Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

Bio-inspired multi-objective algorithms applied on production scheduling problems

Beatriz Flamia Azevedo[a,b,c*], Rubén Montaño-Vega[d], M. Leonilde R. Varela[c] and Ana I. Pereira[a,b,c]

[a]*Research Centre in Digitalization and Intelligent Robotics (CeDRI), Instituto Politécnico de Bragança, Bragança, 5300-253, Portugal*
[b]*Laboratório para a Sustentabilidade e Tecnologia em Regiões de Montanha, Instituto Politecnico de Bragança, Bragança, 300-253, Portugal*
[c]*Algoritmi Research Centre/LASI, University of Minho, Department of Production and Systems, Guimarães, 4804-533, Portugal*
[d]*Department of Mechanical Engineering and Industrial Design, Universidad de Cadiz, Cadiz, 11003, Spain*

C H R O N I C L E

Article history:
Received November 10 2022
Accepted December 16 2022
Available online
December, 16 2022

Keywords:
Bio-inspired algorithms
Metaheuristic
Production scheduling
Decision support
Multi-objective
Clustering algorithm

A B S T R A C T

Production scheduling is a crucial task in the manufacturing process. In this way, the managers must decide the job's production schedule. However, this task is not simple, often requiring complex software tools and specialized algorithms to find the optimal solution. In this work, a multi-objective optimization model was developed to explore production scheduling performance measures to help managers in decision-making related to job attribution under three simulations of parallel machine scenarios. Five important production scheduling performance measures were considered (makespan, tardiness and earliness times, number of tardy and early jobs), and combined into three objective functions. To solve the scheduling problem, three multi-objective evolutionary algorithms are considered (Multi-objective Particle Swarm Optimization, Multi-objective Grey Wolf Algorithm, and Non-dominated Sorting Genetic Algorithm II), and the set of optimum solutions named Pareto Front, provided by each one is compared in terms of dominance, generating a new Pareto Front, denoted as Final Pareto Front. Furthermore, this Final Pareto Front is analyzed through an automatic bio-inspired clustering algorithm based on the Genetic Algorithm. The results demonstrated that the proposed approach efficiently solves the scheduling problem considered. In addition, the proposed methodology provided more robust solutions by combining different bio-inspired multi-objective techniques. Furthermore, the cluster analysis proved fundamental for a better understanding of the results and support for choosing the final optimum solution.

1. Introduction

In today's globalized market, there is a significant increase in production scheduling problems due to the dynamically changing conditions occurring in manufacturing environments and order processing, which requires more complex scheduling systems by combining different kinds of approaches and solving algorithms. In this sense, the so-called Industry 4.0 has emerged (Kagermann et al., 2011) with new technological advances and new challenges within the manufacturing management context (Fu et al., 2018). Thus, production systems are becoming more flexible, varied, personalized, and dynamic to enable higher-quality solutions for managers. The new developments in the Artificial Intelligence domain have repercussions in scheduling research, where the focus is shifting from a traditional, centralized, and static approach to intelligent, predictive, dynamic, and distributed approaches (Zhang et al., 2019), among others, whereby many assumptions of the classic scheduling approaches are no longer competitive (Zhang et al., 2021; Varela et al., 2022).

* Corresponding author
E-mail: beatrizflamia@ipb.pt (B. F. Azevedo)
ISSN 1923-2934 (Online) - ISSN 1923-2926 (Print)
2023 Growing Science Ltd.
doi: 10.5267/j.ijiec.2022.12.001

In this context, one of the most critical challenges of scheduling research is responding dynamically and in real time to the events in a production system (Varela et al., 2022). Specifically, it is necessary to respond effectively to unforeseen events, machine stoppages, the arrival of new orders, lack of material, among others. Thereby, the technological advances of Industry 4.0 include big data analytics, cyber-physical systems, Radio-Frequency IDentification (RFID) (Barenji et al., 2014; Borangiu et al., 2020), and other technology, allowing the obtention of real-time data able to constantly describe what is happening in the manufacturing processes (Varela et al., 2019). Thus, practical and dynamic schedules can be achieved if these data are managed correctly.

The focus of production scheduling optimization is shifting from mass production to mass customization. Thus, the following are becoming key objectives within production planning and scheduling: shortening production order times, increasing the reliability of delivery dates, minimizing stocks, high flexibility, among others. So, in real industrial environments, it is necessary to deal with more than one objective simultaneously. It can be achieved mainly by following multi-objective production optimization approaches (Ojstersek et al., 2020; Zheng et al., 2022). However, despite being the most appropriate method, multi-objective production scheduling is less explored in the literature than the single-objective method due to its complexity (Chen et al., 2022). In this way, the present study proposes a robust and adaptive model to explore the parallel machine problem, considering multi-objective optimization. In this case, bio-inspired algorithms are one of the most widely used tools for solving these multi-objective models (Chaouch et al., 2017; Lu et al., 2016; Jia et al., 2013; Azevedo et al., 2022; Chen et al., 2022), on which this study will be focused.

This paper consists of studying the behaviour of three multi-objective bio-inspired algorithms in a production scheduling problem that are: Multi-objective Particle Swarm Optimization (MOPSO), Multi-objective Grey Wolf Algorithm (MOGWA), and Non-dominated Sorting Genetic Algorithm II (NSGA-II). Each algorithm's results are compared, and the non-dominated solutions will compose a final Pareto Front. This final Pareto Front will be more comprehensive and robust since the information of different bio-inspired algorithms generates it. Besides, an automatic bio-inspired clustering algorithm based on a Genetic Algorithm (GA) is utilized to assist in understanding results and decision support. For this, five important production scheduling performance measures (makespan, tardiness and earliness time, number of tardy and early jobs) were combined into three objective functions. The results aim to help managers make job distribution decisions in a parallel machine environment. Thence, the main contributions of this paper are:

(i) Develop a multi-objective optimization model to solve the parallel machine production scheduling problems.
(ii) Analyse the performance of the different multi-objective evolutionary algorithms in the same problem.
(iii) Propose a robust methodology to strengthen results by the collaboration of different meta-heuristic approaches.
(iv) Provide more flexibility for the decision-maker to customize the production decisions according to established priorities.
(v) Evaluate the results through machine learning techniques (clustering) and support the choice of the most appropriate Pareto Front solution.

This paper is organized as follows. After the introduction, Section 2 presents the literature review of related work, applied evolutionary multi-objective optimization strategies to deal with production scheduling problems. Thereafter, in Section 3 the methodology proposed is described; it is based on a multi-objective approach and corresponding algorithms, as well as on a clustering algorithm utilized to evaluate the final solution. The production scheduling performance measures considered, and the mathematical model developed are presented in Section 4. Section 5 presents and discusses the main results of the proposed approach. Finally, Section 6, concludes the paper and identifies future paths for further research on this subject.

2. State-of-Art Review

Scheduling problems have been extensively researched during the last decades. Thus, various strategies and innovations have been developed in this area. When dealing with a real industrial scenario, it is usually important to deal with multiple objectives at the same time, so the multi-objective approach is the most suitable methodology. Although, multi-objective optimization of parallel machine scheduling is rare in the literature (Chen et al., 2022). This section reviews current research focusing on resolving multi-objective production scheduling problems, especially using bio-inspired algorithms. A multi-objective approach to schedule the operation of an aerospace shell production line is developed by (Wang et al., 2020). In this case, the makespan and the production cost of the line are optimized at the same time, regarding the energy and economic efficiency of the manufacturing processes. In order to overcome this problem, a knowledge-based evolutionary multi-objective algorithm (KD-MOEA) is formulated in which the essential properties of the scheduling plan are extracted and applied to manage the subsequent optimization process. Thereby, the KD-MOEA includes two fundamental operations: knowledge extraction and knowledge use. So, the data contained in the population is sent to the knowledge extraction process; after that, the data is transformed into structural knowledge, which is used to guide the subsequent process optimization. In this way, the obtained knowledge generates new individuals for the population. The proposed method was compared with other algorithms, (i) the Non-dominated Sorting Genetic Algorithm II (NSGA-II), which is considered a classical multi-objective algorithm based on Pareto domination relationship (Deb et al., 2002); and (ii) a multi-objective evolutionary algorithm based on decomposition (MOEA/D), which decomposes a multi-objective optimization problem into a number of scalar optimization subproblems and optimizes them simultaneously (Zhang and Li, 2007). The presented approach showed a better result than the two other methods in terms of hypervolume index and a t-test performed. The KD-MOEA outperformed the MOEA/D on six instances tested (out of six) and presented a significantly better performance in four (out of six) than NSGA-II. However, the statistical values are not significantly different in the other two instances.

In turn, Sheikhalishahi et al. (2019) suggested an open shop floor scheduling model, considering the human error factor and preventive maintenance. In this mathematical model, competing objective functions are considered simultaneously: production time, reducing human error, and optimizing machine availability. To obtain the optimal solution, three meta-heuristic methods, NSGA-II, MOPSO, and strength Pareto Evolutionary Algorithm II (SPEA-II), are used; and the Taguchi method was utilized to determine the parameters of each algorithm considered. The comparison between the algorithms demonstrated that NSGA-II presented more diversified solutions, indicating a better exploration capacity than the other two algorithms. Since NSGA-II could find more solutions, it requires more computation time but in an acceptable way.

Chen et al. (2022) investigated parallel machine scheduling problems with flexible maintenance and job release times to minimize the makespan and the total tardiness time. In this approach, a modified NSGA-II was developed for medium and large-scale instances, considering: (i) a decoding method based on dynamic programming; (ii) a dynamic probability for the crossover and mutation operators and (iii) the incorporation of neighbourhood search method. Besides, the algorithm's parameters definition was defined using the Taguchi technique. So, the proposed algorithm was compared with the traditional NSGA-II and mixed-integer programming approach in terms of operational efficiency and effectiveness, demonstrating the superior performance of the proposed algorithm when solving medium and large-scale instances. A similar occurrence is presented by Sheikhalishahi et al. (2019); again, the algorithm requires more time to solve the instances than the other methods but in an acceptable way.

According to Dai et al. (2019), companies face enormous environmental challenges due to energy consumption and related environmental impacts. In this regard, they see production scheduling improvement as an efficient strategy that can significantly impact energy saving in a manufacturing system. Thus, Dai et al. (2019) proposed a multi-objective optimization model for minimizing energy consumption and makespan for a flexible job shop scheduling problem with transport restrictions. This problem is solved through an enhanced Genetic Algorithm that shows efficient results. On the other hand, Piroozfard et al. (2018) focus their efforts on considering environmental-based objective functions. So, they present a multi-objective flexible job shop scheduling problem to reduce the total carbon footprint and overall job tardiness criterion using an enhanced Multi-objective Genetic Algorithm. The computational results of their proposed algorithm were compared with two representative multi-objective evolutionary algorithms: NSGA-II and SPEA-II. The results show a satisfactory performance; carbon footprint has an important impact on the optimum solutions.

Gong et al. (2018) presented a multi-objective Artificial Bee Colony algorithm (MOABC) for blocking the lot-streaming flow shop scheduling problem. A blocking lot-streaming flow shop scheduling problem is to schedule a number of jobs on more than one machine, where each job is split into a number of sublots, while no intermediate buffers exist between adjacent machines. In this case, the MOABC was modified to solve the described problem, in which the main objective functions are the makespan and earliness time. So, an initialization strategy is proposed by incorporating two effective heuristics; thereafter, three effective updating methods are presented for the three kinds of bees, in which employed bees modify solutions using the information provided by non-dominated solutions, a Pareto local search is applied to improve the solutions selected by onlooker bees, and scout bees employ a destruction and construction operator to generate promising neighbouring solutions. Finally, a new mechanism is given to preserve elitists during evolution. The algorithm proposed was evaluated and compared with four other algorithms (TA, INSGA, NGA, and BBEDA) on eighteen test subsets, outperforming them in terms of convergence and diversity of non-dominated solutions.

Wang et al. (2019) exposed an enhanced Particle Swarm Optimization Algorithm for dynamic job shop scheduling problems with arbitrary arrival of jobs. It considers three objective functions: minimizing makespan, the disruption rate of a new job while processing, and the sequence deviation on machines. In turn, Qin et al., (2019) and Luo et al., (2019), use the Grey Wolf algorithm to solve two types of scheduling problems, showing successful results. The first one (Qin et al., 2019) solves a so-called multi-objective, multi-constraint casting production scheduling problem. This casting problem is usually managed as an independent scheduling problem because it presents specific factors, such as the limitation of the starting time in some casting operations and the transportation between two consecutive operations. Specifically, the objective functions used are the minimization of the makespan, the total product cost, and the delivery delay time. A Tabu Search Algorithm is embedded in the Multi-objective Grey Wolf Algorithm (MOGWA) to avoid premature convergence of solutions. In the next one (Luo et al., 2019), the MOGWA is employed to solve energy-efficient scheduling for multi-objective job shops with variable processing speeds. As mentioned by the authors, green manufacturing has become a fundamental theme, and energy-efficient scheduling is shown to be a significant way to achieve it. Thus, the objective function chosen was minimizing the makespan and total energy consumption. In both articles, promising results are obtained with the proposed multi-objective Grey Wolf algorithms, indicating that they are significantly better than comparable algorithms. Especially in (Luo et al., 2019), it is pointed out that this model allows for solving large-scale problems, making them more suitable for more difficult real-world scenarios.

