INT. J. PROD. RES., 1997, VOL. 35, NO. 8, 2269–2284

A heuristic solution approach to the machine loading problem of an FMS and its Petri net model

M. K. TIWARI†*, B. HAZARIKA†, N. K. VIDYARTHI†, P. JAGGI† and S. K. MUKHOPADHYAY‡

A loading problem in a modern FMS is the allocation of a job from the job pool maintaining the flexibility of the system, reducing the system imbalance and thereby maximizing the throughput. In this context a heuristic has been developed. Ten sample problems have been tested with the proposed heuristic. A comparison has been made with the existing methods and it is found that the results are encouraging and indicate significant improvement. The Petri net model for the problem attempted by the proposed heuristic has been constructed to delineate its graphical representation and subsequent validation.

1. Introduction

An FMS is a group of automated machines and material handling devices which are linked together by a central computer system. Highly sophisticated machine tools enable the system for a wide range of manufacturing operations and allow simultaneous production of multiple part types maintaining a high degree of machine utilization. This system is an attempt to impart the efficiency of mass production while retaining the flexibility of the conventional manufacturing process. The machine loading problem in an FMS is specified so as to assign the machines, operations of selected jobs, and the tools necessary to perform these operations by satisfying the technological constraints (available machine time and tool slot constraint) in order to ensure the minimum system imbalance and maximum throughput, when the system is in operation.

Stecke and Solberg (1981) and Stecke (1983) have formulated a machine loading problem based on a mixed integer programming (MIP) approach. In these papers, the objective was to achieve the maximum expected production rate to be attained at a specific imbalanced work load for a system having unequal size machine groups. When the machines are not pooled into groups, a widely known loading objective is to balance the workload on all machines. Shanthikumar and Stecke (1986) have used this objective to minimize the work-in-process inventories, Berrada and Stecke (1986) adopted a nonlinear integer formulation of a particular FMS loading problem which is to balance the assigned workload on each machine tool. A mixed integer formulation for the loading problem that includes balancing workload and meeting the due date of the job types was suggested by Shankar and Tzen (1985). Liang and Dutta (1993) have proposed an integrated approach to part selection and the

† Department of Mechanical Engineering, North Eastern Regional Institute of Science & Technology (NERIST), Itanagar-791 109 India.

‡ National Institute of Industrial Engineering (Previously the National Institute for Training in Industrial Engineering) Bombay, India.

* To whom correspondence should be addressed.

machine loading problems. Lashkari *et al.* (1987) have also attempted the formulation of operation allocation problem to include the important planning aspects of refixturing and limited tool availability through the MIP approach. Sawik (1989) has also adopted the MIP approach to address the machine loading problem with an objective function to minimize the system imbalance while satisfying the constraints of available production time and tool magazine capacity. Sarin and Chen (1987), Co *et al.* (1990), Ram *et al.* (1990), have made a substantial contribution towards the understanding of the machine loading problem. Rajagopalan (1986) gave two different heuristic solutions with a mixed integer model. To obtain better production schedules without an iterative process, he combined the machine loading problem of FMS with part type selection and production ratio determination. Stecke (1989) proposes a heuristic algorithm using longest processing time (LPT) rules for the machine loading problem. Moreno and Ding (1993) presented two heuristic methods for solving the machine loading problem with the objective of balancing workloads and meeting due dates. Shankar and Srinivasalu (1989) suggested heuristic procedures with a bicriterion objective of minimizing the workload imbalance and maximizing the throughput for resources such as a number of tool slots on machines and available number of working hours in a given planning horizon. Mukhopadhyay *et al.* (1992) have suggested a heuristic solution developing a concept of essentiality ratio for objective of minimizing the system imbalance and thereby maximizing the throughput. Chen and Askin (1990) have considered three FMSs for comparison of the performance of their six loading heuristics.

Workload balance, minimization of inter-machine part movement, routeing flexibility, tool investment and maximum machine utilization are the five basic objectives whose separate evaluation gives the dependence of performance of their heuristics. The MIP approach for solving the machine loading problem of an FMS is computationally infeasible even for deterministic formulation. Computation time required for solving the machine loading problem of a moderate size FMS is considerably large which necessitates the development of heuristic procedures to tackle such problems.

