Optimisation of a machine loading problem using a genetic algorithm-based heuristic

Shrey Ginoria and G.L. Samuel*

Manufacturing Engineering Section,
Department of Mechanical Engineering,
Indian Institute of Technology Madras,
Chennai – 600 036, India
Fax: +91-44-22575705
Email: ginoria.shrey@gmail.com
Email: samuelgl@iitm.ac.in
*Corresponding author

G. Srinivasan

Department of Management Studies,
Indian Institute of Technology Madras,
Chennai – 600 036, India
Email: gsrini@iitm.ac.in

Abstract: In the present work, apart from operating on the structure of a conventional genetic algorithm (GA), a heuristic which uses techniques like differential mutation probability, elitism and local search is used to produce near optimal solutions for large machine loading problems with less computational intensity. Two variants of the machine loading problem are analysed in the present work: single batch model and the multiple batch models. The sensitivity of the problem with respect to the tool capacity constraint is evaluated to find that moderately restricted problems requiring greater computational resources in comparison to lesser restricted and tightly restricted class of problems. The performance of various dispatching rules was compared to infer that the least slack principle fares better than the other tested dispatching rules. It is observed from the results, that the proposed heuristic is efficient in handling large and complex machine loading problems.

Keywords: flexible manufacturing system; FMS; machine loading; load balancing; tool savings; genetic algorithm; branch and bound.

Reference to this paper should be made as follows: Ginoria, S., Samuel, G.L. and Srinivasan, G. (2015) 'Optimisation of a machine loading problem using a genetic algorithm-based heuristic', *Int. J. Productivity and Quality Management*, Vol. 15, No. 1, pp.36–56.

Biographical notes: Shrey Ginoria holds a Bachelor degree in Mechanical Engineering and an MTech in Intelligent Manufacturing (dual degree) from Indian Institute of Technology Madras, Chennai.

G.L. Samuel is currently working as an Assistant Professor at the Department of Mechanical Engineering, Indian Institute of Technology Madras, Chennai. He obtained his BE in Mechanical Engineering from Mysore University in 1991, MTech in Production Engineering and Systems Technology from

Kuvempu University in 1994 and PhD in Manufacturing Engineering from Indian Institute of Technology Madras in 2001. He is a Post-Doctoral Fellow at the School of Mechanical Engineering, Kyungpook National University, South Korea. His active areas of research are: measurements and inspection of freeform surfaces, geometric error compensation in machine tools, evaluation of form errors, micromachining and laser vision systems. He has published over ten papers in refereed international journals.

G. Srinivasan is a Professor in the Department of Management Studies, Indian Institute of Technology – Madras (IIT Madras). He did his BE (Hons.) in Mechanical Engineering in Anna University. He did his MS by Research in Industrial Engineering and PhD in Industrial Management in IIT Madras. He was awarded the BOYSCAST Fellowship of the Ministry of Science and Technology, Government of India for advanced training in total quality management. His areas of interest include fundamentals of operations research, advanced operations research, operations management, supply chain management, manufacturing systems management, services operations management and OR applications.

1 Introduction

A flexible manufacturing system (FMS) is an integrated manufacturing system which consists of a network of multi-functional numerically controlled (NC) machine tools, each with automatic tool changing capabilities and automated material transfer systems. The FMS systems are widely applied in industrial practices because it provides the flexibility of a low volume – high variety manufacturing along with the efficiency of a high volume – low variety manufacturing. As noted by Graves (1981), it involves complex production planning level decisions to utilise the system in an optimal and efficient manner. The conditions for the efficient working of an FMS can be maintained by partaking a lot of important control decisions such as part type selection, machine grouping, production ratio planning, work-in-process inventory planning, machine loading, etc. (Hutchison, 1991).

Several researchers have studied one or more of the important decision making problems in FMSs. Chan et al. (2005) developed a fuzzy goal programming model with an artificial immune system for machine tool selection and operation allocation problem in FMSs. Kim et al. (2003) have addressed the tool requirements planning problem in a FMS. The objective is to minimise tool cost subject to a makespan constraint. They develop four heuristics that start with an infeasible solution and increase the number of tool copies till the makespan constraint is satisfied. Computational tests on randomly generated problems indicate that good solutions can be obtained to the problem within reasonable computational time. Atmani and Lashkari (1998) present a zero one integer programming model for machine tool selection and operation assignment in FMS. A mathematical model which, through constraint optimisation principles, is able to find the optimal distribution of workforce optimising fundamental parameters, such as man-hours, throughput, makespan and work in process was proposed by Gilles and Matteo (2009).

Stecke (1983) discussed the formulation and solution of non-linear integer production planning problems for FMSs. The author listed five important decision making problems in FMS. These are part selection problem, machine grouping problem, production ratio problem, resource allocation problem and machine loading problem. Among the above mentioned problems, it has been found that two problems – part type selection and machine loading – are crucially important for the efficiency of an FMS (Gershwin, 1994). It has also been shown that optimising a machine loading problem automatically maximises the system utilisation, maximises the expected throughput and minimises inventory levels (Shanthikumar and Stecke, 1986).

The machine loading problem can be defined as "assignment to the machines of the operations of the selected part types and the tools necessary to perform these operations, subject to the technological and capacity constraints and according to some loading objective, in a way that will best utilize the machines, or maximize production, when the system is running" (Hwang, 1986). The machine loading problem has been studied in great detail in the recent past. Berrada and Stecke (1986) investigated the problem for several different objectives for a system containing equal sized groups of pooled machines. Shanker and Srinivasulu (1989) considered the loading problem for a non-stationary part mix which could change dynamically during the running time of the system.

2 Literature review

In the past, the loading problem has traditionally been solved by breaking it into three parts: job sequence determination, allocation of jobs to machines and reallocation. However, Liang and Dutta (1993) suggested an integrated approach which undertook these three steps collaboratively and it was shown to produce better results. Owing to the flexibility of an FMS, there are several objectives which are of equal importance while solving a machine loading problem. As Tiwari and Vidyarthi (2000) suggest, the prominent ones among them are workload balance, inter-cell movements, tool changeovers, etc. The machine loading problem is a known hard problem. Hence, various scheduling techniques have been developed to tackle the ever increasing complexity and flexibility of manufacturing systems. The optimisation techniques proposed in the literature include: linear programming (Nascimento, 1993) goal programming (Hoitomt et al., 1993), dynamic programming (Song et al., 1995), branch and bound (Kuhn, 1995) and Lagrangian relaxation (Mukhopadhyay et al., 1998). The approximation techniques include priority rule-based heuristics (Sarma et al., 2002); local search algorithms – iterative search, simulated annealing (Kopfer and Mattfield, 1997), memetic algorithm (Wisut and Karn, 2011), tabu search (Goldberg and Korb, 1989), etc. Ali and Sangari (2010) developed a new algorithm using a simulated annealing, to find the optimal inspection strategy for a serial multistage process. Valente (2007) proposed several dispatching heuristics for the single machine scheduling problem with linear earliness and quadratic tardiness costs, and no machine idle time. Ponnambalam and Kiat (2008) address the machine loading problem in FMSs using particle swarm optimisation. They addressed a bi criteria problem of minimising system imbalance and maximising throughput considering technological constraints on machine availability and tool slots. The test their proposed algorithm on ten problems from the literature and conclude that the proposed algorithm provides good solutions.

