\pi(T)$, conditions of Theorem 4.1 do not hold in common. Moreover, it is clear that violation of condition *a)*, *b)* or *c)* is possible only for these two jobs J_k and J_{k+1}. Using symmetry of two jobs we have to test the following five possible cases (sets N_{M_1}, N_{M_2} and N^* are defined in the first part of this section):

1) $J_k \in N^*$ and $J_{k+1} \in N^*$;

2) $J_k \in N_{M_1}$, $J_{k+1} \in N^*$ and processing time intervals of jobs J_k and J_{k+1} on machine M_1 have more than one common point;

3) $J_k \in N_{M_2}$, $J_{k+1} \in N^*$ and processing time intervals of jobs J_k and J_{k+1} on machine M_2 have more than one common point;

4) $J_k \in N_{M_1}$, $J_{k+1} \in N_{M_1}$ and processing time intervals of jobs J_k and J_{k+1} on machine M_1 have more than one common point

5) $J_k \in N_{M_2}$, $J_{k+1} \in N_{M_2}$ and processing time intervals of jobs J_k and J_{k+1} on machine M_2 have more than one common point.

In case 1) condition *a)* is violated, in cases 2) and 3) condition *c)* is violated, and in cases 4) and 5) condition *b)* is violated. Next, we consider on-line scheduling problem arising often in realization of a schedule in practice.

On-Line Scheduling Problem

Let $c_1(i)$ and $c_2(i)$, respectively, denote completion time of job $J_i \in J$ by machine M_1 and M_2 in the concrete realization of the process.

Let us introduce *on-line* version of the uncertain scheduling problem for the case $S^\pi(T) = \{\pi_u, \pi_v\}$. Let the processing of jobs $J_1, J_2, \ldots, J_{k-1}$ be started (and completed) in this order which is the same in both permutations π_u and π_v. Since jobs J_k and J_{k+1} are in conflict at time $t = c_1(k-1)$, additional scheduling decision has to be used (in on-line mode).

Namely, at time t a scheduler has to choose either job J_k or job J_{k+1} (which only are in conflict) to be processed next (immediately after job J_{k-1}). We can assume that at time $t = c_1(k-1)$ the scheduler already knows the exact values $c_1(k-1)$ and $c_2(k-1)$. For sound decision, the scheduler can test conditions of the following Lemmas 4.1 - 4.3. (We omit the proofs of these lemmas since they are trivial.)

Lemma 4.1 *If* $c_2(k-1) - c_1(k-1) > b_{k1} + b_{k+1,1}$, *then each permutation from set* $\{\pi_u, \pi_v\}$ *is dominant.*

If condition of Lemma 4.1 holds, then the order of jobs J_k and J_{k+1} may be arbitrary in the remaining part of the optimal schedule. Therefore, for this on-line scheduling problem, both permutations π_u and π_v are optimal for the remaining part of the vector $p \in T$ of job processing times. It should be noted that the initial part of each feasible vector of job processing times: $(p_{1,1}, p_{1,2}, \ldots, p_{k-1,1}, p_{k-1,2})$ is already fixed in this on-line version of scheduling problem.

Let condition of Lemma 4.1 do not hold, then scheduler may fix the order of jobs J_k and J_{k+1} if condition of one of the following two lemmas hold.

Lemma 4.2 *If* $c_2(k-1) - c_1(k-1) > b_{k1}$ *and* $c_2(k-1) - c_1(k-1) + a_{k2} > b_{k1} + b_{k+1,1}$, *then permutation* π_u *is dominant.*

Thus, if condition of Lemma 4.2 holds, then the order of jobs J_k and J_{k+1} has to be $J_k \to J_{k+1}$ in the remaining part of the optimal permutation.

Lemma 4.3 *If* $c_2(k-1) - c_1(k-1) > b_{k+1,1}$ *and* $c_2(k-1) - c_1(k-1) + a_{k+1,2} > b_{k1} + b_{k+1,1}$, *then permutation* π_v *is dominant.*

If condition of Lemma 4.3 holds, then the order of jobs J_k and J_{k+1} has to be $J_{k+1} \rightarrow J_k$ in the remaining part of the optimal permutation.

Thus, if condition of one of the Lemmas 4.1 - 4.3 holds, then permutation π_u or (and) π_v is dominant in the remaining part of the process. Scheduler can test all these sufficient conditions for ordering jobs J_k and J_{k+1} in all the above five cases 1) - 5).

Off-Line Scheduling Problem

Claims similar to Lemmas 4.1 - 4.3 may be used for solving off-line scheduling problem as well, if exact difference $c_2(k-1) - c_1(k-1)$ (which is unknown at time $t = 0$) will be substituted by its lower bound.

Indeed, jobs J_k and J_{k+1} can be started by machine M_1 at time $c_1(k-1)$, and machine M_2 is available to process one of the jobs J_k and J_{k+1} from time $c_2(k-1)$. If we can calculate lower bound Δ_{k-1} of the difference $c_2(k-1) - c_1(k-1)$, then before realization of a schedule (i.e., before time $t = 0$) we can test conditions of the following lemmas.

Lemma 4.4 *If $\Delta_{k-1} > b_{k1} + b_{k+1,1}$, then each permutation from set $\{\pi_u, \pi_v\}$ is dominant.*

Lemma 4.5 *If $\Delta_{k-1} > b_{k1}$ and $\Delta_{k-1} + a_{k2} > b_{k1} + b_{k+1,1}$, then permutation π_u is dominant.*

Lemma 4.6 *If $\Delta_{k-1} > b_{k+1,1}$ and $\Delta_{k-1} + a_{k+1,2} > b_{k1} + b_{k+1,1}$, then permutation π_v is dominant.*

Thus, if condition of one of the Lemmas 4.4 - 4.6 holds, then there exists at least one permutation π_u or π_v which is dominant for the problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$. Next, we show how to calculate tight lower bounds Δ_{k-1} for the above cases 1) - 5).

For each case 1) - 4), inclusion $J_i \in N_{M_1}$ certainly holds for indices $i = 1, 2, \ldots, k-1$. Moreover, for each index $i \in \{1, 2, \ldots, k-1\}$ inequality $b_{i1} \leq a_{i2}$ has to hold, and so $p_{i2} - p_{i1} \geq a_{i2} - b_{i1} \geq 0$. Thus, the following inequalities give tight lower bound Δ_{k-1}:

$$c_2(k-1) - c_1(k-1) \geq \sum_{i=1}^{k-1} (p_{i2} - p_{i1}) + p_{1,1}$$

$$\geq \sum_{i=1}^{k-1} (a_{i2} - b_{i1}) + a_{1,1} = \Delta_{k-1}.$$

For case 5), lower bound Δ_{k-1} of difference $c_2(k-1) - c_1(k-1)$ may be calculated recurrently as follows. If $|N_{M_1}| = m$, then we obtain

$$\Delta_m = \sum_{i=1}^{m} (a_{i2} - b_{i1}) + a_{1,1}$$

(similarly to cases 1) - 4)). Further, for each index $l \in \{m+1, m+1, \ldots, k-1)$, one can use recurrent formula

$$\Delta_l = \max\{a_{l,2},\ \ c_2(l-1) - c_1(l-1) + a_{l,2} - b_{l,1}\}.$$

Note that Lemmas 4.1 - 4.6 may be used if $k > 1$. Let $k = 1$, i.e., jobs J_1 and J_2 are in conflict. We have to order these jobs at time $t = 0$ (without additional information).

Let us consider the case when machine M_2 has an idle time before processing job J_3. In this case, machine M_2 can process job J_3 from time when machine M_1 finished processing this job (i.e., at time $t = c_1(3)$). It is easy to prove the following sufficient conditions.

Lemma 4.7 *If $a_{3,1} > b_{2,2} + \max\{0,\ b_{1,2} - a_{2,1}\}$, then order of jobs J_1 and J_2 in the optimal permutation is $J_1 \rightarrow J_2$ (permutation π_u is dominant).*

Lemma 4.8 *If $a_{3,1} > b_{1,2} + \max\{0,\ b_{2,2} - a_{1,1}\}$, then order of jobs J_1 and J_2 in the optimal permutation is $J_2 \rightarrow J_1$ (permutation π_v is dominant).*

Obviously, $c_2(2) - c_1(2) \leq b_{2,2} + \max\{0,\ b_{1,2} - p_{2,1}\}$ and $c_2(2) - c_1(2) \leq b_{1,2} + \max\{0,\ b_{2,2} - a_{1,1}\}$. In the above conditions, difference $b_{1,2} - a_{2,1}$ ($b_{2,2} - a_{1,1}$, respectively) is equal to maximal addition in the case when machine M_2 cannot finish job J_1 (job J_2) before machine M_1 has finished job J_2 (job J_1). Hence, we obtain that $c_1(3) > c_2(2)$ ($c_1(3) > c_2(1)$), and machine M_2 has an idle time before processing job J_3. Therefore, in the opposite case (when the optimal order of jobs J_1 and J_2 cannot be defined by Lemmas 4.7 and 4.8), we cannot decrease value C_{max}. Of course, if $a_{3,1} > b_{2,2} + b_{1,2}$, then both permutations π_u and π_v are dominant and order of jobs J_1 and J_2 may be arbitrary. More precisely, if $a_{3,1} > \max\{b_{1,2} + b_{2,2} - \min\{a_{1,1}, a_{1,2}\},\ \max\{b_{1,2}, b_{2,2}\}\}$, then both permutations π_u and π_v are dominant. In other words, if $a_{3,1} > \max\{0,\ b_{1,2} - a_{2,1},\ b_{2,2} - a_{1,1}\} + \max\{b_{1,2},\ b_{2,2}\}$, then both permutations π_u and π_v are dominant (order of jobs J_1 and J_2 is arbitrary).

It is easy to see that all the above lemmas can be generalized and used in the case $|S^\pi(T)| > 2$ as well. Next, we demonstrate such generalizations on some examples.

Lemmas 4.1 - 4.6 may be used if more than two permutations are in the set $S^\pi(T)$ provided that at each time point no more than two jobs from set J are in conflict. For example, if $|S^\pi(T)| = 8$ and six jobs are in conflict in pairwise manner (e.g., pair of jobs J_k and J_{k+1} are in conflict, pair of jobs J_l and J_{l+1}, and pair of jobs J_m and J_{m+1}). Then we can use the above lemmas for each pair of conflict jobs. More precisely, we can use Lemmas 4.1 - 4.3 in the case of on-line scheduling problem, and Lemmas 4.4 - 4.6 in the case of off-line scheduling problem. Moreover, for the off-line problem, we can test pair of conflict jobs in arbitrary order. And if condition of at least one of Lemmas 4.4 - 4.6 holds for each pairs of jobs, then we have dominant permutation, and $|S^\pi(T)| = 1$. Otherwise, e.g., if no lemma's condition holds for a pair of jobs J_l, J_{l+1}, then we obtain $|S^\pi(T)| = 2$ with $S^\pi(T) = \{\pi_u, \pi_v\}$, where $\pi_u = (J_1,\ J_2, \ldots,\ J_{l-1},\ J_l, J_{l+1}, \ldots, J_n)$ and $\pi_v = (J_1,\ J_2, \ldots,\ J_{l-1},\ J_{l+1},\ J_l, \ldots, J_n)$.

It is worth noting that Lemmas 4.1 - 4.6 may be generalized for the case when more than two jobs are in conflict at time $t > 0$. Let set of jobs $J^0 = \{J_k,\ J_{k+1}, \ldots,\ J_{k+r}\} \subset J$ be in conflict at time $t \geq 0$. Then we can test conditions similar to Lemmas 4.4 - 4.6 and reduce set of permutations which dominate simultaneously set of permutations S^π with respect to T. example, if $|J^0| = 3$ and cardinality of solution $S^\pi(T)$ is equal to $3! = 6$, then we can test the following sufficient sufficient condition.

Lemma 4.9 *If $c_2(k-1) - c_1(k-1) > b_{k1} + b_{k+1,1} + b_{k+2,1}$, then each permutation from set $S^\pi(T)$ is dominant.*

If condition of Lemma 4.9 holds, then in the remaining part of the optimal schedule the order of three jobs J_k, J_{k+1} and J_{k+2} may be arbitrary. Obviously, all the other above

lemmas can be generalized analogously, and we can obtain conditions for the optimal ordering three jobs. For example, we can prove the following claim.

Lemma 4.10 *If* $c_2(k-1) - c_1(k-1) > b_{k1}$, $c_2(k-1) - c_1(k-1) + a_{k2} > b_{k1} + b_{k+1,1}$ *and* $c_2(k-1) - c_1(k-1) + a_{k2} + a_{k+1,2} > b_{k1} + b_{k+1,1} + b_{k+2,1}$, *then permutation* $\{J_1, \ldots, J_k, J_{k+1}, J_{k+2}, \ldots, J_n\}$ *is dominant.*

If condition of Lemma 4.10 holds, then in the remaining part of the optimal permutation the order of jobs J_k and J_{k+1} and J_{k+2} has to be as follows: $J_k \rightarrow J_{k+1} \rightarrow J_{k+2}$. And we can test 6 Lemmas with analogous conditions, but for other orders of conflict jobs.

How to Use Solution $S^{\pi}(T)$ in Supply Chain Scheduling?

Scheduling in supply chain has often to be realized in condition of high uncertainty of numerical input date. We presented two-stage supply chain scheduling problem. In practice, such a problem may be considered for period from one week to three months, i.e., after the problem of *aggregate planning* being already solved (see [CM01])). Within such a planning horizon, the exact values of job processing times are usually unknown, but one can estimate their reliable lower and upper bounds. Moreover, we can reserve some positions for uncertain demands of customers by introducing jobs J_i in set J with zero lower bounds of processing times on both machines: $a_{i1} = a_{i2} = 0$.

The above results may be used as mathematical background for multi-stage scheduling process in some real supply chain. At first, when level of uncertainty of the input date is high, scheduler can solve off-line scheduling problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$. Obviously, if condition of Theorem 4.1 holds, then scheduler can use dominant schedule which is optimal (for off-line scheduling problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$) in spite of uncertainty of numerical date. Otherwise, it is necessary to construct digraph $G = (J, A)$ using Theorem 4.6. If $|S(T)| = 2$ and at least one sufficient condition of Lemmas 4.4 - 4.8 holds, then scheduler can use dominant schedule.

Let dominant schedule for off-line problem be unknown. Digraph G represents solution $S^{\pi}(T)$ to the original scheduling problem and may be considered as generalization of a schedule for the set of on-line scheduling problems which may be generated from the original off-line scheduling problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$. Scheduler has to look for the optimal schedule for on-line scheduling problems which will arise when the supply chain under consideration will be in process. In other words, it may be possible to arrange the jobs which are in conflict in on-line mode later when probably additional information about job processing (completion) times will become available for scheduler. Next, we describe this process in more details.

Similarly to the case of $S^{\pi}(T) = \{\pi_u, \pi_v\}$, the scheduler can arrange set of jobs $J^s = \{J_1, J_2, \ldots, J_{k-1}\}$ in the optimal order $\pi^0 = (J_1, J_2, \ldots, J_{k-1})$ till the first set of conflict jobs $J^0 = \{J_k, J_{k+1}, \ldots, J_{k+r}\}$ will be met. Then at time point $t > 0$, a part of job processing times along with their completion times will be already known. At least, $2(k-1)$-dimensional vector $p^s = (p_{1,1}, p_{1,2}, \ldots, p_{k-1,1}, p_{k-1,2})$ of job processing times will be known before time point t. Then scheduler can test conditions of Lemmas 4.1 - 4.3 for each pair of jobs being in conflict (or their generalizations for case $|S^{\pi}(T)| > 2$). If condition of one of these lemmas holds, then it is possible to select next job for processing

in the optimal way. Otherwise, we need additional agreement for selecting job from set J^0 to be processed next.

In general case, the choice of the job may be based, e.g., on minimization of possible lost of the objective function value. More precisely, for each conflict job $J_k \in J^0$ one can calculate maximal possible difference

$$C_{max}((\pi^s, J_k, J_d, \pi^f), (p^s, p^f))$$

$$- \min_{J_d \in J^0} C_{max}((\pi^s, J_d, J_k, \pi^f), (p^s, p^f)) \tag{4.32}$$

where $(p^s, p^f) \in T$, $J_d \in J^0 \backslash \{J_k\}$ and π^f is any permutation of jobs from set $J \backslash J^s \backslash \{J_k, J_d\}$ which does not contradict to the partial order defined by digraph G. The conflict job which has the minimal value of difference (4.32) will be processed next since such a choice will guarantee the minimal lost of objective function value.

Of course, some other agreements may be used to arrange jobs of the set J^0, if no condition of the above lemmas holds. It is also clear, that the higher uncertainty of the input date will be, the less possibility for right decision scheduler will have. So, there is no sense to use the above approach if the cardinality of set $S^\pi(T)$ is very large.

For example, if condition of Theorem 4.2 holds, then $|S^\pi(T)| = n!$ and digraph G has empty set of arcs: $G = (J, \emptyset)$. For such an off-line scheduling problem, the approach proposed in this paper will be useless, and it will be necessary to look for other more suitable models and methods. It may be more fruitful to look for optimal scheduling policy stochastically minimizing makespan: $E\mathcal{C}_{max}$ (see [ET00, Kam99]). To this end, it is necessary to obtain reliable probability distribution for random job processing times.

In this section, we focus on constructing single schedule which dominates all the other feasible ones (see Theorem 4.1, Lemmas 4.4 - 4.8 for off-line scheduling problem, and Lemmas 4.1 - 4.3 for on-line scheduling problem).

If there is no possibility to construct dominant schedule $\pi_i \in S^\pi$, we propose to construct digraph $G = (J, A)$ (on the basis of constructive sufficiency proof of Theorem 4.6) and use it as a more general schedule form. The partial order defined by digraph G may be extended later in on-line mode when more information about job processing times will be available. So, scheduling decisions have to be continued in on-line mode and have to be involved in supply chain process. If partial order defined by digraph G will be extended to complete order of n jobs (finally, to permutation $\pi_i \in S^\pi$), then the process will be realized in the optimal way. Otherwise, using solution $S^\pi(T)$ one can look for the schedule minimizing the lost of the objective function value.

Several directions for future research may be foreseen.

- To consider problem with several supplier's machines or (and) manufacturer's machines.
- To generalize further Lemmas 4.1 - 4.6, e.g., for several (more than two) jobs being in conflict.
- To consider another objective functions for uncertain scheduling problem, e.g., total completion time $\sum \mathcal{C}_i$ or criteria with job due dates.
- To consider multi-stage supply chain scheduling including not only supplier's machine M_1 and manufacturer's machine M_2, but also distributor's machine M_3, retailer's machine M_4 and customer's machine M_5.

▪ Along with uncertain processing times it is useful to consider uncertain setup times, job release times, due dates, etc.

Note that the whole conditions proven in this paper may be tested in polynomial time. However it is clear, that for the most generalizations of the above scheduling problem existence of polynomial algorithms is unlikely (e.g., if deterministic scheduling counterpart is NP-hard). However, we can remark with respect to algorithm complexity that for practical supply chain planning with horizon between one week and three months algorithm speed may be not very important because of sufficient time for decision making and because of restricted number n of jobs. (But of course, for solving on-line scheduling problem one need fast algorithms.) So, for future more general and so more practical investigation it will be reasonable to look for methods of branch and bound type for solving off-line scheduling problem and it will be also useful to consider heuristic solution instead of optimal one both for off-line and on-line scheduling problems.

4.2 Job Shop with Uncertain Processing Times

In this section, we address the stochastic job shop problem for the case when it is hard to obtain exact probability distributions for random and bounded processing times, and when assuming a specific probability distribution is not realistic. Commonly, schedules obtained after assuming a certain probability distribution may be not close to the optimal schedule. It has been observed that, although the exact probability distribution of job processing times may not be known in advance, upper and lower bounds on job processing times are easy to obtain in many practical cases. We show that information on the bounds of job processing times is important and should be utilized in finding a solution to the shop-scheduling problem with uncertain numerical data.

In this section, the problem of job shop scheduling with random and bounded processing times is studied. It is assumed that processing time of job $J_i \in J$ on machine $M_k \in M$ is not fixed before scheduling, and probability distribution of random processing time between given lower and upper bounds is unknown. In such uncertain version of shop-scheduling problem there may not exist a unique schedule that remains optimal for all possible realizations of job processing times, and therefore, we consider a set of schedules (permutations) which dominates all other feasible schedules (permutations) for the makespan criterion.

In Section 4.1, necessary and sufficient conditions are given when transposition of two jobs may be used to minimize makespan for the uncertain version of a flow shop problem with two machines. In this section, we show how to use the results obtained in Section 4.1 for solving two-machine job shop problem with uncertain job processing times.

Notations and Definition

We consider the following non-preemptive job shop scheduling problem with random and bounded processing times. Two machines $M = \{M_1, M_2\}$ have to process n jobs J with different two-stage routes. Machine repetition in a route is not allowed. Let $J(12) \subseteq J$ denote subset of jobs with route (M_1, M_2): First, job J_i from set $J(12)$ has to be processed by machine M_1, and then by machine M_2. Let $J(21) \subseteq J$ denote subset of jobs with route

(M_2, M_1), and $J(k) \subseteq J$ denote subset of jobs that have to be processed only by machine $M_k, k \in \{1, 2\}$.

Thus, we have equality $J = J(1) \cup J(2) \cup J(12) \cup J(21)$. We denote $q_k = |J(k)|$, where $k \in \{1, 2, 12, 21\}$. In contrast to deterministic job shop problem, it is assumed that processing time p_{ij} of job J_i on machine M_j is not fixed before scheduling. In the realization of the process, p_{ij} may take any real value between lower bound a_{ij} and upper bound b_{ij} being given before scheduling. Moreover, the probability distribution of random processing time is unknown. In such a case there may not exist a unique schedule that remains optimal for all possible realizations of job processing times. Therefore, we consider a set of schedules that dominate all feasible schedules. Similarly as in Section 4.1, we consider criterion $\mathcal{C}_{\max}$, i.e., minimization of schedule length.

Let $T^J \subseteq R^q_+$ define the given set of feasible vectors of job processing times. For the case of uncertain job shop problem it is useful to consider T^J as Cartesian product of four sets:

$$T^J = T_{12} \times T_1 \times T_2 \times T_{21}$$

where T_w is set of feasible vectors defining processing times of jobs from set $J(k)$, $k \in \{1, 2, 12, 21\}$. In what follows, inclusion $p \in T^J$ means that vector p has dimension $q = q_{12} + q_1 + q_2 + q_{21}$ and set of its components consists of the following four ordered subsets $\{p_{ij} : J_i \in J(12), j \in \{1, 2\}\}$, $\{p_{ij} : J_i \in J(1), j = 1\}$, $\{p_{ij} : J_i \in J(2), j = 2\}$ and $\{p_{ij} : J_i \in J(21), j \in \{1, 2\}\}$. We assume that jobs from set J are numbered with respect to the above order of these four subsets of the components of vector p. Then vector $p \in T^J$ may be presented as follows:

$$p = (p_{1,1},\ p_{1,2}, \ldots, p_{q_{12},1},\ p_{q_{12},2};\ p_{q_{12}+1,1}, p_{q_{12}+2,1}, \ldots, p_{q_{12}+q_1,1};$$

$$p_{q_{12}+q_1+1,2}, p_{q_{12}+q_1+2,2}, \ldots, p_{q_{12}+q_1+q_2,2};\ p_{q_{12}+q_1+q_2+1,1},\ p_{q_{12}+q_1+q_2+1,2}, \ldots, p_{q,1},\ p_{q,2})$$

$$= (p(12);\ p(1);\ p(2);\ p(21)),$$

where $p(12) \in T_{12}$, $p(1) \in T_1$, $p(2) \in T_2$ and $p(21) \in T_{21}$. Using three-field notation, this uncertain job shop problem is denoted as $\mathcal{J}2/a_i \leq p_i \leq b_i, n_i \leq 2/\mathcal{C}_{\max}$.

If equality $a_{ij} = b_{ij}$ holds for each job $J_i \in J$ and machine $M_j \in M$, then problem $\mathcal{J}2/a_i \leq p_i \leq b_i, n_i \leq 2/\mathcal{C}_{\max}$ turns into a deterministic job shop problem $\mathcal{J}2/n_i \leq 2/\mathcal{C}_{\max}$ that is polynomially solvable due to the following theorem proven in [Jac56].

Theorem 4.7 *For the problem $\mathcal{J}2/n_i \leq 2/\mathcal{C}_{\max}$ semiactive schedule for processing jobs $J = J(1) \cup J(2) \cup J(12) \cup J(21)$ has minimal length, if machine M_1 processes jobs in the order $(J(12), J(1), J(21))$, machine M_2 in the order $(J(21), J(2), J(12))$ provided that jobs $J(12)$ are ordered for processing by both machines using condition*

$$\min\{p_{i_k 1}, p_{i_{k+1} 2}\} \leq \min\{p_{i_{k+1} 1}, p_{i_k 2}\},\ k = 1, 2, \ldots, q_{12} - 1, \tag{4.33}$$

and jobs $J(21)$ are ordered for processing by both machines using condition

$$\min\{p_{i_k 2}, p_{i_{k+1} 1}\} \leq \min\{p_{i_{k+1} 2}, p_{i_k 1}\},\ k = 1, 2, \ldots, q_{21} - 1, \tag{4.34}$$

jobs $J(1)$ and jobs $J(2)$ may be ordered for processing arbitrarily.

It is easy to see that due to Theorem 4.7, jobs $J(12)$ has to be ordered via Johnson's rule similarly as in a flow shop problem $\mathcal{F}2//\mathcal{C}_{\max}$ since condition (4.33) coincides with condition (4.1) (see Section 4.1, page 186) provided that $J = J(12)$ and $T = T_{12}$. Jobs $J(21)$ has to be ordered via Johnson's rule in a flow shop problem $\mathcal{F}2//\mathcal{C}_{\max}$ with opposite route (M_2, M_1) of each job since condition (4.34) may be obtained from condition (4.1) after interchanging machine M_1 with machine M_2 and setting $J = J(21)$ and $T = T_{21}$.

Thus, the optimal schedule for the problem $\mathcal{J}2/n_i \leq 2/\mathcal{C}_{\max}$ may be defined by the pair of permutations π' and π'' (we call it Jackson's pair of permutations), where $\pi' = (\pi_i^{(12)}, \pi_u^{(1)}, \pi_j^{(21)})$ is permutation of jobs on machine M_1, and $\pi'' = (\pi_j^{(21)}, \pi_v^{(2)}, \pi_i^{(12)})$ is permutation of jobs on machine M_1. Job J_r belongs to permutation $\pi_k^{(w)}$ if and only if $J_r \in J(w)$ with $w \in \{1, 2, 12, 21\}$.

To be more precise, permutation π' of jobs $J(12) \cup J(1) \cup J(21)$ on machine M_1, and permutation π'' of jobs $J(21) \cup J(2) \cup J(12)$ on machine M_2 may be defined as follows. The jobs within permutation $\pi_i^{(12)}$ (permutation $\pi_j^{(21)}$, respectively) have to be ordered by SPT rule (by LPT rule) on machine M_1 (on machine M_2). Rule SPT (shortest processing time) means that the job set have to be sorted in the non-decreasing order of job processing times. Rule LPT (longest processing time) means that the job set have to be sorted in the non-increasing order of job processing times. Since the order of the jobs in permutation $\pi_u^{(1)}$ and in permutation $\pi_v^{(2)}$ may be arbitrary, we can fix these permutations. Let jobs in permutation $\pi_u^{(1)}$ and in permutation $\pi_v^{(2)}$ be ordered with respect to the job numbers:

$$\pi_u^{(1)} = (J_{q_{12}+1}, J_{q_{12}+2}, \ldots, J_{q_{12}+q_1}),\ \pi_v^{(2)} = (J_{q_{12}+q_1+1}, J_{q_{12}+q_1+2}, \ldots, J_{q_{12}+q_1+q_2}).$$

We denote set of all permutations of jobs $J(12)$ as

$$S_{12}^{\pi} = \{\pi_1^{(12)}, \pi_2^{(12)}, \ldots, \pi_{q_{12}!}^{(12)}\},$$

and set of all permutations of jobs $J(21)$ as

$$S_{21}^{\pi} = \{\pi_1^{(21)}, \pi_2^{(21)}, \ldots, \pi_{q_{21}!}^{(21)}\}.$$

Let $\langle S_{12}^{\pi}, S_{21}^{\pi}\rangle$ be such a subset of Cartesian product $(S_{12}^{\pi}, \pi_u^{(1)}, S_{21}^{\pi}) \times (S_{21}^{\pi}, \pi_v^{(2)}, S_{12}^{\pi})$ that elements of set $\langle S_{12}^{\pi}, S_{21}^{\pi}\rangle$ are ordered pairs of two permutations π' and π'', where $\pi' = (\pi_i^{(12)}, \pi_u^{(1)}, \pi_j^{(21)})$ and $\pi'' = (\pi_j^{(21)}, \pi_v^{(2)}, \pi_i^{(12)})$, $1 \leq i \leq q_{12}!$, $1 \leq j \leq q_{21}!$. Since both permutations $\pi_u^{(1)}$ and $\pi_v^{(2)}$ are fixed, and index i (index j) is the same in each permutation from pair π', π'', then we obtain $|\langle S_{12}^{\pi}, S_{21}^{\pi}\rangle| = q_{12}!q_{21}!$ pairs of permutations. We use the following definition of a solution to the problem $\mathcal{J}2/a_i \leq p_i \leq b_i, n_i \leq 2/\mathcal{C}_{\max}$.

Definition 4.3 *Set of pairs of permutations $\langle S_{12}^{\pi}(T_{12}), S_{21}^{\pi}(T_{21})\rangle \subseteq \langle S_{12}^{\pi}, S_{21}^{\pi}\rangle$ is called a solution to the problem $\mathcal{J}2/a_i \leq p_i \leq b_i, n_i \leq 2/\mathcal{C}_{\max}$, if for each feasible vector $p \in T^J$ the set $\langle S_{12}^{\pi}(T_{12}), S_{21}^{\pi}(T_{21})\rangle$ contains at least one optimal Jackson's pair of permutations for the problem $\mathcal{J}2/n_i \leq 2/\mathcal{C}_{\max}$ with vector p of job processing times. If any proper subset of solution $\langle S_{12}^{\pi}(T_{12}), S_{21}^{\pi}(T_{21})\rangle$ is not a solution to the problem $\mathcal{J}2/a_i \leq p_i \leq b_i, n_i \leq 2/\mathcal{C}_{\max}$, then $\langle S_{12}^{\pi}(T_{12}), S_{21}^{\pi}(T_{21})\rangle$ is called minimal solution.*

Next, we show that the results for a flow shop problem obtained in Section 4.1 may be used for solving a job shop problem with uncertain processing times. In particular, we prove the following lemma.

Lemma 4.11 *Let $S^{\pi}_{12}(T_{12})$ be solution to the problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{\max}$ with set of jobs $J = J(12)$ and feasible processing times defined by set of vectors $T = T_{12}$, then $\langle S^{\pi}_{12}(T_{12}), S^{\pi}_{21}\rangle$ is a solution to the problem $\mathcal{J}2/n_i \le 2, a_i \le p_i \le b_i/\mathcal{C}_{\max}$ with set of jobs $J = J(1) \cup J(2) \cup J(12) \cup J(21)$ and feasible processing times defined by set of vectors $T^J = T_{12} \times T_1 \times T_2 \times T_{21}$.*

Proof. By contradiction method, we assume that there exists vector $p = (p(12);\ p(1);\ p(2);\ p(21)) \in T^J$ of job processing times, for which no optimal Jackson's pair of permutations for the problem $\mathcal{J}2/n_i \le 2/\mathcal{C}_{\max}$ with vector p of job processing times belongs to set $\langle S^{\pi}_{12}(T_{12}), S^{\pi}_{21}\rangle$.

Due to Theorem 4.7 and taking into account that set S^{π}_{21} contains all permutations of jobs $J(21)$, violation of optimality of permutation pair from set $\langle S^{\pi}_{12}(T_{12}), S^{\pi}_{21}\rangle$ may be only possible for the permutation of jobs $J(12)$ which will be not optimal Johnson's permutation for the corresponding problem $\mathcal{F}2//\mathcal{C}_{\max}$. To be more specific, set $S^{\pi}_{12}(T_{12})$ does not include optimal Johnson's permutation for the problem $\mathcal{F}2//\mathcal{C}_{\max}$ with job set $J = J(12)$ and vector $p(12) \in T = T_{12}$ of job processing times. Thus, due to Definition 4.1 set $S^{\pi}_{12}(T_{12})$ is not a solution to the problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{\max}$ with set of jobs $J = J(12)$ and set $T = T_{12}$ of feasible vectors of job processing times. We obtain a contradiction with condition of Lemma 4.11. Hence our assumption was wrong.

In fact, for each vector $p \in T^J$ set $\langle S^{\pi}_{12}(T_{12}), S^{\pi}_{21}\rangle$ contains optimal Jackson's pair of permutations for the problem $\mathcal{J}2/n_i \le 2/\mathcal{C}_{\max}$ with vector p of job processing times. Consequently, set $\langle S^{\pi}_{12}(T_{12}), S^{\pi}_{21}\rangle$ is a solution to the problem $\mathcal{J}2/n_i \le 2, a_i \le p_i \le b_i/\mathcal{C}_{\max}$ with set of jobs $J = \{J(12), J(1), J(2), J(21)\}$ and feasible processing times defined by set of vectors $T^J = T_{12} \times T_1 \times T_2 \times T_{21}$.

◇

Similarly we can prove the following claim.

Lemma 4.12 *Let $S^{\pi}_{21}(T_{21})$ be solution to the problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{\max}$ with set of jobs $J = J(21)$ and feasible processing times defined by set of vectors $T = T_{21}$, then $\langle S^{\pi}_{12}, S^{\pi}_{21}(T_{21})\rangle$ is a solution to the problem $\mathcal{J}2/n_i \le 2, a_i \le p_i \le b_i/\mathcal{C}_{\max}$ with set of jobs $J = J(1) \cup J(2) \cup J(12) \cup J(21)$ and feasible processing times defined by set of vectors $T^J = T_{12} \times T_1 \times T_2 \times T_{21}$.*

Using Lemma 4.11 and Lemma 4.12, we prove the following claim.

Theorem 4.8 *If $S^{\pi}_{12}(T_{12})$ is a solution to the flow shop problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{\max}$ with set of jobs $J(12)$ and feasible processing times defined by set of vectors $T = T_{12}$, and $S^{\pi}_{21}(T_{21})$ is a solution to the flow shop problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{\max}$ with set of jobs $J(21)$ and feasible processing times defined by set of vectors $T = T_{21}$, then $\langle S^{\pi}_{12}(T_{12}), S^{\pi}_{21}(T_{21})\rangle$ is a solution to the problem $\mathcal{J}2/n_i \le 2, a_i \le p_i \le b_i/\mathcal{C}_{\max}$ with set of jobs $J = \{J(12), J(1), J(2), J(21)\}$ and feasible processing times defined by set of vectors $T^J = T_{12} \times T_1 \times T_2 \times T_{21}$.*

Proof. By contradiction method, we assume that there exists vector $p = (p(12);\ p(1);\ p(2);\ p(21)) \in T^J$ of job processing times, for which no optimal Jackson's pair of permutations for the problem $\mathcal{J}2/n_i \le 2/\mathcal{C}_{\max}$ with vector p of processing times belongs to set $\langle S^{\pi}_{12}(T_{12}), S^{\pi}_{21}(T_{21})\rangle$.

Due to Theorem 4.7, violation of optimality of permutation pair from set $\langle S^{\pi}_{12}(T_{12}), S^{\pi}_{21}(T_{21})\rangle$ may be possible either for the permutation of jobs $J(12)$ (which will be not optimal Johnson's permutation for the problem $\mathcal{F}2//\mathcal{C}_{\max}$ with job set $J(12)$ and processing times $p(12)$) or for the permutation of jobs $J(21)$ (which will be not optimal Johnson's permutation for the problem $\mathcal{F}2//\mathcal{C}_{\max}$ with job set $J(21)$, technological route (M_2, M_1) and processing times $p(21)$). We call the former case as case a, and the latter case as case b.

In case a, set $S^{\pi}_{12}(T_{12})$ has no optimal Johnson's permutation for the problem $\mathcal{F}2//\mathcal{C}_{\max}$ with job set $J(12)$ and vector $p(12) \in T = T_{12}$ of job processing times. It implies that set $\langle S^{\pi}_{12}(T), S^{\pi}_{21}\rangle$ is not a solution to the problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{\max}$ with set of jobs $J = J(12)$ and set $T = T_{12}$ of feasible vectors of job processing times. However, it is not possible due to Lemma 4.11 (since condition of Lemma 4.11 holds). Thus, in case a we obtain a contradiction. Hence in case a, our assumption was wrong.

Using Lemma 4.12 and arguing similarly we can obtain a contradiction in case b as well.

$\diamond$

Thus, due to Theorem 4.8 solving problem $\mathcal{J}2/a_i \le p_i \le b_i, n_i \le 2/\mathcal{C}_{\max}$ is reduced to solving two problems $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{\max}$. To illustrate the above results we consider the following two examples of uncertain job shop problem.

Example 4.3 *Let the bounds of job processing times for problem $\mathcal{J}2/a_i \le p_i \le b_i, n_i \le 2/\mathcal{C}_{\max}$ be given in Table 4.3. These bounds define set T^J of feasible vectors of job processing times. Set of jobs J consists of four subsets $J(12) = \{J_1, J_2, J_3\}$, $J(1) = \{J_4\}$, $J(2) = \{J_5\}$ and $J(21) = \{J_6, J_7, J_8\}$.*

Table 4.3: *Lower and upper bounds of job processing times in Example 4.3*

i	a_{i1}	b_{i1}	a_{i2}	b_{i2}
1	2	4	5	6
2	3	4	5	6
3	7	8	7	8
4	2	3	-	-
5	-	-	3	5
6	4	5	1	3
7	5	6	2	3
8	7	8	7	8

It is easy to see that Theorem 4.7 used for the problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{\max}$ with job set $J(12)$ implies the following dominance relations between jobs $J(12) = \{J_1, J_2, J_3\}$. Job J_1 precedes job J_3 and job J_2 precedes job J_3. However, we cannot order jobs J_1 and J_2. Therefore, instead of considering $3! = 6$ permutations of the jobs $J(12)$, it is sufficient to consider two permutations, namely, permutations (J_1, J_2, J_3) and (J_2, J_1, J_3).

Next, we consider flow shop problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{\max}$ with job set $J(21) = \{J_6, J_7, J_8\}$ and technological route (M_2, M_1). Due to Theorem 4.7 we find the following dominance relations between jobs from set $J(21)$. Job J_6 precedes job J_8, and job J_7

precedes job J_8. However, we cannot order jobs J_6 and J_7. Therefore, instead of considering $3! = 6$ permutations of these three jobs, it is sufficient to consider two permutations: (J_6, J_7, J_8) and (J_7, J_6, J_8).

Due to Theorem 4.8, for the job shop problem $\mathcal{J}2/a_i \le p_i \le b_i, n_i \le 2/\mathcal{C}_{\max}$ with job set $J = J(12) \cup J(1) \cup J(2) \cup J(21) = \{J_1, J_2, \ldots, J_8\}$ and numerical input data given in Table 4.3 we obtain a solution which consists of the following four pairs of permutations:

$$\pi' = (J_1, J_2, J_3, J_4, J_6, J_7, J_8), \pi'' = (J_6, J_7, J_8, J_5, J_1, J_2, J_3);$$

$$\pi' = (J_1, J_2, J_3, J_4, J_7, J_6, J_8), \pi'' = (J_7, J_6, J_8, J_5, J_1, J_2, J_3);$$

$$\pi' = (J_2, J_1, J_3, J_4, J_6, J_7, J_8), \pi'' = (J_6, J_7, J_8, J_5, J_2, J_1, J_3);$$

$$\pi' = (J_2, J_1, J_3, J_4, J_7, J_6, J_8), \pi'' = (J_7, J_6, J_8, J_5, J_2, J_1, J_3).$$

Thus, instead of considering $q_{12}! \cdot q_{21}! = 3! \cdot 3! = 36$ permutations, it is sufficient to consider four pairs of permutations which certainly includes optimal Jackson's pair of permutations for any fixed vector $p \in T^J$ of job processing times.

Example 4.4 *Let us define the minimal solution to the problem $\mathcal{J}2/a_i \le p_i \le b_i, n_i \le 2/\mathcal{C}_{\max}$ for processing set of jobs $J = J(12) \cup J(1) \cup J(2) \cup J(21) = \{J_1, J_2, \ldots, J_{14}\}$, where jobs $J_1, J_2, \ldots, J_5$ from set $J(12)$ have route (M_1, M_2). Job $J_6 \in J(1)$ and job $J_7 \in J(1)$ have route (M_1). Jobs $M_8 \in J(2)$ and $M_9 \in J(2)$ have route (M_2). Jobs $J_{10}, J_{11}, \ldots, J_{14}$ from set $J(21)$ have route (M_2, M_1). The intervals of feasible job processing times are given in Table 4.4.*

Table 4.4: *Lower and upper bounds of job processing times in Example 4.4*

i	a_{i1}	b_{i1}	a_{i2}	b_{i2}
1	5	7	8	10
2	2	4	6	9
3	8	11	7	10
4	7	10	7	9
5	10	13	5	7
6	4	5	-	-
7	2	3	-	-
8	-	-	4	6
9	-	-	5	7
10	2	3	5	7
11	7	8	3	5
12	8	9	4	5
13	10	11	9	12
14	2	4	6	6

First, we consider uncertain flow shop problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{\max}$ with job set $J(12) = \{J_1, J_2, J_3, J_4, J_5\}$ and technological route (M_1, M_2). Due to Theorem 4.7 we find

the following dominance relations between jobs from set $J(12)$. *Job* J_2 *precedes job* J_1. *Job* J_1 *precedes job* J_3 *and job* J_4. *Job* J_3 *precedes job* J_5. *Job* J_4 *precedes job* J_5. *However, we cannot order jobs* J_3 *and* J_4. *Therefore, instead of considering* $5! = 120$ *permutations of these five jobs, it is sufficient to consider two permutations:* $(J_2, J_1, J_3, J_4, J_5)$ *and* $(J_2, J_1, J_4, J_3, J_5)$.

Next, we consider uncertain flow shop problem $\mathcal{F}2/a_i \le p_i \le b_i/\mathcal{C}_{\max}$ *with job set* $J(21) = \{J_{10}, J_{11}, J_{12}, J_{13}, J_{14}\}$ *and technological route* (M_2, M_1). *Due to Theorem 4.7 we find the following dominance relations between jobs from set* $J(21)$. *Job* J_{11} *precedes job* J_{13}. *Job* J_{12} *precedes job* J_{13}. *Job* J_{13} *precedes job* J_{10} *and job* J_{14}. *However, we cannot order jobs* J_{11} *and* J_{12}. *We cannot order jobs* J_{10} *and* J_{14}. *Therefore, instead of considering* $5! = 120$ *permutations of these five jobs, it is sufficient to consider four permutations:* $(J_{11}, J_{12}, J_{13}, J_{10}, J_{14})$, $(J_{11}, J_{12}, J_{13}, J_{14}, J_{10})$, $(J_{12}, J_{11}, J_{13}, J_{10}, J_{14})$ *and* $(J_{12}, J_{11}, J_{13}, J_{14}, J_{10})$.

