Research Article

A New Multiobjective Time-Cost Trade-Off for Scheduling Maintenance Problem in a Series-Parallel System

Leyla Sadat Tavassoli,1 Reza Massah,2 Arsalan Montazeri,3 Mirpouya Mirmozaffari,4 Guang-Jun Jiang,5,6 and Hong-Xia Chen5,6

1Department of Industrial Manufacturing and Systems Engineering, University of Texas at Arlington, Arlington, TX, USA
2Department of Civil Engineering, University of Texas at Arlington, Arlington, TX, USA
3Department of Chemical Engineering, The University of Isfahan, Isfahan, Iran
4Department of Industrial Manufacturing and Systems Engineering, University of Texas at Arlington, Arlington, TX, USA
5School of Mechanical Engineering, Inner Mongolia University of Technology, Hohhot, Inner Mongolia 010051, China
6Inner Mongolia Key Laboratory of Advanced Manufacturing Technology, Hohhot 010051, Inner Mongolia, China

Correspondence should be addressed to Guang-Jun Jiang; jianggj_2003@163.com

Received 2 March 2021; Revised 25 March 2021; Accepted 21 June 2021; Published 1 July 2021

Academic Editor: Wenyu Zhang

Copyright © 2021 Leyla Sadat Tavassoli et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In this paper, a modified model of Nondominated Sorting Genetic Algorithm 2 (NSGA-II), which is one of the Multiobjective Evolutionary Algorithms, is proposed. This algorithm is a new model designed to make a trade-off between minimizing the cost of preventive maintenance (PM) and minimizing the time taken to perform this maintenance for a series-parallel system. In this model, the limitations of labor and equipment of the maintenance team and the effects of maintenance issues on manufacturing problems are also considered. In the mathematical model, finding the appropriate objective functions for the maintenance scheduling problem requires all maintenance costs and failure rates to be integrated. Additionally, the effects of production interruption during preventive maintenance are added to objective functions. Furthermore, to make a better performance compared with a regular NSGA-II algorithm, we proposed a modified algorithm with a repository to keep more unacceptable solutions. These solutions can be modified and changed with the proposed mutation algorithm to acceptable solutions. In this algorithm, modified operators, such as simulated binary crossover and polynomial mutation, will improve the algorithm to generate convergence and uniformly distributed solutions with more diverse solutions. Finally, by comparing the experimental solutions with the solutions of two Strength Pareto Evolutionary Algorithm 2 (SPEA2) and regular NSGA-II, MNSGA-II generates more efficient and uniform solutions than the other two algorithms.

1. Introduction

In today’s industrial world, it is crucial for manufacturing companies to keep the production rates of machines constant. Since cost reduction and profit increase are the main goals of all manufacturing companies, the breakdown of production machines can cause a decrease in production or stop production in some cases, which will reduce the profits of companies eventually. In this situation, the need for a preventive maintenance (PM) system to keep the machines running is essential [1, 2].

Calculating the optimal time for PM actions will prevent not only the unexpected breakdown of machines but also save cost as too many maintenance operations could potentially increase the cost of production [3]. It is more complex when two preventive maintenance programs coincide. For example, labor shortage or lack of equipment for PM maintenance makes it difficult to perform the PM process on different machines simultaneously [4]. Since the maintenance team is unable to perform more than one maintenance at a time, the maintenance schedule should be changed so that no more than one maintenance is scheduled at a time [5].
Therefore, in cases where the machines’ maintenance schedules overlap, we need to change the optimal time of the parts replacement by modifying the schedule. Hence, our goal is to create a trade-off between the optimal time for preventive maintenance and the limitations of labor and equipment in order to create an executable schedule at the lowest cost.

Preventive maintenance (PM) is an old well-known problem in the manufacturing world with a proven effect on production scheduling and output [6, 7]. It had been studied from the last century, starting from the policy development of replacement which was based on the limit for the repair cost to more complex methods such as the joint optimization of predictive maintenance planning and production scheduling [8].

There are a few approaches to how PM problems can be solved. The first one is to create policies/PM schedules by maximizing system availability. It has been done by using the different methods in [9–13]. Most of the existing research on PM overlooks the interrelationship between maintenance planning and equipment work schedules. As a result, maintenance planning is often done without considering the interactions between these two activities [14, 15]. The most recent research focuses on combining production scheduling with PM scheduling. Thus, the second approach to solve a PM problem is to create a PM scheduling model that considers the production schedule of the machines as the constraints [16, 17]. Another approach is to build the production scheduling model with the PM constraints, such as in [18]. Furthermore, the joint optimization of preventive maintenance and production scheduling planning gained a lot of interest. The goal is to minimize all the costs (maintenance and production-related) and satisfy the demand while creating the optimal production and preventive maintenance schedule [19–21]. Tardiness cost can also have an impact on the production and PM planning and was added to the objective function by [22, 23]. Additionally, the problem can be extended by considering the effect of maintenance on the quality of the products or simultaneously making the quality decision together with the production and maintenance.

Any scheduling problem is known as a complex problem that includes multiple objectives that have to be minimized or maximized at the same time. However, multiple scheduling criteria are rarely considered simultaneously in the literature [24] but, in recent years, it has gained more interest through a trade-off between conflicting objectives such as in [25, 26]. Multiple objectives are typically being used in such problems as the assignment-allocation problem [27] or supply chain network design problems [28]. In production scheduling models, it is very important to find the balance between the time and the total production cost [29, 80]. To buttress this point, [81] uses the multiobjective evolutionary algorithm to minimize the total flow time of jobs and the number of tardy jobs. PM scheduling and rescheduling are also done by minimizing total operational cost (job’s total completion times, maintenance cost, and compression cost) and total completion time deviation simultaneously [30].

