A constraint programming model for the flexible job shop scheduling problem with sequencing flexibility and position based learning effects

Kennedy A. G. Araújo
Departamento de Matemática Aplicada, IME-USP, Universidade de São Paulo
Rua do Matão, 1010, Cidade Universitária, 05508-090 - São Paulo - Brasil
kennedy94@ime.usp.br

Ernesto G. Birgin
Departamento de Ciência da Computação, IME-USP, Universidade de São Paulo
Rua do Matão, 1010, Cidade Universitária, 05508-090 - São Paulo - Brasil
egbirgin@ime.usp.br

Débora P. Ronconi
Departamento de Engenharia de Produção, EPUSP, Universidade de São Paulo
Av. Prof. Almeida Prado, 128, Cidade Universitária, 05508-900, São Paulo SP, Brasil
dronconi@usp.br

ABSTRACT

This paper addresses the flexible job shop scheduling problem with sequencing flexibility (FJS-SF) and learning effects with the objective of makespan minimization. Mixed integer linear programming (MILP) and constraint programming (CP) models are proposed to solve the problem for the position based learning effects. Computational experiments show that CP model is more efficient than MILP model to solve the problem.

KEYWORDS. Job shop scheduling problem. Constraint Programming. Mixed integer linear programming.

Mathematical Programming. Combinatorial Optimization

RESUMO

Este trabalho trata do problema do job shop flexível com flexibilidade de sequência e efeitos de aprendizado com o objetivo de minimizar o makespan. Modelos de programação linear inteira mista e programação por restrições são propostos para resolver o problema para efeitos de aprendizado baseado na posição. Experimentos computacionais mostram que o modelo de programação por restrições é mais eficiente que o modelo de programação linear inteira mista para resolver o problema.

PALAVRAS CHAVE. Job shop scheduling problem, Programação por Restrições, Programação linear inteira mista.

Programação Matemática. Otimização Combinatória

1. Introduction

The flexible job shop scheduling problem (FJS) is one of the most important and well-studied scheduling problems with several applications in real-world problems. In FJS we have a set of jobs and each job consists of a sequence of operations that have to be processed in a known order. Each operation must be processed without preemption exactly once by one machine from a given set of machines, not necessarily identical, and requires a processing time to be concluded. The objective is assigning and scheduling each operation in the available machines minimizing the maximal completion time, i.e the makespan.

The flexible job shop scheduling problem with sequencing flexibility (FJS-SF) is a variant of the FJS in which the precedence relations of the operations from the jobs are defined by a directed acyclic graph (DAG). A job is implicitly defined by a connected component of the DAG. The FJS-SF have several applications in industry, one of them is the printing industry, where each job may be represented by an order of printed copies of some object. Some of the orders correspond to a path-job or a Y-job. Usually, more than one machine may process the operations. Learning effect consideration may occur in some real-world problems, for example when the processing times arise from manual operations. Learning effect implies that the processing times are not constant and may decrease based on a learning function. The FJS-SF with learning effect is NP-hard, since it is also a generalization of the classical job shop scheduling problem [Pinedo, 2012].

