RESEARCH SUPPORT
UNIVERSITY OF MICHIGAN BUSINESS SCHOOL

JUNE 1997

PRODUCTION PLANNING FOR FLEXIBLE FLOW SYSTEMS

WORKING PAPER #9709-17
BY

HEUGNSOONN FELIX LEE
SOUTHERN ILLINOIS UNIVERSITY

KATHRYN E. STECKE
UNIVERSITY OF MICHIGANBUSINESS SCHOOL

REVISION OF 9409-45

Production Planning for Flexible Flow Systems with Limited Machine Flexibility[*]

Heungsoon Felix Lee

Department of Mechanical and Industrial Engineering
Southern Illinois University
Edwardsville, IL 62026-1805
Phone: (618) 692-2805
Fax: (618) 692-2555
E-mail: hflee@siue.edu

and

Kathryn E. Stecke

School of Business Administration
The University of Michigan
Ann Arbor, MI 48109-1234
Phone: (313) 763-0485
Fax: (313) 763-5688
E-mail: kathryn_stecke@umich.edu

July 1996
(revised in June 1997)

[*] This research is supported in part by a grant from National Science Foundation (Grant No. DDM-9201954) and research grants from Southern Illinois University at Edwardsville and from the Business School of The University of Michigan.

Production Planning for Flexible Flow Systems with Limited Machine Flexibility

Abstract

A flexible manufacturing system (FMS) is highly capital-intensive and FMS users are concerned with achieving high system utilization. The production planning function for setting up an FMS prior to production should be developed in order to make the most of the potential benefits of FMSs. We consider two production planning problems of grouping and loading a flexible flow system, which is an important subset of FMSs where the routing of parts is unidirectional. We show that considering this routing restriction as well as limited machine flexibility strongly affects both the solution techniques and the quality of the solutions. Because of the complexity of the problem, we present a heuristic approach that decomposes the original problem into three interrelated subproblems. We show that the proposed approach usually finds a near-optimum solution and is superior to an approach that exists in the literature of FMS production planning. We also introduce effective heuristic methods for two new subproblems that arise because of the unidirectional flow precedence and flexibility constraints. Computational results are reported and future research issues are discussed.

Key Words: Flexible Manufacturing Systems, Flexible Flow Systems, Production Planning, FMS Grouping Problem, FMS Loading Problem, Machine Flexibility, Closed Queueing Networks.

1. Introduction

A flexible manufacturing system (FMS) consists of computer numerically controlled machines that are capable of performing many different operations and linked by a material handling system (MHS). All operations and material movements are monitored and controlled by a computer system. An FMS combines automation suitable for mass production with flexibility suitable for job shop production. The type of FMSs studied in this paper are flexible flow systems (FFSs), where the routing of parts is unidirectional. An FFS is very common for both assembly and machining systems due to easy production control and the efficiency of a flow system. Such an FFS includes most flexible assembly systems (Kamath et al. 1988), and flexible machining systems with U-layouts (Harmon and Peterson 1990), loop layouts (Afentakis 1989), and some group-technology-based layouts.

Making the most of potential benefits of such expensive FMSs requires well-thought out production planning before it begins production for each upcoming time period. Stecke (1983) provides five categories of production planning problems for FMSs, which are part type selection, machine grouping, loading, resource allocation, and production ratio determination. The focus of this paper is on two of them, namely, grouping and loading problems. The machine grouping problem is to partition the machines of the same type into identically tooled machine groups. Each machine in a particular group is able to perform the same operations. The loading problem is to assign operations and their tools to machine groups subject to some technological constraints, such as precedence and which machine tools are capable of performing which operations.

Several researchers have studied grouping and loading problems for FMSs, using different techniques such as mathematical programming, queueing networks, and simulation. Stecke and Solberg (1985) and Dallery and Stecke (1990) address the grouping and loading problems, using closed queueing network models for FMSs. They provide useful guidelines on maximizing system throughput. They found that (1) fewer groups are better, and (2) unbalanced configurations of assigned machines are superior to balanced ones, and (3) unbalanced workloads are better than balanced ones. They report that there can be significant differences in the throughput from balanced versus unbalanced configurations/workloads.

Stecke (1983) provides a nonlinear mixed integer formulation for various realistic loading problems and gives a linearization solution method. Berrada and Stecke (1986) develop a branch-and-bound algorithm to solve a similar formulation in a more user-friendly manner with the workload balancing objective. Kim and Yano (1993) view the loading problem as a multi-dimensional bin-packing problem and presented a heuristic approach using multi-pass algorithms. Some researchers study loading problems with two objectives. Shanker and Tzen (1985) try to balance workloads, while minimizing the number of late parts. Ammons et al. (1985) have an objective of minimizing workload imbalance and material movements between machines. Lashkari et al. (1987) give a nonlinear mixed integer formulation for loading with the two objectives of minimizing transport load and minimizing refixturing activities. There are also studies (Stecke and Solberg 1981 and Greene and Sadowski 1986) which address the loading problem in conjunction with FMS scheduling problems.

Some researchers address both grouping and loading problems for FMSs. Stecke (1986a) presents a hierarchical framework in which the grouping problem is solved and then the loading problem is solved using the input from the grouping problem. Several loading objectives are discussed within the framework. Some iteration is recommended until a satisfactory solution is obtained. Kim and Yano (1992) present an iterative and hierarchical approach to address these two problems with part type selection also. They simplify the part type selection problem by using a prioritized list of part type orders. For selected part types, the iterative approach resorts to an exhaustive search method to solve first the grouping problem and then the loading problem. These approaches model FMSs using closed queueing networks. See Templemeier and Kuhn (1993) for a comprehensive survey of FMS planning papers.

In this paper, the grouping and loading problem for FFSs are studied for the first time, to our knowledge. What is unique in this problem is the consideration of both the unidirectional part flow in FFSs in conjunction with limited machine flexibility. This part flow restriction imposes a new type of constraint on the loading problem since there are precedence relations among operations and parts do not revisit a machine group in FFSs. We show that this flow restriction also affects the choice of machine groups. The previous approaches do not consider material flow and handling aspects of FMSs because there are no fixed precedence relationships for all part types in FMSs and so it was not important to consider part transfer times. This new constraint clearly makes the problem more difficult and greatly affects the solution techniques to be employed. We present a heuristic method that elegantly decomposes the original problem into three subproblems each of which is manageable. We show through numerous test problems that the proposed method usually finds a near-optimum solution and improves on the approach proposed by Kim and Yano (1992) in both effectiveness and efficiency. This approach, however, does not consider precedence requirements and some modification is necessary for the comparative study.

This problem is similar to the line balancing problem (Baybars 1986) in that both involve the assignment of operations among machine groups in a flow line and precedence relations among operations place an important restriction on the operation assignment. However, the line balancing problem deals with only a balanced configuration, typically one machine per machine group, and assumes no limitation in machine flexibility.

The remainder of the paper is organized as follows. In Section 2, we discuss the model for FFSs and state the mathematical formulation for the problems. In Section 3, we present the decomposition-based heuristic approach and the solution methods for the three subproblems. We present our numerical results in Section 4. Finally, in Section 5 we summarize our findings and discuss some future research directions.

2. Problem Formulation

Given available resources such as machines and material handling devices, part types to be produced simultaneously for an upcoming period, and their production requirements, the problem of grouping and loading FFSs is to simultaneously find machine groups and assign operations among the machine groups. The objective here is to maximize the system throughput or system utilization. An FMS is highly capital-intensive and FMS users are concerned with achieving high system utilization (Stecke and Kim 1991). A key objective of planning FMSs that produce part types having independent demands is the maximization of system utilization (Smith and Stecke 1996). By maximizing throughput over the short term, we can also accomplish other goals, such as meeting due dates or reducing operating costs (Kim and Yano 1992). The system or equipment, however, cannot reach 100% utilization since there are limits on the WIP allowed into the system (i.e., because of a limited number of pallets) and there is randomness in the system.

This objective is considered here under the two constraints of the unidirectional flow and limited flexibility capacity. In order to meet this flow constraint, the loading problem needs to explicitly address precedence relations among operations. The flexibility capacity constraint limits the maximum number of operations that can be assigned to each machine group. Each machine type in FFSs has a flexibility capacity, which is measured in terms of the number of operations that this machine type can perform one after another with negligible setup times between operation changes (Sethi and Sethi 1990). For example, in flexible machining systems, CNCs performing various metal-cutting operations are equipped with tool magazines which can hold a certain number of cutting tools, and among these tool changes there are negligible changeover times. The tool magazine capacity is typically 30, 60, or 120 slots in commercial flexible machines (Singh 1996). In flexible assembly systems, automatic insertion machines or assembly robots perform a limited number of assembly tasks because they have a finite work space due to their physical configurations and component feeding mechanisms associated with each assembly task use some of the finite work space (Ammons et al. 1985, Lee and Johnson 1991).

We use a product-form closed queueing network (CQN) model to represent the FFSs. CQN models have been widely-used to represent FMSs (Suri and Hildebrant 1984, Stecke and Solberg 1985, Kamath et al. 1988, Kouvelis and Lee 1995) since Solberg (1977) first suggested its use for FMSs. This is because these models can capture the aspects of material handling systems and product flows, of resource contention, and of random events occurring in FMSs in a reasonably adequate and robust manner. They take into account the interactions and congestion of parts competing for the same machines and represent in an aggregate manner the stochastic behavior of work flows due to the uncertainty and dynamics in FMSs. Our intent here is not to validate CQN models but to use a CQN model to solve production planning problems for FFSs.

For the FFSs being considered, the throughput per period, TH, is computed from the CQN model as a function of the following eight parameters: (a) the number of machine groups M, (b) the number of pallets circulating in the system N, (c) a machine vector $\overline{S}=(S_1,S_2,...,S_M)$, where S_i is the number of identically-tooled machines at machine group i, (d) workload vector $\overline{W}=(W_1,W_2,...,W_M)$, where W_i is the sum of the weighted average operation times assigned to group i (i.e., the average processing time required to process one of the aggregate parts), (e) processing capacity vector $\overline{P}=(P_1,P_2,...,P_M)$, where P_i is the total available processing time of a machine at group i per period, (f) average material handling time required to produce one part W_0, (g) the number of automated guided vehicles (AGVs), S_0, if used, and (h) the total available material handling time (by an AGV or conveyor) per period P_0. See Reiser and Lavenberg (1980) for the CQN TH formula.

