International Journal of Management Science
Vol. 2, No. 1, December 1996

A Job Loading Procedure as a Kernel Part of FMS Integrated Operating System and Its Evaluation

Hiroshi Katayama*

Abstract

FMS operating system consists of several subsystems in general, i.e. tool grouping subsystem, tool / job assignment subsystem, job dispatching subsystem, and papers dealing with each subsystem were published by many researchers [1], [4], [6], [8], [9], [10], [11], [12], [13], [14], [15], [16].

This paper mainly discusses about tool / job assignment subsystem as a job loading procedure, that occupies the kernel position of overall FMS operating system. Its performance is evaluated through simulation experiments of an integrated operating system under a typical FMS hardware configuration implemented in many machining factories, which is composed of the proposed procedure as well as a job dispatching procedure including several heuristic dispatching rules in terms of rule-base.

1. Introduction

Many industries are continuing to introduce FMS in order to accomplish wide range of product capability with small batches for getting competitive edge in the market, of which recent feature is represented by the words of globalization, versatility, rapidity etc.

* Department of Industrial and Management Systems Engineering, School of Science and Engineering, Waseda University

To cope with such environmental difficulties, not only introducing high performance hardware but also operating software system managing hardware is the key element for overall satisfactory performance. Particularly, scale of FMS tends to become larger and most of machining cells are comprised of multi-purpose machining centers. This means more and more massive investment tends to be thrown as fixed cost to implement FMS into factories, and, as a result, operating systems of FMS are getting to be more complex and performance of their software becomes a critical issue [2], [7], [12], [16].

FMS operating system consists of several subsystems in general, i.e. tool grouping subsystem, tool/job assignment subsystem, job dispatching subsystem, and papers dealing with each subsystem were published by many researchers [1], [4], [6], [8], [9], [10], [11], [13], [14], [15]. For example, in order to increase working rate of FMS to recover such huge amount of investment, job loading problem (how jobs and tools have to be assigned to each machining cell) is highlighted as a part of FMS operating system design [1], [4], [6], [9], [10]. Actually, this problem is being studied independently by many researchers in terms of mathematical programming model for example, and most of such approaches use non-linear formulation and get solution with intricate methods[13].

Based on this background, this paper discusses a two stage hierarchical linear formulaltion of job loading problem and compares its performance with the formulation developed by Shanker et al[10] by evaluating typical FMS system performance such as average job loading rate, average working rate, average overtime rate as well as planning computation time through simulation experiments.

2. Loading Procedure

As linear formulation needs huge number of constraints and decision variables in general, many authors try to develop loading procedures in terms of non-linear model [13], and independently, some others try to introduce efficient searching procedures, e.g. branch and bound approach [1], [6], for getting optimal solution, However, it is still expected that huge computing time will be wasted if such formulation is chosen, for example, that of Shanker et al[10]. In addition, another tough obstacle will arise to get satisfactory results by popular mathematical programming package, e.g. LINGO [3], as amount of memory resource and particular functions might be indispensable to deal with such formulations.

The model developed by Shanker et al is given in the following section as a benchmark system, i.e. an example of non-linear formulation, which is well defined compact tool / job assignment problem.

2.1 Shanker et al's Model

In the job loading problems, many authors aim to attain maximum total machine-hour of jobs loaded in the certain planning duration, and the model developed by Shanker et al is categorized into such procedure. Namely, this model consists of expression (1) as the objective function which aim to maximize total processing time of loaded jobs, and expressions (2)~(6) as constraints. Related notations are summarized in Table 1.

$$Max. \sum_{i=1}^{M}\sum_{k=1}^{Y_i}\sum_{j=1}^{N} A_i P_{ikj} x_{ikj} \tag{1}$$

s.t.

