Received 30 January 2024, accepted 9 February 2024, date of publication 13 February 2024, date of current version 21 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3365734

RESEARCH ARTICLE

Preference-Based Stepping Ahead Firefly Algorithm for Solving Real-World Uncapacitated Examination Timetabling Problem

RAVNEIL NAND, (Member, IEEE), EMMENUAL REDDY, (Member, IEEE), KAYLASH CHAUDHARY, (Member, IEEE), AND BIBHYA SHARMA, (Member, IEEE)
School of Information Technology, Engineering, Mathematics and Physics, The University of the South Pacific, Suva, Fiji
Corresponding author: Ravneil Nand (ravneil.nand@usp.ac.fj)

ABSTRACT Swarm-based intelligent optimization algorithms that employ the principles of collective behavior have been gaining traction as viable solutions in optimization research. One area of optimization is the Examination Timetabling Problem (ETP), which presents a significant challenge for many Higher Education Institutes (HEIs). This study proposes a novel approach for solving the Uncapacitated Examination Timetabling Problem (UETP), where a stepping-ahead mechanism is utilized with threshold acceptance in the Firefly Algorithm (FA). The proposed method improves exploration with the use of the stepping-ahead mechanism, while threshold acceptance allows for better exploitation of the search space. Initially, a neighborhood search mechanism is employed as the discretization of FA to improve solution quality, known as Kempe Chain-based neighborhoods. The proposed method is tested on seven UETP problems, with the results showing comparative performance to the best solutions available in the literature for the Toronto exam timetabling dataset. The selection of seven problems is made with exams totaling less than 400, this allows to create a manageable yet representative benchmark. The study further extends the experiment to a real-world dataset collected from an HEI. The use of a real-world dataset allows us to see the potential of the algorithm and at the same time evaluate its performance under realistic conditions and resource constraints. The proposed stepping-ahead mechanism has the potential for use in other domains, such as robotics and engineering. Overall, this paper presents a new methodology for solving the UETP that has the potential to offer superior results when compared to existing approaches.

INDEX TERMS Firefly algorithm, optimization, swarm intelligence, uncapacitated examination timetabling problem.

I. INTRODUCTION

Numerous challenges are encountered in daily livelihood, and one such domain pertains to scheduling. In scheduling, the arrangement of task sets is orchestrated in a manner that seeks to achieve optimal or highly effective outcomes [1]. Various methodologies exist for addressing this challenge; however, a particularly promising avenue lies within the realm of optimization techniques. The application of optimization techniques has been instrumental in addressing functional objectives to achieve optimality or near-optimality [2]. Optimization involves identifying global solutions based on the objectives and any accompanying constraints on the problem [3]. However, tackling problems with multiple objectives introduces an additional layer of complexity [4], as optimization must consider the simultaneous optimization of several objectives, leading to significant shifts in the optimal solution region.

In real-world problem-solving, optimization plays a vital role in addressing complex challenges across various domains, ranging from time-series prediction to devising efficient solutions [4], [5], [6]. Particularly in the education

The associate editor coordinating the review of this manuscript and approving it for publication was Adamu Murtala Zungeru.

sector, optimal solutions are essential for diverse tasks, such as optimizing website response rates [7] or enhancing student performance [8], [9]. In Higher Education Institutes (HEIs), scheduling is a critical area that necessitates optimization, especially in the context of timetabling. Timetabling involves efficiently arranging exams within specific time slots or periods [10]. This challenge extends beyond HEIs to the transportation industry, sport organizations, the health sector, and the aviation industry, all of which require allocating limited resources to specific time slots based on organizational needs and operational rules [11], [12], [13].

The Examination Timetabling Problem (ETP) is a dynamic scheduling challenge that demands an even distribution of exams within available time slots while considering the student workload and adhering to constraints [14], [15]. Solving the ETP is highly complex because of the presence of both hard constraints, which cannot be violated (e.g., room capacity), and soft constraints, which are desirable but can be violated (e.g., two exams on the same day for a student). Researchers have explored various methods, including graph coloring techniques, to address the ETP [16], [17]. In recent years, evolutionary algorithms have emerged as a powerful alternative to traditional graph coloring techniques for solving the ETP, leveraging concepts from natural selection to optimize exam schedules [18], [19], [20], [21]. An evolutionary algorithm that is simple in structure and has the ability to improve exploration and exploitation is the Firefly Algorithm (FA). While FA is a popular swarm intelligence algorithm, its application in the discrete domain, particularly for solving ETP, remains less explored. Some applications of discrete FA have been seen in the vehicle routing problem [22], the traveling salesman problem [23], etc. This has provided motivation to utilize FA in solving ETP, where the problem is an NP-hard problem and real-world application is possible.

This research is an extension of [21], where a modified version of discrete FA, termed the 'Preference-Based Stepping Ahead Firefly Algorithm,' was introduced to the Uncapacitated Examination Timetabling Problem (UETP). It extends the application to a real-world problem together with encroachments to the proposed method using threshold acceptance. The proposed algorithm is coined modified Preference-Based Discrete Stepping Ahead FA (mdFA-Step). It integrates a stepping-ahead mechanism to enhance exploration capabilities and conduct structured searches in the neighborhoods accepted via threshold. The 7 problems of benchmark data set are selected based on exams totalling less than 400, as the real-world data set with 403 exams is also tested. The 7 problems allow better comparison with the real-world problem. This allows evaluation of proposed algorithm's performance under realistic conditions and resource constraints. This inclusion of resource constraints adds a crucial layer of complexity, mirroring the challenges often faced in practical applications, and enables a more comprehensive understanding of the algorithm's robustness and adaptability in scenarios with limited resources.

The main contributions of this research are twofold:

- Preference-Based Discrete Stepping Ahead FA: The algorithm improves a novel approach to achieve better exploration and find improving candidate solutions. The results are better compared to works done by other evolutionary methods [15], [24], [25]. This research opens doors for researchers to explore the preference and stepping-ahead mechanisms in various domains, such as health, sports, and transportation.
- Incorporation of threshold acceptance: It enables fireflies to explore the solution space in steps with new neighborhoods and previous solutions, based on threshold acceptance. mdFA-Step shows superior results compared to dFA-Step [21]. The modification of algorithms to avoid greedy search can be beneficial to researchers where solutions are hard to find.

The paper structure is as follows: Section II reviews related literature on FA and UETP, while Section III presents the real-world problem formulation. In Section IV, we introduce the proposed modified FA with threshold acceptance. Section V details the experimental setup and results, followed by a discussion of the results in Section VI. Finally, Section VII concludes the paper, outlining potential directions for future research.

II. LITERATURE REVIEW

The Discrete Firefly Algorithm (DFA) is a modification of the original Firefly Algorithm, which transforms the continuous domain into a discrete one. FA starts by randomly generating a set of fireflies in the search space. Each firefly represents a potential solution to the optimization problem. The fireflies are attracted to each other based on their brightness, determined by the objective function. The brighter a firefly is, the more attractive it is to other fireflies. At each iteration, the fireflies move towards the brighter ones, and the algorithm updates the brightness of each firefly based on its new position.

The DFA was first proposed by Mirjalili and Gandomi in 2013 [23]. The authors modified the original FA to handle discrete optimization problems by using a transformation process. The process involves discretizing the search space into a set of discrete values, then mapping the continuous variables to the nearest discrete value. The mapping is performed using a rounding function, which rounds the continuous variables to the nearest discrete value. Once the mapping is complete, the DFA performs the optimization process in the discrete domain. The authors demonstrated the effectiveness of the DFA on a set of benchmark problems, including the Traveling Salesman Problem (TSP), the Knapsack Problem, and the Binary Function Optimization Problem.

Since 2013, the DFA has been the subject of several studies, which have focused on improving its performance and evaluating its effectiveness on a range of optimization problems. In a study by Xia et al. [26], the authors proposed a hybrid algorithm that combines the Firefly algorithm with the Particle Swarm Optimization (PSO) algorithm to solve a multi-objective optimization problem. The authors demonstrated that the hybrid algorithm outperforms the DFA and PSO algorithms in terms of solution quality and convergence speed.

In another study by Sababha et al. [27], the authors proposed an improved version of the DFA, called the Enhanced Discrete Firefly Algorithm (EDFA), which modified the original DFA using a mutation operator that randomly changes the values of the fireflies in the search space. The authors demonstrated that the EDFA outperforms the DFA and other state-of-the-art algorithms in solving a range of benchmark problems, including the Knapsack Problem, the Traveling Salesman Problem, and the Quadratic Assignment Problem.

The DFA is a promising algorithm for solving discrete optimization problems, and several studies have focused on improving its performance and evaluating its effectiveness on a range of benchmark problems and real-world optimization problems. Recently, the DFA has also been applied to real-world optimization problems, such as the design of wind turbine blades [28] and the scheduling of chemical production processes [29].

In a study by Tahani et al. [28], the authors used the DFA to optimize the shape of wind turbine blades to maximize power output. The authors demonstrated that the algorithm outperforms other state-of-the-art algorithms in terms of solution quality and convergence speed. In the study by Wang et al. [29], the authors used the DFA to optimize the scheduling of chemical production processes. The authors argued that the algorithm outperforms other state-of-the-art algorithms in terms of solution quality and computational efficiency.

