Proceeding Series of the Brazilian Society of Computational and Applied Mathematics, v. 11, n. 1, 2025.

Trabalho apresentado no XLIII CNMAC, Centro de Convenções do Armação Resort - Porto de Galinhas - PE, 2024

Proceeding Series of the Brazilian Society of Computational and Applied Mathematics

Exact Methods for the Parallel Machines Scheduling Problem with Identical Machines, Controllable Processing Times, and Resource Consumption

Levi R. Abreu[1], Bruno A. Prata[2]
UFC, Fortaleza, CE

Abstract. This paper aims to introduce a new variant of the identical parallel machines scheduling problem with controllable processing times and resource consumption. The performance measure is the minimization of makespan. We developed a mixed-integer linear programming (MILP) formulation and a constraint programming (CP) model. We conducted computational experiments using 2250 randomly generated test instances. The extensive computational experience of tested methods shows that the MILP model is promising for solving small-sized instances, and the CP model got the best average relative deviation and superior performance in large-sized instances.

Keywords.Production sequencing, sustainable manufacturing, green scheduling, combinatorial optimization.

1 Introduction

An important characteristic that has not been largely considered in the production sequencing problems is the controllability of processing times. In several real-world scenarios, the machines can present speed scalability [12]. In other words, they can operate with distinct modes (for example, slow, standard, and fast modes). Different speeds can lead to a reduction of the makespan or total tardiness. Nonetheless, a fast speed consumes numerous resources (usually energy or fuel), possibly with a higher emission of pollutants [5].

The following contributions address the identical parallel machines scheduling problem with controllable processing times. Fang and Lin [4] addressed the unrelated parallel machine scheduling problem aiming to minimize a weighted objective function incorporating total tardiness and power consumption. Their approach considers controllable processing times, due dates, and unitary penalties per tardiness. They present a MILP model alongside a constructive heuristic and a particle swarm optimization (PSO) metaheuristic. Kayvanfar et al. [6] tackled unrelated parallel machines with controllable processing times and sequence-dependent setup times. Their objective function integrates a weighted combination of makespan and earliness/tardiness penalties. They propose a MILP model, a Genetic Algorithm, and an Imperialist Competitive Algorithm (ICA) as solution approaches.

We consider an environment with identical parallel machines featuring controllable processing times and resource consumption. Each available machine can operate in three operational modes (fast, normal, and slow), each requiring varying levels of resources. For instance, the fast mode entails shorter processing times but higher resource consumption. Additionally, we take into account a non-renewable resource, as considered by Prata et al. [11].

[1]levi.abreu@ufc.br
[2]baprata@ufc.br

The main contributions of this paper are the development and analysis of MILP and CP models for the variant proposed. Subsequently, we conducted extensive experimentation to evaluate the developed formulations in solving a testbed of randomly generated test instances. The remaining sections of this paper are described below. Section 2 addresses the problem statement and some proprieties. Section 3 addresses the proposed formulations, while Section 4 presents the computational experiments. Finally, in Section 5, we discuss the main findings and suggestions of potential research avenues.

2 Problem statement

The proposed problems consist of the following notations and parameters: let j be an index for jobs {1,2,...,n}, l an index for speed factors {1,2,...,s}, k an index for job positions in each machine {1, 2, ...,n}, and i an index for machines {1,2,...,m}. For each job j and speed factor l, there is a matrix p_{lj} with processing time of job j with speed factor l and a matrix r_{lj} with the amount of consumed resource to process job j on machine i with speed factor l. Q is the total amount of non-renewable resources available. The problem objective is minimizing the last job operation completion time (makespan). Table 1 illustrates an example instance with two machines and five jobs. The total resources available for use is 40 units ($Q = 40$).

Table 1: Data example for the proposed problem

p_{lj}	J_1	J_2	J_3	J_4	J_5
fast	2	3	1	4	5
normal	4	6	2	8	10
slow	8	12	4	16	20

r_{lj} ($Q = 40$)	J_1	J_2	J_3	J_4	J_5
fast	12	4	24	8	16
normal	6	2	12	4	8
slow	3	1	6	2	4

A possible solution to the problem presented in Table 1 can be illustrated as a sequence of job operations on each machine with their respective chosen speed. Figure 1 illustrates a Gantt chart with the optimal solution of the instance example, with the sequence of jobs on the machines and their respective speeds as follows: $M_1 : \{J_5 : fast, J_3 : slow\}$ and $M_2 : \{J_4 : fast, J_1 : normal, J_2 : fast\}$.

