Journal of Optimization in Industrial Engineering
Vol. 16, Issue 2, Summer & Autumn 2023, 99-114
DOI: 10.22094/JOIE.2023.1967925.1992

Research Article

DOE-based Enhanced Genetic Algorithm for Unrelated Parallel Machine Scheduling to Minimize Earliness and Tardiness Costs

Parsa Kianpour [a,*], Deepak Gupta [a], Krishna Krishnan [a], Bhaskaran Gopalakrishnan [b]

[a] *Industrial, Systems, and Manufacturing Engineering Department, Wichita State University, Wichita, USA*
[b] *Industrial and Management Systems Engineering, West Virginia University, Morgantown, USA*

Received 17 September 2022; Revised 31 July 2023; Accepted 01 August 2023

Abstract

This study presents an enhanced genetic algorithm (E-GA) to minimize earliness/tardiness costs in the job shop environment. It considers an unrelated parallel machine scheduling problem with a limit on maximum tardiness levels. This problem is motivated by the experience of one of the authors in a job shop supporting the local aircraft industry that requires strict control on delivery times. Current literature does not consider this critical restriction and unsuccessfully tries to deal with them using higher penalty costs. The proposed method uses the design of experiment (DOE) concept while optimizing the GA operators. Furthermore, it improves the initial solution using a hybrid dispatch rule through a strategic combination of construction and improvement heuristics. The model was applied to a local job shop. The results indicate that E-GA provides a schedule with lower cost and reduced computational time compared to existing dispatch rules in the literature and existing algorithms (OptQuest).

Keywords: Unrelated Parallel Machine Scheduling; Job Shop; Tardiness, Earliness; Heuristic, GA; DOE

1. Introduction

Scheduling is defined as a decision-making process involving allocating resources to the operations. The proper schedule can help industries reduce costs and time and improve customer satisfaction, which are the most desirable goals in modern industries. There are multiple types of scheduling problems in a manufacturing system. One of the most challenging problems in this area is the unrelated parallel machine scheduling problem in which various jobs should be processed on parallel machines. In this problem, the best sequence of the jobs on the machines should be provided, which optimizes the given performance measurement. There are numerous performance measures in scheduling problem, such as makespan, maximum lateness, total completion time, total flow time, etc. A comprehensive list of performance measures with their distribution in the current literature is provided in the study by Allahverdi (2022).

Unrelated parallel machine scheduling can be solved using exact, metaheuristics, heuristic, and hybrid methods. Some studies assumed a static environment, while others considered uncertainties and dynamic environments (Kianpour, Gupta, Krishnan, & Gopalakrishnan, 2022; Kianpour, Gupta, Krishnan, & Gopalakrishnan, 2021). A study by Moser et al. (2022)focuses on a real-life scheduling problem with complex constraints. In order to determine the best solutions for various instance sizes, it suggests a mathematical model and simulated annealing algorithms, displaying enhanced performance in comparison to current methods in terms of minimizing tardiness and makespan. The other study presents an in-depth literature review on the application of heuristic and metaheuristic methods for solving the unrelated parallel machine scheduling problem. The study identified research trends, it categorized and summarized existing research, and provided direction for additional study in the area (Đurasević & Jakobović , 2023). The other study offered a hybrid meta-heuristic method, called LA-ALNS-TS, for solving the unrelated parallel machine scheduling problem with machine and job sequence-dependent setup times. The algorithm combines learning automata adaptive large neighborhood search (LA-ALNS) and tabu search (TS) to improve search efficiency and address the issue of short-term cycles. Experimental findings on extensive benchmarks validate the suggested strategy's usefulness and efficiency (Fang, Zhu, & Mei, 2022). Mixed Integer Linear Programming (MILP) or Mixed Integer Programming (MIP) has been used frequently to solve scheduling problem in unrelated parallel machine systems (Kianpour, Gupta, Krishna, & Gopalakrishnan, 2021). The study discusses scheduling unrelated parallel machines to reduce overall tardiness. It suggests an ILS algorithm for larger cases of up to 400 tasks and 20 machines and a MILP formulation for smaller cases of up to 150 jobs and 20 machines, demonstrating the effectiveness of both (De-Alba, Nucamendi-Guillén, & Avalos-Rosales, 2022). The other study assesses the problem while considering machine processing costs. They used MILP to minimize makespan and total cost simultaneously for Pareto optimal solutions. The

*Corresponding author Email address: pxkianpour@shockers.wichita.edu

outcomes show that this approach efficiently resolves even large-scale issues, producing a large number of Pareto solutions in a fair amount of time with little divergence from optimality (Safarzadeh & Niaki, 2023). Ozer & Sarac (2019) considered the concept of the shared resource in identical parallel machine scheduling problem with sequence-dependent setup times. The objective of their study was to minimize the total weighted completion time. Optimal schedules were obtained for almost all small problems using a mixed-integer programming model. The paper by Fanjul-Peyro (2020) deals with the unrelated parallel machine scheduling problem with setups and resources with the objective of minimizing the makespan. A MILP is presented to model this problem. In addition, a three-phase algorithm based on an exact mathematical method is introduced to solve up to 50 jobs. In terms of improving the efficiency of the MIP models, the study by Wang & Ye (2019) developed an enhanced MIP formulation based on the concept of "divide and conquer" in the branch-and-bound algorithm. They proposed two MIP formulations based on dummy jobs and linear ordering variables. Computational results for multiple instances proved the effectiveness of the proposed branch-and-bound algorithm. Despite numerous studies to provide the exact solution, it is not computationally efficient to solve unrelated parallel machine scheduling problems using MIP for large-size instances. Therefore, heuristic methods are used to provide the near-optimal solution in a reasonable computational time (Lenstra & Rinnooy Kan 1979). In the study by Athmani et al.(2022), three heuristics, five local search techniques, and three metaheuristics, including Late Acceptance Hill Climbing and two Simulated Annealing versions, are introduced to analyze unrelated parallel machine scheduling with release dates and machine- and sequence-dependent setup times. The study assesses and compares these methods using a three-set 1620-instance benchmark, identifying the best initialization heuristic for metaheuristics and establishing the usefulness of the metaheuristics across different sets. The other study investigated an unrelated parallel machine scheduling problem with the objective of minimizing earliness and tardiness penalties in production systems. By using two mixed-integer linear programming formulations, novel heuristics, and an extension of the adaptive large neighborhood search (ET-ALNS), the research demonstrates, through computational experiments and statistical analysis, that ET-ALNS significantly outperforms other methods (Rolim, Nagano, & de Athayde Prata, 2023).

GA has been used widely for unrelated parallel machine scheduling problem with either single or multiple objectives (Türkylmaz & Bulkan 2015; Chen et al. 2012). One of the common objectives in unrelated parallel machine scheduling problem is minimizing the earliness and/or tardiness cost(s). In this problem, it is assumed that customer requires jobs neither early nor late. Completing jobs before their due dates may lead to inventory costs, while shipping the parts after the due date causes potential penalty costs and customer dissatisfaction. Therefore, a schedule is needed to ship the part with minimum deviation from the due date. Few studies have used the GA to minimize earliness and tardiness costs in scheduling problems (Rohaninejad, Sahraeian, & Nouri 2017; Tari & Niari 2018). The study by Yazdani et al. (2017) considers the problem of scheduling as set of jobs in the job shop environment with an objective of minimizing the sum of maximum earliness and tardiness. They proposed an effective meta-heuristic algorithm based upon an imperialist competitive algorithm that significantly outperformed available algorithms in the literature. A study presents a fuzzy approach to tackle unrelated parallel machine scheduling in production environments with machine reliability and scheduling parameter uncertainties, utilizing a fuzzy mathematical model and a fuzzy-based genetic algorithm. Numerical experiments reveal that this algorithm performs exceptionally well for large instances and surpasses the mathematical model for smaller instances (Yaghtin & Javid, 2023). The other study presents a genetic algorithm that integrates fuzzy logic to tackle the flexible job shop scheduling problem (FJSSP) in industries with uncertain setup and processing times due to windows and sequence-dependent setup times. Tested on a real-world fabric finishing production system, the algorithm notably outperforms four standard heuristics, providing efficient solutions with a performance improvement of over 30% (Campo, Cano, Gómez-Montoya, Rodríguez-Velásquez, & Cortés, 2022).