In another study by Hasanien and Matar [30], the authors proposed a modified version of the DFA, called the Modified Discrete Firefly Algorithm (MDFA), to solve a multi-objective optimization problem related to the optimal design of water distribution networks. The authors demonstrated that the MDFA outperforms other state-of-the-art algorithms in terms of solution quality and computational efficiency.

Moreover, the uncapacitated examination timetabling problem (UETP) can be formulated as an integer linear programming (ILP) problem, where the objective is to minimize the number of periods required to schedule all the exams subject to constraints that ensure there are no clashes between exams taken by the same student. The ILP formulation can be extended to include additional constraints, such as the availability of exam rooms and the preferences of students regarding the timing of their exams.

Several other formulations of the UETP have been proposed, including constraint programming, local search, and metaheuristic algorithms. In a study by Burke et al. [17], the authors proposed a hybrid approach that combined a constraint programming solver with a metaheuristic algorithm. The hybrid approach outperformed both the constraint programming solver and the metaheuristic algorithm when tested on a set of benchmark problems.

Several solution approaches have been proposed for the UETP, including metaheuristic algorithms such as particle swarm optimization (PSO), simulated annealing, tabu search, genetic algorithms, and ant colony optimization, which have been used to solve the UETP with varying degrees of success. In 2012, the authors of [18] proposed a solution for UETP using discrete particle swarm optimization (DPSO) where a new local search called two-staged hill climbing was proposed and utilized to hybridize the DPSO algorithm. In another study by Al-Salem et al. [31], the authors proposed a hybrid algorithm that combined simulated annealing with a genetic algorithm. The hybrid algorithm outperformed both simulated annealing and the genetic algorithm when tested on a set of benchmark problems. In a study by Aldeeb et al. [32], the authors proposed an intelligent water drops algorithm for solving the UETP. The algorithm was used to minimize the number of periods required to schedule all the exams subject to the constraints of the problem. In another study by Bellio et al. [33], the authors proposed an exact method based on a mathematical programming formulation of the UETP. The authors demonstrated that the exact method can solve instances of the UETP with up to 1,000 exams and 10,000 students in a reasonable amount of time.

The UETP has several real-world applications, including the scheduling of exams in universities and schools, scheduling of medical appointments, and scheduling of airline flights. In a study by Bolaji et al. [34], the authors used a genetic algorithm to solve a variant of the UETP that arises in the scheduling of medical appointments. The authors demonstrated that the genetic algorithm can generate high-quality schedules that meet the constraints of the problem. In another study by Aizam et al. [35], the authors used a hybrid algorithm based on a genetic algorithm and a greedy algorithm to solve a variant of the UETP that arises in the scheduling of airline flights. The authors demonstrated that the hybrid algorithm can generate high-quality schedules that meet the constraints of the problem. In [36], the authors proposed a discrete approach that uses a discretization scheme to solve undergraduate thesis defense timetabling. More recently, Nand et al. [21] proposed a stepping-ahead firefly algorithm for UETP where the discretization scheme worked well with the discrete domain.

The UETP remains a challenging optimization problem that has been the subject of extensive research in the optimization community. Several formulations and solution approaches have been proposed, including ILP, metaheuristic algorithms, exact methods, and hybrid approaches. The problem has several real-world applications; however, further research is needed to develop new solution approaches and to apply the uncapacitated examination timetabling problem to other real-world problems.

III. PROBLEM FORMULATION: INSIGHT FOR REAL-WORLD PROBLEM

In the following section, the formulation of the Toronto Uncapacitated Examination Timetabling Problem (UETP) is provided [14] with some highlights about the real-world problem and the methodology.

The UETP formulation is defined as follows:

The objective function is to minimize the weighted sum of pairwise costs of exams, with weights determined by a proximity function:

$$f_c = \frac{1}{M} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} C_{ij} \cdot w_{ij} \tag{1}$$

$$w_{ij} = \begin{cases} 2^{5-|t_i-t_j|} & \text{if } 1 \le |t_i - t_j| \le 5 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Constraints:

$$\sum_{i=1}^{n} \sum_{j=i+1}^{n} \lambda_{ij}(t) \le 1 \quad \text{for all } t \in \{1, 2, \ldots, T\} \tag{3}$$

$$\lambda_{ij}(t) = \lambda_{ji}(t) \quad \text{for all } i, j \in \{1, 2, \ldots, n\}, \; t \in \{1, 2, \ldots, T\} \tag{4}$$

$$\lambda_{ii}(t) = 0 \quad \text{for all } i \in \{1, 2, \ldots, n\}, \; t \in \{1, 2, \ldots, T\} \tag{5}$$

where w_{ij} is the proximity weight between exams i and j. The proximity weight assigns higher weight to pairs of exams that are scheduled close to each other in time, with a maximum weight of $2^5 = 32$ for exams scheduled in the same time slot and a minimum weight of 1 for exams scheduled 5 or more time slots apart.

The constraints ensure that each time slot is assigned to at most one exam, where $\lambda_{ij}(t)$ is a binary variable that takes the value 1 if exams i and j are scheduled in time slot t, and 0 otherwise.

Overall, this formulation provides a clear and concise representation of the uncapacitated exam timetable problem, and it was utilized to develop the proposed algorithm for finding good solutions to this challenging optimization problem. In the real-world problem, periods can have limits; however, in the problem tested, period capacity is omitted similar to the Toronto benchmark problems.

The variables utilized in the real-world problems are as follows:

- N is the number of exams.
- P is the number of periods.
- $E = \{e_1, \ldots, e_n\}$ is the set of exams.
- T is the total number of students.
- $(C_{ij})_{E\times E}$ is the conflict matrix, where each element in the matrix is the number of students taking exam i and j, and where $i, j \in\{1, \ldots, E\}$.
- $t_k (1 \ge t_k \ge T)$ is the time slot associated with exam k $(k \in E)$.

Real-World Dataset Methodology: The data was collected from the university's exam timetabling committee and the student administrative services to form the dataset henceforth called the USP dataset. The 2 files included the following details:

- USP Exam Courses - has 2 columns: course code and number of students in the course. The file contained 403 course records.
- USP Exam Students - has up to 4 columns; each row contains course codes of courses taken by each student. A student is allowed to undertake a maximum of 4 courses in a semester. The file contained 18835 student records.

The following steps were taken to gather and transform the raw data into the final dataset:

- The full Student classlist for Semester 2, 2019, was acquired from the respective department. The student IDs were not required.
- The courses taken by each student were identified and placed in another file with each row containing courses taken by individual students.
- Data cleaning: Non-examinable courses were removed.
- The course codes were replaced with number codes. This file became the student file (STU).
- In a separate file, the number of students enrolled in each course was determined.
- The course codes were also replaced with number codes corresponding to the courses in the student file. This became the Exam course file (CRS).

The USP dataset has similar properties as the Toronto dataset since it also has 2 sets of files with similar features such as 2 columns for the Course file and multiple columns for the Student file. However, the Toronto dataset contains multiple data files from multiple educational institutions while the USP dataset is confined to data from the university's courses and students.

IV. PREFERENCE-BASED STEPPING AHEAD FIREFLY ALGORITHM WITH THRESHOLD ACCEPTANCE

This section highlights the implementation of the algorithm. The proposed algorithm would be called mdFA-Step while the discrete FA will be called dFA.

A. FEASIBLE SOLUTION

In order for any given algorithm to address a specific problem, it is essential that the algorithm aligns its solutions with the requirements of the problem at hand. To facilitate the application of the FA approach in optimizing solutions, a prerequisite is the establishment of a feasible solution. This necessity leads to the discretization of the solution space during the initialization phase of the FA methodology. This phase involves the adoption of the partial exam technique, specifically the exam swap move, which is employed in this study. The partial exam technique has been selected due to its ability to expedite problem-solving processes. This technique, as applied in this investigation, is particularly effective in addressing the UETP as demonstrated by Mandal et al. [37]. It is characterized by its capacity to rapidly assign examinations to time slots, promoting efficiency in the resolution process.

The graph-based heuristics have garnered attention for their utility in ordering strategies where these strategies entail the organization of constraints through distinct methodologies, as discussed in [16]. In this study, the specific heuristic known as the Largest Degree (LD) approach is employed as done in earlier research [21]. The LD heuristic is a strategy used for scheduling exams based on the number of conflicts each exam has with other exams. The idea is to prioritize scheduling exams with a higher number of conflicts early in the timetable construction process. The rationale behind this is that exams leading to infeasible solutions are addressed foremost, given their inherent complexity in scheduling. Partial orderings of exams are done where 5% of exams are moved and the process is repeated, unlike in previous work where 5-10% exams were moved. This assignment process is carried out randomly to ensure a balanced distribution. Should a feasible solution not be achieved through this process, an optimization operator, such as the exam swap operator, comes into play. This operator, denoted as move number 0 in Table 1, functions by reallocating non-assigned exams to random time slots.

TABLE 1. Search space exploration.

