Received 16 April 2023, accepted 11 May 2023, date of publication 22 May 2023, date of current version 9 June 2023.
Digital Object Identifier 10.1109/ACCESS.2023.3278372

RESEARCH ARTICLE

A Novel Metaheuristic Hybrid Parthenogenetic Algorithm for Job Shop Scheduling Problems: Applying an Optimization Model

ATEFEH MOMENIKORBEKANDI AND MAYSAM F. ABBOD, (Senior Member, IEEE)
Department of Electronic and Electrical Engineering, Brunel University London, UB8 3PH Uxbridge, U.K.
Corresponding author: Atefeh Momenikorbekandi (Atefeh.Momeni@outlook.com)

ABSTRACT Metaheuristics are primarily developed to explore optimization techniques in many practice areas. Metaheuristics refer to computational procedures leading to finding optimal solutions to optimization problems. Due to the increasing number of optimization problems with large-scale data, there is an ongoing demand for metaheuristic algorithms and the development of new algorithms with more efficiencies and improved convergence speed implemented by a mathematical model. One of the most popular optimization problems is job shop scheduling problems. This paper develops a novel metaheuristic hybrid Parthenogenetic Algorithm (NMHPGA) to optimize flexible job shop scheduling problems for single-machine and multi-machine job shops and a furnace model. This method is based on the principles of genetic algorithm (GA), underlying the combinations of different types of selections, proposed ethnic GA, and hybrid parthenogenetic algorithm. In this paper, a parthenogenetic algorithm (PGA) combined with ethnic selection GA is tested; the parthenogenetic algorithm version includes parthenogenetic operators: swap, reverse, and insert. The ethnic selection uses different selection operators such as stochastic, roulette, sexual, and aging; then, top individuals are selected from each procedure and combined to generate an ethnic population. The ethnic selection procedure is tested with the PGA types on a furnace model, single-machine job shops, and multi-machines with tardiness, earliness, and due date penalties. A comparison of obtained results of the established algorithm with other selection procedures indicated that the NMHPGA is achieving better objective functions with faster convergence speed.

INDEX TERMS A novel metaheuristic hybrid parthenogenetic algorithm, genetic algorithm, single-machine job shop, multi-machine job shop, metaheuristic optimization.

I. INTRODUCTION

Optimization is a design problem that demands appropriate techniques and methods to provide satisfactory results for a reasonable period to reduce costs. In practice, many design problems are complex, and classical optimization methods based on mathematical features cannot find the best results in a limited period. The most common mathematical methods for optimization are Gradient-based methods, which utilize the objective function. Recent research shows a growing interest in optimization with enhanced efficiency, accuracy, and speed rate for tackling optimization. One of these optimization methods is "Metaheuristic" [1], [2].

This paper focuses on job shop scheduling problems which are examples of optimization problems in the industry. In the following sections, scheduling and job shop scheduling problems will be discussed. Scheduling refers to controlling workloads in a production process to allocate machinery and human resources and plan production processes optimally. Scheduling has a vital role in the manufacturing process since it affects the productivity of the production line by reducing time and energy consumed in the production process; scheduling problems do not have fixed, efficient solution algorithms due to the complexity of these problems [3], [4].

The associate editor coordinating the review of this manuscript and approving it for publication was Giambattista Gruosso.

Scheduling aims to find an optimal processing order to reduce the total makespan of the job shop. A job shop refers to a place where each job is processed on a machine in a limited time. Based on the arrival pattern of the jobs, there are different types of job shops, namely static and dynamic scheduling problems. In static job shops, the jobs need to arrive at the idle shop and be scheduled; in contrast, in the dynamic model, the jobs arrive randomly, and job arrivals are intermittent [5], [6].

Job shop scheduling problems are classified as non-deterministic polynomial-time (NP-hard) problems; such problems are known as 'Hard' problems to solve because as the size of the problem increases linearly, the computation time increases exponentially [3]; in order to solve such complex production scheduling problems, most of the studies focus on the use of artificial intelligence techniques, heuristics and metaheuristic techniques such as neural networks, fuzzy logic, genetic algorithms (GA), particle swarm optimization, simulated annealing, etc. One of the most common techniques to deal with job shop scheduling problems is a genetic algorithm. GA has been the most popular technique in evolutionary computation research. In the traditional GA, the representation used is a fixed-length bit string. Each position in the string represents a particular feature of an individual. Usually, the string is a collection of structural features of a solution with little or no interactions [5], [6].

This paper establishes an intelligence algorithm to deal with flexible job shop scheduling problems via genetic algorithm methodologies based on a parthenogenetic algorithm replacing the crossover operator with three functions: swap, reverse, and insert. The established GA, named novel metaheuristic hybrid parthenogenetic algorithm (NMHPGA), is a combination of ethnic selection GA in which four types of selection, namely stochastic, roulette, sexual, and aging, are combined with the parthenogenetic algorithm applying three functions of PGA, namely swap, reverse, and insert.

A summary of this paper is as follows. Section II discusses the literature and the background of metaheuristics, job shop scheduling, and genetic algorithm. In Section III, the ethnic selection outline is presented. In section IV, the established NMHPGA is discussed. Section V presents some mathematical functions with different characteristics for further utilization in evaluating the developed metaheuristic algorithm, along with other alternative approaches. In section VI, a comprehensive statistical analysis is conducted to compare the results of the new algorithm with the different metaheuristic approaches. Section VII discusses the NMHPGA results tested on a case study of the furnace model. Section VIII presents this paper's main findings, including the conclusions and suggestions for future challenges.

II. LITERATURE REVIEW

A. METAHEURISTIC AND OPTIMIZATION

Metaheuristic was developed by Glover [1] in 1986; the term comprises two main words. Heuristics comes from an old Greek word, "heuristic", meaning to discover, while "meta" means beyond the ordinary or natural limits of something. Metaheuristics are optimization solution techniques that apply higher-level strategies into search processes of designed problems to find optimal solutions avoiding local optima [1], [4], [7], [8].

The history of using metaheuristics is categorized into five distinct periods [1], [9]. In the first period, there was no formal presentation of metaheuristics methods. However, these methods were used for simple optimization problems. The second period, from 1940 to 1980, was the first formal introduction of metaheuristics. In the third period (1980 to 2000), multiple metaheuristics were proposed for specific applications. The metaheuristic methodology was successfully presented in the fourth period, which is from 2000 until now the fifth period, called the "scientific" or "future" period, the designing of new metaheuristics will turn into a matter of science [1], [9].

There are four main categories of metaheuristics in terms of their inspiration:

The first category is "evolutionary algorithms", including the Genetic Algorithm (GA) [9], [10], [11], [13], Memetic Algorithm [7], [8], [13], Differential Evolution [9], [12], [13] and the evolution strategies [10] which are based on biological evolution [14]. In the 1940s, prior to the invention of computers [11], the application of Darwinian Principles (the natural process of evolution) to the approach to scientific problems arose [15].

The second category is swarm intelligence-based algorithms which are based on the cooperative behavior of decentralized and self-organized natural or artificial systems. Some examples of this category are as follows: Ant Colony Optimization [16], Particle Swarm Optimization [17], [18], [19], Cat Swarm Optimization [20], Artificial Bee Colony [21] and firefly algorithm [22], Cuckoo Search [23].

The third category of algorithms is motivated by physical laws; moreover, some methods are based on the lifestyle of humans and animals, which are categorized in the fourth category [9], [24].

The application of GA to the job shop scheduling problem is discussed in the forthcoming catagory.

B. SCHEDULING AND JOB SHOP SCHEDULING PROBLEMS

In today's complex manufacturing environment with multiple product lines, each process requires numerous steps and machines for completion; the manufacturing plant's decision-maker should find a way to manage resources to produce products as efficiently as possible effectively. The decision-maker would create a production schedule that prioritizes on-time delivery and minimizes objectives such as a product's flow time. As a result of increased demand, a field of study known as scheduling problems has been developed [25], [26], [27].

Scheduling problems involve finding the optimal schedule under different objectives, machine environments, and job characteristics. Numerous manufacturing processes are complex and extremely difficult to solve with conventional optimization techniques. They are NP-difficult problems that have set the stage for the application of genetic algorithms to such problems. Among the various scheduling problems, there are (1) Job shop scheduling, (2) multiprocessor scheduling, (3) multitask scheduling, (4) parallel machine scheduling, (5) group job scheduling, and (6) resource-constrained project scheduling and dynamic tasks [5].

C. TYPES OF SCHEDULES

Scheduling is the process of sequencing actions to make the execution optimal; scheduling is classified as a non-deterministic polynomial-time (NP-hard) problem, which refers to a tricky optimization problem to be solved [28], [29], [30], [31]. In this context, job shop scheduling problems (JSSP) are considered one of the most popular machine scheduling problems. They have received considerable attention since they involve a challenging optimization problem with many real-world applications. The production schedule has been subjected to many studies in recent years due to the importance of productivity and sustainability in the manufacturing system [26]. Generally, in a classical $n \times m$ job shop, we have n jobs $J_1, \ J_2 \ of, \ \ldots, \ J_n$ Of different processing times scheduled on m machines. In the specific variant of job shop scheduling, which is precedence constraints, each job has a set of operations $O_1, \ O_2, \ \ldots, \ O_n$ Processing within a particular order. A common type is flexible job shop, where each operation can be processed on any machine.

Furthermore, there is another classification of job shop scheduling, including single machines and flexible multi-machines [28], [29], [30], [31]. This paper focuses on single-machine job shops and multi-machine job shops.

1) SINGLE-MACHINE SCHEDULING PROBLEM WITH TARDINESS AND EARLINESS

Single-resource scheduling with tardiness and earliness penalties is a particular scheduling problem in which each job has a single operation. This model allows several jobs to be optioned at zero timing in a single resource system [5].

The single machine model against common due date is developed, considering that several jobs have to be processed on a single machine where each job has only one operation. All jobs must be ready to be processed at time zero, and for any job finished before the expected due date, the earliness penalty will be applied [5];

The maximum lateness (L_{max}) is the most significant delay for all due dates, calculated as the maximum value o $L_1 L_n$. The total weighted completion time ($\Sigma w_j C_j$) is the sum of the weighted completion times of all n jobs, providing an estimate of the total holding or inventory expenses incurred by the schedule. The sum of completion times is often referred to as flow time, while the weighted sum of completion times is called flow time.

A more general cost function is the discounted total weighted completion time $\Sigma w_j \left(1 - e^{-rC_j}\right)$, which considers costs discounted at a rate of r ($0 < r < 1$) per unit time if job j is not completed by time t, an additional cost of w_j is incurred during the period $[t, t + dt]$. If the job j is completed at time t, the total cost incurred over the period $[0, t]$ i $w_j \left(-e^{-rt}\right)$. The total weighted tardiness ($\Sigma w_j T_j$) is another cost function more general than the total weighted completion time. The weighted number of tardy jobs ($\Sigma w_j U_j$) is a metric of interest as it is simple to record the objective function. However, recent research has focused on objective functions that are not regular, such as earliness penalties, where the earliness of job j is as below when the due date is d_j This penalty decreases as C_j increases; $E_j = max\left(d_j - C_j, 0\right)$; An example of a non-regular objective is total earliness plus total tardiness, $\sum_{j=1}^{n} E_j + \sum_{j=1}^{n} T_j$ and a more general non-regular objective is the total weighted earliness plus total weighted tardiness, $\sum_{j=1}^{n} w'_j E_j + \sum_{j=1}^{n} w''_j T_j$ where the weight associated with earliness (w'_j) may differ from the weight associated with tardiness (w''_j).

2) FLEXIBLE MULTI-MACHINE JOB SHOP SCHEDULING PROBLEM WITH TARDINESS AND EARLINESS PENALTIES

In this type of job shop, operations can be executed on any available machines in the flexible job shop; however, the flexible JSSP is more complicated than the classical JSSP because it introduces more decision levels to determine the job routes to decide what a machine must process among the available options [5].

The following parameters indicate how to count the total time of each machine [5], [32], [33], [34].

Parameters:

n:	Number of jobs
m:	Number of machines
o_j:	Number of operations for job j
M_{ij}:	Set of machines capable of processing operation i of job j
p_{ijm}:	Processing time for operation i of job j on machine m.

3) OBJECTIVE FUNCTION MINIMIZING THE MAKESPAN

In equation (1), p_{ij} Refers to the processing time for an operation, i refers to the machine number, and j refers to the job number. The objective function of open shop scheduling is designed to minimize the upper and lower bound by giving a suitable schedule for ordering operations and jobs [32], [33], [34].

$$\text{minimize } T \ \textit{Total Machines Time} \ (i) = \textit{machine waiting time} + \sum_{j=1}^{n} p_{ij} \quad (1)$$

D. SCHEDULING OPTIMIZATION TECHNIQUES

Scheduling optimization techniques have been widely used in real-world problems, variety of techniques, including metaheuristic algorithms and hybrid algorithms, have been used to solve such problems [35], [36], [37], [38], [39], [40], [41], [42], [43], [44]. The following section illustrates genetic algorithm techniques used to solve scheduling problems.

1) STANDARD GENETIC ALGORITHM APPROACH

A GA is a global search technique used in the computing system to solve optimization problems. A GA can deal with challenging scheduling problems. John Holland of the University of Michigan developed the GA in 1975 based on the idea of the simulation of natural evolutions [6], [45], [46], [47].