The performance of the evolutionary optimization algorithms highly depends on the value of its parameters. Also, the best set of parameters differs from one type of optimization problem to the other. In other words, the best set of parameters to solve an optimization problem with the objective of minimizing makespan in the flow shop is not necessarily the best set for solving the optimization problem of minimizing tardiness and earliness in the job shop. Therefore, proper parameter tuning of an algorithm should be considered for each specific optimization problem. There are numerous studies in the literature regarding the tuning of parameters in evolutionary algorithms (Á. E. Eiben, Hinterding, & Michalewicz 1999; Nobile et al. 2018; Soares & Carvalho 2022). GA is one of the common evolutionary algorithms which grabs scholars' attention in tuning its parameters to improve the efficiency and effectiveness of the algorithms (A. E. Eiben, Michalewicz, Schoenauer, & Smith 2007; Huang, Li, & Yao 2019; Kramer 2017; Pavón, Díaz, Laza, & Luzón 2009). In terms of Job Shop Scheduling Problems (JSSP), few studies investigate the parameter tuning of the GA to enhance their solutions. Arin, Rabadi & Unal (2011) provided comparative studies on the design of experiment (DOE) for tuning of the genetic algorithm parameters in scheduling problem. They considered a single-machine scheduling problem with the objective of minimizing weighted tardiness. They used multiple design of experiment methods to tune the parameters effectively. The results showed that D-optimal and Signal-to-noise (S/N) ratio designs provided the best parameter setting. Wang et al. (2017) also used GA to solve JSSP. To enhance the GA's solution quality, they used the DOE method to identify the parameters with the highest impact on the problem. Then, they established the approximation model and optimized it to get the optimal solution. The study by Mobin, Mousavi, Komaki & Tavana (2018) proposed an approach for parameter tuning in the evolutionary algorithm which simultaneously optimizes all performance metrices of evolutionary optimization algorithm. They used DOE to find the significant parameters of the evolutionary algorithms, as well as an approximate equation for each parameter metric. They considered two multi-objective evolutionary algorithms: multi objective particle swarm optimization algorithm (MOPSO) and fast non-dominated sorting genetic algorithm (NSGA-III). They validated their approach by optimizing performance in solving single machine scheduling problem to minimize makespan, total completion time and total tardiness time. The other study tackles the unrelated parallel machine scheduling problem with sequence-dependent setup times and machine eligibility constraints, aiming to minimize the maximum completion time. Utilizing a lean hybrid genetic algorithm enhanced with a targeted local search operator, the study demonstrates significantly improved performance over other methods, particularly in larger instances, highlighting the critical role of calibration for representative instances (Adan, 2022). Other reseaech proposes a two-stage genetic algorithm, coupled with optimal computing budget allocation (OCBA) and improved Monte-Carlo Policy Evaluation (MCPE), to address a stochastic parallel machine scheduling problem in just-in-time manufacturing, where processing time follows gamma or log-normal distribution. The developed method outperforms existing optimization algorithms, showcasing its applicability in practical systems like semiconductor manufacturing (Cao, Lin, Zhou, Zhou, & Sedraoui, 2023).

While many studies have focused on unrelated parallel machine scheduling problems and employed design of experiments (DOE) to optimize the performance of metaheuristics, none have specifically applied DOE for tuning parameters of the Genetic Algorithm (GA) within the context of minimizing total earliness and tardiness costs considering maximum allowable tardiness. Our research fills this gap by specifically focusing on parameter tuning of GA in this unique problem context. Moreover, we enhance the GA by integrating construction and improvement heuristics to generate an improved initial solution, which expedites the convergence of the GA to the near-optimal solution. While OptQuest solving engine has been employed in various scheduling problems, it has not been directly compared with an enhanced GA approach in this specific context. Our research provides this comparative analysis, offering insights on the performance of our proposed GA in relation to OptQuest's integrated methods. Hence, this study advances understanding of GA parameter tuning within the complex environment of unrelated parallel machine scheduling while providing a valuable comparison for optimization method selection.

1.1. Problem statement

Delivering high-quality products as close to the due date as possible while preserving timely schedules against underlying uncertainties and fierce global competition is the primary difficulty in job shop contexts. The problem in this research is inspired by real-case industrial demands that minimize total earliness and tardiness in unrelated parallel machine scheduling. The problem is formulated using a Mixed Integer Linear Programming (MILP) model. It encompasses N independent jobs and M machines. Each job, indivisible and requiring one machine, has pre-determined processing times and due dates. Processing times differ per machine and are mutually independent, and due dates are unique. Setup time, though excludable, can be incorporated within the processing time. In the scheduling problem, an objective function minimizes total earliness and tardiness. Constraints ensure that jobs are processed by only one machine in a specific sequence, starting with the initial job. Start and completion times are set, and machine completion times are determined based on job processing times. The earliness and tardiness of each job are calculated by comparing its completion time with the due date. The comprehensive mathematical model, inclusive of a detailed objective function and constraints, is elaborated in (Kianpour, Gupta, Krishnan, & Gopalakrishnan, 2021)

In response, the study suggests an Enhanced Genetic Algorithm (E-GA) that is specifically created to reduce both earliness and tardiness costs in the challenging environment of unrelated parallel computers while simultaneously complying to maximum tardiness limitations. This study makes use of the Design of Experiment (DOE) idea to optimize GA settings in order to guarantee quick solutions and informed decision-making. Additionally, it adds hybrid dispatch rules to improve the original solutions, substantially boosting the effectiveness and performance of our E-GA in this difficult scheduling situation.

2. Genetic Algorithm in Unrelated Parallel Machine Scheduling Problem

GA is classified as the evolutionary algorithm based on natural genetic evolution. Darwin investigated the original study on natural evolution in 1859. He claimed that natural populations evolve based on natural selection per “survival of the fittest” in his research.

The idea of using GA in optimization problems to find an optimal or near-optimal solution was first proposed by Holland in the early 1970s (Taylor 1994), which Goldberg then expanded in the late 1980s (Goldberg 1989). The process of natural genetic evolution begins with identifying the fittest individuals from the population. Then, the next generation will be created by producing offspring based on the characteristics of the parents. The offspring from the fit parents will have better fitness than their parents and consequently a better chance of survival. The algorithm keeps creating a new generation until a generation with the fittest individuals is found.

The GA includes five main phases initial population, fitness function, selection, crossover, and mutation. For unrelated parallel machine scheduling problem, the assignment's initial population includes assigning the jobs to the machines with their sequence on that particular machine. The initial population plays a critical role in determining the quality of the solution, in addition to computational time. In this paper, at first, the initial solution was generated randomly. Then, a hybrid dispatch rule has been proposed to improve the initial solution and, consequently, the efficiency and effectiveness of the algorithm. The fitness function is the same as the objective function in the mathematical modeling (e.g., makespan, earliness, tardiness, etc.). This paper uses the total earliness (inventory cost) and tardiness costs (penalty cost) as the fitness function. There are multiple techniques for selection in unrelated parallel machine job shop scheduling problem. Roulette wheel selection, rank selection, steady-state selection, stochastic universal sampling, etc., are frequently used techniques for selection (Nasr et al. 2015). In this paper, we used a steady-state approach. This means that only one solution is replaced at a time instead of the entire generation. When a new solution is generated, two parents are selected from the current population (solutions with high fitness function values are more likely to be selected as parents). The crossover operator has a significant effect on the performance of the GA. The role of the crossover operator is to combine the features of the two chromosomes simultaneously to generate offspring. There are multiple crossover operators presented in the literature (e.g. order crossover, cycle crossover, position-based crossover, etc.) (Cheng, Gen, & Tsujimura 1999).

In the context of the unrelated parallel machine scheduling problem, a genetic algorithm (GA) is employed to find a solution that minimizes earliness and tardiness. The GA utilizes chromosomes to represent potential solutions using binary string encoding. The variables associated with the problem, such as job sequences, start times, and machine assignments, are encoded within the genes of the chromosomes. For instance, the order of tasks can be represented by integers, while start times and machine assignments can be represented by real numbers or integers. To evaluate the fitness of each chromosome, an objective function is utilized that quantifies the degree of earliness and tardiness in the scheduling. This objective function incorporates the specific constraints and requirements of the problem. Through the application of genetic operators, the GA evolves and refines the chromosomes across generations. The selection operator favors chromosomes with higher fitness, allowing them to be chosen as parents for generating the next generation. Crossover then takes place, exchanging genetic information between parent chromosomes to create offspring with new combinations of genes. This process enables the exploration of different solutions and promotes the exploitation of promising regions in the search space. To maintain diversity and explore new areas, the mutation operator introduces random changes in the genes of the chromosomes. By iteratively applying these genetic operators and evaluating the fitness of the resulting chromosomes, the GA progressively improves the solutions. This iterative process continues until a satisfactory or optimal solution is reached, minimizing earliness and tardiness in the unrelated parallel machine scheduling problem.

In this paper, we have used a uniform crossover routine (two children are created by randomly selecting genes in one group or another). This operator chooses genes randomly from one parent, finds their position in the other parent, and copies the remaining genes into the second parent in the same order as they look in the first parent. By applying uniform crossover, we avoid biasing the search with the irrelevant position of variables which may happen in single-point or double-point crossovers. The last GA operator is a mutation, which is used to make slight changes in chromosomes to keep the diversity of the population. Soni & Kumar (2014) provided different types of mutation operators as inversion, insertion, shift mutation, etc. This paper provides a random number between 0 and 1 for each variable. If the variable gets a number less than or equal to the mutation rate, that variable is mutated.

3. Proposed Approach

Mixed-integer linear programming formulation to solve such problems was presented in the model by (Kianpour, Gupta, Krishna, & Gopalakrishnan, 2021). It has been further studied by (Kianpour, Gupta, Krishna, & Gopalakrishnan, 2022) to include uncertainties. The objective of the model is to minimize earliness, and tardiness costs subject to job assignment, machine assignment, and sequencing constraints. Since the model is categorized as an NP-hard problem, it cannot solve medium or large-size issues in a timely manner.

Consequently, we proposed an enhanced heuristic algorithm (GA) to solve large-scale problems. The heuristic algorithms can be classified into construction and improvement heuristics. In construction heuristics, we start without the schedule and start to add one job at a time. In improvement heuristics, we begin with the initial schedule and try to find a better schedule. Dispatching rules are examples of the construction heuristics, while local search methods such as Simulated Annealing, Tabu Search, and GA are examples of the improvement heuristics. The literature review indicates that multiple factors can affect the performance of the GA algorithm in terms of the quality of the solution and computational time. These factors include but are not limited to population size, crossover probability, mutation rate, and the quality of the initial solution. In this paper, we combined construction and improvement heuristics to improve the quality of the initial solution. Also, we have tuned the GA operators using design of experiment to get to the near-optimal solution in less time.