Using Theorem 4.8 we obtain that minimal solution to the problem $\mathcal{J}2/a_i \le p_i \le b_i, n_i \le 2/\mathcal{C}_{\max}$ *with set of jobs* $J = \{J_1, J_2, \ldots, J_{14}\}$ *include the following pairs of permutations (the sequences for processing jobs of set* J_1 *and jobs of set* J_2 *are fixed with respect to job numbers):*

$$\pi' = (J_2, J_1, J_3, J_4, J_5, J_6, J_7, J_{11}, J_{12}, J_{13}, J_{10}, J_{14}),$$

$$\pi'' = (J_{11}, J_{12}, J_{13}, J_{10}, J_{14}, J_8, J_9, J_2, J_1, J_3, J_4, J_5);$$

$$\pi' = (J_2, J_1, J_3, J_4, J_5, J_6, J_7, J_{11}, J_{12}, J_{13}, J_{14}, J_{10}),$$

$$\pi'' = (J_{11}, J_{12}, J_{13}, J_{14}, J_{10}, J_8, J_9, J_2, J_1, J_3, J_4, J_5);$$

$$\pi' = (J_2, J_1, J_3, J_4, J_5, J_6, J_7, J_{12}, J_{11}, J_{13}, J_{10}, J_{14}),$$

$$\pi'' = (J_{12}, J_{11}, J_{13}, J_{10}, J_{14}, J_8, J_9, J_2, J_1, J_3, J_4, J_5);$$

$$\pi' = (J_2, J_1, J_3, J_4, J_5, J_6, J_7, J_{12}, J_{11}, J_{13}, J_{14}, J_{10}),$$

$$\pi'' = (J_{12}, J_{11}, J_{13}, J_{14}, J_{10}, J_8, J_9, J_2, J_1, J_3, J_4, J_5);$$

$$\pi' = (J_2, J_1, J_4, J_3, J_5, J_6, J_7, J_{11}, J_{12}, J_{13}, J_{10}, J_{14}),$$

$$\pi'' = (J_{11}, J_{12}, J_{13}, J_{10}, J_{14}, J_8, J_9, J_2, J_1, J_4, J_3, J_5);$$

$$\pi' = (J_2, J_1, J_4, J_3, J_5, J_6, J_7, J_{11}, J_{12}, J_{13}, J_{14}, J_{10}),$$

$$\pi'' = (J_{11}, J_{12}, J_{13}, J_{14}, J_{10}, J_8, J_9, J_2, J_1, J_4, J_3, J_5);$$

$$\pi' = (J_2, J_1, J_4, J_3, J_5, J_6, J_7, J_{12}, J_{11}, J_{13}, J_{10}, J_{14}),$$

$$\pi'' = (J_{12}, J_{11}, J_{13}, J_{10}, J_{14}, J_8, J_9, J_2, J_1, J_4, J_3, J_5);$$

$$\pi' = (J_2, J_1, J_4, J_3, J_5, J_6, J_7, J_{12}, J_{11}, J_{13}, J_{14}, J_{10}),$$

$$\pi'' = (J_{12}, J_{11}, J_{13}, J_{14}, J_{10}, J_8, J_9, J_2, J_1, J_4, J_3, J_5).$$

Digraph $\Gamma = (J, A)$ *(without transitive arcs), defining the above minimal solution of the problem* $\mathcal{J}2/a_i \le p_i \le b_i, n_i \le 2/\mathcal{C}_{\max}$ *is presented in Figure 4.3.*

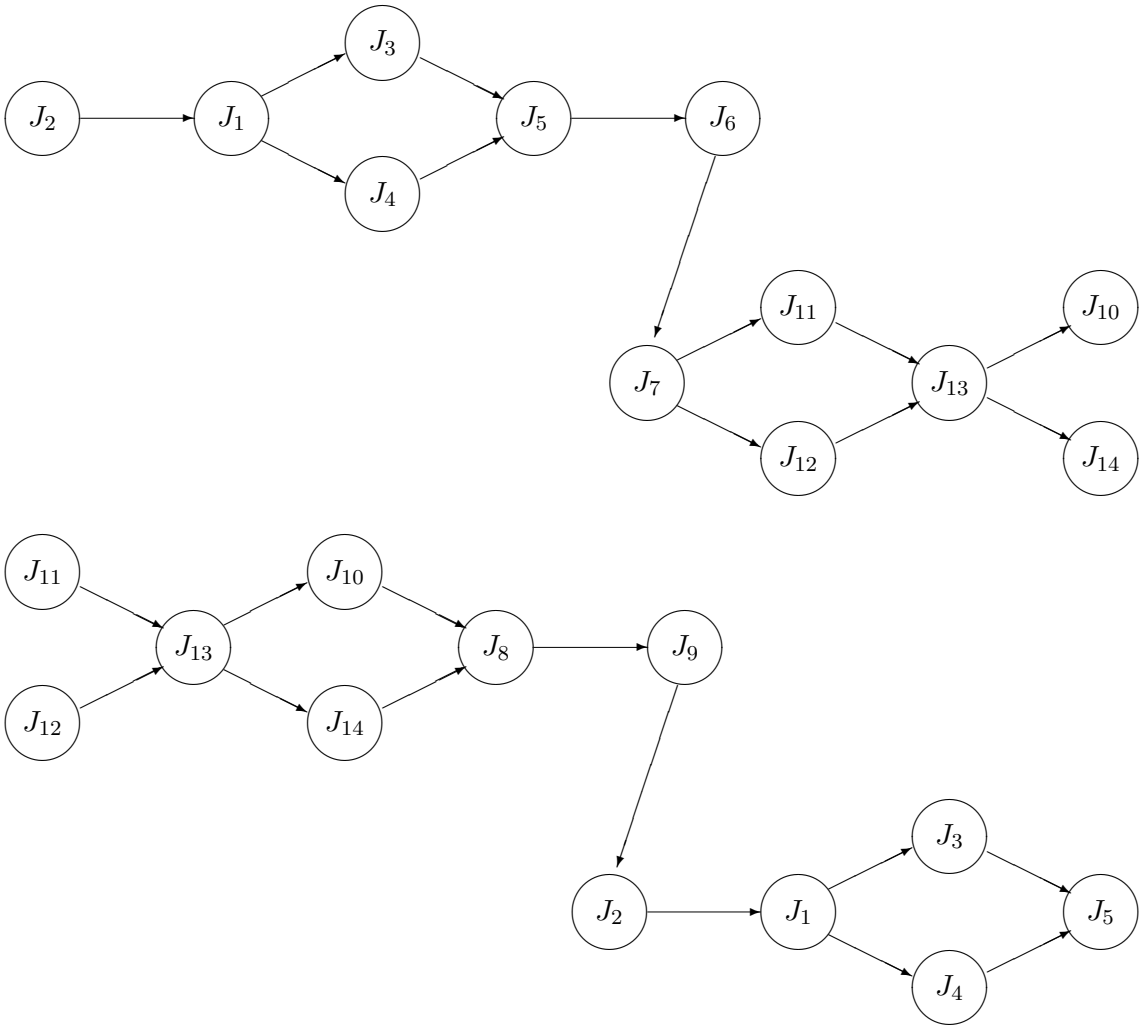

Figure 4.3: *Digraph* Γ *(without transitive arcs) for Example 4.4*

4.3 Flow Shop with Limited Machine Availability

This section deals with the scheduling problem of minimizing the makespan in the two-machine n-job flow shop with w non-availability intervals given on two machines. This problem is binary NP-hard even if there is only one non-availability interval ($w = 1$) either on the first machine or on the second machine in the technological route. If there are no non-availability intervals on any machine ($w = 0$), the two-machine flow shop problem with makespan criterion may be solved in $O(n \log n)$ time using Johnson's permutation of n jobs (see Johnson's rule at page 186).

We derive sufficient conditions for optimality of Johnson's permutation in the case of the given $w \geq 1$ non-availability intervals. The instrument we use is stability analysis which answers the question how stable an optimal schedule is if there are independent changes in the processing times of the jobs. The influence of possible variations of the processing times on the optimality of a schedule was investigated in Chapter 1 where the

stability radius of an optimal schedule was studied for a general shop with $m \geq 2$ machines. In contrast to stability analysis developed in Chapter 1, the variation of processing time studied in this section may have only one direction, since the non-availability intervals of a machine may increase the time interval used for processing a job but cannot decrease this time interval. Therefore, we have to modify the definition of stability radius for the purposes of this section.

Notations and Example

In what follows, we assume that all of the w non-availability intervals are known in advance (notation: *offline,* NC^{off}). Any job $J_i \in J = \{J_1, J_2, \ldots, J_n\}$ that cannot be completed before the non-availability interval of machine $M_k \in \{M_1, M_2\}$ processes this job can be continued just after machine M_k will be available again. To indicate this we use notation *pmtn* since the latter assumption is equivalent to allowance of operation preemptions. Therefore, in this section (and in the next section as well) it is allowed to violate Condition 4 (see Introduction, page 3). Using the three-field notation, the problem under consideration is denoted as $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$.

Let π_k be a Johnson's permutation of n jobs of set J, i.e., jobs in set J are sorted in Johnson's order. Such a permutation is optimal for makespan criterion when all machines are continuously available ($w = 0$) during the planning horizon (see Section 4.1). We study the question whether this permutation remains optimal for the case of offline known $w \geq 1$ non-availability intervals of each of the two machines. In a concrete schedule, enlargements of time intervals used for processing jobs may be caused by non-availability intervals of machines. The main idea realized in this section is to consider the non-availability interval on a machine as an additional part of job processing time. In order to take into account such increasing of total time interval used for processing a job, we compute the *stability polytope* and *stability radius* ρ_j of Johnson's permutation which is the minimum of maximal possible enlargements r_{ij} of jobs $J_i \in J$ on machine $M_j \in \{M_1, M_2\}$ such that Johnson's permutation is not changed. Such stability radius ρ_j can be computed in $O(n)$ time.

Then we compute the *enlargement polytope* and *enlargement radius* δ_j of the processing times of the operations on machine M_j which means the maximum of possible enlargements d_{ij} of time intervals used for processing job J_i on machine M_j caused by non-availability intervals. It is shown that permutation π_k from set $S^{\pi} = \{\pi_1, \pi_2, \ldots, \pi_{n!}\}$ remains *optimal* if inequality $d_{ij} \leq r_{ij}$ holds for each $i \in \{1, 2, \ldots, n\}$ and each $j \in \{1, 2\}$. The enlargement polytopes and radii δ_j can be computed in $O(w^2 + n\ log\ n)$ time. Further, we present sufficient conditions for the stability of an optimal permutation π_k. At the end of this section, the obtained results for stability analysis are demonstrated on a huge number of randomly generated two-machine flow shop problems with $5 \leq n \leq 10000$ and $1 \leq w \leq 1000$.

In this and next sections, we use the following notations, where $j \in \{1, 2\}$ means a machine index and $i \in \{1, 2, \ldots, n\}$ means a job index: w_j denotes number of non-availability intervals on machine M_j; N_{kj} denotes k^{th} non-availability interval on machine M_j; $s(N_{kj})$ denotes starting point of the k^{th} non-availability interval; $f(N_{kj})$ denotes endpoint of the k^{th} non-availability interval; $h(N_{kj})$ denotes length of the k^{th} non-availability interval provided that $h(N_{kj}) = f(N_{kj}) - s(N_{kj})$.

For the problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ under consideration, it is assumed that all w non-availability intervals are known before scheduling. Machine $M_j \in \{M_1, M_2\}$ is not available for processing jobs of set J from starting point $s(N_{kj})$ until endpoint $f(N_{kj}) = s(N_{kj}) + h(N_{kj}),\ k \in \{1, 2, \ldots, w_j\}$. The operation O_{ij} started before point $s(N_{kj})$ but not finished until point $s(N_{kj})$ is suspended during time interval of length $h(N_{kj})$ starting from point $s(N_{kj})$, then processing operation O_{ij} is resumed from point $f(N_{kj}) = s(N_{kj}) + h(N_{kj})$.

Let s_j (c_j, respectively) denote the earliest (latest) possible starting time of a job on machine M_j in a semiactive schedule. Let $s_{ij}(\pi_k)$ ($c_{ij}(\pi_k)$) denote starting (completion) time of operation O_{ij} in the semiactive schedule defined by permutation π_k of n jobs from set J.

Since the minimization of

$$C_{max}(\pi_k) = \max\{c_{ij}(\pi_k) : J_i \in J, j \in \{1, 2\}\}$$

is a *regular* criterion we consider only *semiactive schedules*. Each semiactive schedule is uniquely defined by a permutation of n jobs on each machine. For the case of the *two*-machine flow shop problem it is sufficient to consider the same permutation $\pi_k = (i_1, i_2, \ldots, i_n)$ of jobs on each of the two machines, since a set of such schedules is dominant for minimization the makespan.

If the set J of jobs is ordered due to Johnson's rule on each of the two machines, we say of Johnson's permutation of n jobs. In [Joh54], it has been shown that a semiactive schedule defined by Johnson's permutation is optimal for the case where all machines are continuously available throughout the planning horizon (when $w = 0$). Moreover, preemptions of operations cannot decrease the objective function value $C_{max}(\pi)$, and so the instance of problem $\mathcal{F}2//\mathcal{C}_{max}$ and the instance of problem $\mathcal{F}2/pmtn/\mathcal{C}_{max}$ with the same input data have the same optimal schedule defined by Johnson's permutation. Hereafter, the following example is used to illustrate the calculations.

Example 4.5 *We consider a three-job two-machine flow shop scheduling problem with $w = 9$ non-availability intervals on the two machines defined in Table 4.5.*

Table 4.5: *The non-availability intervals of machines in Example 4.5*

Machine M_1	
$s(N_{1,1}) = 2$	$f(N_{1,1}) = 3$
$s(N_{2,1}) = 11$	$f(N_{2,1}) = 13$
$s(N_{3,1}) = 16$	$f(N_{3,1}) = 17$
$s(N_{4,1}) = 19$	$f(N_{4,1}) = 20$
$s(N_{5,1}) = 24$	$f(N_{5,1}) = 26$
$w_1 = 5$	

Machine M_2	
$s(N_{1,2}) = 1$	$f(N_{1,2}) = 4$
$s(N_{2,2}) = 9$	$f(N_{2,2}) = 10$
$s(N_{3,2}) = 14$	$f(N_{3,2}) = 15$
$s(N_{4,2}) = 21$	$f(N_{4,2}) = 22$
$w_2 = 4$	

Table 4.6 presents the given processing times p_{ij} of jobs $J_i \in J = \{J_1, J_2, J_3\}$ on machines $M_j \in M = \{M_1, M_2\}$.

Table 4.6: *The job processing times for Example 4.5*

Machine M_1	Machine M_2
$p_{1,1} = 3$	$p_{1,2} = 5$
$p_{2,1} = 4$	$p_{2,2} = 1$
$p_{3,1} = 7$	$p_{3,2} = 2$

Next, we show how to compute values $r_{ij}, \rho_j, s_j, c_j, d_{ij}$ and δ_j for jobs $J_i \in J$ and machines $M_j \in \{M_1, M_2\}$. For this example, it will be shown that the schedule with the jobs following Johnson's permutation $\pi_k = (1, 3, 2)$ applied to the situation where the machines are not continuously available remains optimal.

Let π_k be a Johnson's permutation (which is optimal when both machines are continuously available during the planning horizon, i.e., for problem $\mathcal{F}2//\mathcal{C}_{max}$). Our aim is to answer the question whether this permutation remains optimal for problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ as well, i.e., when $w \geq 1$ non-availability intervals are given before scheduling and operation processing times are the same as in the problem $\mathcal{F}2//\mathcal{C}_{max}$.

A non-availability interval N_{kj} on machine M_j may be interpreted as an additional part of the processing time of the operation O_{ij} which processing by machine M_j is preempted due to interval N_{kj}. Indeed, machine M_j cannot process any job during interval N_{kj} and job J_i has to wait until endpoint of interval N_{kj}. In a concrete schedule, enlargement equal to $h(N_{kj})$ of completion time of job J_i is caused by non-availability interval N_{kj}.

Next we show how to compute the *stability radius* ρ_j for machine M_j which is the minimum of all maximal possible enlargements r_{ij} of the processing times of jobs $J_i \in J$ on machine M_j such that Johnson's permutation is not changed. Then we show how to compute the *enlargement radius* δ_j of the processing times of the operations on machine M_j which is the maximum of all maximal possible enlargements d_{ij} of jobs $J_i \in J$ on machine M_j caused by non-availability intervals on machine M_j. Johnson's permutation π_k remains optimal if $d_{ij} \leq r_{ij}$ for all jobs $J_i \in J$ and machines $M_j \in \{M_1, M_2\}$.

Stability Polytopes for Johnson's Permutation

General definition of stability radius of an optimal schedule was given in Chapter 1. Here we need simplified versions of the stability polytope and stability radius since non-availability intervals can increase the time interval used for processing the operation but cannot decrease this time interval. Moreover, stability radius considered in Chapter 1 defines the maximal independent simultaneous variations of the processing times such that the given schedule (digraph G_s) remains *optimal*. In Definition 4.4 which follows we do not take care directly about permutation *optimality* but we are concerned about the property of the given permutation to be a Johnson's one after increasing processing times of all or portion of jobs.

It should be noted that optimality of a permutation of n jobs does not imply that this permutation is Johnson's one even for the problem $\mathcal{F}2//\mathcal{C}_{max}$ with $w = 0$.

Definition 4.4 *The stability radius ρ_j of Johnson's permutation π_s on machine M_j is defined as the minimum value among all maximal possible enlargements r_{ij} of operations O_{ij}, $J_i \in J$, on machine $M_j \in \{M_1, M_2\}$ such that permutation π_s necessarily satisfies Johnson's rules for modified processing times. The closed polytopes*

$$P_j = \{x = (x_{1,j}, x_{2,j}, \ldots, x_{nj}) \ : \ p_{ij} \leq x_{ij} \leq p_{ij} + r_{ij}, \ J_i \in J\}, \ M_j \in \{M_1, M_2\},$$

in the space R^n_+ of non-negative n-dimensional real vectors are called stability polytopes of Johnson's permutation.

Stability polytope P_j defines all possible enlargements of processing times of operations O_{ij} on machine M_j such that the given permutation π_s of jobs J remains Johnson's permutation. As a result, permutation π_s is optimal for the problem $\mathcal{F}2//\mathcal{C}_{max}$ with modified processing times. Note that in this section, we investigate stability properties of the fixed permutation π_s from the set of all permutations $S^\pi = \{\pi_1, \pi_2, \ldots, \pi_{n!}\}$ of n jobs J. That is why for simplicity, we do not indicate permutation π_s or index s in the notations ρ_j, P_j etc.

Next, we prove the following lemma.

Lemma 4.13 *The stability radii ρ_j and stability polytopes P_j, $j \in \{1, 2\}$, can be computed in $O(n)$ time.*

PROOF. Let π_s be a permutation of n jobs following the Johnson's rule. W.l.o.g. we assume that $\pi_s = (J_1, J_2, \ldots, J_n)$. Let the first k jobs belong to set N_1 and the remaining $n-k$ jobs belong to set N_2, i.e., $N_1 = \{J_1, J_2, \ldots, J_k\}$ and $N_2 = \{J_{k+1}, J_{k+2}, \ldots, J_n\}$. (For definition N_1 and N_2, see Johnson's rule given at page 186 in Section 4.1.) To compute numbers r_{ij} we have to distinguish two cases: Either $J_i \in N_1$ or $J_i \in N_2$.

In the case when $J_i \in N_1$, inequality $p_{i,1} \leq p_{i,2}$ holds, and maximal possible enlargements $r_{i,1}$ are the minimum of two values $a^{(i,1)}$ and $b^{(i,1)}$ where $a^{(i,1)} = p_{i,2} - p_{i,1}$ represents the maximum amount of processing time one can add to the value $p_{i,1}$ such that job J_i will remain in set N_1 due to Johnson's rules. The value

$$b^{(i,1)} = \begin{cases} p_{i+1,1} - p_{i,1}, & i = 1, 2, \ldots, k-1, \\ \infty, & i = k, \end{cases}$$

represents the maximum amount of processing time (or infinity if $i = k$) one can add to the value $p_{i,1}$ such that the SPT ordering within the set N_1 is still preserved.

Similar arguments are valid for the case $J_i \in N_2$, i.e., when inequality $p_{i,1} \geq p_{i,2}$ holds. Every computation of an r_{ij} can be done in a constant time, and the stability polytopes P_j and stability radii ρ_j, $j \in \{1, 2\}$, can be computed in $O(n)$ time.

◇

Example 4.5 (continued). *In Table 4.7, the results of calculation values r_{ij} and ρ_j are shown for Example 4.5.*

If no processing time on machine M_1 is enlarged by more than two units and if no processing time on machine M_2 is enlarged by more than one unit, then permutation

Table 4.7: *Stability radii $\rho_j, M_j \in \{M_1, M_2\}$, for Example 4.5*

Machine M_1	
$p_{1,1} = 3$	$r_{1,1} = \min\{2, \infty\} = 2$
$p_{2,1} = 4$	$r_{2,1} = \infty$
$p_{3,1} = 7$	$r_{3,1} = \infty$
	$\rho_1 = 2$

Machine M_2	
$p_{1,2} = 5$	$r_{1,2} = \infty$
$p_{2,2} = 1$	$r_{2,2} = \min\{3, 1\} = 1$
$p_{3,2} = 2$	$r_{3,2} = \min\{5, \infty\} = 5$
	$\rho_2 = 1$

$\pi_k = (1, 3, 2)$ *remains Johnson's permutation for the non-availability intervals given in Table 4.5.*

Enlargement Polytopes

If $w_j = 0$, operation O_{ij} of job J_i on machine M_j may be processed without preemptions and so the length of interval for scheduling operation O_{ij} is equal to p_{ij}. However if $w_j \geq 1$, *scheduling time* (i.e., difference between completion time and starting time) of operation O_{ij} may be increased by the lengths of some non-availability intervals. Next, we determine the maximal possible enlargements of scheduling times of operations O_{ij} on machine $M_j \in \{M_1, M_2\}$ caused by non-availability intervals independently of a concrete schedule under consideration.

Definition 4.5 *The enlargement radius δ_j of the operations on machine M_j is defined as the maximum of possible enlargements d_{ij} of scheduling times of operations O_{ij}, $J_i \in J$, caused by non-availability intervals on machine M_j. The closed polytopes*

$$\Delta_j = \{x = (x_{1,j}, x_{2,j}, \ldots, x_{nj}) \ : \ p_{ij} \leq x_{ij} \leq p_{ij} + d_{ij}, \ J_i \in J\}, \ M_j \in \{M_1, M_2\},$$

in the space R^n_+ are called enlargement polytopes.

The enlargement radius gives the maximum amount that the completion time of an operation on a machine may be delayed by one or more consecutive non-availability intervals which may be included in the scheduling interval for processing the operation. We prove the following lemma.

Lemma 4.14 *The enlargement radii δ_j and enlargement polytopes Δ_j, $M_j \in \{M_1, M_2\}$, can be computed in $O(w^2 + n \log n)$ time.*

Proof. For machine M_1 we do the following. If operation $O_{i,1}$ starts or is resumed immediately after a non-availability interval on machine M_1, we have to add the length of this non-availability interval to the scheduling time of operation $O_{i,1}$.

We compute the maximal sum D_i of lengths of i consecutive non-availability intervals of machine M_1 for $i = 1, 2, \ldots, w_1$:

$$D_i = \max_{a=1}^{w_1+1-i} \left\{ \sum_{k=a}^{a+i-1} h(N_{k,1}) \right\}.$$

Using the given non-availability intervals, we can determine *availability intervals* of machine $M_j \in \{M_1, M_2\}$.

Let A_{kj} denote k^{th} availability interval, i.e., k^{th} maximal interval when machine M_j is available to process a job. Let $s(A_{kj})$, $f(A_{kj})$ and $h(A_{kj})$ be starting point, endpoint and length of interval A_{kj}, respectively.

W.l.o.g. we assume that $s(N_{1,1}) > 0$. Indeed if $s(N_{1,1}) = 0$, we can change earliest possible starting time $s_1 = 0$ on machine M_j by $s_1 = f(N_{1,1})$ and an optimal permutation remains the same. Thus, we obtain $s(A_{1,1}) = 0$, $f(A_{1,1}) = s(N_{1,1})$ and $h(A_{1,1}) = f(A_{1,1}) - s(A_{1,1}) > 0$.

For $k = 2, 3, \ldots, w_1$ we obtain $s(A_{k,1}) = f(N_{k-1,1})$, $f(A_{k,1}) = s(N_{k,1})$ and $h(A_{k,1}) = f(A_{k,1}) - s(A_{k,1}) > 0$.

We set $E_0 = 0$ and compute sum E_i of lengths of i consecutive availability intervals $A_{k,1}$ of machine M_1 for $i = 1, 2, \ldots, w_1 + 1$:

$$E_i = \min_{a=1}^{w_1+1-i} \left\{ \sum_{k=a}^{a+i-1} h(A_{k,1}) \right\}.$$

These computations need $O(w^2)$ time. Using numbers D_k and E_k for $k = 1, 2, \ldots, w_1$ we can calculate the maximal possible enlargement of scheduling time for processing operation $O_{i,1}$ of job J_i on machine M_1. Indeed, to process operation $O_{i,1}$ at most k consecutive availability intervals of machine M_1 have to be used if $E_{k-1} \leq p_{i,1} < E_k$. These k availability intervals are alternated with k non-availability intervals of machine M_1. Therefore the scheduling time for processing operation $O_{i,1}$ may be increased by at most D_k. Hence, the scheduling time for processing each operation $O_{i,1}$ on machine M_1 with $E_{k-1} \leq p_{i,1} < E_k$ may have the maximal enlargement caused by non-availability intervals with total length equal to D_k.

Similar arguments are valid for machine M_2. Thus, the calculation of the enlargement of each job from set J takes $O(w + n\ log\ n)$ time, so we need $O(w^2 + n\ log\ n)$ time for the calculation of the enlargement radii δ_j and enlargement polytope $\Delta_j,\ j \in \{1, 2\}$.

$\diamond$

Example 4.5 (continued). *In Table 4.8, the enlargement radii for machines M_1 and M_2 are presented for Example 4.5. Thus, any scheduling time of operation on machine M_1 (machine M_2) cannot be enlarged by more than three units (by more than two units, respectively) if one sticks non-availability intervals with jobs J.*

Similarly, any scheduling time on machine M_2 cannot be enlarged by more than two units.

Using Lemma 4.13 and Lemma 4.14 we prove the following theorem.

Theorem 4.9 *Johnson's permutation π_s defined for problem $\mathcal{F}2//\mathcal{C}_{max}$ remains optimal for problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ with the same processing times if*

$$d_{ij} \leq r_{ij},\ i \in \{1, 2, \ldots, n\},\ j \in \{1, 2\}. \tag{4.35}$$

To test condition (4.35) takes $O(w^2 + n\ log\ n)$ time.

Table 4.8: *Enlargement radii $\delta_j, M_j \in \{M_1, M_2\}$, for Example 4.5*

Machine M_1	
$p_{1,1} = 3$	$d_{1,1} = 2$
$p_{2,1} = 4$	$d_{2,1} = 3$
$p_{3,1} = 7$	$d_{3,1} = 3$
	$\delta_1 = 3$

Machine M_2	
$p_{1,2} = 5$	$d_{1,2} = 2$
$p_{2,2} = 1$	$d_{2,2} = 1$
$p_{3,2} = 2$	$d_{3,2} = 1$
	$\delta_2 = 2$

PROOF. We prove this claim by contradiction. Let there exist permutation $\pi_k \in S^\pi$ of n jobs J such that

$$C_{max}(\pi_k) < C_{max}(\pi_s) \tag{4.36}$$

for the problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ mentioned in Theorem 4.9. We define modification (we call it Problem 1) of the above problem $\mathcal{F}2//\mathcal{C}_{max}$ which have the following processing times $p_{ij}^{'}$ of operations O_{ij}:

$$p_{ij}^{'} = \begin{cases} h(N_{kj}) + c_{ij}(\pi_k) - s_{ij}(\pi_k), & \text{if there exists } N_{kj} \text{ with } f(N_{kj}) = s_{ij}(\pi_k), \\ c_{ij}(\pi_k) - s_{ij}(\pi_k), & \text{otherwise.} \end{cases}$$

Due to Definition 4.5 we have $p_{ij}^{'} \leq p_{ij} + d_{ij}$. Due to Definition 4.4 and inequalities (4.35), permutation π_s remains Johnson's permutation for Problem 1. As far as $w = 0$ for Problem 1, Johnson's permutation π_s is optimal. Therefore $C_{max}(\pi_s) \leq C_{max}(\pi_k)$ which contradicts to inequality (4.36).

The complexity of calculation values r_{ij} and d_{ij} given in Lemma 4.13 and Lemma 4.14 defines complexity of testing inequalities (4.35).

◇

Next, we present more simple sufficient conditions for optimality of permutation using information about location of some operations O_{ij} and non-availability intervals N_{kj} in the concrete schedule.

Sufficient Conditions for Stability of an Optimal Permutation

Let $\pi_v = (J_{i_1}, J_{i_2}, \ldots, J_{i_n})$ be optimal permutation for the problem $\mathcal{F}2//\mathcal{C}_{max}$, $s(\pi_v)$ be schedule for problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ defined by permutation π_v, and $c_{i_k,j}(s(\pi_v))$ denote completion time of operation $O_{i_k,j}$ in the schedule $s(\pi_v)$ for problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ with $1 \leq k \leq n$.

Theorem 4.10 *Permutation π_v which is optimal for the problem $\mathcal{F}2//\mathcal{C}_{max}$ remains optimal for problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ with the same processing times if there exists a time point $t = c_{i_k,1}(s(\pi_v))$, $1 \leq k \leq n$, for the schedule $s(\pi_v)$ such that*
(i) the smallest k operations on machine M_1 and non-availability intervals of machine M_1 completely fill the time interval $[0, t]$,
(ii) the smallest $n + 1 - k$ operations on machine M_2 and non-availability intervals of machine M_2 completely fill the time interval $[t, c_{i_n,2}(s(\pi_v))]$.

PROOF. Since in the schedule $s(\pi_v)$ there are no idle times on machine M_1 in the interval $[0,t]$ and on machine M_2 in the interval $[t, c_{i_n,2}(s(\pi_v))]$, any transposition of jobs within set $\{i_1, i_2, \ldots, i_k\}$ cannot decrease the value $C_{\max}(s(\pi_v))$. Similar arguments are valid for any transposition of jobs within set $\{i_k, i_{k+1}, \ldots, i_n\}$.

Let s be any semiactive schedule constructed for the problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ under consideration. In any semiactive schedule, machine M_1 has no idle time which belong to availability intervals within interval $[1, c_{i_n,1}(s)]$. Thus, operations $O_{t,1}$, $k+1 \leq t \leq n$, and non-availability intervals of machine M_1 completely fill the interval $[t, c_{i_n,1}(s(\pi_v))]$. From condition (ii) it follows that for schedule $s(\pi_v)$ only interval $[0,t]$ may include idle time of machine M_2 which belong to availability intervals. Let $l(s(\pi_v))$ denote the total length of such idle times. If $l(s(\pi_v)) = 0$, then schedule $s(\pi_v)$ is optimal for problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$. If $l(s(\pi_v)) > 0$, then decreasing value $C_{\max}(s(\pi_v)) = c_{i_n,2}(s(\pi_v))$ is equivalent to decreasing the total length $l(s(\pi_v))$ of idle times.

To finish the proof we have to consider each semiactive schedule s obtained from schedule $s(\pi_v)$ after transposition job J_{i_l}, $1 \leq l \leq k$, and job J_{i_m}, $k+1 \leq m \leq n$. From condition (i) it follows that machine M_1 cannot completely process more than k operations within interval $[0,t]$, and machine M_2 cannot completely process more than $k-1$ operations within interval $[0,t]$. Consequently, from condition (ii) it follows that total idle time of machine M_2 which belong to availability intervals within interval $[0,t]$ in the schedule s cannot be less than $l(s(\pi_v))$, and thus $C_{\max}(s(\pi_v)) \leq C_{\max}(s)$.

◇

Note that in the above proof of Theorem 4.10 we do not use that π_v is a Johnson's permutation for the problem $\mathcal{F}2//\mathcal{C}_{max}$ indicated in Theorem 4.10. Indeed, if conditions (i) and (ii) are valid for any permutation π_v which is optimal for the problem $\mathcal{F}2//\mathcal{C}_{max}$ (but π_v may not satisfy Johnson's rules), then permutation π_v remains optimal for the problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ as well. The following corollaries are also valid for any permutation π_v which is optimal for problem $\mathcal{F}2//\mathcal{C}_{max}$.

Corollary 4.1 *If machine M_2 is filled in the interval $[c_{i_n,1}(s(\pi_v)), c_{i_n,2}(s(\pi_v))]$ with only the smallest operation on machine M_2 and non-availability intervals, then schedule $s(\pi_v)$ is optimal for problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$.*

Corollary 4.2 *If $p_{i_1,1} = \min\{p_{i_k,1} : 1 \leq k \leq n\}$ and machine M_2 has no idle time in the interval $[c_{i_1,1}(s(\pi_v)), c_{i_n,2}(s(\pi_v))]$, then schedule $s(\pi_v)$ is optimal for problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$.*

For the proof of Corollary 4.1 (Corollary 4.2) it is sufficient to note that conditions (i) and (ii) from Theorem 4.10 necessarily hold for $k = n$ (for $k = 1$, respectively). Again it is not assumed that permutation π_v used in Corollary 4.1 and in Corollary 4.2 satisfies Johnson's rules, however these corollaries are often realized just for Johnson's permutation. It is easy to see that testing Corollary 4.1 takes $O(w_2)$ time, while testing Corollary 4.2 takes $O(w_2 + n)$ time.

In order to reduce the number of non-availability intervals that have to be considered for testing Theorem 4.9 and Theorem 4.10, one can use the following lemma.

Lemma 4.15 *The earliest possible starting times s_j and the latest possible completion times c_j of any job from set J on machine $M_j, j \in \{1,2\}$, in a semiactive schedule constructed for the problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ can be computed in $O(w+n)$ time.*

PROOF. On machine M_1, the earliest starting time of a job is equal to

$$s_1 = \max \begin{cases} f(N_{1,1}), & \text{if } s(N_{1,1}) = 0, \\ 0, & \text{otherwise.} \end{cases}$$

The latest completion time of a job from set J on machine M_1 is equal to $c_1 = p_1 + h + g$ with $p_1 = \sum p_{i,1}$ and $h = \sum_{i=1}^{k} h(N_{i,1})$ with $s(N_{k,1}) < p_1$ for maximal k. Let g initially be zero, and k be taken from the former computation. The value g is increased by $h(N_{k+1,1})$ and k is increased by 1 as long as $s(N_{k+1,1}) < p_1 + h + g$. The computation of p_1 takes $O(n)$ time and the computation of h and g takes $O(w)$ time. Hence, we need $O(w+n)$ time to compute the latest completion time of any job from set J on machine M_1.

Let $p_{min}(1)$ be the smallest processing time of an operation on machine M_1. On machine M_2, the earliest possible starting time of any job is the completion time of the first operation (which is assumed to be the smallest one) on machine M_1. Let $d = p_{min}(1) + c$. The values c and k are initially equal to 0. The value c is increased by $h(N_{k+1,1})$ and k is increased by 1 as long as $s(N_{k+1,1}) < p_{min}(1) + c$. Then we have

$$s_2 = \max \begin{cases} f(N_{is}), & \text{if there is a non-availability interval} \\ & N_{i,2} \text{ with } s_{i,2} \le d \text{ and } c_{i2} \ge d, \\ d, & \text{otherwise.} \end{cases}$$

The above computation takes $O(w+n)$ time. Let $p_{max}(2)$ be the largest processing time of an operation on machine M_2. The latest possible completion time of any job on machine M_2 is estimated by the upper bound $c_2 = c_1 + \sum p_{i,2} + d$. The value k is the maximum index with $s(N_{k,2}) \le c_1$. The value d initially is equal to $\max\{f(N_{k,2}) - c_1, 0\}$. The value d is increased by $h(N_{k+1,2})$ and k is increased by 1 as long as $s(N_{k+1,2}) < c_1 + \sum p_{i,2} + d$. This computation takes $O(w+n)$ time.

◇

Example 4.5 (continued). *Table 4.7 and Table 4.8 show the results for the computation of the stability and enlargement radii for machine M_1 and machine M_2. One can see that due to Theorem 4.9 Johnson's permutation $(1,3,2)$ remains optimal for the non-availability case under consideration since $d_{ij} \le r_{ij}$ for all $J_i \in J$ and $j \in \{1,2\}$. Also, Corollary 4.1 and Theorem 4.10 are valid for this example of the problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$.*

In Example 4.5 and in the computational experiments, non-negative *integer* numbers are used as processing times and lengths of non-availability intervals. However, it is obvious that all the above results are valid for non-negative *real* input data as well.

Computational Experiments

Stability analysis was performed on a huge number of randomly generated problems $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$. For each problem instance with $w \ge 1$ we constructed at least

one Johnson's permutation for the corresponding flow shop problem $\mathcal{F}2//\mathcal{C}_{max}$ with the same processing times but without non-availability intervals ($w = 0$), and answered the question: Is this permutation optimal for the original problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ with given $w \geq 1$ non-availability intervals? To this end, we tested sufficient conditions proven in this section. To minimize running time of the algorithm, these conditions were tested in an increasing order of their complexity up to the first positive answer (if any) to the above question. More formally, the following algorithm has been realized.

Algorithm for $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$

Input: Processing times for jobs J and non-availability intervals of machines M.

Output: Optimal schedule for the problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ or feasible schedule if algorithm terminates at *Step 6*.

Step 1: Construct Johnson'permutation π for the problem $\mathcal{F}2//\mathcal{C}_{max}$.

Step 2: **IF** π satisfies Corollary 4.1 **THEN GOTO** *Step 7*.

Step 3: **IF** π satisfies Corollary 4.2 **THEN GOTO** *Step 7*.

Step 4: Construct $\lambda = \min\{\lambda^*, 2^k\}$ Johnson's permutations $\pi_1, \pi_2, \ldots, \pi_\lambda$.

Step 5: **FOR** $i = 1, 2, \ldots, \lambda$ **DO**
BEGIN
Set $\pi := \pi_i$.
IF π satisfies Corollary 4.1 **THEN GOTO** *Step 7*.
IF π satisfies Corollary 4.2 **THEN GOTO** *Step 7*.
IF π satisfies Theorem 4.10 **THEN GOTO** *Step 7*.
IF π satisfies Theorem 4.9 **THEN GOTO** *Step 7*.
END

Step 6: Optimality of permutations π_i, $i = 1, 2, \ldots, \lambda$, for the problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ is not proven **STOP**.

Step 7: Permutation π is optimal for the problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ **STOP**.

Steps 4 and 5 in the above algorithm are realized since there may be more than one Johnson's permutation. Indeed, if there are k equalities $p_{ij} = p_{kj}$ for different jobs $J_i \in J$ and $J_k \in J$, then there may be 2^k possible Johnson's permutations. So, it was tested up to $\min\{\lambda^*, 2^k\}$ Johnson's permutations for each instance. In the experiments, we set $\lambda^* = 1024$.

For each combination of n and w, Tables 4.9 - 4.18 present the percentage of instances in the series for which calculation based on the above algorithm found Johnson's permutation which remains optimal in spite of the given w non-availability intervals.

The above algorithm was coded in C++. For the computational experiments, we used an AMD 1200 MHz processor with 1024 MB main memory. For the *small* and *moderate*

instances similar to those considered in [KBF+02] we made 10000 tests in each series, i.e., for each combination of n and w. These results are presented in Table 4.9 - Table 4.13. For the *large* instances presented in Tables 4.14 - 4.18 we made 1000 tests in each series.

Table 4.9: *Percentage of solved instances with $w_1 > 0$ and $w_2 > 0$*

Number of jobs	Number of non-availability intervals									
	1	2	3	4	5	6	7	8	9	10
5	74.2%	72.1%	72.3%	72.7%	72.3%	73.4%	74.1%	74.6%	75.1%	76.2%
10	87.5%	85.0%	84.6%	83.8%	83.5%	83.4%	83.5%	83.1%	84.4%	83.5%
15	91.1%	90.7%	90.0%	89.9%	89.6%	89.0%	88.7%	87.9%	88.5%	88.5%
20	93.4%	93.2%	92.5%	92.3%	92.1%	91.9%	91.7%	91.3%	90.8%	91.3%
25	94.5%	94.1%	94.3%	93.6%	93.3%	93.7%	93.8%	93.0%	93.2%	92.4%
30	95.1%	94.9%	95.0%	95.1%	94.7%	94.5%	93.9%	94.2%	94.0%	94.4%
35	95.5%	95.7%	95.5%	95.9%	95.4%	95.3%	95.5%	95.4%	95.1%	94.9%
40	96.0%	96.2%	96.1%	96.3%	96.1%	96.1%	95.9%	95.8%	95.5%	95.6%
45	96.6%	96.9%	96.6%	96.4%	96.2%	96.5%	96.3%	96.2%	96.1%	96.1%
50	97.2%	97.0%	96.8%	96.6%	96.7%	96.7%	96.9%	96.6%	96.9%	96.4%
55	97.2%	97.2%	97.2%	97.5%	96.8%	97.1%	97.2%	97.1%	96.8%	96.8%
60	97.7%	97.4%	97.6%	97.2%	97.3%	97.3%	97.2%	97.1%	97.2%	97.1%
65	97.6%	97.4%	97.7%	97.7%	97.3%	97.3%	97.3%	97.2%	97.3%	97.4%
70	97.8%	97.5%	97.9%	98.0%	97.5%	97.9%	97.7%	97.2%	97.8%	97.5%
75	97.7%	98.1%	97.9%	98.2%	97.8%	97.7%	97.8%	97.6%	97.8%	97.9%
80	98.1%	98.0%	98.1%	97.9%	98.1%	98.2%	98.0%	97.9%	97.6%	97.9%
85	98.1%	98.3%	98.2%	98.2%	98.3%	98.0%	98.2%	98.1%	98.0%	98.0%
90	98.5%	98.5%	98.3%	98.2%	98.3%	97.9%	98.1%	98.3%	98.2%	98.0%
95	98.3%	98.3%	98.4%	98.2%	98.1%	98.3%	98.2%	98.3%	98.2%	98.2%
100	98.5%	98.5%	98.5%	98.5%	98.4%	98.4%	98.3%	98.3%	98.2%	98.3%

Lemma 4.15 was used during the generation of random instances: The non-availability intervals were chosen in such a way that every non-availability interval was counted. This means that if c_1 and c_2 are the completion times of the last job on machine M_1 and M_2 (in the schedule with non-availability intervals), then every non-availability interval was located either in the segment $[0, c_1]$ for machine M_1 or in the segment $[0, c_2]$ for machine M_2.

Problems of Small and Moderate Sizes

First, we tested problems similar to those considered in [KBF+02], i.e., with

- $5, 10, 15, \ldots, 100$ jobs with integer processing times uniformly distributed in the range $[1, 1000]$ and with
- $1, 2, 3, \ldots, 10$ non-availability intervals (on both machines) with integer lengths uniformly distributed in the range $[1, 1000]$.

In [KBF+02], a branch-and-bound algorithm was developed and used for constructing optimal schedules for problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$. Due to the limit t of the running time used for each instance (limit t was equal to 1000 seconds) not all instances were optimally solved within limit t. For similar instances, we answered the question of how many schedules following Johnson's order (constructed for problem $\mathcal{F}2//\mathcal{C}_{max}$) remained optimal in spite of the non-availability intervals on the machines (i.e., for problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$). We tested only sufficient conditions and our algorithm does not

guarantee to find an optimal schedule for some problems $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$. However, our experiments shown that such instances arise rather seldom for the most randomly generated problems. We considered 10000 instances in each series of small problems (in [KBF+02], 10 instances were tested in each series).

Tables 4.9 - 4.13 present the percentage of problem instances which were (optimally) solved due to the stability analysis via the algorithm described at page 222. We do not present running times for series of small and moderate instances, since the running times were very close for different instances and very small: The *maximum* running time for each instance in the series presented in Tables 4.9, 4.10 and 4.11 was 0.000125 seconds, and that in the series presented in Tables 4.12 and 4.13 was 0.000731 seconds.

Table 4.10: *Percentage of solved instances with* $w = w_2$

Number of jobs	Number of non-availability intervals									
	1	2	3	4	5	6	7	8	9	10
5	76.7%	75.0%	74.5%	75.2%	76.6%	77.3%	77.9%	77.6%	79.6%	80.3%
10	87.9%	86.4%	85.2%	84.4%	83.1%	83.7%	85.0%	84.5%	85.4%	85.6%
15	91.7%	91.1%	90.4%	88.9%	89.5%	88.9%	89.2%	88.9%	89.2%	88.9%
20	93.6%	92.6%	92.9%	92.2%	92.5%	91.5%	91.7%	91.0%	90.4%	90.5%
25	94.5%	94.5%	94.3%	93.5%	93.7%	92.9%	92.6%	93.2%	92.7%	92.6%
30	95.3%	95.0%	94.6%	94.5%	94.0%	94.1%	94.4%	94.0%	94.0%	93.7%
35	96.0%	95.8%	95.9%	95.7%	95.3%	95.4%	94.8%	94.5%	94.7%	94.8%
40	96.3%	96.4%	95.9%	95.8%	95.8%	96.2%	95.5%	96.0%	95.8%	95.4%
45	96.5%	96.8%	96.5%	96.5%	96.0%	96.4%	96.2%	96.0%	95.7%	96.0%
50	97.2%	96.8%	96.8%	96.7%	96.6%	96.8%	96.8%	96.5%	96.4%	96.0%
55	97.1%	97.4%	97.3%	97.2%	97.0%	96.9%	96.5%	96.8%	96.6%	96.8%
60	97.4%	97.4%	97.5%	97.5%	96.9%	97.1%	97.0%	96.9%	97.0%	96.8%
65	97.5%	97.7%	97.5%	97.5%	97.4%	97.3%	97.2%	97.4%	97.5%	96.8%
70	98.0%	98.0%	97.9%	97.7%	98.0%	97.7%	97.6%	97.4%	97.8%	97.5%
75	98.1%	98.0%	97.8%	97.7%	97.7%	97.8%	97.8%	97.8%	97.8%	97.9%
80	98.2%	98.2%	98.2%	98.0%	98.1%	97.9%	97.9%	97.8%	97.6%	97.6%
85	98.1%	98.3%	98.2%	98.2%	98.0%	97.8%	98.1%	98.1%	97.9%	98.1%
90	98.4%	98.2%	98.3%	98.1%	98.3%	98.2%	98.0%	98.1%	98.2%	98.0%
95	98.6%	98.4%	98.3%	98.4%	98.1%	98.2%	98.3%	98.2%	98.0%	97.8%
100	98.3%	98.3%	98.7%	98.5%	98.4%	98.2%	98.2%	98.2%	98.3%	98.2%

Table 4.9 presents small problems. We can compare Table 4.9 with Table 1 in [KBF+02]. The percentage of solved instances in our experiments were less for $n = 5, n = 10$ and $n = 15$, but almost the same for $15 \leq n \leq 100$. Moreover, the branch-and-bound algorithm used essentially more running time (1000 seconds were not sufficient to solve some small instances).

