Dual Resource Flexible Job Shop Scheduling Problems: the xBTF Algorithm*

Ricardo Magalhães[1][0009−0005−4340−4202], Filipe Santos[1][0009−0008−6083−3617], Susana Vieira[1][0000−0001−7961−1004], and João M. C. Sousa[1][0000−0002−8030−4746]

Instituto Superior Técnico, Universidade de Lisboa, Portugal

Abstract. In this paper, we unveil the xBTF, an innovative advancement to the Biggest Threat First (BTF) algorithm. In our previous research, the BTF achieved groundbreaking results for minimizing the makespan of the Dual Resource Constrained Job Shop Scheduling Problem (DRC-FJSSP). However, it faced limitations due to its inability to effectively balance worker and machine workloads. This was primarily because it constructed schedules sequentially, without considering the resource requirements of subsequent operations. The xBTF introduces a penalty when performing resource allocation, based on the expected worker and machine workload. Preliminary experimentation utilizing the MK1-10 benchmark dataset showcases xBTF's supremacy over its predecessor, particularly in scenarios with higher resource workloads. Moreover, the xBTF utterly outperforms a state-of-the-art metaheuristic, the KGFOA, with extremely small execution times, which makes it excellent also for rescheduling in dynamic scenarios.

Keywords: Scheduling · Job Shop · Dual Resource · Longest Processing Time · Minimum Feasible Time · Heuristic

1 Introduction

With the rise of Industry 4.0, more and more sensors are getting available at the shop floor levels and more data is being generated. Data driven approaches are becoming more popular and being increasingly adopted for optimized process planning, production planning and scheduling.

Production scheduling is crucial to manufacturing companies planning and management. A well-planned schedule may significantly boost productivity and resource use [1]. Manufacturing companies must be able to manage production

* The authors acknowledge Fundação para a Ciência e a Tecnologia (FCT) financial support via the projects LAETA Base Funding (DOI: 10.54499/UIDB/50022/2020) and LAETA Programatic Funding (DOI: 10.54499/UIDP/50022/2020). Filipe Santos' work was supported by the Ph.D. Scholarship 2022. 12077.BDANA from FCT. Ricardo Magalhães' work was funded by FCT, through a PhD scholarship under MIT Portugal program, with the reference MPP2030-FCT ID 22405888735. No potential competing interest was reported by the authors.

with a variety of product configurations as the paradigm shifts from mass production to mass customization [2]. This kind of problem is known as the job shop scheduling problem (JSSP). The JSSP is a NP-hard combinatorial optimization problem [3], of utmost relevance and impact in the fields of manufacturing and production [4], that seeks to discover the ideal order of jobs to be processed at distinct machines.

In most manufacturing scenarios, the route (resource allocation) that each job has to follow is not predefined, in which case it is a Flexible Job Shop Scheduling Problem (FJSSP). Additionally, some problems may be dual resource constrained (DRC). The DRC system is a production system that is restricted by both worker and machine capacity [5].

Overall, the DRC-FJSSP has been extensively addressed in the literature, in a large variety of industrial contexts. All the time, new algorithms and models arise that pretend to enhance schedule optimization and reduce the total amount of execution time. A mixed integer linear programming model (MILP) for quality control laboratory scheduling was presented in [6]. A Multiple-Trial/Best-Move Simulated Annealing algorithm for problems with scarce setup-operators was generated in [7]. A Workload control simulation assessment was performed in [8], considering several labor dispatching rules and environmental factors. A hybrid artificial bee colony algorithm was applied in [9] and a new effective local search method to improve the speed and exploitation ability of the algorithm was developed. A knowledge-guided fruit fly optimisation algorithm (KGFOA) with a new encoding scheme is proposed in [10], that balances global exploration and local exploitation with a combination of a knowledge-guided search and a smell-based search that consists of two types of permutation-based search operators. A Multi-Start Tabu Agents-based Model (MuSTAM) was introduced in [11], where TabuAgents cooperate and communicate between them in order to improve the search quality. Different variants of filter-and-fan (FF) based heuristic solution approaches that combine a local search procedure, used to obtain local optima, with a tree search procedure, that generates compound transitions in order to explore larger neighborhoods, were developed in [12].

