Int J Adv Manuf Technol (2015) 77:1269–1274
DOI 10.1007/s00170-014-6564-2

ORIGINAL ARTICLE

A heuristic approach based on time-indexed modelling for scheduling and tool loading in flexible manufacturing systems

Selin Özpeynirci

Received: 9 August 2014 / Accepted: 28 October 2014 / Published online: 7 November 2014

Abstract In this paper, we simultaneously consider the scheduling and tool loading problems in flexible manufacturing systems. There are various jobs that must be processed on a number of parallel computer numerical control machines. The processing of each job requires a set of machine tools. However, the number of tool copies available in the system is limited due to economic restrictions. The problem, therefore, is to schedule the jobs and the required tools in such a way that the makespan is minimized. We present a time-indexed mathematical model of the problem. A heuristic approach based on the mathematical model is also developed and the computational results are presented. The goal of this study is to develop a new approach for simultaneously scheduling the jobs and loading the tools in flexible manufacturing systems and benefit from the advantages of time-indexed modelling.

Keywords Time-indexed modelling · Scheduling · Flexible manufacturing systems

1 Introduction

Flexible manufacturing systems (FMSs) consist of computer numerical control (CNC) machines connected to an automated material handling system. These provide flexibility in terms of product type and the order of operations executed on a part of a product, which also allow the same operation to be performed on different machines. CNC machines are able to apply different operations provided that the required tools are loaded. Due to these advantages, but also the high level of investments required, FMSs have attracted attention from both industry and academia.

This study considers the scheduling of jobs on parallel CNC machines together with their required tools. The loading, tooling, and scheduling problems in FMSs are generally studied separately in the literature due to their complex nature.

Among the studies that consider tooling and scheduling problems separately, Agnetis et al. [1] show that, given a job sequence for two machines, the tool scheduling problem can be easily solved to optimality. They introduce several decomposition strategies for the problem.

Kellerer and Strusevich [5] consider scheduling problems on parallel dedicated machines under multiple resource constraints, introducing several cases and discussing the computational complexity of the problems. One case they discuss corresponds to the problem currently addressed—a single resource of arbitrary size, where available units of resource may be one or more than one. However, unlike in the current study, these researchers assume that the machine that processes each job is determined in advance. They propose a polynomial time approximation scheme for the problem with a fixed number of machines in which a job uses at most one resource of unit size.

There are few studies that refer to loading and scheduling of jobs and tools simultaneously. One such is Roh and Kim [6], who study part loading, tool loading, and sequencing problems simultaneously with the objective of minimizing total tardiness. They propose three heuristic approaches: simultaneous, sequential, and iterative, respectively. Their simulation study found that the iterative approach performs best.

Ventura and Kim [11] consider a parallel machine scheduling problem where jobs may require resources in addition to the machines. Assuming unit processing time for each job, they consider two cases where there are multiple and single additional resources, respectively. They use Lagrangean relaxation approach to find bounds.

S. Özpeynirci (✉)
Industrial Engineering Department, İzmir University of Economics, Sakarya Cad. No:156 35330, Balçova, İzmir, Turkey
e-mail: selin.ozpeynirci@ieu.edu.tr

In another study, Özpeynirci and Gökgür [7] consider the loading and scheduling problems simultaneously, using an identical problem definition to the current study. They provide a mixed integer mathematical model that fails to reach the optimal solution in reasonable time. They develop a tabu search algorithm and show that it performs well with computational experiments.

In this study, we model the problem of loading and scheduling of jobs with their required tools as a time-indexed mathematical model. We also provide a heuristic approach based on this model. Time-indexed mathematical models have been used to formulate different types of problems, including scheduling problems. They have attracted research interest because of their many advantages, such as strong bounds provided by linear programming relaxations, and approximation algorithms that can be developed based on these models. However, time-indexed formulations also have a major disadvantage, their size. Due to time index, the number of variables and constraints can be extremely large, even for relatively minor problems. Van den Akker et al. [10] suggest the use of Dantzig-Wolfe decomposition techniques to alleviate the effects of the problem size.

Demir and İşleyen [3] compare five different mathematical models for flexible job shop scheduling problem and observe that the time-indexed model requires considerably more computational time compared to others.

