行政院國家科學委員會專題研究計畫　成果報告

考慮共同料限制下多目標主規劃排程問題之研究

計畫類別：個別型計畫
計畫編號：NSC93-2416-H-002-014-
執行期間：93 年 08 月 01 日至 94 年 07 月 31 日
執行單位：國立臺灣大學資訊管理學系暨研究所

計畫主持人：陳靜枝

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 29 日

行政院國家科學委員會補助專題研究計畫 ☑ 成果報告 □期中進度報告

考慮共同料限制下多目標主規劃排程問題之研究

A Heuristic Master Planning Algorithm with Multiple Objectives and Component Commonality

計畫類別：☑ 個別型計畫　　□ 整合型計畫
計畫編號：NSC 93－2416－H－002－014－
執行期間：93 年 08 月 01 日至 94 年 07 月 31 日

計畫主持人：陳靜枝
共同主持人：
計畫參與人員：

成果報告類型(依經費核定清單規定繳交)：☑精簡報告　□完整報告

本成果報告包括以下應繳交之附件：
□赴國外出差或研習心得報告一份
□赴大陸地區出差或研習心得報告一份
□出席國際學術會議心得報告及發表之論文各一份
□國際合作研究計畫國外研究報告書一份

處理方式：除產學合作研究計畫、提升產業技術及人才培育研究計畫、列管計畫及下列情形者外，得立即公開查詢
□涉及專利或其他智慧財產權，□一年□二年後可公開查詢

執行單位：國立臺灣大學資訊管理學系暨研究所

中華民國九十四年九月二十八日星期三

考慮共同料限制下多目標主規劃排程問題之研究

A Heuristic Master Planning Algorithm with Multiple Objectives and Component Commonality

Ching-Chin Chern
Dept. of Information Management, National Taiwan University
50, Lane 144, Sec. 4, Keelung Road, Taipei 10625, Taiwan

摘 要

本研究針對包含多個最終成品的供應鏈網路圖形，且多個最終產品之間有共用性物料的使用。本演算法具有兩個主要目標(1).在考慮成品之產品架構、以及有限產能限制下，規劃與安排未來所有的訂單，選擇適當時間交由適當的廠商生產，希望達到最小生產與運送成本，以及最少訂單的延遲交貨時間，達到供應鏈最佳化的效果。(2).在考慮共用性物料的使用下，針對共用性資源的分配提出考慮公平性的做法。本研究的方法主要分為五大步驟，首先將具有不同生產程序之節點分離，執行產能初始化，第二步驟則是將所有產能轉換為以最終產品為產能單位。第三步驟，根據使用者需求選擇適合訂單排序方式與參數，第四步驟，根據訂單所要求之最終產品從網路結構中取出有效的子網路，第五步驟,再經過訂單排序後便可執行核心的三大演算法，貪婪演算法、平均分配演算法、以及比例分配演算法，每筆訂單依序規劃排程。在對每一張訂單規劃排程時，找出最小成本之廠商組合，然後尋找與安排適當生產量，當此廠商組合無法滿足需求時，調整供應鏈網路圖形以尋找次佳之廠商組合，不斷重複上述步驟直到訂單需求滿足或供應鏈已無任何產能幫助生產為止。當交貨時距限制下需求仍未滿足，延遲一個交貨時距，重複上述之方法，直到訂單需求完全滿足為止。至於產能分配的方式，貪婪演算法會依序將每一張訂單的需求滿足為止；平均分配演算法會將所有會使用同一期同一資源的訂單需求作一個平均的產能分配。最後，比例分配演算法會將所有會使用同一期同一資源的訂單需求依據需求比例分配。

關鍵字：供應鏈管理、先進規劃排程、主規劃排程、線性規劃、啟發性演算法、共用料

ABSTRACT

This study proposes a heuristic algorithm to solve a general master-planning problem of a supply chain network with multiple final products. The objectives of this planning algorithm are: (1)To minimize the processing, transportation , and inventory costs under the constraints of the capacity limits of all the nodes in a given supply chain network graph and the quantity and due day requirements of all the orders; (2)To lower the impact of fairness problem of greedy capacity allocation. This study assumed that multi-finished items are made and shipped on the given supply chain which results in common parts on common nodes for different finished items. Three different ways are proposed to solve the sharing capacity problem caused by common components: greedy, average capacity, and proportional capacity. All the three algorithms are composed of five steps: (1) Split nodes in the supply chain network graph by different functions the nodes perform, and set the initial capacities of all nodes; (2) Transform the capacity units shown on the graph, based on the unit of the final finished product; (3) Sort all the orders by adopting a rule-based sorting method to decide the scheduling sequence; (4) Extract sub-networks from original networks according to final product structure of orders; (5) Finally, for each order, find a minimum cost production tree under the constraints of the order's due date. Then, compute the maximum available capacity of this combination and arrange the suitable quantities of production and transportation. If the demand cannot be fulfilled before the due date, the order will have to be postponed. Repeating the process above until the demand is completely fulfilled. The differences among three different ways of sharing capacities lie on the quota they can allocate for each order: original capacity, average capacity and proportional capacity. These three algorithms result in the same optimal solution as the one by "Linear Programming" in eight different dimensions of scenarios when no delayed orders present. In the four cases with delayed orders, the three orders will still work out a near-optimum solution in a shorter time.

