Management and Production Engineering Review
Volume 14 • Number 4 • December 2023 • pp. 28–40
DOI: 10.24425/mper.2023.147201

An Accurate and Robust Genetic Algorithm to Minimize the Total Tardiness in Parallel Machine Scheduling Problems

Saleem Zeyad RAMADAN[1], Najat ALMASARWAH[2], Esraa S. ABDELALL[3], Gursel A. SUER[4], Nibal T. ALBASHABSHEH[5]

[1] *Industrial Engineering Department, School of Engineering Technology, Al Hussein Technical University, Jordan*
[2] *Department of Industrial Systems Engineering, Mutah University, Jordan*
[3] *Industrial Engineering Department, Jordan University of Science and Technology, Jordan*
[4] *Industrial and Systems Engineering, Ohio University, United States*
[5] *Department of Industrial Engineering, The University of Jordan, Jordan*

Received: 20 January 2023
Accepted: 09 November 2023

Abstract
This paper uses a Genetic Algorithm (GA) to reduce total tardiness in an identical parallel machine scheduling problem. The proposed GA is a crossover-free (vegetative reproduction) GA but used for four types of mutations (Two Genes Exchange mutation, Number of Jobs mutation, Flip Ends mutation, and Flip Middle mutation) to make the required balance between the exploration and exploitation functions of the crossover and mutation operators. The results showed that use of these strategies positively affects the accuracy and robustness of the proposed GA in minimizing the total tardiness. The results of the proposed GA are compared to the mathematical model in terms of the time required to tackle the proposed problem. The findings illustrate the ability of the propounded GA to acquire the results in a short time compared to the mathematical model. On the other hand, increasing the number of machines degraded the performance of the proposed GA.

Keywords
Identical parallel machines; Accurate and Robust Generic Algorithm; Immigration; Surrogate fitness function; Vegetative reproduction.

Introduction

Parallel machine scheduling (PMS) is one of the active research areas in the industry. Typically, parallel machines are classified with respect to processing times as: identical machines, uniform machines, and unrelated machines (Oktafiani & Ardiansyah, 2023). Identical machines are basically when the processing time of a job is the same on all machines. However, processing times vary from one job to the next. Uniform machines show a consistent pattern among machines that is applicable to all jobs and the processing times may vary proportionally. The fastest machine is the same for all jobs, the second fastest machine is the same for all jobs. Unrelated machines are characterized by the fact that there is no regular pattern among processing times and machines (Asadpour et al., 2022; Najat et al., 2019; Wang et al., 2020). This paper focuses on the identical parallel machine scheduling.

In PMS, a group of similar machines is grouped in a certain area to serve one or more of several objectives such as minimizing the total tardiness, number of tardy jobs, maximum tardiness, and flow time (YounesSinaki, et al., 2022). Various methods including metaheuristic, mathematical model, heuristic algorithms and others were proposed in the literature to schedule (n) jobs into (m) identical parallel machines given different objectives such as minimization of the total tardiness, number of tardy jobs, or maximum tardiness and makespan (Almasarwah & Süer, 2021; Kim & Kim 2021; Verma et al., 2021). One effective method for tackling identical PMS is mathematical modeling. However, for small-scale or basic situations, these optimal solutions are only achievable in a reasonable amount of time. The problem is no longer ideally solved in polynomial time as the problem's complexity and/or size increase, as is the case in real-world circumstances. As a result, metaheuristics such as the genetic algorithm, tabu search and others are

***Corresponding author**: Najat Almasarwah – Department of Industrial Systems Engineering, Mutah University, 61710, Alkarak, Jordan, e-mail: najat.eid@mutah.edu.jo*

thought to be effective in generating high-quality solution within reasonable time. On the other hand, the most studied optimization criterion in the literature is the minimization of the schedule's maximum completion time, the criteria known as makespan (Vallada et al., 2011).

This paper deals with the identical parallel machines scheduling problem noticed in the manufacturing system areas. Typically, the identical PMS problem is a NP-hard problem, and it follows the class of intensively studied combinatorial optimization problems, where it does not have exact solutions in polynomial time (Mokotoff, 2004). In this regard, a genetic algorithm that includes a fast local search and a local search enhanced crossover operator among other innovative features that, as we will see, result in a state-of-the-art performance for this type of problems and four different mutation strategies. However, the identical PMS problem could be observed at different manufacturing area such as, plastic container manufacturing (Muñoz-Villamizar et al., 2019), heat treatment furnaces (Baykasoğlu & Ozsoydan, 2018), etc.

The remainder of this paper is as follows: the related literature is reviewed; the notations and assumptions utilized are illustrated; the proposed genetic algorithm for PMS problem is introduced in detail; the numerical experiments and results are explained, respectively; and finally, the overall conclusion and future research directions are stated.

Literature review

A genetic algorithm has been considered one of the optimization methods for solving complex problem (Guo et al., 2010). Many researchers have relied on the genetic algorithm to schedule the jobs in identical parallel machines with the goal of minimizing the total tardiness (e.g., Schaller, 2014; Shim, & Kim, 2007; Tan et al., 2019; Kim et al., 2020; Anghinolfi et al., 2021; Juybari et al., 2021). Others, such as Cheng, Gen, & Tozawa (1995) utilized the genetic algorithm to schedule n jobs in the identical parallel machines with an objective of minimizing the maximum weighted lateness. Some worked on minimizing the maximum tardiness for different jobs in parallel machines with dynamic arrivals time using the genetic algorithm (Malve, & Uzsoy, 2007). To minimize the number of tardy jobs, Ho, & Chang (1995) developed two approaches (Job-Focused and Machine-Focused) to schedule n jobs in m identical parallel machines using genetic algorithm.

Different strategies such as initial population size, crossover, mutation, immigration and selection, should be taken into consideration when the GA is applied to solve a problem (Mahjoob et al., 2022). The initial population is the first step in GA design, and many methods can be used to create and code the initial population. Cochran, Horng, & Fowler (2003) explained two stages of multi-population genetic algorithm to schedule jobs in parallel machines and to implement multi-objective (i.e., Makespan, Total weighted tardiness, and Total weighted completion time). Chang, Chen, & Lin (2005) clarified the importance of using two phases of subpopulation genetic algorithm to schedule jobs in parallel machines to solve several objectives.

