IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 25 September 2022, accepted 6 October 2022, date of publication 14 October 2022, date of current version 9 November 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3213805

RESEARCH ARTICLE

An Experimental Study of a Fuzzy Adaptive Emperor Penguin Optimizer for Global Optimization Problem

MD. ABDUL KADER[1], KAMAL Z. ZAMLI[1,2], (Member, IEEE), AND BASEM YOUSEF ALKAZEMI[3], (Senior Member, IEEE)

[1]Faculty of Computing, College of Computing and Applied Sciences, Universiti Malaysia Pahang, Pekan, Pahang 26600, Malaysia
[2]Faculty of Science and Technology, Universitas Airlangga-C Campus, Surabaya 60115, Indonesia
[3]Department of Computer Science, College of Computer and Information Systems, Umm Al-Qura University, Mecca 24382, Saudi Arabia

Corresponding author: Kamal Z. Zamli (kamalz@ump.edu.my)

This work was supported in part by the Deanship of Scientific Research (DSR), Umm Al-Qura University, under Grant 22UQU4210132DSR02; and in part by the Universiti Malaysia Pahang (UMP) Research Grant titled "An Automatic Research Profiling System for UMP employing UMPIR data from Universiti Malaysia Pahang" under Grant RDU192211.

ABSTRACT Emperor Penguin Optimizer (EPO) is a recently developed population-based meta-heuristic algorithm that simulates the huddling behavior of emperor penguins. Mixed results have been observed on the performance of EPO in solving general optimization problems. Within the EPO, two parameters need to be tuned (namely f and l) to ensure a good balance between exploration (i.e., roaming unknown locations) and exploitation (i.e., manipulating the current known best). Since the search contour varies depending on the optimization problem, the tuning of f and l is problem-dependent, and there is no one-size-fits-all approach. To alleviate these problems, an adaptive mechanism can be introduced in EPO. This paper proposes a fuzzy adaptive variant of EPO, namely Fuzzy Adaptive Emperor Penguin Optimizer (FAEPO), to solve this problem. As the name suggests, FAEPO can adaptively tune the parameters f and l throughout the search based on three measures (i.e., quality, success rate, and diversity of the current search) via fuzzy decisions. A test suite of twelve optimization benchmark test functions and three global optimization problems (Team Formation Optimization - TFO, Low Autocorrelation Binary Sequence - LABS, and Modified Condition/Decision Coverage - MC/DC test case generation) were solved using the proposed algorithm. The respective solution results of the benchmark meta-heuristic algorithms were compared. The experimental results demonstrate that FAEPO significantly improved the performance of its predecessor (EPO) and gives superior performance against the competing meta-heuristic algorithms, including an improved variant of EPO (IEPO).

INDEX TERMS Emperor penguin optimizer, fuzzy adaptive EPO, fuzzy inference system, low autocorrelation binary sequence, meta-heuristic algorithms, MC/DC test case generation, team formation optimization.

I. INTRODUCTION

meta-heuristic algorithms have been known to be effective for solving real-world optimization problems. Although regarded as an approximation approach, meta-heuristic algorithms often excel in the case when the search spaces are significantly large, owing to their ability to produce good enough solutions within polynomial time [1]. A plethora of meta-heuristic algorithms has been introduced in the scientific literature over the last few decades [2], [3]. The main feature of meta-heuristic algorithms is that they do not require much knowledge about the problem space. Mainly, they aim to achieve the global optima of a given problem by balancing the quality of solution and time cost. Moreover, they work with a randomly generated set of candidate solutions known as population and can be implemented easily due to their simple structure.

Many meta-heuristic algorithms have been suggested during the last few decades. For example, swarm intelligence (SI) and population-based methods incorporate many

The associate editor coordinating the review of this manuscript and approving it for publication was Ehab Elsayed Elattar.

nature-inspired algorithms where the swarms are inspired by the collective behavior of the species in nature [4]. Without any centralized control, these species have the ability to position themselves for achieving their goal in a decent way [5]. SI-based algorithms are trendy in the field of global optimization problems. There are many SI based algorithms proposed in the literature, such as Emperor Penguin Optimizer (EPO) [6], Aquila Optimizer (AO) [7], Coot Optimization Algorithm (COA) [8], Salp Swarm Algorithm (SSA) [9], Artificial Bee Colony Algorithm (ABCA) [10], Cuckoo Search Algorithm (CSA) [11], Marine Predators Algorithm (MPA) [12]. Their effectiveness and performance are compared and evaluated by adopting them in many real-world optimization problems.

A successful meta-heuristic algorithm should judiciously combine the benefits of exploration and exploitation to produce optimal results. A good exploration capability of a meta-heuristic algorithm ensures exploring all the potential search regions to find the global solution. In contrast, good exploitation manipulates the solution based on the best solution to enhance existing ones. Although a good balance between them may improve solution accuracy, there is no fundamental theoretical knowledge or framework in the literature on building a credible balance between these two concepts [13]. According to the literature study, putting too much emphasis on exploitation tends to deny a diverse solution and may lead to local optima, whereas putting too much emphasis on exploration consumes significant computing resources and prevent convergence.

Most meta-heuristic algorithms introduce control parameters to balance exploitation and exploration phases efficiently. For example, Crow Search Algorithm (CSA) [14] exploits awareness probability; Salp Swarm Algorithm (SSA) [9] introduces coefficient c1; Owl Search Algorithm (OSA) [15] relies on linearly decreasing constant β; Sooty Tern Optimization Algorithm (STOA) [16] exploits random variable CB and spiral behavior; Squirrel Search Algorithm (SQSA) [17] presents gliding constant G_c. The adjustment of the parameters thus ensures an appropriate quality solution (i.e., based on an intertwine balance and compromise between exploration and exploitation). Nonetheless, fine-tuning these parameters are often time-consuming and problem-specific, since no universal size fits all approaches [18].

A swarm-based algorithm called emperor penguin optimizer (EPO) that emulates the huddling behavior of emperor penguins was proposed by Dhiman and Kumar in 2018 [6]. This newly proposed EPO is successfully adopted in many engineering optimization problems (e.g., [19], [20], [21], [22], [23], [24], [25], [26], and [27]). The advancement of research on EPO from its invention is reviewed in [28] and categorized the variants of EPO as improved EPO [29], [30], [31], [32], [33], hybrid EPO [34], [35], [36], [37], [38], multi-objective EPO [39], [40], [41], [42], and chaotic EPO [43], [44]. Despite its potential, EPO also has the shortcoming of exploration ability [30], weak randomness ability [29], ease to fall into local optimum [29], [30], [31], [32], [33], and premature convergence [39]. These flaws are caused mainly by adequately tuning the responsible parameters that directly impact balancing the exploitation and exploration phases of the search space. This study proposes a fuzzy adaptive EPO (FAEPO) to boost EPO's performance and dynamically balance exploitation and exploration capabilities. A fuzzy-based adaptive parameter tuning method is proposed to balance the exploration and exploitation during the search, thus improving the performance of the algorithm. Specifically, this study makes the following significant contributions:

- FAEPO integrates a Mamdani fuzzy inference system that provides adaptive control of exploration and exploitation by tuning the control parameters, f and l, based on the need for a particular search problem.
- FAEPO improves the time complexity of the original EPO via factoring out the unnecessary fitness calculation for each iteration when no position update is performed.
- FAEPO is subjected to twelve optimization benchmark test functions and three optimization problems (i.e., team formation optimization, low autocorrelation binary sequence-LABS, MC/DC test case generation) as case studies to demonstrate its applicability and generality in many real-world applications.

The remainder of this paper is organized as follows. Section 2 provides an overview and related works on meta-heuristic algorithms that integrate fuzzy inference systems. An overview of the original EPO algorithm is presented in Section 3. A detailed description of the proposed FAEPO algorithm is presented in Section 4. Section 5 provides our empirical evaluation along with our research questions. Research questions are answered elaborately in Section 6. In Section 7, an overall observation and discussion on FAEPO are presented. Finally, we summarize this work in Section 8 and offer directions for future research.

II. OVERVIEW AND RELATED WORKS ON FUZZY INFERENCE SYSTEM INTEGRATION IN META-HEURISTIC ALGORITHM

Humans are far superior to deterministic systems or computers at certain operations, such as obstacle avoidance while driving or planning a strategy. For the complex processes, humans carry them out by simple rules gleaned from their experiences. This could result from humans' exceptional reasoning and complicated cognitive processing capabilities.

Fuzzy logic or many-valued logic is the practical alternative to model human reasoning by using if-then fuzzy rules based on the fuzzy set theory proposed by L. A. Zadeh [45]. The fuzzy set theory offers a systematic methodology for dealing with linguistic information and increases the accuracy of numerical computation through the use of linguistic labels specified by membership functions [46], [47].

Fuzzy Inference Systems (FIS) employs fuzzy logic to map the inputs to outputs in three steps effectively: fuzzification, fuzzy rules inference evaluation (i.e., decision-making), and defuzzification [48]. Fuzzification is the process of

transforming the input data into the appropriate linguistic terms (records in Table 3 can be considered as an example). This stage (Figure 3 can be viewed as an example) involves the application of membership functions to associate system input and output values with fuzzy input and output membership values. The second step, fuzzy rules inference evaluation, employs rules (record in Table 4 can be considered as an example) to infer fuzzy control actions in response to fuzzy inputs. The third step, defuzzification, employs the defuzzification formula and fuzzy output membership values to generate a single crisp value [48].

Mamdani, Sugeno, and Tsukamoto are three types of fuzzy inference systems (FIS) that have been widely employed in both scenarios. Their strategy sets them apart to generate crisp output from fuzzy inputs. Especially the Center of Gravity (COG), the Weighted Average (WA), and the Height Method (HM) are used by the Mamdani, Sugeno, and Tsukamoto FISs, respectively, to determine the crisp output [49]. Mamdani FIS is the most widely used of the three because of its simple structure, which produces reasonable results, and its rule base, which is intuitive and interpretable [50], [51], [52]. The ability to design a system with greater flexibility is another benefit.

On the other hand, parameter settings (PS) of meta-heuristic algorithms are one of the long-standing grand challenges. PS has a substantial impact on the performance and control of the behavior of meta-heuristic algorithms [53]. Thus, the parameters of the meta-heuristic algorithm must be appropriately controlled (i.e., known as the parameter control problem) or fine-tuned (i.e., known as the parameter tuning problem) in order to achieve high performance. Furthermore, there are only a handful of parameter-free algorithms in the literature to date (e.g., Teaching Learning based Optimization (TLBO) [54], Jaya Algorithm (JA) [55], and Symbiotic Optimization Search (SOS) [56]).

Parameter control and parameter tuning problems are clearly related, but bear important differences. Typically, parameter tuning techniques require a large number of runs in order to examine algorithms' performance on a single problem instance or a set of problem instances with varying parameter settings. This adds to the time required for parameter adjustment, which is the primary downside of parameter tuning. A good tuning procedure may be used to tune the parameters of a wide variety of meta-heuristic algorithms that implies the universality of the parameter tuning procedure [57]. On the other hand, parameter control has the obvious disadvantage of not being universal, since the parameter control techniques for one algorithm are not always good for another algorithm [58]. Additionally, a rough idea is required to change parameters dynamically to achieve good performance to properly design parameter control procedures for a meta-heuristic algorithm [59]. Thus, parameter tuning is more convenient and practical than parameter control in the senses mentioned above.

However, the integration of FIS in meta-heuristic algorithms is not a novel concept to solve the parameter control and parameter tuning problems. The focus of this paper is on parameter tuning. Many interesting contributions have been published recently within this field. A number of recent related works on parameter tuning by employing FIS in meta-heuristic algorithms are recorded in Table 1, to list a few. Table 1 records the author names with reference, publication year, standard meta-heuristic algorithm, tuned parameters, fuzzy inputs, and fuzzy outputs data.

As for particle swarm optimization (PSO), the parameters listed in Table 1, which have been tuned by using FIS in various researches, are inertia weight [60], [64], particle velocity [61], [63], acceleration coefficients [62], learning factors [61], [62], [63], [64], [67], cognitive factor [65], [66], social factor [65], [66], and momentum weight factor [67]. Liu et al. [68] and Qi and Chunming [69] tuned the same parameters (i.e., mutation and crossover) to improve the performance of the Genetic Algorithm (GA). In [70], [73], [74], [76], [78], and [80], the iterations are considered as fuzzy input to tune the corresponding parameters for Gravitational Search Algorithm (GSA), Harmony Search Algorithm (HSO), Flower Pollination Algorithm (FPA), Bee Colony Optimization (BCO), and Bird Swarm Algorithm (BSA). Neyoy et al. [71] tuned the alpha parameter using the errors and the changes in the errors to improve Ant Colony Optimization (ACO). Bidar and Kanan [72] use count and delta as fuzzy inputs and alpha and gamma as fuzzy outputs for the Modified Firefly Algorithm (MFA). To tune the mutation parameter of differential evolution (DE), Ochoa et al. [75] use the number of generations as fuzzy input. Bernal et al. [77] used decades as fuzzy input for the β and η parameters of the Imperialist Competitive Algorithm (ICA). In the Chimp Optimization Algorithm (ChOA), Saffari et al. [79] tuned two random variables using the number of repetitions and the parameter f.

The works in the literature that select the most influential operators using FIS are listed in Table 2, to name a few. In Table 2, the authors' names with the corresponding reference, the year of publication, the standard meta-heuristic algorithm, the influential operators, the fuzzy inputs, and the fuzzy outputs data are recorded. It should be noted that the choice of proper control operators requires some expertise. The inappropriate choice of control operators may contribute to poorer performance. According to the research listed in Table 1 and Table 2, integrating fuzzy for parameter tuning and control with meta-heuristic algorithms maximizes the performance of standard meta-heuristic algorithms.

The preceding overview of related research shows that the combination of fuzzy approaches and meta-heuristic algorithms is currently receiving a significant amount of attention from researchers, and the number of studies that have begun to address these two topics together has started to steadily increase day by day. Because of the growing complexity and unpredictability of the problems, it has become essential to combine meta-heuristic algorithms with fuzzy approaches in order to generate results that are both more effective and more reliable. Therefore, this study aims to propose a new fuzzy

TABLE 1. Related works on parameter tuning by employing FIS in meta-heuristics.

SI	Author [Reference]	Year Published	Standard Meta-heuristic	Tuned Parameters	Fuzzy Inputs	Fuzzy Outputs
1	Shi and Eberhart [61]	2001	PSO	◗ inertia weight	◗ current best solution ◗ current inertia weight	◗ updated inertia weight
2	Liu et al. [62]	2007	PSO	◗ minimum velocity threshold of particles	◗ current best performance evaluation ◗ current velocity	◗ scaling factor ◗ velocity threshold
3	Juang et al. [63]	2011	PSO	◗ acceleration coefficients	◗ difference between two successive global bests particle value ◗ iteration counter	◗ two acceleration coefficients
4	Liu and Ma [64]	2011	PSO	◗ velocity		◗ velocity
5	Niknam et al. [65]	2012	PSO	◗ inertia weight ◗ learning factors	◗ best fitness ◗ number of generations	◗ inertia weight ◗ learning factors
6	Melin et al. [66]	2013	PSO	◗ cognitive factor ◗ social factor	◗ iteration ◗ diversity error	◗ cognitive factor ◗ social factor
7	Olivas et al. [67]	2014	PSO	◗ cognitive factor ◗ social factor	◗ iteration ◗ diversity	◗ cognitive factor ◗ social factor
8	Mahmoud and Ahmed [68]	2015	PSO	◗ learning factors ◗ momentum weight factor	◗ normalized best fitness ◗ normalized unchanged best fitness	◗ learning factors ◗ momentum weight factor
9	Liu et al. [69]	2005	GA	◗ mutation ◗ crossover	◗ maximum fitness ◗ average fitness	◗ mutation probability ◗ crossover probability
10	Qi and Chunming [70]	2010	GA	◗ mutation ◗ crossover	◗ maximum fitness ◗ average fitness	◗ mutation probability ◗ crossover probability
11	Sombra et al. [71]	2013	GSA	◗ alpha parameter	◗ iteration	◗ alpha
12	Neyoy et al. [72]	2013	ACO	◗ alpha parameter	◗ errors ◗ changes of errors	◗ alpha
13	Bidar and Kanan [73]	2013	MFA	◗ alpha (controls exploration) ◗ gamma (controls exploitation)	◗ count ◗ delta	◗ alpha ◗ gamma
14	Peraza et al. [74]	2016	HSO	◗ harmony memory accepting parameter ◗ pitch adjustment parameter	◗ iteration	◗ harmony memory accepting ◗ pitch adjusting rate
15	Valdez et al. [75]	2020	HSO	◗ harmony memory accepting parameter	◗ iteration	◗ harmony memory accepting
16	Ochoa et al. [76]	2017	DE	◗ mutation parameter	◗ number of generations	◗ mutation
17	Valenzuela et al. [77]	2017	FPA	◗ pollination probability	◗ iteration	◗ global pollination probability
18	Bernal et al. [78]	2017	ICA	◗ β and ζ parameters	◗ decades	◗ β parameter ◗ ζ parameter
19	Castillo and Amador-Angulo [79]	2018	BCO	◗ alpha ◗ beta	◗ iteration ◗ diversity	◗ alpha ◗ beta
20	Saffari et al. [80]	2022	ChOA	◗ parameters (random variables) a and c	◗ the number of repetitions ◗ parameter f	◗ parameter a ◗ parameter c
21	Melin et al. [81]	2022	BSA	◗ cognitive (c1) and the social (c2) acceleration coefficients	◗ iteration	◗ parameter c1 ◗ parameter c2

adaptive variant of the Emperor Penguin Optimizer to solve global optimization problems.

III. OVERVIEW OF THE ORIGINAL EPO ALGORITHM

In the aquatic world, penguins are recognized as flightless seabirds. When compared to other penguin species, the Emperor Penguin is the tallest and heaviest member of the penguin family [82]. They reside on Antarctic ice, where they spend the entire winter on open ice, and even breed during this difficult season. While the Antarctic is experiencing a hard winter, they cluster together to conserve heat, demonstrating their collectiveness and solidarity through their social behavior [83]. They hunt and forage in groups for the sake of survival. EPO's mathematical model is primarily concerned with identifying efficient movers, updating the position of emperor penguins, and the temperature around the huddle. The primary reason for creating a huddle by emperor penguins is to enhance the ambient temperature and preserve energy within the huddle. The huddle border structure is recognized as a L-shaped polygonal plane. Therefore, the huddle temperature T is dependent on the radius of the huddle polygon R. This relation is represented using Eq. (1) as follows:

$$T = \begin{cases} 0, & \text{if } R > 0.5 \\ 1, & \text{if } R < 0.5 \end{cases} \tag{1}$$

The temperature profile T' is an important factor to take into consideration that has impact on both the exploration and exploitation processes. T' is computed using

TABLE 2. Related works on choosing the most influential operators by employing FIS in meta-heuristics.

SI	Author [Reference]	Year Published	Standard Meta-heuristic	Influential Operators	Fuzzy Inputs	Fuzzy Outputs
1	Cheng and Prayogo [82]	2018	TLBO	◗ teacher phase operator ◗ student phase operator	◗ success rate	◗ probability utilizing bar movement
2	Zamli et al. [49]	2017	TLBO	◗ teacher phase operator ◗ student phase operator	◗ quality measure ◗ intensification measure ◗ diversification measure	◗ selection operator
3	Zamli et al. [4]	2018	PSO	◗ global search operation ◗ local search operation	◗ normalized current fitness ◗ Euclidean distance between the current and local best particle ◗ Euclidean distance between the current and global best particle	◗ inertial weight

Eq. (2) as follows:

$$T' = \left(T - \frac{M_{itr}}{x - M_{itr}}\right), \quad 0 \le x < M_{itr} \tag{2}$$

where

x - number of iterations currently in progress and also used for controlling main loop.
M_{itr} - number of iterations that may be taken as a maximum.

The distance between the emperor penguins and the current ideal solution is calculated when the huddle boundary is generated. On iteration x, the other agents will update their positions using Eq. (3) as follows:

$$\overrightarrow{D_{ep}} = \left|S(\vec{A}) \cdot \overrightarrow{P_{ep}(x)} - \vec{C} \cdot \overrightarrow{P(x)}\right| \tag{3}$$

where

$\overrightarrow{D_{ep}}$ - distance of emperor penguin to best fit emperor penguin.
$\overrightarrow{P}$ - position vector of the current emperor penguin.
$\overrightarrow{P_{ep}}$ - position vector of the fittest emperor penguin (i.e., the best optimal solution).
$S(\vec{A})$ - social forces to identify the best mover (i.e., optimal search agent).

The parameters $(\vec{A})$ and $(\vec{C})$ are used to avoid collisions between surrounding emperor penguins, and they are computed as follows:

$$\vec{A} = M \times \left(T' + P_{grid}(ac)\right) \times Rand() - T' \tag{4}$$

$$P_{grid}(ac) = \left|\vec{P} - \overrightarrow{P_{ep}}\right| \tag{5}$$

$$\vec{C} = Rand() \tag{6}$$

where

M - movement parameter used for collision avoidance, and the value is set to 2.
$P_{grid}(ac)$ - accuracy of polygon grid.
$Rand()$ - random function in the range of [0,1].