Genetic algorithms (GAs) try to mimic evolution and improve the performance of life through the reproduction of each individual, providing their genetic data to produce offspring that are better adapted to their environment and have a higher chance of survival; This is a fundamental aspect of genetic algorithms and genetic programming. Specialized Markov Chains depict the theoretical basis of GA in terms of state transitions and search procedures [6].

Figure 1 depicts a generic cycle of evolution by natural selection in which the best individuals are continually selected and operated by mutation and crossover. After several generations, the population converges on the superior-performing solution [6], [45], [46], [47].

A GA is developed for solving job shop scheduling by trying to represent the ability of crossover operators to generate feasible schedules without affecting the performance [6], [45], [46], [47].

GA addresses a population of potential solutions. A chromosome signifies each solution. The initial step involves encoding all possible solutions into chromosomes. A set of reproduction operators have to be directly applied to the chromosomes to perform mutations and recombination over solutions; selections can compare each individual within a population using a fitness function. The fitness of the solution corresponds to the value of each chromosome. The main objective of the GA is to maximize the fitness function. However, if the objective is to minimize a cost function, the algorithm represents individuals with lower fitness functions. GA begins by generating an initial chromosome population. The initial population is typically generated randomly. Then, GA loops through an iterative procedure to find the optimum solutions. GA iterations consist of the steps outlined below.

- **INITIALISATION AND SELECTION:** The first step is the selection of the individuals; it is made randomly.
- **REPRODUCTION:** In the next step, selected individuals bred offspring in order to generate new chromosomes; the GA can use both crossover and mutation.
- **EVALUATION:** In this stage, the fitness of the new chromosomes is evaluated.
- **REPLACEMENT:** In the last step, individuals from the old population are replaced by the new ones; while the population converges toward the optimal solution, the algorithm will be stopped.

In this context, the breeding process is the main part of the genetic algorithm. The breeding process creates new and fitter individuals. The breeding process includes three steps, selecting parents, crossing the parents to create new individuals, and replacing old individuals with new ones [45], [46], [47].

a: SELECTION

Selection refers to choosing two parents for crossing from the population. After determining an encoding, the next step is to determine how to perform selection, i.e., how to select individuals from the population that will produce offspring for the next generation and how many offspring each individual will produce. The aim of selection is to involve fitter individuals with the expectation that their offspring will also be fitter. Parents for reproduction are selected from the initial population of chromosomes. The problem is selecting these chromosomes. According to Darwin's theory of evolution, only the fittest survive to reproduce.

Selection is a technique that randomly selects chromosomes from a population based on their evaluation function. The greater the fitness function, the greater the chance of selection. The selection pressure is the extent to which the superior individuals are preferred. The greater the selection pressure, the greater the preference for performance and productivity. This selection pressure motivates the GA to enhance the population's fitness over successive generations. Higher selection pressures result in greater convergence rates. GA should be able to identify optimal or nearly optimal solutions under a broad selection scheme pressure range.

Nevertheless, if the selection pressure is too low, the convergence rate will be slow, and the GA will take excessive time to find the optimal solution. If the selection pressure is too decent, there is a higher probability that the GA will prematurely converge on a suboptimal solution. In addition to providing selection pressure, selection schemes should maintain population diversity, as this helps to prevent premature convergence [45], [46], [47].

Selection needs to be balanced with mutation and crossover-induced variation. Substantial selection causes suboptimal, highly fit individuals to represent the population, reducing the diversity required for change and progress; insufficient selection will cause evolution to proceed too slowly [45], [46], [47].

Following is a discussion of the various selection methods used in this paper to generate new algorithms.

b: STOCHASTIC SELECTION

Stochastic is considered a more practical and realistic scheduling problem than the JSSP in the real world. In this work, the GA is modified when dealing with the JSP, where the fitness function can fluctuate under stochastic circumstances [6].

c: ROULETTE SELECTION

The roulette strategy selects the optimum solutions regarding the expected value where each individual has many frequencies during selection operations. The roulette wheel is

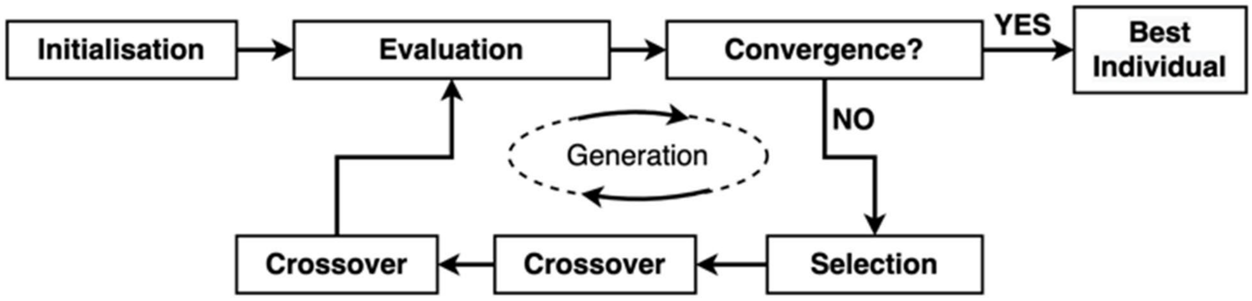


FIGURE 1. Genetic algorithm: the sequence of operators and evaluation of each individual [45].

segmented, and the individuals with the highest fitness are given more extensive segments for a higher probability of being selected [6].

One of the traditional GA selection methods is roulette selection. In the proportionate reproductive operator, a string is selected from the mating pool with a probability proportional to the fitness. The concept of roulette selection is a linear search through a roulette wheel where the slots are weighted according to the individual's fitness values. A target value, a random proportion of the sum of the population's finesses, is determined. The population is iterated until the desired value is reached; This is a moderately effective method of selection, as it is not guaranteed that fit individuals will be chosen, but they are more likely to be selected. A fit individual will contribute more to the target value, but if it doesn't surpass it, the next chromosome in line has a chance, which may be weak. It is important that the population is not sorted by fitness, as this would significantly bias the selection process. In the roulette wheel selection, the expected value of an individual is the individual's fitness divided by the actual population fitness. Each individual is assigned a portion of the roulette wheel proportional to the fitness level. The wheel is spun N times, where N is the total number of selected populations. Each time the wheel is spun, the individual under the marker is chosen to be the next generation's parents. This technique is executed as follows [6], [45]:

(1) Sum the total estimated value of the population's individuals. Let it be T.
(2) Iterate N times: Select an integer 'r' randomly between o and T.
(3) Sum the expected values for each individual in the population till the sum is greater than or equal to 'r'. The individual selected is the one the expected value exceeds this limit.

d: SEXUAL SELECTION

Sexual selection GA (SGA) is an improved version of the GA. In classic GA, the selection process is based on choosing parent chromosomes for reproduction. In the SGA, the idea of male effort inspired it, and female choice is based on the algorithm based on separating the population into males and females. The selection procedure is based on each female; A number of males compete to be selected for reproduction. The remaining steps are similar to those for classic GA [48], [49].

e: AGING SELECTION

Ageing GA is a modified version of a traditional GA in which the age of individuals affects their performance. When a new individual is generated, its age is considered zero. Therefore, with every iteration of age increase in individuals, young individuals are considered less fit than adult individuals; The effectiveness of individuals is measured by considering both the objective function value and their ages [50], [51].

f: CROSSOVER (RECOMBINATION)

Crossover is the process of combining two parent solutions to produce offspring. Following the process of selection (reproduction), the population is enriched with superior individuals. Reproduction duplicates excellent strings but does not generate new ones. The mating pool is treated with a crossover operator in the expectation that it will produce superior offspring. There are different types of crossovers, including single-point crossover, two-point crossover, multi-point crossover, uniform crossover, and so on.

The crossover probability is the fundamental parameter in the crossover study (Pc). Crossover probability is a parameter that describes the frequency of crossover; if there is no genetic crossover, offspring are identical to their parents; if there is chromosome crossover, the offspring contain portions of both parents' chromosomes. If the probability of crossing is 100%, all offspring are produced through crossing; if it is zero percent, the entire new generation is created from exact copies of chromosomes from the old population. Crossover is performed hoping that new chromosomes will contain beneficial portions of old chromosomes and therefore be superior. However, allowing a portion of the aging population to survive in the next generation is beneficial.

The crossover operator is adjusted using the formula [45], [46], [47].

$$P_c = \begin{cases} \dfrac{(p_{c1} - p_{c2})\ (f' - f_{avg})}{\mathrm{f}_{\max} - f_{avg}} & f' \geq f_{avg} \\ p_{c1} & f' < f_{avg} \end{cases} \tag{2}$$

- f_{max} refers to the highest fitness value in the population;
- f_{avg} is the average fitness value in each population;

- f' refers to a higher fitness value between two individuals:

$$Pc1 = 0.9, \quad Pc2 = 0.6$$

Instead of using fixed pm, it is adjusted based on the following formula: the mutation operator [45].

$$p_m = \begin{cases} p_{m1} = \dfrac{(p_{m1} - p_{m2})(f - f_{avg})}{f\_\max - f_{avg}} & f \geq f_{avg} \\ p_{m1} & f < f_{avg} \end{cases} \quad (3)$$

- f_{max} refers to the highest fitness value in the population;
- f_{avg} is the average fitness value in each population;
- f' refers to a higher mutation value between two individuals [45].

g: MUTATION

Following crossover, the strings undertake the mutation preventing an algorithm from becoming trapped at a local minimum. Mutation serves the dual purpose of recovering lost genetic material and randomly altering genetic information. Mutation has traditionally been regarded as a straightforward search operator. The mutation explores the entire search space, whereas crossover is meant to exploit the current solution to find better alternatives. The mutation is viewed as a background process to maintain genetic diversity in a population. It introduces new genetic structures into the population by randomly altering some constituents. Mutation aids in escaping the trap of local minima and maintains population diversity. A search space is ergodic if there is a probability greater than zero of producing any solution from any population state [38], [39], [40], [41], [42], [43], [44], [45], [46], [47].

2) GENETIC ALGORITHM FOR JOB SHOP SCHEDULING PROBLEMS (JSSP)

Scheduling, particularly job shop scheduling, has been studied for a substantial period. Some meta-heuristics, such as Simulated Annealing, Taboo Search, and Genetic Algorithms, have been implemented as pure methods and hybrids of different methods due to the NP-Hard nature of the problem, with hybrid methods being superior to pure methods. The primary issue is how to deal with local minima in a timely manner. GA has been studied and successfully implemented alongside the other problems [45], [46], [47].

The JSSP comprises several machines, denoted by M, and some jobs, marked by J. Each job entails M tasks, each with a predetermined duration. Each task must be performed on a single machine, and each job must only visit each machine once. There is a predetermined order to the functions that comprise a job. A machine can only perform a single task at a time. There are no configuration times, release dates, or due dates. The makespan is the time between the start of the first task and the completion of the last task. The objective is to find start times for each task that minimize the makespan [45].

E. GENETIC ALGORITHM IMPLEMENTATION USING MATLAB

MathWorks's MATLAB (Matrix Laboratory) is a scientific software package designed to provide numerical computation and graphics visualization in an advanced programming language. Dr Moler, Chief Scientist at MathWorks, Inc., developed MATLAB to facilitate access to matrix software created for the LINPACK and EIPACK projects. The initial version was written in the late 1970s for matrix theory, linear algebra, and numerical analysis courses. Therefore, MATLAB is built on a foundation of advanced matrix software, in which the fundamental data element is a one-dimensional matrix [45].

MATLAB offers a vast array of useful functions for genetic algorithm practitioners as well as those desperately hoping to experiment with the algorithm for the first time. Given the versatility of MATLAB's high-level language, problems can be coded in m-files in a fraction of the time it would take to make C or Fortran programs for the same purpose. When combined with MATLAB's advanced data analysis, visualization, and application domain toolboxes, the user is provided with a uniform environment to investigate the potential of GAs.

The GA Toolbox is a set of flexible tools for implementing various genetic algorithm methods. It contains a collection of procedures, written primarily on m-files, which apply the essential functions in genetic algorithms. In this context, due to the low convergence speed of the standard GA, an improved version of GA is established as a novel metaheuristic hybrid parthenogenetic algorithm (NMHPGA) code using MATLAB.

1) DATA STRUCTURES

The only data type supported by MATLAB is a rectangular matrix of real or complex numeric elements. The primary data structures contained within the Genetic Algorithm toolbox are (1) chromosomes, (2) objective function values, and (3) fitness values. The following subsections discuss these data structures [45].

2) CHROMOSOMES

The chromosome data structure stores the whole population in a single matrix of size N_{ind} by L_{ind}, where N_{ind} represents the number of individuals in the population, and Lind represents the length of the genotypic representation of those individuals. Each row represents an individual's genotype, consisting of base-n, ordinarily binary, values *Chrom* (4), as shown at the bottom of the next page [45].

This data representation does not impose a structure on the chromosome structure; Everything that is required is that all chromosomes have equal length. Consequently, structured populations or populations with varied genotypic bases can be utilized with the Genetic Algorithm Toolbox if a proper decoding function, mapping chromosomes to phenotypes, is implemented [45].