Another interesting approach is described by (Safarzadeh & Niaki, 2023), which studied in a multi-objective way, parallel machine scheduling problems with the machine processing cost such as depreciation, energy consumption, carbon emission, and raw materials. So, the objective function aims to minimize the makespan and the total cost simultaneously. To solve the proposed model, the makespan was transferred to the set of constraints by the ϵ-constraint method, which could be solved sequentially by a mixed-integer linear programming strategy. The procedure resulted in an efficient way of estimating the Pareto front, providing a diverse and uniformly distributed set of Pareto solutions. From the literature review, it is possible to conclude that the application of multi-objective strategies has been increasing recently. The research in this area is getting less limited and reduced regarding the single criterion problems. This happens because, in a dynamic industrial environment, several variables (objectives) should be considered together to achieve satisfying results and properly support decision-making. Thus, in this paper, the multi-objective approach developed combines five important production scheduling performance measures (makespan, tardiness and earliness time, number of tardy and early jobs) into three objective functions. The optimal solution is provided by three bio-inspired multi-objective algorithms that are evaluated together in a single Pareto Front. This collaborative methodology between the algorithms is the main differentiation between this work and the previous presented. In this way, it is intended to achieve more robust and strengthened solutions. Moreover, to support the understanding and the selection of the most suitable solutions, the Pareto Front generated by the combination of the three multi-objective algorithms is analysed by an automatic bio-inspired clustering algorithm.

3. Methodology

The approach presented in this work comprises two main processes: multi-objective optimization and machine learning. In the multi-optimization process, the input data is analysed by three different multi-objective evolutionary algorithms (MOEAs) that provide a set of dominated and non-dominated solutions. After that, each algorithm's non-dominated set of solutions are joined in a new data set, and these solutions are compared in terms of dominance. Thus, a new set of non-dominated solutions and, consequently, the final Pareto Front is obtained. Hence, this Pareto Front integrates the answer of different MOEAs strategies. Finally, an automatic bio-inspired clustering algorithm analyses the data composing the Pareto Front to understand the results better. Fig. 1 illustrates this proposed methodology with the underlying processes that are detailed in the following sections.

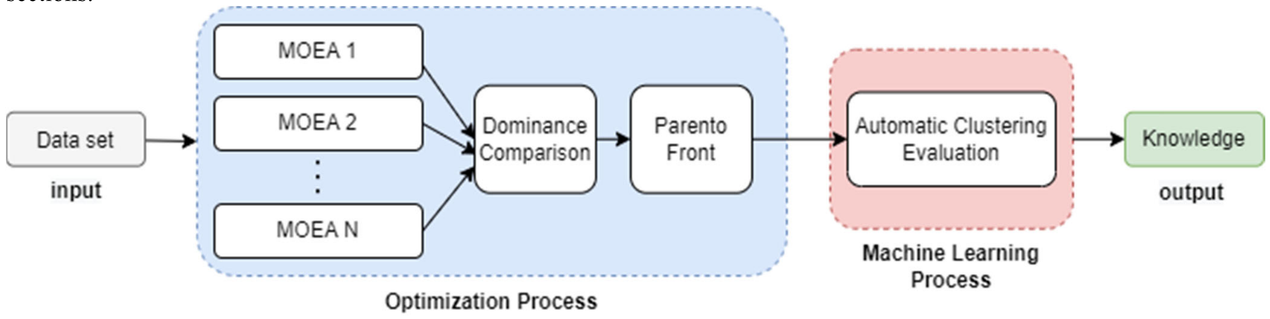


Fig. 1. Results obtaining process.

o *Multi-objective Approach*

To solve a mathematical optimization problem with several objective functions, it is necessary to apply the so-called multi-objective approach, where a decision must be made with multiple criteria simultaneously. Usually, the objectives are conflicting, and multi-objective techniques can offer various possible solutions, representing the trade-offs across the objective functions (Deb, 2011; Miettinen, 1998). In multi-objective optimization, two concepts are essential to understand: dominated and non-dominated solutions. For a better explanation of these concepts, a generic multi-objective optimization problem must first be defined (Miettinen, 1998) as shown in Eq. (1),

$$\min_{x \in S}\{f_1(x), f_2(x), \dots, f_k(x)\}, \tag{1}$$

containing k($\geq$ 2) competing objective functions f_i: $R^n \rightarrow$ R to be minimized at a time. Its decision variable is in the form of vector $x = (x_1, x_2, \dots, x_n)$ belonging to the nonempty feasible region S $\subset R^n$. Using this generic formulation of the problem, the types of constraints that form the feasible region are not fixed. Objective vectors are formed by values of the objective function F(x) $= (f_1(x), f_2(x), \dots, f_k(x))$. In addition, the image of the feasible region within the objective space is denoted as the feasible objective region Z = F (S), (Miettinen, 1998). Optimal solutions following a multi-objective approach will be specified based on a mathematical concept of partial ordering (Deb, 2011). The possible solutions to a multi-objective optimization problem are denoted as the non-dominated solutions, which represent the so-called *Pareto Front*. Specifically, a decision variable vector $x' \in$ S is called a dominated solution if there exist $x \in$ S such that $f_i(x) \leq f_i(x')$ for all $i = 1, \dots, k$. Then, the vectors of the objective function are treated as optimal if no one of its elements can be enhanced without deteriorating at least one of the other elements. For a specific set of solutions, or equivalent points on the feasible objective region Z, a par-wise comparison can be made considering the concepts previously mentioned and whether the domination criteria can be set. Any point that is not dominated by any other set member is known as non-dominated point (Deb, 2011). By contrast, the set of points that do not belong to the non-dominated set is considered a dominated solution. **Fig. 2** illustrates the concepts mentioned and points a and b are an example of non-dominated solutions.

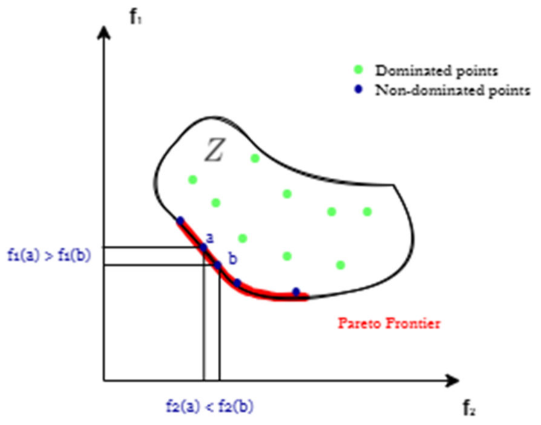


Fig. 2. Pareto Front illustration. Adapted from Miettinen (1998).

Once the points that compose a Pareto Front cannot be ordered completely, all points that belong to Pareto Front can be mathematically considered equally optimal solutions. This particular and intriguing property makes necessary the figure of a decision-maker. A decision maker is a person, or a group of persons collaborating in the decision-making process, who is/are supposed to have a better insight into the problem and who can express preference relation between different solutions, responsible for deciding the most appropriate final solution for the problem (Deb, 2011; Miettinen, 1998).

o *Multi-objective Evolutionary Algorithms*

The algorithm in the evolutionary computation class begins by randomly generating a set of potential solutions (population). The population is represented by individuals arranged in the search space, which is the space where each variable can assume values. The search space is delimited by the domain of the objective function, which ensures that all individuals are admitted solutions for the problem (Sivanandam & Deepa, 2008). By iteratively applying genetic operators like selection, crossover, and mutation (the most common ones), the population is being modified to obtain new feasible solutions. This process stochastically discards poor solutions and evolves more fit (better) solutions (Bansal et al., 2019). Due to the very nature of these operators, which are based on the Darwin's evolution principles (in which the most adapted individuals of a given population survive whereas the less adapted die to be replaced by their offspring (Bansal et al., 2019; Sivanandam & Deepa, 2008)), it is expected that the evolved solutions will become better generation by generation (iteration). Like any iterative process, the evolutionary algorithms require a stopping criterion to interrupt the search and define the optimum solution (Sivanandam & Deepa, 2008). Some examples of stopping criteria are described in (Azevedo, 2020). In this paper, three evolutionary multi-objective algorithms will be used. These algorithms are described below.

▪ *Multi-objective Particle Swarm Optimization*

Multi-objective Particle Swarm Optimization (MOPSO) is a multi-objective variant of Particle Swarm Optimization proposed by (Coello-Coello & Lechuga, 2002). MOPSO integrates the Pareto envelope and the grid-making techniques to deal with multi-objective optimization problems. As in PSO, the particles in MOPSO share information and move towards the best global particles and their local memory. Nevertheless, contrary to PSO, there is more than one criterion to identify and set the best global or local particle. The set of non-dominated particles in the swarm is gathered into a sub-swarm named repository. Each particle selects its global best objective from the individuals in this repository. Domination-based and probabilistic rules are employed for personal (local) best particles. The particles employ this repository to determine a leader for guiding their search. With this approach, a mechanism was applied such that each particle could select a specific leader according to the generation of hypercubes created by dividing the explored search space.

Basically, the MOPSO algorithm initializes the population particles; after that, it initializes the speed of each particle; therefore, every particle in the population is evaluated using the objective function, and those particles that constitute non-dominant vectors are saved in the repository. Done that, it is necessary to produce hypercubes of the search space considered before and place the particles employing these hypercubes as a coordinate system in which the values of their objective functions set the coordinates of each particle. Following, it initializes the memory of each particle; this memory is also stored in the repository and is used as a guide to navigate through the search space. So, while a stoppage criterion has not been reached, six iterative steps undergo (Coello-Coello & Lechuga, 2002):

Step 1: each particle's speed is calculated by means of the Eq. (2),

$$VEL_i = W \times VEL_i + R_1 \times (PBESTS_i - POP_i) + R_2 \times (REP_h - POP_i) \quad (2)$$

being W the inertia weight, R_1 and R_2 are random numbers within the range $[0,1]$; $PBESTS_i$ is the best position that the particle i has occupied; REP_h is a value extracted from the repository.

Step 2: Compute new particle positions by adding the velocity generated in the prior step, as indicated in Eq. (3).

$$POP_i = POP_i + VEL_i \quad (3)$$

Step 3: Retain particles in the search space if they exceed its boundaries.

Step 4: Analyse each of the particles in POP.

Step 5: Update of REP contents along with the geographic representation of particles in the hypercubes.

Step 6: Update the particle's position, as indicated by Equation (4), if the particle's current position is better than the position included in the particle's memory.

$$PBESTS_i = POP_i \quad (4)$$

To decide which memory position should be retained, the argument is simply to apply Pareto dominance (i.e., if the current position is dominated by the position in memory, the position in memory is maintained; otherwise, the current position substitutes the one in memory; if none of them is dominated by the other, one of them is selected arbitrarily) (Coello-Coello & Lechuga, 2002). This process is iterative and stops when a stopping criterion is achieved. More details about the MOPSO can be found in (Coello-Coello & Lechuga, 2002; Coello-Coello et al., 2004), and the algorithm code is available in (Yapiz, 2022b).

- *Multi-objective Grey Wolf Algorithm*

The Multi-objective Grey Wolf Algorithm (MOGWA) is an optimization algorithm inspired by the Grey Wolf hunting mechanism and its social hierarchy. The hunting skill of wolves is recognized in the animal world, placing them at the top of the food chain (Mirjalili et al. 2014, 2016). In order to simulate the behaviour of wolves in mathematical terms, four types of wolves and their leadership hierarchy are considered. The *Alpha* wolves - α are the leaders or the dominant wolf, followed by the second-level wolves in the hierarchy, the *Beta* wolves - β. Beta wolves are subordinated wolves and assist the Alpha in decision-making or other herd activities. After that, there are the *Delta* wolves - δ, which are submitted to Alphas and Betas, but they dominate the *Omega* wolves - ω. The Omega is on the lowest level of the grey wolf hierarchy, playing a role of scapegoat, but they are submitted to all the other dominant wolves (Mirjalili et al., 2014). To model this social hierarchy, the algorithm considers the fittest solution as the α wolves, and the second and third best solutions are the β and δ, respectively. Thus, the rest of the candidate solutions are the ω. In this way, the hunting (optimization) is guided by α, β, and δ. And the ω wolves follow these three wolves (Mirjalili et al., 2014, 2016). Thence, three main hunting steps are implemented to guide the optimization process (Mirjalili et al., 2014, 2016): searching, encircling, and attacking prey. Encircling prey: wolves' common behaviour when hunting is to encircle the prey. Thus, to translate this process in the algorithm the following equations (Equations (5) and (6)) are proposed (Mirjalili et al., 2014).

$$D = | C \cdot X_p(t) - X(t) |, \text{ in which } C = 2 \cdot r_2 \quad (5)$$

$$X(t+1) = X_p(t) - A \cdot D, \text{ in which } A = 2a \cdot r_1 - a \quad (6)$$

where t is the current iteration, A and C are coefficients vectors, X_p is the position vector of the prey, and X indicates the position vector of a grey wolf. And a is linearly decreased from 2 to 0 over the course of iterations and r_1, r_2 are random vectors in [0, 1].

Hunting: the hunting process is guided by the α wolves, but β and δ might also participate occasionally. However, in the stochastic algorithm, initially, it is not possible to have an idea about the location of the prey; it is the optimum solution. Thus, it is supposed that the α (best candidate solution), β, and δ have better knowledge about the potential location of prey. Therefore, the information about the position of the first three best solutions obtained so far is used to update the other wolf's position. Besides, it can be observed that the final position would be in a random place within a circle, which is defined by the positions of α, β, and δ in the search space. The process is represented by the following equations (Eq. (7), Eq. (8), and Eq. (9)) (Mirjalili et al., 2014).

$$D_\alpha = | C_1 \cdot X_\alpha - X |,\ D_\beta = | C_2 \cdot X_\beta - X |,\ D_\delta = | C_3 \cdot X_\delta - X | \quad (7)$$

$$X_1 = X_\alpha - A_1 \cdot (D_\alpha),\ X_2 = X_\beta - A_2 \cdot \left(D_\beta\right), X_3 = X_\delta - A_3 \cdot (D_\delta) \quad (8)$$

$$X(t+1) = \frac{X_1 + X_2 + X_3}{3} \quad (9)$$

At the end of the hunting process, the prey is attacked, and the wolves stop moving.