PM problem is categorized as an NP-hard problem as any scheduling model with a lot of constraints and large datasets. Metaheuristic is a to-go method for many scheduling/assignment problems that provides very good results in a short period of time that is a critical concern in the industry. Metaheuristic methods are a wide class of algorithms that can conduct searching phases based on their stochastic cores [31–41]. They have found their place among viral methods to be utilized for intelligent systems [42–46] and hybrid pattern recognition works [47–55]. These methods can be basic or enhanced methods with other evolutionary bases, and some examples include evolutionary algorithms, ant colony optimization (ACO) [36, 47], memetic and hybrid algorithms [56–60], tabu search, simulated annealing, etc. [47, 61]. Evolutionary algorithms have been found to be very successful in solving multiobjective optimization problems. They also have the potential to be integrated with the hybrid and new machine learning models in the future [62–67]. In the past decade, the nondominated sorting genetic algorithm (NSGA-II) has been one of the most popular and practical evolutionary multiobjective optimization (EMO) algorithms [32, 68–70].

There are many modified versions of heuristics algorithms that solve the scheduling and assignment problems, such as [35, 52, 55]. For example, in many studies, modified versions of NSGA-II were utilized in order to create algorithms to find more accurate solutions because the regular NSGA-II algorithms were not able to find it [71–74]. The other modified algorithms, such as SPEA-II and MOPSO, were generated in order to find solutions for particular problems which could not be solved by regular algorithms [36, 39, 75, 76].

In this study, we first review scheduling maintenance and cost optimization problems and integrate these two problems with other related problems in a series-parallel system. We convert these problems to a mathematical model and propose a modified-NSGA-II algorithm to get optimal solutions. Since many of the generated solutions are not acceptable due to having overlap with other solutions, this algorithm has a repository to keep these unacceptable solutions, which are changed from unacceptable to acceptable solutions by a mutation operator. Therefore, the performance of solution generation in the algorithm is increased. This mutation operator includes a complicated procedure to change the start times of preventive maintenance in solutions in order to convert unacceptable solutions to acceptable solutions in the repository.

In this modified NSGA-II algorithm, we propose to add a repository with a new mutation operator algorithm. Then we analyze the comparison of the results obtained from this algorithm with two regular algorithms.

2. Problems

2.1. Maintenance Scheduling Problem. Let us consider a factory in which several machines are working. Each piece of equipment has parts that wear out (depreciate) over time and need to be repaired or replaced. Obviously, the replacement time of these parts is different. In addition, the
time that the maintenance team spends to replace each of these parts varies. Therefore, considering the number of machines in operation and the several parts that must be repaired or replaced, the possibility of scheduling two different maintenance at the same time is very high. Due to the limited labor and equipment of the maintenance team, it is necessary to change the preventive replacement schedule when two or more maintenance coincides with each other. Potentially, it can help to prevent the accumulation of preventive maintenance over a period of time.

If we assume the total number of PM actions in a period of time is \( m \) so that the maintenance is arranged in order of its start time from 1 to \( m \), then assuming that \( n \) is a number between 0 and \( m \), the connection of the end time of the maintenance \( n \) with the start time of maintenance \( n + 1 \) can have 3 modes.

In the first case, as soon as the maintenance \( n \) is completed, the maintenance \( n + 1 \) begins. Since no time is wasted by the maintenance team between the two maintenance, it is an ideal situation.

The second case is when the period of time between PM actions \( n \) and \( n + 1 \) overlaps. Indeed, the maintenance, \( n + 1 \), begins before the completion of the maintenance, \( n \). Due to the accumulation of the PM program and the limited capacity of the maintenance team, if the volume of maintenance work in a period of time is more than the capacity of the maintenance team, it becomes impossible to perform and the schedule would have to be modified. To solve this problem, we need to shift the maintenance \( n \) slightly backward or the maintenance \( n + 1 \) slightly forward so that the overlap is eliminated and the maintenance planning becomes feasible.

In the last case, we have a gap or time interval between the end time of the maintenance \( n \) and the start time of the maintenance \( n + 1 \), which means the maintenance team has to be idle during this period. This case is acceptable for the maintenance team, but it is not ideal due to the gap. Since our goal is to perform all periodic maintenance within a defined time frame, the presence of different gaps can waste time and cost. Hence, to create the optimal schedule, we can eliminate or reduce the gap or time interval between the two PM actions to get closer to the ideal state.

2.2. Cost Optimization and Failure Rate. A machine that works in a factory has parts that wear out over time. These parts need to be replaced periodically. Since for most of the parts, the possibility of failure increases over time with aging, the late replacement of each part escalates the probability of failure of that part. Failures may result in damage to other parts and components of the machine, which causes additional costs and inactivity of the machine for a period. On the other hand, the early replacement of defective parts can impose additional costs such as increased utilization of spare parts, the maintenance team’s wages, and inactivity of the machine during the replacement period on the factory. Therefore, finding the exact time to replace defective parts is very important in reducing costs.

2.3. Production Problem in a Series-Parallel System. As mentioned earlier, keeping the production rate of machines constant is the main goal of the maintenance team. In a series-parallel system shown in Figure 1, stopping a machine due to a failure or stopping a manufacturing process due to PM activities will reduce production. This reduction rate or \( R_S \) is determined as follows:

\[
R_S = \frac{R_I}{R_T},
\]

where \( R_I \) is the amount of reduction in production and \( R_T \) is the total production rate of machines in one of subsystems 1 to \( N \) while all the machines in that subsystem are working and it can be formulated as follows:

\[
R_T = \sum_{i=1}^{N} \frac{R_n}{n},
\]

Therefore, \( R_S \) is a number between 0 and 1.

2.4. Trade-Off between Problems. As discussed in Section 2.2, early and late replacement of parts can increase costs. The exact time of replacement of each part can be calculated. The problem occurs when the maintenance team has limited working time. Therefore, it will be difficult to execute the schedule accurately. On the other hand, as mentioned in Section 2.3, keeping the production rate constant is also one of the goals of the maintenance unit.

Therefore, a trade-off between a production rate and PM costs is needed considering the limitations of labor and equipment of the maintenance team. In order to obtain this trade-off, a mathematical model is required.

3. Mathematical Model

In this paper, to create a mathematical model, we begin with identifying the problem assumptions and stating the constraints. Then the probability of failure of parts in the machine is calculated and the objective function by specifying the decision variables and the parameters are defined.

