Optimizing Faculty Workloads and Room Utilization using Heuristically Enhanced WOA

Lea D. Austero[1], Ruji P. Medina[4]
Graduate Programs
Technological Institute of the Philippines
Quezon City, Philippines

Ariel M. Sison[2]
School of Engineering and Technology
Emilio Aguinaldo College
Manila, Philippines

Junrie B. Matias[3]
College of Computing and Information Sciences
Caraga State University
Butuan City, Philippines

***Abstract*—The creation and generation of schedules that are free of conflicts manually every academic semester present higher education institutions with a duty that is laborious and demanding of their resources. The course timetabling optimization, as an education timetabling problem, is a popular example of an NP-hard combinatorial problem. Numerous attempts have been made over the course of the past few decades to find a solution to this problem, but no one has yet developed a foolproof approach that can examine all alternatives to find the best method. The promising swarm-based optimization algorithm called Whale Optimization Algorithm was heuristically enhanced in the present study and is called HEWOA. It was designed as a solution to the course timetabling problem discussed in the current study. HEWOA was able to generate an efficient timetable for the large dataset of 1700 events for an average time of 14.92 seconds only, with an average generation of 7.2 and a best time of 8.38 seconds. These results reveal that the performance of HEWOA was better than that of various hybrids of the Genetic Algorithm that was compared in the present study.**

Keywords*—Heuristics; mutation; optimization; swarm; timetabling; whale optimization algorithm

I. Introduction

The application of automated procedures to a time-consuming and resource-intensive task often leads to increased efficiency and productivity, as well as time and cost savings. Among these processes are the preparation and creation of academic schedules. Timetabling problem was solved manually through trial and error, but this was not the greatest option. At present, scientific methods are used to address the problem [1], [2]. Timetabling problems, better known as the university course timetabling problem (UCTTP), are known to be NP-hard, meaning the problem cannot be solved exactly in polynomial time as its size and complexity increase exponentially [3]. It involves allocating non-overlapping classes to given resources such as classrooms and teachers in space-time [4]. The number of courses, the average number of lectures per day, the desired free timeslots each day, and the targeted off-days in a week are a few of the constraints that influence the design of the educational timetable [1], [5]. In the scheduling problem, there are two types of constraints: hard constraints and soft constraints. Hard constraints are rules or restrictions that cannot be broken. Soft constraints are requirements that, if not violated, can improve the effectiveness of the timetable. A timetable is considered efficient if it is able to solve the problem while adhering to all of the hard constraints specified [6].

The process of scheduling classes is often carried out with the assistance of specialized models that are adapted to meet the requirements of the particular educational establishment in question. A significant amount of work devoted to scheduling makes use of streamlined models to investigate and evaluate the performance of various scheduling strategies. The vast majority of research on course scheduling focuses on modeling and computational results, with very little attention paid to actual implementation in the real world [7].

Several strategies have been used to solve course timetabling using benchmark and real-world datasets. This problem was solved over decades using optimization approaches. Heuristic approaches helped resolve timetabling's complex behavior and model [8]. Evolutionary methods are frequently used in solving course timetabling; however, these existing methods were not able to quickly tests all alternatives to find the best solution [8]–[10]. Recently, various research has employed the Whale Optimization Algorithm (WOA), which is appreciated as a simple, flexible, and competitive swarm-based metaheuristic algorithm [11][12][13]. Despite its potential, it has inherent flaws that must be addressed before it can effectively address optimization issues such as course timetabling. WOA, like most metaheuristic algorithms, struggles to have a balanced local and global search.

The present study is an application and enhancement of the algorithm used in the previous work of WOA [14] in solving course timetabling. In this work, WOA was integrated with heuristic mutation to improve further the performance of WOA in solving optimization problems such as timetabling. The aim of the present work is to introduce HEWOA as a heuristically enhanced variant of the WOA, which is used in solving course timetabling problems. A literature review is presented in the next section of this paper, which contains a discussion on timetabling, solutions for solving UCTTP, and WOA. The third section presents the particulars of the methodology, which includes the problem definition, the architecture, and the HEWOA. Section IV presented the observations, results, and discussions on the experiments conducted in solving the timetabling problem. Finally, the conclusion of this study is presented in Section V.

II. Literature Review

A. Timetabling

The process of allocating resources to discrete objects in space-time to achieve desired goals within a given set of constraints is called timetabling. Timetabling encompasses many research-intensive fields. In education timetabling, the timetabling for the course and examination is the most studied. It is challenging to execute a course scheduling solution with the same approach to a problem because each institution has unique characteristics and constraints or limitations [15]. Universities and colleges in the Philippines use a manual procedure to schedule classes, increasing work for program heads and making it difficult to analyze every timetable combination [16]. In each country, accrediting government and private agencies require state universities and colleges to adhere to specific policies and criteria for scheduling classes. One of the recommendations suggests limiting the number of preparations for each faculty member to no more than four distinct subjects so that they have just enough work to do during the semester. The quality of instruction may degrade if faculty members teach more subject courses than the university deems optimal [17], which also leads to student's poor academic performance[18]. This case is often violated in actual practice. There are also cases of an unbalanced allocation of workloads among faculty members, in which some have more than four preparations while others have fewer than four.

More research is encouraged in solving UCTTP as it is unique across institutions due to policies and regulations. A general solution that could solve all the concerns in UCTTP does not exist [9]. Apart from being effective, optimization algorithms should consider simplicity and adaptability to a range of varying real-world UCTTP [10]. Thus, adapting the implementation of state-of-the-art methods on real-world UCTTP is still open to be explored by researchers.

B. Solutions in Solving UCTTP

Over the past decade, several works have shown substantial advancements in timetabling techniques and algorithms. These methods were created to address either benchmark datasets or real-world datasets [9]. In terms of the quantity and hard and soft constraints, benchmark and real-world UCTTPs differ. Creation solutions for benchmark datasets are often generalized and intended for comparing algorithms. The benchmark datasets utilized in international timetabling competitions, such as Socha[19], ITC-02[20], and ITC-07 [21], are the most popular testbeds among researchers in comparing algorithms. Real-world UCTTP, on the other hand, emphasizes the applicability of solutions in academic institutions. Due to varied legislation, educational systems, and cultures, even real-world UCTTP vary in terms of their criteria [10].

