Applied Mathematical Modelling 34 (2010) 1539–1548

Contents lists available at ScienceDirect

Applied Mathematical Modelling

journal homepage: www.elsevier.com/locate/apm

Mathematical models for job-shop scheduling problems with routing and process plan flexibility

Cemal Özgüven [a,*], Lale Özbakır [b], Yasemin Yavuz [a]

[a] Erciyes University, Faculty of Economics and Administrative Sciences, Department of Business, 38039 Kayseri, Turkey
[b] Erciyes University, Faculty of Engineering, Department of Industrial Engineering, 38039 Kayseri, Turkey

ARTICLE INFO

Article history:
Received 31 October 2008
Received in revised form 25 August 2009
Accepted 1 September 2009
Available online 4 September 2009

Keywords:
Job-shop scheduling
Routing flexibility
Process plan flexibility
Mixed-integer programming

ABSTRACT

As a result of rapid developments in production technologies in recent years, flexible job-shop scheduling problems have become increasingly significant. This paper deals with two NP-hard optimization problems: flexible job-shop scheduling problems (FJSPs) that encompass routing and sequencing sub-problems, and the FJSPs with process plan flexibility (FJSP-PPFs) that additionally include the process plan selection sub-problem. The study is carried out in two steps. In the first step, a mixed-integer linear programming model (MILP-1) is developed for FJSPs and compared to an alternative model in the literature (Model F) in terms of computational efficiency. In the second step, one other mixed-integer linear programming model, a modification of MILP-1, for the FJSP-PPFs is presented along with its computational results on hypothetically generated test problems.

1. Introduction

Scheduling may broadly be defined as the allocation of resources to tasks over time in such a way that a predefined performance measure is optimized. From the viewpoint of production scheduling, the resources and tasks are commonly referred to as machines and jobs and the commonly used performance measure is the completion times of jobs. This problem has been extensively researched since the early 1950's. A comprehensive survey of literature on scheduling problems can be found in Graves [1], Lawler et al. [2], and Lee et al. [3].

The classical job-shop scheduling problem (JSP) consists of a set of independent jobs, each having its own processing order through a set of machines. Each job has an ordered set of operations, each of which must be processed on a predefined machine. The problem, known to be strongly NP-hard, is to sequence operations on the machines so that the maximum completion time over all jobs ($C_{\max}$) is minimized. Considerable research has been devoted to this problem in the literature. An overview of history and main techniques used along with their reported results on the available benchmarking problems for JSP can be found in Jain and Meeran [4].

The JSP assumes only one available machine for each operation and one feasible process plan (operation sequence) for each job. It is common, however, in real-world scheduling problems to have a wider space for choices. Typically, in a manufacturing setting there are choices to be made in scheduling from among alternative machines on which to perform an operation or from among alternative process plans of a job through a factory [5]. This paper considers two extensions to the JSP, i.e., *routing flexibility* and *process plan flexibility*. The former implies the availability of alternative machines for each operation, and the latter the availability of alternative process plans for each job.

* Corresponding author. Tel.: +90 352 4374901/30150; fax: +90 352 4375239.
E-mail addresses: cozguven@erciyes.edu.tr (C. Özgüven), lozbakir@erciyes.edu.tr (L. Özbakır), yaseminy@erciyes.edu.tr (Y. Yavuz).

0307-904X/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.apm.2009.09.002

The flexible job-shop scheduling problem (FJSP) is an extension of JSP in that it incorporates the routing flexibility. Due to complications resulting from the addition of routing flexibility to the already difficult JSP, researchers have focused on meta-heuristic approaches to solve the FJSP. Among these are the evolutionary algorithms [6–8], tabu search [9–13], simulated annealing [14,15], ant colony [16] and hybrid approaches [17–22].

The FJSP with process plan flexibility (FJSP-PPF) refers to a more complicated decision problem where jobs can have alternative process plans. The literature for this problem is quite limited: evolutionary algorithms [23–25], tabu search [26], simulated annealing [27], and constraint-directed techniques [5].

As for the mathematical models, the initial formulations of scheduling may be traced back to 1960's. Shapiro [28] has presented mathematical programming models and solution methods that have been applied to several types of production planning and scheduling problems. Pan [29] has provided a review and comparison of mixed-integer linear programming (MILP) formulations for job-shop, flow-shop and permutation flow-shop scheduling problems described in the literature. Kim and Egbelu [30] have developed a MILP model for JSP, where each job has predetermined alternative process plans. They have also presented two algorithms, each consisting of three modules, i.e., selecting a process plan combination, computing the lower bound on makespan, and scheduling. While the scheduling module of the first algorithm depends on the developed MILP model, that of the second algorithm employs earliest completion time (ECT) dispatching rule. Brandimarte [31] has addressed a multi-objective scheduling problem with process plan flexibility and proposed a hierarchical approach, based on a decomposition into machine loading and scheduling sub-problems. Brandimarte's paper deals only with the machine loading problem (selecting a process plan for each job and assigning operations to the machines). The problem is formulated as a bicriterion integer programming model and two different heuristics are proposed, one based on surrogate duality theory and the other based on a genetic descent algorithm.

Low and Wu [14] have addressed the FJSP with the objective of minimizing total tardiness, under setup time consideration. The problem presented as an extended disjunctive graph has been formulated as a mixed-integer programming model, and then quadratic terms in the constraints have been linearized. A simulated annealing-based heuristic has been proposed to solve the problem. The model presented in Low and Wu [14] has been extended to a multi-objective model by Low et al. [19]. Thomalla [32] has proposed a discrete-time integer programming model for the FJSP with the objective of minimizing the sum of the weighted quadratic tardiness of the jobs and developed an algorithm based on Lagrangian relaxation. Choi and Choi [33] have presented a MILP model for the FJSP with sequence-dependent setups, along with a local search algorithm that uses a dispatching rule to obtain an upper bound on the makespan of a subproblem. Gomes et al. [34] have addressed the FJSP with limited intermediate buffers and developed a multi-objective discrete time model that divides the scheduling horizon into a number of intervals of equal duration. This model has subsequently been generalized to account for re-circulation, where jobs may visit certain machine groups more than once. Gao et al. [18] have formulated the FJSP with non-fixed machine availability constraints, where each machine is subject to an arbitrary number of preventive maintenance tasks. To solve this problem, they have proposed a hybrid genetic algorithm. Fattahi et al. [20] have developed a mixed-integer programming model and also six hybrid searching structures that differ in accordance with the searching approaches and heuristics they have used to solve FJSPs. Lee et al. [23] have presented a mathematical model for FJSP-PPF in which each customer order has a due date and outsourcing is available. To solve the model, a genetic algorithm-based heuristic approach has been developed.