The assumptions of the problem are based on the following:

(i) The maintenance team is not able to perform a PM action on two machines at the same time
(ii) The daily and hourly efficiency of the maintenance team is constant
(iii) The maintenance team is available to perform PM action during all working hours of the company
(iv) The spare parts are available, and there are no restrictions on access to the required spare parts
(v) All required PM actions are performed within a specified time frame

where \( T_{i,j,k} \) is the time allocated for PM action in the \( k^{th} \) turn, to repair the part \( j^{th} \) of the machine \( i^{th} \).
3.1. Identification of Failure Rate. We need an exponential function to calculate the failure time of parts in machines. The Weibull function \[77\] is used to check the failure time in the depreciation state. The Weibull function is a function with continuous values. The values of this function are nonnegative real numbers, so this distribution can be used in cases where the random variable is related to longevity. Therefore, to find the failure probability density function, the cumulative distribution function (CDF) would be a good option, which can be computed as follows:

\[
    r(y) = 1 - e^{-(y/\lambda)^k}.
\]

The part's age plays a decisive role in calculating the probability of part failure. If \( r(y) \) is a function of the failure rate of the part, the probability of failure after time \( L \) is obtained from the following equation:

\[
    F(y) = \int_0^L r(y) \, dx,
\]

where \( L \) is the age of the part at time \( t \), and the age of the part at the beginning of the period is equal to 0 due to PM action.

3.2. Identification of Decision Variables and Objective Functions. The decision variable in this problem is \( L_{i,j,k} \), which is a matrix with \( i \) rows, \( j \) columns, and \( k \) dimension. In this matrix, \( i \) is the number of machines in operation, \( j \) is the number of parts of a machine that need preventive maintenance, and \( k \) is the number of times parts that need to be repaired. The number shown in each cell of this matrix is assigned to the start time of the PM action.

The duration of the PM process for each part of the machine is a fixed number and it is indicated by \( T_{i,j} \). The cost of the repair team for each hour of work on the \( j^{th} \) machine \( j \) and \( j^{th} \) part is specified as \( P_{i,j} \). The cost of PM for the part \( j^{th} \) of the machine \( i^{th} \) is obtained from the following equation:

\[
    CPM_{i,j} = \left( T_{i,j} \times (P_{i,j} + M_{i,j}) \right) + S_{i,j},
\]

where \( S_{i,j} \) is the price of spare part \( j^{th} \) of the machine \( i^{th} \) and \( M_{i,j} \) is the cost of one hour of inactivity of the machine \( i^{th} \).

Then the cost objective function is formulated as follows:

\[
    \text{minimize } Z = \sum_{i=1}^{I} \sum_{j=1}^{J} \left( \sum_{k=1}^{K} \left( CPM_{i,j} + \left( \int_0^{L_{i,j,k}} r_{i,j}(y) \, dy \right) \times C_{i,j} \right) \right),
\]

where \( C_{i,j} \) is the cost of part \( j^{th} \) breakdown of the machine \( i^{th} \). Since inactivity of the production line due to failure or PM process leads to decrease in the volume of the production, time objective function that can be formulated as follows:

\[
    \text{minimize } Z = \sum_{i=1}^{I} \sum_{j=1}^{J} \left( \sum_{k=1}^{K} \left( \int_0^{L_{i,j,k}} r_{i,j}(y) \, dy \right) \times R_{i,j} \times T_{i,j} \right) + \left( Q_{i,j} \times T_{i,j} \times R_{i} \right),
\]

where \( R_{i} \) is the cost of inactivity for the machine, which has \( T_{i,j} \) hour inactivity for the machine \( i \) and the part \( j \) and \( Q_{i,j} \) is the number of maintenance actions performed on the machine \( i \) and the part \( j \).

s.t.

\[
    \sum_{i=1}^{I} \sum_{j=1}^{J} T_{i,j,k} \leq T_{\text{total}}, \quad i, j \in N,
\]

\[
    \sum_{j=1}^{J} T_{i,j} \leq F_i, \quad i \in N,
\]

where \( T_{\text{total}} \) is the total number of hours that the maintenance team is available to perform a PM action in the company is defined by \( T_{\text{total}} \). Then we determine how many preventive maintenance activities should be done for each part of a single machine per year. It should be noted that each maintenance team has a certain capacity in terms of labor and equipment. Hence, the total time spent in a certain period should not exceed the capacity of the maintenance team. This period is usually considered to be annual—this constraint model as equation (8).
The constraint modeled by equation (9) allocates to the inactivity time of machine \( i \) in the time interval \( 0 \) to \( T_{\text{total}} \) which cannot be longer than the specified value \( F_i \). The last constraint specifies the number of assignments assigned to the maintenance team at time \( t \), denoted by \( A_t \).

4. Methodology

On a production line, we often use a significant number of machines in the process, which affects the time and cost decisions of PM actions. In order to mitigate the computational difficulties of solving the resulting large-scale problem because of the nonlinear objective functions, we apply a modified NSGA-II heuristic algorithm. The modifications in the mutation and crossover operators are applied, which generate convergence and uniformly distributed solutions.

4.1. Initialization. Before creating the crossover and mutation operators, maintaining a diverse population of candidate solutions is required. This could subsequently be incorporated into a chromosome matrix. To solve the objective functions, the solutions created for this algorithm are in the form of a three-dimensional matrix. We assign a row that corresponds to each machine \( i \) in operation, a column that represents each part \( j \) of a machine in need of PM action, and a dimension that stands for time \( k \) that parts need to be repaired. The number shown in each element indicates the start time of the PM action for the machine \( i \), the part \( j \) in the time \( k \).

4.2. Simulated Binary Crossover (SBX). The simulated binary crossover operator produces children in the vicinity of the parents [78]. In this operator, the parents are generated using the roulette wheel selection. Moreover, the two corresponding elements from both matrices are randomly selected and then for \( u \), a random value between 0 and 1 is generated. Then, the value of \( \beta \) is calculated as follows:

\[
\beta = \begin{cases} 
(2u)^{1/\eta+1}, & \text{if } u \leq 0.5, \\
\left(\frac{1}{2(1-u)}\right)^{1/\eta+1}, & \text{otherwise},
\end{cases}
\]

where \( \eta \) is the distribution index, which is a real number.