Typically, crossover and mutation along with selection are the main strategies to create the next generation, so they are considered the main strategies in genetic operations. Many crossover strategies have been developed in the literature. For example, partially mapped crossover, cycle crossover, order crossover, order-based crossover, position-based crossover, and heuristic crossover are suggested by Larranaga, et al. (1999). Further, they also suggested use of genetic edge recombination crossover, sorted match crossover, maximal preservative crossover, voting recombination crossover, and alternating-position crossover. Other researchers utilized two-point crossover mechanism as crossover operator in their research (Chaudhry, & Drake, 2009; Min, & Cheng, 1999). Schaller (2014) used the uniform – order crossover, and Cheng & Gen (1997) implemented preservation crossover mechanism. Ramadan (2012) proposed the frequency crossover as the crossover strategy in his research.

Mutation mechanism is the second strategy in genetic operations, many strategies were suggested in the literature for mutation. For instance, Ramadan (2012) tested two groups of mutation. The first group consists of (ends exchange mutation, group insertion mutation, and reverse ends mutation, two genes exchange mutation) to implement exploration purpose. In the second group, reverse ends exchange mutation, reverse ends mutation, one position swap mutation, and middle reverse mutation were used for exploitation purpose. Others utilized swapping mutation as mutation operator wherein two genes are selected randomly and their content are swapped (Chaudhry, & Drake, 2009; Kashan et al., 2008).

When designing GA, the designers have to keep in their mind how to balance the search direction to the optimal solution and the search speed. Immigration and stop point strategies are usually added to the GA optimization method to enhance and improve the results. Mensendiek, Gupta, & Herrmann (2015) explained the importance of using immigration to avoid premature convergence to local optimum. Essentially, the ten worst chromosomes are replaced by ten new chromosomes, which are created randomly. Immigration is applied when there is no improvement in the generations due to the premature convergence to local optimum. In addition, Balasubramanian, Mönch, Fowler, & Pfund (2004) and Schaller (2014) defined two stop points to finish the genetic algorithm based on different assumptions. The first stop point is created when the tardiness equals zero. The second stop point accrues when the time limit is finished.

Finally, the last stage in the GA is the selection stage. Different methods have been approached in literature to create the next generation. Chang, Chen, & Lin (2005) applied the elitist strategy to determine the best individual (Chromosome) in each objective. Also, Cheng & Huang (2017) relied on the elitist strategy to create the next generation based on the top 10% of paternal chromosomes with the best offspring chromosomes. While the Mir & Rezaeian (2016) utilized two selection methods to create the new generation roulette wheel selection method and tournament selection procedure. Moreover, the chromosomes with the highest fitness values receive the highest probability of creating the next generation. Several methods are used to calculate the fitness value. They developed an equation to find the fitness value based on the objective function. However, in their research, two types of fitness functions, a surrogate fitness function and an actual fitness function, are implemented to calculate the fitness value.

To the best of our knowledge, there are a limited number of studies that have considered the identical parallel machines scheduling problem with details of minimizing the total tardiness with a large number of machines. In this study, total tardiness is considered to be tackled and a genetic algorithm is proposed. The expanded initial population strategy along with crossover-free strategy (vegetative reproduction technique) were used. Further, four mutation strategies were implemented beside the immigration and elitist selection strategy to minimize the total tardiness for n jobs in m identical parallel machines. The proposed GA will be tested using 13 standard problems found in Tanaka & Araki (2008).

Notations and assumptions

Herein, the assumptions and notations used in this work are listed. Figure 1 shows the general PMS problem where jobs have to be scheduled on machines.

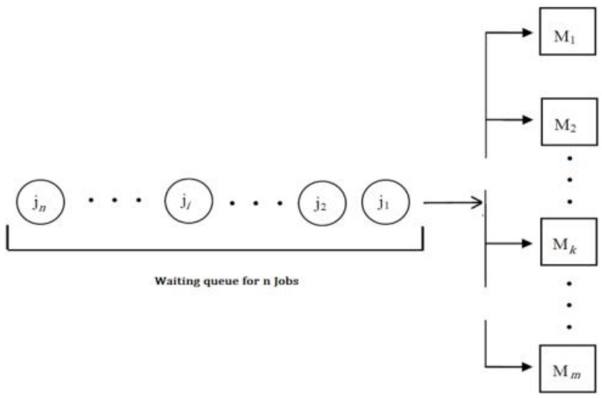

Fig. 1. Schematic diagram for PMS problem

Throughout this paper, the following notations are used:

1. n: number of jobs.
2. m: number of machines.
3. n/m PMS problem: a jobs and machines parallel machine scheduling problem.
4. TT: Total tardiness.
5. nT: Total number of tardy jobs.
6. k: Index for machine number and index for sequence number as each machine has a corresponding sequence.
7. i: Index for the job in its sequence on the machine.
8. j: Index for the job name in the jobs set.
9. S_{ik}: The start time for job i on machine k.
10. P_{ik}: The processing time for job i on machine k.
11. d_{ik}: The due date for job on machine k.
12. ξ_{ik}: The i-th gene on the k-th row of the chromosome.
13. $d_{\xi_{ik}}$: The due date for the job corresponding to the gene ξ_{ik}.
14. $S_{\xi_{ik}}$: The start time for the job corresponding to the gene ξ_{ik}.
15. $P_{\xi_{ik}}$: The processing time for the job corresponding to the gene ξ_{ik}.
16. n_k: Number of jobs in sequence k.
17. PopSize: Population size.
18. E[Nu off]: The expected number of offspring generated in each generation.

The n/m PMS problem is concerned about scheduling jobs on machines such that the TT is minimized. The TT can be calculated as:

$$TT = \sum_{k=1}^{m} \sum_{\forall i:\, d_{ik}<(S_{ik}+P_{ik})} (S_{ik} + P_{ik} - d_{ik}) \qquad (1)$$

The following assumptions are made in this paper:

1. Independent Jobs.
2. Each job has a single operation.

3. The process times include the corresponding set-up times, and the set-up times are independent of the sequence.
4. Machines are identical and have 100% availability and utilization while jobs are waiting.
5. No preemption, no cancelation and no priority for jobs are allowed.
6. All of the jobs are available at time zero.
7. The problem is static and deterministic.
8. The propounded GA is formulated and modeled using MATLAB software program.