Taking into account the processing requirements of small batches of several part types, a random FMS is chosen in which orders arrive in a random manner and each order stands for one product type. Several operations may be required to produce one part type with flexibility in carrying out an operation on more than one machine. In addition to this, the system may have more than one machine of the same type. In this paper, a heuristic has been proposed based on the utilization of the maximum remaining available time on each machine keeping in mind the tool slot requirements of the job for achieving the minimum positive system imbalance resulting in maximum possible throughput. The proposed heuristic has been tested by applying it to the problems of Shankar and Srinivasulu (1989) and Mukhopadhyay *et al.* (1992) and the results obtained are significantly improved over the existing methods. A Petri net model of the machine loading problem solved using the proposed heuristic is constructed to delineate its graphical features. Petri net modelling has been carried out to demonstrate effectively the formulation difficulties related to routeing options, tool slot constraint, concurrency and lot sizes etc.

1.1. *Informal introduction to Petri net models*

Petri nets originated in 1962 from the doctoral dissertation of Carl Adam Petri. Since then, there has been great deal of development of Petri net analytical techniques. For a coherent and concise definition of Petri net terms used in this paper, the reader is referred to Peterson (1981) and Viswanadham and Narahari (1994).

A Petri net is a formal graph model for description and analysis of systems that exhibit both synchronous and concurrent properties and thus are well suited to model the dynamics of an FMS. The following reasons make the Petri net suitable for modelling the loading problems of FMSs.

The precedence relation and structural interaction of concurrent and asynchronous events are reflected from the Petri net. The informative graphical nature is a good visual aid. They can be easily understood as they are logically derived models. Conflicts, deadlocks, buffer sizes can be easily and concisely modelled. The Petri net reveals the set of loadings to be described satisfying the precedence constraints. They may be interrupted depending upon tool slot constraints to restrict the set of possible firing. By the addition of activity duration time and resource allocation condition, the dynamic behaviour of a system can be studied at any one time.

2. Problem statement

A loading problem in FMS is solved and its performance is being judged by determining the system imbalance and the throughput. Shanker and Tzen (1985) have shown that the 'shortest processing time' (SPT) sequencing rule performs best on an average for the loading problem of a random FMS in balancing the workload. Adoption of SPT as the sequencing rule attempts to maximize the throughput in comparison to LPT, FIFO, MOPR etc. Therefore, scheduling of the job is according to the SPT sequencing rule. Ten problems from Mukhopadhyay *et al.* (1992) are studied in detail. Four machines are included and the size of the batches considered are less than twenty five, keeping in view the major economic forces governing the market. The problem is studied by taking three operations. However, it is advisable to preserve the optional operations as long as possible while considering all the possible routes. Optional operation means the operations which can be carried out on more than one machine. Flexibility lies in the selection of a machine for processing the optional operations of the part. To demonstrate the effect of tool slot constraint, each machine has been provided with a maximum of five tool slots. The following assumptions are made to reduce the complexities in analysing the loading problem of the system:

(1) Non splitting of job – which implies that a job undertaken for the processing is to be completed for all its operations before considering a new job.
(2) Unique job routeing – though the flexibility exists in the selection of a machine for optional operation, once a machine is selected for it, the operation must be completed on the same. This is called unique job routeing.
(3) Sharing and duplication of tool slots is not considered.
(4) Number of pallets and fixtures used in the system are sufficient and readily available.
(5) Parts are readily available on machines i.e. material handling time is negligible.

2.1. *Notation for the proposed heuristic*

P_j Individual processing time for each job.
N Batch size of each job.

P_e	Total processing time required for essential operations of job '*j*' on machines.
P_o	Total processing time required for optional operations of job '*j*' on machines.
s	$(j : j$ is a job in the SPT sequence belonging to the job pool).
G	$(j : j$ is a job yet to be allocated).
H	$(j : j$ is a job which has been allocated).
T_{ro}	Remaining tool slots on machine '*m*' after allocating operation '*o*' of job '*j*'.
T_{ao}	Available tool slots on machined '*m*' before allocating operation '*o*' of job '*j*'
T_{co}	Tool slot requirement for performance of operation '*o*'' of job '*j*' on machine '*m*'.
Q_m	Remaining processing time on machine '*m*' after performing operation '*o*' of job '*j*'.
Q_a	Available processing time on machine '*m*' before performing operation '*o*' of job '*j*'.
Q_p	Processing time on machine '*m*' for performing operation '*o*' of job '*j*'.
SUA_o	System unbalance after allocating the operation '*o*' for job '*j*' on machine '*m*'.
SUA'	System unbalance after allocating all essential and optional operations to respective machines for processing job '*j*'.
TP	Throughput for each problem.
SUA	Final system unbalance after allocating all possible jobs.