Two new children are produced using the following formula. We use the following equations to create new solution candidates:

\[
x_1 = 0.5\left[(1 + \beta)x_1 + (1 - \beta)x_2\right], \quad x_2 = 0.5\left[(1 - \beta)x_1 + (1 + \beta)x_2\right],
\]

where \( x_1 \) and \( x_2 \) are the children produced and \( x_1 \) and \( x_2 \) are the parents which are 2 elements of the decision variable matrix.

4.3. Polynomial Mutation. In polynomial mutation, it is possible to find a new solution at any distance from the parent, but it is more likely to be found around the parent than elsewhere [79]. To find the solution, first, the parameter \( \delta \) is computed as follows:

\[
\delta = \begin{cases} 
(2r)^{1/\eta+1} - 1, & \text{if } r < 0.5, \\
1 - [2(1 - r)]^{1/\eta+1}, & \text{if } r \geq 0.5,
\end{cases}
\]

where \( r \) is a random number between 0 and 1, and \( \eta \) is a positive real number that can directly control the probability distribution as an external parameter.

The mutated child is obtained from equation (14), in which \( x_U \) and \( x_L \) are the upper and lower bounds, to extract a new solution from the mutation operator.

\[
x_2 = x_1 + (x_U - x_L)\delta.
\]

4.4. Developed Mutation Algorithm (Mutation Type 2). This mutation operator is designed to convert unacceptable solutions to acceptable ones. Unacceptable solutions are solutions that overlap at least once with the other solutions. To solve this problem, we define a mutation operator (Figure 2). The mutation operator type 2 is designed by picking the first cell of the first row from the first column of the first aisle of the solutions matrix. If there is an overlap, the time interval of PM action for the cell increases or decreases to avoid any overlap with the other cells. Likewise, all cells in a row are reviewed to prevent overlapping between the cells. After reviewing one row, we will review the next row, and after reviewing all the rows of one aisle, we will review the other aisles until the last cell of the matrix is reviewed. After reviewing the last cell, the solution changes from an unacceptable solution to an acceptable solution.

4.5. Nondominated Sorting. To sort the solutions, we use the convexity crowding distance method. For every chromosome, we have the following:

\[
\sum_{i=1}^{I} \frac{f_i(x) - F_i^{\text{max}}}{F_i^{\text{max}} - F_i^{\text{min}}}
\]

where \( f_i(x) \) is the \( i \)th objective function and \( f_i^{\text{max}} \) and \( f_i^{\text{min}} \) are the highest and lowest values of the \( i \)th objective function value, respectively. \( F \) is the closest value generated to the considered objective value. Solutions with higher distance crowding are better solutions.

4.6. Repository. Since the number of acceptable solutions generated per iteration is very small, if we use standard algorithms to solve this problem, many solutions will be unacceptable due to overlap. If these solutions are eliminated, the speed of the algorithm is greatly reduced, and it takes a long time for the algorithm to find the optimal solutions for the problem. So, to solve this problem, we are looking for a new method to convert unacceptable solutions to acceptable ones.
The proposed method is to use a repository to store unacceptable solutions generated by mutation type 1 and crossover operators. Since there is a high probability that some of the solutions generated by mutation and crossover operators are unacceptable solutions because of the time overlap among solution matrix cells, these unacceptable generated solutions are stored in the repository and then after performing mutation operator type 2, they are converted into acceptable solutions. Because the time consumption of mutation operator type 2 is significant, not all solutions are sent to this operator. Therefore, after ranking solutions in the repository, a certain number of the best solutions are sent to mutation operator type 2 and then they are converted into acceptable solutions.

The cycle of entering solutions into the repository and leaving solutions from the repository is shown in Figure 3. The only repository input is the unacceptable solutions generated by crossover and mutation type 1 operators. Since the capacity of the repository is limited and it is not possible to store all unacceptable solutions generated in the repository, in each iteration, a number of lower-ranked solutions in the repository are removed so that the repository has enough space to store the solutions generated in the next iteration. The number of removed solutions in each iteration

![Figure 2: Developed mutation algorithm.](image-url)
depends on the number of solutions entered into the repository and the number of solutions removed from the repository so that the number of solutions stored in the repository does not exceed the capacity of the repository.

The use of a repository in the NSGA2 algorithm also has disadvantages. The first drawback is to increase the computation time for each iteration of the algorithm. Although the presence of a repository in the algorithm causes more optimal solutions to be generated by the algorithm in each iteration, more significant time per iteration in the MNSGA2 algorithm is spent compared to the regular NSGA2 algorithm. It is because of the time required to perform operations in the type 2 mutation operator. A comparison between the time of each iteration between the algorithms is performed in the numerical example and result section. Another disadvantage of using this method is that many of the solutions generated by the type 2 mutation operator may no longer be the optimal solutions. The type 2 mutation operator converts unacceptable solutions into acceptable ones by changing the values of each solution matrix cell that overlaps with other cells in the same solution matrix, but these changes may result in to change the ranking of solutions.

4.7. Modified NSGA-II Algorithm. The idea behind the proposed algorithm is to increase the speed of generating optimal solutions. By adding a repository and a developed mutation algorithm (mutation type 2) to the regular NSGA-II algorithm, the modified NSGA-II algorithm is created, which has an extra loop compared to the regular NSGA-II algorithm. In the new algorithm, unacceptable optimal solutions that are in danger of being deleted because of having overlap between one cell in matrix solution and another cell in the same matrix solution are stored in the repository. These solutions are converted to acceptable solutions with the help of the mutation type 2 operator and then added to the candidate solutions. We conduct a modified NSGA-II algorithm that is illustrated in Figure 4.

5. Numerical Example and Result

We used the real data collected from a company that produced car spare parts for the computational experiments. In this company, there are 40 machines that are producing metal parts in a production line. This production line has 10 stations where there are 4 machines in each station. All machines are milling or turning machine tools that are under preventive maintenance scheduling.

