Proceedings of the 1991 IEEE
International Conference on Robotics and Automation
Sacramento, California - April 1991

SCHEDULING FLEXIBLE FLOW SHOPS OF NO SETUP COST BY A LAGRANGIAN RELAXATION AND NETWORK FLOW APPROACH*

Shi-Chung Chang, Da-Yin Liao, Fu-Shiung Hsieh

Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan, R.O.C.

ABSTRACT

This paper presents further developments of a production scheduling algorithm ([2]) for the class of discrete–part, make–to–order flexible flow shops where setup costs and times are negligible. The scheduling problem is first formulated as a large–scale integer programming problem and a solution approach based on Lagrangian relaxation and minimum cost linear network flow is then developed. As compared to [2], this paper modifies the objective of scheduling to meeting due dates just in time, considers finite buffers, completes the algorithm for finding a good feasible schedule and evaluates the algorithm through numerical experimentations. Numerical results indicate that our scheduling algorithm is near–optimal and has a reasonable computational efficiency for short term scheduling. Algorithmic features and future research issues are also addressed.

1.INTRODUCTION

Scheduling has been well recognized as a very important but difficult part of production control. Although there have been numerous researches on this topic in the literature [5], there exists quite a gap between scheduling theory and practice. Due to the complexity of production operations, finding the optimal schedule efficiently is often beyond the reach of existing theories even for many small–scale systems. In practice, schedules are usually obtained either by simple heuristics but with questionable performance or through time consuming simulations. Today, as many new manufacturing technologies have been developed and installed [11], there are needs for new concepts and methodologies to effectively schedule the operations of modern factories.

In this paper, we consider the production scheduling problem of a make–to–order flexible flow shop, which manufactures medium–volume discrete products. Each type of products has its own due date, desired quantity and associated production process. Homogeneous machines in the shop are put into groups and there are finite buffers among them for intermediate part storage. The production operations of different products may require processing from one same machine group but there is no revisit to it after a part is processed there except when rework is needed. Namely, there are basically no cycles in production flow paths. Setup times and costs are negligible. For a given set of orders, our objective of scheduling is to meet the due dates just in time while satisfying constraints of (1) machine capacity, (2) buffer capacity, (3) end product demand and (4) precedence relationship of manufacturing processes.

We develop an effective methodology for scheduling such a flexible shop based on our earlier study [2]. It consists of four parts :

(1) the dual problem formulation and decomposition of the original problem into subproblems according to part types by applying Lagrangian relaxation to machine and buffer capacity constraints;
(2) application of a minimum cost linear network flow (MCLNF) algorithm to solve each subproblem;
(3) application of a nondifferentiable optimization scheme to solve the dual problem, and
(4) development of a heuristic algorithm that finds a near–optimal, feasible solution based on the solution of the relaxed problem.

Numerical experimentations are conducted to examine both the optimality and computational efficiency of our algorithm. Results indicate that our

* This work was supported in part by the National Science Council of Republic of China under Grants 78-0422-E002-09 and 79-0422-E002-05.

algorithm is feasible and tends to obtain optimal schedules for very small–scale problems. As problem dimensionality increases, the gap between the dual cost and the cost of the feasible schedule usually stays around 10% but may grow up to 30% in cases of low number of product types but high quantity for each type. We find the computational efficiency of the algorithm quite reasonable for short–term scheduling and is most sensitive to scheduling horizon and number of operations per type. We also compare our algorithm with a heuristic scheduling algorithm of a few local manufacturers [4]. All these results demonstrate the potential of our algorithm to be further developed into an effective short–term scheduling tool.

The remainder of this paper is organized as follows. The scheduling problem is formulated in Section 2 and the development of solution algorithm is described in Section 3. Section 4 gives results of our numerical experimentations and addresses features of our algorithm. Concluding remarks and issues for further investigation are discussed in Section 5.

2. PROBLEM FORMULATION

Consider a make–to–order flexible flow shop as described in the previous section. We shall refer to both products and intermediate parts as parts from here on. We assume that

(1) the setup times and costs can be neglected;
(2) there are no initial in–process inventories in the shop;
(3) the capacity for handling rework has been deducted from each machine group; and
(4) the extra processing requirements due to potential scraps have been added to the original production demands.

To focus on our main concept and without loss of generality in developing solution methodology, we assume that all types of parts require the same sequence of processing, i.e., the M machine groups can be organized as a line in Figure 2.1, where buffer m locates before machine group m and buffer M+1 represents the stock of finished parts. Both the first and the last buffers are infinite in size. We also assume that all the demands are released at the beginning of the scheduling horizon.