In the CQN, one aggregate part type, an average part, collectively represents the individual part types. Precedence diagrams of all part types are merged and represented as one super precedence diagram for the aggregate part. This precedence representation is common in mixed-model production, where different part types are simultaneously produced. (See Graves and Redfield 1988, Liu and Sanders 1988, and Lee and Johnson 1991). Demand and operation times for the aggregate part are specified as the sum of demands and the weighted average operation times among the individual part types, respectively. Processing times lost from small but regular disturbances such as tool jams or tool replacements are also added as part of the average operation time. See Lee and Johnson (1991) for an example of an aggregate part.

The problem of simultaneously grouping and loading FFSs can be mathematically stated below. Table 1 provides the notation used throughout this paper.

<Insert Table 1>

P0: Maximize $TH(N, \bar{S}, \bar{W})$

$$\text{subject to: } \sum_{i \in A_c} S_i = K_c, \qquad c=1,\ldots,C$$

$$\sum_{j=1}^{n} t_j X_{ij} = W_i, \qquad i=1,\ldots,M \qquad (1)$$

$$\sum_{j=1}^{n} X_{ij} \leq R_c, \qquad i=1,\ldots,M \text{ and } i \in A_c \qquad (2)$$

$$X_{ij} = 0 \text{ or } 1, \qquad i=1,\ldots,M, \quad j=1,\ldots,n \qquad (3)$$

$$\sum_{i=1}^{M} X_{ij} = 1, \qquad j=1,\ldots,n \qquad (4)$$

$$\sum_{i=1}^{M} i X_{ij} \leq \sum_{i=1}^{M} i X_{ik}, \quad \text{when operation j must precede operation k.} \qquad (5)$$

Equation (1) defines W_i, the workload at machine group i, as the sum of the operation times assigned to group i. Constraint (2) is the flexibility capacity constraint which limits the number of operations assigned to each group. Constraints (3) and (4) are assignment constraints which force each operation to be assigned to exactly one group. Constraint (5) models the precedence relations among operations and ensures that a part does not revisit any group in the flow system. Decision variables are M, $\bar{S}$, and (X_{ij}) and Problem P0 is a nonlinear integer programming problem with a complex objective function, which is clearly hard to solve optimally.

3. Solution Approach

We propose a heuristic method which decomposes the decision variables into two groups: the assignment of operations to groups and the rest of the decisions. The assignment of operations to groups represents the majority of the decision variables, and the two sets of decisions are nicely related through the workloads, $\bar{W}$. Thus, instead of solving the original problem, the methodology solves a relaxed problem where the decision of assigning operations to groups is replaced by the decision of continuously allocating the total workload to machine groups. This replacement allows a reduction in the number of decision variables from nM to M.

For a given number of machine groups, the proposed method solves three subproblems in order. The first subproblem is the workload bound problem (WBP), which is to find workload upper and lower bounds at each group, such that workloads outside the bounds cannot be achieved by any operation assignment. The second subproblem is the workload allocation and grouping problem (WAGP), which is to solve the relaxed problem with the workload bound constraint from the WBP. Since the WAGP deals with the relaxed problem instead of the original problem P0, the maximum throughput obtained here serves an upper bound on the maximum throughput of P0. The WAGP also provides target workloads as part of the solution, which are input to the third subproblem, the operation loading problem (OLP). The OLP concerns only the assignment of operations among groups so that the resulting workloads are as close as possible to the target workloads. The rationale behind this decomposition is that the CQN throughput function, TH, is unimodal and well-behaved with respect to the workloads, $\bar{W}$ (Stecke 1986b, Lee et al. 1991a). The relationship, input, and output among the three subproblems are shown in Figure 2.

<Insert Figure 2>

The proposed method can iterate over the different number of groups, starting with the smallest number and incrementing by one until it equals the total number of machines. The general guideline, however, favors the smallest number (Stecke and Solberg 1985, Dallery and Stecke 1990). The smaller number of machine groups allows larger resource pooling, which results in not only larger throughput but also larger routing flexibility and reliability. In the remainder of this section, we discuss the mathematical formulation and solution method for each subproblem. For simplicity of presentation, we focus on P0 with a single type of flexible machine (C=1), which is capable of performing all operations.

A similar decomposition approach was taken by Lee et al. (1990) to solve a complex design problem for flexible assembly systems, where the decision variables include not only operation assignment but also capacities of machines and material handling devices with the objective of minimizing total design cost. They present a methodology which decomposes the design problem into six subproblems. The focus of their work is a framework of the methodology and relationships among the six subproblems. No details of solution methods for those subproblems are addressed. In this paper, we show that some of the subproblems can be used to solve a production planning problem in FFSs. Solution methods for the WBP and the OLP are presented for the first time with experimental results. We show that the proposed method performs better than an existing method in the literature.

3.1 The Workload Bound Problem (WBP)

The first subproblem, WBP, identifies the feasible region of workload allocation. The WBP finds tight workload upper and lower bounds at each machine group subject to the machine flexibility and unidirectional flow constraints. Thus, the resulting target workloads from the WAGP become more achievable, and consequently, easier to fit in the OLP than those from the WAGP without the workload bounds. The visit sequence is the sequence in which a part visits machine groups and is 1,...,M. The following example clarifies the WBP.

Example 1. We use an aggregate part of Figure 1. Let the machine flexibility R = 5, and M = 2. Denote U_i and L_i as workload upper and lower bounds at group i, respectively. Then, U_1 is the sum of the five largest operation times that can be assigned to group 1, that is, $U_1 = t_1+t_2+t_3+t_4+t_6 = 19$. The flexibility limit of R=5 prevents group 1 from having more than five operations assigned. Although $t_5 > t_3$, operation 5 cannot replace operation 3 because assigning operation 5 to group 1 makes a part visit group 1 twice. At the first visit, operations 1 and 2 can be processed but operations 4, 5, and 6 cannot due to precedence requirements. On the other hand, L_1 is the sum of the two smallest operations that can be assigned to group 1, that is, $L_1 = t_1+t_2 = 9$. Less than two operations cannot be assigned to group 1 since group 2 can have at most 5 operations assigned due to the limited flexibility. Replacing either operation by any other operation would require a part to revisit group 1. Similarly, $U_2 = t_3+t_4+t_5+t_6+t_7 = 20$ and $L_2 = t_5+t_7 = 10$.

Before we present the solution method to the WBP, we give three lemmas which state how many operations should be assigned to each group at the workload bounds. Denote M_0 as the smallest number of groups, i.e., $M_0 = \lceil n / R \rceil$, where $\lceil x \rceil$ is the smallest integer greater than or equal to x. Also let $r = n - R\cdot(M_0 - 1)$.

Lemma 1. When $M_0 \leq M < n+2-R$, U_i is the sum of R operation times for every i.

Lemma 2. When $M = M_0$, L_i is the sum of r operation times for every i.

Lemma 3. When $M > M_0$, L_i is only one operation time for every i.

The lemmas can be easily proved by using the pairwise interchange argument (Baker 1974). For example, in Lemma 1, suppose R-1 operations are assigned to group i and n-(R-1) operations assigned to the other M-1 groups, without violating precedence requirements for a flow system. Then, it is always possible to move an operation to group i through pairwise interchanging between two adjacent groups without violating precedence requirements. This movement increases total workload assigned to group i. We now present the solution procedure to the WBP.

Procedure 1. A solution procedure to the WBP

Step 1. For each j, find the first and last groups, f_j and l_j, in the visit sequence to which operation j can be assigned. Find the sets of operations, V_i, for i = 1 to M, which can be assigned to group i as follows: $V_i = \{j \mid f_j \leq i \leq l_j\}$, where f_j and l_j are obtained from the following equations (Talbot and Patterson 1984):

$f_j = \lceil (1 + \text{the number of operations preceding operation } j) / R \rceil$ and
$l_j = M + 1 - \lceil (1 + \text{the number of operations following operation } j) / R \rceil$.

Step 2. Find U_i for $i = 1$ to M, where $M_0 \leq M < n+2-R$. From Lemma 1, this involves finding a set of R operations in V_i which maximizes the sum of their operation times and allows no revisit by a part to any group.

Step 3. Find L_i for $i=1$ to M. When $M = M_0$, from Lemma 2, this involves finding a set of r operations in V_i which minimizes the sum of their operation times and allows no revisit by a part to any group. When $M > M_0$, L_i is simply the smallest operation time in V_i according to Lemma 3.

We use f_j and l_j in Step 1 in order to find tighter workload bounds in Steps 2 and 3. This is done by exploiting precedence relations among operations and excluding operations which cannot be assigned to a particular group.

Step 2 is difficult to formalize and solve optimally. We transform Step 2 into a variant of a graph problem that is easier to solve but may provide looser workload bounds. Consider a directed graph G_d that is induced by V_i and the precedences among operations in V_i. We use notation $j < k$ for j and k in V_i when there is a directed path from operation j to operation k, and $j \nless k$ when there is no directed path from j to k.

Procedure 2. Transformation into the Maximum-weight Connected Graph Problem (MCGP)

Step 1. Create an undirected graph G_i from G_d by replacing any precedence arc in G_d with an edge and by adding an edge between j and k when $j \nless k$ and $k \nless j$.

Step 2. Find a connected subgraph of G_i with R nodes (i.e., operations) that maximizes the sum of their operation times. We call this problem the maximum-weight connected graph problem (MCGP), where t_j is viewed as the weight of node j.

The motivation behind this transformation is that it is not easy to check for a possible revisit in Step 2 but there is a very efficient algorithm to check connectivity of a graph (Aho et al. 1974). Note, by definition of f_j and l_j, that the optimal solution must include those operations which have both f_j and l_j equal to i. A similar technique can be applied to find L_i in Step 3 for the use of minimization instead of maximization. The following example clarifies the transformation and the MCGP to find workload bounds.

Example 2. Consider the aggregate part of Figure 1. First find (f_j, l_j) for j=1 to 7. These pairs are (1,1), (1,1), (1,2), (1,2), (1,2), (1,2), and (2,2). $V_1 = \{1, 2, 3, 4, 5, 6\}$

and $V_2 = \{3, 4, 5, 6, 7\}$. Note that operations 1 and 2 must be assigned to machine group 1 since their f_j and l_j's are equal to 1. Similarly, operation 7 must be assigned to group 2. Figure 3 shows the transformed undirected graphs. Two edges are newly added in both G_1 and G_2. In G_1, one edge is added between operations 1 and 2 and the other between 5 and 6, since there are no directed paths between them. Since R=5, n=7, $M=M_0=2$, we have r = 3. To find U_1, the MCGP finds five operations that form a connected subgraph of G_1 and maximizes the sum of their operation times. Since the five operations must include operations 1 and 2, $U_1 = t_1+t_2+t_3+t_4+t_6 = 19$. U_2 and L_1 are directly obtained as $U_2 = t_3+t_4+t_5+t_6+t_7 = 20$ and $L_1 = t_1+t_2 = 9$. To find L_2, the MCGP finds two operations that form a connected subgraph of G_2 and minimizes the sum of their operation times. Since the two operations must include operation 7, $L_2 = t_5+t_7 = 10$. Note that all four bounds obtained here happen to be the same as the "true" workload bounds obtained in Example 1.