These machines have some moving parts such as spindle, tool change, turret, pallet change, and moving table, all of which are depreciated due to their mobility. In addition to the mobility of these parts, machining vibration, corrosion of parts due to use of cooling fluid, and damage of machine parts because of metal chips increase the depreciation of machine parts. Typically, ball bearings, spindles, various parts of the cooling system, tool change, and pallet change are subject to preventive maintenance.

5.1. Results for Modified-NSGA-II, NSGA-II, and SPEA-II. We initialized the algorithm by randomly generating a population of 100 solutions. The number of the initial population and the solutions provided by mutation and crossover operators, repository capacity, and the maximum number of unacceptable solutions sent from repository to operator mutation type 2 would be different based on their importance on the problem. For this problem, values of algorithm and repository specifications are demonstrated in Table 1.

To assess the performance of our Modified-NSGA-II algorithm and compare it with the two algorithms NSGA-II and SPEA-II, we defined a reference set based on 100 best candidate solutions obtained by the Modified-NSGA-II, NSGA-II, and SPEA-II. Each was replicated 1000 times after every 100 iterations. Table 2 illustrates the number of nondominated solutions (NNS) by reference in all three algorithms, and as shown in Table 2, the Modified-NSGA-II algorithm has more nondominated solutions than the other.
Figure 4: Modified-NSGA-II algorithm.

Table 1: Algorithm and repository specifications.

| Algorithm specification                                           | Number of produced populations |
|-------------------------------------------------------------------|--------------------------------|
| Initial population                                                | 100                            |
| Mutation (type 1)                                                 | 30                             |
| Crossover                                                         | 30                             |
| Repository capacity                                               | 500                            |
| Maximum number of unacceptable solutions sent from repository to operator mutation type 2 in each iteration | 20                             |
two algorithms. We conducted the experiment using the MATLAB platform on a Windows-based server with 16 GB RAM, i7 CPU, and 1.8GHz.

For further assessment, we compared the performance of the Modified-NSGA-II algorithm with the regular NSGA-II and SPEA-II algorithms using three methods: inverted generated distance, distribution metric, and spacing metric. The results of the comparison among the MNSGA-II, the NSGA-II, and the SPEA-II can be seen in Table 3.

5.2. Inverted Generated Distance. This approach is designed to assess diversity and convergence and can be calculated as follows:

$$
IGD = \frac{\sum_{v \in P} d(v, P)}{|P^*|},
$$

where $P^*$ is equal to uniformly distributed points in a true Pareto front, $P$ is a nondominated solution obtained by a selected algorithm, $v$ is a solution that belongs to $P^*$, and $0$ ($v, P$) is equal to the minimum Euclidean distance between $v$ and the point in $P$. The results shown in Table 3 illustrate that, in general, the modified-NSGA-II algorithm has lower IGD values than the other two algorithms.

5.3. Delta Index. In the delta approach, the distribution of the solutions in the Pareto front is examined considering the following equation:

$$
\Delta(s) = \frac{\sum_{i=1}^{s-1} |d_i - \bar{d}|}{|s| - 1},
$$

where $d_i$ is the Euclidean distance between two consecutive solutions of Pareto front with $s$ optimal solutions and $\bar{d}$ is the average $d_i$. The lower the value of delta is, the more uniformly the Pareto front is distributed. According to the values specified in Table 3, with a slight difference after the SPEA-II algorithm, the modified-NSGA-II algorithm has the lowest value.

5.4. Spacing Metrics. The uniformity of the distance between the Pareto front and the reference set is analyzed in the Spacing Metric method. A lower space parameter (SP) value indicates a more uniform distance between the Pareto front and the reference. The SP value is calculated as follows:

$$
SP(s) = \sqrt{\frac{1}{|S| - 1} \sum_{i=1}^{|S|} (\bar{d} - d_i)^2},
$$

In this case, $d_i$ is the minimum distance of solution $i$ from the reference point. $\bar{d}$ is the average of $d_i$ for $s$ optimal solution set. According to the values specified in Table 3, with a significant difference compared to the other two algorithms, the SPEA2 algorithm has a uniform distance between the reference and Pareto front.

5.5. Operation Time. As mentioned in Section 4.6, although the MNSGA2 algorithm generates better solutions per iteration compared to the SPEA2 and NSGA2 algorithms due to the existence of a repository, it requires more time to perform the calculations. The more complexity the type 2 mutation operator has, the more time is required to perform calculations. Also, the more machines and parts that need preventive maintenance, the more cells the solution matrix has. In this case, the probability of overlap’s existence is higher, and the probability of unacceptable solutions’ existence increases as well. Also, more calculations in the type 2 mutation operator are required to make changes in the solution matrix and produce an acceptable solution. The increase in the computational volume in the type 2 mutation operator increases the whole operation time of the algorithm for each iteration. Figure 5 represents the computation time for this problem between MNSGA2, NSGA2, and SPEA2 algorithms. In Figure 5, the average operation time of NSGA2, MNSGA2, and SPEA2 algorithms is shown for 10 operations so that each one has 1000 iterations.

5.6. Fitness of Generated Solutions. Another problem that the MNSGA2 algorithm has in comparison with the two NSGA2 and SPEA2 algorithms is the change in the fitness or rank of the solutions after converting the unacceptable solutions into the acceptable ones in the type 2 mutation operator. As mentioned earlier, the type 2 mutation operator converts unacceptable solutions into acceptable ones by changing the values of the solution matrix cells that have overlapped with other cells, but after examining the fitness of the solutions, it becomes clear that the values obtained may no longer be our optimal solutions and may be dominated by other solutions obtained in the Pareto front. These solutions are removed after ranking and do not help speed up the algorithm to find optimal solutions. In this case, 73.6% of the total solutions generated by the mutation type 1 and crossover operators

| Algorithm   | IGD    | Delta | Spacing |
|-------------|--------|-------|---------|
| MNSGA-II    | 244900 | 7684  | 6       |
| NSGA-II     | 2324000| 15393 | 408     |
| SPEA2       | 7814000| 6398  | 6110    |
were unacceptable, and 68.7% of the solutions generated by the type 2 mutation operator were dominated by other Pareto front solutions. These results are obtained from the average 10 times operation of the MNSGA2 algorithm, which has 1000 iterations each operation.