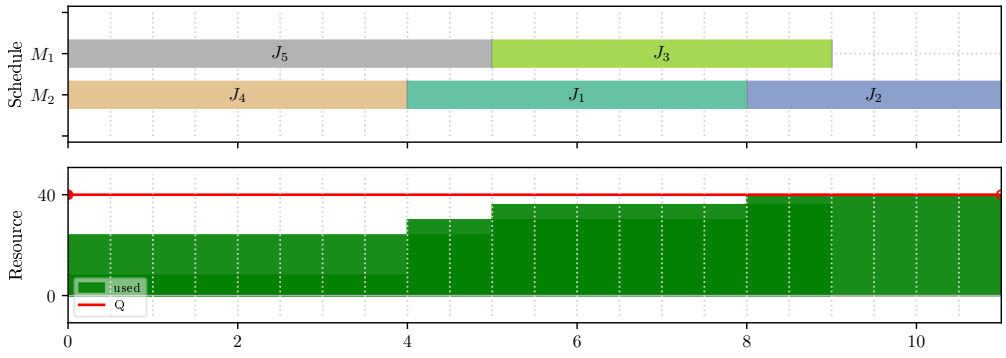


Figure 1: Gantt chart example. Fonte: dos autores

Analyzing Figure 1, we can see the optimum makespan of 11-time units (u.t) with a non-renewable resource consumption of 40 units, respecting the limit of $Q = 40$. The optimal solution mixes jobs processing at different speeds to fully use the amount of non-renewable resources available fully, optimizing the use of scarce resources and improving the makespan by processing jobs

at a fast speed. Next, we illustrate a proposed lower bound for the problem used in the results section to measure the solution quality of the exact methods.

Proposition 2.1. *A lower bound for proposed problem is given by:*

$$LB = \max\left(\left\lceil \frac{1}{m}\sum_{j=1}^{n} p_{l^*j} \right\rceil, \max_{\forall j} p_{l^*j}, p_{l^*m} + p_{l^*m+1}\right) \tag{1}$$

Proof. The lower bound considers the resource consumption $r_{lj} = 0\ \forall l, j$ and assumes that all jobs are processed using the fastest speed $p_{l^*j}\ \forall j$, where l^* is the index of the fastest speed. In addition, the lower bound calculation assumes the processing time vector p_{l^*j} sorted in non-increasing order without loss of generality. With these assumptions, the problem is reduced to the classic case of parallel machines and makespan minimization, with the lower bound proposed by Dell'Amico and Martello [3]. □

From the lower bound proposition, we can estimate the relative percentage deviation indicator of the MILP and CP models solutions.

3 Developed formulations

Next, we present the notation, objective function, and constraints used in the MILP and CP models.

3.1 Proposed MILP model

We define the following decision variables for the MILP model: C_{ik} is the completion time of the job in position k in machine i; C_{max} is the makespan, and Y_{ijkl} is equal to 1 if job j is processed in position k on machine i with speed factor l, 0 otherwise. The MILP model is an extension of the formulation proposed by Prata et al. [11], considering the same indexes and parameters as the problem statement section. The proposed MILP model is as follows.

$$\text{minimize} \quad C_{max} \tag{2}$$

$$\text{subject to} \quad \sum_{i=1}^{m}\sum_{k=1}^{n}\sum_{l=1}^{s} Y_{ijkl} = 1, \qquad \forall j \tag{3}$$

$$\sum_{j=1}^{n}\sum_{l=1}^{s} Y_{ijkl} \leq 1, \qquad \forall i,k \tag{4}$$

$$C_{i0} = \sum_{j=1}^{n}\sum_{l=1}^{s} p_{lj} Y_{ij0l}, \qquad \forall i \tag{5}$$

$$C_{ik} \geq C_{ik-1} + \sum_{j=1}^{n}\sum_{l=1}^{s} p_{lj} Y_{ijkl}, \qquad \forall i, k > 1 \tag{6}$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{s} r_{lj} Y_{ijkl} \leq Q, \tag{7}$$

$$C_{max} \geq C_{ik}, \qquad \forall i,k \tag{8}$$

$$C_{max}, C_{ik} \geq 0, \qquad \forall i,k \tag{9}$$

$$Y_{ijkl} \in \{0,1\}, \qquad \forall i,j,k,l \tag{10}$$

The objective function (2) is the makespan minimization. Constraints (3) force that a given job is processed in a single position of the sequence and a unique machine. Constraints (4) enforce

that each job can be processed at most once on each machine and position with just one speed. Constraints (5) determine the completion time for the job scheduled in the first position of the sequence. Constraints (6) determine the completion time for the jobs processed in the subsequent positions of the sequence. Constraint (7) enforces that the used resources are less than or equal to the maximum available resources. Constraints (8) determine the makespan. Finally, constraint sets (9) and (10) determine the domain of the decision variables.