Several authors studied scheduling problems with learning and deteriorating effects. Wang [2007] studied the single machine scheduling problem with learning and deterioration effects and analyzed several objective function choices. They prove that for some objective functions and special cases the problem can be solved in polynomial time by specific dispatching rules. Biskup [2008] reviewed works concerning scheduling problems with learning effects along with their importance in practical and theoretical studies. The authors discussed several learning functions and types of learning in scheduling literature, divided by position and sum of processing time based. The authors also commented about solution methods and their respective time complexities. Toksarı e Güner [2009] studied the parallel machine scheduling problem considering position based learning and deterioration effects simultaneously, aiming the weighted sum of earliness and tardiness minimization with common due date. The authors proposed integer programming models for the problem with linear and nonlinear deterioration, and introduced a modification of the Baker's algorithm [Baker e Trietsch, 2013] to solve the problem. The authors compared the computational results with respect to a proposed lower bound based on the Lagrangian relaxation of the problem. Yin e Xu [2011] studied single machine scheduling problems with deteriorating and learning effects aiming at different objective functions, such as makespan minimization, total weighted completion time minimization. The authors presented a generalized model for the problem and showed that, for some objective functions, the problem can be polynomially solved by dispatching rules, such as LPT and SPT. Ruiz-Torres et al. [2013] introduced a novel unrelated parallel machine scheduling problem with sequence dependent deteriorating effect aiming the makespan minimization. The authors prove that with single machine environment the problem is solved in polynomial time. List scheduling heuristics and a simulated annealing meta-heuristic are proposed to solve the case with 2 or more machines. Tayebi Araghi et al. [2014] studied the flexible job shop scheduling problem with sequence-dependent setup, position-based learning effect on setup time and deterioration effect on processing times with the objective of makespan minimization. The authors proposed a hybrid metaheuristic combining genetic algorithm and variable neighborhood search. Azzouz et al. [2018] revised scheduling problems with processing times with learning and deteriorating effects. The authors proposed a new classification scheme for the problems and a new cartography. The problem learning functions can be classified into position based, sum-of-processing time-based, truncated, exponential, position-based and sum-of-processing time-based simultaneously, and general learning effects. Future research is suggest, for example the combination of learning effects classes, multi-criteria objective functions, consideration of dynamic events impact such as breakdown machines or dynamic arrival of jobs. Computational complexity of general problems remains open. Wang et al. [2019] treated two variants of permutation flow shop problems with truncated exponential sum of logarithm processing times based and position-based learning effects with makespan and total weighted completion time minimization, respectively. The author presented heuristics based on dispatching rules and an exact branch-and-bound algorithm to solve the problem as well as show lower bounds for both problems. Liu e Jiang [2020] studied the single machine scheduling problem under the consideration with job-dependent position based learning effects and resource allocation simultaneously with common due-date and slack due-date assignment. The authors proved that two special cases of the problem can be solved in polynomial time reducing them to linear assignment problems. A heuristic based on solving linear assignment problems and a branch-and-bound algorithm were proposed to solve the general problem. Peng et al. [2021] address a manufacturing industry flexible job shop scheduling problem considering learning effects with the objectives of time energy consumption and noise minimization. The author propose a MINLP model. A hybrid discrete multi-objective imperial competition algorithm is proposed to solve the problem combining with simulated annealing algorithm to improve the neighborhood search. In the problem sequencing flexibility is not considered.

There are few researchers which studied the flexible job shop scheduling problem with consideration of position based learning effects on processing time. To the best of our knowledge, the consideration of learning effect on processing time and sequence flexibility simultaneously on the flexible job shop scheduling problem has not been previously studied.

The remainder of this work consists of the following sections. In Section 2 the FJS-LE is formally defined. In Sections 3 and 4 a mixed integer linear programming model and a constraint programming model are proposed, respectively. In Section 5 computational experiments are discussed. In Section 6 final conclusions are commented.

2. The flexible job shop scheduling problem with sequencing flexibility and learning effect

In the FJS-SF we have a set a operations $\mathcal{O}$. Each operation i may be processed in a machine k from a set of machines $\mathcal{F}_i$. The operations have their precedence relations represented by a directed acyclic graph (DAG). A job is implicitly defined by a connected component of the DAG. The objective of the FJS-SF is assigning and scheduling every operation in a machine without preemption whilst minimizing the makespan. The FJS is NP-hard, since it is a generalization of the job shop scheduling problem which is proved to be NP-hard Pinedo [2012].

An example of an instance of the FJS-SF is shown as follows. We consider the following instance has two jobs with 12 operations in total (job 1 with operations 1-6, and job 2 with operations 7-12), and 3 machines. The precedence constraints are represented in the directed acyclic graph (DAG) in Figure 1. Normal processing times and machine availability for each operation are detailed in Table 1.

A solution for the problem, associates for every operation $i \in \mathcal{O}$ an unique machine $k \in \mathcal{F}_i$ on which i is processed without preemption and starting time s_i. In the FJS-SF with learning effect on the processing time, the actual processing time may be faster and is calculated by a learning function that can take several parameters such as learning rate, position on which the operation is processed in a machine, or sum of processing time that a machine took processing operations before starting the processing of the current operation. An example of position-based learning function is

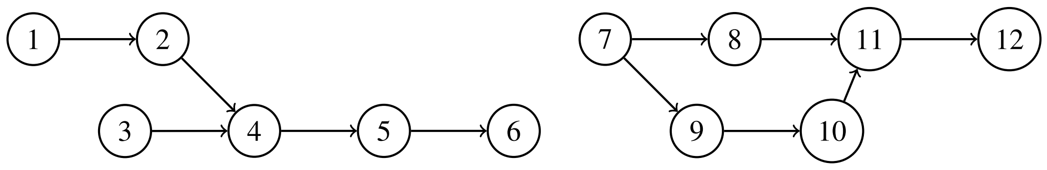

Figure 1: Representation of job precedence constraints in a directed acyclic graph.