$$\sum_{i=1}^{M}\sum_{k=1}^{Y_i} S_{ikj} x_{ikj} - \sum_{p=2}^{M}(-1)^{p+1}\Big\{\sum_{i_1=1}^{M-p+1}\sum_{i_2=i_1+1}^{M-p+2}\cdots\sum_{i_p=i_p-1}^{M}\sum_{k_{i_1}=1}^{Y_{i_1}}\sum_{k_{i_1}=1}^{Y_{i_p}}(Z_{i_1k_{i_1};i_2k_{i_2};\dots;i_pk_{i_p}})(x_{i_1k_{i_1}j})\cdots(x_{i_pk_{i_p}j})\Big\} \le G_j, \quad j=1,\dots,N \tag{2}$$

$$\sum_{r\in B(i,k)} x_{ikr} \le 1, \qquad i=1,\dots,M\,;\ k=1,\dots,Y_i \tag{3}$$

$$\sum_{k=1}^{Y_i}\sum_{j=1}^{N} x_{ikj} = \tilde{x}_i Y_i, \qquad i=1,\dots,M \tag{4}$$

$$\sum_{i=1}^{M}\sum_{k=1}^{Y_i} A_i P_{ikj} x_{ikj} \le \mathrm{H}, \qquad j=1,\dots,N \tag{5}$$

$$\tilde{x}_i,\ x_{ikj} = 0 \text{ or } 1 \qquad i=1,\dots,M\ :\ k=1,\dots,Y_i\ ;\ j=1,\dots,N \tag{6}$$

Where, (2) : Constraint on the number of tool slots available in each machining cell

(3) : Constraint of uniqueness of job operation assignment to capable machine

(4) : Constraint of consistency between number of assigned job operations and operations have to be performed

(5) : Constraint on the total operation time available for each machining cell

(6) : Constraint concerning to the definition of decision variables

Table 1. Symbols of the Shanker et al's Model

Subscripts :

i Job number $1 \leq i \leq M$

k Operation number $1 \leq k \leq Y_i$

j Machine number $1 \leq j \leq N$

*The term "Machine" is used instead of "Machining Cell" for simplification

Parameters :

H Length of one shift [minutes]

P_{ikj} Processing time of operation k of job i on machine j

Y_i Number of operations of job i

A_i Lot size of job i

S_{ikj} Number of tool slots required for processing operation k of job i on machine j

G_j Tool magazine capacity of machine j

$B(i,k)$ Set of machines on which operation k of job i can be performed

$Z_{i_1k_{i_1};i_2k_{i_2};\ldots;i_pk_{i_p}}$

Number of tool or slot usage when operation k_{i1} of job i_1, operation k_{i2} of job i_2,... and operation k_{ip} of job i_p are processed on the same machine

Decision Variables :

$\tilde{x}_i$ =1 If job i is selected

=0 Otherwise

x_{ikj} =1 If operation k of job i is assigned on machine j

=0 Otherwise

In this formulation, there are number of non-linear terms of decision variables Xs as combinatorial terms in expression (2), and also this expression can not be described concretely if the number of jobs is not fixed. This latter feature could be called "parameter dependent formulation" or "data dependent formulation". These two major defects cause serious difficulty to get a solution with a simple procedure, and therefore, an intricate algorithm is developed especially to overcome the first defect.

In this context, some new formulations are desired for job loading phase of FMS operation, which can be calculated by simpler algorithms with fewer computer resources as well as parameter independent formulation of the problem, i.e. concrete formulation for any number of machines and jobs.

In the following two sections, an alternative framework and formulation are described to overcome such difficulties.

2.2 An Integrated FMS Operating System

In this section, an small scale integrated operating system of FMS is introduced to stress the importance of job loading sub−system as well as mutual complementation of each sub−system. A procedure of FMS operating system being discussed here consists of job loading sub−system [Phase (1)] and job dispatching sub−system [Phase (2)], of which each function is executed term by term called shift, which is defined as the minimal planning / execution time bucket.

In advance of the phase (1) procedure, jobs being able to be processed in the coming shift are identified and classified into urgent and normal jobs to avoid due date interference.

Then, jobs to be processed in the considered shift are determined from classified job pool to meet objective function, that is, maximum total operation time criterion is adopted in this paper. Needless to say, necessary operations are pre−determined in the up−stream function, e.g. product design or production engineering division, and tools to complete whole duty should be determined in this phase simultaneously, In this sense, Phase (1) is regarded as a planning phase.