Dynamical settings such as machine breakdowns have been studied. A robust fuzzy-stochastic programming model under machine breakdowns and uncertain processing times was crated in [13]. Discrete event simulation was used in [14] as a research method to assess the impact of machine failures on worker assignment.

Rescheduling is also considered as a viable option for problems in dynamic scenarios. An automobile collision repair shop environment was presented in [15], where re-scheduling is often needed to react to real-time events like due date changes, delay in arrival, changes in job processing time and rush jobs. A rescheduling framework was proposed in [16], which integrates a Machine Learning classification model for identifying rescheduling patterns and a hybrid metaheuristic approach for schedule optimization. A genetic algorithm and simulated annealing algorithm (GASA) coupled with a rescheduling decision making process for optimization of schedules in a dynamic scenario was presented in [17].

Other papers assess the use of Reinforcement Learning to develop an intelligent scheduling agent capable of solving the DRC-FJSSP. A framework for training a Reinforcement Learning agent to schedule diverse dual-resource constrained job shops was established in [18]. The use of an Encoder-Decoder recurrent neural network architecture with Attention Mechanism using Deep Q-Learning, that makes the Reinforcement Learning agent capable of solving problems of variable sizes, was introduced in [19].

Recently published works show there has been a lack of scientific interest in the development of new scheduling heuristics. A situation which reminds the two AI winters in the 1970s and 1980s. Scheduling heuristics seemed to have gone as far as they could and research nowadays is much more focused in the exploration of a solution space, be it through simulation, metaheuristics or even AI. But these algorithms had their own set of features they excelled at. Like having extremely small execution times, which is a tremendous advantage for dynamic scenarios which are becoming more and more important. If only they could achieve competitive performances with current state-of-the-art scheduling methods.

In this work, the xBTF is developed to optimize performances in scenarios when there are highly requested machines and/or workers. The xBTF algorithm introduces a penalty factor at the resource allocation procedure, which considers the expected workload of workers and machines. The goal of this penalty is to release some of the workload for workers and machines whose utilization is expected to be higher. The idea is to release the pressure on these expected bottlenecks.

As this algorithm is an heuristic, it does not take time exploring a search space and looking out for a good solution. It immediately constructs a good solution, taking advantage of the expert knowledge it was built upon. This allows this algorithm to achieve extremely small execution times, which makes it excellent also for recheduling in dynamic scenarios. Also, the xBTF is a deterministic method, which means that the quality of the solutions it creates is not subject to chance. These are substantial advantages comparing with the related work existent in the literature, if the xBTF can prove to achieve as good solutions as those state-of-the-art methods.

The remaining of this paper is organized as follows. In section II the numerical model for a DRC-FJSSP is introduced. Additionaly, the xBTF algorithm is introduced. Results are presented and discussed in section III. Section IV concludes the paper and presents future research steps.

2 Model and methods

2.1 Mathematical Model of a DRC-FJSSP

In a DRC-FJSSP there are a total of a operations that need to processed. They are divided in a set of n jobs $J = \{J_1, ..., J_i, ..., J_n\}$ to be processed at a set of m machines $M = \{M_1, ..., M_k, ..., M_m\}$ operated by a set of w workers $W =$

$\{W_1, ..., W_l, ..., W_w\}$. Each job has a predefined sequence of o_i operations $J_i = \{O_{i1}, ..., O_{ij}, ..., O_{io_i}\}$. Let p_{ij} be the processing time of the operation O_{ij}. Each operation c can only be processed at a subset of eligible worker-machine pairs. c is calculated as in equation 1.

