A HYBRID EVOLUTIONARY METAHEURISTIC PROPOSAL APPLIED TO JOB-SHOP SCHEDULING PROBLEMS WITH EARLINESS AND TARDINESS PENALTIES

UMA PROPOSTA DE METAHEURÍSTICA EVOLUCIONÁRIA HÍBRIDA APLICADA A PROBLEMAS DE SEQUENCIAMENTO DA PRODUÇÃO COM PENALIDADES POR ANTECIPAÇÃO E ATRASO

UNA PROPUESTA DE METAHEURÍSTICA EVOLUTIVA HÍBRIDA APLICADA A PROBLEMAS DE PROGRAMACIÓN DE PRODUCCIÓN CON PENALIZACIONES POR ADELANTO Y ATRASO

Cited as:
Conserva, Júlio C. V. & Silva, Yuri L. T. (2024). A hybrid evolutionary metaheuristic proposal applied to job-shop scheduling problems with earliness and tardiness penalties. Revista Gestão & Tecnologia. Journal of Management and Technology. v 24, nº 1. p .31-57.

Júlio Cesar Vasconcelos Conserva
Graduado em Engenharia de Produção pela Universidade Federal de Campina Grande.

Yuri Laio Teixeira Veras Silva
Professor Adjunto no Departamento de Engenharia de Produção da Universidade Federal de Campina Grande.
https://orcid.org/0000-0003-0683-6194

Scientific Editor: José Edson Lara
Organization Scientific Committee
Double Blind Review by SEER/OJS
Received on 02/04/2023
Approved on 08/03/2024

ABSTRACT
Objective: The present study aims to propose an efficient hybrid evolutionary metaheuristic based on a genetic algorithm with local search structures, which allows for high-performance treatment of Job-Shop Scheduling (JSS) problems with earliness and tardiness penalties (JSS-ETC).
Methodology/Approach: The research is characterized as experimental and was carried out by developing the proposed hybrid heuristic to solve the mathematical model problem, JSS-ETC, aiming to achieve efficient results.
Originality/Relevance: A new hybridization approach was proposed for solving JSS-ETC problems based on two heuristics that adopt mechanisms of different natures, demonstrating a vast potential and relevance in the solvability of complex problems, especially scheduling.
Main results: The results showed that the proposed hybrid algorithm (in its two variations) achieved high performance, mathematically and computationally, compared to consolidated methods in the literature. Superior results were achieved in approximately 30% of the tested instances.
Theoretical/methodological contributions: The present study contributes to a better understanding of the problem addressed and the methods used to solve it, presenting a comprehensive literature review on the problem. Additionally, the proposed hybrid algorithm demonstrated high potential for solving and can be considered for other optimization problems.
Contributions to management: The developed algorithm allows for better production planning and control and efficient management of organizational resources.

Keywords: Scheduling, earliness and tardiness costs, genetic algorithm, local search.

RESUMO
Objetivo do estudo: O presente artigo objetiva propor uma metaheurística evolucionária híbrida eficiente, fundamentada em algoritmo genético com estruturas de busca local, que permite tratar com alto desempenho problemas de *Job-Shop Scheduling* (JSS) com penalidades por antecipação e atraso (JSS-ETC).
Metodologia/Abordagem: A pesquisa é caracterizada como experimental, e foi realizada através do desenvolvimento da heurística híbrida proposta, para solucionar o problema referente ao modelo matemático do problema, o JSS-ETC, buscando alcançar resultados eficientes.
Originalidade/Relevância: Foi proposta uma nova abordagem de hibridização para a resolução de problemas de JSS-ETC, com base em duas heurísticas que adota mecanismos de naturezas diferentes, e que demonstrou um amplo potencial e relevância na resolutividade de problemas complexos, em especial de *scheduling*.
Principais resultados: Os resultados mostraram que o algoritmo híbrido proposto (em suas duas variações) obtiveram um alto desempenho, tanto matematicamente, como computacionalmente, quando comparados a métodos consolidados presentes na literatura. Foram alcançados resultados superiores em aproximadamente 30% das instâncias testadas.
Contribuições teóricas/metodológicas: O presente trabalho contribui para uma melhor compreensão do problema abordado e dos métodos utilizados para sua resolução, apresentando uma vasta revisão de literatura sobre o problema tratado. Adicionalmente, o algoritmo híbrido

proposto demonstrou grande potencial de resolução, podendo ser considerado para outros problemas de otimização existentes.
Contribuições para a gestão: O algoritmo desenvolvido permite alcançar um melhor planejamento e controle da produção, assim como uma gestão de recursos organizacionais eficiente.

Palavras-chave: Scheduling, custos por antecipação e atraso, algoritmo genético, busca local.

RESUMEN
Objetivo del estudio: El presente artículo tiene como objetivo proponer una metaheurística evolutiva híbrida eficiente, basada en algoritmos genéticos con estructuras de búsqueda local, que permita tratar con alto rendimiento problemas de programación de Job-Shop (JSS) con penalizaciones por anticipación y retraso (JSS-ETC).
Metodología/Enfoque: La investigación se caracteriza como experimental y se realizó mediante el desarrollo de la heurística híbrida propuesta para resolver el problema del modelo matemático del problema, el JSS-ETC, buscando lograr resultados eficientes.
Originalidad/Relevancia: Se propuso un nuevo enfoque de hibridación para la resolución de problemas de JSS-ETC, basado en dos heurísticas que adoptan mecanismos de diferentes naturalezas, y que demostraron un amplio potencial y relevancia en la resolución de problemas complejos, especialmente de programación.
Principales resultados: Los resultados mostraron que el algoritmo híbrido propuesto (en sus dos variantes) obtuvo un alto rendimiento, tanto matemático como computacional, en comparación con los métodos consolidados presentes en la literatura. Se alcanzaron resultados superiores en aproximadamente el 30% de las instancias probadas.
Aportes teóricos/metodológicos: Este trabajo contribuye a una mejor comprensión del problema abordado y de los métodos utilizados para su resolución, presentando una amplia revisión de literatura sobre el problema tratado. Además, el algoritmo híbrido propuesto demostró un gran potencial de resolución, pudiendo ser considerado para otros problemas de optimización existentes.
Aportes a la gestión: El algoritmo desarrollado permite lograr una mejor planificación y control de la producción, así como una gestión eficiente de los recursos organizacionales.

Palabras clave: Scheduling, costes por anticipación y retraso, algoritmo genético, búsqueda local.

1 INTRODUCTION

In the current scenario of intense competition in various industrial sectors, organizations have sought ways to obtain consistent competitive advantages to stand out in the market. Amid this challenge, Siqueira and Müller (2022) highlight that organizations have realized that the key to creating superior value to their competitors is primarily in better management of their resources.

The management of the numerous resources that organizations need to deal with daily is a very complicated task, which tends to be even more complex depending on the size and amount of existing information. Among the various problems that encompass decision-making involving resources in production processes, one can mention: production batch sizing, production planning, optimization of installed capacity, and production sequencing.

In this context, production sequencing and scheduling problems have recently received significant attention in organizations. Schaller and Valente (2019) highlight that such problems are related to the allocation of physical resources (machines) for the realization/processing of a certain set of tasks (production orders) over a time horizon. That is, they refer to decision-making processes regarding the order in which production orders should be executed on a specific set of machines and stages.

Scheduling problems have several variants in the literature, including Job-Shop Scheduling (JSS) and Flexible Job-Shop (FJSS), which have various application approaches. The classical JSS and FJSS problems deal with common objectives, such as minimizing maximum execution time, machine load, or the number of delayed tasks (jobs). However, in the real world, organizations have increasingly sought to adopt the Just-in-Time (JIT) philosophy to achieve greater efficiency, making it necessary to adopt criteria in scheduling problems that represent the operational costs and adversities caused by earliness and tardiness costs (ETC).