Tiwari and Vidhyarthi (2000) addressed the machine loading problem in FMS that minimises system imbalance and maximises throughput considering technological constraints on machine availability and tool slots. They test their proposed genetic algorithm on ten randomly generated problems. They also indicate that rules other than SPT can give good performance with respect to loading problems in FMS. The proposed GA has been tested on small sized problems with up to eight jobs.

A special mention is necessary here for the branch and bound technique developed by Berrada and Stecke (1986). A sequence of sub-problems are defined; each sub-problem being solved by an efficient branch and bound technique and the resultant solution being modified to satisfy any violated constraints by solving multiple very small integer problems via the branch and backtrack procedure. Moslehi et al. (2010) used an efficient lower and upper bounds and new dominance rules, a branch-and-bound scheme for solving problem of scheduling *n* jobs on a single machine to minimise the sum of maximum earliness and tardiness. This approach has been demonstrated to be an optimal scheduling technique for a single batch problem irrespective of its nature and variety. However, the only drawback of this technique is that it can handle problems of only a moderate size and fails to deal with problems of greater size. Optimisation techniques are unable to handle greater sized problems. On the other hand, most of the approximation techniques are shown to be very specific for a particular class of problems and cannot be applied universally across the different classes.

Several search and artificial intelligence techniques like reinforcement learning, artificial neural networks, Petri net model, etc., were proposed for optimisation of machine loading problems (Tiwari et al., 1997). Search algorithms are primarily of two types: local search and evolutionary. Local search-based heuristics are known to produce good results in short computational times but they run the risk of being caught in local critical points. Evolutionary algorithms can be formulated independently depending on the nature of the problem and its performance is heavily dependent on the formulation. Maheswaran et al. (2008), proposed hybrid heuristic algorithms based on dynamic dispatching rule, greedy heuristic (backward phase) and backward phase heuristics with an iterated local search to solve the single machine scheduling problems with the objective of minimising the total weighted tardiness. Genetic algorithms can be considered to be a hybrid of the two previous techniques as it applies a local search operator in an evolutionary framework. They combine the advantages of both, efficiency of the local search and robustness of the evolution (Co et al., 1998). Although instances of GA being used to solve a flow shop or a job shop are abundant, applications are relatively scarcer when compared to a parallel machine scheduling problem (Dar-El and Sarin, 1984). Min and Cheng (1999) proposed a GA for solving the identical parallel machine scheduling problem with the objective of minimisation of makespan. Cheng et al. (1995) used genetic algorithms to solve the same problem for minimising the maximum weighted absolute lateness. McCormick and Pinedo (1995) formulated a GA for scheduling with the dual objectives: flow time and makespan. Figielska (1999) devised a GA integrated with column generation technique to solve parallel machine scheduling for the purpose of minimising the makespan and total cost of changeovers.

A large body of literature has been found for solving the parallel machine scheduling problems based on the objectives of makespan and flow time, but there is a relative dearth of work which considers the workflow balancing as its main objective. Also, these techniques are unable to model the dynamic and stochastic nature of the manufacturing systems and take extensively long computing time required for the machine to gain intelligence.

The optimisation and approximation techniques mentioned above are adept at solving moderately sized machine loading problems (Viswanadham and Narahari, 1994).

However, with the evolution of manufacturing environment, an FMS is now expected to handle a far greater number of jobs. In such situations, the machine loading problem can be immensely complex with the addition of multiple batch possibilities, tool savings and tool capacity constraints. Hence, there is a growing need for techniques which would be able to solve such problems of growing size with relative ease. Taking a cue from these observations, the present work attempts to develop a systematic integrated approach to address the entire machine loading problem simultaneously. The genetic algorithm-based heuristic proposed in the present work is capable of solving complex and large sized machine loading problems in a FMS. In the present work, issues such as load balance, system tardiness and material wastage were addressed. Load balance is the primary objective because with the huge capital investment needed for the installation of an FMS, the management expects the system utilisation to be at a maximum which can be easily achieved by the balancing of workloads. Moreover, low tardiness levels facilitate timely supply of orders and help the organisation maintain good customer relations and build an esteemed brand value. Similarly, minimisation of overall material wastage has a directly proportional relationship with the operational costs of the system.

3 Problem definition

A class of problems of scheduling n independent jobs $N = \{1, 2, 3 \ldots\ldots n\}$ on m similar parallel machines $M = \{1, 2, 3 \ldots \ldots m\}$ with the purpose of optimising a predefined criterion is considered in the present work. This class of problems is typically known as parallel machine scheduling problem.

The binary variable x_{ij}, is defined as $x_{ij} = 1$, if job i is operated on machine j. $x_{ij} = 0$, if job i is not operated on machine j. $\sum_j x_{ij} = 1$, to ensures that one job gets processed on only one machine and one machine cannot operate on more than one job at a time. p_{ij} is the time taken by machine j to complete the operation of job i. If few jobs are incompatible on certain machines, x_{ij} would always be 0 for such job machine pairs. These job-machine pairs would form the incompatibility matrix. Also, it is assumed that, all the jobs are available for processing at the start time of the process. Moreover, a job has to complete its processing once it has been loaded on a machine and no pre-emption of jobs is allowed. Present work would mainly deal with the cases of parallel similar unrelated machines. They are more in tune with the industrial reality in comparison with the others as the different machines might have been bought at different times, operating on different technological principles, thus giving rise to independent processing times.

3.1 Tooling constraints

Each machine would have a tool magazine with a finite capacity. The tool capacity of a machine is represented by t_j and is defined by the number of slots present in its tool magazine. Each slot can hold one individual tool. If a job has been allotted to a particular machine, the tool magazine of that machine should house all the tools corresponding to the job, failing which the processing of the job would not be possible. Each job requires the same group of tools irrespective of the machine on which it is processed. If several jobs have been allocated to a particular machine and some of the tools required are common to a few jobs, it is sufficient for the machine to store a single copy of the common tools and not store duplicate copies of the same tool. This leads to significant tool capacity savings for a machine if the allocated jobs share a large number of common tools.

3.2 Single and multiple batch problems

Although a machine can only process one job at a time, multiple jobs can be allocated to a single machine. However, the tool magazine of a machine may house a greater number of tools which is sufficient for processing of several jobs. In that case, the machine can process several jobs sequentially without undergoing any tool changeover as it already houses all the tools required for processing the different jobs. Hence, the group of jobs which can be processed by a machine without undergoing any tool changeover is called a 'batch' of jobs. The parallel machine scheduling problem can operate in two modes: single batch or multiple batches. In the single batch model, a machine can process only one batch of jobs while in the multiple batches model, a machine can handle more than one batch of jobs. Generally, industrial practices favour the multiple batch model as it is less capital intensive due to the lesser number of machines required and an improved percentage of machine utilisation.