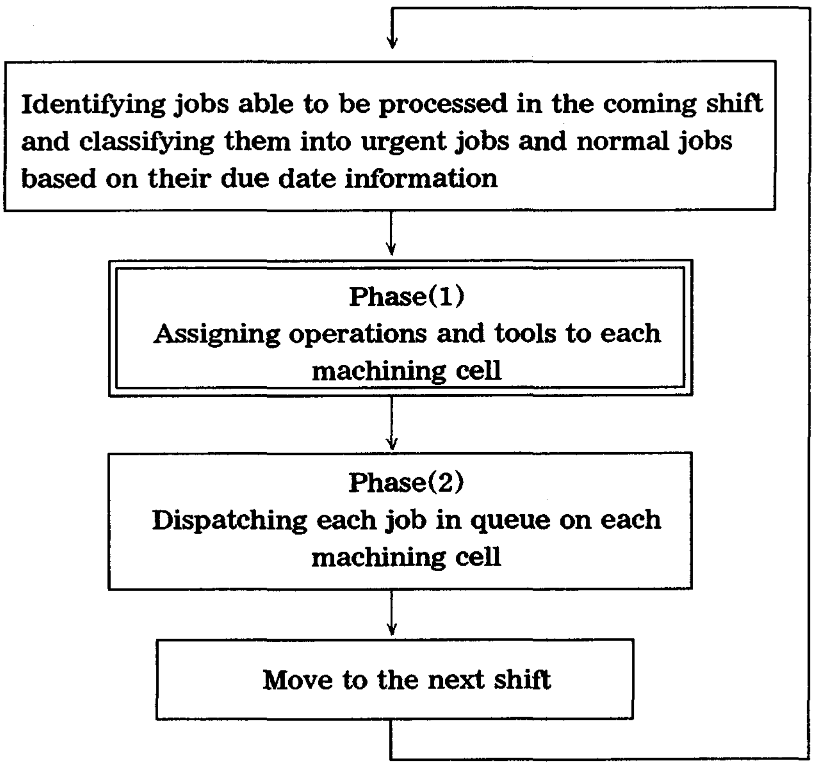
Phase (2), on the other hand, is regarded as a procedure of execution phase, which dispatches each job in queue on each machining cell to avoid inefficient utilization of whole capacity caused by bottle necks occurred in the actual time based operation sequence. Many dispatching procedures are, generally, constructed by simple dispatching rule, class of such rules or network structured rule-base, e.g. [4], [8].

Both phases are hierarchically processed, however, phase (1) plans from relatively macro-scopic point of view to realize maximum utilization and phase (2) is executed from micro-scopic point of view to recover expected inefficiency.

These three step sub-procedures are performed in term-wise as shown in Figure 1. The other functions such as tool classification are not considered for simplified discussion.

Job loading procedure proposed in this paper is described in the next section and is applied as Phase(1) to evaluate its performance by simulation experiments.

Figure 1. Sequence of An Integrated Operating System



2.3 Proposed Loading Procedure

First of all, let introduce the concept of machine class J, in which each machine has exactly the same specification, rather than considering each machine separately such as in the Shanker's model. This machine grouping is often useful to simpler management of the actual FMS.

Detail description of proposed job loading procedure is illustrated in Figure 2, which consists of three major portions such as Initialize Step, Step 0 and subsequent steps denoted as Step sn, and its distinctive feature is two stage "parameter independent" tool / job assignment problem solving, that is, iterative logic of job−tool selection problem solving and tool assignment problem solving. The essential idea is to aim detail optimal solution through tool assignment logic based on the rough cut solution obtained by job−tool selection logic.

Outline of each step is explained as follows.

〈Initialize Step〉

In the first stage of computation, related variables have to be initialized as foollows.

① All of the E_{sn}, the sets of decision variable sets DV_{sn} which is calculated in the step sn, should be cleared by null set ϕ : $E_{sn} \leftarrow \phi$ for $sn=0,\ldots,SN$.