In this paper, a well-known model, which is developed by Manne [35] for JSPs, is modified with the purpose of enabling it to solve the FJSPs. The resulting model is referred to as MILP-1. A mechanism is then incorporated into MILP-1 to take care of the FJSP-PPFs. The resulting model is referred to as MILP-2.

In our search in the literature for the mathematical modelling approaches, all except one of the models we came across are different variations of the scheduling problems either with routing or with process plan flexibility, but not both. The only exception is the model developed by Lee et al. [23]. However, this model is not comparable to MILP-2. Since customer due dates and outsourcing are incorporated into the model, it does not address the FJSP-PPF as described in Section 2.

The remainder of this paper is organized as follows: in Section 2, a formal problem description is given. In Section 3, MILP-1 is presented and compared with the model developed by Fattahi et al. [20]. Section 4 is devoted to the presentation of MILP-2 and to its computational results on test problems. Concluding remarks are made in Section 5.

2. Description of the problems

FJSP consists of a set of n independent jobs $J = \{j_i\}_{i=1}^{n}$, each having its own processing order through a set of m machines $M = \{m_k\}_{k=1}^{m}$. A number of ℓ_i ordered operations $(O_{i1}, \ldots, O_{i\ell_{(i)}})$ has to be performed to complete job i. Operation j of job i (O_{ij}), rather than having to be processed on a predefined machine $m_j \in M$ as in JSP, can be processed by any machine in a given set $M_j \subseteq M$ for a given processing time t_{ijk}. The FJSP can be considered as consisting of many unique JSPs because of the routing flexibility. For example, for a five-job three machine scheduling problem in which each job has four operations to be processed and each operation can be processed by any two of the three machines, the FJSP can be viewed as a set of 2^{20} JSPs [14]. While JSP consists of only a sequencing problem because the assignment of operations to the machines is given in advance, the FJSP turns into a routing as well as a sequencing problem: assigning each operation O_{ij} to a machine selected from the set M_j and ordering operations on the machines so that $C_{\max}$ is minimized.

The following assumptions are made for FJSPs: the processing times are known and fixed. The setup times are either negligible or included in the processing times. Transportation times are ignored. All jobs are available at time zero. No pre-emption is allowed, and a machine can process only one operation at a time. There is only one feasible process plan for each job.

The FJSP-PPF considers multiple process plans for jobs by excluding the final assumption of FJSP. The problem consists of a set of n jobs, each having a set of $\tau_{(i)}$ process plans $P_i = \{\rho_{ip_{(i)}}\}_{p_{(i)}=1}^{\tau_{(i)}}$. The process plan $p_{(i)}$ of job i is an ordered list of $\ell_{ip_{(i)}}$ operations. It is assumed that the process plans are known in advance and represented by linear predence relationships. Operation j in the process plan $p_{(i)}$ of job i $(O_{ip_{(i)}j})$ can be processed by any machine in a given set $M_j \subseteq M$ for a given processing time $t_{ip_{(i)}jk}$. Because only one of the alternative plans is to be adopted for each job, the FJSP-PPF deals with not only routing and sequencing sub-problems but also the process plan selection sub-problem: choosing a process plan for each job i from a given set of $P_{(i)}$, assigning each operation $O_{ip_{(i)}j}$ to a machine selected from the set M_j and ordering operations on the machines so that $C_{\max}$ is minimized.

3. MILP-1 model for FJSP

There are three different definitions for binary variables in the integer programming formulation of scheduling problems [29]:

γ_{irk} = 1, if job i is scheduled in the rth position for processing on machine k; 0, otherwise.
γ_{ikt} = 1, if job i is processed by machine k during period t; 0 otherwise.
$\gamma_{ii'k} = 1$, if job i precedes job i' (not necessarily immediately) on machine k; 0 otherwise.

These three definitions were first proposed by Wagner [36], Bowman [37], and Manne [35], respectively, and they are used mainly to formulate JSPs. The following mixed-integer programming formulation of the FJSP adopts the modelling approach of Manne [35].

3.1. Presentation of the model

The following notation is used for the formulation of MILP-1.

Indices and sets

i	jobs $(i,i' \in J)$
j	operations $(j,j' \in O)$
k	machines $(k \in M)$
J	the set of jobs
M	the set of machines
O	the set of operations
O_i	ordered set of operations of job i $(O_i \subseteq O)$, where $O_{if_{(i)}}$ is the first and $O_{i\ell_{(i)}}$ is the last element of O_i
M_j	the set of alternative machines on which operation j can be processed, $(M_j \subseteq M)$
$M_j \cap M_{j'}$	the set of machines on which operations j and j' can be processed

Parameters

t_{ijk}	the processing time of operation O_{ij} on machine k
L	a large number

Decision variables

X_{ijk}	1, if machine k is selected for operation O_{ij}; 0, otherwise
S_{ijk}	the starting time of operation O_{ij} on machine k
C_{ijk}	the completion time of operation O_{ij} on machine k
$Y_{iji'j'k}$	1, if operation O_{ij} precedes operation $O_{i'j'}$ on machine k; 0, otherwise
C_i	the completion time of job i
$C_{\max}$	maximum completion time over all jobs (makespan)

The proposed mathematical model is defined as follows:
Objective function: Minimize $C_{\max}$
Constraints:

$$\sum_{k \in M_j} X_{ijk} = 1 \quad \forall i \in J,\ \forall j \in O_i, \tag{1}$$

$$S_{ijk} + C_{ijk} \leqslant (X_{ijk}) \cdot L \quad \forall i \in J,\ \forall j \in O_i,\ \forall k \in M_j, \tag{2}$$

$$C_{ijk} \geqslant S_{ijk} + t_{ijk} - (1 - X_{ijk}) \cdot L \quad \forall i \in J,\ \forall j \in O_i,\ \forall k \in M_j, \tag{3}$$

$$S_{ijk} \geqslant C_{i'j'k} - (Y_{iji'j'k}) \cdot L \quad \forall i < i',\ \forall j \in O_i,\ \forall j' \in O_{i'},\ \forall k \in M_j \cap M_{j'}, \tag{4}$$

$$S_{i'j'k} \geqslant C_{ijk} - (1 - Y_{iji'j'k}) \cdot L \quad \forall i < i',\ \forall j \in O_i,\ \forall j' \in O_{i'},\ \forall k \in M_j \cap M_{j'}, \tag{5}$$

$$\sum_{k \in M_j} S_{ijk} \geqslant \sum_{k \in M_j} C_{i,j-1,k} \quad \forall i \in J,\ \forall j \in O_i - \{O_{if_{(i)}}\}, \tag{6}$$

$$C_i \geqslant \sum_{k \in M_j} C_{i,O_{i\ell_{(i)}},k} \quad \forall i \in J, \tag{7}$$

$$C_{\max} \geqslant C_i \quad \forall i \in J, \tag{8}$$

and

$$\begin{aligned}
&X_{ijk} \in \{0,1\} \quad \forall i \in j,\ \forall j \in O_i,\ \forall k \in M_j,\\
&S_{ijk} \geqslant 0 \quad \forall i \in j,\ \forall j \in O_i,\ \forall k \in M_j,\\
&C_{ijk} \geqslant 0 \quad \forall i \in j,\ \forall j \in O_i,\ \forall k \in M_j,\\
&Y_{iji'j'm} \in \{0,1\} \quad \forall i < i',\ \forall j \in O_i,\ \forall j' \in O_{i'},\ \forall k \in M_j \cap M_{j'},\\
&C_i \geqslant 0 \quad \forall i \in j.
\end{aligned}$$