3) PHENOTYPES

The decision variables, or phenotypes, are obtained in GA by mapping the chromosome representation into the variable decision space. Each string in the chromosomal structure is decoded to a row vector of order N_{var}, according to the number of dimensions in the search space and the value of the decision variable vector. The decision variables are kept in a matrix with the dimensions N_{ind} by N_{var}. Again, each row corresponds to the phenotype of a specific individual [45].

4) OBJECTIVE FUNCTION VALUES

The performance of phenotypes within the problem domain is evaluated using an objective function. Objective function values may be scalar or vectorial in multi-objective problems. Note that objective function values and fitness values are not necessarily the same. The objective function values are stored in a $N_{ind} \times N_{obj}$ matrix, where N_{obj} is the number of objectives. Each row corresponds to the objective vector of an individual [45].

$$Objective\ Function = \begin{bmatrix} y_{1,1} & y_{1,2} & y_{1,3} & \cdots & y_{1,Nvar} \\ y_{2,1} & y_{2,2} & y_{2,3} & \cdots & y_{2,Nvar} \\ y_{3,1} & y_{3,2} & y_{3,3} & \cdots & y_{3,Nvar} \\ . & . & . & \cdots & . \\ y_{Nind,1} & y_{Nind,2} & y_{Nind,3} & \cdots & y_{Nind,Nvar} \end{bmatrix} \tag{5}$$

5) FITNESS VALUES

Objective function values are converted into fitness values using a scaling or ranking function. Finesses are nonnegative scalars stored in length-column vectors N_{ind} An example of this subject is shown below, including ranking, an arbitrary fitness function [45].

$$Fitn = ranking\ (ObjV)\ \%\ fitness\ function$$

$$\text{Fitn} = \begin{cases} f_1 & individual\ 1 \\ f_2 & individual\ 2 \\ f_3 & individual\ 3 \\ \cdots & \\ f_{Nind} & individual\ Nind \end{cases}$$

F. FRAMEWORK OF THE PARTHENOGENETIC ALGORITHM

GA is based on the concept of evolution; survival of the fittest has been extensively used to solve NP-hard problems. In GAs, the candidate solutions are indicated as a population of chromosomes (individuals) consisting of a string of genes.

The crossover operator is the primary genetic operator to generate new offspring by mixing two parents. Unique individuals can inherit some features from their parents. There are different types of traditional crossover operators: one-point crossover, two-point crossover, scattered crossover, etc. [45].

Parthenogenetic algorithm (PGA) is a variant of GA that employs gene recombination and selection instead of the traditional crossover operator to produce offspring. PGA deals with the above issue by removing the crossover operator, improving the genetic algorithm's effectiveness and performance; this is due to the shift operator, which is only performed on a single chromosome, preventing the offspring from the crossover operator from jumping to the invalid solutions area. There are three partheno-genetic operators: swap, reverse, and insert. These three operators change the order of genes in a chromosome to generate a new chromosome [52], [53], [54], [55].

III. ETHNIC SELECTION GENETIC ALGORITHM

Ethnic GA (EGA) is based on combining the different populations generated using various selection methods [56]. Some ethnic groups allow heterosexual partners (SGA), some others prefer middle-aged people (AGA), others do not interfere with partner selection (stochastic GA), and lastly, some prefer string and wealthy partners; these techniques affect the speed of convergence and the global solution.

In this paper, an ethnic selection GA combines four types of selections, including stochastic, aging, sexual, and roulette selections, to test the convergence speed; Moreover, the ethnic selection is combined with a parthenogenetic algorithm in order to propose a novel metaheuristic hybrid parthenogenetic algorithm (NMHPGA) to test and compare results with the standard parthenogenetic algorithms.

IV. NOVEL METAHEURISTIC HYBRID PARTHENOGENETIC ALGORITHM

A novel metaheuristic hybrid parthenogenetic algorithm is developed by combining a variant of already existing selections with the parthenogenetic algorithm. Different optimization selections mixed with the parthenogenetic algorithm (PGA) are established to provide a valid comparative study and evaluate the overall performance of the novel metaheuristic hybrid parthenogenetic algorithm. The metaheuristics algorithms for this purpose are Ageing PGA, Sexual selection PGA, Roulette selection PGA, stochastic selection PGA, and ethnic selection PGA, tested on job shops from industry and mathematical benchmark functions to test the accuracy of the algorithm. NMHPGA consists of ethnic selection mixing with 3 parthenogenetic operators, namely swap, reverse and

$$Chrom = \begin{array}{c} \left[\begin{array}{ccccc} g_{1,1} & g_{1,2} & g_{1,3} & \cdots & g_{1,Lind} \\ g_{2,1} & g_{2,2} & g_{2,3} & \cdots & g_{2,Lind} \\ g_{3,1} & g_{3,2} & g_{3,3} & \cdots & g_{3,Lind} \\ . & . & . & \cdots & . \\ g_{Nind,1} & g_{Nind,2} & g_{Nind,3} & \cdots & g_{Nind,Lind} \end{array}\right] \end{array} \begin{array}{l} individual\ 1 \\ individual\ 2 \\ individual\ 3 \\ . \\ individual\ Nind \end{array} \tag{4}$$

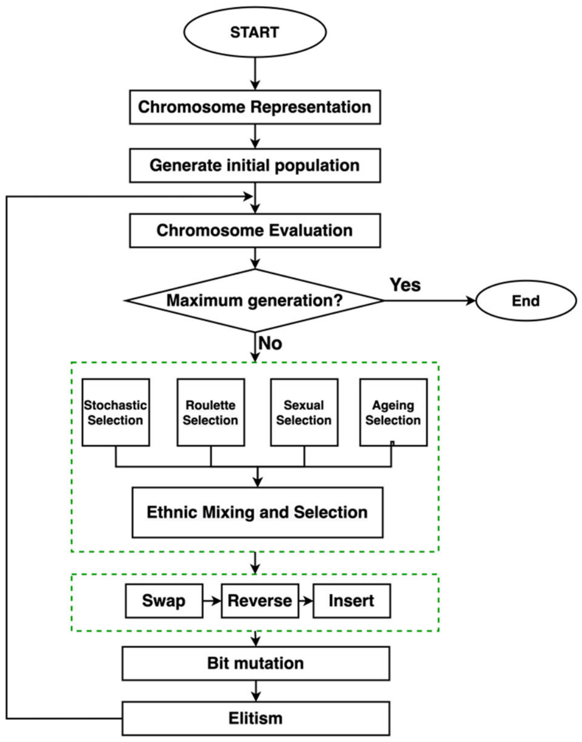


FIGURE 2. NMHPGA algorithm flowchart.

insert, the algorithm's initial population is produced, and then stages of the algorithm start with removing the cross-over operators. In order to test the accuracy of NMHPGA, five different algorithms consisting of mutation only and replacing cross-over operators with swap reverse and insert functions are tested; however, the algorithms differ in selection types. For the stochastic selection parthenogenetic algorithm (STPGA), the selection type is stochastic selection based on random selection. Moreover, the roulette parthenogenetic algorithm is based on roulette wheel selection (RPGA); moving forward to the sexual parthenogenetic algorithm(SPGA), the selection is sexual selection; in comparison, APGA wish is aging parthenogenetic algorithm is based on aging selection; lastly, the novel metaheuristic hybrid parthenogenetic algorithm is based on the ethnic selection which is the combination of stochastic selection, roulette selection, sexual selection, and the aging selection and finding best fitness function from the combination of the selections. The procedure of the NMHPGA algorithm is illustrated as follows:

The first testing stage is based on using standard selection procedures. In contrast, the second stage combines the best individuals selected using different methods into a single population, known as ethnic selection. The NMHPGA does not utilize the crossover function, which is very time-consuming due to the checks to avoid replicated genes in the chromosomes. Figure 2 illustrates flowchart of NMHPGA.

The most recent and improved selection versions are chosen to increase the algorithm's accuracy. Moreover, the algorithm's internal parameters are the most critical in their convergence speed. As a result, the parameters are the most successful configurations based on the literature.

V. MATHEMATICAL BENCHMARK TEST FUNCTIONS

Before testing and resolving an optimization problem, it is essential to identify the functional characteristics that can make the optimization process difficult. In applied mathematics, benchmark functions, also known as artificial landscapes, are primarily employed to evaluate optimization methods' precision, convergence rate, and robustness. In order to develop a new optimization algorithm, it is essential to use benchmark functions to test how the new algorithm performs compared to other algorithms. In this research, various unimodal and multimodal benchmark functions are used to demonstrate the efficacy of the established algorithm. Numerous tests or benchmark functions are indicated in the literature, but no standard set of benchmark functions exists. Test functions should ideally have diverse characteristics to be useful for testing new algorithms. Each metaheuristic method that effectively calculates the optimal points of such procedures makes solving optimization problems more efficient [57].

The benchmark functions used for testing the metaheuristic NMHPGA include Rastrigin, Ackley, Sphere, Rosenbrock, Levy, Griewank, Sum square, Sum of different powers, Rotated Hyper-Ellipsoid and Zakharov function [58], [59], [60]. Figure 3 and Table 1 depict these functions' presentation in more detail. Each function has a unique equation, and their three-dimensional diagrams show the difficulty of locating optimal positions. As demonstrated, Rastrigin, Ackley, and Griewank functions are more complex than the Sphere function due to the presence of both local and global optimal points. Each of the benchmark functions has a global optimal (minimum) at $x = 0, \ y = 0$, with $f(0, 0) = 0$ at this optimal point] [61]. These functions are chosen regarding features like modality, basins, and dimensionality. All test functions are inseparable from increasing the difficulty of optimization. Any technique that reduces the error of looking for optimal spots has a larger capacity to effectively handle optimization problems so that NMHPGA is tested on these benchmark functions.

A. NUMERICAL RESULTS FOR THE BENCHMARK FUNCTIONS

In the following section, numerical results of running the established algorithms on benchmark functions are illustrated; based on the results, the novel algorithm has satisfying results of objective functions. Table 2 presents the 2D Objective function global minima results for different algorithms tested on benchmark functions. In Table 3, the convergence speed comparison on benchmark functions is shown, as the NMHPGA improves the convergence speed, which means it improves the speed. It shows how fast it reaches the best solution and refers to how many generations take to get it.

Table 4 similarly shows the results of testing NMHPGA using the benchmark functions for two dimensions, ten dimensions, and 50 dimensions, and in all three categories results are satisfying. Table 4 illustrate that the objective func-

TABLE 1. Mathematical benchmark functions equations [59].

Name	Equation	Min
Rastrigin - F1	$f(x) = 10d + \sum_{i=1}^{d}(x_i^2 - 10\cos(2\pi x_i))$	0
ACKLEY-F2	$f(x) = -A\exp\left(-B\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(Cx_i)\right) + A + \exp(1)$	0
Sphere-F3	$f(x) = \sum_{i=1}^{d}x_i^2$	0
ROSENBROCK-F4	$f(x) = \sum_{i=1}^{d-1}(100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2)$	0
Levy-F5	$f(x) = sin^2(\pi\omega_1) + \sum_{i=1}^{d-1}(\omega_i - 1)^2\,[1 + 10\,sin^2(\pi\omega_i + 1)] + (\omega_d - 1)^2[1 + sin^2(2\pi\omega_d)]\,, where$ $\omega_i = 1 + \frac{x_i - 1}{4}, for\ all\ i = 1, \dots, d$	0
Griewank-F6	$f(x_1, x_2, \dots, x_n) = \frac{1}{4000}\left(\sum_{i=1}^{n}x_i^2\right) - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	0
Zakharov-F7	$f(x) = \sum_{i=1}^{d}x_i^2 + \left(\sum_{i=1}^{d}0.5ix_i\right)^2 + \left(\sum_{i=1}^{d}0.5ix_i\right)^4$	0
Sum square-F8	$f(x) = \sum_{i=1}^{d}ix_i^2$	0
Sum of different powers-F9	$\sum_{i=1}^{d}\lvert x_i\rvert^{i+1}$	0
Rotated hyper-ellipsoid function-F10	$f(x) = \sum_{i=1}^{d}\left(\sum_{j=1}^{i}x_j^2\right)$	0

TABLE 2. 2D Objective function global minima for different algorithms tested on benchmark functions.

Benchmark-Type	STPGA	RPGA	SPGA	APGA	NMHPGA
Rastrigin	0	1.98E-6	1.947E-6	0	0
Ackley	4E-10	4E-8	6.25E-10	1.434E-6	0
Sphere	2E-8	1.22E-6	1.22E-6	1.22E-6	1.22E-6
Rosenbrock	0	1E-06	1e-06	1E-6	1E-6
Levy	6.2E-10	1.434E-6	6.25E-10	1.434E-6	1.434E-6
Griewank	7.50E-9	3.078E-7	3.078E-7	3.078E-7	3.078E-7
Sum square	3E-8	1.23E-6	1.23E-6	1.23E-6	1.23E-6
Sum of different powers	1.33E-9	1.133-08	1.133E-8	1.133E-8	1.133E-8
Rotated Hyper-Ellipsoid	2E-3	1.23E-5	1.22E-6	1.22E-7	1.14E-8
Zakharov	4.25E-8	1.642E-6	1.642E-6	4.25E-8	1.6425E-6

TABLE 3. Comparison of convergence speed of different algorithms tested on benchmark functions.