In a single batch model, the tool capacity of the machine should be such that it should be able to house the tools necessary for the processing of all the jobs allocated to the machine. Tool capacity of a machine, t_j should always be greater than or equal to number of tools required to process all allocated jobs in one batch. In a multiple batch model, a machine is allowed to undergo any number of tool changeovers. Hence, in case the machine is incapable of processing all the allocated jobs in one batch due to tool capacity constraints, it can process the initial set of jobs, undergo a tool changeover to change the tools in its magazine and then proceed to do the remaining jobs.

3.3 Objective functions

The following three objectives: load balance, total tardiness of the system and total material wastage in the system are used as objectives for optimisation.

3.3.1 Load balance

The makespan of each machine is defined as the completion time of the last job processed on it. It is represented by c_j. Several definitions of load balance exist including minimising the difference between the maximum and minimum loads or between the maximum and the average of the loads. In this paper, we minimise the maximum of the makespan of all machines.

3.3.2 Tardiness

Each job has a due date associated with it which represents the deadline by which it is expected to have been released from the system. In case the job is delayed beyond its due date, it is said to be tardy and the tardiness of the job is represented by:

$$t_j = \max\left(0, C_i - d_i\right) \tag{1}$$

where C_i is the release time or completion time of the job and d_i is the due date of the job.

The total tardiness of the system is represented as:

$$T = \sum_i ti \tag{2}$$

3.3.3 Material wastage

Each job is associated with a parameter which defines the length of the job. When more than one job is processed together in a batch, all the jobs are placed together on a sheet (considered to be one dimensional) and the leftover length of the sheet is considered to be wastage. Thus, the total length of the sheet wasted across all the batches put together is considered as the material wastage of the system.

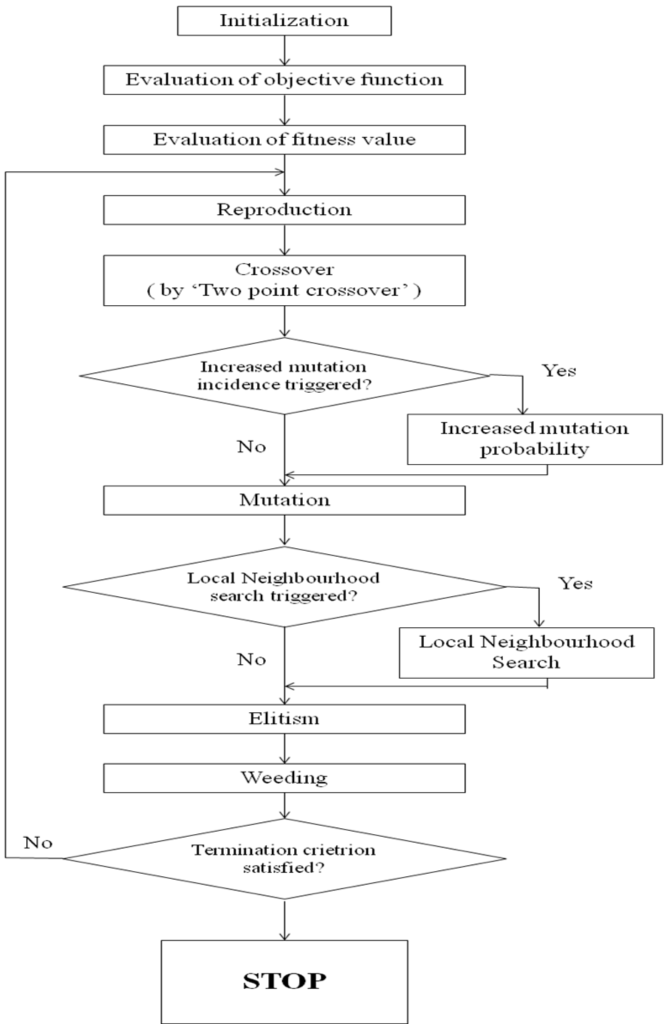
4 Genetic algorithm-based heuristic

A genetic algorithm is an intelligent probabilistic search algorithm that simulates the process of evolution by taking a population of solutions and applying genetic operators in each reproduction. Each solution in the population is accorded a fitness value by evaluating it according to some performance measure. The population is operated on by three main genetic operators: reproduction, crossover and mutation. The reproduction is a process in which the individual strings are copied according to their fitness values which results in more highly fit chromosomes and less weak chromosomes in the intermediate mating pool. During the crossover operation, the chromosomes used for mating are selected through the roulette wheel selection strategy. Mutation is practiced to avoid the solution from getting stuck in a local optimum. One cycle of these genetic operations and the evaluation procedure is known as a generation in GA terminology. The cycle of reproduction – crossover – mutation – selection is continued until the termination criterion is met. Apart from the above described structure of a conventional GA, other techniques such as elitism, local neighbourhood search and increased mutation incidence are used in the present work to produce comparatively better results. The parameters used in the algorithm are given in Table 1 and the steps involved in the proposed algorithm are shown in Figure 1.

Table 1 Parameters used in the genetic algorithm

Sl. no.	*Parameter*	*Value*
1	Population size	50 to 250
2	Mutation probability	0.01
3	Local neighbourhood search threshold	Gets triggered after 25 successive generations of no improvement
4	Increased mutation incidence threshold	Gets triggered after 50 successive generations of no improvement
5	Stagnation threshold	200
6	Maximum number of generations	1,000

Figure 1 Various steps in the proposed genetic algorithm


Initialization
Evaluation of objective function
Evaluation of fitness value
Reproduction
Crossover
(by 'Two point crossover')
Increased mutation incidence triggered?
Yes
Increased mutation probability
No
Mutation
Local Neighbourhood search triggered?
Yes
Local Neighbourhood Search
No
Elitism
Weeding
No
Termination crietrion satisfied?
STOP

Pheno style coding is used and hence the value of the *i*th gene of the chromosome represents the machine on which the *i*th job is to be processed. After the determination of the objective functions for all the chromosomes of a generation, a fitness value is calculated for each chromosome which is representative of its degree of 'fitness'. The entire step by step process of the calculation of fitness values is described below.

Step 1 All the chromosomes of the population are arranged in an ascending order and are given ranks according to their relative.