Constraints (1) make sure that operation O_{ij} is assigned to only one machine. If operation O_{ij} is not assigned to machine k, the constraints (2) set the starting and completion times of it on machine k equal to zero. Otherwise, the constraints (3) guarantee that the difference between the starting and the completion times is equal in the least to the processing time on machine k. Constraints (4) and (5) take care of the requirement that operation O_{ij} and operation $O_{i'j'}$ cannot be done at the same time on any machine in the set $M_j \cap M_{j'}$. Constraints (6) ensure that the precedence relationships between the operations of a job are not violated, i.e. the operation O_{ij} is not started before the operation $O_{i,j-1}$ has been completed. Constraints (7) determine the completion times (of the final operations) of the jobs, and constraints (8) determine the makespan.

3.2. Comparison with Model F and evaluation

The performance of the MILP-1 is compared with the solutions obtained by the mathematical model proposed by Fattahi et al. [20]. The mathematical model developed by Fattahi et al. [20] will henceforth be referred to as Model F. In Fattahi et al. [20], randomly generated test problems are divided into two categories: small size FJSPs (SFJS1- SFJS10) and medium and large size FJSPs (MFJS1- MFJS10). The test problems are defined by the size of the problem ($i \cdot j \cdot k$) in which the indices denote the number of jobs, the number of operations for all jobs, and the number of machines, respectively. LINGO software and branch and bound method are used to solve these problems. Since no method is able to generate optimal solutions for the medium and large size problems in polynomial time, they have presented two bounds (upper lower bounds) for the MFJS1- MFJS10 problems.

MILP-1 has been coded in the mathematical modelling language AIMMS 3.8 and CPLEX 11.0 has been used to solve the test problems of Fattahi et al. [20]. The problems have been run on a PC that has Core 2 CPU and 1.86 GHz processor (1 GB RAM). Running time is limited to 3600 s to compare MILP-1 with Model F on the same grounds. The solutions obtained are compared in Table 1 and Figs. 1 and 2. Since the mean differences between results and model parameters of Model F and MILP-1 is not normally distributed a nonparametric test (Mann–Whitney) is applied to determine the significance of differences between populations. The Mann–Whitney U test results are given in Table 2.

The results reached on the same test problems indicate that the overall performance of MILP-1 is superior to that of the Model F. MILP-1 finds the optimal solution in all problems of SFJS at a lower CPU time. It also finds the optimal solutions of the first five problems of MFJS (MFJS1-MFJS5) which Model F does not. The best integer solutions found in 3600 s for the remaining problems (MFJS6-MFJS10) have lower $C_{\max}$ values than those in Model F.

As shown in Fig. 1, for small sized flexible job shop problems the $C_{\max}$ values obtained by MILP-1 and Model F overlap. However, a significant difference occurs between the two models for medium sized problems. Under the same CPU time limit (3600 s) for medium sized problems, MILP-1 significantly dominates Model F in terms of the $C_{\max}$ results.

The second performance measure is the CPU times of the models. As shown in Fig. 2 MILP-1 outperforms the Model F for both small and medium sized problems.

In order to analyze the statistical significance of performance differences, Mann–Whitney U test is applied to model parameters and performance measures. Mann–Whitney tests for differences between two population medians. As shown in Table 2, probability values prove that there is a statistically significant difference between Model F and MILP-1 in terms of CPU time, number of integer variables and constraints. Probability (P) values show whether there is a significant difference between Model F and MILP-1 in terms of model parameters and CPU time performance measure by using predefined α level ($\alpha = 0.05$). If P value is lower than the α, this means that there is a statistically significant difference between two population medians.

The number of integer variables as well as the number of constrains in MILP-1 are far smaller than those in Model F. In both of the models the bulk of the binary variables and the constraints are used for avoiding the machines from doing more than one operation at the same instant.

Table 1
The computational results of Model F and MILP-1.

Problem No.	Size $(i \cdot j \cdot k)$	Model F					MILP-1				
		Integer	Non-integer	Constraints	CPU time (s)	C_{max}	Integer	Non-integer	Constraints	CPU time (s)	C_{max}
SFJS1	$2 \cdot 2 \cdot 2$	40	26	134	0	66[c]	16	19	42	0.02	66[c]
SFJS2	$2 \cdot 2 \cdot 2$	32	24	108	0	107[c]	10	15	30	0.00	107[c]
SFJS3	$3 \cdot 2 \cdot 2$	72	36	234	7	221[c]	26	24	67	0.02	221[c]
SFJS4	$3 \cdot 2 \cdot 2$	84	38	236	11	355[c]	26	24	67	0.00	355[c]
SFJS5	$3 \cdot 2 \cdot 2$	84	38	272	87	119[c]	36	28	87	0.06	119[c]
SFJS6	$3 \cdot 3 \cdot 2$	189	50	497	129	320[c]	39	34	99	0.03	320[c]
SFJS7	$3 \cdot 3 \cdot 5$	225	55	598	135	397[c]	36	40	93	0.02	397[c]
SFJS8	$3 \cdot 3 \cdot 4$	216	55	589	116	253[c]	45	40	111	0.02	253[c]
SFJS9	$3 \cdot 3 \cdot 3$	243	56	584	319	210[c]	55	40	131	0.03	210[c]
SFJS10	$4 \cdot 3 \cdot 5$	300	66	862	510	516[c]	48	45	124	0.02	516[c]
MFJS1	$5 \cdot 3 \cdot 6$	720	99	1829	3600	(396; 470)[a]	103	72	241	0.44	468[c]
MFJS2	$5 \cdot 3 \cdot 7$	840	106	1986	3600	(396; 484)	128	84	291	6.49	446[c]
MFJS3	$6 \cdot 3 \cdot 7$	1260	131	2819	3600	(396; 564)	190	103	422	4.14	466[c]
MFJS4	$7 \cdot 3 \cdot 7$	1617	149	3789	3600	(496; 684)	250	120	549	1779.03	564[c]
MFJS5	$7 \cdot 3 \cdot 7$	1617	149	3726	3600	(414; 696)	243	118	535	50.98	514[c]
MFJS6	$8 \cdot 3 \cdot 7$	2184	174	4766	3600	(469; 786)	307	133	670	3600	(614; 635)[b]
MFJS7	$8 \cdot 4 \cdot 7$	3584	219	7883	3600	(619; 1433)	475	165	1022	3600	(764; 935)
MFJS8	$9 \cdot 4 \cdot 8$	4896	256	9778	3600	(619; 1914)	519	182	1119	3600	(764; 905)
MFJS9	$11 \cdot 4 \cdot 8$	7040	308	14190	3600	(764; 2908)	751	218	1601	3600	(807.64; 1192)
MFJS10	$12 \cdot 4 \cdot 8$	8832	346	16784	3600	(944; 4960)	899	237	1906	3600	(944; 1276)