Attacking prey (Exploitation): the process of attacking a prey is controlled by a value. Thus, in the model, approaching the prey corresponds to a decrease in the value of a. Note that the fluctuation range of A also provokes a decrement in a. And, when $|A| > 1$ forces the wolves to attack towards the prey (Mirjalili et al., 2014).

Search for prey (Exploration): the grey wolves, mainly the α, β, and δ, diverge their position to search for prey and converge it to attack. Hence, to oblige the search agent to diverge from the prey, it is utilized A with random values greater than 1 or less than -1. Note that $|A| < 1$ forces the wolves to diverge form the prey to find a fitter prey, hopefully. Another component that favours the exploration process is $C \in [0,2]$. This component provides random weight for prey to stochastically emphasize ($C > 1$) or deemphasize ($C < 1$) the effect of prey in defining distance in Equation (5). More details about the MOGWA can be found in (Mirjalili et al. (2016, 2014)) and the algorithm code is available on (Mirjalili, 2022).

- *Multi-objective Particle Swarm Optimization II*

The Non-dominated Sorting Genetic Algorithm II (NSGA-II) is a popular bio-inspired algorithm for solving multi-objective optimization problems. It was developed by (Deb et al., 2002) and is based on Genetic Algorithm (GA) evolutionary procedures.

The NSGA-II uses a fast non-domination sorting procedure to classify solutions according to the level of non-domination and a crowding distance operator to preserve diversity in the evolutionary procedure. Moreover, elitism is achieved through the control of the elite members of the population as the algorithm progresses to keep the diversity of the population until converging to a Pareto-optimal front (Deb et al., 2002).

Basically, this algorithm starts by initializing randomly the population POP_i, $i \in \{0, \dots, n\}$, in which each objective j, $j \in \{1, \dots, k\}$ is valued consequently. The population is composed of N individuals represented by a formed candidate solution, x of $\{f_1, f_2, \dots, f_k\}$, the overall fitness value, F, and neighborhood diversity value, D (Kok et al., 2011). Afterward, POP_i is faced with genetic operations which entail a simulated binary crossover and a polynomial mutation. A generation cycle is completed with respect for each individual POP_i, which is a diversity value according to the crowding distance it constitutes with respect to the adjacent chromosomes on its non-dominated front rank. The NSGA-II algorithm is applied in an iterative way until a specified stopping criterion is met (Kok et al., 2011). More details about the NSGA-II can be found in (Deb et al., 2002; Kok et al., 2011) and the algorithm code is available on Matlab®2019a, specifically by *gamultiobj function* (MATLAB, 2019).

○ *Automatic Clustering Algorithms*

Clustering is one of the most widely used methods for unsupervised learning. It is very useful in any area involving data mining of unlabelled datasets, i.e., datasets with no defined association between input and output. Clustering algorithms consist of performing the task of grouping a set of elements with similarities in the same group and dissimilarities in other groups (Shalev-Shwartz & Ben-David, 2014).

Usually, the traditional clustering algorithms require the indication of the K values, which represents the number of dataset partitioning. However, the estimation of this value is not an easy task. Besides, an unappropriated selection of the number of clusters results in poor performance since, in traditional clustering algorithms, the results often depend on the initial starting points (Ezugwu et al., 2020). In this context, to support the K estimation, automatic bio-inspired clustering techniques have been hardly explored (Qaddoura et al., 2021; Singh, 2021; Chen et al., 2020). So, the automatic clustering process consists of solving an optimization problem, aiming to minimize the similarity within a cluster and maximize the dissimilarity between the clusters.

In this work, the Davies–Bouldin index (DB) (Arbelaitz et al., 2013) is used as a clustering similarity and dissimilarity measure that will define the number of cluster centroids, which is the number of groups into which the dataset will be divided.

DB index is based on a ratio of intra-cluster and inter-cluster distances. It is used to validate cluster quality and to determine the optimal number of clusters. Let us define a dataset $X = \{x_1, x_2, \ldots, x_n\}$, as a set of n elements. Thus, a clustering in X is a cluster that partitions X into K groups: $C = \{c_1, c_2, \ldots, c_k\}$, where $\cup_{c_k \in C} C_k = X, c_k \cap c_l \neq \emptyset, \forall k \neq 1$.

To evaluate the DB index, it is necessary to evaluate the intra-cluster distance, in this case, represented by the average distance between each observation within the cluster and its centroid, which is a dispersion parameter $S(c_k)$, given by Equation (10),

$$S(c_k) = \frac{1}{|\, c_k \,|} \sum_{x_i \in c_k} D(x_i, c_k) \tag{10}$$

in which $D(x_i, c_k)$ is the Euclidean distance between an element x_i and its centroid c_k, Equation (11).

$$D(c_k, c_l) = \| c_k - c_l \|_2 \tag{11}$$

Thus, the DB index is given by Equation (12), where $D(c_k, c_l)$is the Euclidean distance between the centroid c_k and the centroid c_l.

$$DB(C) = \frac{1}{K_i} \sum_{c_k \in C} max\, \{\frac{S(c_k) + S(c_l)}{D(c_k, c_l)}\} \tag{12}$$

Considering the definition of the DB index, a minimization problem can be defined whose objective function is the DB index value. Thus, meta-heuristics can be used to solve this problem, as an evolutionary bio-inspired algorithm. In this case, the meta-heuristic chosen was a Genetic Algorithm, as it is one of the most popular and efficient methods for the purpose. More information about these algorithms can be found at (Kennedy & Eberhart, 1995; Sivanandam & Deepa, 2008), and the automatic clustering code can be found on Yapiz (2022a).

4. Performance Measures and Mathematical Modelling

This section presents the production scheduling performance measures considered in this work and the mathematical multi-objective model developed to solve the parallel machine production scheduling problems.

o *Production Scheduling Performance Measures*

Establishing performance measures is essential to quantify the efficiency and effectiveness of decision actions. According to Varela and Ribeiro (2014) and Santos et al. (2015) performance measures evaluate the scheduling program's efficiency, aiming at a final objective, which may be the maximization or minimization of the criteria used as performance measures. Thus, the optimization criteria allow to achieve several goals (Varela & Ribeiro, 2014; Santos et al., 2015), namely: maximization of production flow, the satisfaction of quality requirements and quick response to customers, minimization of production costs or the combination of all previous cases mentioned. The standard performance measures in the objective function of a scheduling problem are presented below; however other measures can be consulted on (Varela and Ribeiro, 2014; Santos et al., 2015; Pinedo, 2012; Reis, 2020):

Makespan: it is designed by C_{max}, and it defines the time when the last job of a sequence of jobs is complete, where C_j represents the completion date of the job j for $j \in \{1, \ldots, NJ\}$, where NJ are the number of jobs to be performed (Eq. (13)). For jobs consisting of more than one operation, represents the moment when the last one is completed.

$$C_{max} = \max_{j \in \{1,\ldots,NJ\}} (C_j) \tag{13}$$

Tardiness and earliness time: the tardiness concerns the difference between the conclusion date of the job C_j and its the due time d_j. If this difference is positive, T_j, indicates a tardiness in conclusion (Eq. (14)), but if the difference is negative, E_j, means an anticipation or earliness of the conclusion period (Eq. (15)).

$$T_j = C_j - d_j, \text{ if } C_j > d_j \text{ for } j = 1, \ldots, NJ; \tag{14}$$

$$E_j = C_j - d_j, \text{ if } C_j < d_j \text{ for } j = 1, \ldots, NJ. \tag{15}$$

Total number of tardy jobs (NT_{total}): indicates the total number of tardy jobs, NT_{total}, and it can be formulated by Eq. (16), where NJ represents the number of jobs available, and when NTj assumes the value 1 it means that the corresponding job j is late.

$$NT_{total} = \sum_{j=1}^{NJ} NT_j = \begin{cases} 1, & if \quad T_j > 0 \; for \; j = 1,\dots,NJ \\ 0, & if \quad T_j = 0 \; for \; j = 1,\dots,NJ \end{cases} \tag{16}$$

Total number of early jobs (NE_{total}): indicates the total number of early jobs, NE_{total}, and it can be formulated by Eq. (17), where NJ represents the number of jobs available, and when NEj assumes the value 1 it means that the corresponding job j is early:

$$NE_{total} = \sum_{j=1}^{NJ} NE_j = \begin{cases} 1, & if \quad E_j < 0 \; for \; j = 1,\dots,NJ \\ 0, & if \quad E_j = 0 \; for \; j = 1,\dots,NJ \end{cases} \tag{17}$$

- *Mathematical Modelling*

When multiple machines with similar functionalities can work simultaneously without affecting each other, the problem is considered a parallel machine scheduling problem (Lin & Huang, 2021). Based on the features of the employed machines, parallel machine scheduling can be further classified into identical, uniform, and unrelated parallel machine scheduling problems. The machines considered in identical parallel machine scheduling problems are homogeneous, and the processing time of a job at any machine is identical (Lin & Huang, 2021). On the other hand, the unrelated parallel machine scheduling problem aims to assign a set of jobs to a set of unrelated machines that can process the jobs in parallel without affecting or being related to each other regarding their processing times (Lin & Huang, 2021), thus having each job different and unrelated processing times. This model is addressed to simulate an industrial environment composed of m unrelated parallel machines that must execute NJ jobs. So, the five previously mentioned performance measures are combined into three objective functions, which are presented as follows. Nevertheless, some parameters are necessary to define, before to present the objective functions.

Parameters definition

- I is the index set of machines available $i \in I = \{1, \dots, m\}$;
- J is the index set of jobs, $j \in J = \{1, \dots, NJ\}$;
- C_j defines the makespan of job j;
- T_j defines the tardiness of job j;
- E_j defines the earliness of job j;
- NT_j is the number of tardy job j;
- NE_j is the number of early job j ;
- α_1 and α_2 are penalty parameters, with α_1, $\alpha_2 \in$ Z+.

Objective functions

The first objective function, denoted by Equation (18), refers to the makespan, which aims to minimize the maximum execution time of the jobs j.

$$f_1(x) = \max\{C_1, \dots, C_{NJ}\} \tag{18}$$

The second objective function, denoted by Eq. (19), aims to minimize the number of jobs concluded tardily and the jobs concluded early. Once the quantity of late jobs is more critical than quantity of early jobs, a penalty parameter, α_1, is assigned to weight more the tardiness of jobs than the earliness of jobs.

$$f_2(x) = \alpha_1 \sum_{j=1}^{NJ} NT_j + \sum_{j=1}^{NJ} NE_j \tag{19}$$

The third objective function, defined by Eq. (20), aims to minimize the tardiness and the earliness time of the jobs. The first part of the equation refers to the positive difference between the conclusion date of the job and its due time, and the second part refers to the negative difference between both measures. Similar to Eq. (19), a penalty parameter, α_2, is used to penalize more the tardiness times than the earliness times.

$$f_3(x) = \alpha_2 \sum_{j=1}^{NJ} T_j + \sum_{j=1}^{NJ} E_j \tag{20}$$

Finally, the multi-objective problem is defined as $min\{f_1, f_2, f_3\}$.

5. Results and Discussion

This work performs the optimization of unrelated parallel machine scheduling problems. Thereby, to evaluate the model developed, three scenarios are considered: in the first scenario, it is considered a set of 5 machines and 50 jobs; after that, in the second scenario, a set of 10 machines and 100 jobs, and at least a set of 20 machines and 200 jobs at the third scenario. The processing time of each job on each machine and the due time were randomly generated, considering uniform distribution between 10 and 50 (only considering integer values). The genetic operators' parameters used for the MOEAs were the ones recommended by the literature: for the MOPSO was used the initial population size equal to 100, the inertia weight is equal to 0.5, the repository size is equal to 100, the mutation rate is equal to 0.1, and the personal learning and global learning coefficient equal to 1 and 2, respectively (Coello-Coello & Lechuga, 2002; Yapiz, 2022b); for the MOGWO the initial population and the repository size were defined equal to 100, grid inflation parameter equal to 0.1, leader selection pressure parameter equal to 4 (Mirjalili et al. (2016, 2014, 2022); finally for NSGA-II, the parameters are the same defined by the *gamultiobj function* on Matlab®2019a, the population size is equal to 100, the crossover rate is equal to 0.8, and the mutation rate is equal to 0.1 (Deb et al., 2002; MATLAB, 2019a). The parameters α_1 and α_2 were arbitrary chosen, being $\alpha_1 = 5$ and $\alpha_2 = 2$. The maximum number of iterations was considered 200, 300, and 350 for scenarios one, two, and three, respectively. The maximum number of iterations is also the stopping criterion used in the MOEAs that were executed 30 times since they are stochastic algorithms. Thus, comparing the executions provides more robustness for the final solution. For each MOEA, the solution of each execution is compared between them to generate a final set of dominated and non-dominated solutions. After that, the final set of the dominated solution will generate the final Pareto Front that was analysed by a bio-inspired automatic clustering algorithm based on GA techniques. The parameters used for the clustering algorithm were defined by (Yapiz, 2022a), considering a maximum number of clusters equal to 5. So, the initial population is equal to 100, and the maximum number of iterations is equal to 250, which was also the stoppage criterion considered. For the GA were considered the rates of 0.8 for selection and 0.3 for crossover. As the MOEAs, the bio-inspired clustering algorithm is also a stochastic method; like this, it was executed 30 times, and the execution with the smallest DB index was defined as the optimal solution. All the results were obtained by an Inter(R) i5(R) CPU @1.60 GHz with 8 GB of RAM, using Matlab® 2019a software (MATLAB, 2019).