Step 2 Rank value of a chromosome is calculated as:

$$\text{Rank Value} = 1 - (\text{Rank} / \text{Population Size})$$

Step 3 Performance value of a chromosome is calculated as:

$$\text{Performance Value} = \frac{\text{Objective function of the best chromosome of the population}}{\text{Objective function of the chromosome}}$$

Step 4 Goodness value of a chromosome is calculated as:

$$\text{Goodness value} = \text{Rank Value} + \text{Performance Value}$$

Step 5 Finally, the fitness value of a chromosome is calculated as:

$$\text{Fitness Value of a chromosome} = \frac{\text{Goodness value of the chromosome}}{\sum \begin{array}{l}\text{Goodness value of all}\\ \text{chromosomes of the population}\end{array}}$$

Table 2 Calculation of fitness values of chromosome

Chromosome	*Objective value*	*Rank*	*Rank value*	*Performance value*	*Goodness value*	*Fitness value = (Goodness value / 4.35)*
A	40	2	0.6	0.5	1.1	*0.25*
B	50	3	0.4	0.4	0.8	*0.18*
C	20	1	0.8	1	1.8	*0.41*
D	100	5	0	0.2	0.2	*0.05*
E	80	4	0.2	0.25	0.45	*0.10*
Total					*4.35*	*1*

Table 2 presents the calculation of fitness values for a given set of chromosomes. In the present work, the fitness proportional selection, known as roulette wheel selection is used to choose chromosome for various genetic operations like reproduction, crossover, mutation, weeding, etc. After evaluating various crossover techniques such as partial mapped crossover, edge recombination operator, etc., it was found that the two point crossover technique was the best suited for this genetic algorithm design and it is implemented in this work. In the mutation technique used in the proposed algorithm, firstly, a random number between zero and the chromosome length is generated. This random number represents the number of genes of that chromosome which will undergo mutation. The genes to be mutated are selected randomly. The value of the genes is now changed from the present value to a random value between zero and the number of available machines.

In case the best solution of the population has not shown any improvement for a number of successive generations, the process is triggered for the best five chromosomes of the population. The value of one random gene is changed randomly to one of the available machine and this new chromosome is fed into the population. If the best solution of the population does not show any improvement for a number of successive generations, the mutation probability for the next generation increases to an appreciably higher level thus encouraging a larger number of chromosomes to mutate. This promotes a higher amount of diversity in the population which helps the heuristic to break out of the local optima.

To ensure that the best chromosomes of a population are not lost during the selection strategy for the next generation, the best 1% chromosomes of the population are always transferred to the next generation as a rule. The algorithm is terminated when either of the two conditions is satisfied:

- the process has reached the pre-defined limit for the maximum number of generations
- the best chromosome of the population has not shown an improvement for a number of successive generations which is greater than or equal to the stagnation threshold.

Upon termination, the best chromosome of the current population is presented as the near optimal solution of the problem.

5 Results and discussions

In the present work the branch and bound technique developed by Berrada and Stecke (1986) is used to provide a benchmark for a given problem. The basic scheme of the branch and bound techniques is to devise the problem as a series of sub-problems whose solutions would eventually converge to the optimal solution. Each of these sub-problems have a δ_l associated with them and the objective of the sub-problem is to find an allocation of jobs which results in a load balance which is less than δ_l. δ_l is fixed as the midpoint of the interval between the upper bound (UB) and the lower bound (LB) of the problem. These bounds are revised with the solution of each sub-problem in the following manner: initially, UB is set at α while LB is set at 0. If we are able to find an optimal solution to the sub-problem, the bounds get revised according to solution otherwise LB is updated to δ_l while UB remains the same. It is observed that for large and complex problems the branch and bound technique fails to give results within a reasonable CPU time.

5.1 Single batch model

The two techniques have been run for a variety of single batch model problems of different sizes and load balancing is the prime objective which is evaluated.

Table 3 Computational resources required by the branch and bound technique

Sl. no.	*Size*	*Time (in seconds)*		*Number of nodes evaluated*	
		Average	*Standard deviation*	*Average*	*Standard deviation*
1	10 × 3	22.4	3.9	443.8	49.9
2	15 × 4	80.8	9.9	3,521.4	367.6
3	20 × 5	144.1	19.7	15,359.6	1,734.8
4	25 × 6	333.3	36.7	41,646.6	5,011.3
5	30 × 7	621.5	64.1	122,627.4	11,264.7
6	35 × 8	1,182.2	152.8	220,092.8	25,053.6
7	40 × 9	1,738.5	224.2	446,836.8	36,829.2

The branch and bound technique provides optimal solution to the single batch model of a load balancing problem. However, the computation resources required by the technique rise exponentially as the size of the problem increases. This has been illustrated by the following set of graphs and tables. Computational time required by the branch and bound technique to solve seven problems each of different sizes is listed in Table 3. It is observed that, though the branch and bound technique gives optimal solution, the time required increases exponentially as the number of nodes that need to be evaluated increases with increase in size of the problem. The problems have been evaluated on a computer processor of frequency 2.4 GHz. For documentation purposes, the individual values of the time taken and the number of nodes evaluated by the branch and bound technique for the sizes 10 jobs × 3 machines is listed in Table 4.

Table 4 Computational requirements for 10 × 3 sized problems

Problem no.	*Time (in seconds)*	*No. of nodes evaluated*
1	26	410
2	17	354
3	20	440
4	23	456
5	21	497
6	23	423
7	18	406
8	30	439
9	21	521
10	25	493
Total		*4,438*

5.1.1 Comparison of performance of GA heuristic with the branch and bound technique

This section evaluates the degree of near optimality of the solutions given by the GA-based heuristic. The optimal solution used for comparison is evaluated through the branch and bound technique. An identical set of five problems is evaluated by both of the techniques; the GA-based heuristic and the branch and bound technique. Henceforth, the solutions given by the GA-based heuristic are compared with the optimal solution to get a perspective on their efficacy.

Table 5 Average percentage deviation of GA-based heuristic from optimal solution

Size of the problem	*Average percentage deviation of GA-based heuristic's solution from the optimal solution*
10 × 3	2.70 %
15 × 4	5.3 %
20 × 5	9.4 %
25 × 6	13.1 %
30 × 7	8.5 %
35 × 8	11.8 %
40 × 9	10.5 %

The average percentage deviation of the solution to that obtained using branch and bound technique are presented in Table 5. As the table suggests, the solutions obtained from the GA-based heuristic do not deviate far off from the optimal solution of the problems. This study substantiates the claim of the GA-based heuristic to give 'near optimal' solutions to such problems.

5.1.2 Robustness of the GA-based heuristic

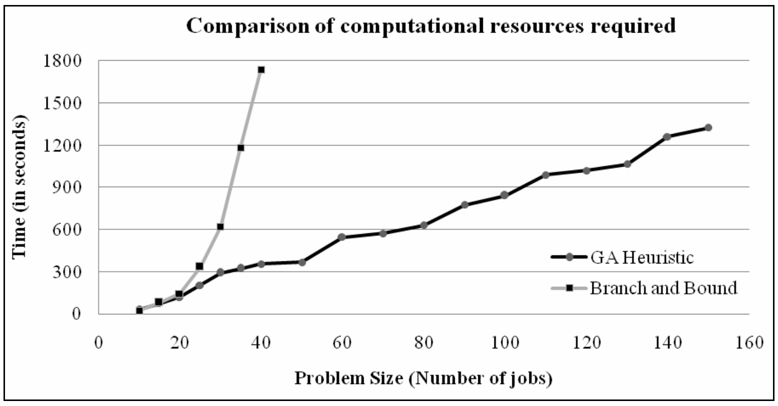
This section analyses the robustness of the GA-based heuristic in terms of the computational resources required while solving problems of larger sizes. The GA-based heuristic is run for ten different problems of various sizes, the sizes ranging from 10 jobs × 3 machines to 150 jobs × 20 machines.