3.2 Proposed CP model

Next, we illustrate the proposed CP model for the problem. This CP model uses intervals to represent the operations of jobs in machines and sequence variables to represent sequences of jobs processed in each machine [2, 7, 10]. The CP model uses the same indexes and similar parameters as the MILP model.

As interval variables, x_{ilj} is a unique optional interval variable for processing the operation of job j in machine i with speed l, and y_j is an interval variable to represents the operation duration of job j. As a sequence variable, Γ_i is a sequence variable with an order of x_{ilj} intervals variables in each machine i. Each interval variable is the sequence jobs j processed in machine i. The proposed CP model is as follows.

$$\text{minimize} \quad \max_{j\in\{1,2,\ldots,n\}} \texttt{endOf}\left(y_j\right) \tag{11}$$

$$\text{subject to} \quad \texttt{noOverlap}\left(\Gamma_i\right), \qquad \forall i \tag{12}$$

$$\texttt{Alternative}\left(y_j, \left[x_{ilj}\right]_{i\in\{1,2,\ldots,m\}:l\in\{1,2\ldots,s\}}\right), \qquad \forall j \tag{13}$$

$$\sum_{i=1}^{m}\sum_{l=1}^{s}\sum_{j=1}^{n} \texttt{stepAtStart}\left(x_{ilj}, r_{lj}\right) \leq Q, \tag{14}$$

$$\text{interval } x_{ilj}, \quad \text{opt}, \quad \text{size} = p_{lj}, \qquad \forall i,l,j \tag{15}$$

$$\text{interval } y_j, \qquad \forall j \tag{16}$$

$$\text{sequence } \Gamma_i, \quad \text{on } \left[x_{ilj}\right]_{l\in\{1,2,\ldots,s\}:j\in\{1,2,\ldots,n\}}, \qquad \forall i \tag{17}$$

The objective function (11) is the makespan minimization. Constraints (12) state that a single job at a time with just one speed can be produced on machine i. Constraints (13) impose that a given job j just be processed by one machine and one speed, the interval variable y_j gets the duration of selected x_{ilj} operation of job j between all m machines and l factors available. Constraint (14) enforces that the used resources are less than or equal to the maximum available resources. Finally, constraint sets (15), (16), and (17) define the scope of decision variables. The variables x_{ilj} and y_j are interval decision variables with a duration, and Γ_i is a variable that stores the sequence of operations on machine i.

4 Computational results

We proposed a new instance set for the proposed problem. The sets of instances are divided in $j \in \{100, 150, 250, 500, 1000\}$ for nine sets of each job size. The processing times and resource consumption matrix are uniformly distributed with U[1,100]. Each set of instances has 90 different test problems, totaling 450 base instances. Each base instance was tested in $m \in \{10, 20, 30, 40, 50\}$ identical machines, totaling 2250 test problems.

To compare the methods implemented, we analyze the results obtained in the computational experiments with Relative Percentage Deviation (RPD) concerning the lower bound of classical parallel machine problem for each instance. Equation (18) shows the RPD calculation of a given

DOI: 10.5540/03.2025.011.01.0393

instance, where sol_{method} is the makespan obtained by a method and LB is the lower bound provided by Equation (1).

$$RPD = \frac{sol_{method} - LB}{LB} \times 100 \tag{18}$$

Regarding computational times, we reported the computational time for the solver to run (Times), the time to reach the best solution (TimeBest), and the time to reach the first solution (TimeFirst). The MILP and CP models were run with a $(300 \times (n/2))/1000$ time limit in seconds, where n is the number of jobs. We implemented these time limits according to Prata et al. [9] for a similar single-machine problem with resource consumption and controllable processing times. The MILP models were executed in the GUROBI solver version 11.0.1, and the CP model was executed in the IBM ILOG CP Optimizer solver version 22.11, both implemented in Python 3.10. The computational experience was performed on a PC with Intel Core i5 CPU 3.00 GHz and 8Gb memory, with the Ubuntu operating system. Table 2 illustrates the Average RPD for all exact algorithms in each instance set. The results with the symbols *** indicate that the method used did not find any feasible solution in the computational time provided.