<Insert Figure 3>

Lee and Dooly (1996) present solution methods for the MCGP, using a variant of the Balas additive method with an imbedded connectivity test and other fathoming methods. Although the workload bounds of the MCGP are theoretically looser than those of the WBP, we show, in the experimental results of Section 4.2, that they are quite effective in finding a near-optimal solution to P0.

3.2 The Workload Allocation and Grouping Problem (WAGP)

Given the numbers of available machines, AGVs, and pallets, (K, S_0, and N), the WAGP is to determine the allocation of K machines among M groups and the continuous allocation of total workload, $TW=\sum_{j=1}^{n} t_j$, among the groups. The objective is to maximize throughput, TH, subject to a constraint that the allocated workloads must lie between the lower and upper bounds obtained from the WBP. Since a continuous allocation of workloads is permitted, the resulting throughput serves as an upper bound to the optimal throughput of the original problem P0. A mathematical formulation of the WAGP is

$$\textbf{WAGP:} \quad \text{Maximize} \quad TH(\bar{S},\bar{W})$$

$$\text{subject to:} \quad \sum_{i=1}^{M} S_i = K$$

$$\sum_{i=1}^{M} W_i = TW$$

$$L_i \le W_i \le U_i \quad \text{for } i=1,\ldots,M,$$

where the decision variables are $\bar{S}$ and $\bar{W}$, which will be referred to as a configuration hereafter. WAGP is a nonlinear mixed integer programming problem and the details of the solution method appear in Lee et al. (1991b). In practice, we find it useful to find all configurations that meet the aggregate demand rather than to find only one configuration maximizing throughput. This is because some configurations may be not possible to implement because of other technical and operating issues that cannot be captured in a mathematical formulation. The solution method to WAGP is modified for this purpose.

3.3 The Operation Loading Problem (OLP)

The OLP involves assigning operations to groups such that the actual workloads are as close as possible to the given target workloads, subject to the flexibility capacity and unidirectional flow constraints. A formulation of the OLP can be stated as:

OLP: Minimize $D(\bar{W},\bar{W}^*)$

subject to: (1), (2), (3), (4), and (5),

where $D(\bar{W},\bar{W}^*)$ is a function which measures the closeness of the actual workload vector $\bar{W}$ to the target workload vector $\bar{W}^*$. Kim and Yano (1993) tested several functions for $D(\bar{W},\bar{W}^*)$ for a different FMS loading problem, and suggested $D(\bar{W},\bar{W}^*) = \max_i (W_i-W_i^*)/W_i^*$. With this substitution, OLP is rewritten as :

OLP': Minimize δ

subject to: constraints (2) through (5) and

$$\sum_{j=1}^{n} q_{ij}\ X_{ij} \ \leq \ \delta \qquad i = 1,\cdots,M, \tag{6}$$

where q_{ij} is equal to t_j / W_i^*. We develop an ε-optimal solution procedure for OLP' by generalizing an optimal algorithm for the assembly line balancing problem (ALBP). OLP' without the flexibility capacity constraint (2) is a variant of the traditional ALBP. In fact, OLP' with $R=\infty$ and balanced target workloads (i.e., W_i^* equals W for all i) is exactly the type II assembly line balancing problem as defined by Baybars (1986), which is to find the minimum δ given the number of groups M. Thus, OLP' can be solved by applying a bisection search method to specify trial values of δ and solving the generalized ALBP for each trial value of δ. The solution procedure for OLP' is given as follows.

Procedure 3. A solution procedure for OLP'

Step 1. Find lower and upper bounds, δ_L and δ_U, on δ. Set the iteration number to 1.

Step 2. If $\delta_U - \delta_L < \varepsilon$, a small value used as a termination tolerance, then terminate with an ε-optimal solution. Otherwise, set δ to $(\delta_L + \delta_U) / 2$ and go to Step 3.

Step 3. Apply an algorithm for the generalized ALBP to determine whether a feasible solution exists for OLP' for the given δ. Increase the iteration number by one.

Step 4. If a feasible solution exists, then update the incumbent solution (operation assignment) and set δ_U to $\max_i (\sum_{j=1}^{n} q_{ij} X_{ij})$; otherwise, increase δ_L to δ. Go to Step 2.

Constraint (6) is violated for any $\delta < 1$. When $\delta = 1$, we have a perfect fit, i.e., $W_i^* = W_i$ for all i. Thus, the initial δ_U and δ_L are specified as follows. For the initial δ_U, an arbitrarily large value can be specified to ensure a feasible solution, but this requires a large number of iterations before termination. Instead, an initial δ_U is set to 2 and increased by one until the first feasible solution is found. The initial δ_L is set to δ_U -1 and the resulting $\delta_L \geq 1$. We found that δ=2 is usually large enough to ensure a feasible solution since with δ=2, the OLP tries to fit operations to a bin with its capacity two times larger than the ideal bin size, W_i^*. In order to solve Step 3, we generalize Johnson's (1988) branch-and-bound algorithm for the traditional ALBP, and the details appear in Lee (1989).

The WAGP usually finds several configurations which have *just different permutations* of a machine vector $\bar{S}$. Solving Procedure 3 for each configuration is time-consuming for even small M since Procedure 3 requires a solution of an integer program several times.

Using the fact that the product-form CQN throughput is permutation-invariant, i.e., $TH(\bar{S},\bar{W}) = TH(\pi(\bar{S}),\pi(\bar{W}))$ for any permutation π, we now present a procedure that reduces computation by considering at most two such permutations. When the WAGP finds multiple configurations, we permute the machine vector for each configuration in lexicographic order such that $S_1 \leq \ldots \leq S_M$ and eliminate any duplicates. We denote this ordered machine vector as $\bar{S}_u$. For each $\bar{S}_u$, we solve the workload allocation problem without the workload bound constraints and find the unconstrained workload vector, $\bar{W}_u$, that maximizes TH.

We derive two configurations by permuting $(\bar{S}_u, \bar{W}_u)$ and use them in Procedure 3 for the OLP. There are two reasons why we use the (permuted) $\bar{W}_u$ as the target workload vector for the OLP. First, the CQN throughput function, TH, is unimodal (maximal at $\bar{W}_u$) and well-behaved with respect to the workload vector (Stecke and Solberg 1985, Stecke 1986, Lee et al. 1991a). Thus, the closer the actual workload vector obtained from the

OLP is to $\bar{W}_u$, the higher the resulting throughput is. Second, those configurations associated with one $\bar{S}_u$ may have several workload vectors of which lexicographically ordered $\bar{W}$'s are not identical. This happens when one or more workload bound constraints bind for some configurations. In this case, $\bar{W}_u$ serves a collective representative for them.

Procedure 4 below finds two effective orderings (i.e., permutations) of $(\bar{S}_u, \bar{W}_u)$. These orderings have small and large target workloads ordered alternately in a zigzag fashion (the machine vector is also ordered accordingly). One ordering starts with a large target workload and the other with a small workload. One reason for this is the opportunity to assign operations with larger processing times, which tend to be the most difficult to "fit" in the context of the OLP, to various machine groups spread throughout the system. We found this to be preferable to having machine groups with large target workloads clustered in one portion of the flow system. This is because clustering target workloads makes it difficult to achieve actual workloads close to the targets while simultaneously satisfying the flexibility capacity and unidirectional flow constraints. Since there are still a large number of possible orderings for each of the two ordering patterns, this procedure uses the workload bounds obtained from the WBP and finds a unique ordering for each pattern. This is achieved by matching large (small) target workloads with large (small) U_is (L_is). These two ordered workload vectors are referred to as the zigzag target workload vectors and denoted as $\bar{z}^1$ and $\bar{z}^2$, respectively. This procedure is appropriate only for *the unbalanced target workloads* since all orderings are identical when the target workloads are balanced. The procedure consists of two parts. The first five steps of the procedure describe how to find $\bar{z}^1$, while Step 6 describes how to find $\bar{z}^2$.

Procedure 4. A procedure to find the two zigzag target workloads, $\bar{z}^1$ and $\bar{z}^2$

Step 1. Partition elements of the target workload vector $\bar{W}_u$ into two sets, Z_L and Z_S, such that the cardinalities of Z_L and Z_S are $\lceil M/2 \rceil$ and $\lfloor M/2 \rfloor$, respectively, and any workload in Z_L is greater than or equal to every workload in Z_S.

Step 2. Assign workloads in Z_L to the odd numbered groups by matching the largest workload in Z_L with the largest U_i among the odd numbered groups and the second largest workload in Z_L with the second largest U_i among the groups and so on. If there is a tie, choose one match arbitrarily.

Step 3. Do pairwise interchanges among the workloads assigned in Step 2 until interchanging any two workloads does not reduce the amount of infeasibility (i.e., $\Sigma_{\text{odd } i} \max \{z_i^1 - U_i, 0, L_i - z_i^1\}$).

Step 4. Assign workloads in Z_S to the even numbered groups by matching the smallest workload in Z_S with the smallest L_i among the even numbered groups and the second smallest workload in Z_S with the second smallest L_i among the groups and so on. If there is a tie, choose one which makes the resulting workloads more zigzagged (i.e., $\max_{i \varepsilon I} | z_i^1 - z_{i-1}^1 |$, where I is a set of groups that are tied and even numbered). If there is still a tie, choose one arbitrarily.

Step 5. Do pairwise interchanges among the workloads assigned in Step 4 until interchanging any two workloads does not reduce the amount of infeasibility.

Step 6. To find $\bar{z}^2$, repeat Steps 1 through 5 with the following changes: in Step 1, exchange the sizes of Z_L and Z_S; in Step 2, replace the odd numbered groups by the even numbered groups; in Step 4, replace even by odd.