Table 6 Average time required by GA-based heuristic to solve a problem

Problem size	*Average time (in seconds)*	*Problem size*	*Average time (in seconds)*
10 × 3	33	80 × 13	631
20 × 5	118.6	90 × 14	775.6
30 × 7	291.5	100 × 15	840.6
40 × 9	354.7	110 × 16	989.1
50 × 10	367.2	120 × 17	1,019.2
60 × 11	546.2	130 × 18	1,066.1
70 × 12	573.2	150 × 20	1,261.3

Several trials were conducted to establish the efficiency of the proposed method for large size problems and the results are reported on Table 6 and shown graphically in Figure 2. Table 6 and the graph establish that the GA-based heuristic is far less computationally intensive than the branch and bound technique and can easily be used to solve problems of greater sizes with near optimal results.

Figure 2 Comparison of the time required by GA-based heuristic and branch and bound technique to solve problems of different sizes



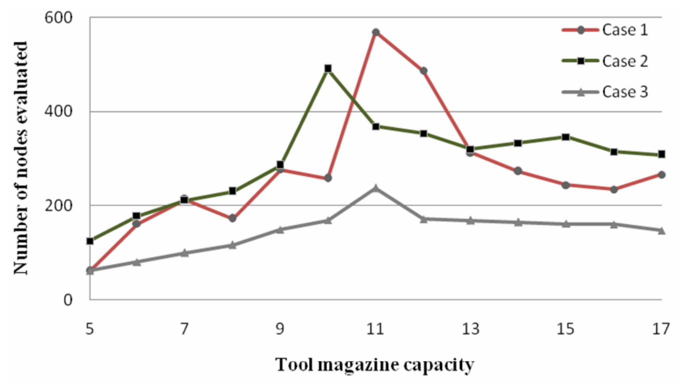
5.1.3 Sensitivity of the branch and bound technique

This section discusses the sensitivity of the branch and bound technique with respect to the tool capacity constraints. The effect of the constraints on the computational resources required by the technique is demonstrated by a series of graphs.

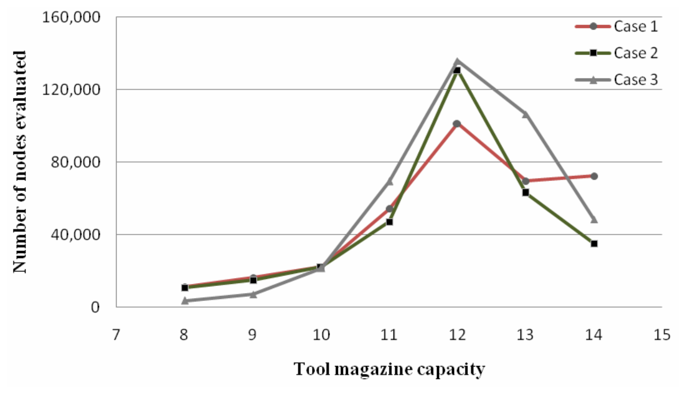
The class of single batch – parallel machine scheduling problems can be segmented into three divisions on the basis of their tool capacity constraints: loosely restricted, moderately restricted and tightly restricted. Loosely restricted problems are such that the tool magazines have a relatively greater capacity and hence, the tool capacity constraint lends a greater degree of freedom and does not have much bearing on the solution. Tightly restricted problems are such that the tool magazines have a relatively lesser capacity and hence very often, the problem is rendered infeasible due to the over bearing tightness of the tool capacity constraints. Moderately restricted problems are the ones which do not fall into either of the categories and operate in a regular manner.

The computational resources required to solve a problem depend significantly on the 'restrictive' class of the problem. This can be demonstrated by evaluating a series of problems for which the entire problem statement is the same but for the tool capacity constraint. The computational resources required for solving the different problems as we gradually increase the tightness of the tool capacity constraint are observed. This analysis is done for each of the various sizes and the results are depicted below in the form of Figure 3(a) to Figure 3(c). For each size, this experiment is repeated thrice thus constituting three different cases for each size.

Figure 3 Sensitivity of branch and bound technique for problems of different sizes, (a) problems of size 10 × 3 (b) problems of size 25 × 6 (c) problems of size 25 × 6 (see online version for colours)

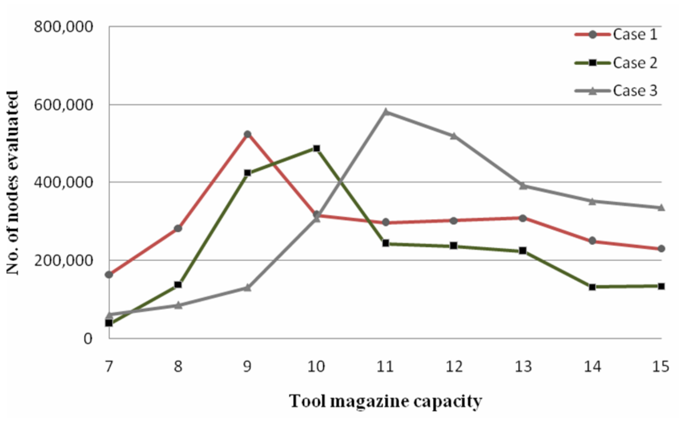


(a)



(b)

Figure 3 Sensitivity of branch and bound technique for problems of different sizes, (a) problems of size 10 × 3 (b) problems of size 25 × 6 (c) problems of size 25 × 6 (continued) (see online version for colours)



(c)

From the above graphs, it can be inferred that while the 'loosely restricted' and the 'tightly restricted' class of problems take relatively lesser amount of computational time than the 'moderately restricted class' thus generating a rough bell shaped curve. This can be attributed to the fact that in the 'loosely restrictive' class of problems, the tool capacity constraint does not play much of a role in the solution does acting as a virtually non-existent constraint while in the 'tightly restricted' class of problems, due to the over bearing tightness of the constraints, the technique is left to evaluate only a handful of feasible solutions as most of the possible allocations are rendered infeasible due to the overt tightness.