Table 2: Average RPD for all exact methods in each instance set.

	n									
	100		250		500		750		1000	
m	CP	MILP	CP	MILP	CP	MILP	CP	MILP	CP	MILP
10	29.69	**21.44**	33.96	**22.41**	**37.53**	***	**48.83**	***	**50.58**	***
20	37.14	**35.19**	**40.34**	1870.43	**50.52**	***	**50.58**	***	**82.68**	***
30	**43.86**	1201.14	**49.75**	***	**50.67**	***	**105.77**	***	**95.52**	***
40	**49.71**	3768.65	**50.26**	***	**135.97**	***	**118.33**	***	**113.88**	***
50	**39.82**	4583.36	**53.70**	***	**149.57**	***	**137.61**	***	***	***
Min	20.99	**18.12**	27.97	**19.23**	**31.04**	***	**32.06**	***	**50.45**	***
Average	**40.04**	1921.96	**45.60**	946.42	**84.85**	***	**92.22**	***	**85.66**	***
Max	**40.04**	1921.96	**45.60**	946.42	**84.85**	***	**92.22**	***	**85.66**	***
# Unsolved	**0**	**0**	**0**	270	**0**	450	**0**	450	**90**	450

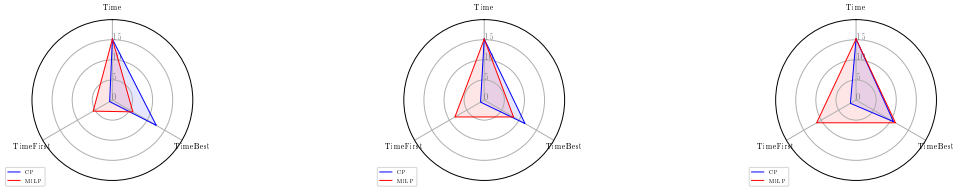
The MILP model was not able to solve the 1620 instances with the largest size (over 30 machines and 200 jobs and over 500 jobs) with the default settings; the CP model was not able to solve the 90 instances with the largest size (over 50 machines and 1000 jobs) with the default settings. Analyzing Table 2, it can be seen that the algorithm with the best Average RPD is the CP model, which got the best results, outperforming the MILP model in instance-sizes over 20 machines and 250 jobs and 30 machines in any job. The MILP obtained the best results in instances sized up to 20 jobs and 100 jobs and instances sized with 10 machines and 250 jobs. The CP model obtained the best average RPD between all instances and most instance sets. To check the results of the exact methods in detail, Figure 2 illustrates the heatmap of average RPD in each instance size $m \times n$ for each method.

Analyzing Figure 2a, the CP model found solutions for most instance sets, producing solutions farther from the lower bound from 40 machines and 500 jobs for larger instances. Analyzing Figure 2b, the MILP model failed to find solutions for most instances tested. Furthermore, for machine numbers larger than 40, the MILP model obtained RPD results greater than 100%, far from the instances' lower bound. As the MILP and CP models got competitive results on the small-sized instances, Figure 3 illustrates a radar chart comparing the computational times Time, Tbest, and TFirst for the exact methods tested in some instances sets with $n = 100$.

DOI: 10.5540/03.2025.011.01.0393

(a) CP model (b) MILP model

Figure 2: Heatmap for avg. RPD for each tested method in each instance set $m \times n$. Fonte: dos autores



(a) $m = 20$ (b) $m = 30$ (c) $m = 40$

Figure 3: Radar chart for avg. computational times (s) for the instance set $n = 100$. Fonte: dos autores

Analyzing Figure 3, the CP model reaches the first solution faster than the MILP model in instances set $m \in \{20, 30, 40\}$. The MILP model reaches the best solution faster than the CP model in instance sets of Figures 3a and 3c, even with better average RPD results than the CP model, as seen in Table 2 and Figure 2b. The CP model found valid initial solutions faster than the MILP model, with a better TimeFirst indicator in each set of instances. As the complexity of the problem increases, the TimeBest indicator becomes closer to the Time indicator in the MILP model, as seen in Figure 3c, indicating that the time to reach the best solution found (TimeBest) is very close to the maximum time available to execute the exact method (Time). Therefore, the MILP model begins to lose efficiency in large-sized instances.