### 6. Conclusion

In this study, we developed a model to solve the scheduling problem in preventive maintenance for series-parallel systems. In this model, we addressed the problems in PM process scheduling in series-parallel systems and then turned this integrated model into a mathematical model. In this mathematical model, we considered a matrix as a decision variable where each cell represents the time scheduling of the PM process for a specific part of a single machine. In the future, the scale of the mathematical model can be changed by adding more decision variables to obtain more results. Then, we defined the objective functions in order to find the optimal solutions. Moreover, we proposed a modified genetic algorithm with customized mutation and crossover operators that allowed us to find a better solution compared to the existing algorithms. We implemented our experiments on a case study drawn from the real data then analyzed and compared the solutions.

Our results confirm the better performance of the MNSGA-II algorithm, which proposes that evolutionary algorithms alone do not necessarily provide more viable optimal solutions and need customized operators for better computational efficiency. In our future works, we will consider more objective functions like the scheduling of special equipment’s use in the PM process, the spare parts supply, and the employment of specially trained personnel by utilizing the modified NSGA-II that presented in this study. Analysis of the scheduling time of PM action provides manufacturing companies with valuable information to determine meticulous scheduling decisions and planning, prevent production interruptions, and increase the amount of production.

### Abbreviations

**Parameters of mathematical model**

- $R_i$: Amount of reduction in production
- $R_T$: Total production rate of machines
- $R_s$: Reduction rate
- $T_{\text{total}}$: Total number of maintenance team working hours
- $T_{i,j,k}$: Time allocated for PM action in the $k$ th turn, to repair the part $j$ th of the machine $i$ th
- $S_{i,j}$: Price of spare part $j$ th of the machine $i$ th
- $CPM_{i,j}$: Cost of PM for the part $j$ th of the machine $i$ th
- $P_{i,j}$: Hours of work on the $i$ th machine and $j$ th part
- $C_{i,j}$: Cost of part $j$ ’s breakdown of the machine $i$ th
- $T_{i,j}$: Hours of inactivity for the machine $i$ and the part $j$
- $Q_{i,j}$: Number of maintenance actions performed on the machine $i$ and the part $j$
- $R_{i}$: Cost of inactivity for machine $i$.

### Data Availability

No data were used to support this study.

### Conflicts of Interest

The authors declare that they have no conflicts of interest.

### Acknowledgments

This work was partially supported by the National Natural Science Foundation of China under the contract number 71761030 and the Natural Science Foundation of Inner Mongolia under the contract number 2019LH07003.

### References

[1] F. Fallahi, M. Yildirim, J. Lin, and C. Wang. "Predictive multi-microgrid generation maintenance: formulation and impact on operations & resilience," *IEEE Transactions on Power Systems*, vol. 36, p. 1, 2021.

[2] M. Mirmozaffari, R. Yazdani, E. Shadkam, L. S. Tavassoli, and R. Massah, "VCS and CVS: new combined parametric and non-parametric operation research models,” *Sustainable Operations and Computers*, vol. 2, pp. 36–56, 2021.

[3] M. Yazdi, "Acquiring and sharing tacit knowledge in failure diagnosis analysis using intuitionistic and Pythagorean assessments," *Journal of Failure Analysis and Prevention*, vol. 19, no. 2, pp. 369–386, 2019.

[4] N. Wang, S. Ren, Y. Liu, M. Yang, J. Wang, and D. Huisinighi, “An active preventive maintenance approach of complex equipment based on a novel product-service system operation mode,” *Journal of Cleaner Production*, vol. 277, Article ID 123365, 2020.

[5] M. Yazdi, N. A. Golilarz, A. Nedjati, and K. A. Adesina, "An improved lasso regression model for evaluating the efficiency of intervention actions in a system reliability analysis," *Neural Computing and Applications*, vol. 33, pp. 1–16.

[6] L. S. Tavassoli, N. Sakhavand, and S. S. Fazeli, “Integrated preventive maintenance scheduling model with redundancy for cutting tools on a single machine,” *Engineering, Technology & Applied Science Research*, vol. 10, no. 6, pp. 6542–6548, 2020.
S. Chen and L. Wang, “ACO-based parallel machine scheduling and production planning,” *Robots and Computer-Integrated Manufacturing*, vol. 51, pp. 238–247, 2018.

J. Pulido, “Using accelerated life testing techniques for preventive maintenance scheduling,” in *Proceedings of the 2014 Reliability and Maintainability Symposium*, pp. 1–6, IEEE, Colorado Springs, CO, USA, January 2014.

Y. Gao, Y. Feng, Z. Zhang, and J. Tan, “An optimal dynamic interval preventive maintenance scheduling for series systems,” *Reliability Engineering & System Safety*, vol. 142, pp. 19–30, 2015.

A. Abdelhadi, “Preventive maintenance operations scheduling based on Eigenvalue and clustering methods,” in *Proceedings of the 2019 IEEE 6th International Conference on Industrial Engineering and Applications (ICIEA)*, pp. 183–186, IEEE, Tokyo, Japan, January 2019.

M. Hamedi, S. Sharafi, and M. T. Delshad, “Optimization of preventive maintenance scheduling based on Monte Carlo simulation in processing plants,” in *Proceedings of the 2019 15th Iran International Industrial Engineering Conference (IIIEC)*, pp. 293–296, IEEE, Yazd, Iran, January 2019.

W. Shan, Z. Qiao, A. A. Heidari, H. Chen, H. Turabieh, and Y. Teng, “Double adaptive weights for stabilization of moth flame optimizer: balance analysis, engineering cases, and medical diagnosis,” *Knowledge-Based Systems*, vol. 214, Article ID 106684, 2021.