Thus, if a job is completed exactly on the due date, its earliness-tardiness cost will be zero. Otherwise, completing the job before or after the due date incurs an earliness or tardiness cost (Ahmadian et al., 2020).

These particularities can be seen more prominently in the food manufacturing sector, which is characterized by production processes with a high degree of complexity, mainly due to the nature of its finished products. Since they usually deal with inputs and products with short expiration dates and high-quality control, decision-making involving production sequencing in organizations of this nature is of fundamental importance to their operational results and the adequate supply of wholesalers and retailers who serve the final links in the supply chain.

Scheduling problems, in their various natures, involving decision-making processes of high mathematical and computational complexity. They can be found in organizations in various

industrial sectors, both in manufacturing and services (Silva et al., 2018). However, solving some production sequencing problems is a very complex task because several of them belong to the NP-Hard class, which makes their solution through exact methods ineffective in instances of high dimensions, given that it is not possible to obtain their optimal solution in polynomial time, especially in production management problems with multiple stages, machines, and operators.

Therefore, heuristic methods have gained considerable attention in optimizing these problems by presenting satisfactory solutions in a shorter time when compared to exact methods. According to Amjad et al. (2018), the Genetic Algorithm (GA) has proven to be one of the most effective techniques for solving JSS and its variants, being one of the most commonly used techniques today.

The GA technique has been widely used in various fields of science and different types of problems, demonstrating solid results not only in scheduling. This method has been applied in a hybrid way with other heuristics, which often allows for a considerable improvement in its performance. This hybridization has presented interesting results in several studies, highlighting the potential of these techniques, as in the study by Yuce et al. (2017).

Given this scenario, this research studies the Job-Shop Scheduling problem with minimizing Earliness and Tardiness Costs (JSS-ETC), aiming to develop a new optimization approach based on a genetic algorithm hybridized with a local search heuristic, seeking to achieve efficient results in solving problems of this nature. In addition, an extensive literature review was carried out on the topic, aiming to identify the progress achieved with recent research and the scientific gaps and emerging studies on the topic.

2 THEORETICAL BACKGROUND

Several works related to JSS-ETC can be found in the literature, such as FJSS, Just-in-time Job-Shop Scheduling (JIT-JSS), Job-Shop with Time Windows (JSS-TW), among others. Therefore, a comprehensive literature review was conducted to understand the state of the art of the problem, as well as emerging optimization techniques and approaches in the area. Initially, this survey focused on better understanding the most commonly used optimization

methods for solving problems related to JSS-ETC. Then, a mapping of works that use GA in their resolution was performed.

For the bi-objective JSS problem that seeks to minimize the weighted average flow time and total penalties of earliness and tardiness, Tavakkoli-Moghaddam et al. (2011) present a new mathematical model and a multi-objective optimization algorithm by Particle Swarm Optimization (PSO) combined with a Variable Neighborhood Search (VNS) heuristic. The study by Gao et al. (2016) deals with a variant of JSS, the bi-objective FJSS, where a weighted combination aims to minimize the makespan and the average of the total earliness. Kianpour et al. (2021) proposed an exact approach based on mixed integer linear programming (MIP) for JSS with dynamic processing times and due dates.

To solve JSS-TW, Huang et al. (2013) developed a modified Ant Colony Optimization (ACO) algorithm, which showed high-performance results. Yazdani et al. (2017) implemented a MIP-based formulation considering a new objective function formulation, which considers the sum of maximum criteria for ET for JSS. Kramer and Subramanian (2019) developed an efficient local search metaheuristic capable of solving scheduling problems considering several distinct characteristics. In addition, a comprehensive literature review was carried out on related problems.

Kayvanfar et al. (2013) developed an exact method and ILS heuristic for the single-machine sequencing problem to minimize ET, considering controllable processing time. Schaller and Valente (2013) proposed a GA to minimize the total ET in a flow-shop permutation, then performed tests to verify if the method used was consistent in generating reasonable solutions compared to others already proposed. While in Schaller and Valente (2019), they worked with the sequencing problem with flow-shop permutation to minimize the total ET with non-forced idle time allowed. Ahmadian et al. (2021) developed a VNS algorithm for JIT-JSS, allowing good computational results to be achieved.

As seen in the previously cited works, several JSS problems in the literature deal directly with minimizing ET. Among so many different variants, the following are discussed as research that addressed the objective of minimizing costs/penalties for ET, in order to detail their resolution strategies better.

Nogueira et al. (2014) proposed three heuristics to solve the problem of unrelated parallel machine sequencing with ETC, a GRASP heuristic, a second one based on Path Relinking, and one based on iterated local search (ILS). Meanwhile, Zambrano-Rey et al. (2015) developed two metaheuristics for JIT-JSS with quadratic ET, namely a GA and a PSO, and two approaches to deal with task release times. Fazlollahtabar et al. (2015) implemented an exact method for a JSS-ETC problem applied to a multiple automated guided vehicle manufacturing system. Ahmadian et al. (2020) studied JSS-ETC in the context of an open-shop system and developed an efficient relaxation heuristic. After conducting experiments on 72 instances with up to 200 tasks, they demonstrated the effectiveness of the proposed method.

In the literature, numerous approaches have been used for other scheduling problems. The main research studies that have used GA-based approaches as an optimization method for these problems are highlighted below. Montoya-Torres et al. (2010) studied the Resource-Constrained Project Scheduling (RCPS) problem, presenting a GA for it and proposing an alternative chromosome representation using a multi-array model. For JSS, Asadzadeh and Zamanifar (2010) proposed an approach in which the initial population creation and GA parallelization are agent-based. Hosseinabadi et al. (2019) investigated the effects of selecting crossover and mutation operators in GA to solve an open-shop scheduling problem.

JSS studies can be found with various applications, such as in energy efficiency. In this regard, May et al. (2015) showed a new GA capable of considering productivity and energy consumption-related objectives. Meanwhile, Salido et al. (2016) developed a GA for JSS where machines can consume different amounts of energy to process tasks at different rates.

There are still approaches that seek to hybridize GA with other methods to improve algorithm efficiency. This fact is what happens in Wang's research (2012) for JSS, which brings a local search operator that can significantly improve the local search capability of GA. The authors also present genetic operators to increase population diversity: a mixed selection operator based on fitness value, new machine-based crossover operators, and a critical path-based mutation operator. Asadzadeh (2015) also presented a hybridization with local search techniques to improve the effectiveness of GA, and then applied it to benchmark instances to attest to the approach's performance. Another work that presents hybrid GA methodologies is that of Kundakci and Kulak (2016), which aims to minimize makespan in a dynamic JSS.

Studies involving FJSS have also made use of GA as an optimization strategy. In this sense, De Giovanni and Pezzella (2010) presented an improved GA to solve the problem in which a system of multiple flexible manufacturing unit processes jobs. In the study by Liangxiao and Zhongjun (2015), an improved GA was presented to minimize makespan.
On the other hand, Li and Gao (2016) implemented a hybrid algorithm that involves a GA with tabu search (TS) structures. Fang (2022) proposed a hybridization of GA with DHS, which allows for improving the quality of individuals through the adopted elitism operator. Some studies have carried out a vast literature mapping to describe the main variations involving the reported issues (Amjad et al., 2018; Gao et al., 2019).

We can also find studies that use GA in hybrid algorithms to solve ET problems and their variants. Yuce et al. (2017) developed a hybrid algorithm using GA and an artificial bee colony algorithm (BA), where a genetic operator is used in the global search of BA. In Khanh Van and Van Hop's (2021) work, a capacity upgrade via MIP combined with a GA was proposed.

3 PROBLEM DESCRIPTION

With the rise of lean production principles, mainly the concepts of JIT, sequencing problems involving costs for anticipation or delay have received significant attention. This fact can be explained by its importance for companies to achieve objectives according to this philosophy. According to JIT, anticipation and delay are harmful to profitability and, therefore, should be minimized.