② Step number sn should be cleared by 0: $sn \leftarrow 0$.

③ Set of machines UD, that operations and tools are fully assigned already, should be cleared by ϕ : $UD \leftarrow \phi$.

④ Any arbitrary machine j should be selected as an element of the set of machines UA, which is tried to assign jobs and tools : $UA \leftarrow \{\exists j\}$.

⑤ Every other machines js not included in UA are defined as elements of the set of machines UR, which any jobs are not assigned yet : $UR \leftarrow \{\forall j\} \backslash UA$.

〈Step 0〉

① In order to get rough cut solution, job−tool selection problem is solved firstly.

② Obtained value of objective function, that is, total job load assigned to the whole FMS which is calculated by setting $\lambda=1$ in (7) and (13), is substituted for upper bound of objective value UB_0.

③ Set of decision variables DV_{sn} denoting adopted jobs and tools, of which values are 1s, are added to initial cumulative set of decision variables E_0.

④ Cumulative objective value F_{UD} is to be cleared by 0 and go to Step 1 (The first stage of Step sn).

⟨Step $sn(sn=1,\dots, SN)$⟩

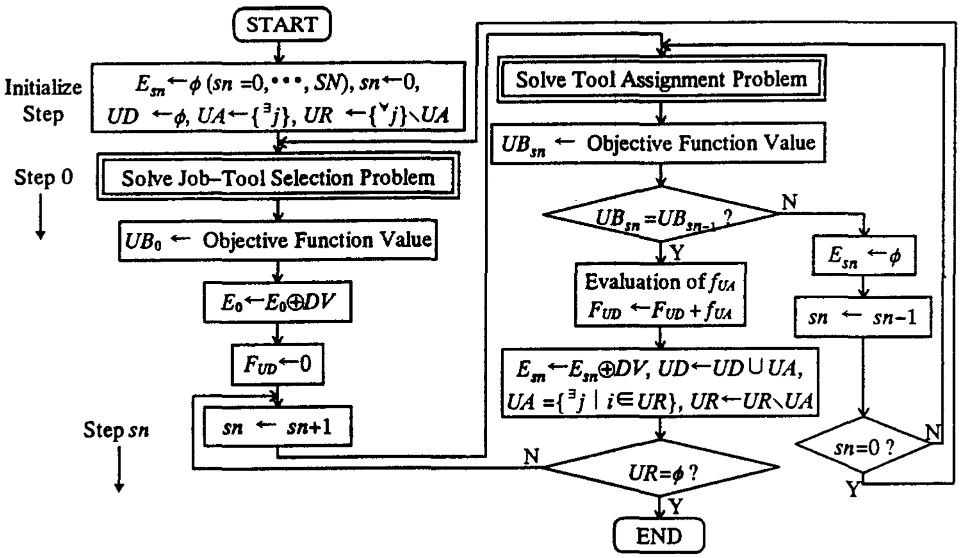
① In order to investigate that whether tools, which is selected in Step 0, are able to be assigned to machines, tool assignment problem is solved.

② Obtained value of objective function is substituted for UB_{sn}.

③ If $UB_{sn} \neq UB_{sn-1}$ then $E_{sn} \leftarrow \phi$, and go to Step $sn-1$
otherwise, DV_{sn} is added to E_{sn}, UA is added to UD, and any arbitrary machine, which is in UR, is transferred to UA.

④ If $UR=\phi$ then this algorithm terminates.
otherwise go to Step $sn+1$.

Figure 2. The Proposed Loading Procedure



Where, ⊕ Addition of Set ∖ Subtraction of Elements
∪ Union of Set + Addition of Scalar

Exact formulation of major job−tool selection problem and tool assignment problem are given in succeeded expressions (7)−(21). Table 2 summarizes related symbols, and symbols do not appeared in there are defined in Table 1. That is, symbols used in the both models follow the common notation unless they are redefined.

