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

Saleem Zeyad RAMADAN¹, Najat ALMASARWAH², Esraa S. ABDELALL³, Gursel A. SUER⁴, Nibal T. ALBASHABSHEH⁵

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 Genetic 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

© 2023 The Author(s). This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/)
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; Juaybari 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 Larrañaga, 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. Mensendieck, Gupta, & Herrmann (2015) ex-
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.

### Notations

- **n**: number of jobs.
- **m**: number of machines.
- **n/m PMS problem**: a jobs and machines parallel machine scheduling problem.
- **TT**: Total tardiness.
- **nT**: Total number of tardy jobs.
- **k**: Index for machine number and index for sequence number as each machine has a corresponding sequence.
- **i**: Index for the job in its sequence on the machine.
- **j**: Index for the job name in the jobs set.
- **S_{ik}**: The start time for job \(i\) on machine \(k\).
- **P_{ik}**: The processing time for job \(i\) on machine \(k\).
- **d_{ik}**: The due date for job \(i\) on machine \(k\).
- **ξ_{ik}**: The \(i\)-th gene on the \(k\)-th row of the chromosome.
- **d_{ξ_{ik}}**: The due date for the job corresponding to the gene \(ξ_{ik}\).
- **S_{ξ_{ik}}**: The start time for the job corresponding to the gene \(ξ_{ik}\).
- **P_{ξ_{ik}}**: The processing time for the job corresponding to the gene \(ξ_{ik}\).
- **n_k**: Number of jobs in sequence \(k\).
- **PopSize**: Population size.
- **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:

\[
TT = \sum_{k=1}^{m} \sum_{i : d_{ik} < (S_{ik} + P_{ik} - d_{ik})} (S_{ik} + P_{ik} - d_{ik})
\]  

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>
<thead>
<tr>
<th>Job</th>
<th>( p_i )</th>
<th>( d_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>22</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>15</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>6</td>
</tr>
</tbody>
</table>

The table displays processing times (\( p_i \)) and due dates (\( d_i \)) 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 \( \xi_{22} \) equals 5, it indicates that job 5 will be the second to be processed on machine 2. On the other hand, if \( \xi_{14} \) 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:

\[
TT = \sum_{k=1}^{m} \sum_{i:(S_{\xi ik} + P_{\xi ik}) + P_{\xi ik} - d_{\xi ik})}
\]

and the actual fitness function is

\[
TT = \sum_{k=1}^{m} \sum_{i:(S_{\xi ik} + P_{\xi ik}) + (S_{\xi ik} + P_{\xi ik} - d_{\xi ik})}
\]

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 \( \xi_{ik} \).

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

\[
nT = \sum_{k=1}^{3} \sum_{i:(S_{\xi ik} + P_{\xi ik}) + (0) + (0) + (1) = 1}
\]
while the actual fitness function value is

\[ TT = \sum_{k=1}^{3} \sum_{i=1}^{5} (S_{ik} + P_{ik} - d_{ik}) \]

\[ = (0) + (0) + (3) = 3 \]

**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.

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 arbitrarily. 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.

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.

A noteworthy advantage of this 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.
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 (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.

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

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

(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>
<thead>
<tr>
<th>Problem</th>
<th>Number of jobs</th>
<th>Number of machines</th>
<th>Optimal value</th>
</tr>
</thead>
<tbody>
<tr>
<td>20_02_02_02_001</td>
<td>20</td>
<td>2</td>
<td>147</td>
</tr>
<tr>
<td>20_02_02_02_002</td>
<td>20</td>
<td>2</td>
<td>102</td>
</tr>
<tr>
<td>20_02_02_02_003</td>
<td>20</td>
<td>2</td>
<td>152</td>
</tr>
<tr>
<td>20_02_02_02_004</td>
<td>20</td>
<td>2</td>
<td>1513</td>
</tr>
<tr>
<td>20_02_02_02_005</td>
<td>20</td>
<td>5</td>
<td>149</td>
</tr>
<tr>
<td>20_02_02_02_008</td>
<td>20</td>
<td>5</td>
<td>111</td>
</tr>
<tr>
<td>20_02_10_10_005</td>
<td>20</td>
<td>10</td>
<td>195</td>
</tr>
<tr>
<td>20_02_10_10_005</td>
<td>20</td>
<td>10</td>
<td>739</td>
</tr>
<tr>
<td>25_02_02_02_001</td>
<td>25</td>
<td>2</td>
<td>163</td>
</tr>
<tr>
<td>25_02_02_02_001</td>
<td>25</td>
<td>2</td>
<td>153</td>
</tr>
<tr>
<td>25_10_02_02_002</td>
<td>25</td>
<td>10</td>
<td>174</td>
</tr>
<tr>
<td>25_10_10_10_005</td>
<td>25</td>
<td>10</td>
<td>1202</td>
</tr>
</tbody>
</table>
The results obtained by the GA, the average fitness value, the standard deviation, and the number of times