Using Eq. (7), we can calculate $S(\vec{A})$, which is responsible for moving the search agent in the direction of the best optimal search agent. Eq. (8), on the other hand, is responsible for updating their position.

$$S(\vec{A}) = \left(\sqrt{f \cdot e^{-\frac{x}{l}} - e^{-x}}\right)^2 \tag{7}$$

$$\vec{P}(x+1) = \overrightarrow{P_{ep}(x)} - \vec{A} \cdot \overrightarrow{D_{ep}} \tag{8}$$

where

e - signifies the expression function.
f and l - the control parameters to maintain the effective exploration and exploitation processes, and their values are in the range of [2,3] and [1.5,2], respectively.
$\vec{P}(x+1)$ - the next updated position of the emperor penguin.

The pseudocode showing the step-by-step procedure for implementing the basic EPO algorithm is shown in Figure 1. EPO begins by defining the initial values of the parameters: $T', \vec{A}, \vec{C}, S(\vec{A}), x, n, D, R, M_{itr}$ and $\vec{P}$, where n and D represent the number of population and dimension of the search space, respectively. Next, EPO generates the initial population of emperor penguins. The main loop begins by calculating the initial fitness of all emperor penguins to find the best emperor penguin. The positions of the other penguins are updated with respect to the position of the best emperor penguin using Eq. (8). While the position of all emperor penguins is updated, the fitness of each penguin is re-evaluated. The current best solution is updated and compared to the next best solution in the next iteration if a better solution was found in the previous step. The best emperor penguin is returned by EPO once the termination criteria are met. For more information, the reader is referred to EPO's original publication in [6].

IV. THE PROPOSED FUZZY ADAPTIVE EPO (FAEPO)

In line with the context of Section II, the performance of meta-heuristic algorithms can be enhanced successfully by adopting fuzzy system in them, which make them adaptive. This inspired us to make the EPO adaptive with the help of fuzzy logic. A detailed description of our proposed fuzzy adaptive variant of EPO is provided in this section. This section is divided into two subsections. The first subsection

Input: the emperor penguin's population $\vec{P}$
Output: the best-obtained search agent $\overrightarrow{P_{ep}}$
Procedure EPO
Initialize the parameters $T', \overrightarrow{A}, \overrightarrow{C}, S(\vec{A}), x, n, D, R, M_{itr}$ *and* $\vec{P}$
while $(x < M_{itr})$ **do**
  FITNESS($\vec{P}(x)$) /* Compute the fitness of each search agent using **FITNESS** function */
  Generate random number R in range [0, 1]
  Compute the temperature profile T' around the huddle using Eq. (2)
  for $i \leftarrow 1$ to n **do**
    for $j \leftarrow 1$ to D **do**
      Compute the vectors $\overrightarrow{A}$ *and* $\overrightarrow{C}$ using Eq. (4) and (6)
      Compute the function $S(\vec{A})$ using Eq. (7)
      Update the position of the current agent using Eq. (8)
    end for
  end for
  Update parameters $T', \overrightarrow{A}, \overrightarrow{C}, S(\vec{A})$
  Amend search agent which goes beyond the region of search space
  FITNESS($\vec{P}(x)$) /* compute the fitness value of updated search agents */
  Update $\overrightarrow{P_{ep}}$ if there is a better solution than previous optimal solution, i.e., (*FITbest*)
  $x \leftarrow x + 1$
end while
return $\overrightarrow{P_{ep}}$
End procedure

FIGURE 1. Pseudocode of original EPO algorithm.

demonstrates the methodology for parameter adaption using fuzzy system. The second subsection illustrates the pseudocode of the proposed FAEPO.

A. METHODOLOGY FOR PARAMETER ADAPTATION

Dynamic parameter adaption utilizing fuzzy system increases the quality of the outcome acquired by conducting a more efficient local and global search than the original EPO approach. Figure 2 illustrates the Mamdani fuzzy inference system [84], [85], which is an integral part of the proposed FAEPO. The metric used in the fuzzy system as input are quality, success rate, and diversity. On the other hand, the fuzzy system outputs are the dynamic adjustment of f and l parameters that control the exploration and exploitation of the search space stated in Eq. (7). The quality measure (Q_M) metric represents the quality of the current potential solution (i.e., FIT[current]) in the normalized form. The Q_M is formulated in Eq. (9) as follows:

$$Q_M = \left| \frac{Fit_{current} - Fit_{best}(\overrightarrow{P})}{Fit_{best}(\overrightarrow{P}) - Fit_{worst}(\overrightarrow{P})} \right| \times 100 \qquad (9)$$

The success measure (S_M) metric represents the average improved tries in the normalized form, where N denotes the population size. The S_M is formulated in Eq. (10) as follows:

$$S_M = \left[\frac{No.\ of\ Improved\ Tries}{N} \right] \times 100 \qquad (10)$$

The diversity measure (D_M) metric is the mean square average of the current fitness compared to the average fitness, which represents the diversity of the current fitness compared to the rest of the population. The D_M is formulated in Eq. (11) as follows:

$$D_M = \left[\frac{\sqrt{\sum_{k=1}^{N} \{Fit_k - Fit_{avg}(\overrightarrow{P})\}^2}}{N} \right] \times 100 \qquad (11)$$

1) FUZZIFICATION

The fuzzification refers to the transition from the real domain to the fuzzy domain. The fuzzification process employs three predefined trapezoidal membership functions, as illustrated in Figure 3. Each membership function utilizes three linguistic terms: high, medium, and low. Each linguistic term may be absolute or partial, depending on its value range. Table 3 summarizes each membership function and the ranges of their corresponding linguistic terms.

2) RULES EVALUATION

A rule base is required after fuzzifying the input parameters. This work combines the inputs with Mamdani Fuzzy Conjunction (AND) fuzzy rules. Table 4 summarizes the six fuzzy rules used for inference evaluation based on Figure 2. It can be observed that a variety of scenarios for the bounds of the fuzzy membership functions can be generated by varying the parameters. In the present study, the bounds and definitions were selected based on the detailed and comprehensive

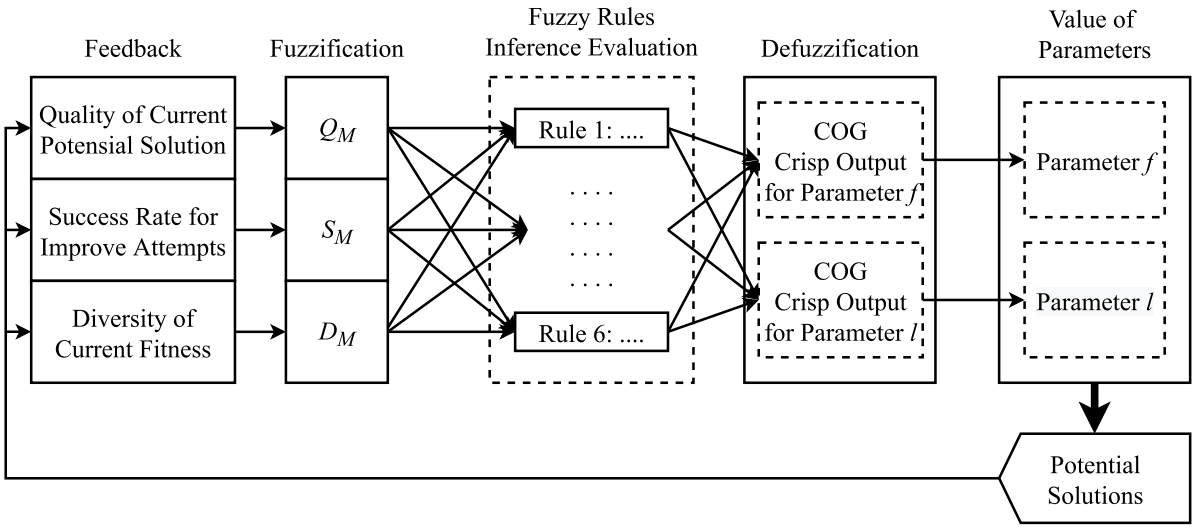


FIGURE 2. Fuzzy inference system for FAEPO.

TABLE 3. Value range of all linguistic terms of each input membership function.

Membership Function	Range of Normalized Value	Linguistic Terms	Status
Q_M	0 - 60	High	Absolute
	60 - 70	High & Medium	Partial
	70 - 80	Medium	Absolute
	80 - 90	Medium & Low	Partial
	90 - 100	Low	Absolute
S_M	0 - 20	Low	Absolute
	20 - 30	Low & Medium	Partial
	30 - 40	Medium	Absolute
	40 - 50	Medium & High	Partial
	50 - 100	High	Absolute
D_M	0 - 30	Low	Absolute
	30 - 40	Low & Medium	Partial
	40 - 60	Medium	Absolute
	60 - 70	Medium & High	Partial
	70 - 100	High	Absolute

TABLE 4. The set of fuzzy rules used in Mamdani FIS for FAEPO.

Rule No.	Fuzzy Inputs			Fuzzy Outputs	
	Q_M	S_M	D_M	f	l
$R1$	Low	×	×	f_1	l_1
$R2$	Low	Medium	×	f_2	l_2
$R3$	Medium	Medium	Low	f_3	l_3
$R4$	Medium	Low	Medium	f_4	l_4
$R5$	High	Low	High	f_5	l_5
$R6$	High	×	×	f_6	l_6

× = Don't Care Condition

sensitivity analyses of the EPO [6] algorithm (as the ancestor of the proposed FAEPO).

3) DEFUZZIFICATION

In this study, defuzzification is achieved using two crisp outputs for the f and l parameters. For each parameter, there are

TABLE 5. Value range of all linguistic terms of f and l parameters.

Parameter f			Parameter l		
Value Range	Linguistic Terms	Status	Value Range	Linguistic Terms	Status
2.0 - 2.1	f_1	Absolute	1.50 - 1.55	l_1	Absolute
2.0 - 2.2	f_1 and f_2	Partial	1.50 - 1.60	l_1 and l_2	Partial
2.1 - 2.3	f_2	Absolute	1.55 - 1.65	l_2	Absolute
2.2 - 2.4	f_2 and f_3	Partial	1.60 - 1.70	l_2 and l_3	Partial
2.3 - 2.5	f_3	Absolute	1.65 - 1.75	l_3	Absolute
2.4 - 2.6	f_3 and f_4	Partial	1.70 - 1.80	l_3 and l_4	Partial
2.5 - 2.7	f_4	Absolute	1.75 - 1.85	l_4	Absolute
2.6 - 2.8	f_4 and f_5	Partial	1.80 - 1.90	l_4 and l_5	Partial
2.7 - 2.9	f_5	Absolute	1.85 - 1.95	l_5	Absolute
2.8 - 3.0	f_5 and f_6	Partial	1.90 – 2.00	l_5 and l_6	Partial
2.9 - 3.0	f_6	Absolute	1.95 – 2.00	l_6	Absolute

six different linguistic terms, where f_1, f_2, f_3, f_4, f_5, and f_6 are linguistic terms belonging to the f parameter, and l_1, l_2, l_3, l_4, l_5, and l_6 are linguistic terms belonging to the l parameter, symbolized by the trapezoidal membership functions shown in Figure 3. The ranges of f and l, discussed in the previous section, are divided into six equal sections. A variety of defuzzification methods, such as Mean of Max, Centroid, and Center of gravity, are available to obtain crisp values for these parameters. Defuzzification has been accomplished using the Center of gravity method in this paper. Table 5 contains the values of all linguistic terms for the parameters f and l.

B. PROPOSED FAEPO IMPLEMENTATION

Figure 5 shows the step-by-step process for implementing the proposed general Fuzzy Adaptive EPO (FAEPO) algorithm. The first step in the FAEPO algorithm is to define the membership functions for the linguistic variables, and the second step is to define fuzzy rules (see lines 1 and 2). After that, the algorithm sets the required parameters: T', $\vec{A}$, $\vec{C}$, $S(\vec{A})$, n, D, M_{itr} and $\vec{P}$, where n and D represent the number of population and dimension of the

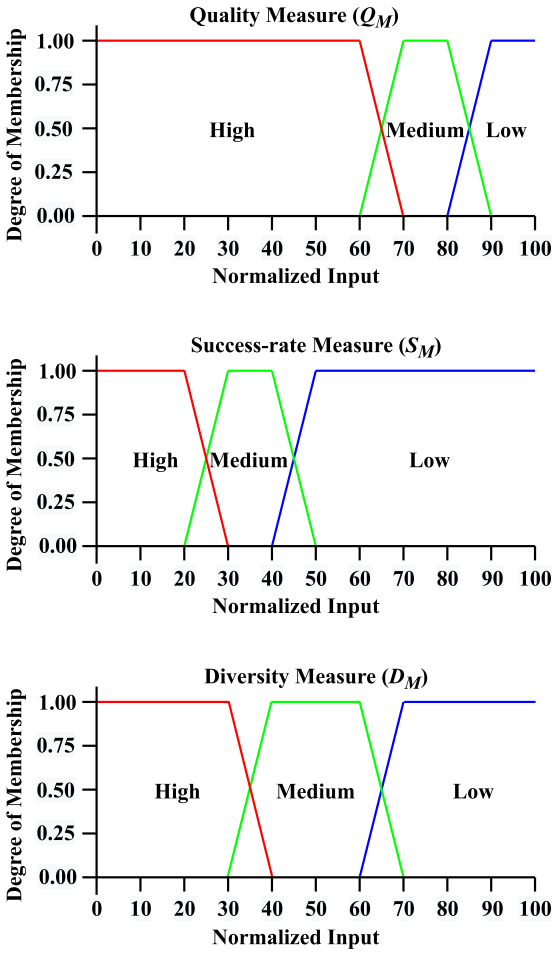


FIGURE 3. Fuzzy input membership functions.

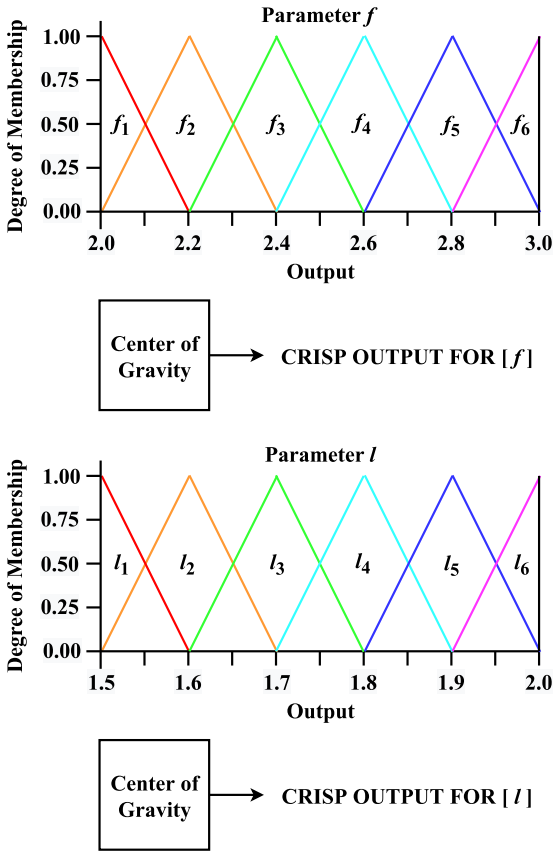


FIGURE 4. Fuzzy output membership functions.

search space, respectively. Prior to entering the main loop, the algorithm utilizes the *FITNESS* function to determine the fitness of each search agent (see line 4). Let's now look at the original EPO algorithm in Figure 1. The initial evaluation of the fitness function (see Figure 1, line 3) is considered redundant because at the end of each iteration, the fitness of all search agents (see Figure 1, line 15) are recomputed to see which one was the best at the end of that iteration. In the proposed FAEPO, this redundant computation, which significantly impacts the time complexity, is eliminated by factorizing it before the main loop. This leads to a reduction in the time complexity of FAEPO. The main loop in Figure 5 begins by computing the temperature profile of the huddle (see line 7). Then the function $S(\vec{A})$ is computed (see line 8). The position of the current search agent is updated within the nested for loops based on the current values of vectors $(\vec{A})$ and $(\vec{C})$ (see lines 11 and 12). After completing the nested loop, the algorithm computes the three measures (quality, success rate, and diversity) (see line 15), fuzzifies the crisp inputs using the predefined membership functions (see line 16), evaluates the fuzzified values based on the fuzzy rules given in Table 4 (see line 17), and finally defuzzifies the crisp outputs for parameters f and l using the predefined membership functions (see line 18). Then the algorithm modifies the search agents that are outside the region of the search space. At the end of the main loop, the algorithm calculates the fitness of the updated search agents and updates the best search agent if a better solution is found (see lines 20 and 21). The main loop is executed until the condition of the maximum number of iterations is met. Finally, FAEPO returns the search agent that has achieved the best results.

V. EMPIRICAL EVALUATION

We conducted a comprehensive evaluation of the proposed algorithm. The evaluation experiments had three distinct goals: (1) to investigate how FAEPO compares to its predecessor EPO; (2) to compare FAEPO to other contemporary meta-heuristic algorithms for global optimization problems; (3) to compare the performance of FAEPO to the benchmark algorithms compared in terms of the Friedman Mean Rank Test. Consistent with the above goals, we focus on answering the following research questions:

RQ1: To what extent does the use of FAEPO improve EPO performance?

RQ2: How are the convergence efficiency and statistical significance of FAEPO compared to other meta-heuristic algorithms?

Input: the emperor penguin's population $\vec{P}$
Output: the best-obtained search agent $\overrightarrow{P_{ep}}$
Procedure FAEPO
Define the membership functions for the linguistic variables
Define the fuzzy rules
Initialize the parameters $T', \overrightarrow{A}, \overrightarrow{C}, S(\vec{A}), x, n, D, R, M_{itr}$ *and* $\vec{P}$
FITNESS($\vec{P}(x)$) /* Compute the fitness of each search agent using **FITNESS** function */
while ($x < M_{itr}$) **do**
  Generate random number R in range [0, 1]
  Compute the temperature profile T' around the huddle using Eq. (2)
  Compute the function $S(\vec{A})$ using Eq. (7)
  for $i \leftarrow 1$ to n **do**
    for $j \leftarrow 1$ to D **do**
      Compute the vectors $\overrightarrow{A}$ *and* $\overrightarrow{C}$ using Eq. (4) and (6)
      Update the position of the current agent using Eq. (8)
    end for
  end for
  Compute Q_M, S_M, and D_M using Eq. (9), (10) and (11), respectively
  Fuzzify inputs Q_M, S_M, and D_M based on the predefined membership functions
  Evaluation of fuzzy rules
  Defuzzify the outputs to produce crisp output for l and f parameters
  Amend search agent which goes beyond the region of search space
  FITNESS($\vec{P}(x)$) /* compute the fitness value of updated search agents */
  Update $\overrightarrow{P_{ep}}$ if there is a better solution than previous optimal solution, i.e., (*FITbest*)
  $x \leftarrow x + 1$
end while
return $\overrightarrow{P_{ep}}$
End procedure

FIGURE 5. Proposed general fuzzy adaptive EPO (FAEPO).

RQ3: How are the performances of FAEPO compared to EPO, IEPO, and other competing meta-heuristic algorithms?
RQ4: Is there any overhead in terms of the time performance of FAEPO implementation?
RQ5: Is FAEPO sufficiently general to handle both minimization and maximization optimization problems?

A. EXPERIMENTAL RUNNING ENVIRONMENT

The following are the specifications of the personal computer that we have utilized in our experiment. RAM is 8.00 GB, and the hard disc (HDD) storage capacity is 500 GB. The processor is an Intel Core i7-6650U running at 2.20 GHz. Windows 10 Pro is the operating system being used, whereas the type of system is a 64-bit operating system and processor based on the x64 architecture. In order to carry out our experiments, we have used the MATLAB 2020b simulation platform. Each experiment is carried out a total of thirty times to establish statistical significance.

B. COMPETING META-HEURISTIC ALGORITHMS

The performance of the proposed FAEPO is compared with seven state-of-the-art meta-heuristic algorithms, including its predecessor EPO and an improved variant of EPO (IEPO). The reason for selecting these algorithms is that they are swarm intelligence-based meta-heuristic algorithms. A brief description of each algorithm (except EPO) can be found below:

- Moth-flame Optimization Algorithm (MFO): MFO is a nature-inspired optimization algorithm presented by Seyedali Mirjalili in 2015 [86]. MFO mimics the transverse orientation, a type of navigation strategy, of moths in nature. The transverse orientation represents the spiral convergence towards the artificial lights eventually. The moths only adjust their position concerning the best flame only during final iterations. In this way, exploration and exploitation of the search space are balanced by gradually reducing the number of flames.
- Salp Swarm Algorithm (SSA): Mirjalili et al. developed the SSA in 2017 [9]. SSA is a swarm intelligence and population-based algorithm that mimics the behavior and social interaction of salp swarms. Salps inhabit the deep oceans and move in search of food in swarms called salp chains. Salp chains can be mathematically divided into two types: the head salp is the leader, while the others are followers. According to Newton's law of motion, the followers update their positions. This algorithm has been used to solve design problems in the field of engineering.
- Sooty Tern Optimization Algorithm (STOA): Dhiman and Kaur [16] proposed the STOA, a population-based meta-heuristic algorithm. The main objective of STOA is to mimic the natural migration and attack behavior of

the sooty tern seabird. The exploration and exploitation of the search space are represented by the migration and attack behavior, respectively. This algorithm has been used to solve industrial engineering problems.