Keywords: Advance Planning and Scheduling (APS), Master Planning, Multiple-Objective Linear Programming, Product Commonality

1. Introduction

A supply chain consists of many different partners, such as material vendors, manufacturers, distribution centers, and retailers. In the supply chain operations, these different partners have many objectives that might be conflicted with each other. Manufacturers want to produce mass amount of products to take advantage of the economic scale while retailers want to customize products to quickly respond to the consumers' requirements. The conflict between different objectives makes the optimization of the supply chain operations very difficult if not impossible. [20] To cope with the new challenges of supply chain management, Advance Planning and Scheduling (APS) is developed [7, 16, 20]. APS combines Material Requirement Planning with Capacity Requirement Planning to create feasible production plans. A complete APS system consists of four major modules [7, 20]: (1) Strategic planning; (2) Demand planning; (3) Master planning; and (4) Factory planning. To solve the problems related to these four planning modules, APS needs to build large mathematical models and apply many complicated optimizing or heuristic algorithms. Kriepl etc [16] study the APO module developed by SAP for planning and scheduling in supply chain operations. APO adopts LP models for solving Master Planning problems and uses the LP optimization engine developed by CPLEX. However, the LP model must be split into several sub-models because the scale of a complete LP model is too big. The solution from APO might not be optimal because it uses solution engines such as Constraint Programming (CP) and Genetic Algorithm (GA). Constraint Programming uses constraint propagation, tree search, forward tracking, backtracking, and consistency techniques to reduce the search region. [3, 20] The quality of the feasible solution found in CP depends heavily on the initial solution point and the feasible region reduction techniques and thus, is not stable because the objective function is not considered in CP. Lee, etc [17, 18] consider the problem of scheduling customer demands under the due date constraints and available outsourcing capacity. They develop a GA-based (Genetic Algorithm) heuristic approach to select the best machine for each operation by minimizing the make span. Kim, etc [15] did a similar study on the same problem but without considering the outsourcing capacity by using the same GA-based (Genetic Algorithm) heuristic approach. Awadh, etc [1] also adopt GA-based Algorithm to find the shortest path route to solve the process planning problem. These three studies did not address the problems faced by the GA algorithm; namely the initial solution, the fitness function, the mutation process, and the stopping condition. This study also proposes a heuristic algorithm to solve the Master Planning problem to provide a feasible and sometimes optimal solution effectively and efficiently.

Among the product commonality studies, Dogramaci [10] demonstrated that component commonality may result in savings due to reduction in the standard error of forecast. Collier [8] introduced the Degree of Commonality Index (DCI) and used statistical methods to show the relationship between the DCI and setup and holding costs. Collier [9] used the DCI and regression analysis to find a linear equation describing the effect of commonality on the safety stock. Wacker etc. [21] presented a modified index, TCCI (Total Constant Commonality Index), which is a relative measure ranging between 0 and 1 and suggested also distinguishing between within-product commonality and between-product commonality. Backer etc. [2] studied the effect of commonality on a two-product, two-level model, single period, and assumed that demand follows a uniform distribution. They minimized the number of units in stock under a required service level, and found that under commonality: 1. The total number of units in inventory was reduced; 2. The inventory level(in units) of the common component was smaller than the total inventory levels (in units) of the two components it had replaced; 3. The inventory level of the specific components (those components that were not replaced) was increased. Gerchak etc. [13] extended the research to arbitrary number of products and minimized the total inventory cost under general joint demand distributions. They show a decrease in total inventory cost caused by commonality. However, since they both assumed a single period model and didn't examine the effect on costs or profits. They did not provide answers to the question of whether commonality is beneficial when common part is more expensive than the part it replaces. Enynan and Rosenblatt [11] used the two-product, two-level inventory model of Baker etc. [2], but allowed the price of the common component to exceed the price of the components that it replaces. In this situation, it is not always desirable to introduce commonality. They proved that under some conditions using common parts will result in lower inventory costs. Eppen [12] considered the related problem of warehouse consolidation or centralization. Using a single-period model with normal demands, Eppen showed that holding and shortage costs were reduced by storing a product in a single location, rather than at several locations. The model in this paper also addressed the issue of centralization, but is more general-the common component may be more expensive. Liu [19] developed an optimization-based available-to-promise allocation model with consideration of common components. They assumed that supply chain network is 3 level structure, final product inventory cost, production cost and lead time of materials are ignored, and one order for one product. The model proved that under the condition of limited capacity, adopting common components achieves higher profits in the long run because of the significant order pooling effects. Moreover, low demand variation results in stable order frequencies of materials and higher profits. As demand variation gets higher, manufacturers achieve cost savings from significant order pooling effects. Zhang [22] constructed a mixed integer linear programming model to find an optimal stocking policy for common components including ordering cost in objective function. When the model is extended to longer period or accommodated with more complex product structure, the mixed integer linear programming can't be solved optimally due to large number of 0-1 variables. Thus, they developed a Lagrangian Relaxation-based heuristics to solve this model more efficiently. These models did not result in feasible production plans. They only discuss about some results or stocking policy model when common parts are adopted. Consequently, we will propose a planning and scheduling algorithm and consider common components.

Chen and Chern [4] have utilized the network flow algorithm such as the shortest path algorithm and the maximum flow algorithm to solve problems related to supply chain network configuration. In their studies, a general supply chain network, with a given product structure of a final finished item, is transformed into a specific network that is ready to apply the shortest path algorithm. Chern and Hsieh [5] later proposed a heuristic master planning algorithm adopting Chen and Chern's [3] algorithm as a core solution engine. Chern and Hsieh [5] did not consider inventory costs and delay penalties in their model. Chern and Kao [6] consider a Master Planning problem from a different angle. Instead of planning by quantity, their MIP model plans demands by time. Their model plans the demands by time which is more suitable for a Factory Planning problem than for a Master Planning problem. This study, following the work of Chern and Hsieh [5], solves the Master Planning problem by including more than one objective and considering product commonality for multiple finished items.

2. Problem Description and LP Model Formulation

Before the construction of a LP model, the following information of products, facility capacities, and demand requirements are given first.