Number	Simple Moves
0	Swap move
1	Kempe Chain move

B. SUPPORTED NEIGHBORHOODS

The generated feasible solution is able to solve the hard constraint, which requires that there can be no conflicts between exams scheduled in the same time slot. Now, the solution needs to be optimized or improved in terms of the soft constraints. As part of the discretization of FA, the solution space is now discrete after using graph heuristic orderings; therefore, to solve soft constraints, a neighborhood operator is needed. The proposed algorithm supports one neighborhood that the mdFA selects in each iteration to generate a candidate of solutions, unlike previous work where two neighborhoods were utilized. The neighborhood is called Kempe chain with single chain exchange. The Kempe chain heuristic is a technique used in exam timetabling problems to address conflicts between exams. It involves identifying connected components of the conflict graph (representing conflicting exams) and then swapping the time slots of exams within these connected components. By doing so, the heuristic aims to break up conflicts and create a more feasible and efficient timetable, ultimately contributing to the successful resolution of scheduling challenges in the exam timetabling domain. The move is presented as move 1 in Table 1.

Move 1 shown in Table 1 is the Traditional Kempe Chain where the given two sets of exams assigned to two periods, a number of chains consisting of exams belonging to either one of the periods is constructed.

It starts with moving 1 randomly picked element from the first random timeslot to another. To maintain feasibility, any conflicts through these moves are solved by moving back and forth exams until no conflicts are present in the two time slots. Only one chain is exchanged unlike the four-color problem where multiple chains are exchanged. In most real-world timetabling problems, moving high-conflicting exams more frequently yields infeasibility and/or induces a higher average increase in the cost function than it does when low-conflicting exams are moved. Therefore, moving an exam from a slot to another conflicting slot without knowing the density of the exams allows movement of multiple exams to make the solution accepted (with the same acceptance condition for all moves).

C. STEPPING AHEAD MECHANISM

The stepping ahead mechanism has been applied to various applications where it was seen to improve the solution quality [21], [38]. The algorithm uses the best solution to search ahead and look for better solutions as a greedy approach. The mechanism allows modifying the current best solution with improved solutions where the number of variables (exams) is limited.

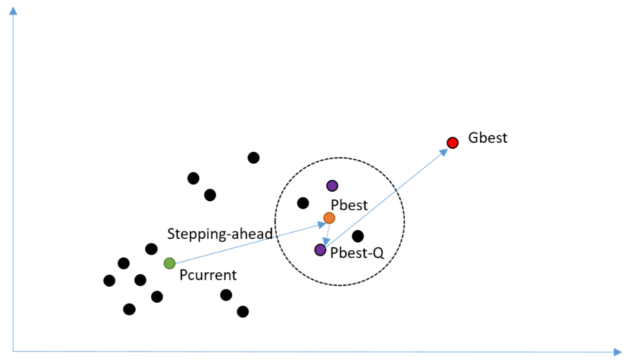

FIGURE 1. Shows stepping ahead move for mdFA.

The concept of "stepping ahead" in the firefly algorithm is illustrated in Figure 1. The figure showcases how a firefly (X_i^t) transitions from its current position to a new best position (X_i^{t+1}) based on the current best firefly position (X_j^t). This movement demonstrates the typical behavior where a sub-optimal solution can lead to the global best solution. The stepping ahead mechanism utilizes a threshold parameter Q to determine the distance each step should cover, allowing the exploration of the search space even with sub-optimal solutions. The notion is to showcase that a lower fitness value in the search space can potentially lead to potentially the global fitness.

The stepping ahead mechanism is represented by the following equation:

$$X_i^{t+1} = X_i^t + Q \cdot (X_j^t - X_i^t), \quad (6)$$

where X_i^{t+1} is the new position of firefly i at time $t+1$, X_i^t is the current position of firefly i at time t, X_j^t is the current best position of firefly j at time t, and Q is the threshold parameter.

Managing the population of an Evolutionary Algorithm (EA) is crucial, as a greedy approach might prematurely discard promising points in the early generations. In the firefly algorithm, even when fireflies are relatively close to each other, they still seek solutions further away from the best solution found. This approach provides the algorithm with an opportunity to break free from local minima, as local minima may mislead the algorithm into believing that the global optimum has been discovered. By allowing fireflies to explore the search space beyond the immediate vicinity of the best solution, the algorithm gains a better chance of finding the true global optimum.

On the contrary, a greedy approach at times gets stuck on local solutions due to not being able to reach solutions that guide to the global best. The threshold setting allows algorithms to select sub-best solutions that can allow a solution to reach the global best or the best solution. At times it is not effective to get the best value; therefore, two extreme ends can be used, being the same to bound the threshold as preference operators.

Furthermore, the integration of the stepping ahead mechanism with the FA is justified by the need for a dynamic and adaptive methodology in addressing the complexities of exam timetabling optimization problems. The stepping ahead mechanism, proven effective in prior applications, aligns with the fundamental goal of continuous improvement in solution quality. By incorporating it into the FA framework, the algorithm gains the ability to actively explore solutions beyond the immediate neighborhood of the current best solution, crucial for avoiding premature convergence and navigating complex solution spaces. The bio-inspired nature of FA, mimicking the mating behavior of fireflies, complements the stepping ahead mechanism's proactive exploration. The introduced threshold parameter adds a layer of adaptability, allowing fine-tuning based on the specific characteristics of exam timetabling instances. Overall, the chosen methodology stands out for its ability to enhance solution quality and navigate diverse problem instances through an integrated, dynamic approach.

D. PREFERENCE OPERATORS

The deterministic approach involves finding the global minimum or maximum of a function by systematically exploring the entire search space. This approach relies on using mathematical techniques designed to find the optimal solution by iteratively improving the candidate solutions until the global optimum is found. On the other hand, the probabilistic approach is based on the use of stochastic optimization algorithms that use probability distributions to explore the search space. These algorithms are often designed to mimic natural phenomena, such as simulating the process of natural selection.

For any algorithm to provide better solutions, preferences can play a significant role. In previous research, restart criteria and threshold difference were utilized to avoid drawbacks of the greedy approach. In this research, the restart criteria are utilized not in 10 iterations but 50 iterations, while the threshold acceptance criteria are changed to suit different problems, unlike previously, which were the same for all. The restart operator allows the algorithm to go back 50 steps and restart the search for good solutions. If the solution is improving, there is no need to activate the restart operator. This allows the preference operator to be algorithm-dependent and only utilized if there is a need. Mathematically, the restart operator can be represented as follows:

$$X_i^{t+1} = \begin{cases} X_i^{t-50}, & \text{if not improving} \\ X_i^t, & \text{otherwise} \end{cases} \tag{7}$$

where X_i^{t+1} is the new position of the solution at time $t+1$, X_i^t is the current position of the solution at time t.

Moreover, the threshold acceptance ranks the solutions, and these steps are iterated until the maximum execution time is reached. While it is common for an algorithm to converge towards the best solution, tackling complex problems necessitates a more sophisticated approach that can produce optimal solutions while exploring the solution space effectively. It is worth noting that the best solution is often surrounded by suboptimal ones. To address this, a smarter technique is employed during the acceptance of solutions when a new solution is not superior to the previous one.

A threshold difference denoted as Q is utilized as previously. However, it is now not generic but problem-dependent. The threshold difference Q dynamically changes for different problems; it is further discussed under the experimental setup. In addition, a probability-based approach is employed as well, where solutions are accepted only if the yield is less than the randomly generated number. This mechanism allows for the acceptance of less favorable solutions based on Equation (8). By incorporating these strategies, the algorithm can effectively explore the search space and consider a wider range of solutions, including those that may initially appear less promising. This enhances the algorithm's ability to find optimal solutions, even in the presence of complex and challenging problem scenarios.

$$\text{rand} \leq e^{-\left(\frac{x_{\text{new}}-x_i}{x_i}/T\right)}, \tag{8}$$

where rand is a uniformly distributed random number between 0 and 1. In Equation (8), x_{new} is the new position of firefly i, while x_i is the current position of the best firefly i, and T is the light intensity. T changes based on the mutation coefficient.

For complex problems with many variables and a large search space, probabilistic methods may be more effective despite their higher computational cost. When parameter tuning is not well-articulated, the solution goes astray but via intervention it gets corrected and redirected. Therefore, the use of deterministic and probabilistic approaches assists in solving the problem better. In this paper, two approaches are used together with the stepping ahead mechanism to allow best and sub-best solutions to get exploration and exploitation.

Furthermore, the impact of the threshold operators in the described algorithm is significant, it's shaping the algorithm's behavior in the quest for optimal timetabling solutions. The dynamic restart criteria, triggered by a threshold-driven decision-making process, ensures that the algorithm avoids premature convergence and remains adaptable to changing solution landscapes. This adaptability is further intensified by the problem-dependent threshold difference (Q) and the probabilistic acceptance criteria. By incorporating these threshold-driven strategies, the algorithm strikes a delicate balance between exploration and exploitation, allowing it to navigate complex timetabling problems more effectively. The threshold mechanism's influence extends beyond traditional deterministic methods, providing the algorithm with the flexibility needed to consider a diverse range of solutions, including those that may initially appear suboptimal. Overall, the threshold mechanism plays a pivotal role in enhancing the algorithm's performance, ensuring robustness, adaptability, and efficacy in addressing the intricacies of exam timetabling problems.

E. ALGORITHM

Algorithm 1 is adapted from the previous work of the authors [21], [38].

