Loading and Batching to Minimize Flow Times in Flexible Manufacturing Systems

by

Jiunn-Min Lee

Department of Information Management
College of Management, Fu-Jen Catholic University
Taipei 24205, Taiwan
Tel: 886-2-2903-1111 ext. 2908
Fax: 886-2-2903-7384, 886-2-2903-2238
E-mail: im1008@mails.fju.edu.tw

and

Chen-Hua Chung

School of Management
Decision Science & Information Systems Area
Carol Martin Gatton College of Business and Economics
University of Kentucky
Lexington, KY 40506-0034
Tel: (859) 257-3258
Fax (859) 257-8031
E-mail: bad180@uky.edu

April, 2003

Loading and Batching to Minimize Flow Times in Flexible Manufacturing Systems

Abstract

A mixed integer nonlinear program is formulated to solve the problem of tool loading and batching production for multi-product types on single flexible machines (SFMs) among several logical workstations. The decisions involve loading tools from each SFM's limited capacity tool maga-zine, allocating work to each SFM, and setting batch-sizes so as to minimize the total flow time. For the case of single-product type on parallel heterogeneous SFMs, the problem can be proved to be NP-hard. A relaxed flow problem and a heuristic solution are used to develop a branch-and-bound algorithm for computing the optimal solution. Based on the analysis of the optimal solution, a ratio comparison method is used to develop a heuristic procedure for obtain-ing a good solution in a short time. Computational results show that the algorithms are quite efficient. Several valuable managerial implications are derived from the ratio comparison method and its excellent performance.

Loading and batching to minimize flow times in flexible manufacturing systems

1. Introduction

A flexible manufacturing system (FMS) is an automated batch manufacturing system consisting of numerically controlled (NC) machines, linked by automated material handling devices that perform the operations required to manufacture parts (or products). Each operation requires one or more cutting tools. The tools for all operations to be performed by a machine are stored in its tool magazine which has limited capacity. Each machine has an automatic tool changing device that can interchange two cutting tools in seconds Stecke [20]. MacCarthy and Liu [18] define a single flexible machine (SFM), a more basic unit of FMS, which consists of a single NC machine with tool changing capacity, a material handling device and a part storage buffer. Due to the intractability of the FMS production planning in its general forms, hierarchical approaches are usually suggested to solve decision problems sequentially [8, 20]. Under each specific objective, the FMS logical cell formation problem may be solved by partitioning the set of multiple products to be processed into disjoint product types, and the set of SFMs into workstations, with each workstation dedicated to processing one product type to improve the FMS efficiency or resource utilization [2, 11, 14, 15]. A work-station is defined to be a group of heterogeneous SFMs and may be organized logically, rather than physically [11]. In this type of situation, the loading and the batching decisions may become the subsequent procedure for the FMS setup problem. Given multi-product types to be produced among workstations during the next period, this study considers the FMS loading and batching problem that simultaneously allocates tools to SFMs and determines each heterogeneous SFM's workload, the number of batches and the relevant batch sizes for minimizing the total flow-time.

In an FMS, the loading problem is defined as the assignment of the operations corresponding to the product types and the needed tools to versatile FMS machines, subject to the technological constraints such as tool magazine capacity, production capacity, and some loading objectives [20].

Since batch manufacturing is the norm for most (if not all) FMSs, batching consideration becomes an important factor for establishing the loading decisions. Unfortunately, such considerations are not adequately reflected in the existing loading research. The batching decisions belong to an important class of scheduling problem. Basically, one is to determine the optimal batch sizes so as to minimize flow times. For an automated system to perform well, it is also necessary to integrate tooling management and the other basic production functions, such as process planning, scheduling, part design, and part programming. The concepts and methods used in a conventional job shop are also applicable to FMSs. Most of the existing loading procedures for both conventional systems and FMSs tend to balance the assigned workload on each machine. The aim is to equalize the total processing time, or workload, of the operations assigned to each machine. As a result, each operation is often assigned to only one machine and consequently each product has a fixed route through the shop. The benefits of the flexibility and other capabilities of an FMS cannot be fully captured by a fixed route processing. Integrating batching considerations into the tool loading decision can remedy the deficiency of the fixed routing processing.

The importance of tool loading management in FMSs has gradually been recognized by both researchers and practitioners. Tool loading has been estimated to comprise about 20% of the cost of new manufacturing systems. For FMSs, the proportion may be even higher since tool loading includes high-cost fixtures and pallets [22]. Despite this high tooling cost, there has been a relative dearth of tool loading research. Zhou and Wysk [25] observed that, although the advent of FMSs has greatly increased the productivity potential of manufacturers, the management of the increased number of tool components and their application requirements have hindered increases in productivity.

Several integer programming models have been proposed to solve the FMS loading problems, but not all of them take tooling into account. When tool loading has to be handled by the model, the load assignment may have to be changed completely [24]. Some researches have proposed models integrating tool loading into production planning problems [1, 6, 9, 10, 12, 13, 19, 20, 21, 23, 24].

Difficulties arise since that an integer programming formulation is usually not the best way for developing heuristics. Also, batching considerations are absent from these models. For the batching decisions in a job shop environment, several mixed integer nonlinear programs for single-product and multi-product cases have been formulated and solved by Dobson, Karmarkar and Rummel (DKR) [3, 4] and Lee and Chung [16, 17] respectively. This study extends the batching problems on parallel heterogeneous machines to the loading and batching problem for producing the homogeneous products (or, the same product type) in the same workstation while the heterogeneous products (or, the different product type) in the other. The FMS considered here consists of a certain number of workstations and each workstation is composed of a relevant number of SFMs with known tool magazine capacity. The problem is formulated as a convex quadratic programming model for minimizing the total batch flow-time. It should be noted that although we deal with only one product type in each workstation, its demand needs to be grouped into multiple batches and allocated to each SFM with different sets of tools. For the homogeneous products, the technique developed by DKR [4] can be used to improve the flow time and it is unnecessary to incorporate the detailed DKR's procedure during the production planning stage.

This rest of the paper is organized as follows: Section 2 formally defines the problem and presents its mathematical formulation with quadratic batch flow-time; Some properties of the optimal solution and the computational complexity are discussed in Section 3; Section 4 provides lower bounds and solution procedures including two efficient optimal solution algorithms and a good heuristic algorithm; The computational results and comparisons between the heuristic solutions and the optimal solutions are reported in Section 5. Finally, Section 6 summarizes the study and suggests directions for future research.

2. Problem Formulations

In this section, we formalize the tool loading and batching problem in an FMS with multiple

workstations. Due to the physical constraints of SFMs and their tool magazines for different product types, it is prohibited to shift SFMs from one logical workstation to another in a production period. Without loss of generality, it is assumed that any workstation and its product type have the same index and that there are D_g pieces of product type g to be processed on m_g separate SFMs at workstation g. For SFM i at workstation g , i = 1,...,m_g , there are | L_{gi} | different tools ready to be selected from each tool magazine and run on each SFM. The demand quantity of product type g , D_g , is assumed to be continuously divisible and will be produced at work-station g in a production cycle. The objective of the problem is to allocate E_{gi} pieces to SFM i , i =1,...,m_g , to choose a tool type j , $j \in L_{gi}$, from the tool magazine of SFM i and then to divide each allocation into batches q_{gijb} , b = 1,..., n_{gij} , at workstation g, g =1,…, w, such that the total batch flow-time for work-in-process is minimized. From the updated tool database, the tool setup and processing data associated with an SFM can be obtained and used immediately.