- Genetic Algorithm (GA): The Genetic Algorithm (GA) [87] is an evolutionary algorithm inspired by natural selection. GA develops three operators: selection, crossover, and mutation. These operators are widely used to find near-optimal solutions. Over generations, the algorithm improves on previous solutions until the termination criterion is met.
- Particle Swarm Optimization (PSO): Particle Swarm Optimization (PSO) [88] is another population-based stochastic optimization algorithm inspired by the social behavior of schools of fish or flocks of birds. Each particle moves in the search space looking for the global best solution and may update its current position if it is better than the previous best solution. The popularity of this algorithm is due to the fact that it has a few adjustable parameters.
- Improved Emperor Penguin Optimizer (IEPO): IEPO is an improved variant of EPO proposed by Tang et al. in 2020 [89]. This improved variant is designed and coupled with eQuest simulation tool for the minimization of energy consumption of residential buildings. The authors introduced two modification in IEPO. First, the incidental parameter shown in Eq. (6) is altered into a balanced equation by the singer process. Second, an incidental walk method (Lévy flight) is applied in this scenario so that proper control of the local search can be maintained.

C. PARAMETER SETTINGS FOR THE COMPETING META-HEURISTIC ALGORITHMS

The main general settings for all algorithms (i.e., population size, maximum iteration, and maximum fitness evaluation) are defined in Table 6. To ensure fairness, the termination criteria for all algorithms is based on the maximum fitness evaluation.

The specific parameter settings for the participating algorithms can be found in Table 7. Here, the best parameter settings are adopted from their original papers.

D. OVERVIEW OF THE CASE STUDY OBJECTS

Accordingly, we design our experimental evaluation to comprehensively focus on four separate case studies to thoroughly assess the proposed FAEPO performance. The overview of the case studies and design of their fitness functions are described in the following subsections.

1) CLASSICAL OPTIMIZATION BENCHMARK TEST FUNCTIONS

Many studies have been conducted to improve existing strategies or propose new algorithms. The use of test problems for benchmarking purposes is one of the common features of these studies. Most of the test problems are mathematical functions designed to mimic the complexity of real search spaces. In the following, twelve benchmark test functions are briefly described with their constraints.

Matyas: This function is a continuous, differentiable, non-separable, non-scalable, and unimodal benchmark test function defined as follows:

$$F_1(z) = 0.26(z_1^2 + z_2^2) - 0.48 z_1 z_2 \tag{12}$$

subject to $-10 \leq z_i \leq 10$. The global minimum is located at $z^* = F_1(0, 0), F_1(z^*) = 0$.

Schaffer N.4: This function is a continuous, differentiable, non-separable, non-scalable, and unimodal benchmark test function defined as follows:

$$F_2(z) = 0.5 + \frac{\cos\left(\sin\left(|z_1^2 - z_2^2|\right)\right) - 0.5}{\left[1 + 0.001\left(z_1^2 + z_2^2\right)\right]^2} \tag{13}$$

subject to $-100 \leq z_i \leq 100$. The global minimum is located at $z^* = F_2(0, 1.25313), F_2(z^*) = 0.292579$.

Powell Sum: This function is a continuous, differentiable, separable, scalable, and unimodal benchmark test function defined as follows:

$$F_3(z) = \sum_{i=1}^{D} |z_i|^{i+1} \tag{14}$$

subject to $-1 \leq z_i \leq 1$. The global minimum is $F_3(z^*) = 0$.

Schwefel's Problem 2.22: This function is a continuous, differentiable, non-separable, scalable, and unimodal benchmark test function defined as follows:

$$F_4(z) = \sum_{i=1}^{D} |z_i| + \prod_{i=1}^{n} |z_i| \tag{15}$$

subject to $-10 \leq z_i \leq 10$. The global minimum is located at $z^* = F_4(0, \ldots, 0), F_4(z^*) = 0$.

Brown: This function is a continuous, differentiable, non-separable, scalable, and unimodal benchmark test function defined as follows:

$$F_5(z) = \sum_{i=1}^{n-1} (z_i^2)^{(z_{i+1}^2+1)} + (z_{i+1}^2)^{(z_{i+1}^2)} \tag{16}$$

subject to $-1 \leq z_i \leq 4$. The global minimum is located at $z^* = F_5(0, \ldots, 0), F_5(z^*) = 0$.

Xin-She Yang N.3: This function is a non-separable unimodal benchmark test function defined as follows:

$$F_6(z) = \left[e^{-\sum_{i=1}^{D} \left(\frac{z_i}{\beta}\right)^{2m}} - 2e^{-\sum_{i=1}^{D} (z_i)^2} \prod_{i=1}^{D} \cos^2(z_i) \right] \tag{17}$$

subject to $-20 \leq z_i \leq 20$. The global minimum is located at $z^* = F_6(0, \ldots, 0), F_6(z^*) = -1$ for $m = 5$ and $\beta = 15$.

Egg Crate: This function is a continuous, separable, and non-scalable benchmark test function defined as follows:

$$F_7(z) = z_1^2 + z_2^2 + 25\left\{sin^2(z_1) sin^2(z_2)\right\} \tag{18}$$

subject to $-5 \leq z_i \leq 5$. The global minimum is located at $z^* = F_7(0, 0), F_7(z^*) = 0$.

TABLE 6. Common parameter settings used in different experiments.

S.N.	Name of the Optimization Problem	Population Size	Maximum Iteration	Maximum Fitness Evaluation
1	Classical optimization benchmark test functions	10	700	7000
2	Team formation optimization problem	10	100	1000
3	Low autocorrelation binary sequence (LABS) problem	10	1000	10000
4	MC/DC test case generation problem	10	1000	10000

TABLE 7. Parameter configurations for the competing meta-heuristic algorithms.

S.N.	Name of the Algorithm	Parameters	Values
1	Moth-flame Optimization Algorithm (MFO)	Awareness Probability (AP)	0.1
		Flight Length f	Total Experts
		Random Number r_1 and r_2	[0, 1]
2	Salp Swarm Algorithm (SSA)	Random Number c_1, c_2, c_3	[0, 1]
		Number of Generations	1000
		Leader Position Update Probability	0.5
3	Sooty Tern Optimization Algorithm (STOA)	Controlling Variable (C_f)	[2, 0]
		Random Variable (C_B)	[0, 0.5]
		Constants u and v	1
		Variable k	$[0, 2\pi]$
4	Particle Swarm Optimizer (PSO)	Inertia Coefficient w	Linearly decreases from 0.99
		Personal Acceleration Coefficient c_1	2
		Social Acceleration Coefficient c_2	2
5	Genetic Algorithm (GA)	Selection	Roulette wheel
		Crossover Probability	0.1
		Mutation Probability	0.02
6	Emperor Penguin Algorithm (EPO)	Parameter M	2
		Parameter f	[2, 3]
		Parameter l	[1.5, 2]
7	Improved Emperor Penguin Algorithm (IEPO)	Parameter M	2
		Parameter f	[2, 3]
		Parameter l	[1.5, 2]
8	Fuzzy Adaptive Emperor Penguin Algorithm (FAEPO)	Parameter M	2
		Parameter f	Fuzzyfied
		Parameter l	Fuzzyfied

Crowned Cross: This function is a multimodal benchmark test function and the negative form of the cross in tray function. It is a complex function to optimize and defined as follows:

$$F_8(z) = 10^{-5}\left(\left|sin(z_1)sin(z_2)e^{\left|100-\frac{(z_1^2+z_2^2)}{\pi}\right|}\right| + 1\right)^{0.1} \quad (19)$$

subject to $-10 \leq z_i \leq 10,\ i = 1, 2$. The global minimum is $F_8(z^*) = 0.0001$.

Rastrigin: This function is a non-convex, non-linear multimodal function defined as follows:

$$F_9(z) = AD + \sum_{i=1}^{D}\left[z_i^2 - A\cos(2\pi z_i)\right] \quad (20)$$

subject to $-5.12 \leq z_i \leq 5.12,\ A = 10$. The global minimum is located at $z^* = F_9(0, \ldots, 0), F_9(z^*) = 0$.

Xin-She Yang N.1: This is a separable, generic stochastic, and non-smooth benchmark test function proposed in [90] and defined as follows:

$$F_{10}(z) = \sum_{i=1}^{D}\epsilon_i |z_i|^i \quad (21)$$

subject to $-5 \leq z_i \leq 5$. The variable $\epsilon_i, i = 1, 2, \ldots, D$ is a uniformly distributed random variable in [0, 1]. The global minimum is located at $z^* = F_10(0, \ldots, 0), F_10(z^*) = 0$.

Xin-She Yang N.2: This is a non-separable, multimodal benchmark test function defined as follows:

$$F_{11}(z) = \frac{\sum_{i=1}^{D}|z_i|}{e^{\sum_{i=1}^{D}\sin(z_i^2)}} \quad (22)$$

subject to $-2\pi \leq z_i \leq 2\pi$. The global minimum is located at $z^* = F_11(0, \ldots, 0), F_11(z^*) = 0$.

Quartic Noise: This function is a continuous, differentiable, separable, and scalable benchmark test function.

$$F_{12}(z) = \sum_{i=1}^{D} iz_i^4 + random[0, 1) \quad (23)$$

subject to $-1.28 \leq z_i \leq 1.28$. The global minimum is located at $z^* = F_12(0, \ldots, 0), F_12(z^*) = 0$.

TABLE 8. Interconnected experts with skills and interaction cost.

Experts	Skills	Interconnection Cost ($\complement$)					
		e_1	e_2	e_3	e_4	e_5	e_6
e_1	s_1, s_2, s_3	0.00	0.00	0.50	1.00	1.00	0.75
e_2	s_1, s_2, s_3	0.00	0.00	0.50	1.00	1.00	0.75
e_3	s_1, s_2, s_4	0.50	0.50	0.00	1.00	1.00	1.00
e_4	s_5, s_6, s_7	1.00	1.00	1.00	0.00	1.00	0.75
e_5	s_8, s_9	1.00	1.00	1.00	1.00	0.00	1.00
e_6	s_3, s_7	0.75	0.75	1.00	0.75	1.00	0.00

Eq. (12) to Eq. (23) are used as the fitness function for the optimization of respective benchmark test function. A summary of the selected benchmark test functions with their constraints is shown in Table 9.

2) TEAM FORMATION OPTIMIZATION PROBLEM

The team formation optimization (TFO) problem is about identifying experts and forming the best group or team with the required skills for specific tasks. TFO is designed to create such a team. Consider an example with six experts, represented by $E = \{e_1, e_2, e_3, e_4, e_5, e_6\}$ with a total of nine unique skills, represented by $S^U = \{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9\}$. The experts with their complementary skills and interconnection costs are listed in Table 8. The interconnection cost, $\complement_{ij}$ ($i = 1, 2, \ldots, 6$ and $j = 1, 2, \ldots, 6$) of all experts with respect to their skills can be calculated using Eq. (24) where S^{e_i} represents the skills of the i^{th} expert. Eq. (24) provides the values between [0,1]. $\complement_{ij} = 0$ means that experts e_i and e_j have no connection to each other, or both experts have the same skills. On the other hand, $\complement_{ij} = 1$ means that both experts have no common skills

$$\complement_{ij} = 1 - \frac{S^{e_i} \cap S^{e_j}}{S^{e_i} \cup S^{e_j}} \tag{24}$$

Take for example, a project P which requires the skills $S^R = \{s_3^R, s_6^R, s_7^R\}$. In accordance with Table 8, 4 experts (i.e., e_1, e_2, e_4, and e_6) possess one or more required skills where s_3^R, s_6^R,and s_7^R possess by $\{e_1, e_2, e_6\}$, $\{e_4\}$, and $\{e_4, e_6\}$, respectively. Only four expert combinations are qualified as potential teams out of the entire pool of possible combinations. The potential teams are: $T_1 = \{e_1, e_2, e_4, e_6\}$, $T_2 = \{e_2, e_4, e_6\}$, $T_3 = \{e_1, e_4, e_6\}$, and $T_4 = \{e_4, e_6\}$. Using Eq. (25), their costs can be calculated as: $T_1^{\complement} = 4.25$, $T_2^{\complement} = 2.5$, $T_3^{\complement} = 2.5$, and $T_4^{\complement} = 0.75$. Moreover, the size of the potential teams are $T_1^s = 4$, $T_2^s = 3$, $T_3^s = 3$, and $T_4^s = 2$. Since T_4 scores the lowest $\complement^T$ and the smallest team size, it is selected as the optimum team or T^*.

$$f(T^*) = \arg\min_{\complement^T} \sum_{i=1}^{i=n-1} \sum_{j=i+1}^{j=n} \complement_{ij} \tag{25}$$

This is a simple example. However, in practical applications, there would be a large number of e_i with a massive set of S, which will generate a huge number of teams. A linear solution to the problem of determining T^* is impractical.

From the above definition, we can formulate our fitness function using Eq. (25) for solving TFO problem, where, $\complement^T$ represents the interconnection cost of the team.

3) LOW AUTOCORRELATION BINARY SEQUENCE (LABS) PROBLEM

Finding a low autocorrelation binary sequence (LABS) is a hard combinatorial optimization problem. However, LABS problems have many applications in diverse areas of signal and information processing. Therefore, much research has been published on these sequences to find low autocorrelation because of their practical importance. Moreover, various methods for constructing low-autocorrelation sequences have been devised. To illustrate the problem, consider the representation of a binary sequence S^b of length N as $s_1 s_2 \ldots s_N$ with $s_i \epsilon \{-1, 1\}$; for $1 \leq i \leq N$, i.e., $S \epsilon \{-1, 1\}^N$. Suppose the sequence S^b has a periodic autocorrelation C with a distance $k (= 0, 1, 2, \ldots, N-1)$, and the formula for this correlation can be defined as follows.

$$C_k(S^b) = \sum_{i=1}^{N-k} s_i s_{i+k} \tag{26}$$

The energy function associated with the sequence S^b is the sum of the squares of its correlations:

$$E(S^b) = \sum_{k=1}^{N-1} C_k^2(S^b) \tag{27}$$

The LABS problem is to find a sequence S^b of given length N that minimizes the $E(S^b)$. Golay [91] devised a new measure known as the merit factor to evaluate the quality of sequences as follows:

$$F(S^b) = \frac{N^2}{2E(S^b)} \tag{28}$$

If we define F_N to be the optimal value of the merit factor for sequences of length N, the LABS(N) problem can be alternatively described as finding F_N such that:

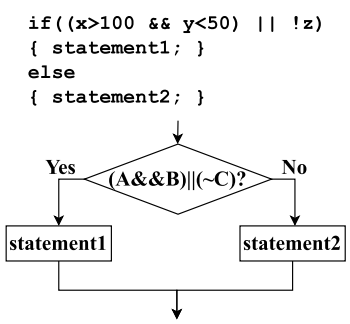
$$F_N = \max_{S^b \epsilon \{-1,1\}^N} F(S^b) \tag{29}$$

The only method for determining the sequence with the optimal F_N is to perform an implicit enumerative search over all 2^L possible sequences.

For LABS problem, Eq. (29) is considered as the fitness function for finding the optimal binary sequence with low autocorrelation.

4) MC/DC TEST CASE GENERATION PROBLEM

NASA promotes the MC/DC as a structural testing coverage criterion for safety-critical software systems. Exists in pairs, MC/DC criteria [92] insist that each variable could influence the overall outcome independently while maintaining the value of the other variable(s). Each pair differs solely in terms of the Boolean value of a single condition, but produces a different outcome for the decision statement. The MC/DC pairs



A	B	C	Output
F	F	F	T
F	F	T	F
F	T	F	T
F	T	T	F
T	F	F	T
T	F	T	F
T	T	F	T
T	T	T	T

FIGURE 6. Illustrative example [93].

for AND operation are $\{\{F,T\},\{T,T\}\},\{\{T,F\},\{T,T\}\}$ where the entry $\{T,T\}$ is redundant. To decrease the number of test cases, the whole MC/DC compatible test predicate can be simplified to $\{F,T\},\{T,F\}$ and $\{T,T\}$. In similar manner, the reduced MC/DC pairs for OR operation are $\{\{F,F\},\{T,F\}\},\{\{F,T\},\{T,F\}\}$. Manual test creation is viable for a specific range of predicates, but when dealing with a large number of predicates, it becomes time-consuming to perform the procedure manually. An illustrative example of *if* statement involving AND, OR, and NOT operations introduced by Haque et al. in [93] is reused and shown in Figure 6 to clear the readers about the MC/DC test data generation. Referring from the truth table shown in Figure 6, there are one pair, one pair, and three pairs for MC/DC coverage for variables A, B, and C, respectively. Three alternative MC/DC solutions (i.e., [0,1,3,5,7], [2,3,5,7], and [3,4,5,7]) are derived by combining the pairs except the repetition. The result is that any MC/DC findings may be transformed to appropriate test cases for all the predicates.

However, the MC/DC gives us a manageable number of tests to perform when the number of combinations is just too high. In order to achieve MC/DC, we will need to locate at least one test pair that satisfies the MC/DC criterion for each condition that is contained in the given boolean expression. If the number of conditions is m, then the total number of test cases will be 2^m. It is proved that with MC/DC, the minimum number of test cases will be $m+1$ for m number of conditions [94]. In this experiment, the fitness function counts the size of the generated test case set for each candidate. The minimum size represents the best fitness of the respective candidate, which can be close to or equal to $m+1$ for m number of conditions.

E. DATASETS USED IN DIFFERENT EXPERIMENTS

This section provides the details of datasets that are used in different experiments to evaluate the performance of the proposed FAEPO.

1) CLASSICAL OPTIMIZATION BENCHMARK TEST FUNCTION EXPERIMENT

In the research that has been published, benchmark test functions have been utilized in order to evaluate the performance of meta-heuristic algorithms. Real-world problems are best tackled with the help of algorithms that have proven successful when applied to a variety of numerical optimization challenges. Due to the absence of a standardized or generally accepted test bed, various researchers make their own unique selections of functions and configurations for their experiments. Because of this, it is hard for researchers to find functions that can accurately measure the robustness of a proposed meta-heuristic algorithm. However, we have selected twelve test functions listed in Table 9, where *Dimensions* denotes the function's dimension, *Range* represents the function's search space boundary, and $F(z^*)$ indicates the optimal value of the function. These functions are divided into two categories. The first six are unimodal ($F_1 - F_6$), and the last six are multimodal ($F_7 - F_{12}$). Unimodal functions have only one global optimum, whereas there are multiple local solutions exists in multimodal categories. Unimodal functions are well-suited for evaluating the exploitation performance of algorithms, whereas the exploration performance of algorithms is measured using multimodal functions. These functions contain numerous local optimums, which the algorithm should avoid. Both type of functions are further divided into two categories: fixed dimensional and variable dimensional. $F1 - F2$ and $F3 - F6$ are belongs to the unimodal fixed dimensional and unimodal variable dimensional test functions, respectively. Moreover, $F7 - F8$ and $F9 - F12$ are belongs to the multimodal fixed dimensional and multimodal variable dimensional test functions, respectively. The dimension of the fixed dimensional test function considered is 2, whereas three different higher dimensions (i.e., 50, 70, and 100) are considered for variable dimensional test functions in the experiment.

2) TEAM FORMATION OPTIMIZATION EXPERIMENT

Two benchmark datasets have been utilized in this experimental evaluation. The first dataset is the IMBD dataset [95], which consists of movie actors (1014 names of actors) and their roles (28 unique roles) by genre. This dataset is owned by Amazon, which is considerably denser than the other datasets and can be used to test the scalability of an algorithm that is currently being evaluated [96]. The dataset is composed of information gathered between the years 2000 and 2002, and the only actors who are regarded as experts are those who made an appearance in eight or more films during this time period. The ability of an actor to perform in a diverse range of genres is a good indicator of the breadth and depth of their acting skills. The second is the DBLP dataset [97], a bibliographic database of scientific publications (includes 5641 authors' information with 3887 unique skills). DBLP is considered to be one of the most widely used real-life benchmark dataset on social networks. This was derived from DBLP XML published in July 2017. The largest number of specialists in database theory, data mining, and artificial intelligence have contributed to the DBLP dataset. The people who are considered to be experts in this dataset are those who have more than one publication indexed on DBLP. The level of expertise of each expert is based on meaningful keywords

TABLE 9. Descriptions of the selected optimization benchmark test functions.

Benchmark Function Type	Function ID	Function Name	Dimensions (D)	Range	$F(z^*)$
Unimodal (Fixed Dimension)	F1	Matyas	2	[-10, 10]	0
	F2	Schaffer N. 4	2	[-100, 100]	0.292579
Unimodal (Variable Dimension)	F3	Powell Sum	50,70,100	[-1, 1]	0
	F4	Schwefel's 2.22	50,70,100	[-10, 10]	0
	F5	Brown	50,70,100	[-1, 4]	0
	F6	Xin-She Yang N.3	50,70,100	[-20, 20]	0
Multimodal (Fixed Dimension)	F7	Egg Crate	2	[-5, 5]	0
	F8	Crowned cross	2	[-10, 10]	0.0001
Multimodal (Variable Dimension)	F9	Rastrigin	50,70,100	[-5.12, 5.12]	0
	F10	Xin-She Yang N.1	50,70,100	[-5, 5]	0
	F11	Xin-She Yang N.2	50,70,100	[-2π, 2π]	0
	F12	Quartic Noise	50,70,100	[-1.28, 1.28]	0

taken from the title of the author's work. For our evaluation, we have adopted five sets of skills to find (i.e., 5, 10, 15, 20, and 25). The datasets are standardized in the same way that other datasets are adjusted so that a single algorithm can be evaluated using many datasets.