- *Results of the first simulation*

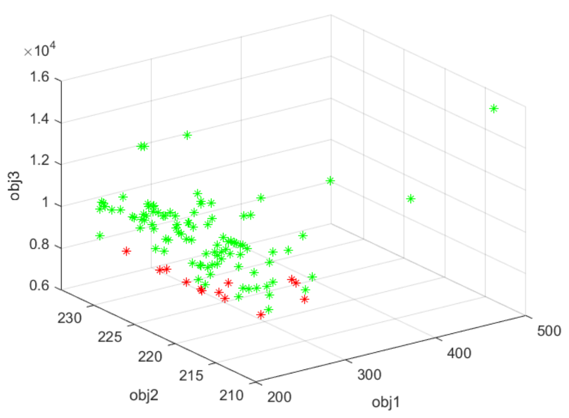
The first simulation considers an environment composed of 5 machines and 50 jobs. Table 1 presents the statistical measure of the solution generated for each algorithm in terms of Mean, Standard Deviation (SD), Minimum (Min), and Maximum (Max) values of each objective, Number of Non-dominated Solutions of the algorithm (NDS), Number of Non-dominated Solutions in the Final Pareto Front (NDSFPF), and the processing time of each execution, in seconds.

Table 1
Algorithms result of the first simulation.

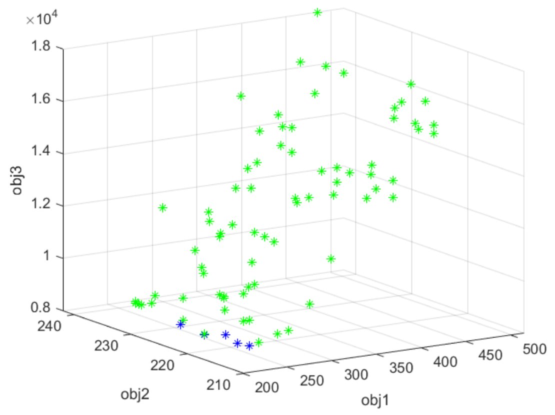
Results	MOPSO			MOGWA			NSGA-II		
	f_1	f_2	f_3	f_1	f_2	f_3	f_1	f_2	f_3
Mean	251.93	219.36	8339.50	232.10	219.20	8573.40	261.81	209.97	9129.70
SD	15.77	4.79	434.79	0.48	3.84	82.48	14.40	2.88	495.67
Min	232	213	7672	232	214	8427	232	205	8065
Max	281	230	9660	233	226	8744	360	217	12147
NDS		14			5			37	
NDSFPF		9			0			9	
Time (s)		90			157			28	

Fig. 3 presents the Pareto Front of each MOEA (Figs. 3a, 3b, and 3c) and the Final Pareto front (Figs. 3d) after the MOEAs comparison. It is important to mention that Figs. 3a, 3b, and 3c were generated by the dominated solution of the 30 executions of each algorithm. These solutions were evaluated in terms of dominance to obtain the Pareto Front of its MOEA. The green points represent the dominated solution, and the highlighted points represent the non-dominated solution of each algorithm. Thereby, Fig. 3a describes the non-dominated solution of the MOPSO, with the non-dominated solution in red; Fig. 3b describes the MOGWA's results, with the non-dominated solutions in blue; and Fig. 3c presents the NSGA-II solutions, in which the black points represent the non-dominated solutions. Finally, Fig. 3d presents the final Pareto Front generated by the dominance comparison of the non-dominated solutions of each MOEA. In this case, the non-dominated solutions of each algorithm are represented with the same colours previously described (MOPSO - red, MOGWA - blue, and NSGA-II - black). It is important to mention that the points of each Pareto Front denote solutions mathematically equals.

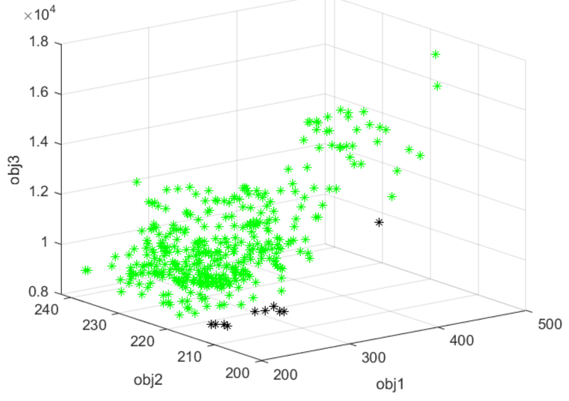
In general, by analysing the mean and standard deviation of this simulation (Table 1), it can be observed that objectives 1 and 2 have more homogeneous solutions than objective 3 since the standard deviation of objective 3 values is much higher than the other objectives. From the results of MOGWA, it is possible to observe that this algorithm has a small range in terms of Pareto Front distribution compared to the other two algorithms. MOPSO and NSGA- II present more variations between the optimal solutions; this characteristic provides the decision-maker greater flexibility in choosing the most appropriate optimal solution, which can vary according to the needs or preferences of the production manager. Regarding the number of non-dominated solutions generated by each MOEAs, the NSGA-II had the larger number; it is 37 solutions in its Pareto Front, also represented by the black points in Fig. 3c. Next is the MOPSO with 14 solutions, represented by the red points in Fig. 3a, and at last, the MOGWA with 5 solutions in its Pareto Front, which are represented by the blue points in Fig. 3b. When these non-dominated solutions (individual Pareto Fronts) are compared with each other, the solutions of MOGWA are dominated by the MOPSO and NSGA-II solutions. Hence, Fig. 3d is only composed of 9 solutions provided by MOPSO and 9 others from NSGA-II. Considering the non-dominated solutions generated individually by each MOEAs, there are a total of 56 solutions. On the other hand, the final Pareto Front, which contemplates all algorithms solutions, only has 18 solutions. This may seem like a significant decrease in the number of decision possibilities, but the remaining solutions are the most dominant ones, although in smaller numbers. Therefore, the methodology provides a refinement of the optimal solutions, which is also very important in supporting decision-making. Regarding the processing time, the NSGA-II is faster than the other two algorithms. Nevertheless, it must be taken into account that the version of the NSGA-II is an internal function of Matlab - *gamultiobj function*, making the encoding much faster than other algorithms that require external codes. However, the MOGWA is the slowest algorithm, requiring approximately 1.8 times more time than the MOPSO.



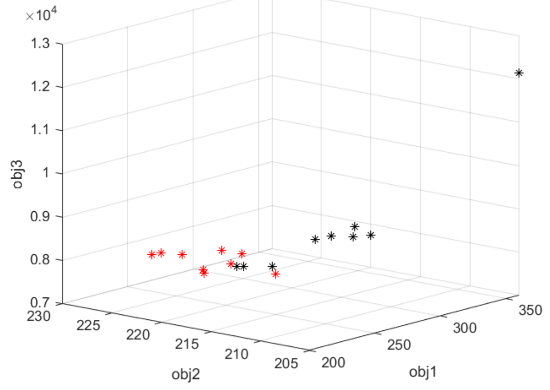
(a) Pareto Front of MOPSO



(b) Pareto Front of MOGWA



(c) Pareto Front of NSGA-II



(d) Final Pareto Front

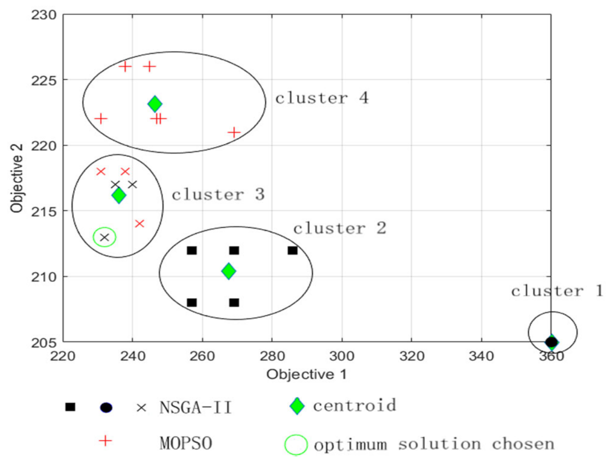
Fig. 3. Pareto Front in the first scenario

As previously mentioned, the final Pareto Front (Fig. 3d) was also analysed by an automatic clustering algorithm based on GA. The analysis was done to assist in the final decision and to analyse patterns of similarity and dissimilarity between the optimal solutions. So, the parameters presented in Section 3.3, result in the inter-cluster distance, which is the Euclidean distance between centroids, denoted by $D(c_1,c_2) = 3243.52$, $D(c_1,c_3) = 3937.29$, $D(c_1,c_4) = 4209.90$, $D(c_2,c_3) = 693.86$, $D(c_2,c_4) = 966.45$, and $D(c_3,c_4) = 273.27$. Whereas the intra-cluster distances are equal to $S_{c1} = 0$, $S_{c2} = 922.32$, $S_{c3} = 835.74$, and $S_{c4} = 920.88$. The GA cluster's centroids coordination are $c_1(f_1,f_2,f_3) = (359.97,205.00,1214.98)$, $c_2(f_1,f_2,f_3) = (267.66,210.4,8904.38)$, $c_3(f_1,f_2,f_3) = (237.00,216.42,8211,66)$, and $c_4(f_1,f_2,f_3) = (246.33,223.86,7938.66)$. Finally, about the DB index value, a value equal to 1.28 was achieved. Fig. 4

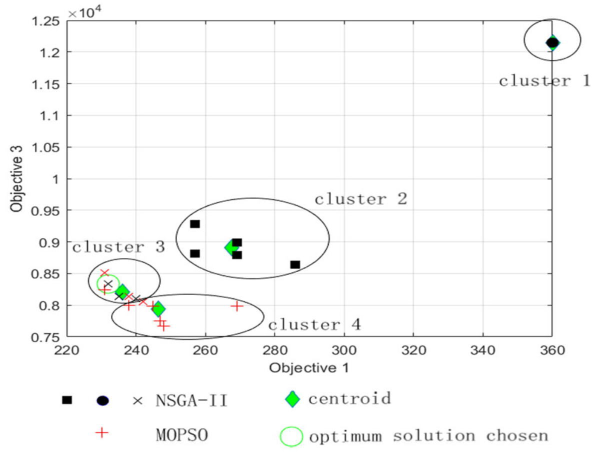
illustrates the clustering division obtained by the GA clustering algorithm, in which the red points were generated by the MOPSO algorithm, and the NSGA-II generated the black ones. Note that, in Fig. 4a, Fig. 4b, and Fig. 4c a dimensional reduction strategy was performed for better visualization and understanding of the results, while Fig. 4d shows the tri-dimensional clustering algorithm result.

This work combined internal (makespan) and external (tardiness and earliness time) production performance measures. Internal measures tend to prioritize production conditions while external measures favour customers more. Thus, the choice of a Pareto Front solution will depend on the priority that the decision-maker wants to assign, i.e., customer, production, or both. When we want to give more priority to production, the solution must minimize objective function 1. In contrast, when the priority is the customer, objectives functions 2 and 3 must be prioritized. If there is no need to give priority, the solution can be one that provides a balance between all the objectives considered, that is, a central solution on the Pareto Front. Through cluster analysis, it is possible to compare the set of solutions available according to the preference or priority that the decision-maker wants to assign to the system. In this way, the optimal solution can be dynamically modified to the industries and corresponding customers' requirements.

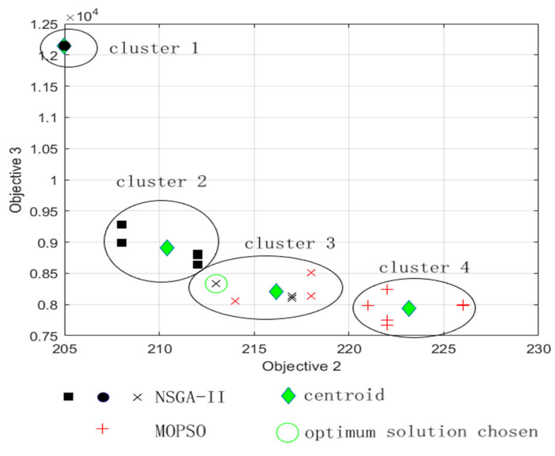
In cluster 1, there is only one solution (point); in statistical terms, the solution of cluster 1 obtained by NSGA-II can be considered an outlier. However, this solution is feasible for the problem considered, so it was chosen to keep it. Therefore, the cluster 1 solution is very useful when it is intended to prioritize objective 2 (number of tardy and early jobs) since it contains the smallest value of objective 2. Nevertheless, the same solution gives less importance to the other objectives, related to makespan (objective 1) and the earliness and tardiness of the jobs (objective 3), since this point has the highest objectives 1 and 3 values, as can be seen in Fig. 4a and Fig. 4b.