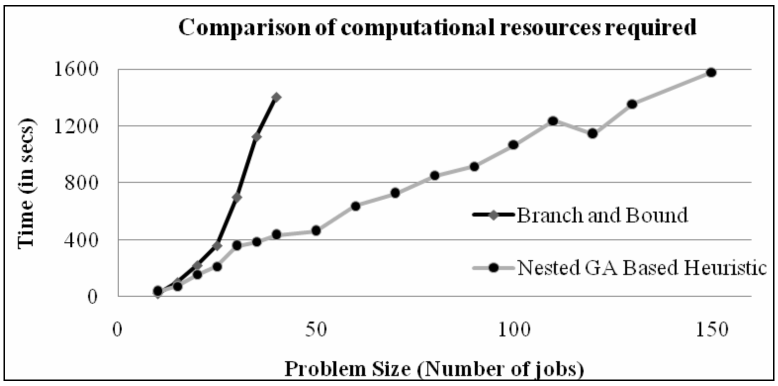
5.2 Multiple batch model

This section of the report analyses the performances of the different heuristics while evaluating multiple batch parallel machine scheduling problems. It also compares the various dispatching rules that can be used for sequencing in multiple batch problems and tries to evaluate the nature of the relationship between the different objectives relevant to the multiple batch model such as load balancing, tardiness, sheet wastage, etc.

5.2.1 Computational requirements of different techniques used

The branch and bound technique and the GA-based heuristic are modified in order to be applicable to the multiple batch model. While the computational time required for the branch and bound technique is roughly comparable to the time required for the single batch model, the time required for the GA-based heuristic increases to a small extent due to the presence of a nested GA algorithm which requires comparatively greater computational resources.

Figure 4 Comparison of computational time required by branch and bound technique and GA-based heuristic for multiple batch models

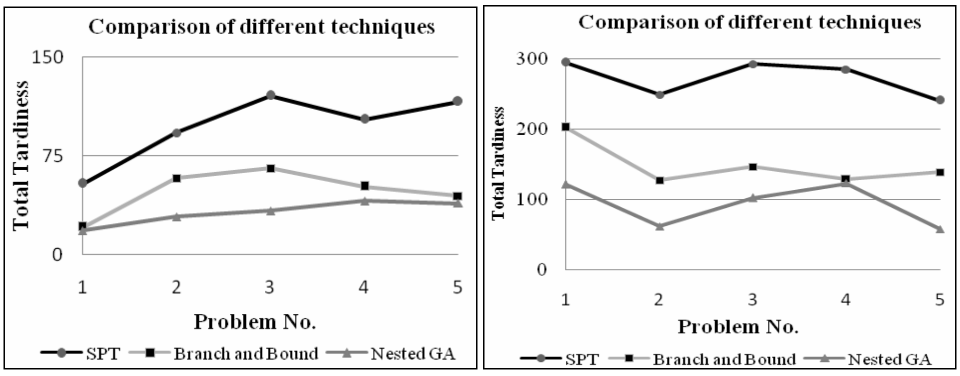


While making a comparison between the computational times required by the two heuristics (Figure 4), it is observed that while the time for the branch and bound technique rise exponentially with the increase in the problem size, the time for the GA-based heuristic increases linearly, thus making it suitable for evaluation of larger sized problems. A graphical comparison between the timings of the two heuristics is presented below.

5.2.2 *Comparison of performance of different techniques*

The efficacy of the solutions given by the different techniques while solving the multiple batch model problems are compared in this section. The comparisons are made by solving an identical set of five problems using the three different dispatching rules; shortest processing time (SPT) deterministic heuristic, branch and bound technique and the GA-based heuristic to allocate jobs to the various batches. It is observed that the SPT heuristic fares poorly in comparison to the branch and bound technique and the GA-based heuristic. The graphs comparing the solutions given by different heuristics for different problem sizes are shown in Figure 5.

Figure 5 Comparison of performance of various techniques while solving problems of different size, (a) problems of size 10 × 3 (b) problems of size 40 × 9



(a) (b)

Moreover, it is also seen that the nested GA-based heuristic is giving comparatively better solutions than the branch and bound technique unlike the single batch model where the latter was seen to be giving the best solution. This is because of the presence of the nested GA which uses a genetic algorithm to determine the sequencing of the jobs allocated to a particular machine while the branch and bound technique uses a pre-defined deterministic rule, in this case the earlier due date principle, to determine the sequencing. Table 7 lists the average improvements of nested GAs solution when compared to the branch and bound's solutions. It should be noted here that the solution obtained by the branch and bound technique is a heuristic solution and is used only for the purpose of comparison of performance.

Table 7 Average percentage improvement of nested GA heuristic over branch and bound

Problem size (number of jobs)	*Average percentage improvement of nested GA heuristic over branch and bound*
10	28.8%
15	25.9%
20	30.2%
25	39.1%
30	36.9%
35	35.4%
40	40.2%

5.2.3 Comparison of performance of various dispatching rules

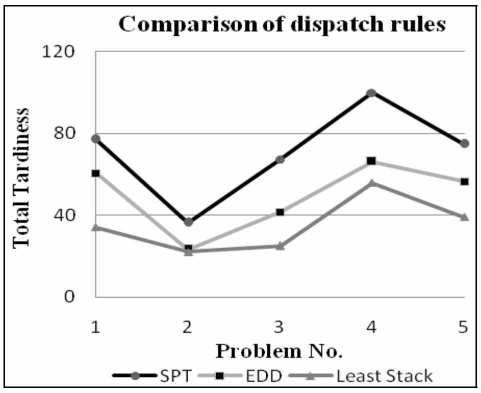
A dispatching rule is the pre-determined rule which determines the sequence in which the allotted jobs are fed to a particular machine. This section compares the performance of a few dispatching rules by evaluating the same problem individually for the different dispatching rules and comparing their results. The three dispatching rules being evaluated here are:

- earliest due date (EDD)
- SPT
- least slack.

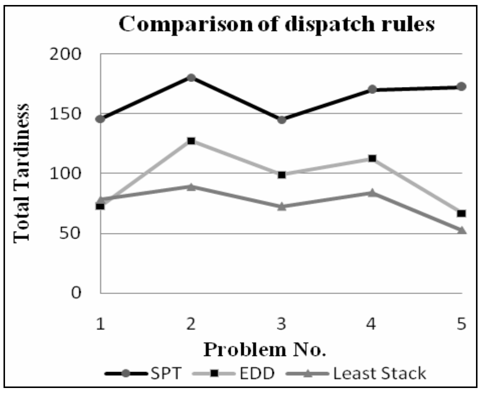
We evaluate five different problems for the five different sizes (30 jobs, 60 jobs, 90 jobs, 120 jobs, 150 jobs). Each problem is evaluated separately for the three different dispatching rules and the results are presented in Figure 6(a) to Figure 6(c).

It is observed that the SPT dispatching rule does not perform well in comparison to the other two rules. Moreover, it is seen that the least slack rule performs marginally better in a majority of the cases. The average improvements of the EDD rule and least slack rule over the SPT rule for the various sizes are listed in Table 8.