Baptiste and Sadykov [2] apply time-indexed formulation to schedule tasks on airborne radars. In their problem, radar corresponds to the single machine, and tasks correspond to jobs to be scheduled including a chain of operations with identical processing times. The operations of a job should be scheduled according to a given frequency. Their objective is to minimize the total penalty of deviations from this frequency. They develop three time-indexed models, two of which can be solved by mixed integer programming solvers, while the other relies on branch-and-price algorithm. Their computational experiments show the superiority of branch-and-price algorithm and one of the other models.

Thörnblad [9] compare different mathematical models for scheduling a multitask production cell, which is a flexible job shop environment. Two of these models are developed using time-indexed variables: one using binary variables that take the value of 1 if a job is processed during a specific time interval (so-called plateau variables) and the other using binary variables equal to 1 if a job starts to be processed at a certain time interval (so-called nail variables). Their computational experiments show that the model with nail variables outperforms the others.

Thörnblad et al. [8] propose a time-indexed mathematical model based on nail variables for flexible job shop problem with preventive maintenance and fixture availability. Their experiments show that time-indexed model can solve the instances that could not be solved by the model with variables most frequently used in scheduling problems.

In the literature, most studies consider the loading and scheduling problems with tooling issues either sequentially or with simplifying assumptions. In contrast, in this paper, we provide an approach with time-indexed modelling to provide a simultaneous solution to these problems, which to the best of the author's knowledge, is the first.

2 Problem definition

Consider n jobs to be processed on m parallel CNC machines. There is no precedence relation between the jobs. Every machine is eligible to process all jobs, but the processing times of the jobs on different machines may vary due to the speed or age of the machine. A job must be assigned to exactly one machine, and preemption is not allowed. Job i requires a set $l(i)$ of tools to be processed, i.e., the tools in set $l(i)$ must be loaded on the machine to be able to process job i.

There are t types of tools, and r_k copies of tool type k are available in the system. Due to economic restrictions, the number of tool copies may be smaller than the number of machines. Therefore, if a tool is required to process a job on a machine and is not loaded on the machine, it must be taken from another machine or tool crib. If all copies are being used on other machines, the process must be delayed until a copy is available.

We make the following assumptions about the tool usage:

- Each tool requires one tool slot in the tool magazine.
- Tools do not break down during processing.
- The time required for tool switches between the machines is negligible.
- The number of tools required by a job does not exceed the tool magazine capacity of a machine.
- Tools cannot be removed from the machine during processing.

The problem is to schedule the jobs and their required tools on parallel CNC machines with the objective of minimizing makespan, i.e., the completion time of the last job. If $r_k \geq m$ for all tool types, there will be no tooling restrictions and our problem is reduced to the parallel machine scheduling problem, which is NP-hard in general [4]. Therefore, we can conclude that our problem is also NP-hard.

3 Mathematical model

In this section, we first define the indices, parameters, and decision variables used and then present the mathematical time-indexed model (TIM). We assume that the planning

horizon is divided into T time intervals each of 1 time unit. The index u represents the time interval that starts at time u and ends at time $u+1$.

Indices
i, q=Job index, i, q=1, 2,…, n
j=Machine index, j=1, 2,…, m
k=Tool type index, k=1, 2,…, t
h_k=Replicate number of tool type k, h_k=1, 2,…, r_k
u=Time interval, u=1, 2,…, T
Parameters
p_{ij}=Processing time of job i on machine j
r_k=Number of tools of type k
$l(i)$=Set of tools required to process job i
Decision variables
$C_{\max}$=Production makespan
C_i=Production completion time of job i

$$X_{iju} = \begin{cases} 1 & \text{if job } i \text{ is scheduled to start processing on} \\ & \text{machine } j \text{ at the beginning of time interval } u \\ 0 & \text{otherwise} \end{cases}$$

$$Z_{ih_k} = \begin{cases} 1 & \text{if replicate } h \text{ of tool } k \text{ is used to process job } i \\ 0 & \text{otherwise} \end{cases}$$