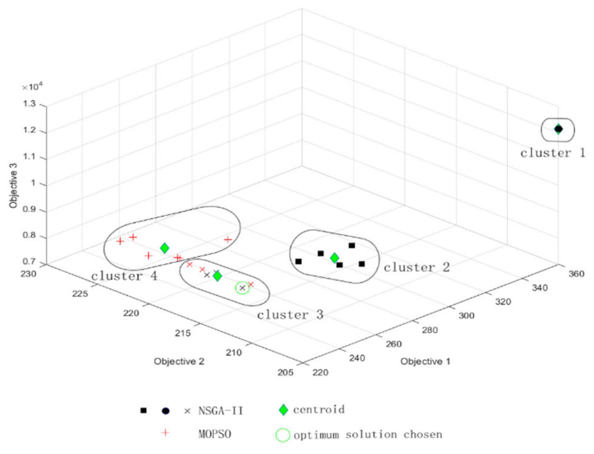
(a) Objective 1 versus 2



(b) Objective 1 versus 3



(c) Objective 2 versus 3



(d) Final Pareto Front

Fig. 4. Pareto Front in the first scenario

In cluster 2, all solutions are exclusively generated by NSGA-II, while in cluster 3, the solutions are provided by MOPSO or NSGA-II. In both clusters, the solutions presented are more balanced between the objectives than the ones of clusters 1 and 4 since clusters 2 and 3 are located more at the centre of the final Pareto Front domain. Comparing both (clusters 2 and 3), the solutions presented in cluster 2 tend to prioritize objective 2 more than the solutions in cluster 3. As can be seen, in Fig. 4c, the highest objective 2 value in cluster 2, is lower than any objective 2 values in cluster 3. Thus, when the decision-makers main objective is to minimize the number of tardy and early jobs, with a balance between makespan and the tardiness and earliness time, the solutions of cluster 2 are the most indicated. On the other hand, the solutions of cluster 3 have the opposite behaviour; all objective 1 values are lower in cluster 3, than in cluster 2. Moreover, regarding objective 3, the solutions have lower values in cluster 3 than in cluster 2. Thus, cluster 3 prioritize more objective 1 and 3 than cluster 2, (Figs. 4b and 4c). Thence, cluster 3 solutions are indicated when the decision-maker intends to prioritize the makespan and the tardiness and earliness time, keeping a balance in terms of the number of tardy and early jobs. Finally, in cluster 4, composed exclusively of the MOPSO solutions, some of them have lower objective 3 values than any other clusters. However, the same cluster has the highest objective 2 values compared to the other clusters.

As previously mentioned, central solutions on the Pareto Front balance the objectives considered in the model, and it is normally chosen when the decision-maker does not want to prioritize any of the three objectives considered. In this paper, as we have three scenarios and three objectives function, each scenario will prioritize one objective function. Thus, in this present scenario, the first one, objective 1 will be prioritized; it is the makespan. So, a solution with the lowest makespan was chosen to be better analysed. The solution chosen is marked with a green circle in Fig. 4, belonging to cluster 3. The same is denoted by $(f_1, f_2, f_3) = (232{,}213{,}8334)$, considering that f_1 corresponds to the makespan minimization, f_2 is the minimization of the number of early and tardy jobs and f_3 represents the minimization of tardiness and earliness time. Coincidentally, this solution can also be considered a central solution on the final Pareto Front; consequently, it minimizes more f_1 than any other solution and keeps a balance between the other objective considered. Therefore, this solution is the ideal choice when production is the main priority, without despising the customer's requirements. Table 2 describes the results provided by this solution in terms of makespan, Number of jobs (N. jobs j), Number of tardy (NT) and early (NE) jobs, earliness (E) and tardiness time (T) for each machine m.

Table 2
Results of the optimum solution chosen in the first scenario.

Machines m	**1**	**2**	**3**	**4**	**5**
Makespan	160	213	228	223	232
N. jobs j	7	10	11	12	10
NT	6	9	9	8	9
NE	1	1	2	3	1
E	-8	-14	-22	-22	-10
T	395	907	1015	900	912

Concerning objective function 2, the solution presented the number of tardy jobs equal to 6, 9, 9, 8, and 9 for the machine 1 to 5, respectively. And, in terms of early jobs, this solution indicates 1, 1, 2, 3, and 1 job for machines 1 to 5, respectively. Regarding objective 3, the solution presents the tardiness time, in minutes, equal to 395, 907, 1015, 900, and 912, and the earliness time, also in minutes, equal to 8, 14, 22, 22, and 10, both for machines 1 to 5, respectively. The Gantt chart of this solution is shown in Fig. 5, being 232 the maximum makespan of the solution (objective function 1), and the completion time of each machine can be verified in Table 2. As can be seen, the maximum conclusion times of the other machines are close, which means that all machines are working at similar times, with a similar number of jobs (N. jobs j), also described in Table 2.

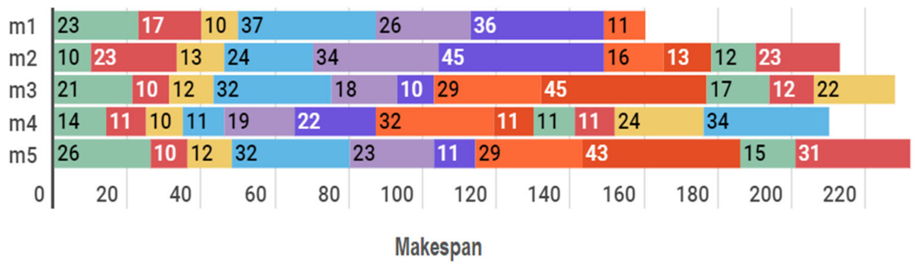


Fig. 5. Gantt chart of solution makespan in the first scenario

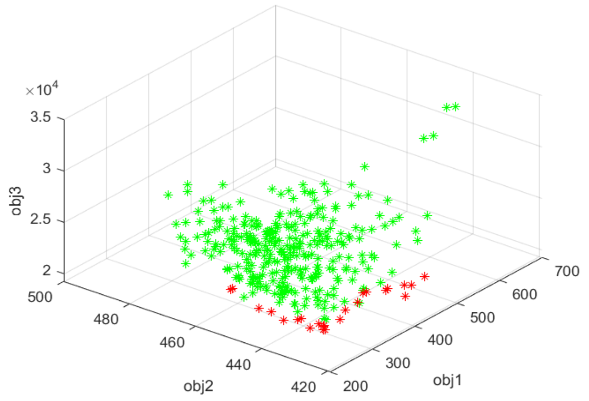
- *Results of the second simulation*

The second scenario considered 10 machines and 100 jobs. The results were obtained in the same way described in the first case. Table 3 describes the results of each algorithm.

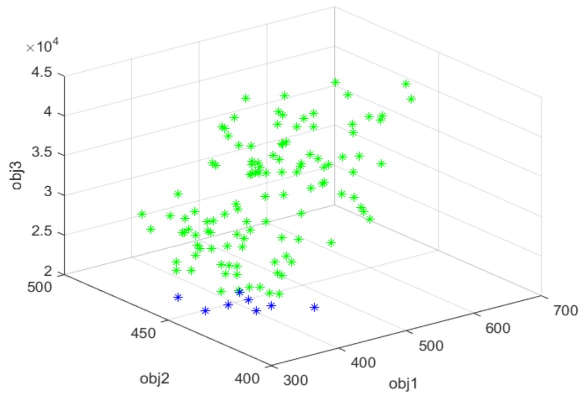
Table 3
Algorithms results of the second scenario

Results	MOPSO			MOGWA			NSGA-II		
	f_1	f_2	f_3	f_1	f_2	f_3	f_1	f_2	f_3
Median	330.62	435.08	20959.54	359.37	43325	22716	271.50	430.20	18659.83
SD	41.39	8.93	1213.87	19.78	8	659.75	21.55	6.91	849.95
Min	275	419	19244	322	413	21628	257	420	17859
Max	446	459	23213	403	455	23905	366	444	19351
NDS		24			8			24	
NDSFPF		1			1			17	
Time (s)		140			240			50	

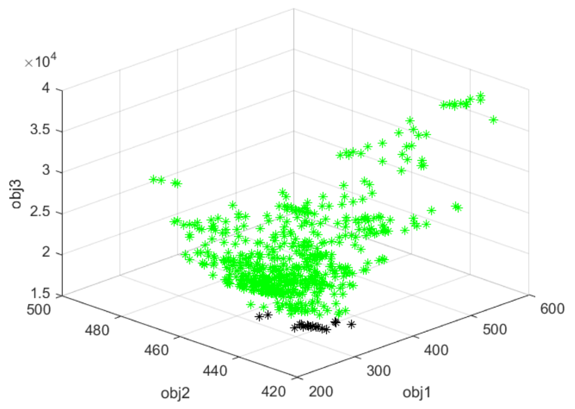
As occurred in the first scenario, objectives 1 and 2 presented more homogeneous solutions than objective 3; the standard deviation of the third solution is higher than the others. About the processing time, the same behaviour observed in the first scenario is observed here: the NSGA-II is approximately three times faster than MOPSO and approximately five times faster than MOGWA. Fig. 6 illustrates the solutions in terms of dominated (green points) and non-dominated solutions (highlighted points). Thereby, Fig. 6a describes the non-dominated solutions of the MOPSO by the red points; Fig. 6b describes the MOGWA non-dominated solutions by the blue points, and Fig. 6c presents the NSGA-II results, with the non-dominated solution in black. Finally, Fig. 6d illustrates the final Pareto Front generated by the dominance comparison of the previous Pareto Fronts.



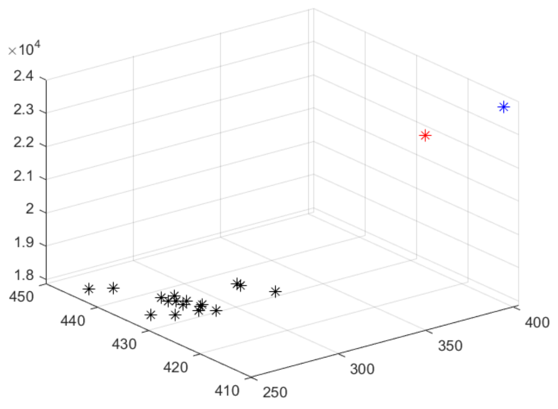
(a) Pareto Front of MOPSO



(b) Pareto Front of MOGWA



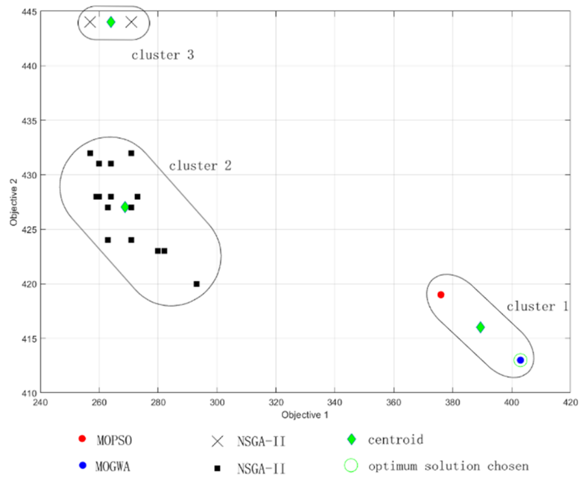
(c) Pareto Front of NSGA-II



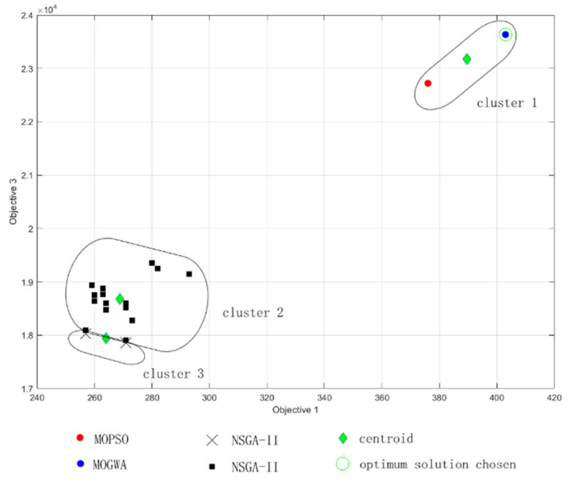
(d) Final Pareto Front

Fig. 6. Pareto Front in the second scenario

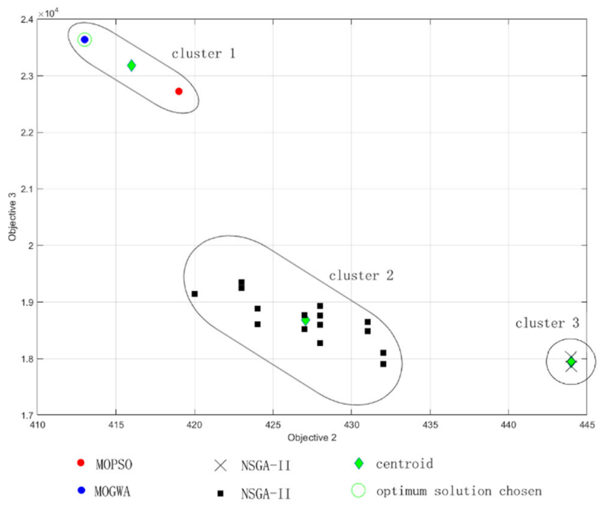
From Fig. 6, it is possible to see that the MOPSO solutions are more distributed than the MOGWA and NSGA-II. Concerning the number of the non-dominated solutions presented in Table 3, as also illustrated in Fig. 6, the MOPSO have 24 non-dominated solutions (Fig. 6a) while the MOGWA had 8 non-dominated. However, only 1 of each continues to be non-dominated when compared with the other two algorithms (Fig. 6d). The NSGA-II generated 24 non-dominated solutions (Fig. 6c), reminding 17 after comparing with the algorithms (Fig. 6d). Thus, the final Pareto Front of the second simulation is composed of 19 non-dominated solutions provided by the three algorithms and illustrated in Fig. 6d. Similar to simulation 1, in the second one, the final Pareto Front was analysed by the bio-inspired clustering algorithm based on GA. According to the clustering algorithm results, the intra-cluster distances are equal to $S_{c1} = 646.59$, $S_{c2} = 1201.11$ and $S_{c3} = 116.38$, being the centroids coordination equal to $c_1(f_1, f_2, f_3) = (389.5, 416, 23177)$, $c_2(f_1, f_2, f_3) = (268.73, 427.06, 18678.27)$ and $c3(f1, f2, f3) = (264, 444, 17941)$, and the DB index obtained is 1.32. Finally, the inter-cluster distances, it is the Euclidean distances between centroids are $D(c_1, c_2) = 4500.36$, $D(c_1, c_3) = 5237.58$, and $D(c_2, c_3) = 737.48$.