The following notations will be used in formulating the problem:

g = the index for workstations or product types; g =1,..., w;

i = the index for SFMs; i =1,…, m_g , g =1,...,w;

j = the index for tool types; $j \in L_{gi}$, i =1,…, m_g , g =1,...,w;

b = the index for batch orders; b =1,…, n_{gij} , $j \in L_{gi}$, i =1,…, m_g , g =1,…,w;

w = the number of workstations or product types;

m_g = the number of SFMs at workstation g;

D_g = the demand quantity of the product type g; g =1,..., w;

L_{gi} = the index set for tool types on SFM i at workstation g;

s_{gij} = the setup time for tool type j selected from L_{gi} at workstation g;

r_{gij} = the processing rate for tool type j selected from L_{gi} at workstation g;

Decision variables for the problem are defined as follows:

E_{gi} = the amount of product types g allocated to SFM i at workstation g;

A_{gij} = the amount of work allocated to SFM i for tool type j at workstation g;

n_{gij} = the number of batches to be run on SFM i as tool type j is selected at workstation g;

q_{gijb} = the size of the b^{th} batch on SFM i as tool type j is selected at workstation g;

x_{gij} = 1, if tool type j is selected on SFM i at workstation g; 0, otherwise.

The batch-flow problem for the multiple workstations case can be formulated as follows:

$$\text{(P1)} \qquad \text{Minimize} \sum_{g=1}^{w} \sum_{i=1}^{m_g} \sum_{j \in L_{gi}} x_{gij} \{ \sum_{b=1}^{n_{gij}} (\sum_{h=1}^{b} s_{gij} + q_{gijh} / r_{gij}) q_{gijb} \} \tag{1}$$

$$\text{subject to} \quad \sum_{b=1}^{n_{gij}} q_{gijb} = A_{gij}, j \in L_{gi}, i = 1,..., m_g, g = 1,\ldots, w, \tag{2}$$

$$\sum_{j \in L_{gi}} x_{gij} A_{gij} = E_{gi}, i = 1,..., m_g, g = 1,\ldots, w, \tag{3}$$

$$\sum_{j \in L_{gi}} x_{gij} \le 1, \quad i = 1,..., m_g, g = 1,\ldots, w, \tag{4}$$

$$\sum_{i=1}^{m_g} E_{gi} = D_g, g = 1,\ldots, w, \tag{5}$$

$$E_{gi} \ge 0, i = 1,..., m_g, g = 1,\ldots, w, \tag{6}$$

$$x_{gij} A_{gij} \ge A_{gij} \ge 0, j \in L_{gi}, i = 1,..., m_g, g = 1,\ldots, w, \tag{7}$$

$$x_{gij} \in \{0, 1\}, j \in L_{gi}, i = 1,..., m_g, g = 1,\ldots, w, \tag{8}$$

$$n_{gij} \in Z^{+}, j \in L_{gi}, i = 1,..., m_g, g = 1,\ldots, w, \tag{9}$$

$$q_{gijb} \ge 0, b = 1,..., n_{gij}, j \in L_{gi}, i = 1,..., m_g, g = 1,\ldots, w, \tag{10}$$

where L_{gi}, $i = 1,..., m_g$, $g = 1,\ldots, w$, are mutually exclusive.

In model P1, the objective is to minimize the sum of the flow-time for each loaded tool on its SFM for all workstations. The first set of constraints makes sure that the sum of batch sizes for the tool type j on SFM i at workstation g equals its allocation amount. The second and third con-straint sets require that there is at most one loaded tool on each SFM and its allocation amount equals E_{gi}, $i = 1,...,$ m_g, $g = 1,\ldots, w$. Constraints (5) through (7) insure that the sum of the allocations on each SFM at workstation g equals to the total demand quantity of product type g, D_g, and $A_{gij} = 0$ when $x_{gij} = 0$, but

allows them to take on a value greater than zero when $x_{gij} = 1$.

Because each product type is only produced at its assigned logical workstation, the optimal flow time can be obtained by partitioning problem P1 into *w* subproblems (i.e., problem P2 described in Section 3) corresponding to *w* product types and then solving each subproblem for its assigned product type iteratively. We will discuss the loading and batching problem for single-product type in the subsequent sections and the computational results for multi-product types in Section 5.

3. Computational Complexity

In this section, we analyze the computational complexity of problem P1. First, suppressing the workstation subscript *g*, we derive a problem P2 equivalent to the single-product case of P1 and then show that P2 is NP-hard. Some lemmas pertaining to problem P2 will be derived. For any tool loading problem, these results can be used to calculate the batch flow-time with ease. We here define $\sigma(\bullet)$ as a tooling assignment function on a particular SFM; $\sigma(i)$ represents the assignment index of the tool loading on SFM *i*. Also, $R_\sigma = (r_{1\sigma(1)}, r_{2\sigma(2)},...,r_{m\sigma(m)})$ denotes the set of processing rates, $S_\sigma = (s_{1\sigma(1)}, s_{2\sigma(2)}, ...,s_{m\sigma(m)})$ the set of setup times, and $f_b(m, R_\sigma, S_\sigma, D)$ the function of the optimal flow time for processing a total amount of work *D* at *m* SFMs with setup times $s_{1\sigma(1)}, s_{2\sigma(2)},..., s_{m\sigma(m)}$, and processing rates $r_{1\sigma(1)}, r_{2\sigma(2)},...,r_{m\sigma(m)}$. It is obvious that the following problem is equivalent to the case of single-product type of P1.

(P2) $$\text{Minimize} \sum_{i=1}^{m}\sum_{j\in L_i}\sum_{b=1}^{n_{ij}}\left(\sum_{h=1}^{b} s_{ij} + q_{ijh}/r_{ij}\right)q_{ijb} \tag{11}$$

subject to $$\sum_{b=1}^{n_{ij}} q_{ijb} = A_{ij}, \quad j\in L_i,\ i = 1,..., m, \tag{12}$$

$$\sum_{j\in L_i} A_{ij} = E_i, \quad i = 1, ..., m, \tag{13}$$

$$\sum_{i=1}^{m} E_i = D, \tag{14}$$

$$E_i \geq 0\ ,\ i = 1, 2,..., m, \tag{15}$$

$$|\ M_i\ | \leq 1\ ,\ M_i = \{\ j\ |\ \exists\ A_{ij} > 0\ ,\ j \in L_i\ \},\ i = 1, 2, ..., m \tag{16}$$

$$A_{ij} \geq 0\ ,\ j \in L_i\ ,\ i = 1, ..., m, \tag{17}$$

$$n_{ij} \in Z^+, j \in L_i\ ,\ i = 1, ..., m, \tag{18}$$

$$q_{ijb} \geq 0,\ b = 1,..,\ n_{ij}\ ,\ j \in L_i\ ,\ i = 1, ..., m, \tag{19}$$

where L_i , i = 1,...,m, are mutually exclusive.

