Received: 18 January 2023 | Accepted: 5 February 2023

DOI: 10.1111/poms.13977

POMS

ORIGINAL ARTICLE

On the multiresource flexible job-shop scheduling problem with arbitrary precedence graphs

Gregory A. Kasapidis[1,2] | **Stéphane Dauzère-Pérès**[3,4] | **Dimitris C. Paraskevopoulos**[5] | **Panagiotis P. Repoussis**[6] | **Christos D. Tarantilis**[2]

[1]Department of Operations and Supply Chain Management, Management School, University of Liverpool, Liverpool, UK

[2]Department of Management Science and Technology, School of Business, Athens University of Economics and Business, Athens, Greece

[3]Department of Manufacturing Sciences and Logistics, Mines Saint-Etienne, CNRS, UMR 6158 LIMOS, Gardanne, France

[4]Department of Accounting and Operations Management, BI Norwegian Business School, Oslo, Norway

[5]Bayes Business School (formerly Cass), City, University of London, London, UK

[6]Department of Marketing and Communication, School of Business, Athens University of Economics and Business, Athens, Greece

Correspondence
Gregory A. Kasapidis, Department of Operations and Supply Chain Management, Management School, University of Liverpool, Liverpool, UK.
Email: gkasapidis@aueb.gr

Stéphane Dauzère-Pérès, Department of Manufacturing Sciences and Logistics, Mines Saint-Etienne, CNRS, UMR 6158 LIMOS, Gardanne, France.
Email: dauzere-peres@emse.fr

Handling Editor: Panos Kouvelis

Abstract

This paper aims at linking the work presented in Dauzère-Pérès et al. (1998) and more recently in Kasapidis et al. (2021) on the multiresource flexible job-shop scheduling problem with nonlinear routes or equivalently with arbitrary precedence graphs. In particular, we present a mixed integer linear programming (MIP) model and a constraint programming (CP) model to formulate the problem. We also compare the theorems introduced in Dauzère-Pérès et al. (1998) and Kasapidis et al. (2021) and propose a new theorem extension. Computational experiments were conducted to assess the efficiency and effectiveness of all propositions. Lastly, the proposed MIP and CP models are tested on benchmark problems of the literature and comparisons are made with state-of-the-art algorithms.

KEYWORDS
arbitrary precedence graphs, constraint programming, flexible job shop scheduling, integer linear programming, multiple resources, nonlinear precedence constraints

1 | INTRODUCTION

The flexible job-shop scheduling problem (FJSP) is an extension of the classical job-shop scheduling problem, where each operation has a subset of machines on which it can be processed. Hence, operations must also be assigned to, and not only sequenced on, machines. Several relevant extensions of the FJSP have been considered in the literature. In this paper, we are studying the relationships between the work presented in Dauzère-Pérès et al. (1998) and Kasapidis et al. (2021) on the FJSP with nonlinear routes, also called arbitrary precedence graphs. Next, we focus on the multiresource FJSP with arbitrary precedence graphs that was first introduced in Dauzère-Pérès et al. (1998). We present a mixed integer linear program (MILP) and a constraint programming model (CP) for the problem as well as thorough computational experimentation.

To our knowledge, the combination of arbitrary precedence graphs and multiple necessary resources has very rarely been considered in the literature. The FJSP where an arbitrary directed acyclic graph models general precedence constraints between operations has been named differently in Ivens and Lambrecht (1996) (assembly and split structures), Dauzère-Pérès et al. (1998) (nonlinear routes), Schutten (1998) (convergent and divergent job routings), Birgin et al. (2015) (sequencing flexibility), Lunardi et al. (2020), and Kasapidis et al. (2021) (arbitrary precedence constraints). Multiple necessary resources for an operation in the FJSP are explicitly

Accepted by Panos Kouvelis.

considered for the first time in Dauzère-Pérès et al. (1998). Another related extension of the FJSP, where each operation may have multiple modes, is initially studied in Brucker and Neyer (1998). A mode corresponds to a predefined set of resources required by an operation. However, considering multiple modes is different than considering multiple necessary resources as the latter allows more flexibility.

The remainder of this paper is structured as follows. Section 2 provides a detailed description of the problem, while Section 3 introduces the MILP and the CP models. Section 4 presents some comparisons among the theorems presented in Dauzère-Pérès et al. (1998) and Kasapidis et al. (2021), while also proposing a new theorem extension. Section 5 presents and discusses numerical results on benchmark instances, and Section 6 provides conclusions and some future research prospects.

2 | PROBLEM DESCRIPTION

In this section, we adopt the nomenclature for the FJSP with arbitrary precedence graphs as presented in Kasapidis et al. (2021) and extend it so as to incorporate multiple resources following Dauzère-Pérès et al. (1998). The FJSP with arbitrary precedence graphs and multiple resources can be described as follows: There is a set of jobs $J = \{1, \dots, l\}$ to be processed on a set of resources $R = \{1, \dots, m\}$. Every job $u \in J$ consists of a set of operations O_u, and let set $\Omega = \{1, \dots, n\}$ denote the set of all operations of the problem, that is, $\Omega = \bigcup_{u \in J} O_u$. Every operation $i \in \Omega$ requires a set $G_i = \{1, \dots, g_i\}$ of different resources, called necessary resources. Every necessary resource $j \in G_i$ must be selected in a set of available resources $R_{i,j} \in R$. Note that two sets $R_{i,j}$ and $R_{i,j'}$ such that $j \neq j'$ are not necessarily disjoint. However, the same resource cannot be assigned to operation i for multiple necessary resources.