Benchmark-Type	STPGA	RPGA	SPGA	APGA	NMHPGA
Rastrigin	6	5	2	3	3
Ackley	8	6	6	3	2
Sphere	4	3	3	2	2
Rosenbrock	7	2	2	2	2
Levy	3	7	5	2	2
Griewank	4	3	2	3	2
Sum square	2	3	2	2	2
Sum of different powers	2	3	2	6	2
Rotated Hyper-Ellipsoid	4	3	2	3	2
Zakharov	6	5	2	16	2

tion of NMHPGA is based on the objective function of each benchmark function in order to measure the performance of the algorithm. The objective function of the NMHPGA is close to the defined objective function of each benchmark function equation, demonstrating the algorithm's good performance.

VI. STATISTICAL ANALYSIS OF THE NOVEL METAHEURISTIC HYBRID PARTHENOGENETIC ALGORITHM

Different algorithms' objective function values and convergence speeds are calculated and utilized for statistical analysis. To this end, the comparison of the convergence speed of different algorithms tested on other job shops is shown in the following section.

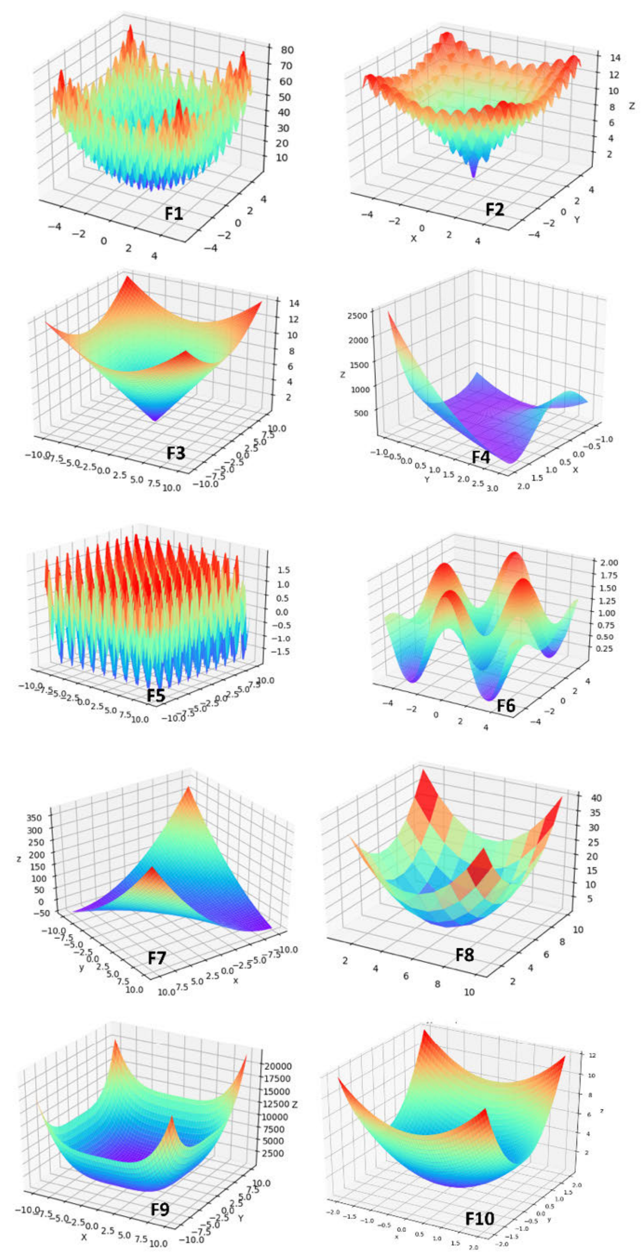


FIGURE 3. Mathematical benchmark functions.

TABLE 4. Objective functions and convergence speed of NMHPGA tested on benchmark functions for 2D,10D, and 50D.

Benchmark-Type	2D		10D		50D	
	Objective function	Conv speed	Objective function	Conv spee d	Objectiv e function	Conv spee d
Rastrigin	0	3	5.078E-2	24	7.34E-5	440
Ackley	0	2	2.321E-4	12	6.24E-4	73
Sphere	1.220E-6	2	2.561E-4	11	3.70E-7	62
Rosenbrock	1.000E-6	2	0	3	0	15
Levy	1.433E-6	2	6.327E-6	18	7.95E-2	66
Griewank	3.078E-7	2	0.16355	157	0.55349	78
Sum square	1.230E-6	2	0.0055565	8	0.00088	253
Sum of different powers	1.133E-8	2	1.8212E-9	7	1.0E-10	11
Rotated Hyper-Ellipsoid	1.140E-8	2	3.561E-4	10	3.80E-6	72
Zakharov	1.642E-6	2	1.937E-2	6	2.97E-1	32

TABLE 5. Single machine job shops attribute (category-A).

JS-Type	Number of machines	Number of jobs
SM1	1	32
SM2	1	40
SM3	1	60
SM4	1	80
SM5	1	100
SM6	1	120
SM7	1	150
SM8	1	200
SM9	1	250
SM10	1	300

TABLE 6. Comparison of objective functions of different algorithms tested on job shops in single machines category-A.

JS-Type	STPGA	RPGA	SPGA	APGA	NMHPGA
SM1	41251	39982	39947	40225	39444
SM2	159697	181734	187035	159802	160871
SM3	479639	599840	584763	471184	474325
SM4	1154879	1280561	1301604	1118500	1120440
SM5	2405190	2566361	2609437	2299699	2297979
SM6	14135674	11540670	11591499	11849996	11009721
SM7	9342492	10172814	9832276	8546759	8549596
SM8	19156403	19650883	19532952	15980532	15952550
SM9	39430578	40136718	40668916	31142579	30845566
SM10	74190404	61548166	60259784	52838449	52260543

A. JOB SHOP OPTIMIZATION RESULTS

1) JOB SHOP SCHEDULES

In this paper, three categories of simple benchmarks are tested; the benchmarks focus on only two elements, the number of jobs and arrival pattern, to make the job shops as simple as possible to focus on optimization results and compare the effectiveness of results. The job shops are generated for a simple production line. The schedules are generated randomly using a constrained open-shop algorithm. The first category is category-A (SM), consisting of 10 single-machine job shops with earliness, tardiness, and due date (Table 5). Moving forward to the next category, the second category is category-B consisting of 10 multi-machine job shops(MM) with 4 machines, eight jobs with earliness, tardiness, and due date, and the last category is category-C consisting of 9 multi-machine job shops with earliness, tardiness, and due date (Table 8). Table 5 and Table 8 illustrate different job shop types used in this paper. Besides, the initial random selection state is equally selected to form a comparative model.

The scheduling problem is based on finding the best scheduling time with the objective function of minimizing the execution time and penalties. The is set to a population size of

TABLE 7. Comparison of convergence speed of different algorithms tested on job shops in single machines category-A.

JS-Type	STPGA	RPGA	SPGA	APGA	NMHPGA
SM1	859	113	113	800	500
SM2	311	53	143	210	747
SM3	894	113	150	203	142
SM4	906	145	362	143	178
SM5	899	173	159	272	271
SM6	917	740	943	975	581
SM7	984	475	259	396	338
SM8	991	382	302	604	394
SM9	993	728	518	647	578
SM10	988	636	353	747	606

TABLE 8. Category-B Multimachine Job shops 4 Machines,8 jobs (MM1-MM10) and Category-C multi-machine job shops (MM11-MM19) attributes.

JS-Type	Number of machines	Number of jobs
MM1	4	8
MM2	4	8
MM3	4	8
MM4	4	8
MM5	4	8
MM6	4	8
MM7	4	8
MM8	4	8
MM9	4	8
MM10	4	8
MM11	4	10
MM12	4	20
MM13	4	30
MM14	4	40
MM15	4	100
MM16	4	150
MM17	4	200
MM18	4	250
MM19	4	300

300 and a generation of 1000. Testing aims to investigate the NMHPGA performance with simple mutation and advanced regeneration and the effect of the selected types of roulette selection, sexual selection, aging selection, and ethnic selection. In this research, MATLAB R2021a has been used.

2) JOB SHOP SCHEDULING AND RELIABILITY TEST RESULTS

Simulation results are shown in Figures 6–15 for the single-machine and multi-machines in Figures 16–34, respectively. Besides, three simulation results for SM3, MM3, and MM11 are shown in Figure 4, selected randomly to be illustrated in the paper. Each figure indicates the objective function regarding the generation number. The objective function represents the time taken to finish the job shop schedule.

Table 6 illustrates the Comparison of objective functions of different algorithms tested on job shops in single machines category A. It is illustrated that NMHPGA decreases the objective function in most cases compared to other algorithms, which shows the better performance of the NMHPGA compared to other algorithms tested in this research. Besides, Table 7 represents a comparison of the convergence speed of different algorithms tested on job shops in single machines category A. As illustrated, the objective function for NMHPGA has the lowest value with a higher convergence speed; This is concluded from the number of generations shown to

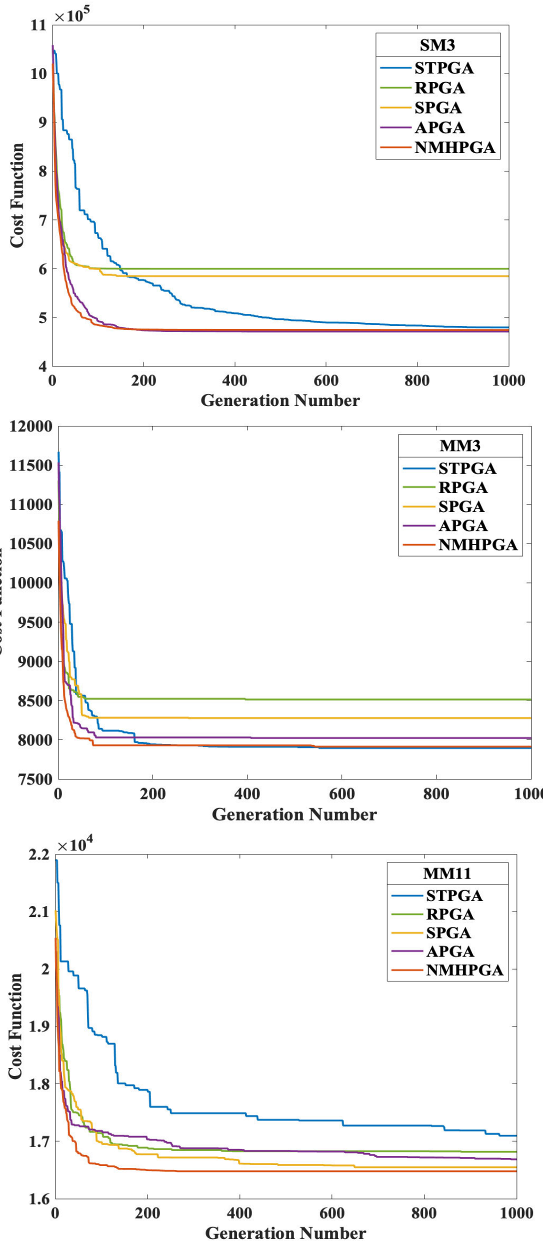


FIGURE 4. Convergence speed results of various PGAs, SM3, MM3, and MM11, were chosen randomly; the rest of the results are shown in Figures 6–34.

reach the best solution; the number of generations is lower for NMHPGA, which means it improves the convergence speed.

Moving forward to the next section, Table 9 illustrates a Comparison of objective functions of different algorithms tested on job shops in multi-machines category B. Moreover, Table 10 indicates a Comparison of the convergence speed of different algorithms tested on job shops in multi-machines category B; Similar to category A results, NMHPGA reduces the objective functions with higher speed. Simulation results of category C are illustrated in Tables 11 and 12. In this cat-

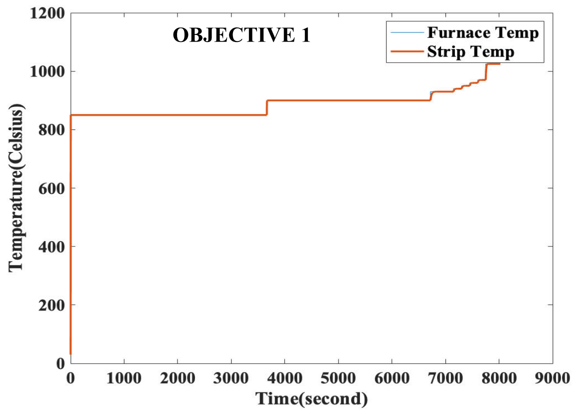
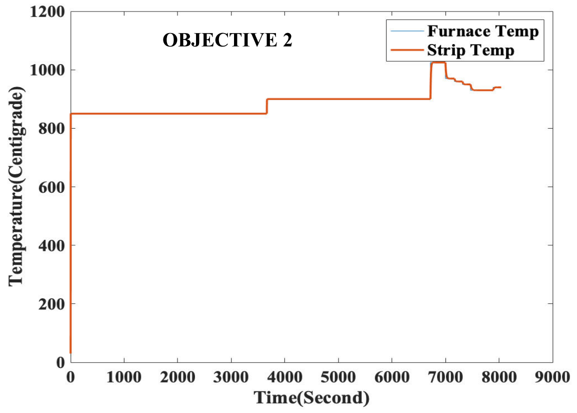
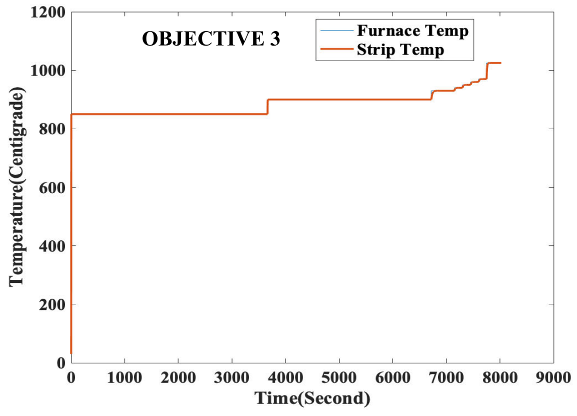




FIGURE 5. The furnace model function for three objectives.