Once the tooling assignment problem is solved, we may obtain the number of active SFMs and the multiplier value using the same procedure given by DKR [3]. Lemma 2 presents the calculation of the total batch flow-time in a simple form. Without loss of generality, we assume that $s_{1\sigma(1)} \leq\ s_{2\sigma(2)}\ \leq ... \leq s_{m\sigma(m)}$.

Lemma 1. P2 is equivalent to the case of single-product type of P1.

Proof. See Appendix.

Lemma 2. For any tooling assignment, σ , if p ($\leqq m$) SFMs are active and its multiplier value is λ_σ , then $f_b(m, R_\sigma, S_\sigma, D) = \sum_{i=1}^{p}(n_{i\sigma(i)} + 1)\{r_{i\sigma(i)} n_{i\sigma(i)}[\lambda_\sigma^2 / (n_{i\sigma(i)} + 1)^2 - s_{i\sigma(i)}^2 (n_{i\sigma(i)} + 2)/12]\}/2$

$= \sum_{i=1}^{p}(n_{i\sigma(i)} + 1)\{r_{i\sigma(i)} s_{i\sigma(i)}^2 n_{i\sigma(i)} (1 - n_{i\sigma(i)})/12 + E_i[s_{i\sigma(i)} + E_i / (n_{i\sigma(i)}\ r_{i\sigma(i)})]\}/2$, where $n_{i\sigma(i)} = \lfloor [1/4 + 2E_i/(s_{i\sigma(i)} r_{i\sigma(i)})]^{1/2} + 1/2 \rfloor = \lfloor [1/4 + 2\lambda_\sigma / s_{i\sigma(i)}]^{1/2} - 1/2 \rfloor$, $\lambda_\sigma = E_i(1 + 1/n_{i\sigma(i)})/r_{i\sigma(i)} +$

$s_{i\sigma(i)}(n_{i\sigma(i)} + 1)/2$, and $E_i = r_{i\sigma(i)}[\lambda_\sigma\ n_{i\sigma(i)} / (n_{i\sigma(i)} + 1) - n_{i\sigma(i)} s_{i\sigma(i)}/2]$, $i = 1, 2, \ldots, p$.

Proof. See Appendix.

Corollary 1. For any tooling assignment, σ , if it contains m active SFMs with the multiplier value equal to λ_σ and $s_{i\sigma(i)} r_{i\sigma(i)} > D$, i=1,..., m, then $f_b(m, R_\sigma, S_\sigma, D) = \sum_{i=1}^{m} E_i(s_{i\sigma(i)} + E_i / r_{i\sigma(i)}) =$

$\sum_{i=1}^{m} r_{i\sigma(i)}(\lambda_\sigma^2 - s_{i\sigma(i)}^2)/4$, where $\lambda_\sigma = (2D + \sum_{i=1}^{m} s_{i\sigma(i)} r_{i\sigma(i)}) \Big/ (\sum_{i=1}^{m} r_{i\sigma(i)})$, and $E_i = r_{i\sigma(i)}(\lambda_\sigma - s_{i\sigma(i)})/2$,

i =1, ..., m.

Proof. See Appendix.

For a two-tool system with $2m$ single-tool SFMs, we have $|L_i| = 2$, $i = 1,..., m$, the setup time s_{11}, s_{12}, s_{21}, s_{22},..., s_{m1}, s_{m2} and the associated processing rates r_{11}, r_{12}, r_{21}, r_{22},..., r_{m1}, r_{m2}. To facilitate our discussion, we define the following notations: $S = (s_{11}, s_{12}, ..., s_{m1}, s_{m2})$, $R = (r_{11}, r_{12}, ..., r_{m1}, r_{m2})$, $S_\sigma = (s_{1\sigma(1)}, ..., s_{m\sigma(m)})$, $R_\sigma = (r_{1\sigma(1)}, ..., r_{m\sigma(m)})$ and $S_\delta = (s_{1\delta(1)}, ..., s_{m\delta(m)})$, $R_\delta = (r_{1\delta(1)}, ..., r_{m\delta(m)})$. And, we note that $\{\sigma(i), \delta(i)\} = \{1, 2\}$, $i = 1, 2, ..., m$.

Lemma 3. For a two-tool case, given any amount of work D such that $2D < s_{ij} r_{ij}$ and $s_{ij}^h >> s_{ij}^l >> D$ whenever $h > l$, $h, l \in Z^+$, $j = 1, 2$, $i = 1, 2, ..., m$. Assume that all SFMs associated with $f_b(2m, R, S, 2D)$ and $f_b(m, R_\sigma, S_\sigma, D)$ are active. Then, $f_b(2m, R, S, 2D) = 2f_b(m, R_\sigma, S_\sigma, D)$ if and only if $\sum_{i=1}^{m} s_{i\sigma(i)}^{2-h} r_{i\sigma(i)} = \sum_{i=1}^{m} s_{i\delta(i)}^{2-h} r_{i\delta(i)}$, $h = 0, 1, 2$.

Proof. See Appendix.

Theorem 1. P2 is NP-hard.

Proof. See Appendix.

Since P2 is NP-hard, it is desirable to find the associated batch-flow relaxation problem on the binary searching tree which in turn can be solved by the DKR algorithm [3]. In the next section, the basic rationale will be discussed in developing two efficient optimal solution algorithms and a heuristic procedure.

4. Algorithms for Solving P2: Optimal Solution & A Heuristic Procedure

From the analysis above, it is obvious that problem P2 can be divided into $\prod_{i=1}^{m} |L_i|$ smaller problems. Suppose $\overline{A}_{ij}$ is the amount of work allocated to SFM i as tool type j is loaded. The resulting subproblem for tool type j on SFM i is:

(P3) $$f_{ij}(\overline{A}_{ij}) \equiv \text{Minimize} \sum_{b=1}^{n_{ij}} \left(\sum_{h=1}^{b} s_{ij} + q_{ijh}/r_{ij}\right) q_{ijb} \tag{20}$$

subject to $$\sum_{b=1}^{n_{ij}} q_{ijb} = \overline{A}_{ij}, \tag{21}$$

$$n_{ij} \in Z^{+}, j \in L_i, i = 1, \ldots, m, \tag{22}$$

$$q_{ijb} \geq 0, b = 1, \ldots, n_{ij.}. \tag{23}$$

Without Constraint (14), problem P2 can be solved by the DKR algorithm. Since the convex quadratic programming relaxation problem can be solved first as an allocation problem and then as a batching problem, the original problem P2 can be rewritten as:

(P4) Minimize $$\sum_{i=1}^{m} \sum_{j \in L_i} f_{ij}(A_{ij}), \tag{24}$$

subject to $$\sum_{j \in L_i} A_{ij} = E_i, i = 1, ..., m, \tag{25}$$

$$\sum_{i=1}^{m} E_i = D, \tag{26}$$

$$|M_i| \leq 1, M_i = \{j | \exists A_{ij} > 0, j \in L_i\}, i = 1, \ldots, m, \tag{27}$$

$$E_i \geq 0, i = 1, \ldots, m, \tag{28}$$

$$A_{ij} \geq 0, j \in L_i, i = 1, \ldots, m, \tag{29}$$

where L_i, $i = 1, 2, ..., m$, are mutually exclusive.