2.2. *Proposed heuristic*

Step 1. Determine the individual processing time for each job (P_j).

$$P_j = N(P_e + P_o)$$

.

Step 2. Adopt SPT sequence for job allocation.

Step 3. Define initial conditions.

$$S = G$$
$$H = (0)$$

Step 4. Calculate T_{ro} for first operation '*o*' of job '*j*' on machine '*m*' entertaining essential operation and preserving optional operations.

$$T_{ro} = T_{ao} - T_{co}$$

Step 5. If T_{ro} is negative, reject job else go to Step (6).

Step 6. Allocate operation virtually to machine '*m*' having the maximum remaining processing time 'Q_m'.

$$Q_m = Q_a - Q_p$$

Step 7. Calculate SUA_o

$$SUA_o = \sum_{m=1}^{n} Q_m$$

Step 8. Repeat steps 4–7 for all operations of job '*j*'.

Step 9. Assign SUA_o = SUA′. If SUA′ is positive allocate job to set 'H' and proceed for next job i.e. from step 4 else go to step 10.
Step 10. Reallocate the last job of the allocated set with the jobs in set G according to SPT sequence and repeat steps 4–10.
Step 11. Final job allocation set is to be chosen based on minimum positive system imbalance.

$$\text{Assign SUA}' = \text{SUA}$$

Step 12. Determine throughput for each shift.

$$\text{TP} = \sum_{j=(1,2,\ldots X)}^{x} \text{Batch size of jobs allocated to set H}$$

2.3. *Solution methodology*

The proposed heuristic has been used to solve the machine loading problem of FMS given in Table 1.

Step 1. P_j = Batch size × unit processing time
$8 \times 18 = 144$
Step 2. P_j for each job is calculated and is arranged in increasing order.
Step 3. Define initial condition.

$$S = (1, 4, 5, 6, 3, 2, 7, 8)$$

Step 4. T_{ro} for first operation of job 1 on machine 3.

$$T_{ro} = 5 - 1 = 4$$

Step 5. If T_{ro} is not negative, go to step (6).
Step 6. Operation 1 of job 1 is allocated to machine 3 and Q_3 is calculated.

$$Q_3 = 480 - 144 = 336$$

Step 7. Calculate

$$\begin{aligned} SUA_o &= (Q_1 + Q_2 + Q_3 + Q_4) \\ &= (480 + 480 + 336 + 480) \\ &= 1776 \end{aligned}$$

Step 8. Allocate **all operations.**
Step 9. Assign SUA′ = SUA_o = 1776 (which is positive). Therefore, new initial condition.

$$S = (1, 4, 5, 6, 3, 2, 7, 8)$$
$$G = (4, 5, 6, 3, 2, 7, 8)$$
$$H = (1)$$

Similarly, we proceed to allocate the next job according to sequence.

The complete solution to Problem 1 is given in Table 2.

3. Modelling methodology

In the proposed model, the execution of various processes are sequential. All the places and transitions are immediate i.e. time is neither associated with places nor

Job number	Operation number	Batch size	Unit processing time (min)	Machine number	Slot needed
1	1	8	18	3	1
2	1	9	25	1	1
				4	
	2		24	4	1
	3		22	2	1
3	1	13	26	4	2
				1	
	2		11	3	3
4	1	6	14	3	1
	2		19	4	1
5	1	9	22	2	2
				3	
	2		25	2	1
6	1	10	16	4	1
	2		7	4	1
				2	
				3	
	3		21	2	1
				1	
7	1	12	19	3	1
				2	
				4	
	2		13	2	1
				3	
				1	
	3		23	4	3
8	1	13	25	1	1
				2	
				3	
	2		7	2	1
				1	
	3		24	1	3

Table 1. Job description (adopted from Mukhopadhay *et al.* (1992)).