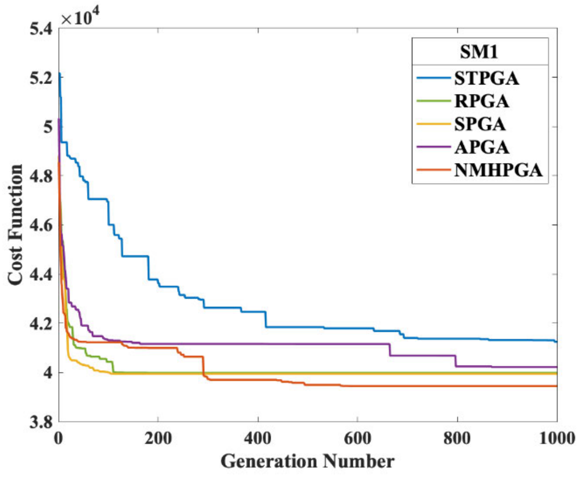


FIGURE 6. Single machine scheduling using the SM1 model.

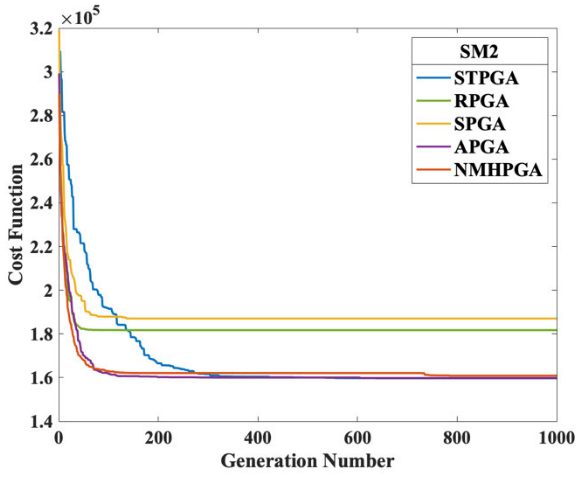


FIGURE 7. Single machine scheduling using the SM2 model.

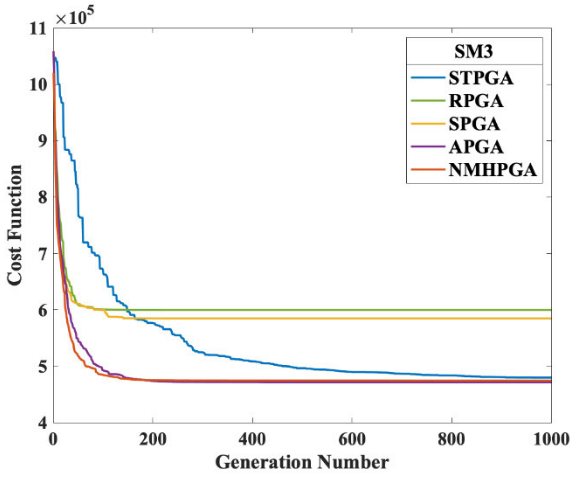


FIGURE 8. Single machine scheduling using the SM3 model.

egory, the objective functions of NMHPGA decrease and are lower than those of other algorithms with faster convergence.

The convergence speed comparison using the benchmark functions is shown in Table 3, NMHPGA improves the convergence speed, which means it improves the speed of reaching the best solution. Table 4 similarly shows the results of testing NMHPGA on benchmark functions for 2, 10, and 50 dimensions. In all three categories, the results are satisfying because the results are close to the objective functions of zero.

In General, NMHPGA objective function results are better and lower, which means the parthenogenetic algorithm with a combination of ethnic selection can lead to better results in terms of the cost function. The developed algorithm selects the best solutions from the various selection algorithms, giving comprehensive and diverse solutions (from the different

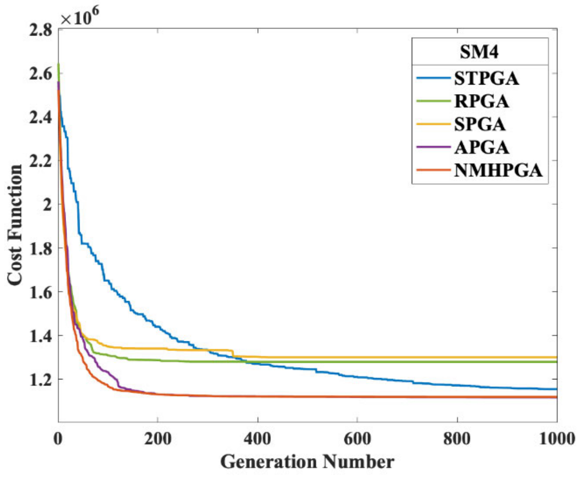


FIGURE 9. Single machine scheduling using the SM4 model.

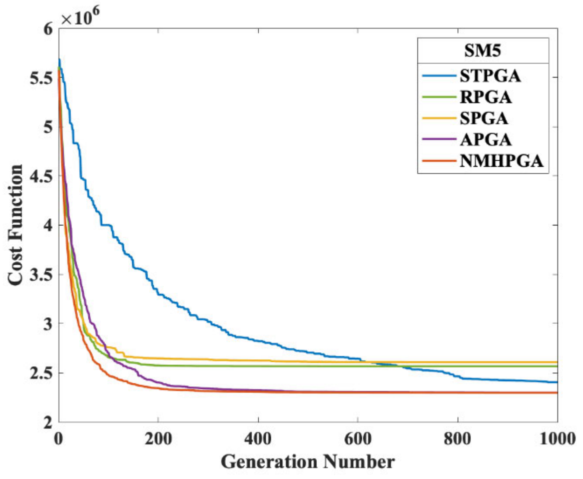


FIGURE 10. Single machine scheduling using the SM5 model.

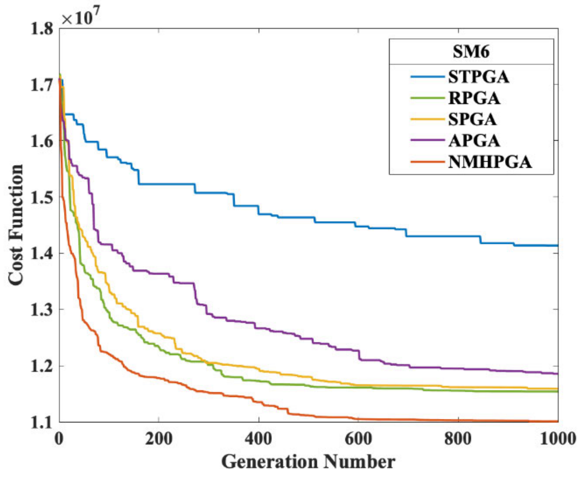


FIGURE 11. Single machine scheduling using the SM6 model.

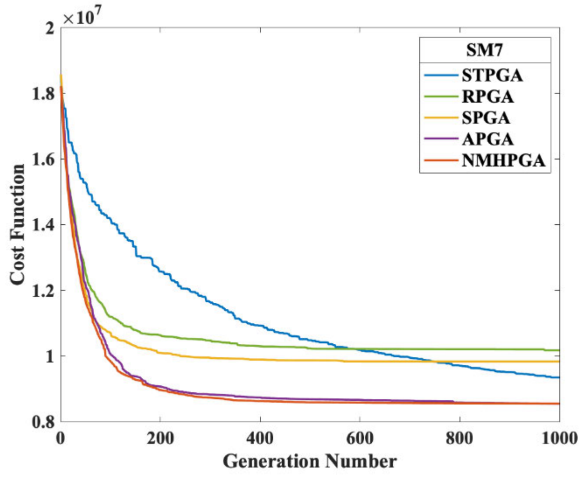


FIGURE 12. Single machine scheduling using the SM7 model.

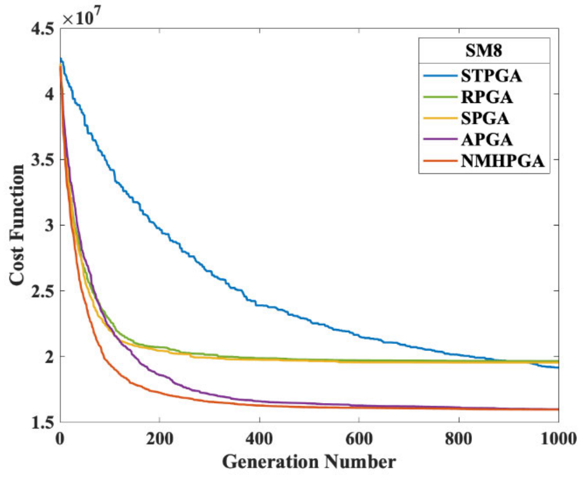


FIGURE 13. Single machine scheduling using the SM8 model.

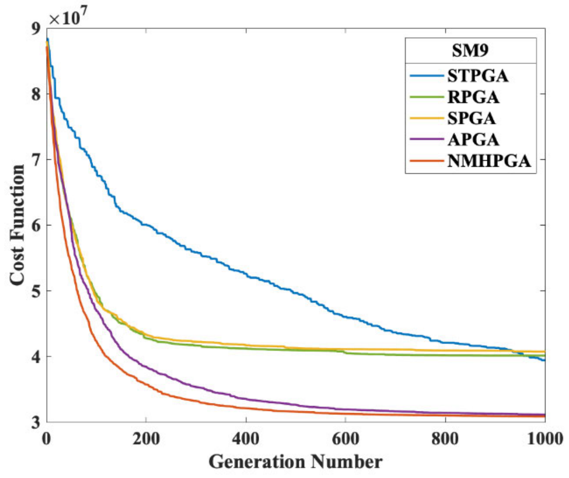


FIGURE 14. Single machine scheduling using the SM9 model.

selections) to be merged to generate new solutions by missing chromosomes from different search areas. On the other hand, the stochastic and roulette selection methods show slower convergence and do not reach the global minima due to the random nature of the solution generation constrained to a specific search area of the solution space.

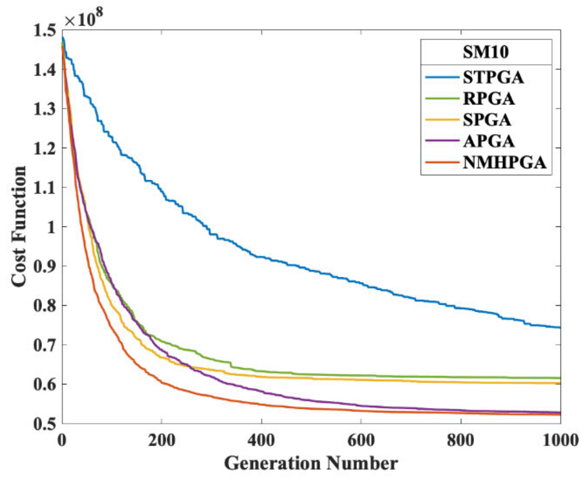


FIGURE 15. Single machine scheduling using the SM10 model.

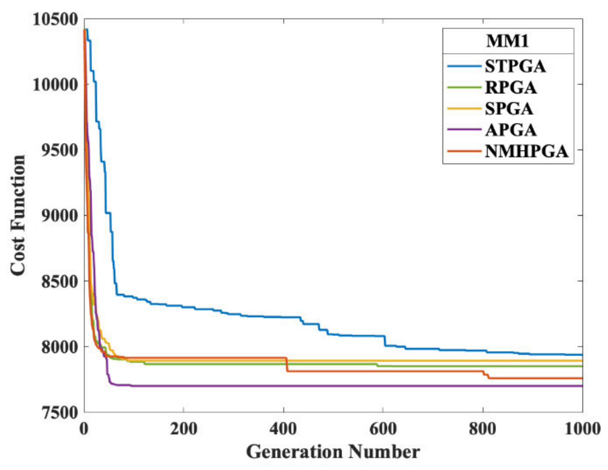


FIGURE 16. Multi-machine scheduling using MM1 model.

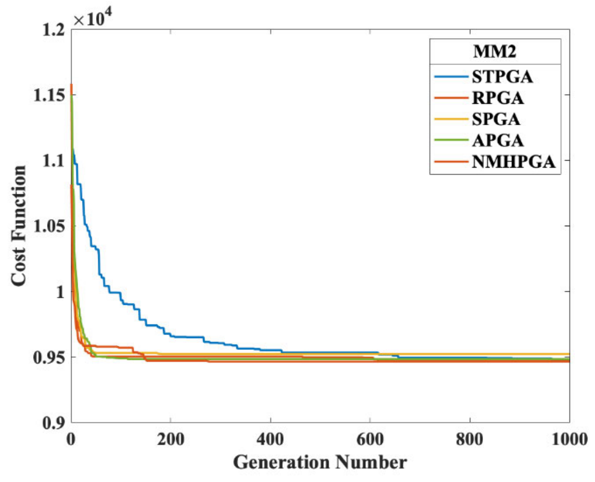


FIGURE 17. Multi-machine scheduling using the MM2 model.