Removing constraint (27) from P4, we can easily implement the allocation procedure of the relaxation problem, and then test the feasibility of constraint (27) at the active SFMs. If the allocation procedure shows that all active tool types are feasible for P4, then it will proceed to calculate the multiplier value and each batch size. For P3, a solution in the closed form has been developed by DKR [4]. Based on the same logic, we develop two optimal solution algorithms and a heuristic procedure. The extensive numerical experiments seem to suggest that the performance of the depth first search combined with a branch-and-bound algorithm on the binary searching tree is most efficient. A ratio comparison method is adopted in the heuristic algorithm. By testing the ratio of the setup time raised to the fifth power over the processing rate between idle tool and active tool on each SFM, the solution can be improved by approximately 1/10 of the heuristic objective value for a small size problem. Theoretically, a tighter lower bounding function will always improve the efficiency of a

branch-and-bound algorithm. We take advantage of this property to facilitate the search for the optimal solution on the binary searching tree. The following rules are used in selecting variable A_{ij} for branching SFM i where $|M_i| > 1$:

(a) Choose SFM i^* for branching such that $|M_{i^*}| = \max_i \{|M_i| \,||M_i| > 1\}$ (30)

(b) Choose SFM i^* for branching such that $|M_{i^*}| = \min_i \{|M_i| \,||M_i| > 1\}$ (31)

where M_i ($\subseteq L_i$), $i = 1,..., m$, is the index set of variables, A_{ij} , which are not yet fixed at the current node. Two algorithms BES1 and BES2 implement the depth first search combined with the branch-and-bound algorithms, each of which adopts (30) and (31) respectively.

If the variable set M_{i^*} is selected, then the variable $A_{i^*j^*}$ in M_{i^*} , with the following property are used for branching: $\overline{A}_{i^*j^*} = \max_{j \in M_{i^*}} \{\overline{A}_{i^*j}\}$, (32)

where $\overline{A}_{i^*j}$ can be obtained from the following equations:

$$n_{i^*j} = \left\lfloor (1/4 + 2\hat{\lambda}/s_{i^*j})^{1/2} - 1/2 \right\rfloor \text{ , and } \overline{A}_{i^*j} = r_{i^*j}[\hat{\lambda}\, n_{i^*j}/(n_{i^*j} + 1) - n_{i^*j}\, s_{i^*j}/2], \quad j \in M_{i^*}.$$

The approximation of the multiplier value, $\hat{\lambda}$, is obtained using the same procedure as presented in DKR[3]. We set $A_{i^*j} = 0$ for $j \neq j^*$ in the left child of the current candidate node and $A_{i^*j^*} = 0$ in the right child. It should be noted that if the current relaxation problem is infeasible, the true value of the multiplier will not be calculated. The notations used in the development of the algorithms are listed as follows.

P_o = initial optimization problem,

F_o = the current least upper bound of the objective,

λ_o = the multiplier value for producing the assigned product type corresponding to F_o ,

T_o = the index set of the tools selected corresponding to F_o ,

T_{o_i} = the index of tool selected on SFM i corresponding to T_o ,

P_c = the current candidate problem being explored,

F_c = the optimal batch flow-time corresponding to P_c ,

λ_c = the multiplier value for producing the assigned product type corresponding to F_c ,

T_c = the index set of the tools selected corresponding to F_c ,

T_{c_i} = the index of tool selected on SFM i corresponding to T_c ,

W_{c_j} = the index of SFM which selects tool j corresponding to F_c ,

N_c = the index set of the current candidate tools on searching tree,

R_c= the set of processing rates corresponding to N_c ,

S_c = the set of setup times corresponding to N_c ,

$PARENT(P_c)$ = the parent of P_c ,

$CHILD(P_c)_I$ = the I^{th} child of P_c , I = 1, 2 ,

$(P_c)_R$ = the relaxation problem of P_c , obtained by removing constraint (16),

(i, j) = the assignment for tool j selected on SFM i,

N = the index set of all assignments,

IWC = the index of the current candidate SFM to be assignd,

F_L = the lower bound value for the current candidate problem,

F_H = the batch flow-time for the initial heuristic solution,

λ_H = the multiplier value for producing the assigned product type corresponding to F_H ,

T_H = the index set of tools selected corresponding to F_H ,

T_{H_i} = the index of tool selected on SFM i corresponding to T_H ,

W_{H_j} = the index of SFM which selects tool j corresponding to F_H ,

F_p = the previous batch flow-time before improving the heuristic objective value.

We will state the lower bound function and the fathoming rule in Lemma 4 and Lemma 5.

Lemma 4. If tool type j and tool type h are available on SFM i such that $s_{ij} \le s_{ih}$, $r_{ij} > r_{ih}$, j , $h \in L_i$, i =1, ..., m , then the optimal solution will not choose tool type h on SFM i .

Proof. See Appendix.

Lemma 5. In the current candidate problem, define $S_L = \{S_{L_i} \mid i = 1,...,m\}$, $R_L = \{R_{L_i} \mid i = 1,...,m\}$.

If $S_{L_i} = \min_{j \in L_i}\{s_{ij}\}$, $i = 1,..., m$, and $R_{L_i} = \max_{j \in L_i}\{r_{ij}\}$, $i = 1,..., m$, then $f_b(m, R_L, S_L, D) \leq F_c$.

Proof. See Appendix.

Using Lemma 5 in the optimal solution algorithms *BES1* and *BES2* respectively, we are able to reduce the computing time by about 25% in obtaining the optimal solution for the small size problem.

A Lower Bound

Suppose that the optimal objective value of P2 equals F^* and that the upper limit of the searching depth is set to equal l. Because P2 is NP-hard, we may not be able to obtain the optimal solution in some cases as the depth of the binary searching tree reaches l. In this situation, we define the candidate problem P^l_c whose depth equals l and its relaxation problem is infeasible. The set of all P^l_c will be V_l. If the optimal solution algorithm cannot yield the solution at the depth l of searching tree, then Lemma 5 clearly implies that $\min_{P^l_c \in V_l} f_b(|N_c|, R_c, S_c, D) \leq F^*$.

In the following algorithms, all input data are preprocessed such that if $s_{ij} < s_{ih}$, then $r_{ij} < r_{ih}$, $j, h \in L_i$, $i = 1,...,m$.

Algorithm BHS

Step 1. (Initialization) Set $N_c = \bigcup_{i=1}^{m} L_i$ and $D_H = |N_c| D/m$. Find R_c, S_c, $f_b(|N_c|, R_c, S_c, D_H)$ and $\overline{A}_{ij}$, $j \in L_i$, $i = 1,\ldots, m$. Go to Step 2.

Step 2. (Heuristic assignment) For $i = 1,..., m$, find T_{H_i} such that $\overline{A}_{ij^*} = \max_{j \in L_i}\{\overline{A}_{ij} \mid j \in L_i, \overline{A}_{ij} > 0\}$ and $T_{H_i} = \{j^*\}$. If $T_{H_i} \neq \phi$, go to Step 3. Otherwise, find $s_{ij^*} = \min_{j \in L_i}\{s_{ij}\}$, set $T_{H_i} = \{j^*\}$ and go to Step 3.