$$c = j + \sum_{d=1}^{i-1} o_d \tag{1}$$

Eligible worker-machine pairs are made by a machine that can process a given operation and a worker that can operate that machine. Thus, there is an eligibility matrix E_{bc}, where the element (b, c) is a 1, if b is an eligible worker-machine pair for operation c, otherwise it is a 0. Where, b is calculated as in equation 2 and c is a number between 1 and a.

$$b = (k-1)w + l \tag{2}$$

Each machine can process only one operation at a time and there is no preemption, which means that since an operation has started it cannot be stopped until it is finished. All jobs, machines and workers are available at time 0. The goal is to minimise the maximum completion time, the makespan C, by assigning an eligible worker-machine pair to each operation, as well as arranging the processing order of operations on each machine.

Let s_{ij} be the starting time of $O_{i,j}$, and r_{kl} be the ready time of machine M_k operated by worker W_l, and N be a large enough number. Mathematically, the DRC-FJSSP with makespan minimisation can be formulated as follows:

$$\min\ C \tag{3}$$

Subject to:

$$C = \max_{i,j}(s_{ij} + p_{ij}),\ J_i \in J,\ j = 1, 2, ..., o_i - 1 \tag{4}$$

$$s_{i(j+1)} \geq s_{ij} + \sum_k \sum_l p_{ij}\eta_{ijkl},\ J_i \in J,\ j = 1, 2, ..., o_i - 1,\ \sum_{c=1}^{a} E_{bc} > 0 \tag{5}$$

$$s_{i'j'} + (1 - \zeta_{ijm-i'j'm})N \geq s_{ij} + \sum_l p_{ij}\eta_{ijkl},$$

$$J_i \in J,\ j = 1, 2, ..., o_i,\ \sum_{c=1}^{a} E_{bc} > 0 \tag{6}$$

$$r_{k'l} + (1 - \xi_{kl-k'l})N \geq r_{kl},\ W_l \in W,\ \sum_{c=1}^{a} E_{bc} > 0,\ \sum_{c=1}^{a} E_{b'c} > 0 \tag{7}$$

$$r_{kl} + (1 - \eta_{ijkl})N \leq s_{ij},\ J_i \in J,\ j = 1, 2, ..., o_i,\ \sum_{c=1}^{a} E_{bc} > 0 \tag{8}$$

$$\sum_{k}\sum_{l}\eta_{ijkl}=1,\ J_i\in J,\ j=1,2,...,o_i,\ \sum_{c=1}^{a}E_{bc}>0 \tag{9}$$

where

$$\eta_{ijkl}=\begin{cases}1, & \text{if } O_{i,j} \text{ is processed on } M_k \text{ operated by } W_l\\ 0, & \text{otherwise}\end{cases} \tag{10}$$

$$\zeta_{ijk-i'j'k}=\begin{cases}1, & \text{if } O_{i,j} \text{ is processed before } O_{i'j'} \text{ on } M_k\\ 0, & \text{otherwise}\end{cases} \tag{11}$$

$$\xi_{kl-k'l}=\begin{cases}1, & \text{if } M_k \text{ is operated before } M_{k'} \text{ by } W_l\\ 0, & \text{otherwise}\end{cases} \tag{12}$$

Equation 4 defines the makespan C. Equation 5 ensures that the precedence constraints are not violated. Equation 6 guarantees that a machine can process only one operation at a time. Equation 7 ensures that a worker can only operate one machine at a time. Equation 8 ensures that an operation cannot start unless the assigned resources are ready. Equation 9 guarantees that each operation is assigned to only one compatible machine operated by one eligible worker.

2.2 The Improved Biggest Threat First Algorithm

The new Improved Biggest Threat First algorithm, generates the blacklist B as in its predecessor [20], but it introduces a penalty at the resource allocation process, i.e., when generating the deployment vector D. The penalty is calculated based on the expected workload of each eligible worker and machine.