Metaheuristic methods promise precise and optimal timetable scheduling solutions and are popular for timetabling and other optimization challenges. They are simple to implement, faster than the standard mathematical-based optimization process, and achieve optimal results [11]. Evolutionary methods like Genetic algorithms, ant colony, local search, simulated annealing, and tabu search are frequently used in course scheduling; however, none of these was considered the best [8], [9]. Hybrid techniques or combining two or more algorithms are also prevalent and have produced more high-quality outcomes than other techniques, as proven in prior studies [22]–[24]. Hybrid methods are appropriate for maximizing the benefits of separate techniques. Single solution-based meta-heuristics and population-based meta-heuristics are the most popular approaches for the benchmark UCTTP, while in the case of real-world or actual datasets such as the one used in this work, the most popular methods used include single solution-based meta-heuristic, Operations Research, population-based meta-heuristic, hyper-heuristic and hybrid approaches [9].

C. Whale Optimization Algorithm

The Whale Optimization Algorithm (WOA) is a swarm-based optimization algorithm that is inspired by the hunting behavior of humpback whales[13]. WOA is among the most promising and competitive optimization techniques [11], [25]. The whales, while encircling the prey, create specific bubbles along a circular path. The bubble-net attacking technique assists in exploitation. The prey for the search state of WOA represents the exploration phase. For exploitation, the whale position is updated using either spiral movement or shrinking encirclement. For exploration, the humpback whale finds the best solution and updates its position according to other whales. Having this inspiration, WOA is composed of three operators: encircling prey, bubble-net attacking method, and search prey.

The following are the relevant equations implemented by WOA on its operations. Eq. 1-4 captures the procedures for the encircling prey.

$$\vec{D} = |\vec{C}.\vec{X}^{*}(t) - \vec{X}(t)| \tag{1}$$

$$\vec{X}(t+1) = \vec{X}^{*}(t) - \vec{A}.\vec{D} \tag{2}$$

$$\vec{A} = 2\vec{a}.\vec{r} - \vec{a} \tag{3}$$

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

where t indicates the current iteration, $A^{\rightarrow}$ and $C^{\rightarrow}$ are coefficient vectors, and $X*$ is the position vector of the best solution obtained so far. $X^{\rightarrow}$ is the position vector, | | is the absolute value, and is an element-by-element multiplication, and $\vec{r}$ is a random vector in [0, 1]. It should be noted that $X*$ should be updated in each iteration if there is a better solution. The vectors $A^{\rightarrow}$ and $C^{\rightarrow}$ are calculated as shown in Eq. 3 and 4, respectively.

Bubble-net attacking method:

$$\vec{X}(t+1) = \vec{D}'.e^{bl}.\cos(2\pi l) + \vec{X}^{*}(t) \tag{5}$$

$$f(x) = \begin{cases} \vec{X}^{*}(t) - \vec{A}.\vec{D}, \; if\ p < 0.5 \\ \vec{D}'.e^{bl}.\cos(2\pi l) + \vec{X}^{*}(t), \; if\ p \geq 0.5 \end{cases} \tag{6}$$

Where p represents a constant for explaining the shape of the logarithmic spiral and l is a random number uniformly distributed in the range of [-1, 1].

and Search for prey:

$$\vec{D} = |\vec{C}.\vec{X}_{rand} - \vec{X}| \tag{7}$$

$$\vec{X}(t+1) = \vec{X}_{rand} - \vec{A}.\vec{D} \quad (8)$$

where $\vec{X}_{rand}$ is nominated arbitrarily from whales in the current iteration.

Many recent research has employed the Whale Optimization Algorithm (WOA), which is appreciated as a simple, flexible, and competitive swarm-based metaheuristic algorithm [11][12][13]. Whale bubble-net hunting inspired WOA's algorithm. Its effectiveness and adaptability attract researchers from many fields. It is used in electrical, computer, aeronautical, and construction engineering [26]. Despite WOA's promising features, it has some unavoidable flaws, including being designed for continuous search space [13], requiring too many parameters tuning [27], having no theoretical convergent property [28], and having a probability distribution that changes with iterations [12]. It may also prematurely converge, trapping it in local optima [29], [30]. WOA, like most metaheuristic algorithms, struggles to balance local and global searches. The present study is an application and enhancement of the algorithm used in the previous work of WOA [14] in solving course timetabling. In this work, WOA was integrated with heuristic mutation.

III. Methodology

A. Problem Definition

In the allocation of schedules to resources, the course timetabling problem must fulfill both hard and soft restrictions. This section presents the problem description and objective functions to be used in the implementation of the timetable. This study's constraints and objective functions are similar to the previous work [14] but were implemented with a different algorithm to enhance WOA.

- Timetable Guidelines

The number of courses, the average number of lectures per day, the desired free timeslots each day, and the desired off-days in a week are a few of the variables that influence the design of an educational timetable. Each course in a curriculum is a class which could be a lecture, a lab, or both. Each of these classes is allocated a teacher and a classroom at a time that should not conflict with other classes scheduled for that day. Classrooms are utilized either for lecture classes or laboratory work. The mathematical formulation must fulfill all the relevant variables in order to produce a timetable that is both efficient and feasible. The following variables have been taken into consideration throughout this study. Let:

- E be the set of scheduled classes for a teacher and students with specified courses and classrooms,
- S be the set of students grouped through blocks in a program,
- T be the set of teachers wherein each teacher can handle many courses with a maximum of 4 unique subjects to handle,
- C is the set of courses wherein each lecture unit in a course is equivalent to one hour, whereas each laboratory unit is equivalent to three hours of class;
- R be the set of rooms, either lecture or laboratory, that will be assigned to classes, and;
- K is the set of period slots in a day from 7:30 in the morning to 7:30 in the evening, wherein the days of the week are paired as Monday-Thursday, Tuesday-Friday, and Wednesday-Saturday.
- Constraints

The constraints in course timetabling problems are classified as soft and hard. Soft constraints are optional, while hard constraints must be satisfied completely [9], [13]. For this work, the minimum requirements set as policies by the Accrediting Agency of Chartered Colleges and Universities in the Philippines and the Commission on Higher Education for state universities such as Bicol University are also taken into account in these constraints. Table I presents the constraints considered in this study which were identified as the most common constraints being used in solving course timetabling [9].