Example 3. Suppose M = 5, $\bar{W}_u = (10, 18, 30, 32, 35)$, $\bar{L} = (3, 33, 4, 4, 20)$, and $\bar{U} = (28, 45, 47, 48, 35)$. In this example, there are 120 possible orderings of $\bar{W}_u$ and Procedure 4 provides two potential orderings among them. After Step 1, we have $Z_L = \{30,32,35\}$ and $Z_S = \{10,18\}$ for $\bar{z}^1$, and after Step 5, we have $\bar{z}^1 = (30, 18, 35, 10, 32)$. Similarly, for $\bar{z}^2$, we have $Z_L = \{32,35\}$ and $Z_S = \{10,18,30\}$. Initially, 32 in Z_L is matched with $U_2 = 45$ and 35 with $U_4 = 48$, but they are interchanged to reduce the amount of infeasibility. Thus, $\bar{z}^2 = (10, 35, 18, 32, 30)$. We show the effectiveness of these zigzag orderings in the experimental results of Section 6.1.

3.4 The Proposed Methodology

We now present the overall methodology to solve Problem P0, using the subproblems and their solution methods discussed in the previous sections. An example follows.

Procedure 5. Methodology to solve Problem P0

Step 1. Solve the WBP using Procedures 1 and 2.

Step 2. Solve the WAGP and find all configurations, $(\bar{S}, \bar{W})$, that meet demand. If none, terminate. We either need to acquire more resources or find another set of part types. Otherwise, sort configurations in decreasing order of throughput. Order a machine vector in each configuration lexicographically from the top of the sorted list, and eliminate any duplicates. Pick the first configuration in the list.

Step 3. If the current configuration is balanced, then solve the OLP using Procedure 3 with balanced target workloads and go to Step 4. Otherwise, solve the workload allocation problem without the workload bound constraints and find two zigzag target workload vectors from Procedure 4. For each of the two vectors, solve the OLP using Procedure 3.

Step 4. Calculate the actual throughput using the operation assignment from the OLP. If this throughput is greater than the incumbent value, then update the incumbent solution.

Step 5. If either all configurations in the list are exhausted, or the incumbent throughput is no less than the theoretical throughput found from the WAGP for the next configuration in the list, then write the incumbent solution and terminate. (In the latter case, a better solution cannot be found from the remaining configurations since the theoretical throughput from the WAGP is no less than its corresponding actual throughput from the OLP). Otherwise, pick the next configuration and go to Step 3.

Example 4. Suppose that an aggregate part consists of 14 operations with their processing times and precedence as shown in Figure 4a. The flexibility capacity, R, is set to 5. Suppose that the aggregate demand, d, is 650 parts per period and the processing capacity, P, is 10,000 minutes per machine per period. Conveyors are used to move parts and the total material handling time for one part, W_0, is 20 minutes. Available resources, (K, N), are (8, 7). The number of groups, M, is set to the minimum, which is 3. The workload bounds from Step 1 of Procedure 5 are obtained as $\bar{L}$ = (18, 11, 11) and $\bar{U}$ = (34, 34, 33). The WAGP in Step 2 provides two configurations which meet the demand: (a) $\bar{S}_1$ = (3, 3, 2), $\bar{W}_1^*$ = (29.9, 29.9, 15.2) and (b) $\bar{S}_2$ = (3, 2, 3), $\bar{W}_2^*$ = (29.9, 15.2, 29.9). Their throughputs are TH=657.4. The second configuration is eliminated in Step 3, since $\bar{S}_2$ can be permuted into $\bar{S}_1$. $\bar{W}_1^*$ is optimal for the unconstrained workload allocation problem, since $\bar{L} < \bar{W}_1^* < \bar{U}$. The two zigzag target workload vectors found in Step 3 are (29.9, 15.2, 29.9) and (29.9, 29.9, 15.2). The solution of the OLP with the first workload vector is the higher throughput of 653.1 with the actual workload $\bar{W}$ = (31, 18, 26) and $\bar{S}$ =(3, 2, 3). The corresponding operation assignments are shown in Figure 4b. For this small problem, we can verify that this solution is optimal for P0 by enumerating all feasible operation assignments and machine vectors and computing the corresponding throughputs.

<Insert Figures 4a and 4b>

4. Experimental Results

This section consists of three parts of experimental results. The first part shows the effectiveness of the zigzag target workload vectors obtained from Procedure 4. The second part shows that Procedure 5 finds a near-optimal solution for P0. The third part shows that Procedure 5 outperforms the existing method in both effectiveness and efficiency.

4.1 Effectiveness of the zigzag target workload vectors

A number of experiments were conducted to investigate the effectiveness of the two zigzag target workload vectors that Procedure 4 identifies. M is equal to four or five. This limits the maximum number of orderings to be 24 and 120, respectively, which we considered to be reasonable for enumeration. In addition, the following set of parameters are used to generate test problems: (a) two numbers of operations (n=20 for M = 4 and n=30 for M = 5), (b) two densities of precedence diagram (.05 and .50), where density is defined as the ratio of a number of present precedent arcs to the total number of possible precedent arcs, i.e., $\binom{n}{2}$, and precedent arcs are randomly generated such that each arc is equally likely, (c) operation time t_j is randomly generated from a discrete uniform distribution between 1 and 9 minutes, (d) aggregate demand per period d=400, (e) processing capacity per machine per period P=10,000 minutes, (f) flexibility capacity R=6, and (g) use of a conveyor with 5 minutes for average material handling time between two machines. The numbers of available machines and pallets, i.e., (K, N) are assigned such that the maximum throughput from the WAGP is greater than or equal to the demand, 400. Five problems are tested for each of four combinations (two values each of both M and density) of the parameter set. Procedure 3 is solved with the termination tolerance ε replaced by the number of iterations limited to five.

The following two statistics are collected for each unbalanced configuration found by the WAGP that meets demand: (i) ranking recorded as a / b which means that the zigzag workload vector with the larger throughput of the two achieved the a^{th} largest throughput among the b distinct $\pi(\bar{W}_u)$s; (ii) two throughput percentages (zigzag, worst), where 'zigzag' is the throughput achieved by the better zigzag workload vector divided by the largest throughput among all distinct $\pi(\bar{W}_u)$s and 'worst' is the smallest throughput divided by the largest throughput. The 'b' is usually less than the maximum number, since many $\pi(\bar{W}_u)$s are identical when several W_{iu}'s are equal. These statistics are summarized in Tables 2 and 3.

< Insert Tables 2 and 3 >

Experimental results show that the zigzag workload vectors are very effective. When M = 4, the better zigzag workload vector of the two achieved the largest throughput 10 times out of 17 unbalanced configurations. The better zigzag workload vector also had at least the third largest throughput 16 times out of 17. On average, the zigzag workload vector achieved 99.8 percent of the largest throughput for both densities. When M = 5, the better zigzag workload vector achieved the largest throughput 11 times out of 21 and at least the third largest throughput 20 times out of 21. On average, the workload vector achieved 99.7 percent for density = .05 and 99.1 percent for density = .50. Further evidence on the effectiveness of the zigzag workloads is presented for M > 5 in the following section. The results also reveal that the ordering of the target workload vector can significantly affect the quality of the solution. For example, in Table 3, when the density is .50, the ordering that gives the smallest throughput was only 72.7 percent of the largest throughput for the second configuration of Problem 3, and only 77 percent for the first configuration of Problem 2.

4.2 Experiments with Procedure 5

A number of experiments were conducted for the proposed method to solve P0, Procedure 5. Procedure 5 was coded in PASCAL and FORTRAN and run on a mainframe computer, IBM 3090-600. The same parameter values are used as before except for the following ones. Two demand levels are d=100 and 200. The number of operations, n, is 100. Two different flexibility capacities are R=15 and 30, since R=30 was used by Ammons et al. (1985) for a PCB assembly system manufacturing computers. Average material handling between two groups is 10, about the average processing time for two operations. Conveyors or stop-and-go AGVs are used to move pallets between groups.

Experimental results for three aggregate parts, eight problems for each aggregate part, are summarized in Tables 4, 5, and 6. For each problem, the following statistics are recorded: the (K, N) used, the upper bound on throughput obtained from the WAGP, the number of $\overline{S}_u$'s, the actual throughput and machine vector obtained from the proposed method, the ratio of the actual throughput to the upper bound, the number of operations assigned to each group, and the CPU time.

<Insert Tables 4, 5, and 6>

The proposed method found a feasible solution for 20 problems. It did not find a feasible solution for 4 problems, although their upper bound throughputs are greater than the target demand. These 4 problems occurred when R=15 and d=200 for aggregate parts 1 and 3. When R=30, the FFS needs only four groups to accommodate 100 operations, compared to seven groups required for R=15. Machines that are spread over a smaller number of groups lead to a smaller number of material handling operations and a larger number of parallel machines (i.e., the more pooling of resources). Consequently, under the larger flexibility capacity (i.e., when R=30), the proposed method achieves an average of 13.6 percent higher throughput than when R=15. A smaller M results in a larger number of $\overline{S}_u$'s to which OLP can be applied, and consequently, a longer CPU time. A larger number of parallel machines also enhances both system reliability and routing flexibility. When R=30, a better fit is achieved in the operation assignments of the OLP. The actual throughput deviates from its upper bound by an average of 0.8% when R=30, compared to 2.9% when R=15. This is due to the fact that more flexible machines can process a larger number of operations, allowing more freedom to maneuver operation assignments.

As demand increases from 100 to 200, a larger numbers of machines and pallets are required, which leads to a larger number of $\overline{S}_u$'s and longer CPU time. As density increases from 0.05 to 0.50, there are more precedence arcs among operations. This leads to tighter workload bounds from the WBP, a smaller upper bound throughput, and a smaller number of $\overline{S}_u$'s from the WAGP that meet demand, and shorter CPU time. The actual throughputs obtained are insensitive to density except for one case, where R=15 and d=200 for aggregate part 3. In this case, the throughput decreases from 193.4 to 169.1 as density increases from 0.05 to 0.50. This insensitiveness was against our expectation that the high density of the precedence diagram restricts the assignment of operations in the OLP and results in poor fit and low throughput. One possible interpretation for this is that the flexibility capacity R=15, which allows machines to accommodate up to 15 operations, is still large enough to temper an adverse effect of additional precedence arcs on operation assignment.

Overall, the proposed method works very well under various experimental conditions, providing a near-optimal solution most of the time. The average difference between the actual throughput and the corresponding upper bound throughput is only 1.8 percent. This also gives evidence that the workload bounds obtained from Procedures 1 and 2 to solve the WBP are effective and serve the tight workload bound constraint for the WAGP. All 24 problems require less than 30 seconds of CPU time when (K, N) does not exceed (12,35). Twenty one problems among them require less than 15 seconds. This is reasonable since this problem is addressed in short-term planning and would be solved about once a day or week.