The processing time required to process operation i on a resource $k \in R$ is denoted by $p_{i,k}$. Also, an operation i is assumed to be completed when the processing of all its assigned resources is completed. The total processing time of an operation i is denoted by p_i. Assuming that $\alpha(i,j) \in R_{i,j}$ denotes the resource selected as the necessary resource j of operation i, p_i can be calculated as follows: $p_i = \max_{j \in G_i} p_{i,\alpha(i,j)}$. Moreover, every operation i may have multiple predecessors and successor operations that are denoted by sets PJ_i and SJ_i, respectively. Furthermore, let i_u° and i_u^* denote two dummy operations that correspond to the first and the last operations of a job $u \in J$.

For the sake of completeness, let sets $\mathcal{P}_i$ and $\mathcal{F}_i$ denote the sets of all predecessor and successor operations of operation i. Let also $p^k(i)$ and $f^k(i)$ denote the immediate resource predecessor and successor operations of i on resource $k \in \bigcup_{j \in G_i} R_{i,j}$. Lastly, as in Dauzère-Pérès et al. (1998), we assume that an operation starts simultaneously on all the resources $k \in R$ assigned to the operation and that the resources are occupied for the same amount of time. This policy is called "simultaneous occupation" in this paper.

3 | PROBLEM MODELING

In this section, we present two formulations for the problem: A MILP model in Section 3.1 and a CP model in Section 3.2.

3.1 | MILP model

This section introduces a MILP model for the FJSP with arbitrary precedence graphs and multiple resources with simultaneous occupation constraints. The following variables are considered. Let t_i denote the completion time of operation $i \in \Omega$ and $t_{i,j}$ the completion time of operation $i \in \Omega$ on its jth necessary resource, where $j \in G_i$. Binary variable $Y_{i,j,k}$ is equal to one if resource $k \in R_{i,j}$ is assigned as the jth necessary resource of operation i and zero otherwise. Binary variable $X_{i,i',k}$ is equal to one if two operations i and i' are assigned to the same resource $k \in R$ and i' is processed after i and zero otherwise.

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

subject to

$$\sum_{k \in R_{i,j}} Y_{i,j,k} = 1 \qquad \forall i \in \Omega, \forall j \in G_i \tag{2}$$

$$\sum_{\forall j \in G_i} \sum_{k \in R_{i,j}} Y_{i,j,k} \leq 1 \qquad \forall i \in \Omega \tag{3}$$

$$p_i \geq \sum_{k \in R_{i,j}} Y_{i,j,k} p_{i,k} \qquad \forall i \in \Omega, \forall j \in G_i \tag{4}$$

$$t_i \geq t_{i'} + p_i \qquad \forall i \in \Omega, \forall j \in G_i, \forall i' \in PJ_i \tag{5}$$

$$t_i \geq t_{i'} + p_i - \mathcal{M}(2 + X_{i,i',k} - Y_{i,j,k} - Y_{i',j',k}) \quad \forall i, i' \in \Omega, \forall j \in G_i, \forall j' \in G_{i'}, \forall k \in R_{i,j} \cap R_{i',j'} \tag{6}$$

$$t_{i'} \geq t_i + p_{i'} - \mathcal{M}(3 - X_{i,i',k} - Y_{i,j,k} - Y_{i',j',k}) \quad \forall i, i' \in \Omega, \forall j \in G_i, \forall j' \in G_{i'}, \forall k \in R_{i,j} \cap R_{i',j'} \tag{7}$$

$$t_i \geq 0 \qquad \forall i \in \Omega \tag{8}$$

$$C_{max} \geq t_{i_u^*} \qquad \forall u \in J \tag{9}$$

$$X_{i,i',k} \in \{0,1\} \qquad \forall i, i' \in \Omega, \forall k \in R \tag{10}$$

$$Y_{i,j,k} \in \{0,1\} \qquad \forall i \in \Omega, \forall j \in G_i, \forall k \in R_{i,j} \tag{11}$$

As for the classical FJSP, the objective is to minimize the makespan, see (1). Constraints (2) enforce one available resource k to be used for the processing of the jth necessary resource of operation i. Constraints (3) ensure that an available resource k cannot be used more than once for the requirements of operation i. Constraints (4) are responsible for the calculation of the actual time that the execution of an operation i requires. Constraints (5) ensure that the completion time of every operation i is larger than the completion time of any predecessor operation $j \in PJ_i$. Constraints (6) and (7) guarantee that all operations are processed sequentially by the available resources. Constraints (8) enforce all completion times to be positive. Constraints (9) are used to calculate the makespan, while Constraints (10) and (11) set the domain values for the binary variables X and Y, respectively.

3.2 | CP formulation

In this section, we present the CP formulation for the FJSP with arbitrary precedence graphs and multiple resources with simultaneous occupation. The nomenclature of the IBM CP Optimizer is used. We refer the reader to Kasapidis et al. (2021) for a comprehensive discussion of the key constraint expressions and variable types supported by the IBM CP Optimizer used to model the FJSP and its variants.