Due to an NP-hardness proof and experiments, in [KBF+02], it was shown that instances in which the values $p_{i,2}$ were a double of the values $p_{i,1}$ (i.e., $p_{i,2} = 2p_{i,1}$) were much harder than instances with processing times uniformly generated for both machines. Our computations confirmed this property for small problem size (compare Table 4.9 with Table 4.12, and Table 4.11 with Table 4.13). From comparing Table 4.12 with Table 3 in [KBF+02] it follows that our approach has a close percentage of solved problems (since limit t was used for the branch-and-bound algorithm) and essentially outperforms the branch-and-bound algorithm in the running time.

Along with the case when non-availability intervals are on both machines: $w_1 > 0$ and $w_2 > 0$, we tested the cases (classes) of problems when either $w_1 = 0$ or $w_2 = 0$. Theoretically, the case with $w_1 = 0$ seems to be harder than the case with $w_2 = 0$

Table 4.11: *Percentage of solved instances with $w = w_1$*

Number of jobs	Number of non-availability intervals									
	1	2	3	4	5	6	7	8	9	10
5	71.4%	67.4%	67.0%	69.0%	69.6%	71.0%	72.5%	73.7%	73.8%	76.4%
10	86.2%	83.3%	82.2%	80.9%	81.2%	81.0%	81.2%	82.6%	81.6%	82.2%
15	91.2%	89.7%	89.5%	88.1%	87.7%	87.6%	87.1%	87.1%	88.0%	88.1%
20	93.0%	92.9%	92.0%	91.7%	90.6%	90.7%	90.5%	90.0%	90.5%	90.2%
25	94.1%	93.8%	93.8%	93.1%	92.8%	92.8%	92.5%	92.2%	92.4%	92.1%
30	95.1%	94.9%	94.2%	95.2%	94.7%	94.1%	94.2%	93.8%	93.9%	93.5%
35	96.1%	95.8%	95.5%	95.3%	94.9%	94.4%	95.1%	94.7%	94.3%	94.3%
40	96.5%	96.1%	95.8%	95.8%	95.9%	95.5%	95.2%	95.6%	95.1%	95.2%
45	97.0%	96.6%	96.4%	96.5%	96.1%	96.2%	95.7%	95.9%	95.9%	95.9%
50	96.9%	96.6%	96.4%	96.7%	96.9%	96.7%	96.5%	96.5%	96.5%	96.2%
55	97.4%	97.2%	96.7%	96.9%	96.9%	97.1%	97.0%	97.0%	96.8%	96.6%
60	97.5%	97.7%	97.2%	97.0%	97.3%	97.0%	97.1%	96.8%	97.1%	97.0%
65	97.9%	97.5%	97.5%	97.6%	97.6%	97.3%	97.4%	97.3%	97.2%	97.2%
70	97.5%	97.7%	98.0%	97.9%	97.8%	97.5%	97.6%	97.1%	97.5%	97.8%
75	97.9%	98.0%	97.9%	97.8%	97.7%	97.8%	97.8%	97.6%	97.5%	97.7%
80	98.2%	98.4%	98.0%	98.0%	98.0%	97.6%	97.8%	97.5%	97.8%	97.7%
85	98.2%	98.2%	98.3%	97.8%	98.0%	98.1%	98.1%	97.7%	97.9%	97.8%
90	98.3%	98.1%	98.4%	98.0%	98.2%	98.3%	98.2%	98.0%	98.2%	97.7%
95	98.5%	98.4%	98.4%	98.2%	98.3%	98.3%	98.1%	97.9%	98.3%	98.1%
100	98.4%	98.3%	98.3%	98.4%	98.5%	98.4%	98.4%	98.2%	98.1%	98.3%

since non-availability interval on machine M_1 may cause idle time on machine M_2 (while the opposite claim is not true). So, we tested the above two cases (see Table 4.10 for $w = w_1$, and Table 4.11 for $w = w_2$). From Tables 4.10 and 4.11, it follows that there are computational differences between cases $w = w_1$ and $w = w_2$ (problem with $w = w_1$ are often harder than problems with $w = w_2$), however these differences are not so significant.

The hardest problems for our stability analysis were obtained due to the union of the above two difficulties. Table 4.13 presents the computational results for the problems with both equalities $w = w_1$ and $p_{i,2} = 2p_{i,1}$. The worst results were obtained for series with a small number of jobs ($n = 5, n = 10, n = 15$ and $n = 20$) and a large number of non-availability intervals (see right-upper corner of Table 4.13). Thus, sufficient conditions derived in this section are disappointing for the problems with $p_{i,2} = 2p_{i,1}$ and $w = w_1 > n$. Such a class of problems needs to be studied in more details.

Note that the easiest problems for stability analysis based on the above computational scheme were obtained for the union of equality $w = w_2$ and equality $p_{i,2} = 2p_{i,1}$: All of the 10000000 instances of such problems were optimally solved in our experiments.

Problems of Large Size

Due to the small running time, we were able to investigate much larger instances than those considered in [KBF$^+$02]. We tested problems $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$) with

- $1000, 2000, \ldots, 10000$ jobs with integer processing times uniformly distributed in the range $[1, 1000]$ and with
- $10, 100, 500, 1000$ non-availability intervals with integer lengths uniformly distributed in the range $[1, 1000]$ on both machines.

Table 4.12: *Percentage of solved instances with* $p_{i,2} = 2p_{i,1}$, $w_1 > 0$ *and* $w_2 > 0$

Number of jobs	Number of non-availability intervals									
	1	2	3	4	5	6	7	8	9	10
5	64.5%	50.6%	44.4%	42.1%	40.0%	38.2%	37.7%	37.8%	36.5%	36.0%
10	66.6%	54.2%	47.4%	42.5%	40.3%	36.8%	35.6%	34.6%	33.9%	33.1%
15	68.0%	55.0%	49.2%	45.0%	42.2%	39.1%	37.5%	35.9%	35.6%	34.1%
20	69.4%	56.7%	50.8%	46.7%	43.9%	41.6%	39.9%	38.6%	36.6%	35.2%
25	68.4%	58.2%	52.2%	47.8%	45.8%	42.2%	40.2%	39.0%	37.4%	37.5%
30	68.8%	58.5%	52.7%	48.9%	45.4%	44.2%	42.4%	40.9%	39.8%	38.0%
35	68.6%	58.7%	53.4%	50.0%	46.2%	44.9%	43.0%	41.7%	40.3%	39.4%
40	69.7%	58.2%	52.6%	49.1%	47.2%	45.1%	43.0%	41.1%	41.8%	39.6%
45	69.9%	59.2%	53.2%	50.0%	47.6%	44.9%	43.7%	42.3%	41.2%	41.1%
50	70.3%	60.1%	54.5%	49.5%	47.6%	47.0%	44.5%	44.8%	43.3%	41.0%
55	68.9%	59.1%	54.1%	50.5%	48.0%	46.5%	46.8%	43.6%	42.1%	42.1%
60	70.0%	59.5%	53.8%	50.6%	49.2%	47.4%	46.0%	43.8%	43.7%	42.2%
65	70.4%	60.0%	53.7%	51.5%	49.1%	47.7%	46.5%	44.8%	43.4%	42.7%
70	70.1%	60.4%	54.7%	50.9%	49.5%	48.2%	47.7%	45.5%	43.6%	42.6%
75	68.8%	60.7%	54.9%	51.5%	49.5%	48.4%	46.1%	45.5%	44.3%	43.7%
80	70.0%	60.3%	54.2%	51.5%	50.3%	48.3%	47.0%	45.0%	44.4%	44.6%
85	70.8%	60.5%	55.1%	52.1%	50.4%	48.2%	46.6%	45.5%	44.7%	45.0%
90	70.2%	60.8%	55.4%	52.3%	50.9%	48.0%	47.2%	46.7%	45.5%	43.9%
95	71.2%	60.0%	55.5%	52.5%	51.0%	47.7%	47.5%	46.5%	45.9%	44.6%
100	70.4%	60.8%	57.2%	53.1%	50.8%	48.6%	47.1%	46.9%	45.9%	45.7%

The computational results for the latter problems are given in Table 4.14 which is analogue to Table 4.9.

Tables 4.14 - 4.18 give the percentage of (optimally) solved instances due to stability analysis based on the above computational scheme and the average running time in seconds for each series of instances (in parenthesis). Comparing Table 4.9 with Table 4.14 show that increasing simultaneously both numbers n and w increases the number of solved instances. The same issue follows from comparing Table 4.10 with Table 4.15, Table 4.11 with Table 4.16, Table 4.12 with Table 4.17, and Table 4.13 with Table 4.18.

Of course, the running time increases with increasing the product nw. Fortunately, the running time for the stability analysis via algorithm given at page 222 remains rather small even for large problem sizes. Also we can conclude that the order of considered cases (classes) of problems with respect to an increasing of their complexity in our experiments was as follows.

$$[(p_{i,2} = 2p_{i,1})\&(w = w_2)] \rightarrow [(w_1 > 0)\&(w_2 > 0)] \rightarrow [(w = w_2)] \rightarrow [(w = w_1)]$$

$$\rightarrow [(p_{i,2} = 2p_{i,1})\&(w_1 > 0)\&(w_2 > 0)] \rightarrow [(p_{i,2} = 2p_{i,1})\&(w = w_1)].$$

The hardest class of large problems was that with $p_{i,2} = 2p_{i,1}$ and $w = w_1$, and the easiest class of large problems was that with $p_{i,2} = 2p_{i,1}$ and $w = w_2$. All problems from the latter class were optimally solved (the table for this class is omitted). Note that the above order of problem classes had exceptions in our experiments, and it was more expressive for small problems. Within a problem class, the complexity of the instances usually increased with decreasing the difference $n - w$.

Remarks and Discussion

In this section, sufficient conditions are proven for Johnson's permutation to be optimal

Table 4.13: *Percentage of solved instances with* $p_{i,2} = 2p_{i,1}$ *and* $w = w_1$

Number of jobs	Number of non-availability intervals									
	1	2	3	4	5	6	7	8	9	10
5	27.4%	16.2%	9.5%	5.0%	3.0%	1.4%	0.7%	0.5%	0.2%	0.1%
10	33.0%	26.1%	20.8%	15.3%	11.4%	7.5%	5.3%	3.0%	1.9%	0.8%
15	35.0%	31.6%	27.8%	23.3%	19.6%	16.1%	13.8%	10.7%	8.2%	6.1%
20	36.2%	34.0%	29.9%	27.1%	26.6%	23.0%	20.3%	17.6%	15.2%	13.3%
25	36.9%	35.4%	32.5%	30.3%	28.2%	25.9%	24.2%	21.9%	19.6%	17.9%
30	38.4%	35.8%	33.7%	32.9%	30.2%	29.0%	26.6%	25.6%	23.8%	22.6%
35	38.4%	35.9%	35.6%	32.9%	31.9%	30.7%	30.0%	27.1%	26.2%	24.8%
40	38.9%	37.5%	35.6%	33.9%	33.0%	31.8%	30.7%	29.7%	28.7%	26.8%
45	38.9%	38.0%	36.1%	36.1%	34.0%	32.9%	32.9%	30.5%	29.9%	27.5%
50	39.5%	38.4%	36.5%	36.3%	35.1%	34.4%	33.1%	31.8%	30.7%	30.1%
55	39.5%	38.7%	37.5%	36.8%	36.0%	34.7%	33.8%	33.2%	33.1%	30.7%
60	40.7%	38.6%	37.9%	37.3%	36.8%	34.8%	33.8%	34.1%	32.8%	31.7%
65	40.7%	39.6%	38.8%	37.4%	36.9%	36.6%	35.7%	34.2%	33.3%	32.6%
70	39.8%	40.6%	39.2%	38.0%	37.7%	36.5%	36.3%	34.4%	34.6%	33.7%
75	41.2%	39.7%	39.3%	38.0%	38.4%	36.7%	36.6%	35.6%	35.0%	34.2%
80	40.6%	39.9%	40.4%	39.2%	38.7%	37.5%	36.6%	36.1%	36.1%	35.9%
85	41.5%	40.1%	40.2%	39.4%	38.0%	38.7%	37.4%	36.6%	35.5%	34.6%
90	41.2%	39.6%	40.4%	39.5%	39.4%	38.4%	37.2%	37.4%	37.1%	37.0%
95	41.6%	40.6%	40.2%	40.2%	38.9%	38.8%	37.9%	38.3%	35.8%	36.6%
100	41.3%	40.6%	40.0%	39.3%	39.3%	39.3%	38.2%	38.5%	37.8%	36.5%

Table 4.14: *Average running time and percentage of solved instances with* $w_1 > 0$ *and* $w_2 > 0$

Number of jobs	Number of non-availability intervals			
	10	100	500	1000
1000	99.9% (0.008 s)	100.0% (0.008 s)	99.9% (0.010 s)	99.8% (0.027 s)
2000	100.0% (0.029 s)	100.0% (0.029 s)	100.0% (0.031 s)	100.0% (0.037 s)
3000	100.0% (0.065 s)	100.0% (0.065 s)	99.8% (0.101 s)	100.0% (0.073 s)
4000	100.0% (0.116 s)	100.0% (0.116 s)	100.0% (0.118 s)	100.0% (0.123 s)
5000	100.0% (0.180 s)	100.0% (0.180 s)	100.0% (0.182 s)	100.0% (0.188 s)
6000	100.0% (0.259 s)	100.0% (0.259 s)	100.0% (0.261 s)	100.0% (0.266 s)
7000	100.0% (0.352 s)	100.0% (0.352 s)	100.0% (0.354 s)	100.0% (0.359 s)
8000	100.0% (0.459 s)	100.0% (0.459 s)	100.0% (0.461 s)	100.0% (0.466 s)
9000	100.0% (0.580 s)	100.0% (0.580 s)	100.0% (0.582 s)	100.0% (0.588 s)
10000	100.0% (0.715 s)	100.0% (0.716 s)	100.0% (0.717 s)	100.0% (0.723 s)

in the case of given non-availability intervals on machines M_1 and M_2 in the *two*-machine flow shop problem. Due to Theorem 4.9 and Lemmas 4.13 and 4.14, these conditions may be tested in polynomial time of the number n of jobs and the number w of non-availability intervals.

In Chapter 3, the notion of *relative stability radius* was used for a job shop problem. Such a relative stability radius may be used instead of a stability polytope defined in this section. It should be noted that the stability analysis may be used also for other scheduling problems with limited machine availability if an optimal schedule for the corresponding *pure setting* of the problem (i.e., when all machines are continuously available during the whole planning horizon) may be constructed applying a priority rule of jobs such as SPT, LPT and so on.

Moreover, one can use the above results for some kind of *on-line settings* of scheduling problems when there is no prior information about the exact location of the non-

Table 4.15: *Average running time and percentage of solved instances with* $w = w_2$

Number of jobs	Number of non-availability intervals			
	10	100	500	1000
1000	100.0% (0.008 s)	99.9% (0.008 s)	99.3% (0.020 s)	99.8% (0.073 s)
2000	100.0% (0.029 s)	100.0% (0.029 s)	100.0% (0.032 s)	100.0% (0.041 s)
3000	99.9% (0.067 s)	100.0% (0.066 s)	100.0% (0.068 s)	100.0% (0.077 s)
4000	100.0% (0.116 s)	100.0% (0.116 s)	100.0% (0.119 s)	100.0% (0.127 s)
5000	100.0% (0.180 s)	99.9% (0.187 s)	100.0% (0.183 s)	100.0% (0.191 s)
6000	100.0% (0.259 s)	100.0% (0.259 s)	100.0% (0.262 s)	100.0% (0.270 s)
7000	100.0% (0.352 s)	100.0% (0.352 s)	100.0% (0.354 s)	100.0% (0.363 s)
8000	100.0% (0.459 s)	100.0% (0.459 s)	100.0% (0.461 s)	100.0% (0.470 s)
9000	100.0% (0.580 s)	100.0% (0.580 s)	100.0% (0.583 s)	100.0% (0.591 s)
10000	100.0% (0.716 s)	100.0% (0.716 s)	100.0% (0.719 s)	100.0% (0.727 s)

Table 4.16: *Average running time and percentage of solved instances with* $w = w_1$

Number of jobs	Number of non-availability intervals			
	10	100	500	1000
1000	99.8% (0.008 s)	99.9% (0.008 s)	99.7% (0.017 s)	100.0% (0.019 s)
2000	99.9% (0.030 s)	99.8% (0.031 s)	99.8% (0.041 s)	100.0% (0.041 s)
3000	99.9% (0.070 s)	100.0% (0.065 s)	100.0% (0.068 s)	100.0% (0.077 s)
4000	100.0% (0.116 s)	100.0% (0.116 s)	100.0% (0.118 s)	100.0% (0.127 s)
5000	100.0% (0.180 s)	100.0% (0.180 s)	100.0% (0.183 s)	100.0% (0.191 s)
6000	100.0% (0.259 s)	100.0% (0.259 s)	100.0% (0.262 s)	100.0% (0.270 s)
7000	100.0% (0.351 s)	100.0% (0.352 s)	100.0% (0.354 s)	100.0% (0.363 s)
8000	100.0% (0.459 s)	100.0% (0.459 s)	100.0% (0.461 s)	100.0% (0.470 s)
9000	100.0% (0.580 s)	100.0% (0.580 s)	100.0% (0.583 s)	100.0% (0.591 s)
10000	100.0% (0.715 s)	100.0% (0.716 s)	100.0% (0.719 s)	100.0% (0.727 s)

availability intervals on the time axis but values d_{ij} or values $\delta_j,\ J_i \in J,\ j = \{1, 2\}$, are known before scheduling. In a straightforward matter it is possible to extend the results of this section to a *two*-machine n-job job shop problem (this is done in the next section).

The above computational results show that our stability analysis is efficient for small problems ($n \leq 100$ and $w \leq 10$) and especially for large problems ($1000 \leq n \leq 10000$ and $10 \leq w \leq 1000$). For the most classes of the randomly generated problems only a few instances were not optimally solved within a few seconds of running time. The only exception are the problems in which the processing times on machine M_1 are a double of the processing times on machine M_2, the number of jobs is less than the number of non-availability intervals, and machine M_2 is continuously available during the planning horizon. For such kind of problems $\mathcal{F}2{,}NC^{off}/pmtn/\mathcal{C}_{max}$ other sufficient conditions have to be derived.

4.4 Job Shop with Limited Machine Availability

This section deals with the scheduling problem of minimizing the length of a schedule (makespan) for processing n jobs by two machines with the given w non-availability intervals. Processing each job has no more than two stages (i.e., there are at most two operations per job), and technological routes through machines may differ from job to job (multi-stage system of a job shop type). If there is no non-availability interval ($w = 0$),

Table 4.17: *Average running time and percentage of solved instances with* $p_{i,2} = 2p_{i,1}$, $w_1 > 0$ *and* $w_2 > 0$

Number of jobs	Number of non-availability intervals			
	10	100	500	1000
1000	80.5% (0.032 s)	75.9% (0.036 s)	60.3% (0.175 s)	48.9% (1.329 s)
2000	95.4% (0.074 s)	92.3% (0.084 s)	85.1% (0.155 s)	73.5% (0.802 s)
3000	99.2% (0.137 s)	97.6% (0.147 s)	90.4% (0.233 s)	85.4% (0.602 s)
4000	99.3% (0.238 s)	98.9% (0.245 s)	93.4% (0.338 s)	88.0% (0.696 s)
5000	99.9% (0.359 s)	99.5% (0.368 s)	96.2% (0.454 s)	91.7% (0.752 s)
6000	100.0% (0.514 s)	99.6% (0.526 s)	95.6% (0.664 s)	91.3% (0.984 s)
7000	99.9% (0.703 s)	99.3% (0.728 s)	96.6% (0.853 s)	92.1% (1.221 s)
8000	100.0% (0.912 s)	99.6% (0.934 s)	97.1% (1.081 s)	93.4% (1.444 s)
9000	99.9% (1.168 s)	99.3% (1.210 s)	98.3% (1.286 s)	93.9% (1.740 s)
10000	100.0% (1.447 s)	99.7% (1.472 s)	97.4% (1.678 s)	93.0% (2.209 s)

Table 4.18: *Average running time and percentage of solved instances with* $p_{i,2} = 2p_{i,1}$ *and* $w = w_1$

Number of jobs	Number of non-availability intervals			
	10	100	500	1000
1000	79.4% (0.032 s)	69.6% (0.042 s)	37.4% (0.457 s)	0.3% (4.968 s)
2000	95.0% (0.075 s)	89.7% (0.093 s)	69.5% (0.352 s)	46.6% (2.856 s)
3000	99.1% (0.136 s)	96.2% (0.158 s)	83.5% (0.360 s)	66.1% (2.054 s)
4000	99.4% (0.237 s)	98.6% (0.248 s)	86.5% (0.498 s)	72.8% (1.937 s)
5000	99.7% (0.363 s)	98.3% (0.393 s)	89.0% (0.661 s)	79.6% (1.797 s)
6000	99.8% (0.520 s)	97.8% (0.581 s)	90.8% (0.854 s)	83.2% (1.858 s)
7000	99.9% (0.703 s)	98.1% (0.778 s)	92.7% (1.049 s)	83.0% (2.246 s)
8000	99.8% (0.923 s)	99.3% (0.950 s)	92.8% (1.350 s)	87.8% (2.181 s)
9000	100.0% (1.161 s)	99.1% (1.224 s)	93.5% (1.653 s)	89.0% (2.469 s)
10000	99.9% (1.455 s)	99.1% (1.524 s)	94.5% (1.956 s)	89.3% (2.899 s)

this problem may be polynomially solved using Jackson's pair of job permutations (see Section 4.2), otherwise it is binary NP-hard even if there is only one non-availability interval ($w = 1$), and all the jobs have the same machine route (flow shop). The latter problem becomes unary NP-hard if the number of non-availability intervals may be arbitrarily large. In practice, the limited machine availability may be coursed, e.g., by unfinished jobs from the previous schedule, machine breakdowns and machine maintenance, appearance unexpected job with high priority or with close due date, etc.

We find some sufficient conditions when Jackson's pair of permutations remains optimal for the two-machine job shop problem with the given $w > 1$ non-availability intervals. Extensive computational studies show the effectiveness (in the number of problems solved) and efficiency (in computational time) of these sufficient conditions for the randomly generated instances with $n \leq 10000$ jobs and $w \leq 1000$ non-availability intervals.

Definitions and Notations

Let set of jobs $J = \{J_1, J_2, \ldots, J_n\}$ have to be processed in a job shop with set $M = \{M_1, M_2\}$ of two machines. Each machine $M_j \in M$ can process any job $J_i \in J$ no more than once. Let $J(12) \subseteq J$ be set of jobs with machine route (M_1, M_2) (at first, job $J_i \in J(12)$ has to be processed by machine $M_1 \in M$, and then by machine $M_2 \in M$).

Let $J(21) \subseteq J$ be set of jobs with opposite machine route (M_2, M_1), and $J(k) \subseteq J$ a set of jobs which have to be processed by only one machine $M_k \in M$. Thus, we have $J = J(1) \cup J(2) \cup J(12) \cup J(21)$, and we set $n_l = |J_l|$ where $l \in \{1, 2, 12, 21\}$.

Let O_{ir} denote operation of job $J_i \in J$ on machine $M_{i_r} \in M$ at stage $r \in \{1, 2\}$ of the machine route (M_{i_1}, M_{i_2}), if $J_i \in J(12) \cup J(21)$, or of the machine route (M_{i_1}), if $J_i \in J(1) \cup J(2)$ and $r = 1$. Processing time p_{ii_r} of operation O_{ir} of job $J_i \in J$ on machine $M_{i_r} \in M$ is known before scheduling. All the n jobs are available at time $t = 0$. Criterion $\mathcal{C}_{max}$ under consideration denotes minimization of schedule length (makespan):

$$\mathcal{C}_{max} = \max_s C_{max}(s) = \max_s \{\max\{C_i(s) \ : \ J_i \in J\}\}$$

where $C_i(s)$ denotes completion time of job $J_i \in J$ at schedule s.

In this section, we assume that all of the w non-availability intervals are known in advance (before scheduling), and all the jobs are assumed to be *resumable*. In other words, if processing operation O_{ir} is preempted at the beginning of the unavailable interval of machine $M_{i_r} \in M$, then it can be resumed at the end of this interval without increasing its processing time p_{ii_r} given before scheduling. (Of course, the processing interval used for operation O_{ir} will be increased due to unavailable interval of machine M_{i_r}.) To indicate this we use notation *pmtn* since the latter assumption is equivalent to allowance of operation preemption. Using three-field notation, the problem under consideration may be denoted as $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$.

Let (π', π'') be a Jackson's pair of permutations (see Section 4.2) of the jobs of set J (here π' is a permutation of jobs $J(12) \cup J(21) \cup J(1)$ on machine M_1, and π'' is a permutation of jobs $J(12) \cup J(21) \cup J(2)$ on machine M_2) which is optimal when all machines are continuously available ($w = 0$) during the planning horizon (see Theorem 4.7 at page 206). The main aim of this section is to study the question whether this pair of permutations remains optimal for the case of offline known $w \geq 1$ non-availability intervals. To this end, we use the approach similar to that proposed in Section 4.3 for the case of a flow shop problem with limited machine availability.

First, we present some properties of optimal schedule and remind useful results for problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ proven in Section 4.3. Then we prove sufficient conditions for optimality of Jackson's pair of permutations in the case of the given $w > 1$ intervals of machine non-availability. The end of this section contains computational results for stability analysis of Jackson's pair of permutations.

We use the notations similar to that used in Section 4.3. For the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$, it is assumed that all the non-availability intervals are known before scheduling. Machine $M_j \in M$ is not available for processing any job from set J from starting point $s(N_{kj})$ until endpoint $f(N_{kj}) = s(N_{kj}) + h(N_{kj}),\ k \in \{1, 2, \ldots, w_j\}$. The operation O_{ii_r} started before time point $s(N_{kj})$ with $j = i_r$ but not finished until time point $s(N_{kj})$ is suspended during time period of length $h(N_{kj})$ starting from time point $s(N_{kj})$. Then processing operation O_{ii_r} is resumed from time point $f(N_{kj})$.

Since the minimization of completion time is a *regular* criterion we can consider only *semiactive schedules* (see Definition 1.1 at page 11). Each semiactive schedule is uniquely defined by permutation of jobs on machine M_1 and that on machine M_2. For the semiactive schedule s, let $c_1(s)$ and $c_2(s)$ denote completion time of all jobs by machine M_1 and machine M_2, respectively. Then for problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$, completion time of

schedule s (and so length or makespan of schedule s) may be defined as follows: $C_{max}(s) = \max\{c_1(s), c_2(s)\}$. We use the following definition of the main machine at schedule s.

Definition 4.6 *Machine $M_j \in M$ is called the main machine at schedule s if under this schedule the following equality holds: $C_{max}(s) = c_j(s)$.*

If equality $w = 0$ holds, then problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ turns into a classical job shop problem $\mathcal{J}2/pmtn/\mathcal{C}_{max}$ with two machines that is polynomially solvable. Note that using operation preemptions cannot decrease length of an optimal schedule. Thus, if $w = 0$, it is sufficient to construct optimal schedule for the problem $\mathcal{J}2//\mathcal{C}_{max}$ which is defined by the two permutations (sequences) (π', π'') where π' is a sequence of jobs $J(1) \cup J(12) \cup J(21)$ on machine M_1, and π'' is a sequence of jobs $J(2) \cup J(12) \cup J(21)$ on machine M_2 (see Theorem 4.7 at page 206).

We use notation $s(\pi', \pi'')$ for the semiactive schedule (uniquely) defined by permutation π' of jobs on machine M_1 and by permutation π'' of jobs on machine M_2. If schedule $s(\pi', \pi'')$ is optimal for the problem $\mathcal{J}2//\mathcal{C}_{max}$ (for the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$), then pair of permutations (π', π'') is called optimal for the problem $\mathcal{J}2//\mathcal{C}_{max}$ (for the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$, respectively). Note that here pair of permutations (π', π'') is the same for both problems $\mathcal{J}2//\mathcal{C}_{max}$ and $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$, while the schedules $s(\pi', \pi'')$ usually are different (in spite of the same notation $s(\pi', \pi'')$ used).

In the schedule $s(\pi', \pi'')$ that is optimal for problem $\mathcal{J}2//\mathcal{C}_{max}$, jobs of the set $J(12) \subset J$ (of the set $J(21) \subset J$) are processed with respect to Johnson's order of job processing times (see Johnson's rule at page 186 in Section 4.1). Therefore, it is sufficient to look for solution to the problem $\mathcal{J}2//\mathcal{C}_{max}$ using the set of pairs of permutations: $(\pi' = (\pi_{12}, \pi_1, \pi_{21}), \pi'' = (\pi_{21}, \pi_2, \pi_{12}))$ where permutations π_{12} and π_{21} are defined by Johnson's rule. Hereafter, job J_i belongs to permutation π_l if and only if $J_i \in J(l)$, $l \in \{1, 2, 12, 21\}$. Moreover, processing order of the jobs of set $J(1)$ (set $J(2)$) may be arbitrary in such an optimal schedule, therefore we fix both sequences π_1 and π_2 (e.g., in lexicographical order of jobs numbers). If in sequences π_{12} and π_{21} jobs are ordered due to Johnson's rule, then we say of Jackson's pair (π', π'') of permutations of $(n_{12} + n_{21} + n_1)$ jobs and of $(n_{12} + n_{21} + n_2)$ jobs, respectively (see Theorem 4.7 at page 206 in Section 4.2).

Because of operation preemption in optimal semiactive schedule s cannot decrease value $C_{max}(s$ of the objective function, optimal schedule defined by Jackson's pair of permutations for the instance of problem $\mathcal{J}2//\mathcal{C}_{max}$ remains optimal schedule for the instance of problem $\mathcal{J}2/pmtn/\mathcal{C}_{max}$ with the same input data.

Properties of an Optimal Schedule

Let (π', π'') be a Jackson's pair of permutations (which is optimal when both machines are continuously available during the planning horizon, i.e., for problem $\mathcal{J}2//\mathcal{C}_{max}$). Our aim is to answer the question when this pair of permutations remains optimal for the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ with the given $w \geq 1$ non-availability intervals and with operation processing times being the same as in the problem $\mathcal{J}2//\mathcal{C}_{max}$. In the proofs of the sufficient conditions, we will use following obvious claim.

Lemma 4.16 *If the main machine M_j at schedule $s(\pi', \pi'')$ is completely filled in the closed interval $[0, c_j(s)]$ by operations on machine M_j and by non-availability intervals of this machine (in this case, we say that machine M_j works without idles), then pair of permutations (π', π'') is optimal for the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$.*

It should be noted that in the above schedule $s(\pi', \pi'')$ pair of permutations (π', π'') may be not only a Jackson's pair of permutations for the problem $\mathcal{J}2//\mathcal{C}_{max}$. Nevertheless, when conditions of Lemma 4.16 hold for some pair of permutations (π', π''), then schedule $s(\pi', \pi'')$ would be optimal for the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ without fail. So, conditions of Lemma 4.16 are sufficient for optimality of the schedule $s(\pi', \pi'')$ for problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ regardless permutation pair (π', π'') are Jackson's pair of permutations or not.

Let us assume that for the Jackson's pair of permutations (π', π'') main machine for the schedule $s(\pi', \pi'')$ works with idle. We can assume that it is machine M_2 w.l.o.g., since in a two-machine job shop problem there is machine symmetry. Main machine M_2 processes jobs of the set $J(2) \cup J(12) \cup J(21)$ with respect to sequence $\pi'' = (\pi_{21}, \pi_2, \pi_{12})$. It is easy to see that all the jobs from sequences π_{21} and π_2 have to be processed by machine M_2 without idles at any semiactive schedule. Therefore, idle on machine M_2 may arise only due to operations of jobs from set $J(12)$. Hence, jobs of the set $J(12)$ define the makespan for the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$.

Therefore in what follows, we can concentrate mainly on the processing only this set $J(12)$ of jobs. Note that set $J(12)$ of jobs forms a special case of a flow shop problem generated by the original job shop problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ under consideration. Namely we will consider the following problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ (we call it as problem A) of processing n_{12} jobs of set $J(12)$ with the same machine route (M_1, M_2). Problem A has the same non-availability intervals on machines M_1 and includes all the non-availability intervals on machine M_2 as original job shop problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ has. Furthermore, problem A has one additional non-availability interval on machine M_2 as follows: $[0, c_2(\pi_{21}, \pi_2))$.

Thus, job shop problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ under consideration may be reduced to flow shop problem A in the sense as problem $\mathcal{J}2//\mathcal{C}_{max}$ was reduced to problem $\mathcal{F}2//\mathcal{C}_{max}$ in [Jac56]. Therefore, a lot of results obtained for problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ so far may be transformed to that for problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$. Next, we demonstrate how to use the results for the flow shop problem presented in Section 4.3 for solving problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$.

If jobs of the set $J(12)$ are processed by machines M_1 and M_2 with respect to the same sequence π_{12} which is a Johnson's permutation of these jobs, then the schedule defined by permutation π_{12} is optimal for the problem $\mathcal{F}2//\mathcal{C}_{max}$. The next our question is either permutation π_{12} is optimal for the new problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ or not.

Section 4.3 contains sufficient conditions for a Johnson's permutation π_{12} for problem $\mathcal{F}2//\mathcal{C}_{max}$ to be optimal permutation for problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ as well. More precisely, to answer the above question we can test sufficient conditions for stability of optimal permutation $\sigma = \pi_v$ given in Theorem 4.10, Corollary 4.1 and Corollary 4.2 (see Section 4.3, pages 219, 220 and 220, respectively). Let at least one of these sufficient conditions hold for permutation $\sigma = \pi_{12}$ of the jobs from set $J(12)$ for flow shop problem

A. Then this set $J(12)$ of jobs defines minimum makespan value for the original job shop problem $\mathcal{J}2{,}NC^{off}/pmtn/\mathcal{C}_{max}$. Hence, due to these conditions, pair of permutations $s(\pi', \pi'')$ remains optimal for the problem $\mathcal{F}2{,}NC^{off}/pmtn/\mathcal{C}_{max}$.

Next, we prove that in the semiactive schedule $s(\pi', \pi'')$ at least one machine has to work without idles, i.e., idle times (do not caused by non-availability intervals) are only possible on one of the two machines of set M in the semiactive schedule $s(\pi', \pi'')$.

W.l.o.g., we can assume that machine M_2 works with idles in the schedule $s(\pi', \pi'')$, and let $t = s(N_{1,2})$ be starting point of the first such idle time of machine M_2, i.e., the first idle time (do not caused by non-availability intervals) in the order from left to right at the numerical axis. Obviously, such an idle time of machine M_2 is only possible if some job J_i from set $J(12)$ is still in process by machine M_1, while this job J_i could be processed by machine M_2 at the same time (i.e., machine M_2 has already completed all the jobs of sets $J(21)$ and $J(2)$, and machine M_2 is available at this time). Since schedule $s(\pi', \pi'')$ is defined by the pair of permutations (π', π'') where $\pi' = (\pi_{12}, \pi_1, \pi_{21})$, the following inequality must hold:

$$c_2(\pi_{21}, \pi_2) < c_1(\pi_{12}). \tag{4.37}$$

Analogously, such an idle time of machine M_1 may arise only due to some jobs from set $J(21)$ being still in process by machine M_2, while they could be processed by machine M_1 (i.e., machine M_1 has already completed all the jobs of sets $J(12)$ and $J(1)$, and machine M_1 is available).

Since inequality (4.37) holds, machine M_1 can freely processed jobs of set $J(21)$, and therefore, machine M_1 has to work without idles. So, the following claim is proved.

Lemma 4.17 *Under schedule $s(\pi', \pi'')$, at least one of the two machines in set M works without idles.*

Lemma 4.17 implies the following claim.

Corollary 4.3 *If under schedule $s(\pi', \pi'')$ equality $c_1(s) = c_2(s)$ holds, then pair of permutations (π', π'') is optimal for the problem $\mathcal{J}2{,}NC^{off}/pmtn/\mathcal{C}_{max}$.*

Indeed, since at least one machine works without idles (due to Lemma 4.17), then due to equality $c_1(s) = c_2(s)$ it is impossible to decrease makespan $C_{max}(s(\pi', \pi''))$.

Sufficient Conditions for Optimality of a Permutation Pair

We will say that problem $\mathcal{J}2//\mathcal{C}_{max}$ *corresponds* to problem $\mathcal{J}2{,}NC^{off}/pmtn/\mathcal{C}_{max}$ (and wise versa), if equality $w = 0$ holds for the former problem, inequality $w > 0$ holds for the latter problem, and all the other parameters and conditions are the same for both problems. We want to answer the following question. When pair of Jackson's permutations (π', π'') constructed for problem $\mathcal{J}2//\mathcal{C}_{max}$ would be optimal for the corresponding problem $\mathcal{J}2{,}NC^{off}/pmtn/\mathcal{C}_{max}$?

Let (π', π'') be Jackson's pair of permutations constructed for the problem $\mathcal{J}2//\mathcal{C}_{max}$. Then w.l.o.g., we can assume that machine M_2 is the main machine in the schedule $s(\pi', \pi'')$ for the problem $\mathcal{J}2//\mathcal{C}_{max}$. In the following claim the same notation $s(\pi', \pi'')$ used twice but in different sense: At first, it is a schedule for problem $\mathcal{J}2{,}NC^{off}/pmtn/\mathcal{C}_{max}$, and then it is a schedule for problem $\mathcal{J}2//\mathcal{C}_{max}$.

Theorem 4.11 *Let (π', π'') be Jackson's pair of permutations constructed for problem $\mathcal{J}2//\mathcal{C}_{max}$. The schedule $s(\pi', \pi'')$ is optimal for the corresponding problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$, if for the corresponding problem $\mathcal{J}2//\mathcal{C}_{max}$ main machine M_2 works without idles at schedule $s(\pi', \pi'')$, and the following two conditions hold:*

$$\sum_{k=1}^{w_1} h(N_{k1}) \leq c_2(\pi'') - c_1(\pi'), \tag{4.38}$$

$$s(N_{1,1}) \geq \sum_{J_i \in J(12)} p_{i1}. \tag{4.39}$$

Proof. We will show that due to the above conditions, machine M_2 remains the main machine for the corresponding problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ and works without idles. As a result, conditions of Lemma 4.16 will hold, and so schedule $s(\pi', \pi'')$ will be optimal for the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ as well.

Indeed, from inequality (4.39) we obtain that non-availability intervals of machine M_1 cannot increase idle time of machine M_2. Therefore, machine M_2 is completely filled in the closed interval $[0, c'_2(\pi'')]$ by processing times of jobs from permutation π'' and by non-availability intervals of machine M_2. Hence, machine M_2 works without idles, and so the following equality holds:

$$c'_2(\pi'') = c_2(\pi'') + \sum_{k=1}^{w_2} h(N_{k2}).$$

Next, we prove that machine M_2 remains the main machine for the corresponding problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$. Note, that non-availability intervals of machine M_2 can increase idle time of machine M_1. This additional idle time (IT) of machine M_1 (if it exists) is not greater than sum of the lengths of the non-availability intervals of the machine M_2:

$$IT \leq \sum_{i=1}^{w_2} h(N_{i2}).$$

Therefore, endpoint of all the jobs from permutation π' on machine M_1 cannot increase more than by the sum of the lengths of all the non-availability intervals:

$$c'_1(\pi') \leq c_1(\pi') + IT + \sum_{l=1}^{w_1} h(N_{l1}) \leq c_1(\pi') + \sum_{l=1}^{w_1} h(N_{l1}) + \sum_{i=1}^{w_2} h(N_{i2}).$$

We obtain

$$c'_2(\pi'') = c_2(\pi'') + \sum_{k=1}^{w_2} h(N_{k2}) = c_2(\pi'') - c_1(\pi') + c_1(\pi') + \sum_{k=1}^{w_2} h(N_{k2})$$

$$\geq \sum_{l=1}^{w_1} h(N_{l1}) + c_1(\pi') + \sum_{k=1}^{w_2} h(N_{k2}) \geq c'_1(\pi').$$

Therefore, machine M_2 remains the main machine for the corresponding problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$.

Thus, conditions of Lemma 4.16 hold, and therefore, schedule $s(\pi', \pi'')$ is optimal for the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$. This completes the proof.

$\diamond$

Next, we show that all the above conditions of Theorem 4.11 are essential. In other words, if at least one condition of Theorem 4.11 does not hold, then Jackson's pair of permutations (π', π'') mentioned in Theorem 4.11 may be not optimal for the corresponding problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$. We consider the following three examples (one example for each condition of Theorem 4.11 which is violated).

Example 4.6 *Let us consider problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ with the processing times of $n = 4$ jobs given in Table 4.19 where subset of jobs $J(kl)$ is indicated in the first column such that inclusion $J_i \in J(kl)$ holds with $\{k, l\} = \{1, 2\}$. Let only one non-availability interval $(5, 7)$ be given on machine M_2.*

Table 4.19: *Processing times for Example 4.6*

Job J_i	i	p_{i1}	p_{i2}
$J(12)$	1	5	4
	2	4	3
$J(21)$	3	1	2
	4	1	1

The following condition of Theorem 4.11 does not hold: The main machine M_2 works with idle for the corresponding problem $\mathcal{J}2//\mathcal{C}_{max}$. Jackson's pair of permutations for the corresponding problem $\mathcal{J}2//\mathcal{C}_{max}$ is $((J_1, J_2, J_4, J_3), (J_4, J_3, J_1, J_2))$, and machine M_2 works with idle from time point 3 to time point 5.

It is easy to see that for the original problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ pair of permutations $((J_1, J_2, J_4, J_3), (J_4, J_3, J_1, J_2))$ is not optimal. Namely, for the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$, optimal pair of permutations is $((J_2, J_1, J_4, J_3), (J_4, J_3, J_2, J_1))$ (by the way, it is not a Jackson's pair of permutations for the corresponding problem $\mathcal{J}2//\mathcal{C}_{max}$).

Let now inequality (4.38) of Theorem 4.11 be violated, i.e., let the sum of the non-availability intervals on machine M_2 be more than difference $c_2(\pi'') - c_1(\pi')$. Then an example exists (see the following Example 4.7) with optimal schedule defined by pair of permutations which is certainly not Jackson's pair of permutations.

Example 4.7 *Let job processing times for this example be given in Table 4.20, and let one non-availability interval (5, 7) be given on machine M_1.*

Jackson's pair of permutations for the corresponding problem $\mathcal{J}2//\mathcal{C}_{max}$ is $((J_1, J_2, J_3, J_4), (J_3, J_4, J_1, J_2))$. Here, we have $c_1(\pi') = 12$, $c_2(\pi'') = 13$, and so

$$c_2(\pi'') - c_1(\pi') = 1 < \sum_{i=1}^{w_1=1} h(N_{i1}) = h(N_{1,1}) = 2.$$

It is easy to see that for the original problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ pair of permutations $((J_1, J_2, J_3, J_4), (J_3, J_4, J_1, J_2))$ is not optimal. (For the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ optimal pair of permutations is $((J_1, J_2, J_4, J_3), (J_4, J_3, J_1, J_2))$ which is certainly not a Jackson's pair of permutations.)

Table 4.20: *Processing times for Example 4.7*

Job J_i	i	p_{i1}	p_{i2}
$J(12)$	1	2	2
	2	1	2
$J(21)$	3	4	5
	4	3	4

Example 4.8 *Let now inequality (4.39) of Theorem 4.11 be violated, i.e., starting point of the first non-availability interval of machine M_1 is less than endpoint of the processing all the jobs of set $J(12)$ by machine M_1.*

Let processing times of jobs be given in Table 4.21. We consider two non-availability intervals: The first non-availability interval (2, 3) on machine M_1, and the second non-availability interval (4, 7) on machine M_2.

Table 4.21: *Processing times for Example 4.8*

Job J_i	i	p_{i1}	p_{i2}
J_{12}	1	3	2
	2	2	1
J_{21}	3	3	1
	4	2	1

Optimal Jackson's pair of permutations for corresponding problem $\mathcal{J}2//\mathcal{C}_{max}$ is $((J_1, J_2, J_4, J_3), (J_4, J_3, J_1, J_2))$. Here, we have $c_1(\pi_{12}) = 5$ while $s(N_{1,1}) = 2$, and so start of the first non-availability interval of machine M_1 is less than endpoint of the processing all the jobs of set $J(12)$ by machine M_1. For the original problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ pair of permutations $((J_1, J_2, J_4, J_3), (J_4, J_3, J_1, J_2))$ is not optimal.

For problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ optimal pair of permutations is $((J_2, J_1, J_4, J_3), (J_4, J_3, J_2, J_1))$ which is not Jackson's pair of permutations.

In the following claim the same notation $s(\pi', \pi'')$ used again twice but in different sense: At first, it is schedule for problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$, and then it is schedule for problem $\mathcal{J}2//\mathcal{C}_{max}$.

Theorem 4.12 *Let (π', π'') be Jackson's pair of permutations for the problem $\mathcal{J}2//\mathcal{C}_{max}$. Then schedule $s(\pi', \pi'')$ is optimal for the corresponding problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ if for the corresponding problem $\mathcal{J}2//\mathcal{C}_{max}$ main machine M_2 works without idles at schedule $s(\pi', \pi'')$, and the following three conditions hold:*

$$\sum_{k=1}^{w_1} h(N_{k1}) \geq \sum_{k=1}^{w_2} h(N_{k2}), \tag{4.40}$$

$$s(N_{1,2}) \geq \sum_{J_i \in J(12)} p_{i1}, \tag{4.41}$$

$$s(N_{1,1}) \geq \sum_{J_i \in J(21)} p_{i2}. \tag{4.42}$$

PROOF. Next, we show that if all the above conditions hold in common, then machine M_2 remains the main machine for the corresponding problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ and works without idles. Therefore, conditions of Lemma 4.16 hold, and therefore schedule $s(\pi', \pi'')$ has to be optimal for the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$.

Indeed, inequality (4.41) implies that non-availability intervals of machine M_1 cannot increase idle time on machine M_2. Hence, machine M_2 is completely filled in the closed interval $[0, c'_2(\pi'')]$ with processing times of all jobs from permutation π'' and non-availability intervals of this machine. So, machine M_2 works without idles and we obtain equality

$$c'_2(\pi'') = c_2(\pi'') + \sum_{k=1}^{w_2} h(N_{k2}).$$

Now, we have to show that machine M_2 remains the main machine for the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$. Inequality (4.42) implies that non-availability intervals on machine M_2 cannot increase idle time on machine M_1. Therefore, endpoint on machine M_1 cannot be enlarged more than by the sum of the lengths of all the non-availability intervals on this machine, i.e., we have

$$c'_1(\pi') \leq c_1(\pi') + \sum_{k=1}^{w_1} h(N_{k1}).$$

We obtain

$$c'_1(\pi') \leq c_1(\pi') + \sum_{k=1}^{w_1} h(N_{k1}) \leq c_2(\pi'') + \sum_{k=1}^{w_2} h(N_{k2}) = c'_2(\pi'').$$

Therefore, machine M_2 is the main machine for the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$.

Thus, the conditions of Lemma 4.16 hold, and hence schedule $s(\pi', \pi'')$ is optimal for the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$. This completes the proof.

◇

Computational Results

Our computational study of the above sufficient conditions was performed on a huge number of randomly generated problems $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$. For each randomly generated instance $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ we constructed a Jackson's pair of permutations for the corresponding job shop problem $\mathcal{J}2//\mathcal{C}_{max}$ (i.e., that with the same job processing times) and answered the question: Is this pair of permutations optimal for the original problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$? To this end, we tested sufficient conditions proven in Section 4.3 for problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ and that proven in this section for problem

$\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$. To minimize running time of the software, these sufficient conditions were tested in an increasing order of their complexity up to the first positive answer (if any) to the above question. More formally, the following algorithm was realized.

Algorithm for $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$

Input: Processing times for jobs J and non-availability intervals of machines M.
Output: Optimal schedule for the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ or feasible schedule if algorithm terminates at *Step 6*.

Step 1: Construct Jackson's pair of permutations (π', π'') for problem $\mathcal{J}2//\mathcal{C}_{max}$ and Johnson's permutation σ for flow shop problem A.

Step 2: **IF** schedule $s(\sigma) = s(\pi_{12})$ for problem A satisfies Corollary 4.1 **THEN GOTO** *Step 7*.

Step 3: **IF** schedule $s(\sigma) = s(\pi_{12})$ for problem A satisfies Corollary 4.2 **THEN GOTO** *Step 7*.

Step 4: Construct $\lambda = \min\{\lambda^*, 2^k\}$ Johnson's permutations $\sigma_1, \sigma_2, \ldots, \sigma_\lambda$ for problem A, and λ Jackson's pairs of permutations $(\pi'_1, \pi''_1), (\pi'_2, \pi''_2), \ldots, (\pi'_\lambda, \pi''_\lambda)$ for problem $\mathcal{J}2//\mathcal{C}_{max}$.

Step 5: **FOR** $i = 1, 2, \ldots, \lambda$ **DO**
BEGIN
Set $\sigma := \sigma_i$.
IF schedule $s(\sigma)$ for problem A satisfies Corollary 4.1
THEN GOTO *Step 7*.
IF schedule $s(\sigma)$ for problem A satisfies Corollary 4.2
THEN GOTO *Step 7*.
IF schedule $s(\pi', \pi'')$ satisfies Theorem 4.11 **THEN GOTO** *Step 7*.
IF schedule $s(\pi', \pi'')$ satisfies Theorem 4.12 **THEN GOTO** *Step 7*.
END

Step 6: Optimality of pairs of permutations (π'_i, π''_i), $i = 1, 2, \ldots, \lambda$, for the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ is not proven **STOP**.

Step 7: Pair of permutation (π', π'') is optimal for the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ **STOP**.

Steps 4 and 5 of the above algorithm are realized since there may be more than one Jackson's pair of permutation (if there are equalities $p_{ij} = p_{kj}$ for different jobs $J_i \in J$ and $J_k \in J$). We tested up to $\min\{\lambda^*, 2^k\}$ pairs of Jackson's permutations for each instance. In the experiments, we set $\lambda^* = 1024$.

For each combination of n and w, Tables 4.22 - 4.33 present the percentage of instances

in the series for which calculation based on the above algorithm found Jackson's pair of permutations which remains optimal in spite of the given non-availability intervals.

Table 4.22: *Percentage of solved small and moderate (easy) instances with* $w_1 > 0$ *and* $w_2 > 0$

Number of jobs	Number of non-availability intervals									
	1	2	3	4	5	6	7	8	9	10
5	86.2%	85.5%	87.8%	85.9%	85.7%	85.2%	85.0%	83.2%	83.4%	85.8%
10	92.1%	93.5%	93.3%	92.4%	92.8%	92.1%	92.6%	93.2%	93.0%	94.3%
15	95.0%	95.1%	95.7%	95.8%	96.2%	96.5%	95.7%	96.2%	95.2%	95.4%
20	97.3%	97.4%	97.2%	97.9%	97.6%	97.5%	96.7%	97.5%	96.4%	97.4%
25	97.7%	98.2%	98.5%	97.8%	99.1%	97.5%	98.5%	98.7%	98.6%	97.6%
30	99.1%	99.2%	98.9%	99.3%	99.0%	98.9%	99.0%	98.7%	98.5%	99.4%
35	99.3%	99.0%	99.4%	99.3%	98.6%	98.7%	99.6%	99.7%	99.7%	99.5%
40	99.6%	99.1%	99.2%	99.4%	99.0%	99.8%	99.2%	99.5%	99.4%	99.5%
45	99.1%	99.2%	99.3%	99.5%	99.8%	98.8%	99.6%	99.4%	98.7%	99.7%
50	99.3%	99.1%	99.7%	99.2%	99.6%	99.8%	99.9%	100.0%	99.7%	99.6%
55	99.6%	99.7%	99.5%	99.8%	99.8%	99.6%	99.8%	99.5%	99.4%	99.4%
60	99.7%	99.8%	99.9%	99.8%	99.9%	99.9%	99.7%	100.0%	99.8%	99.7%
65	99.6%	99.6%	99.8%	99.7%	99.9%	99.9%	99.7%	99.9%	99.6%	99.6%
70	99.7%	99.8%	99.7%	99.9%	99.6%	99.7%	99.8%	99.6%	100.0%	100.0%
75	100.0%	99.9%	100.0%	99.9%	99.8%	99.7%	99.6%	99.9%	100.0%	99.9%
80	99.8%	100.0%	99.9%	99.7%	99.9%	100.0%	99.9%	99.8%	99.9%	99.8%
85	99.9%	100.0%	99.9%	100.0%	99.4%	99.9%	100.0%	100.0%	99.8%	99.9%
90	100.0%	99.9%	99.8%	99.9%	99.9%	99.9%	99.8%	99.9%	100.0%	99.5%
95	100.0%	99.9%	99.9%	100.0%	99.9%	99.7%	100.0%	99.9%	100.0%	100.0%
100	99.7%	99.8%	99.8%	99.9%	99.9%	99.9%	100.0%	100.0%	99.9%	99.9%

Small and Moderate Problems

First, we tested problems similar to those considered in [KBF$^+$02, BLSS02], i.e., with

- $5, 10, 15, \ldots, 100$ jobs with integer processing times uniformly distributed in the range $[1, 1000]$ and with
- $1, 2, 3, \ldots, 10$ non-availability intervals (on both machines) with integer lengths uniformly distributed in the range $[1, 1000]$.

The above algorithm was coded in C++. For the computational experiments, we used an AMD 1200 MHz processor with 1024 MB main memory. For the *small* and *moderate* instances similar to those considered in [KBF$^+$02] and in [BLSS02], we made 10000 tests in each series (i.e., for each combination of n and w). For the *large* instances, we made 1000 tests in each series.

We answered the question of how many schedules following Jackson's order (constructed for problem $\mathcal{J}2//\mathcal{C}_{max}$) remained optimal in spite of the non-availability intervals on the machines (i.e., for problem $\mathcal{J}2,NC^{\mathit{off}}/pmtn/\mathcal{C}_{max}$). We tested only sufficient conditions and our algorithm does not guarantee to find an optimal schedule for some problems $\mathcal{J}2,NC^{\mathit{off}}/pmtn/\mathcal{C}_{max}$ (however, our experiments showed that such instances arise rather seldom for the most randomly generated problems). We considered 10000 instances in each series of small problems.

Table 4.23: *Percentage of solved small and moderate (hard) instances with* $w_1 > 0$ *and* $w_2 > 0$

Number of jobs	Number of non-availability intervals									
	1	2	3	4	5	6	7	8	9	10
5	64.9%	68.2%	69.0%	69.1%	70.0%	70.2%	69.6%	65.7%	67.7%	66.3%
10	81.3%	80.4%	80.8%	81.0%	81.8%	79.0%	80.4%	81.7%	81.5%	82.4%
15	87.9%	88.8%	87.8%	89.0%	88.2%	88.6%	89.1%	89.8%	88.6%	88.8%
20	91.1%	92.1%	92.3%	93.0%	91.0%	91.9%	92.2%	92.1%	90.3%	91.2%
25	95.3%	94.7%	93.9%	94.9%	92.5%	94.1%	95.3%	92.9%	94.0%	93.3%
30	94.8%	94.9%	95.7%	96.3%	96.1%	96.2%	95.2%	95.7%	96.0%	94.9%
35	96.4%	97.0%	97.4%	95.9%	95.9%	96.6%	96.9%	95.3%	95.4%	96.6%
40	96.6%	97.5%	96.6%	96.9%	97.4%	96.1%	95.0%	95.9%	96.5%	97.1%
45	97.8%	97.4%	98.1%	97.5%	96.3%	97.4%	97.4%	97.9%	96.9%	97.2%
50	98.4%	98.7%	97.7%	97.0%	97.7%	97.9%	97.2%	97.9%	97.5%	97.7%
55	97.8%	98.0%	97.6%	97.0%	98.0%	97.8%	97.9%	98.4%	97.6%	97.7%
60	98.1%	97.1%	97.9%	98.2%	98.7%	98.2%	97.7%	98.1%	98.5%	98.3%
65	98.4%	98.3%	98.2%	98.5%	97.7%	99.1%	97.8%	98.7%	98.3%	98.4%
70	98.1%	98.3%	98.9%	98.4%	98.8%	98.7%	97.6%	98.3%	98.0%	98.2%
75	97.5%	98.4%	98.3%	98.5%	98.9%	98.4%	98.7%	98.3%	98.4%	98.6%
80	99.1%	98.3%	98.2%	98.4%	98.3%	98.2%	98.7%	98.6%	99.3%	98.6%
85	98.5%	98.7%	98.7%	98.4%	98.7%	99.0%	98.8%	98.7%	98.5%	98.3%
90	99.0%	98.6%	98.8%	98.9%	98.6%	98.6%	99.0%	98.8%	98.7%	97.9%
95	98.5%	98.8%	99.0%	98.9%	98.8%	99.2%	98.9%	99.2%	98.9%	98.4%
100	98.4%	98.7%	98.5%	98.6%	99.4%	99.1%	98.7%	98.5%	98.4%	98.7%

Tables 4.22 - 4.27 present the percentage of problem instances which were (optimally) solved due to the stability analysis via the above computational scheme. We do not present running times for series of small instances, since the running times were very close for different instances and very small: The *maximum* running time of an instance in the series presented in the Tables 4.22 - 4.27 was 0.001 seconds.

Along with the case when non-availability intervals are on both machines: $w_1 > 0$ and $w_2 > 0$, we tested the cases (classes) of problems when either $w_1 = 0$ or $w_2 = 0$.

Large Problems

Due to the small running time, we were able to investigate very large instances (as in [BLSS02]. We tested instances of the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ with

- $1000, 2000, \ldots, 10000$ jobs with integer processing times uniformly distributed in the range $[1, 1000]$ and with
- $10, 100, 500, 1000$ non-availability intervals with integer lengths uniformly distributed in the range $[1, 1000]$ on both machines.

The computational results for the large instances of the problem $\mathcal{J}2,NC^{off}/pmtn/\mathcal{C}_{max}$ are given in Tables 4.29 - 4.33 which are analogous to Tables 4.22 - 4.27.

Tables 4.29 - 4.33 give the percentage of (optimally) solved instances due to stability analysis based on the above computational scheme and the average running time in seconds for each series of instances (in parenthesis). The maximum running time for one instance was 0.01000 second.

Table 4.24: *Percentage of solved small and moderate (easy) instances with $w = w_1$*

Number of jobs	Number of non-availability intervals									
	1	2	3	4	5	6	7	8	9	10
5	89.3%	89.7%	88.9%	90.6%	89.6%	89.5%	89.9%	89.9%	88.9%	90.0%
10	96.5%	96.0%	95.0%	96.8%	96.6%	95.8%	95.8%	96.1%	96.6%	95.3%
15	97.5%	98.7%	97.6%	97.3%	97.5%	98.5%	98.4%	98.8%	98.2%	97.4%
20	98.4%	98.6%	99.0%	98.6%	98.7%	99.1%	98.3%	98.6%	98.2%	98.4%
25	98.8%	99.4%	99.5%	99.5%	98.7%	98.7%	98.7%	99.3%	99.2%	99.3%
30	99.3%	99.6%	99.9%	99.1%	99.1%	99.1%	99.6%	99.3%	99.3%	99.3%
35	99.5%	99.4%	99.2%	99.6%	99.7%	99.8%	99.7%	99.6%	99.5%	99.5%
40	99.3%	99.6%	99.6%	99.8%	99.8%	99.4%	99.6%	99.6%	100.0%	98.9%
45	99.7%	99.7%	99.7%	99.7%	99.1%	100.0%	99.7%	99.7%	99.8%	99.8%
50	99.6%	99.9%	99.8%	99.8%	100.0%	99.8%	99.9%	99.7%	100.0%	99.8%
55	99.7%	99.7%	99.8%	99.7%	99.9%	99.8%	99.9%	99.7%	99.6%	99.6%
60	99.7%	99.7%	99.9%	99.8%	99.8%	99.8%	99.9%	99.6%	99.7%	99.8%
65	99.8%	99.9%	99.9%	99.9%	99.8%	99.8%	100.0%	99.9%	100.0%	99.9%
70	99.7%	99.5%	99.8%	99.7%	100.0%	99.9%	99.9%	99.8%	99.9%	99.9%
75	100.0%	99.4%	100.0%	99.9%	99.7%	99.7%	100.0%	99.8%	99.8%	100.0%
80	99.9%	99.9%	100.0%	100.0%	100.0%	99.9%	99.8%	99.9%	100.0%	100.0%
85	100.0%	99.9%	99.9%	99.8%	100.0%	99.8%	100.0%	99.8%	99.5%	99.9%
90	99.9%	99.7%	99.8%	99.8%	99.8%	99.7%	99.9%	100.0%	99.9%	100.0%
95	99.8%	100.0%	99.7%	99.8%	99.8%	99.9%	100.0%	100.0%	99.9%	100.0%
100	99.9%	100.0%	99.8%	99.8%	99.8%	99.8%	99.9%	100.0%	99.8%	100.0%

In this section, sufficient conditions are proven for Jackson's pair of permutations to be optimal in the case of the given w non-availability intervals on machines M_1 and M_2 in the two-machine job shop. Due to Theorems 4.11 and 4.12, Corollaries 4.1 and 4.2 (see Section 4.3), these conditions may be tested in polynomial time of the number n of jobs and the number w of non-availability intervals.

However, there are instances of the problems that cannot be solved exactly using these sufficient conditions. And it is not surprising since the problem is NP-hard even for $w = 1$ and single machine route. But it is worth noticing that the number of unsolved instances decreases with the increase of the number n of jobs. Also within the huge computational study, we did not find the type of combination of numbers n and w and type of the relation between the lengths of non-availability intervals on machine M_1 and machine M_2 for which our sufficient conditions in common give bad results. Note that such types of bad instances exist for a flow shop problems considered in Section 4.3.

The above computational results show that our sufficient conditions are very efficient in computational time and effective (in the number of problem solved) for small problems ($n \leq 100$ and $w \leq 10$) and especially for large problems ($1000 \leq n \leq 10000$ and $10 \leq w \leq 1000$). For most classes of the randomly generated problems, only a few instances were not optimally solved within a few seconds of running time.

In Section 4.3, analogous sufficient conditions have been proven and used for a special case of the problem under consideration, namely for the case of flow shop problem. Such sufficient conditions may also be used for other scheduling problems with limited machine availability if an optimal schedule for the corresponding pure setting of the problem (i.e., when all machines are continuously available during the whole planning horizon) can be constructed applying a priority rule to jobs such as SPT, LPT and so on. Moreover, one can use some of the above results for some kind of online settings of scheduling problems when there is no prior information about the exact location of the non-availability intervals

Table 4.25: *Percentage of solved small and moderate (hard) instances with* $w = w_1$

Number of jobs	Number of non-availability intervals									
	1	2	3	4	5	6	7	8	9	10
5	69.3%	65.7%	68.3%	65.6%	69.7%	69.8%	69.0%	70.8%	68.9%	70.5%
10	79.4%	79.2%	79.2%	79.4%	79.6%	79.6%	79.3%	77.9%	79.5%	79.1%
15	84.8%	86.3%	84.8%	85.3%	85.0%	83.6%	86.2%	83.4%	84.5%	85.4%
20	90.6%	88.9%	90.3%	89.4%	91.7%	89.5%	90.6%	88.9%	90.3%	90.3%
25	91.9%	95.1%	93.5%	92.0%	93.2%	93.2%	93.0%	93.4%	92.1%	92.2%
30	95.1%	96.2%	95.4%	95.0%	94.8%	95.8%	94.9%	93.8%	95.5%	95.1%
35	96.2%	96.6%	96.4%	96.3%	95.4%	95.7%	96.2%	96.6%	95.9%	95.6%
40	97.1%	95.9%	96.1%	96.8%	95.7%	97.2%	96.4%	96.2%	96.5%	97.6%
45	96.0%	97.1%	97.4%	97.9%	97.3%	97.6%	97.1%	98.1%	97.3%	96.5%
50	97.0%	97.2%	96.9%	97.4%	98.5%	98.2%	98.6%	97.5%	97.0%	97.0%
55	98.0%	98.1%	97.6%	97.8%	97.9%	98.2%	97.8%	97.6%	97.9%	96.8%
60	97.9%	97.9%	98.1%	97.8%	98.0%	97.7%	97.4%	98.6%	97.7%	97.8%
65	98.7%	98.8%	97.8%	98.4%	98.0%	98.1%	98.1%	98.9%	98.8%	98.5%
70	98.5%	98.2%	97.5%	98.3%	98.8%	98.7%	99.0%	98.7%	98.8%	98.5%
75	98.9%	99.1%	98.2%	98.7%	98.8%	98.6%	98.6%	98.1%	98.8%	98.7%
80	98.9%	98.9%	99.0%	99.3%	98.7%	98.9%	98.4%	98.6%	98.3%	98.3%
85	98.8%	98.9%	98.8%	98.2%	98.2%	98.7%	98.6%	98.6%	98.7%	98.6%
90	99.2%	98.8%	99.2%	99.1%	98.8%	99.1%	98.5%	98.5%	98.9%	98.9%
95	98.9%	99.1%	98.3%	99.2%	98.8%	98.6%	98.6%	99.1%	98.8%	99.0%
100	98.7%	99.2%	99.2%	99.0%	99.0%	98.9%	98.7%	98.9%	98.7%	99.1%

on the time axis. The other topic for future research may be connected with using the above sufficient conditions in the framework of an exact solution algorithm like a branch-and-bound one.

4.5 Flow Shop with Bounded Setup Times

The two-machine flow shop problem to minimize makespan or total completion time is addressed where setup times are considered to be separate from processing times. Setup times are relaxed to be random variables as opposed to the common assumption in the literature that setup times are exactly known (deterministic) in advance. The probability distributions of setup times are unknown, and only the lower and upper bounds of setup times are given before scheduling. In such cases, there may not exist a unique schedule that remains optimal for all possible realizations of setup times, and therefore, a set of schedules has to be considered which dominates all other schedules for the given makespan or total completion time. In this section, sufficient conditions are obtained when transposition of jobs minimizes makespan or total completion time.

Definition and Formulation

The two-machine flow shop problem is known to have an optimal solution when the objective is to minimize makespan (see [Joh54] or Section 4.1). The problem, however, is shown to be NP-hard when the performance measure is total completion time (see [GS78]) In these and the vast majority of subsequent research on the two-machine flow shop problem, it is assumed that the setup time of a job is included in the processing time of the job on each machine or the setup times are zero. While this assumption may be justified for some scheduling problems, other situations call for explicit (separate) setup time consideration.

Table 4.26: *Percentage of solved small and moderate (easy) instances with $w = w_2$*

Number of jobs	Number of non-availability intervals									
	1	2	3	4	5	6	7	8	9	10
5	83.7%	86.3%	86.7%	87.4%	86.6%	83.9%	86.7%	83.9%	85.0%	86.1%
10	93.2%	93.9%	93.0%	92.8%	91.5%	93.6%	93.5%	91.4%	93.6%	92.8%
15	96.2%	95.3%	96.1%	96.6%	96.7%	95.3%	94.4%	95.3%	95.9%	96.1%
20	97.5%	98.0%	96.6%	97.3%	97.3%	97.9%	98.1%	96.8%	97.3%	97.1%
25	98.8%	98.4%	98.2%	98.9%	98.7%	98.3%	97.6%	98.5%	98.3%	98.4%
30	98.6%	99.2%	99.5%	98.4%	99.2%	98.5%	99.0%	98.6%	99.0%	99.1%
35	99.3%	99.4%	99.2%	99.2%	99.2%	99.4%	99.7%	99.4%	99.5%	99.1%
40	99.5%	99.4%	99.1%	99.5%	99.4%	99.1%	98.9%	99.4%	99.3%	99.5%
45	99.6%	99.6%	99.7%	99.6%	99.5%	99.0%	99.9%	99.7%	99.5%	99.2%
50	99.8%	99.1%	99.6%	99.2%	99.6%	99.7%	99.4%	99.8%	99.3%	99.6%
55	99.8%	99.5%	99.8%	99.9%	99.8%	99.6%	99.7%	99.8%	99.7%	99.5%
60	99.4%	99.6%	100.0%	99.8%	99.7%	99.9%	99.6%	99.6%	99.9%	99.5%
65	99.6%	99.6%	99.8%	99.6%	99.5%	99.8%	99.8%	99.5%	99.7%	99.8%
70	99.6%	99.8%	99.4%	99.9%	99.7%	99.7%	99.9%	99.9%	99.8%	99.9%
75	99.9%	99.9%	99.5%	99.7%	99.7%	99.9%	99.9%	99.9%	99.8%	99.7%
80	99.9%	99.8%	99.7%	99.9%	99.8%	100.0%	99.6%	99.9%	99.6%	99.9%
85	99.9%	99.9%	99.9%	100.0%	99.7%	99.7%	100.0%	99.8%	99.8%	99.9%
90	99.9%	99.8%	100.0%	99.9%	99.5%	99.9%	99.9%	99.8%	99.9%	99.8%
95	99.9%	99.8%	99.8%	99.8%	99.8%	99.8%	99.8%	99.9%	99.8%	100.0%
100	100.0%	100.0%	100.0%	99.9%	99.9%	99.9%	100.0%	99.7%	99.9%	99.9%

The practical situations in which setup times must be considered as separate include the chemical, pharmaceutical, printing, food processing, metal processing, and semiconductor industries (see [AGA99]).

One advantage of considering setup times separate is that when there exists some idle time on the second machine (usually the case), then the setup time for a job on the second machine can be performed prior to the completion time of the job on the first machine. This means that the performance measures of makespan and total completion time may be improved by considering setup times as separate from processing times.

Therefore, some researchers considered the problem with separate setup times, i.e., relaxing the assumption that setup times are included in processing times. The papers [All95, KB84, YH79] are addressed the problem of minimizing makespan by considering setup times as separate. The papers [All00, BK86] are addressed the two-machine separate setup time problem with respect to total completion time criterion. In the research mentioned so far, the setup times are considered as separate from processing times but assumed to be known in advance, i.e., deterministic. In reality this assumption is not valid in many cases since setup times may be random variables. In fact, it is sometimes even difficult to find an appropriate probability distribution for setup times.

In this section, we address the two-machine flow shop scheduling problem to minimize total completion time or makespan when assuming deterministic setup times is not realistic and when it is hard to obtain exact probability distributions for random setup times. As such, a solution obtained by assuming fixed setup times or by assuming a certain probability distribution may not be even close to the optimal solution for the realization of the process.

It has been observed that although the exact probability distribution of setup times may not be known, upper and lower bounds on job setup times are easy to obtain in many practical cases. This information on the bounds of job setup times is important and it

Table 4.27: *Percentage of solved small and moderate (hard) instances with* $w = w_2$

Number of jobs	Number of non-availability intervals									
	1	2	3	4	5	6	7	8	9	10
5	68.0%	68.8%	69.3%	70.8%	70.8%	69.2%	69.1%	71.0%	70.2%	70.5%
10	78.5%	76.9%	80.2%	78.2%	79.1%	78.4%	78.5%	76.6%	77.6%	78.9%
15	84.2%	85.4%	85.8%	85.0%	86.2%	83.0%	86.6%	85.1%	83.7%	84.3%
20	90.5%	90.9%	91.5%	90.3%	89.1%	88.3%	91.2%	90.6%	90.9%	90.4%
25	93.0%	94.0%	93.0%	92.9%	93.1%	93.7%	93.5%	89.7%	93.2%	93.8%
30	94.2%	93.7%	94.7%	94.3%	95.3%	95.4%	94.5%	95.2%	94.9%	94.6%
35	96.6%	96.2%	96.4%	95.7%	96.8%	97.1%	96.0%	96.4%	96.5%	96.6%
40	96.6%	96.7%	96.4%	96.5%	97.6%	95.7%	96.2%	96.4%	97.7%	96.3%
45	96.4%	96.7%	98.3%	96.9%	97.3%	97.0%	96.7%	96.4%	96.4%	97.3%
50	97.1%	96.8%	98.0%	97.8%	97.2%	98.0%	97.7%	97.3%	97.5%	98.2%
55	98.4%	96.5%	98.0%	97.9%	98.1%	97.8%	98.2%	98.4%	97.8%	98.0%
60	97.3%	97.8%	97.1%	98.6%	97.7%	98.6%	97.8%	97.9%	97.8%	98.6%
65	98.1%	98.1%	97.6%	97.9%	98.7%	98.1%	98.3%	98.1%	98.4%	97.1%
70	98.4%	98.1%	98.7%	99.2%	97.9%	99.0%	98.5%	98.2%	98.2%	98.8%
75	97.9%	98.1%	99.2%	98.5%	98.7%	98.5%	99.1%	98.1%	98.5%	98.9%
80	98.8%	98.6%	98.5%	98.9%	98.5%	98.3%	98.3%	98.2%	98.3%	99.2%
85	97.9%	98.7%	99.0%	99.1%	98.7%	98.7%	99.0%	98.0%	98.8%	99.0%
90	98.4%	98.4%	98.1%	98.4%	98.8%	98.5%	99.2%	98.7%	98.7%	98.0%
95	98.8%	99.0%	99.0%	98.4%	99.4%	98.5%	98.7%	99.1%	98.5%	99.5%
100	99.1%	99.1%	98.6%	99.4%	99.0%	98.2%	98.6%	98.9%	98.5%	99.3%

Table 4.28: *Average running time and percentage of solved instances with* $w_1 > 0$ *and* $w_2 > 0$

Number of jobs	Number of non-availability intervals			
	10	100	500	1000
1000	100.0% (0.004 s)	100.0% (0.004 s)	100.0% (0.004 s)	100.0% (0.004 s)
2000	100.0% (0.016 s)	100.0% (0.016 s)	100.0% (0.016 s)	100.0% (0.016 s)
3000	100.0% (0.036 s)	100.0% (0.034 s)	100.0% (0.036 s)	100.0% (0.036 s)
4000	100.0% (0.064 s)	100.0% (0.062 s)	100.0% (0.059 s)	100.0% (0.062 s)
5000	100.0% (0.101 s)	100.0% (0.102 s)	100.0% (0.097 s)	100.0% (0.100 s)
6000	100.0% (0.144 s)	100.0% (0.144 s)	100.0% (0.145 s)	100.0% (0.155 s)
7000	100.0% (0.196 s)	100.0% (0.201 s)	100.0% (0.203 s)	100.0% (0.212 s)
8000	100.0% (0.263 s)	100.0% (0.282 s)	100.0% (0.272 s)	100.0% (0.281 s)
9000	100.0% (0.348 s)	100.0% (0.359 s)	100.0% (0.334 s)	100.0% (0.347 s)
10000	100.0% (0.444 s)	100.0% (0.416 s)	100.0% (0.434 s)	100.0% (0.425 s)

should be utilized in finding a solution for the scheduling problem.

We consider the environments in which setup times are random variables with unknown probability distributions when only a lower bound $l_{ik} > 0$ and an upper bound $u_{ik} \geq l_{ik}$ of the setup time s_{ik} of job J_i on machine M_k are given. The two-machine scheduling problem with bounded setup times may be denoted as $\mathcal{F}2/l_{ik} \leq s_{ik} \leq u_{ik}/\sum \mathcal{C}_i$ for total completion time criterion and $\mathcal{F}2/l_{ik} \leq s_{ik} \leq u_{ik}/\mathcal{C}_{max}$ for the makespan criterion. Notice that these problems can be considered as stochastic flow shop problems but under strict uncertainty when there is no prior information about probability distributions of the random setup times. It is only known that setup times will fall between the given lower and upper bounds with probability one. Such problems have been addressed in Chapter 3 and in Sections 4.1 and 4.2 of this chapter for the case where processing times are random variables but setup times are assumed to be zero.

Let p_{jk} denote the processing time of job $J_j \in J = \{J_1, J_2, \ldots, J_n\}$ on machine $M_k \in$

Table 4.29: *Average running time and percentage of solved instances with* $p_{i,2} = 2p_{i,1}, J_i \in J(12)$, $p_{k,1} = 2p_{k,2}, J_k \in J(21))$, $w_1 > 0$ *and* $w_2 > 0$

Number of jobs	Number of non-availability intervals			
	10	100	500	1000
1000	99.9% (0.008 s)	99.9% (0.008 s)	99.8% (0.008 s)	100.0% (0.008 s)
2000	99.9% (0.030 s)	100.0% (0.030 s)	100.0% (0.031 s)	99.8% (0.029 s)
3000	100.0% (0.069 s)	99.9% (0.070 s)	100.0% (0.068 s)	100.0% (0.069 s)
4000	100.0% (0.125 s)	100.0% (0.118 s)	99.8% (0.118 s)	100.0% (0.123 s)
5000	100.0% (0.206 s)	100.0% (0.191 s)	99.9% (0.191 s)	99.8% (0.191 s)
6000	100.0% (0.269 s)	100.0% (0.273 s)	100.0% (0.284 s)	99.9% (0.294 s)
7000	100.0% (0.387 s)	100.0% (0.377 s)	99.9% (0.398 s)	99.9% (0.382 s)
8000	100.0% (0.486 s)	100.0% (0.549 s)	100.0% (0.520 s)	100.0% (0.515 s)
9000	100.0% (0.700 s)	100.0% (0.644 s)	100.0% (0.654 s)	100.0% (0.677 s)
10000	100.0% (0.812 s)	100.0% (0.866 s)	100.0% (0.853 s)	100.0% (0.823 s)

Table 4.30: *Average running time and percentage of solved instances with* $w = w_1$

Number of jobs	Number of non-availability intervals			
	10	100	500	1000
1000	100.0% (0.004 s)	100.0% (0.005 s)	100.0% (0.004 s)	100.0% (0.004 s)
2000	100.0% (0.016 s)	100.0% (0.016 s)	100.0% (0.016 s)	100.0% (0.016 s)
3000	100.0% (0.036 s)	100.0% (0.035 s)	100.0% (0.035 s)	100.0% (0.036 s)
4000	100.0% (0.063 s)	100.0% (0.061 s)	100.0% (0.060 s)	100.0% (0.061 s)
5000	100.0% (0.098 s)	100.0% (0.099 s)	100.0% (0.097 s)	100.0% (0.101 s)
6000	100.0% (0.152 s)	100.0% (0.142 s)	100.0% (0.151 s)	100.0% (0.141 s)
7000	100.0% (0.195 s)	100.0% (0.205 s)	100.0% (0.196 s)	100.0% (0.201 s)
8000	100.0% (0.257 s)	100.0% (0.273 s)	100.0% (0.285 s)	100.0% (0.267 s)
9000	100.0% (0.345 s)	100.0% (0.331 s)	100.0% (0.344 s)	100.0% (0.347 s)
10000	100.0% (0.430 s)	100.0% (0.427 s)	100.0% (0.432 s)	100.0% (0.424 s)

$M = \{M_1, M_2\}$ at stage k, and let the bracket $[j,k]$ indicate the job in position j of the permutation of n jobs processed on machine M_k. That is, $p_{[ik]}$ denotes the processing time of the job in position i in job permutation processed on machine M_k. Similarly, $s_{[ik]}$, $l_{[ik]}$ and $u_{[ik]}$ denote setup time, lower bound of setup time and upper bound of setup time of the job in position i in job permutation processed on machine M_k, respectively.

In [YH79] for the deterministic problem where setup times are fixed (i.e., $s_{ik} = l_{ik} = u_{ik}$), the following formula for completion time $C[j]$ was established for the job in position j:

$$C_{[j]} = \max_{0\le u\le j}[\sum_{i=1}^{u}(s_{[i1]} - s_{[i2]} + p_{[i1]}) - \sum_{i=1}^{u-1} p_{[i2]}] + \sum_{i=1}^{j}(s_{[i2]} + p_{[i2]}).$$

This equation can be written as

$$\begin{aligned} C_{[j]} &= \max_{0\le u\le j}[\sum_{i=1}^{u}(s_{[i1]} + p_{[i1]}) - \sum_{i=1}^{u-1}(s_{[i2]} + p_{[i2]}) - s_{[u2]}] + \sum_{i=1}^{j}(s_{[i2]} + p_{[i2]}) \\ &= \max_{0\le u\le j}[SP_{[u1]} - (SP_{[u-1,2]} + SP_{[u2]})] + SP_{[j2]}, \end{aligned}$$

where $SP_{[jk]} = \sum_{r=1}^{j}(s_{[rk]} + p_{[rk]}), j = 1,2,\ldots,n$ and $M_k \in M = \{M_1, M_2\}$. Let $D_j = \max\{0, \sigma_1, \sigma_2, \ldots, \sigma_j\}$, where $\sigma_j = SP_{[j1]} - (SP_{[j-1,2]} + s_{[j2]}), j = 1,2,\ldots,n$. Then, $C[j]$ can be written as

$$C_{[j]} = SP_{[j2]} + D_j.$$

Table 4.31: *Average running time and percentage of solved instances with* $p_{i,2} = 2p_{i,1}, J_i \in J(12)$, $p_{k,1} = 2p_{k,2}, J_k \in J(21)$, $w = w_1$

Number of jobs	Number of non-availability intervals			
	10	100	500	1000
1000	100.0% (0.008 s)	99.8% (0.008 s)	99.9% (0.008 s)	99.9% (0.008 s)
2000	99.9% (0.031 s)	100.0% (0.030 s)	100.0% (0.029 s)	100.0% (0.028 s)
3000	100.0% (0.067 s)	99.9% (0.067 s)	100.0% (0.069 s)	100.0% (0.068 s)
4000	100.0% (0.123 s)	99.9% (0.118 s)	100.0% (0.119 s)	100.0% (0.118 s)
5000	100.0% (0.205 s)	100.0% (0.191 s)	100.0% (0.205 s)	100.0% (0.196 s)
6000	100.0% (0.280 s)	100.0% (0.269 s)	100.0% (0.293 s)	100.0% (0.286 s)
7000	100.0% (0.383 s)	100.0% (0.410 s)	100.0% (0.393 s)	100.0% (0.410 s)
8000	100.0% (0.519 s)	100.0% (0.544 s)	100.0% (0.533 s)	100.0% (0.526 s)
9000	100.0% (0.689 s)	100.0% (0.693 s)	100.0% (0.659 s)	100.0% (0.666 s)
10000	100.0% (0.847 s)	100.0% (0.848 s)	100.0% (0.861 s)	100.0% (0.830 s)

Table 4.32: *Average running time and percentage of solved instances with* $w = w_2$

Number of jobs	Number of non-availability intervals			
	10	100	500	1000
1000	100.0% (0.004 s)	100.0% (0.004 s)	100.0% (0.005 s)	100.0% (0.004 s)
2000	100.0% (0.016 s)	100.0% (0.016 s)	100.0% (0.016 s)	100.0% (0.016 s)
3000	100.0% (0.036 s)	100.0% (0.035 s)	100.0% (0.035 s)	100.0% (0.036 s)
4000	100.0% (0.062 s)	100.0% (0.063 s)	100.0% (0.059 s)	100.0% (0.063 s)
5000	100.0% (0.098 s)	100.0% (0.100 s)	100.0% (0.101 s)	100.0% (0.100 s)
6000	100.0% (0.145 s)	100.0% (0.142 s)	100.0% (0.141 s)	100.0% (0.142 s)
7000	100.0% (0.200 s)	100.0% (0.199 s)	100.0% (0.204 s)	100.0% (0.205 s)
8000	100.0% (0.267 s)	100.0% (0.285 s)	100.0% (0.265 s)	100.0% (0.268 s)
9000	100.0% (0.331 s)	100.0% (0.357 s)	100.0% (0.346 s)	100.0% (0.356 s)
10000	100.0% (0.422 s)	100.0% (0.430 s)	100.0% (0.439 s)	100.0% (0.428 s)

Once completion times of the jobs are known, then, total completion time, $\sum \mathcal{C}_i$, and the makespan, $\mathcal{C}_{max}$, are obtained as follows:

$$\sum \mathcal{C}_i = \sum_{j=1}^{n} (SP_{[j2]} + D_j);$$

$$\mathcal{C}_{max} = SP_{[n2]} + D_n. \tag{4.43}$$

It is clear from equation (4.43) that the term $SP_{[n2]}$ is a constant term which is independent of the sequence, and hence, minimization of $\mathcal{C}_{max}$ is equivalent to the minimization of the term D_n, which is the total idle time on machine M_2, until all the n jobs are completed.

For each job $J_j \in J = \{J_1, J_2, \ldots, J_n\}$ and machine $M_k \in M = \{M_1, M_2\}$, any feasible realization s_{jk} of setup time satisfies the inequalities

$$l_{jk} \leq s_{jk} \leq u_{jk}. \tag{4.44}$$

The exact value of setup time s_{jk} is not known before scheduling. On the other hand, we know the lower and upper bounds of setup times given by inequalities (4.44). These inequalities define polytope ST of feasible vectors $\overline{s} = (s_{1,1}, s_{1,2}, \ldots, s_{n1}, s_{n2})$ of setup times as

$$ST = \{\overline{s} \ : \ l_{jk} \leq s_{jk} \leq u_{jk}, \ J_j \in J, M_k \in M\}.$$

Table 4.33: *Average running time and percentage of solved instances with* $p_{i,2} = 2p_{i,1}, J_i \in J(12)$, $p_{k,1} = 2p_{k,2}, J_k \in J(21))$, $w = w_2$

Number of jobs	Number of non-availability intervals			
	10	100	500	1000
1000	99.9% (0.008 s)	100.0% (0.008 s)	100.0% (0.008 s)	99.8% (0.008 s)
2000	100.0% (0.029 s)	100.0% (0.030 s)	100.0% (0.029 s)	100.0% (0.030 s)
3000	100.0% (0.070 s)	99.9% (0.064 s)	100.0% (0.062 s)	100.0% (0.068 s)
4000	100.0% (0.124 s)	99.9% (0.116 s)	99.9% (0.118 s)	100.0% (0.123 s)
5000	100.0% (0.193 s)	100.0% (0.189 s)	100.0% (0.197 s)	100.0% (0.196 s)
6000	100.0% (0.265 s)	100.0% (0.296 s)	100.0% (0.261 s)	100.0% (0.291 s)
7000	100.0% (0.393 s)	100.0% (0.399 s)	100.0% (0.391 s)	100.0% (0.410 s)
8000	100.0% (0.495 s)	100.0% (0.543 s)	100.0% (0.550 s)	100.0% (0.505 s)
9000	100.0% (0.678 s)	100.0% (0.675 s)	100.0% (0.666 s)	100.0% (0.676 s)
10000	100.0% (0.836 s)	100.0% (0.864 s)	100.0% (0.855 s)	100.0% (0.846 s)

Permutation schedules are dominant for the deterministic two-machine flow shop with separate processing time problem with respect to any regular performance measure, e.g., makespan and total completion time (see [YH79] or [All00]). That is, one only needs to consider the same sequence of jobs on each machine M_1 and M_2 in order to find the optimal schedule. Permutation schedules are also dominant for the considered problems with random job completion times (see [All95]). Setup times are random variables for the two considered problems which result in the random job completion times, and therefore, permutation schedules are also dominant for the problems under consideration. Thus, there are $n!$ sequences (permutations) $S^\pi = \{\pi_1, \pi_2, \ldots, \pi_{n!}\}$ for the problems of $\mathcal{F}2/l_{jk} \leq s_{jk} \leq u_{jk}/\mathcal{C}_{\max}$ and $\mathcal{F}2/l_{jk} \leq s_{jk} \leq u_{jk}/\sum \mathcal{C}_i$ that needs to be considered in finding the optimal schedule for each problem.

Similar to Chapter 3, we will use the following definition of a solution to the problems of $\mathcal{F}2/l_{jk} \leq s_{jk} \leq u_{jk}/\mathcal{C}_{\max}$ and $\mathcal{F}2/l_{jk} \leq s_{jk} \leq u_{jk}/\sum \mathcal{C}_i$.

Definition 4.7 *A set of sequences $S^\pi(ST) \subseteq S^\pi$ is a solution to the problem $\mathcal{F}2/l_{jk} \leq s_{jk} \leq u_{jk}/\mathcal{C}_{\max}$ (problem $\mathcal{F}2/l_{jk} \leq s_{jk} \leq u_{jk}/\sum \mathcal{C}_i$), if for each fixed feasible vector $\overline{s} \in ST$ of setup times, the set $S^\pi(ST)$ contains at least one optimal sequence for the corresponding deterministic flow shop problem.*

Thus the whole set S^π of sequences is an obvious solution for the problems of $\mathcal{F}2/l_{jk} \leq s_{jk} \leq u_{jk}/\sum \mathcal{C}_i$ and $\mathcal{F}2/l_{jk} \leq s_{jk} \leq u_{jk}/\mathcal{C}_{\max}$. However, to construct the whole set S^π is only possible for a small number n of jobs. It is also hard or practically infeasible for a scheduler to choose the best sequence from a large set $S^\pi(ST)$ of candidates as the processing of jobs evolves. Therefore, it is important to minimize the cardinality of solution $S^\pi(ST)$ constructed for problems of $\mathcal{F}2/l_{jk} \leq s_{jk} \leq u_{jk}/\sum \mathcal{C}_i$ and $\mathcal{F}2/l_{jk} \leq s_{jk} \leq u_{jk}/\mathcal{C}_{\max}$. To this end, we introduce the following two types of dominance relations on the set of sequences S^π.