In this context, the problem of single-machine production sequencing has been widely studied. In this problem, *n* jobs are available at time zero to be processed on a single machine and delivered by a due date d_i, for each job *i*. Each job *i* requires exactly one operation, and its processing time p_i is known. If job *i* is completed before the due date, its anticipation is given by $E_i = d_i - C_i$, where C_i is the completion time of job *i*. On the other hand, if task *i* is completed after the desired date, its delay is given by $T_i = C_i - d_i$. Each *job i* has its own unit cost for anticipation (α_i) and delay (β_i). he objective of the problem is to obtain an optimal schedule of task execution that minimizes the sum of the penalties for anticipation and delay (Ronconi & Kawamura, 2010). The main parameters for the problem are presented below:

d_i → Due date of job *i*;

α_i → Penalty per unit time of anticipation for the job *i*;

β_i → Penalty per unit time of delay for job *i*;

p_i → Processing time of job *i*;

R → A very large number.

Consider the following decision variables:

x_{ik} → A binary variable equal to 1 if job *i* is sequenced before job *k* and 0 otherwise;

C_i → Completion time of job *i*;

E_i → Anticipation of job *i*;

T_i → Delay of job *i*;

Thus, the mathematical model that represents the problem is given by:

$$Min \sum_{i=1}^{n} \alpha_i E_i + \sum_{i=1}^{n} \beta_i T_i \quad (1)$$

$$s.t. \quad T_i - E_i = C_i - d_i, \qquad i = 1, 2, \dots, n, \quad (2)$$

$$C_i \leq C_k - p_k + R(1 - x_{ik}), \qquad i = 1, 2, \dots, n-1, \qquad k = i+1, \dots, n, \quad (3)$$

$$C_k \leq C_i - p_i + R x_{ik}, \qquad i = 1, 2, \dots, n-1, \qquad k = i+1, \dots, n, \quad (4)$$

$$C_i - p_i \geq 0, \qquad i = 1, 2, \dots, n, \quad (5)$$

$$T_i \geq 0, \qquad i = 1, 2, \dots, n, \quad (6)$$

$$E_i \geq 0, \qquad i = 1, 2, \dots, n, \quad (7)$$

$$x_{ik} \in \{0,1\}, \qquad i = 1, 2, \dots, n-1, \qquad k = i+1, \dots, n. \quad (8)$$

In the formulation above, (1) represents the objective function of the problem, which consists of minimizing the sum of penalties for delay and anticipation. The set of constraints (2) indicates that the difference between the completion time of a task and its deadline must be equal to the difference between the delay and the anticipation of the order. Constraints (3) and (4) ensure feasibility in the calculations related to the completion times of each order. Constraint (5) ensures that the initial time of each order is non-negative. The set of constraints (6) and (7)

defines the non-negativity of the variables T_i and E_i, while constraint (8) defines the variable x_{ik} as binary.

4 PROPOSED HYBRID GENETIC ALGORITHM WITH LOCAL SEARCH

In the pursuit of increasingly efficient methods for solving optimization problems, heuristics have been studied and developed to consider nature's survival and reproduction behaviors in their search mechanisms. In this context, evolutionary algorithms emerge, which are generally inspired by these biological processes of evolution and were initially presented by Holland (1992).

Evolutionary heuristics work with a population of candidate solutions simultaneously, allowing for a broader sweep of the solution space, thus exploring different regions. The search process allocates a number of individuals for exploration in each region, prioritizing the most promising ones without neglecting the others. This strategy is possible due to memory and learning mechanisms among individuals of a particular population and the randomness present in the method, which contributes to escaping local optimal solutions (Pacheco, 2017).

Freitas (2018) emphasizes that in evolutionary algorithms, some biology terms denote optimization elements so that generations are iterations, individuals are solutions, the population is a set of solutions from each iteration, and the fitness function is the objective function of the algorithm.

This study aimed to develop a hybrid evolutionary algorithm based on Genetic Algorithm (GA) and local search mechanisms to solve the problem highlighted in Section 3. Generally, a GA is an evolutionary method that draws inspiration from the principles of natural selection, where fitter individuals have a higher probability of reproduction, and individuals with more offspring have a greater chance of perpetuating their genetic code for the next generation.

Similar to other evolutionary heuristics, the GA operates with a set of solutions called a population, and each member of this population can be referred to as an individual or a chromosome. Each individual is evaluated and assigned a fitness value, where the best solutions to the problems are correlated with the best fitness values. Based on the analysis of the fitness of each individual, another operator of the genetic algorithm comes into action, which is known as the selection mechanism, which is biased towards fitter individuals in the population,

meaning that the best solutions have a higher chance of being selected, as occurs in the process of natural selection. Figure 1 illustrates the functioning of the GA proposed in this work and the operators that were considered.

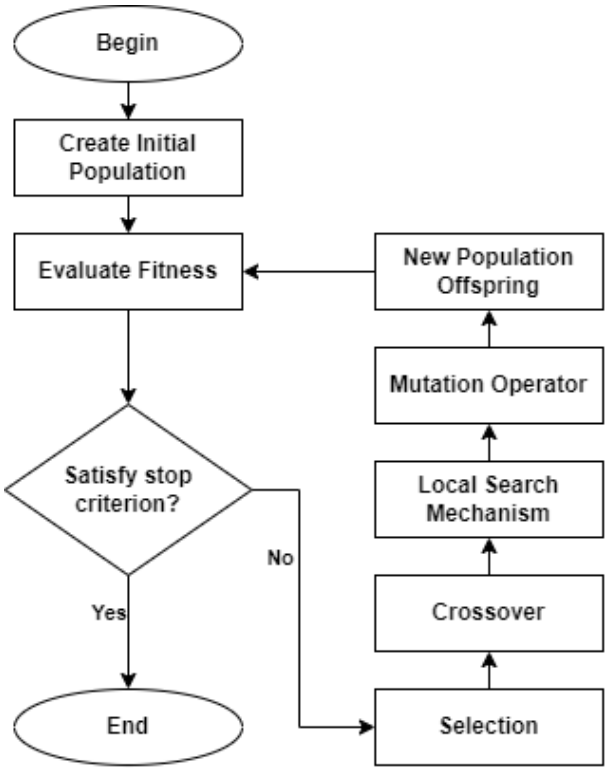


Figure 1 - hybrid genetic algorithm proposed
Source: Elaborated by the authors (2022).

4.1 Population representation

In GA, each individual can be represented by a chain of genes. These genes can be symbolized in binary form or with real numbers, depending on the characteristics of the problem being worked on (Pacheco, 2017). In this research, each chromosome should represent an ordered list of jobs (production orders), which symbolizes the sequence in which these jobs should be processed, i.e., each gene represents a production order in its respective order. Figure 2 illustrates this representation of individuals and the population.

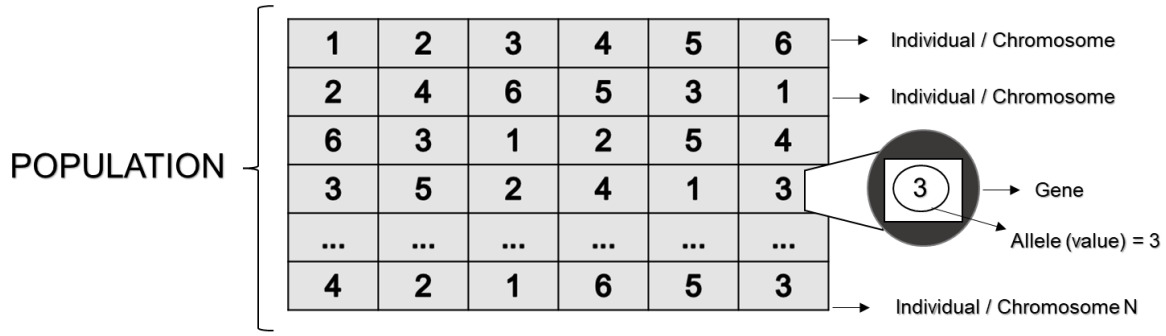


Figure 2 - Population Representation

4.2 Fitness function

Asadzadeh (2015) explains in his work that in GA the fitness of individuals in the population is calculated through the fitness function, which measures the quality of a candidate solution to a given problem. For this purpose, the function returns a value obtained through this evaluation, which will depend on the objective of the problem being analyzed.