3.1. Hybrid dispatch rule to generate the initial solution

The first step in implementing the GA algorithm is generating an initial population. Studies show that good initial solutions help GA get to the good solution faster (Brucker 2007; Zitzler, Deb, & Thiele 2000). Also, if the initial solution is not good or large enough, GA would have difficulty finding a good solution. The random generation of the initial solution will affect the search space and selection process, which consequently increases the problem difficulty. This research proposed the hybrid dispatching rule to provide the initial solution. There are multiple ways to dispatch the jobs on the machines in the job shop environment. Jobs can be assigned based on their due dates, processing times, or arrival times to the shop. A sample list of dispatching rules in the job shop environment is presented in (Shahzad & Mebarki 2016). This study combined the shortest processing time (SPC) with the highest penalty cost to dispatch jobs on the machines. In this process, we assign jobs to the machines using this strategy based on processing time and maximum earliness and tardiness unit costs. In the first step, we assign the job with the highest earliness/tardiness cost to the machine with the shortest processing time. We repeat the first step until all machines have at least one job to complete in the second step. Then in the last step, we repeat the first two steps until all jobs are assigned to the machines. The implementation process will be explained in detail in section 4.2.

3.2. Identifying the best set of the ga parameters

Generating the new population in the GA algorithm is highly dependent on the GA operators, such as population size, crossover rate, and mutation rate. A suitable parameter setting affects the performance of the algorithm. Selecting an appropriate parameter tuning technique for each specific optimization problem should be considered. There are various ways in the literature to identify the best parameter setting for the GA algorithm, such as one-factor-at-a-time (OFAT) (Daniel 1994). But the disadvantage of such methods is that they don't consider the interaction between parameters which may change the whole solution process. To overcome this drawback, the design of experiment (DOE) has been widely used to consider the interaction of parameters (Li et al. 2009). The DOE process includes identifying the key parameters, transforming the input parameters if needed to find the best combination of them, analyzing the best relationship between input parameters and responses, constructing the empirical formula, and developing an approximation model. Once the significant factors have been identified and the approximation model has been built, the next step is to determine the best set of these parameters to achieve the desired objective. A detailed introduction to the design of experiment is provided in (Goupy & Creighton 2007). In this paper, we used the DOE approach to identify the significant parameters on the performance of the GA and optimized those parameters to minimize the total cost and computational time simultaneously. The details of the implementation process will be provided in section 4.1. A detailed flow chart is shown in Figure 4.

4. Case Studies from Local Job Shops

In this section, we first optimize the GA operators (e.g., population size, crossover rate, and mutation rate) using DOE. Then, we propose the improved initial solution for the GA using the hybrid dispatching rule and compare results with the known dispatching rules in the literature. Finally, we compare our enhanced GA with the OptQuest algorithm, a powerful search engine that integrates Tabu Search, Neural Network, Scatter Search, and Linear/Integer Programming into a single composite method. In this paper, the GA algorithm is coded by visual basic language in the Excel platform, and the Evolver solver is used to solve the problem. The GA algorithm's stoppage criteria is no more than 0.01% difference between two consecutive optimal solutions for 50,000 trials. All computations are carried out on an Intel ® Core ™ 3.20 GHz with 8.00 GB RAM.

4.1. Hybrid dispatch rule to generate the initial solution

To optimize the GA, six different problem sizes with random combinations of jobs and machines have been considered. The details of the problems are provided in table 1.

Table 1
Problem sizes for GA operator's optimization

No.	# of Jobs	# of Machines
1	9 Jobs	5 Machines
2	10 Jobs	2 Machines
3	18 Jobs	7 Machines
4	25 Jobs	10 Machines
5	31 Jobs	13 Machines
6	40 Jobs	20 Machines

The processing times of the jobs on the machines, due dates of each job, and unit inventory cost and penalty cost are different for each problem size. The two-level factorial design has been used to run the experiment in which population size, crossover rate, and mutation rates are factors while each factor gets minimum and maximum value. The response parameters in this experiment are total cost and computation time. The details of the factors and their levels are presented in table 2. Each experiment has been run five (5) times; therefore, the total number of experiments for each problem size is 40. In this step, the initial solution for the GA is provided randomly. In the first step of the GA operators' optimization, we need to identify if the residuals are randomly and normally distributed around zero. If not, then the transformation is required to achieve normality before using some form of the general linear model.

Table 2
List of factors and their levels

Factor	Min Level	Max Level
A: Population Size	500	2,000
B: Crossover Rate	0.25	0.75
C: Mutation Rate	0.05	0.2

With the number of levels as (L), the number of factors as (F), and the number of runs as (R), the total number of experiments (Ex_t) for each problem size is calculated as follows:

$$Ex_t = L^F \times R = 2^3 \times 5 = 40 \tag{1}$$

In this paper, the normal plot of the residuals is used to test the normality of the data. The plot with residuals versus predicted responses is used to indicate the existence of a pattern in the data. Finally, the Box-Cox plot is used to identify which transformation type is needed (e.g., square root, natural log, inverse, power, etc.). This study selected only the main factors and two-factor interactions to approximate the model (A,B,C,AB,AC,and BC). ANOVA is used to assess the significance of each factor and their interactions (a p-value less than 0.0500 indicates that the factor's effect is significant). Then, we calculate the model coefficients, and finally, the approximation model is built to be used in the optimization process. A summary of the DOE process for the studied problem sizes is presented in table 3. Problem number 3 in table 1 (10 jobs and 2 machines) provide the same cost as the exact model regardless of the population size, crossover rate, and mutation rate. The GA operators, in this case, only affect the computational time and convergence process to get to the optimal solution. Table 3 evaluates the cost model and the computational time model separately in terms of transformation type, a significant factor, and positive and negative factors. Negative factors have a negative effect on the cost and computational time, while positive factors have a positive effect on the total cost and computational time. For instance, in the case of 40 jobs and 20 machines, mutation rate (factor C) and population size (factor A) are considered negative factors. In other words, the increase in mutation rate and population size will increase the total cost, which is against the objective of this optimization problem. Therefore, they affect the objective function negatively. In this case, the effect of factor C is more significant than A.

Table 3
The summary of the DOE analysis

Problem Size	Response Variable	Transformation	Significant Factors	Negative Factors	Positive Factors
$\mathbf{9 \times 5}$	Cost	-	-	C-AB-A-AC	B, BC
	Time	Natural Log	A	A-AC-C-B	AB-BC
$\mathbf{10 \times 2}$	Cost	Inverse	-	-	-
	Time	Natural Log	-	AC-A-BC	B-C-AB
$\mathbf{18 \times 7}$	Cost	-	-	AB-A-C-BC	B-AC
	Time	Natural Log	A	A-B-C	AB-BC-AC
$\mathbf{25 \times 10}$	Cost	-	C-B-A	C-A-BC-AC	B-AB
	Time	Natural Log	C	B-A-AC-AB	C-BC
$\mathbf{31 \times 13}$	Cost	Inverse	C-B-A	B-AC-AB	C-A-BC
	Time	-	C	A-B-AC	C-BC-AB
$\mathbf{40 \times 20}$	Cost	Natural Log	C-A-B	C-A-BC	B-AC-AB
	Time	Square Root	C	C-BC	B-AC-A-AB

*A: population size; B: crossover rate; C: mutation rate

The general structure of the approximation model is presented as follows:

$$M = \varepsilon + \alpha \times A + \beta \times B + \gamma \times C + \rho \times AB +$$

$$\sigma \times AC + \delta \times BC \tag{2}$$

Where M can be either cost or time, based on the type of transformation, M can be different. For instance, in the case of 40 jobs and 20 machines, M equals to ln(cost) since the natural log transformation has been used for normalization. The coefficients (e.g., α, β, γ, ρ, σ and δ) are determined based on the experimental data. (A, B, C, AB, AC, and BC) represent the primary factors and interactions. In the optimization process, we search for the combination of factor levels that simultaneously satisfy the criteria placed on each factor or response. The approximation model provided by the analysis is used to include a response in the optimization criteria. The desired goal for each factor or response should be selected in this process. The options for desired goals include maximize, minimize, target, within range, none (for responses only), and set to an exact value (factors only). Each parameter, either factors or responses, should have a minimum or maximum level. Also, if we have a more important factor than the other ones, we can assign a higher weight to that factor. The goals are integrated into an overall desirability function. The program's objective is to maximize this function by beginning from the random starting point and proceeding up the steepest slope to the maximum. In the numerical optimization process, the factors (population size, crossover rate, and mutation rate) are set in range (they can get any value between their maximum and minimum level). The responses (total cost and computational time) are set to minimize. Figures 1-3 show the optimization results for problem No. 3. Figures 1.a, 1.b, and 1.c evaluate the effect of population size on the desirability, cost, and computational time respectively. The increase in population size from 500 to 2,000 reduces the desirability of the optimization process from 60% to less than 40%. Also, increasing the population size reduces the probability of getting the near-optimal solution. When the population size is 500, the value for minimum cost is approximately $14,500, while this number increases to $15,500 when the population size is 2,000. The same trend is observed for computational time. When the population size is 500, it takes approximately 250 seconds to solve the problem. This computational time doubles by increasing the population size from 500 to 2,000. Contrary to the general belief, this result indicates that providing a large population size does not necessarily lead GA to a good solution, and a smaller population size can provide a better solution in less time. However, the disadvantage of the smaller population size is that the algorithm might be easily trapped in local optimal. To overcome this issue, the other GA operators (crossover rate and mutation rate) should be optimized to help the algorithm with the exploration and exploitation process. Therefore, a population size of 500 will be the best population size for this problem.