Definition 4.8 *Sequence π_u dominates sequence π_v with respect to polytope ST if the inequality $\sum C_i(\pi_u) \leq \sum C_i(\pi_v)$ holds for any vector $\overline{s} \in ST$ of setup times for the problem $\mathcal{F}2/l_{jk} \leq s_{jk} \leq u_{jk}/\sum \mathcal{C}_i$. Similarly, sequence π_u dominates sequence π_v with respect to polytope ST if the inequality $C_{max}(\pi_u) \leq C_{max}(\pi_v)$ holds for any vector $\overline{s} \in ST$ of setup times for the problem $\mathcal{F}2/l_{jk} \leq s_{jk} \leq u_{jk}/\mathcal{C}_{\max}$.*

By Definition 4.7 and Definition 4.8, a set of sequences $S^{\pi}(ST) \subseteq S^{\pi}$ is a solution to the problem $\mathcal{F}2/l_{jk} \leq s_{jk} \leq u_{jk}/\sum \mathcal{C}_i$ (problem $\mathcal{F}2/l_{jk} \leq s_{jk} \leq u_{jk}/\mathcal{C}_{\max}$) if for each sequence $\pi_v \in S^{\pi}$ there exists sequence $\pi_u \in S^{\pi}(ST)$ which dominates sequence π_v.

Transposition of Two Jobs

Let q_r denote a subsequence of a given complete sequence $\pi_u \in S^{\pi}$ of all the n jobs. That is, the notations of $\pi_u = (q_1, J_i, q_2, J_k, q_3)$ and $\pi_v = (q_1, J_k, q_2, J_i, q_3)$ mean that the two sequences of π_u and π_v have the same jobs in all positions except that the jobs J_i and J_k are interchanged. When the jobs J_i and J_k are adjacent in the sequence, these similar sequences of $\pi_g \in S^{\pi}$ and $\pi_h \in S^{\pi}$ can be expressed as follows: $\pi_g = (q_1, J_i, J_k, q_2)$ and $\pi_h = (q_1, J_i, J_k, q_2)$.

Theorem 4.13 *For a two-machine flow shop problem $\mathcal{F}2/l_{jk} \leq s_{jk} \leq u_{jk}/\mathcal{C}_{\max}$ of minimizing makespan, the sequence $\pi_u = (q_1, J_i, q_2, J_k, q_3) \in S^{\pi}$ dominates the sequence $\pi_v = (q_1, J_i, q_2, J_k, q_3) \in S^{\pi}$ with respect to polytope ST if the two inequalities of $p_{k2} \leq p_{i2}$ and $u_{i1} + p_{i1} + u_{k2} \leq l_{k1} + p_{k1} + l_{i2}$ hold.*

Theorem 4.14 *For a two-machine flow shop problem $\mathcal{F}2/l_{jk} \leq s_{jk} \leq u_{jk}/\sum \mathcal{C}_i$ of minimizing total completion time, the sequence $\pi_u = (q_1, J_i, q_2, J_k, q_3) \in S^{\pi}$ dominates the sequence $\pi_v = (q_1, J_i, q_2, J_k, q_3) \in S^{\pi}$ with respect to polytope ST if the three inequalities of $p_{k2} \leq p_{i2}$, $u_{i2} + p_{i2} \leq l_{k2} + p_{k2}$ and $u_{i1} + p_{i1} + u_{k2} \leq l_{k1} + p_{k1} + l_{i2}$ hold.*

PROOF. We combine the proof of Theorem 4.13 with that of Theorem 4.14 since the former is a part of the latter.

Let us consider two job sequences π_u and π_v where $\pi_u \in S^{\pi}$ is a sequence in which job J_i is in position α and job J_k in position β, with $\alpha < \beta$, whereas sequence $\pi_v \in S^{\pi}$ is obtained from sequence π_u by interchanging only the jobs in positions α and β.

If it will be necessary, we will add permutation $\pi_r \in S^{\pi}$ to the above notations $C_{[j]}$, D_j, σ_j, $SP_{[jk]}$, $\sum C_i$, i.e., we will also use the following notations: $C_{[j]}(\pi_r)$, $D_j(\pi_r)$, $\sigma_j(\pi_r)$, $SP_{[jk]}(\pi_r)$, $\sum C_i(\pi_r)$.

It should be clear that $SP_{[\alpha-1,1]}(\pi_u) = SP_{[\alpha-1,1]}(\pi_v)$ and $SP_{[\alpha-1,2]}(\pi_u) = SP_{[\alpha-1,2]}(\pi_v)$, since both sequences π_u and π_v have the same job in each position of $1, 2, \ldots, \alpha - 1$.

For position$j = \alpha$, we obtain

$$\sigma_{\alpha}(\pi_2) = SP_{[\alpha-1,1]}(\pi_2) + s_{k1} + p_{k1} - (SP_{[\alpha-1,2]}(\pi_2) + s_{k2}),$$

$$\sigma_{\alpha}(\pi_1) = SP_{[\alpha-1,1]}(\pi_1) + s_{i1} + p_{i1} - (SP_{[\alpha-1,2]}(\pi_1) + s_{i2}).$$

From the above two equations we obtain inequality

$$\sigma_{\alpha}(\pi_u) \leq \sigma_{\alpha}(\pi_v), \tag{4.45}$$

since inequality $u_{i1} + p_{i1} + u_{k2} \leq l_{k1} + p_{k1} + l_{i2}$ always implies inequality $s_{i1} + p_{i1} + s_{k2} \leq s_{k1} + p_{k1} + s_{i2}$. For positions $j = \alpha + 1, \alpha + 2, \ldots, \beta - 1$, we obtain equalities

$$\sigma_j(\pi_v) = SP_{[\alpha-1,1]}(\pi_v) + s_{k1} + p_{k1} + \sum_{r=\alpha+1}^{j} (s_{[r1]} + p_{[r1]}) - [SP_{[\alpha-1,2]}(\pi_v)$$

$$+s_{k2}+p_{k2}+\sum_{r=\alpha+1}^{j-1}(s_{[r2]}+p_{[r2]})+s_{[j2]}];$$

$$\sigma_j(\pi_u)=SP_{[\alpha-1,1]}(\pi_u)+s_{i1}+p_{i1}+\sum_{r=\alpha+1}^{j}(s_{[r1]}+p_{[r1]})-[SP_{[\alpha-1,2]}(\pi_u)$$

$$+s_{i2}+p_{i2}+\sum_{r=\alpha+1}^{j-1}(s_{[r2]}+p_{[r2]})+s_{[j2]}],$$

where $\sum_{r=\alpha+1}^{\alpha}(s_{[r2]}+p_{[r2]})=0$. Both sequences π_u and π_v have the same job in all positions except for position α, and hence, it follows that $\sigma_j(\pi_1)-\sigma_j(\pi_2)=(s_{i1}+p_{i1}+s_{k2}+p_{k2})-(s_{k1}+p_{k1}+s_{i2}+p_{i2})$, which results in

$$\sigma_j(\pi_1)\leq\sigma_j(\pi_2), \tag{4.46}$$

since $p_{k2}\leq p_{i2}$ and $s_{i1}+p_{i1}+s_{k2}\leq s_{k1}+p_{k1}+s_{i2}$ (again this is implied by the fact that inequality $u_{i1}+p_{i1}+u_{k2}\leq l_{k1}+p_{k1}+l_{i2}$ holds). For position $j=\beta$, we obtain equations

$$\sigma_\beta(\pi_v)=SP_{[\alpha-1,1]}(\pi_v)+s_{k1}+p_{k1}+\sum_{r=\alpha+1}^{\beta-1}(s_{[r1]}+p_{[r1]})+s_{i1}+p_{i1}$$

$$-[SP_{[\alpha-1,2]}(\pi_v)+s_{k2}+p_{k2}+\sum_{r=\alpha+1}^{\beta-1}(s_{[r2]}+p_{[r2]})+s_{i2}];$$

$$\sigma_\beta(\pi_u)=SP_{[\alpha-1,1]}(\pi_u)+s_{i1}+p_{i1}+\sum_{r=\alpha+1}^{\beta-1}(s_{[r1]}+p_{[r1]})+s_{k1}+p_{k1}$$

$$-[SP_{[\alpha-1,2]}(\pi_u)+s_{i2}+p_{i2}+\sum_{r=\alpha+1}^{\beta-1}(s_{[r2]}+p_{[r2]})+s_{k2}].$$

It follows from the last two equations that

$$\sigma_\beta(\pi_u)\leq\sigma_\beta(\pi_v), \tag{4.47}$$

since $p_{k2}\leq p_{i2}$. It is obvious that $\sigma_j(\pi_1)=\sigma_j(\pi_2)$ for each position $j=1,2,\ldots,\alpha-1$ since both sequences π_u and π_v have the same job in each of these positions.

It can easily be shown that $s_j(\pi_u)=s_j(\pi_v)$ for each position $j=\beta+1,\beta+2,\ldots,n$. From these facts and inequalities (4.45), (4.46), (4.47), we obtain inequality

$$\sigma_j(\pi_1)\leq\sigma_j(\pi_2) \tag{4.48}$$

for each $j=1,2,\ldots,n$. Therefore, $D_n(\pi_u)\leq D_n(\pi_v)$. This shows that the sequence $\pi_u=(q_1,J_i,q_2,\ J_k,q_3)\in S^\pi$ dominates the sequence $\pi_2=(q_1,J_k,q_2,J_i,q_3)\in S^\pi$ with respect to minimization makespan $\mathcal{C}_{max}$. Thus, Theorem 4.13 is proven.

For the proof of Theorem 4.14, observe that

$$C_{[j]}(\pi_u)=C_{[j]}(\pi_v) \tag{4.49}$$

for each position $j = 1, 2, \ldots, \alpha - 1$ since both sequences have the same job in each of these positions. It should be also observe that $D_{\alpha-1}(\pi_u) = D_{\alpha-1}(\pi_v)$. It can easily be shown that for each position $j = \beta, \beta + 1, \ldots, n$ the following equality holds:

$$C_{[j]}(\pi_u) - C_{[j]}(\pi_v) = \max\{D_{\alpha-1}(\pi_u), \sigma_\alpha(\pi_u), \sigma_{\alpha+1}(\pi_u), \ldots, \sigma_j(\pi_u)\}$$
$$- \max\{D_{\alpha-1}(\pi_v), \sigma_\alpha(\pi_v), \sigma_{\alpha+1}(\pi_v), \ldots, \sigma_j(\pi_v)\}.$$

But it follows from (4.48) that

$$\max\{D_{\alpha-1}(\pi_u), \sigma_\alpha(\pi_u), \sigma_{\alpha+1}(\pi_u), \ldots, \sigma_j(\pi_u)\}$$
$$\leq \max\{D_{\alpha-1}(\pi_v), \sigma_\alpha(\pi_v), \sigma_{\alpha+1}(\pi_v), \ldots, \sigma_j(\pi_v)\}.$$

Therefore, inequality

$$C_{[j]}(\pi_u) \leq C_{[j]}(\pi_v) \tag{4.50}$$

holds for each position $j = \beta, \beta + 1, \ldots, n$. Observe that for $j = \alpha$ the following equalities hold:

$$C_{[\alpha]}(\pi_v) = SP_{[\alpha-1,2]}(\pi_v) + s_{k2} + p_{k2} + \max\{D_{\alpha-1}(\pi_v), \sigma_\alpha(\pi_v)\};$$
$$C_{[\alpha]}(\pi_1) = SP_{[\alpha-1,2]}(\pi_u) + s_{i2} + p_{i2} + \max\{D_{\alpha-1}(\pi_u), \sigma_\alpha(\pi_u)\}.$$

For $j = \alpha + 1, \alpha + 2, \ldots, \beta - 1$ the following equalities hold:

$$C_{[j]}(\pi_v) = SP_{[\alpha-1,2]}(\pi_v) + s_{k2} + p_{k2} + \sum_{r=\alpha+1}^{j} (s_{[r2]} + p_{[r2]})$$
$$+ \max\{D_{\alpha-1}(\pi_v), \sigma_\alpha(\pi_v), \sigma_{\alpha+1}(\pi_v), \ldots, \sigma_j(\pi_v)\};$$
$$C_{[j]}(\pi_u) = SP_{[\alpha-1,2]}(\pi_u) + s_{i2} + p_{i2} + \sum_{r=\alpha+1}^{j} (s_{[r2]} + p_{[r2]})$$
$$+ \max\{D_{\alpha-1}(\pi_u), \sigma_\alpha(\pi_u), \sigma_{\alpha+1}(\pi_u), \ldots, \sigma_j(\pi_u)\}.$$

Hence, for $j = \alpha, \alpha + 1, \ldots, \beta - 1$, we obtain

$$C_{[j]}(\pi_u) - C_j(\pi_v) = (s_{i2} + p_{i2}) - (s_{k2} + p_{k2})$$
$$+ \max\{D_{\alpha-1}(\pi_u), \sigma_\alpha(\pi_u), \sigma_{\alpha+1}(\pi_u), \ldots, \sigma_j(\pi_u)\}$$
$$- \max\{D_{\alpha-1}(\pi_v), \sigma_\alpha(\pi_v), \sigma_{\alpha+1}(\pi_v), \ldots, \sigma_j(\pi_v)\}.$$

But condition $u_{i2} + p_{i2} \leq l_{k2} + p_{k2}$ implies $s_{i2} + p_{i2} \leq s_{k2} + p_{k2}$ and

$$\max\{D_{\alpha-1}(\pi_u), \sigma_\alpha(\pi_u), \sigma_{\alpha+1}(\pi_u), \ldots, \sigma_j(\pi_u)\}$$
$$\leq \max\{D_{\alpha-1}(\pi_v), \sigma_\alpha(\pi_v), \sigma_{\alpha+1}(\pi_v), \ldots, \sigma_j(\pi_v)\}.$$

Therefore, inequality

$$C_{[j]}(\pi_u) \leq C_{[j]}(\pi_v), \tag{4.51}$$

holds for each $j = \alpha, \alpha + 1, \ldots, \beta - 1$. Now it follows from equality (4.49) and inequalities (4.50) and (4.51) that the sequence $\pi_u = (q_1, J_i, q_2, J_k, q_3) \in S^\pi$ dominates the sequence $\pi_v = (q_1, J_k, q_2, J_i, q_3) \in S^\pi$ with respect to total completion time minimization. This completes the proof.

$\diamond$

Remark 4.1 Note that sufficient conditions given in Theorem 4.13 are included to sufficient conditions given in Theorem 4.14. Thus, using sufficient conditions given in Theorem 4.14, one can minimize both objective functions $\mathcal{C}_{max}$ and $\sum \mathcal{C}_i$.

Transposition of Two Adjacent Jobs

Next, we prove analogous sufficient conditions for permutation domination in the case when transposed jobs are adjacent.

Theorem 4.15 *For a two-machine flow shop problem $\mathcal{F}2/l_{jk} \le s_{jk} \le u_{jk}/\mathcal{C}_{\max}$ of minimizing makespan, the sequence $\pi_u = (q_1, J_i, J_k, q_2) \in S^\pi$ dominates the sequence $\pi_v = (q_1, J_k, J_i, q_2) \in S^\pi$ with respect to polytope ST if the inequality $u_{i1}+p_{i1}+u_{k2} \le l_{k1}+p_{k1}+l_{i2}$ and one of the following inequalities of*

(i) $u_{k1} + p_{k1} \le p_{i2} + l_{k2}$,
(ii) $u_{i1} + p_{i1} \le l_{i2} + p_{i2}$,
(iii) $p_{k2} \le p_{i2}$

hold.

Theorem 4.16 *For a two-machine flow shop problem $\mathcal{F}2/l_{jk} \le s_{jk} \le u_{jk}/\sum \mathcal{C}_i$ of minimizing total completion time, the sequence $\pi_u = (q_1, J_i, J_k, q_2) \in S^\pi$ dominates the sequence $\pi_v = (q_1, J_i, J_k, q_2) \in S^\pi$ with respect to polytope ST if the inequalities $u_{i1}+p_{i1}+u_{k2} \le l_{k1}+p_{k1}+l_{i2}$, $u_{i2}+p_{i2} \le l_{k2}+p_{k2}$ and one of the following inequalities of*

(i) $u_{k1} + p_{k1} \le p_{i2} + l_{k2}$,
(ii) $u_{i1} + p_{i1} \le l_{i2} + p_{i2}$,
(iii) $p_{k2} \le p_{i2}$

hold.

Proof. Again, we combine the proof of Theorem 4.15 with that of Theorem 4.16 since the former is a part of the latter.

We assume the sequence π_u has job $J_i \in J$ in position α and job $J_k \in J$ in position $\alpha+1$, and the sequence π_v is obtained from the sequence π_u by interchanging only the jobs in positions α and $\alpha+1$. For these two sequences, we obtain

$$\sigma_\alpha(\pi_u) = SP_{[\alpha-1,1]}(\pi_u) + s_{i1} + p_{i1} - SP_{[\alpha-1,2]}(\pi_u) - s_{i2};$$

$$\sigma_\alpha(\pi_v) = SP_{[\alpha-1,1]}(\pi_v) + s_{k1} + p_{k1} - SP_{[\alpha-1,2]}(\pi_v) - s_{k2};$$

$$\sigma_{\alpha+1}(\pi_u) = SP_{[\alpha-1,1]}(\pi_u) + s_{i1} + p_{i1} + s_{k1} + p_{k1} - SP_{[\alpha-1,2]}(\pi_u) - s_{i2} - p_{i2} - s_{k2};$$

$$\sigma_{\alpha+1}(\pi_v) = SP_{[\alpha-1,1]}(\pi_v) + s_{k1} + p_{k1} + s_{i1} + p_{i1} - SP_{[\alpha-1,2]}(\pi_v) - s_{k2} - p_{k2} - s_{i2}.$$

Since both sequences have the same job in each of positions $1, 2, \ldots, \alpha-1$, we obtain $D_{\alpha-1}(\pi_u) = D_{\alpha-1}(\pi_v)$. We denote $D_{\alpha-1} = D_{\alpha-1}(\pi_u) = D_{\alpha-1)}(\pi_v)$, and obtain

$$C_{[\alpha]}(\pi_u) = SP_{[\alpha-1,2]}(\pi_u) + s_{i2} + p_{i2} + \max\{D_{\alpha-1}, \sigma_\alpha(\pi_u)\};$$

$$C_{[\alpha]}(\pi_v) = SP_{[\alpha-1,2]}(\pi_v) + s_{k2} + p_{k2} + \max\{D_{\alpha-1}, \sigma_\alpha(\pi_v)\};$$

$$C_{[\alpha+1]}(\pi_u) = SP_{[\alpha-1,2]}(\pi_u) + s_{i2} + p_{i2} + s_{k2}$$

$$+p_{k2} + \max\{D_{\alpha-1}, \sigma_\alpha(\pi_u), \sigma_{\alpha+1}(\pi_u)\};$$

$$C_{[\alpha+1]}(\pi_v) = SP_{[\alpha-1,2]}(\pi_v) + s_{k2} + p_{k2} + s_{i2}$$

$$+p_{i2} + \max\{D_{\alpha-1}, \sigma_\alpha(\pi_v), \sigma_{\alpha+1}(\pi_v)\}.$$

Observe that $SP_{[\alpha-1,1]}(\pi_u) = SP_{[\alpha-1,1]}(\pi_v)$ and $SP_{[\alpha-1,2]}(\pi_u) = SP_{[\alpha-1,2]}(\pi_v)$, because both sequences have the same job in each of positions $1, 2, \ldots, \alpha-1$. As a result, it holds that

$$\sigma_j(\pi_u) = \sigma_j(\pi_v), \tag{4.52}$$

for each $j = 1, 2, \ldots, \alpha-1$. It can easily be shown that

$$\sigma_j(\pi_u) = \sigma_j(\pi_v) \tag{4.53}$$

for each $j = \alpha+2, \alpha+3, \ldots, n$. It follows from inequalities (4.52) and (4.53) that

$$\sigma_\alpha(\pi_u) \leq \sigma_\alpha(\pi_v), \tag{4.54}$$

since $u_{i1} + p_{i1} + u_{k2} \leq l_{k1} + p_{k,1} + l_{i,2}$. From (4.54) it follows

$$\max D_{\alpha-1}, \sigma_\alpha(\pi_u) \leq \max D_{\alpha-1}, \sigma_\alpha(\pi_v). \tag{4.55}$$

From the above inequalities we obtain the following claims.

If $u_{k1} + p_{k1} \leq p_{i2} + l_{k2}$, then

$$\sigma_{\alpha+1}(\pi_u) \leq \sigma_\alpha(\pi_v). \tag{4.56}$$

If $u_{i1} + p_{i1} \leq l_{i2} + p_{i2}$, then

$$\sigma_{\alpha+1}(\pi_u) \leq \sigma_\alpha(\pi_v). \tag{4.57}$$

If $p_{k2} \leq p_{i2}$, then

$$\sigma_{\alpha+1}(\pi_u) \leq \sigma_{\alpha+1}(\pi_v). \tag{4.58}$$

It any one of the inequalities (4.56), (4.57) or (4.58) holds, then

$$\max\{D_{\alpha-1}, \sigma_\alpha(\pi_u), \sigma_{\alpha+1}(\pi_u)\} \leq \max\{D_{\alpha-1}, \sigma_\alpha(\pi_v), \sigma_{\alpha+1}(\pi_v)\}. \tag{4.59}$$

Therefore, from inequalities (4.52), (4.53), (4.54) and (4.59), it follows $D_n(\pi_u) \leq D_n(\pi_v)$, which shows that the sequence $\pi_u = (q_1, J_i, J_k, q_2) \in S^\pi$ dominates the sequence $\pi_v = (q_1, J_k, J_i, q_2) \in S^\pi$ with respect to makespan minimization. Theorem 4.15 is proven.

Taking the difference between completion times of the jobs in positions α and $\alpha+1$ for the two sequences under consideration yields

$$[C_{[\alpha]}(\pi_u) + C_{[\alpha+1]}(\pi_u)] - [C_{[\alpha]}(\pi_v) + C_{[\alpha+1]}(\pi_v)] = (s_{i2} + p_{i2})$$

$$-(s_{k2} + p_{k2}) + \max\{D_{\alpha-1}, \sigma_\alpha(\pi_u)\} - \max\{D_{\alpha-1}, \sigma_\alpha(\pi_v)\}$$

$$+ \max\{D_{\alpha-1}, \sigma_\alpha(\pi_u), \sigma_{\alpha+1}(\pi_u)\} - \max\{D_{\alpha-1}, \sigma_\alpha(\pi_v), \sigma_{\alpha+1}(\pi_v)\}.$$

From (4.55), (4.59) and the fact that $u_{i2} + p_{i2} \leq l_{k2} + p_{k2}$ it follows

$$C_{[\alpha]}(\pi_u) + C_{[\alpha+1]}(\pi_u) \leq C_{[\alpha]}(\pi_v) + C_{[\alpha+1]}(\pi_v). \tag{4.60}$$

It can also be shown that for each $j = \alpha + 2, \alpha + 3, \ldots, n$ it follows

$$C_{[j]}(\pi_u) - C_{[j]}(\pi_v) = \max\{D_{\alpha-1}, \sigma_\alpha(\pi_u), \sigma_{\alpha+1}(\pi_u), \sigma_{\alpha+2}(\pi_u), \ldots, \sigma_j(\pi_u)\}$$

$$- \max\{D_{\alpha-1}, \sigma_\alpha(\pi_v), \sigma_{\alpha+1}(\pi_v), \sigma_{\alpha+2}(\pi_v), \ldots, \sigma_j(\pi_v)\}.$$

Observe that $\sigma_k(\pi_3) = \sigma_k(\pi_4)$ for each $k = \alpha + 2, \alpha + 3, \ldots, n$. Therefore, from the inequality of (4.59) we obtain

$$C_{[j]}(\pi_u) \leq C_{[j]}(\pi_v) \tag{4.61}$$

for each $j = \alpha + 2, \alpha + 3, \ldots, n$. It is obvious that

$$C_{[j]}(\pi_u) = C_{[j]}(\pi_v) \tag{4.62}$$

for each $j = 1, 2, \ldots, \alpha - 1$. Clearly from (4.55) and (4.59) – (4.62), it follows that $\sum C_i(\pi_u) \leq \sum C_i(\pi_v)$ which completes the proof of Theorem 4.16.

◇

The survey paper [AGA99] on separate setup scheduling problems indicates that they are classified as either sequence-independent or sequence-dependent with the latter being more general. If the setup time depends only on the job to be processed, it is called sequence-independent. On the other hand, if the setup time depends on both the job to be processed and the previous job, it is called sequence-dependent.

In both sequence-dependent and sequence-independent cases, the setup times are deterministic and known beforehand. In this section, we relax the assumption that they are deterministic and consider setup times to be random variables with free distributions where only a lower and upper bounds are given.

4.6 Comments and References

Lai et al. (1997) introduced such a solution to scheduling problem with uncertain job processing times.

The approach under consideration was originally proposed by Lai et al. (1997), Lai and Sotskov (1999) for makespan criterion, and was developed by Lai et al. (2004) for total completion time criterion $\sum C_i$. In particular, the formula for calculating the *stability radius* of the optimal schedule (i.e., the largest value of simultaneous independent variations of job processing times that this schedule remains optimal) has been provided by Lai et al. (1997). In the work of Lai et al. (2004), stability analysis of a schedule minimizing total completion time was involved in a branch and bound method for solving jobshop problem $Jm|p_{ij}^L \leq p_{ij} \leq p_{ij}^U|\sum C_i$ with m machines and different technological routes of jobs. In recent work of Allahverdi and Sotskov (2003), for a two-machine flowshop problem $F2|p_{ij}^L \leq p_{ij} \leq p_{ij}^U|C_{max}$ sufficient conditions have been found when transposition of two jobs minimizes makespan. Sotskov et al. (2004) studied total completion time in a flowshop with uncertain processing times. In particular, the geometrical algorithm has been developed for solving flowshop problem $Fm|p_{ij}^L \leq p_{ij} \leq p_{ij}^U, n = 2|\sum C_i$ with m machines and two jobs. For a flowshop problem with two machines and that with three machines, sufficient conditions have been found when transposition of two jobs

minimizes total completion time. Recent work of Allahverdi et al. (2003) is devoted to the case of separate setup times to minimize makespan C_{max} or total completion time $\sum C_i$. Namely, processing times are fixed while setup times are relaxed to be distribution-free random variables with only the lower and upper bound of setup times being given before scheduling. Allahverdi et al. (2003) found local and global dominance relations for such a flowshop problem with two machines.

In this section, we find a combination of *sufficient conditions* proved by Allahverdi and Sotskov (2003) which put together *necessary conditions* for fixing the order of two jobs in a solution to problem $F2|p_{ij}^L \leq p_{ij} \leq p_{ij}^U|C_{max}$.

In contrast to papers by Lai et al. (1997) and Lai and Sotskov (1999), where exponential algorithms based on exhausting enumeration of the feasible schedules have been derived for a jobshop problem $Jm|p_{ij}^L \leq p_{ij} \leq p_{ij}^U|C_{max}$, in this paper, we develop polynomial procedures for solving special case of the problem when $m = 2$ and all the jobs have the same technological route.

Allahverdi and Sotskov (2003) proved the following sufficient conditions for such fixing the order of two jobs.

We find a combination of *sufficient conditions* proved by Allahverdi and Sotskov (2003) which put together *necessary conditions* for fixing the order of two jobs in a solution to problem $\mathcal{F}2/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$.

A number of recent papers have been devoted to the makespan minimization in the flow shop scheduling problem with two machines $\{1, 2\}$ which have $w \geq 1$ non-availability intervals, see Sections 5 and 6 in the survey by Schmidt (2000).

It should be noted that only flow-shop systems (among multi-stage ones) with limited machine availability have been treated in the OR literature so far.

In [KBF+02], it was proven that problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ is unary NP-hard if an arbitrary number w of non-availability intervals occur on one of the two machines. In [Lee97], it has been shown that this problem is binary NP-hard even if there is a single non-availability interval ($w = 1$) on either machine M_1 or machine M_2. In [Lee97], it was also provided a heuristic with a makespan which is at most (3/2) times larger than the minimal makespan if this non-availability interval is on the first machine, and (4/3) times larger than the minimal makespan if the non-availability interval is on the second machine.

In [CW00], it was shown that the error bound of (3/2) for the situation with a non-availability interval on the first machine is tight. For the problem with availability constraints imposed on each machine when the non-availability interval on one machine is followed immediately by the non-availability interval on the other machine, paper [CW99] provided a heuristic with a worst-case error bound of (5/3). In [KBF+02], the branch-and-bound algorithm was developed which outperformed the dynamic programming algorithm proposed in [Lee97]) for the case of one non-availability interval.

In [NK04], for problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$ with one non-availability interval *a fully polynomial approximation scheme* has been developed with complexity $O(n^5/\epsilon^4)$ where ϵ is upper bound of the exceeding of the minimal makespan in the approximate schedule that may be obtained. In [BLSS02], stability of Johnson's schedule constructed for problem $\mathcal{F}2//\mathcal{C}_{max}$ has been tested for the case of problem $\mathcal{F}2,NC^{off}/pmtn/\mathcal{C}_{max}$. This results are presented in Section 4.3.

The reviews in this area are given in [LLP97, KBF$^+$02, SS98, Sch00].

Table 4.34: *Notations for the two-machine flow shop and job shop*

Symbols	Description
π_k	Permutation of jobs from set J: $\pi_k = \{J_{k_1}, J_{k_2}, \ldots, J_{k_n}\}$ where $\{k_1, k_2, \ldots, k_n\} = \{1, 2, \ldots, n\}$
S^π	Set of all permutations $S^\pi = \{\pi_1, \pi_2, \ldots, \pi_{n!}\}$ of n jobs J defining set of all *permutation schedules*
$S^\pi(T)$	Set of permutations $S^\pi(T) \subseteq S^\pi$ which is a solution to the *uncertain* flow shop problem
N_1	Set of jobs jobs with $p_{i1} \leq p_{i2}$
N_2	Set of jobs jobs with $p_{i1} \geq p_{i2}$
N_{M_1}	Set of jobs with left intervals of job processing times on machine M_1
N_{M_2}	Set of jobs with left intervals of job processing times on machine M_2
$J(ij)$	Subset of jobs of set J with machine route (M_i, M_j) where $\{i, j\} = \{1, 2\}$
$J(i)$	Subset of jobs of set J with one-machine route (M_i) where $i \in \{1, 2\}$
$C_i(\pi_k, p)$	Completion time of job $J_i \in J$ in schedule $\pi_k \in S^\pi$ provided that processing times are given by vector $p \in T$
$C_{max}(\pi_k, p)$	Maximal completion time (makespan) of schedule $\pi_k \in S^\pi$ provided that processing times are given by vector $p \in T$: $C_{max}(\pi_k, p) = \max\{C_i(\pi_k, p)$
T^J	Set of feasible vectors of job processing times in uncertain job shop problem: $T^J = T(1) \cup T(2) \cup T(12) \cup T(21)$ where $T(w)$ defines intervals of feasible processing times of jobs $J(k)$, $k \in \{1, 2, 12, 21\}$
w_j	Number of non-availability intervals on machine M_j
w	Total number of non-availability intervals: $w = w_1 + w_2$
N_{kj}	k^{th} non-availability interval on machine M_j
$s(N_{kj})$	Starting point of the k^{th} non-availability interval on machine M_j
$f(N_{kj})$	Endpoint of the k^{th} non-availability interval on machine M_j
$h(N_{kj})$	Length of the k^{th} non-availability interval on machine M_j: $h(N_{kj}) = f(N_{kj}) - s(N_{kj})$
r_{ij}	Maximal possible enlargement of the processing time of job J_i on machine M_j such that Johnson's permutation is not changed
ρ_j	Stability radius of Johnson's permutation on machine M_j
A_{kj}	k^{th} availability interval on machine M_j
s_j	Earliest possible starting time of a job on machine M_j
c_j	Latest possible completion time of a job on machine M_j
d_{ij}	Maximal possible enlargement of scheduling time of operation O_{ij} forced by the non-availability intervals of machine M_j
δ_j	Enlargement radius of scheduling times of operations on machine M_j
$s_{ij}(\pi_k)$	Starting time of operation O_{ij} in schedule defined by permutation π_k
$c_{ij}(\pi_k)$	Completion time of operation O_{ij} in schedule defined by permutation π_k
s_{ik}	Setup time of job J_i on machine M_k
l_{ij}	Lower bound for setup time of job J_i on machine M_k
u_{ij}	Upper bound for setup time of job J_i on machine M_k
$[j, k]$	Indication of the job in position j of the permutation of jobs processed on machine M_k, e.g., $p_{[ik]}$ denotes processing time of this job
ST	Polytope of feasible vectors $\overline{s} = (s_{1,1}, s_{1,2}, \ldots, s_{n1}, s_{n2})$ of setup times

Chapter 5

Complexity and Computational Results

In Sections 5.1 - 5.4 of this chapter, we present computational results for randomly generated job shop problems $\mathcal{J}//\mathcal{C}_{max}$ and $\mathcal{J}//\sum\mathcal{C}_i$. In Section 5.1, we develop algorithms for calculating the stability radii $\widehat{\varrho}_s(p)$ and $\overline{\varrho}_s(p)$ on the basis of the formulas from Chapter 1. Section 5.2 investigates the influence of errors and possible changes of the processing times on the property of a schedule to be optimal. To this end, extensive numerical experiments with randomly generated job shop problems were performed. Computational results for randomly generated job shop problems $\mathcal{J}m/n=k, a_i\leq p_i\leq b_i/\mathcal{C}_{max}$ and $\mathcal{J}m/n=k, a_i\leq p_i\leq b_i/\sum\mathcal{C}_i$ with uncertain numerical input data by testing exact and heuristic algorithms derived in Section 3.6 are discussed in Section 5.4. All algorithms were coded in Fortran-77 and tested on a PC.

5.1 Calculation of the Stability Radius

This section is devoted to the calculation of the stability radius of an optimal schedule for a job shop problem, when the objective is to minimize mean or maximum flow times. The used approach may be regarded as an *a posteriori* stability analysis when an optimal schedule has already been constructed and the question is to determine such changes in the processing times of operations, which do not destroy the optimality of the schedule at hand.

We investigate the influence of errors and possible changes of the processing times on the property of a schedule to be optimal. To this end, extensive numerical experiments with randomly generated job shop problems are performed. Due to the developed software, we have the possibility to compare the values of the stability radii, the numbers of optimal schedules and some other 'numbers' for the two criteria $\mathcal{C}_{max}$ and $\sum\mathcal{C}_i$. The main question we try to answer is how large the stability radius is, on average, for randomly generated job shop problems.

The formulas for calculating the stability radii $\widehat{\varrho}_s(p)$ and $\overline{\varrho}_s(p)$ of an optimal digraph $G_s(p)$, derived in Chapter 1, were coded in Fortran-77. Due to these formulas (1.28), (1.29) and (1.36), (1.37), the calculation of the stability radii based on a direct comparison of the paths of an optimal digraph G_s and of each feasible digraph $G_k\in\Lambda(G), k\neq s$, for $\mathcal{C}_{max}$

and subsets of paths of G_s and of $G_k \in \Lambda(G), k \neq s$, for $\sum \mathcal{C}_i$ is time-consuming (even for the small Example 1.1 it is only for the makespan criterion possible to do this calculation 'by hand' without a computer). Nevertheless, such an 'unpractical' calculation for sample problems allows to derive some properties of the job shop problems, which may be used in practically efficient methods for determining lower and/or upper bounds for the stability radii. Computational results for randomly generated job shop problems are presented in this chapter. The stability radii have been calculated for more than 10,000 randomly generated job shop problems.

Next, we present the formal algorithm for calculating the stability radius $\widehat{\varrho}_s(p)$ on the basis of the coded formulas (1.28) and (1.29). We calculate the set of stability radii $\widehat{\mathcal{R}} = \{\widehat{\varrho}_1(p), \widehat{\varrho}_2(p), \ldots, \widehat{\varrho}_{opt}(p)\}$ for the set of all optimal digraphs $G_1(p), G_2(p), \ldots, G_{opt}(p)$ from the set $\Lambda(G)$ generated from a weighted mixed graph (Q, A, E). Here *opt* indicates the number of optimal schedules.

Algorithm *RAD_*$\widehat{\varrho}_s(p)$

Input: A weighted mixed graph $G(p) = (Q, A, E)$ with
a vector $p \in R^q_+$ of job processing times.

Output: The set $\widehat{\mathcal{R}}$ of the stability radii for all optimal digraphs.

Step 1: Construct the set of all feasible digraphs
$\Lambda(G) = \{G_1(p), G_2(p), \ldots, G_{opt}(p), \ldots, G_\lambda(p)\}$ generated from
mixed graph $G(p) = (Q, A, E)$ numbered in non-decreasing order of
their makespans: $l^p_1 = l^p_2 = \ldots = l^p_{opt} < l^p_{opt+1} \leq l^p_{opt+2} \leq \ldots \leq l^p_\lambda$.
Set $\widehat{\mathcal{R}} = \emptyset$.

IF $opt = 1$ **THEN** $s = 1$ **GOTO** *Step 4*.

Step 2: **FOR** $s = 1$ **TO** *opt* **DO**

BEGIN

Step 3: **IF** there exists a path $\mu^* \in H_s(p)$ such that for digraph
$G_k(p), k \neq s, k \leq opt$, $l^p_s = l^p_k$, there does not exist
path $\nu^* \in H_k(p)$ with $[\mu^*] \subseteq [\nu^*]$
THEN $\widehat{\varrho}_s(p) = 0$; $\widehat{\mathcal{R}} := \widehat{\mathcal{R}} \cup \{\widehat{\varrho}_s(p)\}$
IF $s < opt$ **THEN GOTO** *Step 2* **ELSE STOP**
ELSE

Step 4: $\widehat{\varrho}_s(p) := \infty$
IF conditions of Theorem 1.3 hold for digraph $G_s(p)$
THEN $\widehat{\mathcal{R}} := \widehat{\mathcal{R}} \cup \{\widehat{\varrho}_s(p)\}$
IF $s < opt$ **THEN GOTO** *Step 2* **ELSE STOP**
ELSE

Step 5: **FOR** $k = 1, k \neq s$ **TO** λ **DO**

BEGIN

Step 6: Construct set $H_{sk} = \{\mu \in H_s :$ There is no path $\nu \in H_k$ such that $[\mu] \subseteq [\nu]\}$.
IF $H_{sk} = \emptyset$
Step 7: **IF** $k = \lambda$
IF $H_{st} = \emptyset$ for each digraph $G_t(p), t \neq s, t \in \{1, 2, \ldots, \lambda\}$
THEN $\widehat{\mathcal{R}} := \widehat{\mathcal{R}} \cup \{\widehat{\varrho}_s(p)\}$
IF $s < opt$ **THEN GOTO** *Step 2* **ELSE STOP**
ELSE
ELSE GOTO *Step 5*
ELSE
$r_k = 0$
Step 8: **FOR** $\mu \in H_{sk}$ **DO**

BEGIN
Step 9: **FOR** $\nu \in H_k :\ l^p(\nu) \geq l^p_s$ **DO**

BEGIN
Step 10: Set $r_\beta = 0$.
Construct a sequence $(p^{\nu\mu}_{(0)}, p^{\nu\mu}_{(1)}, \ldots, p^{\nu\mu}_{(w_{\nu\mu})})$, where $p^{\nu\mu}_{(0)} = 0$ and $(p^{\nu\mu}_{(1)}, p^{\nu\mu}_{(2)}, \ldots, p^{\nu\mu}_{(w_{\nu\mu})})$ is a non-decreasing sequence of the processing times of the operations from the set $[\nu] \setminus [\mu]$ with $w_{\nu\mu} = |[\nu] \setminus [\mu]|$.

FOR $\beta = 0$ **TO** $\omega_{\nu\mu}$ **DO**
BEGIN
$r_\beta = \max\left\{r_\beta, \frac{l^p(\nu) - l^p(\mu) - \sum_{\alpha=0}^{\beta} p^{\nu\mu}_{(\alpha)}}{|[\mu]+[\nu]|-\beta}\right\}$
END
END
$r_k := \max\{r_k, r_\beta\}$
END
Step 11: $\widehat{\varrho}_s(p) := \min\{\widehat{\varrho}_s(p), r_k\}$
Step 12: **FOR** $k := k + 1$ **TO** $\lambda + 1$ **Do**
BEGIN
IF $\widehat{\varrho}_s(p) > \frac{l^p_k - l^p_s}{q}$ **THEN GOTO** *Step 5*
END

$\widehat{\mathcal{R}} := \widehat{\mathcal{R}} \cup \{\widehat{\varrho}_s(p)\}$
IF $s < opt$ **THEN GOTO** *Step 2* **ELSE STOP**
END
END STOP.

At the worst, the calculation of $\widehat{\varrho}_s(p)$ (in just the same way, as the calculation of $\overline{\varrho}_s(p)$) implies not only to have an optimal digraph $G_s(p)$, which already is an NP-hard problem, but to construct all feasible digraphs $G_1(p), G_2(p), \ldots, G_\lambda(p)$ (see Step 1) and for each of

them, which has to be compared with the optimal digraph, Algorithm $RAD\_\widehat{\varrho}_s(p)$ finds all dominant paths (see Step 6) introduced in Definition 1.3 at page 17. We can avoid such a time-consuming comparison in the following cases.

First, if there are two or more optimal digraphs it is possible that the stability radius of one of them or the radii of both are equal to zero (see Theorem 1.1). In Step 3, we check such a condition for $\widehat{\varrho}_s(p) = 0$. Second, there are two cases of an infinitely large stability radius $\widehat{\varrho}_s(p) = \infty$. One of them follows from the graph construction: There is identified a problem class whose optimal solutions are implied only by the given structural input data and even independently from the numerical input data (see Theorem 1.3). Thus, the necessary and sufficient conditions of Theorem 1.3 for an infinitely large stability radius $\mathcal{J}//\mathcal{C}_{max}$ can be verified in polynomial time $O(q^2)$ in Step 4 (q is the number of operations, $q = |Q|$). The second condition for an infinitely large stability radius follows directly from Theorem 1.2 and it is checked in Step 7. More exactly, from Theorem 1.2 and the definition of the set H_{sk} it follows that, if $H_{sk} = \emptyset$ for each feasible digraph $G_k(p), k \neq s$, then $\widehat{\varrho}_s(p) = \infty$.

From Step 7 to Step 11, we calculate the value $\widehat{r}_{ks}$ according to formula (1.29). In Algorithm $RAD\_\widehat{\varrho}_s(p)$, the value $\widehat{r}_{ks} := \widehat{\varrho}_s(p) := \min\{\widehat{\varrho}_s(p), r_k\}$ is finally defined in Step 11. To restrict the number of digraphs G_k with which an optimal digraph has to be compared, Algorithm $RAD\_\widehat{\varrho}_s(p)$ uses the bounds (3.35) in Step 12.

Using Algorithm $RAD\_\widehat{\varrho}_s(p)$, we construct a set $\widehat{\mathcal{R}} = \{\widehat{\varrho}_1(p), \widehat{\varrho}_2(p), \ldots, \widehat{\varrho}_{opt}(p)\}$ of the relative stability radii. As it follows from Remark 3.4, this algorithm is more effective than Algorithm $SOL\_\mathcal{C}_{max}(2)$. So, if $\widehat{\mathcal{R}}$ is not a single-element set, then a decision-maker can use one of the optimal digraphs $G_s(p), s = 1, 2, \ldots, opt$, which is more stable, i.e. a schedule with the largest value of the stability radius $\widehat{\varrho}_1(p) \in \widehat{\mathcal{R}}$.

Next, we present the formal algorithm for the calculation of the stability radii $\overline{\varrho}_s(p)$, which uses the formulas (1.36) and (1.37) derived for the job shop problem.

Algorithm $RAD\_\overline{\varrho}_s(p)$

Input: A weighted mixed graph $G(p) = (Q, A, E)$ with a vector $p \in R^q_+$ of processing times.

Output: The set $\overline{\mathcal{R}}$ of the stability radii for all optimal digraphs.

Step 1: Construct the set of all feasible digraphs $\Lambda(G) = \{G_1(p), G_2(p), \ldots, G_{opt}(p), \ldots, G_\lambda(p)\}$ generated from a weighted mixed graph (Q, A, E) and numbered in non-decreasing order of the mean flow time objective function values: $L^p_1 = L^p_2 = \ldots = L^p_{opt} < L^p_{opt+1} \leq L^p_{opt+2} \leq \ldots \leq L^p_\lambda$; set $\overline{\mathcal{R}} = \emptyset$.