3) FINDING LABS EXPERIMENT

Finding binary sequences of low autocorrelation (LABS) is a notoriously hard combinatorial optimization problem [98]. There is no dilemma involved with LABS. Because of this, the solution is dictated by an objective function, and the goal is to acquire the objective function values that are the lowest possible. The value of the objective function, sometimes referred to as the solution quantity, of each potential candidate solution needs to be determined. Due to the fact that the objective function needs to be minimized, the LABS problem is a minimization problem. The minimum values of energy or the maximum values of merit factor for the binary sequences through exhaustive search are stated in [99]. We have used binary sequences of length $7 \leq N \leq 31$ in our experiment to evaluate the performance of the proposed algorithm to find the near-optimal merit factor of LABS.

4) MC/DC TEST CASE GENERATION EXPERIMENT

We opt to select a list of eight logical expressions/functions (LE1 to LE8) for MC/DC test case generation. Tests are performed on eight functions listed in Table 10. The classification of triangles is the task of the first function (i.e., LE1) to be examined through testing. This is a well-known problem that has been cited in a great deal of previous testing work. The triangle function takes as input three real numbers that represent the lengths of the three sides of a triangle and then determines whether the triangle is irregular, scalene, isosceles, or equilateral. The code for the triangle function is eighty lines long. The second function (i.e., LE2), which is known as *NextDate*, accepts a date as its input, verifies that date, and then calculates the date of the next daytime. The input consists of a day, a month, and a year, each of which is represented by an integer. The LE3, LE4, LE6, and LE7 are the instances of functions that are considered arbitrary. The Traffic Alert and Collision Avoidance System (TCAS) is an on-board aircraft embedded system for conflict detection and resolution. *Alt_sep_test* (i.e., LE5) is the highest-level function of TCAS. This function takes fourteen global variables as input. The last logical function (i.e., LE8) is taken from TCAS of an avionics system. This function list is used to ensure a fair comparison of all competing meta-heuristic algorithms. The logical functions considered in this experiment comprise three, four, and five predicates only.

VI. EXPERIMENTAL RESULTS ALIGNED WITH RQs

The section provides the answers to the given research questions (RQs) in relation to our experimental findings.

A. TO WHAT EXTENT DOES THE USE OF FAEPO IMPROVE EPO PERFORMANCE? (RQ1)

The improvements that the proposed FAEPO brings over the EPO are twofold: (1) adaptive tuning of control parameters; (2) elimination of unnecessary fitness calculations. This study provides a novel Fuzzy Adaptive EPO (FAEPO) algorithm that integrates a Mamdani fuzzy inference system (see Figure 2), which enables adaptive control of exploration and exploitation by tuning the control parameters f and l according to the need of a particular search problem. The fuzzy rules shown in Table 4 and input-output membership functions are shown in Figure 3 and Figure 4, respectively. Moreover, the proposed FAEPO improves the time complexity of the original EPO by excluding the unnecessary fitness calculation for each agent in each iteration when no position update is performed. Instead of the redundant fitness evaluation in EPO (i.e., in line 3 of Figure 1), the proposed FAEPO

TABLE 10. List of tested logical expression.

S.N.	Function Name	Logical Expression (LE)	LE#	No. of Predicates
1	Triangle function [101]	(A||B||C)	LE1	3
2	NextDate function [101]	((A||B)||C)	LE2	3
3	Instance of function	(A&&B)||($\sim C$)	LE3	3
4	Instance of function	A||(B&&C)	LE4	3
5	Alt_sept_test [102]	(A&&(B&&C)||($\sim D$))	LE5	4
6	Instance of function	(A||B) && (C||D)	LE6	4
7	Instance of function	(A&&B)||(C&&D)	LE7	4
8	Traffic Collision Avoidance System (TCAS) [103]	A&&(($\sim B$)||($\sim C$))&&D||E	LE8	5

evaluates the initial fitness of all emperor penguins once before entering the main loop shown in line 4 of Figure 5.

B. HOW ARE THE CONVERGENCE EFFICIENCY AND STATISTICAL SIGNIFICANCE OF FAEPO COMPARED TO OTHER META-HEURISTIC ALGORITHMS? (RQ2)

In this section, the FAEPO algorithm is evaluated against a set of classical optimization benchmark test functions to answer research question 2 (RQ2). Test problems are considered necessary in the development of optimization algorithms. Therefore, there are standard test functions that many researchers have used. For a similar reason, some test functions are also used in this experiment to evaluate the performance of the proposed FAEPO. The performance of the proposed algorithm, FAEPO, is compared with that of its predecessor (i.e., EPO), it's an improved variant (i.e., IEPO) and the other five benchmark meta-heuristic algorithms, which are listed in Table 7. Next, Tables 11 and 12 summarize the statistical results (i.e., best, mean, standard deviation, and rank) next to the different algorithms compared for various dimensions. Finally, the last two rows of the tables provide the Friedman mean ranking and the overall ranking. The first measure, the Friedman mean rank, is calculated by considering the ranks of the algorithms, with the positions determined by sorting the means. Similarly, the overall rankings of the competing meta-heuristic algorithms are determined by sorting the average rankings. Moreover, the Friedman mean rank of the competing meta-heuristic algorithms for the benchmark test functions is shown in Figure 7 and Figure 8 for unimodal (F1-F6) and multimodal (F7-F12) benchmark functions, respectively.

According to the experimental results of the unimodal functions in Table 11 for different dimensions, FAEPO outperforms the other competitors and secures the first place due to the improved balance between exploitation and exploration, followed by STOA, PSO, SSA, EPO, GA, MFO, and IEPO. Figure 7 illustrates the Friedman mean ranking of the competing meta-heuristic algorithms based on their average results. As mentioned in Section 3, tuning the parameters f and l improves the algorithm's ability for exploration and exploitation. A close competitor of FAEPO in terms of ranking is STOA. FAEPO obtains the first rank, while EPO and IEPO obtain the fifth and eighth rank, respectively. So the average rank of FAEPO is 1 and is clearly ahead of EPO and IEPO.

Table 11 also records the standard deviation of the identified the best solutions for fourteen different dimensional tests to indicate the consistency of the solutions. FAEPO has the lowest standard deviation in nine out of fourteen different dimensional tests of the unimodal functions, while MFO, SSA, STOA, PSO, GA, EPO, and IEPO have the lowest standard deviation in 3, 4, 6, 4, 0, 3, 3 tests, respectively. These results demonstrate the consistency of FAEPO in solving problems. This is also true for all dimension of F6, where all competing algorithms except GA have identical standard deviations (i.e., 0.000E+00).

Table 12 summarizes the results obtained for multimodal functions. Unlike unimodal functions, multimodal functions have multiple optimal solutions. However, in our experiments, all competing meta-heuristic algorithms have difficulties in finding the global optimal solution. However, if multiple global optimal solutions exist due to equal fitness values, identifying one of these solutions would be considered a success.

Multimodal functions show results identical to those of unimodal functions. In this scenario, FAEPO again achieves first place. However, the STOA algorithm improves and secures the second position. The EPO and IEPO algorithms, on the other hand, performs poorly and secures fourth and eighth place. The ranks of the remaining algorithms MFO, SSA, STOA, PSO, and GA are 7, 6, 2, 5, and 3, respectively.

Figure 8 illustrates the Friedman mean rank of all competing meta-heuristic algorithms for multimodal benchmark test functions. For the standard deviations recorded in Table 12, FAEPO, and STOA achieve a minimum standard deviation of 0.00E+00 in five different dimensional test cases. The PSO, EPO, and IEPO achieve a minimum standard deviation of 0.00E+00 in 1, 1, and 3 different dimensional test cases, respectively, whereas MFO, SSA, and GA failed to achieve minimum standard deviation.

Convergence analysis is primarily used to better understand the behavior of optimization algorithms. Figure 9 and Figure 11 for unimodal and Figure 10 and Figure 12 for multimodal test functions show the best convergence curves (among 30 runs) of FAEPO and other competing meta-heuristic algorithms.

Figure 11 shows an intense competition between FAEPO and EPO. On the other hand, PSO manages to converge fairly. The other competing meta-heuristic algorithms, namely MFO, SSA, STOA, GA, and IEPO get stuck in the local optimum and cannot recover. On the contrary, FAEPO has

TABLE 11. Statistical results for the unimodal benchmark test functions.

Function ID	D	–	MFO	SSA	STOA	PSO	GA	EPO	IEPO	FAEPO
F1	2	Best	2.41E-25	2.27E-17	1.48E-58	4.85E-76	3.11E-84	8.43E-152	3.39E-17	5.79E-156
		Mean	1.93E-05	2.77E-15	1.09E-40	7.11E-53	2.81E-17	2.08E-124	1.28E-01	5.44E-150
		Stdev	1.03E-04	3.68E-15	5.85E-40	3.83E-52	1.52E-16	9.86E-124	3.21E-01	2.93E-149
		Rank	7	6	4	3	5	2	8	1
F2	2	Best	2.93E-01	2.93E-01	2.93E-01	2.93E-01	2.93E-01	2.93E-01	3.24E-01	2.93E-01
		Mean	2.98E-01	2.93E-01	2.93E-01	2.93E-01	3.35E-01	3.86E-01	4.30E-01	2.95E-01
		Stdev	1.08E-02	2.71E-08	6.50E-06	1.78E-06	4.37E-02	6.79E-02	5.65E-02	1.21E-02
		Rank	5	2	2	2	6	7	8	4
F3	50	Best	1.09E-06	2.19E-06	1.21E-53	1.16E-24	3.67E-14	3.51E-125	3.50E-02	3.00E-128
		Mean	3.77E-02	8.73E-06	4.09E-19	4.14E-13	4.75E-07	1.11E-01	9.88E-01	2.73E-119
		Stdev	1.36E-01	6.07E-06	2.12E-18	2.01E-12	1.25E-06	4.12E-01	5.59E-01	9.45E-120
		Rank	6	5	2	3	4	7	8	1
	70	Best	2.08E-04	1.81E-06	2.06E-46	5.60E-22	2.71E-15	2.53E-59	5.31E-01	3.11E-76
		Mean	7.35E-02	1.15E-05	4.86E-04	4.14E-13	2.53E-07	9.88E-01	1.42E+00	1.89E-31
		Stdev	1.17E-01	6.31E-06	1.91E-03	1.33E-12	5.61E-07	7.64E-01	5.60E-01	7.06E-31
		Rank	6	4	5	2	3	7	8	1
	100	Best	3.22E-03	1.49E-06	1.29E-32	1.48E-16	8.75E-15	9.63E-01	7.07E-01	4.19E-02
		Mean	1.59E-01	1.35E-05	2.35E-02	3.01E-11	1.61E-07	1.97E+00	1.61E+00	2.10E-01
		Stdev	2.10E-01	9.46E-06	5.59E-02	6.67E-11	5.35E-07	5.37E-01	5.54E-01	3.41E-01
		Rank	5	3	4	1	2	8	7	6
F4	50	Best	4.15E+02	1.75E+25	1.41E-09	4.07E+02	9.30E+02	2.78E+02	3.08E+43	2.00E+02
		Mean	9.11E+02	7.70E+47	5.09E-08	8.02E+02	7.95E+23	7.16E+74	1.02E+75	2.29E+02
		Stdev	3.42E+02	3.43E+48	4.68E-08	2.17E+02	4.15E+24	3.15E+75	2.98E+75	5.72E+01
		Rank	4	6	1	3	5	7	8	2
	70	Best	1.31E+03	1.04E+43	6.46E-09	8.66E+02	1.19E+03	3.69E+02	1.37E+49	2.12E+02
		Mean	1.80E+03	1.19E+77	4.73E-07	1.37E+03	1.54E+41	4.30E+106	1.04E+106	3.45E+02
		Stdev	2.60E+02	6.42E+77	8.01E-07	2.29E+02	7.94E+41	2.31E+107	4.99E+106	8.08E+01
		Rank	4	6	1	3	5	8	7	2
	100	Best	2.49E+03	3.24E+78	1.46E-07	1.59E+03	1.02E+09	5.03E+02	2.36E+80	4.26E+02
		Mean	3.75E+03	1.39E+105	2.89E-06	1.07E+31	2.81E+74	1.05E+152	2.30E+150	5.92E+02
		Stdev	4.92E+02	7.29E+105	4.73E-06	5.75E+31	1.51E+75	5.62E+152	1.06E+151	1.74E+02
		Rank	3	6	1	4	5	8	7	2
F5	50	Best	4.20E+01	7.79E-04	8.36E-18	3.61E+00	1.56E-01	5.29E-34	1.46E+01	3.97E-36
		Mean	5.88E+02	5.85E-01	8.60E-15	3.04E+01	7.44E+00	3.01E-27	3.14E+11	7.14E-34
		Stdev	5.62E+02	1.38E+00	2.05E-14	2.59E+01	6.98E+00	1.57E-26	1.58E+12	2.12E-33
		Rank	7	4	3	6	5	2	8	1
	70	Best	1.59E+02	1.19E+00	3.91E-15	1.21E+01	1.20E+01	3.82E-29	1.89E+01	2.52E-30
		Mean	1.42E+03	3.16E+01	1.87E-12	8.09E+01	3.39E+01	1.60E-23	2.61E+13	1.98E-27
		Stdev	1.12E+03	3.06E+01	2.68E-12	8.30E+01	2.02E+01	3.43E-23	9.84E+13	8.99E-28
		Rank	7	4	3	6	5	2	8	1
	100	Best	1.40E+03	8.91E+01	7.73E-14	2.85E+01	4.73E+01	2.25E-22	8.66E+01	4.57E-22
		Mean	7.14E+07	2.56E+02	2.85E-10	1.10E+02	1.28E+02	1.29E-18	3.90E+14	1.26E-21
		Stdev	2.92E+08	1.51E+02	6.95E-10	8.06E+01	4.23E+01	3.19E-18	1.75E+15	2.53E-21
		Rank	7	6	3	4	5	2	8	1
F6	50	Best	0.00E+00	4.46E-287	0.00E+00	0.00E+00	6.20E-67	0.00E+00	0.00E+00	0.00E+00
		Mean	1.13E-301	9.19E-234	0.00E+00	0.00E+00	2.63E-49	0.00E+00	0.00E+00	0.00E+00
		Stdev	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.12E-48	0.00E+00	0.00E+00	0.00E+00
		Rank	6	7	3	3	8	3	3	3
	70	Best	0.0E+00	0.0E+00	0.0E+00	0.0E+00	4.2E-92	0.0E+00	0.0E+00	0.0E+00
		Mean	0.0E+00	5.4E-300	0.0E+00	0.0E+00	2.8E-56	0.0E+00	0.0E+00	0.0E+00
		Stdev	0.0E+00	0.0E+00	0.0E+00	0.0E+00	1.5E-55	0.0E+00	0.0E+00	0.0E+00
		Rank	3.5	7	3.5	3.5	8	3.5	3.5	3.5
	100	Best	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.23E-122	0.00E+00	0.00E+00	0.00E+00
		Mean	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.54E-85	0.00E+00	0.00E+00	0.00E+00
		Stdev	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.09E-85	0.00E+00	0.00E+00	0.00E+00
		Rank	4	4	4	4	8	4	4	4
Friedman Mean Rank			5.32	5.00	2.82	3.39	5.29	5.04	6.82	2.32
Overall Rank			7	4	2	3	6	5	8	1

TABLE 12. Statistical results for the multimodal benchmark test functions.

Function ID	D	–	MFO	SSA	STOA	PSO	GA	EPO	IEPO	FAEPO
F7	2	Best	1.46E-135	2.26E-16	1.58E-78	5.75E-245	3.42E-15	8.94E-289	4.40E-22	2.01E-290
		Mean	1.31E+00	1.26E-13	8.97E-62	9.82E-01	1.64E+00	1.10E-260	1.00E+00	3.63E-278
		Stdev	3.33E+00	1.51E-13	4.83E-61	2.94E+00	3.65E+00	0.00E+00	2.86E+00	0.00E+00
		Rank	7	4	3	5	8	2	6	1
F8	2	Best	1.00E-04	5.07E-02	1.41E-03	1.00E-04	5.54E-03	1.00E-04	1.09E-01	1.00E-04
		Mean	1.18E-02	1.71E-01	5.13E-01	3.64E-03	8.99E-02	4.57E-01	6.96E-01	9.32E-02
		Stdev	1.68E-02	4.78E-02	1.59E-01	9.72E-03	8.11E-02	1.96E-01	2.25E-01	1.63E-01
		Rank	2	5	7	1	3	6	8	4
F9	50	Best	2.97E+02	1.13E+02	8.93E-11	9.88E+01	9.27E+01	9.18E-09	4.09E+02	5.68E-14
		Mean	3.97E+02	1.96E+02	9.08E+00	1.66E+02	1.35E+02	7.68E+01	6.62E+02	6.59E-08
		Stdev	6.56E+01	4.10E+01	1.40E+01	3.45E+01	2.83E+01	1.23E+02	1.00E+02	3.55E-07
		Rank	7	6	2	5	4	3	8	1
	70	Best	4.97E+02	2.01E+02	1.31E-09	1.65E+02	1.33E+02	5.75E-08	7.18E+02	6.53E-10
		Mean	6.13E+02	3.00E+02	8.23E+00	2.60E+02	1.94E+02	1.22E+02	9.86E+02	4.42E-09
		Stdev	6.01E+01	4.69E+01	1.40E+01	3.93E+01	3.36E+01	1.95E+02	1.12E+02	9.59E-09
		Rank	7	6	2	5	4	3	8	1
	100	Best	7.50E+02	3.12E+02	5.00E-12	3.37E+02	1.55E+02	4.55E-13	1.01E+03	0.00E+00
		Mean	9.53E+02	4.33E+02	4.99E+00	4.75E+02	2.56E+02	1.83E+02	1.46E+03	1.08E+02
		Stdev	6.86E+01	5.43E+01	1.38E+01	7.20E+01	4.95E+01	3.65E+02	1.86E+02	1.81E+02
		Rank	7	5	1	6	4	3	8	2
F10	50	Best	1.10E+15	1.44E+04	8.04E-13	3.68E-01	1.26E+10	3.55E+05	9.87E+12	1.79E-13
		Mean	1.39E+22	5.13E+10	6.73E-04	1.94E+11	5.20E+16	2.14E+20	9.55E+25	3.11E+08
		Stdev	3.74E+22	1.93E+11	1.94E-03	9.03E+11	1.47E+17	5.64E+20	3.25E+26	1.49E+09
		Rank	7	3	1	4	5	6	8	2
	70	Best	6.90E+27	6.39E+07	1.56E-15	1.87E+04	5.17E+14	2.40E+16	1.58E+25	4.80E-04
		Mean	1.57E+34	1.93E+21	3.52E-04	1.32E+27	4.69E+28	1.31E+36	1.28E+41	1.63E+23
		Stdev	4.77E+34	6.69E+21	9.28E-04	7.11E+27	2.47E+29	4.45E+36	5.67E+41	8.77E+23
		Rank	6	2	1	4	5	7	8	3
	100	Best	2.82E+43	1.57E+14	3.66E-09	1.30E+16	2.28E+25	2.38E+38	4.62E+46	2.13E+32
		Mean	1.19E+52	2.32E+33	2.77E-02	2.84E+31	1.67E+47	1.17E+57	2.77E+59	4.35E+41
		Stdev	4.28E+52	8.44E+33	8.91E-02	1.53E+32	9.01E+47	5.59E+57	1.20E+60	1.63E+42
		Rank	6	3	1	2	5	7	8	4
F11	50	Best	7.08E-20	4.50E-19	1.63E-19	2.78E-20	1.62E-20	1.65E-19	1.72E-19	1.66E-19
		Mean	1.03E-19	4.86E-18	1.69E-19	1.55E-19	2.25E-20	1.74E-19	1.79E-19	1.73E-19
		Stdev	1.59E-20	6.58E-18	2.64E-21	1.26E-19	3.78E-21	3.13E-21	1.30E-21	3.09E-21
		Rank	2	8	4	3	1	6	7	5
	70	Best	2.83E-28	1.34E-26	7.39E-28	2.66E-28	5.61E-29	7.51E-28	7.72E-28	7.19E-28
		Mean	4.20E-28	3.17E-24	7.64E-28	3.05E-26	1.04E-28	7.82E-28	7.94E-28	7.35E-28
		Stdev	1.11E-28	6.51E-24	1.02E-29	8.69E-26	6.32E-29	1.13E-29	6.71E-30	1.12E-29
		Rank	2	8	4	7	1	5	6	3
	100	Best	7.80E-41	1.97E-36	1.90E-40	9.06E-41	1.34E-41	1.95E-40	2.04E-40	1.90E-40
		Mean	3.13E-40	5.15E-33	1.97E-40	6.95E-36	1.13E-40	2.01E-40	2.04E-40	1.93E-40
		Stdev	6.91E-40	1.88E-32	2.24E-42	2.59E-35	1.37E-40	2.49E-42	4.15E-56	1.26E-42
		Rank	6	8	3	7	1	4	5	2
F12	50	Best	5.14E+00	3.17E-01	3.41E-03	1.81E-01	1.82E-01	2.44E-03	3.97E+01	9.61E-04
		Mean	4.84E+01	1.08E+00	9.34E-03	7.31E-01	3.20E-01	1.42E-02	2.14E+02	1.00E-02
		Stdev	4.18E+01	3.38E-01	6.96E-03	1.18E+00	6.79E-02	1.26E-02	1.18E+02	9.54E-03
		Rank	7	6	1	5	4	3	8	2
	70	Best	4.33E+01	1.61E+00	2.39E-03	6.86E-01	2.90E-01	4.59E-03	1.99E+02	1.92E-03
		Mean	1.48E+02	2.49E+00	1.37E-02	1.28E+00	4.34E-01	2.56E-02	6.07E+02	2.56E-03
		Stdev	7.87E+01	6.12E-01	9.32E-03	6.66E-01	8.37E-02	2.47E-02	2.37E+02	8.25E-04
		Rank	7	6	2	5	4	3	8	1
	100	Best	1.34E+02	3.86E+00	4.66E-03	2.03E+00	4.04E-01	7.40E-03	9.05E+02	4.91E-03
		Mean	3.62E+02	7.66E+00	1.75E-02	5.35E+00	7.15E-01	4.13E-02	1.59E+03	5.12E-03
		Stdev	1.42E+02	2.13E+00	9.21E-03	5.49E+00	1.58E-01	2.53E-02	3.65E+02	1.05E-03
		Rank	7	6	2	5	4	3	8	1
Friedman Mean Rank			5.71	5.43	2.43	4.57	3.79	4.36	7.43	2.29
Overall Rank			7	6	2	5	3	4	8	1

TABLE 13. Experimental results were obtained through 30 independent runs using IMDB dataset.