The proposed genetic algorithm for PMS problem

Chromosome representation

The chromosome consists of m rows, one row for each machine, and $n-(m-1)$ genes in each row to enforce that at least one job will be processed on each machine. In this representation, gene $\xi_{ik}=j$ means that the i^{th} job on the k^{th} machine is job j. Hence, each gene carries three pieces of information: the position of the gene represents both the machine number and the sequence of the job on the machine while the value of the gene represents the job number. This representation ensures the feasibility of the offspring, hence; avoid the repair actions to the offspring. To illustrate the Chromosome representation, consider the data for 7/3 PMS problem in Table 1.

Table 1
Illustrative example data

Job	p_i	d_i
1	4	12
2	6	10
3	3	8
4	12	15
5	10	22
6	7	15
7	2	6

The table displays processing times (pi) and due dates (di) for seven jobs. As an illustration, Job 1 has a processing time of 4 and a due date of 12. The notation “7/3 PMS” signifies that seven jobs will be processed on three machines. Additionally, the chromosome is structured with three rows ($m=3$), and each row contains 5 genes determined by the formula $n-(m-1)=7-(3-1)$. Figure 2 provides a visual representation of a possible chromosome corresponding to the information in Table 1.

3	1	6	0	0
7	5	0	0	0
2	4	0	0	0

Fig. 2. Feasible chromosome for the 7/3 PMS problem in Table 1

The phenotype can be readily extracted from the genotype in this representation. For instance, if gene ξ_{22} equals 5, it indicates that job 5 will be the second to be processed on machine 2. On the other hand, if ξ_{14} equals 0, it signifies that no jobs will be processed in slot 4 or beyond for machine 1.

Fitness function

Two fitness functions are used in this work. A surrogate fitness function is used in the first 100 generations and the actual fitness function is used for the rest of the runs. The surrogate fitness function is the total number of tardy jobs nT while the actual fitness function is the total tardiness TT. The purpose of the surrogate fitness function is to locate the promising regions for the rest of the generation as, from authors experience, finding the minimum nT is easier than finding the minimum TT and often the solution (Schedule) at the minimum is close to the solution at the minimum TT.

The surrogate fitness function is:

$$TT=\sum_{k=1}^{m}\sum_{\substack{d_{\xi ik}<(S_{\xi ik}+P_{\xi ik})\\ i=1:n-(m-1)}} i \tag{2}$$

and the actual fitness function is

$$TT=\sum_{k=1}^{m}\sum_{\substack{d_{\xi ik}<(S_{\xi ik}+P_{\xi ik})\\ i=1:n-(m-1)}} (S_{\xi ik}+P_{\xi ik}-d_{\xi ik}) \tag{3}$$

where $S_{\xi_{ik}}$, $P_{\xi_{ik}}$ and $d_{\xi_{ik}}$ are the start time, processing time, and due date for the job represented by gene ξ_{ik}.

The surrogate fitness function value for the chromosome in Figure 2 is:

$$nT=\sum_{k=1}^{3}\sum_{\substack{d_{\xi ik}<(S_{\xi ik}+P_{\xi ik})\\ i=1:5}} i=(0)+(0)+(1)=1$$

while the actual fitness function value is

$$
\begin{aligned}
TT &= \sum_{k=1}^{3} \sum_{\substack{d_{\xi ik} < (S_{\xi ik} + P_{\xi ik}) \\ i=1:5}} (S_{ik} + P_{ik} - d_{ik}) \\
&= (0) + (0) + (3) = 3
\end{aligned}
$$

Initial population

An expanded initial population strategy was adopted in this work. In this strategy, the initial population size was the population size PopSize multiplied by a factor of c. The jobs were assigned to the machines randomly such that at least one job is assigned for each machine.

Mutation

In this study, a crossover-free Genetic Algorithm (GA) was employed, exclusively utilizing mutation to generate offspring from a single parent. This crossover-free approach emulates vegetative reproduction observed in plants. Each selected chromosome underwent mutation with four distinct types:

1. Two Genes Exchange Mutation: This mutation type exclusively affects two genes within the same machine sequence, similarly to traditional mutations. It induces a limited change in the chromosome, primarily serving exploitation purposes.
2. Number of Jobs Mutation: This mutation alters the number of jobs in the machine sequence, consequently impacting two machines' sequences.
3. Flip Ends Mutation: This mutation involves flipping the ends of sequences in the machines.
4. Flip Middle Mutation: This mutation entails flipping the middle of sequences in the machines.

These four mutation types collectively serve the roles of both traditional mutation and crossover. Two Genes Exchange Mutation, resembling traditional mutation, generates subtle changes for exploitation. On the other hand, the remaining three mutations, by introducing more significant disturbances in the chromosome, function similarly to crossovers, playing a role in exploration.

A noteworthy advantage of these mutation types lies in their consistent production of feasible offspring, eliminating the need for post-mutation adjustments commonly required in crossovers or other mutation approaches.

Figure 3 shows a schematic diagram of the *Two Genes Exchange* mutation. In this mutation, two random genes were selected from a randomly selected sequence and switched. It should be noticed here that the resulting offspring is always feasible.

3	1	6	0	0
7	5	0	0	0
2	4	0	0	0

Parent

6	1	3	0	0
7	5	0	0	0
2	4	0	0	0

Offspring

Fig. 3. Schematic diagram for Two Genes Exchange mutation

Figure 4 shows a schematic diagram of the Number of Jobs mutation. In this mutation, the machine sequence with the largest tardiness is selected and the last job in that sequence is removed and added after the last job in the sequence with the lowest tardiness providing that the minimum of one job per machine constraint is conserved. Ties are broken arbitrary. In Figure 4, the tardiness for the three sequences in the parent chromosome are [0 0 3] respectively. Therefore, Job 4 will be removed from sequence 3 and added after Job 6 in sequence 1 to form the offspring. The main function of this type of mutation is to balance out the load on the machines.

3	1	6	0	0
7	5	0	0	0
2	4	0	0	0

Parent

3	1	6	4	0
7	5	0	0	0
2	0	0	0	0

Offspring

Fig. 4. Schematic diagram for Number of Jobs mutation

Figure 5 shows a schematic diagram for the Flip Ends mutation for 13/3 PMS problem. In this mutation, one of the sequences is chosen at random and then two random numbers, less than or equal to the number of jobs in that sequence, are generated. The jobs between 1 and the smallest number will be flipped and the jobs between the largest number and will be flipped also. For this case, the sequence of Machine 1 is selected, and the two random numbers are 3 and 6.