(1) It is assumed that the planning time horizon is cut into different intervals, called "time buckets." It is also assumed that the goods receiving is done at the beginning of the time bucket and the goods delivery is done at the end of the time bucket.

(2) It is assumed that more than one final product is made or transferred in the given supply chain network. The structure of these final products is given as graphs of product trees with nodes standing for items and links standing for parent-and-child relations. The number in the parentheses beside their name shows the quantity per child item required for making one parent item.

(3) A directed graph, $G(V, L)$, is used to represent a supply chain network. Nodes in $G(V, L)$ stand for facilities to produce, process, store, and sell products and links stand for logistical linkages between two nodes. Four types of information should be accompanied for each node: item name, unit production cost, capacity (in units), and unit holding cost per unit time bucket. Three types of information are given with each link: item name, transportation lead time (in time bucket), and unit transportation cost.

(4) Accompanied with each demand are quantities, due dates, and delay penalty per unit time bucket. It is assumed that the demands are dividable, which means that each demand can be

fulfilled by several different combinations of vendors, manufacturers, wholesalers, and retailers.

(5) No fixed setup cost is considered in this model. It is assumed that all participants of the given supply chain absorb their own setup time and costs.

The following indices and parameters are defined and followed by the construction of a LP model.

- **Indices:** *i*, *j*, and *g* are for nodes, *r* for demands with *R* equal to total number of demands, *t* for time buckets with *T* equal to total number of planning time buckets, *m* and *n* for parent items and components, *p* for final finished products with *P* equal to total number of final finished products, and *k* for production tree with *K* equal to total number of production trees.

- **Parameters:**
  - V: The set of nodes in a supply chain network. Specifically, *Start* node and *End* node are source (origin) and sink (destination) respectively.
  - V': The set of nodes in a supply chain network except *End* node.
  - V^M: The set of manufacturing nodes in a supply chain network.
  - V^P: The set of distributing nodes in a supply chain network.
  - L: The set of links in a supply chain network.
  - LT_{ijm}: Lead Time to transfer item *m* on link (i, j) and $(i, j) \in L$.
  - DUE_{rp}: Due date (in time bucket) for final product *p* of demand *r*.
  - DEM_{rp}: Demand quantity for final product *p* of demand *r*.
  - CTX_{imt}: The capacity limit of node *i* to produce item *m* in time bucket *t* where $i \in V^M$.
  - CTP_{imt}: The capacity limit of node *i* to produce item *m* in time bucket *t* where $i \in V^P$.
  - TC_{ijm}: The incremental unit shipping cost of item *m* from node *i* to node *j* where $(i, j) \in L$.
  - PC_{im}: The incremental unit processing cost of item *m* at node *i* where $i \in V^M$.
  - QC_{im}: The incremental unit processing cost of item *m* at node *i* where $i \in V^P$.
  - HC_{im}: The unit holding cost of item *p* at node *i* and $i \in V$.
  - DC_{rp}: The unit delay penalty per unit time bucket for item *p* of demand *r*.
  - Par_m: The set of parent items of component *m*.
  - Prd_i: The set of items produced on node *i* where $i \in V^M$.
  - Com_i: The set of components shipped to node *i* where $i \in V^M$.
  - Use_{mn}: Quantity of component *n* needed for per unit of item *m* where $m \in Par_n$.
  - RAT_{pm}: Quantity of component *m* needed for per unit of final finished item *p*.
  - NK_{ik}: = 1 if node *i* belongs to product tree *k*; = 0 otherwise.
  - IE_{ik}: = 1 if node *i* belonging to product tree *k* is an outsource facility; = 0 otherwise.

- **Decision Variables:**
  - X_{imprtk}: The production quantity of component *m* for final product *p* of demand *r* at node *i* of production tree *k* in time bucket *t*.
  - $S_{ijmprtk}$: The shipping quantity of component *m* for final product *p* of demand *r* from node *i* to node *j* of production tree *k* in time bucket *t*.
  - I_{imprtk}: The ending inventory quantity of component *m* for final product *p* of demand *r* at node *i* of production tree *k* in time bucket *t*.

- **The Constraints:**

(a) $\sum_{p=1}^{P}\sum_{r=1}^{R}\sum_{k=1}^{K} X_{imprtk} \le CTX_{imt} \quad \forall i \in V^M,\ m,\ t$

(b) $\sum_{p=1}^{P}\sum_{r=1}^{R}\sum_{g \in V'}\sum_{k=1}^{K} S_{gimpr(t-LT_{gim})k} \le CTP_{imt} \quad \forall i \in V^P, (g,i) \in L,\ m,\ t$

(c) $\sum_{i \in V'}\sum_{t=1}^{T}\sum_{k=1}^{K} S_{iEndppr(t-LT_{iEndp})k} = DEM_{rp} \quad \forall p, r$

(d) $\sum_{p=1}^{P}\left(I_{impr(t-1)k} + X_{imprtk} - \sum_{(g,i)\in L} S_{gimprtk}\right) = \sum_{p=1}^{P} I_{imprtk} \quad \forall i \in V^M, m = Prd_i, r,\ t,\ k$

(e) $\sum_{p=1}^{P}\left(I_{impr(t-1)k} + \sum_{(i,g)\in L} S_{igmpr(t-LT_{igm})k} - \sum_{p \in Par_m} X_{imprtk} \times Use_{pm}\right) = \sum_{p=1}^{P} I_{imprtk} \ \forall i \in V^M, m \in Com_i, r, t, k$