Machines	Operations											
–	1	2	3	4	5	6	7	8	9	10	11	12
1	10	20	10	–	30	20	–	40	–	10	20	–
2	20	15	–	30	40	–	10	10	40	20	10	–
3	15	5	20	–	10	30	20	–	20	10	–	15

Table 1: Normal processing times for the 12 operations for each of the three machines. If a cell is filled with "–", it means that the operation cannot be processed in the machine.

$f(\alpha, p_{ik}) = r^{\alpha} p_{ik}$, in which $\alpha \leq 0$ stands for the learning rate, the more negative, the faster the actual processing time is, and r stands for the position in which the operation is processed. Note that, the processing times may be calculated previously.

3. Mixed integer linear model for the FJS-SF with position-based learning effect

In this section, a MILP model for the FJS-SF with position-based learning effect is proposed. Position-based decision variables is the basis for modeling position-based learning effect problems since is more natural to express constraints on the processing time change. The MILP model proposed as follows is based on Birgin et al. [2014]'s model and on the model with position based variables for scheduling first introduced by Wagner [1959]. Wilson [1989] also used position based variables for flowshop scheduling problems. The following notation is presented.

Sets:

$\mathcal{O}$: operations,

$\mathcal{F}$: machines,

$\mathcal{O}_k$: operations that can be processed by machine k,

$\mathcal{F}_i$: machines that can process operation i,

A: directed arcs that represent job precedence constraints (a directed acyclic graph),

Parameters:

p_{ik}: normal processing time of operation i in machine k,

$f(p_{ik}, r)$: position based learning function,

Decision variables:

x_{ikr}: 1 if operation i is processed in the r-th position by machine k; 0, otherwise,

s_i, starting time of operation i,

h_{kr}, starting time of processing the operation associated to the r-th position of machine k,

p'_i: actual processing time of operation i,

We present the MILP model for general linear position-based learning functions in Equations (1)-(14).

$$\text{Minimize} \quad C_{\max} \tag{1}$$

subject to

$$\sum_{k\in\mathcal{F}_i}\sum_{r=1}^{|\mathcal{O}_k|} x_{ikr} = 1, \qquad i \in \mathcal{O}, \tag{2}$$

$$\sum_{i\in\mathcal{O}_k} x_{ikr} \le 1, \qquad k \in \mathcal{F}, r = 1,\dots,|\mathcal{O}_k|, \tag{3}$$

$$\sum_{i\in\mathcal{O}_k} x_{i,k,r+1} \le \sum_{i\in\mathcal{O}_k} x_{ikr}, \qquad k \in \mathcal{F}, r = 1,\dots,|\mathcal{O}_k|-1, \tag{4}$$

$$p'_i = \sum_{k\in\mathcal{F}_i}\sum_{r=1}^{|\mathcal{O}_k|} f(p_{ik}, r)\cdot x_{ikr}, \qquad i \in \mathcal{O} \tag{5}$$

$$h_{kr} + \sum_{i\in\mathcal{O}_k} f(p_{ik}, r)\cdot x_{ikr} \le h_{k,r+1}, \qquad k \in \mathcal{F}, r = 1,\dots,|\mathcal{O}_k|-1, \tag{6}$$

$$h_{kr} + \sum_{i\in\mathcal{O}_k} f(p_{ik}, r)\cdot x_{ikr} \le C_{max}, \qquad k \in \mathcal{F}, r == 1,\dots,|\mathcal{O}_k|, \tag{7}$$

$$s_i + p'_i \le s_j, \qquad \forall (i,j) \in A \tag{8}$$

$$s_i + p'_i - \left(2 - x_{ikr} - \sum_{r'=r+1}^{|\mathcal{O}_k|} x_{jkr'}\right)\cdot M \le s_j, \qquad \forall i,j \in \{\mathcal{O} | i \ne j\}, \forall k \in \mathcal{F}_i \cap \mathcal{F}_j,$$
$$r = 1,\dots,|\mathcal{O}_k|-1, \tag{9}$$