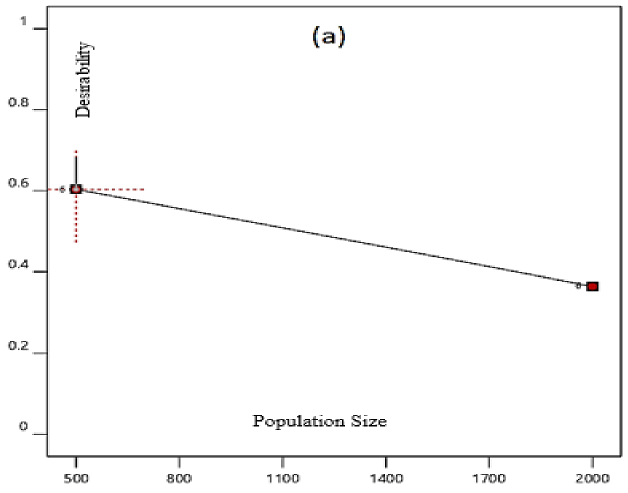


Fig. 1. (a) Population Size vs. Desirability

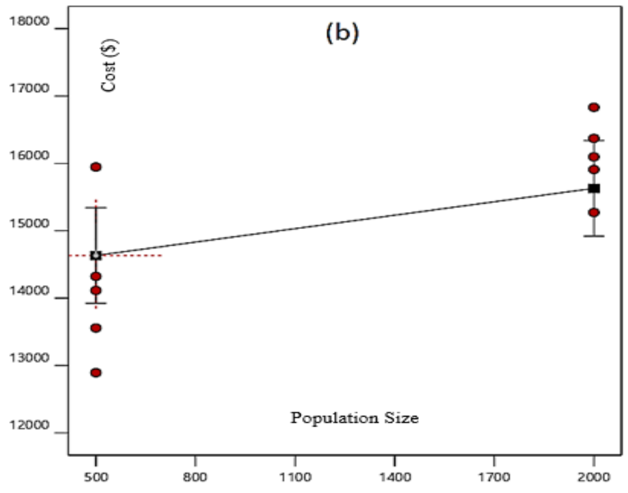


Fig. 1. (b) Population Size vs. Cost ($)

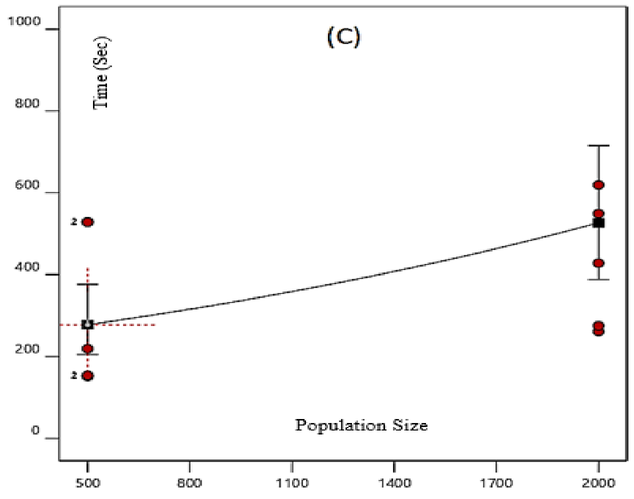


Fig. 1. (c) Population Size vs. Time (Sec)

The crossover rate does not affect the desirability significantly. However, increasing the crossover rate from 0.25 to 0.75 improves desirability (Figure 2.a). Also, increasing the crossover rate helps GA find the better near-optimal solution (Figure 2.b).

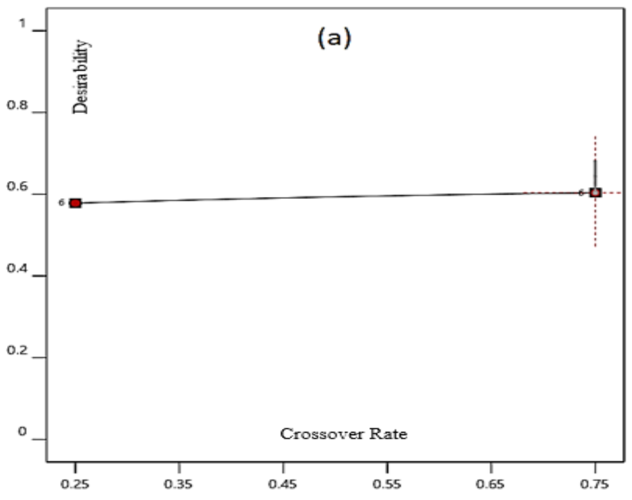


Fig. 2. (a) Crossover Rate vs. Desirability

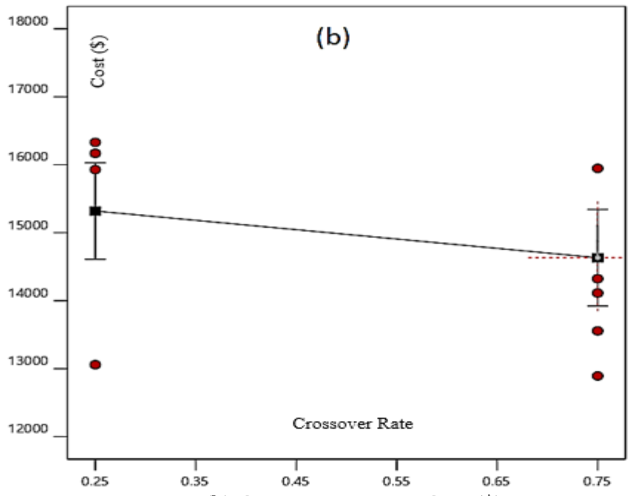


Fig. 2. (b) Crossover Rate vs. Cost ($)

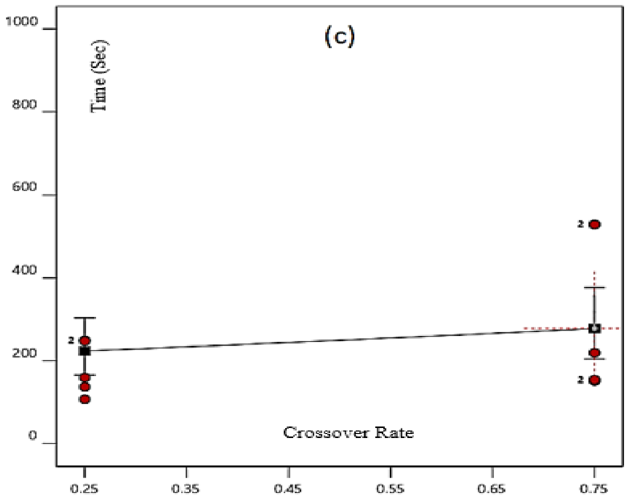


Fig. 2. (c) Crossover Rate vs. Time (sec)

The increase in the crossover rate will increase the computational time slightly (Figure 2.c). However, the main objective of this problem is reducing costs. Therefore, the slight increase in computational time can be ignored. Therefore, the crossover rate of 0.75 has been selected as the optimal crossover rate. The mutation rate does not affect desirability considerably.

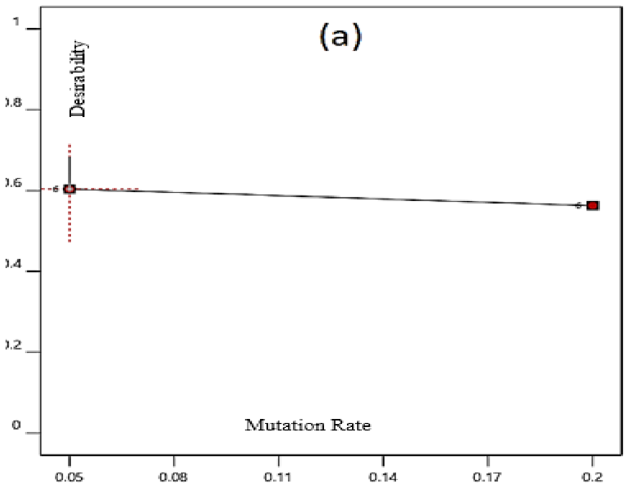


Fig. 3. (a) Mutation Rate vs. Desirability

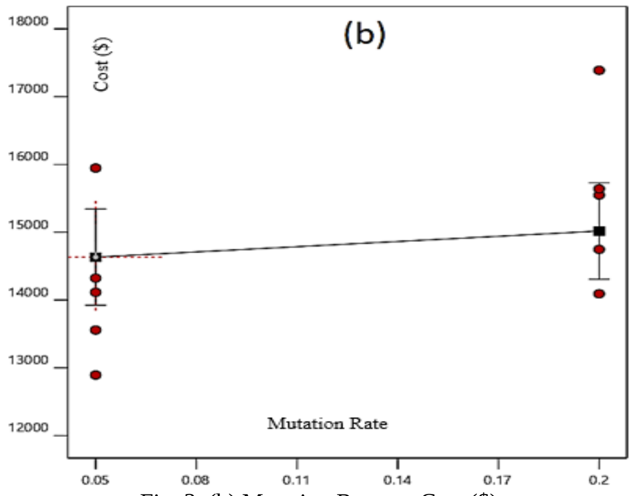


Fig. 3. (b) Mutation Rate vs. Cost ($)

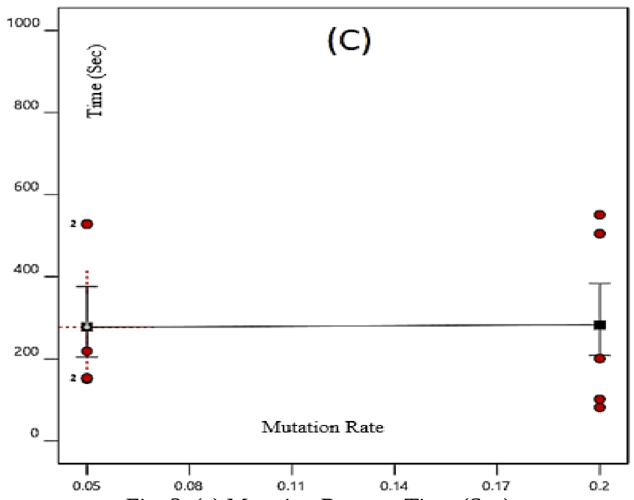


Fig. 3. (c) Mutation Rate vs. Time (Sec)

The increase in mutation rate from 0.05 to 0.2 will lead to a small reduction in desirability (Figure 3.a). However, a smaller mutation rate (0.05) helps GA find a better schedule than the larger mutation rate (0.2) as shown in

Figure 3.b. The goal of mutation is to explore the promising region provided by the crossover operator. Therefore, a large mutation can act against the crossover operator and push the algorithm away from the concentrated point provided by crossover to search other spaces. Also, the increase in mutation rate has no significant change in the computational time (Figure 3.a). As a result, the mutation rate of 0.05 has been considered as the optimal level for this problem. The same analysis has been done for problems #4, 5, and 6 in table 1, and the results are summarized in table 4. We optimized the GA operators to minimize total cost and computational time simultaneously until this point. However, the primary objective of this study is minimizing total cost. Therefore, numerical optimization has been implemented to optimize the GA operators with the single objective of the total cost. The results are provided in table 5. The problem with 10 jobs and 2 machines is excluded from the study since it provided the same total cost (equal to the cost provided by the exact method) in all scenarios. Based on the results in tables 4 and 5, the population size of 500, crossover rate of 0.75, and mutation rate of 0.05 have been selected to solve the scheduling problem in this paper.