with transitions. Places represent activities or resources and transitions represent initiation or termination of involved activities. Tokens in the places represent either the availability of the resources, readiness of the part for the next process or a part being processed. Intermediate places are dummy processes. Potential deadlock is implicitly avoided by using intermediate places. Token in these places show the readiness of the job for the next execution. Numerical value in the places represent the lot size of the corresponding sets (tool slots and jobs). In our Petri net model, job representation has only one initial and final place. A marking is said to be an initial marking M_o when it represents the initial state of the system. This marking changes whenever a transition is fired. For example, the initial marking in Fig. 2 represents M_o, where $M_o = (1, 1, 1, 0)$. Once the transition '*t*' is fired the marking M_o results in $M_1 = (1, 1, 0, 1)$. Directed arcs show the material and information flow in the system. By putting weight (w) on these arcs, a condition has been imposed that the transition will be fired if and only if *w* is less than or equal to the number of tokens in the place from which the arc is directed.

Jobs in sequence	Machine number	Process P_{joms}	Available time on m/cs	Processing time required on m/c: Essential	Processing time required on m/c: Operational	Remaining time on machines	Available tool slots on each machine	Tool slots required	Remaining tool slots on each machine	System imbalance	Remarks
1	1		480			480	5		5	1776	Job selected for processing on machine 3.
	2		480			480	5		5		
	3	1131	480	144		336	5	1	4		
	4		480			480	5		5		
4	1		480			480	5		5	1578	Job selected for processing on machine 3 & 4.
	2		480			480	5		5		
	3	4132	336	84		252	4	1	3		
	4	4242	480	114		366	5	1	4		
5	1		480			480	5		5	1353	
	2	5223	480	225		255	5	1	4		
	3		252			252	3		3		
	4		366			366	4		4		
	1		480			480	5		5	1155	Job selected for processing in machine 2.
	2	5123	255		198	57	4	2	2		
	3		252			252	3		3		
	4		366			366	4		4		
6	1		480			480	5		5	995	
	2		57			57	2		2		
	3		252			252	3		3		
	4	6144	366	160		206	4	1	3		
	1		480			480	5		5	925	
	2		57			57	2		2		
	3	6434	252		70	182	3	1	2		
	4		206			206	3		3		

Table 2 (*cont.*)

	1	6314	480		210	270	5	1	4	715	Job selected for processing in machines 4, 3 & 1
	2		57			57	2		2		
	3		182			182	2		2		
	4		206			206	3		3		
3	1		270			270	4				Job rejected due to tool slot constraint
	2		57			57	2				
	3	3235	182	143		39	2	3			
	4		206			206	3				
2	1		270			270	4		4	301	
	2	2325	57	198		- 141	2	1	1		
	3		182			182	2		2		
	4	2245	206	216		- 10	3		2		
	1	2115	270		225	45	4		3	76	Job selected for processing in machines 2, 4 & 1
	2		- 141			- 141	1		1		
	3		182			182	2		2		
	4		- 10			- 10	2		2		
7	1		45			45	3				Job rejected due to tool slot constraint
	2		- 141			- 141	1				
	3		182			182	2				
	4	7346	- 10	276		- 286	2	3			
8	1	8316	45	312		- 267	3	3	0	- 236	Job rejected due to negative system imbalance
	2		- 141			- 141	1		1		
	3		182			182	2		2		
	4		- 10			- 10	2		2		

Table 2. Illustration of problems given in Table 1 by the proposed heuristic.

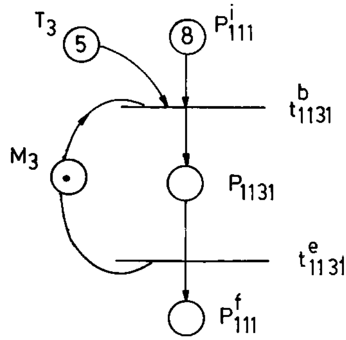

Figure 1. The Petri net model of job 1.

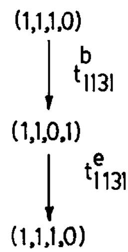

Figure 2. Reachability tree of the Petri net in Fig. 1.

An enabled transition fires by removing the number of tokens equal to the weight of the corresponding input arc from each of its input places. The concept of weight of arc has been used for tool slots only. When the enabled transition is fired, the number of tool slots consumed which are equal to the weight of input arc is shown in each of the input places.

The Petri net model of the system is decomposed according to each job and for job 1 is shown in Fig. 1. Its reachability tree is shown in Fig. 2. Similarly, the Petri net model for job 6 is shown in Fig. 3. Its reachable markings are tabulated in Table 3. The transitions fired for obtaining the new markings are shown in Table 4. The places representing machine resource are being repeated for clarity of representation. A combined Petri net model showing the sequential operation of jobs 1, 4, 5 and rejection of job 3 is shown in Fig. 4.