$$h_{kr} - M\cdot(1 - x_{ikr}) \le s_i, \qquad i \in \mathcal{O}, k \in \mathcal{F}_i, r = 1,\dots,|\mathcal{O}_k|, \tag{10}$$

$$s_i - M\cdot(1 - x_{ikr}) \le h_{kr}, \qquad i \in \mathcal{O}, k \in \mathcal{F}_i, r = 1,\dots,|\mathcal{O}_k|, \tag{11}$$

$$s_i \ge 0, \qquad i \in \mathcal{O}, \tag{12}$$

$$h_{kr} \ge 0, \qquad k \in \mathcal{F}_i, r = 1,\dots,|\mathcal{O}_k| \tag{13}$$

$$x_{ikr} \in \{0,1\} \qquad i \in \mathcal{O}, k \in \mathcal{F}_i, r = 1,\dots,|\mathcal{O}_k|. \tag{14}$$

Objective function (1) stands for the makespan. Constraints (2) define that every operation must be processed by one machine and takes only one position. Constraints (3) impose that a position of one machine may only be associated to at most one operation. Constraints (4) force that idle positions, i.e., positions that are not associated to any operation, are at the end of the usage of the machines. Constraints (5) define the actual processing time of operation i in order to simplify the model. Constraints (6) force that the operations do not overlap in the machines. Constraints (7) set that the makespan is greater or equal to the completion time of every operation, combining such constraints with the function (1) minimization, the makespan is set as the maximum completion time among the operations. Constraints (8) impose that the precedence constraints among operations in the DAG is respected. Constraints (9) state that, if both operations i and j were assigned to the same machine k and operation i precedes operation j, i and j do not overlap. Constraints (10) and (11) associate starting times of operations to its corresponding position and machine, i.e., if $x_{ikr} = 1$ then $s_i = h_{kr}$. Constraints (12) and (14) refer to the decision variables domain. M is a sufficiently big number and may assume value $\sum_{i\in\mathcal{O}}\sum_{k\in\mathcal{F}} p_{ik}$. Function $f(p_{ik}, r)$ is a general function f tha calculates the actual processing time of operation i in machine k in the r-th position, such function may have additional parameters such as learning rate $\alpha_i \leq 0$ or truncation parameter $0 < \beta_i < 1$ for each operation i.

4. Constraint programming model for the FJS-SF with position-based learning effect

Constraint programming (CP) is well a powerful technique to solve scheduling problems in the literature. CP optimizer is an optimization commercial solver based on CP and has constraint and variable concepts that make modeling for scheduling problems easier. In this section, a CP model that can be used with CP Optimizer is presented. The CP optimizer syntax are defined as soon as they appear in the formulation. The CP model for the FJS-SF with position-based learning effect is presented in (15) - (21).

$$\text{Minimize} \quad \max_{i \in V} \text{endOf}(o_i) \tag{15}$$

$$\text{subject to}$$

$$\text{endBeforeStart}(o_i, o_j), \qquad (i,j) \in A, \tag{16}$$

$$\text{alternative}\left(o_i, [a_{ikr}]_{k \in \mathcal{F}_i, r=1,\ldots,|\mathcal{O}_k|}\right), \qquad i \in \mathcal{O}, \tag{17}$$

$$\text{noOverlap}\left([a_{ikr}]_{i \in \mathcal{O}_k, r=1,\ldots,|\mathcal{O}_k|}\right), \qquad k \in \mathcal{F}, \tag{18}$$

$$\text{or}([\text{presenceOf}(a_{ik,r+1})]_{i \in \mathcal{O}_k}) \implies \text{or}([\text{presenceOf}(a_{ikr}])_{i \in \mathcal{O}_k}), \qquad k \in \mathcal{F}_k, r = 1, \ldots, |\mathcal{O}_k| - 1, \tag{19}$$

$$\text{interval } o_i, \qquad i \in \mathcal{O}, \tag{20}$$

$$\text{interval } a_{ikr}, \text{ opt}, \text{ size } = f(p_{ik}, r), \qquad i \in \mathcal{O}, k \in \mathcal{F}_k, r = 1, \ldots, |\mathcal{O}_k|. \tag{21}$$