Table 4
Optimal GA Operators

Problem Size	Desirability	Optimal Population Size	Optimal Crossover Rate	Optimal Mutation Rate
$\mathbf{10 \times 2}$	64%	500	0.75	0.2
$\mathbf{9 \times 5}$	64%	500	0.25	0.2
$\mathbf{18 \times 7}$	64%	500	0.75	0.05
$\mathbf{25 \times 10}$	58%	500	075	0.19
$\mathbf{31 \times 13}$	57%	500	0.75	0.19
$\mathbf{40 \times 20}$	52%	500	0.75	0.14

Table 5
Optimal GA Operators

Problem Size	Desirability	Optimal Population Size	Optimal Crossover Rate	Optimal Mutation Rate
$\mathbf{9 \times 5}$	71%	2000	0.75	0.2
$\mathbf{18 \times 7}$	65%	500	0.75	0.05
$\mathbf{25 \times 10}$	80%	500	075	0.05
$\mathbf{31 \times 13}$	69%	500	0.75	0.05
$\mathbf{40 \times 20}$	82%	500	0.75	0.05

4.2. Improving the initial solution for the GA

In this section, we combined the shortest processing time with the maximum penalty or inventory cost (as applicable for late and early jobs, respectively) to provide the hybrid dispatch rule, which is used as the initial solution in the GA. In this method, we assigned the jobs on the machines based on their processing time and maximum earliness and tardiness unit costs. For instance, if we have 10 jobs to be assigned to six (6) machines, we assign the job with the maximum earliness/tardiness unit cost to the machine with the minimum processing time. We continue this cycle until all machines have at least one job to process. Then, we start to assign the second round of jobs to the machines following the same rule. We continue this process until all jobs have been assigned. The algorithm to determine the initial solution is illustrated in figure 4. In the case of a tie, random selection is made from the jobs or machines that are tied with respect to their rank or least jobs/processing times, respectively.

In order to create a starting solution for the problem with 31 jobs and 13 machines, We rank the jobs based on their maximum earliness (inventory) or tardiness (penalty) cost in the first step. The ranking is presented in table 6. Then, we need to follow the rest of the steps (note: in the second step, if the machine with the least processing time has already been assigned, we pick the machine with the next least processing time). The assignment of the jobs on the machines based on the proposed method is presented in table 7. Table 7 represents the initial solution for GA while using optimized operators given in the previous section.

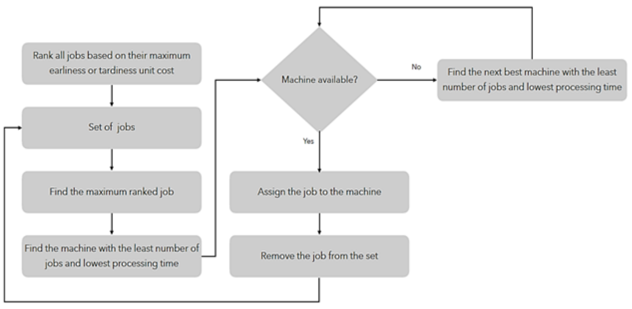

Rank all jobs based on their maximum earliness or tardiness unit cost
Set of jobs
Find the maximum ranked job
Find the machine with the least number of jobs and lowest processing time
Machine available?
No
Find the next best machine with the least number of jobs and lowest processing time
Yes
Assign the job to the machine
Remove the job from the set

Fig. 4. The summary of the hybrid dispatch rule algorithm

The same concept has been applied to the other problem sizes. The proposed hybrid dispatch rule was compared with other dispatch rules in literature, such as shortest processing time (SPT), longest processing time (LPT), biggest weight (WI), earliest release date (ERD), and earliest due date (EDD). The biggest weight in this study is the maximum of either inventory or penalty unit cost of the job. Each approach was run five (5) times, and results are compared in terms of the mean optimal cost, percentage cost deviation, and mean computational time. The percentage cost deviation is the average of the absolute difference between the results of each run with the minimum cost provided by the GA in that problem size. In this process, the proposed model was compared with cases with random initial solution (RI) and random operators (RO), RI and best operators (BO), SPT, LPT, WI, ERD, and EDD. Figures 5-7 evaluate five (5) different problem sizes in terms of the mean of the minimum cost, %Cost deviation, and computational time.

Table 6
The ranking of the jobs

Rank	Job	Rank	Job	Rank	Job	Rank	Job
1	4	10	25	19	19	28	6
2	7	11	2	20	9	29	31
3	20	12	10	21	27	30	26
4	12	13	1	22	28	31	30
5	17	14	22	23	29		
6	18	15	5	24	3		
7	24	16	21	25	8		
8	13	17	11	26	14		
9	23	18	16	27	15		

Table 7
Assignment of the jobs on the machines

Machine	1	Sequence 2	Sequence 3
1	J20	J16	J31
2	J13	J21	J30
3	J25	J28	-
4	J1	J3	J26
5	J12	J8	J15
6	J10	J11	-
7	J2	J19	-
8	J17	J5	-
9	J4	J14	-
10	J24	J27	-
11	J23	J9	-
12	J18	J22	J6
13	J7	J29	-

Figure 5 compares the optimal total cost provided by the schedule from the proposed model with the cost of the schedules provided by other dispatch rules in the literature. In all the studied problem sizes, the proposed model provided the schedule with the least earliness and tardiness costs compared to the other methods. The shortest processing time (SPT) ranked second in 80% of the studied problems since it is used in the development of the proposed hybrid method. As we expected, the random initial solution with random operators (RI-RO) provided the worst optimal cost in 60% of the cases. This method offers almost the same cost with random initial-optimized (best) operators (RI-BO) in two smaller size problems. There was no significant difference between the costs provided by ERD and EDD, which are slightly better than the costs provided by WI. One key performance indicator for heuristic algorithms is the deviation of the results provided in different runs. Less deviation shows that the difference in the results provided by solving the same problem is negligible, so it would not be necessary to run the algorithm multiple times. Based on Figure 6, the proposed algorithm provides results with about 1% average cost deviation (with no deviation for small size problems), which is at least five times lower than the %cost deviation of the other methods. Although reducing the computational time is not the primary objective of this paper, the results indicate that the proposed method provides a better solution with lower computational time (Figure 7). On average, the computational time of the proposed methods is 70% less than the computational time provided by all the other methods.

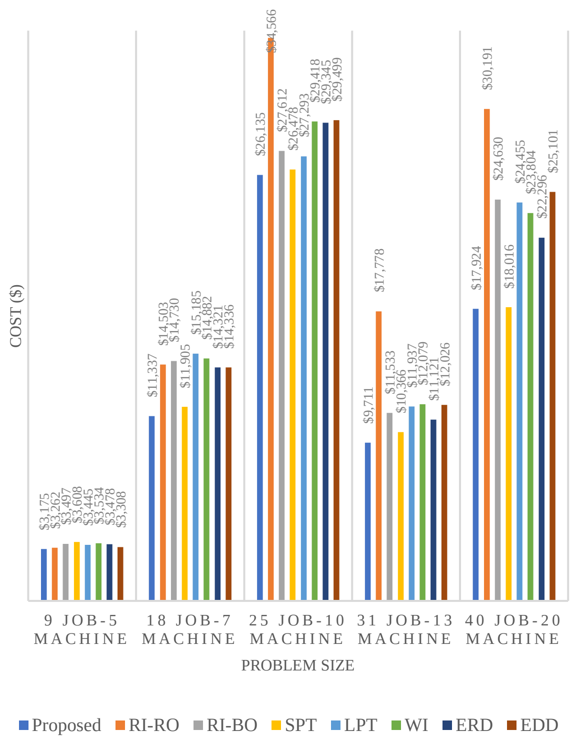


Fig. 5. Comparison of the minimum cost provided by the proposed method with other

In general, the results show that the initial solution improves the efficiency and effectiveness of the GA drastically. Also, the optimized population size helps GA search for the solution in a smaller area, reducing the computational time required to find the near-optimal solution. Moreover, the optimized crossover and mutation rate help the convergence of the algorithm, which improves the quality of the solution and reduces computational time.