A decision interval variable τ_i is defined for every operation $i \in \Omega$ and a decision interval variable $\tau_{i,j}$ for every operation $i \in \Omega$ and necessary resource $j \in G_i$. In addition, the decision interval variable $\phi_{i,j,k}$ is used to represent the different execution modes of the jth necessary resource of an operation i on a resource $k \in R_{i,j}$. Note that the *Size* attribute of decision interval variables $\phi_{i,j,k}$ $\forall i \in \Omega, \forall j \in G_i, \forall k \in R_{i,j}$, is not constrained since the resources are occupied for the entire execution of operation i. The set $\mu_{i,j} = \{\phi_{i,j,k}, \forall k \in R_{i,j}\}$ is used to represent all the available execution modes on the jth necessary resource of operation i. Note that $\mu_{i,j}$ is the domain set of variable $\tau_{i,j}$. Lastly, a sequence interval decision variable σ_k is defined per resource k over the set of interval variables $\sigma_k = \{\phi_{i,j,k}, \forall i \in \Omega, \forall j \in G_i\}$.

$$\text{minimize}\, C_{max} \tag{12}$$

subject to

$$Alternative(\tau_{i,j}, \mu_{i,j}) \qquad \forall i \in \Omega, \forall j \in G_i \tag{13}$$

$$\begin{aligned} &PresenceOf(\phi_{i,j',k}) + PresenceOf(\phi_{i,j,k}) \leq 1 \\ &\quad \forall i \in \Omega, \forall j, j' \in G_i, j' > j, \\ &\quad \forall k \in R_{i,j'} \cap R_{i,j}, j' > j \end{aligned} \tag{14}$$

$$StartOf(\tau_{i,j}) = StartOf(\tau_i) \qquad \forall i \in \Omega, \forall j \in G_i \tag{15}$$

$$EndOf(\tau_i) \geq EndOf(\tau_{i,j}) \qquad \forall i \in \Omega, \forall j \in G_i \tag{16}$$

$$StartOf(\tau_i) \geq EndOf(\tau_{i'}) \qquad \forall i \in \Omega, \forall i' \in PJ_i \tag{17}$$

$$SizeOf(\phi_{i,j,k}) \geq Sizeof(\tau_i) \qquad \forall i \in \Omega, \forall j \in G_i, \forall k \in R_{i,j} \tag{18}$$

$$SizeOf(\phi_{i,j,k}) \geq p_{i,k} \qquad \forall i \in \Omega, \forall j \in G_i, \forall k \in R_{i,j} \tag{19}$$

$$NoOverlap(\sigma_k) \qquad \forall k \in R \tag{20}$$

$$C_{max} \geq EndOf(\tau_i) \qquad \forall i \in \Omega \tag{21}$$

Objective (12) refers to the minimization of the makespan. Constraints (13) are used to select only one available resource k per necessary resource j of operation i, while Constraints (14) ensure that the same resource is not used more than once for the same operation. Constraints (15) ensure that all the necessary resources are occupied simultaneously as soon as operation i starts. Constraints (16) are used to calculate the completion time of operation i. Constraints (17) make sure that precedence relations between operations are respected. Constraints (18) ensure that all the available resources $k \in R_{i,j}$, $\forall j \in G_i$ are occupied for the entire execution of operation i, while Constraints (19) set a lower bound for variables $\phi_{i,j,k}$. Constraints (20) make sure that resources execute only one operation at a time. Lastly, Constraints (21) calculates the objective.

4 | MOVE FEASIBILITY CHECK AND EVALUATION IN A NEIGHBORHOOD-BASED METAHEURISTIC

A common way to model and solve scheduling problems is through a disjunctive graph $D(V, A, E)$, where the set of nodes V represents the operations $i \in \Omega$, plus the dummy start and finish operations 0 and $*$, while the conjunctive arcs in A model the immediate precedence relationships between operations in the route of a job, and disjunctive arcs in E link operations that can be assigned to the same resource $k \in R$.

A solution s of the problem can be represented by a conjunctive graph $G(V, A, S) \subset D$, where S is obtained by replacing a conjunctive arc (when two operations are assigned to the same resource) or deleting (if two operations are not assigned to the same resource) each disjunctive arc in set E. Since the available resources are only capable of processing operations sequentially and operations are processed only once, any graph G that represents a feasible solution should be a directed acyclic graph.

A popular and efficient way to solve the FJSP is to use neighborhood-based metaheuristics that rely on the disjunctive graph model, by performing local "moves" from one conjunctive graph to another. The first integrated move for the FJSP is proposed in Dauzère-Pérès and Paulli (1997), where operation i is indifferently resequenced on the same machine or reassigned to another machine between two operations v and w sequenced consecutively on a machine. Two critical questions need to be answered when designing a neighborhood-based solution approach for the FJSP or one of its extensions: (1) "Is a move feasible?" and (2) "What is the value of the objective function after performing a move?" Both questions can be answered by actually performing a move to check its feasibility and calculate the value of the objective function, that is, the makespan, which requires to traverse the directed graph after the move. However, when the number of possible moves to evaluate is very large, as in the connected neighborhood structure and Tabu Search of Dauzère-Pérès and Paulli (1997), the resulting computational times are prohibitive. Hence, conditions have been proposed in the literature to guarantee feasibility and estimate the makespan without actually performing any move. These conditions rely on the head (length of the longest path from operation 0 to operation i) r_i, the tail (length of the longest path from the end of operation i to operation $*$) q_i, the set $\mathcal{P}_i$ of all predecessors in G and the set $\mathcal{S}_i$ of all successors in G of each operation $i \in \Omega$ (see, e.g., Dauzère-Pérès & Paulli, 1997, for more details).

Regarding move feasibility, Remark 1 specifies that the conditions in Dauzère-Pérès et al. (1998) and Kasapidis et al. (2021), both extended from the ones in Dauzère-Pérès and Paulli (1997), are equivalent. This is because, since the operation is moved on only one resource at a time in Dauzère-Pérès et al. (1998), the graph can be seen as an arbitrary precedence graph (or with nonlinear routes) for the arcs associated with the resources that are not reassigned.