3.1. *Notations used in the Petri net model*

O_{joms}	Operation of jth job, oth operation in mth machine and sth sequence loading.
t^b_{joms}	Beginning of transition of operation o_{joms}.
t^e_{joms}	End of transition of operation o_{joms}.
P_{joms}	Place representing operation place of o_{joms}.
P^n_{joms}	mth intermediate place of job 'j' (Jj) after the 'oth process in the sth sequence of loading.
P^i_{jms}	mth initial place of Jj which represents the beginning of job type Jj in sth sequence of loading.

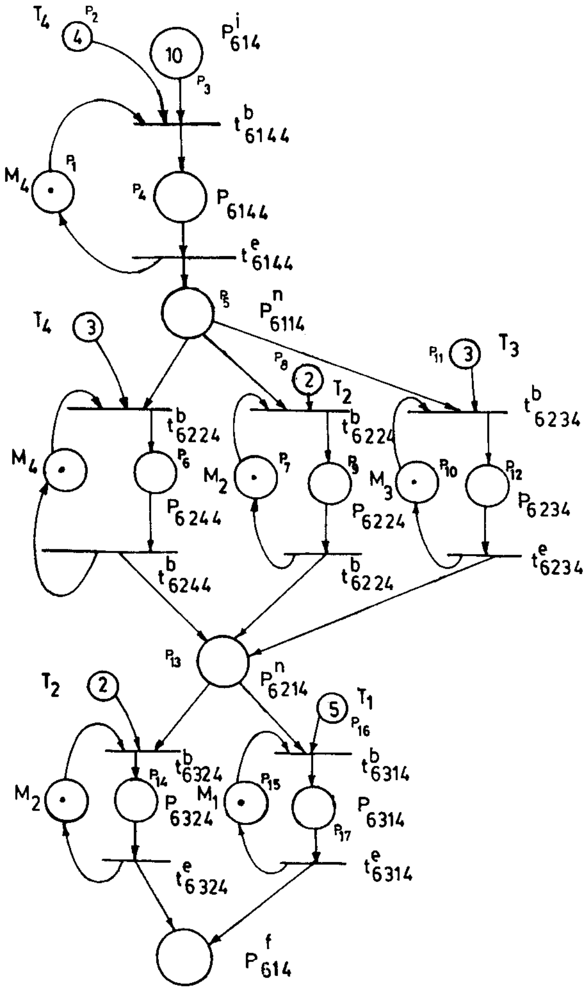

Figure 3. The Petri net model: job 6.

P^f_{jms}	*m*th final place of J*j* which represents the end of job type J*j* in *s*th sequence of loading.
P^m_q	Resource place for *q*th machine.
M_i	*i*th machine.
T_i	*i*th machine's tool slot.
S	Sequence of loading
$1 \ldots n$	for job selected for processing.
$(-1) \ldots (-n)$	for jobs rejected.

3.2. *Petri net modelling for loading problems*

Several attempts have been made by various researchers to address the problems in FMS using Petri net models. Lee and Dicesare (1994) formulated a scheduling problem with a Petri net model which employs global search and limits the search space by the use of heuristic functions. Liu and Wu (1993) have used a Petri net

	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}	P_{16}	P_{17}
M_0	1	1	1	0	0	0	1	1	0	1	1	0	0	0	1	1	0
M_1	1	1	0	1	1	0	1	1	0	1	1	0	0	0	1	1	0
M_2	1	1	0	0	1	0	1	1	0	1	1	0	0	0	1	1	0
M_3	1	1	0	0	0	1	1	1	0	1	1	0	0	0	1	1	0
M_4	1	1	0	0	0	0	1	1	1	1	1	0	0	0	1	1	0
M_5	1	1	0	0	0	0	1	1	0	1	1	1	0	0	1	1	0
M_6	1	1	0	0	0	0	1	1	0	1	1	0	1	0	1	1	0
M_7	1	1	0	0	0	0	1	1	0	1	1	0	0	1	1	1	0
M_8	1	1	0	0	0	0	1	1	0	1	1	0	0	0	1	1	1

Table 3. Reachable marking of Petri net for Job 6.