[a] (IP bound; best IP).
[b] (Best LP bound; best solution).
[c] Optimal C_{max} value.

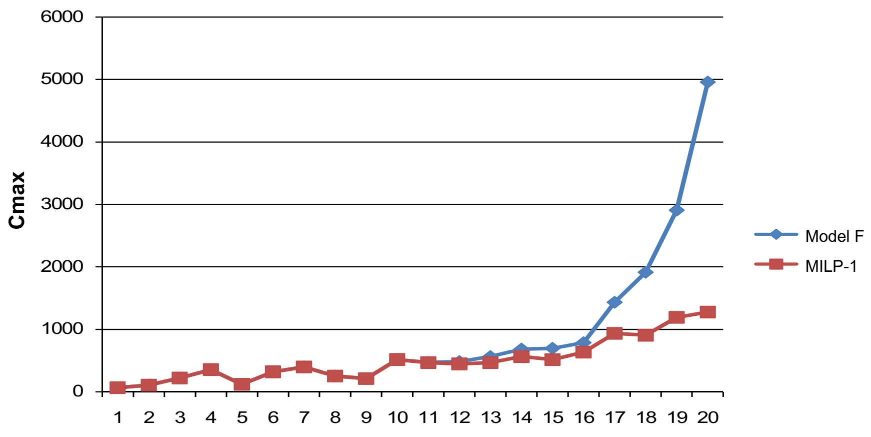

Fig. 1. C_{max} comparison of the results for SFJS and MFJS.

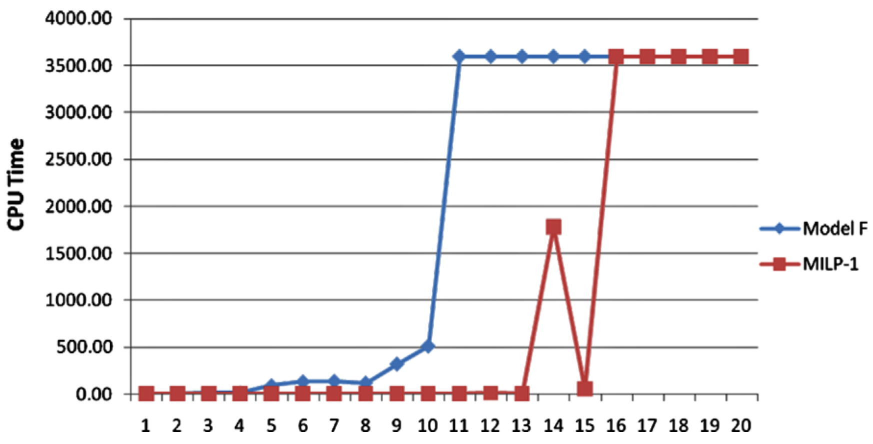

Fig. 2. CPU time comparison of the results for SFJS and MFJS.

Table 2
Mann–Whitney *U* test results for Model F and MILP-1.

	$C_{\max}$	CPU time	# Integer variables	# Non-integer variables	# Constraints
P	0.4154	0.0003	0.0001	0.1012	0.0000

In Model F, each operation of each job is assigned to a machine and then is further assigned to a position on that machine. The immediate predecessors of the operations on the machines are thereby implicitly determined. The authors of Model F have accordingly defined binary variables X_{ijhk}, in line with the approach first adopted by Wagner [36], where i ($i = 1,2,\ldots,m$), j ($j = 1,2,\ldots,n$), h ($h = 1,2,\ldots,h_j$), and k ($k = 1,2,\ldots,k_i$) denote the machines, the jobs, the operations, and the positions on the machines, respectively. The number of X_{ijhk} variables adds up to $mnh_jk_i = m(nh_j)^2$ under the assumptions that h_j is the same for each job and that each operation of each job can be assigned to each position on each one of the machines.

We have instead used, in line with the approach first adopted by Manne [35], the binary variables $Y_{iji'j'k}$ where operation O_{ij} is not necessarily the immediate predecessor of operation $O_{i'j'}$. The number of $Y_{iji'j'k}$ variables adds up to $m(n-1)h_j^2$ under the same assumptions excluding the one about the positions on the machines.

In terms of the $2 \cdot 2 \cdot 2$ problem (the first one in Table 1) the number of X_{ijhk} variables is 32, while the number of $Y_{iji'j'k}$ variables is only 8. The only other binary variables Y_{ijh} in Model F and X_{ijk} in MILP-1 used for assigning the operations of the jobs to the machines are the same in number, being equal to 8 in the case of Problem $2 \cdot 2 \cdot 2$.

As indicated in Table 1, the difference between the models in terms of the total number of binary variables and of the total number of the constraints grows as the size of the problems gets larger. This results from the fact that Model F rests on the notion of the positions on the machines, whereas our model does not.

The outcomes of MILP-1 are also compared to those of the six different heuristic algorithms proposed by Fattahi et al. [20]. On each one of the same test problems, Fattahi et al. [20] have run the algorithms 10 times. The best $C_{\max}$ values they have found are presented in Table 3. No information is available as to whether the CPU times are the average or the lowest times.

In terms of the 10 problems in the SFJS category, the CPU times of MILP-1 are all below 1 s and lower than those of the six heuristic algorithms developed by Fattahi et al. [20]. In five of the problems, $C_{\max}$ values found by HTS/SA, ISA, and ITS are higher than the optimum values found by MILP-1.

As for the 10 problems in the MFJS category, in the first three problems the CPU times of MILP-1 are all lower than those of the heuristic algorithms. Except for MFJS9, $C_{\max}$ values found by all of the heuristic algorithms are higher than the ones found by MILP-1.

Table 3
The computational results of MILP-1 and the heuristic algorithms.