The new algorithm starts by estimating the expected worker utilization, v. This utilization is estimated as a linear combination of the operations processing times p with an allocation multiplication factor, x. x_{cl} is a fraction, whose numerator is the number of eligible resource pairs for operation c with worker l as the chosen operator, and whose denominator is the number of eligible resource pairs for operation c. The algorithm then calculates the penalized minimum feasible time h_l at which operation c could be allocated to worker l. The chosen worker z is the one with the lowest h. A similar procedure is taken to choose machine y for allocation. At each step, operation B_c is put into the schedule at time step $\min(f)$ allocated to resource pair D_c.

3 Results

The xBTF was tested for a widely used benchmark dataset, the MK1-10 [21]. A hundred test samples were generated for each of the MK1-10 benchmark dataset problem instances. The types of instances in the dataset are described in table 2, with information regarding the number of jobs, machines and workers, the number of operations per job, the maximum number of equivalent machines per

Table 1. Nomenclature

Sets	
e	Set of eligible worker-machine resource pairs
J	Set of jobs, J=$\{J_1, ..., J_n\}$
M	Set of machines, M=$\{M_1, ..., M_m\}$
W	Set of workers, W=$\{W_1, ..., W_w\}$
Variables	
f_d	Minimum feasible time for resource pair d
g	c-th operation in the Blacklist
h	Penalized minimum feasible time
q_i	Remaining sum of processing times of job i
r_{kl}	Ready time of machine M_k operated by worker l
s_{ij}	Starting time of operation O_{ij}
v	Expected worker utilization
y	Chosen machine
z	Chosen worker
B	Blacklist
C	Makespan
D	Deployment
L	Lower Bound
O_{ij}	j-th operation of job i
meq	maximum number of equivalent machines per operation
η	Operation allocation flag
σ_i	Number of processed operations from job i
ζ	Operation sequence flag
ξ	Machine sequence flag

operation and the processing times of each operation. The full test dataset is available at https://github.com/Ricardo-Mag/hundredMK1-10.

The KGFOA was chosen as a state-of-the-art metaheuristic for comparison, as it is the method highlighted in the related work section with the most number of citations and it proved to achieve an outstanding performance for DRC-FJSSP problems. It was implemented as described in [10] for 1000 generations.

The average performance of all algorithms was assessed for each group of instances in the dataset, as well as their respective standard deviations. These results are presented in table 3. The distance to bound represents the percentual distance of the makespan to the Lower Bound. The lower bound of a job shop problem is a theoretical threshold which is guarantied to be smaller or equal to the optimal makespan. It is the sum of all operations processing times divided by the total number of units of the bottleneck resource, in this case the number of workers.

First and foremost, the xBTF utterly outperforms the KGFOA for all instances and metrics. These outstanding results show that this algorithm can

Algorithm 1 Penalized generation of the Deployment and scheduling

Initialize empty schedule
for $c = 1, ..., a$ **do**
  $g = B_c$
  for $l = 1, ..., w$ **do**
    $v_l = \sum_{c=1}^{a} x_{cl} p_c$ ▷ $v_l = NaN$, if worker l is not eligible
  end for
  for $l = 1, ..., w$ **do**
    $f_l = \text{MFT}(g, l)$ ▷ Check worker l availability and job precedences
    $h_l = f_l + (\max(v) - v_l)$
  end for
  $z = \text{argmin}(h)$
  for $k = 1, ..., m$ **do**
    $v_k = \sum_{c=1}^{a} x_{ck} p_c$ ▷ $v_k = NaN$, if machine k is not eligible with worker z
  end for
  for $k = 1, ..., m$ **do**
    $f_k = \text{MFT}(g, k, z)$ ▷ Check resource (k,z) availability and job precedences
    $h_k = f_k + (\max(v) - v_k)$
  end for
  $y = \text{argmin}(h)$
  $D_c = (y - 1)w + z$
  Put B_c in schedule at $\min(f)$ in D_c
end for

Table 2. MK1 to MK10 instances.

	n	m	w	o_i	meq	p_{ij}
MK1	10	6	4	5-7	3	1-7
MK2	10	6	4	5-7	6	1-7
MK3	15	8	6	10	5	1-20
MK4	15	8	6	3-10	3	1-10
MK5	15	4	3	5-10	2	5-10
MK6	10	15	8	15	5	1-10
MK7	20	5	4	5	5	1-20
MK8	20	10	6	10-15	2	5-20
MK9	20	10	6	10-15	5	5-20
MK10	20	15	8	10-15	5	5-20

Table 3. Average KGFOA, BTF and xBTF test results and standard deviation.