TABLE I. THE CONSTRAINTS

Code	Type	Description
H1	Hard	No teacher may be assigned to the same group of students in two separate classes.
H2	Hard	A teacher should handle only one course in one classroom at each time slot.
H3	Hard	Exactly only one class is assigned per timeslot per day in a classroom.
H4	Hard	The size of the classroom should be considered.
H5	Hard	For all required courses for a group of students must be given a scheduled
H6	Hard	All the teaching periods required in the curriculum must be given a schedule
H7	Hard	An uninterrupted period of time required for a class should be assigned precisely on a given day.
S1	Soft	At least one timeslot in a day should be vacant for a group of students
S2	Soft	The maximum number of straight classroom teaching hours is three in a schedule
S3	Soft	The total number of teaching hours in a day should not exceed 6 hours.

- Objective Functions

The degree to which the timetable can satisfy the constraints effectively will determine how much each solution will cost. In each generation of the candidate solution, a time slot is allotted to a pair of student and faculty groups associated with a classroom and a course. This is done with the goal of satisfying all hard requirements while keeping the expense of satisfying soft constraints to a minimum. The problem may also be expressed using the formulation that follows. Let X be the set of all possible solutions, $HC = \{h_1 \dots h_6\}$ the set of hard constraints, $SC = \{s_1 \dots s_5\}$ the set of soft constraints and $x \subseteq X$ the set of all candidate solutions.

The goal of this procedure is to find the most efficient timetable with the least cost, as presented in Eq. 1 by the penalty charged per violation of a hard constraint:

$$hp(x) = \sum_{t\in T} f_1(x,t) + \sum_{t\in T} f_2(x,t) + \sum_{r\in R} f_3(x,r) + \sum_{r\in R} f_4(x,r) + \sum_{s\in S} f_5(x,s) + \sum_{s\in S} f_6(x,s) + \sum_{s\in S} f_7(x,s) \quad (9)$$

TABLE II. THE OBJECTIVE FUNCTIONS FOR THE HARD CONSTRAINTS

Function	Penalty instance	Purpose
$f_1(x,t)$	Two or more classes are assigned to one teacher to handle	Ensures no conflict in faculty loading
$f_2(x,t)$	More than one teacher is assigned to a class	Ensures that only one teacher is assigned to a class
$f_3(x,r)$	One classroom is assigned with multiple classes.	Ensures no conflict in the classroom assignment
$f_4(x,r)$	If the classroom size is not considered.	Ensures that the classroom can accommodate the class size
$f_5(x,s)$	a course for a block of students has no schedule assigned	Ensures that all courses enrolled by the students are assigned a schedule
$f_6(x,s)$	The number of required hours on the curriculum is not satisfied.	Ensures that the required hours offered in the curriculum are equal to the scheduled classes
$f_7(x,s)$	More than a total of six hours of teaching load for a teacher	Ensures that there is a balance distribution of workload within a week

The penalty function for hard constraints is represented by Equation 9, and the value of the objective function for each solution can be determined as follows when soft constraints are also considered:

$$f(x) = \left(\sum_{s\in S} f_8(x,s) + \sum_{t\in T} f_9(x,t) + \sum_{t\in T} f_{10}(x,t)\right) + (hp(x) \times W) \quad (10)$$

The function $f(x)$ shown in Eq. 10 represents the sum of all penalties from hard and soft constraints. The function examines each solution for possible violations on the constraints wherein a value of one (1) will be given had there been violations; otherwise, zero (0). It can be observed that there is a variable W that is applied to the hard constraints $hp(x)$, where a value of three (3) will be multiplied by the counted violations on hard constraints. This was done so that the objective function would give more weight to violations on hard constraints.

TABLE III. THE OBJECTIVE FUNCTIONS FOR THE SOFT CONSTRAINTS

Function	Penalty instance	Purpose
$f_8(x,s)$	There is no vacant period for a day to a group of students	Ensures that students have at least one vacant period daily
$f_9(x,s)$	The total teaching hours for a course totaled to more than 3 hours	Ensures that the schedules are not straight and that having such would be tiresome both for the students and teachers.
$f_{10}(x,s)$	the total teaching hours to be handled by a teacher is more than 6 hours	Ensures that the schedules are distributed throughout the week.

B. Architecture

The courses and classes to be included in the scheduling were collected from various colleges at Bicol University. The lecture and laboratory rooms available for utilization and faculty members teaching these courses were also identified. These raw data were used in the generation of the timetable. Various sizes of datasets were considered for the experimentations to test the algorithms: 400, 800, 1200, and 1700 as the largest dataset.

Shown in Fig. 1 is the framework of the method for generating an optimized and fairly distributed timetable. The classes to be scheduled, faculty schedule, room availability, and specified constraints serve as the inputs. These data will be processed using HEWOA as objective functions. A penalty will be applied to violations of constraints, and the process will be repeated through generations until the cost approaches zero to produce an efficient and optimized timetable.

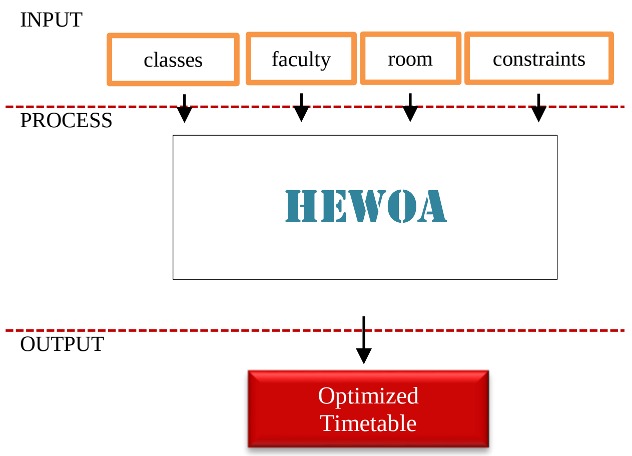


Fig. 1. Framework of the Study.