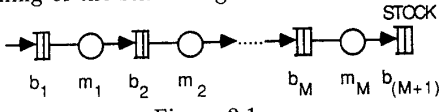

Figure 2.1

Let us now define some notations for describing such a flexible flow shop.

Notations

I : total number of part types;
i : part type index, $i = 1, \cdots, I$;
D_i : demand of type i parts;
d_i : due date of type i parts;
M : total number of machine groups;
m : machine group index, $m = 1, \cdots, M$;
C_m : capacity of machine group m;
P_{im} : processing time of a type–i part on machine group m;
T : scheduling time horizon;
t : time index, $t = 1, \cdots, T$;
b : buffer index, $b = 1, \cdots, M+1$;
S_b : capacity of buffer b;
X_{ibt} : number of type i parts in buffer b at the beginning of time period t;
u_{imt} : number of type i parts loaded onto machine group m for processing at time t;
Z_{it} : number of type i parts arriving at the stock at period t, where $Z_{it} = u_{iM(t-P_{iM}+1)}$.

In the flow line, a batch of type i parts loaded onto machine group (m–1) for processing at period $t-P_{i(m-1)}$ go into buffer m at period t after $P_{i(m-1)}$ periods of processing. The first buffer of the flow line serves as a source with initial level D_i of type i parts while the last buffer serves as a sink and accumulates finished parts. Flows of parts are not compressible and must satisfy the following flow balance equations:

Flow Balance Equations

$$X_{i11} = D_i; \tag{2.1.a}$$

$$X_{i1(t+1)} = X_{i1t} - u_{i1t}; \tag{2.1.b}$$

$$X_{im(t+1)} = X_{imt} - u_{imt} + u_{i(m-1)(t-P_{i(m-1)})}, \quad m=2, \ldots, M; \tag{2.1.c}$$

$$X_{i(M+1)(t+1)} = X_{i(M+1)t} + u_{iM(t-P_{iM})}; \tag{2.1.d}$$

for $t = 1, \ldots, T-1$ and $i = 1, \ldots, I$.

The quantity $\sum_{i=1}^{I} \sum_{\tau=t-P_{im}+1}^{t} u_{im\tau}$ is the total

number of parts being processed by machine group m during time period t and must not exceed the processing capacity, i.e.,

Machine Capacity Constraints

$$\sum_{i=1}^{I} \sum_{\tau=t-P_{im}+1}^{t} u_{im\tau} \le C_m, \quad \forall\, t, \forall\, m. \quad (2.2)$$

Similarly, the total number of parts in buffer b during time period t must satisfy

Buffer Capacity Constraints

$$\sum_{i=1}^{I} X_{ibt} \le S_b, \qquad \forall\, b, \forall\, t, \quad (2.3)$$

where we assume that all parts take the same amount of buffer space individually. Moreover,

$$X_{ibt} \text{ and } u_{imt} \text{ are nonnegative integers,} \qquad \forall\, i, \forall\, m, \forall\, b, \forall\, t. \quad (2.4)$$

Our objective of production scheduling has two folds : (1) to meet due dates if possible and (2) to produce just in time. Penalty costs are incurred by early and tardy productions[3]. We therefore formulate the production scheduling problem as

$$(P) \qquad \min_{\mathbf{u}} \sum_{i=1}^{I} \sum_{t=1}^{T} \psi_{it} Z_{it}$$

subject to constraints (2.1—2.4),

where the cost function ψ_{it} is chosen such that

$$\psi_{it} = \begin{cases} B_i(d_i - t), & t \le d_i \\ A_i(t - d_i), & t > d_i \end{cases},$$

and B_i and A_i are positive penalty coefficients on earliness and tardiness respectively.

3. SOLUTION ALGORITHM

The production scheduling problem (P) formulated above is NP hard[10, Sec. 4.2]. In this section, we describe a near—optimal but computationally efficient solution algorithm. We shall elaborate on the part of finding a good feasible solution from the dual solution and briefly describe the steps for solving the dual problem since the latter ones are similar to those in [2].