Step 3. (Calculation) Set $N_c = \bigcup_{i=1}^{m} T_{H_i}$. Find R_c , S_c , $f_b(|N_c|$, R_c , S_c , $D)$, F_H and λ_H . Set $F_p = F_H$, $IWC = 1$. Go to Step 4.

Step 4. (Iterative improvement)

If $IWC > m$ and $F_p = F_H$, go to Step 8.

If $IWC > m$ and $F_p > F_H$, set $IWC = 1$ and $F_p = F_H$. Go to Step 5.

If $IWC \le m$, go to Step 5.

Step 5. (Current candidate tool) Find J^* such that $\{J^*\} = T_{H_{IWC}}$. If $\{J^*\} \neq \phi$, go to Step 6. Otherwise, set $IWC = IWC + 1$ and go to Step 4.

Step 6. (Ratio comparison test)

Find J^{**} such that $Q_{J^{**}} = \max_J \{Q_J | \ Q_J = [s_{IWC, J^*} / s_{IWC, J}]^5 [r_{IWC, J} / r_{IWC, J^*}] > 1\}$. Set $T_{c_{IWC}} = \{J^{**}\}$.

If $T_{c_{IWC}} = \phi$, set $IWC = IWC + 1$ and go to Step 4. Otherwise, set $T_{c_i} = T_{H_i}$, $i = 1,\ldots,m$, $i \neq IWC$. Go to Step 7.

Step 7. (Calculation, F_c) Set $N_c = \bigcup_{i=1}^{m} T_{c_i}$. Find R_c , S_c , f_b ($| N_c |$, R_c , S_c , D), F_c and λ_c . If $F_c \ge F_H$, set $IWC = IWC + 1$ and go to Step 4. Otherwise, set $F_H = F_c$, $\lambda_H = \lambda_c$, $T_{H_{IWC}} \quad T_{c_{IWC}}$, and $IWC = IWC + 1$. Go to Step 4.

Step 8. (Termination) Output the heuristic solution, F_H , λ_H and T_{H_i} , $i = 1, 2,..., m$.

Algorithm BES

Step 1. (Initialization) Use Algorithm *BHS* to find F_H , λ_H and T_H . Set $F_o = F_H$, $\lambda_o = \lambda_H$, $T_o = T_H$ and $P_c = P_o$. Go to Step 2.

Step 2. (Solving the allocation relaxation) Find active tool types for $(P_c)_R$. If the allocation is feasible for P_c , go to Step 5. Otherwise, go to Step 3.

Step 3. (Lower bound test) Find F_L . If $F_L \ge F_o$, go to Step 6. Otherwise, go to Step 4.

Step 4. (DFS-search, Branching) Find $CHILD(P_c)_I$, I = 1, 2. Choose an SFM for branching using search strategy (30) or (31) and generate $CHILD(P_c)_I$, I = 1, 2, using the branching rule (32). Set $P_c = CHILD(P_c)_1$, go to Step 2.

Step 5. (Calculation) Find F_c , T_c and λ_c . If $F_c \geq F_o$, go to Step 6. Otherwise, set $F_o = F_c$, $T_o = T_c$ and $\lambda_o = \lambda_c$. Go to Step 6.

Step 6. (Backtracking) Set $P_c = PARENT(P_c)$. If we have searched all children of P_c , go to Step 7. Otherwise, go to Step 8.

Step 7. If $P_c = P_o$, go to Step 9. Otherwise, go to Step 6.

Step 8. Set $P_c = CHILD(P_c)_2$, go to Step 2.

Step 9. (Termination) Output the optimal solution, F_o , λ_o , T_{o_i} , i = 1,..., m and each SFM's batch number and its relevant batch sizes.

5. Computational Results

We conduct several experiments to evaluate the effectiveness of the heuristic and to compare the heuristic solution with the optimal solution and lower bound over different ranges of problem parameters. Since problem P2 is NP-hard, it is rather difficult, if not impossible, to solve the large size problem in a reasonably short time. We here present some sample computational results of these different algorithms. When we encounter the worst case in the total branch-and-bound enumeration, we may expect that $\sum_{g=1}^{w}\prod_{i=1}^{m}|L_{gi}|$ number of operations is required to solve the batch-flow relaxation problem. Algorithms BES1 and BES2, which adopt branching rules (30) and (31) respectively, are developed to obtain the optimal solution. BHS adopts the heuristic assignment and an iterative ratio comparison method to find a satisfactory solution. In order to evaluate the proposed algorithms for solving the loading and batching problem, we generate a total of 77 problems which are divided into two separate sets. The first set of 53 problems are solved by the following algorithm combinations: two

efficient optimal solution algorithms, BES1 and BES2, and the heuristic algorithm BHS. The second set of 24 problems are solved by algorithms BHS and a lower bound. The lower bound is produced by fixing the level on the searching tree of BES1. Test data were randomly generated using the following distribution parameters. Setup times (s_{ij}) are uniformly distributed between 5.0 and 40.0. Process-ing rates (r_{ij}) are uniformly distributed between 5.0 and 50.0. A normal distribution $N(k\mu, k\sigma)$ is assumed for the demand for each product type, where k is a positive number, μ = 1,000 and σ = 200. The rest of the data are shown in Tables 1 and 2. Average CPU seconds required for each problem by the optimal solution algorithm, and heuristic algorithm are also included.

< Insert Tables 1 - 2 here>

Table 1 shows that BES1 performs better than BES2 while the heuristic algorithm, BHS, does provide a good solution. In all the test problems, BHS performs extremely well. In case of large size problems, a lower bound is used to evaluate the heuristic solutions and the computational results are shown in Table 2. For the small and medium size problems, the optimal solutions of P2 can be easily obtained by algorithm BES1. When the size of P2 becomes large, BHS can produce a good solution in a short time. Analyzing the solution structure of problem P2, we find that the ratio comparison method is quite useful. It helps obtain the optimal solution quickly and all SFMs' are almost active by their flexible tool magazines in the solution. The same conclusion can be drawn for the lower bound. Moreover, some implications for the tool loading and batching decisions can be derived from the ratio comparison method and its computational results. It is demonstrated that each tool's setup time raised to the fifth power over its processing rates can determine tool loading and then the batch-sizes setting on each SFM.

6. Conclusions

Existing FMS research does not incorporate batching consideration into the loading decision. In this paper, we study the problem of tool loading and batch sizing on multiple single flexible machines (SFMs) among workstations with the objective of minimizing the total flow time. The problem is

proved to be NP-hard. By using the IBM-3090 supercomputer, we were able to obtain the optimal solutions for small size problems. These optimal solutions provide the basis for analyzing the optimal decisions and for evaluating the performance of the heuristic procedure. Also, the ratio comparison provides an efficient method for obtaining the heuristic solution and some implications for the tool loading and batch sizing decisions. These results may be applied to other similar production environments.

Further studies should be directed to the integration of the tool loading and batching problem with other system setup subproblems such as machine grouping problem, production ratio problem, resource allocation problem, etc. In terms of the solution procedure, hierarchical, iterative and simultaneous approaches are suggested for such an integration effort. The heuristic algorithms developed in this paper provide a good foundation for these future studies.