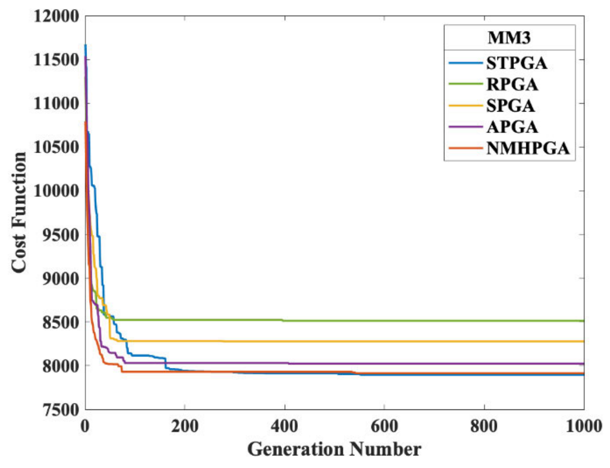


FIGURE 18. Multi-machine scheduling using the MM3 model.

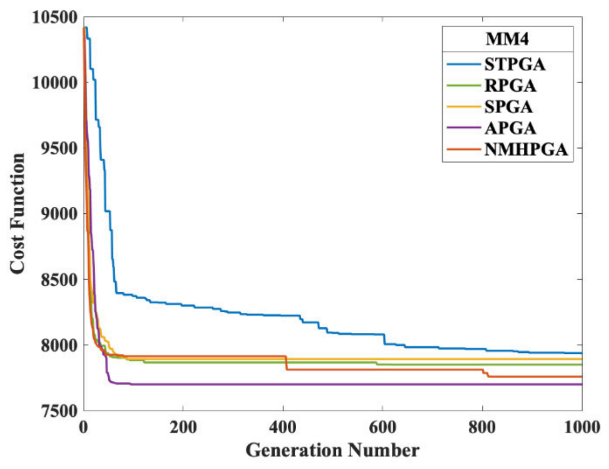


FIGURE 19. Multi-machine scheduling using the MM4 model.

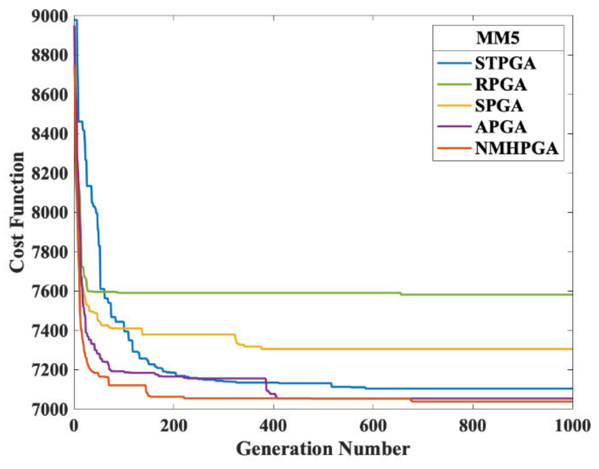


FIGURE 20. Multi-machine scheduling using the MM5 model.

VII. CASE STUDY: FURNACE MODEL

Furnace designs vary in purpose, heating operations, fuel type, and method of introducing combustion air, however, most process furnaces share specific characteristics. This paper optimizes the scheduling of a furnace model created by Yoshitani and Hasegawa [62]. The quality of a furnace's design is determined by the fuel type, combustion efficiency,

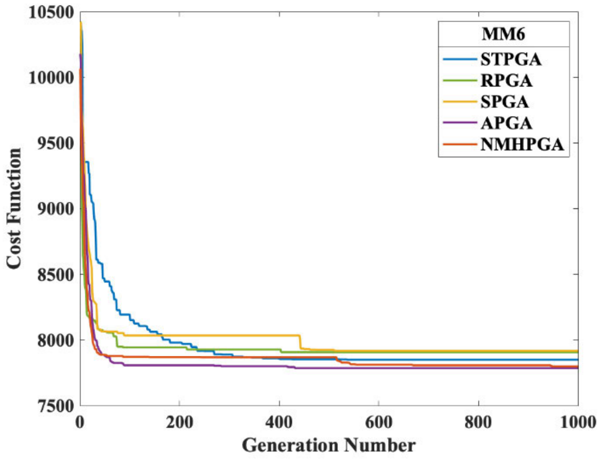


FIGURE 21. Multi-machine scheduling using the MM6 model.

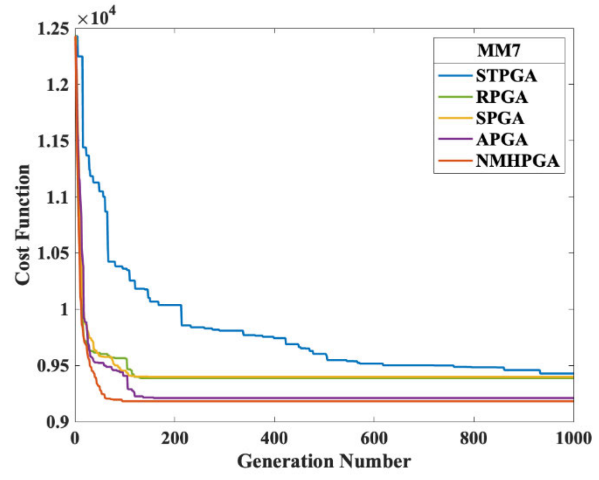


FIGURE 22. Multi-machine scheduling using the MM7 model.

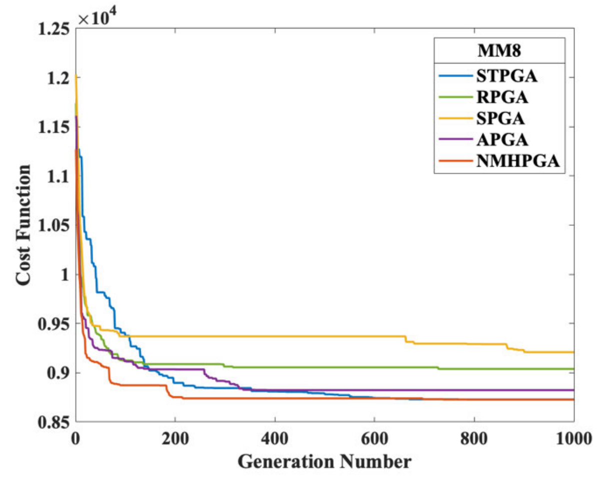


FIGURE 23. Multi-machine scheduling using the MM8 model.

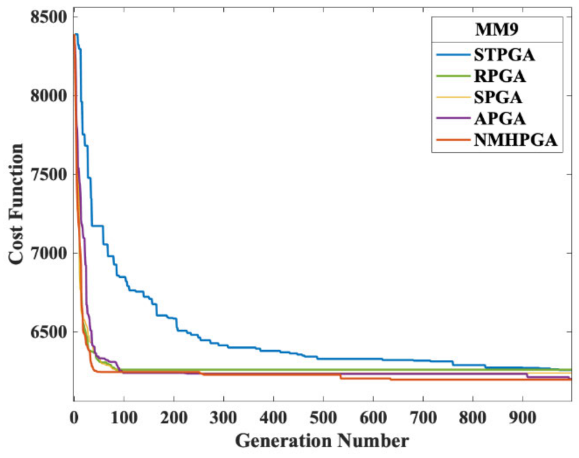


FIGURE 24. Multi-machine scheduling using the MM9 model.

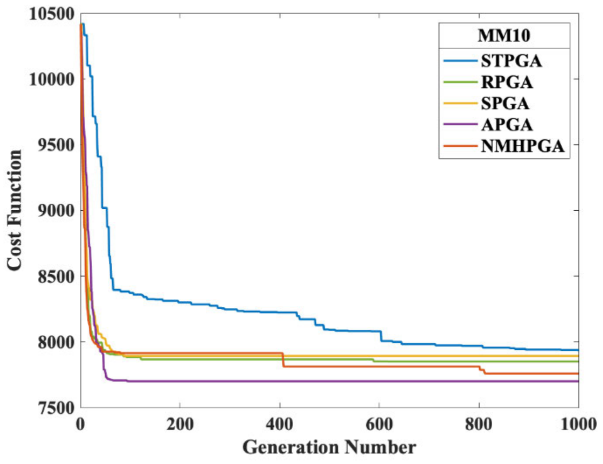


FIGURE 25. Multi-machine scheduling using the MM10 model.

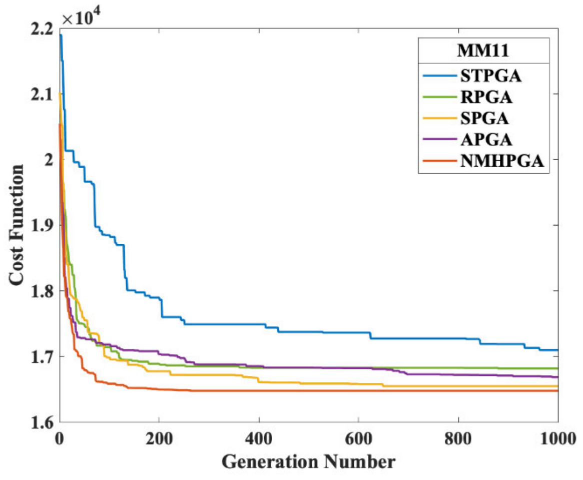


FIGURE 26. Multi-machine scheduling using the MM11 model.

standby and cycling losses, and heat transfer [62]. The heating schedule for the materials is optimized to reduce energy consumption and time. Five Algorithms are used to optimize the schedule of the furnace model. However, NMHPGA achieves superior results in reducing the furnace model's time and energy consumption. For this purpose, industrial data is utilized.

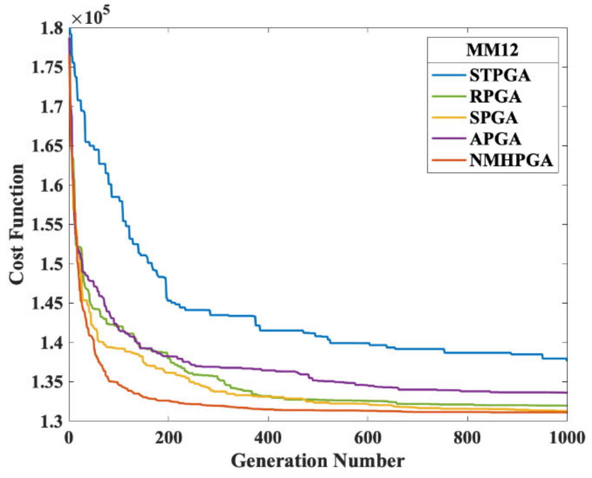


FIGURE 27. Multi-machine scheduling using the MM12 model.

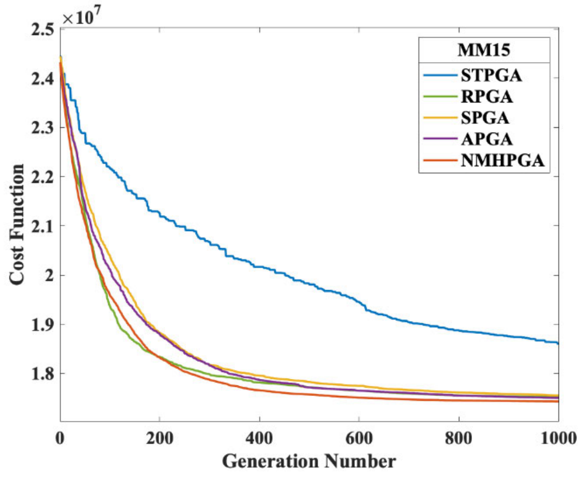


FIGURE 30. Multi-machine scheduling using the MM15 model.

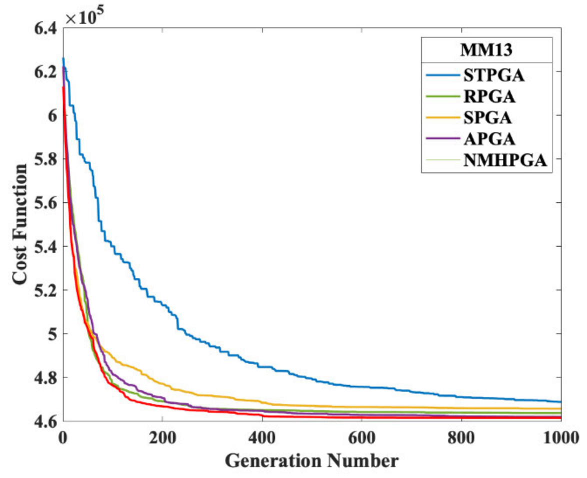


FIGURE 28. Multi-machine scheduling using the MM13 model.