3.1 Lagrangian Relaxation and Decomposition

Observing that the coupling among production flows of different part types in problem (P) is through their competition for processing and storage resources, therefore, we apply Lagrangian relaxation to machine and buffer capacity constraints (2.2) and (2.3) and form the dual problem of (P) as

$$(D) \qquad \max_{\boldsymbol{\lambda} \ge 0,\, \boldsymbol{\pi} \ge 0} \Phi(\boldsymbol{\lambda}, \boldsymbol{\pi}),$$

subject to (2.1) and (2.4),

where λ_{mt} and π_{bt} are the associated Lagrange multipliers,

$$\Phi(\boldsymbol{\lambda}, \boldsymbol{\pi}) \equiv \sum_{i=1}^{I} \min_{\mathbf{u}_i} L_i(\mathbf{u}_i, \boldsymbol{\lambda}, \boldsymbol{\pi}) - \sum_{t=1}^{T} \sum_{m=1}^{M} \lambda_{mt} C_m - \sum_{t=1}^{T} \sum_{b=1}^{M} \pi_{bt} S_b, \quad (3.1)$$

and

$$L_i(\mathbf{u}_i, \boldsymbol{\lambda}, \boldsymbol{\pi}) \equiv \sum_{t=1}^{T} \Big(\psi_{it} Z_{it} + \sum_{m=1}^{M} \lambda_{mt} \sum_{\tau=t-P_{im}+1}^{t} u_{im\tau} + \sum_{b=1}^{M} \pi_{bt} X_{ibt} \Big). \quad (3.2)$$

Note that for a given set of $\boldsymbol{\lambda}$ and $\boldsymbol{\pi}$, i.e., after relaxing machine and buffer capacity constraints, the scheduling subproblem for type i parts is

$$(P\text{—}i) \qquad \min_{\mathbf{u}_i} L_i(\mathbf{u}_i, \boldsymbol{\lambda}, \boldsymbol{\pi})$$

subject to (2.1) and (2.4) for type i only.

All these scheduling subproblems can be solved independently from each other.

3.2 A Network Model for Subproblems

It can be seen after a little thinking that material flow equations (2.1) of (P—i) render themselves naturally to a network representation. Each node of the network corresponds to one flow balance equation. The arcs represent either part processing paths with u_{imt}'s as flows on them or parts carried over in buffers between two time periods with X_{imt}'s as flows. Buffer 1 corresponds to the source node while buffer M+1 corresponds to the sink. Machine and buffer capacity constraints impose flow bounds on the arcs.

Figure 3.1 illustrate the network representation of a 2—machine, 3—buffer production line over a 5—period time horizon, where the processing time at machine group 1 is two units and one at machine group 2. Furthermore, the objective function of (P—i)

is linear in arc—flows. Thus, a subproblem (P—i) is essentially a minimum cost linear network flow (MCLNF) problem, which has an integer optimal solution [7]. Interested readers may refer to [2] for more details. We adopt here the RELAX code of Bertsekas and Tseng [1] to solve each subproblem.

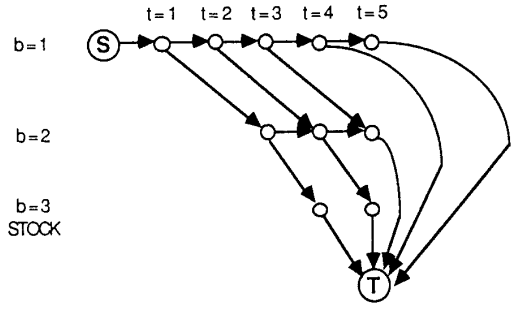

Figure 3.1

3.3 Solving the Dual Problem

After solving all the subproblems for a given set of Lagrange multipliers (λ,π), we use the obtained solution to update (λ,π). Aware of the non—differentiability of the dual objective function Φ due to the integrality in subproblems, we calculate the subgradient of Φ with respect to (λ,π) and update (λ,π) according to the subgradient method of [8] and [6]. We then solve the dual problem by iteratively conducting 3.2 and 3.3. Please also refer to [2] for more details.

3.4 Construction of a Good Feasible Schedule

The primal problem (P) is not a convex optimization problem because of discrete decision variables. So the schedule obtained from solving the dual problem is generally infeasible, i.e., some of the relaxed capacity constraints (2.2) or (2.3) may not be satisfied. However, the dual solution does provide a lower bound to the optimum. We now develop an iterative, heuristic algorithm that adjusts the dual solution to a near—optimal and feasible schedule. Major steps of the heuristic are as follows.

Algorithm

Step 0 Initialize with the schedule obtained from solving the dual problem.

Do for t from *1* to T

Do for m from *1* to M

Step 1 Check if capacity constraints of machine m and the associated buffer are violated at time period t.

Step 2 If so, determine, according to the descending order of priority factors (PFs) among different types, the type(s) of parts that should be removed from facility (machine or buffer) m at time period t to eliminate the excessive production flow. The definition of PF is given in the APPENDIX.