IF $opt = 1$ **THEN** set $s = 1$ **GOTO** *Step 4*;

Step 2: **FOR** $s = 1$ **TO** *opt* **DO**

begin

Step 3: **IF** there exists a set $\Omega^{v^*}_s \in \Omega_s(p)$ such that for any set $\Omega^{u^*}_k \in \Omega_k(p)$,

there exists an operation $O_{ij} \in Q$ such that condition $n_{ij}(\Omega_s^{v^*}) \geq n_{ij}(\Omega_k^{u^*})$ (or condition $n_{ij}(\Omega_s^{v^*}) \leq n_{ij}(\Omega_k^{u^*})$) holds and this inequality has the sign $>$ (or $<$) for at least one set $\Omega_k^{u^0} \in \Omega_k(p)$
THEN set $\overline{\varrho}_s(p) = 0$ and $\overline{\mathcal{R}} := \overline{\mathcal{R}} \cup \{\overline{\varrho}_s(p)\}$;
IF $s < opt$ **THEN GOTO** *Step 2*; **ELSE stop**;
ELSE

Step 4: set $\overline{\varrho}_s(p) := \infty$;

Step 5: **FOR** $k = 1, k \neq s$ **TO** λ **DO**

begin

Step 6: construct the set $\Omega_{sk} = \{\Omega_s^v :$ There does not exist a set Ω_k^u such that $n_{ij}(\Omega_s^v) \leq n_{ij}(\Omega_k^u)$ for each $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, n_i\}$;
set $r_k = 0$;

Step 7: **FOR** $v = 1$ **TO** ω_s **DO**

begin

Step 8: **FOR** each $\Omega_k^u \in \Omega_k, u = 1, 2, \ldots, \omega_k$, with $\sum_{\nu \in \Omega_k^u} l^p(\nu) \geq L_s^p$ **DO**

begin

Step 9: set $r_\beta = 0$;
order the set of operations Q in the following way: $O_{ij_{(1)}}, O_{ij_{(2)}}, \ldots, O_{ij_{(m)}}, O_{ij_{(m+1)}}, \ldots, O_{ij_{(q)}}$, where for all $\alpha = 1, 2, \ldots, m$ inequality $n_{ij_{(\alpha)}}(\Omega_k^u) \leq n_{ij_{(\alpha)}}(\Omega_s^v)$ holds and for each $\alpha \in \{m+1, m+2, \ldots, q\}$ the inequalities $n_{ij_\alpha}(\Omega_k^u) > n_{ij_\alpha}(\Omega_s^v)$ and $p_{ij_{(m+1)}} \geq p_{ij_{(m+2)}} \geq \ldots \geq p_{ij_{(q)}}$ have to be satisfied;
FOR $\beta = 0$ **TO** $q - m$ **DO**
begin
$r_\beta = \max\left\{r_\beta, \frac{\sum_{\alpha=1}^{m+\beta} p_{ij_{(\alpha)}}(n_{ij_{(\alpha)}}(\Omega_k^u) - n_{ij_{(\alpha)}}(\Omega_s^v))}{\sum_{\alpha=1}^{m+\beta} |n_{ij_{(\alpha)}}(\Omega_k^u) - n_{ij_{(\alpha)}}(\Omega_s^v)|}\right\}$
end
end
set $r_k := \max\{r_k, r_\beta\}$;
end

Step 10: set $\overline{\varrho}_s(p) := \min\{\overline{\varrho}_s(p), r_k\}$;

Step 11: **FOR** $k := k + 1$ **TO** $\lambda + 1$ **DO**
begin
IF $\overline{\varrho}_s(p) > \frac{L_k^p - L_s^p}{nq - n}$ **THEN GOTO** *Step 5*
end

$\overline{\mathcal{R}} := \overline{\mathcal{R}} \cup \{\overline{\varrho}_s(p)\}$;
IF $s < opt$ **THEN GOTO** *Step 2*; **ELSE stop**
end

end stop.

If there exist at least two optimal schedules, i.e. if $opt > 1$, we verify in Step 3 the condition for a zero stability radius on the basis of Theorem 1.6. In Step 4, we set $\overline{\varrho}_s(p) := \infty$ (note that $\overline{\varrho}_s(p) < \infty$ due to Theorem 1.7 and Remark 1.1). Theorem 1.5 is used for the calculation of the stability radius $\overline{\varrho}_s(p)$, $0 < \overline{\varrho}_s(p) < \infty$, for each optimal digraph $G_s, s = 1, 2, \ldots, opt$, (see Steps 6 - 10). In Step 11, we can reduce the set of digraphs in our considerations due to Lemma 3.8.

Both above formal algorithms were coded in Fortran-77. So, for a small problem size the program starts with generating all feasible digraphs and for each of them, which has to be compared with the optimal digraph, it finds dominant paths (see Definition 1.3). Then formulas (1.28) and (1.29) from Section **??** are used for calculating $\widehat{\varrho}_s(p)$ and formulas (1.36) and (1.37) from Section 1.4 are used for calculating $\overline{\varrho}_s(p)$. To restrict the number of digraphs G_k with which an optimal digraph has to be compared, we use the bound (3.35) (see Chapter 3) for the makespan criterion and the bound (3.64) (see Chapter **??**) for the mean flow time criterion.

Note that the software developed is rather general. In principle, it allows to calculate the exact or approximate values of $\widehat{\varrho}_s(p)$ and $\overline{\varrho}_s(p)$ for most scheduling problems (since there exists a possibility to represent them as extremal problems on a mixed graph, see Section 1.1). The only 'theoretical' requirement for such problems is the prohibition of preemptions of operations (see Assumption 2). However, in the simulation study we are forced to take into account also 'practical' requirements: The running time and the memory of the computers. Remind that the most critical parameter of the problem under consideration is the number of edges in the mixed graph G because the whole number of feasible (without a circuit) and infeasible (with circuit) digraphs generated by G is equal to $2^{|E|}$. Moreover, for each feasible digraph G_k, we have to find all dominant paths for C_{max} and (what is essentially larger) all subsets of the set of dominant paths for $\sum C_i$.

5.2 Experimental Design and Results

In this section, computations were restricted to job shop problems. We considered three different levels of the simulation study in dependence on running time and memory limits. The stability region of the optimal digraph G_s (the whole set of non-negative q-dimensional vectors, for which G_s is optimal) is a closed cone [84, p. 326]. Indeed, if G_s is optimal for the vector $p \in R^q_+$ of the processing times, it remains optimal for the processing times $\alpha p_{11}, \alpha p_{12}, \ldots, \alpha p_{nn_n}$ with any real $\alpha > 0$ (obviously, the stability radius is the largest radius of a stability ball, which is fully contained in the stability region). So, when considering the influence of 'load leveling' factors (numbers and distributions of operations per machines and per jobs) to the stability radius, we consider the same range of variations of the processing times for the problems of the first level: The processing times of the operations are uniformly distributed real numbers (with four digits after the decimal point) between the same bounds 10 and 100.

First, we generated small instances with 12 operations in each case, for which the exact values of the stability radii $\widehat{\varrho}_s(p)$ and $\overline{\varrho}_s(p)$ may be calculated on a PC 386 usually within some seconds using only internal memory of the computer. For each combination

of the number of jobs from 3 to 7 and of the number of machines from 4 to 8, we randomly generated and solved 50 instances. Moreover, at the first level simulation includes four different types of problems in dependence on the distribution of the number of operations to the machines (evenly or randomly) and the operations, distributed to the same machine, to the jobs (evenly or randomly). Thus, we consider at the first level problems of the four types

EE (evenly, evenly),
ER (evenly, randomly),
RE (randomly, evenly) and
RR (randomly, randomly).

At the first level, we calculated the stability radii for 5000 job shop problems ($4 \cdot 5 \cdot 5 \cdot 50 =$ 4 (types) $\cdot$ 5 (combinations of the number of jobs n) $\cdot$ 5 (combinations of the number of machines m) $\cdot$ 50 (randomly generated instances in each series)) with 12 operations in each instance. Note that, if there were two or more optimal schedules for a sample problem, we calculated the stability radius for each of them.

After solving the above problems (without using external memory on a hard disk), we considered series of instances for each combination of the number of jobs from 8 to 10 and of the number of machines from 4 to 8, and for each combination of the number of jobs from 3 to 10 and $m = 3$. The number of operations in each instance was equal to 12. Since the number of edges in the mixed graph exceeded 20 (and so the number of generated feasible and infeasible digraphs exceeded $2^{20} = 1,048,576$), we had to use external memory on a hard disk for such instances and the running time for some of them achieved one or even two hours on a PC 486. So, we were forced to restrict the number of considered instances in the most difficult series for such combinations of the numbers of jobs and the numbers of machines to 10.

On the basis of the obtained information within the first level of experiments (for the instances with 12 operations), we designed the second and third ones. First, we decided to consider only instances generated for an evenly distributed number of operations to the machines and evenly distributed operations on the same machine to the jobs (i.e. type EE). At the second level, we calculated the exact values of the stability radii for job shop problems with 16 and 20 operations, considering 10 instances in each series while considering the influence of 'load leveling' factors. Note that for some of the instances at the second level, the CPU time of a Pentium PC exceeded 10 hours.

Along with 'load leveling' factors, other ones also influence the complexity and stability of scheduling problems, e.g. the variability of $p_{ij}, O_{ij} \in Q$, across the entire shop and the variability of the average processing time from job to job or from machine to machine are also important factors of the complexity of shop scheduling problems (remind the famous job shop problem with 10 jobs and 10 machines given in [FT63], which was so difficult to attain due to a special processing time variability). Therefore, at the second level we also investigated the influence of the latter factors for random modifications of the processing times of the job shop problem with the same mixed graph G. More precisely, for the same randomly generated mixed graph G (see Chapter 1) at the second level of the simulation study, we considered six different ranges of variations of the given processing times, namely: [1, 10], [1, 100], [1, 1000], [10, 100], [10, 1000] and [100, 1000].

Figure 5.1: *Maximal, average and minimal values of $\widehat{\varrho}_s(p)$ for the problems of type EE*

Figure 5.2: *Maximal, average and minimal values of $\overline{\varrho}_s(p)$ for the problems of type EE*

Figure 5.3: *Maximal, average and minimal values of $\widehat{\varrho}_s(p)$ for the problems of types ER, RE and RR*

Obviously, intervals [10, 100] and [100, 1000] may be obtained from the interval [1, 10] after multiplying with 10 and 100, respectively. However, the number of optimal schedules, and the number of problems with a zero value of stability radii may be different for these three intervals, since we consider all real numbers with fixed number of decimal places. Due to the same reason, we consider the intervals [1, 100] and [10, 1000]. For the above segments, we calculated $\widehat{\varrho}_s(p)$ and $\overline{\varrho}_s(p)$ for each optimal schedule s in series with 50 instances. Moreover, we investigated instances in which different jobs had different ranges of variations of the given processing times. At the third level, we considered a well-known job shop test problem with 6 jobs and 6 machines from [FT63] with different ranges of variations of the given processing times across the entire shop and across different jobs.

In Figures 5.1 and 5.2, we present the maximal, average and minimal values of the stability radii for each combination of the number of jobs n and the number of machines m, considered at the first level when the number of operations are evenly distributed to the machines and the operations on a machine are evenly distributed to the jobs (type EE). While the processing times are real numbers between 10 and 100, the stability radii are approximately between 0.001 and 50 for $\mathcal{C}_{max}$ and between 0.001 and 35 for $\sum \mathcal{C}_i$. Similar data for the other three types of distributing the operations are given in Figures 5.3 and 5.4 (types ER, RE and RR). The largest value of $\widehat{\varrho}_s(p)$ was about 90, and the largest value of $\overline{\varrho}_s(p)$ was about 70. For all types EE, ER, RE and RR, the average value of $\widehat{\varrho}_s(p)$ was larger than that of $\overline{\varrho}_s(p)$. An obvious conclusion from these diagrams is that an optimal makespan schedule (Figures 5.1 and 5.3) is more stable than an optimal mean flow time schedule (Figures 5.3 and 5.4). An important issue from Figures 5.1 - 5.4 is also that for each series of instances the smallest value of $\widehat{\varrho}_s(p)$ and $\overline{\varrho}_s(p)$ is greater than zero.

The results for the sample problems of the second level for 'load leveling' factors are presented in Table 5.1, where the minimal (MIN), average (AVE) and maximal (MAX) values of the stability radius divided by the maximal processing times (p_{MAX}) are given in columns 2, 3, and 4, and similar values divided by the average processing times (p_{AVE}) are given in columns 5, 6, and 7. During our experiments, we also determined the largest number γ of competitive digraphs in the sequence $(G_{i_1}, G_{i_2}, \ldots, G_{i_\gamma}, \ldots, G_{i_{\lambda'}}, \ldots, G_{i_\lambda})$ (where the digraphs are ordered according to non-decreasing objective function values) and the number λ' of the digraph, which was the last considered one in this sequence, while calculating the exact value of the stability radius. Column 8 contains the average values of the percentage of digraphs, which may be a competitive digraph for the optimal one $(100t/\lambda)$. For the set of instances presented in Table 5.1 with the mean flow time criterion,

Figure 5.4: *Maximal, average and minimal values of $\overline{\varrho}_s(p)$ for the problems of types ER, RE and RR*

Table 5.1: *Randomly generated problems*

n x m	RADIUS / p_{MAX}			RADIUS / p_{AVE}			$\frac{100\gamma}{\lambda}$	NOS		DIFF	
	MIN	AVE	MAX	MIN	AVE	MAX		AVE	MAX	AVE	MAX
1	2	3	4	5	6	7	8	9	10	11	12
Maximum flow time											
6 x 6	0.01	0.62	4.26	0.02	1.09	7.40	-	21.50	78	1.60	3.83
7 x 7	0.07	1.76	11.16	0.12	3.45	23.59	-	15.60	43	2.71	9.72
8 x 8	0.07	3.43	12.66	0.13	6.00	17.80	-	17.00	70	4.91	12.20
9 x 9	0.00	3.97	11.52	0.00	6.91	22.14	4.43	28.90	144	6.07	11.38
10 x 10	0.18	3.33	21.90	0.32	5.97	41.38	1.54	12.40	48	5.68	18.84
Mean flow time											
6 x 6	0.33	1.28	5.10	0.59	2.25	8.67	2.27	1.10	2	0.00	0.00
7 x 7	0.23	1.33	6.57	0.40	2.32	11.19	2.42	1.20	2	0.00	0.00
8 x 8	0.26	1.86	6.54	0.51	3.28	11.85	0.03	1.20	2	0.00	0.00
9 x 9	0.60	2.20	4.41	1.10	3.84	8.22	0.10	1.10	2	0.00	0.00
10 x 10	0.46	3.83	8.05	0.75	6.79	13.69	0.57	1.00	1	0.00	0.00

these values are bounded by 2.42%. When minimizing the makespan, these values are larger, but the latter results are mostly due to the large numbers of optimal makespan schedules (the average and maximal numbers NOS of optimal semiactive schedules for an instance are given in columns 9 and 10, respectively). Note that for some 6 x 6 instances (i.e. those with 6 jobs and 6 machines), 7 x 7 instances and 8 x 8 instances, the number λ of all semiactive schedules was not calculated in our experiments, and therefore the values of $100\gamma/\lambda$ are not presented for these series. If there is more than one optimal schedule, we calculate the differences of their stability radii. The average and maximal values of these differences (DIFF) are presented in columns 11 and 12, respectively. We can also note that for the mean flow time criterion, an optimal schedule is usually uniquely determined, and even if there are two optimal mean flow time schedules, they have often the same stability radius. Consequently, for the mean flow time criterion we have not much need to look for an optimal schedule with the largest stability radius.

Next, we present the randomly generated mixed graph G for the job shop problem $\mathcal{J}6/n = 4/\Phi$ with 4 jobs and 6 machines, which is used for the simulation study of the influence of the variability of the processing times: $Q = \{O_{11}, O_{12}, \ldots, O_{44}\}$; $E = \{[O_{11}, O_{21}], [O_{11}, O_{34}], [O_{21}, O_{34}]; \ [O_{12}, O_{23}]; \ [O_{13}, O_{22}], \ [O_{13}, O_{42}], [O_{22}, O_{42}]; \ [O_{14}, O_{32}], [O_{14}, O_{41}], [O_{32}, O_{41}]; \ [O_{24}, O_{31}], \ [O_{24}, O_{44}], [O_{31}, O_{44}]; \ [O_{33}, O_{43}]\}$ (see Figure 5.5). Computational results for this mixed graph are given in Table 5.2 and Table 5.3. Table 5.2 presents the computational results for different ranges of the processing times for the same mixed graph G, which is described above. Note also that both criteria $\mathcal{C}_{max}$ and $\sum \mathcal{C}_i$ are considered for the same 50 examples for which the obtained results are presented in the corresponding rows of Table 5.2 (row i for $\mathcal{C}_{max}$ corresponds to row $i + 6$ for $\sum \mathcal{C}_i$).

Table 5.3 presents the computational results for different ranges of the values p_{ij} of the operations of different jobs. Along with the columns defined for Table 5.1, we also present the percentage of considered digraphs while calculating the exact value of the stability radius (column 10 in Table 5.2 and column 16 in Table 5.3) and the number NMO of problems with two or more optimal schedules (column 13 in Table 5.2 and column 19 in

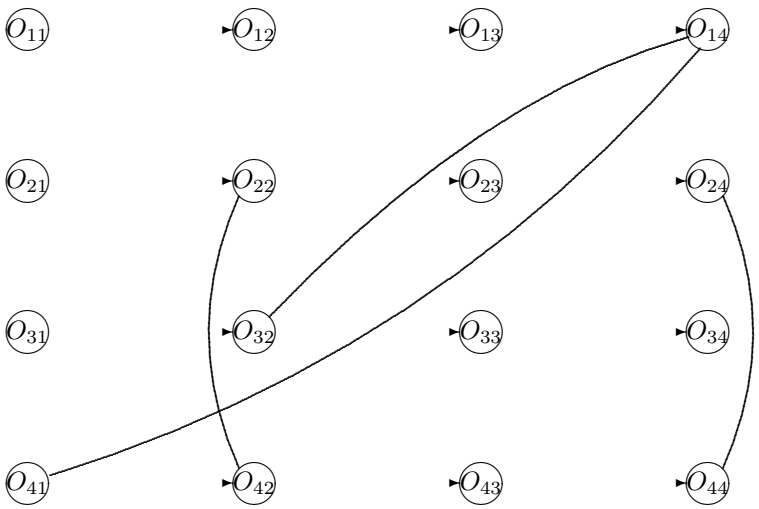

Figure 5.5: *Randomly generated mixed graphs for problem* $\mathcal{J}6/n=4/\Phi$

Table 5.2: *Problem* $\mathcal{J}6/n=4/\Phi$, $\Phi \in \{\mathcal{C}_{max}, \sum \mathcal{C}_i\}$, *with different ranges of variations of* p_{ij}

Bounds for p_{ij}		RADIUS/p_{MAX}			RADIUS/p_{AVE}			$\frac{100\gamma}{\lambda}$	$\frac{100\lambda'}{\lambda}$	NOS		NMO	DIFF	
LB	UB	MIN	AVE	MAX	MIN	AVE	MAX			AVE	MAX		AVE	MAX
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Maximum flow time														
1	10	0.07	2.37	8.48	0.13	4.40	17.15	0.48	2.93	2.80	12	34	0.15	0.75
1	100	0.02	2.31	12.11	0.04	4.91	27.38	0.71	5.02	3.58	21	34	2.35	10.53
1	1000	0.00	3.63	13.83	0.00	8.33	36.71	0.73	8.66	4.26	36	32	30.93	132.28
10	100	0.13	2.52	10.78	0.26	4.64	21.18	0.50	4.71	2.74	12	31	2.36	9.53
10	1000	0.01	3.08	13.06	0.01	6.23	31.36	0.62	8.07	4.60	30	41	26.90	123.57
100	1000	0.04	2.88	11.89	0.07	5.20	24.85	0.53	4.18	2.30	12	25	20.63	108.90
Mean flow time														
1	10	0.06	2.56	10.17	0.11	4.76	17.89	0.30	3.87	1.02	2	1	0.13	0.13
1	100	0.07	2.47	9.90	0.13	5.06	20.54	0.34	4.44	1.02	2	1	0.00	0.00
1	1000	0.03	2.07	10.21	0.05	3.95	16.96	0.30	3.40	1.04	2	2	1.80	3.59
10	100	0.12	2.41	7.30	0.21	4.51	15.63	0.30	3.42	1.00	1	0	-	-
10	1000	0.17	2.26	8.67	0.37	4.52	16.29	0.32	3.40	1.12	2	6	6.18	22.48
100	1000	0.05	2.53	11.43	0.11	4.48	19.78	0.33	4.22	1.04	2	2	16.37	32.75

Table 5.3).

For the problems considered at the second level, the 'superiority' of the stability radius for the makespan criterion is lost in most cases. At least the minimal values of $\overline{\varrho}_s(p)$ became larger than those of $\widehat{\varrho}_s(p)$. Of course, the large number of optimal makespan schedules has influenced this relation essentially, but even on average, we cannot find a large superiority of the stability radius of one criterion over the other (for the considered classes of randomly generated job shop problems).

Next, we discuss some questions on the basis of our experimental calculation of the

stability radii of the optimal schedules for small randomly generated job shop problems.

How often is the stability radius equal to zero? In the experiments at the first and the second levels, we obtained only once a stability radius equal to zero for criterion $\mathcal{C}_{max}$ and never for criterion $\sum \mathcal{C}_i$ although it takes not much effort to construct such an example by hand (see Theorem 1.1 for $\mathcal{C}_{max}$ and Theorem 1.6 for $\sum \mathcal{C}_i$). So, in principle, to find an optimal schedule for almost all problems generated in our experiments has sense. On the other hand, in many series there are instances with very small values of the stability radius (even less than 0.001). So, if for such an instance the precision of the processing times is not sufficiently high, we have no guarantee that the (a priori) constructed optimal schedule will be indeed the best one in its practical realization.

May the stability radius be infinitely large? From theoretical results it follows that for any given n and m, there exist job shop problems with an optimal makespan schedule s, which remains optimal for any feasible variation of the processing times, i.e. $\widehat{\varrho}_s(p) = \infty$ (see Theorem 1.2 and Theorem 1.3). In particular, an easily verifiable characterization of such a schedule has been derived for criterion $\mathcal{C}_{max}$ (see Theorem 1.3). In contrast, it was shown that for mean flow time, we have $\overline{\varrho}_s(p) \leq \max_{O_{ij} \in Q}\{p_{ij}\}$ for any job shop problem (see Theorem 1.7). Although in [KSW95] a practical example of an infinitely large stability radius was presented (for a traffic-light on the intersection of two roads), nevertheless such a shop appears to be rather artificial for large numbers of jobs and machines. Surprisingly, in our randomly generated job shop problems with the makespan criterion an infinitely large stability radius was obtained not seldom, at least essentially more often than a zero stability radius (of course, we did not include infinite stability radii while calculating the average and maximal values of $\widehat{\varrho}_s(p)$). So, our experiments indicate that the results derived in [KSW95, Sot91b] will have not only theoretical significance.

How much 'best' schedules do we need to consider? As already mentioned, we also determined the number γ of competitive digraphs and the number λ' of considered digraphs, while calculating the exact value of the stability radius. For the problems of the first level, the diagrams for the percentage of the numbers γ and for the percentage of the numbers λ' for the problems of type EE are presented in Figure 5.6 (Figure 5.7) for criterion $\mathcal{C}_{max}$ (for criterion $\sum \mathcal{C}_i$, respectively). In the front part of the diagrams in Figures 5.6 and 5.7, the minimal, average and maximal values of the percentages $100\gamma/\lambda$ are presented, while in the background of these diagrams the minimal, average and maximal values of the percentages $100\lambda'/\lambda$ are presented. As it follows from Figures 5.6 and 5.7, the value $100\gamma/\lambda$ may be smaller than 1 % and it is not greater than 73 % for $\mathcal{C}_{max}$ and not greater than 56 % for $\sum \mathcal{C}_i$.

Table 5.3: *Problem $\mathcal{J}6/n=4/\Phi$, $\Phi \in \{\mathcal{C}_{max}, \sum \mathcal{C}_i\}$, with variability of p_{ij} from job to job*

Lower bound (LB_i) and upper bound (UB_i) for the processing times of job J_i								RADIUS/p_{MAX}			RADIUS/p_{AVE}			$\frac{100\gamma}{\lambda}$	$\frac{100\lambda'}{\lambda}$	NOS		NMO	DIFF	
LB_1	UB_1	LB_2	UB_2	LB_3	UB_3	LB_4	UB_4	MIN	AVE	MAX	MIN	AVE	MAX			AVE	MAX		AVE	MAX
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Maximum flow time																				
10.	20.	30.	40.	70.	80.	90.	100.	0.01	0.87	3.81	0.01	1.59	6.92	3.83	9.15	29.90	54	50	0.48	3.24
10.	20.	40.	50.	60.	70.	90.	100.	0.00	0.98	4.37	0.01	1.79	8.06	1.10	5.86	7.64	16	50	0.57	3.54
10.	30.	30.	50.	60.	80.	80.	100.	0.01	1.41	6.62	0.02	2.56	12.01	1.70	7.95	15.48	32	50	1.28	5.05
10.	40.	30.	60.	50.	80.	10.	100.	0.00	1.66	8.56	0.01	2.99	15.59	1.21	6.97	7.68	22	50	1.40	6.92
10.	60.	30.	60.	50.	80.	50.	100.	0.06	1.84	8.03	0.09	3.32	15.27	0.84	4.25	5.54	24	49	1.00	7.08
10.	60.	10.	60.	50.	100.	50.	100.	0.00	3.30	12.05	0.01	6.02	22.56	1.30	12.52	11.98	24	50	3.77	10.57
10.	40.	40.	70.	10.	100.	10.	100.	0.01	4.28	14.81	0.01	8.31	31.23	0.65	18.38	15.92	58	50	6.26	14.01
10.	30.	10.	50.	60.	80.	60.	100.	0.01	1.78	9.82	0.02	3.20	18.20	1.24	6.81	10.62	32	50	1.58	9.31
10.	20.	30.	50.	60.	80.	80.	100.	0.00	1.03	4.69	0.00	1.87	8.85	2.36	9.54	19.38	54	50	1.10	4.13
10.	25.	25.	75.	25.	75.	50.	100.	0.09	2.73	14.34	0.16	5.30	25.56	0.81	7.93	3.62	12	42	2.03	14.07
Mean flow time																				
10.	20.	30.	40.	70.	80.	90.	100.	0.08	1.08	2.97	0.14	1.95	5.36	0.27	1.56	1.00	1	0	-	-
10.	20.	40.	50.	60.	70.	90.	100.	0.04	1.21	2.82	0.07	2.21	5.22	0.26	1.28	1.00	1	0	-	-
10.	30.	30.	50.	60.	80.	80.	100.	0.02	1.15	4.11	0.04	2.08	7.33	0.31	1.64	1.00	1	0	-	-
10.	40.	30.	60.	50.	80.	70.	100.	0.09	1.54	4.85	0.15	2.78	8.02	0.31	1.78	1.00	1	0	-	-
10.	60.	30.	60.	50.	80.	50.	100.	0.05	1.87	6.14	0.09	3.35	10.85	0.35	2.39	1.00	1	0	-	-
10.	60.	10.	60.	50.	100.	50.	100.	0.01	1.45	4.41	0.01	2.62	7.53	0.32	2.02	1.04	3	1	0.00	0.00
10.	40.	40.	70.	70.	100.	10.	100.	0.07	2.32	7.83	0.12	4.23	13.62	0.33	3.40	1.04	2	2	0.00	0.00
10.	30.	10.	50.	60.	80.	60.	100.	0.04	1.31	4.66	0.08	2.34	7.97	0.33	1.51	1.00	1	0	-	-
10.	20.	30.	50.	60.	80.	80.	100.	0.01	0.94	3.26	0.01	1.70	5.95	0.25	1.14	1.00	1	0	-	-
10.	25.	25.	75.	25.	75.	50.	100.	1.01	2.18	5.51	0.02	4.17	9.92	0.31	3.27	1.00	1	0	-	-

Figure 5.6: *Percentage of the number γ (and λ') of competitive (considered) digraphs for the problems of type EE for $\widehat{\varrho}_s(p)$*

Figure 5.7: *Percentage of the number γ (and λ') of competitive (considered) digraphs for the problems of type EE for $\overline{\varrho}_s(p)$*

It should be noted that for the case of a large number of machines and a small number of operations (at the first level when $q = 12$), there often exist only a few feasible semi-active schedules which make the relative values of γ and λ' rather large. Moreover, for criterion $\mathcal{C}_{max}$, we have a relatively large number of optimal schedules which also enlarges the relative values of γ and λ'. Thus, calculating the exact value of the stability radius on the basis of bounds (3.35) and (3.64) may require to consider the whole set $\Lambda(G)$ of digraphs for the problems considered at the first level of our simulation study.

From Tables 5.2 and 5.3, it follows that the competitive digraphs are within 3.83 % of the whole set of feasible digraphs for criterion $\mathcal{C}_{max}$ and within 0.35 % for criterion $\sum \mathcal{C}_i$, and the percentage of digraphs which have been considered while calculating the stability radius is no more than 18.38 % for criterion $\mathcal{C}_{max}$ and no more than 4.44 % for criterion $\sum \mathcal{C}_i$. So, it is not necessary to construct the whole set of feasible digraphs for calculating the stability radius of an optimal digraph for these types of problems.

After studying the obtained results at the first and second levels of our experiments, we enlarged the size of problems, which are still suitable for calculating the exact value of the stability radius (or at least its upper bound). For calculating the stability radius for instances of larger size, we constructed for each of them only the k best schedules (with $k = 100$ in most cases) by a direct enumeration of the whole set of feasible digraphs. Then, considering only these k best digraphs, we intended to calculate the stability radius of an optimal digraph (or optimal digraphs). If this process has stopped before the whole k best digraphs were compared with the optimal one, we have obtained the exact value of the stability radius due to the bounds (3.35) or (3.64), otherwise we have obtained at least an upper bound for the stability radius. Moreover, to shorten the running time we used the branch-and-bound method for calculating the k best digraphs.

How can one combine this approach with the branch-and-bound method? The following approach to stability analysis for scheduling problems seems to be practically efficient. Using a branch-and-bound method (e.g. [BJS94, CP89]), one can construct not only one optimal but the k best schedules. In particular, in our computational study we used a branch-and-bound algorithm with the conflict resolution strategy. Due to an implicit enumeration of the feasible mixed graphs $G_{(s)}(p) = (Q, A_{(s)}, E_{(s)})$, we construct the k best ones and calculate the exact value or an upper bound for the stability radius of an optimal schedule in the same manner as described in the above paragraph "How much 'best' schedules do we need to consider?". Note that, while an explicit enumeration of the digraphs $G_1(p), G_2(p), \ldots, G_\lambda(p)$ gives the exact value of $\widehat{\varrho}_s(p)$ for $|E| \leq 30$, the branch-and-bound algorithm gives the possibility to calculate $\widehat{\varrho}_s(p)$ for $|E| \leq 100$ (often within the same CPU time).

In particular, at the third level of the experiments we considered the well-known classical job shop problem from [FT63] with 6 jobs and 6 machines. The assignment of

the operations $Q = \{O_{11}, O_{12}, \ldots, O_{66}\}$ to the set of machines $M = \{M_1, M_2, \ldots, M_6\}$ is as follows:

$$\begin{aligned}
Q_1 &= \{O_{12}, O_{25}, O_{34}, O_{42}, O_{55}, O_{64}\},\\
Q_2 &= \{O_{13}, O_{21}, O_{35}, O_{41}, O_{52}, O_{61}\},\\
Q_3 &= \{O_{11}, O_{22}, O_{31}, O_{43}, O_{51}, O_{66}\},\\
Q_4 &= \{O_{14}, O_{26}, O_{32}, O_{44}, O_{56}, O_{62}\},\\
Q_5 &= \{O_{16}, O_{23}, O_{36}, O_{45}, O_{53}, O_{65}\},\\
Q_6 &= \{O_{15}, O_{24}, O_{33}, O_{46}, O_{54}, O_{63}\}.
\end{aligned}$$

For this problem, each job has to be processed on each machine exactly once and hence we have $q = 6$ x $6 = 36$ and $|E| = 6$ x $\binom{6}{2} = 90$. By the branch-and-bound algorithm we constructed $k = 150$ best schedules: 22 of them are optimal with $\mathcal{C}_{max} = 55$ and 54 other schedules have a makespan value equal to 56, and at least 74 schedules have a makespan value equal to 57. We calculated an upper bound for $\widehat{\varrho}_s(p)$ for each optimal makespan schedule. It turned out that 14 of them have a zero stability radius and the other 8 optimal schedules have an upper bound for $\widehat{\varrho}_s(p)$ equal to 0.08333. The existence of unstable optimal schedules for this test problem is implied mainly by the fact that its processing times are integers from 1 to 10.

We also randomly generated 50 instances with 6 jobs, 6 machines and 36 operations. Again each job has to be processed on each machine exactly once (i.e. we considered classical job shop problems), but in contrast to the problem from [FT63], the processing times were uniformly distributed real numbers between 1 and 10. For each generated problem with 36 operations, we constructed 50 best schedules (for the makespan criterion) on the basis of the branch-and-bound algorithm and calculated upper bounds for $\widehat{\varrho}_s(p)$ for each optimal makespan schedule which was constructed. Note that 45 of these instances had more than one optimal makespan schedule and among them, 7 instances had 50 or even more optimal makespan schedules. The average value of the stability radius $\widehat{\varrho}_s(p)$ was equal to 0.12939, and for all calculated optimal makespan schedules s the following bounds were satisfied: $0.001 \leq \widehat{\varrho}_s(p) \leq 0.87455$. We calculated also the differences between $\widehat{\varrho}_s(p)$ for different optimal makespan schedules $s \in S$ of the same instance (if this instance had two or more optimal makespan schedules). The maximum of this difference was equal to 0.84636, the average difference was 0.11709 and some optimal makespan schedules had the same stability radius. Among the 50 instances, there was no optimal schedule with a zero stability radius.

To investigate the influence of the variability of the processing times p_{ij} on the stability radius, we considered again the test problem with 6 jobs and 6 machines given in [FT63], but with different distributions of the processing times to the operations. More precisely, the mixed graph $G = (Q, A, E)$ was defined in accordance with [FT63], but the processing times were randomly generated real numbers with the same lower and upper bounds for all jobs (see rows 1 - 6 in Table 5.4) and with different lower and upper bounds for different jobs (rows 7 - 10 in Table 5.4). Each row in Table 5.4 presents the results obtained for a series of 10 instances. For each instance, we calculated the stability radius using 100 best schedules generated by the branch-and-bound algorithm.

For row 7 in Table 5.4, the lower bound LB_i^1 and the upper bound UB_i^1 for job J_i are as follows:

Table 5.4: *Test problem $\mathcal{J}6/n=6/\mathcal{C}_{max}$ with variability of p_{ij}*

Bounds for p_{ij}		RADIUS/p_{MAX}			RADIUS/p_{AVE}			NOS		NMO	DIFF	
LB	UB	MIN	AVE	MAX	MIN	AVE	MAX	AVE	MAX		AVE	MAX
1	2	3	4	5	6	7	8	9	10	11	12	13
Common bounds for p_{ij} for different jobs												
1.	10.	0.0067	0.1843	0.8744	0.0138	0.3374	1.5340	13.5	52	8	0.0106	0.0393
1.	100.	0.0077	0.3265	1.2092	0.0163	0.6158	2.0630	26.7*	100*	10	0.1705	1.1278
1.	1000.	0.0749	0.6679	2.4344	0.1639	1.4461	5.1843	31.5	90	10	6.3276	22.6764
10.	100.	0.0440	0.7733	3.9813	0.0820	1.4277	7.4289	17.5*	100*	9	0.5540	3.6507
10.	1000.	0.0070	0.4436	1.7260	0.0116	0.8298	3.2587	27.4*	100*	10	3.5290	10.9608
100.	1000.	0.0308	0.5694	1.9779	0.0564	1.0109	3.8182	17.0	54	10	7.2888	18.3886
Different bounds for p_{ij} for different jobs												
LB_i^1	UB_i^1	0.0000	0.6429	3.9997	0.0000	1.1009	7.2044	41.2*	100*	10	0.9175	3.1636
LB_i^2	UB_i^2	0.0216	0.5046	1.3379	0.0383	0.8487	2.3764	5.2	12	9	0.0000	0.0000
LB_i^3	UB_i^3	0.0000	1.0051	4.2719	0.0000	1.7247	7.4870	74.6*	100*	10	1.7654	4.1433
LB_i^4	UB_i^4	0.0031	0.9716	9.2608	0.0053	1.7292	16.4208	82.2*	100*	10	2.1819	8.6476

$$
\begin{aligned}
LB_1^1 &= 10, \quad UB_1^1 = 40;\\
LB_2^1 &= 20, \quad UB_2^1 = 50;\\
LB_3^1 &= 30, \quad UB_3^1 = 60;\\
LB_4^1 &= 50, \quad UB_4^1 = 80;\\
LB_5^1 &= 60, \quad UB_5^1 = 90;\\
LB_6^1 &= 70, \quad UB_6^1 = 100.
\end{aligned}
$$

For row 8 these bounds are:

$$
\begin{aligned}
LB_1^2 &= 10, \quad UB_1^2 = 60;\\
LB_2^2 &= 30, \quad UB_2^2 = 60;\\
LB_3^2 &= 40, \quad UB_3^2 = 60;\\
LB_4^2 &= 50, \quad UB_4^2 = 70;\\
LB_5^2 &= 50, \quad UB_5^2 = 80;\\
LB_6^2 &= 50, \quad UB_6^2 = 100.
\end{aligned}
$$

For row 9 these bounds are:

$$
\begin{aligned}
LB_1^3 &= 10, \quad UB_1^3 = 40;\\
LB_2^3 &= 20, \quad UB_2^3 = 50;\\
LB_3^3 &= 40, \quad UB_3^3 = 70;\\
LB_4^3 &= 60, \quad UB_4^3 = 90;\\
LB_5^3 &= 70, \quad UB_5^3 = 100;\\
LB_6^3 &= 10, \quad UB_6^3 = 100.
\end{aligned}
$$

For row 10 these bounds are:

$$
\begin{aligned}
LB_1^4 &= 10, \quad UB_1^4 = 30;\\
LB_2^4 &= 20, \quad UB_2^4 = 40;\\
LB_3^4 &= 30, \quad UB_3^4 = 50;\\
LB_4^4 &= 60, \quad UB_4^4 = 80;\\
LB_5^4 &= 70, \quad UB_5^4 = 90;\\
LB_6^4 &= 80, \quad UB_6^4 = 100.
\end{aligned}
$$

In Table 5.4, we marked the series of instances, for which the number of optimal schedules is larger than 100 by an asterisk. Since we calculated only 100 best schedules for each instance, we had not the exact number of optimal semiactive makespan schedules.

Unfortunately, the developed software did not allow us to find $\overline{\varrho}_s(p)$ for most of the above instances with 36 operations and 90 edges since the calculation of the stability radius for the mean flow time criterion is essentially more time-consuming than for the makespan.

How to use this approach for problems of practical size? For large instances, for which a direct enumeration of all feasible digraphs was practically impossible, we constructed only a subset of feasible digraphs, selected then the best digraph G_s among them and calculated an upper bound for the 'stability radius' of G_s by a comparison with all other digraphs that have been constructed. This variant of the implementation of the software may be useful for some practical problems. Indeed, in reality OR workers have at most one or only a few feasible schedules (usually without an exact information about their quality). In the case when a set of feasible schedules is known, we can investigate the stability radius of the best of them in comparison with the others at hand.

Even if we have not the possibility to find an optimal schedule by a branch-and-bound method and only an approximate schedule (with information about its quality) or a heuristic schedule has been constructed, we can investigate the 'stability radius' of this schedule in comparison with the other $k-1$ schedules that have been constructed.

5.3 Remarks on the Stability Radii

The main issue from our experiments is that an optimal schedule is usually stable: Its stability radius is not equal to zero and so there exists a ball with the center p of the processing times in the space R_+^q of input data, within which the schedule remains optimal. Thus, such a radius may be useful as a measure of the stability of an optimal schedule. Moreover, on the basis of the above computational experiments (though limited problem sizes), one can make the conclusion that an optimal schedule for criterion $\mathcal{C}_{max}$ is often more stable than an optimal schedule for criterion $\sum \mathcal{C}_i$ when the size of the problem is small.

Moreover, our approach gives not only the exact value or a bound for the stability radius but also competitive schedules (competitive digraphs) which along with an optimal schedule have to be considered as candidates for the practical realization, when the stability radius or its upper bound is less than the possible error of the processing times known in advance.

Note that the problem of calculating the stability radius of the digraph $G_s(p)$ is NP-hard even provided that an optimal schedule s is known. It is even NP-hard to find the 'tolerances' of a single processing time p_{ij}, which do not violate the optimality of the optimal digraph. The latter result follows from [RC95] since the problem considered in that paper may be presented as a special case of the job shop problem.

Another insight is that an optimal mean flow time schedule is usually uniquely determined, while two or more optimal makespan schedules are very usual (at least in our simulation study). So, in the latter case it makes sense to look for an optimal makespan schedule with the largest value of the stability radius (the difference of the stability radii

for different optimal schedules of the same problem may be very large for the makespan criterion). Such a schedule has a better chance to be makespan optimal in its practical realization. However, this is not valid for the mean flow time criterion, for which one can be satisfied by the first constructed optimal schedule because even if there are two or more optimal mean flow time schedules, they usually have the same value (or close values) of the stability radii.

Moreover, there exist shops for which we can look for an optimal makespan schedule with an infinitely large stability radius. In particular, if one can influence the properties of the shop (i.e. technological routes of the jobs, the number of used machines and the distribution of the operations to the machines, etc.), one can design a shop that has an optimal makespan schedule with an infinitely large stability radius (see Theorem 1.3). In this case the variations of the processing times have no influence on such a schedule to be optimal. For some scheduling problems, such a property may be practically important.

Since a zero stability radius of the optimal schedule s is rather seldom, there exists an $\epsilon > 0$ such that s will remain optimal for any variations $p_{ij} \pm \epsilon$ of the processing times. In particular, this is true for almost all problems with the mean flow time criterion, which were considered in our experiments, since for these problems an optimal schedule is often uniquely determined, and as a result, it has a strictly positive stability radius. On the other hand, it has practical sense to make the error in the determination of the processing times as small as possible in order to guarantee the real optimality of a schedule at hand: Almost in all series there were schedules with very small (but strictly positive) values of the stability radii.

After the analysis of the influence of possible changes of the given processing times of the operations, i.e. the largest quantity of independent variations (stability radius) within which an optimal schedule of problems $\mathcal{J}//\mathcal{C}_{max}$ and $\mathcal{J}//\sum \mathcal{C}_i$ remains optimal, we make experimental investigations of job shop problems with *uncertain* processing times $\mathcal{J}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ and $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$, which satisfy Assumptions 1, 2 and 3 (see Introduction). These computational results are described in Section 5.4.

5.4 Problems $\mathcal{J}/a_i \leq p_i \leq b_i/\Phi,\ \Phi \in \{\mathcal{C}_{max}, \Sigma \mathcal{C}_i\}$

The algorithms derived in Section 3.6 were coded in Fortran-77 and were tested on a PC 486 (120 MHz) for the exact solution and on a PC 486 (50 MHz) for the heuristic solution of problem $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$ and on a PC 486 (133 MHz) for the exact and the heuristic solution of problem $\mathcal{J}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$. Here the term 'exact solution' is used for indicating a set $\Lambda^*(G)$ which satisfies Definition 3.5 in contrast to the 'heuristic solution' indicating a set $\Lambda \subset \Lambda(G)$ which generally may not contain an optimal schedule for each vector $p \in T$.

The experimental design was as follows. First, we considered series of instances of problem $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$ with small n and m for which an exact solution and the exact minimal solution may be calculated within one hour on a PC 486 (120 MHz). After finding upper bounds for such n and m, we started experiments with medium size problems in order to find at least their heuristic solution. From this moment, the experiments were continued simultaneously on both computers in order to find upper bounds on n and m for a 'good' heuristic solution on a PC 486 (50 MHz) (see Table 5.9), and to increase the

problem size for the exact solution on a PC 486 (120 MHz) (see Table 5.7).