3	1	2	9	4	11	10	0	0	0	0
5	6	0	0	0	0	0	0	0	0	0
8	7	12	13	0	0	0	0	0	0	0

Parent

2	1	3	9	4	10	11	0	0	0	0
5	6	0	0	0	0	0	0	0	0	0
8	7	12	13	0	0	0	0	0	0	0

Offspring

Fig. 5. Schematic diagram for Flip Ends mutation

Figure 6 shows a schematic diagram for the *Flip Middle* mutation for 13/3 PMS problem. In this mutation, one of the sequences is chosen at random and then two random numbers, less than or equal to the number of jobs in that sequence n_k, are generated. The jobs between (and including) the smallest number and the largest number will be flipped. For this case, the sequence of Machine 1 is selected, and the two random numbers are 3 and 6.

3	1	2	9	4	11	10	0	0	0	0
5	6	0	0	0	0	0	0	0	0	0
8	7	12	13	0	0	0	0	0	0	0

Parent

3	1	11	4	9	2	10	0	0	0	0
5	6	0	0	0	0	0	0	0	0	0
8	7	12	13	0	0	0	0	0	0	0

Offspring

Fig. 6. Schematic diagram for Flip Middle mutation

It should be noticed here that all of the four mutation types are designed such that a feasible offspring is generated when using them. Therefore, there is no need for any type of repair for the offspring as repairs consume computational time. Figure 7 shows the general framework for the proposed GA.

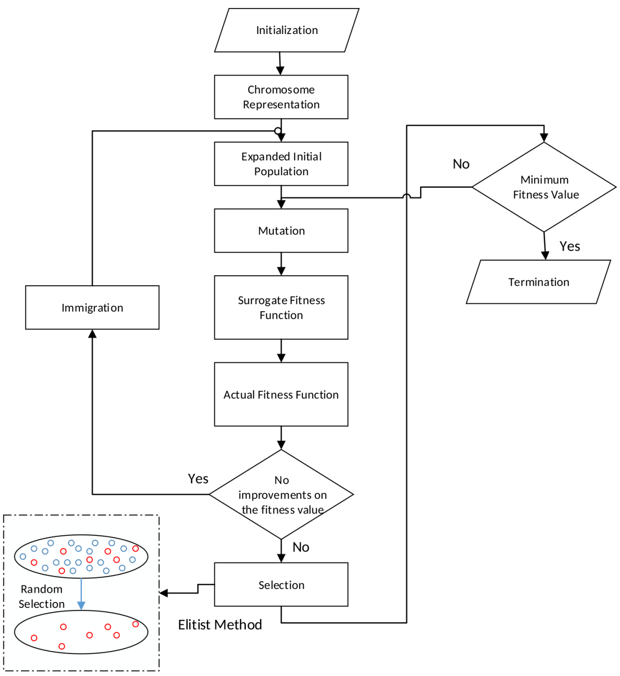

Fig. 7. The framework of the proposed GA

In this GA, a 25% mutation rate is adopted. This means that $E[\text{Nu off}]$ is the same as as PopSize:

$$E[\text{Nu off}] = \text{PopSize} \times 0.25 \times 4 = \text{PopSize} \tag{4}$$

Immigration

This strategy was important in this GA as it helped in reducing the effect of premature convergence, therefore helping the GA to reach the optimal solution many times. The immigration operator in this GA simply replaced the entire population with a fresh random population if 250 generations elapsed before any improvement on the fitness value was achieved.

Selection

The Elitist selection strategy was adopted in this work. Under this strategy, all the parents and the offspring competed according to their fitness values, both in the surrogate fitness function stage and in the actual fitness function stage, and the best PopSize chromosomes, corresponding to the minimum fitness values, were selected to be the parents of the next generation.

Experimentations

Thirteen benchmark instances from Tanaka & Araki (2008) were considered for experimentation as in Table 2. The data given in these problems include

Table 2
The 13 benchmark instances used in the experimentation

Problem	Number of jobs	Number of machines	Optimal value
20_02_02_02_001	20	2	147
20_02_02_02_002	20	2	102
20_02_02_02_003	20	2	152
20_02_02_06_001	20	2	0
20_02_10_10_005	20	2	1513
20_05_02_02_001	20	5	149
20_05_02_02_002	20	5	111
20_10_02_02_001	20	10	195
20_10_10_10_005	20	10	739
25_02_02_02_001	25	2	163
25_05_02_02_001	25	5	153
25_10_02_02_001	25	10	174
25_10_10_10_005	25	10	1202

Table 3
The results obtained by the GA, the average fitness value, the standard deviation, and the number of times

Problem	Average TT	Standard deviation for TT	Number of times optimal value found	Average time to the best solution (s)	Average number of generations	% Average deviation from optimal
20_02_02_02_001	147	0	100	53.7	4133.6	0.0%
20_02_02_02_002	102	0	100	2.1	160.4	0.0%
20_02_02_02_003	152	0	100	8.1	624.8	0.0%
20_02_02_06_001	0	0	100	2.3	180.1	0.0%
20_02_10_10_005	1517.6	2.5	16	273.1	20952	0.3%
20_05_02_02_001	151.5	2.7	44	248.8	8070.7	1.7%
20_05_02_02_002	*111*	*0*	*100*	90.0	2912.6	0.0%
20_10_02_02_001	199.9	2.5	4	286.9	4559.0	2.5%
20_10_10_10_005	739	0	100	12.7	201.6	0.0%
25_02_02_02_001	163	0	100	9.5	713.2	0.0%
25_05_02_02_001	160.4	3.17	3	297.2	9738.9	4.8%
25_10_02_02_001	198.7	4.7	0	280.9	4394.1	14.2%
25_10_10_10_005	1235.5	7.83	0	294.0	4600.4	2.8%

number of jobs, number of machines, processing time, and due date. Each instance was solved 100 times, and the TT value was recorded each time. The average, the standard deviation, the number of times the GA was able to find the optimal solution and the percentage of the average deviation from the optimal value along with the average time and average generation number were recorded in Table 3.