Interval decision variables of the problem are described in (20) and (21). In (20), an interval variable o_i for each operation i is defined. In (21), an *optional* interval variable a_{ikr} is being defined for each possible assignment of operation i to a machine $k \in \mathcal{F}_i$ and positions $r = 1, \ldots, |\mathcal{O}_k|$. *Optional* means that the interval variable may exist or not; and the remaining of the constraint says that, in case it exists, its size must be given by $f(p_{ik}, r)$. The objective function (15) is to minimize the makespan, given by the maximum end value of all the operations represented by the interval variables o_i. Precedence constraints between operations are posted as endBeforeStart constraints between interval variables in constraints (16). Constraints (17) state that each operation i must be allocated to exactly one machine $k \in \mathcal{F}_i$ in exactly one position r, that is, one and only one interval variable a_{ikr} must be present and the selected interval a_{ikr} will start and end at the same values as interval o_i. Constraints (18) state that, for a machine k, the intervals a_{ikr} representing the assignment of the operations to this machine do not overlap. Constraints (19) force that the absent position of machine k are the last ones, i.e., that operations are processed in the first positions of the machines without absent intervals among them.

The CP model can be strengthened by redundant constraints stating that, at any moment in time, there are never more than m machines being used simultaneously, and that operation associated to interval variable a_{ikr} ends before operation associated to interval variable $a_{ik,r+1}$ starts. Such constraints are given by Equations (22) and (23).

$$\sum_{i \in \mathcal{O}} pulse(o_i, 1) \leq m, \tag{22}$$

$$\text{endBeforeStart}(a_{ikr}, a_{ik,r+1}), \qquad i \in \mathcal{O}, k \in \mathcal{F}_i, r = 1, \ldots, |\mathcal{O}_k| - 1, \tag{23}$$

Adding redundant constraints reduces the domains of variables at each search node, which gets to a reduced search space.

5. Computational experiments

In this section, we evaluate the performance of the MILP and CP models proposed in the previous sections. We carried out the experiments with respect to a set of 50 benchmark instances. The experiments were carried out with an Intel Xeon X5690 3.47 GHz machine with 64 GB of RAM. The MILP and CP models were solved using CPLEX and CP Optimizer solvers, respectively, with IBM ILOG CPLEX Optimization Studio 22.1 software using default parameters with the concert library and C++ programming language. The code was compiled using the g++ 10.2.1 compiler. Benchmark instances and detailed experiments are available at `https://github.com/kennedy94/FJSModels`.

The set of benchmark instances was proposed by Birgin et al. [2014] for the FJS-SF with no learning effect. The first set, named YFJS, contains 20 instances with Y-shaped jobs, and the second set, named DAFJS, contains 30 instances with general type of jobs. The size of each instance is detailed in Table 2, where the triple (n, o, m) represents the number of jobs, number of operations per job, and number of machines, respectively.

In the computational tests, we considered the learning function $f(\alpha, p_{ik}) = r^{\alpha}p_{ik}$, where α and p_{ik} stand for the learning rate and normal processing time, respectively. The tests were carried out with time limit of 3,600 seconds. In Table 2, we present the final duality gap for each instance, varying the learning rate from $\alpha = -0.5$ to $\alpha = -0.1$. The duality gap is given by $100\times$ *(best feasible solution value - best lower bound value) / best feasible solution value* . In order to compare the two models with respect to the same lower bound, we decided to get the CP model best lower bound value as reference.

In Table 2, "–" states that no feasible solution was found within the time limit. CP model was able to find feasible solution for every instance with the every tested α value used. Although, proven optimal solutions were found by CP model only for instance YFJS03, for every α value. MILP model could not find feasible solutions in 116 test out of 250. Note that the solutions of instance YFJS03 are not equal,

With respect to the CP model, the average gaps for $\alpha \in \{-0.5, -0.1\}$ with instances of group YFJS are 63.00, 58.02, 51.13, 41.17, and 31.27, respectively, and with instances of group DAFJS are 81.72, 75.82, 68.76, 58.83, and 45.48, respectively. On the other hand, with respect to the MILP model, the average gaps for $\alpha \in \{-0.5, -0.1\}$ with instances of group YFJS are 76.11, 73.62 70.60, 64.55, and 61.95, respectively, and with instances of group DAFJS are 89.44, 87.89, 84.26, 81.23, and 76.36, respectively.

