Optimizing Decentralized Exam Timetabling with a Discrete Whale Optimization Algorithm

Emily Sing Kiang Siew[1], San Nah Sze[2], Say Leng Goh[3]

i-CATS University College, Kuching, Sarawak 93350, Malaysia[1]

Faculty of Computer Science and Information Technology, Universiti Malaysia Sarawak, Kota Samarahan, Sarawak 94300, Malaysia[2]

Optimization and Visual Analytics Research Group, Faculty of Computing and Informatics, Universiti Malaysia Sabah, Labuan International Campus, Labuan 87000, Malaysia[3]

***Abstract*—In recent years, there has been increasing interest in intelligent optimization algorithms, such as the Whale Optimization Algorithm (WOA). Initially proposed for continuous domains, WOA mimics the hunting behavior of humpback whales and has been adapted for discrete domains through modifications. This paper presents a novel discrete Whale Optimization Algorithm approach, integrating the strengths of population-based and local-search algorithms for addressing the examination timetabling problem, a significant challenge many educational institutions face. This problem remains an active area of research and, to the authors' knowledge, has not been adequately addressed by the WOA algorithm. The method was evaluated using real-world data from the first semester of 2023/2024 for faculties at the Universiti of Sarawak, Malaysia. The problem incorporates standard and faculty-specified constraints commonly encountered in real-world scenarios, accommodating online and physical assessments. These constraints include resource utilization, exam spread, splitting exams for shared and non-shared rooms, and period preferences, effectively addressing the diverse requirements of faculties. The proposed method begins by generating an initial solution using a constructive heuristic. Then, several search methods were employed for comparison during the improvement phase, including three Variable Neighborhood Descent (VND) variations and two modified WOA algorithms employing five distinct neighborhoods. These methods have been rigorously tested and compared against proprietary heuristic-based software and manual methods. Among all approaches, the WOA integrated with the iterative threshold-based VND approach outperforms the others. Furthermore, a comparative analysis of the current decentralized approach, decentralized with re-optimization, and centralized approaches underscores the advantages of centralized scheduling in enhancing performance and adaptability.**

Keywords*—Examination timetabling; discrete whale optimization algorithm; variable neighborhood descent; capacitated; decentralized

I. Introduction

Educational timetabling involves assigning specific times to resources, events, and spaces while adhering to a predefined set of hard constraints and optimizing soft constraints. Resources typically encompass lecturers, teachers, students, administrative staff, or specialized equipment. Events may include lectures, classes, exams, or other academic activities. Spaces refer to physical locations such as lecture halls, classrooms, or exam rooms.

Numerous formulations have been proposed for this problem, with the two most notable being the uncapacitated formulation introduced by [1] and the capacitated formulation featured as Exam 1 in the Second International Timetabling Competition, discussed by [2]. This study addresses a capacitated formulation of a real-world faculty exam timetabling problem (ETP) at the Universiti of Sarawak, Malaysia (UNIMAS). This problem stands out due to its unique combination of two approaches: online exam scheduling, which solely considers designated periods without considering physical room allocation, and physical exam scheduling, which involves assigning each exam to a specific period and room. Both exam scheduling strategies aim to prevent conflicts and optimize exam spacing, but the latter necessitates meeting room allocation constraints, such as dividing or sharing spaces.

Since ETP is an NP-complete decision problem [3], diverse approaches have been employed to address it. According to a recent study [4], there are six types of solution methods in the ETP. These are mathematical optimization, matheuristics, heuristics, metaheuristics, hyper-heuristics, and hybrid approaches. The survey found that metaheuristics had been the approach most employed over the past 12 years.

Metaheuristics generally outperform exact search methods, as the latter often involves generating all possible solutions, which can be computationally intensive. Metaheuristic algorithms can be broadly divided into two categories: population-based algorithms, which emphasize exploration, and single-solution-based algorithms, which focus on exploitation. Effective metaheuristic design requires balancing two criteria: diversification, which involves exploring the search space broadly, and intensification, which focuses on refining and exploiting the most promising solutions [5].

An effective way to balance exploration and exploitation is by using a hybrid approach that integrates various techniques to enhance the performance of search algorithms. In this study, we introduce and design a novel hybrid method that combines the recently developed Whale Optimization Algorithm (WOA) with local search techniques to solve a real-world ETP. The following outline summarizes the main contributions of this work.

- Discrete WOA algorithm: A solution methodology that combines the WOA algorithm with local search methods is developed. This approach performs better than other VND variants in optimizing exam timetabling.
- Decentralized faculty exam timetabling: We propose a novel model that accommodates the preferences of multiple faculties with contradictory constraints, accounts for varied exam types, and ensures a more inclusive and flexible scheduling framework.
- Utilization of real-world data: The proposed discrete WOA approach is validated using real-world data, showcasing its robustness and practical relevance across various educational settings.

The paper is structured as follows: Section II presents a review of related works, followed by Section III, which outlines the problem description. Section IV discusses the original WOA, other applied methods and neighborhood structures. Section V details the algorithms of the proposed discrete WOA approaches. Section VI presents the experimental results, and Section VII compares the centralized and decentralized approaches. Lastly, Section VIII provides the conclusion of the study.