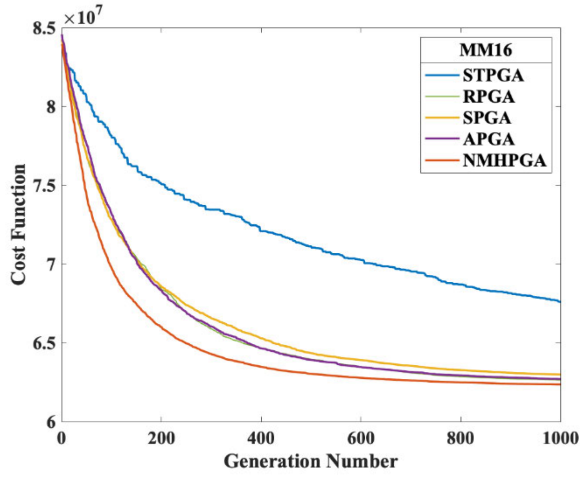


FIGURE 31. Multi-machine scheduling using the MM16 model.

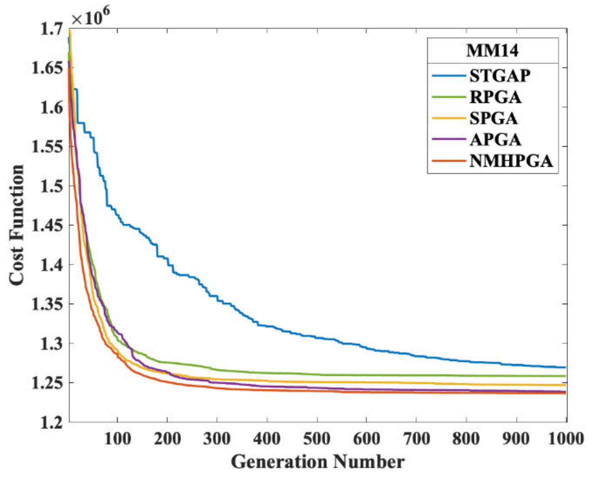


FIGURE 29. Multi-machine scheduling using the MM14 model.

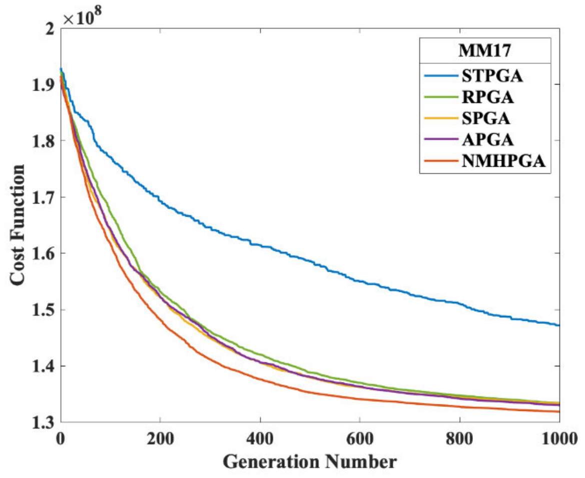


FIGURE 32. Multi-machine scheduling using the MM17 model.

The furnace model for this project should accurately account for all requirements, such as the amount of time and fuel consumed. The applied furnace model is a multi-objective function with three objectives; objective one is primarily focused on reducing time, objective two is based on reducing energy, and objective three is simultaneously reducing time and energy. Before applying the three objectives, the furnace takes 139.2167 (h) and consumes

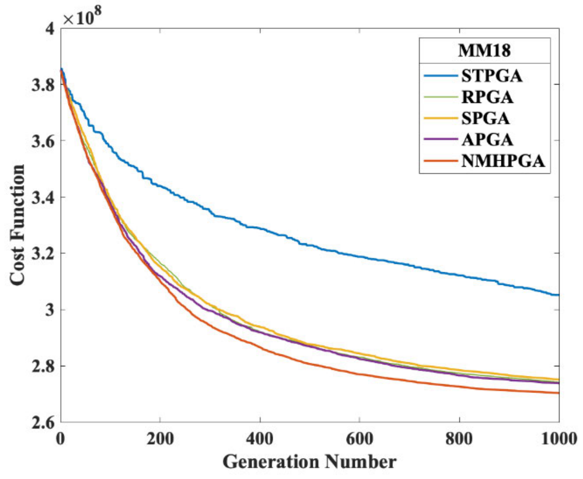


FIGURE 33. Multi-machine scheduling using the MM18 model.

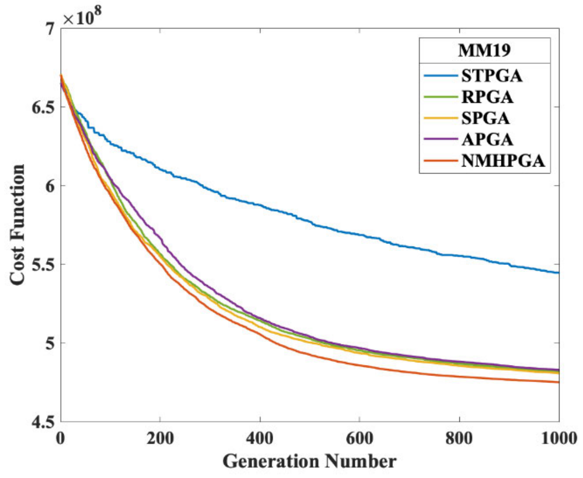


FIGURE 34. Multi-machine scheduling using the MM19 model.

TABLE 9. Comparison of objective functions of different algorithms tested on job shops in multi-machines category-B.

JS-Type	STPGA	RPGA	SPGA	APGA	NMHPGA
MM1	7931	7850	7893	7700	7758
MM2	9481	9469	9523	9485	9465
MM3	7895	8514	8277	8023	7913
MM4	7931	7850	7893	7700	7758
MM5	7104	7582	7306	7054	7039
MM6	7849	7907	7918	7785	7798
MM7	9428	9387	9399	9210	9181
MM8	8728	9039	9209	8824	8727
MM9	6255	6258	6237	6198	6195
MM10	7931	7850	7893	7700	7758
Mean	8053.3	8170.6	8154.8	7967.9	7959.2
STD	979.2	967.2	1011.4	988.6	969.4

86.8673E6 (m^3/h) fuel. Table 13 illustrates the consumed energy for different algorithms for three objectives; objective three is more efficient due to the focus on both time and energy consumption simultaneously; as illustrated, NMHPGA consumed 83.9666E6 (m^3/h) of fuel, with more efficient and faster results compared to other algorithms. Table 14 shows the elapsed time after optimizing the schedules achieving more efficient results by applying NMHPGA for this research. Figure 5 illustrates the furnace model function for three objectives using NMHPGA.

TABLE 10. Comparison of convergence speed of different algorithms tested on job shops in multi-machines category-B.

JS-Type	STPGA	RPGA	SPGA	APGA	NMHPGA
MM1	991	597	127	55	818
MM2	435	152	177	115	50
MM3	219	385	67	85	76
MM4	908	590	99	60	812
MM5	583	664	382	410	336
MM6	429	406	447	420	543
MM7	938	136	120	147	90
MM8	565	733	905	343	212
MM9	952	141	81	230	635
MM10	986	592	104	76	816

TABLE 11. Comparison of objective functions of different algorithms tested on job shops in multi-machines category-C.

JS-Type	STPGA	RPGA	SPGA	APGA	NMHPGA
MM11	17043	16503	16634	16546	16421
MM12	137736	131970	131274	133618	131115
MM13	468806	463734	465696	461970	461538
MM14	1269516	1258531	1246911	1238453	1236658
MM15	18603560	17521764	17555825	17503567	17432983
MM16	67602116	62639047	62988711	62694341	62354613
MM17	147200593	133407491	133306194	133005781	131840576
MM18	305201786	274169417	275108844	273802285	270317679
MM19	544267800	482106036	480728988	482871193	475082286

TABLE 12. Comparison of convergence speed of different algorithms tested on job shops in multi-machines category-C.

JS-Type	STPGA	RPGA	SPGA	APGA	NMHPGA
MM11	995	364	661	915	143
MM12	993	825	744	816	401
MM13	993	396	445	283	279
MM14	993	329	403	518	302
MM15	998	454	641	532	493
MM16	990	715	908	878	774
MM17	992	942	954	922	920
MM18	986	982	984	979	979
MM19	993	984	988	988	933

TABLE 13. Consumed energy x1E6 (m^3/h).

FS Model	STPGA	RPGA	SPGA	APGA	NMHPGA
OBJ 1	83.9666	84.0094	84.0703	84.0942	83.9666
OBJ 2	84.3083	84.3982	84.2927	84.2316	84.0847
OBJ 3	83.9666	84.1739	84.0167	84.1451	83.9666

TABLE 14. Elapsed time after optimization (H).

FS Model	STPGA	RPGA	SPGA	APGA	NMHPGA
OBJ 1	133.7500	133.8167	133.9833	133.9500	133.7500
OBJ 2	134.4500	134.5833	134.4000	134.3167	134.0000
OBJ 3	133.7500	134.2167	133.8333	134.1500	133.7500

VIII. CONCLUSION AND FUTURE WORK

This paper established a novel hybrid metaheuristic method based on the combination of different types of selection of genetic algorithms.

Ten groups of mathematical benchmark functions, three categories of benchmarks, and a furnace made from the industry were selected to evaluate the established algorithm's performance. The algorithm performance was compared with four other algorithms.

The most important findings and summary of results of this paper are as follows:

(i) This paper considers different types of GAs with varying kinds of selections.

(ii) The PGAs are tested with different selection procedures, which conclude that a combined solution is better than an individual.

(iii) The ethnic selection procedure is the best, as it combines the best individuals from different groups. Combining the ethnic selection with the PGA shows better results can be achieved without lengthy crossover procedures.

(iv) The advantage of this approach is an improvement in the speed of convergence and the global search point.

(v) The NMHPGA, which removes the crossover function and replicates it with swap, insert, and reverse functions, combined with ethnic selection, improves effectiveness and performance due to the operators performing on a single chromosome.

Three categories of job shop benchmarks have been applied to test the established NMHPGA. However, other selection and combination functions can be integrated into future works to improve efficiency with fewer genes. Moreover, more complex benchmarks and industrial case studies can be applied to test the algorithm. As for the convergence speed, it is about finding what iteration the error (cost) reaches to a steady state; this means the best solution is found, and there is no need to keep the algorithm running as the error will not change; this is useful in finding the best solution in a few generations, which takes less time and resources.

REFERENCES

[1] F. Glover, "Future paths for integer programming and links to artificial intelligence," *Comput. Oper. Res.*, vol. 13, no. 5, pp. 533–549, Jan. 1986.

[2] H. de Garis, "Introduction to evolutionary computing," *Evol. Comput.*, vol. 12, no. 2, pp. 269–271, Jun. 2004, doi: 10.1162/evco.2004.12.2.269.

[3] D. Biskup, "Single-machine scheduling with learning considerations," *Eur. J. Oper. Res.*, vol. 115, no. 1, pp. 173–178, 1999, doi: 10.1016/S0377-2217(98)00246-X.

[4] Q.-V. Dang, T. van Diessen, T. Martagan, and I. Adan, "A matheuristic for parallel machine scheduling with tool replacements," *Eur. J. Oper. Res.*, vol. 291, no. 2, pp. 640–660, Jun. 2021, doi: 10.1016/J.EJOR.2020.09.050.

[5] M. L. Pinedo, "Scheduling: Theory, algorithms, and systems," in *Scheduling: Theory, Algorithms, Systems*, vol. 9781461423614, 4th ed., Feb. 2012, pp. 1–673, doi: 10.1007/978-1-4614-2361-4.

[6] D. E. Goldberg and J. H. Holland, "Genetic algorithms in search, optimization, and machine learning David E. Goldberg. The University of Alabama T," *Mach. Learn.*, vol. 3, no. 2, pp. 95–99, 1979.

[7] K. Tamssaouet, S. Dauzère-Pérès, and C. Yugma, "Metaheuristics for the job-shop scheduling problem with machine availability constraints," *Comput. Ind. Eng.*, vol. 125, pp. 1–8, Nov. 2018, doi: 10.1016/J.CIE.2018.08.008.

[8] N. Mladenović and P. Hansen, "Variable neighborhood search," *Comput. Oper. Res.*, vol. 24, no. 11, pp. 1097–1100, Nov. 1997, doi: 10.1016/S0305-0548(97)00031-2.

[9] S. Talatahari, M. Azizi, M. Tolouei, B. Talatahari, and P. Sareh, "Crystal structure algorithm (CryStAl): A Metaheuristic optimization method," *IEEE Access*, vol. 9, pp. 71244–71261, 2021, doi: 10.1109/ACCESS.2021.3079161.

[10] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1992.

[11] Q. Lin, Q. Zhu, P. Huang, J. Chen, Z. Ming, and J. Yu, "A novel hybrid multi-objective immune algorithm with adaptive differential evolution," *Comput. Oper. Res.*, vol. 62, pp. 95–111, Oct. 2015, doi: 10.1016/J.COR.2015.04.003.

[12] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997.

[13] P. Moscato, "On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms," *Caltech Concurrent Comput. Program*, vol. 826, p. 1989, Sep. 1989.

[14] H.-G. Beyer and H.-P. Schwefel, "Evolution strategies A comprehensive introduction," *ACM Comput. Classification*, vol. 1, pp. 3–52, Mar. 2002.

[15] D. B. Fogel, "Evolutionary computation: The fossil record. Selected readings on the history of evolutionary algorithms," 1998, p. 641. [Online]. Available: https://ieeexplore.ieee.org/book/5263042

[16] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: Optimization by a colony of cooperating agents," *IEEE Trans. Syst., Man, Cybern., B Cybern.*, vol. 26, no. 1, pp. 29–41, Feb. 1996, doi: 10.1109/3477.484436.