The mathematical model TIM is given below:

$$\text{Min } C_{\max} \tag{1}$$

subject to

$$C_{\max} \geq C_i \qquad \forall i \tag{2}$$

$$C_i = \sum_{j=1}^{m} \sum_{u=0}^{T} \left[\left(u + p_{ij} \right) X_{iju} \right] \qquad \forall i \tag{3}$$

$$\sum_{j=1}^{m} \sum_{u=0}^{T} X_{iju} = 1 \qquad \forall i \tag{4}$$

$$X_{iju} = 0 \qquad \forall i, j, u = T - p_{ij} + 1, \ldots, T \tag{5}$$

$$\sum_{i=1}^{n} \sum_{\mu=u-p_{ij}+1}^{u} X_{ij\mu} \leq 1 \qquad \forall j, u \tag{6}$$

$$\sum_{h_k=1}^{r_k} Z_{ih_k} = 1 \qquad \forall i, k \in l(i) \tag{7}$$

$$\sum_{j=1}^{m} \sum_{\mu=u-p_{ij}+1}^{u} X_{ij\mu} + \sum_{j=1}^{m} \sum_{\mu=u-p_{qj}+1}^{u} X_{qj\mu} \leq 1 - M\left(Z_{ih_k} + Z_{qh_k} - 2 \right) \forall i < q, h, k \in l(i) \cap l(q), u \tag{8}$$

$$C_{\max} \geq 0 \tag{9}$$

$$C_i \geq 0 \qquad \forall i \tag{10}$$

$$X_{iju} \in \{0, 1\} \qquad \forall i, j, u \tag{11}$$

$$Z_{ih_k} \in \{0, 1\} \qquad \forall i, k \in l(i) \tag{12}$$

The objective function (1) minimizes the production makespan. Constraint (2) ensures that the completion time of any job is less than or equal to the production makespan. Constraint (3) defines the completion time of the jobs as the summation of starting time and processing time. Constraint (4) schedules a job to be processed exactly once. Constraint (5) guarantees that a job is not scheduled to start at a time that the job would not be completed at T. Constraint (6) ensures that a machine processes only one job at a time. Constraint (7) allows each job to use exactly one copy of the required tool. Constraint (8) guarantees that a tool copy is used by one job at a time. Constraints (9–12) are the set constraints.

The length of planning horizon, T, has a great impact on the number of decision variables and constraints; hence, it can significantly affect the solution time. The value of T should be sufficiently long to create an optimal schedule for the planning horizon but should be kept as short as possible in order to reduce the computation times [9].

A convenient way to find an upper bound for $C_{\max}$ is summing up the minimum processing times of jobs among the machines. In this way, we assume that each job is processed on the machine in the shortest possible processing time and also that the processing times of jobs do not overlap; hence, a feasible schedule can be obtained.

$$T = \sum_{i=1}^{n} \left(\min_j \left\{ p_{ij} \right\} \right)$$

Another way to find T is using a greedy heuristic that will provide an upper bound on the optimal makespan value. Özpeynirci and Gökgür [7] develop a heuristic algorithm to find an initial feasible solution for their tabu search algorithm. We use the same heuristic to determine T value. For the sake of completeness, their heuristic algorithm is defined below:

Let S_0 be the set of jobs that are not assigned to a machine yet and S_1 be the set of jobs that are assigned to a machine.

Step 0 Set $S_0=\{1, 2,\ldots, n\}$ and $S_1=\{\}$. List the job-machine pairs in a non-decreasing order of processing times. Assign the jobs to the machines in the list if the jobs are not yet assigned and the machine is empty. Continue until one job is assigned to each machine. If it is impossible to start any job at time 0 due to unavailability of the required tools, insert idle time until at least one copy of all required tools are free, i.e., a feasible solution is found. Update sets S_0 and S_1.

Step 1 Find the machine that becomes idle first. Let the machine be j. Calculate the priority values for all jobs in set S_0 on machine j using the following equation:

$$\pi_{ij} = p_{ij} \times a_{ij} \times b_{ij}^2$$

where

π_{ij}=Priority value of job i on machine j

a_{ij}=The number of tools needed additionally to assign job i to machine j that we have available copy

b_{ij}=The number of tools needed additionally to assign job i to machine j that we do not have available copy

By calculating the priorities of the jobs as above, we give higher importance to jobs with shorter processing times and therefore demand fewer additional tools. If a job requires an unavailable tool, we need to insert idle time until the tool becomes free. Hence, we take the square of b_{ij} to increase the π_{ij} values of these jobs.

Select the job with minimum priority value. Let the job be i.

Step 2 Remove the tools from machine j that are not elements of $l(i)$.

Step 3 Load the tools that are elements of $l(i)$ to machine j and those that are not already loaded on machine j. If a required tool is not free, then delay the starting time of job i on machine j until the tool becomes free, i.e., the job that uses the tool is completed. Update sets S_0 and S_1. If $S_0=\{\}$, then stop. Otherwise, go to step 1.

We find upper bounds for $C_{\max}$ by using the two approaches defined above. Then we set the value of T to the minimum of these two upper bounds.