Problem No.	Size ($i \cdot j \cdot k$)	MILP-1		HSA/SA[a]		HSA/TS[b]		HTS/TS[c]		HTS/SA[d]		ISA[e]		ITS[f]	
		CPU	$C_{\max}$	CPU	$C_{\max}$	CPU	$C_{\max}$	CPU	$C_{\max}$	CPU	$C_{\max}$	CPU	$C_{\max}$	CPU	$C_{\max}$
SFJS1	$2 \cdot 2 \cdot 2$	0.02	66[g]	12	66[g]	1	66[g]	1	66[g]	2	66[g]	25	66[g]	1	66[g]
SFJS2	$2 \cdot 2 \cdot 2$	0.00	107[g]	13	107[g]	1	107[g]	1	107[g]	3	107[g]	35	107[g]	1	107[g]
SFJS3	$3 \cdot 2 \cdot 2$	0.02	221[g]	14	221[g]	1	221[g]	1	221[g]	5	221[g]	40	221[g]	1	221[g]
SFJS4	$3 \cdot 2 \cdot 2$	0.00	355[g]	14	355[g]	1	355[g]	1	355[g]	7	355[g]	45	355[g]	1	390
SFJS5	$3 \cdot 2 \cdot 2$	0.06	119[g]	14	119[g]	2	119[g]	1	119[g]	9	119[g]	50	119[g]	1	137
SFJS6	$3 \cdot 3 \cdot 2$	0.03	320[g]	18	320[g]	3	320[g]	1	320[g]	7	320[g]	50	320[g]	2	320[g]
SFJS7	$3 \cdot 3 \cdot 5$	0.02	397[g]	16	397[g]	4	397[g]	1	397[g]	9	397[g]	55	397[g]	2	397[g]
SFJS8	$3 \cdot 3 \cdot 4$	0.02	253[g]	17	253[g]	5	253[g]	2	253[g]	10	256	35	253[g]	2	253[g]
SFJS9	$3 \cdot 3 \cdot 3$	0.03	210[g]	19	210[g]	6	210[g]	2	210[g]	11	210[g]	55	215	3	215
SFJS10	$4 \cdot 3 \cdot 5$	0.02	516[g]	21	516[g]	7	516[g]	4	516[g]	10	516[g]	55	516[g]	3	617
MFJS1	$5 \cdot 3 \cdot 6$	0.44	468[g]	22	479	55	491	15	469	30	469	60	488	9	548
MFJS2	$5 \cdot 3 \cdot 7$	6.49	446[g]	62	495	55	482	12	482	30	468	60	478	8	457
MFJS3	$6 \cdot 3 \cdot 7$	4.14	466[g]	82	553	70	538	20	533	50	538	107	599	8	606
MFJS4	$7 \cdot 3 \cdot 7$	1779.03	564[g]	102	656	85	650	27	634	80	618	195	703	9	870
MFJS5	$7 \cdot 3 \cdot 7$	50.98	514[g]	105	650	110	662	40	625	64	625	240	674	10	729
MFJS6	$8 \cdot 3 \cdot 7$	3600	635	125	762	130	785	96	717	102	730	330	856	50	816
MFJS7	$8 \cdot 4 \cdot 7$	3600	935	197	1020	290	1081	129	964	190	947	480	1066	240	1048
MFJS8	$9 \cdot 4 \cdot 8$	3600	905	230	1030	325	1122	405	970	182	922	610	1328	370	1220
MFJS9	$11 \cdot 4 \cdot 8$	3600	1192	330	1180	660	1243	660	1105	330	1105	840	1148	680	1124
MFJS10	$12 \cdot 4 \cdot 8$	3600	1276	425	1538	600	1615	960	1404	430	1384	850	1546	763	1737

[a] Hierarchical approach and simulated annealing (SA) heuristic for assignment problem and SA heuristic for sequencing problem.
[b] Hierarchical approach and SA heuristic for assignment problem and tabu search (TS) heuristic for sequencing problem.
[c] Hierarchical approach and TS heuristic for assignment problem and TS heuristic for sequencing problem.
[d] Hierarchical approach and TS heuristic for assignment problem and SA heuristic for sequencing problem.
[e] Integrated approach with SA heuristic.
[f] Integrated approach with TS heuristic.
[g] Optimal $C_{\max}$ value.

4. MILP-2 model for FJSP-PPF

MILP-1 is extended to enable it to handle the process plan flexibility. The outcome is MILP-2.

4.1. Presentation of the model

The following notation is used for the formulation of MILP-2.

Indices and sets (in addition to the ones used in Section 3)

$p_{(i)}$	process plans of job i ($p_{(i)} \in P_i$)
P_i	the set of alternative process plans for job i
$O_{ip_{(i)}}$	ordered set of operations in the process plan $p_{(i)}$of job i ($O_{ip_{(i)}} \subseteq O$), where $O_{ip_{(i)}f_{(i)}}$ is the first and $O_{ip_{(i)}\ell_{(i)}}$ is the last element of $O_{ip_{(i)}}$
O_i	the set of operations in all process plans of job i, ($O_i = \bigcup_{p_{(i)} \in P_i} O_{ip_{(i)}}$)

Parameters

$t_{ip_{(i)}jk}$	the processing time of operation $O_{ip_{(i)}j}$ on machine k
L	a large number

Decision variables

$Z_{ip_{(i)}}$	1, if process plan $p_{(i)}$ of job i is selected; 0, otherwise
$X_{ip_{(i)}jk}$	1, if machine k is selected for operation $O_{ip_{(i)}j}$; 0, otherwise
$S_{ip_{(i)}jk}$	the starting time of operation $O_{ip_{(i)}j}$ on machine k
$C_{ip_{(i)}jk}$	the completion time of operation $O_{ip_{(i)}j}$ on machine k
$Y_{iji'j'k}$	1, if operation O_{ij} precedes operation $O_{i'j'}$ on machine k; 0, otherwise
C_i	the completion time of job i
$C_{\max}$	maximum completion time over all jobs (makespan)