The CP model could find better solutions than MILP model in every test. Figures 2 and 3 shows that, for both models, the lower the learning rate, the harder the problem.

6. Conclusions

This work addressed the flexible job shop problem with sequencing flexibility (FJS-SF) with position-based learning effects. For the best of authors knowledge, sequencing flexibility and position-based learning effects were not studied in the flexible job shop environment simultaneously. We argued that FJS-SF with learning effects is NP-hard, proposed a constraint programming model and mixed integer linear model to solve the problem.

Computational tests with 50 benchmark instances, with a specific learning function with different parameter values, suggest that the CP model can find better solutions than MILP model. The MILP model cannot find feasible solutions for large-sized instances, while CP model found

instance	size (n, o, m)	CP model					MILP model				
		duality gap for $\alpha \in \{-0.5, \ldots, -0.1\}$					duality gap for $\alpha \in \{-0.5, \ldots, -0.1\}$				
		-0.5	-0.4	-0.3	-0.2	-0.1	-0.5	-0.4	-0.3	-0.2	-0.1
YFJS01	4, 10, 7	62.30	56.85	46.73	37.33	25.92	89.19	87.86	85.54	82.58	81.13
YFJS02	4, 10, 7	53.91	48.58	39.27	30.20	17.54	91.32	90.07	75.40	86.55	84.36
YFJS03	6, 4, 7	0.00	0.00	0.00	0.00	0.00	3.50	0.00	1.63	0.00	0.00
YFJS04	7, 4, 7	50.95	45.35	41.05	33.10	24.21	61.43	57.14	57.14	33.10	39.07
YFJS05	8, 4, 7	59.24	52.37	44.04	32.98	12.68	72.37	60.88	66.82	55.36	32.55
YFJS06	9, 4, 7	67.57	65.46	58.65	47.90	38.04	86.79	89.61	83.61	77.33	82.85
YFJS07	9, 4, 7	56.58	49.93	38.93	28.97	20.46	79.57	79.85	76.39	75.83	74.72
YFJS08	9, 4, 12	53.16	44.60	37.80	0.01	0.01	62.00	49.39	40.97	13.86	8.20
YFJS09	9, 4, 12	62.18	58.38	54.23	48.04	33.87	94.25	94.28	94.00	93.16	86.42
YFJS10	10, 4, 12	57.03	52.03	43.57	35.98	24.47	71.67	73.74	64.73	54.45	56.26
YFJS11	10, 5, 10	58.80	53.98	42.54	31.80	19.97	93.58	91.87	91.29	89.20	88.42
YFJS12	10, 5, 10	53.42	48.28	44.75	37.09	32.77	92.41	91.63	91.07	88.19	82.37
YFJS13	10, 5, 10	57.87	53.64	49.50	36.94	29.21	91.39	90.79	89.28	89.51	88.97
YFJS14	13, 17, 26	63.86	57.44	43.65	34.56	20.17	–	–	–	–	–
YFJS15	13, 17, 26	63.70	57.24	48.81	32.65	21.66	–	–	–	–	–
YFJS16	13, 17, 26	65.63	54.86	48.15	36.88	28.97	–	–	–	–	–