(f) $\sum_{p=1}^{P}\left(I_{impr(t-1)k} + \sum_{(i,g)\in L} S_{igmpr(t-LT_{igm})k} - \sum_{(g,j)\in L} S_{gjmprtk}\right) = \sum_{p=1}^{P} I_{gmprtk} \ \forall g \in V^P, m, r, t, k$

(g) $\sum_{t=1}^{T}\sum_{k=1}^{K} X_{imprtk} = DEM_{rp} \times RAT_{pm} \quad \forall i \in V^{M}, m, r, p$

(h) $\sum_{r=1}^{R}\sum_{p=1}^{P} X_{imprtk} \leq CTX_{imt} \times NK_{ik} \quad \forall i \in V^{M}, m, t, k$

(i) $\sum_{r=1}^{R}\sum_{p=1}^{P}\sum_{(g,i)\in L}\sum_{k=1}^{K} S_{gimpr(t-LT_{gim})k} \leq \sum_{k=1}^{K} CTP_{imt} \times NK_{ik} \quad \forall i \in V^{P}, (g,i)\in L, m, t$

(j) $I_{gmpr0k} = 0 \quad \forall g \in V, m, p, r, k$

(k) $I_{(End)mprtk} = 0 \quad \forall m, p, r, t < DUE_{rp}, k$

(l) $S_{ijmpr0k} = 0 \quad \forall (i,j) \in L, m, p, r, k$

(m) $X_{imprtk} \geq 0 \quad \forall i \in V^{M}, m, p, r, t, k$

(n) $I_{imprtk} \geq 0 \quad \forall i \in V, m, p, r, t, k$

(o) $S_{ijmprtk} \geq 0 \quad \forall (i,j) \in L, m, p, r, t, k$

Constraint (a) represents the production capacity limitations for all components on all manufacturing nodes in each time bucket. Constraint (b) shows the distribution capacity limitations for all components on all distribution nodes in each time bucket. Constraint (c) requires the total transportation quantity finished on End node after due date to be equal to the total demand quantity for each demand. Constraint (d) is the inventory balancing equation for each semi-finished or finished product on each manufacturing node in each time bucket. Constraint (e) is also the inventory balancing equation for each component on each manufacturing node in each time bucket. Constraint (f) represents the inventory balancing equation for each items on each distribution node in each time bucket. Constraint (g) sets the total production quantity of each component finished in all the time buckets to be equal to the total demand quantity for each demand. The total production quantity of each demand on each manufacturing node of each production tree can not exceed the capacity limitation, which is shown in constraint (h). The total transportation quantity of each demand on each distribution node of each production tree can not exceed the capacity limitation, shown in constraint (i). The initial inventory of each node at time bucket one is set to 0 in constraint (j). The inventory of End node for demand r is set to 0 after due time bucket in constraint (k). The initial transportation quantity of each node at time bucket one is set to 0 in constraint (l). The nonnegative requirement of each decision variable is shown in constraints (m), (n), and (o).

♦ **Multiple Objective Functions:**

To solve a multiple objective LP model, the goal programming technique is adopted. The multiple goals are sorted by their priorities and are achieved one by one while adding the previous goal as a constraint when solving the next one. In this study, the two goals are to be achieved: (1) Minimize order delay penalty; and (2) Minimize the sum of production cost, transportation cost, processing cost, and inventory holding cost. Three objective functions lead to three phases of LP optimization processes as follows:

- Phase I:

$$\text{Min} \quad Z_1 = \sum_{r=1}^{R}\sum_{p=1}^{P}\sum_{m=1}^{M}\sum_{t=DUE_{rp}}^{T} DC_{rp}\left(DEM_{rp} - \sum_{k=1}^{K} I_{(End)mprtk}\right)$$

- Phase II:

$$\begin{aligned}\text{Min} \quad Z_2 = &\sum_{r=1}^{R}\sum_{p=1}^{P}\sum_{m=1}^{M}\sum_{t=1}^{T}\sum_{k=1}^{K}\sum_{i\in V^{M}} PC_{im}X_{imprtk}\\ &+\sum_{r=1}^{R}\sum_{p=1}^{P}\sum_{m=1}^{M}\sum_{i\in V^{P}}\sum_{(g,i)\in L}\sum_{t=LT_{gim}}^{T}\sum_{k=1}^{K} QC_{im}S_{gimpr(t-LT_{gim})k}\\ &+\sum_{r=1}^{R}\sum_{p=1}^{P}\sum_{m=1}^{M}\sum_{(i,j)\in L}\sum_{t=1}^{T}\sum_{k=1}^{K} TC_{im}S_{ijmprtk}\\ &+\sum_{r=1}^{R}\sum_{p=1}^{P}\sum_{m=1}^{M}\sum_{i\in V^{\cdot}}\sum_{t=1}^{T}\sum_{k=1}^{K} HC_{im}I_{imprtk}\end{aligned}$$

Add constraints:

(r) $\sum_{r=1}^{R}\sum_{p=1}^{P}\sum_{m=1}^{M}\sum_{t=DUE_{rp}}^{T} DC_{rp}\left(DEM_{rp} - \sum_{k=1}^{K} I_{(End)mprtk}\right) \leq Z_1$

In the first phase, the total delay penalty is minimized together with all the constraints from (a) to (q). In the second phase, the sum of production, transportation, processing, and inventory holding cost is minimized together with constrains from (a) to (q) and one extra constraint from Phases I. Though the problem can be formulated as a LP model, difficulties arise when trying to solve this model. Therefore, this study proposed a heuristic algorithm to solve this MP problem.

3. Heuristic Commonality Master Planning Algorithm (HCMPA)

The heuristic MP algorithm constructed in this study is by nature a greedy algorithm. To simplify the explanation of the MP heuristic algorithm developed later, two important terms related to the production tree selected should be defined first.