The proposed mathematical model is defined as follows:
Objective function: Minimize $C_{\max}$
Constraints:

$$\sum_{k \in M_j} X_{ip_{(i)}jk} = Z_{ip_{(i)}} \quad \forall i \in J,\ \forall p_{(i)} \in P_i,\ \forall j \in O_{ip_{(i)}}, \tag{9}$$

$$\sum_{p_{(i)} \in P_i} Z_{ip_{(i)}} = 1 \quad \forall i \in J, \tag{10}$$

$$S_{ip_{(i)}jk} + C_{ip_{(i)}jk} \leqslant (X_{ip_{(i)}jk}) \cdot L \quad \forall i \in J,\ \forall p_{(i)} \in P_i,\ \forall j \in O_{ip_{(i)}},\ \forall k \in M_j, \tag{11}$$

$$C_{ip_{(i)}jk} \geqslant S_{ip_{(i)}jk} + t_{ip_{(i)}jk} - (1 - X_{ip_{(i)}jk}) \cdot L \quad \forall i \in J,\ \forall p_{(i)} \in P_i,\ \forall j \in O_{ip_{(i)}},\ \forall k \in M_j, \tag{12}$$

$$\sum_{p_{(i)} \in P_i} S_{ip_{(i)}jk} \geqslant \sum_{p'_{(i)} \in P_{i'}} C_{i'p'_{(i)}j'k} - (Y_{iji'j'k}) \cdot L \quad \forall i < i',\ \forall j \in O_{ip_{(i)}},\ \forall j' \in O_{i'p'_{(i')}},\ \forall k \in M_j \cap M_{j'}, \tag{13}$$

$$\sum_{p'_{(i)} \in P_{i'}} S_{i'p'_{(i)}j'k} \geqslant \sum_{p_{(i)} \in P_i} C_{ip_{(i)}jk} - (1 - Y_{iji'j'k}) \cdot L \quad \forall i < i',\ \forall j \in O_{ip_{(i)}},\ \forall j' \in O_{i'p'_{(i')}},\ \forall k \in M_j \cap M_{j'}, \tag{14}$$

$$\sum_{k \in M_j} S_{ip_{(i)}jk} \geqslant \sum_{k \in M_j} C_{ip_{(i)}j-1,k} \quad \forall i \in J, \forall p_{(i)} \in P_i,\ \forall j \in O_{ip_{(i)}} - \{O_{ip_{(i)}f_{(i)}}\}, \tag{15}$$

$$C_i \geqslant \sum_{k \in M_j} C_{ip_{(i)},O_{ip_{(i)}\ell_{(i)}},k} \quad \forall i \in J,\ \forall p_{(i)} \in P_i, \tag{16}$$

$$C_{\max} \geqslant C_i \quad \forall i \in J, \tag{17}$$

and

$$Z_{ip_{(i)}} \in \{0,1\} \quad \forall i \in J,\ \forall p_{(i)} \in P_i,$$
$$X_{ip_{(i)}jk} \in \{0,1\} \quad \forall i \in J,\ \forall p_{(i)} \in P_i,\ \forall j \in O_{ip_{(i)}},\ \forall k \in M_j,$$
$$S_{ip_{(i)}jk} \geqslant 0 \quad \forall i \in J,\ \forall p_{(i)} \in P_i,\ \forall j \in O_{ip_{(i)}},\ \forall k \in M_j,$$
$$C_{ip_{(i)}jk} \geqslant 0 \quad \forall i \in J,\ \forall p_{(i)} \in P_i,\ \forall j \in O_{ip_{(i)}},\ \forall k \in M_j,$$
$$Y_{iji'j'k} \in \{0,1\} \quad \forall i < i',\ \forall j \in O_i,\ \forall j' \in O_{i'},\ \forall k \in M_j \cap M_{j'},$$
$$C_i \geqslant 0 \quad \forall i \in J$$

Constraints (9) and (10) make sure that only one process plan is adopted for job i and that operation $O_{ip_{(i)}j}$ is assigned to only one machine. If operation $O_{ip_{(i)}j}$ is not assigned to machine k, the constraints (11) set the starting and completion times of it on machine k equal to zero. If it is assigned, the constraints (12) guarantee that the difference between the starting and the completion times is at least the processing time on machine k. Constraints (13) and (14) take care of the requirement that operation O_{ij} and operation $O_{i'j'}$ in their adopted process plans cannot be done at the same time on any machine in the set $M_j \cap M_{j'}$. Constraints (15) ensure that the precedence relationships between the operations of a job are not violated. Constraints (16) determine the completion times of the jobs and constraints (17) determine the makespan.

4.2. Computational results

To our knowledge, there is no comparable mathematical model in the literature. Therefore, this section is confined only to the application of MILP-2 on the test problems generated by Özbakır [38].

Özbakır [38] has generated hypothetical flexible job shop problems with different sizes to analyze the solution capability of MILP-2 on the FSJP-PPF. Due to the existence of different process plan and routing flexibility levels, each problem size represents 4 different FJSPs. Table 4 shows the process plan and routing flexibility levels with respect to each problem size. Table 5 presents the systematical approach used for determining the operations, machine alternatives for operations and processing times of operations for each job.

The details of the first test problem P1–11 (5×5, low process plan flexibility-low routing flexibility) are given in Table 6. In this test problem with five jobs, two process plans, six operations, and five machines, there are two alternative process plans with a given ordered set of operations for each job and two alternative machines for each operation. The number of the operations in the process plans range from 2 to 4.

The solution obtained by MILP-2 for P1–11 and the corresponding Gantt chart is presented in Table 7 and Fig. 3. Each operation in the adopted process plan is assigned to the machine indicated in the parenthesis. The resulting $C_{\max}$ value is 193.

Table 4
Process plan and routing flexibility levels for different problem sizes.

Problems	Problem size ($n \times m$)	Process plan flexibility		Routing flexibility	
		Low	High	Low	High
P1	5×5	2	3	2	3
P2	10×10	2	3	2	5
P3	20×5	2	3	2	3

Table 5
Test problems with different routing flexibility levels.

Number of jobs (n)	5, 10, 20
Number of machines (m)	5, 10
# Operations for job i	Uniform[$m/2, m+1$]
m_j (machine for $j \cdot$ operation)	Uniform[$1, m$]
t_{ijk} ($i \cdot$ job $j \cdot$ operation processing time)	t_{ij1} Uniform[1,99]
$k \in M_j$	$t_{ij2\ldots}$ Uniform[t_{ij1}, $3 \times t_{ij1}$]

Table 6
The data for P1–11.

Alternative process plans for jobs			Alternative machines for operations and processing times						
Jobs	Process plans	Operation sequence	Operations	Machines	Jobs				
					1	2	3	4	5
1	1	$4 \to 5 \to 2 \to 3$	1	3	–	68	58	–	55
	2	$5 \to 4 \to 2 \to 3$		5	–	84	74	–	85
2	1	$1 \to 2 \to 4$	2	2	40	10	–	7	–
	2	$1 \to 4 \to 2$		3	88	15	–	13	–
3	1	$3 \to 5 \to 1 \to 4$	3	5	5	–	18	40	17
	2	$1 \to 5 \to 3 \to 4$		4	10	–	52	47	42
4	1	$2 \to 3 \to 5 \to 4$	4	4	71	87	14	11	64
	2	$3 \to 2 \to 5 \to 4$		2	74	88	22	30	65
5	1	$4 \to 1 \to 3$	5	2	18	–	56	66	–
	2	$4 \to 6$		3	22	–	64	80	–
			6	2	–	–	–	–	74
				1	–	–	–	–	88

Table 7
The solution of P1–11.