C. The Heuristically Enhanced Whale Optimization Algorithm

In this work, preliminary results implemented in solving course timetabling using WOA indicate that updating the entire solution using these traditional procedures of WOA has a greater likelihood of destroying rather than improving it. This is because the whale operator is designed to update all values using the same parameter values, and the event number is large. Additionally, the more constraints applied to a problem, the more complicated the search process becomes. A solution emerged during the experimentation: when the first procedure (encircling prey) was not used, the performance of WOA was improved. This is due to the nature of the equation, which has a high probability of destroying the solution. Moreover, the heuristic mutation operator is integrated into the process to enhance further WOA's exploration and exploitation capability.

Fig. 2 now depicts the pseudocode of HEWOA, taking into account the aforementioned modifications to the method as well as the section where the heuristic mutation seen in Fig. 3 will be implemented. It is shown that the encircling procedure was removed from the process.

Instead of updating the position of the current search whale using the encircling prey process, remove it, retaining the Bubble-net attacking method for the exploitation phase and searching for prey for the exploration phase.

```
Initialize the whales population $X_i$(i = 1, 2,...,n)
Calculate the fitness of each search agent
X*=the best search agent
  while (t < maximum number of iterations)
    for each search agent
    Update a, A, C, l, and p
      if1 (p<0.5)
        if2(|A| < 1)
          • Select a random search agent (Xrand)
          • Update the position of the current search
            agent by the Eq. 8
      else if1 (p ≥ 0.5)
        Update the position of the current search by
        the Eq. 5
      end if1
      Update the position of the current search
      agent using Heuristic Mutation (Fig. 4)
    end for
    Check if any search agent goes beyond the search
    space and amend it Calculate the fitness of each
    search agent Update X* if there is a better
    solution
    t=t+1
  end while
return X*
```

Fig. 2. The HEWOA Pseudocode.

Accordingly, in WOAs encircling prey section, search agents update to the best agent. It is easy to trap the algorithm in a local solution, causing premature convergence [31]. Consequently, updating the position of the current search agent using heuristic mutation is integrated to further improve the algorithm's exploitation and exploration capability. In this work, the heuristic mutation focuses on invalid classes and repairs them using random pairs of room and period.

- The Encoding Method

The genetic operator avoids illegal offspring by encoding all events in each candidate solution in the same index, as shown in Fig. 3.

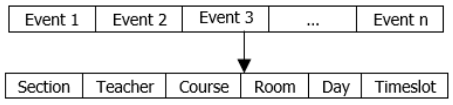


Fig. 3. The Representation of the Solution.

The subset of solutions in the current generation is a 3-dimension array containing scheduled events. Each of these events contains the codes of other constraints such as student section, teacher, course, room, day, and timeslot. This data structure is similar to the one used in the work of [32]. In addition, prioritizing constraints are applied, and the events of the teacher who has the highest workload and the events of the section of students who have the most classes that need to be scheduled are attached to the solution first.

- Fitness Function

The whale's fitness is the weighted sum of penalty cost based on Equations 9 and 10. Every constraint violation incurs a penalty of 1. However, the cost for hard constraints is multiplied by the weight value W such that the algorithm prioritizes these constraints while finding and ranking all candidate solutions. In addition, the impact of soft constraints will highlight the superior solution since it meets preferences. In other words, a solution with the same cost in hard constraints will be differentiated by the number of violations on soft constraints.

- Heuristic Mutation

Mutation modifies genes to create new individuals. The heuristic mutation is designed to produce better offspring wherein a set of chromosomes is transformed from a parent by exchanging some genes (neighborhood) [33]. In Fig. 4, the operator gets all invalid gene indexes and then mutates them.

The operator does not implement mutation probability parameters in mutating invalid genes since it generates ten percent random genes or classes. The operator selects one of these randomly generated genes that fulfill stringent criteria. If no valid gene is found, the room or periods are altered at random (day and timeslots). It guarantees that randomly selected times are distinct from nearby occurrences in order to ensure diversity. Additionally, the rate of ten percent random genes or classes can be decreased or increased during the configuration or before running the algorithm.

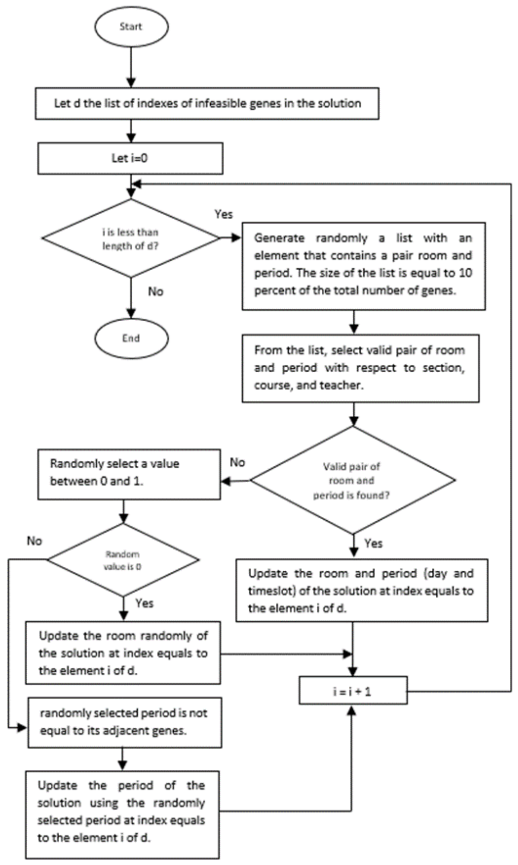

Start
Let d the list of indexes of infeasible genes in the solution
Let i=0
i is less than length of d?
Yes
No
End
Generate randomly a list with an element that contains a pair room and period. The size of the list is equal to 10 percent of the total number of genes.
From the list, select valid pair of room and period with respect to section, course, and teacher.
Valid pair of room and period is found?
No
Yes
Randomly select a value between 0 and 1.
Random value is 0
No
Yes
Update the room and period (day and timeslot) of the solution at index equals to the element i of d.
Update the room randomly of the solution at index equals to the element i of d.
i = i + 1
randomly selected period is not equal to its adjacent genes.
Update the period of the solution using the randomly selected period at index equals to the element i of d.

Fig. 4. Heuristic Mutation Applied to WOA.