II. Related Work

Real-world exam timetabling constraints are categorized into four main types [4]: exam-related, period-related, room-related, and invigilator-related. Real-world scenarios more commonly give rise to the capacitated formulation of ETPs, treating room capacities as adhered-to constraints. The constraints on room usage can vary significantly across problem formulations, ranging from limits on the number of exams allowed per room to considerations of individual room capacities and overall seating availability. Some studies extend this by considering the total seating capacity across all rooms within a time slot and the capacities of individual rooms [6–8]. In such cases, several exams may be assigned in the same room without restrictions on the number of exams if the total room capacity is sufficient to accommodate all the students requiring seating.

Dammak et al. [9] proposed a heuristic algorithm that modeled the exam-room assignment problem, allowing multiple exams in a single room. In contrast, other studies have also explored the possibility of scheduling multiple exams in a single room [10, 11]. Other room-related constraints studied include the distance between exam halls [12, 13], the allocation of exams across multiple rooms [12, 14] and assigning specific exams to designated rooms. We incorporate all these constraints—one exam per room, multiple exams per room, splitting exams across multiple rooms, and distance between rooms for split exams—into our approach on a faculty-specific basis.

Researchers have recently designed many intelligent algorithms, such as the Archimedes optimization algorithm [15], Fire Hawks algorithm [16], and WOA, to address various optimization challenges. Notably, the WOA, a swarm intelligence-based approach [17], models the hunting strategies of humpback whales, mimicking their collective feeding behavior. Recent studies have enriched the growing literature by highlighting its successful practical applications and reporting enhanced results and performance [18]. Additionally, research suggests that WOA surpasses other optimization algorithms concerning global search capabilities and convergence speed [19]. The WOA offers several advantages, including simplicity of operation, minimal control parameters, and a robust capability to avoid local optima. These attributes have inspired researchers to employ WOA to address diverse practical challenges.

Although the WOA was originally developed for continuous problems, several studies have explored the use of the WOA for discrete optimization problems, including the knapsack problem [20, 21], feature selection [22–24], and workshop scheduling [25–27]. The primary strength of the WOA lies in its ability to maintain a balance between exploration and exploitation throughout the iterative process. While WOA has shown promise in various optimization problems across multiple domains, to the best of our knowledge, its application to real-world exam timetabling remains unexplored.

Hence, this study bridges this gap by adapting the WOA approach to meet the specific needs of our real-world ETP, offering a novel solution for discrete optimization in educational scheduling. Since WOA was originally designed for continuous optimization tasks, it relies on continuous updates to individual positions, making it unsuitable for discrete scheduling problems like exam timetabling. To overcome this limitation, we propose a modified discrete WOA, incorporating discrete updating strategies to tailor the algorithm to the discrete nature of timetabling. The real-world experimentation outlined in the following sections demonstrates its effectiveness in addressing practical exam timetabling while fulfilling institutional constraints.

III. Problem Definition

This paper presents a solution method for the ETP at UNIMAS. Specifically, we study the decentralized ETP within the Faculty of Economics and Business (FEB) and the Faculty of Computer Science and Information Technology (FCSIT). The data on students within the faculties has been collected and analyzed to evaluate the proposed solution algorithm.

The problem definitions are as follows:

- The exams will take place over two weeks.
- Each day is divided into two blocks.
- Exams have varying durations, such as 120, 150, and 180 minutes.
- If assigning an exam to a single room within a timeslot is not feasible, the exam must be split across multiple rooms.
- The exam day reserved for pre-assigned common courses should not be used for other exams.
- The exams include online exams conducted via an online platform and physical exams held in rooms.

- Online exams must be assigned to a predesignated slot.
- Shared or non-shared exam rooms will depend on each faculty's specific practices.
- There are two types of exam rooms: exam halls and faculty-owned exam rooms.
- Exam halls vary in availability based on the schedule set by the Centre, which range in size from medium to large.
- Faculty-owned exam rooms are consistently available for faculty exams and are typically small-sized.

The hard constraints are:

- H1: Each student may attend only one exam at any given time.
- H2: Each exam can only be scheduled once within the exam period.
- H3: The exam period must not exceed the designated days.
- H4: Rooms must be able to accommodate all students taking an exam during each timeslot.
- H5: Rooms can only be shared if the faculty permits; otherwise, no sharing is allowed.

The soft constraints include:

- S1: Minimize the number of rooms utilized.
- S2: Minimize proximity costs to ensure adequate time gaps between exams.
- S3: Minimize the splitting of exams across different areas or rooms.
- S4: Minimize violations of exams assigned to preferred timeslots.

A solution that violates soft constraints is not considered infeasible; this allows for defining specific objective values for each soft constraint. Consequently, the objective function f aims to minimize the total soft constraint violations, directing the optimization process toward reducing their overall impact. The end user typically determines the weights assigned to different types of soft conflicts. However, this study standardizes the weights by assigning fixed values: 1 for *S1*, *S2*, *S3*, and 2 for *S4* to ensure reproducibility.

IV. Methods

A. Constructive Heuristic Method

The proposed algorithm starts by generating initial feasible solutions, using a constructive heuristic method as the starting point. The process begins with assigning prioritized exams to their preferred time slots, followed by the allocation of online exams and, finally, the allocation of physical exams. We use a best-fit strategy for room allocation, choosing the smallest room that fits, minimizing room splits, and allocating several exams to the same room whenever feasible. The algorithm continues to allocate exams to rooms and periods while ensuring compliance with hard constraints and adhering to the soft constraint of preferred slot assignments.