Jobs	Process plans	Operation (machine) sequence	Starting and completion times of operations
1	2	5(2) → 4(2) → 2(2) → 3(5)	(0–18), (74–148), (148–188), (188–193)
2	2	1(3) → 4(4) → 2(3)	(0–68), (68–155), (161–176)
3	1	3(5) → 5(2) → 1(5) → 4(4)	(0–18), (18–74), (74–148), (172–186)
4	2	3(5) → 2(3) → 5(3) → 4(4)	(18–58), (68–81), (81–161), (161–172)
5	2	4(4) → 6(1)	(0–64), (64–152)

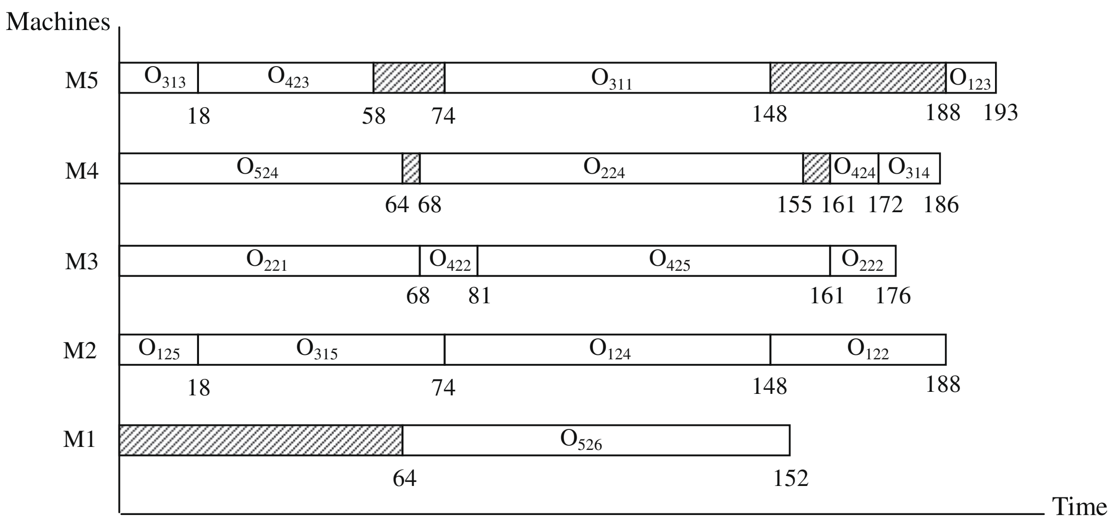

Fig. 3. Gantt chart for the solution of P1–11.

Table 8
The computational results of MILP-2.

Problem No.	Problem size								Model size			Result	
	Jobs	Machines	PPFL[a]	RFL[b]	Process plans[c]	Operations[d]	Op_Range[e]	Mac_Op[f]	Integers	Non-Integers	Constraints	CPU time (s)	C_{max}
P1–11	5	5	Low	Low	2	6	2–4	2	220	146	500	119.22	193[h]
P1–12	5	5	Low	High	2	6	2–4	3	391	216	842	2411.08	171[h]
P1–21	5	5	High	Low	3	7	2–4	2	273	214	630	699.97	193[h]
P1–22	5	5	High	High	3	7	2–4	3	477	318	1038	3600	(165; 179)[g]
P2–11	10	10	Low	Low	2	11	6–10	2	1614	567	3486	3600	(519; 570)
P2–12	10	10	Low	High	2	11	6–10	5	7818	1401	15,894	3600	(519; 674)
P2–21	10	10	High	Low	3	11	6–10	2	1766	851	3912	3600	(519; 598)
P2–22	10	10	High	High	3	11	6–10	5	8183	2111	16,746	3600	(519; 1456)
P3–11	20	5	Low	Low	2	7	2–4	2	3568	561	7366	3600	(301; 808)
P3–12	20	5	Low	High	2	7	2–4	3	6653	831	13,536	3600	(301; 819)
P3–21	20	5	High	Low	3	7	2–4	2	3720	825	7762	3600	(301; 855)
P3–22	20	5	High	High	3	7	2–4	3	6871	1227	14064	3600	(301; 775)

[a] Process plan flexibility level.
[b] Routing flexibility level.
[c] Number of the alternative process plans for each job.
[d] The total number of the operations.
[e] Range of the number of the operations in the process plans of the jobs.
[f] Number of alternative machines for each operation.
[g] (Best LP bound; best solution).
[h] Optimal C_{max} value.

The test problems are run under the same environment mentioned in Section 3.2 for Model F and MILP-1. The solutions obtained within a time limit of 3600-s are presented in Table 8.

MILP-2 finds the optimal solutions to the first three problems. For the remaining ones, as the size of the problem increases, the gap between the best LP bound and the best integer solution widens. It is also observed that the number of binary variables and constraints in MILP-2 are sensitive mainly to the number of alternative machines in the test problems.

5. Conclusion

This study started with the FJSP, the generalized form of the JSP. In the first step, a mixed-integer linear programming model (MILP-1) is developed for the FJSPs. MILP-1 is then compared to the only alternative model we found in the literature and its superiority over Model F in terms of model size, CPU time and $C_{\max}$ value is displayed. In the second step, process plan flexibility is incorporated into FJSP, and MILP-1 is modified accordingly to give rise to MILP-2, for which there appears to be no comparable mathematical model in the literature. Hypothetical test problems that involve routing and process plan flexibility are used for testing MILP-2. The computational results on a variety of test problems with different sizes and flexibility levels are presented and evaluated.

Makespan is adopted as the single performance measure in MILP-1 due to the fact that it is the single measure used in the model that MILP-1 is compared with in this paper, namely Model F. Since it is an extension of MILP-1, we chose to retain the makespan as the single measure in MILP-2. It must however be mentioned that additional measures such as total tardiness, the number of tardy jobs and the load balance of the machines can readily be incorporated into MILP-2.

The authors of this paper are planning as a future study to develop a general framework model that embodies sequence-dependent setup times and process plan costs along with routing and process plan flexibilities.