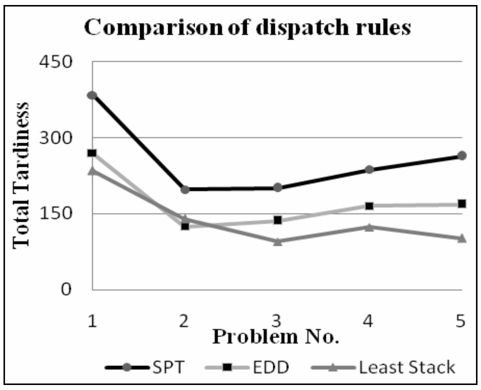
Figure 6 Comparison of performance of dispatch rules for problems of different size, (a) 30 job problems (b) 60 job problems (c) 90 job problem



(a)



(b)



(c)

Table 8 Improvement of performance of dispatching rules over SPT rule

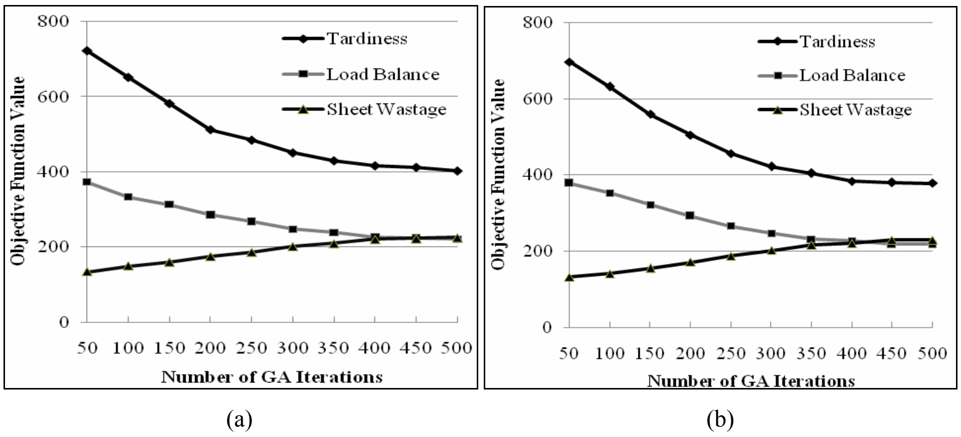
Problem size (number of jobs)	*Average improvement over SPT rule by*	
	EDD rule	*Lease slack rule*
30	30.9%	50.0%
60	41.2%	53.4%
90	33.1%	46.1%
120	34.3%	50.9%
150	37.3%	48.0%

5.2.4 Relationship among different objectives

Tardiness, load balancing and material wastage are the few objectives which are of prime importance in an industrial scenario. It is observed that the total tardiness of the system and the load balancing are non-conflicting objectives and share a directly proportional relationship. Moreover, material wastage and tardiness of a system turn out to be conflicting objectives and have an inversely proportional relationship.

The relationship between the various objectives of a system is evaluated in the genetic algorithm-based heuristic by tracking their values through the different generations. The values of the three above mentioned objectives are recorded at intervals of 50 generations and the recorded values are plotted in a graph, given in Figure 7, to understand the relationship in a better manner.

Figure 7 Relationship among different objectives for different problems, (a) 120 job sized problem (b) 150 job sized problems



It is observed that load balancing and tardiness of the system are non-conflicting objectives. We see that as the generations progress, both the load balance and the tardiness decrease simultaneously. This is because of the fact that if the load balance decreases, it implies that the entire load of the system is being distributed in an even manner across the present machines. This ensures that the jobs are released at relatively earlier times because in this case, jobs are not stuck in machines which are loaded heavily in comparison to other machines, thus increasing the overall tardiness of the system.

Moreover, it is also seen that material wastage and tardiness of a system are conflicting objectives. As the generations progress, the tardiness of a system decreases while the sheet wastage of the system increases thus displaying an inversely proportional relationship. This is because of the fact that identical jobs have same due dates while non-identical jobs do not have same due dates. Sheet wastage is minimised only when we process all identical jobs in one batch which facilitates efficient nesting thus leading to lesser material wastage but this would lead to increased tardiness as the entire stock of other parts would have to wait for the first geometry to finish. This explains why material wastage and tardiness act as conflicting objectives.

In summary, the GA-based heuristic produces solutions which are within 15% of the optimal solution for the single batch problem. The GA-based heuristic is shown to be capable of solving large sized problems within reasonable times. Moreover, it is also seen that the nested GA-based heuristic gives better solutions than the algorithm where jobs are allocated to machines using dispatching rules and then optimised using a branch and bound technique.

The practicing managers would find the proposed algorithms useful since they provide very good results within reasonable computational effort. Algorithms when used in practice have to provide quick and good solutions rather than provide optimum solutions consuming enormous time. This research also brings out the tradeoff between tardiness and nesting problems which would bring together these two independent problems in practice to obtain good implementable results.

6 Conclusions

The unique contribution of the present work is the development of a genetic algorithm-based heuristic to solve complex and large sized machine loading problems in a FMS within reasonable computational effort. Other contributions include the result that the time taken by a prevalent branch and bound technique is highly sensitive to the tool magazine capacity of the machines. A comparison of performance among the different dispatching rules: EDD rule, SPT rule and the least slack rule for the multiple batch size problem indicated that the least slack dispatching rule produces the best results. It is also concluded observed that load balance and total tardiness are non-conflicting objectives while material wastage and total tardiness are conflicting objectives and share an inversely proportional relationship.

A limitation of the work is that the proposed GA is a heuristic technique and cannot guarantee optimal solutions always. Another limitation is that we have compared the nested GA (for multiple batches) with an algorithm that uses a dispatching rule in the first stage and a branch and bound in the second stage.

Further work can concentrate on development of a heuristic which is capable of handling a situation where in orders of jobs keep coming in dynamically and it is capable of generating real time schedules for every modification. Moreover, there is a need for a technique which would be capable of handling multiple stage flow shops as well which increases the complexity level of the problem to a significant extent. Apart from these, several other factors which can be incorporated into the proposed heuristic to make it more efficient are breakdown of machines, non-availability of tools, grouping of jobs and non-deterministic processing times.

Acknowledgements

The authors are thankful to the reviewers for their valuable comments on the paper, which helped us to improve the quality and relevance of the paper.

References

Ali, A. and Sangari, M.S. (2010), 'A metaheuristic method for optimising inspection strategies in serial multistage processes', *International Journal of Productivity and Quality Management*, Vol. 6, No. 3, pp.289–303.

Atmani, A. and Lashkari, R.S. (1998) 'A model of machine tool selection and operation allocation in FMS', *International Journal of Production Research*, Vol. 36, No. 5, pp.1339–1349.

Berrada, M. and Stecke, K.E. (1986) 'A branch and bound approach for machine load balancing in flexible manufacturing systems', *Management Science*, Vol. 32, No. 10, pp.210–225.