4.3 Comparison with an existing method

Although solving the grouping and loading problem simultaneously for FFSs has not been studied before, we took a solution method for FMSs available in the literature (Kim and Yano 1992) and modified it for comparative study with our proposed method. The Kim and Yano method (KY) is basically an enumeration scheme that consists of the following three steps: (1) generate all possible machine group configurations; (2) find the ideal continuous workload allocation that maximizes throughput with no workload bounds for each configuration; (3) rank configurations in decreasing order of throughput and solve the loading problem one by one in the sorted list until either a feasible solution is found or the list is exhausted.

In order to solve the loading problem in (3) for KY, Procedure 3 is used for a given ordering since KY consider a special case with density 0.0, i.e., no precedence relations among operations. Since it is not realistic to evaluate all possible orderings (for example, 5040 orderings for M=7), Procedure 4 is applied to generate two zigzag orderings with workload bounds specified as $L_i = 0$ and U_i=TW for every group. In order to avoid unnecessary computation for KY, all configurations with their ideal throughputs less than demand are eliminated after (2). KY is applied to the same set of test problems that were used in the previous section. The results are summarized in Tables 7 to 9 including the following statistics: the number of $\bar{S}_u$'s with their ideal throughputs no less than demand, the actual throughput and machine vector obtained, ratio of the actual throughput to the ideal upper bound throughput, and CPU time. However, the tables exclude those cases where K=7 and M=7. In these cases, there is only one possible $\bar{S}_u$ which is balanced, and both methods found the same solution.

<Insert Tables 7, 8, and 9>

The proposed method is superior to KY in both effectiveness and efficiency. The proposed method finds solutions with an average of 9.5% (i.e., 14.2 parts) larger throughput than KY. There is no significant difference in CPU time when demand is 100, but when demand is 200, KY requires an average of 156% (i.e., 10.3 seconds) longer CPU time than the proposed method. This is because KY deals with a larger number of $\bar{S}_u$'s, since the ideal throughput is obtained without the workload bound constraints. We expect that this will become more evident for higher demand since higher demand requires the larger number of machines K and the total number of $\bar{S}_u$'s exponentially increases with respect to K.

Ranking $\bar{S}_u$'s in decreasing order of ideal throughput in Step 3 also causes longer CPU time since larger ideal throughput is associated with more unbalanced $\bar{S}_u$'s and more unbalanced ideal workloads, but may not be achievable in Step 4 due to limited flexibility capacity and tight precedence relationships. As a result, a feasible solution is not found on many occasions until the lower part of the ranked list is reached. On the other hand, the proposed method avoids this problem by obtaining effective workload bounds from the WBP. These workload bounds are used in the WAGP to give more realistic target workloads and ideal throughputs. Thus, ordering $\bar{S}_u$'s based on these throughputs in Step 2 of Procedure 5 helps to find a good solution faster than ordering by KY. Another advantage of the proposed method over the other is that it always gives a tighter upper bound on the throughput from the WAGP, which better serves in determining the quality of the solutions obtained from the heuristics. The proposed method achieves an average of 97.7% of its upper bound throughput, whereas KY achieves an average of 87.2%.

5. Summary and Future Research Issues

In this paper, we studied two important production planning problems for FFSs: grouping and loading problems. We present a method which solves these two problems simultaneously with the objective of maximizing system utilization. With the precedence requirements, the complex throughput function and the machine flexibility constraint inhibit seeking an optimum solution. We present a heuristic method which decomposes the large optimization problem into three interrelated subproblems. The specific contributions and findings are summarized as follows:

(1) We show that the proposed method is effective and finds a near-optimum solution most of the time for a moderate size of problems with up to 100 operations, 7 groups, 12 machines, and 35 pallets. Experiments with 24 various test problems show that the throughput of the solution obtained is within an average of 1.8 percent of its upper bound. Computation time is also reasonable without exceeding 30 seconds on a mainframe computer.

(2) We show that the proposed method is superior to an existing method from the FMS literature. Experiments with 18 test problems show that the proposed method achieves 9.5% larger throughput than the existing method. The proposed method is also more efficient than the iterative approach, which is based on an enumeration scheme. The latter requires 156% longer CPU time when demand is 200 parts per period. This becomes more evident as the number of machines or demand increases. This result shows the importance of addressing the grouping and loading problems for FFSs simultaneously rather than hierarchically. Another advantage of the proposed method is that it always provides a tighter upper bound on the throughput than the other, which helps to better assess the quality of the solution obtained.

(3) We define the WBP for the first subproblem and present a solution method. Even this subproblem is hard to solve optimally due to a complex combinatorial nature and so we present a heuristic method by exploiting the flexibility capacity and part flow constraints and transforming it to the MCGP, which is easier to solve. Experimental results show that workload bounds obtained by this method are effective since the upper bound throughput from the WAGP with these workload bounds is close to the actual throughput obtained.

(4) We define the OLP for the third subproblem and present a solution method. Since this subproblem is more difficult than the assembly line balancing problem, a heuristic method is developed. The ordering of the target workload vector elements can make a significant impact on operation assignment and the actual throughput. This is unique and is

not shared with FMS loading problems and the line balancing problem. FMS loading problems do not need to deal with precedences and are independent of the ordering, whereas the line balancing problem considers precedences but deal with a balanced configuration only. We develop a procedure which finds two effective zigzag orderings. Experiments with 38 test problems in Section 4.1 show that the throughput from the OLP with these two orderings is at least 96.4% and average 99.5% of the maximum throughput obtained from all possible orderings. The effectiveness of these two orderings is further reinforced by experimental results with an additional 24 test problems in Section 4.2.

We leave two issues for future research. First, a similar proposed method can be applied to other types of flow systems such as open asynchronous lines which can be modeled as an open tandem queueing network. In this case, the objective is to minimize the total number of WIP parts rather than to maximize the system utilization. The same proposed method can be applied except that the WAGP requires a different solution method like that given by Calabrese (1992). Second, the scope of the proposed method can be broadened to include other planning problems such as part type selection. Sometimes, all part types required to be produced for one period cannot be produced at the same time because machine flexibility is limited and all the necessary tools cannot be loaded into tool magazines. Then the issue is how to divide part types into batches so as to minimize total makespan to complete all production requirements. To each selected batch of part types, the proposed method can be applied.

References

Afentakis, P. 1989. A Loop Layout Design Problem for Flexible Manufacturing Systems. *International Journal of Flexible Manufacturing Systems*, Vol. 1, pp. 175-196.

Aho, A., Hopcroft, J. and Ullman, J. 1974. *The Design and Analysis of Computer Algorithms.* Addison-Wesley, Reading, MA.

Ammons, J. C., Lofgren, C. B. and McGinnis, L. F. 1985. A Large Scale Machine Loading Problem in Flexible Assembly. *Annals of Operations Research*, Vol. 3, pp. 319-332.

Baker, K. E. 1974. *Introduction to Sequencing and Scheduling*, John Wiley & Son, New York, NY.

Baybars, I. 1986. A Survey of Exact Algorithms for the Simple Assembly Line Balancing Problem. *Management Science*, Vol. 32, pp. 909-932.

Berrada, M. and Stecke, K. E. 1986. A Branch and Bound Approach for Machine Load Balancing in Flexible Manufacturing Systems. *Management Science*, Vol. 32, pp. 1316-1335.

Calabrese, J. M. 1992. Optimal Workload Allocation in Open Networks of Multiserver Queues. *Management Science*, Vol. 38, pp. 1792-1802.

Dallery, Y. and Stecke, K. E. 1990. On the Optimal Allocation of Servers and Workloads in Closed Queueing Networks. *Operations Research*, Vol. 38, pp. 694-703.

Graves, S. C. and Redfield, C. H. 1988. Equipment Selection and Task Assignment for Multiproduct Assembly System Design. *International Journal of Flexible Manufacturing Systems*, Vol. 1, pp. 31-50.

Greene, T. J. and Sadowski, R. P. 1986. A Mixed Integer Program for Loading and Scheduling Multiple Flexible Manufacturing Cells. *European Journal of Operational Research,* Vol. 24, pp. 379-386.

Harmon, R. L. and Peterson, L. D. 1990. *Reinventing the Factory*, The Free Press, New York, NY.

Johnson, R. V. 1988. Optimally Balancing Large Assembly Lines with 'FABLE'. *Management Science*, Vol. 34, pp. 240-253.

Kamath, M., Suri, R. and Sanders, J. 1988. Analytical Performance Models for Closed-Loop Flexible Assembly Systems. *International Journal of Flexible Manufacturing Systems*, Vol. 1, pp. 51-84.

Kim, Y.-D. and Yano, C. A. 1992. An Iterative Approach to System Setup Problems in Flexible Manufacturing Systems. *International Journal of Flexible Manufacturing Systems*, Vol. 4, pp. 183-209.

Kim, Y.-D. and Yano, C. A. 1993. Heuristic Approaches for Loading Problems in Flexible Manufacturing Systems. *IIE Transactions*, Vol. 25, pp. 26-39.

Kouvelis, P. and Lee, H. L. 1995. An Improved Algorithm for Optimizing a Closed Queueing Network Model of a Flexible Manufacturing System. *IIE Transactions*, Vol. 27, pp. 1-8.

Lashkari, R. S., Dutta, S. P. and Padhye, A. M. 1987. A New Formulation of Operation Allocation Problem in Flexible Manufacturing Systems: Mathematical Modeling and Computational Experience. *International Journal of Production Research*, Vol. 25, pp. 1267-1283.

Lee, H. F. 1989. A Methodology for Capacity Planning in Flexible Assembly Systems. Ph.D. dissertation, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, MI.

Lee, H. F. and Dooly, D. R. 1996. Algorithms for the Constrained Maximum-Weight Connected Graph Problem. *Naval Research Logistics*, Vol. 43, pp. 985-1008.

Lee, H. F. and Johnson, R. V. 1991. A Balancing Strategy for Designing Flexible Assembly Systems. *International Journal of Flexible Manufacturing Systems,* Vol. 3, pp. 91-120.

Lee, H. F., Srinivasan, M. M. and Yano, C. A. 1990. A Framework for Capacity Planning and Machine Configuration for Flexible Assembly Systems. Technical Report 90-10, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, MI.