In the problem studied, the fitness function represents the sum of costs for anticipation and delay of candidate solutions. Since it is a minimization problem, the solutions that have better fitness are those that achieve a lower fitness value. This characteristic will favor higher quality solutions throughout each iteration of the GA, making them more likely to be selected when generating new solutions.

At each new population generation, the fitness function is responsible for comparing whether the new solutions have improved with respect to the best global solution already found. The candidate solution becomes the new best global solution if it has better fitness.

4.3 Procedures and operators

Even before implementing the GA operators, the initial step to be taken is generating an initial population for the problem, which can be constructed randomly (greedy) or even with the use of more traditional heuristics such as local search. To do this is necessary to establish the size of the population to be generated and the number of genes.

4.3.1 Selection procedure

The selection process is a fundamental principle of GA, as it is the one that will ensure that, after several generations, the initial population of individuals will generate more fit individuals. Such selection methods are developed to prioritize individuals with better fitness, but not purely these, as it is necessary to maintain population diversity (Goldberg & Deb, 1991).

A well-established mechanism for selection is the Roulette wheel, where individuals are chosen by means of a draw, in which each chromosome is represented on the wheel proportionally to its fitness. Thus, those with higher values have a greater chance of being selected. However, because the problem studied here involves costs, it is necessary to make a representation to provide individuals with lower costs a better fitness, i.e., an inversely proportional choice probability. For this purpose, an adaptation is made to the equation presented by Goldberg and Deb (1991) for calculating the selection probability of an individual by the proportional reproduction method. Thus, if f_i is the fitness value of individual i in the population, the probability of being selected is:

$$p_i = \frac{\frac{1}{f_i}}{\sum_{j=i}^{n} \frac{1}{f_i}} \qquad (9)$$

In Figure 3, the operation of the selection operator is illustrated:

Individual S_i	Fitness $f(S_i)$	Relative Fitness
S_1	90	0,07
S_2	50	0,12
S_3	15	0,41
S_4	40	0,15
S_5	25	0,25

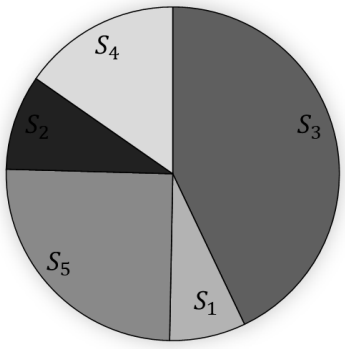


Figure 3 - Selection Procedure

4.3.2 Crossover mechanism

Another operator of the GA is the crossover, which aims to generate new chromosomes by crossing the selected individuals (parents) from the selection operator. This crossing causes the recombination of parent characteristics, providing future generations with inherited

characteristics. This operator can be applied using a crossover probability rate and can be used in various ways, such as one-point crossover and two-point crossover.

In one-point crossover, a point within the range of 0 to n (chromosome length) is randomly selected. Then, the genes within the selected interval are passed to the first child from the second parent and the second child from the second parent. In the first child, empty genes are filled by sequentially traversing the genetic material of the second parent and assigning the values of the genes not yet present in this child. The same happens in the second child, but the genetic material of the first parent is traversed to get the remaining values.

In this research, the two-point crossover was based on an adaptation of the Order Crossover as presented in Scofield (2002), where the children are generated by choosing a piece of the chromosome and then maintaining the relative sequence of jobs from another chromosome. First, two cut points are chosen for the two parent chromosomes. After choosing the cut points, the segments are passed on to the children.

The following steps are used to generate the children: In the generation of the first child, missing values are filled by traversing the second parent to select the values in order, omitting the values already in the first child segment. In the generation of the second child, missing values are filled by traversing the first parent to select the values in order, omitting the values already in the second child segment. Figure 4 illustrates the crossover processes described.

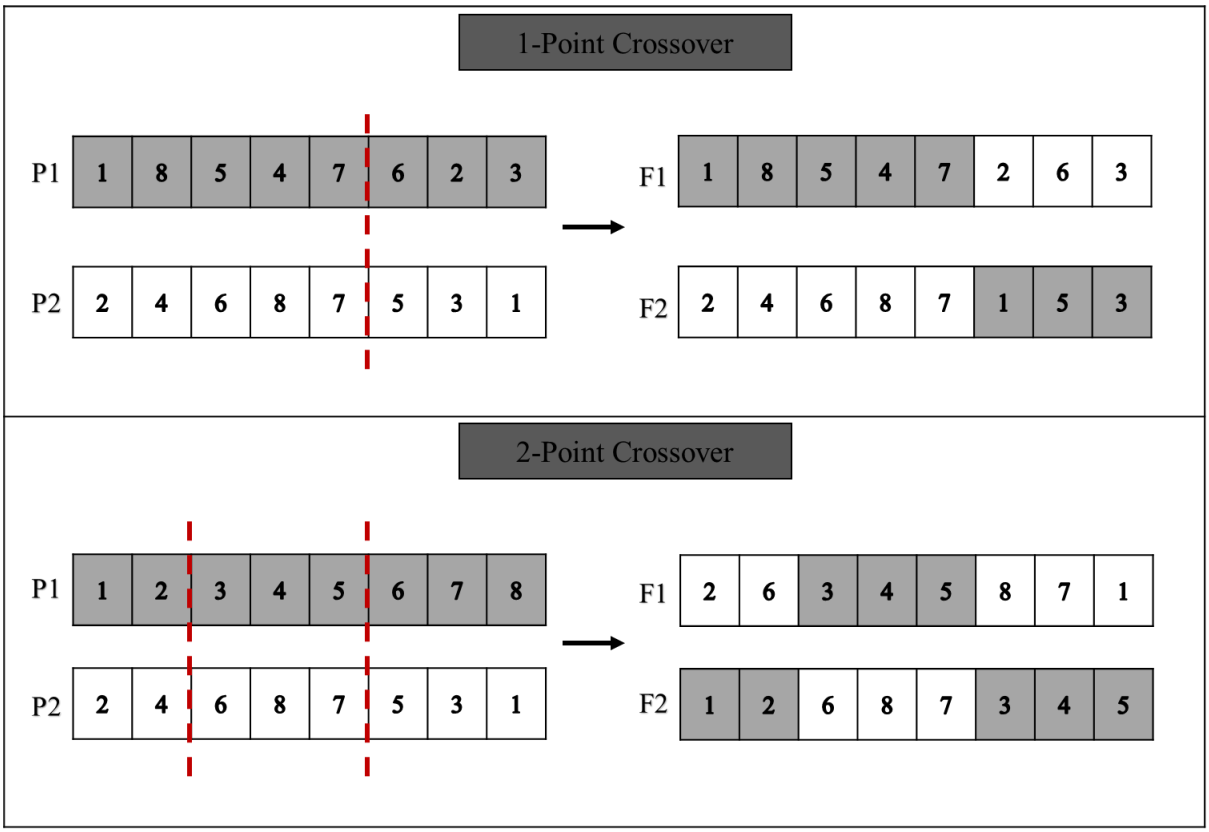


Figure 4 - Crossover Operator

4.3.3 Local search operator

The local search operator aims to improve the solution to the problem by altering the current solution through one or more exchanges. In this research, we specifically used the Swap mechanism, which involves exchanging the position of two production orders, considering the probability of this inversion occurring.