[17] M. K. Tiwari and F. T. S. Chan, "Swarm intelligence, focus on ant and particle swarm optimization," in *Swarm Intelligence, Focus on Ant and Particle Swarm Optimization*, Mar. 2019, p. 548, doi: 10.5772/48.

[18] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. MHS 6th Int. Symp. Micro Mach. Hum. Sci.*, Oct. 1995, pp. 39–43.

[19] A. Kumar, D. Kumar, and S. K. Jarial, "A review on artificial bee colony algorithms and their applications to data clustering," *Cybern. Inf. Technol.*, vol. 17, no. 3, pp. 3–28, Sep. 2017, doi: 10.1515/cait-2017-0027.

[20] S.-C. Chu, P.-W. Tsai, and J.-S. Pan, "Cat swarm optimization," in *Trends in Artificial Intelligence* (Lecture Notes in Computer Science), vol. 4099. Berlin, Germany: Springer, 2006, pp. 854–858, doi: 10.1007/11801603_94.

[21] F. de Rango, N. Palmieri, and M. Tropea, "Multirobot coordination through bio-inspired strategies," in *Nature-Inspired Computation and Swarm Intelligence*, 2020, pp. 361–390, doi: 10.1016/B978-0-12-819714-1.00030-0.

[22] X. Yang, "Nature-inspired Metaheuristic algorithms," in *Nature-Inspired Metaheuristic Algorithms*, 2nd ed., 2010, p. 115.

[23] S. Walton, O. Hassan, K. Morgan, and M. R. Brown, "A review of the development and applications of the cuckoo search algorithm," in *Swarm Intelligence and Bio-Inspired Computation: Theory and Applications*, Jan. 2013, pp. 257–271, doi: 10.1016/B978-0-12-405163-8.00011-9.

[24] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, May 1983, doi: 10.1126/SCIENCE.220.4598.671.

[25] V. T'Kindt and J. C. Billaut, *Multicriteria scheduling*. Cham, Switzerland: Springer, 2006, doi: 10.1007/b106275.

[26] B. Jarboui, P. Siarry, and J. Teghem, "Metaheuristics for production scheduling," 2013, p. 501. [Online]. Available: https://www.wiley.com/en-gb/Metaheuristics+for+Production+Scheduling-p-9781118731567, doi: 10.1002/9781118731598.

[27] H. Xiong, S. Shi, D. Ren, and J. Hu, "A survey of job shop scheduling problem: The types and models," *Comput. Oper. Res.*, vol. 142, Jun. 2022, Art. no. 105731, doi: 10.1016/J.COR.2022.105731.

[28] S. Tian, T. Wang, L. Zhang, and X. Wu, "An energy-efficient scheduling approach for flexible job shop problem in an internet of manufacturing things environment," *IEEE Access*, vol. 7, pp. 62695–62704, 2019, doi: 10.1109/ACCESS.2019.2915948.

[29] W. Shao, Z. Shao, and D. Pi, "Modeling and multi-neighborhood iterated greedy algorithm for distributed hybrid flow shop scheduling problem," *Knowl.-Based Syst.*, vol. 194, Apr. 2020, Art. no. 105527, doi: 10.1016/J.KNOSYS.2020.105527.

[30] T. Liu, Y. Chen, and J. Chou, "Solving distributed and flexible job-shop scheduling problems for a real-world fastener manufacturer," *IEEE Access*, vol. 2, pp. 1598–1606, 2014, doi: 10.1109/ACCESS.2015.2388486.

[31] J. Zhang, G. Ding, Y. Zou, S. Qin, and J. Fu, "Review of job shop scheduling research and its new perspectives under industry 4.0," *J. Intell. Manuf.*, vol. 30, no. 4, pp. 1809–1830, Apr. 2019, doi: 10.1007/s10845-017-1350-2.

[32] S. Tao, "An improved genetic algorithm for solving TSP," *J. Phys., Conf. Ser.*, vol. 1952, no. 4, Jun. 2021, Art. no. 042067, doi: 10.1088/1742-6596/1952/4/042067.

[33] K. Gao, Z. Cao, L. Zhang, Z. Chen, Y. Han, and Q. Pan, "A review on swarm intelligence and evolutionary algorithms for solving flexible job shop scheduling problems," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 4, pp. 904–916, Jul. 2019, doi: 10.1109/JAS.2019.1911540.

[34] C. Cheng, S. Li, K. Ying, and Y. Liu, "Scheduling jobs of two competing agents on a single machine," *IEEE Access*, vol. 7, pp. 98702–98714, 2019, doi: 10.1109/ACCESS.2019.2929582.

[35] E. Jiang, L. Wang, and J. Wang, "Decomposition-based multi-objective optimization for energy-aware distributed hybrid flow shop scheduling with multiprocessor tasks," *Tsinghua Sci. Technol.*, vol. 26, no. 5, pp. 646–663, Oct. 2021.

[36] J. Li, Y. Du, K. Gao, P. Duan, D. Gong, Q. Pan, and P. N. Suganthan, "A hybrid iterated greedy algorithm for a crane transportation flexible job shop problem," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 3, pp. 2153–2170, Jul. 2022, doi: 10.1109/TASE.2021.3062979.

[37] D. Trentesaux, C. Pach, A. Bekrar, Y. Sallez, T. Berger, T. Bonte, P. Leitão, and J. Barbosa, "Benchmarking flexible job-shop scheduling and control systems," *Control Eng. Pract.*, vol. 21, no. 9, pp. 1204–1225, Sep. 2013, doi: 10.1016/J.CONENGPRAC.2013.05.004.

[38] J. Xiong, L.-N. Xing, and Y.-W. Chen, "Robust scheduling for multi-objective flexible job-shop problems with random machine breakdowns," *Int. J. Prod. Econ.*, vol. 141, no. 1, pp. 112–126, Jan. 2013, doi: 10.1016/J.IJPE.2012.04.015.

[39] P. Brucker and R. Schlie, "Job-shop scheduling with multi-purpose machines," *Computing*, vol. 45, no. 4, pp. 369–375, Dec. 1990, doi: 10.1007/BF02238804.

[40] S.-W. Lin, K.-C. Ying, and C.-Y. Huang, "Minimising makespan in distributed permutation flowshops using a modified iterated greedy algorithm," *Int. J. Prod. Res.*, vol. 51, no. 16, pp. 5029–5038, Aug. 2013, doi: 10.1080/00207543.2013.790571.

[41] H. Bargaoui, O. B. Driss, and K. Ghédira, "A novel chemical reaction optimization for the distributed permutation flowshop scheduling problem with makespan criterion," *Comput. Ind. Eng.*, vol. 111, pp. 239–250, Sep. 2017, doi: 10.1016/J.CIE.2017.07.020.

[42] L. Meng, C. Zhang, X. Shao, and Y. Ren, "MILP models for energy-aware flexible job shop scheduling problem," *J. Cleaner Prod.*, vol. 210, pp. 710–723, Feb. 2019, doi: 10.1016/J.JCLEPRO.2018.11.021.

[43] N. Al-Hinai and T. Y. ElMekkawy, "Robust and stable flexible job shop scheduling with random machine breakdowns using a hybrid genetic algorithm," *Int. J. Prod. Econ.*, vol. 132, no. 2, pp. 279–291, Aug. 2011, doi: 10.1016/J.IJPE.2011.04.020.

[44] M. Dai, D. Tang, A. Giret, and M. A. Salido, "Multi-objective optimization for energy-efficient flexible job shop scheduling problem with transportation constraints," *Robot. Comput.-Integr. Manuf.*, vol. 59, pp. 143–157, Oct. 2019, doi: 10.1016/J.RCIM.2019.04.006.

[45] S. N. Sivanandam and S. N. Deepa, "Genetic algorithms," in *Introduction to Genetic Algorithms*, 2008, pp. 15–37, doi: 10.1007/978-3-540-73190-0_2.

[46] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: Past, present, and future," *Multimedia Tools Appl.*, vol. 80, no. 5, pp. 8091–8126, Feb. 2021, doi: 10.1007/s11042-020-10139-6.

[47] J. Luo, D. El Baz, R. Xue, and J. Hu, "Solving the dynamic energy aware job shop scheduling problem with the heterogeneous parallel genetic algorithm," *Future Gener. Comput. Syst.*, vol. 108, pp. 119–134, Jul. 2020, doi: 10.1016/J.FUTURE.2020.02.019.

[48] M. M. Raghuwanshi and O. G. Kakde, "Genetic algorithm with species and sexual selection," in *Proc. IEEE Conf. Cybern. Intell. Syst.*, Jun. 2006, pp. 1–8.

[49] K. Omori, S. Maekawa, H. Tamaki, and S. Kitamura, "Parallelization of genetic algorithm with sexual selection," *Electr. Eng. Jpn.*, vol. 150, no. 1, pp. 42–49, Jan. 2005, doi: 10.1002/eej.20029.

[50] A. Ghosh, S. Tsutsui, and H. Tanaka, "Individual aging in genetic algorithms," in *Proc. Austral. New Zealand Conf. Intell. Inf. Syst. (ANZIIS)*, Nov. 1996, pp. 276–279.

[51] N. Kubota and T. Fukuda, "Genetic algorithms with age structure," *Soft Comput. Fusion Found., Methodolog. Appl.*, vol. 1, no. 4, pp. 155–161, Dec. 1997.

[52] M. J. Li and T. S. Tong, "A partheno-genetic algorithm and analysis of its global convergence," *Zidonghua Xuebao/Acta Automatica Sinica*, vol. 25, no. 1, pp. 68–72, 1999.

[53] X. Yu, X. Liao, W. Li, X. Liu, and Z. Tao, "Logistics automation control based on machine learning algorithm," *Cluster Comput*, vol. 22, no. 6, pp. 14003–14011, Nov. 2019, doi: 10.1007/s10586-018-2169-0.

[54] J. Wang, W. Huang, G. Ma, and S. Chen, "An improved partheno genetic algorithm for multi-objective economic dispatch in cascaded hydropower systems," *Int. J. Electr. Power Energy Syst.*, vol. 67, pp. 591–597, May 2015, doi: 10.1016/J.IJEPES.2014.12.037.

[55] R. Shuai, W. Jing, and X. Zhang, "Research on chaos partheno-genetic algorithm for TSP," in *Proc. Int. Conf. Comput. Appl. Syst. Model. (ICCASM)*, Oct. 2010, p. 290, doi: 10.1109/ICCASM.2010.5619417.

[56] A. Momenikorbekandi and M. F. Abbod, "Multi-ethnicity genetic algorithm for job shop scheduling problems," Tech. Rep., 2022, doi: 10.5013/IJSSST.a.22.01.13.

[57] D. B. Fogel, T. Bäck, and Zbigniew. *Michalewicz, Evolutionary Computation* (Advanced Algorithms and Operators). vol. 2. London, U.K.: Institute of Physics, 2000. Accessed: Jan. 5, 2023. [Online]. Available: https://www.routledge.com/Evolutionary-Computation-2-Advanced-Algorithms-and-Operators/Baeck-Fogel-Michalewicz/p/book/9780750306652

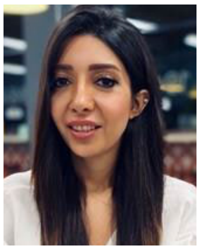
[58] M. Locatelli, "A note on the griewank test function," *J. Global Optim.*, vol. 25, no. 2, pp. 169–174, Feb. 2003.

[59] *Optimization Test Functions and Datasets*. Accessed: Jan. 5, 2023. [Online]. Available: https://www.sfu.ca/~ssurjano/optimization.html

[60] H. Cho, F. Olivera, and S. D. Guikema, "A derivation of the number of minima of the griewank function," *Appl. Math. Comput.*, vol. 204, no. 2, pp. 694–701, Oct. 2008, doi: 10.1016/j.amc.2008.07.009.

[61] M. Molga and C. Smutnicki, "Test functions for optimization needs," Tech. Rep., 2005.

[62] N. Yoshitani and A. Hasegawa, "Model-based control of strip temperature for the heating furnace in continuous annealing," *IEEE Trans. Control Syst. Technol.*, vol. 6, no. 2, pp. 146–156, Mar. 1998.



ATEFEH MOMENIKORBEKANDI is a B.Sc. and M.Sc. Doctoral Researcher in AI in manufacturing systems, Department of Electronic and Electrical Engineering Brunel University London. She completed her bachelor's degree in industrial engineering, working on automation systems and robotics in the manufacturing system of the automobile industry. Her Ph.D. project is the application of AI in the manufacturing systems, specifically optimizing job shop scheduling problems and machine learning. In this project, based on AI algorithms, a novel algorithm will be developed to increase the productivity and optimization of the production line.

MAYSAM F. ABBOD (Senior Member, IEEE) received the Ph.D. degree in control engineering from The University of Sheffield, in 1992. From 1993 to 2006, he was with the Department of Automatic Control and Systems Engineering, The University of Sheffield, as a Research Associate and a Senior Research Fellow. His developed systems were applied to industrial and biomedical modeling and computer control of manufacturing systems. His main research interests include intelligent systems for modeling, control, and optimization.