Remark 1. Theorem 1 in Kasapidis et al. (2021) is equivalent to Theorem 1 in Dauzère-Pérès et al. (1998).

Regarding the criterion estimation of a move, Remark 2 specifies that the evaluation in Dauzère-Pérès et al. (1998) and Kasapidis et al. (2021) is different. While the evaluation in Kasapidis et al. (2021) is a direct extension of an arbitrary precedence graph of the evaluation proposed in Dauzère-Pérès and Paulli (1997), the evaluation in Dauzère-Pérès et al. (1998) aims at reducing the computational effort by avoiding enumerating all paths in graph G. More precisely, the evaluation in Dauzère-Pérès et al. (1998) only requires to consider the heads and tails of operations.

Remark 2. Theorem 2 in Kasapidis et al. (2021) is not equivalent to Theorem 5 in Dauzère-Pérès et al. (1998).

Hence, following Remark 2, we propose to further extend Theorem 5 in Dauzère-Pérès and Paulli (1997), already extended in Kasapidis et al. (2021) for an arbitrary precedence graph, to consider multiple necessary resources for operations in the FJSP with an arbitrary precedence graph. Theorem 1 below presents the resulting lower bound.

Theorem 1. *The makespan after moving operation i between two consecutive operations v and w in the available resource $k \in R_{i,j}$, $\forall j \in G_i$, and such that Theorem 1 in Dauzère-Pérès et al. (1998) holds, is always larger than or equal to*

$$LB(i,v,w) = \max\left(\hat{r}_v + p_v, \max_{\forall e \in PJ_i}(r_e + p_e)\right) + \tilde{p}_i + \max\left(\hat{q}_w + p_w, \max_{\forall e \in SJ_i}(q_e + p_e)\right), \quad (22)$$

where

$$\hat{r}_v = \begin{cases} r_v - r_{sm_i} + \max\left(\max_{\forall e \in PJ_{sm_i}}(r_e + p_e), r_{pm_i} + p_{pm_i}\right) & \text{if } i \in \mathcal{P}_v, \\ r_v & \text{if } i \notin \mathcal{P}_v, \end{cases} \quad (23)$$

$$\hat{q}_w = \begin{cases} q_w - q_{pm_i} + \max\left(\max_{\forall e \in SJ_{pm_i}}(q_e + p_e), q_{sm_i} + p_{sm_i}\right) & \text{if } i \in \mathcal{S}_w, \\ q_w & \text{if } i \notin \mathcal{S}_w \end{cases} \quad (24)$$

Proof. *The proof follows the ones of Theorem 5 in Dauzère-Pérès and Paulli (1997) and Theorem 2 in Kasapidis et al. (2021). The only difference lies in the processing times, which are now calculated considering multiple resources as shown in Section 2. Note that the processing time $\tilde{p}_i$ corresponds to the processing time of operation i after the move,that is, $\tilde{p}_i = \max_{j \in G_i} p_{i,\tilde{\alpha}(i,j)}$, where $\tilde{\alpha}(i,j)$ denotes the selected necessary resources after the move.* □

The numerical results of Section 5 show that the evaluation in Dauzère-Pérès et al. (1998) does not significantly reduce the computational times compared to the evaluation in Theorem 1, although the accuracy of the former is poorer than the latter.

Another way of evaluating a move, called the Lpath method, is proposed in Dell'Amico and Trubian (1993) for the classical JSP, that is, when operations are moved to the same machine in the FJSP. The Lpath method is extended for the FJSP in González et al. (2015), and for the FJSP with arbitrary precedence graphs in Kasapidis et al. (2021).

Lastly, note that Dauzère-Pérès et al. (1998) also show that the resulting neighborhood structure is connected, that is, it allows an optimal solution to be reached in a finite number of moves.

5 | COMPUTATIONAL EXPERIMENTS

In this section, we present and discuss the results of the computational experiments conducted in this paper. More specifically, Section 5.1 includes the assessment of Theorems 1 and 5 of Dauzère-Pérès et al. (1998) and the extended Lpath method, while Section 5.2 compares the results of the proposed MILP and CP models to state-of-the-art results using well-known benchmarks of the literature.

TABLE 1 Accuracy assessment of move evaluations on FJSP instances with linear precedence graphs.

Method	$> C_{max}$	$< C_{max}$	C_{max}	Accuracy (%)	Time(s)
Lpath	185,655	580,405	19,233,940	96.17	0.017
Theorem 1	4,123,396	0	15,876,604	79.38	0.010
Theorem 5 of Dauzère-Pérès et al. (1998)	11,470,719	0	8,529,281	42.65	0.008

TABLE 2 Accuracy assessment of move evaluations on FJSP instances with arbitrary precedence graphs.

Method	$> C_{max}$	$< C_{max}$	C_{max}	Accuracy (%)	Time(s)
Lpath	703,986	438,907	18,857,107	94.29	0.016
Theorem 1	2,701,880	0	17,298,120	86.49	0.008
Theorem 5 of Dauzère-Pérès et al. (1998)	10,638,390	0	9,361,610	46.81	0.007

With regard to implementation, the IBM ILOG CPLEX Solver (v22.1.0), respectively the IBM ILOG CP Optimizer (v22.1.0), was used for the MILP model, respectively for the CP model. An Intel Core i7-7700 processor and 16.0GB of RAM were used, with a common time limit of 10,800 s for both the MILP and the CP models.