B. The Original WOA

The WOA is a recently developed swarm intelligence optimization algorithm commonly used to solve optimization and classical engineering problems. When whales locate prey, they swim in a spiral toward it while encircling and foraging using a bubble net. This process involves three hunting strategies: shrinking and attacking with a bubble-net attack, encircling prey, and randomly searching for prey. The first two strategies guide exploitation, while exploration is supported by the third within the WOA.

We present the mathematical model for each phase below, employing a uniform distribution to generate random numbers in the equations. In the following equations, t signifies the current iteration, x refers to the position vector, and *MaxIter* indicates the maximum number of permitted iterations.

1) Exploitation phase – encircling prey: Humpback whales employ strategies described by the mathematical models in Eq. (1) and Eq. (2) to encircle and hunt their prey. As per Eq. (2), acting as search agents, whales adjust their positions relative to the prey—the current optimal solution, x^*. The coefficient vectors C and A, calculated using Eq. (3) and Eq. (4), adjust the search area to determine the whale's position relative to its prey. In both phases, the value of a decreases linearly from 2 to 0, while the vector r exhibits a uniform distribution within the interval [0,1].

$$D = |C \cdot x^* (t) - x (t)| \tag{1}$$

$$x (t+1) = x^* (t) - A \cdot D \tag{2}$$

$$\mathrm{A} = 2ar + a \tag{3}$$

$$C = 2r \tag{4}$$

2) Exploitation phase – bubble-net attacking: The shrinking, encircling behavior is governed by Eq. (5), while the position of a neighboring search agent is determined using a spiral equation as described in Eq. (6). D' denotes the distance from the i-th whale to the optimal solution, with b defining the shape of the logarithmic spiral and l being a random value within the range [-1, 1].

$$a = 2 - t \cdot (2 / MaxIter) \tag{5}$$

$$x (t+1) = D' \cdot e^{bl} \cdot cos(2\pi l) + x^* (t) \tag{6}$$

3) Exploration phase – searching for prey: For exploration, a random search agent is selected to guide the process, as mathematically represented by Eq. (7) and Eq. (8). Vector A contains random values exceeding one or falling below -1, while x_{rand} represents a randomly chosen whale from the population.

$$D = |C \cdot x_{rand} - x| \tag{7}$$

$$x (t+1) = | x_{rand} - A \cdot D | \tag{8}$$

Algorithm 1 delineates pseudocode for the original WOA, which starts by generating an initial population and evaluating it with a fitness function. During each iteration, a random value determines the update of a solution's position using either Eq. (2), Eq. (8), or Eq. (6) methods. The system returns the best solution x* upon meeting the termination criteria.

Algorithm 1: Original WOA

```
Generate initial population x_i for i = 1, 2, ..., n
Compute each solution's fitness
Set the best search solution x*
t = 0
While (t < MaxIter) do
  | For each solution do
  |   | Update C, A, p, a, and l
  |   | If p < 0.5 then
  |   |   | If |A| < 1 then
  |   |   |   | Update the current solution's position by (2)
  |   |   | Else
  |   |   |   | Update the current solution's position by (8)
  |   |   | End If
  |   | Else
  |   |   | Update the current solution's position by (5)
  |   | End If
  | End For
  | Verify if any solution goes beyond the search space and amend it
  | Compute each solution's fitness
  | t = t + 1
  | Update x* if a better solution is found
End While
return x*
```

C. Variable Neighborhood Descent

Exploring a single neighborhood structure may result in finding a local optimum specific to that structure, but this is unlikely to be the global optimum. Conversely, identifying a solution that serves as a local optimum across multiple neighborhood structures enhances the likelihood of reaching the global optimum. This principle forms the foundation of the VND method. Specifically, a VND algorithm is employed to refine the solutions. VND is a deterministic variation of the Variable Neighborhood Descent framework initially proposed by [28]. It has been widely adopted as a local search method in numerous metaheuristics and implemented in diverse forms [29]. During its process, VND systematically explores different neighborhoods of a given solution to enhance its quality.

Algorithm 2 provides the pseudocode for the VND. The algorithm explores the neighborhood structures defined by the operators N_k, where $1 \leq k \leq k_{max}$, following a predefined order. LocalSearch(x, N_k) indicates that a local search is performed using the current neighborhood N_k, starting from solution x. The first-improvement strategy is applied to all neighborhood structures, as described in Subsection D. Specifically, when an improved solution is found within a specific neighborhood, the corresponding move is made, and the next neighborhood structure is explored. This procedure repeats until the maximum iteration limit is reached.

VND can employ various rules to transition between neighborhoods on its list and adopt diverse strategies to explore each. This flexibility gives rise to multiple VND variants, including Basic Sequential VND (BVND), Cyclic VND, Pipe VND (PVND), and Nested VND. These variants may utilize either the first-improvement or best-improvement search strategies. Algorithms 3 and 4 outline the neighborhood change procedure for both BVND and PVND, respectively. For the former, if a better candidate solution is found within a given neighborhood structure, the search resumes in the initial neighborhood structure based on the specified order. Otherwise, the search continues in the next neighborhood structure. For the latter, if the current solution improves within a particular neighborhood, exploration continues within that neighborhood.