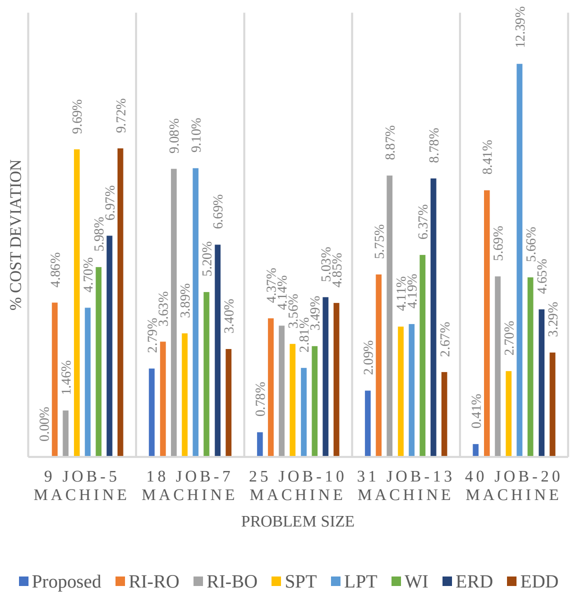
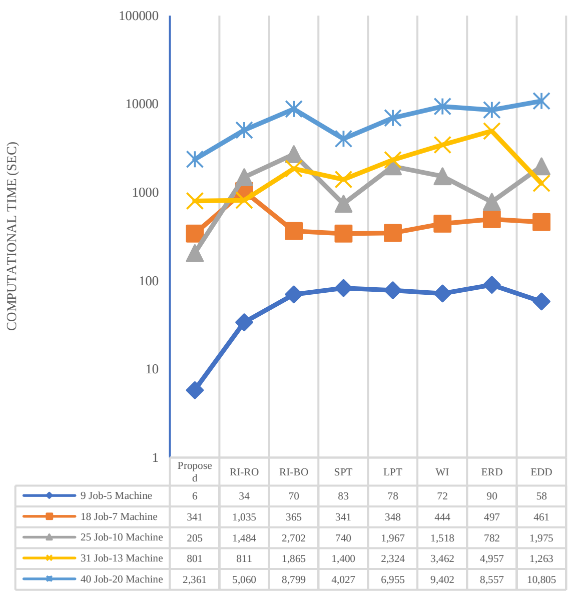


Fig. 6. Comparison of the % cost deviation provided by the proposed method with others



	Proposed	RI-RO	RI-BO	SPT	LPT	WI	ERD	EDD
9 Job-5 Machine	6	34	70	83	78	72	90	58
18 Job-7 Machine	341	1,035	365	341	348	444	497	461
25 Job-10 Machine	205	1,484	2,702	740	1,967	1,518	782	1,975
31 Job-13 Machine	801	811	1,865	1,400	2,324	3,462	4,957	1,263
40 Job-20 Machine	2,361	5,060	8,799	4,027	6,955	9,402	8,557	10,805

Fig. 7. Comparison of the computational time provided by the proposed method with others

4.3 *Comparing proposed E-GA with OptQuest*

In this section, the performance of the enhanced GA method (E-GA) is compared with OptQuest, which is defined as a unique set of powerful algorithms and sophisticated analysis techniques. In the first step, the cost of the schedule provided by the E-GA method is compared with the cost from OptQuest in two cases. In the first case, we use the proposed hybrid dispatch rule to provide the initial solution for the OptQuest algorithm (P-OptQuest). In the second case, the initial solution is selected randomly (R-OptQuest).

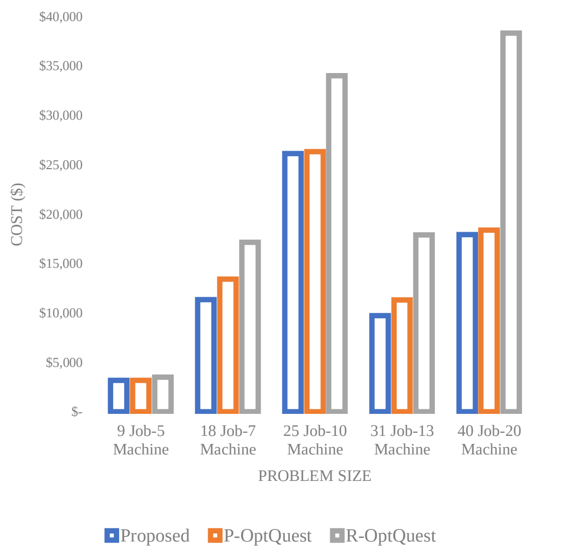


Fig. 8. Comparison of the minimum cost provided by E-GA and OptQuest

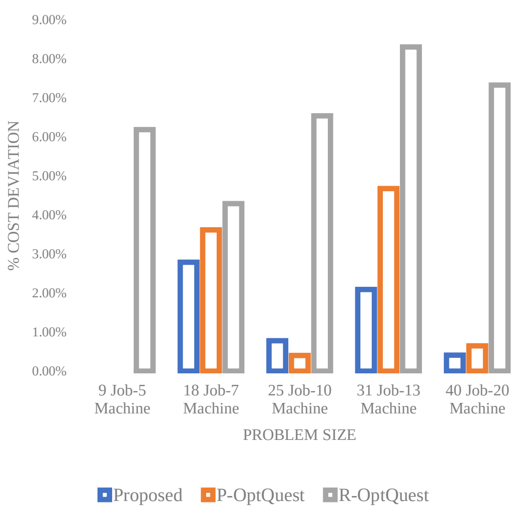


Fig. 9. Comparison of the % cost deviation provided by the E-GA and OptQuest

Comparative results are illustrated in Figure 8. The results show that E-GA and OptQuest perform the same for the small size problem (9J-5M). However, E-GA provides a better solution (schedule with minimum cost) for other problem sizes. The study shows that the schedule provided by E-GA can reduce the total cost by up to 15% and 46% compared to P-OptQuest and R-OptQuest, respectively. In terms of % cost deviation, both E-GA, and P-OptQuest provide zero deviation for the small size problem. For the rest of the problem sizes, E-GA provides lower deviation compared to P-OptQuest and R-OptQuest (average deviation of 1.21% for E-GA compared to 1.97% and 6.53% for P-OptQuest and R-OptQuest, respectively). Figure 10 shows that P-OptQuest converges faster compared to E-GA in 80% of the studied problems. However, the solution quality provided by the E-GA is better (schedule with lower earliness and tardiness costs).

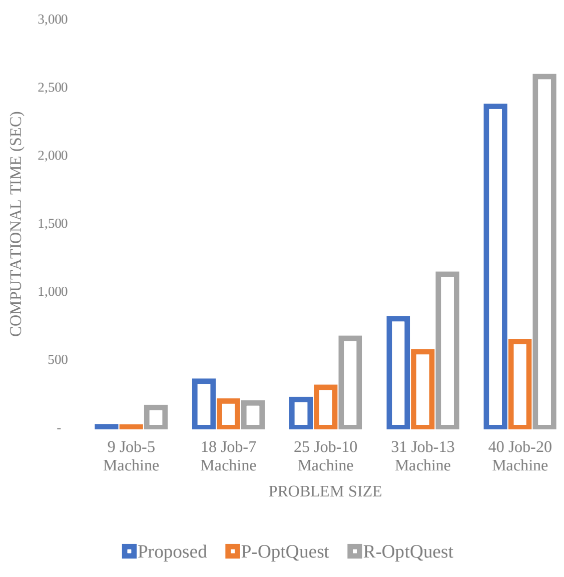


Fig. 10. Comparison of the computational time provided by the E-GA and OptQuest

To compare the computational time performance between E-GA and P-OptQuest, the analysis has been implemented to identify how long it would take for E-GA to provide the same cost provided by P-OptQuest. The small size problem and the problem with 25 jobs and 10 machines are excluded since the difference for the small size problem is negligible, and E-GA provides the best schedule in less time for 25J-10M problem. For other problems, the analysis results are illustrated in Figure 10. Figure 11 compares how long it would take for E-GA and P-OptQuest to provide the schedule with the costs of $13,408, $11,326, and $18,382 provided by OptQuest for problems sizes 18J-7M, 31J-13M, and 40J-20M, respectively.

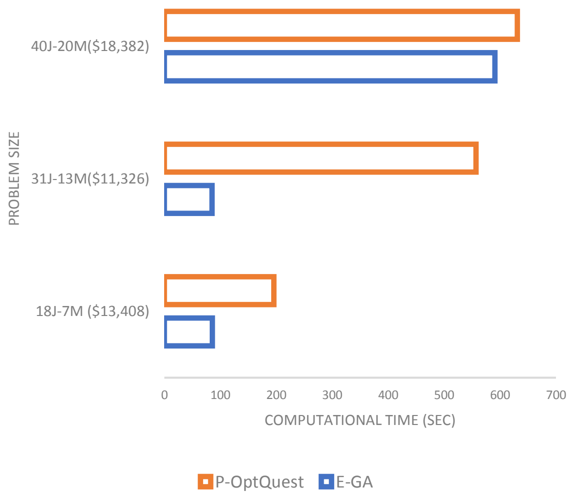


Fig. 11. Comparison of the computational time provided by the E-GA and P-OptQuest

The results indicate that for problem size 18J-7M, it takes 56% less time for E-GA to provide the schedule with a cost of $13,408. This number is 85% and 6% for 31J-13M and 40J-20M problems, respectively. Since the primary objective of this research is to minimize the total cost than computational time, E-GA is deemed most effective since it provides the schedule with the lowest total earliness and tardiness costs.

4.4. Computational complexities of the algorithm

This study provides a runtime versus the number of inputs (n) plot as a scale to measure the efficiency of the proposed algorithm. This plot originated from the big O notation concept used in computer science to describe the performance or complexity of an algorithm. According to big O notation, the algorithm can run in constant time (regardless of the number of inputs), linear time, quadratic time, etc. The list of common types of big O notation is provided in (Danziger, 2010).