Table 5.5: *Types of problems considered in the experiments*

	Types of problems	Errors of the processing times				Types of problems	
Exact solutions:	A	5%,	10%,	15%,	20%		
sets $B, \Lambda^*(G)$	B	2%,	6%,	8%,	10%	B	Heuristic
and $\Lambda^T(G)$	C	1%,	3%,	5%,	7%	C	solution:
		1%,	2%,	3%,	4%	D	set B
		0.1%,	0.2%,	0.3%,	0.4%	E	

Table 5.6: *The minimal lower and maximal upper bounds of processing times*

Errors	Lower bound	Upper bound	The actual processing time p^*_{ij}
20%	$(1-0.2)p_{ij}$	$(1+0.2)p_{ij}$	$0.8p_{ij} \le p^*_{ij} \le 1.2p_{ij}$
15%	$(1-0.15)p_{ij}$	$(1+0.15)p_{ij}$	$0.85p_{ij} \le p^*_{ij} \le 1.15p_{ij}$
10%	$(1-0.1)p_{ij}$	$(1+0.1)p_{ij}$	$0.9p_{ij} \le p^*_{ij} \le 1.1p_{ij}$
8%	$(1-0.08)p_{ij}$	$(1+0.08)p_{ij}$	$0.92p_{ij} \le p^*_{ij} \le 1.08p_{ij}$
7%	$(1-0.07)p_{ij}$	$(1+0.07)p_{ij}$	$0.93p_{ij} \le p^*_{ij} \le 1.07p_{ij}$
6%	$(1-0.06)p_{ij}$	$(1+0.06)p_{ij}$	$0.94p_{ij} \le p^*_{ij} \le 1.06p_{ij}$
5%	$(1-0.05)p_{ij}$	$(1+0.05)p_{ij}$	$0.95p_{ij} \le p^*_{ij} \le 1.05p_{ij}$
4%	$(1-0.04)p_{ij}$	$(1+0.04)p_{ij}$	$0.96p_{ij} \le p^*_{ij} \le 1.04p_{ij}$
3%	$(1-0.03)p_{ij}$	$(1+0.03)p_{ij}$	$0.97p_{ij} \le p^*_{ij} \le 1.03p_{ij}$
2%	$(1-0.02)p_{ij}$	$(1+0.02)p_{ij}$	$0.98p_{ij} \le p^*_{ij} \le 1.02p_{ij}$
1%	$(1-0.01)p_{ij}$	$(1+0.01)p_{ij}$	$0.99p_{ij} \le p^*_{ij} \le 1.01p_{ij}$
0.4%	$(1-0.004)p_{ij}$	$(1+0.004)p_{ij}$	$0.996p_{ij} \le p^*_{ij} \le 1.004p_{ij}$
0.3%	$(1-0.003)p_{ij}$	$(1+0.003)p_{ij}$	$0.997p_{ij} \le p^*_{ij} \le 1.003p_{ij}$
0.2%	$(1-0.002)p_{ij}$	$(1+0.002)p_{ij}$	$0.998p_{ij} \le p^*_{ij} \le 1.002p_{ij}$
0.1%	$(1-0.001)p_{ij}$	$(1+0.001)p_{ij}$	$0.999p_{ij} \le p^*_{ij} \le 1.001p_{ij}$

We tested the algorithms for the makespan criterion from Section 3.6 with all corresponding changes for criterion $\mathcal{C}_{max}$ for the same randomly generated test problems. Heuristic solutions of problem $\mathcal{J}/a_i \le p_i \le b_i/\mathcal{C}_{max}$ are presented in Table 5.10 and exact solutions are given in Table 5.8.

For criterion $\Phi = \sum \mathcal{C}_i$ (Tables 5.8 and 5.10 for criterion $\Phi = \mathcal{C}_{max}$), both Tables 5.7 and 5.9 present computational results for classical job shop problems only (see Table 1.1 at page 38). So, each randomly generated instance $\mathcal{J}m/n=k, a_i \le p_i \le b_i/\Phi$ has $|Q| = mn$ operations and the corresponding mixed graph G has $(m-1)n$ arcs and $\binom{n}{2}m$ edges (note that the latter parameter has the most influence on the running times of our algorithms). For more than 700 classical job shop problems with different combinations of $n \le 10$ and $m \le 8$, we calculated the average number of all feasible schedules λ, the average

cardinality $|B|$ of the set B, the average cardinality $|\Lambda^*(G)|$ of the set $\Lambda^*(G)$, and the average cardinality $|\Lambda^T(G)|$ of the set $\Lambda^T(G)$ for both criteria $\sum \mathcal{C}_i$ and $\mathcal{C}_{max}$.

For each combination of n and m under consideration, three types of series (called A, B and C) of instances were considered for the case of an exact solution (see Table 5.7 and Table 5.8). Each series consists of 10 instances with randomly generated technological routes. The expected processing times, which form the *input vector* p, are real numbers uniformly distributed in the segment $[10, 100]$. In each instance of types A, B and C, all operations are partitioned into four approximately equal parts with different maximal errors of the processing times (see Table 5.5). For an instance of type C, these errors are 1%, 3%, 5% and 7%, for an instance of type B, errors are 2%, 6%, 8% and 10%, and for an instance of type A, errors are 5%, 10%, 15% and 20% (see Table 5.5). In particular, the operations of the fourth part of an instance of type A have the most uncertain processing times: If the *input* (expected) processing time is supposed to be equal to p_{ij}, then the lower bound for the actual processing time is equal to $(1 - 0.2)p_{ij}$ and the upper bound is equal to $(1 + 0.2)p_{ij}$ (see Table 5.6). On the other hand, the operations of the first part of an instance of type C have the processing times with the smallest error: If the *input* processing time is supposed to be equal to p_{ij}, then the lower bound is equal to $(1 - 0.01)p_{ij}$ and the upper bound is equal to $(1 + 0.01)p_{ij}$. Table 5.7 for $\sum \mathcal{C}_i$ (Table 5.8 for $\mathcal{C}_{max}$) presents the results for the following three computational schemes, in which Algorithms $EXPL$, $B\&B1$ and $B\&B2$ are used with $\Phi_s^p = L_s^p$ (with $\Phi_s^p = l_s^p$, respectively) for the mean flow time criterion (for the makespan criterion).

Scheme I: Algorithm $EXPL \rightarrow$
Algorithm $SOL\_\sum \mathcal{C}_i$ (Algorithm $SOL\_\mathcal{C}_{max}$) $\rightarrow$
Algorithm $MINSOL^*\_\sum \mathcal{C}_i$ (Algorithm $MINSOL^*\_\mathcal{C}_{max}$)

Scheme II: Algorithm $B\&B1 \rightarrow$
Algorithm $SOL\_\sum \mathcal{C}_i$ (Algorithm $SOL\_\mathcal{C}_{max}$) $\rightarrow$
Algorithm $MINSOL^*\_\sum \mathcal{C}_i$ (Algorithm $MINSOL^*\_\mathcal{C}_{max}$)

Scheme III: Algorithm $B\&B2 \rightarrow$
Algorithm $SOL\_\sum \mathcal{C}_i$ (Algorithm $SOL\_\mathcal{C}_{max}$) $\rightarrow$
Algorithm $MINSOL^*\_\sum \mathcal{C}_i$ Algorithm $MINSOL^*\_\mathcal{C}_{max}$

Each of these schemes constructs first a solution B, then a solution $\Lambda^*(G)$ by Algorithm $SOL\_\sum \mathcal{C}_i$ ($SOL\_\mathcal{C}_{max}$) and finally a minimal solution $\Lambda^T(G)$ by Algorithm $MINSOL^*\_\sum \mathcal{C}_i$ ($MINSOL^*\_\mathcal{C}_{max}$). In Table 5.7 (Table 5.8), λ denotes the average number of schedules (third row in column 1), λ' the average cardinality of the set B, λ^* the average cardinality of the set $\Lambda^*(G)$, and λ^T the average cardinality of the set $\Lambda^T(G)$ (first, second and third rows in column 2, respectively). Of course, for each instance $\lambda, \lambda', \lambda^*$ and λ^T are integers, but their average values are real numbers given in Table 5.7, Table 5.8 and the tables below with one decimal place.

The application of Scheme I to Example 3.2 of problem $\mathcal{J}3/n = 3, a_i \le p_i \le b_i/\sum \mathcal{C}_i$ described in Section 3.4, gives the following sets of schedules. Firstly, Algorithm $EXPL$ constructs the set $\Lambda(G) = \{G_1, G_2, \ldots, G_{22}\}$ of all schedules and set $B \subset \Lambda(G)$ with $|B| = 12$ (see Step 4, where sufficiency from Lemma 3.5 is used). Then using solution B, Algorithm $SOL\_\sum \mathcal{C}_i$ constructs the set $\Lambda^*(G) = \{G_1, G_2, G_5\}$ which is also a solution.

Finally, Algorithm $MINSOL\_\sum \mathcal{C}_i$ shows that solution $\Lambda^*(G)$ is minimal, i.e. $\Lambda^T(G) = \Lambda^*(G)$. Thus, for Example 3.2 we have $\lambda = 22, \lambda' = 12$ and $\lambda^* = \lambda^T = 3$.

The average CPU time (in seconds) for constructing set B, set $\Lambda^*(G)$ and set $\Lambda^T(G)$ (first, second and third rows) are presented in columns 3, 4 and 5 for Schemes I, II and III, respectively. As follows from Table 5.7 and Table 5.8, in most cases for both criteria $\sum \mathcal{C}_i$ and $\mathcal{C}_{max}$, Scheme III based on Algorithm B&B2 is the best for the problems of type C, while Scheme I based on Algorithm $EXPL$ is the best for the problems of types A and B. As it was mentioned, Steps 8 and 9 for the branch-and-bound algorithm are not so fast as Step 1 of Algorithm $EXPL$. Moreover, due to a large uncertainty of the input vector p for problems A and B, Algorithms $B\&B1$ and $B\&B2$ have to construct a lot of intermediate digraphs $G_{(t)}$ in the branching tree which are not in the set $\Lambda(G)$. Unfortunately, the exact minimal solution was obtained within 1.5 hours by the worst of the Schemes I, II or III only for some combinations of n and m with $n \leq 4$ and $m \leq 8$ and the exact solution was not obtained by Scheme I for some combinations of n and m for the reason 'not enough memory' or 'limit of time' (such series are marked in the first column of Table 5.7 and Table 5.8 by an asterisk).

To solve problems with larger size, we were forced to consider restricted variants of the branch-and-bound algorithms: Algorithm $B\&B1^*$ (Algorithm $B\&B2^*$) denotes Algorithm $B\&B1$ (Algorithm $B\&B2$, respectively) without Steps 8 and 9. In general, such modifications do not guarantee to obtain a solution B, but they are essentially faster. Fortunately, for almost all problems presented in Table 5.7 and Table 5.8, the restricted variants of the branch-and-bound algorithms still give a solution, i.e. for each $p \in T$ the set B constructed contains an optimal schedule. The main reason for this computational result is that Steps 8 and 9 often generate only schedules which are dominated by other ones. Therefore, it is possible to exclude these schedules due to Theorem 3.4. Columns 6 – 9 of Table 5.7 (of Table 5.8) present computational results on a PC 486 (120 MHz) (on a PC 486 (133 MHz), respectively) for the following two computational schemes:

Scheme IV: Algorithm $B\&B1^*$ with $\Phi_s^p = L_s^p$ (with $\Phi_s^p = l_s^p$) $\rightarrow$
Algorithm $SOL\_\sum \mathcal{C}_i$ (Algorithm $SOL\_\mathcal{C}_{max}$) $\rightarrow$
Algorithm $MINSOL^*\_\sum \mathcal{C}_i$ (Algorithm $MINSOL^*\_\mathcal{C}_{max}$)

Scheme V: Algorithm $B\&B2^*$ with $\Phi_s^p = L_s^p$ (with $\Phi_s^p = l_s^p$) $\rightarrow$
Algorithm $SOL\_\sum \mathcal{C}_i$ (Algorithm $SOL\_\mathcal{C}_{max}$) $\rightarrow$
Algorithm $MINSOL^*\_\sum \mathcal{C}_i$ (Algorithm $MINSOL^*\_\mathcal{C}_{max}$)

More precisely, column 6 presents the average approximate values λ' (first row), λ^* (second row) and λ^T (third row) calculated by Algorithm $B\&B1^*$, and column 7 presents the average running times for constructing approximations of the sets $B, \Lambda^*(G)$ and $\Lambda^T(G)$ by Algorithm $B\&B1^*$. Similarly, column 8 presents the average approximate values λ', λ^* and λ^T calculated by Algorithm $B\&B1^*$, and column 9 presents the average running times for constructing approximations of the sets $B, \Lambda^*(G)$ and $\Lambda^T(G)$ by Algorithm $B\&B2^*$. From Table 5.7 for criterion $\sum \mathcal{C}_i$ and Table 5.8 for criterion $\mathcal{C}_{max}$, it follows that Algorithm $B\&B2^*$ (with the corresponding criterion) in Scheme V is often faster than Algorithm $B\&B1^*$ in Scheme IV: Only for some series of type A, Algorithm $B\&B1^*$ is, on average, faster than Algorithm $B\&B2^*$. Note also that Algorithm $B\&B2^*$ gives more often an exact solution than Algorithm $B\&B1^*$.

As it follows from Table 5.7 and Table 5.8, even the heuristic Schemes IV and V require rather large running times. So, for larger problem sizes, we used only Algorithms $B\&B1^*$ and $B\&B2^*$ for constructing the sets B heuristically, i.e. without a guarantee that the constructed set B is indeed a solution. Obviously, the cardinality of a solution increases not only with increasing the size of the problem (which in turn increases the running time), but also with increasing the uncertainty of the numerical input data. Therefore, to reduce the cardinality of a solution, we consider along with instances of types A, B, and C also problems of the following two types D and E with smaller errors of the given processing times, namely: The problems of type D with the errors of the processing times equal to 1 %, 2 %, 3 % and 4 %, and problems of type E with the errors of the processing times equal to 0.1 %, 0.2 %, 0.3 % and 0.4 % (see Table 5.5).

Heuristic solutions are represented in Table 5.9 for problems $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$ and in Table 5.10 for problems $\mathcal{J}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ with the same structural and numerical input data. Next, we describe the design of Table 5.9. The values of k used for criterion $\sum \mathcal{C}_i$ are given in column 2. Columns 3 - 6 (and columns 7 - 10) present computational results for Algorithm $B\&B1^*$ (and Algorithm $B\&B2^*$, respectively). Column 3 (column 7) gives the minimal value of the cardinality λ' of the set B constructed, column 4 (column 8) the average value of λ', and column 5 (column 9) the maximal value of λ'. The average CPU times are given in column 6 for Algorithm $B\&B1^*$ and in column 10 for Algorithm $B\&B2^*$. Table 5.10 has a similar design with the exception of the column with the values of k used. For criterion $\mathcal{C}_{max}$, we set $k = 150$ for all computational results presented for problems $\mathcal{J}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$. As follows from Table 5.9 and Table 5.10, if the problems have small size, Algorithm $B\&B2^*$ is preferable to Algorithm $B\&B1^*$ (in both running time and the quality of the solution constructed). However, if the number of potentially optimal schedules is large (due to a large problem size or due to a large uncertainty of the numerical input data), then Algorithm $B\&B1^*$ has a smaller running time, however the quality of the solution constructed by Algorithm $B\&B2^*$ remains still better. Moreover, the value of k has a large influence on the quality and the running time of Algorithm $B\&B1^*$ in contrast to Algorithm $B\&B2^*$ which is independent of k. In principle, we use the parameter k in Algorithm $B\&B2^*$ mainly to have the same conditions for the comparison with Algorithm $B\&B1^*$.

5.5 Remarks on the Scheduling Problems with Uncertainty

On the basis of the characterizations of a solution and a minimal solution derived in Section 3.5, we have developed an explicit enumeration scheme for the flow shop and for the classical job shop problems and branch-and-bound schemes for the general job shop problem. Instead of Algorithm $B\&B1$, one can use any known branch-and-bound method developed for problems $\mathcal{J}//\Phi$ (with fixed numerical input data) after a simple modification with the aim to construct the k best schedules instead of only an optimal one. However, the question which still remains open, is how to choose k to have a guarantee that the k best schedules contain an exact solution of problem $\mathcal{J}/a_i \leq p_i \leq b_i/\Phi$. To answer this question, we have available only experimental results given in Section 5.2 for calculating the stability radius of an optimal schedule for randomly generated job shop

problems with $n \leq 10,\ m \leq 7$ and small numbers of conflict edges in the mixed graph G. In particular, in all experiments presented in Table 5.7 and Table 5.8, we used $k = 150$ which was sufficient to obtain an exact solution for almost all problems considered. For the computational results presented in Table 5.9 for criterion $\sum \mathcal{C}_i$, we used $k = 150, k = 100$ and $k = 50$ depending on the problem size and on the uncertainty of the input vector of the processing times (which is defined by the problem type A, B, C, D or E). As follows from Table 5.9 and Table 5.10 (in which $k = 150$), these values of k were not sufficient for some instances in the sense that the cardinality of the set of schedules which may be optimal for some vectors $p \in T$ was larger than the value of k used. E.g., for each instance of type C with $n = m = 6$, the number of such schedules was larger than 50 (although $k = 50$ was used).

Algorithm $B\&B2$ constructs the set of all 'potentially optimal schedules' for problem $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$: If a schedule may be optimal for some feasible vector of the processing times, it has to be contained in the set constructed by Algorithm $B\&B2$. Thus, Algorithm $B\&B2$ guarantees an exact solution after a complete realization. Moreover, its running time was often less than that of Algorithm $B\&B1$. The 'heuristic' Algorithm $B\&B2^*$ was also often preferable in comparison with Algorithm $B\&B1^*$ in both running times and in the numbers of instances for which a better solution was obtained.

The comparison with Algorithm $EXPL$ may be as follows. On the one hand, Algorithm $EXPL$ takes often smaller running times for problems of types B and C. On the other hand, it is suitable only for a classical job shop problem (and for a flow shop problem as its special case), while Algorithms $B\&B1$ and $B\&B2$ are suitable for the general case of a job shop problem. Moreover, Algorithm $EXPL$ may be realized efficiently only in the exact version in contrast to Algorithms $B\&B1$ and $B\&B2$ which have efficient heuristic versions as well (namely Algorithms $B\&B1^*$ and $B\&B2^*$). Note also that some advantage of Algorithm $EXPL$ is based on the fast generation of all digraphs $G_s^T,\ G_s \in \Lambda(G)$, which have only dominant paths (see Steps 1 and 2). The dominance relation $\preceq_T$ between digraphs is tested in Step 4 of Algorithm $EXPL$. In contrast to Algorithm $EXPL$, both above branch-and-bound algorithms (and their four heuristic versions) first test dominance relation $\preceq_T$ (see Step 5) and then use a more time-consuming procedure (but which is suitable for the general case of a job shop problem) for excluding redundant paths from the digraph $G_s \in \Lambda(G)$.

It should be noted that the software developed allows to solve within one hour problems $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$ exactly with n x $m \leq 25$ on a PC 486 (120 MHz) and with n x $m \leq 50$ heuristically on a PC 486 (50 MHz). The cardinality of the set $\Lambda^T(G)$ (and as a result the running time of the above algorithms) grows very quickly with increasing the problem size or/and the size of the polytope T. So, the software developed may be practically efficient only if at least one of the above sizes is sufficiently small.

The developed approach seems to be useful for a preliminary analysis of a scheduling environment with uncertain numerical data and fixed structural data. After calculating a solution $\Lambda^*(G)$ or (what is better but more time-consuming) a minimal solution $\Lambda^T(G)$, a decision-maker may quickly choose the best schedule if additional information on the numerical data will be available at the stage of the realization of a schedule. To this end, it is desirable to construct at the stage of scheduling a small number of schedules which may be considered as possible candidates for a realization. Note that, for criterion $\sum \mathcal{C}_i$, the average value of $\lambda^T = |\Lambda^T(G)|$ (see column 2) was equal to 8 for the instances

presented in Table 5.7, and for the instances presented in Table 5.9, the average value of $\lambda' = |B|$ (see columns 4 and 8) was equal to 38. Unfortunately, in the latter case for 12 % of the instances under consideration, the number of potentially optimal schedules exceeded the value of k used.

As it was noted in [Alt00a], "typically an optimal solution for a model of the problem situation is generated (often by a computer), and this solution is translated back to a solution for the original real world problem". Usually, real world problems are of a large size and with a large uncertainty of the numerical input data. Since algorithms for constructing k best solutions do not work in such cases, Althöfer [Alt00b] presents an approach where 'true alternatives' are generated one after the other by introducing penalties. The development of such algorithms for scheduling problems may be a direction of some future work.

Next, we discuss how it is possible to use the results of this dissertation for the following two-stage practical processes. We propose to consider two sequential stages. At the first stage (*scheduling problem*), a set of *potentially* optimal schedules has to be constructed under the conditions of uncertain numerical input data. In other words, problem $\mathcal{G}/a_i \leq p_i \leq b_i/\Phi$ has to be solved, i.e. a minimal solution $\Lambda^T(G)$ has to be found. Each schedule from the set $\Lambda^T(G)$ is a *potentially* optimal schedule, and at the first stage, a decision-maker does not know which schedule from the set $\Lambda^T(G)$ will be the best in reality.

At the second stage (*control problem*), it is required to choose an optimal schedule from the set $\Lambda^T(G)$ and to realize it taking into account the additional information about the processing times of the operations. By a solution of the *control problem*, we mean a digraph $G_s \in \Lambda^T(G)$, which is optimal for the vector $p^* = (p_1^*, p_2^*, \ldots, p_q^*) \in T$ of the processing times: $\Phi_s^{p^*} = \min\{\Phi_k^{p^*} : G_k \in \Lambda^T(G)\}$, where p_i^* denotes the actual processing time of operation $i \in Q$. In the worst case, p_i^* may become known only after the completion of operation $i \in Q$. The digraph $G_s \in \Lambda^T(G)$, for which equality $\Phi_s^{p^*} = \min\{\Phi_k^{p^*} : G_k \in \Lambda^T(G)\}$ holds, is *definitely* optimal.

The problem of the first stage (*scheduling problem*) and the problem of the second stage (*control problem*) are distinguished by the time which is acceptable for the decision-making: To solve the *scheduling problem*, it is possible to use essentially more time than for the *control problem*, which has to be solved in a very short time as in on-line scheduling (see [CV97, HZ97, San95, Sei98]).

In contrast to the *scheduling problem*, which necessarily has a solution if digraph $(Q, A, \emptyset)$ does not contain a circuit, a solution (i.e. a definitely optimal schedule) of the *control problem* may not exist from some instant (i.e. an optimal continuation of a partial schedule which was realized may not exist after previous false decisions).

Some more details about the above two-stage processes are given in [98, Sot00]. In particular, sufficient conditions have been derived for some cases when *control problem* has a solution.

5.6 Comments and References

In Section 5.1, we develop algorithms for calculating the stability radii $\widehat{\varrho}_s(p)$ and $\overline{\varrho}_s(p)$ on the basis of the formulas from [BSW96, Sot91b, SWW98]. The computational results were also published in [LSSW98, SSW97, Sot99a].

Table 5.7: *Exact solutions of randomly generated problems* $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$

$n \times m$ type λ	λ' λ^* λ^T	CPU time for Scheme I	Scheme II	Scheme III	λ' λ^* λ^T	CPU time for Scheme IV	λ' λ^* λ^T	CPU time for Scheme V
1	2	3	4	5	6	7	8	9
3×3	5.2	.6	.7	.6	4.8	.4	4.8	.2
C	1.9	1.0	1.1	1.0	1.9	.8	1.9	.6
91.2	1.9	1.1	1.3	1.1	1.9	.9	1.9	.7
3×3	9.3	.7	.8	.5	6.1	.2	6.6	.2
B	2.9	1.2	1.3	1.0	2.7	.6	2.7	.6
90.7	2.6	1.4	1.5	1.2	2.4	.9	2.4	.9
3×3	16.5	.7	.8	.9	10.6	.3	11.3	.3
A	3.3	1.5	1.7	1.8	2.9	1.0	3.0	1.0
77.4	3.3	1.7	1.9	2.0	2.9	1.4	3.0	1.4
3×4	3.4	1.9	1.6	1.1	3.0	.8	3.0	.3
C	1.7	2.2	1.9	1.8	1.5	1.0	1.5	.6
261.9	1.7	2.3	2.0	1.5	1.5	1.1	1.5	.6
3×4	15.1	2.4	2.6	2.5	9.5	.9	10.0	.7
B	3.0	5.9	5.9	5.9	2.5	3.2	2.5	3.0
300.8	2.7	6.0	6.0	6.1	2.4	3.5	2.4	3.3
3×4	32.5	2.7	3.2	3.4	15.2	.8	16.8	.9
A	5.1	12.0	12.1	12.3	3.8	5.6	3.9	6.4
276.8	4.1	14.1	14.1	14.3	3.1	7.5	3.2	8.3
3×5	4.7	8.3	6.3	5.8	4.1	1.4	4.2	1.0
C	1.5	8.6	6.6	6.2	1.5	1.8	1.5	1.4
604.8	1.4	8.7	6.8	6.3	1.5	1.9	1.5	1.4
3×5	12.9	10.5	11.6	10.6	9.5	2.2	10.2	1.1
B	3.4	12.8	13.7	12.8	3.3	3.8	3.3	2.9
894.1	3.1	13.0	14.0	13.0	3.0	4.3	3.0	3.3
3×5	77.6	12.9	20.8	21.3	30.5	2.4	33.0	2.8
A	11.9	63.4	68.9	70.1	8.7	19.4	9.0	27.1
896.7	10.8	96.5	100.6	102.2	7.7	40.5	8.0	52.8
3×6	9.3	48.8	51.4	34.2	7.1	2.5	7.3	1.5
C	3.5	49.2	52.0	34.8	2.8	2.9	2.8	2.0
1555.9	2.7	49.5	52.8	35.6	2.1	3.2	2.1	2.3
3×6	21.0	49.1	66.5	58.9	13.9	3.6	14.1	2.3
B	4.6	71.9	88.2	80.6	4.5	19.5	4.5	18.3
1760.9	4.2	77.2	93.2	85.7	4.1	24.7	4.1	23.6
3×6	65.3	48.3	119.2	139.5	19.2	2.4	23.4	2.7
A	7.6	198.3	262.0	282.7	4.8	51.8	6.0	54.1
1559.0	7.0	476.0	526.2	548.3	4.5	111.5	5.6	289.4
3×7	5.4	307.6	343.0	310.0	4.1	5.4	4.4	1.6
C	1.5	308.3	343.7	310.7	1.5	5.9	1.5	2.2
4611.1	1.5	308.4	343.9	310.8	1.5	6.0	1.5	2.3
3×7	38.6	313.8	751.0	769.7	21.2	7.3	23.0	7.2
B	6.6	363.8	797.2	817.0	4.9	39.9	5.1	43.3
4805.1	5.7	371.1	804.1	824.0	4.3	45.3	4.5	49.3
3×7	156.0	279.7	1319.0	1274.7	21.5	2.4	27.0	3.7
A	19.2	923.4	1934.1	1897.3	9.2	91.4	10.2	108.6
2742.8	17.9	1032.7	2038.5	2003.1	8.2	123.0	9.2	144.8

Table 5.7 (continuation)**:** *Exact solutions of randomly generated problems* $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$

$n \times m$ type λ	λ' λ^* λ^T	CPU time for Scheme I	Scheme II	Scheme III	λ' λ^* λ^T	CPU time for Scheme IV	λ' λ^* λ^T	CPU time for Scheme V
1	2	3	4	5	6	7	8	9
3×8	20.4	2338.2	2106.5	1935.0	15.7	19.7	15.8	7.1
C	4.7	2348.2	2115.2	1938.5	4.0	28.1	4.0	15.5
21923.1	4.6	2354.6	2121.7	1943.6	3.9	33.2	3.9	20.6
$3 \times 8^*$	28.7	2060.8	4192.0	4018.8	26.0	9.9	27.0	7.1
B	7.4	2172.5	4282.2	4107.4	7.2	57.5	7.2	55.4
8810.2	7.3	2196.4	4302.5	4120.9	6.3	80.3	6.3	78.2
$3 \times 8^*$	141.5	2054.5	4119.2	3742.9	56.3	9.0	59.2	12.7
A	17.3	3999.9	5938.1	5556.0	19.1	1477.5	21.0	1463.0
8059.0	16.3	4501.5	6039.7	5811.8	17.6	1952.4	19.6	1963.1
4×3	13.3	19.2	14.2	9.5	11.4	6.1	11.7	2.3
C	3.2	20.3	15.4	10.7	3.2	7.3	3.2	3.5
2906.6	2.8	21.0	17.4	12.7	2.8	8.1	2.8	4.3
4×3	38.9	17.5	27.8	28.8	29.9	5.4	30.7	3.4
B	6.3	41.4	50.8	51.9	5.5	24.5	5.5	22.6
2217.4	5.5	45.4	54.8	55.8	4.8	28.3	4.8	26.4
4×3	290.9	28.6	70.7	69.0	110.1	5.9	115.2	8.1
A	31.0	414.0	440.4	442.0	22.5	288.4	22.8	293.9
2990.2	27.7	688.6	699.1	703.9	19.6	501.9	19.9	509.8
4×4	34.2	303.1	867.3	852.4	23.9	15.8	24.1	5.9
C	7.5	328.5	891.5	876.5	6.5	33.4	6.5	23.6
17159.1	6.3	330.5	893.0	878.0	5.5	35.1	5.5	25.3
$4 \times 4^*$	88.3	308.2	1501.2	1354.7	52.2	14.4	52.9	9.0
B	16.1	1293.8	2444.9	2297.2	13.5	782.8	13.5	780.5
17767.7	14.5	1574.0	2771.7	2580.5	12.0	999.6	12.0	997.6
4×4	477.7	319.4	2682.9	2505.1	131.7	15.4	132.0	23.8
A	30.8	3070.1	5355.7	5180.0	24.8	2680.7	24.8	2639.4
16142.6	30.1	3466.2	5771.4	5594.7	24.0	2905.8	24.0	2879.2

Table 5.8: *Exact solutions of randomly generated problems* $\mathcal{J}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$

$n \times m$ type λ	λ' λ^* λ^T	CPU time for Scheme I	Scheme II	Scheme III	λ' λ^* λ^T	CPU time for Scheme IV	λ' λ^* λ^T	CPU time for Scheme V
1	2	3	4	5	6	7	8	9
3×3	4.2	.17	.31	.15	3.4	.22	3.4	.10
C	1.3	.17	.31	.16	1.3	.26	1.3	.15
91.2	1.1	.17	.31	.16	1.2	.26	1.3	.16
3×3	16.0	.18	.28	.28	7.4	.14	7.8	.14
B	2.6	.18	.29	.30	2.0	.19	2.0	.19
90.7	2.4	.20	.30	.30	1.7	.20	1.6	.19
3×3	12.8	.14	.31	.30	8.3	.19	8.5	.17
A	2.2	.15	.32	.31	2.2	.24	2.2	.23
77.4	2.0	.15	.32	.31	2.0	.25	2.0	.23
3×4	7.0	.76	.86	.53	4.0	.48	4.3	.22
C	1.5	.77	.86	.54	1.5	.52	1.5	.25
261.9	1.3	.77	.86	.54	1.3	.53	1.3	.25
3×4	18.1	.85	1.0	.92	10.9	.55	11.0	.42
B	3.0	.90	1.1	.95	2.5	.61	2.7	.49
300.8	2.2	.90	1.1	.96	2.0	.64	2.0	.51
3×4	36.4	.80	1.22	1.14	13.6	.46	15.0	.45
A	7.1	.85	1.27	1.19	5.1	.54	5.5	.55
276.8	5.8	.89	1.29	1.21	4.2	.57	4.5	.59
3×5	8.0	4.05	2.87	2.24	5.4	.87	5.5	.40
C	2.2	4.06	2.88	2.25	1.9	.93	1.9	.45
604.8	1.6	4.07	2.89	2.25	1.7	.94	1.7	.45
3×5	11.8	4.79	4.48	3.22	7.2	1.33	7.6	.44
B	3.1	4.83	4.50	3.23	2.4	1.41	2.4	.52
894.1	2.3	4.83	4.51	3.23	2.1	1.42	2.1	.52
3×5	103.7	5.25	9.19	8.80	24.9	1.43	29.2	1.32
A	17.8	5.48	9.41	9.02	8.2	1.61	8.5	1.52
896.7	13.8	5.62	9.48	9.08	6.6	1.70	6.8	1.63
3×6	7.0	25.98	20.83	11.42	4.1	1.49	4.7	.57
C	2.4	25.99	20.83	11.43	2.1	1.54	2.3	.62
1555.9	2.2	25.99	20.83	11.43	1.9	1.54	2.1	.63
3×6	21.6	25.73	22.48	19.36	11.5	2.15	12.2	1.08
B	4.3	25.79	22.54	19.42	3.9	2.26	3.9	1.23
1760.9	3.5	25.83	22.57	19.43	3.1	2.31	3.1	1.29
3×6	67.8	25.37	54.80	54.62	18.1	1.48	19.5	1.30
A	12.5	25.64	55.06	54.89	7.2	1.68	8.0	1.51
1559.0	8.8	25.78	55.18	55.01	4.8	1.83	5.4	1.68
3×7	7.5	169.47	114.97	99.24	4.0	3.18	4.1	.60
C	1.7	169.51	114.99	99.26	1.4	3.26	1.4	.68
4611.1	1.5	169.51	115.00	99.27	1.3	3.28	1.3	.70
3×7	42.4	180.42	264.59	261.54	17.8	3.93	19.7	2.29
B	7.1	180.65	264.80	261.74	4.4	4.17	5.3	2.55
4805.1	5.7	180.71	269.88	261.78	3.9	4.29	4.4	2.69
3×7	90.4	152.22	604.40	523.97	16.6	1.39	19.2	1.66
A	17.7	152.76	604.89	524.45	7.6	1.59	8.6	1.91
2742.8	13.2	153.00	605.10	524.66	5.6	1.72	6.3	2.06

Table 5.8 (continuation)**:** *Exact solutions of randomly generated problems* $\mathcal{J}/a_i \le p_i \le b_i/\mathcal{C}_{max}$

$n \times m$ type λ	λ' λ^* λ^T	CPU time for Scheme I	Scheme II	Scheme III	λ' λ^* λ^T	CPU time for Scheme IV	λ' λ^* λ^T	CPU time for Scheme V
1	2	3	4	5	6	7	8	9
3×8	19.3	1297.43	841.55	795.93	11.3	11.39	11.2	2.66
C	5.2	1297.54	841.65	796.02	3.9	11.53	4.3	2.83
21923.1	4.6	1297.58	841.68	796.05	3.4	11.60	3.6	2.92
$3 \times 8^*$	32.9	1190.78	1959.74	1938.03	13.5	5.74	14.4	2.10
B	7.2	1191.12	1960.05	1938.35	4.6	6.01	4.9	2.37
8960.8	4.4	1191.19	1960.11	1938.39	3.2	6.18	3.2	2.55
$3 \times 8^*$	160.3	1161.51	3411.75	3022.93	40.9	5.21	44.8	6.73
A	23.6	1164.83	3414.66	3026.13	12.2	6.39	13.1	7.99
8296.1	19.0	1166.78	3415.74	3027.31	9.9	7.69	10.2	9.42
4×3	18.1	8.47	6.35	3.93	12.1	3.47	12.4	1.11
C	3.3	8.55	6.41	3.99	3.1	3.56	3.1	1.20
2906.6	2.1	8.56	6.41	3.99	2.4	3.61	2.1	1.24
4×3	40.9	7.35	10.01	8.67	27.7	3.19	29.3	1.29
B	7.2	7.48	10.10	8.77	5.0	3.34	7.0	1.45
2217.4	4.2	7.50	10.12	8.78	2.0	3.39	3.8	1.53
4×3	286.4	8.94	23.07	21.36	92.3	3.50	96.3	3.64
A	24.8	9.61	23.64	21.95	17.2	4.08	16.3	4.25
2990.2	21.3	9.74	23.75	22.03	14.1	4.45	13.3	4.61
4×4	41.8	164.20	199.87	201.76	19.9	7.84	19.9	2.53
C	6.4	164.41	200.05	201.95	3.8	7.99	3.8	2.67
17159.1	2.4	164.43	200.05	201.96	2.4	7.04	2.4	2.71
$4 \times 4^*$	79.0	169.08	199.25	190.39	27.1	7.52	27.3	2.47
B	14.7	169.58	199.69	190.85	7.2	7.79	6.9	2.73
17763.3	9.5	169.62	199.72	190.88	4.5	7.94	4.4	2.86
4×4	434.9	164.40	729.63	638.22	104.2	8.27	112.8	9.52
A	43.5	165.76	730.90	639.48	20.5	9.34	25.7	10.74
16142.6	34.8	166.43	731.42	640.01	15.6	10.17	20.0	11.70

Table 5.9: *Heuristic solutions of randomly generated problems* $\mathcal{J}/a_i \le p_i \le b_i/\sum \mathcal{C}_i$

n x m; type	k	$B\&B1^*$				$B\&B2^*$			
		λ'			CPU	λ'			CPU
		MIN	AVE	MAX	time	MIN	AVE	MAX	time
1	2	3	4	5	6	7	8	9	10
4 x 4; B	150	18	52.3	102	34.3	18	52.9	103	21.5
4 x 4; C	150	4	22.4	53	35.8	4	22.6	53	13.0
4 x 4; D	150	1	11.1	39	47.1	1	12.2	50	12.2
4 x 5; C	150	5	35.8	140	102.9	5	37.0	145	41.3
4 x 5; D	150	2	8.6	18	72.8	2	8.6	18	18.8
4 x 6; C	150	15	33.8	78	148.5	15	35.6	78	63.1
4 x 6; D	150	1	7.2	12	100.5	1	7.3	12	22.3
4 x 7; C	150	6	52.0	134	170.6	5	54.6	136	98.9
4 x 7; D	150	3	12.8	29	177.3	3	12.9	29	56.4
4 x 8; C	150	15	54.4	120	416.4	14	58.7	122	292.0
4 x 8; D	150	7	27.5	57	287.8	7	27.8	55	134.2
4 x 9; C	150	6	78.0	150	495.1	8	80.3	150	335.6
4 x 9; D	150	3	22.9	56	458.4	4	22.9	54	156.1
4 x 10; C	150	25	86.6	150	682.9	24	87.8	150	852.5
4 x 10; D	150	3	28.5	70	707.9	3	29.1	66	362.5
5 x 3; C	150	19	62.0	146	85.9	19	62.8	147	65.2
5 x 3; D	150	2	38.6	150	95.1	2	38.5	150	51.9
5 x 4; C	150	11	63.1	150	191.8	11	64.3	150	154.0
5 x 4; D	150	2	23.2	50	182.6	2	23.4	52	106.7
5 x 5; C	150	63	114.5	150	500.5	62	116.2	150	854.6
5 x 5; D	150	11	36.9	133	499.0	11	37.4	139	291.1
5 x 5; E	100	1	1.7	4	366.0	1	1.7	4	86.6
5 x 6; C	150	15	81.4	150	862.3	16	82.7	150	1220.3
5 x 6; D	150	7	49.0	89	761.5	7	48.6	88	493.6
5 x 7; D	150	9	47.9	150	1390.3	9	48.9	150	1642.0
5 x 7; E	50	1	2.6	7	539.3	1	2.6	7	214.7
5 x 8; D	100	18	78.5	100	1803.5	18	80.5	100	2446.7
5 x 8; E	50	1	3.2	6	1054.5	1	3.2	6	328.1
5 x 9; E	50	1	2.5	6	1531.3	1	2.5	6	653.4
5 x 10; E	50	1	2.5	5	2071.7	1	2.5	5	617.9
6 x 3; D	150	19	101.3	150	538.4	19	100.3	150	621.4
6 x 3; E	50	1	4.2	18	456.8	1	4.2	18	309.8
6 x 4; D	150	20	99.9	150	1197.8	18	81.3	150	1858.1
6 x 4; E	100	1	2.3	6	936.7	1	2.3	6	403.6
6 x 5; D	100	6	90.1	100	1671.0	6	88.1	100	3022.7
6 x 5; E	50	1	2.8	8	1382.4	1	2.8	8	724.1
6 x 6; C	50	50	50	50	2389.6	50	50	50	7350.4
6 x 6; D	50	15	46.5	50	1997.6	15	46.5	50	5252.0
6 x 6; E	50	1	4.1	12	1997.6	1	3.5	12	1226.2
7 x 3; D	150	42	122.5	150	1311.9	76	131.8	150	2302.3
7 x 4; E	100	1	7.1	20	2204.5	1	7.0	24	3608.4
7 x 5; E	50	1	8.4	39	3074.2	2	15.7	50	6139.9
8 x 3; E	50	1	4.5	9	1781.5	1	5.1	11	3103.3
9 x 2; E	100	1	14.1	100	1297.3	1	14.9	100	1958.7
10 x 2; E	50	2	14.1	50	1651.6	2	9.3	50	2781.4

Table 5.10: *Heuristic solutions of randomly generated problems* $\mathcal{J}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$

n x m; type		$B\&B1^*$				$B\&B2^*$			
		λ'			CPU	λ'			CPU
		MIN	AVE	MAX	time	MIN	AVE	MAX	time
1		2	3	4	5	6	7	8	9
4 x 4;	B	5	27.2	51	7.29	5	27.3	50	2.46
4 x 4;	C	2	19.9	62	7.59	2	19.9	64	2.51
4 x 4;	D	4	11.8	28	9.70	4	12.5	35	1.81
4 x 5;	C	7	32.1	69	15.97	7	33.1	74	6.58
4 x 5;	D	1	5.2	19	14.10	1	5.4	20	1.56
4 x 6;	C	6	28.0	79	27.53	6	29.7	82	8.69
4 x 6;	D	2	10.6	45	21.91	2	10.7	44	3.82
4 x 7;	C	6	38.7	118	31.93	5	41.3	129	16.43
4 x 7;	D	1	12.4	45	35.68	1	13.4	45	7.85
4 x 8;	C	14	60.6	143	63.53	18	66.0	134	38.91
4 x 8;	D	3	18.3	45	51.04	3	18.5	45	13.59
4 x 9;	C	8	55.1	145	76.46	8	60.6	150	42.13
4 x 9;	D	1	24.5	101	74.88	1	25.8	109	25.55
4 x 10;	C	16	74.9	150	100.86	17	82.3	150	78.79
4 x 10;	D	5	29.6	81	95.54	5	31.8	86	51.35
5 x 3;	C	22	101.0	150	13.06	22	100.4	150	11.64
5 x 3;	D	19	93.1	150	13.15	19	93.3	150	12.18
5 x 4;	C	17	86.1	150	26.86	17	88.4	150	21.81
5 x 4;	D	17	42.1	102	29.31	17	44.1	102	15.63
5 x 5;	B	57	133.1	150	46.29	60	136.4	150	128.36
5 x 5;	C	86	142.2	150	55.33	88	143.3	150	125.10
5 x 5;	D	13	59.7	150	55.67	13	60.7	150	31.70
5 x 5;	E	1	14.6	84	55.43	1	15.1	89	12.20
5 x 6;	C	19	91.3	150	83.44	23	95.9	150	94.73
5 x 6;	D	7	49.8	104	88.35	7	54.0	131	52.46
5 x 7;	D	35	103.9	150	148.23	35	104.6	150	364.76
5 x 7;	E	1	8.2	24	112.03	1	8.2	24	21.18
5 x 8;	D	22	92.1	150	211.49	21	91.9	150	331.82
5 x 8;	E	2	11.3	58	179.53	2	10.9	58	49.70
5 x 9;	E	1	8.9	57	233.89	1	9.1	57	49.22
5 x 10;	E	1	5.9	21	309.72	1	5.9	21	53.87
6 x 3;	D	25	132.7	150	57.70	25	137.5	150	141.25
6 x 3;	E	5	102.9	150	65.60	8	119.7	150	160.52
6 x 4;	D	78	139.7	150	130.70	86	140.5	150	327.15
6 x 4;	E	2	59.2	150	100.75	2	59.6	150	62.80
6 x 5;	D	109	135.8	150	155.49	112	139.3	150	319.53
6 x 5;	E	1	37.0	143	157.26	1	37.0	143	125.21
6 x 6;	C	150	150	150	271.68	150	150	150	1136.78
6 x 6;	D	25	110.0	150	255.49	42	126.3	150	550.78
6 x 6;	E	1	17.6	43	255.48	1	17.7	43	133.24
7 x 3;	D	150	150	150	169.66	110	146.0	150	414.17
7 x 4;	E	4	87.1	150	268.30	12	96.2	150	505.38
7 x 5;	E	5	87.1	150	342.51	2	124.7	150	1057.61
8 x 3;	E	2	134.4	150	234.02	36	138.6	150	585.55
9 x 2;	E	150	150	150	157.68	150	150	150	416.59
10 x 2;	E	150	150	150	209.69	150	150	150	606.73

Table 5.11: *Common notations*

Symbols	Description
$\mathcal{J}$	Symbol for job shop (in the three-field notation)
$\mathcal{F}$	Symbol for flow shop
$\mathcal{O}$	Symbol for open shop
$\mathcal{G}$	Symbol for general shop
M	Set of machines: $M = \{M_1, M_2, \ldots, M_m\}$
J	Set of jobs: $J = \{J_1, J_2, \ldots, J_n\}$
$C_i(s)$	Completion time of job $J_i \in J$ at schedule s
$\Phi(C_1, C_2, \ldots, C_n)$	Objective function of job completion times where $C_i = C_i(s)$
Φ_k^p	Value of the objective function calculated for digraph $G_k \in \Lambda(G)$ (schedule k) with processing times given by vector $p \in R_+^q$
Φ	Regular criterion
$\mathcal{C}_{max} = \max_{i=1}^n C_i$	Criterion of minimizing the maximum flow time (makespan)
$\sum \mathcal{C}_i = \sum_{i=1}^n C_i$	Criterion of minimizing the mean flow time
$C_i(s)$	Completion time of job $J_i \in J$ at schedule s
$\|B\|$	Cardinality of set B (number of elements in set B if it is finite)
$\|b\|$	Absolute value of real number $b \in R^1$, i.e., $\|b\| = b$, if $b \in R_+^1$, $\|b\| = -b$, if $b \in R^1 \setminus R_+^1$
R^n	Space of n-dimensional real vectors with the maximum metric
R_+^n	Space of non-negative real vectors with the maximum metric
$d(p, p')$	Distance between vector $p \in R^n$ and vector $p' \in R^n$: $d(p, p') = \max_{i=1}^n \|p_i - p_i'\|$
$\lceil a \rceil$	The smallest integer greater than or equal to real number a
$[\mu] + [\nu]$	'Symmetric difference' of set $[\mu]$ and set $[\nu]$: $[\mu] + [\nu] = [\mu] \cup [\nu] \setminus [\mu] \cap [\nu]$
$G = (Q, A, E)$	Mixed graph defining the structural input data
q	Number of operations: $q = \|Q^J\| = \sum_{i=1}^n n_i = \sum_{k=1}^m \|Q_k^J\|$
$G_k = (Q, A \cup E_k, \emptyset)$	Acyclic digraph generated from mixed graph G
E_k	Signature of schedule k with $k \in S$ and $G_k \in \Lambda(G)$
$G(p) = (Q(p), A, E)$	Weighted mixed graph with vector p of job processing times
$G_k(p) = (Q(p), A \cup E_k, \emptyset)$	Acyclic weighted digraph
G_k^T	Minimal subgraph of G_k containing all dominant paths with respect to polytope T
$\Lambda(G) = \{G_1, G_2, \ldots, G_\lambda\}$	Set of acyclic digraphs generated from mixed graph G
$S = \{1, 2, \ldots, \lambda\}$	Set of semiactive schedules
λ	Number of semiactive schedules
$S^\Phi(p)$	Set of optimal semiactive schedules with respect to criterion Φ
$O_\varrho(p)$	Stability ball of optimal digraph with radius ϱ and center $p \in R_+^q$
$\varrho_s(p)$	Stability radius of optimal digraph G_s for a regular criterion
$\widehat{\varrho}_s(p)$	Stability radius of optimal digraph G_s for makespan criterion
$\overline{\varrho}_s(p)$	Stability radius of optimal digraph G_s for mean flow time criterion
$[\mu]$	Set of vertices (operations) which are contained in path μ
$l^p(\mu)$	Weight of path μ in the digraph with processing times $p \in R_+^q$
$\tilde{H}_k^i$	Set of paths in digraph G_k ending in the last operation of job J_i
H_k^i	Set of dominant paths in $\tilde{H}_k^i$

Conclusions

In spite of a large number of papers and books published about optimal sequencing and scheduling, the utilization of numerous results of the scheduling theory in most production environments is far from the desired volume. One of the reasons for the gap between scheduling theory and practice is connected with the usual assumption that the processing times of the jobs are known exactly before scheduling (for deterministic models) or that they are random values with known probability distributions (for stochastic models).