- The minimum cost production tree, *S*: The tree selected from the given supply chain network has the minimum sum of the unit cost.
- Final Effective Available Capacity of *S* based on DUE_{rp}, $FEAC_S$: This available capacity is computed from the minimum available capacity on each node of the minimum cost production tree, *S*, based on the due time bucket of final product *p* for demand *r*.

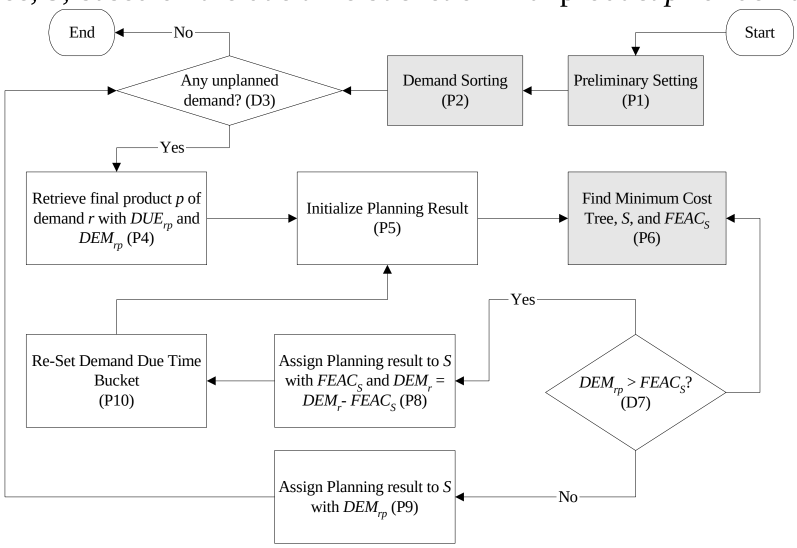

Figure 3.1: Main Steps of HCMPA

Throughout the proposed heuristic MP algorithm, *S* is identified and $FEAC_S$ is computed. The heuristic MP algorithm developed later in this section is called Heuristic Commonality Master Planning Algorithm (HCMPA). Ten steps as seen in Figure 3.1, included in HCMPA, are listed as follows:

(P1): Initialize cost information for all nodes of the given *G*(*V*, *L*). Convert the supply chain network to include outsourcing capacities.

(P2): Sort demands through the algorithm proposed by this study and explained later in this section.

(D3): If any unplanned demand exists, go to (P4). Otherwise, stop and output the final plan.

(P4): Retrieve the next demand with final product *p* and index *r* based on the sequences determined in (P2) with DUE_{rp} and DEM_{rp}.

(P5): Initialize planning results to be *0*.

(P6): Find the minimum cost production tree, *S*, and available capacity of *S*, $FEAC_S$.

(D7): If DEM_{rp}, $\geq FEAC_S$, then continue to (P8). Otherwise go to (P9).

(P8): Plan the demand by $FEAC_S$ found in (P6) and allocating $FEAC_S$ of all the nodes on *S* to this demand. $DEM_{rp} = DEM_{rp} - FEAC_S$ and go to (P10).

(P9): Allot DEM_{rp} of all nodes on tree *S* found in (P10) to this demand. Go to (D3).

(P10): Re-Set DUE_{rp} to the original due time bucket and go to (P5) to find the planning result for the residual DEM_{rp}.

HCMPA seems straightforward but difficulties remain in finding *S* based on the delay and inventory holding conditions together with other cost factors. The searching of the bottleneck nodes for computing $FEAC_S$ is also a great challenge. This study proposes three ways of allocating capacities to orders in step (P6) in the following subsection.

3.1 Greedy Capacity Allotment Algorithm

The idea of this algorithm is to treat the common parts as other parts, so when planning starts, the capacities of the common parts are used for orders that are placed in the front position. Once the capacities of the common parts are exhausted, the orders placed in the rear position are delayed. The planning algorithm of the greedy capacity allotment is similar with the algorithm given by

Hsieh [12]. The algorithm decides how many order quantity to fulfill by comparing the available capacity, $FEAC_S$, and demand quantity, DEM_{rp}. If $DEM_{rp} < FEAC_S$, it implies that the selected tree have enough capacity and quota to fulfill this order. Assign the order to this selected tree. The scheduler will then go to step (D3) to fetch another unplanned order. If the minimum is $FEAC_S$, the order can be fulfilled partially up to the level of $FEAC_S$. The remaining quantity of this order, DEM_{rp}, is updated as DEM_{rp} - $FEAC_S$. The scheduler goes to step (P10).

3.2 Greedy Capacity Allotment Algorithm

The idea of this algorithm is to treat the common parts as scarce resources. Thus, when allocating capacities of the nodes providing common parts, the available capacities are computed according to the average required quantity of the orders with the same order due time bucket. In this algorithm, the capacity using quota will be obtained from average number of several orders that will use the same common parts on the same day. Here we have to find out the time bucket that the orders use common parts. If the differences between the due date and the lead time of a common part are the same for two orders, the two orders will have a competition on their common part resources. For example, two orders use the same part from vendor V1. Because the two orders have the same due date, they have to compete for the same resource. We compute average demand quantity of these two orders as their capacity quota on V1. As a result, the quota will be (4000+2000)/2=3000.