		Makespan	Distance Bound (%)	Exec Time (s)
MK1	KGFOA	72.9 ± 7.3	21.1 ± 8.3	264.47 ± 19.05
	BTF	**67.9 ± 6.1**	**12.6 ± 4.7**	**0.20 ± 0.01**
	xBTF	**68.0 ± 5.8**	**12.9 ± 4.2**	0.21 ± 0.01
MK2	KGFOA	75.1 ± 7.4	23.4 ± 9.5	273.16 ± 16.72
	BTF	**64.0 ± 4.6**	**5.0 ± 2.8**	**0.21 ± 0.01**
	xBTF	**63.9 ± 4.3**	**4.9 ± 2.4**	0.22 ± 0.01
MK3	KGFOA	367.6 ± 23.9	40.0 ± 7.6	998.66 ± 20.65
	BTF	**289.1 ± 12.1**	10.1 ± 2.7	**0.33 ± 0.02**
	xBTF	**286.5 ± 11.4**	**9.1 ± 2.3**	0.38 ± 0.02
MK4	KGFOA	124.1 ± 15.4	38.0 ± 8.3	522.20 ± 67.22
	BTF	**109.9 ± 12.9**	22.3 ± 6.9	**0.25 ± 0.02**
	xBTF	**107.7 ± 12.0**	**19.8 ± 5.5**	0.27 ± 0.02
MK5	KGFOA	322.9 ± 22.3	14.0 ± 3.8	661.80 ± 57.21
	BTF	**308.2 ± 19.9**	**8.9 ± 3.7**	**0.27 ± 0.02**
	xBTF	**311.4 ± 20.3**	10.0 ± 3.4	0.29 ± 0.02
MK6	KGFOA	172.4 ± 13.6	66.9 ± 10.8	1125.95 ± 16.35
	BTF	**126.9 ± 6.4**	**22.9 ± 4.4**	**0.35 ± 0.02**
	xBTF	**127.3 ± 6.4**	**23.2 ± 4.1**	0.40 ± 0.02
MK7	KGFOA	313.5 ± 21.5	19.0 ± 5.7	536.50 ± 13.59
	BTF	**274.0 ± 15.8**	**4.0 ± 2.1**	**0.26 ± 0.01**
	xBTF	**273.0 ± 14.8**	**3.6 ± 1.5**	0.27 ± 0.02
MK8	KGFOA	662.3 ± 32.1	28.3 ± 4.2	2089.35 ± 114.43
	BTF	**586.5 ± 23.0**	**13.6 ± 3.3**	**0.47 ± 0.03**
	xBTF	**587.8 ± 25.1**	**13.9 ± 3.5**	0.52 ± 0.03
MK9	KGFOA	673.5 ± 34.5	29.0 ± 3.8	2146.64 ± 109.55
	BTF	**542.3 ± 22.0**	**3.9 ± 1.0**	**0.51 ± 0.03**
	xBTF	**544.4 ± 22.3**	4.3 ± 1.1	0.57 ± 0.03
MK10	KGFOA	545.2 ± 29.9	39.7 ± 5.0	2192.05 ± 121.28
	BTF	**416.7 ± 17.8**	**6.8 ± 1.6**	**0.51 ± 0.03**
	xBTF	**416.7 ± 17.4**	**6.8 ± 1.6**	0.60 ± 0.04

Table 4. BTF vs xBTF p-value test results.

p-values	Makespan	Distance to Bound (%)	Execution Time (s)
MK1	0.85	0.66	**<0.05**
MK2	0.88	0.74	**<0.05**
MK3	0.11	**<0.05**	**<0.05**
MK4	0.20	**<0.05**	**<0.05**
MK5	0.27	**<0.05**	**<0.05**
MK6	0.72	0.61	**<0.05**
MK7	0.65	0.16	**<0.05**
MK8	0.71	0.62	**<0.05**
MK9	0.50	**<0.05**	**<0.05**
MK10	1.00	0.99	**<0.05**

perform at a top notch level. Also, its extremely low execution time proves that it can also be applied for rescheduling in dynamic scenarios.