Lee, H. F., Srinivasan, M. M. and Yano, C. A. 1991a. Characteristics of Optimal Workload Allocation in Closed Queueing Networks. *Performance Evaluation,* Vol. 12, pp. 255-268.

Lee, H. F., Srinivasan, M. M. and Yano, C. A. 1991b. The Optimal Configuration and Workload Allocation Problem in Flexible Manufacturing Systems. *International Journal of Flexible Manufacturing Systems,* Vol. 3, pp. 213-230.

Liu, C.-M. and Sanders, J. 1988. Stochastic Design Optimization of Asynchronous Flexible Assembly Systems. *Annals of Operations Research,* Vol. 15, pp. 131-154.

Reiser, M. and Lavenberg, S. 1980. Mean-Value Analysis of Closed Multichain Queueing Networks. *Journal of Assoc. Computing Machinery,* Vol. 27, pp. 313-322.

Sethi, A. K. and Sethi, S. P. 1990. Flexibility in Manufacturing: A Survey. *International Journal of Flexible Manufacturing Systems,* Vol. 2, pp. 289-328.

Shanker, K. and Tzen, Y. J. 1985. A Loading and Dispatching Problem in a Random Flexible Manufacturing System. *International Journal of Production Research,* Vol. 23, pp. 579-595.

Singh, N. 1996. *Systems Approach to Computer-Integrated Design and Manufacturing,* John Wiley & Sons, New York, NY.

Smith, T. and K.E. Stecke. 1996. On the Robustness of Using Balanced Part Mix Ratios to Determine Cyclic Part Input Sequences into Flexible Flow Systems. *International Journal of Production Research,* Vol. 34, pp. 2925-2941.

Solberg, J. J. 1977. A Mathematical Model of Computerized Manufacturing Systems. *The 4th International Conference on Production Research.*, Tokyo, Japan (London: Taylor & Francis), pp. 22-30.

Stecke, K. E. 1983. Formulation and Solution of Nonlinear Integer Production Planning Problems for Flexible Manufacturing Systems. *Management Science*, Vol. 29, pp. 273-288.

Stecke, K. E. 1986a. A Hierarchical Approach to Solving Machine Grouping and Loading Problems of Flexible Manufacturing Systems. *European Journal of Operations Research*, Vol. 24, pp. 369-378.

Stecke, K. E. 1986b. On the Nonconcavity of Throughput in Certain Closed Queueing Networks. *Performance Evaluation*, Vol. 6, pp. 293-305.

Stecke, K. E. and Kim, I. 1991. A Flexible Approach to Part Type Selection in Flexible Flow Systems Using Part Mix Ratios. *International Journal of Production Research*, Vol. 29, pp. 53-75.

Stecke, K. E. and Solberg, J. J. 1981. Loading and Control Policies for a Flexible Manufacturing System. *International Journal of Production Research*, Vol. 19, pp. 481-490.

Stecke, K. E. and Solberg, J. J. 1985. The Optimality of Unbalancing Both Workloads and Machine Group Sizes in Closed Queueing Networks of Multi-Server Queues. *Operations Research*, Vol. 33, pp. 882-910.

Suri, R. and Hildebrant, R. 1984. Modeling Flexible Manufacturing Systems Using Mean-Value Analysis. *Journal of Manufacturing Systems*, Vol. 3, pp. 27-38.

Talbot, F. B. and Patterson, J. H. 1984. An Integer Programming Algorithm with Network Cuts Solving the Assembly Line Balancing Problem. *Management Science*, Vol. 30, pp. 85-99.

Templemeier, H. and Kuhn, H. 1993. *FMSs: Decision Support For Design and Operation*, John Wiley & Sons, New York, NY.

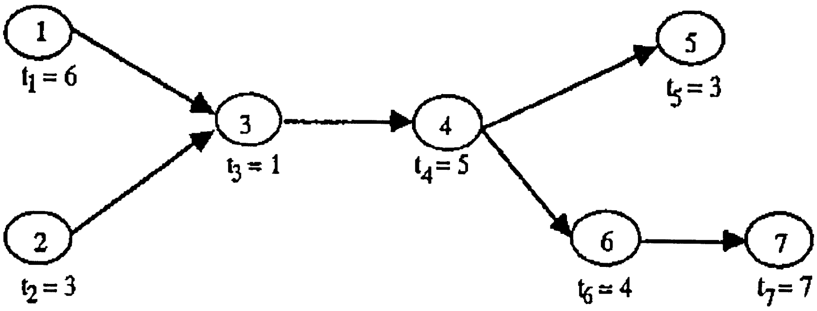


Figure 1. Precedence diagram for an aggregate part.

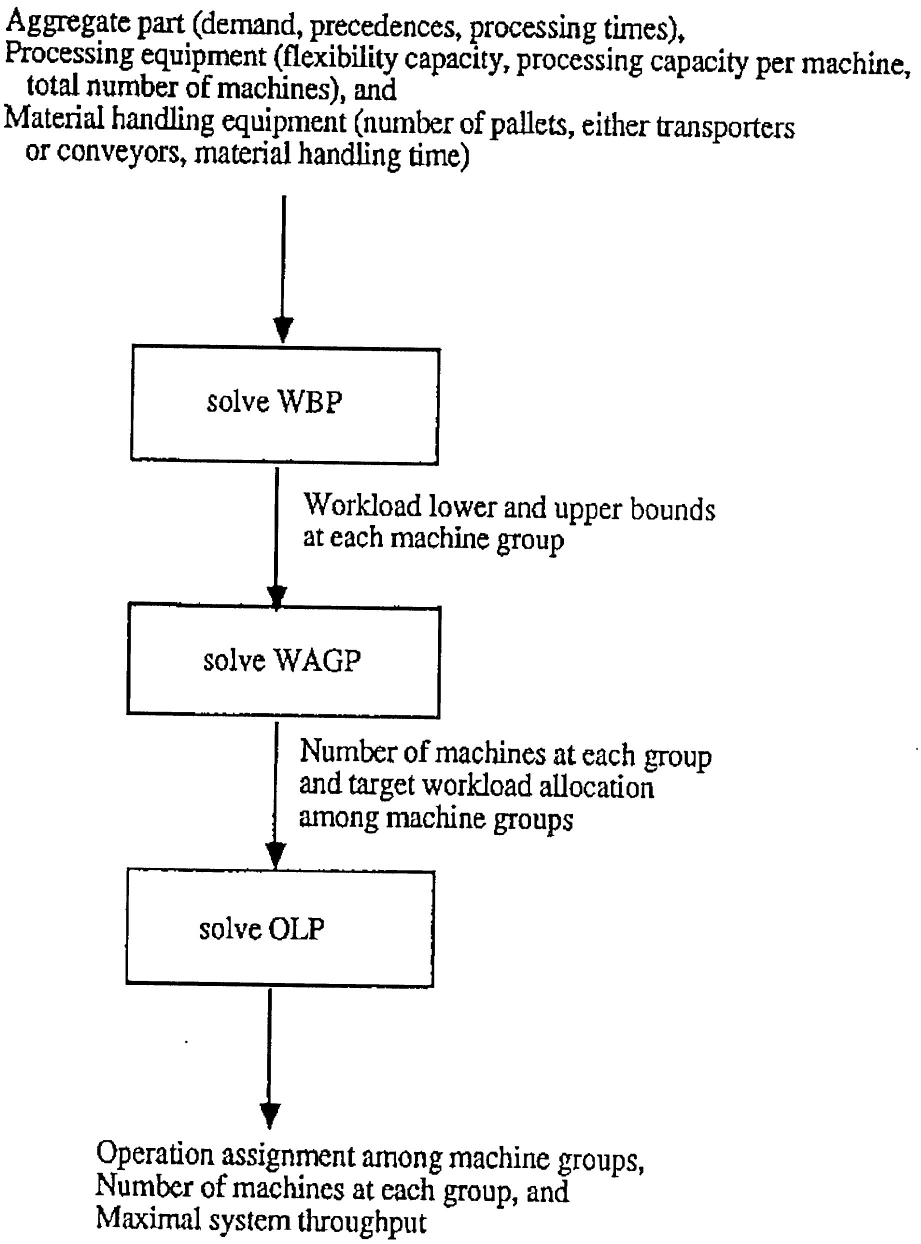
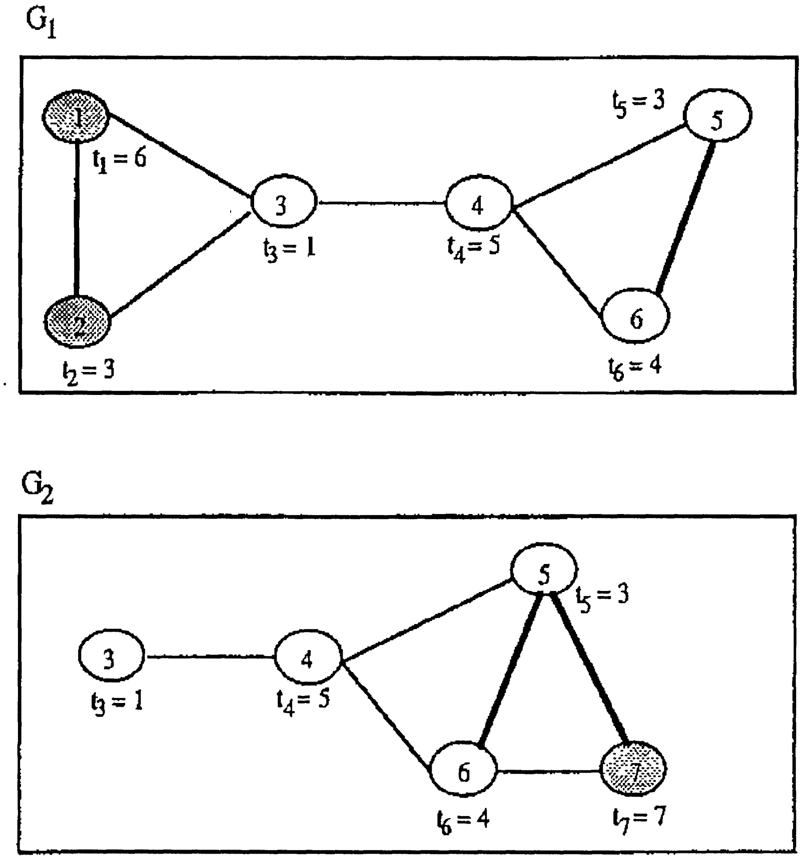

Aggregate part (demand, precedences, processing times),
Processing equipment (flexibility capacity, processing capacity per machine, total number of machines), and
Material handling equipment (number of pallets, either transporters or conveyors, material handling time)
solve WBP
Workload lower and upper bounds at each machine group
solve WAGP
Number of machines at each group and target workload allocation among machine groups
solve OLP
Operation assignment among machine groups,
Number of machines at each group, and
Maximal system throughput

Figure 2. The proposed decomposition method: relationship among three subproblems.