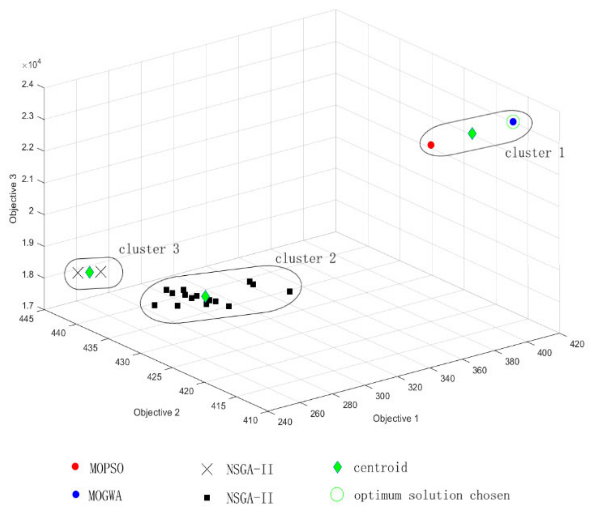
(a) Objective 1 versus 2



(b) Objective 1 versus 3



(c) Objective 2 versus 3



(d) Final Pareto Front

Figure 7. Pareto Front in the second scenario.

Fig. 7 illustrates the clustering algorithm solution, in which the MOPSO provided the red solution, the blue comes from the MOGWA, and the black ones describe the NSGA-II solutions. Again, a dimensional reduction strategy was performed for better visualization and understanding of the results (Figs. 7a, 7b and 7c), and Fig. 7d shows the tri-dimensional clustering algorithm results. In the second scenario, the clustering algorithm solution results in three clusters. Cluster 1 is composed of one solution of MOPSO and another from MOGWA algorithm, while clusters 2 and 3 are exclusively composed of the NSGA-II solutions. The solutions of cluster 1 prioritize objective function 2, as they present the lowest values of this objective than any other solution in the Pareto Front (Fig. 7a). However, objectives 1 and 3 were penalized since they have the highest values in this cluster (Fig. 7b). Thus, if the production line needs a minimal number of tardy and early jobs, the solution of cluster 1 is the best choice. In its turn, cluster 2 is composed of more central solutions. Naturally, these solutions are more balanced than the solutions of the other clusters. The solutions of cluster 2 are indicated when the decision-maker wants to give similar priority to both objectives. Finally, cluster 3 prioritizes objectives 1 and 3, but it penalizes objective 2, since the highest objective 2 values are found in cluster 3 (Fig. 7c). Thus, the solutions of cluster 3 are the better choice when the makespan, tardiness, and earliness time are the main priorities. In the second scenario, a solution that prioritizes objective function 2 was chosen to be better analysed. This solution was provided by MOGWA algorithm, which belongs to cluster 1, and it is marked with a green circle in Fig. 7. In terms of objective function, the mentioned solution, results $(f_1, f_2, f_3) = (403{,}413{,}23634)$, in which f_1 is the makespan, f_2 is the number of early and tardy jobs, and f_3 is tardiness and the earliness time. Table 4 describes the results provided by this solution.

Table 4
Results of the optimum solution chosen in the second scenario.

Machines m	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Makespan	81	90	229	141	331	398	403	329	233	142
N. jobs j	4	5	10	9	9	11	17	16	12	10
NT	2	2	7	6	10	16	14	11	8	3
NE	2	3	3	2	1	1	2	1	1	2
E	-19	-28	-50	-29	-3	-2	-13	-5	-12	-19
T	63	72	822	389	1711	3200	2578	1743	929	215

The solution mentioned defines the second objective function with the values of tardy jobs equal to 2, 2, 7, 6, 10, 16, 14, 11, 8, and 3 and the number of early jobs equal 2, 3, 3, 2, 1, 1, 2, 1, 1, and 2 for machines 1 to 10, respectively. In relation to objective 3 the tardiness times are equal to 63, 72, 822, 389, 1711, 3200, 2578, 1743, 929, and 215, and the earliness time equal to 19, 28, 50, 29, 3, 2, 13, 5, 12, and 19 for machines 1 to 10, respectively. Regarding objective 1, the Gantt chart of this solution is shown in Fig. 8, being 403 the makespan of the solution; and the completion time of each machine can be verified in Table 4. In this case, the completion time of the other machines are not close, generating a long waiting time for some machine, such as machines 1, 2, 3, 4, 9, and 10, since the makespan is not the priority of the solution chosen, as occur in the first scenario.

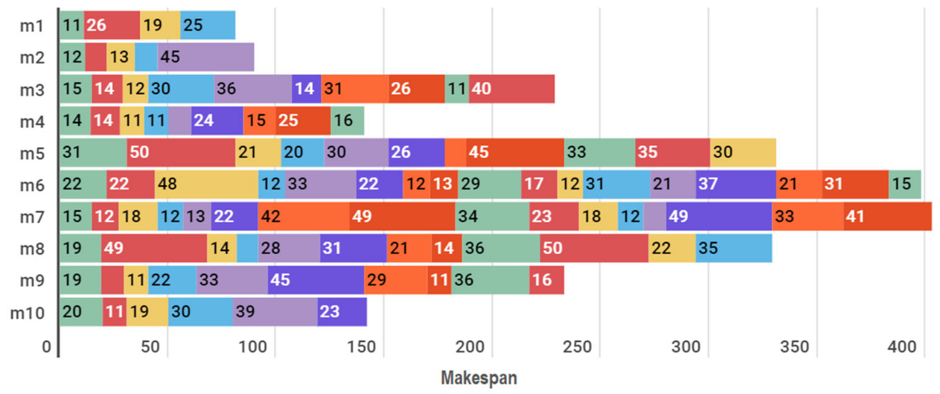


Fig. 8. Gantt chart of solution makespan in the second scenario

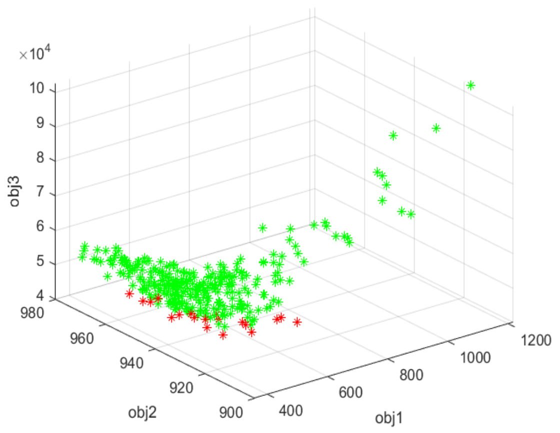
- *Results of the third simulation*

The third scenario considered 20 machine and 200 jobs, double the value of the second scenario. **Table 5** shows the results obtained by each algorithm in the same way considered in the first and second scenarios.

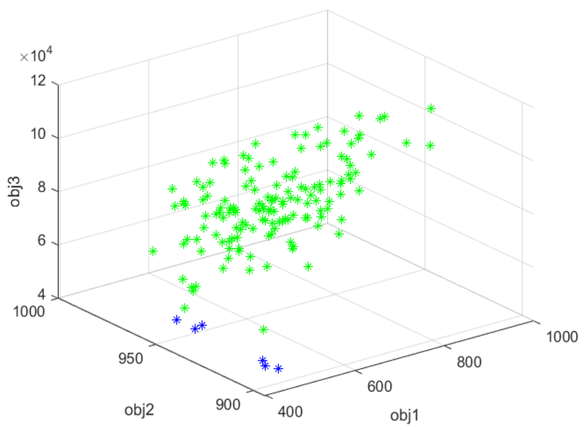
Table 5
Algorithms results of the third scenario.

Results	**MOPSO**			**MOGWA**			**NSGA-II**		
	f_1	f_2	f_3	f_1	f_2	f_3	f_1	f_2	f_3
Median	417.94	925.63	52964.73	409.33	914.83	51413	321.60	895.52	44105
SD	55.79	13.17	2061.02	10.91	18.25	1886.89	19.10	12.79	1059.94
Min	536	952	57527	396	894	49001	297	870	42476
Max	355	904	49843	426	939	54450	386	923	46810
NDS		19			6			57	
NDSFPF		0			0			57	
Time (s)		240			400			95	

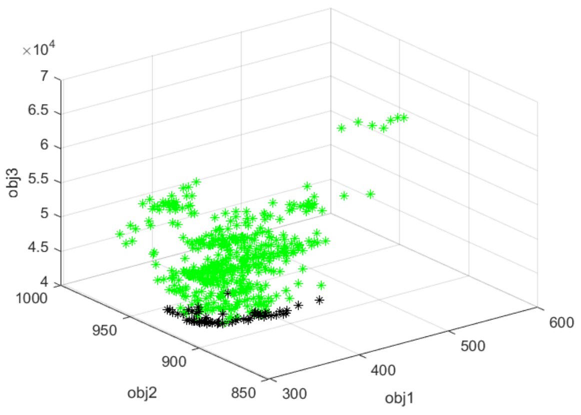
Since the value range of objective 3 is wider than those for objective 1 and 2, the minor standard deviation patterns in those were kept. Concerning the processing time, it was possible to observe the same pattern of the two previous simulations; the NSGA-II is faster, followed by the MOPSO and the MOGWA algorithms. Concerning the individual solution of each algorithm, Fig. 9 illustrate the results achieved in the 30 executions in term of dominated (green points) and non-dominated (highlighted points). Thereby, Fig. 9a describes the non-dominated solutions of the MOPSO by the red points; Fig. 9b describes the MOGWA non-dominated solutions by the blue points, and Fig. 9c presents the NSGA-II results, with the non-dominated solution in black. Furthermore, Fig. 9d illustrates the final Pareto Front generated by the dominance comparison of the previous Pareto Fronts. Concerning Table 5 and Fig. 9, the MOPSO initially had 19 non-dominated solutions, and the MOGWA had 6. However, after the comparison with the 57 solutions of the NSGA-II, all the MOPSO and MOGWA solution becomes dominated. So, the NSGA-II could dominate all other solutions generated by the other two MOEAs.



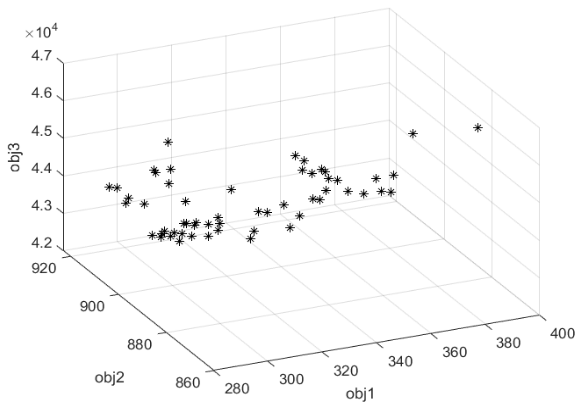
(a) Pareto Front of MOPSO



(b) Pareto Front of MOGWA



(c) Pareto Front of NSGA-II

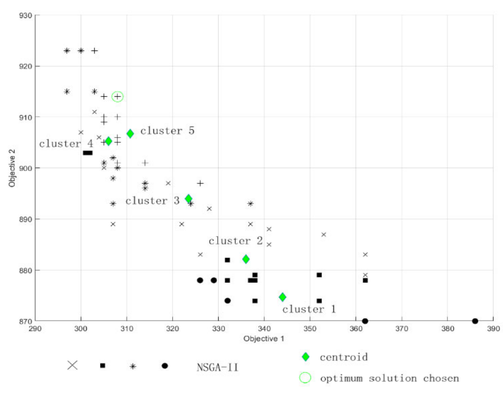


(d) Final Pareto Front

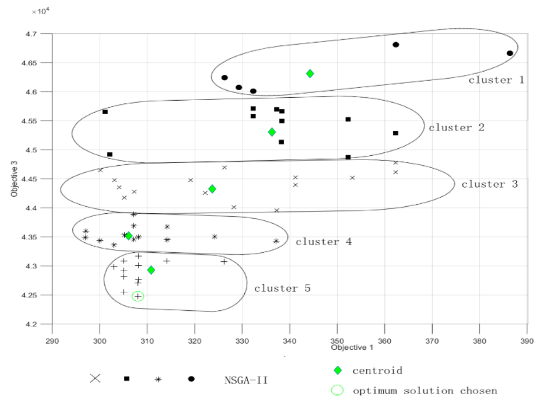
Fig. 9. Pareto Front in the third scenario.