J. Hu, H. Chen, A. A. Heidari et al., “Orthogonal learning based preventive maintenance policy for series production lines,” *Expert Systems with Applications*, vol. 160, Article ID 113701, 2020.

S. Chen and L. Wang, “ACO-based parallel machine scheduling considering both setup time and run-based preventive maintenance with reliability constraints,” in *Proceedings of the 2018 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pp. 227–231, IEEE, Bangkok, Thailand, December 2018.

S. Chen and L. Wang, “ACO-based parallel machine scheduling and imperfect preventive maintenance policy for single machine system,” *International Journal of Operational Research*, vol. 18, no. 2, pp. 140–156, 2013.

M.-C. Fitouhi and M. Nourafkh, “Integrating noncyclical preventive maintenance scheduling and production planning for multi-state systems,” *Reliability Engineering & System Safety*, vol. 121, pp. 175–186, 2014.

A. Wolter and S. Helber, “Simultaneous production and maintenance planning for a single capacitated resource facing both a dynamic demand and intensive wear and tear,” *Central European Journal of Operations Research*, vol. 24, no. 3, pp. 489–513, 2016.

J. Ye and H. Ma, “Multiobjective joint optimization of production scheduling and maintenance planning in the flexible job-shop problem,” *Mathematical Problems in Engineering*, vol. 2015, Article ID 725460, 9 pages, 2015.
[37] H. Chen, A. A. Heidari, H. Chen, M. Wang, Z. Pan, and A. H. Gandomi, “Multi-population differential evolution-assisted Harris hawks optimization: framework and case studies,” Future Generation Computer Systems, vol. 111, pp. 175–198, 2020.

[38] X. Zhao, X. Zhang, Z. Cai et al., “Chaos enhanced grey wolf optimization wrapped ELM for diagnosis of paraquat-poisoned patients,” Computational Biology and Chemistry, vol. 78, pp. 481–490, 2019.

[39] Y. Xu, H. Chen, J. Luo, Q. Zhang, S. Jiao, and X. Zhang, “Enhanced Moth-flame optimizer with mutation strategy for global optimization,” Information Sciences, vol. 492, pp. 181–203, 2019.

[40] M. Wang and H. Chen, “Chaotic multi-swarm whale optimizer boosted support vector machine for medical diagnosis,” Applied Soft Computing Journal, vol. 88, Article ID 105946, 2020.

[41] M. Wang, H. Chen, B. Yang et al., “Toward an optimal kernel extreme learning machine using a chaotic moth-flame optimization strategy with applications in medical diagnoses,” Neurocomputing, vol. 267, pp. 69–84, 2017.

[42] W. Zhu, C. Ma, X. Zhao et al., “Evaluation of sino foreign cooperative education project using orthogonal sine cosine optimized kernel extreme learning machine,” IEEE Access, vol. 8, pp. 61107–61123, 2020.

[43] G. Liu, W. Jia, M. Wang et al., “Predicting cervical hyperextension injury: a covariance guided sine cosine support vector machine,” IEEE Access, vol. 8, pp. 46895–46908, 2020.

[44] Y. Wei, H. Lv, M. Chen et al., “Predicting entrepreneurial intention of students: an extreme learning machine with Gaussian Barebone Harris Hawks optimizer,” IEEE Access, vol. 8, pp. 76841–76855, 2020.

[45] H. Tang, Y. Xu, A. Lin et al., “Predicting green consumption behaviors of students using efficient firefly grey-assisted K-nearest neighbor classifiers,” IEEE Access, vol. 8, pp. 35546–35562, 2020.

[46] A. Lin, Q. Wu, A. A. Heidari et al., “Predicting intentions of students for master programs using a chaos-induced sine cosine-based fuzzy K-Nearest neighbor classifier,” IEEE Access, vol. 7, pp. 67235–67248, 2019.

[47] X. Zhao, D. Li, B. Yang, C. Ma, Y. Zhu, and H. Chen, “Feature selection based on improved ant colony optimization for online detection of foreign fiber in cotton,” Applied Soft Computing, vol. 24, pp. 585–596, 2014.

[48] Y. Zhang, R. Liu, A. A. Heidari et al., “Towards augmented kernel extreme learning models for bankruptcy prediction: algorithmic behavior and comprehensive analysis,” Neurocomputing, vol. 430, pp. 185–212, 2020.

[49] Y. Zhang, R. Liu, X. Wang, H. Chen, and C. Li, “Boosted binary Harris hawks optimizer and feature selection,” Engineering with Computers, vol. 36, pp. 1–30, 2020.

[50] H.-L. Chen, G. Wang, C. Ma, Z.-N. Cai, W.-B. Liu, and S.-J. Wang, “An efficient hybrid kernel extreme learning machine approach for early diagnosis of Parkinson’s disease,” Neurocomputing, vol. 184, pp. 131–144, 2016.

[51] L. Hu, G. Gong, J. Ma, X. Wang, and H. Chen, “An efficient machine learning approach for diagnosis of paraquat-poisoned patients,” Computers in Biology and Medicine, vol. 59, pp. 116–124, 2015.

[52] L. Shen, H. Chen, Z. Yu et al., “Evolving support vector machines using fruit fly optimization for medical data classification,” Knowledge-Based Systems, vol. 96, pp. 61–75, 2016.

[53] J. Xia, H. Chen, Q. Li et al., “Ultrasound-based differentiation of malignant and benign thyroid Nodules: an extreme learning machine approach,” Computer Methods and Programs in Biomedicine, vol. 147, pp. 37–49, 2017.

[54] C. Li, L. Hou, B. Y. Sharma et al., “Developing a new intelligent system for the diagnosis of tuberculous pleural effusion,” Computer Methods and Programs in Biomedicine, vol. 153, pp. 211–225, 2018.

[55] X. Xu and H.-L. Chen, “Adaptive computational chemotaxis based on field in bacterial foraging optimization,” Soft Computing, vol. 18, no. 4, pp. 797–807, 2014.