5 Final remarks

This paper presented a new variant for parallel machines considering controllable processing times and resource consumption. We proposed a new CP and MILP scheduling model. As a result, the new CP model performed superiorly compared to the proposed MILP model, but neither model solved large instances of over 50 machines and 1000 jobs. As extensions of this work, other parallel machine variants could be investigated, considering different performance measures, such as total tardiness or the number of tardy jobs. A new execution with a larger time limit could improve the quality of the proposed exact methods. Finally, developing hybrid algorithms, such as CP matheuristics [1] and developing valid inequalities [8], are also research opportunities to find feasible solutions in large-sized instances.

Acknowledgment

This work was supported by the Brazilian National Council for Scientific and Technological Development [Grants 309755/2021-2, 407151/2021-4, and 404264/2023-9] and the Brazilian Coordination for the Improvement of Higher Education Personnel [Finance Code 001].

References

[1] L. R. de Abreu, K. A. G. Araújo, B. A. de Prata, M. S. Nagano, and J. V. Moccellin. "A new variable neighbourhood search with a constraint programming search strategy for the open shop scheduling problem with operation repetitions". In: **Engineering Optimization** 54.9 (2022), pp. 1563–1582. DOI: 10.1080/0305215X.2021.1957101.

[2] L. R. Abreu, B. A. Prata, M. S. Nagano, and J. M. Framinan. "A constraint programming-based iterated greedy algorithm for the open shop with sequence-dependent processing times and makespan minimization". In: **Computers & Operations Research** 160 (2023), p. 106386. DOI: 10.1016/j.cor.2023.106386.

[3] M. Dell'Amico and S. Martello. "Optimal scheduling of tasks on identical parallel processors". In: **ORSA Journal on Computing** 7.2 (1995), pp. 191–200. DOI: 10.1287/ijoc.7.2.191.

[4] K. T. Fang and B. M. T. Lin. "Parallel-machine scheduling to minimize tardiness penalty and power cost". In: **Computers & Industrial Engineering** 64.1 (2013), pp. 224–234. DOI: 10.1016/j.cie.2012.10.002.

[5] V. Fernandez-Viagas and J. M. Framinan. "Controllable processing times in project and production management: Analysing the trade-off between processing times and the amount of resources". In: **Mathematical Problems in Engineering** 2015 (2015). DOI: 10.1155/2015/826318.

[6] V. Kayvanfar, G. H. M. Komaki, A. Aalaei, and M. Zandieh. "Minimizing total tardiness and earliness on unrelated parallel machines with controllable processing times". In: **Computers & Operations Research** 41 (2014), pp. 31–43. DOI: 10.1016/j.cor.2013.08.003.

[7] P. Laborie. "An update on the comparison of MIP, CP and hybrid approaches for mixed resource allocation and scheduling". In: **International conference on the integration of constraint programming, artificial intelligence, and operations research**. Springer. 2018, pp. 403–411. DOI: 10.1007/978-3-319-93031-2_29.

[8] E. Mokotoff. "An exact algorithm for the identical parallel machine scheduling problem". In: **European Journal of Operational Research** 152.3 (2004), pp. 758–769. DOI: 10.1016/S0377-2217(02)00726-9.

[9] B. A. Prata, L. R. de Abreu, and J. Y. F. Lima. "Heuristic methods for the single-machine scheduling problem with periodical resource constraints". In: **Top** 29.2 (2021), pp. 524–546. DOI: 10.1007/s11750-020-00574-x.

[10] B. A. Prata, L. R. Abreu, and M. S. Nagano. "Applications of constraint programming in production scheduling problems: A descriptive bibliometric analysis". In: **Results in Control and Optimization** 14 (2024), p. 100350. DOI: 10.1016/j.rico.2023.100350.

[11] B. A. Prata, V. Fernandez-Viagas, J. M. Framinan, and C. D. Rodrigues. "Matheuristics for the flowshop scheduling problem with controllable processing times and limited resource consumption to minimize total tardiness". In: **Computers & Operations Research** 145 (2022), p. 105880. DOI: 10.1016/j.cor.2022.105880.

[12] D. Shabtay and G. Steiner. "A survey of scheduling with controllable processing times". In: **Discrete Applied Mathematics** 155.13 (2007), pp. 1643–1666. DOI: 10.1016/j.dam.2007.02.003.