5.1 | Move evaluation assessment

To assess Theorems 1 and 5 of Dauzère-Pérès et al. (1998) and the extended Lpath method, we used well-known benchmark problems of the literature for the FJSP and the FJSP with arbitrary precedence graphs. Even though these sets of problems do not consider multiple resources, they serve as a suitable test bed. In particular, two different sets of experiments are conducted on two different sets of benchmark problem instances. At first, regarding the problem instances of the FJSP, the following problem instances were used: DP15a and DP18a from the DPData benchmark set (see Dauzère-Pérès & Paulli, 1997) as well as Mk6 and Mk10 from the BRData benchmark set (see Brandimarte, 1993). Second, regarding the FJSP with arbitrary precedence graphs, the five largest available problem instances were used: DAFJS10, DAFJS29, DAFJS30, YFJS19, and YFJS20 from the DAFJS and YFJS benchmark sets provided in Birgin et al. (2014).

In both sets of experiments, the local search procedure of Kasapidis et al. (2021) was used. Each method was evaluated a total of 20 million times, and the results are presented in Tables 1 and 2. The former includes the results on problem instances of the FJSP, that is, with linear precedence graphs, while the latter includes the results on problem instances of the FJSP with arbitrary precedence graphs, that is, with nonlinear routes.

Both tables share the same structure. The first column includes the name of the method, while the next three columns denote the number of times when the estimate was larger than, lower than, or equal to the actual makespan of the move, respectively. The fifth column includes the accuracy of the estimation method, that is, how frequently the estimation method was able to accurately estimate the actual makespan of the move. Lastly, the sixth column includes the time in microseconds (s) that was required on average for a single evaluation of each method.

Overall, one can observe that Lpath shows high precision for all problems. More specifically, Lpath estimates the makespan with an accuracy of 96.17% and 94.29% in the case of linear and nonlinear precedence constraints, respectively. Regarding the other move evaluation methods, we can confirm that both Theorems 1 and 5 of Dauzère-Pérès et al. (1998) produce valid lower bounds since there was no case where the calculated estimate was greater than the actual makespan of a move. We also notice that the accuracy of both methods is lower compared to Lpath.

More specifically, Theorem 1 has an accuracy of 79.38% and 86.49% for problems with linear and nonlinear precedence constraints, respectively. Whereas Theorem 5 of Dauzère-Pérès et al. (1998) has an accuracy of 42.65% and 46.81% for problems with linear and nonlinear precedence constraints, respectively. In terms of performance, Lpath is more computationally expensive than Theorems 1 and 5 of Dauzère-Pérès et al. (1998). More specifically, in both sets of experiments, Lpath is twice as time consuming as the other two move evaluation methods. Note that Theorem 5 of Dauzère-Pérès et al. (1998) is marginally faster compared to Theorem 1. While both Theorems 1 and 5 of Dauzère-Pérès et al. (1998) produce valid lower bounds, one could prefer Lpath as it is more accurate despite the fact that it is more computationally expensive.

5.2 | Comparison of MILP and CP models

In this section, we assess the performance of the MILP and CP models introduced in Sections 3.1 and 3.2, respectively. We used benchmark problem instances of the literature for the multiresource FJSP with arbitrary precedence graphs, in particular, the benchmark set (denoted by MJS) introduced in Dauzère-Pérès et al. (1998). This benchmark set includes a total of 70 instances that can be extended by assuming: (a) linear precedence graphs and (b) a common nonlinear precedence graph, resulting in a total of 140 different instances. As

TABLE 3 Results on the benchmark set of Dauzère-Pérès et al. (1998) (denoted by MJS) with linear precedence constraints.