Algorithm 2: Variable Neighborhood Descent

```
Procedure VND (x, N)
While (t < MaxIter) do
  | stop = false
  | k = 1
  | x' = x
  | While (k <= k_max)
  |   | x'' = LocalSearch (x, N_k)
  |   | neighborhood_change (x, x'', k)
  | End While
End While
return x'
```

Algorithm 3: Sequential Neighborhood Change for BVND

```
Procedure Sequential_neighborhood_change (x, x', k)
If f (x') < f (x)  then
    | x = x'
    | k = 1
Else
    | k = k + 1
End
```

Algorithm 4: Pipe Neighborhood Change for PVND

```
Procedure Pipe_neighborhood_change (x, x', k)
If f (x') < f (x)  then
    | x = x'
Else
    | k = k + 1
End
```

D. Neighborhood Structure

The five types of neighborhood structures are implemented and described in the following:

- *Kick*: Assign exam e_1 to the period currently designated to exam e_2, then reassign exam e_2 to a different period from the available options. The room for both exams may be available within the designated period.
- *Swap*: Exchange the periods of two exams, while their rooms may be swapped or assigned to different rooms.
- *Shift*: Move an exam to a different period and/or room(s).
- *Reallocate*: Move an exam assigned to a shared room to an unoccupied one.
- *Compact*: Relocate an exam to a shared room during the same period.

Swap, *Shift*, and *Kick* are moves related to both period and room assignments, whereas *Reallocate* and *Compact* focus specifically on room-related adjustments with contradictory objectives. *Reallocate* aims to address room-sharing preferences, especially in cases where certain faculties prohibit sharing. In contrast, *Compact*, which applies to most faculties, aims to reduce the number of rooms utilized, thereby encouraging room sharing. The neighborhood structure is set to N = {*Swap*, *Shift*, *Kick*, *Reallocate*} for faculty exam timetabling where shared exam rooms are prohibited. Otherwise, the neighborhood structure is set to N = {*Swap*, *Shift*, *Kick*, *Compact*}.

V. PROPOSED APPROACH

This study proposes two algorithms: one based on the PVND algorithm and the other on WOA. Subsection A describes the first algorithm, Iterative Threshold-based Variable Neighborhood Descent (ITVND), while Subsection B presents the second algorithm, the modified discrete WOA.

A. Iterative Threshold-based Variable Neighborhood Descent

We propose a variant of the classic PVND, ITVND, with the pseudocode presented in Algorithm 5. The threshold-based pipe neighborhood change procedure is outlined in Algorithm 6. The ITVND algorithm incorporates a control parameter, the objective function threshold cT, and an input parameter, the iteration count L. We initially assign the objective value of the starting solution to cT. The iteration count increments in steps of L, and at every L-th iteration (cT mod L = 0), cT is updated to the current cost. The algorithm accepts all improving or sideways moves with an objective value below cT.

The algorithm explores the solution space using multiple neighborhood structures, where the neighborhood structure N_k is defined for k =1…, *kmax*. The core concept of ITVND is to maintain the objective value threshold across L iterations for various neighborhood structures. It permits accepting inferior solutions within the objective value threshold, introducing a more flexible acceptance condition, which slows the current cost reduction and prolongs the time needed to reach convergence.

Algorithm 5: Iterative Threshold-based Pipe Variable Neighborhood Descent

```
Procedure ITVND (x, N)
t = 0
While (t < MaxIter) do
  | k = 1
  | x′ = x
  | While (k <= k_max)
  |   | x″ = LocalSearch (x, N_k)
  |   | Threshold_pipe_neighborhood_change (x, x″, k, cT)
  | End While
  | If (t mod L = 0) then
  |   | cT = f (x)
  | End If
  | t = t + 1
End While
return x′
```

Algorithm 6: Threshold Pipe Neighborhood Change

```
Procedure Threshold-based_pipe_neighborhood_change (x, x′, k, cT)
If f (x′) < cT or f (x′) < f (x) then
    | x = x′
Else
    | k = k + 1
End
```

B. Discrete Whale Optimization Algorithm

As whales adjust their positions within a continuous domain using specific equations and operators, the original WOA becomes unsuitable for tasks like timetabling, which exhibit discrete characteristics. While the classical WOA algorithm relies on whale interactions to solve optimization problems, its simple neighborhood structure and limited disturbance tend to trap it in local optima. To address this problem, we replaced these equations and operators with a local search mechanism. The proposed modified discrete WOA approach considers two methods: (1) WOA-VD and (2) WOA-IVD. As shown in Algorithm 7, the process begins with a set of solutions generated using a constructive heuristic, then iteratively refined throughout the search process.

During the exploitation phase, the modified discrete WOA algorithm updates individual solutions using information from the best current solution. When probability p is less than 0.5, the algorithm utilizes local search with an improvement criterion, accepting a new solution only if it enhances the current solution's fitness, thereby supplanting Eq. (2). If p is equal to or greater than 0.5, PVND is chosen as a local search method to improve the solution in WOA-VD algorithm. In contrast, the ITVND local search method is used in the WOA-IVD algorithm to compare the effectiveness of the VND variation during the search process. Both methods replace Eq. (5) and take advantage of its ability to search for a larger area by changing neighborhoods in a planned way.