The algorithm decides how many order quantity to fulfill by comparing the available capacity, $FEAC_S$, demand quantity, DEM_{rp}, and the scheduling quota, MIN_{irp}. If DEM_{rp} is the minimum, it means that we have enough capacity and quota to fulfill this order. Scheduling of the order will be completed. If minimum quantity of the three is MIN_{irp}, the order can only use capacity of the size MIN_{irp}. After the scheduling quota is used up, the order currently being scheduled will be switch out and the successive order will be switched in. The demand quantity DEM_{rp} of the switch-out order will be updated as DEM_{rp} - MIN_{irp} and its inventory information of every node will be stored in the database. The counter will tick to show whether this order is ever scheduled. Eventually, the program will go to step (D3) and continue scheduling. If the minimum is $FEAC_S$, the order can be fulfilled partially up to the level of $FEAC_S$. The remaining quantity of this order, DEM_{rp}, is updated as DEM_{rp} - $FEAC_S$. The scheduler goes to step (P10).

3.3 Proportional Capacity Allotment Algorithm

The idea of this algorithm is to allocate the capacities of the nodes providing the common parts to orders with the same due time bucket proportionally to their order quantities. In this algorithm, the capacity using quota proportion of an order will be obtained from its demand proportion to other orders that will use the same common parts on the same day. If the differences between the due date and the lead time of a common part are the same for two orders, the two orders will have a competition on their common part resources. For example, two orders use the same parts from vendor V1. Because the two orders have the same due date, they have to compete for the same resource at the same time bucket. We compute proportion number of the demand of these two orders as its capacity using quota proportion. As a result, the proportion of order 1 will be 4000/(4000+2000)=2/3, and the proportion of order 2 will be 2000/(4000+2000)=1/3. In this algorithm, the capacity using quota will be obtained from demand quantities of all orders that will use the same parts at the same time bucket. The quota proportion is computed before, the sum of all available capacity with consideration of inventory at the same time bucket is computed as the common part capacity. That is, we can use inventory before the specific time bucket when there isn't enough capacity. Of course, if some capacity has been used before planning time, it should be subtracted from sum of the available capacity. The proportion multiplying the sum of available capacity will be the order's capacity quota: POR_{irp}.

The algorithm decides how many order quantity to fulfill by comparing the available capacity, $FEAC_S$, demand quantity, DEM_{rp}, and the scheduling quota, POR_{irp}. If DEM_{rp} is the minimum, it means that we have enough capacity and quota to fulfill this order. Scheduling of the order will be completed. If minimum quantity of the three is POR_{irp}, the order can only use capacity of the size POR_{irp}. After the scheduling quota is used up, the order currently being scheduled will be switch out and the successive order will be switched in. The demand quantity

DEM_{rp} of the switch-out order will be updated as DEM_{rp} - POR_{irp} and its inventory information of every node will be stored in the database. The counter will tick to show whether this order is ever scheduled. Eventually, the program will go to step (D3) and continue scheduling. If the minimum is $FEAC_S$, the order can be fulfilled partially up to the level of $FEAC_S$. The remaining quantity of this order, DEM_{rp}, is updated as DEM_{rp} - $FEAC_S$. The scheduler goes to step (P10).

4. Computational Analysis

In this section, 12 scenarios are demonstrated to justify the effectiveness and efficiency of HCMPA in this research. A prototype of APS based on MOMPA has been constructed on a PC server with a Pentium IV 3.5 GB CPU and 1 GB RAM. The LP models are solved by ILOG 9.0 on Microsoft Window Server 2000 environment. The scenario categories are shown as Table 4.1. Three factors are taken into accounts: capacity status, quantities of orders, and degree of commonality by TCCI [21].Two kinds of demand quantities are considered: small scale and large scale. Small scale includes 2 orders and each with demand quantity of 400 units. Large scale includes 10 orders and each with demand quantity of 800 units. As to the capacity status, there are three kinds of situations: loose vacancy where each node of the network is 100% available; tight vacancy where each node of the network is about 40% available; insufficient vacancy where each node of the network can fulfill only half of all the demands on time. The BOM used in these scenarios are shown in Fig 4.1 for low commonality (TCCI = 11.1) and Fig 4.2 for high commonality (TCCI = 33.3).The supply chain network topology is shown in Fig 4.3 for high commonality and Fig 4.4 for low commonality.

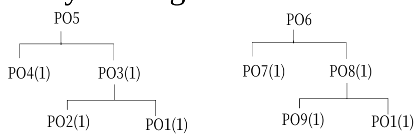

Figure 4.1: BOM for Low Commonality

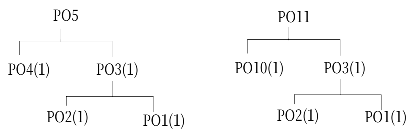

Figure 4.2: BOM for High Commonality

Table 4.1: Scenarios in Computational analysis

Factor	Level	Notes
Capacity	Loose	100 percent available
	Tight	40 percent available
	Insufficient	Only half of the needed capacity
Demand	Large	10 orders of 800 units
	Small	2 orders of 400 units
Commonality	High	All common parts except one
	Low	One common part component