《Job−Tool Selection Problem : Formulation of Step 0》

$$Max. \sum_{i \in D_s} \sum_{k=1}^{Y_i} \lambda \, A_i P_{ik} \cdot \tilde{x}_i + \sum_{i \in S - D_s} \sum_{k=1}^{Y_i} A_i P_{ik} \cdot \tilde{x}_i \qquad (7)$$

s.t.

$$\sum_{h=1}^{Q} S_h \tilde{w}_h \leq \sum_{j=1}^{N} G_j \qquad (8)$$

$$\tilde{x}_i Y_i \leq \sum_{k=1}^{Y_i} \tilde{w}_h, \quad h = C_{ik} \qquad i=1,\ldots,M \qquad (9)$$

$$\sum_{i=1}^{M} \sum_{k=1}^{Y_i} A_i P_{ik} \cdot \tilde{x}_i \leq \eta NH \qquad (10)$$

$$\text{In case of } E_0 \neq \phi : \sum_{dv \in DV_c} dv < |DV_0|, \; DV_0 \in E_0 \qquad (11)$$

$$\tilde{x}_i, \; \tilde{w}_h = 0 \text{ or } 1 \qquad (12)$$

Where, (7) : Objective function aiming to maximize weighted total job loading

(8) : Constraint of total tool slot number available in entire FMS

(9) : Constraint of necessary condition which number of assigned tools have to satisfy, that is, it should be greater than or equal to the required number of operations

(10) : Constraint to total operation time available on the entire FMS

(11) : Constraint to exclude solutions obtained before the current optimization step

(12) : Constraint concerning to the definition of decision variables

〈Tool Assignment Problem : Formulation of Step *sn*〉

$$Max.\ F_{UD_{sn}} + \sum_{J\in U}\sum_{i\in D_s}\sum_{k=1}^{Y_i} \lambda\ A_i P_{ikJ} x_{ikJ} + \sum_{J\in U}\sum_{i\in S-D_s}\sum_{k=1}^{Y_i} A_i P_{ikJ} x_{ikJ} \tag{13}$$

s.t.

$$\sum_{h\in T} S_{hJ} W_{hJ} \le G_J, \qquad J\in U \tag{14}$$

$$\sum_{J\in U} W_{hJ} \le NT_{h,sn}, \qquad h\in T \tag{15}$$

$$\sum_{r\in B(i,k)} x_{ikr} \le 1, \qquad i\in S : k=1,\dots,Y_i \tag{16}$$

$$\sum_{k=1}^{Y_i}\sum_{J\in U} x_{ikJ} \le \tilde{x}_i Y_i, \qquad i\in S \tag{17}$$

$$x_{ikJ} \le w_{hJ}, \qquad h=C_{ik} \qquad i\in S : k=1,\dots,Y_i : J\in U \tag{18}$$

$$\sum_{i\in S}\sum_{k=1}^{Y_i} A_i P_{ikJ} x_{ikJ} \le \eta\,|J|\,H, \qquad J\in U \tag{19}$$

$$\text{In case of } E_{sn}\neq\phi : \sum_{dv\in DV_c} dv < |DV_{sn}|,\ DV_{sn}\in E_{sn} \tag{20}$$

$$x_{ikJ},\ w_{hJ}=0 \text{ or } 1 \tag{21}$$

Where, (13) : Objective function aiming to maximize weighted total job loading in the step *sn*

(14) : Constraint of total tool slot number available in each machine set, *i.e.* *UA* and *UR*

(15) : Constraint of total tool number available in the step *sn*

(16) : Constraint of uniqueness of job operation assignment to capable machine

(17) : Constraint of necessary condition which number of assigned job operations have to satisfy, that is, it should be less than or equal to the required number of operation

(18) : Constraint of necessary condition which decision variable x_{ikJ} have to satisfy, that is, tool *h* have to be assigned to machine set *J* if job operation requiring tool *h* is assigned to *J*.