4 Heuristic approach

Due to the difficulty of the problem and large T values, the optimal solution of even very small-sized problems cannot be found within a reasonable time limit. Therefore, there is a need for an effective heuristic that finds good feasible solutions within a short time. For this purpose, we design a heuristic approach and suggest an improvement on this heuristic based on the time-indexed model given above.

An important factor that affects the solution time is the T value. When we assume that the length of each interval is 1 time unit, the total number of intervals is T. Reducing the total number of intervals would reduce the computational burden. Therefore, we may assume that the length of each time interval is l where $l \geq 1$ and integer. Then we update the processing times by dividing each p_{ij} by l and rounding up to guarantee feasible solutions.

$$p'_{ij} = \left\lceil \frac{p_{ij}}{l} \right\rceil$$

Once we obtain the updated processing times, we find new upper bounds for the T value and solve the time-indexed model. Let us call this model TIM-H. The l value can be increased until the T value is lowered enough to find optimal solutions to the TIM-H.

Lastly, the optimal solution of TIM-H is multiplied by l to find an upper bound on the original $C_{\max}$.

Solution of the above heuristic can be improved significantly with a little additional effort. The sequence found by TIM-H is taken, and $C_{\max}$ is calculated using the original p_{ij} values. This improvement can be applied if TIM-H gives an optimal solution.

5 Computational experiments

In order to measure the performance of proposed heuristics, a series of experiments are conducted using the procedure of Özpeynirci and Gökgür [7]. In our experiments, we set the number of jobs to 8, 10, and 15; the number of machines to 2 and 3; and the number of tool types to 5 and 8. The number of tools in the set $l(i)$ is generated from a discrete uniform distribution in the interval [2, 5]. The tools in set $l(i)$ are generated randomly. The value of r_k is randomly selected from the set {1,2}. The p_{ij} values are generated from a discrete uniform distribution in the interval [25, 150]. Ten problem instances are generated under each parameter setting.

The models are solved by IBM ILOG CPLEX Optimization Studio 12.4 using an Intel Xeon, 3.2 GHz (two processors) computer with 10 GB RAM. We set an upper time limit of 1 h, and after which, the execution of the algorithm is terminated if the optimal solution has not been found.

The results of computational experiments are reported in Tables 1 and 2. In the first three columns of each table, the number of jobs (n), the number of machines (m), and the number of tool types (t) are given. Table 1 gives the average

Table 1 Solution time performance of heuristic

n	*m*	*t*	Average *T* values			Number of problems solved to optimality			Solution times (s)		
			l=2	*l*=4	*l*=8	*l*=2	*l*=4	*l*=8	*l*=2	*l*=4	*l*=8
8	2	5	233.5	118.8	61.5	8	10	10	643.00	83.01	11.51
8	2	8	192.9	98.7	51.9	9	10	10	559.40	43.78	7.54
10	2	5	318.0	160.9	82.1	0	9	10	–	508.61	43.61
10	2	8	259.9	134.3	68.9	2	10	10	1117.00	582.81	53.27
10	3	5	258.5	134.7	69.5	0	10	10	–	670.91	26.66
10	3	8	237.1	120.1	59.7	3	10	10	911.76	324.45	18.83
15	2	8	406.9	206.9	103.9	0	0	7	–	–	1081.13

T values, the number of problems among ten instances solved to optimality by the TIM-H, and the solution times of TIM-H in seconds for *l* value set to 2, 4, and 8. The time required for the improvement step is negligible and, therefore, not reported. Table 2 shows the average percent deviations of heuristic solution before and after improvement from the optimal solution for three levels of *l*. Optimal solutions are taken from Özpeynirci and Gökgür [7]. If the TIM-H is unable to reach an optimal solution, the deviation of upper bound from the optimal is reported.

It can be seen from Table 1 that the complexity of the problem is affected by the number of operations, the number of machines, and the *T* value. As the values of these parameters increase, the solution time increases and the number of instances solved by TIM-H decreases. The number of tool types has no significant effect on the model performance. The highest improvement in the performance is observed as *l* value increases, which causes a decrease in *T* value.

The deviations of heuristic solution from optimal, given in Table 2, increase as *l* value increases, due to increasing errors caused by rounding up the processing times after dividing by *l*. When we apply the improvement step, however, the deviations decrease substantially, and in most instances, it is possible to find the optimal solution to the original model TIM.