Column 7 of Table 3 shows the percentage of the deviation between the average TT and the optimal TT. One can see that the proposed GA gave accurate results. In all the 13 instances, the GA was able to provide solutions with a percentage of the deviation between the average TT and the optimal TT of less than 5% except for one instance, 25_10_02_02_001. In fact, in 7 instances out of the 13 instances the GA was able to find the exact optimal value consistently. Moreover, the table showed that the GA was robust as the standard deviations in column 3 were small. In fact, 7 out of the 13 instances had zero standard deviation, which means that in each replication of these instances, the GA was able to reach the same TT value. Furthermore, the maximum time average for the 13 instances was less than 5 minutes with an overall average of 2.3 minutes which indicates that this GA was efficient in terms of computational time. Table 3 also displays the average number of generations for the reader's convenience.

The optimal solutions found for the different instances are shown in Table 4. One should notice here that there may be alternative optimal for each instance and hence these optimal solutions may vary from one run to another, but their corresponding optimal values are the same. The proposed GA was able to find the optimal TT for 10 out of the 13 instances.

The worst result among the 13 instances was for instance 25_10_02_02_001 where the average TT found was about 14% more than the optimal one. The accuracy of the GA was measured by the deviation between the average fitness value and the optimal value. The robustness of the GA was measured by the standard deviation of the TT values. Finally, six tests of hypothesis were performed to test the claims about the effectiveness of the immigration, surrogate fitness function, and the expanded initial population sets of strategies used in the proposed GA.

Immigration operator effect on GA performance testing

The first two tests of hypotheses were designed to see if the immigration operator improved the performance of the GA or not in terms of accuracy and robustness. To do so, the problem 25_10_02_02_001 which is the one with the worst GA performance, was selected. The problem was solved again 100 times with the same GA setup used to construct Table 3 except that the immigration operator was eliminated. The average optimal value found was 215.0 with a standard deviation of 14.5.

Table 4
Optimal solutions found for the different problems

Problem	Optimal solution
20_02_02_02_001	M1: 13 10 1 6 2 19 14 5 11 20 18 M2: 8 3 15 16 7 17 12 9 4
20_02_02_02_002	M1: 12 4 11 9 20 6 16 3 10 15 M2: 14 7 17 5 19 1 18 13 8 2
20_02_02_02_003	M1: 13 17 18 12 1 2 14 11 3 4 M2: 19 15 9 10 16 5 8 7 6 20
20_02_02_06_001	M1: 14 12 2 5 8 19 11 15 4 M2: 6 3 7 10 1 13 16 17 9 20 18
20_02_10_10_005	M1: 15 2 3 8 16 18 20 19 9 M2: 6 1 4 11 13 17 12 10 7 14 5
20_05_02_02_001	M1: 17 13 15 9 M2: 5 7 14 18 M3: 16 19 11 20 M4: 2 8 10 4 M5: 1 3 6 12
20_05_02_02_002	M1: 17 9 7 2 M2: 5 18 8 14 M3: 20 12 6 16 4 M4: 10 19 3 M5: 1 13 11 15
20_10_02_02_001	Not Found
20_10_10_10_005	M1: 18 19 M2: 16 9 M3: 6 17 14 M4: 8 20 M5: 12 2 M6: 5 M7: 3 1 M8: 15 M9: 13 11 M10: 4 10 7
25_02_02_02_001	M1: 3 25 12 24 18 9 23 10 6 20 17 13 15 M2: 2 14 21 16 19 1 22 11 5 8 7 4
25_05_02_02_001	M1: 3 18 2 7 25 M2: 24 9 1 15 M3: 16 20 17 19 23 M4: 5 11 12 22 13 M5: 14 6 10 21 8 4
25_10_02_02_001	Not Found
25_10_10_10_005	Not Found

Figure 8 displays the normality probability plot illustrating the outcomes from 100 replicates. The obtained p-value is 0.59, which is less than the significance level of 0.05, suggesting that the optimal results across the 100 replicates adhere to a normal distribution.

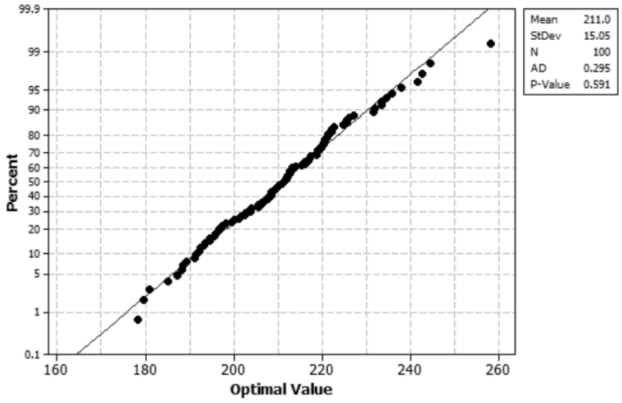

Fig. 8. Normality probability plot for the 100 optimal values

The following set of hypotheses were tested:

H01: The average TT value found with immigration operator is the same as the average TT found without immigration factor; hence immigration factor did not improve the accuracy of the GA.

H11: The average TT found with immigration operator is less than the average TT found without immigration operator; hence immigration operator improved the accuracy of the GA.

The t-test for the difference between two means showed that the p-value is 0.000, which allows us to reject H01 and accept H11, i.e., accept that the immigration operator improved the accuracy of the GA.

H02: The standard deviation for the fitness values found for the 100 replications with immigration operator is the same as the standard deviation for the fitness values found for the 100 replications without immigration operator; hence immigration operator did not improve the robustness of the GA.

H12: The standard deviation for the fitness values found for the 100 replications with immigration operator is less than the standard deviation for the fitness values found for the 100 replications without immigration operator; hence immigration operator improved the robustness of the GA.

The F-test of equality of two variances showed that the p-value is 0.000, which allows us to reject H02 and accept H12, i.e., accept that the immigration operator improved the robustness of the GA.

Surrogate fitness effect on GA performance testing

The second set of hypotheses were designed to see if the use of the surrogate fitness function improved the performance of the GA or not in terms of accuracy and robustness. To do so, the problem 25_10_02_02_001 was solved again 100 times with the same GA setup used to construct Table 3 except that the surrogate fitness function was not used. The average optimal value found was 204.9 with a standard deviation of 6.6. The following two sets of hypotheses were tested:

H03: The average TT value found with the use of the surrogate fitness function is the same as the average TT value found without the use of the surrogate fitness function; hence surrogate fitness function did not improve the accuracy of the GA.

H13: The average TT value found with the use of the surrogate fitness function is less than the average TT value found without the use of the surrogate fitness function; hence surrogate fitness function improved the accuracy of the GA.