(19) : Constraint of total operation time available on each machine set

(20) : Constraint of exclude solutions obtained before the current optimization step

(21) : Constraint concerning to the definition of decision variables

Table 2. Symbols of the Proposed Model

Subscripts :

h	Tool number $1 \le h \le Q$
J	Set of machines $J \in U$

Parameters :

sn	Step number(sn=1,..., SN)
DV_{sn}	Set of decision variables obtained in the step sn, of which values are 1s
DV_c	Set of decision variables obtained in the current step
E_{sn}	Family of DV that is constructed in the step sn
UD	Set of machines on which job operations and tools are fully assigned already
UA	Machine on which operations are assigned in the current step
UR	Set of machines on which operations are not assigned yet
U	$UA \cup UR$
$F_{UD_{sn}}$	Cumulative objective value contributed by machines of UD in the step sn
f_{UA}	Objective value contributed by machine of UA
UB_{sn}	Upper bound of objective value in the step sn
S_{hJ}	Number of tool slots required for processing by tool h on set of machines J
G_J	Tool magazine capacity of set of machines J
C_{ik}	Tool which operation k of job i requires
D	Set of higher priority jobs
λ	Arbitrary large number, which is for giving higher priority to urgent jobs
η	Work load parameter (in this paper, η=1)
$S_{h\cdot}$	$\min_j S_{hj}$ $(j=1,\ldots,N)$
$P_{ik\cdot}$	$\min_j P_{ikj}$ $(j=1,\ldots,N)$

$NT_{h,sn}$	Number of tool h available in the step sn
S	Set of jobs which are selected from job pool in the step 0
D_s	$S \cap D$
T	Set of tools which are selected in the step 0
P_{ikJ}	$\min_j P_{ikj} \ (j \in J)$

Decision Variables :

$\tilde{w}_h$	=1	If tool h is selected
	=0	Otherwise
x_{ikJ}	=1	If operation k of job i is assigned on J(set of machines)
	=0	Otherwise
w_{hJ}	=1	If tool h is Assigned on set of machines J
	=0	Otherwise

Others :

$\lvert\cdot\rvert$	Number of elements contained in set

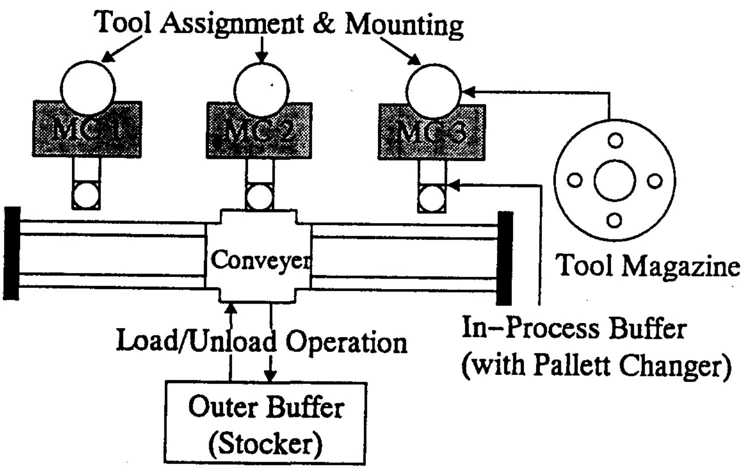
3. Performance Evaluation

In order to evaluate the performance of the proposed procedure, typical hardware configuration of FMS observed in actual machining factories are considered, and comparison analysis between the proposed procedure and Shanker et al's [10] model is performed on this hardware domain.

3.1 FMS Hardware Model

Figure 3 shows the FMS hardware configuration under consideration and major features related to the hardware, which is simplified model of actual FMS, are summarized as follows.

Figure 3. A FMS Hardware Configuration



1) Number of machining cells is three and their specification are machining centers (MC), that is, each cell can perform required operation through proper tool assignment.
2) Total number of machining operations necessary and able to processed by MCs are 21, which are shown in Table 3.