Secondly, xBTF outperforms the BTF for problems with a high resource workload. The p-values of the results between the BTF and xBTF were calculated and are presented in table 4, where all the significant p-values (≤ 0.05) are highlighted. For the Distance to Bound metric, there are four problems for which the p-values are significant. The xBTF proves to be superior at solving two of them (MK3 and MK4). The MK3 and MK4 are two of the most complex problems of the MK1-10 dataset, as they have highly variant processing times and a small number of operations. This can lead to heavy workloads for certain resources because the heuristic might not prioritize their availability, resulting in long processing time tasks being scheduled last, leaving insufficient room for smaller tasks to balance the workload on those resources.

This complexity can be verified considering the very high Distance to Bound results that the BTF had for these problems ($\geq 10\%$). For every problem where the BTF scores greater than 10% at the Distance to Bound metric, the xBTF is either statistically superior or equivalent, which makes it a better alternative for this kind of complex problems with high resource workloads.

Only at the execution times there is a clear superiority of the BTF algorithm. Which comes as no surprise, since, unlike the xBTF, it does not calculate resource allocation penalties. Even so, the execution times of the xBTF are still so small (≤ 0.60 s) that this superiority has no practical meaning. In fact, the execution times are so small that it is reasonable to consider running both algorithms for a new unknown problem and choose the best of these two solutions.

4 Conclusions

The xBTF algorithms was presented in this paper and its performance evaluated. With xBTF a resource allocation penalty was introduced trying to achieve better results at the most complex problems with higher resource demands.

The preliminary results have shown that the xBTF utterly outperforms a high performance metaheuristic, the KGFOA, and it proved to be superior at problems with higher resource demands, for which the BTF scored extremely high at the Distance to Bound metric.

Regarding the execution times, the BTF and xBTF are so fast that there is no practical difference between applying one or the other. In fact, it can be extremely easy and handy to run both algorithms for a new unknown problem and elect the best solution as the final schedule. These very low execution times also make the xBTF algorithm an excellent solution for rescheduling in dynamic job shop problems.

The tremendous performance revealed by the xBTF at solving DRC-FJSSP's may help reignite some of the scientific interest in heuristic algorithms. Keeping their well known advantages of execution speed, in a world where dynamic industrial scenarios are becoming more and more important, this heuristic surprising performance results may open new research opportunities.

Future work may assess the xBTF performance in problems with different performance metrics, such as the weighted tardiness. It should be easy to adapt it for these problems, considering a new threat prioritization rule: the due date minus the total remaining processing times of jobs.

Moreover, research may be carried to apply the xBTF to problems where the processing time of operations varies with the allocated worker-machine pair. Additionally, DRC-FJSSP's where workers do not operate machines throughout the whole processing times of the operations (partial allocation) are also quite interesting for future evaluation of the performance of the algorithm. Considering its nature and design it should be capable of addressing these problems without significant changes.

References

[1] Xie, Jin, et al. 2022. "A new neighbourhood structure for job shop scheduling problems." *International Journal of Production Research*: 1-15.

[2] Liu, Renke, Rajesh Piplani, and Carlos Toro. 2022. "Deep reinforcement learning for dynamic scheduling of a flexible job shop." *International Journal of Production Research*: 1-21.

[3] Garey, Michael R., David S. Johnson, and Ravi Sethi. 1976. "The complexity of flowshop and jobshop scheduling." *Mathematics of operations research* 1 (2): 117-129.