Therefore, this study proposes a hybridization of a genetic algorithm with a local search, a procedure frequently found in the literature. This high usage is explained by the fact that such methods significantly improve the quality of the solutions found by algorithms. Figure 5 exemplifies the operation of the local search heuristic using the swap mechanism:

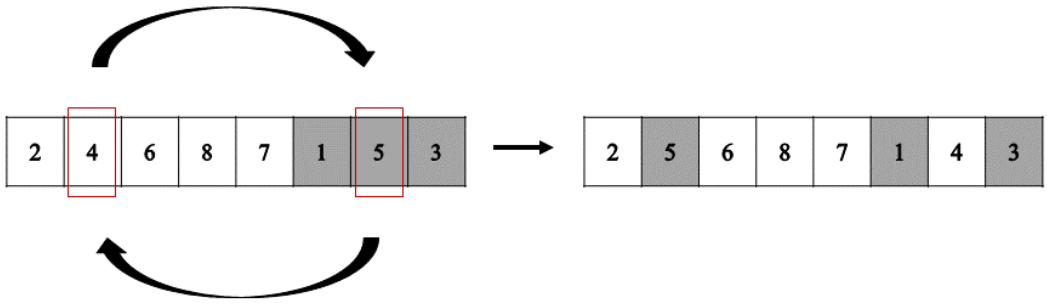


Figure 5 - Local Search Operator

4.3.4 Mutation

This operator is used as an essential strategy for maintaining population diversity by swapping the values of two components (genes) of a certain individual in the population. This strategy allows for exploring different regions of the search space, thus enabling escape from local minima or maxima.

Due to the characteristics of the problem being addressed, this mutation operator can be put into practice using the same swap principle used in local search. Simply swapping the position of two genes (production orders) is enough to perform the mutation. This procedure was previously illustrated in Figure 5.

4.4 Population update

This process is responsible for updating the current population. Thus, chromosomes generated in the previous stages and those with lower fitness are removed from the population, while those newly generated chromosomes are inserted. This procedure, known as Elitism, prevents the population from worsening over iterations by ensuring that only the fittest individuals survive for the next generation, thus improving the algorithm's memory mechanism. From there, the population is re-evaluated, and the algorithm's execution flow continues until the established stopping criterion is reached.

4.5 Stopping criterion

The stopping criterion is a crucial factor in GA as it can determine the algorithm's optimization and execution time. The algorithm proposed here considered a criterion where the number of generations the algorithm should perform was defined as equal to 5000 generations.

5 PRESENTATION AND DISCUSSION OF RESULTS

This section is dedicated to presenting the computational tests developed in this work. Initially, the problems applied in this study and how the instances considered for the experiments are composed are described. Next, the necessary parameters for the algorithms and the values used in the tests are presented. Finally, the computational results are presented. Firstly, the performance of the two proposed hybridized Genetic Algorithms was compared. Subsequently, a comparison was made with other procedures existing in the literature.

5.1 Computational experiments

The algorithms proposed in this research were tested on two production sequencing problems. In the first experiment, instances were generated according to instances present in the OR-Library for the Common Due Date (CDD) scheduling problem. The second experiment used benchmark instances from Tanaka et al. (2009) in the OR-Library for the Different Due Date (DDD) scheduling problem.

For CDD, instances with six different numbers of Jobs $n \in \{40, 20, 50, 100, 200, 500\}$ and four restrictive factors $h \in \{0.2, 0.4, 0.6, 0.8\}$. The h factor represents how constrained the production line is at the beginning of the schedule and is used to find the common due date according to the expression (10): $= h \sum_{i=1}^{n} p_i$, where p_i represents the processing time of job i. The instances used in this experiment consider processing times as integers with a uniform distribution between [1, 20], early penalty costs between [1, 10], and tardiness penalty costs between [1, 15]. Each file available in the OR-Library contains ten instances that, when evaluated considering different n of jobs and different h values, generate a total of 240 instances experimented for this problem.

For DDD, instances with three different numbers of jobs n ∈ {40, 50, 100}, and processing times between [1, 100] are considered. These instances also consider early and tardiness penalty costs in the range of [1, 10] for both cases. For each group with a different n of jobs, there are 125 instances, resulting in a total of 375 instances considered for this problem.

Before testing the proposed algorithms on the set of instances for both mentioned problems, the necessary parameters were defined to conduct these experiments. Table 1 shows the parameters: number of generations ($Generations$), which is used as the stopping criterion

for the algorithm, population size (P_{Size}), crossover probability (P_C), mutation probability (P_M), mutation probability per number of iterations without finding a better solution $(P_{M/I})$, local search probability (P_{LS}), number of iterations without improvement ($Iteration$), to then apply mutation $(P_{M/I})$, and number of individuals in the Elitism ($Elitism$).

Table 1
Values of Parameters Used

Parameter	Value
($Generations$) Stopping criterion	5000
(P_{Size}) Population size	20
(P_C) Crossover probability	0.9
(P_M) Mutation probability	0.04
$(P_{M/I})$ Mutation probability per lack of improvement	0.5
(P_{LS}) Local search probability	0.2
($Iteration$) Iterations without improvement to apply $(P_{M/I})$	0.2($Generations$)
($Elitism$) Individuals in elitism	0.2 (P_{Size})

5.2 Computational experiments

The optimization algorithms developed were implemented using Julia version 1.7.3, a high-level dynamic programming language widely used lately due to its excellent numerical and scientific performance. All experiments were conducted on a 64-bit computer with an Intel(R) Core (TM) i5-7200U 2.5 GHz processor and 8 GB of RAM.

First, the performance of the two proposed heuristics was analyzed on a set of instances for Common Due Date scheduling problem. The preliminary results of the experiments are shown in Table 2, where column 1 represents the number of jobs in the instance studied, column 2 the value of parameter h, and columns 3, 4, and 5 represent, respectively, the minimum, mean, and maximum values found by algorithm HGA_1, while column 6 shows the computational execution time in seconds. The following columns present the results of the same items analyzed for HGA_1, but now for algorithm HGA_2.

Table 2
Performance of the Algorithms in the CDD Problem

n	h	HGA_1				HGA_2			
		Fitness			CPU Time (s)	Fitness			CPU Time (s)
		Min	Average	Max	Average	Min	Average	Max	Average
10	0,2	1001	1651	2139	0,94	1001	1651	2139	1,96
	0,4	619	961	1374	0,91	619	961	1374	1,94
	0,6	550	779	1083	0,93	550	779	1083	1,97
	0,8	560	969	1432	0,94	560	969	1432	1,94
20	0,2	3412	6161	10349	1,35	3408	6170	10352	5,26
	0,4	2090	3621	6191	1,41	2091	3636	6209	5,16
	0,6	1710	2904	4137	1,39	1729	2924	4160	5,18
	0,8	2125	3718	5567	1,37	2127	3715	5557	5,19
50	0,2	28671	36192	44648	2,73	28381	36354	44756	27,34
	0,4	17655	21251	26080	2,74	17819	21461	26527	27,53
	0,6	14788	16827	23290	2,74	14686	17032	23353	27,86
	0,8	17084	21938	31391	2,76	17401	22160	31486	27,73
100	0,2	122055	137946	166280	5,32	122979	138023	165599	110,83
	0,4	77065	84047	100846	5,18	75236	84056	101756	111,23
	0,6	62968	72459	90174	5,29	61702	72204	90471	110,37
	0,8	73778	95287	127103	5,44	76394	95244	127463	111,78
200	0,2	487522	543150	611538	10,86	493688	546044	612755	516,84
	0,4	310218	336437	378761	11,05	307947	337790	381489	519,20
	0,6	258892	290443	321970	11,08	267954	295086	324137	521,15
	0,8	344174	388476	438290	11,40	342647	390263	436429	523,98
500	0,2	3079966	3434003	3680198	28,99	3045332	3395916	3665630	4760,95
	0,4	1889502	2126900	2265862	29,82	1845837	2103469	2259273	4718,87
	0,6	1712169	1898729	2036687	31,01	1673721	1864544	2026031	4741,85
	0,8	2241324	2519253	2681528	31,04	2280477	2510578	2686596	4757,90

The HGA_1 algorithm has a faster execution time compared to HGA_2, regardless of the group of instances analyzed. It can be explained by analyzing the characteristics of the genetic operators present in each algorithm. While HGA_1 uses 1-point crossover, HGA_2 uses 2-point crossover, demanding more iterations to fill the genes that will have to receive genetic material from the second parent without repeating the inherited genes from the first parent.