Appendix

Proof of Lemma 1. The only difference between the single-product type case of P1 and P2 lies in that (15) replaces (4) on SFM i. So each constraint in the proposed case of P1 can be transferred into the corresponding constraint of P2. This completes the proof.

Proof of Lemma 2. As any specific tooling assignment σ incurs, P2 reduces to the batching problem. In this case, $A_{i\sigma(i)} = E_i$, and $A_{ij} = 0$, $j \neq \sigma(i)$, $j \in L_i$, $i \leq p$. Define $g_b(r_{i\sigma(i)}, s_{i\sigma(i)}, E_i)$ as the flowtime incurs on SFM i. Then, $f_b(m, R_\sigma, S_\sigma, D) \equiv \sum_{i=1}^{p} g_b(r_{i\sigma(i)}, s_{i\sigma(i)}, E_i)$. Using the results of DKR's closed form [3], we obtain the relationship between $n_{i\sigma(i)}$ and $q_{i\sigma(i)b}$, $b=1,..., n_{i\sigma(i)}$, ($q_{i\sigma(i)b} > 0$) for the optimal solution. Suppressing the subscript, $i\sigma(i)$, it becomes the following expression: $q_b = E_i/n + sr(n+1)/2 - bsr$, for $q_b > 0$, $b = 1,.., n$. By incorporating this expression into the objective function for SFM i , $g_b(r, s, E_i)$ becomes a function of n , s , r , and E_i . Then, we have $g_b(r, s, E_i) = \sum_{b=1}^{M} q_b \sum_{h=1}^{b} (s + q_h/r) = s\sum_{b=1}^{n} bq_b + [\sum_{b=1}^{n} q_b \sum_{h=1}^{b} q_h]/r$, where $1 \leq n \leq M$, $q_b > 0$, $b = 1, ..., n$,

and M is an upper bound on the total number of batches (DKR) [4]. Let $A = E_i/n + sr(n+1)/2$ and $B = sr$. Then, $g_b(r, s, E_i) = n(n+1)[sA/2 - (2n+1)sB/6 + sA^2/(2B) - sA/2 + sB/12 + n(7sB/24 - sA/2) + sn^2B/8] = (r/24)[-s^2 n^3 + (12 E_i s/r + s^2)n + (12 E_i^2/r^2 + 12 E_i s/r) + 12 E_i^2/(r^2 n)] = (n+1)\{r s^2 n(1-n)/12 + E_i [s + E_i/(nr)]\}/2$. Since $E_i = r[\lambda_\sigma n/(n+1) - sn/2]$,

$g_b(r, s, E_i) = (n+1)\{r n[\lambda_\sigma^2/(n+1)^2 - s^2(n+2)/12]\}/2$. By definition, we have $f_b(m, R_\sigma, S_\sigma, D) \equiv \sum_{i=1}^{p} g_b(r_{i\sigma(i)}, s_{i\sigma(i)}, E_i)$. Hence, $f_b(m, R_\sigma, S_\sigma, D) = \sum_{i=1}^{p} (n_{i\sigma(i)} + 1)\{r_{i\sigma(i)} s_{i\sigma(i)}^2 n_{i\sigma(i)} (1 - n_{i\sigma(i)})/12 + E_i [s_{i\sigma(i)} + E_i/(n_{i\sigma(i)} r_{i\sigma(i)})]\}/2 = \sum_{i=1}^{p} (n_{i\sigma(i)} + 1)\{r_{i\sigma(i)} n_{i\sigma(i)} [\lambda_\sigma^2/(n_{i\sigma(i)} + 1)^2 - s_{i\sigma(i)}^2 (n_{i\sigma(i)} + 2)/12]\}/2$,

where $n_{i\sigma(i)} = \lfloor [1/4 + 2E_i/(s_{i\sigma(i)} r_{i\sigma(i)})]^{1/2} + 1/2 \rfloor = \lfloor [1/4 + 2\lambda_\sigma/s_{i\sigma(i)}]^{1/2} - 1/2 \rfloor$, $\lambda_\sigma = E_i(1 + 1/n_{i\sigma(i)})/r_{i\sigma(i)} + s_{i\sigma(i)}(n_{i\sigma(i)} + 1)/2$, and $E_i = r_{i\sigma(i)}[\lambda_\sigma n_{i\sigma(i)}/(n_{i\sigma(i)} + 1) - n_{i\sigma(i)} s_{i\sigma(i)}/2]$,

$i = 1, 2, \ldots, p$. This completes the proof.

Proof of Corollary 1. From Lemma 2 and the fact that $s_{i\sigma(i)} > r_{i\sigma(i)} > D$, it is obvious that $n_{i\sigma(i)} = 1, i = 1, \ldots, m$. Hence, $f_b(m, R_\sigma, S_\sigma, D) = \sum_{i=1}^{m} E_i (s_{i\sigma(i)} + E_i/r_{i\sigma(i)}) = \sum_{i=1}^{m} r_{i\sigma(i)} (\lambda_\sigma^2 - s_{i\sigma(i)}^2)/4$, where $\lambda_\sigma = 2E_i/r_{i\sigma(i)} + s_{i\sigma(i)}$, $E_i = r_{i\sigma(i)}(\lambda_\sigma - s_{i\sigma(i)})/2$, $i = 1, \ldots, m$. Also, $\lambda_\sigma \cdot r_{i\sigma(i)} = 2E_i + s_{i\sigma(i)} r_{i\sigma(i)}$.

Summing both sides of the preceding equation from $i = 1$ to m, we obtain the following expression.

$\lambda_\sigma = (2D + \sum_{i=1}^{m} s_{i\sigma(i)} r_{i\sigma(i)}) \Big/ (\sum_{i=1}^{m} r_{i\sigma(i)})$. This completes the proof.

Proof of Lemma 3. In the case of $2m$ single-tool SFMs, because of the condition that $2D < s_{ij} r_{ij}$, we can obtain the batch flow-time by means of Corollary 1. Hence, $f_b(2m, R, S, 2D) = \sum_{i=1}^{m} \sum_{j=1}^{2} A_{ij} s_{ij} + A_{ij}^2/r_{ij} = \sum_{i=1}^{m} \sum_{j=1}^{2} (\lambda^2 - s_{ij}^2) r_{i\sigma(i)}/4$, in which A_{ij} is the amount allocated to the $(2(i-1) + j)^{th}$ SFM for the new system. Since all SFMs are active, $A_{ij} > 0$, $i = 1, \ldots, m$, $j = 1, 2$,

$\lambda = [4D + \sum_{i=1}^{m} \sum_{j=1}^{2} s_{ij} r_{ij}] \Big/ [\sum_{i=1}^{m} \sum_{j=1}^{2} r_{ij}]$. For simplicity, define $W \equiv \sum_{i=1}^{m} \sum_{j=1}^{2} r_{ij}$, $T \equiv \sum_{i=1}^{m} \sum_{j=1}^{2} s_{ij} r_{ij}$, and