During exploration, local search with a threshold-based acceptance criterion replaces Eq. (8). A new candidate solution is accepted if its objective value falls below a dynamically updated cost bound. This approach helps explore the search space more effectively by mitigating the effects of premature convergence and promoting broader exploration outside the immediate neighborhood of the current optimal solution.

Algorithm 7: Discrete WOA-VD

```
Generate initial population xi for i = 1, 2, ..., n
Compute each solution's objective value
Initialize the best search solution x*
t = 0
While (t < MaxIter) do
  | For each solution individual do
  |   | Update a, A, C, l and p
  |   | If p < 0.5 then
  |   |   | If |A| < 1 then
  |   |   |   | Generate new candidate x'
  |   |   |   | If f (x') < f (x) then
  |   |   |   |   | x = x'
  |   |   |   | End If
  |   |   | Else
  |   |   |   | Generate new candidate x'
  |   |   |   | If f (x') < cBound then
  |   |   |   |   | x = x'
  |   |   |   | End If
  |   |   |   | If (t mod L = 0) then
  |   |   |   |   | cBound = f (x)
  |   |   |   | End If
  |   |   | End If
  |   | Else
  |   |   | Apply PVND (x, N) to improve x
  |   | End If
  |   |
  | End For
  | Calculate each solution's objective value
  | t = t + 1
  | Update x* if a better solution is found
End While
return x*
```

Our search method is based on local search and leverages the following characteristics:

- *Termination Criterion*: A fixed number of iterations as the termination criterion ensures that the algorithm's runtime remains consistent and independent of other parameters.
- *Search Space*: The search space is restricted to the feasible region, which only includes scheduling that fully satisfies all constraints, such as precedence and conflict avoidance.

VI. Experiments

A. Experimental Settings

The experiment is conducted on a computer with Windows 11, equipped with an Intel® Core™ i7 processor, 16.0 GB of memory, and an integrated graphics card. We built the algorithms presented using IntelliJ IDEA, the Integrated Development Environment (IDE), and Java, specifically JDK 1.8, as the programming language. Table I presents the characteristics of the datasets collected from two faculties. These datasets specifically pertain to the first semester of the 2023/2024 academic year.

TABLE I. Characteristics of Datasets from Two Faculties

Description	Dataset	
	FCSIT	FEB
No. of enrolment	2,837	7,035
No. of students	1,026	2,011
No. of exams	31	55
Exam range per student	1 to 5	1 to 8
Conflict density	0.234	0.182
Exam Types	Online & Physical	Physical
Total exam period	12 days	12 days
Average Shared Hall Usage per Timeslot	1.8	3.1
Faculty-Owned Exam Room Count	10	6
Scheduling method	Manually arranged	Proprietary System

$cT = f(x)$

To evaluate the performance of our proposed algorithms, we compared them against other methods and the existing scheduling method, which generates the current solution. The five include three variations of VND: BVND, PVND, and ITVND, along with two variations of WOA: WOA-VD and WOA-IVD. For each of these algorithms, we conducted 30 independent runs per dataset.

B. Experimental Results

Table II below presents the descriptive statistics for the three employed algorithms compared to the existing scheduling method based on 30 runs. The bold formatting indicates the best values achieved by all the methods. We conducted all statistical analyses using SPSS version 29.

TABLE II. Comparison Between Different Methods

Dataset	Method	Slot	Best	Mean	Worst	Std Dev
FCSIT	Manual	12	55.6			
	BVND	8	50.4	53.7	57.3	1.97
	PVND	8	50.1	53.1	56.3	1.48
	ITVND	8	49.4	51.6	54.7	1.41
	WOA-VD	8	49.9	52.9	57.8	2.09
	WOA-IVD	8	**48.7**	**51.2**	**53.7**	**1.28**
FEB	Proprietary System	12	148.6			
	BVND	12	131.8	134.6	138.3	1.77
	PVND	12	132.2	134.7	137.0	1.24
	ITVND	12	130.3	132.4	134.6	1.22
	WOA-VD	12	131.0	132.9	135.3	1.13
	WOA-IVD	12	**129.9**	**131.9**	**133.6**	**1.04**

For the FCSIT dataset, Table II shows that WOA-IVD emerges as the best-performing method among the tested methods, with the lowest objective value (48.7) and the lowest standard deviation (1.28), demonstrating consistent performance. The second-best method is ITVND, which has a slightly higher objective value (49.4) but maintains low variability (1.41). In contrast, despite its competitive mean value, the WOA-VD method has the highest variability (2.09), indicating less consistent results than others.

In comparing the performance of various methods for the FEB dataset, all our proposed methods outperform the proprietary system [30], which employed two-stage heuristic methods across all instances, with a moderate gap. The best-performing method, WOA-IVD, achieved an average of 131.9 with a standard deviation 1.04. ITVND followed closely with an average value of 132.4 and a standard deviation 1.22. However, the results for BVND and PVND are somewhat inferior, with some instances where they perform poorly compared to the ITVND and WOA models.

Overall, the results demonstrate that adopting all the VND and WOA variation methods reduces exam session and objective value compared to the manual approach. The WOA-IVD method performs the best across both datasets, consistently achieving the lowest mean values and demonstrating superior efficiency compared to other methods. On the other hand, VND tends to perform the worst, showing higher values in comparison.