The t-test for the difference between two means results showed that the p-value is 0.000 which allows us to reject H03 and accept H13, i.e., accept that the surrogate fitness function improved the accuracy of the GA.

H04: The standard deviation for the TT values found with the use of the surrogate fitness function is the same as the standard deviation for the TT values found without the use of the surrogate fitness function; hence the use of the surrogate fitness function did not improve the robustness of the GA.

H14: The standard deviation for the TT values found with the use of the surrogate fitness function is less than the standard deviation for the TT values found without the use of the surrogate fitness function; hence the use of the surrogate fitness function improved the robustness of the GA.

The F-test for the equality of two variances showed that the p-value is 0.000, which allows us to reject H04 and accept H14, i.e., accept that the use of the surrogate fitness function improved the robustness of the GA.

The expanded initial population effect on GA performance testing.

H05: The average value TT found with the expanded initial population is the same as the average TT found without the expanded initial population; hence the expanded initial population did not improve the accuracy of the GA.

H15: The average TT found with the expanded initial population is less than the average TT found without the expanded initial population; hence the expanded initial population improved the accuracy of the GA.

The t-test for the difference between two means showed that the p-value is 0.000, which allows us to reject H05 and accept H15, i.e., accept that the expanded initial population improved the accuracy of the GA.

H06: The standard deviation for the TT values found with the use of the expanded initial population is the same as the standard deviation for the TT values found without the use of the expanded initial population hence the use of the expanded initial population did not improve the robustness of the GA.

H16: The standard deviation for the TT values found with the use of the expanded initial population is less than the standard deviation for the TT values found without the use of the expanded initial population; hence the use of the expanded initial population improved the robustness of the GA.

The F-test for the equality of two variances results showed that the p-value is 0.000, which allows us to reject H06 and accept H16, i.e., accept that the use of the expanded initial population improved the robustness of the GA.

The test of hypothesis that was performed to test the accuracy and robustness of the three strategies, i.e., immigration, surrogate fitness function, and the expanded initial population, showed a p-value of 0.00 for all of the tests. This result showed that it is important to make use of the three strategies to improve the robustness and the accuracy of the proposed GA.

Balancing time and quality

Several studies have been proposed in the literature to solve the identical parallel machine scheduling problem to minimizing the total tardiness using the mathematical model (Biskup et al., 2008). Mathematical modeling is a powerful problem-solving tool. However, these optimal solutions are only attainable within a tolerable time for simple or small-scale problems. As the problem complexity and/or size increases, as found in real-life scenarios, the problem is no longer optimally solved in polynomial time.

One of the essential features that could be considered during experiments with genetic algorithm and mathematical models is the tradeoff between the solutions and the time needed to acquire solutions. It has been observed that the genetic algorithm could obtain the near optimal solutions or duplicate the results of the mathematical model in a short time. Meanwhile, the mathematical model takes longer to obtain the optimal solutions. In this regard, the time to obtain the optimal solution in the mathematical model was considered to make a fair comparison between the results of the mathematical and genetic algorithm. The results tabulated in Table 5, compares the solutions time for the mathematical model and genetic algorithm. The results show the ability of the genetic algorithm to obtain the near optimal solution or duplicate the results of the mathematical model in short time compared to the mathematical model.

Table 5
Comparison of the results of the mathematical model and genetic algorithm in terms of the total tardiness and the time value to obtain the results

Problem	Mathematical model results		Proposed GA	
	Total tardiness	Time (second)	Total tardiness	Time (second)
20-02-02-02-001	147	67.2	147	53.7
20-02-02-02-002	102	20.43	102	2.1
20-02-02-02-003	152	50.46	152	8.1
20-02-02-06-001	0	19.63	0	2.3
20-02-10-10-005	1513	285	1517.6	273.1
20-05-02-02-001	149	348.1	151.5	248.8
20-05-02-02-002	111	134.74	111	90
20-10-02-02-001	195	379.35	199.9	286.9
20-10-10-10-005	739	62.05	739	12.7
25-02-02-02-001	163	30.03	163	9.5
25-05-02-02-001	153	541.85	160.4	297.2
25-10-02-02-001	174	651.91	198.7	280.9
25-10-10-10-005	1202	646	1235.5	294

Results

The proposed GA was tested using the benchmark instances for PMP as given in Table 3. As shown in Figure 9, the proposed GA can find the optimal solution for seven problems out of 13 problems. In this case, the system is capable of finding the solution to 53.85% of problems.

1. Comparison the Total Tardiness Value for the Optimal Value with the Average TT Using Proposed GA.
2. Comparison the Number of Machines and % Average Deviation from Optimal

On the other hand, the number of machines was found to be the main factor that affects the performance of the proposed GA. In this regard, increasing the number of machines were found to reduce the system performance. Therefore, the system was incapable of finding the optimal solution when the number of machines increases. For example, in problem nine, 25 jobs are assigned to be processed on 10 machines. In this case, the percentage of Average Deviation from Optimal equals to 14.2%, and it represents the worst case compared to the other cases where the percentage of Average Deviation from Optimal is less than 5% as shown in Figure 10.

To test the accuracy and robustness of the proposed genetic algorithm for minimizing the total tardiness, two tests of hypothesis (t-test and F-test) were implemented to examine the performance of the three strategies (immigration, surrogate fitness function, and the expanded initial population) which were developed in the proposed GA at p-value 0.00. The t-test was applied to check the ability of the suggested strategy to improve the accuracy of the proposed GA. While the F-test is implemented to test the importance of implementing the strategy on the robustness of the proposed GA. The results showed the importance of applying the three strategies to improve the accuracy and robustness of the proposed GA. To summarize, it is important to employ the three