<table>
<thead>
<tr>
<th>Problem</th>
<th>Average $TT$</th>
<th>Standard deviation for $TT$</th>
<th>Number of times optimal value found</th>
<th>Average time to the best solution (s)</th>
<th>Average number of generations</th>
<th>% Average deviation from optimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>20_02_02_02_001</td>
<td>147</td>
<td>0</td>
<td>100</td>
<td>53.7</td>
<td>4133.6</td>
<td>0.0%</td>
</tr>
<tr>
<td>20_02_02_02_002</td>
<td>102</td>
<td>0</td>
<td>100</td>
<td>2.1</td>
<td>160.4</td>
<td>0.0%</td>
</tr>
<tr>
<td>20_02_02_02_003</td>
<td>152</td>
<td>0</td>
<td>100</td>
<td>8.1</td>
<td>624.8</td>
<td>0.0%</td>
</tr>
<tr>
<td>20_02_02_06_001</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>2.3</td>
<td>180.1</td>
<td>0.0%</td>
</tr>
<tr>
<td>20_02_10_10_005</td>
<td>1517.6</td>
<td>2.5</td>
<td>16</td>
<td>273.1</td>
<td>20952</td>
<td>0.3%</td>
</tr>
<tr>
<td>20_05_02_02_001</td>
<td>151.5</td>
<td>2.7</td>
<td>44</td>
<td>248.8</td>
<td>8070.7</td>
<td>1.7%</td>
</tr>
<tr>
<td>20_05_02_02_002</td>
<td>1111</td>
<td>0</td>
<td>100</td>
<td>90.0</td>
<td>2912.6</td>
<td>0.0%</td>
</tr>
<tr>
<td>20_10_02_02_001</td>
<td>199.9</td>
<td>2.5</td>
<td>4</td>
<td>286.9</td>
<td>4559.0</td>
<td>2.5%</td>
</tr>
<tr>
<td>20_10_10_10_005</td>
<td>739</td>
<td>0</td>
<td>100</td>
<td>12.7</td>
<td>201.6</td>
<td>0.0%</td>
</tr>
<tr>
<td>25_02_02_02_001</td>
<td>163</td>
<td>0</td>
<td>100</td>
<td>9.5</td>
<td>713.2</td>
<td>0.0%</td>
</tr>
<tr>
<td>25_05_02_02_001</td>
<td>160.4</td>
<td>3.17</td>
<td>3</td>
<td>297.2</td>
<td>9738.9</td>
<td>4.8%</td>
</tr>
<tr>
<td>25_10_02_02_001</td>
<td>198.7</td>
<td>4.7</td>
<td>0</td>
<td>280.9</td>
<td>4394.1</td>
<td>14.2%</td>
</tr>
<tr>
<td>25_10_10_10_005</td>
<td>1235.5</td>
<td>7.83</td>
<td>0</td>
<td>294.0</td>
<td>4600.4</td>
<td>2.8%</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Problem</th>
<th>Optimal solution</th>
</tr>
</thead>
</table>
| 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: 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_001    | 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  
                          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  
| 20_10_10_10_005    | Not Found  
                          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_02_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_05_02_02_001    | Not Found  
| 25_10_02_02_001    | Not Found  
| 25_10_10_10_005    | Not Found  

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 results 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. 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

<table>
<thead>
<tr>
<th>Problem</th>
<th>Mathematical model results</th>
<th>Proposed GA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total tardiness</td>
<td>Time (second)</td>
</tr>
<tr>
<td>20-02-02-02-001</td>
<td>147</td>
<td>67.2</td>
</tr>
<tr>
<td>20-02-02-02-002</td>
<td>102</td>
<td>20.43</td>
</tr>
<tr>
<td>20-02-02-02-003</td>
<td>152</td>
<td>50.46</td>
</tr>
<tr>
<td>20-02-02-06-001</td>
<td>0</td>
<td>19.63</td>
</tr>
<tr>
<td>20-02-10-10-005</td>
<td>1513</td>
<td>285</td>
</tr>
<tr>
<td>20-05-02-02-001</td>
<td>149</td>
<td>348.1</td>
</tr>
<tr>
<td>20-05-02-02-002</td>
<td>111</td>
<td>134.74</td>
</tr>
<tr>
<td>20-10-02-02-001</td>
<td>195</td>
<td>379.35</td>
</tr>
<tr>
<td>20-10-10-10-005</td>
<td>739</td>
<td>62.05</td>
</tr>
<tr>
<td>25-02-02-02-001</td>
<td>163</td>
<td>30.03</td>
</tr>
<tr>
<td>25-05-02-02-001</td>
<td>153</td>
<td>541.85</td>
</tr>
<tr>
<td>25-10-02-02-001</td>
<td>174</td>
<td>651.91</td>
</tr>
<tr>
<td>25-10-10-10-005</td>
<td>1202</td>
<td>646</td>
</tr>
</tbody>
</table>

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
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