References

[1] S.C. Graves, A review of production scheduling, Oper. Res. 29 (4) (1981) 646–675.
[2] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, D.B. Shmoys, Sequencing and scheduling: algorithms and complexity, in: S.C. Graves et al. (Eds.), Handbooks in OR and MS, vol. 4, Elsevier Science Publishers B.V., 1993, pp. 445–522.
[3] C.-Y. Lee, L. Lei, M. Pinedo, Current trends in deterministic scheduling, Ann. Oper. Res. 70 (1997) 1–41.
[4] A.S. Jain, S. Meeran, Deterministic job-shop scheduling: past, present and future, Eur. J. Oper. Res. 113 (1999) 390–434.
[5] J.C. Beck, M.S. Fox, Constraint-directed techniques for scheduling alternative activities, Artif. Intell. 121 (2000) 211–250.
[6] I. Kacem, S. Hammadi, P. Borne, Pareto-optimality approach for flexible job-shop scheduling problems: hybridization of evolutionary algorithms and fuzzy logic, Math. Comput. Simulat. 60 (2002) 245–276.
[7] F.T.S. Chan, T.C. Wong, L.Y. Chan, Flexible job-shop scheduling problem under resource constraints, Int. J. Prod. Res. 44 (11) (2006) 2071–2089.
[8] N.B. Ho, J.C. Tay, E.M.-K. Lai, An effective architecture for learning and evolving flexible job-shop schedules, Eur. J. Oper. Res. 179 (2007) 316–333.
[9] P. Brandimarte, Routing and scheduling in a flexible job shop by tabu search, Ann. Oper. Res. 41 (3) (1993) 157–183.
[10] S. Dauzère-Pérès, J. Paulli, An integrated approach for modeling and solving the general multiprocessor job-shop scheduling problem using tabu search, Ann. Oper. Res. 70 (1997) 281–306.
[11] S. Dauzère-Pérès, W. Roux, J.B. Lasserre, Multi-resource shop scheduling with resource flexibility, Eur. J. Oper. Res. 107 (1998) 289–305.
[12] C.R. Scrich, V.A. Armentano, M. Laguna, Tardiness minimization in a flexible job shop: a tabu search approach, J. Intell. Manuf. 15 (2004) 103–115.
[13] M. Saidi-Mehrabad, P. Fattahi, Flexible job shop scheduling with tabu search algorithms, Int. J. Adv. Manuf. Technol. 32 (2007) 563–570.
[14] C. Low, T.-H. Wu, Mathematical modelling and heuristic approaches to operation scheduling problems in an FMS environment, Int. J. Prod. Res. 39 (4) (2001) 689–708.
[15] T. Loukil, J. Teghem, P. Fortemps, A multi-objective production scheduling case study solved by simulated annealing, Eur. J. Oper. Res. 179 (2007) 709–722.
[16] A. Rossi, G. Dini, Flexible job-shop scheduling with routing flexibility and separable setup times using ant colony optimisation method, Robot. Comput. Integr. Manuf. 23 (2007) 503–516.
[17] W. Xia, Z. Wu, An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems, Comput. Ind. Eng. 48 (2005) 409–425.
[18] J. Gao, M. Gen, L. Sun, Scheduling jobs and maintenances in flexible job shop with a hybrid genetic algorithm, J. Intell. Manuf. 17 (2006) 493–507.
[19] C. Low, Y. Yip, T.-H. Wu, Modelling and heuristics of FMS scheduling with multiple objectives, Comput. Oper. Res. 33 (2006) 674–694.
[20] P. Fattahi, M. Saidi-Mehrabad, F. Jolai, Mathematical modeling and heuristic approaches to flexible job shop scheduling problems, J. Intell. Manuf. 18 (2007) 331–342.
[21] J. Gao, L. Sun, M. Gen, A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems, Comput. Oper. Res. 35 (2008) 2892–2907.
[22] G. Vilcot, J.-C. Billaut, A tabu search and a genetic algorithm for solving a bicriteria general job shop scheduling problem, Eur. J. Oper. Res. 190 (2008) 398–411.
[23] Y.H. Lee, C.S. Jeong, C. Moon, Advanced planning and scheduling with outsourcing in manufacturing supply chain, Comput. Ind. Eng. 43 (2002) 351–374.
[24] Y.K. Kim, K. Park, J. Ko, A symbiotic evolutionary algorithm for the integration of process planning and job shop scheduling, Comput. Oper. Res. 30 (2003) 1151–1171.
[25] B.J. Park, H.R. Choi, A genetic algorithm for integration of process planning and scheduling in a job shop, in: A. Sattar, B.H. Kang (Eds.), AI 2006, LNAI, vol. 4304, Springer-Verlag, Berlin, Heidelberg, 2006, pp. 647–657.
[26] A. Baykasoğlu, L. Özbakır, A.İ. Sönmez, Using multiple objective tabu search and grammars to model and solve multi-objective flexible job-shop scheduling problems, J. Intell. Manuf. 15 (6) (2004) 777–785.
[27] A. Baykasoğlu, Linguistic-based meta-heuristic optimization model for flexible job-shop scheduling, Int. J. Prod. Res. 40 (17) (2002) 4523–4543.
[28] J.F. Shapiro, Mathematical programming models and methods for production planning and scheduling, in: S.C. Graves et al. (Eds.), Handbooks in OR and MS, vol. 4, Elsevier Science Publishers B.V., 1993, pp. 371–443.
[29] C.-H. Pan, A study of integer programming formulations for scheduling problems, Int. J. Syst. Sci. 28 (1) (1997) 33–41.
[30] K.-H. Kim, P.J. Egbelu, Scheduling in a production environment with multiple process plans per job, Int. J. Prod. Res. 37 (12) (1999) 2725–2753.
[31] P. Brandimarte, Exploiting process plan flexibility in production scheduling: a multi-objective approach, Eur. J. Oper. Res. 114 (1999) 59–71.
[32] C.S. Thomalla, Job shop scheduling with alternative process plans, Int. J. Prod. Econ. 74 (2001) 125–134.
[33] I.-C. Choi, D.-S. Choi, A local search algorithm for jobshop scheduling problems with alternative operations and sequence-dependent setups, Comput. Ind. Eng. 42 (2002) 43–58.
[34] M.C. Gomes, A.P. Barbosa-Povoa, A.Q. Novais, Optimal scheduling for flexible job shop operation, Int. J. Prod. Res. 43 (11) (2005) 2323–2353.
[35] A.S. Manne, On the job-shop scheduling problem, Oper. Res. 8 (1960) 219–223.
[36] H.M. Wagner, An integer linear-programming model for machine scheduling, Nav. Res. Logist. Quart. 6 (1959) 131–140.
[37] E.H. Bowman, The scheduling-sequence problem, Oper. Res. 7 (1959) 621–624.
[38] L. Özbakır, Modeling, Analyzing and Solution of Multiple Objective Flexible Job Shop Scheduling Problems by Using Meta-heuristic Algorithms, Ph.D. Dissertation, Erciyes University, Social Sciences Institute, Kayseri, Turkey, 2004 (in Turkish).