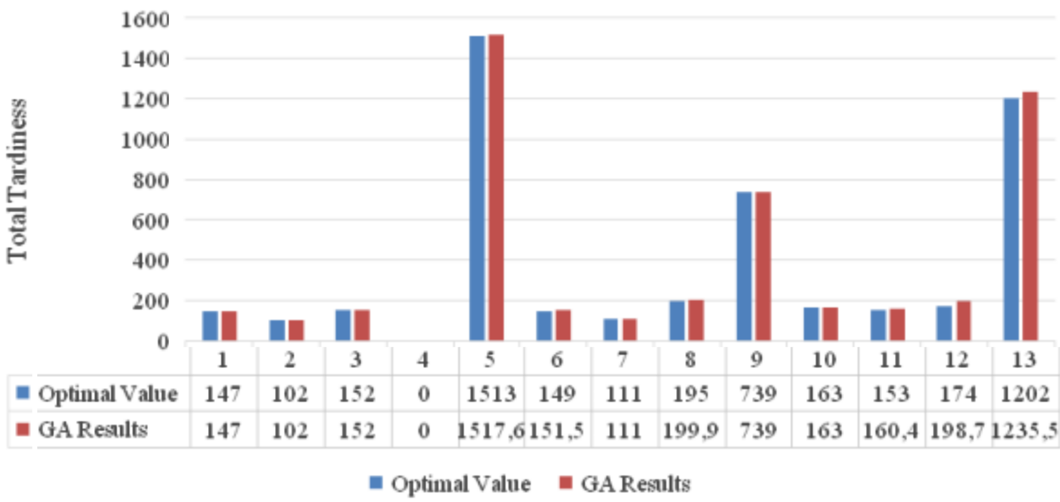

Fig. 9. Comparison the Total Tardiness Value for the Optimal Value with the Average TT Using Proposed GA

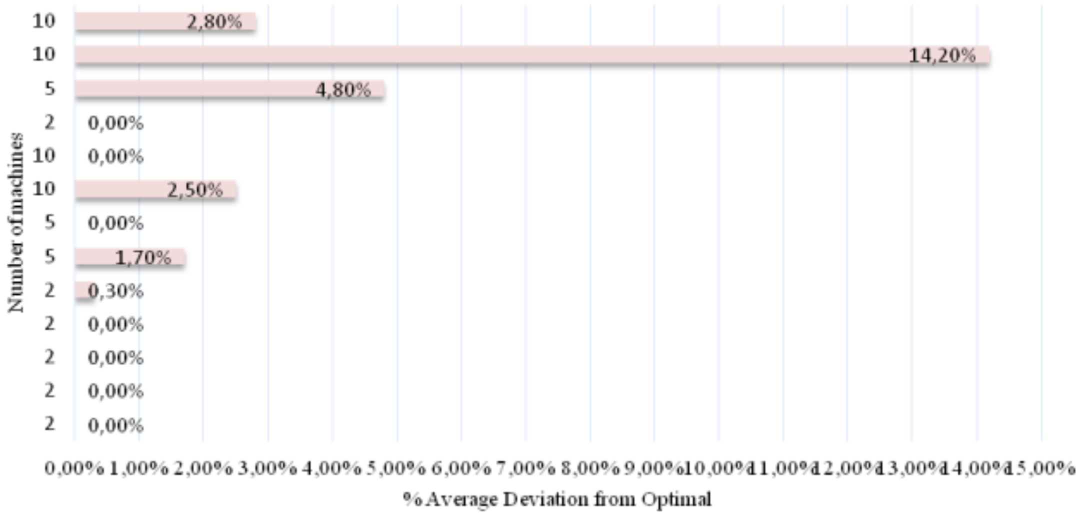

Fig. 10. Comparison the Number of Machinesand % Average Deviation from Optimal

sets of strategies (immigration, surrogate fitness function and the expanded initial population) to enhance and improve the accuracy and robustness of the proposed GA.

Conclusions

A proposed GA is developed in this research for minimizing the total tardiness for jobs in identical parallel machines. The proposed algorithm considered three strategies, expanded initial population, surrogate fitness function and immigration. Next, the effect of immigration, surrogate fitness function, and the expanded initial population strategies on improving the robustness and the accuracy of the proposed GA were tested. The obtained results led to conclude that using these strategies improve the accuracy and robustness of the GA to schedule the jobs in the identical parallel machines under the objective of minimizing the total tardiness. Moreover, the system performance was found to depend on the number of machines in the system with indirect relation. Increasing the number of machines reduced the accuracy and robustness of the proposed GA. On the other hand, minimizing the total tardiness in the systems would improve the customer satisfaction level; thus, this creates an environmentally friendly system.

Even of the benefits provided by the genetic algorithm in solving a large part of problem, where the near-optimal solutions are obtained, coding the fitness function to achieve a higher fitness and generate better solutions or the near-optimal solution for a given problem is a significant challenge/limitation. Additionally, the efficacy of the propounded heuristic decreases as the number of machines and jobs in the manufacturing systems increases. For future research, it is recommended to consider the unrelated and uniform parallel machine scheduling problem and develop an algorithm to minimize the total energy consumption in a green parallel machine. The uncertainty in the processing time and demand for jobs in machines should also be taken into consideration. The setup and the maintenance times for the machines could be considered.

Acknowledgments

Conflict of interest: All authors declare that no conflict of interest.

Ethical approval: This article does not contain any studies with human participants or animals performed by any of the authors.

Informed consent: Informed consent was obtained from all individual participants included in the study.

References

Almasarwah, N., & Süer, G.A. (2021). Consideration of processing time dissimilarity in batch-cyclic scheduling of flowshop cells. *International Journal of Production Research*, 59(21), 6544–6563.

Anghinolfi, D., Paolucci, M., & Ronco, R. (2021). A bi-objective heuristic approach for green identical parallel machine scheduling. *European Journal of Operational Research*, 289(2), 416–434.

Asadpour, M., Hodaei, Z., Azami, M., Kehtari, E., & Vesal, N. (2022). A green model for identical parallel machines scheduling problem considering tardy jobs and job splitting property. *Sustainable Operations and Computers*, 3, 149–155.

Balasubramanian, H., Mönch, L., Fowler, J., & Pfund, M. (2004). Genetic algorithm based scheduling of parallel batch machines with incompatible job families to minimize total weighted tardiness. *International Journal of Production Research*, 42(8), 1621–1638.

Baykasoğlu, A., & Ozsoydan, F.B. (2018). Dynamic scheduling of parallel heat treatment furnaces: A case study at a manufacturing system. *Journal of manufacturing systems*, 46, 152–162.

Biskup, D., Herrmann, J., & Gupta, J.N. (2008). Scheduling identical parallel machines to minimize total tardiness. *International Journal of Production Economics*, 115(1), 134–142.

Chang, P.C., Chen, S.H., & Lin, K.L. (2005). Two-phase sub population genetic algorithm for parallel machine-scheduling problem. *Expert Systems with Applications*, 29(3), 705-712.