$Q \equiv \sum_{i=1}^{m}\sum_{j=1}^{2} s_{ij}^2 r_{ij}$. We then have $Q>>T>>W$, $\lambda = (4D+T)/W$ and $f_b(2m,R,S,2D) = 4D^2/W + 2TD/W + T^2/(4W) - Q/4$. Similarly, define $W_\sigma \equiv \sum_{i=1}^{m} r_{i\sigma(i)}$ $T_\sigma \equiv \sum_{i=1}^{m} s_{i\sigma(i)} r_{i\sigma(i)}$, and $Q_\sigma \equiv \sum_{i=1}^{m} s_{i\sigma(i)}^2 r_{i\sigma(i)}$. It follows that $Q_\sigma >> T_\sigma >> W_\sigma$, $\lambda_\sigma = (2D+T_\sigma)/W_\sigma$ and $f_b(m,R_\sigma, S_\sigma, D) = D^2/W_\sigma + DT_\sigma/W_\sigma + [T_\sigma^2/W_\sigma - Q_\sigma]/4$. From the algebraic fundamentals ([7, p.218]), and the conditions that $Q >> T >> W$, D, and $Q_\sigma >> T_\sigma >> W_\sigma$, D, we can prove the Lemma through the following deduction: $f_b(2m,R, S,2D) = 2f_b(m,R_\sigma, S_\sigma, D)$, if and only if $4/W = 2/W_\sigma$, $2T/W = 2T_\sigma/W_\sigma$, and $(T^2/W - Q)/4 = 2(T_\sigma^2/W_\sigma - Q_\sigma)/4$, if and only if $W_\sigma = W/2$, $T_\sigma = T/2$ and $Q_\sigma = Q/2$, if an only if $\sum_{i=1}^{m} s_{i\sigma(i)}^{2-h} r_{i\sigma(i)} = \sum_{i=1}^{m} s_{i\delta(i)}^{2-h} r_{i\delta(i)}$, $h = 0, 1, 2$. Q.E.D.

Proof of Theorem 1. We will show that an NP-hard problem is reducible to P2 in the case where $2D < s_{ij} r_{ij}$, $j = 1, 2$, $i = 1$ „..., m. Based on the original conditions, we have $\{\sigma(i), \delta(i)\} = \{1, 2\}$, $i = 1, ..., m$. Define a problem DP: Given positive numbers D, s_{ij} and r_{ij}, such that $s_{ij} r_{ij} > 2D$, $j = 1, 2$, $i = 1, ..., m$, and $f_b(2m,R,S, 2D)$. Question: Does there exist an assignment, σ, such that $f_b(2m,R, S,2D) = 2f_b(m, R_\sigma, S_\sigma, D)$? Clearly, if problem DP cannot be solved in deterministic polynomial time, neither can problem P2. From Lemma 3, DP is equivalent to finding an σ such that $\sum_{i=1}^{m} s_{\sigma(i)}^{2-h} r_{i\sigma(i)} = \sum_{i=1}^{m} s_{i\delta(i)}^{2-h} r_{i\delta(i)}$, h=0, 1, 2, where $\sigma(i) \in \{1,2\} \setminus \{\delta(i)\}$, i=1,…, m. Suppose $s_{i\sigma(i)}^{2-h} r_{i\sigma(i)}, s_{i\delta(i)}^{2-h} r_{i\delta(i)} << M$, $h = 0, 1, 2$. Then, we can construct a weighted set problem as follows. Question: Does there exist an assignment, σ, such that $\sum_{i=1}^{m}\sum_{h=0}^{2} r_{i\sigma(i)} (s_{i\sigma(i)} M)^{2-h} = \sum_{i=1}^{m}\sum_{h=0}^{2} r_{i\delta(i)} (s_{i\delta(i)} M)^{2-h}$? This problem has been proved to be NP-hard ([5, p.223]). Hence, P2 is NP-hard.

Proof of Lemma 4. The conclusion follows directly from the concept of dominant tool loading. Suppose that tool type h was selected on SFM i in our optimal solution. Then, it would contradict the fact that the objective would be improved should we exchange tool type j with tool type h.

Proof of Lemma 5. From Lemma 4 and the concept of dominant tool loading, the result follows

immediately.

References

[1] Berrada, M. and Stecke, K. E. (1984) A branch and bound approach for FMS machine loading. *Proceedings of the First ORSA/TIMS Conference on FMS,* University of Michigan, Ann Arbor, 256-271.

[2] Chan, D.Y. and Bedworth, D. D. (1990) Design of a scheduling system for flexible manufacturing cells. *International Journal of Production Research*, **28** , 2037-2049.

[3] Dobson, G., Karmarkar, U. S. and Rummel, J. L. (1989) Batching to minimize flow times on parallel heterogeneous machines. *Management Science*, **35**, 607-613.

[4] Dobson, G., Karmarkar, U. S. and Rummel, J. L. (1987) Batching to minimize flow times on one machine. *Management Science*, **33**,784-799.

[5] Garey, M. R. and Johnson, D. S. (1979) *Computers and Intractability-A Guide to the Theory of NP Completeness,* Bell Laboratories, Murray Hill, New Jersey.

[6] Hsu, V. N. and de Matta, R. (1997) An efficient heuristic approach to recognize the infeasibility of a loading problem. *International Journal of Flexible Manufacturing Systems*, **9**, 31-50.

[7] Jacobson, N. (1974) *Basic Algebra I* , W. H. Freeman and Company.

[8] Jaikumar, R. and Van Wassenhove, L. N.(1989) A production planning framework for flexible manufacturing systems. *Journal of Manufacturing and Operations Management*, **2**, 52-79.

[9] Kim, Y. D. and Yano, C. A. (1993) Heuristic approach for loading problems in flexible manufacturing systems. *IIE Transactions*, **25**, 26-39.

[10] Kim, Y. D. and Yano, C. A. (1994) New branch and bound algorithms for loading problems in flexible manufacturing systems. *International Journal of Flexible Manufacturing*

Systems, **6**, 361-382.

[11] Kochikar, V. P. and Narendran, T.T. (1998) Logical cell formation in FMS, using flexibility-based criteria. *International Journal of Flexible Manufacturing systems*, **10**, 163-181.

[12] Kouvelis, P. and Lee, H. L. (1991) Block angular structures and the loading problem in flexible manufacturing systems. *Operations Research*, **39**, 666-676.

[13] Kuhn, H. (1995) A heuristic algorithm for the loading problem in flexible manufacturing systems. *International Journal of Flexible Manufacturing systems*, **7**, 229-254.

[14] Kumar, R., Kusiak, A. and Vanelli, A. (1986) Grouping parts and components in flexible manufacturing systems. *European Journal Operational Research*, **24**,387-397.

[15] Kusiak, A. (1985) The parts families problem in FMS. *Annals of Operations Research,* **3**,279-284.

[16] Lee, J. M. and Chung, C.H. (1998) Batching multiple products on parallel heterogeneous machines in a closed job shop, *International Journal of Production Research*, **36**, 2793-2811.

[17] Lee, J. M. and Chung, C.H. (1993) A generalized model for batching multiple products on parallel heterogeneous machines", Working Paper, College of Business and Economics, University of Kentucky.

[18] Maccarthy, B. L. and Liu, J. (1993) A new classification scheme for flexible manufacturing system. *International Journal of Production Research*, **31**, 299-309.