Step 3 Remove the excessive production flow(s) from the production schedule. Since the type(s) of parts to remove is identified in *Step 2*, we consider again the material flow network of the type and focus on the upstream and downstream subnets of the arc with the excessive flow. We pull the excessive amount of flow out of both subnets in a way that results in minimum production cost change (i.e. solving a MCLNF problem for each subnet). We then update the material flow network according to the removal.

Step 4 Reschedule the removed production flow. This is done by rerouting the flow in minimum cost through the material flow network obtained in *Step 3* (a MCLNF problem again). Update the production schedule.

Enddo

Enddo

Note that in all the MCLNF problems encountered in the above feasibility adjustment algorithm, arc costs are the same as those of the dual solution, i.e., we adjust only the primal variables but not the dual variables.

4. NUMERICAL RESULTS

Numerical experimentations are conducted in this section to demonstrate the feasibility, optimality and computational efficiency of our algorithm. We first apply the algorithm to a very simple example whose optimal schedule can be obtained by observation. We then test it against a set of cases designed from the data of a local manufacturer in order to study its algorithmic properties. All of our experimentations are performed on a SUN/SPARC— IPC workstation.

In addition, we compare the performance of our algorithm with that of a heuristic algorithm [4]. The heuristic assumes infinite buffer capacity and intends to shorten the total makespan. It breaks production demands into work orders(WOs) and assign WOs one by one to machines for processing according to the earliest starting dates and PF values (APPENDIX)

of WOs; the lower the PF value of a WO, the higher its priority in getting assigned.

4.1 A Simple Example

There are 3 machine groups and 2 types of parts. Processing requirements and system capacity data are given in Tables 4.1 and 4.2. We set cost function coefficients A_i=20 and B_i=1 for all part types. Note that from the scheduling point of view, the only difference between the two types is the due date.

In applying our algorithm to this example, we initialize all the multipliers as zero and set initial estimate of the optimal cost to 1000.0. The step size adjustment parameter in the subgradient method is 1.9. The resultant(feasible) schedule is shown in a Gantt chart (Figure 4.1) with production cost 115.0 and computation time 0.63 CPU seconds. It is easy to see that the schedule is optimal, which is consistent with our earlier results in [2].

Table 4.1

Part	Quantity	Due Date	Routing
1	25	5	m1 -> m2 -> m3
2	25	9	m1 -> m2 -> m3*

* The notation "mn" refers to machine group n.

Table 4.2

Machine Group		1	2	3
Machine Capacity		10	20	15
Buffer Capacity		100	20	15
Proc. Time	Part 1	1	2	1
	Part 2	1	2	1

Time	1	2	3	4	5	6	7	8	9	10
m1	1,10*	1,10	1,5	2,10	2,10	2,5				
m2		1,10	1,10	1,5	2,5	2,10	2,10			
m3				1,5	1,15	1,5		2,10	2,15	

* i,q : q units of type i parts

Figure 4.1 Schedule from our algorithm

Time	1	2	3	4	5	6	7	8	9	10
m1	1,10	1,10	1,5 2,5	2,10	2,10					
m2		1,10	1,10	1,5 2,5	2,10	2,10				
m3				1,10	1,10	1,5 2,5	2,10	2,10		

Figure 4.2 Schedule from the heuristic

4.2 Algorithmic Features

The performance of our algorithm is a function of the length of scheduling horizon(T), number of machine groups involved(M), number of part types(I) and part quantity per type. We now use a set of realistic data to create a set of 6 test scenarios and examine the effects of the aforementioned factors on solution optimality and computational efficiency.

The major computational loads of our algorithm are from solving MCLNFs and subgradient iterations. There are two known facts : (a) that the computational complexity of the RELAX code for solving a MCLNF is $O(N^3 \log NC)$, where N is the number of nodes while C is the range of arc cost coefficients, and (b) that the convergence of subgradient iterations slows down as the number of Lagrange multipliers increases. We therefore project that

(1) as the number of part type (I) increases, the number of subproblems increases, but not the dimension of each subproblem or the Lagrange multipliers; so CPU time probably increases linearly with respect to I;

(2) as the number of machine groups (M) and/or scheduling horizon (T) increase, the number of nodes in each MCLNF subproblem increases ($N = M \cdot T$) and the number of Lagrange multipliers also increase with $M \cdot T$; so the computation time may increase faster than linearly.

Numerical results are listed in Table 4.3. Comparison of scenarios S1, S2 and S6 confirms conjecture (1), and S2 vs. S5 and S1 vs. S2 justify (2).