Chaudhry, I.A., & Drake, P.R. (2009). Minimizing total tardiness for the machine scheduling and worker assignment problems in identical parallel machines using genetic algorithms. *The International Journal of Advanced Manufacturing Technology*, 42(5), 581-594.

Cheng, R., & Gen, M. (1997). Parallel machine scheduling problems using memetic algorithms. *Computers & Industrial Engineering*, 33(3-4), 761–764.

Cheng, C.Y., & Huang, L.W. (2017). Minimizing total earliness and tardiness through unrelated parallel machine scheduling using distributed release time control. *Journal of Manufacturing Systems*, 42, 1–10.

Cheng, R., Gen, M., & Tozawa, T. (1995). Minmax earliness/tardiness scheduling in identical parallel machine system using genetic algorithms. *Computers & Industrial Engineering*, 29(1–4), 513–517.

Cochran, J.K., Horng, S.M., & Fowler, J.W. (2003). A multi-population genetic algorithm to solve multi-objective scheduling problems for parallel machines. *Computers & Operations Research*, 30(7), 1087–1102.

Guo, P., Wang, X., & Han, Y. (2010, October). The enhanced genetic algorithms for the optimization design. In: *2010 3rd International Conference on Biomedical Engineering and Informatics (BMEI)*, IEEE, 7, 2990–2994.

Ho, J.C., & Chang, Y.L. (1995). Minimizing the number of tardy jobs for m parallel machines. *European Journal of Operational Research*, 84(2), 343–355.

Juybari, J.K., Juybari, S.K., & Hasanzadeh, R. (2021). Parallel machines scheduling with time-dependent deterioration, using meta-heuristic algorithms. SN Applied Sciences, 3(3), 1–13.

Kashan, A.H., Karimi, B., & Jenabi, M. (2008). A hybrid genetic heuristic for scheduling parallel batch processing machines with arbitrary job sizes. *Computers & Operations Research*, 35(4), 1084–1098.

Kim, J.G., Song, S., & Jeong, B. (2020). Minimising total tardiness for the identical parallel machine scheduling problem with splitting jobs and sequence-dependent setup times. *International Journal of Production Research*, 58(6), 1628–1643.

Kim, J., & Kim, H.J. (2021). Parallel machine scheduling with multiple processing alternatives and sequence-dependent setup times. *International Journal of Production Research*, 59(18), 5438–5453.

Larranaga, P., Kuijpers, C.M., Murga, R.H., Inza, I., & Dizdarevic, S. (1999). Genetic algorithms for the travelling salesman problem: A review of representations and operators. *Artificial Intelligence Review*, 13(2), 129–170.

Mahjoob, M., Fazeli, S.S., Milanlouei, S., Tavassoli, L.S., & Mirmozaffari, M. (2022). A modified adaptive genetic algorithm for multi-product multi-period inventory routing problem. *Sustainable Operations and Computers*, 3, 1–9.

Malve, S., & Uzsoy, R. (2007). A genetic algorithm for minimizing maximum lateness on parallel identical batch processing machines with dynamic job arrivals and incompatible job families. *Computers & Operations Research*, 34(10), 3016–3028.

Mensendiek, A., Gupta, J.N., & Herrmann, J. (2015). Scheduling identical parallel machines with fixed delivery dates to minimize total tardiness. *European Journal of Operational Research*, 243(2), 514–522.

Min, L., & Cheng, W. (1999). A genetic algorithm for minimizing the makespan in the case of scheduling identical parallel machines. *Artificial Intelligence in Engineering*, 13(4), 399–403.

Mir, M.S.S., & Rezaeian, J. (2016). A robust hybrid approach based on particle swarm optimization and genetic algorithm to minimize the total machine load on unrelated parallel machines. *Applied Soft Computing*, 41, 488–504.

Mokotoff, E. (2004). An exact algorithm for the identical parallel machine scheduling problem. *European Journal of Operational Research*, 152(3), 758–769.

Muñoz-Villamizar, A., Santos, J., Montoya-Torres, J., & Alvaréz, M. (2019). Improving effectiveness of parallel machine scheduling with earliness and tardiness costs: A case study. *International Journal of Industrial Engineering Computations*, 10(3), 375–392.

Najat, A., Yuan, C., Gursel, S., & Tao, Y. (2019). Minimizing the number of tardy jobs on identical parallel machines subject to periodic maintenance. *Procedia Manufacturing*, 38, 1409–1416.

Oktafiani, A., & Ardiansyah, M.N. (2023). Scheduling Splitable Jobs on Identical Parallel Machines to Minimize Makespan using Mixed Integer Linear Programming. *International Journal of Innovation in Enterprise System*, 7(01), 41–54.

Ramadan, S.Z. (2012). Reducing premature convergence problem in genetic algorithm: application on travel salesman problem. *Computer and Information Science*, 6(1), 47.

Schaller, J.E. (2014). Minimizing total tardiness for scheduling identical parallel machines with family setups. *Computers & Industrial Engineering*, 72, 274–281.

Shim, S.O., & Kim, Y.D. (2007). Scheduling on parallel identical machines to minimize total tardiness. *European Journal of Operational Research*, 177(1), 135–146.

Tanaka and M. Araki (2008). A branch-and-bound algorithm with Lagrangian relaxation to minimize total tardiness on identical parallel machines. *International Journal of Production Economics*. 113(5), 446–458.

Tan, M., Yang, H.L., & Su, Y.X. (2019). Genetic algorithms with greedy strategy for green batch scheduling on non-identical parallel machines. *Memetic Computing*, 11(4), 439–452.

Vallada, E., & Ruiz, R. (2011). A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times. *European Journal of Operational Research*, 211(3), 612–622.

Verma, S., Pant, M., & Snasel, V. (2021). A comprehensive review on NSGA-II for multi-objective combinatorial optimization problems. *IEEE Access*, 9, 57757–57791.

Wang, S., Wu, R., Chu, F., & Yu, J. (2020). Identical parallel machine scheduling with assurance of maximum waiting time for an emergency job. *Computers & Operations Research*, 118, 104918.

YounesSinaki, R., Sadeghi, A., Mosadegh, H., Almasarwah, N., & Suer, G. (2022). Cellular manufacturing design 1996–2021: a review and introduction to applications of Industry 4.0. *International Journal of Production Research*, 1–52.