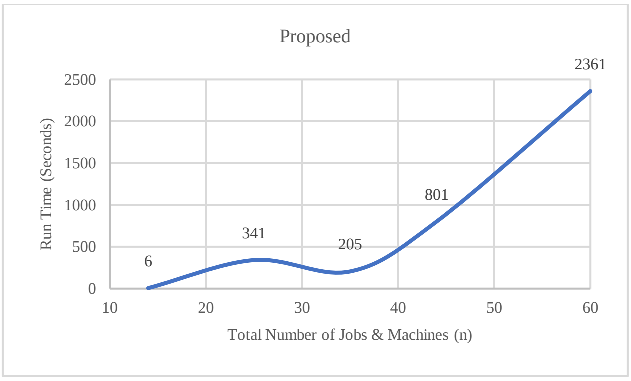


Fig. 12. Run Time vs. n for Proposed Model

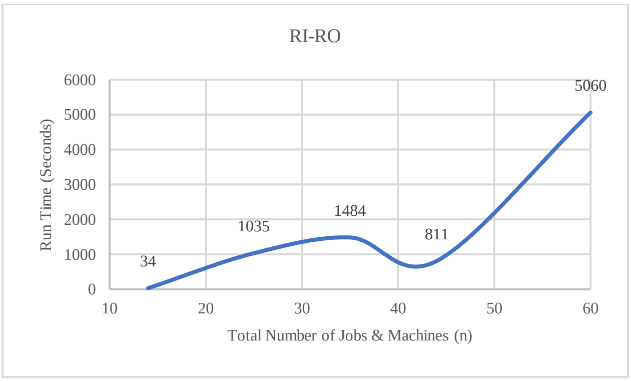


Fig. 13**.** Run Time vs. n for RI-RO Mode

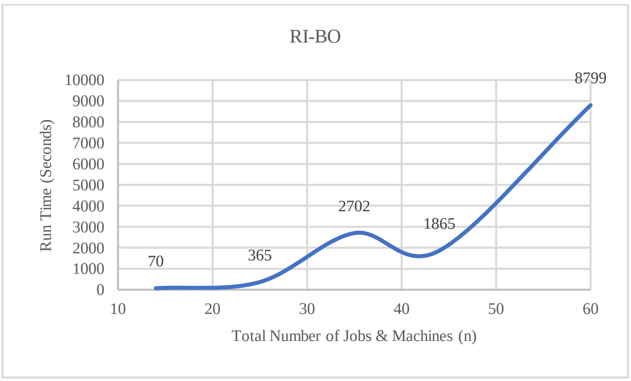


Fig. 14. Run Time vs. n for RI-BO Model

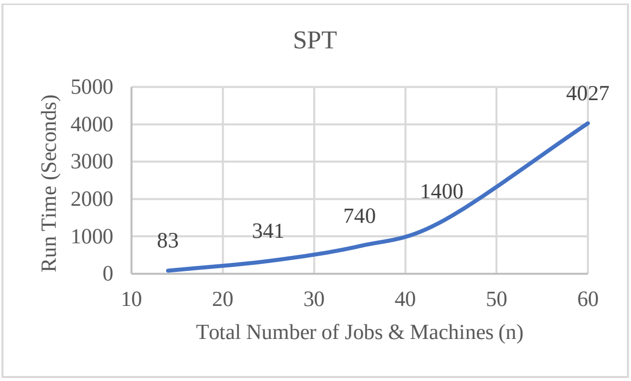


Fig. 15. Run Time vs. n for SPT Model

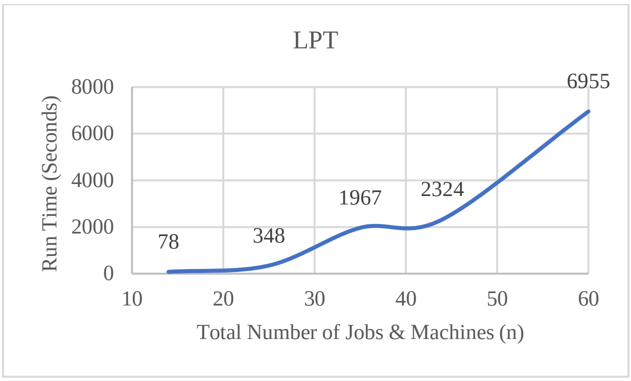


Fig 16. Run Time vs. n for LPT Model

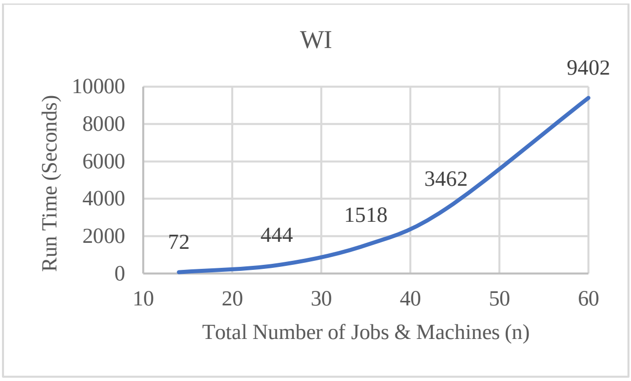


Fig. 17. Run Time vs. n for WI Model

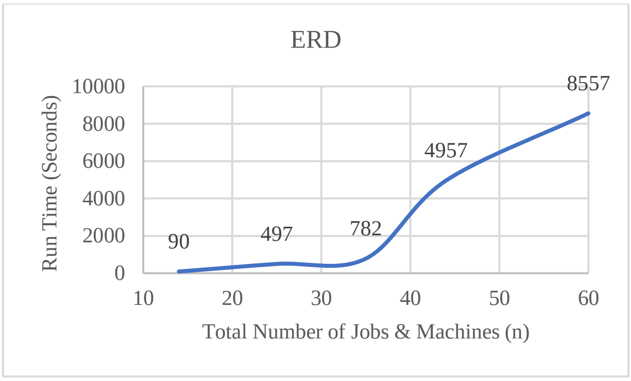


Fig. 18. Run Time vs. n for ERD Model

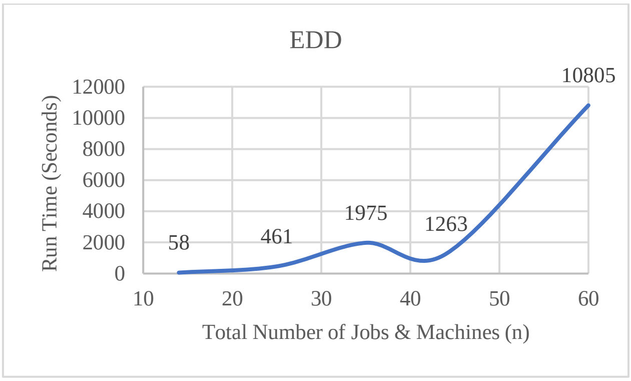


Fig. 19. Run Time vs. n for EDD Model

Figures 12-19 show the runtime vs. input plots for the proposed algorithm and the other methods in this study. Assuming the linear relationship between the total number of jobs & machines and runtime (average R2 = 0.81), we see that the proposed model provides the most efficient algorithm since it provides the minimum coefficient for n among all other compared methods in the literature. The summary of the results is provided in table 8. Applying other methods, such as exponential and second-order polynomial, validated that the proposed algorithm outperforms other studied algorithms in terms of efficiency.

Table 8
Coefficient of n (total number of jobs & machines) for different algorithms

Method	Proposed	SPT	RI-RO	LPT	RI-BO	ERD	WI	EDD
Coefficient of n	48.5	83.6	94.5	146.7	179.4	194.9	201.8	214.4

4.5. Summary and discussion of the results

Genetic Algorithm has been used extensively to obtain an optimal and near-optimal solution for NP-hard optimization problems, which can take a lifetime to solve. The most important GA parameters based on function-based biology-inspired concepts are initial population, population size, crossover, and mutation operators. To incorporate those parameters to obtain the optimal or near-optimal solution, this study followed the basic GA structure presented (Kumar, 2019) in figure 20. The numerical results provided in this study indicated that the selection of different GA parameters and their interaction directly affects the quality of the solution. Providing controlled instead of random initial solutions will reduce search time to identify an optimal solution and decrease the probability of resulting in infeasible solutions since it approximates where the minimal points for a function lie. In this study, the hybrid dispatch rule was proposed to provide a feasible solution by taking unit earliness/tardiness cost and processing time of the jobs into account. However, using the intended initial solution may decrease the exploration capacity of the GA and may lead to a local optimum rather than a global optimum. This problem has been solved by using optimized crossover and mutation operators. An optimized crossover operator was used to explore the search space by selecting the pair of individuals among those who survived from the previous generation and then generating offspring that inherits valid characteristics of the two parents. As crossover proposed better solutions, a mutation was used to make solutions closer to the best solution. In other words, the mutation operator was used to provide exploration, and the crossover operator was used to lead the population to converge on one of the good solutions found so far (exploitation).
Consequently, while crossover tried to converge to a specific point in the landscape, the mutation did its best to avoid convergence and explore more areas. Their probabilities impact the effect of these parameters. For instance, the probability rate of 100% for crossover provides a completely different result compared to the probability rate of 50%. The same rule applies to mutation probability as well. Therefore, maintaining the balance between these parameters is one of the most significant factors in improving the GA's performance.