Initial marking	Transitions fired	Final marking
M_0	t^b_{6144}	M_1
M_1	t^e_{6144}	M_2
M_2	t^b_{6244}	M_3
M_2	t^b_{6224}	M_4
M_2	t^b_{6234}	M_5
M_3	t^e_{6244}	M_6
M_4	t^e_{6224}	M_6
M_5	t^e_{6234}	M_6
M_6	t^b_{6324}	M_7
M_6	t^b_{6324}	M_8
M_7	t^e_{6324}	M_0
M_8	t^e_{6314}	M_0

Table 4. Firing of transitions and subsequent markings for Job 6.

model to analyse an FMS to check buffer overflows, to schedule the earliest starting time for a sequence of operation in an FMS and to detect deadlock phenomenon. Reddy *et al.* (1992) applied Petri nets to address the tool management issues at machine group level to get a better insight into the working of the elements that constitutes the tool management. Timed and untimed Petri nets can be used to model the loading problem of an FMS. In timed Petri nets, time can be associated with places or transition to study the dynamic behaviour of the system. However, based on themes and concepts used by the aforesaid researchers, an attempt has been made to construct an untimed Petri net model for a machine loading problem which has been solved by the proposed heuristic.

3.3. *Validation of the Petri net model*

The proposed heuristic for solving the machine loading problem resolves the conflict in the Petri net model. The model is pure, since there exists no place which is an input and output place of same transition. The model is not safe, because the number of tokens in each place exceeds one but it is bounded because there exists an integer x ($X = 8$ for job no. 1) such that the number of tokens in any place cannot

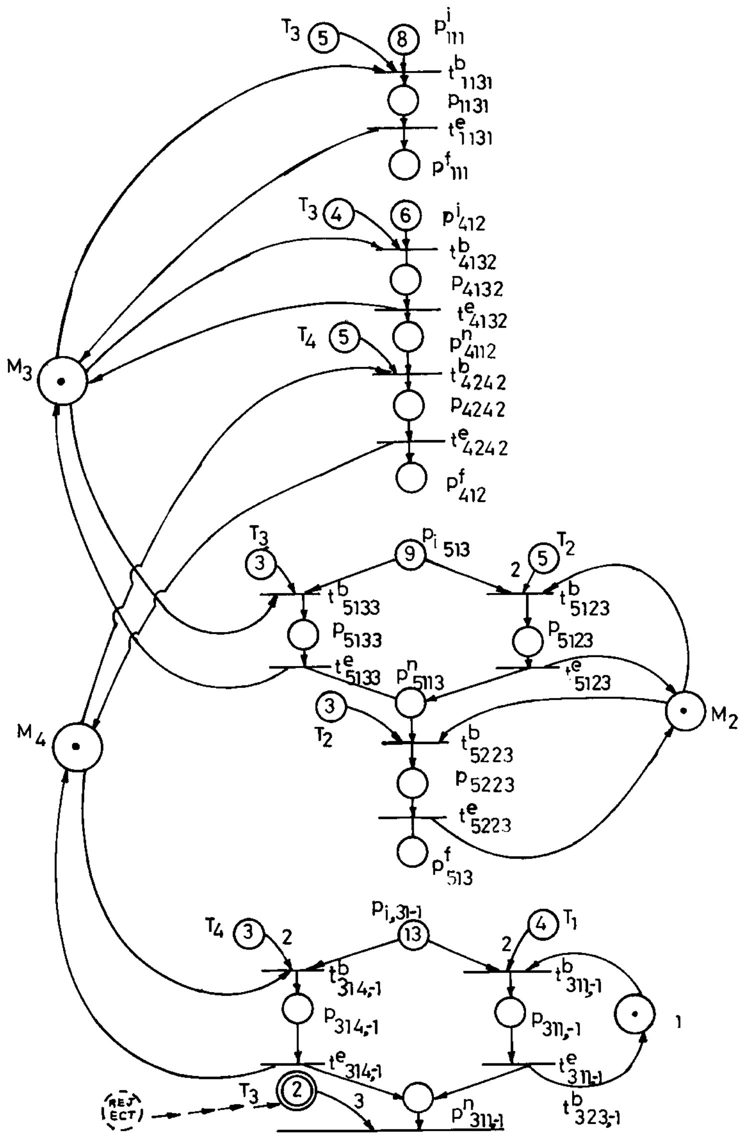

Figure 4. The Petri net model of jobs 1, 4, 5 and 3.