IV. Results and Discussion

Using real-world data, the efficiency of HEWOA was evaluated. These are actual datasets that include the courses to be scheduled for the various programs offered by Bicol University, Legazpi City, Albay, Philippines, as well as the rooms being utilized for the conduct of classes and the assigned faculty that will handle the class. The HEWOA uses a population of 10 whales and will stop generating solutions if all hard constraints are satisfied by any whale or when it reaches an iteration of 1000.

TABLE IV. PERFORMANCE OF THE HEWOA COMPARED TO OTHER METHODS

Methods	Events	Average Generation	Best Time	Average Time	SD
GA Using Heuristic Mutation	400	3.4	0.590s	1.061s	0.353
	800	5.2	2.370s	**3.455s**	0.650
	1200	8.0	6.430s	8.554s	1.402
	1700	15.5	16.670s	21.670s	4.247
GA Using Invalid Genes Focused Random Resetting Mutation	400	4.1	1.280s	1.595s	0.198
	800	7.0	5.145s	7.608s	1.475
	1200	14.0	16.670s	23.015s	3.979
	1700	33.9	55.700s	81.355s	16.402
HEWOA	400	**2.4**	**0.458s**	**0.74s**	0.160
	800	**4.1**	**2.009s**	3.514s	0.812
	1200	**5.8**	**5.299s**	**8.362s**	1.964
	1700	**7.2**	**8.384s**	**14.942s**	4.671
EWOA [14]	117	-	-	3s	-
	195	-	-	15s	-
	304	-	-	270s	-
Guided GA [34]	878	136.5	-	-	-
	1140	409.5	-	-	-
Parallel GA and Local Search [35]	166	900	-	-	-
Greedy and Genetic Fusion Algorithm [36]	300	900	-	-	-

Table IV shows the various sizes of events or the number of classes used to test the various techniques. As part of the execution of the generating schedules in the timetable, the objective functions (Eq. 9) are executed, which assess penalties (Tables II and III) for schedules that violate the constraints stated in Table I. These costs determine the fitness function. Initial generations or runs would incur corresponding penalties, which would gradually decrease until they approached zero, at which point there would be no violations.

The performance of the HEWOA was compared to other competitive methods that solved course timetabling: GA using heuristic mutation; GA using invalid genes focused on random resetting mutation; guided GA [34], parallel GA and local search [35]; and greedy and genetic fusion algorithm [36]. The result in Table IV of their performances in terms of total execution time is based on ten (10) runs per method. In terms of average generation, HEWOA was able to perform better compared to other work which uses Hybrid GA as the base method. In terms of execution time, HEWOA was also able to perform better except on a dataset with 800 classes.

Fig. 5, 6, and 7 illustrate the pace at which the indicated techniques are approaching closer to zero, which is defined as the state in which no more penalties are incurred on the constraints. The research indicates that HEWOA is the approach with the fastest pace among these solutions, followed by GA with heuristic mutation. The least performing method is the GA when random resetting mutation is applied to infeasible genes.

An example of manually generated faculty workloads that shows unoptimized class schedules and loading is shown in Fig. 8. It can be observed in the workload that on Tuesday and Friday, the faculty had teaching hours of more than 6 hours. Thus, violating soft constraint 3, the total number of teaching hours per day should not exceed 6 hours. The same scenario can be repeated with other faculty since that method of plotting schedules is manual, which is prone to errors, especially when the schedulers are adjusting or transferring loads from one faculty to another.

Table V shows an example of faculty workloads generated by HEWOA. It can be seen that the workloads are equally distributed in the week, although the schedules do not show the consultation hours. It is also noted that that days are paired, like Monday-Thursday and Tuesday-Friday. In each timeslot, the total hour is one and a half hours, and the other one and a half hours are to be lectured on a corresponding pair day.

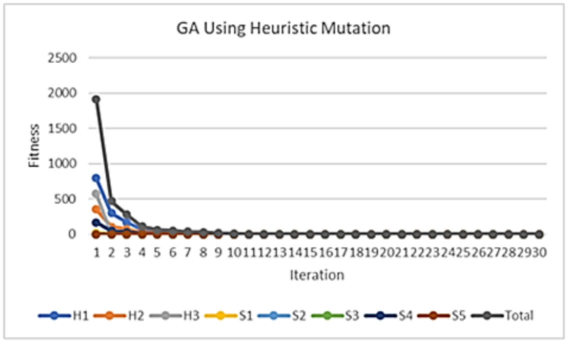


Fig. 5. Performance of Genetic Algorithm when a Heuristic Mutation is Applied.

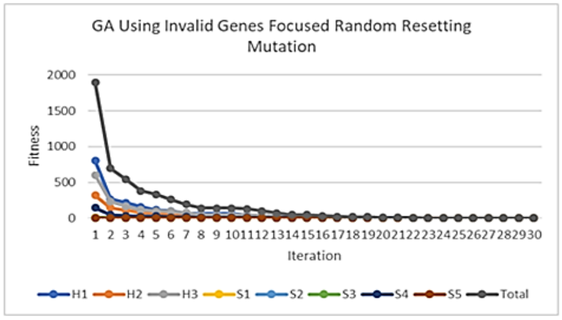


Fig. 6. Performance of Genetic Algorithm when Random Resetting Mutation is Applied on Infeasible Genes.