The steps highlighted in the algorithm are as follows:

- In Step 1, the variables are initialized in terms of the largest degree (LD) constraint sorting. A random solution is initialized for the population array using the fitness function, where the solution generated is feasible. The swap move is utilized to get to the feasible solutions.
- In Step 2 of the algorithm, the new solutions are evaluated in terms of the discrete domain where the population is evaluated based on the objective function. The transformation operators are Kempe chain technique. Initially, a selected solution is evaluated for 5 cycles.
- In Step 3, if the solution of the current firefly in relation to the compared firefly is not better in terms of the fitness function in 10 generations, the algorithm searches for a new better solution that is further away. The term coined is *Stepping Ahead mechanism* [21], [38]. The algorithm uses a new solution as the input of the next move to find better solutions or closer to the best solutions. The concept is based on the ideology of the surroundings of good solutions with bad solutions in terms of fitness.
- Step 4 is where the first preference operator is utilized. Rank the fireflies in the population based on their fitness values and update the best solution found so far. Use a preference-based heuristic to generate a new set of firefly solutions that respect the exam scheduling preferences of the stakeholder. Optionally, activate a Stepping Ahead mechanism that moves each firefly towards the best solution found, with a probability threshold and a distance constraint. If the resulting solution has the same fitness as the original solution, continue to move the firefly using the Kempe move technique until the fitness improves or until a maximum number of iterations is reached.
- In Step 5, if no improvement is observed after 10 iterations, activate the Stepping Ahead mechanism. If no improvement is still observed after 20 iterations, restart the algorithm from the point where an improvement was last observed. This is the second preference operator.
- Post-process the results and visualize the final exam timetable.

Algorithm 1 Proposed Discrete Stepping Ahead Firefly Algorithm (Adapted from [21]).

```
1:  while $t$ < MaxGen do
2:      for $i = 1$ to $n$ (all $n$ fireflies) do
3:          for $j = 1$ to $n$ (all $n$ fireflies) do
4:              Step 2: Evaluation
5:              if $I_j \leq I_i$ then
6:                  Move exam (firefly) based on transformation; $I_{new}$
7:                  while count $\leq 5$ and $I_i = I_{new}$ do
8:                      Move exam (firefly) based on transformation; $I_{new}$
9:                  end while
10:             end if
11:             Step 3: Stepping ahead
12:             if Stepping ahead is selected then
13:                 Move exam (firefly) based on the transformation; $I_{new}$
14:                 while count $\leq 5$ and $(I_{new} - I_j > Q$ or $I_i = I_{new})$ do
15:                     Move exam (firefly) in relation to $I_{new}$ using the transformations; $I_{new}$
16:                 end while
17:             else
18:                 nothing
19:             end if
20:         end for
21:     end for
22:     Step 4: Preference operator 1
23:     Rank fireflies and update the best using threshold and probability; Preference utilized;
24:     if No improvement then
25:         Stepping ahead is activated;
26:     else
27:         Stepping ahead is deactivated;
28:     end if
29:     Step 5: Preference operator 2
30:     if No improvement due to stepping ahead then
31:         Restart a few steps back;
32:     end if
33: end while
34: Post-processing the results and visualization
```

The mdFA-Step uses the firefly metaheuristic to find a good solution to the uncapacitated exam timetabling problem, with the addition of the Kempe move technique, Stepping Ahead mechanism, and preference operators to enhance the exploration and exploitation of the search space. The preference-based heuristic and the restart mechanism help avoid local optima and improve the quality of the solutions.

In summary, parameter tuning is an important step in machine learning that can significantly affect the performance of the model. The method uses a probabilistic model to approximate the performance of the model as a function of the hyperparameters and then selects the hyperparameters that are expected to give the best performance based on the model. Evolutionary algorithms can be effective when the search space is complex and the evaluation of the model is expensive, but they can be computationally expensive and require a lot of computational resources. The modified parameter tuning on the acceptance criterion allows superior solutions to get accepted, but diversity is lost as it becomes a greedy search. To have diversity, when solutions are not better than the current best, a threshold difference and probability would be used to allow non-superior or sub-best solutions to be accepted. There is a need for intelligent solutions where diversity and superior solutions are kept. Therefore, using a threshold difference with Q on all problems cannot give good performance. Q needs to be dependent on the problem, as the probability approach. The impact of parameter tuning on mdFA-Step's performance is significate. It serves as a dynamic mechanism that allows the algorithm to intelligently navigate complex search spaces, avoid local optima, and adapt to the diverse demands of different timetabling scenarios. The carefully tuned parameters collectively contribute to the algorithm's robustness, efficiency, and effectiveness in generating high-quality solutions for real-world exam timetabling problems.

F. DETAILED EXECUTION PROCEDURE

The overall framework of the mdFA-Step approach for the Uncapacitated Examination Timetabling Problem is based on four steps:

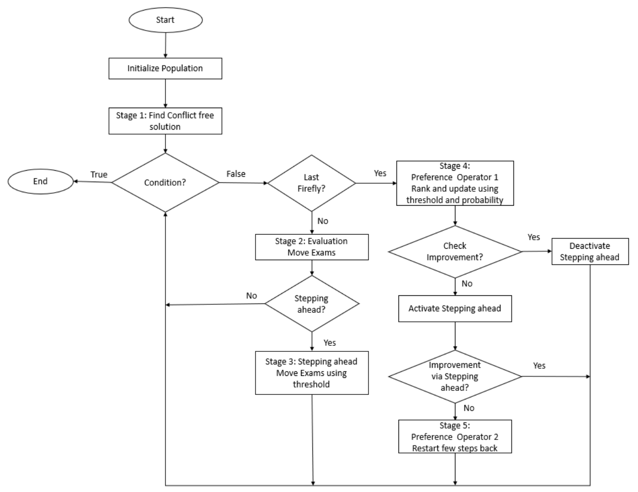
Start
Initialize Population
Stage 1: Find Conflict free solution
End
True
Condition?
False
Last Firefly?
Yes
Stage 4: Preference Operator 1 Rank and update using threshold and probability
No
Stage 2: Evaluation Move Exams
Check Improvement?
Yes
Deactivate Stepping ahead
No
Activate Stepping ahead
No
Stepping ahead?
Yes
Stage 3: Stepping ahead Move Exams using threshold
Improvement via Stepping ahead?
Yes
No
Stage 5: Preference Operator 2 Restart few steps back

FIGURE 2. Flowchart of mdFA.

- **Step 1 - Find Feasible Solutions**: The solution is constructed using a partial exam assignment strategy used together with Largest Degree (LD) graph heuristic orderings. In LD, exams are sorted in descending order according to the number of conflicts each exam has with others. Later, using the partial exam technique, the sorted exams are placed in non-conflicting slots. This ensures finding feasible solutions during initializing by placing the exams in different slots based on the degree of conflict of exams.
- **Step 2 - Apply Kempe Chain Move**: The initial solution needs to be optimized. The neighborhood operators based on the Kempe Chain move are used to improve the best solution, and solution acceptance is possible via threshold acceptance.
- **Step 3 - Apply Stepping Ahead Mechanism**: The next step of the algorithm is to use the stepping ahead mechanism with preference. Here, Kempe Chain move is used to improve the solution where if the first move does not find a different solution than the current best, it repeats the step based on cycles. This is the greedy approach.
- **Step 4 - Preference Operator**: The operators allow mdFA-Step to select sub-best solutions to find better solutions in the form of threshold acceptance and restart. This is orchestrated using a promising solution that has already been found from the previous move.

The flowchart of the proposed algorithm is shown in Fig. 2.

V. EXPERIMENTS AND RESULTS

This section presents the experimental setup and results of the mdFA-Step. The experimental setup highlights the parameters used in the algorithm and details about the Uncapacitated Examination Timetabling Problem (UETP). The results presented include the mean, median, and the best and worst results from the experiments.

Setting: The mdFA-Step was implemented in the MATLAB language and executed on a laptop with the following configurations: Intel Core i7-8665U (CPU @ 1.90GHz with 16GB RAM) and Windows 10 OS. The datasets are utilized in the same way as in the literature without modifications. This allows an observation of the true performance of the proposed algorithm as reported in the literature. The parameters needed for dFA, as described in Table 2, are obtained from the literature.

TABLE 2. Parameter setting.

Parameter	Value
Initial Population size	50
No. of Runs	10
Initial Light intensity	0.1
Damping Ratio	0.99
Light Absorption Coefficient	1
Attraction Coefficient Base Value	2
Mutation Coefficient	0.9
Q (stepping acceptance)	0.01

The benchmark dataset tested in this paper consists of 7 problems compared to the 13 problem set. The selected 7 problems have exams totaling less than 400. The real-world dataset tested has 403 exams, providing a good benchmark to test the algorithm's performance on a similar number of exams to verify its effectiveness.

The new modification of the mdFA-Step using the threshold mechanism not only allows the acceptance of sub-best solutions but also gives a chance to other solutions to contribute to finding better solutions in their solution space. Usually, the approaches used are deterministic and probabilistic. Deterministic gradually updates the parameter, while probabilistic is based on mathematical formulation. For individual test problems, the parameter is set based on multiple experiments, where it's a fixed value or a threshold set with two values. Table 3 shows the threshold setting for the tested problems.

TABLE 3. Setting of threshold Q for Toronto problems.

Problem	Q1	Q2
EAR83	0.05	0.1
HEC92	0.1	0.1
LSE91	0.05	0.075
STA83	0.01	0.01
TRE92	0.75	0.75
UTE92	0.01	0.01
YOR83	0.05	0.05

A. DATASET

Table 4 shows the characteristics of the 7 benchmark functions. The data were introduced by Carter [16] and can be retrieved from [39]. The dataset has the property of a very steep learning curve due to its challenging nature.