We conducted a one-way ANOVA analysis of variance, as presented in Table III for dataset FCSIT and Table IV for dataset FEB, to statistically demonstrate the differences among all employed methods, BVND, PVND, ITVND, WOA-VD, and WOA-IVD, presented in Table II. The results for both datasets indicate a statistically significant difference between the tested approaches, with a p-value below 0.001.

TABLE III. ANOVA FOR FCSIT DATASET OF THE ALGORITHMS

Source	DF	Sum of Squares	Mean Square	F	Signature
Between Groups	4	132.209	33.052	11.772	<.001
Within Groups	145	407.131	2.808		
Total	149	539.340			

TABLE IV. ANOVA FOR FEB DATASET OF THE ALGORITHMS

Source	DF	Sum of Squares	Mean Square	F	Signature
Between Groups	4	198.958	49.739	29.289	<.001
Within Groups	145	246.247	1.698		
Total	149	445.204			

Fig. 1 and 2 present the box plots for two datasets generated using Tableau software. The lower mean, median, and distribution values in both box plots show that WOA-IVD is consistently the best method. The fact that its box plot height is lower than other algorithms in both datasets further demonstrates this. For the FCSIT dataset, as shown in the box plot in Fig. 1, WOA-VD performs the worst due to its higher spread and maximum values, suggesting poor performance in worst-case scenarios. For the FEB dataset, as shown in the box plot in Fig. 2, BVND has the largest spread and includes higher maximum values, indicating it performs less reliably and worse in the worst-case scenarios. The proposed WOA-IVD approach effectively balances the objectives of exploration and exploitation, as the tested approaches' performance ranks BVND, PVND, WOA-VD, ITVND, and WOA-IVD.

The results highlight the consistent effectiveness of our discrete WOA method, achieving an objective value reduction of approximately 12.41% for the FCSIT dataset and 12.58% for the FEB dataset. This consistency underscores the method's reliability and adaptability, making it a practical solution for diverse scenarios. However, the proposed discrete modified WOA may be constrained by the need to adapt the local search method in the WOA model, which is currently tailored to our ETP problem instance and might require modification to address different problem constraints or domains.

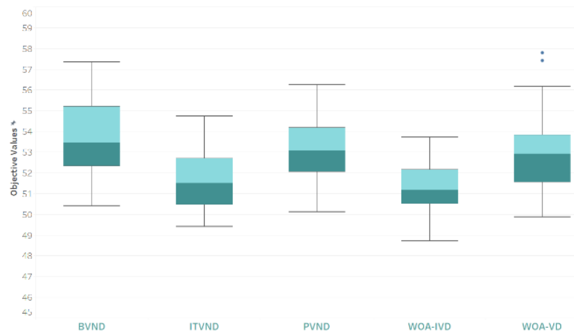

Fig. 1. Box plots of objective values for FCSIT dataset.

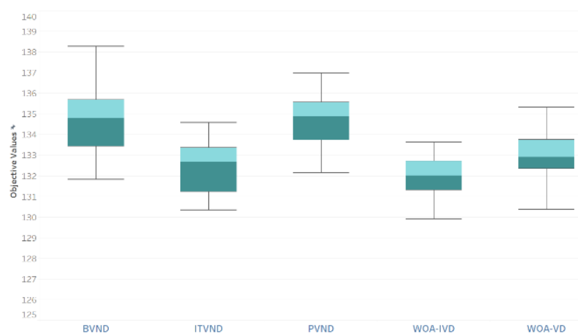

Fig. 2. Box plots of objective values for FEB dataset.

VII. CENTRALIZED AND DECENTRALIZED

A comparison of centralized and decentralized approaches to university exam timetabling during the pandemic was conducted by Modirkhorasani and Hoseinpour [31], focusing on minimizing costs and ensuring social distancing, with the study underscoring the advantages of decentralization. Building on this and given that our current practice employs a decentralized approach at the faculty level, we intend to conduct a similar comparison using our objective function for cost evaluation to better align with our specific operational context. A comparative analysis is conducted to evaluate the effects of three scheduling methods:

1) Decentralized approach: the current practice whereby faculty schedule their timetables independently.

2) Centralized approach: All faculty exams are scheduled, with resources managed centrally. It employs a uniform 12-timeslot structure, ensuring consistency in completing exam sessions across datasets.

3) Decentralized approach with re-optimization: Similar to the first approach, only that re-optimization is performed after post-resource reallocation. If a faculty's exam session concludes earlier than others, the shared resources are reallocated for use by other faculties that take longer exam sessions.

The comparative analysis aims to determine the relative impacts of these approaches on scheduling efficiency and overall outcomes. The centralized and decentralized approaches are compared in Table V, with and without the re-optimization strategy for the decentralized approach. The comparison is made across four soft constraints (*S1*–*S4*) based on their objective values. The bold formatting indicates the best values achieved by the approaches.

TABLE V. PERFORMANCE OF SCHEDULING UNDER CENTRALIZED AND DECENTRALIZED APPROACHES

Value	Decentralized		Centralized
	Without Re-optimization	With Re-optimization	
S1	92.0	**82.3**	86.0
S2	37.2	43.9	**19.5**
S3	33.9	**25.4**	29.1
S4	16.0	20.4	**9.8**
Total	179.3	172.0	**144.5**

The decentralized approach with re-optimization has shown superior efficiency in resource allocation, as evidenced by its lowest values in both *S1* and *S3*, which pertain to room usage and splitting. However, it is less effective for other constraints, such as *S2* and *S4*, which are related to spread and preferred slot, where it performs worse than the decentralized approach without re-optimization. While re-optimization enhances performance in certain aspects, it may not universally improve outcomes across all constraints. Despite this, the centralized approach remains the most efficient overall.