		DP	CP				MILP			
Instance	Best LB	C_{max}	LB	C_{max}	Gap (%)	Time (s)	LB	C_{max}	Gap (%)	Time (s)
mjs01	361	**361***	361	**361***	0.00	17	354	**361***	0.00	10,800
mjs02	381	384	381	**381***	0.00	25	380	**381***	0.00	10,800
mjs03	376	378	376	**376***	0.00	51	364	381	1.33	10,800
mjs04	391	394	391	**391***	0.00	34	391	**391***	0.00	640
mjs05	623	643	623	**623***	0.00	659	572	659	5.78	10,800
mjs06	547	585	547	**547***	0.00	1171	487	570	4.20	10,800
mjs07	610	644	610	**610***	0.00	10553	520	625	2.46	10,800
mjs08	552	575	552	**552***	0.00	1424	511	585	5.98	10,800
mjs09	563	568	563	**563***	0.00	104	549	585	3.91	10,800
mjs10	454	928	444	**828**	82.30	10,800	454	998	119.73	10,800
mjs11	487	1057	487	**901**	85.01	10,800	444	-	-	10,800
mjs12	446	859	446	**790**	77.11	10,800	446	926	107.60	10,800
mjs13	434	827	434	**791**	82.26	10,800	419	889	104.84	10,800
mjs14	552	946	552	**910**	64.86	10,800	460	1045	89.31	10,800
mjs15	655	1469	655	**1292**	97.25	10,800	655	-	-	10,800
mjs16	581	1312	581	**1198**	106.20	10,800	566	-	-	10,800
mjs17	647	1572	647	**1407**	117.47	10,800	614	-	-	10,800
mjs18	668	1544	668	**1396**	108.98	10,800	655	-	-	10,800
mjs19	674	1572	674	**1321**	95.99	10,800	642	-	-	10,800
mjs20	500	1033	500	**902**	80.40	10,800	499	1156	131.20	10,800
mjs21	438	916	438	**836**	90.87	10,800	419	1021	133.11	10,800
mjs22	467	924	467	**865**	85.22	10,800	444	1176	151.82	10,800
mjs23	475	957	475	**849**	78.74	10,800	452	-	-	10,800
mjs24	433	918	419	**790**	82.44	10,800	433	1025	136.72	10,800
mjs25	653	1513	653	**1315**	101.38	10,800	613	-	-	10,800
mjs26	620	1481	620	**1203**	94.03	10,800	567	-	-	10,800
mjs27	633	1566	633	**1327**	109.64	10,800	612	-	-	10,800
mjs28	610	1395	610	**1325**	117.21	10,800	601	-	-	10,800
mjs29	690	1336	690	**1215**	76.09	10,800	638	1760	155.07	10,800
mjs30	216	218	216	**216***	0.00	50	216	227	5.09	10,800
mjs31	218	**218***	218	**218***	0.00	18	218	220	0.92	10,800
mjs32	216	219	216	**216***	0.00	429	210	236	9.26	10,800
mjs33	217	224	217	**217***	0.00	99	211	230	5.99	10,800
mjs34	213	**213***	213	**213***	0.00	8	213	217	1.88	10,800
mjs35	265	**265***	265	**265***	0.00	4	265	**265***	0.00	1401
mjs36	223	225	223	**223***	0.00	27	219	226	1.35	10,800
mjs37	202	207	202	**202***	0.00	64	189	220	8.91	10,800
mjs38	241	**241***	241	**241***	0.00	5	241	246	2.07	10,800
mjs39	210	**210***	210	**210***	0.00	56	210	217	3.33	10,800
mjs40	241	**241***	241	**241***	0.00	3	241	**241***	0.00	1339
mjs41	210	218	210	**210***	0.00	680	204	232	10.48	10,800
mjs42	250	**250***	250	**250***	0.00	2	250	**250***	0.00	1836
mjs43	219	**219***	219	**219***	0.00	6	219	**219***	0.00	814
mjs44	252	258	252	**252***	0.00	8	252	253	0.40	10,800
mjs45	294	296	294	**294***	0.00	73	294	318	8.16	10,800
mjs46	296	300	296	**296***	0.00	556	292	356	20.27	10,800

(Continues)

TABLE 3 (Continued)

Instance	Best LB	DP	CP				MILP			
		C_{max}	LB	C_{max}	Gap (%)	Time (s)	LB	C_{max}	Gap (%)	Time (s)
mjs47	330	333	330	**330***	0.00	109	330	-	-	10,800
mjs48	315	327	315	**315***	0.00	164	299	-	-	10,800
mjs49	356	**356***	356	**356***	0.00	8	356	-	-	10,800
mjs50	279	327	279	**326**	16.85	10,800	279	-	-	10,800
mjs51	289	373	289	**367**	26.99	10,800	277	-	-	10,800
mjs52	286	**317**	286	**317**	10.84	10,800	286	-	-	10,800
mjs53	267	**353**	267	**353**	32.21	10,800	266	-	-	10,800
mjs54	241	311	241	**299**	24.07	10,800	235	-	-	10,800
mjs56	380	**508**	380	534	40.53	10,800	372	-	-	10,800
mjs59	346	490	346	**476**	37.57	10,800	345	-	-	10,800
mjs60	246	268	246	**246***	0.00	301	243	-	-	10,800
mjs61	301	303	301	**301***	0.00	101	301	-	-	10,800
mjs62	284	**284***	284	**284***	0.00	63	284	298	4.93	10,800
mjs63	286	289	286	**286***	0.00	44	286	297	3.85	10,800
mjs64	240	**240***	240	**240***	0.00	61	240	-	-	10,800
mjs65	375	381	375	**375***	0.00	357	368	-	-	10,800
mjs66	423	**423***	423	**423***	0.00	796	423	-	-	10,800
mjs67	400	408	400	**400***	0.00	1000	399	-	-	10,800
mjs68	382	400	382	**382***	0.00	1189	381	-	-	10,800

Abbreviations: CP, constraint programming; DP, Dauzère-Pérès et al. (1998); LB, lower bound, MILP, mixed integer linear program.
Values in bold are best solutions found, and Values with * are optimal solutions.

TABLE 4 Results on the benchmark set of Dauzère-Pérès et al. (1998) (denoted by MJS) with Arbitrary Precedence Constraints.

Instance	Best LB	DP	CP				MILP			
		C_{max}	LB	C_{max}	Gap (%)	Time (s)	LB	C_{max}	Gap (%)	Time (s)
mjs01	1052	1136	1052	**1052***	0.00	0	1052	**1052***	0.00	2
mjs02	1104	1160	1104	**1104***	0.00	6	1104	**1104***	0.00	5
mjs03	1133	1166	1133	**1133***	0.00	1	1133	**1133***	0.00	9
mjs04	1086	1097	1086	**1086***	0.00	4	1086	**1086***	0.00	14
mjs05	1761	1809	1761	**1761***	0.00	21	1761	**1761***	0.00	50
mjs06	1648	1712	1648	**1648***	0.00	13	1648	**1648***	0.00	52
mjs07	1828	1841	1828	**1828***	0.00	39	1828	**1828***	0.00	198
mjs08	1627	1693	1627	**1627***	0.00	71	1627	**1627***	0.00	148
mjs09	1557	1585	1557	**1557***	0.00	20	1557	**1557***	0.00	40
mjs10	1610	1739	1610	**1610***	0.00	1658	1549	1631	1.30	10,800
mjs11	1637	1817	1637	**1637***	0.00	3472	1495	1697	3.67	10,800
mjs12	1560	1759	1560	**1560***	0.00	1074	1453	1595	2.24	10,800
mjs13	1518	1709	1518	**1518***	0.00	682	1437	1571	3.49	10,800
mjs14	1658	1898	1658	**1658***	0.00	305	1641	**1658***	0.00	10,800
mjs15	2184	2679	2182	**2450**	12.28	10,800	2184	2641	20.92	10,800
mjs16	2096	2458	2096	**2193**	4.63	10,800	1998	2369	13.02	10,800
mjs17	2221	2679	2221	**2454**	10.49	10,800	2166	2624	18.14	10,800
mjs18	2302	2755	2302	**2415**	4.91	10,800	2199	2563	11.34	10,800
mjs19	2275	2832	2275	**2402**	5.58	10,800	2211	2592	13.93	10,800
mjs20	1704	1951	1704	**1704***	0.00	1404	1571	1809	6.16	10,800