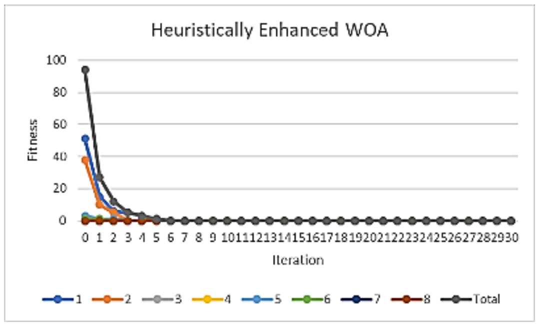
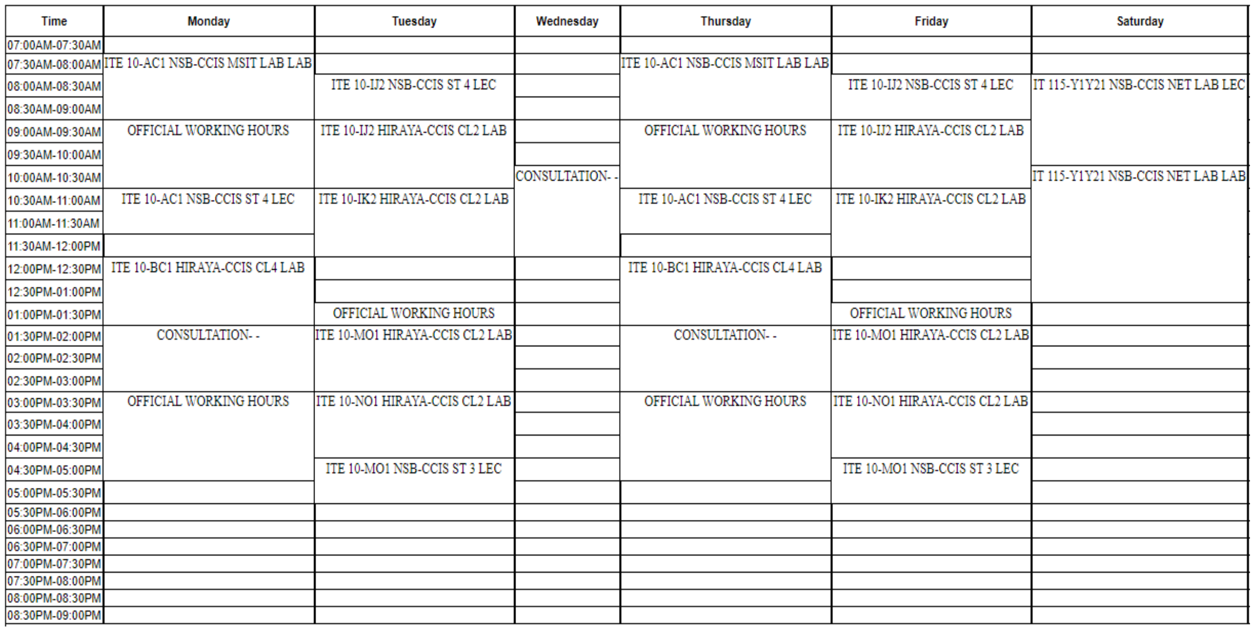


Fig. 7. Performance of HEWOA when the Heuristic Mutation is Integrated.



Time	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
07:00AM-07:30AM						
07:30AM-08:00AM	ITE 10-AC1 NSB-CCIS MSIT LAB LAB			ITE 10-AC1 NSB-CCIS MSIT LAB LAB		
08:00AM-08:30AM		ITE 10-IJ2 NSB-CCIS ST 4 LEC			ITE 10-IJ2 NSB-CCIS ST 4 LEC	IT 115-Y1Y21 NSB-CCIS NET LAB LEC
08:30AM-09:00AM						
09:00AM-09:30AM	OFFICIAL WORKING HOURS	ITE 10-IJ2 HIRAYA-CCIS CL2 LAB		OFFICIAL WORKING HOURS	ITE 10-IJ2 HIRAYA-CCIS CL2 LAB	
09:30AM-10:00AM						
10:00AM-10:30AM			CONSULTATION- -			IT 115-Y1Y21 NSB-CCIS NET LAB LAB
10:30AM-11:00AM	ITE 10-AC1 NSB-CCIS ST 4 LEC	ITE 10-IK2 HIRAYA-CCIS CL2 LAB		ITE 10-AC1 NSB-CCIS ST 4 LEC	ITE 10-IK2 HIRAYA-CCIS CL2 LAB	
11:00AM-11:30AM						
11:30AM-12:00PM						
12:00PM-12:30PM	ITE 10-BC1 HIRAYA-CCIS CL4 LAB			ITE 10-BC1 HIRAYA-CCIS CL4 LAB		
12:30PM-01:00PM						
01:00PM-01:30PM		OFFICIAL WORKING HOURS			OFFICIAL WORKING HOURS	
01:30PM-02:00PM	CONSULTATION- -	ITE 10-MO1 HIRAYA-CCIS CL2 LAB		CONSULTATION- -	ITE 10-MO1 HIRAYA-CCIS CL2 LAB	
02:00PM-02:30PM						
02:30PM-03:00PM						
03:00PM-03:30PM	OFFICIAL WORKING HOURS	ITE 10-NO1 HIRAYA-CCIS CL2 LAB		OFFICIAL WORKING HOURS	ITE 10-NO1 HIRAYA-CCIS CL2 LAB	
03:30PM-04:00PM						
04:00PM-04:30PM						
04:30PM-05:00PM		ITE 10-MO1 NSB-CCIS ST 3 LEC			ITE 10-MO1 NSB-CCIS ST 3 LEC	
05:00PM-05:30PM						
05:30PM-06:00PM						
06:00PM-06:30PM						
06:30PM-07:00PM						
07:00PM-07:30PM						
07:30PM-08:00PM						
08:00PM-08:30PM						
08:30PM-09:00PM						

Fig. 8. Example of Faculty Workloads Generated Manually Captured from the University Database.

TABLE V. EXAMPLE OF FACULTY WORKLOADS GENERATED BY HEWOA CAPTURED FROM THE UNIVERSITY DATABASE

Faculty	**Section**	**Course Code**	**Room**	**Day**	**Day Description**	**Timeslot**	**Timeslot Description**
114	BSIS 1-105	ITE 10	18.03	1	Mon-Thu	3	10:30-12:00
114	BSIT 2-113	IS 105	18.04	1	Mon-Thu	4	12:00-13:30
114	BSIS 1-105	ITE 10	7.02	1	Mon-Thu	8	18:00-19:30
114	BSIS 1-107	ITE 10	18.01	2	T-TF	3	10:30-12:00
114	BSIT 2-113	IS 105	7.02	2	T-TF	4	12:00-13:30
114	BSIS 3-111	IS 116	7.01	2	T-TF	6	15:00-16:30

On room utilization, plotting classes manually can result in an inefficient room allocation. These would result in more classroom usage; it could increase the cost of maintenance and energy. In this work, the utilization of classrooms and laboratories is optimized since the HEWOA can produce timetables with fewer rooms compared to classrooms. For example, when we retrieved the schedules in one semester with 1700 classes, the classrooms and laboratories utilized were more than 120 compared to schedules generated by HEWOA and GA, which only used 91.