For the decentralized approach, values without re-optimization are generally higher, indicating that re-optimization improves efficiency. In contrast, the centralized approach consistently produces lower values than both decentralized approaches, underscoring its overall efficiency. The total objective values further support this trend, with the centralized approach achieving the lowest total (144.5), followed by the decentralized approach with re-optimization (172.0) and without re-optimization (179.3). It shows a reduction of approximately 15.9% in the total objective value, calculated as (172.0–144.5) / 172.0 × 100. This improvement highlights the effectiveness of the centralized approach in enhancing the solution's overall quality. However, the comparative analysis of centralized and decentralized approaches is incomplete, as it encompasses only a limited subset of faculties rather than the entire scope, thereby leaving this aspect open for further exploration.

VIII. CONCLUSION

This study examines decentralized faculty exam timetabling to optimize resource allocation and satisfy institutional constraints while designing an approach that can be adopted across multiple faculties. The ETP under consideration accommodates two distinct exam modes and formulations within the same timetable. This structure is notably different from those commonly found in literature and, to our knowledge, has not been previously studied. We propose two approaches: ITVND, an improved version of VND, and a novel discrete WOA. Specifically, we embedded the different local search strategies in the WOA algorithm to ensure they work well in the discrete scheduling domain. We used a real-world dataset to validate the proposed algorithm's practicality, highlighting its applicability across various faculties while adhering to their specific constraints. Our search methods have been rigorously tested and compared internally and against proprietary software developed using heuristic and manual methods. These comparisons highlight that the discrete WOA outperforms other approaches, demonstrating superior performance, though it takes slightly longer. While the preliminary results provide proof of concept, further experimentation with additional examination timetabling datasets, such as benchmark sets, could provide valuable insights. We consider hybridizing the WOA algorithm with other metaheuristic algorithms for future studies.

ACKNOWLEDGMENT

This work was funded by *i*-CATS University College under the *i*-CATS Research and Innovation Grant Scheme 2024.

REFERENCES

[1] Carter, Michael W., Gilbert Laporte, and Sau Yan Lee, "Examination timetabling: Algorithmic strategies and applications," Journal of the operational research society 47.3, 1996, pp.373-383. doi: 10.2307/3010580.

[2] McCollum B, McMullan P, Parkes AJ, Burke EK, Qu R, "A New Model for Automated Examination Timetabling," Annals of Operations Research 194:291–315, 2012, doi: 10.1007/s10479-011-0997-x.

[3] de Werra D, "The combinatorics of timetabling," European Journal of Operational Research 96(3):504–513, 1997, doi: 10.1016/S0377-2217(96)00111-7.

[4] E.G. Talbi, "Metaheuristics: From Design to Implementation," Wiley Online Library, 2009.

[5] Burke, E., Bykov, Y., Newall, W., Petrovic, S., "A time-predefined local search approach to exam timetabling problems," IIE Trans. 36 (6), pp. 509–528, 2004, doi: 10.1080/07408170490438410.

[6] Caramia, M., Dell'Olmo, P., "Coupling stochastic and deterministic local search in exam timetabling," Oper. Res. 55 (2), pp. 351–366, 2007, doi: 10.1287/opre.1060.0354.

[7] Cheong, C.Y., Tan, K.C., Veeravalli, B., "A multi-objective evolutionary algorithm for exam timetabling," J. Sched. 12, pp. 121–146, 2007, doi: 10.1007/s10951-008-0085-5.

[8] Dammak, A., Elloumi, A., Kamoun, H., "Classroom assignment for exam timetabling," Adv. Eng. Softw. 37, pp. 659–666, 2006, doi: 10.1016/j.advengsoft.2006.02.001.

[9] Akbulut, A., Yilmaz, G., "University exam scheduling system using graph coloring algorithm and rfid technology," Int. J. Innov. Manag. Technol. 4 (1), 66, 2013, doi: 10.7763/IJIMT.2013.V4.359.

[10] Hassan, Mohammad Al-Haj, Osama Al-Haj Hassan, "Constraints aware and user friendly exam scheduling system," Int. Arab J. Inf. Technol. 13 (1A), pp. 156–162, 2016.

[11] Kahar, M.N.M., Kendall, G., "The exam timetabling problem at Universiti Malaysia Pahang: Comparison of a constructive heuristic with an existing software solution," Eur. J. Oper. Res. 207 (2), pp. 557–565, 2010, doi: 10.1016/j.ejor.2010.04.011.

[12] Cataldo, A., Ferrer, J. C., Miranda, J., Rey, P. A., & Sauré, A., "An integer programming approach to curriculum-based examination timetabling," Ann Oper Res 258, pp. 369–393, 2017, doi: 10.1007/s10479-016-2321-2.

[13] Ozturk, Z. K., Gundogan, H. S., Mumyakmaz, E., & Kececioglu, T., "Exam scheduling under pandemic conditions: A mathematical model and decision support system," Technological Forecasting and Social Change, 208, 123687, 2024, doi: 10.1016/j.techfore.2024.123687.