Table 3. Operations and Corresponding Standard Time Data

Operation Number	1	2	3	4	5	6	7	8	9	10	11
Operation Time(min.)	4	5	6	4	5	6	4	5	6	4	5

Operation Number	12	13	14	15	16	17	18	19	20	21
Operation Time(min.)	6	4	5	6	4	5	6	4	5	6

3) One and only one tool is supposed to be required for each operation.
4) Number of tools which each tool magazine can hold, that is, capacity is five.
5) Because of negligible small transfer time between machines comparing with operation time in Table 3, these are supposed to be zeros.

3.2 Assumptions and Preconditions on Operation Environment

Features of operation environment considered and artificial conditions introduced for simulation are as follows.

1) Length of each shift is 480 minutes (8 hours), and duration of simulation is 110 shifts. Data are captured from the 11th shift to the 110th shift for statistical evaluation.
2) Distribution of job arrival is exponential distribution with arrival rate 1 / 30 (mean time between arrival=30 minutes).
3) Lot size of each job is discrete rectangular distribution varies on [2, 9] domain.
4) Due date of each job is 10 shifts forward after arrival.
5) Number of operations required for each job is discrete rectangular distribution varies on [2, 4] domain.
6) Operations and their process times for each job are selected from Table 3 through discrete rectangular distribution varies on [1, 21] domain.
7) Jobs which has two shifts until due dates are treated as higher priority jobs.
8) Simulation experiments are performed on DOS / V PC(CPU: 486DX2, 33MHz).
9) Calculation of job−tool selection problem and tool assignment problem in the proposed procedure is executed on LINGO [3], one of the most popular mathematical programming package.
10) Dispatching method adopted is the rule−based system developed in Katayama et al [4], [5].

3.3 Criteria

Following four criteria values are calculated to evaluate performance of overall integrated system whose main feature is the proposed tool / job assignment procedure of job loading phase.

1) Average Computation Time(ACT) per Shift : (Total Computation Time for 100 Shifts) / (100 Shifts)
2) Average Loading Rate(ALR) per Shift : (Total Process Time of Jobs Loaded in 100 Shifts) / (100 Shifts)
3) Average Working Rate(AWR) per Shift : (Total Elapsed Time in 100 Shifts) / (100 Shifts)
4) Average Overtime Working Rate(AOWR) per Shift : (Total Overworking time in 100 Shifts) / (100 Shifts)

3.4 Results of Simulation

Table 4 is given as an example results of simulation experiments, which try to clarify quantitative advantage of the proposed procedure for loading phase comparing with Shanker et al's procedure.

Table 4. Summary Results

	Shanker	Proposed Loading Procedure
Average Compuation Time(min.)	21.52	1.83
Average Loading Rate	0.945	
Average Working Rate	0.979	0.992
Average Overtime Rate	0.003	0.005

As the proposed procedure is not facilitated smoothing function of job duties between machining cells, which is considered as load balancing in Shanker et al and is considered to be relevant when idle time of machining cells in execution phase can not be eliminated by job dispatching procedure, their algorithm gets better performance than the proposed procedure in terms of average working rate, that is, AWR of the proposed procedure (0.992) contains more idle time than that of Shanker procedure (0.979) mainly because of biased job loading on machining cells.

On the other hand, average computation time of proposed procedure is significantly shorter than that, because area of feasible solution in this example, which will diverge in proportion to the increase of job numbers in a shift, is too large to be enumerated by their algorithm.

From these results, proposed procedure is considered as an effective tool if jobs able to process in a shift are substantial, and parameters as well as data related to operations are suitable for load balancing, i.e. unbiased operation times of each job.

4. Conclusion

In this paper, a job loading procedure, of which formulation contains two stage linear mathematical programming models and is independent of facility and job configuration parameters, is developed as a part of FMS integrated operating system, and its performance is evaluated through simulation experiments based on the data investigated at the actual FMS shop floor in some factory. The proposed procedure is qualified by collaborated companies to be applicable for practical use.

References

[1] Berrada M. and K. E. Stecke, "A Branch and Bound Approach for Machine Load Balancing in Flexible Manufacturing Systems", *Management Science*, Vol. 32, No. 10(1986), pp.1316-1335.