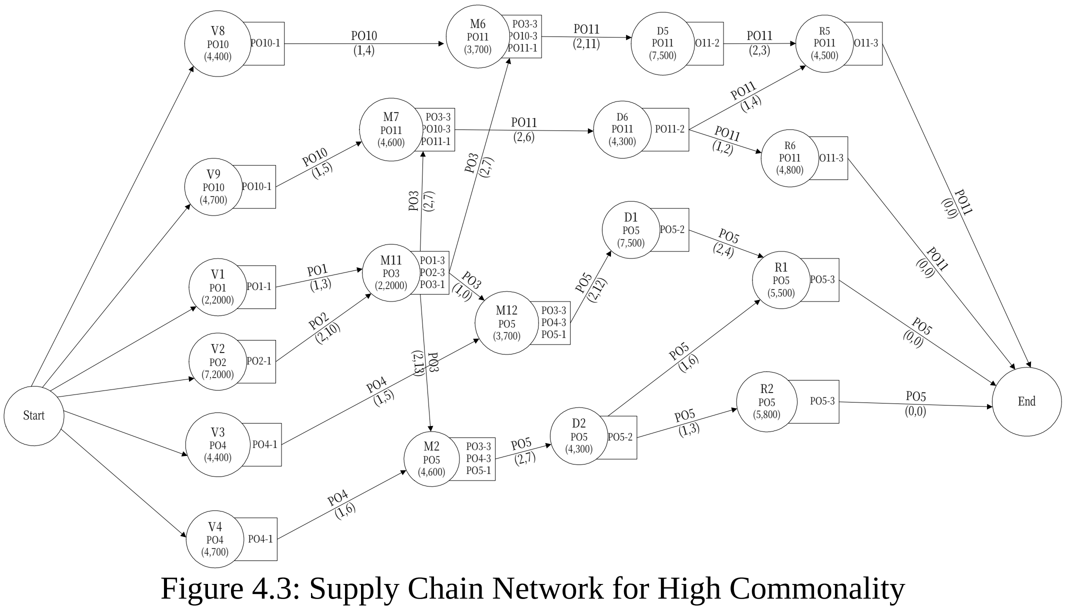

Figure 4.3: Supply Chain Network for High Commonality

In Table 4.2 we can find that the three heuristic algorithms: greedy, average capacity allotment, and proportional capacity allotment all result in solutions that are very close to result of linear programming. The result shows that the 3 heuristic algorithms can obtain near-optimal solution under these scenarios. However, the execution time of linear programming is much greater than other 3 heuristic algorithms.

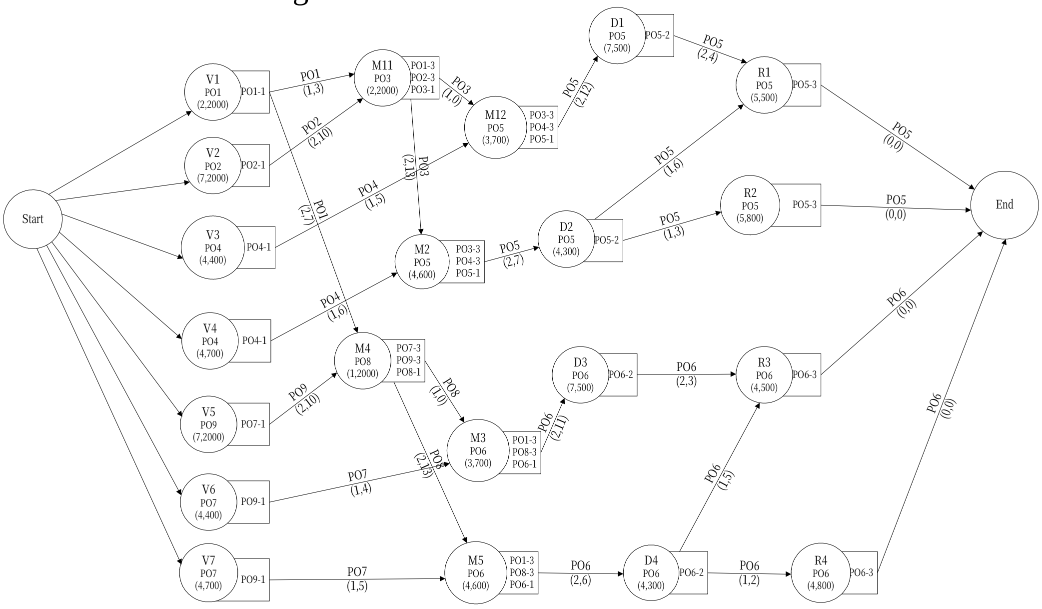

Figure 4.4: Supply Chain Network for Low Commonality

Table 4.2: Cost Comparison between HCMPA and LP for all Scenarios

Scenario	Greedy	Average	Proportion
Loose, Large-Demand, High-Commonality	3.12%	3.12%	3.12%
Loose, Large-Demand, Low-Commonality	3.10%	3.10%	3.10%
Loose, Small-Demand, High-Commonality	0.00%	0.00%	0.00%
Loose, Small-Demand, Low-Commonality	0.00%	0.00%	0.00%
Tight, Large-Demand, High-Commonality	3.35%	3.35%	3.35%
Tight, Large-Demand, Low-Commonality	3.22%	3.22%	3.22%
Tight, Small-Demand, High-Commonality	0.00%	0.00%	0.00%
Tight, Small-Demand, Low-Commonality	0.00%	0.00%	0.00%
Insufficient, Large-Demand, High-Commonality	10.58%	6.16%	5.06%
Insufficient, Large-Demand, Low-Commonality	11.46%	8.85%	8.83%
Insufficient, Small-Demand, High-Commonality	2.97%	1.87%	2.15%
Insufficient, Small-Demand, Low-Commonality	0.00%	0.00%	0.00%