(Continues)

TABLE 4 (Continued)

		DP	CP				MILP			
Instance	**Best LB**	C_{max}	**LB**	C_{max}	**Gap (%)**	**Time (s)**	**LB**	C_{max}	**Gap (%)**	**Time (s)**
mjs21	1486	1649	1486	**1486***	0.00	639	1411	1543	3.84	10,800
mjs22	1573	1670	1573	**1573***	0.00	930	1500	1598	1.59	10,800
mjs23	1541	1773	1541	**1541***	0.00	923	1448	1608	4.35	10,800
mjs24	1448	1634	1448	**1448***	0.00	713	1348	1525	5.32	10,800
mjs25	2233	2771	2233	**2379**	6.54	10,800	2138	2542	13.84	10,800
mjs26	2061	2359	2061	**2204**	6.94	10,800	2023	2429	17.86	10,800
mjs27	2214	2703	2214	**2386**	7.77	10,800	2168	2594	17.16	10,800
mjs28	2132	2540	2132	**2279**	6.89	10,800	2072	2585	21.25	10,800
mjs29	2267	2452	2267	**2267***	0.00	7948	2081	2415	6.53	10,800
mjs30	710	721	710	**710***	0.00	2	710	**710***	0.00	73
mjs31	746	772	746	**746***	0.00	2	746	**746***	0.00	189
mjs32	722	743	722	**722***	0.00	3	722	**722***	0.00	75
mjs33	710	730	710	**710***	0.00	0	710	**710***	0.00	50
mjs34	697	760	697	**697***	0.00	1	697	**697***	0.00	403
mjs35	842	849	842	**842***	0.00	0	842	**842***	0.00	35
mjs36	673	690	673	**673***	0.00	2	673	**673***	0.00	137
mjs37	626	687	626	**626***	0.00	2	626	**626***	0.00	493
mjs38	754	774	754	**754***	0.00	2	754	**754***	0.00	66
mjs39	682	695	682	**682***	0.00	2	682	**682***	0.00	147
mjs40	688	698	688	**688***	0.00	1	688	**688***	0.00	250
mjs41	725	750	725	**725***	0.00	1	725	**725***	0.00	37
mjs42	757	773	757	**757***	0.00	1	757	**757***	0.00	163
mjs43	630	687	630	**630***	0.00	0	630	**630***	0.00	28
mjs44	750	828	750	**750***	0.00	2	750	**750***	0.00	53
mjs45	966	986	966	**966***	0.00	12	966	**966***	0.00	3555
mjs46	1010	1034	1010	**1010***	0.00	4	1010	**1010***	0.00	8999
mjs47	1018	1059	1018	**1018***	0.00	1	1018	**1018***	0.00	303
mjs48	1074	1128	1074	**1074***	0.00	9	1074	**1074***	0.00	5812
mjs49	1202	1251	1202	**1202***	0.00	3	1202	**1202***	0.00	241
mjs50	849	949	849	**849***	0.00	10	849	**849***	0.00	8700
mjs51	919	1049	919	**919***	0.00	309	919	922	0.33	10,800
mjs52	880	948	880	**880***	0.00	6	880	**880***	0.00	675
mjs53	911	1018	911	**911***	0.00	60	906	1038	13.94	10,800
mjs54	832	945	832	**832***	0.00	78	818	833	0.12	10,800
mjs56	1257	1417	1257	**1257***	0.00	33	1257	-	-	10,800
mjs59	1273	1440	1273	**1273***	0.00	153	1273	-	-	10,800
mjs60	773	846	773	**773***	0.00	15	773	**773***	0.00	369
mjs61	1003	1015	1003	**1003***	0.00	22	1003	**1003***	0.00	3225
mjs62	931	979	931	**931***	0.00	3	931	**931***	0.00	163
mjs63	1056	1164	1056	**1056***	0.00	3	1056	**1056***	0.00	82
mjs64	823	828	823	**823***	0.00	3	822	**823***	0.00	270
mjs65	1277	1322	1277	**1277***	0.00	1	1277	**1277***	0.00	3436
mjs66	1461	1501	1461	**1461***	0.00	1	1461	**1461***	0.00	3288
mjs67	1370	1459	1370	**1370***	0.00	89	1370	**1370***	0.00	2812
mjs68	1398	1513	1398	**1398***	0.00	29	1398	**1398***	0.00	9013

Abbreviations: CP, constraint programming; DP, Dauzère-Pérès et al. (1998); LB, lower bound; MILP, mixed integer linear program.
Values in bold are best solutions found, and Values with * are optimal solutions.

there were consistency problems in the data of five instances, only 130 (two times 65) instances were considered.