Again, a clustering analyse was performed. According to the bio-inspired clustering algorithm, the final Pareto Front of the third simulation, results in 5 clusters, being the DB index equal to 1.74. The intra-cluster distances are equal to $S_{c1} = 695.85$, $S_{c2} = 963.97$, $S_{c3} = 729.84$, $S_{c4} = 370.31$ and $S_{c5} = 694.30$, and the centroids coordination are $c_1(f_1, f_2, f_3) = (344{,}874.66{,}46311.5)$, $c_2(f_1, f_2, f_3) = (336{,}882.15{,}45307.2)$, $c_3(f_1, f_2, f_3) = (323.5{,}894{,}44326)$, $c_4(f_1, f_2, f_3) = (306{,}905.27{,}43516.09)$ and $c_5(f_1, f_2, f_3) = (310.76{,}906.76{,}42927)$. Finally, the inter-cluster distances, results in $D(c_1, c_2) = 1004.36$, $D(c_1, c_3) = 1985.7$, $D(c_1, c_4) = 2795.84$, $D(c_1, c_5) = 3384.82$, $D(c_2, c_3) = 981.35$, $D(c_2, c_4) = 810.18$, $D(c_2, c_5) = 2380.46$, $D(c_3, c_4) = 810.18$, $D(c_3, c_5) = 1399.12$, and $D(c_4, c_5) = 589.11$.

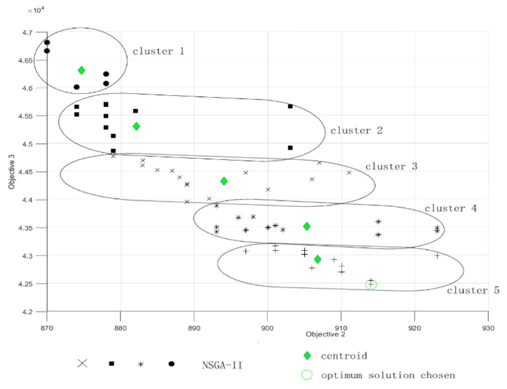
As previously mentioned, the final Pareto Front was divided into 5 clusters, as illustrated in Fig. 10. According to Fig. 10a it is not possible to obtain a clear description of each cluster division since objectives 1 and 2 are mixed in the domain; thus, the better cluster division is given by objective 3, which means that this Pareto Front arrangement is very useful when it intended to divide objective 3 into sections. In this way, when comparing objective 3 versus objective 1 (Fig. 10b) and objective 2 (Fig. 10c), it is possible to have a clear division between the elements that belong to each cluster.



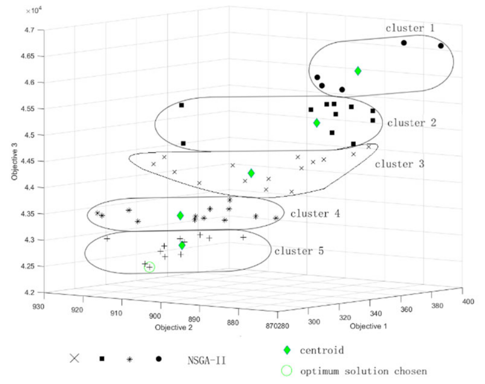
(a) Objective 1 versus 2



(b) Objective 1 versus 3



(c) Objective 2 versus 3



(d) Final Pareto Front

Fig. 10. Pareto Front in the third scenario

By Fig. 10b we have the highest objective 3 values in cluster 1, with a decrease of this value in clusters 2, 3, and 4, until achieving the lowest values in cluster 5. Besides, clusters 4 and 5 have the lowest values of objective 1, whereas cluster 1 has the highest values of this variable. In its turn, Fig. 10c confronts objectives 2 and 3, the highest objective 3 values are in cluster 1, with a decrease of this value in clusters 2, 3, and 4, until achieving the lowest values in cluster 5. Nevertheless, in this case, the lowest objective 2 values are in cluster 1, and the highest values are in clusters 4 and 5. In the third scenario, a solution that prioritizes objective function 3 was chosen as the optimum solution to be better analysed. This solution has the lowest tardiness and early time and belongs to cluster 5, and it is marked with a green circle in Fig. 10. So, its coordinator is $(f_1, f_2, f_3) = (308{,}914{,}42476)$, in which f_1 is the makespan, f_2 is the number of early and tardy jobs, and f_3 is tardiness and the earliness time. Table 6 describes the results provided by this solution.

Table 6
Results of the optimum solution chosen in the third scenario.

Machines *m*	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Makespan	183	271	243	274	197	252	282	242	225	207
N. jobs *j*	9	10	9	11	10	10	12	9	10	8
NT	9	10	7	10	8	9	10	7	8	8
NE	0	0	1	1	2	1	2	2	1	0
E	0	0	-5	-14	-14	-2	-16	-33	-13	0
T	732	1277	943	1325	696	1104	1220	921	893	783
Machines *m*	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
Makespan	257	159	305	245	288	283	279	308	251	298
N. jobs *j*	9	8	12	9	10	12	10	11	10	11
NT	7	8	11	8	10	11	10	8	10	10
NE	2	0	1	1	0	1	0	3	0	1
E	-27	-0	-4	-21	0	-5	-0	-23	0	-13
T	899	472	1501	900	1288	1223	1325	1249	1023	1369

The solution mentioned defines the second objective function with the number of tardy jobs equal to 9, 10, 7, 10, 8, 9, 10, 7, 8, 8, 7, 8, 11, 8, 10, 11, 10, 8, 10, and 10, and the number of early jobs equal to 0, 0, 1, 1, 2, 1, 2, 2, 1, 0, 2, 0, 1, 1, 0, 1, 0, 3, 0, and 1 for machines 1 to 20, respectively. In relation to objective 3 the tardiness times are equal to 732, 1277, 943, 1325, 696, 1104, 1220, 921, 893, 783, 899, 472, 1501, 900, 1288, 1223, 1325, 1249, 1023, and 1369; and the earliness time equal to 0, 0, 5, 14, 14, 2, 16, 33, 13, 0, 27, 0, 4, 21, 0, 5, 0, 23, 0, and 13 for machines 1 to 20, respectively. Regarding objective 1, the Gantt chart of this solution is shown in Fig. 11, begin 308 the makespan of the solution; and the completion time of all 20 machines can be verified in Table 6. As occurred in the second scenario, the maximum completion time of the other machines are not close, generating long waiting time for some machine, such as machines 1, 5, 9, 10, and 12.

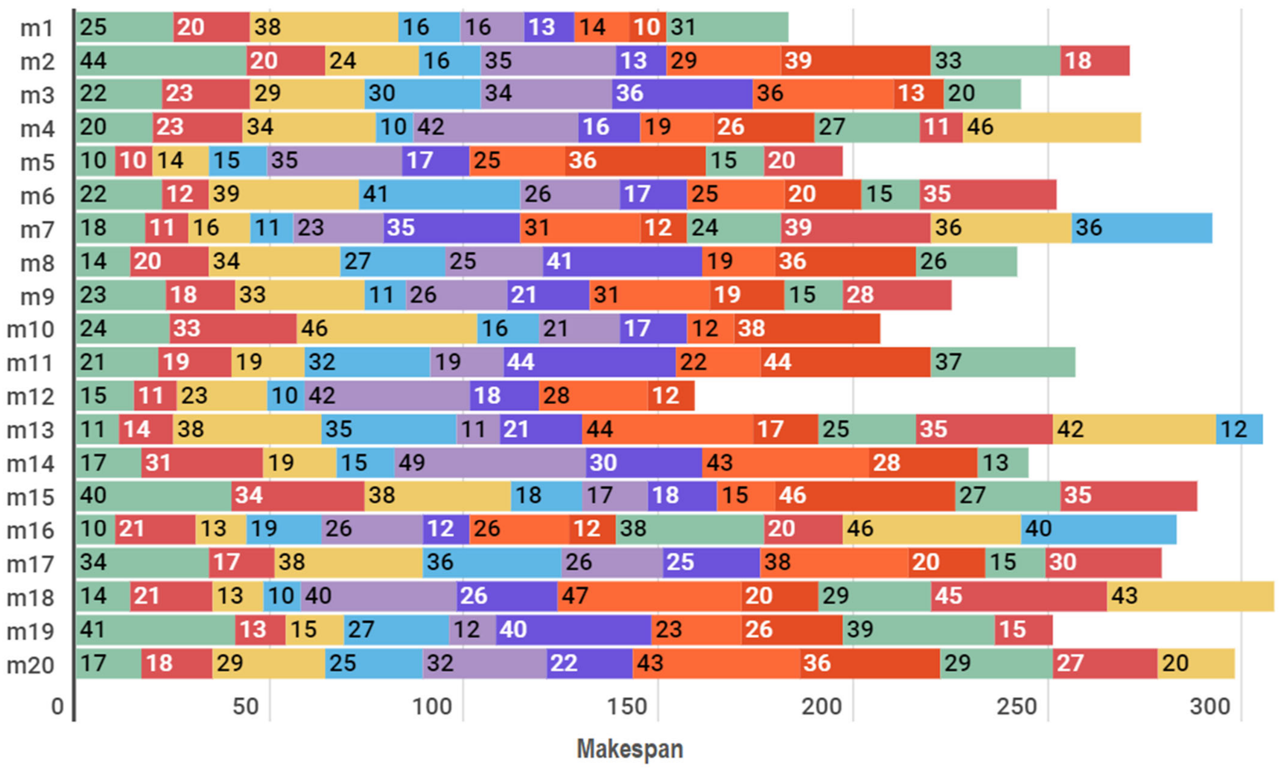


Fig. 11. Gantt chart of solution makespan in the third scenario

6. Conclusion

This work explored production scheduling analysis under different bio-inspired approaches. Three job scheduling problems were studied using unrelated parallel machines. The first was composed of 5 parallel machines and 50 jobs, the second contained 10 parallel machines and 100 jobs, and the third one had double the values of the second problem, it is 200 jobs and 20 machines. Five important production scheduling performance measures (makespan, tardiness and earliness time, number of tardy and early jobs) were combined into three objective functions, and a multi-objective mathematical optimization model was proposed. The methodology adopted to solve the model considered three multi-objective evolutionary algorithms: MOPSO, MOGWA, and NSGA-II. The results of these three algorithms were compared between them in order to generate a final Pareto Front composed of the non-dominated solutions of all algorithms. Moreover, an automatic bio-inspired clustering algorithm based on Genetic Algorithm methods was utilized to enrich the decision support. The results of the proposed methodology were very satisfactory since it presented a final solution (final Pareto Front) strengthened through different bio-inspired techniques. These results made it possible to perceive the impact of combining different bio-inspired methods to solve a problem. Since bio-inspired algorithms are stochastic methods, they do not guarantee the exact solution. Therefore, the answers tend to vary when different techniques evaluate the same data set. In the case of multi-objective optimization, such variations tend to increase even more, giving rise to conflict between objectives. As established by the no-free-lunch theorem Wolpert and Macready (1997), some algorithms can be better than others in solving a given problem since there is no single best optimization algorithm, so if one algorithm performs better than another algorithm on one class of problems, then it will perform worse on another class of problems (Goel et al., 2020; Wolpert & Macready, 1997). For this reason, it is hard to identify the best algorithm for each data set before confronting the results with the algorithms. However, there are many bio-inspired techniques, making it impossible to compare them in time to choose the perfect one for each problem. Hence, methodologies capable of combining different techniques and comparing them to find a more robust solution is a great asset in the universe of bio-inspired techniques. In this approach, if we considered only one algorithm, the final answer would not be as enriched, although feasible for the decision-maker. This characteristic demonstrates one of the main advantages and innovation of the proposed methodologies: the coverage of a greater variety of decisions since the Pareto Front completes the decision of different methods. Moreover, in Industry 4.0, the request for customized solutions is increasingly important, so providing a wide range of optimal options is always a positive outcome.

As mentioned, all solutions that complete a Pareto Front are considered optimal for the problem because the priorities given to each objective differentiate them. Through clusters analysis, it is possible to group the solutions. These groups provide a visual analysis of the possible solutions for more effectively identifying a given preference, which is of upmost importance for the decision-maker. Furthermore, the decision-maker preferences may vary from moment to moment since the industrial environment is generally very dynamic. Therefore, when a production system has high demand, it is necessary to prioritize production or more internal performance measures, e.g., the makespan. On the other hand, when there are fewer tasks to be accomplished, it is possible that more attention should or could be given to providing higher or special importance or priority to customer-oriented performance measures, such as the ones considered in this work, related to production orders due dates accomplishment, in order to deliver tasks, ideally just in time, which is, not being late nor delivered much in advance, to increase customer-company relation.

Sustainability issues, big and complex data processing, and customization for enabling and supporting real-time decision-making in a dynamically changing production environment, under the constant transformation of the market, configure major trends currently, in Industry 4.0, and further in Industry 5.0. Thus, flexible and adaptive production planning is an excellent differential for production management. Considering this, the methodology presented in this work proved to be of great value for intelligent production scheduling since it offers excellent flexibility to the decision-maker so that he/she can customize production decisions according to established priorities, either of the company and/or the customers' preferences.

In general, the approach proved to be effective in supporting scheduling decision-making to explore a set of most widely considered performance measures regarding more production or customer-oriented performance measures. It is noteworthy that the final decision of the optimum solution is up to the manager, who, through the presented methodology, can define different solutions according to the priority of the production system versus customer requirements. As a suggestion for the future path of this work, it is intended to apply the methodology and the model to a real problem and compare the developed approach with other ones available in the literature.

Acknowledgements

This work has been supported by FCT – Fundação para a Ciência e Tecnologia within the R&D Units Project Scope: UIDB/00319/2020 and EXPL/EME-SIS/1224/2021. Beatriz Flamia Azevedo is supported by FCT Grant Reference SFRH/BD/07427/2021 The authors are grateful to the Foundation for Science and Technology (FCT, Portugal) for financial support through national funds FCT/MCTES (PIDDAC) to CeDRI (UIDB/05757/2020 and UIDP/05757/2020) and SusTEC (LA/P/0007/2021).