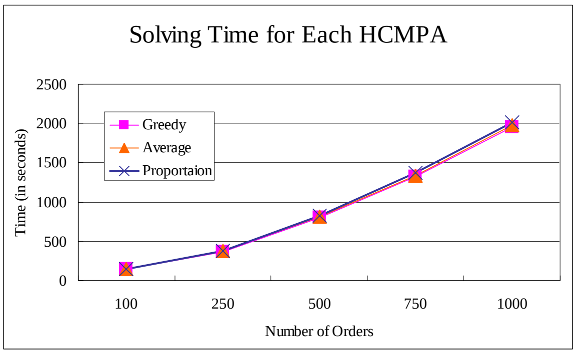

Figure 4.5: Time Performance of three HCMPAs

The impact of number of orders to HCMPA will be discussed here. The example repeats the same order pattern as the one in scenario “Loose, Large-Demand, High-Commonality”. The numbers of orders are 100, 250, 500, 750, and 1000, respectively. The time performance of HCMPA is shown in Fig 4.5, which indicates that the performance of these three algorithms are closed to each other and the line is almost linear. As to the LP model, it comes out with no feasible solution when number of orders is greater than 10. A large example is also tested with 2000 demands and a seven-layer supply chain network and product structure for three final products: A, T, X. The data comes from a large NB assembly plant based in Taiwan. The inventory cost is assumed 1% of the unit processing cost per period and the delay penalty is 18% of the retail price or NT$6000 per period. The greedy HSMPA takes 47 minutes and the proportion HCMPA take 63 minutes to plan all the 2000 demands while the average HCMPA takes about 2 hours. The gap between the lower bound and the result from HCMPA is only 0.13%. In the real-world business situation, the number of demands is often more than 1000 per month and the planning time horizon is more than 6 months with lead time equal to one day to several months. The scale of a MP problem is often gigantic and impossible to be solved through a LP model but it is not a problem to HCMPA as demonstrated in the section.

Reference

[1] Awadh, B., N. Sepehri, and O. Hwaaleshka, "A Computer-Aided Process Planning Model Based on Genetic Algorithms," *Computers & Operations Research*, Vol. 22, No. 8, pp. 841—856, 1995.

[2] Baker, K.R., M.J. Magazine, and H.L.W. Nuttle, "The Effect of Commonality on Safety Stocks in a Simple Inventory Model," *Management Science*, Vol. 32, pp.982—988, 1986.

[3] Brailsford, S. C., C. N. Potts, and B. M. Smith, "Constraint Satisfaction Problems: Algorithms and Applications," *European Journal of Operational Research*, Vol.119, pp. 557—581, 1999.

[4] Chen, S. Y. and C. C. Chern, "Shortest Path for a Supply Chain Network," *Proceedings of the 4th Asian Pacific Decision Sciences Institute Conference*, Shanghai, China, 1999, pp. 579—582, 1999.

[5] Chern, C. C. and C. H. Hsieh, "A Heuristic Master Planning Algorithm to Schedule Orders Under Due Date Requirements and Capacity Limits," *Proceedings of the 33rd Annual Meeting of Decision Science Institute*, San Diego, California, USA, pp. 1842—1847, 2002.

[6] Chern, C. C. and Y. K. Kao, "Heuristic Master Planning Algorithm (HMPA) by Allowing Order Delay," *Proceedings of the 35th Annual Meeting of Decision Science Institute*, Boston, USA, pp. 3841—3846, 2004.

[7] Chopra, S. and P. Meindl, "Supply Chain Management: Strategy, Planning, and Operation," USA, Pearson Prentic Hall, 2004.

[8] Collier, D.A., "The Measurement and Operating Benefits of Component Part Commonality," *Decision Science*, Vol. 16, pp.85—96, 1981.

[9] Collier, D.A., "Aggregate Safety Stock Levels and Component Part Commonality," *Management Science*, Vol. 28, pp.1296—1303, 1982.

[10] Dogramaci, A., "Design of Common Components Considering Implications of Inventory Costs and Forecasting," *AIIE Transactions*, Vol. 11, pp.129—135, 1919.

[11] Enyan, A. and M. Rosenblatt, "Component Commonality Effects on Inventory Costs," *IIE Transactions*, Vol. 28, pp.93—104, 1996.

[12] Eppen, G.D., "Effects of Centralization on Expected Costs in a Multi-Location Newsboy Problem," *Management Science*, Vol. 25, pp.498—501, 1978.

[13] Gerchak, Y., M.J. Magazine, and A.B. Gamble, "Component Commonality with Service Level Requirements," *Management Science*, Vol. 34, pp.753—760, 1988.

[14] Jayaraman, V. and H. Pirkul, "Planning and Coordination of Production and Distribution Facilities for Multiple Commodities," *European Journal of Operational Research*, Vol. 133, pp. 394—408, 2001.

[15] Kim, Y. K., K. Park, and J. Ko, "A Symboitic Evolutionary Algorithm for the Integration of Process Planning and Job Shop Scheduling," *Computers & Operations Research*, Vol. 30, pp. 1151—1171, 2003.

[16] Kreipl, S. and M. Pinedo, "Planning and Scheduling in Supply Chains: An Overview of Issues in Practice," *Production and Operation Management*, Vol. 13, No.1, pp. 77—92, 2004

[17] Lee, Y. H., C. S. Jeong, and C. Moon, "Advanced Planning and Scheduling with Outsourcing in Manufacturing Supply Chain," *Computer & Industrial Engineering*, Vol. 43, pp. 351—374, 2002.

[18] Lee, Y. H. and S. H. Kim, "Production—Distribution Planning in Supply Chain Considering Capacity Constraints," *Computers & Industrial Engineering*, Vol. 43, pp. 169—190, 2002.

[19] Liu, X. R., "Optimization-Based Available To Promise Allocation Model with Common Components," *Unpublished Master Thesis, National Taiwan University*, 2003.

[20] Stadtler, H. and C. Kilger, "Supply Chain Management and Advanced Planning: Concepts, Models, Software and Case Studies," Springer-Verlag Berlin Heidelberg, 2000.

[21] Wacker, J. G., and M. Treleven, "Component Part Standardization: Analysis of Commonality Sources and Indices," *Journal of Operations Management*, Vol. 6, pp.347—368.

[22] Zhang, S. H., "Optimal Material Stocking Policy Model in Assemble-To-Order Systems by Considering Material Commonality," *Unpublished Master Thesis, National Taiwan University, Taipei, Taiwan*, 2003.