TABLE 4. Dataset problem characteristics (Toronto).

Problem	Exams	Students	Admissions	Density	Slots
EAR83	190	1125	8109	0.27	24
HEC92	81	2823	10632	0.42	18
LSE91	381	2726	10918	0.06	18
STA83	139	611	5751	0.14	13
TRE92	261	4360	14901	0.18	23
UTE92	184	2749	11793	0.08	10
YOR83	181	941	6034	0.29	21

Results: Tables 5–7 show the results obtained on the benchmark dataset, while Table 8 shows results for the real-world dataset. Table 9 shows a comparison of results with selected work from the literature.

TABLE 5. Statistical summary of results on three datasets for dFA, dFA-Step, and mdFA-step.

Algorithm	Instance	Best	Median	Worst	Mean
dFA	HEC92	10.31	10.58	10.62	10.51
	STA83	**157.03**	**157.05**	157.20	157.10
	YOR83	38.15	38.15	38.15	38.15
dFA-Step	HEC92	10.17	10.40	10.77	10.42
	STA83	**157.03**	**157.05**	157.14	157.08
	YOR83	36.23	37.33	**37.70**	37.19
mdFA-Step	HEC92	**10.03**	**10.33**	**10.46**	**10.31**
	STA83	**157.03**	**157.05**	**157.08**	**157.05**
	YOR83	**34.58**	**37.13**	**37.91**	**36.89**

For the USP problem, Q1 and Q2 are set at 0.075. The number of exams is 403 with 18,835 students, while total admissions were 47,046. The dataset is named based on the number of periods it's tested on; for example, if the number of periods was set at 20, its name is given as USP20. Note that this is a simple measure of density, and there may be other factors to consider when analyzing the density of an Examination Timetabling problem, such as the distribution of exams across time slots, the duration of exams, and the preferences of students and faculty members.

TABLE 6. Results of Wilcoxon signed rank testing and ANOVA P-test for mdFA-step on three datasets.

	Instance	Rank	P-Value
dFA	HEC92	1.190e-01	9.30e-02
	STA83	7.301e-01	2.54e-01
	YOR83	1.111e-02	1.42e-01
dFA-Step	HEC92	3.175e-01	1.45e-01
	STA83	6.032e-01	3.58e-01
	YOR83	1.000e-02	4.72e-01

In Table 5, the performance of dFA, dFA-Step, and mdFA-Step methods is shown on selected 3 benchmark datasets where the selected datasets are small in size. The results in bold indicate the best results obtained for the algorithm. It can be seen from the table that mdFA-Step has improved performance in all 3 datasets. The modification shows better results obtained for the mdFA-Step when compared to the dFA-Step algorithm. The utilization of the preference operator with the stepping-ahead mechanism has improved the performance. The significance level of dFA,

TABLE 7. Statistical summary of results from mdFA-step.

Problem	Best	Median	Worst	Mean
EAR83	34.33	36.25	36.92	36.02
HEC92	10.03	10.33	10.46	10.31
LSE91	11.02	11.42	11.61	11.37
STA83	157.03	157.05	157.08	157.05
TRE92	8.53	8.68	8.95	8.72
UTE92	24.92	25.06	25.31	25.09
YOR83	34.58	37.13	37.91	36.89

dFA-Step, and mdFA-Step has been conducted by the use of the Wilcoxon signed-rank testing [40]. The test results are shown in Table 6, where the mdFA-Step has a significantly different distribution in HEC92 and YOR83, while in STA83, it has a similar distribution. Additionally, the Anova p-value presented in Table 6 shows there is a significant difference in result distribution of all the datasets, as this is evident where the p-value is less than the threshold of 0.05.

The Wilcoxon signed-rank test is a non-parametric statistical test used to compare two related samples. The Wilcoxon signed-rank test is particularly useful in this context because it does not assume that the data is normally distributed, which is often not the case for performance metrics in optimization problems. Instead, the test ranks the performance of the two algorithms or approaches and calculates a test statistic based on the differences in these ranks. If the test statistic is statistically significant, it indicates that there is a significant difference in performance between the two approaches.

Overall, mdFA-Step provides the most effective solutions to this challenging optimization problem.

In Table 7, the statistical results of mdFA-Step on 7 problems from the University of Toronto benchmark exam timetabling problems (version I) [16] are shown. Version I was utilized for this research because it has been mostly reported in the literature. The results show a good result distribution as the best and worst are not very far apart, indicating a good distribution of the results. In Table 8,

TABLE 8. Statistical summary of results from mdFA-step for the real-world dataset.

Problem	Best	Median	Worst	Mean
USP20	3.29	3.38	3.39	3.36
USP19	3.74	3.80	3.84	3.79
USP18	4.14	4.21	4.39	4.24

the statistical results of mdFA-Step on 3 problems from University of the South Pacific are shown. The 3 periods are tested on the problem as that is the feasible periods utilized in the real-world. The real-world data has similar exam size as the selected 7 benchmark problems. The results show good result distribution as best and worst are not very apart, indicating good distribution of the results. For larger period, the fitness is less while for smaller period the fitness increases; however, the problem is able to be solved.

In Table 9, the comparison of results of mdFA-Step is carried out with methods from literature that are similar to the proposed method. The results from the proposed method are indicated by mdFA-Step, while results from other methods are listed below it. Some methods from literature are not compared as those methods used pre-testing phases or additional stages other than the stages used by dFA-Step. It can be seen that the proposed method has comparable results in all 7 cases. The previous results on dFA-Step are also compared to show the importance of modification and its performance on the 7 datasets.

Moreover, the Table 9 provides a comprehensive evaluation of the proposed algorithm mdFA-Step. While looking at the exams Yor83, Sta83 and Hec92, its shows good algorithmic performance. Notably, lower values associated with mdFA-Step across diverse datasets suggest a heightened level of computational efficiency, indicating its ability to achieve superior results with less computational effort compared to other algorithms. This consistency in competitive performance underscores the algorithm's reliability across various problem instances which is visible when compared to its predecessor dFA-Step. In terms of scalability, mdFA-Step's performance across datasets of varying sizes is commendable. A sustained efficiency across different scales indicate favorable scalability where it ranks fifth out of the thirteen algorithms.

The Friedman test is a non-parametric test used to determine if there are statistically significant differences between groups. Table 10 shows the low p-value (closer to zero), indicating that there are significant differences among the groups. Since the null hypothesis of the Friedman test is that there are no differences between the groups, a low p-value suggests that the null hypothesis is rejected, and there are significant differences. The Friedman test itself does not identify which specific algorithms are different from each other; therefore, to determine which algorithms are different, a post-hoc test was done. In Table 11, the Tukey post-hoc test results show pairwise comparisons between mdFA-Step and different algorithms. The "reject" column indicates whether the null hypothesis of no difference between the compared groups is rejected or not. All comparisons result in "False" in the "reject" column. This means that, according to the Tukey test, there are no significant differences between any pair of algorithms. Since the sample size is small, for a larger sample size, the Tukey test does not identify specific pairs of algorithms that are significantly different from each other in terms of their performance.

Figures 3(a) to 3(c) display the average convergence graphs of dFA, dFA-Step and mdFA-Step. Figure 3(a) shows the convergence graph of the HEC92 dataset, while Figure 3(b) shows the average convergence graph of the STA83 dataset, where uniform performance is evident after 100 generations for all algorithms. Similarly, Figure 3(c) illustrates the convergence of the YOR83 dataset, indicating that the graph becomes uniform after 500 iterations for dFA and dFA-Step, while mdFA-Step gives uniform performance after 500, 1000, 1500, 2000 and 2500. Since YOR83 was a challenging problem to solve, the use of transformation, stepping-ahead mechanism, and preference operators allowed the algorithm to move out of local minima.

B. ANALYSIS OF COMPUTATIONAL COMPLEXITY

mdFA-Step has demonstrated its effectiveness in solving the examination timetabling problem. The computational complexity is key to find the theoretical efficiency as done in [47]. The primary procedures of the time complexity is outlined below:

Stage 1 (Feasible Solution:)

- Initializing N fireflies: $3O(N)$.
- Reproduction of feasible solutions has its own time complexity where the periods are filled with exams and later moved around if not feasible. In the worst-case scenario, it is $O(N^2)$.
- Due to the use of sorting, the time complexity is $O(\log N)$.
- Overall, the time complexity of the algorithm in this stage is $O(N^2 \log N)$.

Stage 2 (Main Loop:)

- Executing the main loop until $t < \text{MaxGen}$: $O(\text{MaxGen})$.
- Two nested loops over fireflies (i and j): $O(N^2)$.
- Fitness evaluation involves moving fireflies based on transformations: $O(5)$.
- Fireflies may move based on Stepping ahead: $O(5)$.
- Ranking and updating the best fireflies involve threshold and probability considerations: $2O(N)$.

TABLE 9. Best results from the literature compared with mdFA-step.