YFJS17	17, 17, 26	92.89	89.62	84.65	76.96	62.15	–	–	–	–	–
YFJS18	17, 17, 26	92.84	88.84	84.93	78.14	67.22	–	–	–	–	–
YFJS19	17, 17, 26	94.18	91.82	84.89	82.51	73.75	–	–	–	–	–
YFJS20	17, 17, 26	93.89	91.12	86.43	81.28	72.40	–	–	–	–	–
DAFJS01	4, 5-9, 5	64.42	55.36	47.09	37.23	22.42	71.81	77.41	66.49	47.64	49.39
DAFJS02	4, 5-7, 5	68.19	61.59	53.25	42.72	30.11	76.44	73.03	68.28	71.57	40.77
DAFJS03	4, 10-17, 10	68.01	60.23	49.83	35.81	20.11	91.42	88.93	84.33	–	–
DAFJS04	4, 9-14, 10	62.97	54.26	43.12	29.81	17.74	85.48	81.65	75.14	69.81	58.59
DAFJS05	6, 5-13, 5	70.99	63.55	58.80	46.54	36.98	88.51	86.62	83.17	79.89	75.57
DAFJS06	6, 5-13, 5	72.77	67.48	56.48	50.90	38.66	89.50	87.26	82.98	81.39	77.58
DAFJS07	6, 7-23, 10	78.46	71.40	61.71	49.53	28.87	–	–	–	–	–
DAFJS08	6, 6-23, 10	72.76	65.56	54.90	42.33	23.94	92.64	90.43	87.70	–	–
DAFJS09	8, 4-9, 5	76.55	70.29	64.07	55.72	45.39	91.16	90.03	88.07	85.39	82.21
DAFJS10	8, 4-11, 5	77.42	72.65	68.17	58.52	51.35	92.00	88.86	87.80	85.03	81.73
DAFJS11	8, 10-23, 10	86.86	74.12	69.13	55.02	34.33	–	–	–	–	–
DAFJS12	8, 9-22, 10	89.37	85.57	78.72	68.00	50.60	–	–	–	–	87.53
DAFJS13	10, 5-11, 5	84.32	80.52	76.08	70.57	63.97	94.09	92.89	91.17	89.10	86.62
DAFJS14	10, 4-10, 5	83.73	79.88	77.17	69.93	65.30	94.29	92.69	90.72	88.34	85.56
DAFJS15	10, 8-19, 10	91.31	88.63	80.38	71.29	57.63	–	–	–	–	–
DAFJS16	10, 6-20, 10	87.93	80.72	75.02	61.17	44.39	–	–	–	–	83.67
DAFJS17	12, 4-11, 5	87.97	86.19	81.76	77.81	73.44	95.49	94.34	92.97	91.33	89.44
DAFJS18	12, 5-9, 5	88.36	84.36	81.21	75.48	69.38	95.64	94.55	92.41	91.27	88.65
DAFJS19	8, 7-13, 7	75.04	67.00	57.92	46.43	27.16	93.73	91.84	88.45	86.47	81.78
DAFJS20	10, 6-17, 7	91.38	82.27	75.92	65.47	57.01	–	–	–	–	–
DAFJS21	12, 5-16, 7	93.05	87.38	81.65	77.90	63.06	–	–	–	–	–
DAFJS22	12, 5-17, 7	93.35	88.98	84.64	75.04	64.31	–	–	–	–	–
DAFJS23	8, 6-17, 9	77.13	70.72	60.85	49.93	36.27	–	–	–	88.80	–
DAFJS24	8, 6-25, 9	83.72	78.62	70.75	57.83	43.29	–	–	–	–	–
DAFJS25	10, 9-19, 9	91.86	87.88	82.30	73.30	57.55	–	–	–	–	–
DAFJS26	10, 8-17, 9	93.06	87.84	83.67	72.60	58.14	–	–	–	–	–
DAFJS27	12, 7-22, 9	92.22	90.65	84.52	77.15	67.12	–	–	–	–	–
DAFJS28	8, 8-15, 10	79.76	74.61	62.13	57.11	31.87	–	–	–	–	–
DAFJS29	8, 7-19, 10	81.20	74.42	69.24	50.61	35.17	–	–	–	–	–
DAFJS30	10, 8-19, 10	87.43	81.99	72.41	63.21	48.77	–	–	–	–	–