[4] Pinedo, M. 2016. Scheduling: theory, algorithms and systems. 5-th ed. Springer.

[5] Hashemi-Petroodi, S. Ehsan, et al. 2021. "Workforce reconfiguration strategies in manufacturing systems: a state of the art." *International Journal of Production Research* 59 (22): 6721-6744.

[6] Cunha, Mariana M., et al. 2019. "Dual resource constrained scheduling for quality control laboratories." *IFAC-PapersOnLine* 52 (13): 1421-1426.

[7] Defersha, Fantahun M., Dolapo Obimuyiwa, and Alebachew D. Yimer. 2020. "Multiple-Trial/Best-Move Simulated Annealing for Flexible Job Shop Scheduling with Scarce Setup-Operators." *2020 7th International Conference on Soft Computing Machine Intelligence (ISCMI)*. IEEE.

[8] Thürer, Matthias, Mark Stevenson, and Paolo Renna. 2019. "Workload control in dual-resource constrained high-variety shops: an assessment by simulation." *International Journal of Production Research* 57 (3): 931-947.

[9] Gong, Guiliang, et al. 2020. "A hybrid artificial bee colony algorithm for flexible job shop scheduling with worker flexibility." *International Journal of Production Research* 58 (14): 4406-4420.

[10] Zheng, Xiao-long, and Ling Wang. 2016. "A knowledge-guided fruit fly optimization algorithm for dual resource constrained flexible job-shop scheduling problem." *International Journal of Production Research* 54 (18): 5554-5566.

[11] Farjallah, Farah, Houssem Eddine Nouri, and Olfa Belkahla Driss. 2022. "Multi-start Tabu Agents-Based Model for the Dual-Resource Constrained Flexible Job Shop Scheduling Problem." *Conference on Computational Collective Intelligence Technologies and Applications*. Springer, Cham.

[12] Müller, David, and Dominik Kress. 2022. "Filter-and-fan approaches for scheduling flexible job shops under workforce constraints." *International Journal of Production Research* 60 (15): 4743-4765.

[13] Soofi, Parham, et al. 2021. "Robust Fuzzy-Stochastic Programming Model and Meta-Heuristic Algorithms for Dual-Resource Constrained Flexible Job-Shop Scheduling Problem Under Machine Breakdown." *IEEE Access* 9: 155740-155762.

[14] Fernandesa, N. O., et al. 2022. "Worker Assignment in Dual Resource Constrained Systems Subject to Machine Failures: A Simulation Study."

[15] Andrade-Pineda, Jose L., et al. 2020. "Scheduling a dual-resource flexible job shop with makespan and due date-related criteria." *Annals of Operations Research* 291 (1): 5-35.

[16] Li, Yuanyuan, et al. 2020. "Machine learning and optimization for production rescheduling in Industry 4.0." *The International Journal of Advanced Manufacturing Technology* 110 (9): 2445-2463.

[17] Tao, Ze, and Xiaoxia Liu. 2019. "Dynamic Scheduling of Dual-Resource Constrained Blocking Job Shop." *International Conference on Intelligent Robotics and Applications*. Springer, Cham.

[18] Martins, Miguel S. E., et al. 2020. "Reinforcement learning for dual-resource constrained scheduling." *IFAC-PapersOnLine* 53 (2): 10810-10815.

[19] Magalhães, Ricardo, et al. 2021. "Encoder-Decoder Neural Network Architecture for solving Job Shop Scheduling Problems using Reinforcement Learning." *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE.

[20] Magalhães, Ricardo, Susana Vieira, and João Sousa. 2022. "Threat elimination algorithm for Dual Resource Constrained Flexible Job Shop Scheduling Problems." *IFAC-PapersOnLine* 55 (10): 2288-2293.

[21] Brandimarte, Paolo. 1993. "Routing and scheduling in a flexible job shop by tabu search." *Annals of Operations research* 41 (3) : 157-183.