Algorithms	Ear83	Hec92	Lse91	Sta83	Tre92	Ute92	Yor83
mdFA-Step	34.3	10.0	11.0	157.0	8.5	24.9	34.6
dFA-Step	34.7	10.2	11.3	157.0	8.5	25.0	36.2
Abdullah et al. [41]	34.9	10.3	10.2	159.2	8.4	26.0	36.2
Alefragis et al. [42]	32.7	10.0	10.0	157.0	7.9	24.8	35.1
Burke and Bykov [43]	32.7	10.1	9.9	157.0	7.7	27.8	34.8
Burke et al. [17]	33.2	10.3	10.4	156.9	8.3	24.9	36.3
Caramia et al. [44]	29.3	9.2	9.6	158.2	9.4	24.4	36.2
Carter et al. [16]	36.4	10.8	10.5	161.5	9.6	25.8	41.7
Demeester et al. [45]	32.5	10.0	10.0	157.0	7.7	24.8	34.6
Di Gaspero [24]	45.7	12.4	15.5	160.8	10.0	29.0	41.0
Eley [15]	36.8	11.1	11.3	157.3	8.6	26.4	39.4
Merlot et al. [25]	35.1	10.6	10.5	157.3	8.4	25.1	37.4
Yang and Petrovic [46]	33.7	10.8	10.4	158.4	7.9	25.4	36.4

TABLE 10. Friedman test results.

Statistic	P-value
54.81	1.96×10^{-7}

- If no improvement due to stepping ahead, a few steps are restarted: $O(5)$.
- Overall, the time complexity of the algorithm in this stage is $O(N^2)$. This is the same time complexity for a Firefly Algorithm in the continuous domain.

Overall Time Complexity: Combined time complexity for mdFA-Step is $O(N^2 \log N)$.

VI. DISCUSSION

Tables 5 to 7 demonstrate promising results for uncapacitated exam timetabling benchmark problems. Specifically, Table 5 presents a comparison between dFA-Step and dFA, revealing the consistent performance of dFA-Step across all datasets. The stepping-ahead mechanism fine-tunes the results, proving particularly beneficial for complex datasets where dFA alone may require additional guidance in the search space.

In Table 6, significance test results for mdFA-Step indicate its significant deviation from dFA and dFA-Step in 2 out of 3 experimented datasets. mdFA-Step outperforms due to the threshold acceptance modification, introducing improved solution hunting through the stepping-ahead strategy. This strategy extends the search area within the same search space by incorporating sub-best solutions alongside the best ones. The preference parameter enables the activation or deactivation of this mechanism, allowing the algorithm to perform efficiently in global searches, potentially avoiding local optima.

Furthermore, Table 7 provides a statistical summary of dFA-Step results, showcasing consistent performance across multiple runs. The proximity of minimum and worst results indicates stability, and valuable insights into the algorithm's behavior can be derived from the minimum, maximum, mean, and median columns.

Table 8 displays results for the USP problems, illustrating mdFA-Step's ability to provide feasible solutions with a well-distributed range of results. In Table 9, a comparison between the proposed algorithm and selected works from the literature shows competitive results across various problems. mdFA-Step consistently delivers favorable outcomes, and the flexibility of the framework allows easy experimentation with different algorithms and neighborhoods.

The proposed algorithm mdFA-Step exhibits robustness by consistently producing high-quality solutions across diverse timetabling scenarios. The algorithm's resilience to variations in data, demonstrated through its competitive performance in benchmark problems (Table 7) and real-world datasets (Table 8), underscores its reliability in addressing uncertainties and complexities inherent in practical timetabling applications. The minimal differences between the best and worst fitness values indicate a stable and consistent performance, highlighting mdFA-Step's ability to navigate through different problem sizes and structures with efficacy. In addition to, the adaptability of mdFA-Step is evident in its successful application to various timetabling problem types. The algorithm's versatility is showcased by its ability to handle various constraints and characteristics specific to each problem type. For instance, its ability to generate good solutions for real-world problem, as seen in the fitness values achieved for problems like USP18, USP19, and USP20 in the real-world datasets. This adaptability is crucial for addressing the unique constraints and requirements associated with different timetabling instances. The mdFA-Step's underlying optimization mechanisms allow it to dynamically adjust to the characteristics of each problem type, making it a robust and flexible solution for diverse timetabling challenges.

Moreover, Table 10 shows the Friedman test where low p-value indicates that there are significant differences among the algorithms. Since, there is significant difference, post-hoc test was done. In Table 11, the Tukey post-hoc test results show pairwise comparisons between mdFA-Step and different algorithms and since False was obtained under reject column for all pairwise comparison, there are no significant differences between any pair of the algorithms. The sample size was small as it was tested using best results obtained from literature, not much analysis could be made. However,

TABLE 11. Tukey HSD results for comparisons with mdFA-step.

Group 1	Group 2	Mean Diff.	P-Value	Lower CI	Upper CI	Reject
Abdullah et al.	mdFA-Step	-0.7	1.0	-97.6465	96.2465	False
Alefragis et al.	mdFA-Step	0.4	1.0	-96.5465	97.3465	False
Burke and Bykov	mdFA-Step	0.0429	1.0	-96.9037	96.9894	False
Burke et al.	mdFA-Step	0.0	1.0	-96.9465	96.9465	False
Caramia et al.	mdFA-Step	0.5714	1.0	-96.3751	97.518	False
Carter et al.	mdFA-Step	-2.2857	1.0	-99.2322	94.6608	False
Demeester et al.	mdFA-Step	0.5286	1.0	-96.418	97.4751	False
Di Gaspero	mdFA-Step	-4.8714	1.0	-101.818	92.0751	False
Eley	mdFA-Step	-1.5143	1.0	-98.4608	95.4322	False
Merlot et al.	mdFA-Step	-0.5857	1.0	-97.5322	96.3608	False
Yang and Petrovic	mdFA-Step	-0.3857	1.0	-97.3322	96.5608	False
dFA-Step	mdFA-Step	-0.3714	1.0	-97.318	96.5751	False

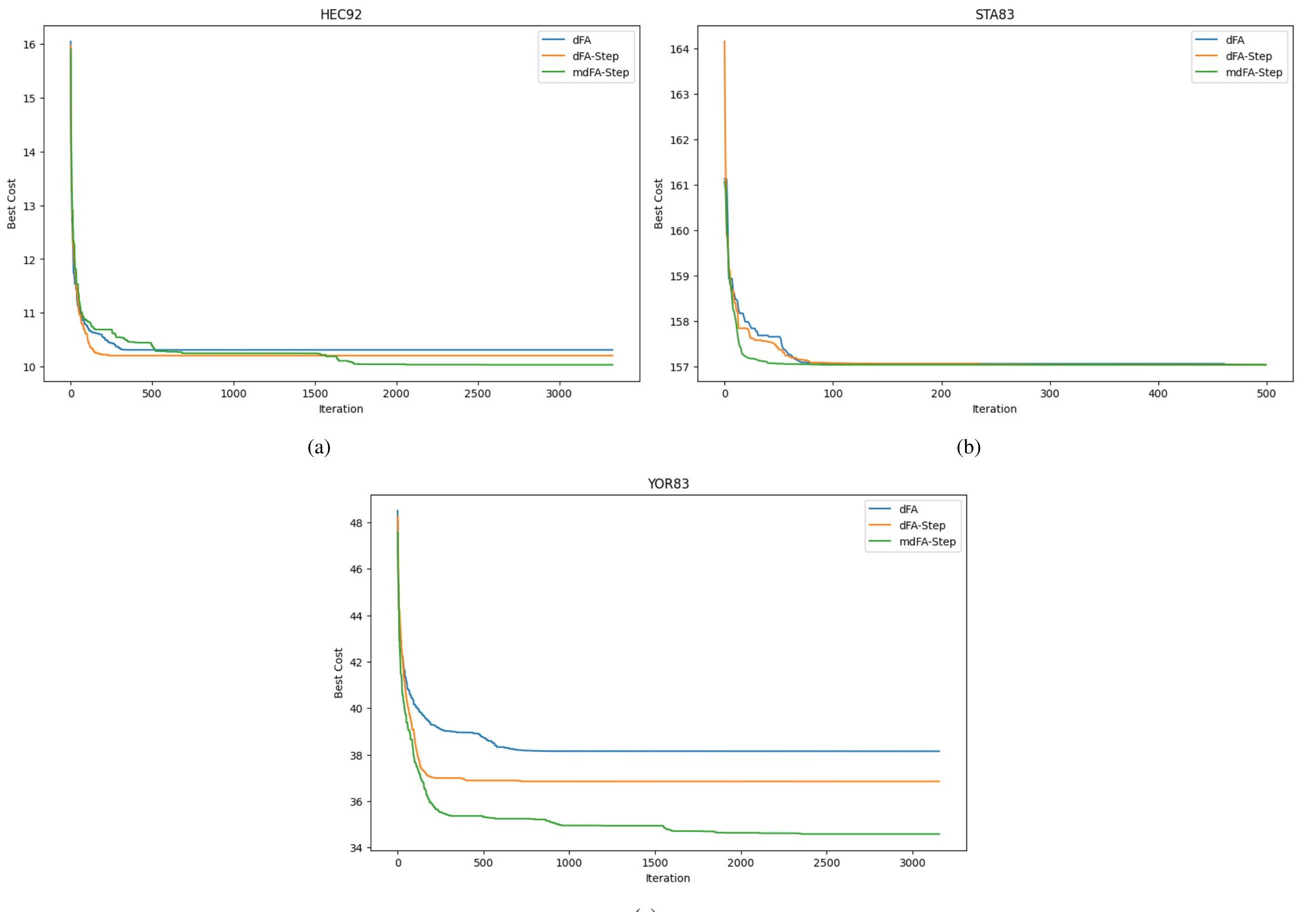

FIGURE 3. Average convergence graphs of dFA, dFA-step and mdFA-step. (a) HEC92. (b) STA83. (c) YOR83.

the Figures 3(a) to 3(c) offer further analysis of mdFA-Step, demonstrating convergence superiority among other two algorithms. The activation of the stepping-ahead parameter leads to swift improvements, showcasing the algorithm's ability to drive towards better solutions.