Tables 3 and 4 present the numerical results. In both tables, the first column includes the name of the problem instance, while the second column provides the best-known lower bound (LB). The third column shows the results of Dauzère-Pérès et al. (1998), while the next two multicolumns provide the results of the proposed CP and MILP models, respectively. Each multicolumn includes LB, C_{max}, the percentage of the optimality gap from the LB, and the total elapsed time, respectively.

Overall, the CP model gives the best results compared to both the MILP model and the metaheuristic approach of Dauzère-Pérès et al. (1998), although sometimes at the expense of significant computational times. More specifically, regarding the instances with linear precedence constraints, the CP model has determined 49 new best solutions and 38 optimal solutions with an average optimality gap of 31.12%. On the other hand, the MILP model improves five solutions, while solving seven instances to optimality with an average gap of 33.51%. Note that the MILP cannot produce any feasible solution in 28 instances of this group. The same behavior is observed in the results regarding instances with arbitrary precedence constraints. In this case, the CP model gives 65 new best solutions and solves 56 instances to optimality with an average optimality gap of 1.02%. The MILP model produces 41 new best solutions and solves 41 instances to optimality with an average optimality gap of 3.18%. Note that, in this benchmark set, the MILP model cannot find a feasible solution for only two instances.

Note that, in this experiment, the addition of arbitrary precedence constraints induces a significant reduction of the average optimality gaps of both the CP and MILP models. This may be related to the fact that, when arbitrary precedence graphs are included, less operation sequences compete in parallel over the available resources at the same time, which typically leads to a reduction of the complexity of the problem.

6 | CONCLUSIONS

The multiresource FJSP with arbitrary precedence graphs, also called nonlinear routes, is considered in this paper. The theorems that were introduced in Dauzère-Pérès et al. (1998) and more recently in Kasapidis et al. (2021) are compared, and the extension to multiple resources is studied. In particular, a MILP model and a CP model are proposed, and computational results are discussed. They show that the CP model is more effective, although time-consuming for some instances and that Theorem 1 proposed in this paper is more effective than Theorem 5 of Dauzère-Pérès et al. (1998).

In terms of future research, it is worth studying the policies discussed in Dauzère-Pérès and Pavageau (2003), where all resources assigned to an operation may not be simultaneously occupied; instead, an operation may not start or end simultaneously on all its assigned resources.

ACKNOWLEDGMENTS

We would like to thank the editors of the journal for offering us this opportunity to show the link and future research prospects between Dauzère-Pérès et al. (1998) and Kasapidis et al. (2021).

ORCID

Gregory A. Kasapidis https://orcid.org/0000-0002-5272-9261
Stéphane Dauzère-Pérès https://orcid.org/0000-0002-3566-3248
Panagiotis P. Repoussis https://orcid.org/0000-0001-7717-1981

REFERENCES

Birgin, E. G., Feofiloff, P., Fernandes, C. G., De Melo, E. L., Oshiro, M. T., & Ronconi, D. P. (2014). A MILP model for an extended version of the flexible job shop problem. *Optimization Letters*, *8*, 1417–1431.

Birgin, E. G., Ferreira, J. E., & Ronconi, D. P. (2015). List scheduling and beam search methods for the flexible job shop scheduling problem with sequencing flexibility. *European Journal of Operational Research*, *247*, 421–440.

Brandimarte, P. (1993). Routing and scheduling in a flexible job shop by tabu search. *Annals of Operations Research*, *41*, 157–183.

Brucker, P., & Neyer, J. (1998). Tabu-search for the multi-mode job-shop problem. *Operations-Research-Spektrum*, *20*, 21–28.

Dauzère-Pérès, S., & Paulli, J. (1997). An integrated approach for modeling and solving the general multiprocessor job-shop scheduling problem using tabu search. *Annals of Operations Research*, *70*, 281–306.

Dauzère-Pérès, S., & Pavageau, C. (2003). Extensions of an integrated approach for multi-resource shop scheduling. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *33*, 207–213.

Dauzère-Pérès, S., Roux, W., & Lasserre, J. B. (1998). Multi-resource shop scheduling with resource flexibility. *European Journal of Operational Research*, *107*, 289–305.

Dell'Amico, M., & Trubian, M. (1993). Applying tabu search to the job-shop scheduling problem. *Annals of Operations Research*, *41*, 231–252.

González, M. A., Vela, C. R., & Varela, R. (2015). Scatter search with path relinking for the flexible job shop scheduling problem. *European Journal of Operational Research*, *245*, 35–45.

Ivens, P., & Lambrecht, M. (1996). Extending the shifting bottleneck procedure to real-life applications. *European Journal of Operational Research*, *90*, 252–268.

Kasapidis, G. A., Paraskevopoulos, D. C., Repoussis, P. P., & Tarantilis, C. D. (2021). Flexible job shop scheduling problems with arbitrary precedence graphs. *Production and Operations Management*, *30*, 4044–4068.

Lunardi, W. T., Birgin, E. G., Laborie, P., Ronconi, D. P., & Voos, H. (2020). Mixed integer linear programming and constraint programming models for the online printing shop scheduling problem. *Computers and Operations Research*, *123*, 105020.

Schutten, J. M. (1998). Practical job shop scheduling. *Annals of Operations Research*, *83*, 161–177.

How to cite this article: Kasapidis, G. A., Dauzère-Pérès, S., Paraskevopoulos, D. C., Repoussis, P. P., & Tarantilis, C. D. (2023). On the multiresource flexible job-shop scheduling problem with arbitrary precedence graphs. *Production and Operations Management*, *32,* 2322–2330. https://doi.org/10.1111/poms.13977