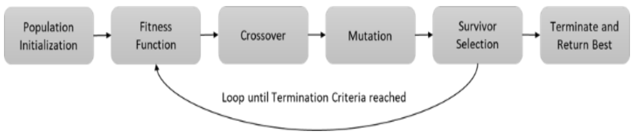

Population Initialization
Fitness Function
Crossover
Mutation
Survivor Selection
Terminate and Return Best
Loop until Termination Criteria reached

Fig. 20. The basic structure of the GA algorithm

5. Conclusion

This paper developed an enhanced GA method to solve the unrelated parallel machines job shop scheduling problem with a maximum allowable tardiness limit. In the first step, the numerical optimization method is used to tune GA operators (population size, crossover rate, and mutation rate) using design of experiment (DOE) methodology. The computational time of the DOE process has not been considered in the study since it was negligible. In the second step, the construction heuristics based on dispatch rules were combined with improvement heuristics. The hybrid dispatch rule was proposed to provide an initial solution for GA in order to help the efficiency and effectiveness of the algorithm's convergence. In the last step, the solution provided by E-GA was compared with OptQuest, which is one of the most powerful heuristic solvers. OptQuest uses advanced techniques like tabu search, scatter search, and neural networks to analyze different scenarios and find the optimal solution. Tabu Search uses memory structures to avoid cycling and to promote diversification in the search process, aiming to escape local optima to find better solutions. Scatter Search creates a diverse set of reference solutions as a base to generate potential solutions, making it effective for combinatorial optimization problems. Neural Networks, though not a traditional metaheuristic, can be integrated within a metaheuristic framework. Therefore, we can conclude that the E-GA was compared with the combination of the most advanced metaheuristics approaches and provided more satisfactory results overall. The method was implemented on data obtained from the local job shop. With a hybrid dispatch rule for the initial solution, the proposed E-GA works better in finding the near-optimal solution compared to other dispatch rules (on average, the solution provided by the hybrid dispatch rule leads to the schedule with 17% lower cost compared to other dispatch rules). Although OptQuest provides the results slightly faster, the quality of the solution provided by the proposed E-GA is better. Also, it is identified that, on average, E-GA would provide the same results which had been provided by OptQuest in 49% less time. The results provided by E-GA deviate 60% less than the results provided by P-OptQuest. Therefore, the proposed E-GA is recommended to solve the unrelated parallel machine scheduling problem. The proposed model can help decision-makers with scheduling jobs on the machines in an efficient and effective manner. The result of the proposed model provides an explicit view to production, budgeting, and financial managers. As a future research direction, uncertainty may be incorporated before solving the problem using E-GA or a similar heuristic method. Most of the jobs in the job shop under consideration are completed by only one machine. However, production cells, including multiple machines (with known processing sequences) may be considered future research opportunities.

References

Adan, J. (2022). A hybrid genetic algorithm for parallel machine scheduling with setup times. *Journal of Intelligent Manufacturing,* 33, 2059–2073.

Allahverdi, A. (2022). A survey of scheduling problems with uncertain interval/bounded processing/setup times. *Journal of Project Management,* 7, 255-264.

Athmani, M. E., Arbaoui, T., Mimene, Y., & Yalaoui, F. (2022). Efficient heuristics and metaheuristics for the unrelated parallel machine scheduling problem with release dates and setup times. *GECCO '22: Proceedings of the Genetic and Evolutionary Computation Conference*, 177-185.

Campo, E. A., Cano, J. A., Gómez-Montoya, R., Rodríguez-Velásquez, E., & Cortés, P. (2022). Flexible Job Shop Scheduling Problem with Fuzzy Times and Due-Windows: Minimizing Weighted Tardiness and Earliness Using Genetic Algorithms. *algorithms,* 15(10), 334.

Cao, Z., Lin, C., Zhou, M., Zhou, C., & Sedraoui, K. (2023). Two-Stage Genetic Algorithm for Scheduling Stochastic Unrelated Parallel Machines in a Just-in-Time Manufacturing Context. *IEEE Transactions on Automation Science and Engineering,* 20(2), 936-949.

Cheng, R., Gen, M., & Tsujimura, Y. (1999). A tutorial survey of job-shop scheduling problems using genetic algorithms, part II: hybrid genetic search strategies. *Computers & Industrial Engineering,* 36(2), 343-364.

Daniel, C. (1994). Factorial One-Factor-at-a-Time Experiments. *The American Statistician,* 48(2), 132-135.

De-Alba, H. G., Nucamendi-Guillén, S., & Avalos-Rosales, O. (2022). A mixed integer formulation and an efficient metaheuristic for the unrelated parallel machine scheduling problem: Total tardiness minimization. *EURO Journal on Computational Optimization,* 10, 100034.

Ɖurasević, M., & Jakobović , D. (2023). Heuristic and metaheuristic methods for the parallel unrelated machines scheduling problem: a survey. *Artificial Intelligence Review,* 56, 3181–3289.

Eiben, A. E., Hinterding, R., & Michalewicz, Z. (1999). Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation,* 3(2), 124 - 141.

Eiben, A. E., Michalewicz, Z., Schoenauer, M., & Smith, J. E. (2007). *Parameter Control in Evolutionary Algorithms. In: Lobo F.G., Lima C.F., Michalewicz Z. (eds) Parameter Setting in Evolutionary Algorithms. Studies in Computational Intelligence.* Berlin: Springer.

Fang, W., Zhu, H., & Mei, Y. (2022). Hybrid meta-heuristics for the unrelated parallel machine scheduling problem with setup times. *Knowledge-Based Systems*, 241, 108193.

Fanjul-Peyro, L. (2020). Models and an exact method for the Unrelated Parallel Machine scheduling problem with setups and resources. *Expert Systems with Applications: X,* 5, 100022

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and.* Addison-Wesley Longman Publishing Co.

Goupy, J., & Creighton, L. (2007). *Introduction to Design of Experiments with JMP Examples, Third Edition.* SAS Press.

Holland, J. H. (1975). *Adaptation in Natural And Artificial Systems.* Ann Arbor, Mich, USA: University of Michigan Press.

Huang, C., Li, Y., & Yao, X. (2019). A Survey of Automatic Parameter Tuning Methods for Metaheuristics. *IEEE Transactions on Evolutionary Computation*, 24(2), 201-216.

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

Kianpour, P., Gupta, D., Krishnan, K., & Gopalakrishnan, B. (2021). Optimising unrelated parallel machine scheduling in job shops with maximum allowable tardiness limit. *International Journal of Industrial and Systems Engineering,* 37(3), 359-381.

Kianpour, P., Gupta, D., Krishnan, K., & Gopalakrishnan, B. (2022). Investigating total earliness and tardiness costs through unrelated parallel machine scheduling in uncertain job shop environment using robust optimisation and design of experiment. *International Journal of Operational Research,* 45(4), 511-539.

Kramer, O. (2017). *Genetic Algorithm Essentials.* Cham: Springer.

Lenstra, J. K., & Kan, A. H. (1979). Computational Complexity of Discrete Optimization Problems. *Annals of Discrete Mathematics,* 4, 121-140.

Li, G., Li, M., Azarm, S., Al-Hashimi, S., Al-Ameri, T., & Al-Qasas, N. (2009). Improving multi-objective genetic algorithms with adaptive design of experiments and online metamodeling. *Structural and Multidisciplinary Optimization,* 37(5), 447-461.

Moser, M., Musliu, N., Schaerf, A., & Winter, F. (2022). Exact and metaheuristic approaches for unrelated parallel machine scheduling. *Journal of Scheduling*, 25, 507-534.

Nasr, S. M., El-Shorbagy, M. A., El-Desoky, I. M., Hendawy, Z. M., & Mousa, A. A. (2015). Hybrid Genetic Algorithm for Constrained Nonlinear Optimization Problems. *British Journal of Mathematics & Computer Science,* 7(6), 466-480.

Nobile, M. S., Cazzaniga, P., Besozzi, D., Colombo, R., Mauri, G., & Pasi, G. (2018). Fuzzy Self-Tuning PSO: A settings-free algorithm for global optimization. *Swarm and Evolutionary Computation*, 39, 70-85.

Pavón, R., Díaz, F., Laza, R., & Luzón, V. (2009). Automatic parameter tuning with a Bayesian case-based reasoning system. A case of study. *Expert Systems with Applications*, 36(2), 3407-3420.

Pezzella, F., Morganti, G., & Ciaschetti, G. (2008). A genetic algorithm for the Flexible Job-shop Scheduling Problem. *Computers & Operations Research,* 35(10), 3202-3212.

Rohaninejad, M., Sahraeian, R., & Nouri, B. V. (2017). Minimising the total cost of tardiness and overtime in a resumable capacitated job shop scheduling problem by using an efficient hybrid algorithm. *International Journal of Industrial and Systems Engineering,* 26(3), 318 - 343.

Rolim, G. A., Nagano, M. S., & de Athayde Prata, B. (2023). Formulations and an adaptive large neighborhood search for just-in-time scheduling of unrelated parallel machines with a common due window. *Computers & Operations Research,* 153, 106159.

Safarzadeh, H., & Niaki, S. T. (2023). Unrelated parallel machine scheduling with machine processing cost. *International Journal of Industrial Engineering Computations,* 14(1), 33-48.

Shahzad, A., & Mebarki, N. (2016). Learning Dispatching Rules for Scheduling: A Synergistic View Comprising Decision Trees, Tabu Search and Simulation. *Computers,* 5(1), 3.

Sharma, P., & Jain, A. (2016). A review on job shop scheduling with setup times. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture,* 230(3), 517-533.

Soares, L. C., & Carvalho, M. A. (2022). Application of a hybrid evolutionary algorithm to resource-constrained parallel machine scheduling with setup times. *Computers & Operations Research,* 139, 105637.

Yaghtin, M., & Javid, Y. (2023). Genetic algorithm based on greedy strategy in unrelated parallel-machine scheduling problem using fuzzy approach with periodic maintenance and process constraints. *International Journal of Supply and Operations Management*.

Zhu, X., & Wilhelm, W. E. (2007). Scheduling and lot sizing with sequence-dependent setup: A literature review. *IIE Transactions,* 38(11), 987-1007.

Zimmermann, J., & Neyer, F. J. (2013). Do We Become a Different Person When Hitting the Road?Personality Development of Sojourners. *Journal of Personality and Social Psychology,* 105(3), 515-530.

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation,* 8(2), 173-195.