Darker edges are newly added.
Shaded circles 1 and 2 indicate operations that must be assigned to machine group 1, while shaded circle 7 must be assigned to machine group 2.

Figure 3. Transformed undirected graphs for the MCGP.

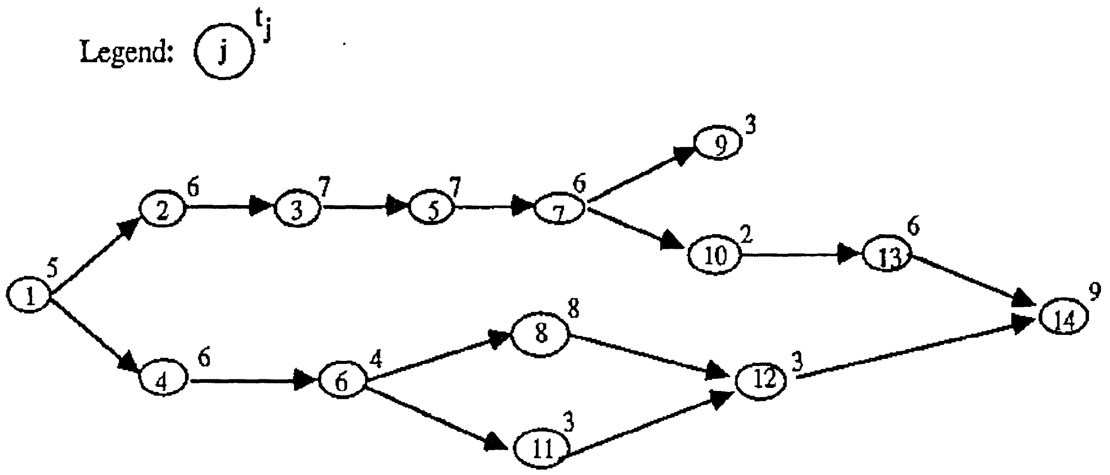
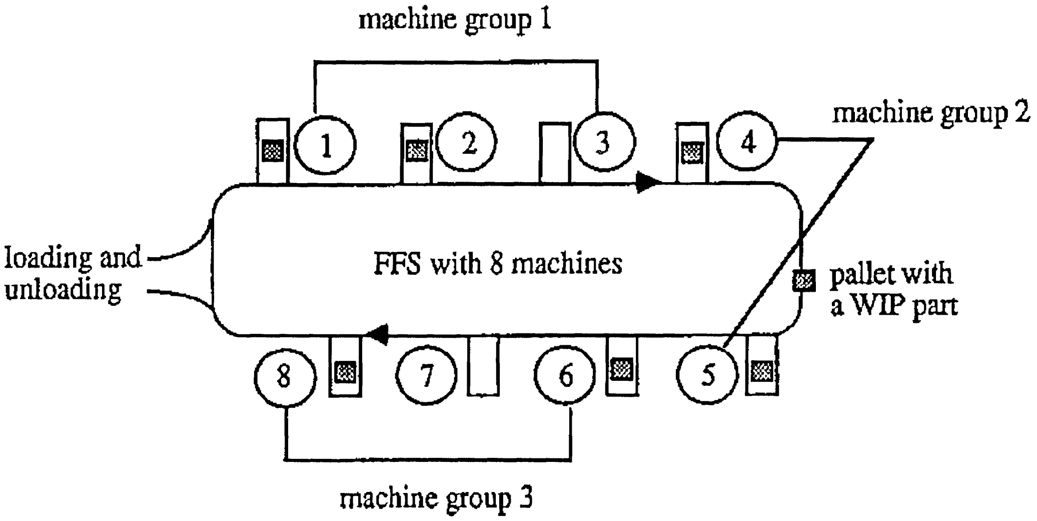


Figure 4a. Precedence diagram for Example 4.



Machines 1, 2, and 3 form machine group 1, each processing operations 1, 2, 3, 4, and 5.
Machines 4 and 5 form machine group 2, each processing operations 6, 7, 9, 10, and 12.
Machines 6, 7, and 8 form machine group 3, each processing operations 8, 11, 13, and 14.

Figure 4b. The grouping and loading solution for Example 4.

Table 1. Notation.

A_c = set of groups using machines of type c

C = number of machine types

d = demand of the aggregated part

f_j = the first machine group in the visit sequence to which operation j can be assigned

K_c = number of available machines of type c

l_j = the last machine group in the visit sequence to which operation j can be assigned

$\bar{L} = (L_i)$, where L_i is the lower bound of the workload at group i, W_i

M = number of machine groups in the system

M_0 = the smallest M possible

n = total number of operations in the aggregate part

N = number of pallets circulating in the system

$\pi(\cdot)$ = a permutation function among elements of a vector

P_0 = total available material handling time by a transporter or conveyor per period

$\bar{P} = (P_i)$, where P_i is the total available processing time by a machine at group i per period

R_c = flexibility capacity of machines of type c

S_0 = number of AGVs or transporters if required

$\bar{S} = (S_i)$, where S_i is the number of machines at group i

$\bar{S}_u$ = lexicographically ordered $\bar{S}$ such that $S_1 \leq \ldots \leq S_M$

t_j = average processing time of operation j of the aggregated part

V_i = set of operations that can be assigned to machine group i, i.e., $V_i = \{j \mid f_j \leq i \leq l_j\}$

$\bar{U} = (U_i)$, where U_i is the upper bound of the workload at group i, W_i

W_0 = average total material handling time required to produce a part

$\bar{W} = (W_i)$, where W_i is the workload (average total processing time) at machine group i required to produce a part

$\bar{W}_u$ = optimal workload allocation that maximizes TH for a given $\bar{S}_u$ under no workload bound constraints

TH = throughput of the CQN for given N, M, P_0, S_0, W_0, $\bar{P}$, $\bar{S}$, and $\bar{W}$

TW = total workload (i.e., average total processing time) required to produce a part

X_{ij} = the assignment variable $= \begin{cases} 1 & \text{if operation j is assigned to group i;} \\ 0 & \text{otherwise.} \end{cases}$

$\bar{z}$ = zigzag target workload vector

Table 2. Experiments with the zigzag target workloads for M = 4.

	Density = .05			Density = .50		
Replication no.	Ranking	Throughput Percentage		Ranking	Throughput Percentage	
(K, N)	a / b	Zigzag	Worst	a / b	Zigzag	Worst
Problem 1 (1)[&]	1 / 4	100	98.8	1 / 4	100	97.7
(7, 7)						
Problem 2 (1)	1 / 4	100	99.6	2 / 4	99.9	98.8
(7, 7) (2)	2 / 12	99.5	85.9	1 / 12	100	87.0
Problem 3 (1)	2 / 12	99.9	97.2	3 / 12	98.4	90.6
(8, 7) (2)	3 / 6	99.3	92.3	1 / 6	100	89.6
Problem 4 (1)	1 / 6	100	95.8	1 / 6	100	90.5
(6, 7) (2)	1 / 4	100	95.1	–[*]	–	–
Problem 5 (1)	1 / 4	100	99.0	2 / 4	99.9	99.8
(7, 9) (2)	4 / 12	99.1	92.5	1 / 12	100	86.5
Average		99.8	95.1		99.8	92.6

[&] (i): i indicates the ith unbalanced configuration found by the WAGP that meets demand for the associated test problem.

[*] For Problem 4, the WAGP finds two configurations that meet demand when density = 0.05, but it finds only one configuration when density=0.50.

Table 3. Experiments with the zigzag target workloads for M = 5.

	Density = .05			Density = .50		
Replication no.	Ranking	Throughput Percentage		Ranking	Throughput Percentage	
(K, N)	a / b	Zigzag	Worst	a / b	Zigzag	Worst
Problem 1 (1)[&]	1 / 5	100	99.7	1 / 5	100	97.5
(11, 12) (2)	3 / 30	99.8	85.4	–*	–	–
(3)	1 / 10	100	84.0	–*	–	–
Problem 2 (1)	3 / 20	99.9	90.9	1 / 20	100	77.0
(10, 10) (2)	2 / 30	99.9	86.0	–	–	–
Problem 3 (1)	1 / 5	100	98.0	1 / 5	100	84.1
(9, 9) (2)	3 / 30	99.8	84.2	4 / 30	97.4	72.7
(3)	3 / 10	98.4	86.2	–	–	–
Problem 4 (1)	1 / 5	100	98.6	3 / 5	96.4	95.4
(11, 10) (2)	2 / 30	99.9	83.8	–	–	–
(3)	1 / 10	100	91.3	–	–	–
Problem 5 (1)	2 / 5	99.5	98.6	1 / 5	100	84.0
(9, 10) (2)	1 / 30	100	87.3	1 / 30	100	79.5
(3)	3 / 10	99.1	93.2	–	–	–
Average		99.7	90.5		99.1	84.3

[&](i): i indicates the i^{th} unbalanced configuration found by the WAGP that meets demand for the associated test problem.

*For Problem 1, the WAGP finds three configurations that meet demand when density = 0.05, but it finds only one configuration when density=0.50.

Table 4. Experiments with the proposed method: aggregate part 1.

Demand/period	100			
density *d*	.05		.50	
Problem No.	P-1	P-2	P-3	P-4
flex. capacity R	15	30	15	30
M_0, N, K	7, 25, 7	4, 25, 7	7, 25, 7	4, 25, 7
no. of $\bar{S}_u$'s	1	3	1	2
upper bound TH*	111.6	124.7	111.4	123.8
actual TH	111.6	123.8	109.3	123.8
adopted $\bar{S}$	(1,1,1,1,1,1,1)	(1,2,2,2)	(1,1,1,1,1,1,1)	(2,2,2,1)
oper. assignment	(10,15,15,15,15,15,15)	(10,30,30,30)	(15,14,13,15,15,14,14)	(27,28,30,15)
actual TH / TH*	100%	99.3%	98.1%	100%
CPU time (sec.)	2.7	5.7	0.4	0.2

Demand/period	200			
density *d*	.05		.50	
Problem No.	P-5	P-6	P-7	P-8
flex. capacity R	15	30	15	30
M_0, N, K	7, 35, 12	4, 35, 12	7, 35, 12	4, 35, 12
no. of $\bar{S}_u$'s	3	11	1	7
upper bound TH*	201.3	220.7	201.0	218.6
actual TH	194.0	217.9	194.0	217.1
adopted $\bar{S}$	(2,2,2,2,1,2,1)	(4,2,4,2)	(2,2,2,2,1,2,1)	(2,4,3,3)
oper. assignment	(15,15,15,15,14,15,11)	(30,17,30,23)	(15,15,15,15,14,15,11)	(16,30,29,25)
actual TH / TH*	96.4	98.7%	96.6%	99.3%
CPU time (sec.)	11.0	29.7	0.2	1.8

Table 5. Experiments with the proposed method: aggregate part 2.