exceed 'X'. Since the Petri net model is bounded, the number of reachable markings are finite. Using the reachability tree we have found that the Petri net is proper i.e. initial marking is reachable from all reachable markings. This implied reinitializability of the system. Conservativeness of the model reveals no buffer overflows in the system. Construction of the reachability tree helped in searching all the possible states and checks the absence of deadlock. Deadlocks are detected when a sequence of transition firing results in a state from which no further transition can be fired. Absence of deadlock implies the liveness of the Petri net model. The Petri net models shown belongs to the class of reversible nets and its reversibility depends upon the number of tokens in the input places.

4. Discussion

The proposed heuristic uses system imbalance as a major criterion to justify the validity of the solution. The maximum available capacity of the system is 1920 min (= 480×4) which is nothing but the maximum possible system imbalance. Therefore, minimizing the system imbalance can also be viewed as maximization of the capacity utilization of the system. In this context, the question related to overloading of machines is to be answered. If the overloading of machines is permitted, the optimum value of system imbalance must be maintained within a limit so that the flexibility of the system could not be compromised. Mukhopadhyay *et al.* (1992) have considered the un-utilized machine time and over-utilized machine time in their system imbalance while Shankar and Srinivasulu (1989) have taken only un-utilized machine time into their system imbalance. In the proposed heuristic, the overloading of the machines is being taken into consideration in the system imbalance. Mukhopadyay *et al.* (1992) have considered the positive value of summation of remaining machine time after allocating the jobs in sequence. Next, the job from the set of unassigned jobs will be considered, when the value of system imbalance (modular value) starts increasing with respect to previous value. This happens even when the actual system imbalance is found negative.

The proposed heuristic considers the allocation of jobs from the set of unassigned jobs until a positive minimum value of system imbalance is achieved. Therefore, the system's flexibility has been restored to a certain extent in our solution. A comparative study of system imbalance and throughput for the problems given in Mukhopadhyay *et al.* (1992) has been carried out and the solutions are given in Table 5.

A Petri net model for a machine loading problem solved by the proposed heuristic has been included in this paper to demonstrate effectively the following points.

(1) A good visual aid to express the interactions of all the variables of the problem.
(2) Dynamic behaviour of a system can be studied at any time.
(3) Deadlocks, conflicts and buffer sizes can be modelled easily.

Entire reachability graph generation for the given loading problem is an infeasible proposition due to the complexities involved. Therefore, a search heuristic algorithm is to be employed for generating that portion of the reachability graph which has been in demand for finding the next allocation of a machine, keeping in mind the technological and capacity constraints.

5. Conclusion

The proposed heuristic employs the backward procedure to maximize the assigned workload and simultaneously restore the flexibility of the system with considerable computational simplicity. It provides a reasonably good solution to the machine loading problems. However, there are certain drawbacks such as neglecting the tool changing time and using a predetermined sequencing rule, tool overlapping etc. The proposed heuristic allocates the operations on machines by taking into account the maximum remaining process time and tool slot constraints of each machine of the system. The Petri net suffers from the problem of state space explosion. Transformation methods can be used to reduce the size of the net, while maintaining the properties of interest. The complexities in developing the entire

		Shankar and Srinivasulu		Mukhopadhyay *et al.*		Proposed heuristic	
Shift number	Total number of jobs	System imbalance	Throughput	System imbalance	Throughput	System imbalance	Throughput
1	8	253	39	122	42	76	42
2	6	388	51	202	63	234	63
3	5	288	63	286	79	152	69
4	5	819	51	819	51	819	51
5	6	467	62	364	76	264	61
6	6	548	51	365	62	314	63
7	6	189	54	147	66	996	48
8	7	459	36	459	36	158	43
9	7	462	79	315	88	309	88
10	6	518	44	320	56	166	55

Table 5. Comparison of the proposed heuristic with the heuristic developed by Mukhopadhyay *et al.* (1992) and Shankar and Srinivasulu (1989).

reachability graph for the machine loading problem demands a heuristic search algorithm which is an interesting field to be investigated further by researchers.

Acknowledgments

The authors wish to express their gratitude to the editor and anonymous referee for their valuable suggestions regarding this paper.