Table 2: Duality gaps (%)for the experiments with benchmark instances.

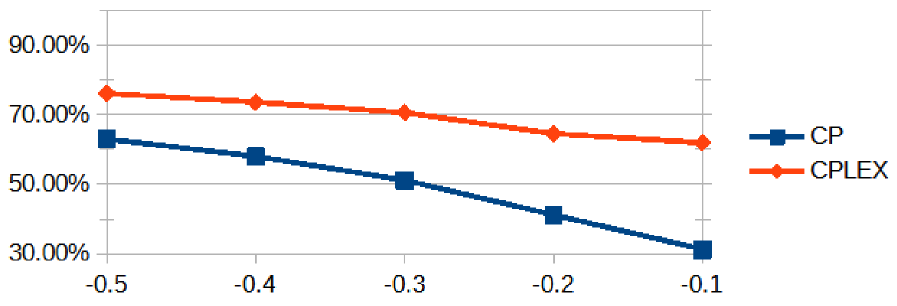

Figure 2: Average duality gaps for CP and MILP models with instance set YFJS. x and y-axis stand for α and duality gap, respectively.

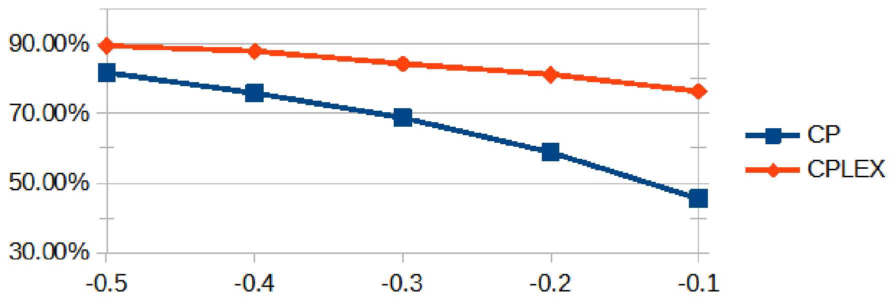

Figure 3: Average duality gaps for CP and MILP models with instance set DAFJS. x and y-axis stand for α and duality gap, respectively.

feasible solutions for every test considered. Although proven optimal solutions were found only for one instance.

For future work, several opportunities may be explored. Such as, multiobjective variants, different learning or deteriorating functions and/or parameterization. Heuristic or metaheuristics methods may be used as alternative methods to get to reasonable solutions in acceptable time.

7. Acknowledgments

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

References

Azzouz, A., Ennigrou, M., e Ben Said, L. (2018). Scheduling problems under learning effects: classification and cartography. *International Journal of Production Research*, 56(4):1642–1661.

Baker, K. R. e Trietsch, D. (2013). *Principles of sequencing and scheduling*. John Wiley & Sons.

Birgin, E. G., Feofiloff, P., Fernandes, C. G., De Melo, E. L., Oshiro, M. T., e Ronconi, D. P. (2014). A milp model for an extended version of the flexible job shop problem. *Optimization Letters*, 8 (4):1417–1431.

Biskup, D. (2008). A state-of-the-art review on scheduling with learning effects. *European Journal of Operational Research*, 188(2):315–329.

Liu, W. e Jiang, C. (2020). Due-date assignment scheduling involving job-dependent learning effects and convex resource allocation. *Engineering Optimization*, 52(1):74–89.

Peng, Z., Zhang, H., Tang, H., Feng, Y., e Yin, W. (2021). Research on flexible job-shop scheduling problem in green sustainable manufacturing based on learning effect. *Journal of Intelligent Manufacturing*, p. 1–22.

Pinedo, M. L. (2012). *Scheduling*, volume 29. Springer.

Ruiz-Torres, A. J., Paletta, G., e PéRez, E. (2013). Parallel machine scheduling to minimize the makespan with sequence dependent deteriorating effects. *Computers & Operations Research*, 40 (8):2051–2061.

Tayebi Araghi, M., Jolai, F., e Rabiee, M. (2014). Incorporating learning effect and deterioration for solving a sdst flexible job-shop scheduling problem with a hybrid meta-heuristic approach. *International Journal of Computer Integrated Manufacturing*, 27(8):733–746.

Toksarı, M. D. e Güner, E. (2009). Parallel machine earliness/tardiness scheduling problem under the effects of position based learning and linear/nonlinear deterioration. *Computers & Operations Research*, 36(8):2394–2417.

Wagner, H. M. (1959). An integer linear-programming model for machine scheduling. *Naval research logistics quarterly*, 6(2):131–140.

Wang, J.-B. (2007). Single-machine scheduling problems with the effects of learning and deterioration. *Omega*, 35(4):397–402.

Wang, J.-B., Liu, F., e Wang, J.-J. (2019). Research on m-machine flow shop scheduling with truncated learning effects. *International Transactions in Operational Research*, 26(3):1135–1151.

Wilson, J. (1989). Alternative formulations of a flow-shop scheduling problem. *Journal of the Operational Research Society*, 40(4):395–399.

Yin, Y. e Xu, D. (2011). Some single-machine scheduling problems with general effects of learning and deterioration. *Computers & Mathematics with Applications*, 61(1):100–108.