Subsequently, after performing tests with the algorithms on the CDD problem, new experiments were conducted with the same algorithms on the Different Due Date scheduling problem. The preliminary results of the DDD experiments are presented in Table 3, where they

are organized following the same pattern seen in Table 1, except for the restrictive factor h, which was not used in the DDD tests. Thus, the fitness values and computational time for the HGA_1 and HGA_2 algorithms were presented in each group of instances of the problem.

Table 3
Performance of Algorithms in DDD Problem

n	HGA_1				HGA_2			
	Fitness			CPU Time (s)	Fitness			CPU Time (s)
	Min	Average	Max	Average	Min	Average	Max	Average
40	16433	58409	160070	3,70	16423	58170	160389	24,91
50	26279	86131	241681	3,85	26354	85485	243065	38,10
100	129372	370533	924691	6,51	126912	374157	925917	157,04

The better performance of HGA_1 regarding computational execution time remained when compared to HGA_2. However, when analyzing the average fitness obtained for each instance group, it can be observed that HGA_2 performed better on instances of groups with 40 and 50 jobs. In comparison, HGA_1 was better on instances with 100 jobs.

5.3 Benchmark with CDD literature

The results obtained from the proposed metaheuristic procedures were compared with the main results in the literature by confronting the values obtained for the objective function in each Common Due Date Scheduling instance group. The objective function values (fitness) used for benchmarking were obtained from the work of Biskup and Feldmann (2001). Table 4 provides the minimum, average, and maximum Gap values obtained by each proposed algorithm (F_{PA}) concerning the Benchmark value (F_{BF}) of Biskup and Feldmann (2001), called BF in this study. These values were calculated according to Equation (11).

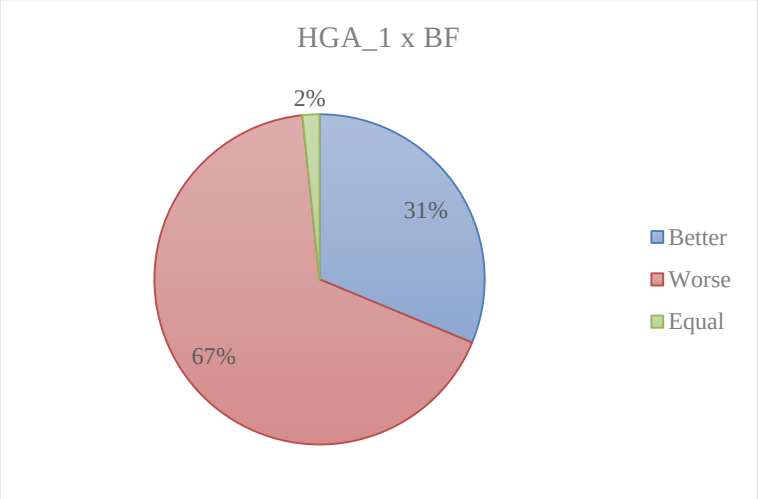
$$GAP = \frac{(F_{PA})-(F_{BF})}{(F_{BF})} \qquad (11)$$

Table 4
Gap between values obtained by HGA and BF (Biskup & Feldmann, 2001)

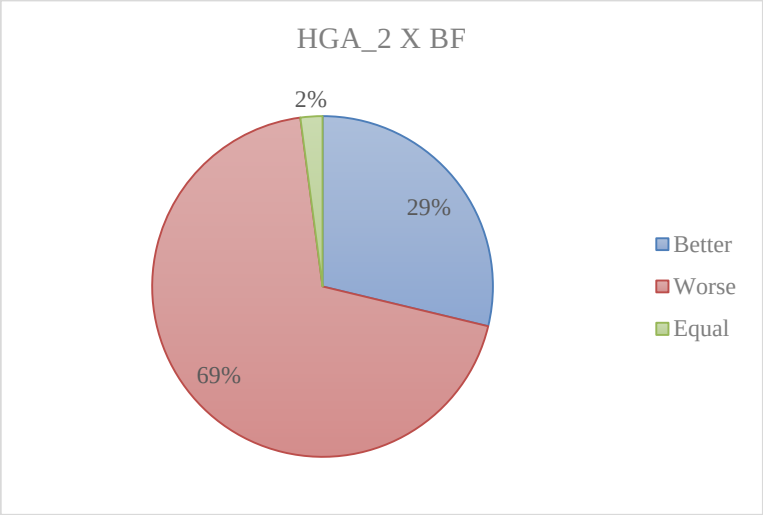
n	h	HGA_1			HGA_2		
		GAP_{min} %	GAP_{avg} %	GAP_{max} %	GAP_{min} %	GAP_{avg} %	GAP_{max} %
10	0,2	-12,17%	-7,71%	-7,71%	-12,17%	-7,71%	-3,36%
	0,4	-9,15%	-3,22%	-3,22%	-9,15%	-3,22%	2,11%
	0,6	-1,72%	6,76%	6,76%	-1,72%	6,76%	42,60%

	0,8	3,70%	36,79%	36,79%	3,70%	36,79%	132,85%
20	0,2	-9,85%	-4,16%	-4,16%	-9,50%	-4,00%	-0,45%
	0,4	-62,15%	-7,94%	-7,94%	-62,04%	-7,57%	1,24%
	0,6	-1,04%	2,66%	2,66%	-0,36%	3,39%	9,11%
	0,8	12,36%	36,21%	36,21%	12,46%	36,10%	61,90%
50	0,2	-7,09%	-3,76%	-3,76%	-6,52%	-3,38%	1,71%
	0,4	-6,01%	-0,83%	-0,83%	-5,31%	0,14%	4,43%
	0,6	2,78%	5,47%	5,47%	3,20%	6,78%	11,23%
	0,8	22,45%	37,74%	37,74%	24,72%	39,23%	51,45%
100	0,2	-5,54%	-2,27%	-2,27%	-4,90%	-2,19%	0,43%
	0,4	-4,61%	2,40%	2,40%	-3,96%	2,34%	6,71%
	0,6	8,90%	11,48%	11,48%	8,50%	11,02%	13,46%
	0,8	33,47%	46,16%	46,16%	33,06%	46,08%	57,67%
200	0,2	-3,01%	-0,07%	-0,07%	-1,76%	0,49%	2,93%
	0,4	2,56%	6,83%	6,83%	3,60%	7,23%	10,26%
	0,6	8,31%	13,31%	13,31%	9,04%	15,14%	18,79%
	0,8	38,72%	51,58%	51,58%	40,20%	52,27%	60,42%
500	0,2	0,00%	2,57%	2,57%	-1,13%	1,42%	3,09%
	0,4	7,85%	11,00%	11,00%	6,27%	9,75%	11,85%
	0,6	17,96%	19,84%	19,84%	16,15%	17,69%	18,84%
	0,8	52,64%	58,95%	58,95%	55,23%	58,45%	63,32%

By analyzing the results presented in Table 4, it can be observed that the HGA_1 and HGA_2 algorithms obtained satisfactory results in several instances, in some cases even surpassing the values obtained by BF. Graphs 1 and 2 present graphically the comparison results, showing in percentage terms how many instances the proposed algorithms achieved better, equal, or worse performance than the one used for comparison.