As the problem size increases, *l* value can be increased to the extent that it reduces the *T* value. Once an optimal solution to TIM-H has been found, the improvement step leads us to near-optimal solutions.

Table 2 Average deviation of heuristic solutions from optimal

n	*m*	*t*	Heuristic approach			Improvement		
			l=2	*l*=4	*l*=8	*l*=2	*l*=4	*l*=8
8	2	5	0.82	2.41	6.47	0.00	0.00	0.27
8	2	8	0.72	2.24	5.99	0.00	0.20	0.41
10	2	5	3.24	2.05	4.92	–	0.00	0.20
10	2	8	0.76	2.10	4.99	0.00	0.00	0.07
10	3	5	1.60	2.22	6.07	–	0.06	0.28
10	3	8	0.38	1.92	5.61	0.00	0.05	0.20
15	2	8	–	7.20	4.39	–	–	0.23

6 Conclusion

In this study, we consider the scheduling of jobs and their required tools on parallel CNC machines. Job assignment and tool loading and scheduling problems are solved simultaneously, with the objective of minimizing makespan.

We develop a time-indexed model and observe that even small-sized instances cannot be solved by this model to optimality. This is due to the time index that increases the problem size substantially. The high value of *T*, which is affected by the processing times, increases the number of variables and constraints.

We suggest a heuristic approach, which involves modifying the processing times, thus leading to a reduction in the planning horizon and a decrease in the number of constraints and variables. Assuming that the length of each time interval is greater than 1 unit, we obtain reduced processing times and hence reduced *T*. With the new processing times, we can solve the model to optimality and find upper bounds on the makespan.

We also improve these solutions by using the schedule obtained by the heuristic approach and finding the makespan with the real processing times. Our experiments show that the improvement step enables the optimal solution to be found in most instances. On average, the deviation from the optimal solution is approximately 0.2 %, which is better than the deviations from the tabu search solution developed by Özpeynirci and Gökgür [7].

In this study, we assume that the tool switching times are negligible. However, future studies can take into account tool switching times in order to develop the relevant solution approaches. Also, studies can be conducted that apply other effective approaches to scheduling problems, such as constraint programming or Lagrangean relaxation.

Acknowledgments This study is supported by the Scientific and Technological Research Council of Turkey, project no: 110M492.

References

1. Agnetis A, Alfieri A, Brandimarte P, Prinsecchi P (1997) Joint job/tool scheduling in a flexible manufacturing cell with no on-board tool magazine. Comput Integr Manuf Syst 10:61–68
2. Baptiste P, Sadykov R (2010) Time-indexed formulations for scheduling chains on a single machine: an application to airborne radars. Eur J Oper Res 203:476–483
3. Demir Y, İşleyen SK (2013) Evaluation of mathematical models for flexible job-shop scheduling problems. Appl Math Model 37:977–988
4. Garey MR, Johnson DS (1979) Computer and intractability. A guide to the theory of NP-completeness. *Bell Laboratories*
5. Kellerer H, Strusevich VA (2004) Scheduling problems for parallel dedicated machines under multiple resource constraints. Discret Appl Math 133:45–68
6. Roh H-K, Kim Y-D (1997) Due-date based loading and scheduling methods for a flexible manufacturing system with an automatic tool transporter. Int J Prod Res 35:2989–3003
7. Özpeynirci S, Gökgür B (2011) Esnek İmalat Sistemlerinde Çizelgeleme ve Makine Ucu Atama Problemi için Tabu Arama Algoritması. 31. Ulusal Yöneylem Araştırması ve Endüstri Mühendisliği Bildiri Kitabı, 708–714
8. Thörnblad K, Almgren T, Patriksson M, Strömberg AB (2012) Mathematical optimization of a flexible job shop problem including preventive maintenance and availability of fixtures. Proceedings of the 4th world P&OM conference/19th international annual EurOMA conference, Amsterdam, Netherlands
9. Thörnblad K (2011) On the optimization of schedules of a multitask production cell. Licentiate thesis, Department of Mathematical Sciences, Chalmers University of Technology and the University of Gothenburg
10. Van den Akker JM, Hurkens CAJ, Savelsbergh MWP (2000) Time-indexed formulations for machine scheduling problems: column generation. INFORMS J Comput 12:111–124
11. Ventura JA, Kim D (2003) Parallel machine scheduling with earliness-tardiness penalties and additional resource constraints. Comput Oper Res 30:1945–1958