While these results serve as a proof of concept at the preliminary level, the comparison with state-of-the-art algorithms and the utilization of benchmark datasets emphasize the algorithm's utility in this domain. The consistent performance is attributed to the introduction of the preference-based stepping-ahead parameter, enabling proactive exploration of the search space by incorporating both best and worst solutions.

A. CONVERGENCE ANALYSIS

This sub-section discusses the convergence of the mdFA-Step algorithm.

Theorem: Let x^* be the global optimum of the objective function $f(x)$, and let x_t be the best solution found by a stochastic optimization algorithm after t iterations. Then, if the algorithm satisfies the following conditions:

- The algorithm generates a sequence of solutions that converges to x^* with probability 1.
- The objective function $f(x)$ is continuous and bounded.
- The algorithm is unbiased, meaning that the expected value of the algorithm's updates is equal to the true gradient of the objective function.

Then, the probability that x_t is within a distance ϵ of x^* converges to 1 as t approaches infinity.

Proof: To demonstrate the convergence of d_t to 0, we consider the following:

Define $d_t = \|x_t - x^*\|$ as the distance between the algorithm's solution and the global optimum at iteration t. By the triangle inequality, we have:

$$d_t \leq d_{t-1} + \|x_t - x_{t-1}\|$$

Assuming unbiased updates, we have:

$$\mathbb{E}[\|x_t - x_{t-1}\|] \leq \alpha_t,$$

where α_t is a non-increasing sequence of positive real numbers.

While α_t is non-increasing, the algorithm's progress may lead to larger improvements after small steps. To address this, we consider the non-increasing property as a general trend rather than a strict rule.

Now, let's show how d_t converges to 0:

With probability 1, there exists T such that $\|x_t - x_{t-1}\| \leq \alpha_t$ for all $t \geq T$. Therefore, we have:

$$d_t \leq \sum_{i=T}^{t} \alpha_i$$

Since α_t is non-increasing, this sum converges. Specifically, $\lim_{t\to\infty} \sum_{i=T}^{t} \alpha_i = L$, where L is a finite limit.

Thus, d_t converges to a finite value:

$$\lim_{t\to\infty} d_t \leq \lim_{t\to\infty} \sum_{i=T}^{t} \alpha_i = L$$

As d_t is always non-negative, this implies that $\lim_{t\to\infty} d_t$ is a non-negative finite value. In the context of convergence analysis, achieving $\lim_{t\to\infty} d_t = 0$ implies that the algorithm's solution is approaching the global optimum as the number of iterations increases, supporting the convergence claim.

Therefore, the probability that the best solution found by the algorithm is within a distance ϵ of the global optimum converges to 1 as the number of iterations approaches infinity, under the conditions stated in the theorem.

VII. CONCLUSION

This research introduces a novel approach to tackle optimization problems, specifically focusing on the Uncapacitated Examination Timetabling Problems (UETP). The proposed method involves implementing a discrete Firefly Algorithm (dFA) with a modified "preference-based stepping ahead" parameter, creating a new technique termed mdFA-Step. The mdFA-Step algorithm underwent extensive evaluation by addressing seven benchmark Toronto problems sourced from the literature, in addition to a real-world university problem. The experimental findings highlight the superior performance of the mdFA-Step algorithm when compared to standalone dFA, stepping-ahead dFA, and selected methods from the literature.

The primary contribution of this research lies in the preference-based stepping ahead mechanism. This mechanism not only explores the solution space around the best solution but also effectively exploits it by incorporating sub-best solutions through threshold acceptance in the form of preferences. The stepping-ahead mechanism enables fireflies to comprehensively explore the solution space, while threshold acceptance ensures efficient exploitation of the area. Consequently, the new algorithm can uncover high-quality solutions that may otherwise be overlooked when trapped in local optima, enhancing the search through neighborhood exploration to identify even more superior solutions. The preference-based approach acknowledges that optimal solutions can be in proximity to suboptimal solutions at various stages, utilizing progressive solutions as inputs to generate new solutions based on problem-specific preferences. Convergence analysis demonstrates that mdFA-Step can reach the best solution in the discrete domain. Comparison with a modified Firefly Algorithm from the literature solidifies the efficacy of the proposed mdFA-Step method on the selected datasets, indicating its potential application in various domains, including robotics and healthcare.

It is essential to emphasize that the results obtained in this research serve as a proof of concept at the preliminary stage, focusing on the University of Toronto benchmark exam timetabling problems (7 problems) and a real-world university problem. In future research, we aim to extend the evaluation to encompass all datasets from the Toronto benchmark and other real-world university datasets. Furthermore, our research can be expanded to incorporate additional benchmark datasets featuring more intricate constraints, such as room allocations, as observed in the ITC 2007 dataset. Pursuing these avenues aims to further validate and refine the mdFA-Step algorithm, ensuring its applicability and effectiveness across diverse scenarios in the field of optimization. Future research will also include a comprehensive analysis of parameter settings to find the optimal configuration for the algorithm across both existing and novel problem domains, enhancing its performance and applicability with a more detailed statistical analysis of the results which includes confidence intervals and significance testing.

REFERENCES

[1] T. Moustakas and K. Kolomvatsos, "Correlation adaptive task scheduling," *Computing*, vol. 105, no. 11, pp. 2459–2486, Nov. 2023.

[2] Q. Chen, B. Liu, Q. Zhang, and J. Liang, "Evaluation criteria for CEC special session and competition on bound constrained single-objective computationally expensive numerical optimization," in *Proc. CEC*, 2015, pp. 25–28.

[3] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[4] M.-C. Yuen, S.-C. Ng, M.-F. Leung, and H. Che, "A metaheuristic-based framework for index tracking with practical constraints," *Complex Intell. Syst.*, vol. 8, no. 6, pp. 4571–4586, Dec. 2022.

[5] S. C. M. S. De Sirisuriya, T. G. I. Fernando, and M. K. A. Ariyaratne, "Algorithms for path optimizations: A short survey," *Computing*, vol. 105, no. 2, pp. 293–319, Feb. 2023.

[6] M.-C. Yuen, S.-C. Ng, and M.-F. Leung, "A competitive mechanism multi-objective particle swarm optimization algorithm and its application to signalized traffic problem," *Cybern. Syst.*, vol. 52, no. 1, pp. 73–104, Oct. 2020.

[7] M. Yunfeng, "A study on tactics for corporate website development aiming at search engine optimization," in *Proc. 2nd Int. Workshop Educ. Technol. Comput. Sci.*, vol. 3, Mar. 2010, pp. 673–675.

[8] R. Nand, A. Chand, and M. Naseem, "Analyzing students' online presence in undergraduate courses using clustering," in *Proc. IEEE Asia–Pacific Conf. Comput. Sci. Data Eng. (CSDE)*, Dec. 2020, pp. 1–6.

[9] H. Turabieh, S. A. Azwari, M. Rokaya, W. Alosaimi, A. Alharbi, W. Alhakami, and M. Alnfiai, "Enhanced Harris hawks optimization as a feature selection for the prediction of student performance," *Computing*, vol. 103, no. 7, pp. 1417–1438, Jul. 2021.

[10] K. R. Baker and D. Trietsch, *Principles of Sequencing and Scheduling*. Hoboken, NJ, USA: Wiley, 2013.

[11] S. Daskalaki and T. Birbas, "Efficient solutions for a university timetabling problem through integer programming," *Eur. J. Oper. Res.*, vol. 160, no. 1, pp. 106–120, Jan. 2005.

[12] S. Daskalaki, T. Birbas, and E. Housos, "An integer programming formulation for a case study in university timetabling," *Eur. J. Oper. Res.*, vol. 153, no. 1, pp. 117–135, Feb. 2004.

[13] Y. Saadi, S. Jounaidi, S. El Kafhali, and H. Zougagh, "Reducing energy footprint in cloud computing: A study on the impact of clustering techniques and scheduling algorithms for scientific workflows," *Computing*, vol. 105, no. 10, pp. 2231–2261, Oct. 2023.

[14] E. Burke, Y. Bykov, J. Newall, and S. Petrovic, "A time-predefined local search approach to exam timetabling problems," *IIE Trans.*, vol. 36, no. 6, pp. 509–528, Jun. 2004.

[15] M. Eley, "Ant algorithms for the exam timetabling problem," in *Proc. Int. Conf. Pract. Theory Automated Timetabling*. Berlin, Germany: Springer, 2006, pp. 364–382.

[16] M. W. Carter, G. Laporte, and S. Y. Lee, "Examination timetabling: Algorithmic strategies and applications," *J. Oper. Res. Soc.*, vol. 47, no. 3, pp. 373–383, Mar. 1996.