References

Berrada, M., and Stecke, K. E., 1986, A branch and bound approach for machine load balancing in flexible manufacturing systems. *Management Science*, **32** (10), 1316–1335.

Chen, Y. J., and Askin, R. G., 1990, A multiobjective evaluation of flexible manufacturing system loading heuristics. *International Journal of Production Research*, **28** (5), 895–911.

Co, H., Biermann, J. S., and Chen, S. K., 1990, A methodical approach to flexible manufacturing system batching, loading and tool configuration problems. *International Journal of Production Research*, **28** (12), 2171–2187.

Dicesare, F., Harhalakis, G., Proth, J. M., Silva, M., and Vernadat, F. B., 1993, *Practices of Petri Nets in Manufacturing* (London: Chapman and Hall).

Lashkari, R. S., Dutta, S. P., and Padhe, A. M., 1987, A new formulation of operation allocation problem in flexible manufacturing system and computational experience. *International Journal of Production Research*, **25**, 1267–1283.

Lee, D. Y., and Dicesare, F., 1994, Scheduling flexible manufacturing system using Petri nets and heuristic search. *IEEE Transactions on Robotics and Automation*, **10** (2), 123–132.

Liang, M., and Dutta, S. P., 1993, An integrated approach to part selection and machine loading problem in a class of flexible manufacturing systems. *European Journal of Operational Research*, **67**, 387–404.

Liu, C.-M., and Wu, E.-C., 1993, Using Petri nets to solve FMS problems. *International Journal of Computer Integrated Manufacturing*, **6** (3), 175–185.

Mukhopadhyay, S. K., Midha, S., and Murlikrishna, V., 1992, A heuristic procedure for loading problems in flexible manufacturing systems. *International Journal of Production Research*, **30** (9), 2213–2228.

Murata, T., 1989, Petri nets: properties, analysis and applications. *Proceedings of the IEEE*, **77** (4), 541–580.

Moreno, A. A., and Ding, F.-Y., 1993, Heuristics for the FMS loading and part type selection problems. *International Journal of Flexible Manufacturing Systems*, **5**, 287–300.

Peterson, J. L., 1981, *Petri Net Theory and the Modeling of Systems* (Englewood Cliff, NJ: Prentice Hall).

Rajagopalan, S., 1986, Formulation and heuristic solutoins for part grouping and tool loading in flexible manufacturing systems. *Proceedings of the 2nd ORSA/TIMS conference on FMS*, University of Michigan, Ann Arbor, Michigan, USA, **5**, 477–486.

Ram, B., Sarin, S., and Chen, C. S., 1990, A model and solution approach for the machine loading and tool allocation problem in a flexible manufacturing system. *International Journal of Production Research*, **28** (4), 637–645.

Reddy, C. E., Krishanaiah Chetty, O. V., and Chaudhuri, D., 1992, A Petri net based approach for analysing tool management issues in FMS. *International Journal of Production Research*, 1427–1446.

Sarin, S. C., and Chen, C. S., 1987, Machine loading and tool allocation problem in a flexible manufacturing system. *International Journal of Production Research*, **25**, 1081–1094.

Santhikumar, J. G., and Stecke, K. E., 1986, Reducing work-in-process inventory in certain classes of flexible manufacturing systems. *European Journal of Operational Research*, **26** (2), 266–271.

Sawik, T. J., 1989, Production scheduling in a flexible manufacturing system. *Systems Science*, **15** (4), 113–124.

Shankar, K., and Srinivasulu, A., 1989, Some solution methodologies for the loading problem in a flexible manufacturing system. *International Journal of Production Research*, **27** (6), 1019–1034.

Shankar, K., and Tzen, Y.-J. J., 1985, A loading and dispatching problem in a random FMS. *International Journal of Production Research*, **23** (3), 579–593.

Stecke, K. E., and Solberg, J. J., 1981, Loading and control policies for a flexible manufacturing system. *International Journal of Production Research*, **19**, 481–490.

Stecke, K. E., 1983, Formulation and solution of nonlinear integer production planning problem for flexible manufacturing system. *Management Science*, **29**, 3–12.

Stecke, K. E., 1989, Algorithms for efficient planning and operation of particular FMS. *International Journal of Flexible Manufacturing Systems*, **1**, 287–324.

Viswanadham, N., and Narahari, Y., 1994, *Performance Modeling of Automated Manufacturing Systems* (New Delhi: Prentice Hall).