Chan, F.T.S., Swarnkar, R. and Tiwari, M.K. (2005) 'Fuzzy goal-programming model with an artificial immune system (AIS) approach for a machine tool selection and operation allocation problem in a flexible manufacturing system', *International Journal of Production Research*, Vol. 43, No. 19, pp.4147–4163.

Cheng, R., Gen, M. and Tozawa, T. (1995) 'Minmax earliness/tardiness scheduling in identical parallel machine system using genetic algorithms', *Computers and Industrial Engineering*, Vol. 29, No. 14, pp.513–517.

Co, H.C., Jaw, T.J. and Chen, S.K. (1988) 'Sequencing in flexible manufacturing systems and other short queue length systems', *Journal of Manufacturing Systems*, Vol. 7, No. 1, pp.1–9.

Dar-El, E.M. and Sarin, S.C. (1984) 'Scheduling parts in FMS to achieve maximum machine utilization', *Proceedings of 1st ORSA/TIMS Conference on FMS*, Ann Arbor, MI.

Figielska, E. (1999) 'Preemptive scheduling with changeovers: using column generation technique and genetic algorithm', *Computers and Industrial Engineering*, Vol. 37, Nos. 1–2, pp.81–84.

Gershwin, S.B. (1994) *Manufacturing Systems Engineering*, Prentice-Hall, Upper Saddle River, NJ.

Gilles, N. and Matteo, S.M. (2009) 'Flow shop operator scheduling through constraint satisfaction and constraint optimisation techniques', *International Journal of Productivity and Quality Management*, Vol. 4, Nos. 5/6, pp.549–568.

Goldberg, D.E. and Korb, D.B. (1989) 'Messy genetic algorithms: motivation analysis and first results', *Complex Systems*, Vol. 3, No. 5, pp.493–530.

Graves, S.C. (1981) 'A review of production scheduling', *Operations Research*, Vol. 29, No. 3, pp.646–675.

Hoitomt, D.J., Luh, P.B. and Pattipati, K.R. (1993) 'A practical approach to job shop scheduling problems', *IEEE Transactions on Robotics and Automation*, Vol. 9, No. 1, pp.1–13.

Hutchison, J. (1991) 'Current and future issues concerning FMS scheduling', *Omega International Journal of Management Science*, Vol. 19, No. 6, pp.529–537.

Hwang, S.S. (1986) *Models for Production Planning in Flexible Manufacturing System*, PhD thesis, University of California, Berkley.

Kim, Y, Lee, G, Lim, S. and Choi, S. (2003) 'Tool requirements planning in flexible manufacturing system: minimizing tool costs subject to a makespan constraint', *International Journal of Production Research*, Vol. 41, No. 14, pp.3339–3357.

Kopfer, H. and Mattfield, D.C. (1997) 'A hybrid search algorithm for the job shop problem', *Proceedings of the First International Conference on Operations and Quantitative Management*, Jaipur India, 5–8 January, Vol. 2, pp.498–505.

Kuhn, H. (1995) 'A heuristic algorithm for the loading problem in flexible manufacturing systems', *International Journal of Flexible Manufacturing Systems*, Vol. 7, No. 3, pp.229–254.

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*, Vol. 67, No. 3, pp.387–404.

Maheswaran, R., Ponnambalam, S.G. and Jawahar, N. (2008) 'Hybrid heuristic algorithms for single machine total weighted tardiness scheduling problems', *Int. J. of Intelligent Systems Technologies and Applications*, Vol. 4, Nos. 1/2, pp.34–56.

McCormick, S.T. and Pinedo, M.L. (1995) 'Scheduling n independent jobs on m uniform machines with both flow time and makespan objectives: a parametric analysis', *ORSA Journal of Computing*, Vol. 7, No. 1, pp.63–77.

Min, L. and Cheng, W. (1999) 'A genetic algorithm for minimizing the makespan in the case of scheduling identical parallel machines', *Artificial Intelligence Engineering*, Vol. 13, No. 4, pp.399–403.

Moslehi, G., Mahnam, M., Amin-Nayeri, M. and Azaron, A. (2010) 'A branch-and-bound algorithm to minimise the sum of maximum earliness and tardiness in the single machine', *International Journal of Operational Research*, Vol. 8, No. 4, pp.458–482.

Mukhopadhyay, S.K., Singh, M.K. and Srivastava, R. (1998) 'FMS loading: a simulated annealing approach', *International Journal of Production Research*, Vol. 36, No. 6, pp.1529–1547.

Nascimento, M.A. (1993) 'Giffler and Thompsons algorithm for job shop scheduling is still good for flexible manufacturing systems', *Journal of Operations Research Society*, Vol. 44, No. 5, pp.521–524.

Ponnambalam, S.G. and Kiat, L.S. (2008) 'Solving machine loading problem in flexible manufacturing systems using particle swarm optimization', *World Academy of Science Engineering and Technology*, Vol. 15, pp.14–19.

Sarma, U.M.B.S., Kant, S., Rai, R. and Tiwari, M.K. (2002) 'Modeling the machine loading problem of FMSs and its solution using a tabu-search based heuristic', *International Journal of Computer Integrated Manufacturing*, Vol. 15, No. 4, pp.285–295.

Shanker, K. and Srinivasulu, A. (1989) 'Some solution methodologies for loading problems in a flexible manufacturing system', *International Journal of Production Research*, Vol. 27, No. 3, pp.1019–1034.

Shanthikumar, G.J. and Stecke, K.E. (1986) 'Reducing work-in-process inventory in certain types of flexible manufacturing systems', *European Journal of Operational Research*, Vol. 16, No. 4, pp.301–314.

Song, C.Y., Hwang, H. and Kim, Y.D. (1995) 'Heuristic algorithm for the tool movement policy in flexible manufacturing systems', *International Journal of Manufacturing Systems*, Vol. 14, No. 3, pp.160–168.

Stecke, K.E. (1983) 'Formulation and solution of nonlinear integer production planning problems for flexible manufacturing systems', *Management Science*, Vol. 29, No. 3, pp.273–288.

Tiwari, M.K. and Vidyarthi, N.K. (2000) 'Solving machine loading problems in a flexible manufacturing system using a genetic algorithm based heuristic approach', *International Journal of Production Research*, Vol. 38, No. 14, pp.3357–3384.

Tiwari, M.K., Hazarika, B., Vidhyarthi, N.K, Jaggi, P. and Mukhopadhyay, S.K. (1997) 'A heuristic solution approach to the machine loading problem of an FMS and its Petri net model', *International Journal of Production Research*, Vol. 35, No. 8, pp.2269–2284.

Valente, J.M.S. (2007) 'Heuristics for the single machine scheduling problem with early and quadratic tardy penalties', *European J. of Industrial Engineering*, Vol. 1, No. 4, pp.431–448.

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

Wisut, S. and Karn, P. (2011) 'Memetic algorithm for non-identical parallel machines scheduling problem with earliness and tardiness penalties', *International Journal of Manufacturing Technology and Management*, Vol. 22, No. 1, pp.26–38.