[14] S. Mirjalili, A. Lewis, "The whale optimization algorithm," Adv. Eng. Softw. 95, pp. 51–67, 2016, doi: 10.1016/j.advengsoft.2016.01.008.

[15] Hashim, F. A., Hussain, K., Houssein, E. H., Mabrouk, M. S., & Al-Atabany, W., "Archimedes optimization algorithm: a new metaheuristic algorithm for solving optimization problems," Applied intelligence, 51, 1531-1551, 2021, doi:10.1007/s10489-020-01893-z.

[16] Ashraf, A., Anwaar, A., Haider Bangyal, W., Shakir, R., Ur Rehman, N., Qingjie, Z., "An Improved Fire Hawks Optimizer for Function Optimization," In: Tan, Y., Shi, Y., Luo, W. (eds) Advances in Swarm Intelligence. ICSI 2023. Lecture Notes in Computer Science, vol 13968. Springer, Cham, 2023, doi:10.1007/978-3-031-36622-2_6.

[17] Mahmood, S., Bawany, N.Z., Tanweer, M.R., "A comprehensive survey of whale optimization algorithm: modifications and classification," Indones. J. Electr. Eng. Comput. Sci. 29 (2), 899, 2023, doi: 10.11591/ijeecs.v29.i2.pp899-910.

[18] Nadimi-Shahraki, M.H., Zamani, H., Asghari Varzaneh, Z., Mirjalili, S., "A Systematic Review of the Whale Optimization Algorithm: Theoretical Foundation, Improvements, and Hybridizations," Arch. Comput. Methods Eng. 30, pp. 4113–4159, 2023, doi: 10.1007/s11831-023-09928-7.

[19] Abdel-Basset, M., El-Shahat, D. & Sangaiah, "A modified nature inspired meta-heuristic whale optimization algorithm for solving 0–1 knapsack problem," Int. J. Mach. Learn. & Cyber. 10, pp. 495–514, 2019, doi: 10.1007/s13042-017-0731-3.

[20] Abdel-Basset, M., El-Shahat, D. & Sangaiah, "A modified nature inspired meta-heuristic whale optimization algorithm for solving 0–1 knapsack problem," Int. J. Mach. Learn. & Cyber. 10, 495–514 (2019). https://doi.org/10.1007/s13042-017-0731-3.

[21] Li, Y., He, Y., Liu, X. et al., "A novel discrete whale optimization algorithm for solving knapsack problems," Appl Intell 50, pp. 3350–3366, 2020, doi: 10.1007/s10489-020-01722-3.

[22] Majdi M. Mafarja, Seyedali Mirjalili, "Hybrid Whale Optimization Algorithm with simulated annealing for feature selection," Neurocomputing, 260, pp. 302–312, 2017, doi: 10.1016/j.neucom.2017.04.053.

[23] Tawhid, M.A., Ibrahim, A.M., "Feature selection based on rough set approach, wrapper approach, and binary whale optimization algorithm," Int. J. Mach. Learn. & Cyber. 11, pp. 573–602, 2020, doi: 10.1007/s13042-019-00996-5.

[24] Mafarja, M., Thaher, T., Al-Betar, M.A., Too, J., Awadallah, M.A., Abu Doush, I. and Turabieh, H., "Classification framework for faulty-software using enhanced exploratory whale optimizer-based feature selection scheme and random forest ensemble learning," Appl Intell 53, pp. 18715–18757, 2023, doi:10.1007/s10489-022-04427-x.

[25] T. Jiang, C. Zhang and Q. -M. Sun, "Green Job Shop Scheduling Problem With Discrete Whale Optimization Algorithm," in IEEE Access, vol. 7, pp. 43153-43166, 2019, doi: 10.1109/ACCESS.2019.2908200.

[26] Liu, M.; Yao, X.; Li, Y, "Hybrid whale optimization algorithm enhanced with Lévy flight and differential evolution for job shop scheduling problems," Appl. Soft Comput., 87, 105954, 2020, doi: 10.1016/j.asoc.2019.105954.

[27] Zhao, F.; Xu, Z.; Bao, H.; Xu, T.; Zhu, N., "A cooperative whale optimization algorithm for energy-efficient scheduling of the distributed blocking flow-shop with sequence-dependent setup time," Comput. Ind. Eng., 178, 109082, 2023, doi:10.1016/j.cie.2023.109082.

[28] Mladenovi´c, N., Hansen, P., "Variable neighborhood search," Comput. Oper. Res. 24, pp. 1097–1100, 1997, doi: 10.1016/S0305-0548(97)00031-2.

[29] Mjirda, A., Todosijevi´c, R., Hanafi, S., Hansen, P., Mladenovi´c, N., "Sequential variable neighborhood descent variants: an empirical study on the traveling salesman problem," Int. Trans. Oper. Res. 24, pp. 615–633, 2017, doi: 10.1111/itor.12282.

[30] Sze, S. N., Phang, M. H., & Chiew, K. L., "Real-Life Faculty Examination Timetabling to Utilise Room Used," Journal of Telecommunication, Electronic and Computer Engineering, 9(3-11), pp. 51-54, 2017.

[31] Modirkhorasani, A., & Hoseinpour, P., "Decentralized exam timetabling: A solution for conducting exams during pandemics," Socio-Economic Planning Sciences, 92, p.101802, 2024, doi:10.1016/j.seps.2024.101802.