A. Implications

Population-based metaheuristics such as Genetic Algorithms, Particle Swarm Optimization, and Ant Colony Optimization are superior to other methods in solution space exploration. Still, these approaches require a higher processing time necessary to generate solutions of high quality [37]. A good quality solution means no violations of the specified constraints whose purpose is explained and presented in Tables II and III. It was observed in the presented results in Table IV and Fig. 5, 6, and 7 that HEWOA was able to generate good-quality solutions that require lesser computational time. HEWOA is a multi-objective implementation of WOA and can be used for other similar solutions.

The typical technique of timetabling is inefficient, manual, and not very robust against changes [10]. As a result, the number of course conflicts is considerable, which reduces the effectiveness of the instruction [38], [39]. The utilization of an intelligent approach for UCTTP has been the subject of much research and widespread debate worldwide. Using the method in the present work would aid in developing efficient software for course scheduling in academic institutions.

V. CONCLUSION

This paper introduces HEWOA, a heuristically enhanced Whale Optimization Algorithm designed to solve UCTTP. The results of the experiments on various sizes of real-world data indicate that both GA and HEWOA could generate a feasible and efficient timetable that could satisfy all the identified constraints set by educational institutions. Generating class schedules using GA and HEWOA can optimize classroom and laboratory utilization which could help decrease the cost of maintenance and energy. Optimized classroom and laboratory utilization could also help increase the number of students since there would be more available resources for classes. Faculty workloads also can be improved using automated scheduling; thus, it helps satisfy soft constraints and enhance the quality of schedules.

Moreover, it is observed that HEWOA outperformed Hybrid GAs [34] and other methods [35], [36] in terms of execution time and average solution generation for the majority of utilized event sizes.

Lastly, future work would include testing the performance of this method using benchmark UCTTP, which provides different constraints than the real-world data used in this study.

ACKNOWLEDGMENT

The authors express their appreciation and gratitude to Bicol University and Caraga State University for their cooperation in the conduct of the study. The same gratitude is given to the CHED for the scholarship grant, which provided aid to this work.

REFERENCES

[1] R. Ganguli and S. Roy, "A Study on Course Timetable Scheduling using Graph Coloring Approach," 2017. [Online]. Available: http://www.ripublication.com

[2] R. R. Iwańkowicz and M. Taraska, "Self-classification of assembly database using evolutionary method," Assembly Automation, vol. 38, no. 3, pp. 268–281, May 2018, doi: 10.1108/AA-06-2017-071.

[3] M. Nouiri, A. Bekrar, A. Jemai, S. Niar, and A. C. Ammari, "An effective and distributed particle swarm optimization algorithm for flexible job-shop scheduling problem," J Intell Manuf, vol. 29, no. 3, pp. 603–615, Mar. 2018, doi: 10.1007/s10845-015-1039-3.

[4] Ş. Gür and T. Eren, "Scheduling and planning in service systems with goal programming: Literature review," Mathematics, vol. 6, no. 11, Nov. 2018, doi: 10.3390/math6110265.

[5] M. Andrey, V. Voronkin, P. Svetlana, and S. Alexey, "Software and hardware infrastructure for timetables scheduling in university," Oct. 2018, pp. 15–20. [Online]. Available: http://ceur-ws.org

[6] J. S. Tan, S. L. Goh, G. Kendall, and N. R. Sabar, "A survey of the state-of-the-art of optimisation methodologies in school timetabling problems," Expert Syst Appl, vol. 165, Mar. 2021, doi: 10.1016/j.eswa.2020.113943.

[7] M. Mühlenthaler, "Real-world academic course timetabling," Lecture Notes in Economics and Mathematical Systems, vol. 678, pp. 107–128, 2015, doi: 10.1007/978-3-319-12799-6_4.

[8] H. Alghamdi, T. Alsubait, H. Alhakami, and A. Baz, "A Review of Optimization Algorithms for University Timetable Scheduling," 2020. [Online]. Available: www.etasr.com

[9] M. C. Chen, S. N. Sze, S. L. Goh, N. R. Sabar, and G. Kendall, "A Survey of University Course Timetabling Problem: Perspectives, Trends and Opportunities," IEEE Access, vol. 9, pp. 106515–106529, 2021, doi: 10.1109/ACCESS.2021.3100613.

[10] R. A. Oude Vrielink, E. A. Jansen, E. W. Hans, and J. van Hillegersberg, "Practices in timetabling in higher education institutions: a systematic review," Ann Oper Res, vol. 275, no. 1, pp. 145–160, Apr. 2019, doi: 10.1007/s10479-017-2688-8.

[11] U. Can and B. Alatas, "Performance comparisons of current metaheuristic algorithms on unconstrained optimization problems," Periodicals of Engineering and Natural Sciences, vol. 5, no. 3, pp. 328–340, 2017, doi: 10.21533/pen.v5i3.120.

[12] Y. Ling, Y. Zhou, and Q. Luo, "Lévy Flight Trajectory-Based Whale Optimization Algorithm for Global Optimization," IEEE Access, vol. 5, pp. 6168–6186, 2017, doi: 10.1109/ACCESS.2017.2695498.

[13] S. Mirjalili and A. Lewis, "The Whale Optimization Algorithm," Advances in Engineering Software, vol. 95, pp. 51–67, May 2016, doi: 10.1016/j.advengsoft.2016.01.008.

[14] L. D. Austero, A. M. Sison, J. B. Matias, and R. P. Medina, "Solving course timetabling problem using Whale Optimization Algorithm," in 2022 8th International Conference on Information Technology Trends (ITT), May 2022, pp. 160–166. doi: 10.1109/ITT56123.2022.9863951.

[15] J. B. Matias, A. C. Fajardo, and R. P. Medina, "A fair course timetabling using genetic algorithm with guided search technique," in 2018 5th International Conference on Business and Industrial Research (ICBIR), 2018, pp. 77–82.

[16] H. Bellardo, "Preference-driven university course scheduling system," 2010.

[17] The New Times, "Heavy workload affects quality of teaching," https://www.newtimes.co.rw/author/36/times-reporter, Sep. 19, 2017.