In the first part of these Conclusions, we summarize what we have learned from studying the scheduling paradigm when the processing times of the operations (and/or other numerical input data) are uncertain before scheduling. In the second part, we outline some topics for future research which follow directly from the above results. The common notations used in this dissertation are combined in Table 5.11, see also special notations for the general shop (Table 1.3 at page 65) and special notations for the job shop (Table 4.34 at page 256).

1. In this dissertation, a model of more realistic scheduling scenarios was considered. It was assumed that in the practical realization of a schedule, the processing time of an operation may take any real value between lower and upper bounds (within the polytope T), which are given before applying a scheduling procedure, and there is no prior information about the probability distributions of the random processing times. For such an uncertain scheduling problem, there does usually not exist a unique schedule that remains optimal for all possible realizations of the processing times and a set of schedules has to be considered which dominates all other schedules for the given criterion. To find such a set of schedules, our idea was to use a stability analysis of an optimal schedule with respect to the perturbations of the processing times (a survey of the main results on stability of an optimal schedule was given in Chapter 1).

In Chapter 2 and Chapter 3, we introduced the notion of the relative stability radius of an optimal schedule s as the maximal value of the radius of a stability ball (in the space of real vectors of the processing times) within which schedule s remains the best among the given set B of schedules (see Definition 3.2 at page 118 for the maximum flow time criterion and Definition 3.6 at page 147 for the mean flow time criterion). The relativity is considered with respect to the polytope T of feasible vectors of the processing times and with respect to the set B of semiactive schedules for which the superiority of a schedule s at hand has to be guaranteed.

We used the mixed (disjunctive) graph model which is suitable for the whole scheduling process from the initial mixed graph G representing the input data until a final digraph G_s representing a semiactive schedule s. The mixed graph model may be used for different

requirements on the numerical input data (see Table 3.1 at page 111). The most results obtained in this dissertation (see Chapters 2 and 3) are formulated in terms of paths in the digraphs G_s.

In Chapter 2, we focused on dominance relations between feasible schedules taking into account the given polytope T (Section 3.2). We established necessary and sufficient conditions for the case of an infinitely large relative stability radius of an optimal schedule s for the maximum flow time criterion (Theorem 3.2 at page 124). Under such conditions, schedule s remains optimal (has the minimal length) for any feasible perturbations of the processing times.

We established also necessary and sufficient conditions for the case of a zero relative stability radius of an optimal schedule s (Theorem 3.1 at page 121). Under such conditions, the optimality of schedule s is unstable: There are some small changes of the given processing times which imply that another schedule from the set B will be better (will have a smaller length) than schedule s.

Formulas for calculating the exact value of the relative stability radius are based on a comparison of an optimal schedule s with other schedules from the set B (Theorem 3.3 at page 126), and we show how it is possible to restrict the number of schedules from the set B examined for such a calculation of the relative stability radius (Lemma 3.4 at page 136). To this end, we considered the schedules from the set B in non-decreasing order of the values of the objective function until some inequalities hold (Corollary 3.4 at page 138).

In Chapter 3, analogous results were obtained for the mean flow time criterion, and the focus was on the dominance relations between feasible schedules taking into account the given criterion (Definition 3.5 at page 140). Formulas for calculating the exact value of the relative stability radius were given in Theorem 3.6 at page 151. A possibility to restrict the number of schedules under consideration has been discussed in Section **??**. We established necessary and sufficient conditions for an infinitely large relative stability radius of an optimal schedule for the mean flow time criterion (Theorem 3.7 at page 152) and necessary and sufficient conditions for a zero relative stability radius of an optimal schedule (Theorem 3.8 at page 153).

Using these results, we developed several exact and heuristic algorithms for constructing a solution and a minimal solution (see Definition 3.1 at page 110) of a scheduling problem with uncertain processing times. The developed software was tested on randomly generated job shop problems, and the computational results were discussed in Chapter 4. For the maximum and mean flow time criteria, we calculated the stability radii of the optimal schedules for more than 10,000 randomly generated instances. For a randomly generated uncertain scheduling problems with the same criteria, we constructed solutions and minimal solutions as well. In the experiments both the numbers of jobs and machines were restricted by 10. The most critical parameter for the running time of the programs was the number of edges E in the mixed graph $G = (Q, A, E)$.

2. In conclusion, we present some topics for future research. We can note that the most part of this dissertation (Chapters 3, **??** and 5) is devoted to the *scheduling problem* (with uncertain numerical input data) for the criteria $\mathcal{C}_{max}$ and $\sum \mathcal{C}_i$. Other regular criteria for the *scheduling problem* may be the subject of further research.

In Section 5.5, we introduced the *control problem* (when uncertain numerical data are

realized), which follows after the *scheduling problem*. For the *control problem*, only very preliminary results are known, however the *control problem* seems to be very important for practice. The *control problem* (in the setting presented in Section 5.5) may be a subject for future research.

The next important direction for further research is to construct more efficient algorithms for the *scheduling problem*, in particular, to use this approach for *scheduling problems* whose deterministic versions have polynomial algorithms for constructing optimal schedules.

Another direction of future research may be connected with the consideration of a minimal solution of an uncertain scheduling problem with respect to the cardinality of the solution obtained (the minimal solution $\Lambda^T(G)$ was determined with respect to inclusion).

After carrying out the computational experiments with the calculation of the stability radii of an optimal schedule (see Chapter 4), we can select the following topics for future research. For practical aims, it is useful to develop further a branch-and-bound algorithm for constructing the k best schedules (instead of one, which is usually constructed) and to combine such a calculation with a stability analysis on the basis of the results discussed in Chapters 1 and 4.

Another possible topic is to improve the bounds (3.35) at page 136 and (3.64) at page 167 in order to restrict the number of digraphs G_s, with which an optimal digraph has to be compared, while calculating its stability radius.

A more complex question is to find simpler (practical) formulas for calculating the stability radius or at least lower and/or upper bounds for it (without considering the paths of the digraph G_s).

If the calculation of a bound for the stability radius will be simplified considerably, it seems to be useful to calculate this bound within a branch-and-bound framework. One can obtain a new type of stopping rule (or some other advantages) due to a possible connection between the stability radius of the best constructed schedule and its proximity to the optimum.

Sections 4.1 and 4.2 were basically devoted to an overall enumeration scheme for calculating the stability radii, and an implicit enumeration scheme was only used *before* performing the stability analysis (i.e. for calculating optimal and near optimal schedules). The application of the stability analysis *within* an implicit enumeration framework should have practical utility and it may be a topic for future research, too.

The scheduling problem with uncertainty remains an interesting and challenging subject for the studies, which may combine some theoretical results with practical problems.

Index

Bibliography

[84] .. , .. , and .. . ., 1984.

[98] .. and .. . . , 42(5):8–12, 1998.

[AG99] M.S. Akturk and E. Gorgulu. Match-up scheduling under a machine breakdown. *Journal of the Operations Research Society*, 112:81–97, 1999.

[AGA98] A. Allahverdi, J.N.D. Gupta, and T. Aldowaisan. A review of scheduling research involving setup considerations. Department of Mechanical and Industrial Engineering Kuwait University, Safat, Kuwait, 1998.

[AGA99] A. Allahverdi, J.N.D. Gupta, and T. Aldowaisan. A review of scheduling research involving setup considerations. *OMEGA The International Journal of Management Sciences*, 27:219–239, 1999.

[All95] A. Allahverdi. Two-stage production scheduling with separated setup times and stochastic breakdowns. *Journal of the Operations Research Society*, 46:896–904, 1995.

[All97] A. Allahverdi. Scheduling in stochastic flowshop with independent setup, processing and removal times. *Computers & Operations Research*, 24:955–960, 1997.

[All99] A. Allahverdi. Stochastically minimizing total flowtime in flowshops with no waiting space. *European Journal of Operational Research*, 113:101–112, 1999.

[All00] A. Allahverdi. Minimizing mean flowtime in a two-machine flowshop with sequence independent setup times. *Computers & Operations Research*, 27:111–127, 2000.

[Alt00a] I. Althöfer. Decision support systems with multiple choice structure. In I. Althöfer, N. Cai, and et al., editors, *Numbers, Information and Complexity*, volume 16, pages 72–108. Kluwer, Amsterdam, 2000.

[Alt00b] I. Althöfer. K-alternative algorithms for sum type problems. In S. Dempe, H. Schreier, and I. Schiermeyer, editors, *14. Workshop on Discrete Optimization*, pages 5 – 8, Holzhau, Erzgebirge - Germany, May 23 -26, 2000. Technische Universität Bergakademie Freiberg.

[AM94] A. Allahverdi and J. Mittenthal. Two-machine ordered flowshop scheduling under random breakdowns. *Mathematical and Computer Modelling*, 20:9–17, 1994.

[AM98] A. Allahverdi and J. Mittenthal. Dual criteria scheduling on a two machine flow shop subject to random breakdowns. *Internat. Transactions in Oper. Res.*, 5:317–324, 1998.

[Bak74] K.R. Baker. *Introduction to Sequencing and Scheduling.* John Wiley, New York, 1974.

[Bay86] I. Baybars. A survey of exact algorithms for the simple assembly line balancing problem. *Management Science*, 32(8):909–932, 1986.

[BBMN91] J.C. Bean, J.R. Birge, J. Mittenehal, and C.E. Noon. Match-up scheduling with multiple resources, release dates and disruption. *Operations Research*, 39:470–483, 1991.

[BDP96] J. Blazewicz, W. Domschke, and E. Pesch. The job shop scheduling problem: Conventional and new solution techniques. *European Journal of Operational Research*, 93:1–33, 1996.

[BEP+96] J. Blazewicz, K.H. Ecker, E. Pesch, G. Schmidt, and J. Weglarz. *Scheduling in Computer and Manufacturing Processes.* Springer-verlag, Germany, Berlin, 1996.

[BESW93] J. Blazewicz, K. Ecker, G. Schmidt, and J. Weglarz. *Scheduling in Computer and Manufacturing Systems.* Springer-Verlag, Germany, Berlin, 1993.

[BHTW99a] H. Bräsel, M. Harboth, T. Tautenhahn, and P. Willenius. On the hardness of the classical job shop problem. *Annals of Operations Research*, 92:265–279, 1999.

[BHTW99b] H. Bräsel, M. Harboth, T. Tautenhahn, and P. Willenius. On the set of solutions of the open shop problem. *Annals of Operations Research*, 92:241–263, 1999.

[BJS94] P. Brucker, B. Jurisch, and B. Sievers. A fast branch and bound algorithm for the job shop scheduling problem. *Discrete Applied Mathematics*, 49:107–127, 1994.

[BK86] P.C. Bagga and K. Khurana. Two-machine flowshop with separated sequence-independent setup times: mean completion time criterion. *Indian Journal of Management and Systems*, 2:47–57, 1986.

[BK96a] H. Bräsel and M. Kleinau. New steps in the amazing world of sequences and schedules. *Math. Methods Oper. Res.*, 43:195–214, 1996.

[BK96b] P. Brucker and A. Krämer. Polynomial algorithms for resource-constrained and multiprocessor task scheduling problems. *European Journal of Operational Research*, 90:214–226, 1996.

[BKS99] P. Brucker, S.A. Kravchenko, and Yu.N. Sotskov. Preemptive job-shop scheduling problems with fixed number of jobs. *Math. Meth. Oper. Res.*, 49:41–76, 1999.

[BLSS02] O. Braun, T.-C. Lai, G. Schmidt, and Yu.N. Sotskov. Stability of johnson's schedule with respect to limited machine availability. *International Journal of Production Research*, 40(17):4381–4400, 2002.

[Bru95] P. Brucker. *Scheduling Algorithms*. Springer, Berlin, Berlin, 1995.

[BSW96] H. Bräsel, Yu.N. Sotskov, and F. Werner. Stability of a schedule minimizing mean flow time. *Mathematical and Computer Modelling*, 24(10):39–53, 1996.

[BT00] J. Bukchin and M. Tzur. Design of flexible assembly line to minimize equipment cost. *IIE Transactions*, 32:585–598, 2000.

[CCLe95] P. Chretienne, E.G. Coffman, J.K. Lenstra, and Z. Liu (editors). *Scheduling Theory and its Applications*. John Wiley & Sons, 1995.

[CCP98] H. Chen, C. Chu, and J.M. Proth. Cyclic scheduling of a hoist with time window constraints. *IEEE Transactions on Robotics and Automation*, 14:144–152, 1998.

[CDL94] G. Chrysslouris, K. Dicke, and M. Lee. An approach to real-time flexible scheduling. *Internat. J. Flexible Manufacturing Systems*, 6:235–253, 1994.

[CG00] C. Chu and V. Gordon. TWK due date determination and scheduling model: NP-hardness and polynomially solvable case. In J.-M. Proth and V. Tanaev, editors, *Proceedings of the International Workshop on Discrete Optimization Methods an Scheduling and Computer-aided Design*, pages 99–101, Minsk, Belarus, September 5-6, 2000.

[CJK98] T.C.E. Cheng, A. Janiak, and M.Y. Kovalyov. Bicriterion single machine scheduling with resource dependent processing time. *SIAM Journal on Computing*, 8(2):617–630, 1998.

[CM01] S. Chopra and P. Meindl. *Supply Chain Management: Strategy, Planning, and Operation*. Prentice Hall, Upper Saddle River, NJ, USA, 2001.

[CMM67] R.W. Conway, W.L. Maxwell, and L.W. Miller. *Theory of Scheduling*. Addison-Wesley, Reading, MA, USA, 1967.

[CP89] J. Carlier and E. Pinson. An algorithm for solving the job shop problem. *Management Science*, 35:164–176, 1989.

[CS98a] T.C.E. Cheng and N. Shakhlevich. Proportionate flow shop with controllable processing times. Working Paper No. 13/97-8, The Hong Kong Polytechnic University, 1998.

[CS98b] T.C.E. Cheng and N. Shakhlevich. Single machine scheduling of unit-time jobs with controllable release dates. Working Paper No. 01/98-9, The Hong Kong Polytechnic University, 1998.

[CT96] X. Cai and F.S. Tu. Scheduling jobs with random processing times on a single machine subject to stochastic breakdowns to minimize early-tardy penalties. *Naval Research Logistics*, 43:1127–1146, 1996.

[CTU96] S. Chand, R. Traub, and R. Uzsoy. Single-machine scheduling with dynamic arrivals: Decomposition results and an improved algorithm. *Naval Research Logistics*, 43:709–716, 1996.

[CV97] B. Chen and A.P.A. Vestjens. Scheduling on identical machines: How good is LPT in an on-line setting? *Operations Research Letters*, 21:165–169, 1997.

[CW99] T.C.E. Cheng and G. Wang. Two-machine flowshop scheduling with consecutive availability constraints. *Information Processing Letters*, 71:49–54, 1999.

[CW00] T.C.E. Cheng and G. Wang. An improved heuristic for two-machine flowshop scheduling with an availability constraint. *Operations Research Letters*, 26:223–229, 2000.

[DGHL02] A. Dolgui, N. Guschinski, Y. Harrath, and G. Levin. Une approche de programmation linaire pour la conception des lignes de transfert. *European Journal of Automated Systems (APII-JESA)*, 36(1):11–33, 2002.

[DHM96] R.L. Daniels, B.J. Hoopes, and J.B. Mazzola. Scheduling parallel manufacturing cells with resource flexibility. *Management Science*, 42:1260–1276, 1996.

[Dij59] E.W. Dijkstra. A note on two problems in connection with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[DJ88] W.J. Davis and A.T. Jones. A real-time production scheduler for a stochastic manufacturing environment. *International Journal of Production Research*, 1(2):101–112, 1988.

[DK95] R.L. Daniels and P. Kouvelis. Robust scheduling to hedge against processing time uncertainty in single stage production. *Management Science*, 41:363–376, 1995.

[DL82] V. Dumitru and F. Luban. Membership functions, some mathematical programming models and production scheduling. *Fuzzy Sets and Systems*, 8:19–33, 1982.

[ES76] M.J. Eisner and D.G. Severance. Mathematical techniques for efficient record segmentation in large shared databases. *Journal of the Association for Computing Machinery*, 23:619–635, 1976.

[ET00] S. Elmaghraby and K.A. Thoney. Two-machine flowshop problem with arbitrary processing time distributions. *IIET*, 31:467–477, 2000.

[FT63] H. Fisher and M.L. Thompson. Probabilistic learning combinations of local job-shop scheduling rules. In J.F. Muth and G.L. Thompson, editors, *Industrial Scheduling*, pages 225–251. Prentice-Hall, Englewood Cliffs, 1963. : , . and , .., , , 260-290, 1966.

[Gan19] H.L. Gantt. Efficiency and democracy. *Trans. Amer. Soc. Mech. Engrg.*, 40:799–808, 1919.

[GG94] B. Grabot and L. Geneste. Dispatching rules in scheduling: A fuzzy approach. *International Journal of Production Research*, 32(4):903–915, 1994.

[GJS76] M.R. Garey, D.S. Johnson, and R. Sethi. The complexity of flowshop and jobshop scheduling. *Math. Oper. Res.*, 1:117–129, 1976.

[GL80] E.N. Gordeev and V.K. Leontev. Stability in bottleneck problems. *U.S.S.R. Comput. Maths. Math. Phys*, 20(4):275–280, 1980.

[GL85] E.N. Gordeev and V.K. Leontev. The complexity of the tabulation of trajectory problems. *U.S.S.R. Comput. Maths. Math. Phys*, 25(4):199–201, 1985.

[GLLRK79] R.L. Graham, E.L. Lawler, J.K. Lenstra, and A.H.G. Rinnoy Kan. Optimization and approximation in deterministic sequencing and scheduling. *Annals of Disc. Appl. Math.*, 5:287–326, 1979.

[GLS83] E.N. Gordeev, V.K. Leontev, and I.Ch. Sigal. Computational algorithms for finding stability radius in choice problems. *U.S.S.R. Comput. Maths. Math. Phys.*, 23(4):128–132, 1983.

[Gor89] E.N. Gordeev. Solution stability of the shortest path problem. *Discrete Math.*, 1(3):45–56, 1989. (In Russian).

[Gra81] S.C. Graves. A review of production scheduling. *Operations Research*, 29(4):646–675, 1981.

[Gre97] H.J. Greenberg. An annotated bibliography for post-solution analysis in mixed integer programming and combinatorial optimization. University of Colorado at Denver, Mathematics Department, 1997.

[GS78] T. Gonzalez and S. Sahni. Flow shop and job shop schedules. *Operations Research*, 26:36–52, 1978.

[GTL96] M. Gen, Y. Tsujimura, and Y. Li. Fuzzy assembly line balancing using genetic algorithms. *Computers and Industrial Engineering*, 31(3/4):631–634, 1996.

[Gus83] D. Gusfield. Parametric combinatorial computing and a problem of program module distribution. *Journal of Association of Computer Machinery*, 30:551–563, 1983.

[HZ97] O. Holthaus and H. Ziegler. Look ahead job demanding for improving job shop perfomance. *Operations Research Spektrum*, 19:23–29, 1997.

[IMN87] H. Ishii, T. Masuda, and T. Nishida. Two machine mixed shop scheduling problems with controllable machine speeds. *Discrete Applied Mathematics*, 17:29–38, 1987.

[IN86] H. Ishii and T. Nishida. Two machine open shop scheduling problem with controllable machine speeds. *Journal of the Operations Research Society of Japan*, 29:123–131, 1986.

[IT95] H. Ishii and M. Tada. Single machine scheduling problem with fuzzy precedence relation. *European Journal of Operational Research*, 87:284–288, 1995.

[Jac56] J.R. Jackson. An extension of johnson's results on job lot scheduling. *Naval Research Logistics Quarterly*, 3(3):201–203, 1956.

[Jan88] A. Janiak. General flow-shop scheduling with resource constraints. *International Journal of Production Research*, 26:125–138, 1988.

[JM99] A.S. Jain and S. Meeran. Deterministic job-shop scheduling: Past, present and future. *European Journal of Operational Research*, 113:390–434, 1999.

[Joh54] S.M. Johnson. Optimal two and three stage production schedules with set up times included. *Naval Research Logistics Quarterly*, 1(1):61–68, 1954.

[Kam99] J. Kamburowski. Stochastically minimizing the makespan in two-machine flow shops without blocking. *European Journal of Operational Research*, 112:304–309, 1999.

[KB84] K. Khurana and P.C. Bagga. Minimizing the makespan in a two-machine flowshop with time lags and setup conditions. *Zeitschrift Operations Research*, 28:163–174, 1984.

[KBF+02] W. Kubiak, J. Blazewicz, P. Formanowicz, J. Breit, and G. Schmidt. Two-machine flowshops with limited machine availability. *European Journal of Operational Research*, 136:528–540, 2002.

[KDV00] P. Kouvelis, R.L. Daniels, and G. Vairaktarakis. Robust scheduling of a two-machine flow shop with uncertain processing times. *IIE Transactions*, 32:421–432, 2000.

[KK00] G. Kyparisis and C. Koulamas. Open shop scheduling with makespan and total completion time criteria. *Computers & Operations Research*, 27:15–27, 2000.

[KKG92] P. Kouvelis, A.A. Kurawarwala, and G.J. Gutierrez. Algorithms for robust single- and multiple-period layout planning for manufacturing systems. *European Journal of Operational Research*, 63:287–303, 1992.

[KN86] P.S. Ku and S.C. Niu. On johnson's two-machine flow-shop with random processing times. *Operations Research*, 34:130–136, 1986.

[KRKvHW94] A.W.H. Kolen, A.H.G. Rinnooy Kan, C.P.M. van Hoesel, and A.P.M. Wagelmans. Sensitivity analysis of list scheduling algorithms. *Discrete Applied Mathematics*, 55:145–162, 1994.

[KSL98] P. Kaminsky and D. Simchi-Levi. Probabilistic analysis and practical algorithms for the flow shop weighted completion time problem. *Operations Research*, 46(6):172–882, 1998.

[KSW95] S.A. Kravchenko, Yu.N. Sotskov, and F. Werner. Optimal schedules with infinitely large stability radius. *Optimization*, 33:271–280, 1995.

[KW96] M. Kuroda and Z. Wang. Fuzzy job shop scheduling. *International Journal Production Economics*, 44:45–51, 1996.

[LC95] W. Li and J. Cao. Stochastic scheduling on a single machine subject to multiple breakdowns according to different probabilities. *Operations Research Letters*, 18:81–91, 1995.

[Lee97] C.Y. Lee. Minimizing the makespan in the two-machine flowshop scheduling problem with an availability constraint. *ORL*, 20:129–139, 1997.

[Leo75] V.K. Leontev. The stability of the traveling salesman problem. *U.S.S.R. Comput. Math. and Math. Phys.*, 15(5):199–213, 1975.

[Leo76] V.K. Leontev. Stability in combinatorial choice problems. *Soviet Mathematics Doklady*, 17:635–638, 1976.

[Lib91] M. Libura. Sensitivity analysis for minimum Hamiltonian path and traveling salesman problems. *Discrete Applied Mathematics*, 30(2):197–211, 1991.

[Lib99] M. Libura. On accuracy of solutions for discrete optimization problems with perturbed coefficients of the objective function. *Annals of Operations Research*, 86:53–62, 1999.

[LJ91] H.F. Lee and R.V. Johnson. A line-balancing strategy for designing flexible assembly systems. *International Journal of Flexible Manufacturing Systems*, 3:91–120, 1991.

[LLLH00] H. Li, Z. Li, L.X. Li, and B. Hu. A production rescheduling expert simulation system. *European Journal of Operational Research*, 124:283–293, 2000.

[LLP97] C.Y. Lee, L. Lei, and M. Pinedo. Current trends in deterministic scheduling. *AOR*, 70:1–41, 1997.

[LLRKS93] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys. Sequencing and scheduling: Algorithms and complexity. In S.S. Graves, A.H.G. Rinnooy Kan, and P. Zipkin, editors, *Handbooks in Operations Research and Management Science*, volume 4, pages 445–522. Logistics of Production and Inventory, North-Holland, New York, 1993.

[LRK79] J.K. Lenstra and A.H.G. Rinnooy Kan. Computational complexity of discrete optimization problems. *Annals of Discrete Mathematics*, 4:121–140, 1979.

[LRKB77] J.K. Lenstra, A.H.G. Rinnooy Kan, and P. Brucker. Complexity of machine scheduling problems. *Annals of Discrete Mathematics*, 1:343–362, 1977.

[LS99] T.-C. Lai and Yu.N. Sotskov. Sequencing with uncertain numerical data for makespan minimization. *Journal of the Operations Research Society*, 50:230–243, 1999.

[LSSW97] T.-C. Lai, Yu.N. Sotskov, N.Y. Sotskova, and F. Werner. Optimal makespan scheduling with given bounds of processing times. *Mathematical and Computer Modelling*, 26(3):67–86, 1997.

[LSSW98] T.-C. Lai, Yu.N. Sotskov, N.Y. Sotskova, and F. Werner. Mean flow time minimization with uncertain processing times. Otto-von-Guericke-Universität, FMA, Preprint No. 15/98, Magdeburg, Germany, 1998.

[LvdPSvdV96] M. Libura, E.S. van der Poort, G. Sierksma, and J.A.A. van der Veen. Sensitivity analysis based on k-best solutions of the traveling salesman problems. Technical Report 96A14, Research Institute Systems, Organizations and Management, University of Groningen, The Netherlands, Groningen, 1996.

[LW91] L. Lei and T.J. Wang. The minimum common-cycle algorithm for cycle scheduling of two hoists with time window constraints. *Management Science*, 37:1629–1639, 1991.

[LWS94] V.J. Leon, S.D. Wu, and R.H. Storer. Robustness measures and robust scheduling for job shops. *IIE Transactions*, 26:32–43, 1994.

[Mel78] O. I. Melnikov. Optimal schedule stability for the Bellman-Johnson problem. *Vesti Akad. Navuk BSSR, Ser. Fiz.-Mat. Navuk.*, 6:99–101, 1978. (In Russian).

[ML93] B.L. MacCarthy and J. Liu. Addressing the gap in scheduling reserach: A review of optimization and heuristic methods in production research. *International Journal of Production Research*, 31:59–79, 1993.

[MVZ95] J.M. Mulvey, R.J. Vanderbei, and S.A. Zenios. Robust optimization of large-scale systems. *Operations Research*, 43:264–281, 1995.

[NK04] C.T. Ng and M. Y. Kovalyov. An fptas for scheduling a two-machine flow-shop with one unavailability interval. *Naval Research Logistics*, 51:307–315, 2004.

[OD99] E.C. Özelkan and L. Duckstein. Optimal fuzzy counterparts of scheduling rules. *European Journal of Operational Research*, 113:593–609, 1999.

[PD91] H. Parunak and Van Dyke. Characterizing the manufacturing scheduling problem. *J. Manufacturing Systems*, 10(3):241–259, 1991.

[PDK83] P.A. Pinto, D.G. Dannenbring, and B.M. Khumawala. Assembly line balancing with processing alternatives: An application. *Management Science*, 29:817–830, 1983.

[Pin95a] M. Pinedo. *Scheduling: Theory, Algorithms, and Systems*. Prentice-Hall, Englewood Cliffs, NJ, USA, 1995.

[Pin95b] E. Pinson. The job shop scheduling problem: A concise survey and some recent development. In P. Chretienne, E.G. Coffman, J.K. Lenstra, and Z. Liu, editors, *Scheduling Theory and its Applications*, pages 277–293. John Wiley & Sons, 1995.

[PL94] N.M. Paz and W. Leigh. Maintance scheduling: Issues, results, and research needs. *Internat. J. Oper. and Production Management*, 47:47–69, 1994.

[PQ78] J. Picard and M. Queyranne. The time-dependent traveling salesman problem and its application to the tardiness problem in one machine scheduling. *Operations Research*, 26:86–110, 1978.

[PZ76] C.J. Piper and A.A. Zoltners. Some easy postoptimality analysis for zero-one programming. *Management Science*, 22(7):759–765, 1976.

[RC95] R. Ramaswamy and N. Chakravarti. Complexity of determining exact tolerances for min-sum and min-max combinatorial optimization problems. Technical Report WPS-247/95, Indian Institute of Management, Calcutta, 1995.

[RL87] J.J. Rosenblatt and H.L. Lee. A robustness approach to facilities design. *International Journal of Production Research*, 25:479–486, 1987.

[RR96] D.J. Robb and T.R. Rohleder. An evaluation of scheduling heuristics for dynamic single-processor scheduling with early/tardy costs. *Naval Research Logistics*, 43:349–364, 1996.

[RS64] B. Roy and B. Sussmann. Les problèmes d'ordonnancement avec contraintes disjonctives. Note DS No. 9 bis, SEMA, Montrouge, 1964.

[San95] E. Sanlaville. Nearly on line scheduling of preemptive independent tasks. *Discrete Applied Mathematics*, 57:229–241, 1995.

[Sch99] A. Scholl. *Balancing and Sequencing of Assembly Lines*. Physica-Verlag, A Springer-Verlag Company, Heidelberg, 1999.

[Sch00] G. Schmidt. Scheduling with limited machine availability. *European Journal of Operational Research*, 121:1–15, 2000.

[SD01] Yu.N. Sotskov and A. Dolgui. Stability radius of the optimal assembly line balance with fixed cycle time. In *Proceedings of the IEEE Conference ETFA2001*, pages 623–628, Troyes, France, 2001.

[SDP05] Yu.N. Sotskov, A. Dolgui, and M.-C. Portmann. Stability analysis of optimal balance for assembly line with fixed cycle time. *EJOR*, 2005. ( ).

[SEDE99] S.C. Sarin, E. Erel, and E.M. Dar-El. A methodology for solving single-model, stochastic assembly line balancing problem. *International Journal of Management Science - OMEGA*, 27:525–535, 1999.

[Sei98] S.S. Seiden. Randomized online interval scheduling. *Operations Research Letters*, 22:171–177, 1998.

[Sga98] J. Sgall. On-line scheduling. In A. Fiat and G.J. Woeginger, editors, *On-Line Algorithms, Lecture Notes in Computer Science*. Springer, Berlin, 1998.

[SH99] R. Slowinski and M. Hapke. *Scheduling Under Fuzziness*. Physica-Verlag, Heidelberg, New York, Heidelberg, New York, 1999.

[SJ97] C. Stein and W. Joel. On the existence of schedules that are near-optimal for both makespan and total weighted completion time. *Operations Research Letters*, 21:115–122, 1997.

[SK94] E. Szelke and R.M. Kerr. Knowledge-based reactive scheduling. *Production Planning and Control*, 5(2):124–145, 1994.

[SK98] A. Scholl and R. Klein. Balancing assembly lines effectively: A computational comparison. *European Journal of Operational Research*, 114:51–60, 1998.

[SK00] M. Sakawa and R. Kubota. Fuzzy programming for multiobjective job shop scheduling with fuzzy processing time and fuzzy duedate through genetic algorithms. *European Journal of Operational Research*, 120:393–407, 2000.

[SLG95] Yu.N. Sotskov, V.N. Leontev, and E.N. Gordeev. Some concepts of stability analysis in combinatorial optimization. *Discrete Applied Mathematics*, 58:169–190, 1995.

[SOM+90] S.F. Smith, P.S. Ow, N. Muscettola, J. Potvin, and D.C. Matthy. An integrated framework for generating and revising factory schedules. *Journal of the Operations Research Society*, 41(6):539–552, 1990.

[Sot91a] Yu.N. Sotskov. The complexity of shop-scheduling problems with two or three jobs. *European Journal of Operational Research*, 53:326–336, 1991.

[Sot91b] Yu.N. Sotskov. Stability of an optimal schedule. *European Journal of Operational Research*, 55:91–102, 1991.

[Sot93] Yu.N. Sotskov. The stability of the approximate boolean minimization of a linear form. *U.S.S.R. Comput. Math. and Math. Phys.*, 33(5):699–707, 1993.

[Sot99a] N.Y. Sotskova. Optimal makespan schedules for a job-shop with uncertain processing times. In M.F. Carvalho and F.M. Müller, editors, *Proceedings of the 15th International Conference on CAD/CAM, Robotics & Factories of the Future CARS & FOF99*, volume 1, pages MW4:7 – MW4:12, Águas de Lindóia - Brazil, August 18-20, 1999. Technological Center for Informatics Foundation Automation Institute, Campinas-SP-Brazil.

[Sot99b] N.Y. Sotskova. Optimization in a job shop with given bounds of processing times. In J. Soldek and J. Pejas, editors, *Proceedings of the Sixth International Conference on Advanced Computer Systems ACS'99*, pages 462–467, Szczecin - Poland, November 18-19, 1999. Technical University of Szczecin.

[Sot99c] N.Y. Sotskova. Solution of an uncertain job-shop problem with regular criterion. In *Book of Abstracts. Symposium on Operations Research SOR'99*, page 103, Magdeburg - Germany, September 1 - 3, 1999. Otto-von-Guericke-Universität Magdeburg.

[Sot00] N.Y. Sotskova. Two-stage scheduling optimization of requirements maintenance with different routes. In *Proceedings of the Eighth Belarussian Mathematical Conference*, page 24, Minsk, Belarus, June 19-24, 2000. (In Russian).

[SS95] Yu.N. Sotskov and N.V. Shakhlevich. NP-hardness of shop-scheduling problems with three jobs. *Discrete Applied Mathematics*, 59:237–266, 1995.

[SS98] E. Sanlaville and G. Schmidt. Machine scheduling with availability constraints. *Acta Informatika*, 35:795–811, 1998.

[SSW97] Yu.N. Sotskov, N.Y. Sotskova, and F. Werner. Stability of an optimal schedule in a job shop. *OMEGA - International Journal of Management Science*, 25(4):397–414, 1997.

[Str95] V.A. Strusevich. Two machine flow shop scheduling problem with no wait in process: Controllable machine speeds. *Discrete Applied Mathematics*, 59:75–86, 1995.

[Str99] V.A. Strusevich. A heuristic for the two-machine open-shop scheduling problem with transportation times. *Discrete Applied Mathematics*, 93:287–304, 1999.

[STW98] Yu.N. Sotskov, V.S. Tanaev, and F. Werner. Stability radius of an optimal schedule: A survey and recent developments. In G. Yu, editor, *Industrial Applications of Combinatorial Optimization*, volume 16, pages 72–108. Kluwer Academic Publishers, Boston, MA, 1998.

[Sus72] B. Sussmann. Scheduling problems with interval disjunctions. *Mathematical Methods of Operations Research*, 16:165–178, 1972.

[SW80] D.R. Shier and G. Witzgall. Arc tolerances in shortest path and network flow problems. *Networks*, 10:277–291, 1980.

[SW00] N.Y. Sotskova and F. Werner. Dominance relations on the set of schedules for uncertain job-shop. In J.-M. Proth and V. Tanaev, editors, *Proceedings of the International Workshop on Discrete Optimization Methods an Scheduling and Computer-aided Design*, pages 99–101, Minsk, Belarus, September 5-6, 2000.

[SWW98] Yu.N. Sotskov, A.P.M. Wagelmans, and F. Werner. On the calculation of the stability radius of an optimal or an approximate schedule. *Annals of Operations Research*, 83:213–252, 1998.

[Tar82] R.E. Tarjan. Sensitivity analysis of minimum spanning trees and shortest path trees. *Information Processing Letters*, 14:30–33, 1982.

[TGK95] Y. Tsujimura, M. Gen, and E. Kubota. Solving fuzzy assembly-line balancing problem with genetic algorithms. *Computers and Industrial Engineering*, 29(1-4):543547, 1995.

[Tri94] M.A Trick. Scheduling multiple variable-speed machines. *Operations Research*, 42:234–248, 1994.

[TSS94] V.S. Tanaev, Yu.N. Sotskov, and V.A. Strusevich. *Scheduling Theory: Multi-Stage Systems*. Kluwer Academic Publishers, Dordrecht, The Netherlands, The Netherlands, Dordrecht, 1994.

[vdP97] E.S. van der Poort. *Aspects of Sensitivity Analysis for the Traveling Salesman Problem*. PhD thesis, Graduate School/Research Institute Systems, Organizations and Management, University of Groningen, P.O. Box 800, Groningen, The Netherlands, 1997.

[vHW99] S. van Hoesel and A. Wagelmans. On the complexity of postoptimality analysis of 0/1 programs. *Discrete Applied Mathematics*, 91:251–263, 1999.

[Wag95] H.M. Wagner. Global sensitivity analysis. *Operations Research*, 43:948–969, 1995.

[WBS99] S.D. Wu, E.-S. Byeon, and R.H. Storer. A graph-theoretic decomposition of the job shop scheduling problem to achieve scheduling robustness. *Operations Research*, 47(1):113–124, 1999.

[WJ88] G.R. Wilson and H.K. Jain. An approach to postoptimality and sensitivity analysis of zero-one goal programs. *Naval Research Logistics*, 35:73–84, 1988.

[YH79] T. Yoshida and K. Hitomi. Optimal two-stage production scheduling with setup times separated. *AIIE Transactions*, 11:261–263, 1979.

Zusammenfassung

Scheduling ist ein bedeutender Bestandteil des Decision-Making in Unternehmen vieler Industriezweige. Jedoch ist die Nutzung verschiedener Resultate der klassischen Scheduling-Theorie bei praktischen Problemstellungen aus der Industrie bisher sehr begrenzt. Insbesondere schränkt die in der Scheduling-Theorie übliche Voraussetzung, da die Bearbeitungszeiten der Operationen vor Nutzung eines Scheduling-Algorithmus bekannt sind, die praktische Anwendung ein. Diese Dissertation ist ein Versuch, einige theoretische Resultate für breitere Anwendungen nutzbar zu machen.

Neben den üblichen Voraussetzungen, da die strukturellen Daten fixiert sind und da zu jeder Zeit eine Maschine nur einen Auftrag bearbeiten kann und jeder Auftrag auf höchstens einer Maschine bearbeitet werden kann, wird in dieser Dissertation die folgende Annahme getroffen: Die tatsächliche Bearbeitungszeit p_i einer Operation i kann jeden reellen Wert zwischen einer gegebenen unteren Schranke a_i und einer gegebenen oberen Schranke b_i annehmen. Ein Job-Shop Problem mit derartigen Unsicherheiten in den Eingangsdaten wird im Fall der Minimierung der Gesamtbearbeitungszeit (Makespan) mit $\mathcal{J}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ bezeichnet. Gilt für jede Operation $i \in Q = \{1, 2, \ldots, q\}$ eines Problems $\mathcal{J}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ die Beziehung $a_i = b_i$, so liegt ein deterministisches Job-Shop Problem vor, andernfalls ein stochastisches Problem ohne a priori Information über die Verteilungen der Zufallsgröen der Bearbeitungszeiten.

Für Problem $\mathcal{J}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ mu nicht notwendig ein eindeutig bestimmter Plan existieren, der für alle möglichen Realisierungen der Bearbeitungszeiten optimal bleibt. Daher mu für ein Problem mit unsicheren Eingangsdaten eine Menge semiaktiver Pläne konstruiert werden, die die Menge der restlichen Pläne für das gegebene Zielkriterium dominiert. Um eine solche Menge von Plänen zu bestimmen, wird eine Stabilitätsanalyse eines optimalen Planes bezüglich der möglichen Störungen der Bearbeitungszeiten durchgeführt. Es wird der Begriff des relativen Stabilitätsradius eines optimalen Plans eingeführt. Dabei wird die Relativität bezüglich des Polytops $T = \{p' = (p'_1, p'_2, \ldots, p'_q) : a_i \leq p'_i \leq b_i, i \in Q\}$ der zulässigen Vektoren p' der Bearbeitungszeiten und bezüglich der Teilmenge B von semiaktiven Plänen, für die die Überlegenheit eines verfügbaren Planes garantiert werden mu, betrachtet. In der Arbeit wird das gemischte (disjunktive) Graphenmodell benutzt, welches für den gesamten Scheduling Proze von dem gemischten Ausgangsgraphen bis zum Graphen G_s, der einen vollständigen semiaktiven Plan s repräsentiert, geeignet ist.

Die formale Definition des relativen Stabilitätsradius ist wie folgt. Sei $O_\varrho(p)$ eine abgeschlossene Kugel im Raum R^q der q-dimensionalen reellen Vektoren. Angenommen, der Plan $s \in B$ habe für jeden Vektor $p' \in O_\varrho(p) \cup T$ von Bearbeitungszeiten eine minimale Gesamtbearbeitungszeit (Makespan). Dann wird der maximale Wert des Radius ϱ einer

solchen Kugel als relativer Stabilitätsradius bezeichnet und mit $\varrho_s^B(p \in T)$ abgekürzt. In der Dissertation werden notwendige und hinreichende Bedingungen für den Fall $\varrho_s^B(p \in T) = 0$ und für den Fall eines unendlich groen Wertes von $\varrho_s^B(p \in T)$ abgeleitet.

Die abgeleitete Formel für die Berechnung des exakten Wertes von $\varrho_s^B(p \in T)$ basiert of dem Vergleich des Planes s mit anderen zulässigen Plänen von der Menge B, und es wird gezeigt, wie man die Anzahl der zu betrachtenden Digraphen der Menge B bei diesem Vergleich reduzieren kann. hnliche Resultate wie zuvor für das Makespan-Kriterium beschrieben wurden für den Fall der Minimierung der Summe der Bearbeitungsendtermine der Aufträge (Mean Flow Time) erhalten. Insbesondere wurden notwendige und hinreichende Bedingungen für einen unendlich groen Wert von $\varrho_s^B(p \in T)$ im Fall des Mean-Flow-Time Kriteriums abgeleitet.

Auf der Grundlage dieser Resultate wurden exakte und heuristische Algorithmen zur Lösung der Job-Shop Probleme $\mathcal{J}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ und $\mathcal{J}/a_i \leq p_i \leq b_i/\sum \mathcal{C}_i$ und des General-Shop Problems $\mathcal{G}/a_i \leq p_i \leq b_i/\mathcal{C}_{max}$ abgeleitet. Die entwickelte Software wurde an zufällig erzeugten Job-Shop Problemen getestet.

Curriculum Vitae

Nadezhda Sotskova was born in May 21, 1975, in Minsk (Belarus). From 1982 to 1992 she studied in the Secondary School in Minsk (mathematical class). From 1992 to 1997 she studied mathematics at the Belarussian State University (Department of Applied Mathematics and Computer Science). She graduated in June 1997, on the subject of economical cybernetics, and got qualification of a mathematician-economist and qualification of a teacher of mathematics and computer science. The title of her Master's thesis was "The experimental investigations of the stability radius of the optimal graphs (schedules)". In January 1998 she started as a PhD student at the University of Magdeburg (Department of Mathematics, Institute of Mathematical Optimization).

List of Tables

List of Figures