Demand/period	100			
density d	.05		.50	
Problem No.	P-9	P-10	P-11	P-12
flex. capacity R	15	30	15	30
M_0, N, K	7, 25, 7	4, 25, 7	7, 25, 7	4, 25, 7
no. of $\bar{S}_u$'s	1	3	1	2
upper bound TH*	108.2	120.9	108.2	120.0
actual TH	108.2	120.0	108.1	120.0
adopted $\bar{S}$	(1,1,1,1,1,1,1)	(2,1,2,2)	(1,1,1,1,1,1,1)	(2,1,2,2)
oper. assignment	(10,15,15,15,15,15,15)	(29,13,30,28)	(14,15,13,15,14,15,14)	(29,13,30,28)
actual TH / TH*	100%	99.3%	99.9%	100%
CPU time (sec.)	2.5	10.2	0.3	0.3

Demand/period	200			
density d	.05		.50	
Problem No.	P-13	P-14	P-15	P-16
flex. capacity R	15	30	15	30
M_0, N, K	7, 35, 13	4, 35, 13	7, 35, 13	4, 35, 13
no. of $\bar{S}_u$'s	4	14	3	8
upper bound TH*	211.8	232.3	209.7	229.9
actual TH	210.2	229.2	208.0	227.9
adopted $\bar{S}$	(2,2,2,2,1,2,2)	(4,1,4,4)	(2,2,1,2,2,2,2)	(2,4,3,4)
oper. assignment	(12,15,15,15,14,14,15)	(30,10,30,30)	(15,15,10,15,15,15,15)	(15,30,25,30)
actual TH / TH*	99.3%	98.7%	99.2%	99.1%
CPU time (sec.)	13.0	25.6	0.7	2.3

Table 6. Experiments with the proposed method: aggregate part 3.

Demand/period	100			
density *d*	.05		.50	
Problem No.	P-17	P-18	P-19	P-20
flex. capacity R	15	30	15	30
M_0, N, K	7, 21, 7	4, 21, 7	7, 21, 7	4, 21, 7
no. of $\bar{S}_u$'s	1	3	1	2
upper bound TH*	112.3	127.6	112.2	126.3
actual TH	112.2	126.4	111.7	126.2
adopted $\bar{S}$	(1,1,1,1,1,1,1)	(2,2,1,2)	(1,1,1,1,1,1,1)	(2,2,2,1)
oper. assignment	(10,15,15,15,15,15,15)	(21,30,19,30)	(13,14,15,13,15,15,15)	(27,28,30,15)
actual TH / TH*	99.9%	99.1	99.6%	99.9%
CPU time (sec.)	1.7	13.0	0.2	0.3

Demand/period	200			
density *d*	.05		.50	
Problem No.	P-21	P-22	P-23	P-24
flex. capacity R	15	30	15	30
M_0, N, K	7, 30, 12	4, 30, 11	7, 30, 12	4, 30, 11
no. of $\bar{S}_u$'s	3	11	2	7
upper bound TH*	206.6	227.3	203.9	224.7
actual TH	193.4	223.4	169.1	223.2
adopted $\bar{S}$	(2,2,2,1,2,1,2)	(2,4,3,3)	(2,2,1,2,2,2,1)	(4,2,3,3)
oper. assignment	(14,15,15,11,15,15,15)	(11,30,29,30)	(15,15,13,15,15,15,12)	(30,16,27,27)
actual TH / TH*	93.6%	98.3%	82.9%	99.3%
CPU time (sec.)	8.9	24.8	0.6	2.0

Table 7. Comparison among the methods: aggregate part 1.

Problem No.		KY	Proposed Method
P-2	no. of $\bar{S}_u$'s	3	3
	adopted $\bar{S}$	(3,1,2,1)	(1,2,2,2)
	upper bound TH^*	126.2	124.7
	actual TH	114.8	123.8
	actual TH / TH^*	91.0	99.3
	CPU time (sec.)	7.2	5.7
P-4	no. of $\bar{S}_u$'s	3	2
	adopted $\bar{S}$	(1,2,1,3)	(2,2,2,1)
	upper bound TH^*	126.2	123.8
	actual TH	103.0	123.8
	actual TH / TH^*	81.6	100
	CPU time (sec.)	0.4	0.2
P-5	no. of $\bar{S}_u$'s	7	3
	adopted $\bar{S}$	(2,2,2,1,2,1,2)	(2,2,2,2,1,2,1)
	upper bound TH^*	207.9	201.3
	actual TH	179.8	194.0
	actual TH / TH^*	86.5	96.4
	CPU time (sec.)	27.6	11.0
P-6	no. of $\bar{S}_u$'s	15	11
	adopted $\bar{S}$	(5,1,3,3)	(4,2,4,2)
	upper bound TH^*	226.1	220.7
	actual TH	205.2	217.9
	actual TH / TH^*	90.8	98.7
	CPU time (sec.)	42.9	29.7
P-7	no. of $\bar{S}_u$'s	7	1
	adopted $\bar{S}$	(1,2,1,2,2,2,2)	(2,2,2,2,1,2,1)
	upper bound TH^*	207.9	201.0
	actual TH	154.6	194.0
	actual TH / TH^*	74.4	96.6
	CPU time (sec.)	1.4	0.2
P-8	no. of $\bar{S}_u$'s	15	7
	adopted $\bar{S}$	(1,4,3,4)	(2,4,3,3)
	upper bound TH^*	226.1	218.6
	actual TH	202.7	217.1
	actual TH / TH^*	89.7	99.3
	CPU time (sec.)	2.0	1.8

Table 8. Comparison among the methods: aggregate part 2.

Problem No.		KY	Proposed Method
P-10	no. of $\bar{S}_u$'s	3	3
	adopted $\bar{S}$	(1,2,1,3)	(2,1,2,2)
	upper bound TH*	122.3	120.9
	actual TH	103.7	**120.0**
	actual TH / TH*	84.8	**99.3**
	CPU time (sec.)	10.4	10.2
P-12	no. of $\bar{S}_u$'s	3	2
	adopted $\bar{S}$	(2,2,2,1)	(2,1,2,2)
	upper bound TH*	122.3	120.0
	actual TH	119.9	**120.0**
	actual TH / TH*	98.0	**100**
	CPU time (sec.)	0.6	0.3
P-13	no. of $\bar{S}_u$'s	11	4
	adopted $\bar{S}$	(2,2,2,2,2,1,2)	(2,2,2,2,1,2,2)
	upper bound TH*	219.3	211.8
	actual TH	200.1	**210.2**
	actual TH / TH*	91.3	**99.3**
	CPU time (sec.)	42.1	13.0
P-14	no. of $\bar{S}_u$'s	18	14
	adopted $\bar{S}$	(1,6,3,3)	(4,1,4,4)
	upper bound TH*	237.7	232.3
	actual TH	201.3	**229.2**
	actual TH / TH*	84.7	**98.7**
	CPU time (sec.)	54.5	25.6
P-15	no. of $\bar{S}_u$'s	11	3
	adopted $\bar{S}$	(2,2,2,2,2,1,2)	(2,2,1,2,2,2,2)
	upper bound TH*	219.3	209.7
	actual TH	198.2	**208.0**
	actual TH / TH*	90.4	**99.2**
	CPU time (sec.)	3.1	0.7
P-16	no. of $\bar{S}_u$'s	18	11
	adopted $\bar{S}$	(4,4,4,1)	(2,4,3,4)
	upper bound TH*	237.7	229.9
	actual TH	220.0	**227.9**
	actual TH / TH*	92.5	**99.1**
	CPU time (sec.)	3.6	2.3

Table 9. Comparison among the methods: aggregate part 3.

Problem No.		KY	Proposed Method
P-18	no. of $\bar{S}_u$'s	3	3
	adopted $\bar{S}$	(3,1,2,1)	(2,2,1,2)
	upper bound TH*	129.5	127.6
	actual TH	118.1	**126.4**
	actual TH / TH*	91.2	**99.1**
	CPU time (sec.)	7.3	13.0
P-20	no. of $\bar{S}_u$'s	3	2
	adopted $\bar{S}$	(3,1,2,1)	(2,2,2,1)
	upper bound TH*	129.5	126.3
	actual TH	101.0	**126.2**
	actual TH / TH*	78.0	**99.9**
	CPU time (sec.)	0.4	0.2
P-21	no. of $\bar{S}_u$'s	7	3
	adopted $\bar{S}$	(2,2,2,1,2,1,2)	(2,2,2,1,2,1,2)
	upper bound TH*	212.2	206.6
	actual TH	**193.4**	**193.4**
	actual TH / TH*	91.2	**93.6**
	CPU time (sec.)	26.1	8.9
P-22	no. of $\bar{S}_u$'s	15	11
	adopted $\bar{S}$	(4,3,4,1)	(2,4,3,3)
	upper bound TH*	233.8	227.3
	actual TH	217.8	**223.4**
	actual TH / TH*	93.2	**98.3**
	CPU time (sec.)	38.3	24.8
P-23	no. of $\bar{S}_u$'s	7	2
	adopted $\bar{S}$	(2,2,2,1,2,1,2)	(2,2,1,2,2,2,1)
	upper bound TH*	212.2	203.9
	actual TH	155.9	**169.1**
	actual TH / TH*	73.5	**82.9**
	CPU time (sec.)	1.2	0.6
P-24	no. of $\bar{S}_u$'s	15	7
	adopted $\bar{S}$	(5,1,3,3)	(4,2,3,3)
	upper bound TH*	233.8	224.7
	actual TH	202.7	**223.2**
	actual TH / TH*	86.7	**99.3**
	CPU time (sec.)	1.7	2.0