[56] Y. Fan, P. Wang, A. A. Heidari et al., “Rationalized fruit fly optimization with sine cosine algorithm: a comprehensive analysis,” Expert Systems with Applications, vol. 157, Article ID 113486, 2020.

[57] E. Rodriguez-Esparza, L. A. Zanella-Calzada, D. Oliva et al., “An efficient Harris hawks-inspired image segmentation method,” Expert Systems with Applications, vol. 155, Article ID 113428, 2020.

[58] S. Jiao, G. Chong, C. Huang et al., “Orthogonally adapted Harris hawks optimization for parameter estimation of photovoltaic models,” Energy, vol. 203, Article ID 117804, 2020.

[59] Z. Xu, Z. Hu, A. A. Heidari et al., “Orthogonally-designed adapted grasshopper optimization: a comprehensive analysis,” Expert Systems with Applications, vol. 150, Article ID 113282, 2020.

[60] M. Mahjoob, S. S. Fazeli, S. Milanloquei, and L. S. Tavassoli, “A modified adaptive genetic algorithm for multi-product multi-period inventory routing problem,” 2021, https://arxiv.org/abs/2104.09031.

[61] Z. Mujkic, A. Qorri, and A. Kraslawski, “Sustainability and optimization of supply chains: a literature review,” Operations and Supply Chain Management: An International Journal, vol. 11, no. 4, pp. 186–199, 2018.

[62] X. Zhang, J. Wang, T. Wang, R. Jiang, J. Xu, and L. Zhao, “Robust feature learning for adversarial defense via hierarchical feature alignment,” Information Sciences, vol. 560, pp. 256–270, 2020.

[63] X. Zhang, T. Wang, W. Luo, and P. Huang, “Multi-level fusion and attention-guided CNN for image dehazing,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 30, p. 1, 2020.

[64] X. Zhang, M. Fan, D. Wang, P. Zhou, and D. Tao, “Top-k feature selection framework using robust 0-1 integer programming,” IEEE Transactions on Neural Networks and Learning Systems, vol. 31, pp. 1–15, 2020.

[65] X. Zhang, D. Wang, Z. Zhou, and Y. Ma, “Robust low-rank tensor recovery with rectification and alignment,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 43, no. 1, pp. 238–255, 2019.

[66] X. Zhang, R. Jiang, T. Wang, and J. Wang, “Recursive neural network for video deblurring,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 30, p. 1, 2020.

[67] X. Zhang, T. Wang, J. Wang, G. Tang, and L. Zhao, “Pyramid channel-based feature attention network for image dehazing,” Computer Vision and Image Understanding, vol. 197–198, Article ID 103003, 2020.

[68] F. Zhao, L. Huan, Y. Zhang, W. Ma, and C. Zhang, “A novel multiobjective optimization algorithm based on differential evolution and NSGA-II,” in Proceedings of the 2018 IEEE 22nd International Conference on Computer Supported Cooperative Work in Design ((CSCWD)), pp. 570–575, IEEE, Nanjing, China, May 2018.

[69] N. A. Gollarz, A. Addeh, H. Gao et al., “A new automatic method for control chart patterns recognition based on
ConvNet and harris hawks meta heuristic optimization algorithm,” *IEEE Access*, vol. 7, pp. 149398–149405, 2019.

[70] N. Amiri Golilarz, H. Gao, and H. Demirel, "Satellite image de-noising with Harris hawks meta heuristic optimization algorithm and improved adaptive generalized Gaussian distribution threshold function,” *IEEE Access*, vol. 7, pp. 57459–57468, 2019.

[71] S. Bandyopadhyay and R. Bhattacharya, “Solving multi-objective parallel machine scheduling problem by a modified NSGA-II,” *Applied Mathematical Modelling*, vol. 37, no. 10-11, pp. 6718–6729, 2013.

[72] F. T. S. Chan, A. Jha, and M. K. Tiwari, "Bi-objective optimization of three echelon supply chain involving truck selection and loading using NSGA-II with heuristics algorithm,” *Applied Soft Computing*, vol. 38, pp. 978–987, 2016.

[73] S. H. R. Pasandideh, S. T. A. Niaki, and K. Asadi, “Bi-objective optimization of a multi-product multi-period three-echelon supply chain problem under uncertain environments: NSGA-II and NRGA,” *Information Sciences*, vol. 292, pp. 57–74, 2015.

[74] S. Bandyopadhyay and R. Bhattacharya, “Applying modified NSGA-II for bi-objective supply chain problem,” *Journal of Intelligent Manufacturing*, vol. 24, no. 4, pp. 707–716, 2013.

[75] A. K. Paul and P. C. Shill, "New automatic fuzzy relational clustering algorithms using multiobjective NSGA-II,” *Information Sciences*, vol. 448-449, pp. 112–133, 2018.

[76] M. Wang and H. Chen, “Chaotic multi-swarm whale optimizer boosted support vector machine for medical diagnosis,” *Applied Soft Computing*, vol. 88, Article ID 105946, 2020.

[77] D. P. Murthy, M. Xie, and R. Jiang, *Weibull Models*, John Wiley & Sons, Hoboken, NJ, USA, 2004.

[78] S. Ramesh, S. Kannan, and S. Baskar, "Application of modified NSGA-II algorithm to multiobjective reactive power planning.” *Applied Soft Computing*, vol. 12, no. 2, pp. 741–753, 2012.

[79] K. Deb, "Multiobjective optimisation using evolutionary algorithms: an introduction,” in *Multiobjective Evolutionary Optimisation for Product Design and Manufacturing*, pp. 3–34, Springer, London, UK, 2011.

[80] M. Mahjoob, S. S. Fazeli, S. Milanlouei, A. K. Mohammadzadeh, and L. S. Tavassoli, "Green supply chain network design with emphasis on inventory decisions,” *Sustainable Operations and Computers*, vol. 2, 2021.

[81] M. Mahjoob, S. S. Fazeli, L. S. Tavassoli, M. Mirmohammadzada, and S. Milanlouei, "A green multi-period inventory routing problem with pickup and split delivery: a case study in flour industry,” *Sustainable Operations and Computers*, vol. 2, 2021.