Table 4.3

Scenario	I	M	T	CPU Time (sec)	Primal Cost	Dual Cost	Duality Gap
S1	4	7	145	2085.68	595.000	438.950	26.23%
S2	10	7	145	6040.00	1120.00	967.998	13.57%
S3	10	8	145	6490.08	1680.00	1561.30	7.07%
S4	10	8	110	4021.50	1720.00	1566.02	8.95%
S5	10	14	145	11861.40	2494.00	1684.52	32.46%
S6	20	7	145	8622.97	1975.00	1564.26	20.80%

In terms of the optimality measure, the duality gap, we observe that those of S1, S5 and S6 are not so satisfactory. After a closer examination of the

detailed schedule of S1, we find that due to the high part quantity per type, a great amount of parts of the same type compete for the processing resources during the same intervals and they cannot be differentiated in priority by our algorithm. In the dual solution to S1, there are totally 28 periods during which the resources constraints are violated in high excessive quantities. Such an observation implies that the dual solution obtained only provides a loose lower bound. As for S5 and S6, we believe that the dual solutions only converge to local maxima due to large number of dual variables. Fine tuning of the subgradient method may alleviate this problem.

5. CONCLUDING REMARKS

The scheduling algorithm we developed above has not been fine tuned. Its current performance in solution optimality and computational efficiency for application to medium–sized short–term scheduling problems such as hourly schedule over a week or daily schedule over a season is not completely satisfactory but acceptable. We believe that its performance can be further improved by fine tuning the subgradient method and carefully initializing the Lagrange multipliers with respect to specific application systems. Besides, we are extending the algorithm to handle more realistic issues such as rework, scrap, assembly and fast rescheduling. Results will be reported in an upcoming paper.

ACKNOWLEDGEMENTS

The authors would like to thank Mr. Chi-Tsai Yang and Mr. Hong-Mo Yeh of Fu-Sheng Industrial Co., and Dr. Debra Hoitomt and Prof. Peter B. Luh of University of Connecticut for their valuable discussions and comments.

APPENDIX

$$PF \equiv (CR - LB) \cdot FQ$$

where

$$CR = \frac{\text{allowance from now to due date}}{\text{residual processing time}}$$

is the critical ratio,

$$LB = \frac{\text{residual processing time}}{\text{lead time}}$$

is the load ratio, and

$$FQ = \frac{\text{finished quantity}}{\text{demand}}$$

is the finished quantity ratio.

REFERENCES

[1] Bertsekas, D. P., and P. Tseng, "Relaxation Methods for Minimum Cost Ordinary and Generalized Network Flow Problems," *Operations Research*, Vol. 36, No. 1, 1988, pp. 93–114.

[2] Chang, S. C., D. Y. Liao, F. S. Hsieh, and C. C. Yang, "Flow Shop Scheduling by a Lagrangian Relaxation and Network Flow Approach," *Proceedings of the 29th IEEE Conference on Decision and Control*, Hawaii, Dec. 1990, pp. 122–124.

[3] De, P., J. B. Ghosh and C. E. Wells, "Scheduling about a Common Due Date with Earliness and Tardiness Penalties," *Computers Opns Res.*, Vol. 17, No. 2, 1990, pp. 231–241.

[4] *Technical Memo*, Information and Administration Dept., Fu Sheng Industrial Co., San–Chung, Taiwan, 1989

[5] Grave, S. C., "A Review of Production Scheduling," *Operstions Research*, Vol. 29, July–August, 1981, pp. 646–675.

[6] Hoitomt, D. J., P. B. Luh, E. Max, and K.R. Pattipati, "Scheduling Jobs with Simple Precedence Constraints on Parallel Machines," *Control Systems Magazine*, Vol. 10, No. 2, Feb. 1990, pp. 34–40.

[7] Papadimitriou, C. H., and K. Steiglitz, *Combinatorial Optimization : Algorithms and Complexity*, Prentice–Hall, Inc., 1982.

[8] Polyak, B. T. "Minimization of Unsmooth Functionals," *USSR Computional Math. and Math. Physics*, Vol. 9, 1969, pp. 14–29.

[9] Rosenberg, E., "A Geometrically Convergent Subgradient Optimization Method for Nonlinearly Constrained Convex Programs,", *Math. Oprn Res.*, Vol. 13, No. 3, August 1988, pp. 512– 523.

[10] Syslo, M. M., N. Deo, and J. S. Kowalik, *Discrete Optimization Algorithms with PASCAL Program*, Prentice Hall, Inc., 1983.

[11] Vollmann, T. E., W. L. Berry and D. C. Whybark, *Manufacturing Planning and Control Systems*, 2nd ed., the Dow Jonew–Irwin/APICS Series in Production Management, Homewood, IL, 1988.