[19] Sarin, S.C. and Chen, C.S. (1987) The machine loading and tool allocation problem in a flexible manufacturing system. *International Journal of Production Research*, **25**, 1081-1094.

[20] Stecke, K. E. (1983) Formulation and solution of nonlinear integer production planning problems for flexible manufacturing systems. *Management Science*, **29**, 273-288.

[21] Tang, C. S. and Denardo, E. V. (1988) Models arising from a flexible manufacturing

machine: part I: minimization of the number of tool switches. part II: minimization of the number of switching instants. *Operations Research*, **36**,767-784.

[22] Tomek, P. (1986) Tooling strategies related to FMS management. *FMS Magazine*, **4***,*102-107.

[23] Vinod, B. and Sabbagh, M. (1986) Optimal performance analysis of manufacturing systems subject to tool availability. *European Journal Operational Research*, **24**,398-409.

[24] de Werra D. and Widmer, M. (1991) Loading problems with tool management in flexible manufacturing systems: a few integer programming models. *International Journal of Flexible Manufacturing Systems*, **3**, 71-82.

[25] Zhou, C. and Wysk, R. A. (1989) Tool management for computer integrated manuafcturing. *1989 International Industrial Engineering Conference & Societies' Manufacturing and Productivity Symposium Proceedings,* 603-607.

Table 1. Computational results for algorithms BES1, BES2 and BHS

Numbers of workstations / SFMs / tools *	Distribution of each product type's demand **	CPU time in seconds of algorithm BES1	CPU time in seconds of algorithm BES2	α = (heu. sol. – opt. sol.) / opt. sol. × 100%	CPU time in seconds of algorithm BHS
10 / 5 / 5	N(μ , σ)	0.332024	0.335236	0.00%	0.04125
10 / 5 / 10	N(μ , σ)	0.171732	0.115141	0.00%	0.013692
10 / 5 / 15	N(μ , σ)	0.612018	0.756041	0.00%	0.043802
10 / 5 / 20	N(μ , σ)	0.390542	0.377353	0.00%	0.021314
10 / 5 / 25	N(μ , σ)	0.162474	0.187291	0.00%	0.012272
10 / 5 / 30	N(μ , σ)	0.526431	1.028072	0.00%	0.031471
10 / 5 / 40	N(μ , σ)	0.707721	0.554191	0.00%	0.027623
10 / 5 / 50	N(μ , σ)	0.583164	0.429504	0.00%	0.025416
10 / 5 / 60	N(μ , σ)	0.557502	0.671957	0.00%	0.046794
10 / 5 / 70	N(μ , σ)	0.321477	0.635921	0.00%	0.054977
10 / 5 / 80	N(μ , σ)	1.460843	1.876706	0.00%	0.027525
10 / 5 / 90	N(μ , σ)	1.267749	1.682032	0.00%	0.046794
10 / 5 / 100	N(μ , σ)	1.185041	1.015274	0.00%	0.036868
10 / 10 / 10	N(2μ , 2σ)	3.359976	5.546256	0.00%	0.079320
10 / 10 / 20	N(2μ , 2σ)	2.875952	6.164336	0.00%	0.054164
10 / 10 / 30	N(2μ , 2σ)	15.476321	17.313212	0.00%	0.117614
10 / 10 / 40	N(2μ , 2σ)	72.044533	82.006573	0.00%	0.104797
10 / 10 / 50	N(2μ , 2σ)	84.535737	45.909851	0.00%	0.114582
10 / 10 / 60	N(2μ , 2σ)	252.249141	248.948763	0.00%	0.179143
10 / 10 / 70	N(2μ , 2σ)	230.244476	260.088892	0.00%	0.202622
10 / 10 / 80	N(2μ , 2σ)	252.129641	407.676588	0.00%	0.175447
10 / 15 / 20	N(3μ , 3σ)	390.079165	1031.269551	0.00%	0.217302
10 / 15 / 30	N(3μ , 3σ)	468.837888	747.024348	0.00%	0.190164
5 / 15 / 40	N(3μ , 3σ)	245.543125	2235.728545	0.00%	0.123875
5 / 15 / 50	N(3μ , 3σ)	1097.361855	3487.869145	0.00%	0.112307
5 / 15 / 70	N(3μ , 3σ)	1078.098543	1204.733530	0.00%	0.093121
Total Replications = 130				Average Percentage = 0.00%	

Note: BES1 = an optimal solution algorithm using the branching rule (30) for solving problem P2. BES2 = an optimal solution algorithm using the branching rule (31) for solving problem P2. BHS = a heuristic solution algorithm of problem P2, and α = percentage of the heuristic solution above the optimal solution of problem P2.
* w / m / l = l tool types on each SFM, m heterogeneous SFMs at each workstation and w logical workstations.
μ = 1000, σ = 200, ** five replications of each problem settings.

Table 2. Computational results for algorithm BHS as the ratio above the lower bound

Numbers of workstations / SFMs	Number of magazine tools	Distribution of each product type's demand *	Upper limits of the searching tree depth **	α = (heu. sol. – (lower bound) / (lower bound)	CPU time in seconds of algorithm BHS
10 / 20	90	N(4 μ , 4 σ)	15	0.043017	0.395197
	100	N(4 μ , 4 σ)	15	0.043592	0.353215
10 / 25	20	N(5 μ , 5 σ)	20	0.068913	0.564182
	25	N(5 μ , 5 σ)	20	0.073171	0.862433
	30	N(5 μ , 5 σ)	15	0.055204	1.023062
	40	N(5 μ , 5 σ)	15	0.060203	0.712795
	50	N(5 μ , 5 σ)	15	0.057798	1.261289
	60	N(5 μ , 5 σ)	15	0.046115	1.345223
	70	N(5 μ , 5 σ)	15	0.050697	1.069793
	80	N(5 μ , 5 σ)	15	0.053384	0.891183
	90	N(5 μ , 5 σ)	15	0.052182	0.917596
	100	N(5 μ , 5 σ)	15	0.034591	0.755201
5 / 30	10	N(6 μ , 6 σ)	20	0.063012	0.468954
	15	N(6 μ , 6 σ)	20	0.045484	0.301698
	20	N(6 μ , 6 σ)	15	0.069660	0.353853
	25	N(6 μ , 6 σ)	15	0.068894	0.538252
	30	N(6 μ , 6 σ)	15	0.070768	0.390159
	40	N(6 μ , 6 σ)	15	0.058197	0.380096
	50	N(6 μ , 6 σ)	15	0.044606	0.456688
	60	N(6 μ , 6 σ)	15	0.064882	0.607871
	70	N(6 μ , 6 σ)	15	0.048888	0.508413
	80	N(6 μ , 6 σ)	15	0.041359	0.530154
	90	N(6 μ , 6 σ)	15	0.057705	0.772320
	100	N(6 μ , 6 σ)	15	0.051616	0.623085
Total replications = 24				Average ratio = 0.055164	

Note: * one replications of each problem setting.**The searching tree is obtained by using Algorithm BES1. BES1 = an optimal solution algorithm using the branching rule (30) for solving problem P2, BHS = a heuristic solution algorithm of problem P2, and α = percentage of the heuristic solution above the optimal solution of problem P2.
μ = 1000, σ = 200.