References

Arbelaitz, O., Gurrutxaga, I., Muguerza, J., Perez, J. M., & Perona I. (2013). An extensive comparative study of cluster validity indices. *Pattern Recognition, 46*(1), 243–256, (DOI: 10.1016/j.patcog.2012.07.021).

Azevedo, B. F. (2020). Study of genetic algorithms for optimization problems. Master's thesis, Instituto Politécnico de Bragança Escola Superior de Tecnologia e Gestão, Portugal, Bragança, Portugal.

Azevedo, B. F., Varela, M. L. R., & Pereira, A. I. (2022). Production scheduling using multi-objective optimization and cluster approaches. In: Abraham A et al (eds) Innovations in Bio-Inspired Computing and Applications IBICA 2021 - *Lecture Notes in Networks and Systems 419*, (DOI: 10.1007/ 978-3-030-96299-9 12).

Bansal, J. C., Singh, P. K., & Nikhil, R. P. (2019). Evolutionary and swarm intelligence algorithms. *Studies in Computational Intelligence*, (DOI: 10.1007/ 978-3-319-91341-4).

Barenji, R.V., Barenji, A.V., & Hashemipour, M. (2014). A multi-agent RFID-enabled distributed control system for a flexible manufacturing shop. *The International Journal of Advanced Manufacturing Technology, 71*(9-12), 1773–1791, (DOI: 10.1007/s00170-013-5597-2).

Borangiu, T., Morariu, O., Raileanu, S., Trentesaux, D., Leitão, P., & Barata, J. (2020) Digital transformation of manufacturing. industry of the future with cyber-physical production systems. *Romanian Journal of Information Science and Technology 23*, 3–37.

Chaouch, I., Driss, O. B., & Ghedira, K. (2017) A modified Ant Colony Optimization algorithm for the Distributed Job shop Scheduling Problem. *Procedia Computer Science, 112*, 296–305, (DOI: 10.1016/j.procs.2017.08.267).

Chen, J., Qi, X., Chen, L., Chen, F., & Cheng, G. (2020) Quantum-inspired ant lion optimized hybrid k-means for cluster analysis and intrusion detection. *Knowledge-Based Systems, 203*, 106167, (DOI: 10.1016/j.knosys.2020.106167).

Chen, Y., Guan, Z., Wang, C., Chou, F., & Yue, L. (2022). Bi-objective optimization of identical parallel machine scheduling with flexible maintenance and job release times. *International Journal of Industrial Engineering Computations,* 13(4), 457–472, (DOI: 10.5267/j.ijiec.2022.8.003).

Coello-Coello, C. A., & Lechuga, M. S. (2002). MOPSO: a proposal for multiple objective particle swarm optimization. In: *Proceedings of the 2002 Congress on Evolutionary Computation*. CEC'02 (Cat. No.02TH8600), *2,* pp 1051–1056 vol.2, (DOI: 10.1109/CEC.2002.1004388).

Coello-Coello, C. A., Pulido, G. T., & Lechuga, M. S. (2004) Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation, 8*(3), 256–279, (DOI: 10.1109/TEVC.2004.826067).

Dai, M., Tang, D., Giret, A., & Salido, M. A. (2019). Multi-objective optimization for energy-efficient flexible job shop scheduling problem with transportation constraints. *Robotics and Computer-Integrated Manufacturing, 59*, 143–157, (DOI: 10.1016/j.rcim.2019.04.006).

Deb, K. (2011). Multi-objective optimization using evolutionary algorithms: An introduction. In: *Multi-objective Evolutionary Optimisation for Product Design and Manufacturing*, Wang, L. and Amos, H. C. Ng and Deb, K. (eds.), 1st edn, Springer-Verlag London.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002) A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation, 6*(2), 182–197, (DOI: 10.1109/4235.996017).

Ezugwu, A. E., Shukla, A. K., Agbaje, M. B., Oyelade, O. N., José-García, A., & Agushaka, J. O. (2021). Automatic clustering algorithms: a systematic review and bibliometric analysis of relevant literature. *Neural Computing and Applications*, *33*(11), 6247-6306.

Fu, Y., Ding, J., Wang, H., & Wang, J. (2018). Two-objective stochastic flow-shop scheduling with deteriorating and learning effect in Industry 4.0-based manufacturing system. *Applied Soft Computing*, *68*, 847-855.

Goel, L., Raman, S., Dora, S. S., Bhutani, A., Aditya, A. S., & Mehta, A. (2020) Hybrid computational intelligence algorithms and their applications to detect food quality. *Artificial Intelligence Review, 53*(2), 1415–1440, (DOI: 10.1007/s10462-019-09705-8).

Gong, D., Han, Y., & Sun, J. (2018). A novel hybrid multi-objective artificial bee colony algorithm for blocking lot-streaming flow shop scheduling problems. *Knowledge-Based Systems*, *148*, 115-130. (DOI: 10.1016/j.knosys.2018.02.029).

Jia, S., Yi, J., Yang, G., Du, B., & Zhu, J. (2013). A multi-objective optimisation algorithm for the hot rolling batch scheduling problem. *International Journal of Production Research, 51*(3), 667–681, (DOI: 10.1080/00207543.2011.654138).

Kagermann, H., Lukas, W., & Wahlster, W. (2011). Industrie 4.0: Mitdem Internet der Dinge auf dem Weg zur 4. industriellen Revolution. VDI nachrichten 13.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In: *Proceedings of ICNN'95 - International Conference on Neural Networks*, *4*, 1942–1948 vol.4, (DOI: 10.1109/ICNN.1995.488968).

Kok, J., Gonzalez, F., Kelson, N., & Periaux, J. (2011). An FPGA-based approach to multi-objective evolutionary algorithm for multi-disciplinary design optimisation. In Poloni, C, Gauger, N, Periaux, J, Quagliarella, D, & Giannakoglou, K (Eds.) *Proceedings of the International Conference on Evolutionary and Deterministic Methods for Design, Optimization and Control with Applications to Industrial and Societal Problems (Eurogen 2011).* CIRA - Italian Aerospace Research Centre, Italy, pp. 1-10.

Lin, D. Y., & Huang, T. Y. (2021) A hybrid metaheuristic for the unrelated parallel machine scheduling problem. *Mathematics, 9*(7), 768, (DOI:10.3390/math9070768).

Lu, C., Xiao, S., Li, X., & Gao, L. (2016) An effective multi-objective discrete grey wolf optimizer for a real-world scheduling problem in welding production. *Advances in Engineering Software, 99*, 161–176, (DOI: 10.1016/j.advengsoft. 2016.06.004).

Luo, S., Zhang, L., & Fan, Y. (2019) Energy-efficient scheduling for multi-objective flexible job shops with variable processing speeds by grey wolf optimization. *Journal of Cleaner Production, 234*, 1365–1384, (DOI: 10.1016/j.jclepro.2019. 06.151).

MATLAB (2019). The mathworks inc 2019a. https://www.mathworks.com

Miettinen, K. (1998) *Nonlinear multiobjective optimization*, 1st ed. International Series in Operations Research & Management Science, Springer.

Mirjalili, S. (2022) Multi-objective grey wolf optimizer (mogwo). https://www.mathworks.com/matlabcentral/fileexchange/55979-multi-objective-grey-wolf-optimizer-mogwo, retrieved February 2, 2022.

Mirjalili, S., Mirjalili, S. M., & Lewis A. (2014). Grey wolf optimizer. *Advances in Engineering Software, 69*, 46–61, (DOI: 10.1016/j.advengsoft.2013.12.007).

Mirjalili, S., Saremi, S., Mirjalili, S. M., & dos S Coelho, L. (2016). Multi-objective grey wolf optimizer: a novel algorithm for multi-criterion optimization. *Expert Systems with Applications, 47*, 106–119, (DOI: 10.1016/j.eswa.2015.10.039).

Ojstersek, R., Brezocnik, M., & Buchmeister, B. (2020). Multi-objective optimization of production scheduling with evolutionary computation: A review. *International Journal of Industrial Engineering Computations*, *11*(3), 359-376.

Pinedo, M. L. (2012). *Scheduling: theory, algorithms, and systems*. Springer (DOI:10.1007/978-1-4614-2361-4).

Piroozfard, H., Wong, K.Y., & Wong, W. P. (2018). Minimizing total carbon footprint and total late work criterion in flexible job shop scheduling by using an improved multi-objective genetic algorithm. *Resources, Conservation and Recycling 128*, 267–283, (DOI: 10.1016/j.resconrec.2016.12.001).

Qaddoura, R., Faris, H., & Aljarah, I. (2021). An efficient evolutionary algorithm with a nearest neighbour search technique for clustering analysis. *Journal of Ambient Intelligence and Humanized Computing, 12*, 8387–8412, (DOI: 10.1007/s12652-020-02570-2).

Qin, H., Fan, P., Tang, H., Huang, P., Fang, B., & Pan, S. (2019). An effective hybrid discrete grey wolf optimizer for the casting production scheduling problem with multi-objective and multi-constraint. *Computers & Industrial Engineering*, *128*, 458-476. (DOI: 10.1016/j.cie.2018.12.061).

Reis, P. C. S. O. (2020). Ferramenta de apoio ao escalonamento da produção. Master's thesis, Instituto Superior de Engenharia do Porto - Departamento de Engenharia Mecânica.

Safarzadeha, H., & Niakia, S. T. A. (2023). Unrelated parallel machine scheduling with machine processing cost. *International Journal of Industrial Engineering Computations, 14*(1), 33–48, (DOI: 10.5267/j.ijiec.2022.10.004).

Santos, A. S., Madureira A. M., & Varela M. L. R. (2015). An ordered heuristic for the allocation of resources in unrelated parallel machines*. International Journal of Industrial Engineering Computations, 6*(2)

Shalev-Shwartz, S., & Ben-David, S. (2014). *Understanding machine learning: from theory to algorithms*. Cambridge University Press.

Sheikhalishahi, M., Eskandari, N., Mashayekhi, A., & Azadeh A. (2019). Multi-objective open shop scheduling by considering human error and preventive maintenance. *Applied Mathematical Modelling, 67*, 573–587, (DOI: 10.1016/j.apm.2018.11.015).

Singh, T. (2021). A novel data clustering approach based on whale optimization algorithm. *Expert Systems, 38*, 8387–8412, (DOI: doi.org/10.1111/exsy.12657).

Sivanandam, S. N., & Deepa, S. N. (2008). *Introduction to Genetic Algorithms*, 1st ed. Springer, (DOI: 10.1007/978-3-540-73190-0).

Varela, M. L. R., & Ribeiro R. A. (2014). Distributed manufacturing scheduling based on a dynamic multi-criteria decision model. Springer, (DOI:10.1007/978-3-319-06323-2_6).

Varela, M. L., Putnik, G. D., Manupati, V. K., Rajyalakshmi, G., Trojanowska, J., & Machado, J. (2021). Integrated process planning and scheduling in networked manufacturing systems for I4. 0: a review and framework proposal. *Wireless Networks*, *27*(3), 1587-1599. Springer. (DOI: 10.1007/s11276-019-02082-8).

Varela, M. L. R., Putnik, G. D., Alves, C. F., Lopes, N., & Cruz-Cunha, M. M. (2022). A Systematic Review of Manufacturing Scheduling for the Industry 4.0. 1st International Symposium on Industrial Engineering and Automation (ISIEA 2022), Managing and Implementing the Digital Transformation, 21st-22nd June 2022, Bozen-Bolzano, Italy. Lecture Notes in Networks and Systems (pp. 237-249), Springer.

Wang, Q., Wang, X., Luo, H., & Xiong, J. (2020) An improved multi-objective evolutionary approach for aerospace shell production scheduling problem. *Symmetry, 12*(4), (DOI: 10.3390/sym12040509).

Wang, Z., Zhang, J., & Yang, S. (2019). An improved particle swarm optimization algorithm for dynamic job shop scheduling problems with random job arrivals. *Swarm and Evolutionary Computation, 51*, 100594, (DOI: 10.1016/j.swevo.2019.100594).

Wolpert, D. H., & Macready, W. G., (1997) No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1*(1), 67–82, (DOI: 10.1109/ 4235.585893).

Yapiz (2022a) Evolutionary clustering and automatic clustering. (URL: https://www.mathworks.com/matlabcentral/fileexchange/52865-evolutionary-clustering-and-automatic-clustering), retrieved February 2, 2022.

Yapiz (2022b) Multi-objective particle swarm optimization (MOPSO). (URL: https://www.mathworks.com/matlabcentral/fileexchange/52870-multi-objective-particle-swarm-optimization-mopso), retrieved February 2, 2022.

Zhang, J., Ding, G., Zou, Y., Qin, S., & Fu, J. (2019) Review of job shop scheduling research and its new perspectives under Industry 4.0. *Journal of Intelligent Manufacturing, 30*(4), 1809–1830, (DOI: 10.1007/s10845-017-1350-2).

Zhang, Q., & Li, H. (2007). MOEA/D: A multi-objective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation, 11*(6), 712-731, (DOI: 10.1109/TEVC.2007.892759).

Zhang, S., Tang, F., Li, X., Liu, J., & Zhang, B. (2021). A hybrid multi-objective approach for real-time flexible production scheduling and rescheduling under dynamic environment in Industry 4.0 context. *Computers & Operations Research, 132*, 105267, (DOI: 10.1016/j.cor.2021.105267).

Zheng, F., Jin, K., Xu, Y., & Liu, M. (2022). Unrelated parallel machine scheduling with processing cost, machine eligibility and order splitting. *Computers & Industrial Engineering, 171*. (DOI: 10.1016/j.cie.2022.108483).