Number of Test Skills	Mean Team Size (TS) Mean Team Cost (TC)	MFO	SSA	STOA	PSO	GA	EPO	IEPO	FAEPO
5	TS	3.0000	3.0000	3.0000	3.0000	3.1667	3.0000	3.0000	3.0000
	TC	2.1638	2.2215	2.1439	2.3089	2.7279	2.3710	2.5024	2.0782
10	TS	4.6333	4.7333	5.0333	4.7000	4.5000	5.4333	5.9000	4.1333
	TC	6.2722	6.5357	7.5985	6.3536	5.7384	9.3560	11.5432	5.0501
15	TS	7.3000	7.0000	7.3333	7.1000	6.8667	7.8333	8.0667	6.7667
	TC	19.8670	18.3549	20.0965	18.5965	17.6289	23.2339	24.8830	17.0822
20	TS	8.5333	8.5333	8.9667	8.2667	8.4667	9.7333	10.2667	7.9333
	TC	26.5503	26.8599	30.8383	24.9286	26.3670	35.5611	41.3109	23.2583
25	TS	11.2000	11.7000	11.7667	11.4000	11.7000	13.1000	12.8000	11.1333
	TC	48.7030	52.7929	53.6963	49.7990	52.4914	66.7576	65.2642	47.7404

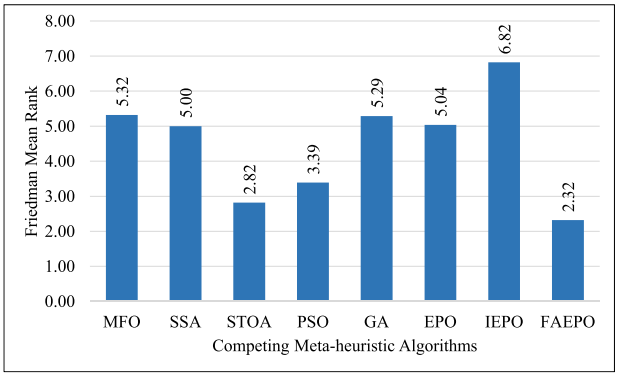


FIGURE 7. Friedman mean rank for unimodal benchmark functions F1-F6.

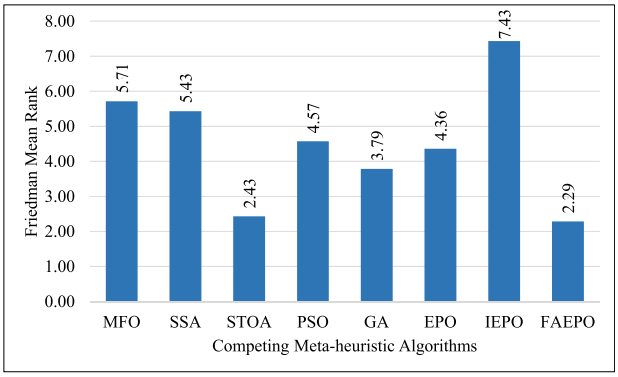


FIGURE 8. Friedman mean rank for multimodal benchmark functions F7-F12.

better convergence for F1-F6, while PSO converges better for F3 for dimension 100.

Analogous to the convergence properties for unimodal functions, FAEPO is also shown to converge faster than other algorithms for multimodal functions, as shown in Figure 12. Moreover, a competing convergence characteristic between FAEPO, EPO, and IEPO can be observed, with the former dominating the latter for F7-F12 except F11.

C. HOW IS THE PERFORMANCE OF FAEPO COMPARED TO EPO, IEPO, AND OTHER META-HEURISTIC ALGORITHMS? (RQ3)

To answer this research question (RQ3), we used our proposed FAEPO and other competing meta-heuristic algorithms in three global optimization problems: team formation optimization, binary sequence with low autocorrelation - LABS, MC /DC test case generation. Numerical results of all experiments are plotted and performance is compared using the Friedman Mean Rank test. The performance evaluation of FAEPO for each optimization problem is described as follows.

1) PERFORMANCE EVALUATION OF FAEPO ON TEAM FORMATION OPTIMIZATION PROBLEM

FAEPO is compared with state-of-the-art meta-heuristic algorithms including its predecessor EPO, and an improved variant of EPO on two benchmark datasets, namely IMDB and DBLP. Tables 13 and 14 show the results obtained for all competing meta-heuristic algorithms in 30 independent runs for TFO using the benchmark datasets: IMDB and DBLP, respectively.

Although the average results are scattered across different cells of the tables, FAEPO obtains the lowest average value in terms of team size (TS) and team cost (TC) for both datasets. With the exception of GA, all competing meta-heuristic algorithms show comparable performance for the lowest number of test skills values (i.e., 5) with respect to the mean team size for the IMDB dataset. The performance differences become more pronounced with a higher number of test skills values. Moreover, FAEPO exhibits the lowest mean values for all S^R, which shows the superiority of the proposed algorithm in finding optimal solutions over the previously compared meta-heuristic algorithms. Moreover, FAEPO performs better in mean values than any other algorithm of interest for the IMDB datasets. FAEPO's closest competitor is PSO and MFO for the IMDB dataset in terms of average team size and average team cost, respectively.

Similar to IMDB, the results of DBLP also demonstrate the superiority of the proposed algorithm, FAEPO, over all other competing meta-heuristic algorithms for $S^R = \{5, 10\}$. For example, although MFO and SSA obtain the best average for all $S^R = \{15, 20, 25\}$, FAEPO obtains a better average in terms of team size and team cost than EPO and IEPO for all S^R values. Friedman's mean ranking and overall ranking are shown in Table 15 and Table 16 for the IMDB and DBLP datasets, respectively. FAEPO outperforms all other competing meta-heuristic algorithms by securing the overall rank of

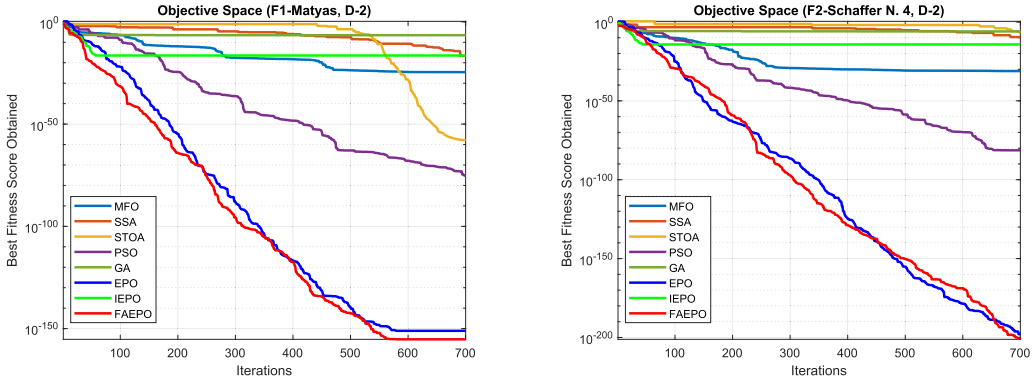


FIGURE 9. Convergence curves of the competing meta-heuristic algorithms on unimodal fixed dimensional benchmark test functions (F1-F2).

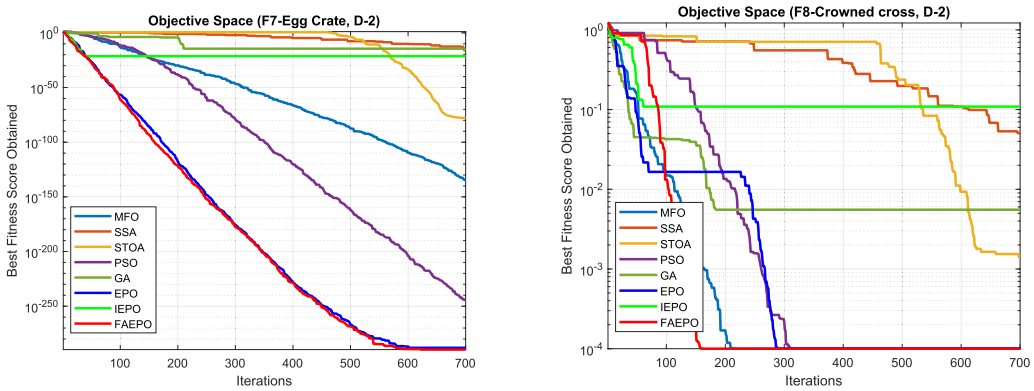


FIGURE 10. Convergence curves of the competing meta-heuristic algorithms on multimodal fixed dimensional benchmark test functions (F7-F8).

1 for the IMDB datasets, as can be seen in Table 15. On the other hand, FAEPO secures rank 1 in DBLP dataset for $S^R = \{5, 10\}$, but performs poorly for $S^R = \{15, 20, 25\}$. FAEPO tied for third place among all the meta-heuristic algorithms tested on the DBLP dataset. Friedman's mean rankings for the IMDB and DBLP datasets are shown in Figure 13 and Figure 14, respectively.

In terms of the fairness of the comparison with other meta-heuristic algorithms, several points can be further discussed. First, we compared our work with the same category of algorithms (nature-inspired meta-heuristic algorithms). Therefore, for these algorithms, we consider the best parameter values proposed in their original works and listed in Table 7. Second, all competing meta-heuristic algorithms stop working when the number of maximum fitness scores is reached, which means that all competing meta-heuristic algorithms get equal chances to update their search agents.

From the discussions, it can be concluded that the proposed algorithm, FAEPO, is the best performing algorithm for the team formation optimization problem among all the competing meta-heuristic algorithms, especially EPO and IEPO.

2) PERFORMANCE EVALUATION OF FAEPO ON LABS PROBLEM

Table 17 shows the experimental results of the best obtained optimal merit factor (F_N) of the LABS problem for sequence lengths up to 31. The obtained average merit factors and actual merit factors [97] are also shown in Table 17. The sequence lengths below seven are not considered, since these sequences provide comparable performance for all algorithms compared.

In this experiment, we used two ranking methods for performance evaluation: Difference mean ($\overline{D}_{F_N}$) rank and Friedman mean rank. Here, the difference mean represents the mean of absolute difference between the actual merit factors calculated by exhaustive search [97] and the best obtained merit factors. The mathematical formulation of the difference mean is shown in Eq. (30) as follows:

$$\overline{D}_{F_N} = \frac{1}{25}\sum_{N=7}^{31}\left|F_{N(actual)} - F_{N(best\ obtained)}\right| \tag{30}$$

The goal of the competing meta-heuristic algorithms is to find the merit factor that is equal or close to the actual merit factor. Therefore, the difference mean can determine the overall performance of the competing meta-heuristic algorithms by comparing the closeness to the $\overline{D}_{F_N}$ of the actual merit factor (i.e., zero). The ranking of the difference means provides information about the performance of the competing meta-heuristic algorithms in this experiment. The ranking of the competing meta-heuristic algorithms are shown in the last row of Table 17.

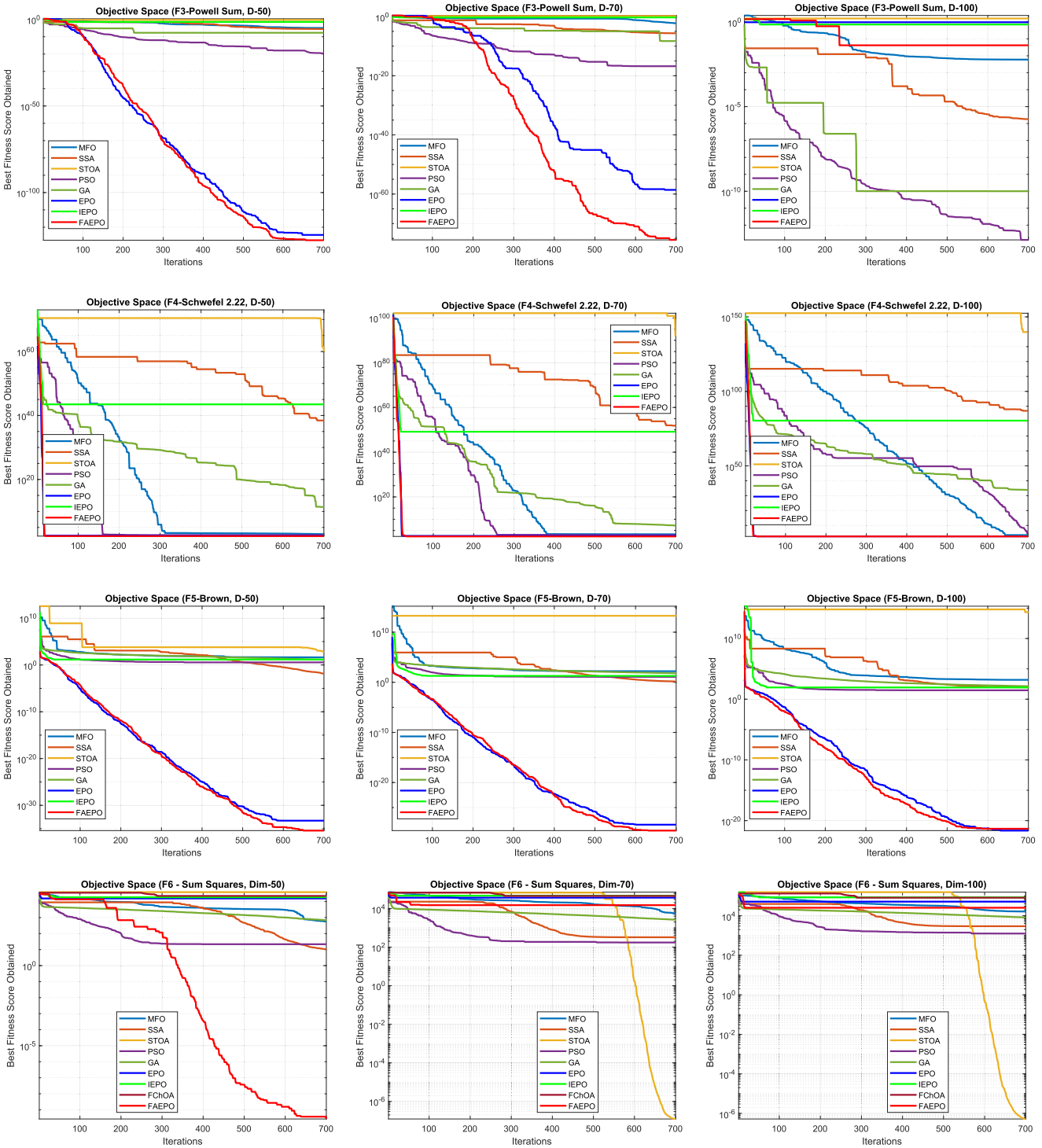


FIGURE 11. Convergence curves of the competing meta-heuristic algorithms on unimodal variable dimensional benchmark test functions (F3-F6).

The Friedman mean ranking was also used in this experiment to rank the competing meta-heuristic algorithms based on the average merit factors. Table 18 shows the Friedman mean ranking of the competing meta-heuristic algorithms based on the performance (average merit factor) recorded in Table 17. Friedman's mean ranking of all competing meta-heuristic algorithms for LABS problem is shown in Figure 15. FAEPO achieves the rank 1 in each ranking among all the competing meta-heuristic algorithms.

The experimental results uncover that the proposed FAEPO quickly finds the near-optimal (best found) merit factors equal to or close to actual merit factors. Moreover, the

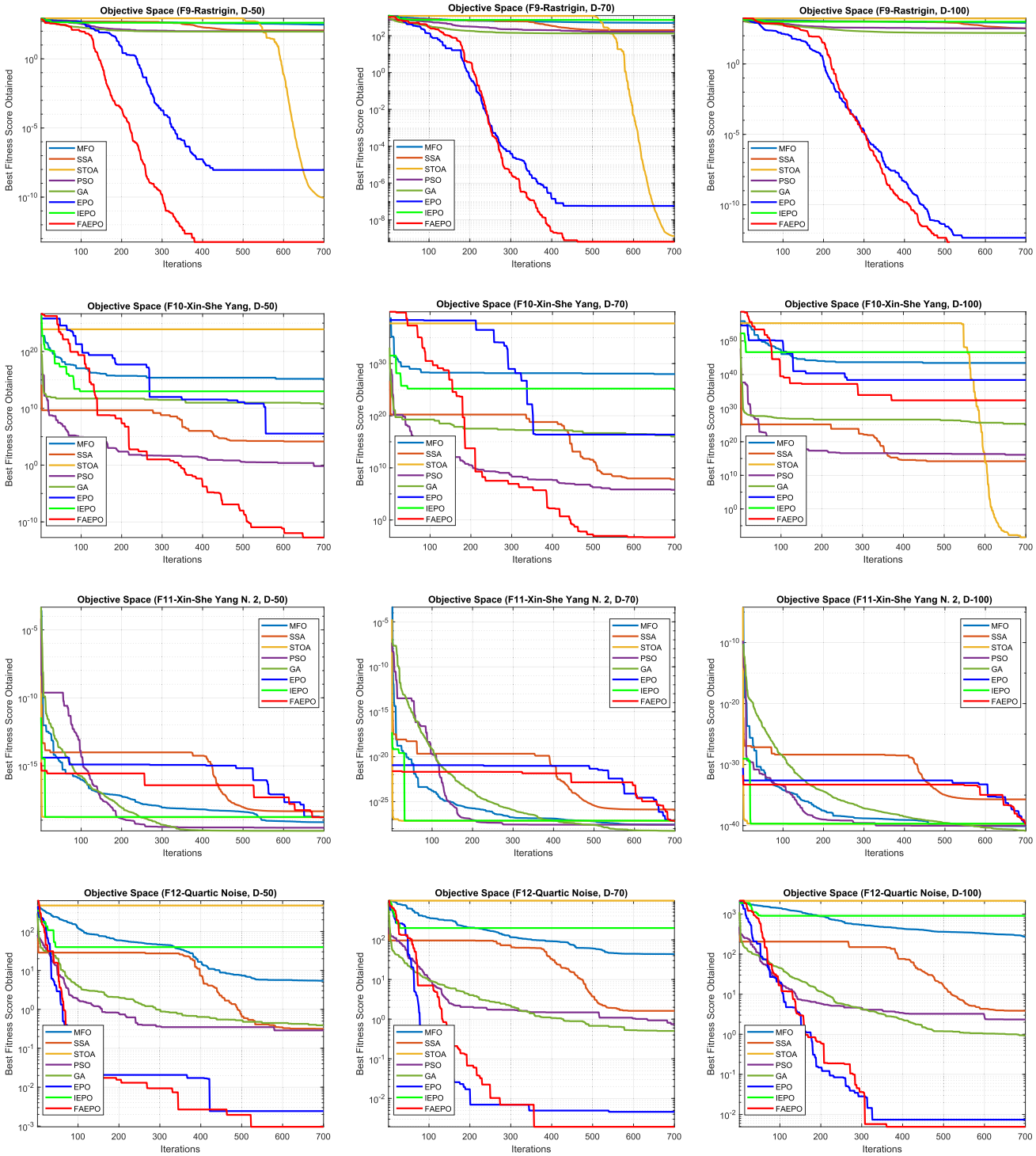


FIGURE 12. Convergence curves of the competing meta-heuristic algorithms on multimodal variable dimensional benchmark test functions (F9-F12).

acquired maximum average merit factors and minimum difference mean results demonstrate that the proposed FAEPO outperforms the other algorithms.