[18] I. Gwambombo, "The Effect of teacher's workload on students' academic performance in community secondary schools," 2013.

[19] R. Lewis and B. Paechter, "Finding feasible timetables using group-based operators," IEEE Transactions on Evolutionary Computation, vol. 11, no. 3, pp. 397–413, Jun. 2007, doi: 10.1109/TEVC.2006.885162.

[20] Y. Bykov, "The description of the algorithm for international timetabling competition," 2014. [Online]. Available: https://www.researchgate.net/publication/228388131

[21] L. di Gaspero, B. Mccollum, and A. Schaerf, "The Second International Timetabling Competition (ITC-2007): Curriculum-based Course Timetabling (Track 3) Multiobjective Timetabling with Fairness View project PHD works View project The Second International Timetabling Competition (ITC-2007): Curriculum-based Course Timetabling (Track 3)-preliminary presentation," 2007. [Online]. Available: http://www.idsia.ch/

[22] S. L. Goh, G. Kendall, and N. R. Sabar, "Improved local search approaches to solve the post enrolment course timetabling problem," Eur J Oper Res, vol. 261, no. 1, pp. 17–29, Aug. 2017, doi: 10.1016/j.ejor.2017.01.040.

[23] P. Daru Kusuma and A. Sayid Albana, "University Course Timetabling Model in Joint Courses Program to Minimize the Number of Unserved Requests," IJACSA) International Journal of Advanced Computer Science and Applications, vol. 12, no. 10, pp. 121–127, 2021, [Online]. Available: www.ijacsa.thesai.org

[24] S. Al-Negheimish, F. Alnuhait, H. Albrahim, S. Al-Mogherah, M. Alrajhi, and M. Hosny, "An Intelligent Bio-Inspired Algorithm for the Faculty Scheduling Problem," IJACSA) International Journal of Advanced Computer Science and Applications, vol. 9, no. 5, 2018, [Online]. Available: www.ijacsa.thesai.org

[25] H. M. Mohammed, S. U. Umar, and T. A. Rashid, "A Systematic and Meta-Analysis Survey of Whale Optimization Algorithm," Computational Intelligence and Neuroscience, vol. 2019. Hindawi Limited, 2019. doi: 10.1155/2019/8718571.

[26] N. Rana, M. S. A. Latiff, S. M. Abdulhamid, and H. Chiroma, "Whale optimization algorithm: a systematic review of contemporary applications, modifications and developments," Neural Computing and Applications, vol. 32, no. 20. Springer Science and Business Media Deutschland GmbH, pp. 16245–16277, Oct. 01, 2020. doi: 10.1007/s00521-020-04849-z.

[27] R. Sivalingam, S. Chinnamuthu, and S. S. Dash, "A modified whale optimization algorithm-based adaptive fuzzy logic PID controller for load frequency control of autonomous power generation systems," Automatika, vol. 58, no. 4, pp. 410–421, 2017, doi: 10.1080/00051144.2018.1465688.

[28] R. Salgotra, U. Singh, and S. Saha, "On Some Improved Versions of Whale Optimization Algorithm," Arab J Sci Eng, vol. 44, no. 11, pp. 9653–9691, Nov. 2019, doi: 10.1007/s13369-019-04016-0.

[29] H. S. Alamri, Y. A. Alsariera, and K. Z. Zamli, "Opposition-based Whale Optimization Algorithm," J Comput Theor Nanosci, vol. 24, no. 10, pp. 7461–7464, 2018, doi: 10.1166/asl.2018.12959.

[30] N. Singh and H. Hachimi, "A New Hybrid Whale Optimizer Algorithm with Mean Strategy of Grey Wolf Optimizer for Global Optimization," Mathematical and Computational Applications, vol. 23, no. 1, p. 14, Mar. 2018, doi: 10.3390/mca23010014.

[31] K. Lu and Z. Ma, "A modified whale optimization algorithm for parameter estimation of software reliability growth models," Journal of Algorithms & Computational Technology, vol. 15, p. 17483026211034442, 2021.

[32] J. B. Matias, A. C. Fajardo, and R. P. Medina, "Examining genetic algorithm with guided search and self-adaptive neighborhood strategies for curriculum-based course timetable problem," in 2018 Fourth International Conference on Advances in Computing, Communication & Automation (ICACCA), 2018, pp. 1–6.

[33] W. Ho and P. Ji, “A hybrid genetic algorithm for component sequencing and feeder arrangement,” J Intell Manuf, vol. 15, no. 3, pp. 307–315, Jun. 2004, doi: 10.1023/B:JIMS.0000026569.88191.46.

[34] M. Fachrie and A. F. Waluyo, “Guided Genetic Algorithm to Solve University Course Timetabling with Dynamic Time Slot,” in 2020 3rd International Seminar on Research of Information Technology and Intelligent Systems, ISRITI 2020, Dec. 2020, pp. 583–587. doi: 10.1109/ISRITI51436.2020.9315448.

[35] D. Kristiadi and R. Hartanto, “Genetic Algorithm for lecturing schedule optimization,” IJCCS (Indonesian Journal of Computing and Cybernetics Systems), vol. 13, no. 1, p. 83, Jan. 2019, doi: 10.22146/ijccs.43038.

[36] K. Wang, W. Shang, M. Liu, and W. Lin, “A Greedy and Genetic Fusion Algorithm for Solving Course Timetabling Problem,” in 2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS), 2018, pp. 344–349.

[37] C. K. Teoh, A. Wibowo, and M. S. Ngadiman, “Review of state of the art for metaheuristic techniques in Academic Scheduling Problems,” Artif Intell Rev, vol. 44, no. 1, pp. 1–21, Jun. 2015, doi: 10.1007/s10462-013-9399-6.

[38] Ç. H. Erdoğan and R. Topuz, “Investigation of Stress Perceptions of Physical Education Teachers,” Asian Journal of Education and Training, vol. 6, no. 2, pp. 144–148, 2020, doi: 10.20448/journal.522.2020.62.144.148.

[39] J. A. Bowden and P. J. Green, “Completion mindsets and contexts in doctoral education: Pursuing efficiency and quality with integrity,” 2019, pp. 77–99. doi: 10.1007/978-981-13-6990-2_5.