[17] E. K. Burke, A. J. Eckersley, B. McCollum, S. Petrovic, and R. Qu, "Hybrid variable neighbourhood approaches to university exam timetabling," *Eur. J. Oper. Res.*, vol. 206, no. 1, pp. 46–53, Oct. 2010.

[18] M. A. Ahandani, M. T. V. Baghmisheh, M. A. B. Zadeh, and S. Ghaemi, "Hybrid particle swarm optimization transplanted into a hyper-heuristic structure for solving examination timetabling problem," *Swarm Evol. Comput.*, vol. 7, pp. 21–34, Dec. 2012.

[19] N. Leite, F. Melício, and A. C. Rosa, "A fast simulated annealing algorithm for the examination timetabling problem," *Expert Syst. Appl.*, vol. 122, pp. 137–151, May 2019.

[20] Z. Naji Azimi, "Hybrid heuristics for examination timetabling problem," *Appl. Math. Comput.*, vol. 163, no. 2, pp. 705–733, Apr. 2005.

[21] R. Nand, B. Sharma, and K. Chaudhary, "An introduction of preference based stepping ahead firefly algorithm for the uncapacitated examination timetabling," *PeerJ Comput. Sci.*, vol. 8, p. e1068, Sep. 2022.

[22] E. Osaba, X.-S. Yang, F. Diaz, E. Onieva, A. D. Masegosa, and A. Perallos, "A discrete firefly algorithm to solve a rich vehicle routing problem modelling a newspaper distribution system with recycling policy," *Soft Comput.*, vol. 21, no. 18, pp. 5295–5308, Sep. 2017.

[23] G. K. Jati, R. Manurung, and Suyanto, "Discrete firefly algorithm for traveling salesman problem: A new movement scheme," in *Swarm Intelligence and Bio-Inspired Computation*. Amsterdam, The Netherlands: Elsevier, 2013, pp. 295–312.

[24] L. D. Gaspero and A. Schaerf, "Tabu search techniques for examination timetabling," in *Proc. Int. Conf. Pract. Theory Automated Timetabling*. Berlin, Germany: Springer, 2000, pp. 104–117.

[25] L. T. Merlot, N. Boland, B. D. Hughes, and P. J. Stuckey, "A hybrid algorithm for the examination timetabling problem," in *Proc. Int. Conf. Pract. Theory Automated Timetabling*. Berlin, Germany: Springer, 2002, pp. 207–231.

[26] X. Xia, L. Gui, G. He, C. Xie, B. Wei, Y. Xing, R. Wu, and Y. Tang, "A hybrid optimizer based on firefly algorithm and particle swarm optimization algorithm," *J. Comput. Sci.*, vol. 26, pp. 488–500, May 2018.

[27] M. Sababha, M. Zohdy, and M. Kafafy, "The enhanced firefly algorithm based on modified exploitation and exploration mechanism," *Electronics*, vol. 7, no. 8, p. 132, Jul. 2018.

[28] M. Tahani, N. Babayan, S. Mehrnia, and M. Shadmehri, "A novel heuristic method for optimization of straight blade vertical axis wind turbine," *Energy Convers. Manage.*, vol. 127, pp. 461–476, Nov. 2016.

[29] Z. Wang, L. Shen, X. Li, and L. Gao, "An improved multi-objective firefly algorithm for energy-efficient hybrid flowshop rescheduling problem," *J. Cleaner Prod.*, vol. 385, Jan. 2023, Art. no. 135738.

[30] H. M. Hasanien and M. Matar, "Water cycle algorithm-based optimal control strategy for efficient operation of an autonomous microgrid," *IET Gener., Transmiss. Distrib.*, vol. 12, no. 21, pp. 5739–5746, Nov. 2018.

[31] M. Al-Salem, M. Almomani, M. Alrefaei, and A. Diabat, "On the optimal computing budget allocation problem for large scale simulation optimization," *Simul. Model. Pract. Theory*, vol. 71, pp. 149–159, Feb. 2017.

[32] B. A. Aldeeb, M. A. Al-Betar, and M. Norita, "Intelligent water drops algorithm for university examination timetabling," in *Proc. Int. Parallel Conf. Res. Ind. Appl. Sci.*, Dubai, United Arab Emirates, 2014, pp. 18–30.

[33] R. Bellio, S. Ceschia, L. Di Gaspero, and A. Schaerf, "Two-stage multi-neighborhood simulated annealing for uncapacitated examination timetabling," *Comput. Oper. Res.*, vol. 132, Aug. 2021, Art. no. 105300.

[34] A. L. Bolaji, A. F. Bamigbola, and P. B. Shola, "Late acceptance hill climbing algorithm for solving patient admission scheduling problem," *Knowl.-Based Syst.*, vol. 145, pp. 197–206, Apr. 2018.

[35] N. A. H. B. Aizam, Z. F. B. Ismail, and P. Kovindan, "General mathematical model and timetabling problems: Expansion and application to real-time timetabling," *AIP Conf. Proc.*, vol. 2184, no. 1, 2019, Art. no. 040010.

[36] F. Ghaisani and S. Suyanto, "Discrete firefly algorithm for an examination timetabling," in *Proc. Int. Seminar Res. Inf. Technol. Intell. Syst. (ISRITI)*, Dec. 2019, pp. 1–4.

[37] A. K. Mandal, M. N. M. Kahar, and G. Kendall, "Addressing examination timetabling problem using a partial exams approach in constructive and improvement," *Computation*, vol. 8, no. 2, p. 46, May 2020.

[38] R. Nand, B. N. Sharma, and K. Chaudhary, "Stepping ahead firefly algorithm and hybridization with evolution strategy for global optimization problems," *Appl. Soft Comput.*, vol. 109, Sep. 2021, Art. no. 107517.

[39] R. Qu. (2021). *Benchmark Exam Timetabling Datasets*. Accessed: Jul. 24, 2021. [Online]. Available: http://www.cs.nott.ac.uk/~rxq/data.html

[40] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, Mar. 2011.

[41] S. Abdullah, S. Ahmadi, E. K. Burke, and M. Dror, "Investigating Ahuja–Orlin's large neighbourhood search approach for examination timetabling," *OR Spectr.*, vol. 29, no. 2, pp. 351–372, Feb. 2007.

[42] P. Alefragis, C. Gogos, C. Valouxis, and E. Housos, "A multiple metaheuristic variable neighborhood search framework for the uncapacitated examination timetabling problem," in *Proc. 13th Int. Conf. Pract. Theory Automated Timetabling (PATAT)*, vol. 1, 2021, pp. 159–171.

[43] E. K. Burke and Y. Bykov, "A late acceptance strategy in hill-climbing for exam timetabling problems," in *Proc. PATAT Conf.*, Montreal, QC, Canada, Canada, 2008, pp. 1–7.

[44] M. Caramia, P. Dell'Olmo, and G. F. Italiano, "New algorithms for examination timetabling," in *Algorithm Engineering*. Berlin, Germany: Springer, 2001, pp. 230–241.

[45] P. Demeester, B. Bilgin, P. De Causmaecker, and G. V. Berghe, "A hyper-heuristic approach to examination timetabling problems: Benchmarks and a new problem from practice," *J. Scheduling*, vol. 15, no. 1, pp. 83–103, Feb. 2012.

[46] Y. Yang and S. Petrovic, "A novel similarity measure for heuristic selection in examination timetabling," in *Proc. Int. Conf. Pract. Theory Automated Timetabling*. Berlin, Germany: Springer, 2004, pp. 247–269.

[47] Y. Wang, S. Gao, M. Zhou, and Y. Yu, "A multi-layered gravitational search algorithm for function optimization and real-world problems," *IEEE/CAA J. Autom. Sinica*, vol. 8, no. 1, pp. 94–109, Jan. 2021.

RAVNEIL NAND (Member, IEEE) received the B.Sc. and M.Sc. degrees in computer science, and the B.Tech degree from The University of the South Pacific. He is currently a Research Student and an Assistant Lecturer in computer science and information systems with the School of Information Technology, Engineering, Mathematics and Physics, The University of the South Pacific (USP). His research interests include artificial intelligence, optimization, and data mining. He has published several papers from his research in leading international conference proceedings and journals.

EMMENUAL REDDY (Member, IEEE) received the B.Sc. degree in mathematics and computer science, the Postgraduate Diploma degree in information technology, and the M.Sc. degree in information systems from The University of the South Pacific (USP), where he is currently pursuing the Ph.D. degree in information systems. He is a Research Student and an Assistant Lecturer in computer science and information systems with the School of Information Technology, Engineering, Mathematics and Physics, USP. His research interests include technology-based learning, data mining, blockchain, neural networks, and natural language processing.

KAYLASH CHAUDHARY (Member, IEEE) received the Ph.D. degree in computer science, majoring in formalization, modeling, and analysis of virtual payment systems, from The University of the South Pacific (USP). He is currently a Senior Lecturer in computer science and information systems with the School of Computing, Information and Mathematical Sciences, USP. His research interests include software engineering, distributed system design and implementation, software architecture, electronic micro-payment systems for file sharing in peer-to-peer networks, machine learning, and optimization.

BIBHYA SHARMA (Member, IEEE) received the Ph.D. degree in applied mathematics from The University of the South Pacific (USP). He is currently a Professor of mathematics and the Head of School of Information Technology, Engineering, Mathematics and Physics (STEMP), USP. He has published more than 100 articles in mathematics, science education, robotics, biologically inspired processes, and higher education.