Graph 1 - HGA_1 Performance Compared to BF



Graph 2 - HGA_2 Performance Compared to BF

The results presented in Graph 1 demonstrate that out of all the instances used in the present study, HGA_1 obtained better results compared to BF in 31% of them, identical results in 2%, and worse results in 67%. While Graph 2 shows that HGA_2 achieved better results in 29% of the instances, identical results in 2%, and worse results in 69%, thus slightly decreasing performance compared to HGA_1.

To perform a more detailed analysis of the results and better understand which instance groups the algorithms obtained better performance, some additional graphs were developed, allowing a better interpretation of the results. These graphs make it possible to visualize the behavior of each algorithm by instance group, being able to see how the fitness (objective function) obtained by a specific algorithm is impacted by the increase of the restrictive factor h, which directly impacts the increase of the delivery date of orders (jobs). Figure 6, which presents the benchmark between the developed algorithms and the one of BF, is shown below.

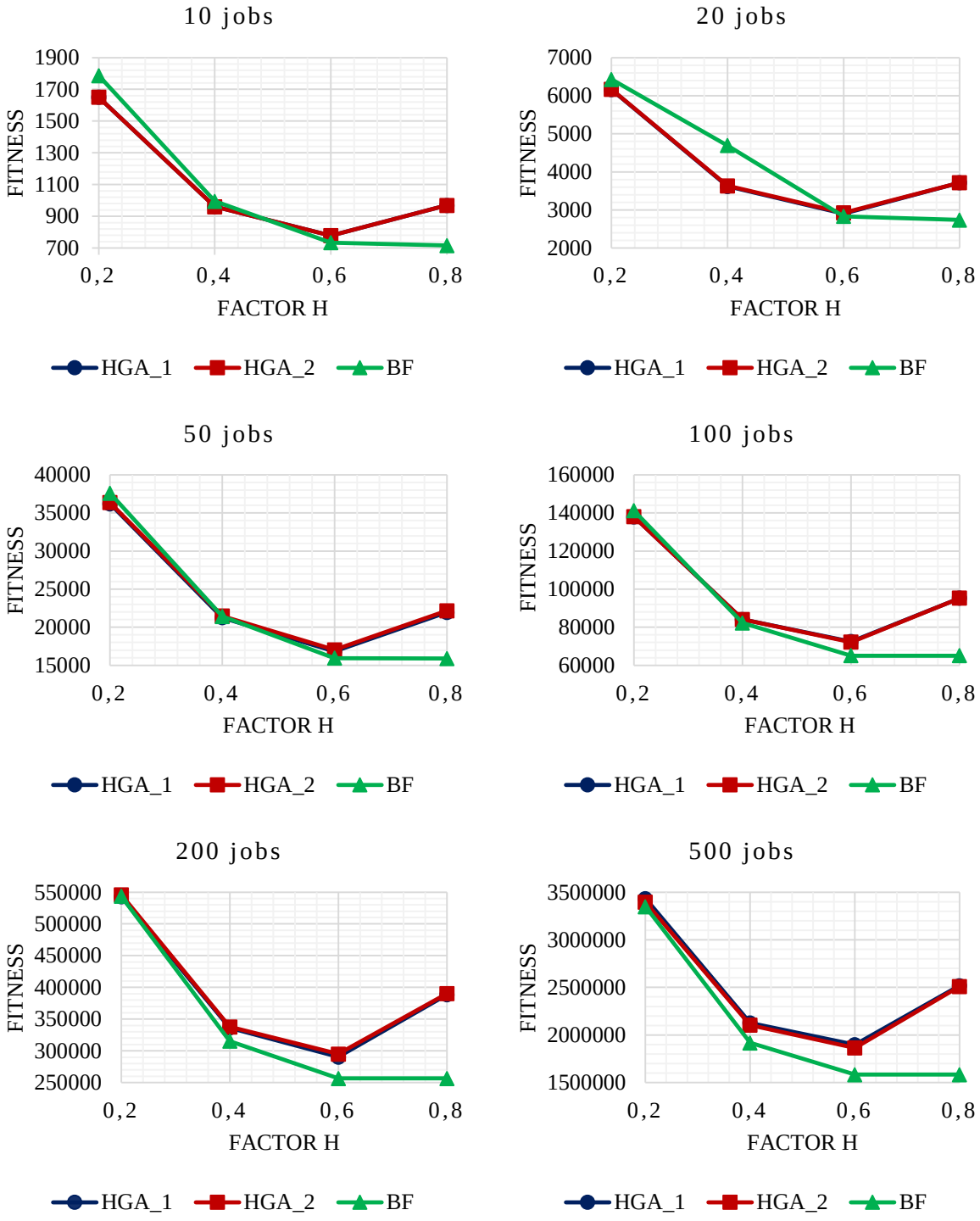


Figure 6 - Comparison Between HGA_1, HGA_2 and BF

Observing the above graphs makes it possible to verify that the proposed metaheuristics can be more efficient, mainly in instances with up to 100 jobs containing the restrictive factor

h equal to 0.2 and 0.4. In larger instances, that is, with 200 and 500 jobs, the HGA_1 and HGA_2 algorithms lose performance compared to BF. When analyzing instances with h equal to 0.6 and 0.8, regardless of the number of jobs, the BF algorithm obtains better results.

It was uncovered that as the h factor increases, the due dates of jobs increase, and the results obtained by the work of Biskup and Feldmann (2001) improve, showing a downward trend in the objective function. On the other hand, the HGA_1 and HGA_2 algorithms showed worse results when the h factor passed from 0.6 to 0.8. This phenomenon can be easily explained by the fact that the works that are precursors in CDD consider solutions in which the first job to be produced does not necessarily need to start at time zero, while HGA_1 and HGA_2 require it. This restriction ends up causing that, as the delivery time increases, the initial orders generate more penalties for anticipation, negatively impacting the objective function.

6 CONCLUDING REMARKS

The present work proposed and presented a hybrid metaheuristic (with two variations in its search strategy) capable of solving scheduling problems related to the Just-in-Time philosophy, precisely problems involving setup costs and production delays. To this end, a hybrid approach between genetic algorithms and local search mechanisms was developed and implemented using the Julia language.

The proposed algorithms contributed effectively to achieving good results. Both showed considerable performance in obtaining production sequencing, considering the objectives of minimizing setup costs and delays in the two problems tested. Additionally, the developed approach achieved efficient solutions in a good computational time.

Furthermore, in terms of comparison with other methods proposed in the literature, computational experiments showed the competitive performance of the proposed metaheuristics, presenting a benchmark of the results obtained with the main existing methods for solving the studied problem. The performance of the proposed approaches presented significant efficiency in smaller instances, especially in the problem that considers common delivery dates.

These computational tests also showed that both algorithms obtained superior results to those reported in the benchmark literature in some parameters, especially when considering

instances of up to 100 production orders, which do not occur in instances with higher orders. Thus, the results show gaps for improvement with the strategies used in the algorithms in future research, mainly to deal with instances with a more significant number of jobs.

For a better direction of future research, it is proposed that methods for solving scheduling problems involving setup costs and delays (ETC) that consider the preventive maintenance schedule of machines be considered, a topic recently proposed in the literature.

REFERENCES

Ahmadian, M. M., Salehipour, A., & Cheng, T. C. E. (2021). A meta-heuristic to solve the just-in-time job-shop scheduling problem. *European Journal of Operational Research*, *288*(1), 14-29.

Ahmadian, M. M., Salehipour, A., & Kovalyov, M. (2020). An efficient relax-and-solve heuristic for open-shop scheduling problem to minimize total weighted earliness-tardiness. *Available at SSRN 3601396*.

Amjad, M. K., Butt, S. I., Kousar, R., Ahmad, R., Agha, M. H., Faping, Z., ... & Asgher, U. (2018). Recent research trends in genetic algorithm based flexible job shop scheduling problems. *Mathematical Problems in Engineering*, *2018*, 1-32.

Asadzadeh, L. (2015). A local search genetic algorithm for the job shop scheduling problem with intelligent agents. Computers & Industrial Engineering, 85, 376-383.