3) PERFORMANCE EVALUATION OF FAEPO ON MC/DC TEST CASE GENERATION PROBLEM

We conducted our MC/DC test case generation experiment to evaluate our proposed algorithm in terms of the size of generated test cases. This experiment was performed on a benchmark set of eight logical functions, which are listed in Table 10. All the experimental results are shown in Table 19. As shown in Table 19, all competing meta-heuristic algorithms achieve identical results for the LE1 and LE2 functions. Similarly, all algorithms beat GA for functions LE3, LE4, LE5, LE6, LE7, and LE8 in generating average test cases.

TABLE 14. Experimental results were obtained through 30 independent runs using the DBLP dataset.

Number of Test Skills	Mean Team Size (TS) Mean Team Cost (TC)	MFO	SSA	STOA	PSO	GA	EPO	IEPO	FAEPO
5	TS	3.967	3.967	4.000	3.933	4.000	5.000	4.933	3.833
	TC	5.474	5.466	5.683	5.376	5.689	9.645	9.389	5.345
10	TS	6.167	6.267	6.300	6.400	7.167	7.633	8.133	6.000
	TC	15.138	15.676	16.096	16.669	21.574	24.596	28.132	14.458
15	TS	9.767	9.700	10.667	9.867	11.067	13.800	13.100	10.033
	TC	41.049	40.030	49.877	42.125	54.063	85.569	77.292	43.491
20	TS	13.167	13.067	14.633	14.000	14.800	16.700	16.567	14.100
	TC	78.412	76.981	97.685	88.774	100.044	129.079	127.283	90.459
25	TS	16.700	16.933	18.267	17.667	19.633	21.000	20.933	18.000
	TC	127.384	131.090	153.465	143.640	178.573	202.515	201.476	147.352

TABLE 15. Friedman mean ranking of competing meta-heuristic algorithms using IMDB dataset based on the performance in Table 13.

Number of Test Skills	TS-Team Size TC-Team Cost	MFO	SSA	STOA	PSO	GA	EPO	IEPO	FAEPO
5	TS	4	4	4	4	8	4	4	4
	TC	3	4	2	5	8	6	7	1
10	TS	3	5	6	4	2	7	8	1
	TC	3	5	6	4	2	7	8	1
15	TS	5	3	6	4	2	7	8	1
	TC	5	3	6	4	2	7	8	1
20	TS	4.5	4.5	6	2	3	7	8	1
	TC	4	5	6	2	3	7	8	1
25	TS	2	4.5	6	3	4.5	8	7	1
	TC	2	5	6	3	4	8	7	1
Friedman Mean Rank (TS)		3.7	4.2	5.6	3.4	3.9	6.6	7	1.6
Overall Rank (TS)		3	5	6	2	4	7	8	1
Friedman Mean Rank (TC)		3.4	4.4	5.2	3.6	3.8	7	7.6	1
Overall Rank (TC)		2	5	6	3	4	7	8	1

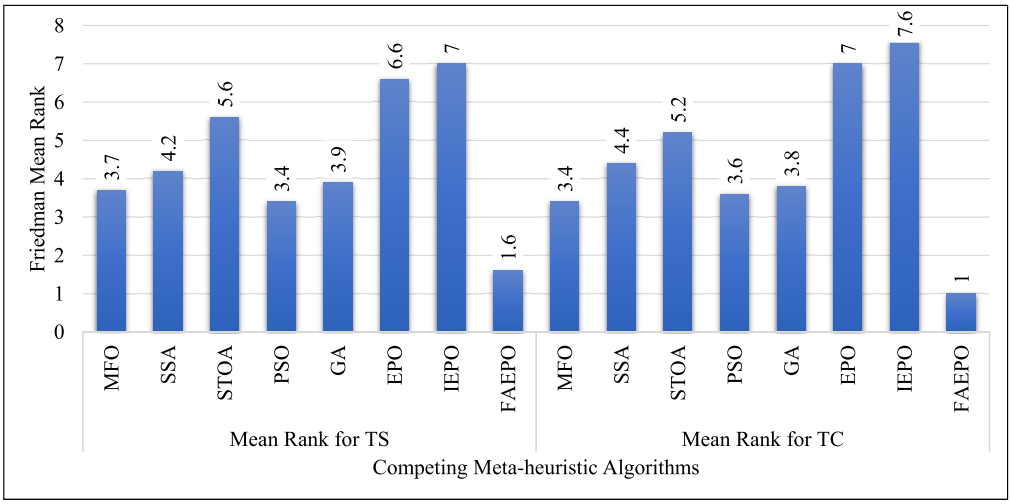


FIGURE 13. Friedman mean rank for IMDB dataset.

Moreover, PSO, GA, and EPO give poor results in generating average test cases for function LE6. Moreover, PSO and GA provide poor results for LE7 and LE8 functions. On the other hand, MFO, SSA, STAO, IEPO, and FAEPO perform equally better than the other competing meta-heuristic algorithms in all cases.

The entries in Table 20 represent the Friedman mean ranking and the overall ranking of the competing meta-heuristic

TABLE 16. Friedman mean ranking of competing meta-heuristic algorithms using DBLP dataset based on the performance in Table 14.

Number of Test Skills	TS-Team Size TC-Team Cost	MFO	SSA	STOA	PSO	GA	EPO	IEPO	FAEPO
5	TS	3.5	3.5	5.5	2	5.5	8	7	1
	TC	4	3	5	2	6	8	7	1
10	TS	2	3	4	5	6	7	8	1
	TC	2	3	4	5	6	7	8	1
15	TS	2	1	4	3	6	8	7	4
	TC	2	1	4	3	6	8	7	4
20	TS	2	1	4	3	6	8	7	4
	TC	2	1	4	3	6	8	7	4
25	TS	1	2	5	3	6	8	7	4
	TC	1	2	5	3	6	8	7	4
Friedman Mean Rank (TS)		2.1	2.1	4.9	3.2	5.9	7.8	7.2	2.8
Overall Rank (TS)		1.5	1.5	5	4	6	8	7	3
Friedman Mean Rank (TC)		2.2	2	4.8	3.2	6	7.8	7.2	2.8
Overall Rank (TC)		2	1	5	4	6	8	7	3

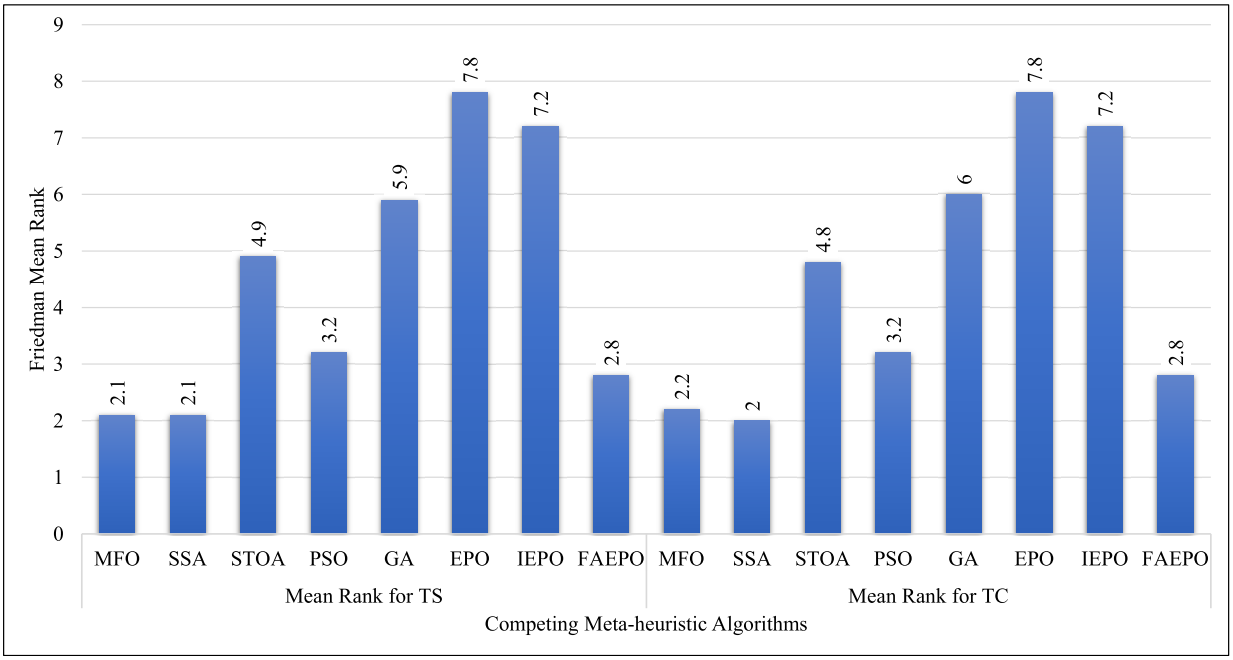


FIGURE 14. Friedman mean rank for DBLP dataset.

algorithms based on the performance shown in Table 19. The overall ranking of the competing meta-heuristic algorithms in Table 20 shows that FAEPO, IEPO, STOA, SSA, and MFO achieve the same rank of 3 for computing the average MC/DC test cases. On the other hand, EPO, PSO, and GA achieve the overall rank of 6, 7, and 8, respectively.

The experimental results suggest that the proposed FAEPO performs better than its predecessor EPO and the other competing meta-heuristic algorithms in terms of ranking. Although five algorithms, including the proposed FAEPO, achieve the same top ranking, EPO receives a poor ranking (i.e., 6). Figure 16 shows Friedman's mean ranking of all competing meta-heuristic algorithms for generating MC/DC test cases.

4) IS THERE ANY OVERHEAD IN TERMS OF THE TIME PERFORMANCE OF FAEPO IMPLEMENTATION?

Complexity is one of the most influential metrics for evaluating the performance of an algorithm. Big Oh (O) notation is commonly used to represent the complexity of an algorithm. Many factors affect the time complexity of a meta-heuristic algorithm. Generally, the number of population (n), dimension (d), maximum iteration (M_{itr}), and fitness function evaluation (f_e) are considered when evaluating the time

TABLE 17. Numerical results of obtained merit factors (F_N) for sequence length $7 \leq N \leq 31$.

Sequence Length (N)	Merit Factor (F_N)	MFO	SSA	STOA	PSO	GA	EPO	IEPO	FAEPO	Actual F_N [99]
7	Best Found	8.167	8.167	8.167	8.167	8.167	8.167	8.167	8.167	8.167
	Average	7.700	8.167	8.167	4.442	7.389	8.167	8.167	8.167	
8	Best Found	4.000	4.000	4.000	4.000	4.000	4.000	4.000	4.000	4.000
	Average	4.000	4.000	4.000	3.400	3.822	4.000	4.000	4.000	
9	Best Found	3.375	3.375	3.375	3.375	3.375	3.375	3.375	3.375	3.375
	Average	3.375	3.375	3.375	2.734	3.150	3.375	3.375	3.375	
10	Best Found	3.846	3.846	3.846	3.846	3.846	3.846	3.846	3.846	3.846
	Average	3.749	3.846	3.846	2.782	3.651	3.846	3.846	3.846	
11	Best Found	12.100	12.100	12.100	12.100	12.100	12.100	12.100	12.100	12.100
	Average	5.422	7.881	12.100	3.132	4.221	12.100	12.100	12.100	
12	Best Found	7.200	7.200	7.200	7.200	7.200	7.200	7.200	7.200	7.200
	Average	5.279	6.994	7.200	2.894	4.376	7.200	7.200	7.200	
13	Best Found	14.083	14.083	14.083	6.036	6.036	14.083	14.083	14.083	14.083
	Average	5.537	7.064	11.669	2.642	3.790	12.742	8.450	14.083	
14	Best Found	5.158	5.158	5.158	5.158	5.158	5.158	5.158	5.158	5.158
	Average	4.343	5.107	5.158	2.635	3.446	5.158	5.158	5.158	
15	Best Found	4.891	7.500	7.500	4.891	4.891	7.500	7.500	7.500	7.500
	Average	4.182	4.873	5.239	2.744	3.091	5.326	5.065	6.370	
16	Best Found	5.333	5.333	5.333	4.000	4.000	5.333	5.333	5.333	5.333
	Average	4.052	4.679	5.257	2.765	3.000	5.283	4.991	5.333	
17	Best Found	4.516	4.516	4.516	3.613	4.516	4.516	4.516	4.516	4.516
	Average	3.702	4.047	4.389	2.453	2.902	4.348	4.275	4.516	
18	Best Found	4.909	4.909	4.909	3.951	3.951	6.480	4.909	6.480	6.480
	Average	3.583	4.366	4.686	2.306	2.822	4.802	4.622	5.328	
19	Best Found	4.402	4.402	5.470	2.959	3.684	5.470	4.878	6.224	6.224
	Average	3.454	4.027	4.538	2.276	2.785	4.610	4.442	5.419	
20	Best Found	4.000	5.263	5.263	2.564	5.263	5.882	5.882	5.882	7.692
	Average	3.274	4.130	4.297	2.071	2.996	4.552	4.509	5.370	
21	Best Found	4.410	4.410	4.410	3.150	3.341	4.410	5.803	8.481	8.481
	Average	3.368	3.966	4.150	2.127	2.584	4.356	4.274	5.248	
22	Best Found	4.410	4.410	4.410	3.150	3.341	4.410	5.803	8.481	6.205
	Average	3.368	3.966	4.150	2.127	2.584	4.356	4.274	5.248	
23	Best Found	4.410	4.410	4.410	3.150	3.341	4.410	5.803	8.481	5.628
	Average	3.368	3.966	4.150	2.127	2.584	4.356	4.274	5.248	
24	Best Found	4.000	4.500	5.143	3.429	3.600	4.500	5.539	5.539	8.000
	Average	3.263	3.681	3.940	1.943	2.480	4.051	4.177	4.743	
25	Best Found	3.906	4.596	4.112	3.255	3.125	4.883	4.596	6.010	8.681
	Average	3.116	3.800	3.840	2.120	2.379	4.105	4.016	4.726	
26	Best Found	4.390	4.390	4.390	3.101	3.101	4.899	5.541	5.541	7.511
	Average	3.193	3.708	3.840	2.044	2.400	4.253	4.331	4.695	
27	Best Found	3.609	4.993	4.734	2.382	3.609	5.283	5.283	5.975	9.851
	Average	3.134	3.879	3.791	1.863	2.337	4.246	4.236	4.887	
28	Best Found	4.558	4.781	4.356	2.761	3.564	5.297	5.026	7.840	7.840
	Average	3.031	3.871	3.763	1.960	2.339	4.063	4.112	4.853	
29	Best Found	3.689	4.473	4.473	2.362	3.138	5.682	4.890	6.007	6.782
	Average	2.947	3.701	3.594	1.769	2.430	4.170	4.150	4.834	
30	Best Found	4.206	4.546	4.206	2.514	3.237	4.945	4.206	5.422	7.627
	Average	2.943	3.546	3.541	1.795	2.270	4.058	3.927	4.431	
31	Best Found	3.784	4.491	4.665	2.626	2.684	4.854	5.058	5.789	7.172
	Average	2.902	3.632	3.497	1.877	2.155	4.028	3.983	4.620	
Difference Mean		3.327	2.607	2.131	4.737	4.059	1.916	2.140	1.426	0.000
Difference Mean Rank		6	5	3	8	7	2	4	1	

TABLE 18. Friedman's mean ranking of competing meta-heuristic algorithms based on the performance in Table 17.

Sequence Length (N)	MFO	SSA	STOA	PSO	GA	EPO	IEPO	FAEPO
7	6	3	3	8	7	3	3	3
8	3.5	3.5	3.5	8	7	3.5	3.5	3.5
9	3.5	3.5	3.5	8	7	3.5	3.5	3.5
10	6	3	3	8	7	3	3	3
11	6	5	2.5	8	7	2.5	2.5	2.5
12	6	5	2.5	8	7	2.5	2.5	2.5
13	6	5	3	8	7	2	4	1
14	6	5	2.5	8	7	2.5	2.5	2.5
15	6	5	3	8	7	2	4	1
16	6	5	3	8	7	2	4	1
17	6	5	2	8	7	3	4	1
18	6	5	3	8	7	2	4	1
19	6	5	3	8	7	2	4	1
20	6	5	4	8	7	2	3	1
21	6	5	4	8	7	2	3	1
22	6	5	4	8	7	2	3	1
23	6	5	4	8	7	2	3	1
24	6	5	4	8	7	3	2	1
25	6	5	4	8	7	2	3	1
26	6	5	4	8	7	3	2	1
27	6	4	5	8	7	2	3	1
28	6	4	5	8	7	3	2	1
29	6	4	5	8	7	2	3	1
30	6	4	5	8	7	2	3	1
31	6	4	5	8	7	2	3	1
Friedman Mean Rank	5.80	4.52	3.62	8.00	7.00	2.42	3.10	1.54
Overall Rank	6	5	4	8	7	2	3	1

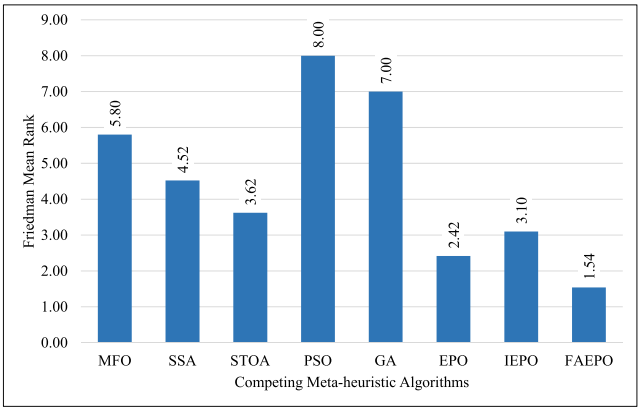


FIGURE 15. Friedman mean rank for LABS problem.

TABLE 19. Cumulative results of obtained MC/DC test cases using benchmark algorithms.

Algorithm	Test Case Pairs	LE1	LE2	LE3	LE4	LE5	LE6	LE7	LE8
MFO	Min	4	4	4	4	5	5	5	6
	Max	4	4	4	4	5	5	5	6
	Average	4	4	4	4	5	5	5	6
SSA	Min	4	4	4	4	5	5	5	6
	Max	4	4	4	4	5	5	5	6
	Average	4	4	4	4	5	5	5	6
STOA	Min	4	4	4	4	5	5	5	6
	Max	4	4	4	4	5	5	5	6
	Average	4	4	4	4	5	5	5	6
PSO	Min	4	4	4	4	5	5	5	6
	Max	4	4	4	4	5	8	8	8
	Average	4	4	4	4	5	6.33	6.4	6.63
GA	Min	4	4	4	4	5	5	6	6
	Max	4	4	5	5	6	8	8	8
	Average	4	4	4.33	4.37	5.77	6.7	6.97	7.37
EPO	Min	4	4	4	4	5	5	5	6
	Max	4	4	4	4	5	6	5	6
	Average	4	4	4	4	5	5.47	5	6
IEPO	Min	4	4	4	4	5	5	5	6
	Max	4	4	4	4	5	5	5	6
	Average	4	4	4	4	5	5	5	6
FAEPO	Min	4	4	4	4	5	5	5	6
	Max	4	4	4	4	5	5	5	6
	Average	4	4	4	4	5	5	5	6

TABLE 20. Friedman mean ranking of competing meta-heuristic algorithms based on the performance in Table 19.

LE#	MFO	SSA	STOA	PSO	GA	EPO	IEPO	FAEPO
LE1	4.5	4.5	4.5	4.5	4.5	4.5	4.5	4.5
LE2	4.5	4.5	4.5	4.5	4.5	4.5	4.5	4.5
LE3	4	4	4	4	8	4	4	4
LE4	4	4	4	4	8	4	4	4
LE5	4	4	4	4	8	4	4	4
LE6	3	3	3	7	8	6	3	3
LE7	3.5	3.5	3.5	7	8	3.5	3.5	3.5
LE8	3.5	3.5	3.5	7	8	3.5	3.5	3.5
Friedman Mean Rank	3.9	3.9	3.9	5.3	7.1	4.3	3.9	3.9
Overall Rank	3	3	3	7	8	6	3	3

complexity of meta-heuristic algorithms. The time complexity of a meta-heuristic algorithm (A_M) can be expressed as follows:

$$O(A_M) = O(Fitness\ Function\ Evaluation + Agent\ Update\ in\ Memory + Dimension\ Update \quad (31)$$

The time complexity of the meta-heuristic algorithms compared in this study is summarized using Eq. (31) in Table 21. The time complexity of each meta-heuristic algorithm in Table 21 can be approximated as $\approx O(M_{itr} \times n \times (f_e + d))$ for sufficiently large n, provided all other operations can be completed in constant time. This approximation shows that adding FIS to FAEPO has no negative impact on performance.

For a more fair comparison, the average running time comparison of all competing meta-heuristic algorithms for each experiment is considered in this study. Tables 22 to 25 record the comparative results for the competing meta-heuristic algorithms that have been executed within the same environment. The results highlight the average running time in seconds. The bolded results represent the best average running time obtained. Overall, EPO outperforms all other competing meta-heuristic algorithms in terms of average running time

TABLE 21. Comparative time complexity.