[2] Carrie A. S., E. Adhami, A. Stephens and I. C. Murdoch, "Introducing a flexible manufacturing system", *International Journal of Production Research*, Vol. 22, No. 6(1984), pp.907-916.

[3] Cunningham K. and L. Schrage, LINGO User's Manual, LINDO System Inc. (1988).

[4] Katayama H., K. Oto and Y. Hirakawa, "Loading and Dispatching Poli-

cies for a Flexible Manufacturing System", *International Operations* (Ed. by R. H. Hollier et al), Elsevier Science Publishers B.V.(1992), pp. 347-353.

[5] Katayama H., T. Shinya and Y-L. Chen, "A Rule-based dispatching decision system for FMS and its performance", Proceedings of the first Asia Pacific Decision Science Institute Conference, Vol. 2, The Hong Kong University of Science and Technology(1996), pp. 491-500.

[6] Kouvelis P. and H. L. Lee, "Block angular structures and the loading problem in flexible manufacturing systems", *Operations Research*, Vol. 39, No. 4(1991), pp.666-676.

[7] Kusiak A., "Flexible manufacturing systems: a structural approach", *International Journal of Production Research*, Vol. 23, No. 6(1985), pp. 1057-1073.

[8] Nakamura Y., I. Hatono and J. Tamura, "Scheduling of Flexible Manufacturing Systems using Rule Base", *Journal of Measure and Automatic Control Association*, Vol. 23, No. 1(1987), pp.66-71.

[9] Shanker K. and Y. J. Tzen, "A loading and dispatching in a random flexible manufacturing system", *International Journal of Production Research*, Vol. 23, No. 3(1985), pp.579-595.

[10] Shanker K. and A Srinvasulu, "Some solution methodologies for loading problems in a flexible manufacturing system", *International Journal of Production Research*, Vol. 27, No. 6(1989), pp.1019-1034.

[11] Shen S. and Y. L. Chang, "Schedule Generation in a Flexible Manufacturing System: A Knowledge-based Approach", *Decision Support Systems*, Vol. 4(1988), pp.157-166.

[12] Solberg J. J., "A mathematical model of computerized manufacturing systems", *Proceedings of the 4th International Conference on Production Research*, Tokyo(1977), pp.1265-1275.

[13] Stecke K. E., "Formulation and Solution of Nonlinear Integer Production Planning Problems for Flexible Manufacturing Ststems", *Management Science*, Vol. 29, No. 3(1983), pp.273-288.

[14] Stecke K. E. and J. E. Aronson, "Review of operator/machine interference models", *International Journal of Production Research*, Vol. 23, No. 1(1985), pp.129-151.

[15] Vinod B. and J.J. Solberg, "Performance Models for Unreliable Flex-

ible Manufacturing Systems", *OMEGA*, Vol. 12, No. 3(1984), pp.299-308.

[16] Vinod B. and J.J. Solberg, "The optimal design of flexible manufacturing systems", *International Journal of Prduction Research*, Vol. 23, No. 6(1985), pp.1141-1151.

Appendix

〈Structure of Adopted Dispatching Rule-base〉

The dispatching procedure adopted in the considered integrated FMS operating system is shown in Table A-1, which is originally developed in [5] based on [4]. The feature of this procedure is: focus on reduction of real time monitored idle rate.

Table A-1. A Dispatching Procedure in terms of Idle Rate Reduction Rule-base

		Number of jobs queuing at the buffer of the current operation	
		Which is more than average between queues of every operations	Which is the minimum between queues of every operations
Number of Jobs at the Buffer of the next operation	There is a unique minimum number	Is there only one job having minimum queue at the next operation? Y → *PR*: Job having Minimum Queue of Next Operation N → *PR*: Job having Minimum Current Operation Time	Is there only one job having minimum queue at the next operation? Y → *PR*: Job having Minimum Queue of Next Operation N → *PR*: Job having Maximum Current Operation Time
	There are some minimum numbers	*PR*: Job having Minimum Same Operation Jobs in the Queue of Next Operation	*PR*: Job having Maximum Current Operation Time