Asadzadeh, L., & Zamanifar, K. (2010). An agent-based parallel approach for the job shop scheduling problem with genetic algorithms. *Mathematical and Computer Modelling*, *52*(11-12), 1957-1965.

Biskup, D., & Feldmann, M. (2001). Benchmarks for scheduling on a single machine against restrictive and unrestrictive common due dates. *Computers & Operations Research*, *28*(8), 787-801.

De CM Nogueira, J. P., Arroyo, J. E. C., Villadiego, H. M. M., & Gonçalves, L. B. (2014). Hybrid GRASP heuristics to solve an unrelated parallel machine scheduling problem with earliness and tardiness penalties. *Electronic Notes in Theoretical Computer Science*, *302*, 53-72.

De Giovanni, L., & Pezzella, F. (2010). An improved genetic algorithm for the distributed and flexible job-shop scheduling problem. *European journal of operational research*, *200*(2), 395-408.

Fang, J. (2022). An Effective Hybrid Multiobjective Flexible Job Shop Scheduling Problem Based on Improved Genetic Algorithm. *Scientific Programming*, *2022*.

Fazlollahtabar, H., Saidi-Mehrabad, M., & Balakrishnan, J. (2015). Mathematical optimization for earliness/tardiness minimization in a multiple automated guided vehicle manufacturing system via integrated heuristic algorithms. *Robotics and Autonomous Systems*, *72*, 131-138.

Freitas, D. M. D. (2018). Geração evolucionária de heurísticas para localização de defeitos de software.

Gao, K. Z., Suganthan, P. N., Pan, Q. K., Chua, T. J., Cai, T. X., & Chong, C. S. (2016). Discrete harmony search algorithm for flexible job shop scheduling problem with multiple objectives. *Journal of Intelligent Manufacturing*, *27*, 363-374.

Gao, K., Cao, Z., Zhang, L., Chen, Z., Han, Y., & Pan, Q. (2019). A review on swarm intelligence and evolutionary algorithms for solving flexible job shop scheduling problems. *IEEE/CAA Journal of Automatica Sinica*, *6*(4), 904-916.

Goldberg, D. E., & Deb, K. (1991). A comparative analysis of selection schemes used in genetic algorithms. In *Foundations of genetic algorithms* (Vol. 1, pp. 69-93). Elsevier.

Holland, J. H. (1992). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press.

Huang, R. H., Yang, C. L., & Cheng, W. C. (2013). Flexible job shop scheduling with due window—a two-pheromone ant colony approach. *International Journal of Production Economics*, *141*(2), 685-697.

Kayvanfar, V., Mahdavi, I., & Komaki, G. M. (2013). Single machine scheduling with controllable processing times to minimize total tardiness and earliness. *Computers & Industrial Engineering*, *65*(1), 166-175.

Khanh Van, B., & Van Hop, N. (2021). Genetic algorithm with initial sequence for parallel machines scheduling with sequence dependent setup times based on earliness-tardiness. *Journal of Industrial and Production Engineering*, *38*(1), 18-28.

Kianpour, P., Gupta, D., Krishnan, K. K., & Gopalakrishnan, B. (2021). Automated job shop scheduling with dynamic processing times and due dates using project management and industry 4.0. *Journal of Industrial and Production Engineering*, *38*(7), 485-498.

Kramer, A., & Subramanian, A. (2019). A unified heuristic and an annotated bibliography for a large class of earliness–tardiness scheduling problems. *Journal of Scheduling*, *22*(1), 21-57.

Kundakcı, N., & Kulak, O. (2016). Hybrid genetic algorithms for minimizing makespan in dynamic job shop scheduling problem. *Computers & Industrial Engineering*, *96*, 31-51.

Li, X., & Gao, L. (2016). An effective hybrid genetic algorithm and tabu search for flexible job shop scheduling problem. *International Journal of Production Economics*, *174*, 93-110.

Liangxiao, J., & Zhongjun, D. (2015). An improved genetic algorithm for flexible job shop scheduling problem. In *2015 2nd International Conference on Information Science and Control Engineering* (pp. 127-131). IEEE.

May, G., Stahl, B., Taisch, M., & Prabhu, V. (2015). Multi-objective genetic algorithm for energy-efficient job shop scheduling. *International Journal of Production Research*, *53*(23), 7071-7089.

Montoya-Torres, J. R., Gutierrez-Franco, E., & Pirachicán-Mayorga, C. (2010). Project scheduling with limited resources using a genetic algorithm. *International Journal of Project Management*, *28*(6), 619-628.

Pacheco, A. (2017). O algoritmo genético – GA. *Computação Inteligente* [Web page]. *https://computacaointeligente.com.br/algoritmos/o-algoritmo-genetico/*

Rahmani Hosseinabadi, A. A., Vahidi, J., Saemi, B., Sangaiah, A. K., & Elhoseny, M. (2019). Extended genetic algorithm for solving open-shop scheduling problem. *Soft computing*, *23*, 5099-5116.

Ronconi, D. P., & Kawamura, M. S. (2010). The single machine earliness and tardiness scheduling problem: lower bounds and a branch-and-bound algorithm. *Computational & applied mathematics*, *29*, 107-124.

Salido, M. A., Escamilla, J., Giret, A., & Barber, F. (2016). A genetic algorithm for energy-efficiency in job-shop scheduling. *The International Journal of Advanced Manufacturing Technology*, *85*, 1303-1314.

Schaller, J., & Valente, J. M. (2013). A comparison of metaheuristic procedures to schedule jobs in a permutation flow shop to minimise total earliness and tardiness. *International Journal of Production Research*, *51*(3), 772-779.

Schaller, J., & Valente, J. M. (2019). Heuristics for scheduling jobs in a permutation flow shop to minimize total earliness and tardiness with unforced idle time allowed. *Expert Systems with Applications*, *119*, 376-386.

Scofield, W. (2002). Aplicação de Algoritmos Genéticos ao Problema Job-Shop. *Universidade Federal de Ouro Preto. MG. Brasil*.

Silva, Y. L. T., Subramanian, A., & Pessoa, A. A. (2018). Exact and heuristic algorithms for order acceptance and scheduling with sequence-dependent setup times. *Computers & operations research*, *90*, 142-160.

Siqueira, F. R. D., & Müller, C. A. D. S. (2022). Integração entre Teoria dos Stakeholders e Visão Baseada em Recursos: trajetória percorrida pela literatura de Administração.

Tanaka, S., Fujikuma, S., & Araki, M. (2009). An exact algorithm for single-machine scheduling without machine idle time. *Journal of Scheduling*, *12*, 575-593.

Tavakkoli-Moghaddam, R., Azarkish, M., & Sadeghnejad-Barkousaraie, A. (2011). A new hybrid multi-objective Pareto archive PSO algorithm for a bi-objective job shop scheduling problem. *Expert Systems with Applications*, *38*(9), 10812-10821.

Wang, Y. (2012). A new hybrid genetic algorithm for job shop scheduling problem. *Computers & Operations Research*, *39*(10), 2291-2299.

Yazdani, M., Aleti, A., Khalili, S. M., & Jolai, F. (2017). Optimizing the sum of maximum earliness and tardiness of the job shop scheduling problem. *Computers & Industrial Engineering*, *107*, 12-24.

Yuce, B., Fruggiero, F., Packianather, M. S., Pham, D. T., Mastrocinque, E., Lambiase, A., & Fera, M. (2017). Hybrid Genetic Bees Algorithm applied to single machine scheduling with earliness and tardiness penalties. *Computers & Industrial Engineering*, *113*, 842-858.

Zambrano Rey, G., Bekrar, A., Trentesaux, D., & Zhou, B. H. (2015). Solving the flexible job-shop just-in-time scheduling problem with quadratic earliness and tardiness costs. *The International Journal of Advanced Manufacturing Technology*, *81*, 1871-1891.