Algorithm	Time Complexity	Remarks
MFO	$O(M_{itr} \times n \times (f_e + 1 + d + n))$	MFO requires sorting ($O(n^2)$ for worst-case) of flames in each iteration depending upon their fitness value.
SSA	$O(n^2 + M_{itr} \times n \times (f_e + 1 + d))$	Before entering the main loop, SSA requires that the locations of salps be sorted in ascending order based on their fitness value.
STOA	$O(M_{itr} \times n \times (f_e + 1 + d))$	STOA only performs a single agent update per iteration.
PSO	$O(M_{itr} \times n \times (f_e + 1 + d))$	PSO only performs a single agent update per iteration.
GA	$O(M_{itr} \times n \times (f_e + 1 + d))$	GA only performs a single agent update per iteration.
EPO	$O(M_{itr} \times n \times (2 \times f_e + 1 + d))$	The time complexity of EPO includes the multiplier two, which represents the number of fitness function evaluations done in each iteration.
IEPO	$O(M_{itr} \times n \times (f_e + 1 + d \times L))$	The time complexity of IEPO includes L, which represents the number of levy function evaluations done for each dimension in each iteration.
FAEPO	$O(M_{itr} \times n \times (f_e + 1 + d)) + r$	Unlike the original EPO, the FAEPO calculates the fitness of all search agents once in each iteration, resulting in time complexity improvement. However, the time complexity also contains the constant value, r, for fuzzy decision-making.

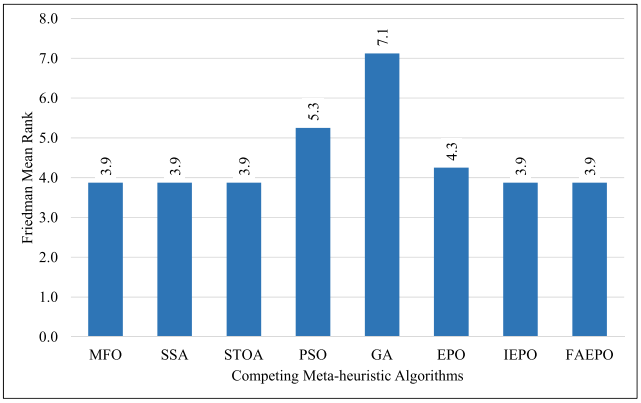


FIGURE 16. Friedman mean ranking for MC/DC test case generation.

in all experiments except MC/DC test case generation and TFO for the DBLP dataset. However, it fails to generate optimal results in any of the configurations for the given case studies. Although not the fastest, FAEPO outperforms the other strategies in terms of generating optimal results, better than EPO and IEPO in all experiments. On the other hand, the average running time of IEPO is higher than the other competing meta-heuristic algorithms and provides very poor results in all cases.

5) IS FAEPO SUFFICIENTLY GENERAL TO HANDLE BOTH MINIMIZATION AND MAXIMIZATION OPTIMIZATION PROBLEMS?

The answer to question 5 (RQ5) refers to the performance evaluation of FAEPO on three global optimization problems. Among the optimization problems, TFO and MC /DC test case generation are the minimization problems, while LABS is considered as a maximization problem in terms of finding the maximum merit factor of sequences. Unlike typical meta-heuristic algorithms that require presetting of control parameters, FAEPO tunes its control parameters using FIS based on the quality, success rate, and diversity of the current search. As a result, FAEPO becomes more general and can handle different optimization problems. The experimental results show that the proposed FAEPO outperforms the other competing meta-heuristic algorithms for both minimization and maximization optimization problems.

TABLE 22. Average running time comparison for benchmark test function experiment.

Function ID	D	MFO	SSA	STOA	PSO	GA	EPO	IEPO	FAEPO
F1	2	0.022	0.031	0.016	0.130	0.234	**0.015**	1.185	0.288
F2	2	0.029	0.041	0.020	0.154	0.270	**0.019**	1.288	0.334
F3	50	0.132	0.120	0.101	0.238	0.390	**0.099**	39.271	0.488
	70	0.155	0.133	0.122	0.218	0.336	**0.111**	42.390	0.457
	100	0.308	0.277	0.251	0.396	0.525	**0.221**	76.154	0.746
F4	50	0.074	0.064	0.076	0.166	0.319	**0.047**	37.053	0.358
	70	0.084	0.065	0.097	0.136	0.262	**0.056**	41.005	0.312
	100	0.159	0.111	0.184	0.235	0.359	**0.101**	80.138	0.517
F5	50	0.126	0.160	0.128	0.249	0.406	**0.074**	34.903	0.410
	70	0.161	0.204	0.187	0.265	0.412	**0.097**	43.054	0.416
	100	0.356	0.459	0.440	0.586	0.792	**0.243**	89.370	0.822
F6	50	0.070	0.061	0.072	0.164	0.309	**0.046**	35.569	0.351
	70	0.067	0.054	0.072	0.133	0.237	**0.045**	37.558	0.295
	100	0.137	0.097	0.160	0.191	0.309	**0.095**	60.427	0.447
F7	2	0.020	0.026	0.015	0.104	0.181	**0.014**	0.906	0.208
F8	2	0.023	0.032	0.018	0.117	0.236	**0.017**	1.184	0.266
F9	50	0.055	0.046	0.059	0.108	0.190	**0.037**	21.667	0.238
	70	0.085	0.071	0.087	0.159	0.270	**0.056**	40.892	0.342
	100	0.167	0.128	0.183	0.219	0.330	**0.104**	61.687	0.473
F10	50	0.139	0.129	0.144	0.222	0.392	**0.114**	39.333	0.512
	70	0.166	0.147	0.173	0.218	0.349	**0.137**	43.533	0.493
	100	0.287	0.255	0.313	0.335	0.465	**0.246**	59.178	0.768
F11	50	0.057	0.050	0.061	0.114	0.199	**0.044**	21.992	0.283
	70	0.078	0.066	0.082	0.144	0.250	**0.054**	38.224	0.327
	100	0.088	0.068	0.100	0.119	0.202	**0.064**	42.311	0.280
F12	50	0.122	0.112	0.127	0.189	0.316	**0.099**	30.087	0.418
	70	0.161	0.144	0.169	0.216	0.348	**0.132**	43.510	0.483
	100	0.409	0.357	0.448	0.494	0.668	**0.334**	95.691	1.027

VII. OVERALL OBSERVATIONS AND DISCUSSION ON FAEPO

The effectiveness of our algorithm can be further justified by considering the research questions raised earlier. As mentioned earlier, FAEPO is compared with seven other well-known meta-heuristic algorithms, including its prede-

TABLE 23. Average running time comparison for TFO experiment.

IMDB Dataset								
Number of Test Skills	MFO	SSA	STOA	PSO	GA	EPO	IEPO	FAEPO
5	0.166	0.139	0.141	0.141	0.140	**0.124**	42.637	0.259
10	0.100	0.081	0.081	0.089	0.089	**0.064**	38.397	0.134
15	0.122	0.100	0.102	0.098	0.103	**0.092**	39.002	0.168
20	0.087	0.071	0.073	0.078	0.080	**0.058**	38.475	0.134
25	0.117	0.097	0.098	0.101	0.104	**0.083**	39.516	0.148
DBLP Dataset								
Number of Test Skills	MFO	SSA	STOA	PSO	GA	EPO	IEPO	FAEPO
5	119.972	116.011	119.799	124.986	**87.799**	137.799	298.026	131.171
10	28.255	38.792	32.048	47.991	61.516	22.506	218.297	**20.562**
15	41.328	52.486	46.124	73.666	96.298	30.691	221.681	**28.327**
20	53.679	60.473	57.982	77.303	89.196	40.552	232.240	**34.562**
25	85.569	84.675	84.026	83.017	96.898	**69.403**	282.970	70.456

TABLE 24. Average running time comparison for LABS experiment.

Sequence Length (N)	MFO	SSA	STOA	PSO	GA	EPO	IEPO	FAEPO
7	0.072	0.062	0.063	0.208	0.329	**0.054**	4.685	0.408
8	0.081	0.074	0.070	0.230	0.358	**0.059**	5.894	0.429
9	0.081	0.068	0.067	0.219	0.353	**0.059**	6.460	0.425
10	0.083	0.073	0.070	0.223	0.361	**0.062**	7.154	0.429
11	0.085	0.072	0.070	0.228	0.357	**0.061**	7.734	0.432
12	0.087	0.075	0.074	0.228	0.361	**0.064**	8.552	0.441
13	0.089	0.074	0.075	0.227	0.356	**0.063**	9.245	0.435
14	0.090	0.074	0.079	0.233	0.375	**0.068**	10.122	0.451
15	0.089	0.073	0.073	0.217	0.345	**0.061**	10.725	0.412
16	0.086	0.071	0.073	0.209	0.327	**0.063**	10.456	0.407
17	0.086	0.073	0.076	0.213	0.329	**0.066**	11.055	0.400
18	0.091	0.079	0.084	0.235	0.371	**0.066**	13.310	0.427
19	0.163	0.135	0.141	0.379	0.590	**0.119**	22.253	0.758
20	0.097	0.077	0.081	0.206	0.336	**0.072**	12.995	0.391
21	0.089	0.073	0.081	0.208	0.327	**0.065**	13.733	0.384
22	0.089	0.073	0.081	0.208	0.327	**0.065**	13.733	0.384
23	0.089	0.073	0.081	0.208	0.327	**0.065**	13.733	0.384
24	0.112	0.090	0.100	0.252	0.392	**0.085**	18.752	0.476
25	0.173	**0.132**	0.154	0.391	0.641	0.140	29.307	0.708
26	0.099	0.078	0.090	0.217	0.338	**0.071**	16.855	0.400
27	0.119	0.094	0.109	0.246	0.399	**0.089**	20.566	0.484
28	0.172	0.132	0.156	0.350	0.543	**0.119**	30.230	0.727
29	0.110	0.087	0.100	0.227	0.361	**0.083**	19.818	0.437
30	0.112	0.087	0.099	0.228	0.348	**0.086**	19.789	0.448
31	0.112	0.091	0.100	0.220	0.344	**0.086**	20.719	0.412

TABLE 25. Average running time comparison for MC/DC test case generation experiment.

LE#	MFO	SSA	STOA	PSO	GA	EPO	IEPO	FAEPO
LE1	**3.735**	5.143	5.720	4.570	4.390	5.160	8.362	5.683
LE2	**2.668**	3.580	4.002	3.435	3.175	3.561	5.774	4.054
LE3	2.418	**2.411**	2.832	3.480	3.472	2.515	6.457	2.861
LE4	3.238	4.086	3.678	**2.478**	2.570	3.718	6.895	4.054
LE5	**5.130**	6.198	12.073	7.857	7.950	13.178	364.290	15.175
LE6	5.347	6.359	13.164	4.730	**4.272**	16.308	364.290	17.627
LE7	5.087	5.009	11.287	4.833	**4.495**	13.152	18.162	13.523
LE8	70.378	82.034	332.950	48.161	**32.293**	610.184	546.419	1500.0

cessor EPO and its improved variant IEPO. The value of the default/common parameters was kept identical to make a fair comparison between the competing meta-heuristic algorithms during the experiments, as listed in Table 21. Based on the obtained results of all experiments and the above discussions, some thought-provoking observations can be made, which are explained below.

- Our analysis of the overall ranking results shows that FAEPO can maintain an appropriate balance between exploration and exploitation processes. Unlike the original EPO, where exploration was performed by predefining control parameters, FAEPO can use the FIS to tune the control parameters. However, FAEPO outperforms EPO and other competing meta-heuristic algorithms due to its adaptive behavior, which is the major factor for FAEPO's superior performance. Statistical analysis shows that FAEPO outperforms all other meta-heuristic algorithms (see Tables 12 and 13), except for its predecessor EPO, which outperforms all other meta-heuristic algorithms in only one case (see Table 11).
- The benchmarking experiments show that the integration of FIS into FAEPO has improved its exploration and exploitation capabilities. By breaking the default of control operators, fuzzy rules make the exploration and exploitation of FAEPO more dynamic. Unlike typical meta-heuristic algorithms, FAEPO behaves like a parameterized algorithm without requiring significant tuning.
- When comparing the two algorithms, there are no significant differences between FAEPO and EPO in terms of convergence across different experiments. In fact, our experimental results show that both FAEPO and EPO can achieve convergence in fewer iterations than the other competing meta-heuristic algorithms.
- It is convenient to use Big O notation to compare the performance of each meta-heuristic algorithm considered in terms of time complexity. In most cases, the time complexity is determined by the number of search agents (n), the number of dimensions (d), the number of maximum iterations (M_{itr}) and the evaluation of the fitness function (f_e). According to Eq. (31), the time complexity of all competing meta-heuristic algorithms can be summarized in Table 21. When n is sufficiently large, the time complexity of all meta-heuristic algorithms in Table 21 can be

approximated as $\approx O(M_{itr} \times n \times (f_e + d))$, assuming that all other operations can be performed in a constant time. From this approximation, it can be deduced that the introduction of FIS within FAEPO does not directly degrade the performance of the system.

- With respect to the fairness of comparisons with other meta-heuristic algorithms, several issues can be discussed in more detail. First, instead of using the maximum number of iterations, our research chose the maximum fitness evaluation as the stopping criterion, while maintaining the same population size. This choice ensures that each algorithm performs the same number of agent updates regardless of the iteration number. Second, implementation efficiency, choice of language implementation, and data structure directly affect the runtime performance of an algorithm. For this reason, we have considered the average running time comparison of the competing meta-heuristic algorithms, and all of them are implemented within the same environment. Even though it's not the fastest, FAEPO is better than the other meta-heuristic algorithms because it always gives the best results. Finally, we focused our Friedman Mean Rank analysis on the mean results rather than the best. The performance of all meta-heuristic algorithms can be affected by chance, since they all use randomness to update their solutions (through trial and error). By chance, one algorithm may produce the optimal result once, but produces suboptimal results on subsequent runs. Therefore, using the mean instead of the best results can give a more accurate indication of performance.
- Our design decisions can affect our FAEPO performance in terms of fuzzy design on the subject of constraints. We chose a Mamdani-based approach over a Sugeno-based approach and a triangular/trapezoidal membership function over the Gaussian membership function due to the simplicity of maintaining the fuzzy rules. We also opted for three and six overlapping linguistic terms for each input and output membership function, respectively (see Figure 3 and Figure 4). In addition, our design decisions highlight some limitations of our approach. First, the choice of linguistic terms can be described from two opposing perspectives. On the one hand, an excessive number of linguistic terms requires additional rules that require additional computations. On the other hand, an insufficient number of linguistic terms cannot meet the requirements of the problem.
- In terms of our design decisions, we have created a fuzzy FIS. However, other approaches may be more effective. It is not guaranteed that our choices are sufficiently general for other optimization problems. Different design decisions (e.g., Mamdani versus Sugeno approach, Gaussian versus Trapezoidal membership function) may lead to different results.

VIII. CONCLUSION AND FUTURE WORKS

In this study, an adaptive variant of the EPO (i.e., FAEPO) is proposed for solving global optimization problems by tuning its control parameters. The proposed FAEPO is a combination of the EPO algorithm and the Mamdani fuzzy interference system for adaptive tuning of two control parameters based on setting their range of values by a fuzzy decision process. In order to perform a comprehensive evaluation, the proposed algorithm is applied to twelve optimization benchmark test functions consisting of unimodal, multimodal, and three global optimization problems (TFO, LABS and MC/DC). Based on the experimental findings obtained, FAEPO outperforms not only its predecessor (i.e., EPO) but also other competing meta-heuristic algorithms for all experiments conducted in this study. Moreover, we concluded that FAEPO is sufficiently general and can be applied to other optimization problems. Finally, it is important to point out that our FIS approach can also be applied to other meta-heuristic algorithms such as the Aquila Optimizer (AO) [7], Cat Swarm Optimization (CSO) [11], Orca Predation Algorithm (OPA) [103], and Mayfly Optimization Algorithm (MOA) [104] to solve the optimization problems highlighted in this paper. Moreover, instead of dealing with the tuning of the control operators, as proposed in this work, the FIS can automatically select the search operators while the algorithm is running. Meta-heuristic algorithms need more work to make them even better at what they do. Here are some possible directions for research that could help to improve the performance of FAEPO:

- FAEPO can be combined with other algorithms like PSO, GWO, and EO to create a hybrid algorithm.
- FAEPO can be extended for solving multi objective and many objective optimization problems.
- FAEPO can be enhanced for feature selection problems as a binary variant.
- FAEPO can be adopted to solve a wide variety of constrained and unconstrained real-world optimization problems.
- FAEPO can be combined with chaotic maps and levy flight.

REFERENCES

[1] D. Prayogo, M.-Y. Cheng, Y.-W. Wu, A. A. N. P. Redi, V. F. Yu, S. F. Persada, and R. Nadlifatin, "A novel hybrid Metaheuristic algorithm for optimization of construction management site layout planning," *Algorithms*, vol. 13, no. 5, p. 117, May 2020.

[2] R. K. Chandrawat, R. Kumar, B. Garg, G. Dhiman, and S. Kumar, "An analysis of modeling and optimization production cost through fuzzy linear programming problem with symmetric and right angle triangular fuzzy number," in *Proc. 6th Int. Conf. Soft Comput. Problem Solving*. Singapore: Springer, 2017, pp. 197–211.

[3] P. Singh and G. Dhiman, "A fuzzy-LP approach in time series forecasting," in *Proc. Int. Conf. Pattern Recognit. Mach. Intell.*, vol. 10597. Cham, Switzerland: Springer, 2017, pp. 243–253.

[4] K. Z. Zamli, B. S. Ahmed, T. Mahmoud, and W. Afzal, "Fuzzy adaptive tuning of a particle swarm optimization algorithm for variable-strength combinatorial test suite generation," in *Swarm Intelligence: Applications* (Control, Robotics and Sensors), vol. 3. Institution of Engineering and Technology, 2018, pp. 639–662, doi: 10.1049/PBCE119H_ch22.

[5] A. A. Hassan, S. Abdullah, K. Z. Zamli, and R. Razali, "Combinatorial test suites generation strategy utilizing the whale optimization algorithm," *IEEE Access*, vol. 8, pp. 192288–192303, 2020.
[6] G. Dhiman and V. Kumar, "Emperor penguin optimizer: A bio-inspired algorithm for engineering problems," *Knowl. Based Syst.*, vol. 159, pp. 20–50, Nov. 2018.
[7] L. Abualigah, D. Yousri, M. Abd Elaziz, A. A. Ewees, M. A. A. Al-Qaness, and H. A. Gandomi, "Aquila optimizer: A novel meta-heuristic optimization algorithm," *Comput. Ind. Eng.*, vol. 157, pp. 1–37, Jul. 2021.
[8] I. Naruei and F. Keynia, "A new optimization method based on coot bird natural life model," *Expert Syst. Appl.*, vol. 183, pp. 1–25, Nov. 2021.
[9] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, "Salp swarm algorithm: A bio-inspired optimizer for engineering design problems," *Adv. Eng. Softw.*, vol. 114, pp. 163–191, Dec. 2017.
[10] L. Wang, X. Zhang, and X. Zhang, "Antenna array design by artificial bee colony algorithm with similarity induced search method," *IEEE Trans. Magn.*, vol. 55, no. 6, pp. 1–4, Jun. 2019.
[11] N. Y. Et. al., "Cat swarm optimization algorithm for antenna array synthesis," *Turkish J. Comput. Math. Educ.*, vol. 12, no. 2, pp. 1466–1474, Apr. 2021.
[12] A. Durmus, "The concentric elliptical antenna array patterns synthesis using marine predators algorithm," *Arabian J. Sci. Eng.*, vol. 46, no. 10, pp. 9485–9495, Oct. 2021.
[13] R. Habachi, A. Touil, A. Charkaoui, and A. Echchatbi, "Eagle strategy based crow search algorithm for solving unit commitment problem in smart grid system," *Indonesian J. Electr. Eng. Comput. Sci.*, vol. 12, no. 1, pp. 17–29, 2018.
[14] A. Askarzadeh, "A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm," *Comput. Struct.*, vol. 169, pp. 1–12, Jun. 2016.
[15] M. Jain, S. Maurya, A. Rani, and V. Singh, "Owl search algorithm: A novel nature-inspired heuristic paradigm for global optimization," *J. Intell. Fuzzy Syst.*, vol. 34, no. 3, pp. 1573–1582, 2018.
[16] G. Dhiman and A. Kaur, "STOA: A bio-inspired based optimization algorithm for industrial engineering problems," *Eng. Appl. Artif. Intell.*, vol. 82, pp. 148–174, Jun. 2019.
[17] M. Jain, V. Singh, and A. Rani, "A novel nature-inspired algorithm for optimization: Squirrel search algorithm," *Swarm Evol. Comput.*, vol. 44, pp. 148–175, Feb. 2018.
[18] F. Din and K. Z. Zamli, "Fuzzy adaptive teaching learning-based optimization strategy for pairwise testing," in *Proc. 7th IEEE Int. Conf. Syst. Eng. Technol. (ICSET)*, Oct. 2017, pp. 17–22.
[19] S. Ganesh, V. Vengatesan, J. A. R. Jimreeves, and B. Ramasubramanian, "Simultaneous network reconfiguration and pmu placement in the radial distribution system," *Adv. Math., Sci. J.*, vol. 9, no. 10, pp. 8143–8151, Sep. 2020.
[20] H. Shingrakhia and H. Patel, "Emperor penguin optimized event recognition and summarization for cricket highlight generation," *Multimedia Syst.*, vol. 26, no. 6, pp. 745–759, Dec. 2020.
[21] D. Mehta and S. Saxena, "Hierarchical WSN protocol with fuzzy multi-criteria clustering and bio-inspired energy-efficient routing (FMCB-ER)," *Multimedia Tools Appl.*, vol. 81, pp. 1–34, Sep. 2020.
[22] D. Pandey, B. K. Pandey, and S. Wairya, "Hybrid deep neural network with adaptive galactic swarm optimization for text extraction from scene images," *Soft Comput.*, vol. 25, no. 2, pp. 1–18, 2020.
[23] S. L. Tade and V. Vyas, "Hybrid deep emperor penguin classifier algorithm-based image quality assessment for visualisation application in HDR environments," *IET Image Process.*, vol. 14, no. 11, pp. 2579–2587, Sep. 2020.
[24] M. Singh, B. M. Mehtre, and S. Sangeetha, "Insider threat detection based on user behaviour analysis," *Commun. Comput. Inf. Sci.*, vol. 1241, pp. 559–574, Jul. 2020.
[25] K. Cheena, T. Amgoth, and G. Shankar, "Emperor penguin optimised self-healing strategy for WSN based smart grids," *Int. J. Sensor Netw.*, vol. 32, no. 2, pp. 87–95, 2020.
[26] P. Shrivastava, "EPO: An optimization technique for urban traffic management while limiting the pollution using WSN," *Int. J. Commun. Syst.*, vol. 33, no. 5, pp. 1–14, 2020.
[27] J. S. P. Dharshini and M. V. Subramanyam, "Emperor penguin optimized user association scheme for MMWAVE wireless communication," *Wireless Pers. Commun.*, vol. 113, no. 2, pp. 1097–1113, Jul. 2020.
[28] M. A. Kader, K. Z. Zamli, and B. S. Ahmed, "A systematic review on emperor penguin optimizer," *Neural Comput. Appl.*, vol. 33, no. 23, pp. 1–21, 2021.
[29] Z. Xing, "An improved emperor penguin optimization based multi-level thresholding for color image segmentation," *Knowl.-Based Syst.*, vol. 194, pp. 1–20, Apr. 2020.
[30] S. Min, Z. Tang, and B. D. Rouyendegh, "Inspired-based optimisation algorithm for solving energy-consuming reduction of chiller loading," *Int. J. Ambient Energy*, vol. 43, pp. 1–11, Dec. 2020.
[31] F. Tang, J. Li, and N. Zafetti, "Optimization of residential building envelopes using an improved emperor penguin optimizer," *Eng. Comput.*, vol. 38, pp. 1–13, Jul. 2020.
[32] D. L. Bhuyar and A. K. Kureshi, "EPOWT: A denoising technique of the electrocardiography signal transmission via 5G wireless communications," *Trans. Emerg. Telecommun. Technol.*, vol. 31, no. 3, pp. 1–17, Mar. 2020.
[33] H. Jia, K. Sun, W. Song, X. Peng, C. Lang, and Y. Li, "Multi-strategy emperor penguin optimizer for RGB histogram-based color satellite image segmentation using Masi entropy," *IEEE Access*, vol. 7, pp. 134448–134474, 2019.
[34] S. K. Baliarsingh, W. Ding, S. Vipsita, and S. Bakshi, "A memetic algorithm using emperor penguin and social engineering optimization for medical data classification," *Appl. Soft Comput.*, vol. 85, pp. 1–15, Dec. 2019.
[35] G. Dhiman, "ESA: A hybrid bio-inspired metaheuristic optimization approach for engineering problems," *Eng. Comput.*, vol. 37, pp. 1–31, Jan. 2019.
[36] G. Dhiman, "MOSHEPO: A hybrid multi-objective approach to solve economic load dispatch and micro grid problems," *Int. J. Speech Technol.*, vol. 50, no. 1, pp. 119–137, Jan. 2020.
[37] G. Dhiman and M. Garg, "MoSSE: A novel hybrid multi-objective meta-heuristic algorithm for engineering design problems," *Soft Comput.*, vol. 24, no. 24, pp. 18379–18398, Dec. 2020.
[38] J. Yang and H. Gao, "Cultural emperor penguin optimizer and its application for face recognition," *Math. Problems Eng.*, vol. 2020, pp. 1–16, Nov. 2020.
[39] Y. Cao, Y. Wu, L. Fu, K. Jermsittiparsert, and N. Razmjooy, "Multi-objective optimization of a PEMFC based CCHP system by meta-heuristics," *Energy Rep.*, vol. 5, pp. 1551–1559, Nov. 2019.
[40] D. Kumar, V. Kumar, and R. Kumari, "Automatic clustering using quantum-based multi-objective emperor penguin optimizer and its applications to image segmentation," *Modern Phys. Lett. A*, vol. 34, no. 24, pp. 1–18, 2019.
[41] H. Kaur, A. Rai, S. S. Bhatia, and G. Dhiman, "MOEPO: A novel multi-objective emperor penguin optimizer for global optimization: Special application in ranking of cloud service providers," *Eng. Appl. Artif. Intell.*, vol. 96, pp. 1–21, Nov. 2020.
[42] M. Naresh, D. V. Reddy, and K. R. Reddy, "Multi-objective emperor penguin handover optimisation for IEEE 802.21 in heterogeneous networks," *IET Commun.*, vol. 14, no. 18, pp. 3239–3246, Nov. 2020.
[43] S. K. Baliarsingh, S. Vipsita, K. Muhammad, and S. Bakshi, "Analysis of high-dimensional biomedical data using an evolutionary multi-objective emperor penguin optimizer," *Swarm Evol. Comput.*, vol. 48, pp. 262–273, Aug. 2019.
[44] S. K. Baliarsingh and S. Vipsita, "Chaotic emperor penguin optimised extreme learning machine for microarray cancer classification," *IET Syst. Biol.*, vol. 14, no. 2, pp. 85–95, Apr. 2020.
[45] L. A. Zadeh, "Fuzzy sets," *Inf. Control*, vol. 8, no. 3, pp. 338–353, Jun. 1965.
[46] J. S. R. Jang, C. T. Sun, and E. Mizutani, "Neuro-fuzzy and soft computing—A computational approach to learning and machine intelligence," *IEEE Trans. Autom. Control*, vol. 42, no. 10, pp. 1482–1484, Oct. 1997.
[47] L. A. Zadeh, "Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic," *Fuzzy Sets Syst.*, vol. 90, pp. 111–127, Sep. 1997.
[48] K. Z. Zamli, F. Din, S. Baharom, and B. S. Ahmed, "Fuzzy adaptive teaching learning-based optimization strategy for the problem of generating mixed strength t-way test suites," *Eng. Appl. Artif. Intell.*, vol. 59, pp. 35–50, Mar. 2017.
[49] W. E. Sari, O. Wahyunggoro, and S. Fauziati, "A comparative study on fuzzy Mamdani-Sugeno-Tsukamoto for the childhood tuberculosis diagnosis," in *Proc. Amer. Inst. Phys. Conf.*, vol. 1755, 2016, pp. 1–28.

[50] S. Napitupulu, E. B. Nababan, and P. Sihombing, "Comparative analysis of fuzzy inference Tsukamoto Mamdani and Sugeno in the horticulture export selling price," in *Proc. 3rd Int. Conf. Mech., Electron., Comput., Ind. Technol. (MECnIT)*, Jun. 2020, pp. 183–187.

[51] E. Sonalitha, B. Nurdewanto, S. Ratih, N. R. Sari, A. B. Setiawan, and P. Tutuko, "Comparative analysis of Tsukamoto and Mamdani fuzzy inference system on market matching to determine the number of exports for MSMEs," in *Proc. Electr. Power, Electron., Commun., Controls Informat. Seminar (EECCIS)*, Oct. 2018, pp. 440–445.

[52] M. Irfan, L. P. Ayuningtias, and J. Jumadi, "Analisa perbandingan logic fuzzy metode tsukamoto, sugeno, Dan Mamdani (Studi kasus: Prediksi jumlah pendaftar mahasiswa Baru fakultas sains Dan teknologi universitas Islam Negeri sunan gunung djati Bandung)," *Jurnal Teknik Informatika*, vol. 10, no. 1, pp. 9–16, 2017.

[53] H. H. Hoos, *Automated Algorithm Configuration and Parameter Tuning*. Berlin, Germany: Springer, 2012, pp. 37–71.

[54] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems," *Comput.-Aided Des.*, vol. 43, no. 3, pp. 303–315, Mar. 2011.

[55] R. V. Rao, "Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems," *Int. J. Ind. Eng. Comput.*, vol. 7, no. 1, pp. 19–34, 2016.

[56] M.-Y. Cheng and D. Prayogo, "Symbiotic organisms search: A new metaheuristic optimization algorithm," *Comput. Struct.*, vol. 139, pp. 98–112, Jul. 2014.

[57] C. Huang, Y. Li, and X. Yao, "A survey of automatic parameter tuning methods for metaheuristics," *IEEE Trans. Evol. Comput.*, vol. 24, no. 2, pp. 201–216, Apr. 2020.

[58] E. S. Skakov and V. N. Malysh, "Parameter meta-optimization of metaheuristics of solving specific np-hard facility location problem," *J. Phys., Conf.*, vol. 973, pp. 2–13, Mar. 2018.

[59] G. Karafotias, M. Hoogendoorn, and A. E. Eiben, "Parameter control in evolutionary algorithms: Trends and challenges," *IEEE Trans. Evol. Comput.*, vol. 19, no. 2, pp. 167–187, Apr. 2015.

[60] Y. Shi and R. C. Eberhart, "Fuzzy adaptive particle swarm optimization," in *Proc. Congr. Evol. Comput.*, vol. 1, May 2001, pp. 101–106.

[61] H. Liu, A. Abraham, and W. Zhang, "A fuzzy adaptive turbulent particle swarm optimisation," *Int. J. Innov. Comput. Appl.*, vol. 1, no. 1, pp. 39–47, 2007.

[62] Y.-T. Juang, S.-L. Tung, and H.-C. Chiu, "Adaptive fuzzy particle swarm optimization for global optimization of multimodal functions," *Inf. Sci.*, vol. 181, no. 20, pp. 4539–4549, Oct. 2011.

[63] Y. Liu and L. Ma, "Solving TSP by fuzzy particle swarm algorithm," in *Proc. Int. Conf. Bus. Manage. Electron. Inf.*, vol. 5, 2011, pp. 202–204.

[64] T. Niknam, E. Azadfarsani, and M. Jabbari, "A new hybrid evolutionary algorithm based on new fuzzy adaptive PSO and NM algorithms for distribution feeder reconfiguration," *Energy Convers. Manage.*, vol. 54, no. 1, pp. 7–16, Feb. 2012.

[65] P. Melin, F. Olivas, O. Castillo, F. Valdez, J. Soria, and M. Valdez, "Optimal design of fuzzy classification systems using PSO with dynamic parameter adaptation through fuzzy logic," *Expert Syst. Appl.*, vol. 40, no. 8, pp. 3196–3206, Jun. 2013.

[66] F. Olivas, F. Valdez, O. Castillo, and P. Melin, "Dynamic parameter adaptation in particle swarm optimization using interval type-2 fuzzy logic," *Soft Comput.*, vol. 20, no. 3, pp. 1057–1070, 2016.

[67] T. Mahmoud and B. S. Ahmed, "An efficient strategy for covering array construction with fuzzy logic-based adaptive swarm optimization for software testing use," *Expert Syst. Appl.*, vol. 42, no. 22, pp. 8753–8765, 2015.

[68] H. Liu, Z. Xu, and A. Abraham, "Hybrid fuzzy-genetic algorithm approach for crew grouping," in *Proc. 5th Int. Conf. Intell. Syst. Design Appl. (ISDA)*, 2005, pp. 332–337.

[69] Z. Qi and P. Chunming, "An improved fuzzy genetic algorithm with fuzzy adjusted crossover and mutation probabilities," in *Proc. 3rd Int. Conf. Adv. Comput. Theory Eng. (ICACTE)*, Aug. 2010, pp. 581–585.

[70] A. Sombra, F. Valdez, P. Melin, and O. Castillo, "A new gravitational search algorithm using fuzzy logic to parameter adaptation," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2013, pp. 1068–1074.

[71] H. Neyoy, O. Castillo, and J. Soria, *Dynamic Fuzzy Logic Parameter Tuning for ACO and its Application in TSP Problems*, vol. 451. Berlin, Germany: Springer, 2013, pp. 259–271.

[72] M. Bidar and H. Rashidy Kanan, "Modified firefly algorithm using fuzzy tuned parameters," in *Proc. 13th Iranian Conf. Fuzzy Syst. (IFSC)*, Aug. 2013, pp. 1–4.

[73] C. Peraza, F. Valdez, M. Garcia, P. Melin, and O. Castillo, "A new fuzzy harmony search algorithm using fuzzy logic for dynamic parameter adaptation," *Algorithms*, vol. 9, no. 4, pp. 1–19, 2016.

[74] F. Valdez, O. Castillo, and C. Peraza, "Fuzzy logic in dynamic parameter adaptation of harmony search optimization for benchmark functions and fuzzy controllers," *Int. J. Fuzzy Syst.*, vol. 22, no. 4, pp. 1198–1211, Jun. 2020.

[75] P. Ochoa, O. Castillo, and J. Soria, *Differential Evolution Using Fuzzy Logic and a Comparative Study With Other Metaheuristics*, vol. 667. Cham, Switzerland: Springer, 2017, pp. 257–268.

[76] L. Valenzuela, F. Valdez, and P. Melin, *Flower Pollination Algorithm With Fuzzy Approach for Solving Optimization Problems*, vol. 667. Cham, Switzerland: Springer, 2017, pp. 357–369.

[77] E. Bernal, O. Castillo, J. Soria, and F. Valdez, "Imperialist competitive algorithm with dynamic parameter adaptation using fuzzy logic applied to the optimization of mathematical functions," *Algorithms*, vol. 10, no. 1, p. 18, Jan. 2017.

[78] O. Castillo and L. Amador-Angulo, "A generalized type-2 fuzzy logic approach for dynamic parameter adaptation in bee colony optimization applied to fuzzy controller design," *Inf. Sci.*, vols. 460–461, pp. 476–496, Sep. 2018.

[79] A. Saffari, M. Khishe, and S.-H. Zahiri, "Fuzzy-ChOA: An improved chimp optimization algorithm for marine mammal classification using artificial neural network," *Anal. Integr. Circuits Signal Process.*, vol. 111, no. 3, pp. 403–417, Jun. 2022.

[80] P. Melin, I. Miramontes, O. Carvajal, and G. Prado-Arechiga, "Fuzzy dynamic parameter adaptation in the bird swarm algorithm for neural network optimization," *Soft Comput.*, vol. 26, no. 18, pp. 9497–9514, Sep. 2022.

[81] M.-Y. Cheng and D. Prayogo, "Fuzzy adaptive teaching–learning-based optimization for global numerical optimization," *Neural Comput. Appl.*, vol. 29, no. 2, pp. 309–327, Jan. 2018.

[82] G. L. Kooyman and T. G. Kooyman, "Diving behavior of emperor penguins nurturing chicks at coulman island, Antarctica," *Condor*, vol. 97, no. 2, pp. 536–549, May 1995.

[83] A. Waters, F. Blanchette, and A. D. Kim, "Modeling huddling penguins," *PLoS ONE*, vol. 7, no. 11, pp. 1–8, 2012.

[84] F. Camastra, A. Ciaramella, V. Giovannelli, M. Lener, V. Rastelli, A. Staiano, G. Staiano, and A. Starace, "A fuzzy decision system for genetically modified plant environmental risk assessment using Mamdani inference," *Expert Syst. Appl.*, vol. 42, no. 3, pp. 1710–1716, Feb. 2015.

[85] O. Cordón, "A historical review of evolutionary learning methods for mamdani-type fuzzy rule-based systems: Designing interpretable genetic fuzzy systems," *Int. J. Approx. Reasoning*, vol. 52, no. 6, pp. 894–913, Sep. 2011.

[86] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," *Knowl.-Based Syst.*, vol. 89, pp. 228–249, Nov. 2015.

[87] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Boston, MA, USA: Addison-Wesley, 1989.

[88] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE ICNN*, vol. 4, Nov./Dec. 1995, pp. 1942–1948.

[89] F. Tang, J. Li, and N. Zafetti, "Optimization of residential building envelopes using an improved emperor penguin optimizer," *Eng. Comput.*, vol. 38, no. 2, pp. 1395–1407, Apr. 2022.

[90] X.-S. Yang, *Test Problems in Optimization* (Engineering Optimization: An Introduction with Metaheuristic Applications). Hoboken, NJ, USA: Wiley, 2010.

[91] M. Golay, "The merit factor of long low autocorrelation binary sequences (corresp.)," *IEEE Trans. Inf. Theory*, vol. IT-28, no. 3, pp. 543–549, May 1982.

[92] K. Z. Zamli, A. A. Al-Sewari, and M. H. M. Hassin, "On test case generation satisfying the MC/DC criterion," *Int. J. Adv. Soft Comput. Appl.*, vol. 5, no. 3, pp. 104–115, 2013.

[93] A. Haque, I. Khalil, and K. Z. Zamli, "An automated tool for MC/DC test data generation," in *Proc. IEEE Symp. Comput. Informat.*, Jun. 2014, pp. 1–4.

[94] K. J. Hayhurst and D. S. Veerhusen, "A practical approach to modified condition/decision coverage," in *Proc. 20th Digit. Avionics Syst. Conf.*, 2001, pp. 1–10.

[95] (May 2019). *IMDB Dataset*. [Online]. Available: https://github.com/MAK660/Dataset/blob/master/IMDB_DataSet.txt
[96] X. Wang, Z. Zhao, and W. Ng, "A comparative study of team formation in social networks," in *Database Systems for Advanced Applications*, M. Renz, C. Shahabi, X. Zhou, and M. A. Cheema, Eds. Cham, Switzerland: Springer, 2015, pp. 389–404.
[97] (May 2019). *DBLP Dataset*. [Online]. Available: https://github.com/MAK660/Dataset/blob/master/DBLP_DataSet.txt
[98] T. Packebusch and S. Mertens, "Low autocorrelation binary sequences," *J. Phys. A, Math. Theor.*, vol. 49, no. 16, pp. 1–18, 2016.
[99] J. E. Gallardo, C. Cotta, and A. J. Fernández, "Finding low autocorrelation binary sequences with memetic algorithms," *Appl. Soft Comput.*, vol. 9, no. 4, pp. 1252–1262, Sep. 2009.
[100] Z. Awedikian, "Automatic data generation for MC/DC test criterion using metaheuristic algorithms," M.S. thesis, Dèpartement De Génie Informatique Et Génie Logiciel, École Polytechnique de Montréal, Montreal, QC, Canada, 2009.
[101] A. Gotlieb, "TCAS software verification using constraint programming," *Knowl. Eng. Rev.*, vol. 27, no. 3, pp. 343–360, 2012.
[102] T. Kitamura, Q. Maissonneuve, E.-H. Choi, C. Artho, and A. Gargantini, "Optimal test suite generation for modified condition decision coverage using SAT solving," in *Computer Safety, Reliability, and Security*, B. Gallina, A. Skavhaug, and F. Bitsch, Eds. Cham, Switzerland: Springer, 2018, pp. 123–138.
[103] Y. Jiang, Q. Wu, S. Zhu, and L. Zhang, "Orca predation algorithm: A novel bio-inspired algorithm for global optimization problems," *Expert Syst. Appl.*, vol. 188, pp. 1–48, Feb. 2021.
[104] K. Zervoudakis and S. Tsafarakis, "A mayfly optimization algorithm," *Comput. Ind. Eng.*, vol. 145, pp. 1–23, Jul. 2020.

MD. ABDUL KADER received the bachelor's degree in computer science and engineering from the Khulna University of Engineering and Technology (KUET), Bangladesh, in 2009, and the M.Sc. degree in image processing from Universiti Malaysia Perlis (UniMAP), Malaysia, in 2012. He is currently pursuing the Ph.D. degree in computational intelligence with the Faculty of Computing, Universiti Malaysia Pahang (UMP), Malaysia. He has more than nine years teaching and research experience in different Universities. His research interests include computational intelligence, image processing, optimization, software engineering, and data science.

KAMAL Z. ZAMLI (Member, IEEE) received the degree in electrical engineering from the Worcester Polytechnic Institute, Worcester, MA, USA, in 1992, the M.Sc. degree in real-time software engineering from Universiti Teknologi Malaysia, in 2000, and the Ph.D. degree in software engineering from the University of Newcastle upon Tyne, U.K., in 2003. His research interests include (combinatorial t-way) software testing and search-based software engineering.

BASEM YOUSEF ALKAZEMI (Senior Member, IEEE) received the Ph.D. degree from Newcastle University, U.K. He is currently a Professor with the College of Computer and Information System, Umm Al-Qura University, Saudi Arabia. He is also appointed as the Dean with the Deanship of Scientific Research (DSR), Umm Al-Qura University. He supervised several postgraduate students those conducted their theses in software continuous delivery (CD), BPM, the IoT, big data for retails, machine translation, and machine learning. His main research interests include software engineering, data mining, big data, and machine learning. He served as a reviewer in a number of international conferences and reputable journals.