Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 234 (2024) 1714–1721

Procedia Computer Science

www.elsevier.com/locate/procedia

Seventh Information Systems International Conference (ISICO 2023)

Development of a Web-based Course Timetabling System based on an Enhanced Genetic Algorithm

Dexter Romaguera[a], Jenie Plender-Nabas[a], Junrie Matias[a]*, Lea Austero[b]

[a]*Caraga State University, Butuan City, Philippines*
[b]*Bicol University, Legazpi City, Philippines*

Abstract

This paper presents the development of a web-based course timetabling system based on an enhanced genetic algorithm. The enhanced method utilizes a heuristic mutation which concentrates on mutating the infeasible genes to improve the algorithms' exploration and exploitation capability. The method was implemented using a free and open-source application and can be accessed online. Based on the actual datasets from Caraga State University, the enhanced method optimized the use of classroom resources by using a smaller number of rooms. The generated timetable is more efficient as it satisfies not just hard constraints, which are conflicting schedules, but also soft constraints.

Keywords: Course Timetabling; Enhanced Genetic Algorithm; Metaheuristics

1. Introduction

Course timetabling presents many challenges, especially in under-resourced educational institutions. Course timetabling demands that, at the beginning of each semester, classes be assigned to specific days, timeslots, and teachers in a limited classroom [1] while satisfying several objectives [2], [3]. Classes consist of activities involving teachers, students, and a course. Typically, the activities occur in a particular classroom and must be arranged so that no two groups of students are scheduled simultaneously. The course timetabling problem is a combinatorial optimization and NP-complete problem that has been addressed analytically and heuristically [2], [4]. Finding an exact

* Corresponding author.
E-mail address: jbmatias@carsu.edu.ph

10.1016/j.procs.2024.03.177

or perfect solution using conventional optimization techniques is challenging due to the variations in the fast growth of students' numbers, policies, constraints (hard and soft), and objectives across the different institutions [5]. The work [2] suggests pursuing heuristic approaches and minimizing the cost of unsatisfied soft constraints.

Consequently, many universities in the Philippines are still creating course timetables and allocating teaching loads manually. Timetables are maintained in a database for viewing, analysis, and administration [6]. However, directly changing classrooms and reassigning teachers to other teachers may result in conflicts and violation of other preferences like unavailability and preference. The most common issues are teachers' prepared timeslots that are not satisfied, and some of them are given poor or imbalanced schedules like the first class being plotted in the earliest timeslot of the day and the last class of the same day being scheduled on the last time slot on the same day. Hence, the teachers prepare to have a workload that is evenly distributed in a week.

Changes in constraints and goals among institutions make it challenging to reconfigure or create new class schedules [7]. Numerous metaheuristic methods are implemented to solve various course timetabling and other fields of scheduling problems [8]. A strategy based on the metaheuristic method does not depend on the nature of the problem in any way. The metaheuristic approach is classified into two types which is the population-based and single trajectory-based solutions. Because of their capacity to address a wide range of real-world problems, meta-heuristics methods have gained increasing popularity over the years [9]–[13].

One popular metaheuristic method based on population that has been proven successful in solving many timetabling problems is the Genetic Algorithm (GA) [14]. GA is a general-purpose search method based on natural genetics principles. Because they can efficiently search a vast space of potential solutions, genetic algorithms are a popular solution for timetabling problems [15]. However, many challenges and constraints are associated with using genetic algorithms for timetabling. One of the causes is the genetic operators' destructive nature can recombine the solution, violating further limitations and resulting in considerable computational time in generating a feasible solution [16]. This can be traced to the operators of the algorithms being stochastic could recombine the solutions and not meet the constraints of the problem [16]–[18]. Another challenge is the constraints and the objective function, the timetabling problem may involve multiple constraints, and it can be challenging to balance conflicting objectives, such as minimizing conflicts and other preferences [16], [19]. As the number of resources and constraints increases, GA can become less effective and require more computational time [16], [19]. GAs can solve timetabling problems but need proper problem formulation and parameter optimization.

To address these limitations, this study proposes a unique mutation operator based on a heuristic to enhance the computational performance of the genetic Algorithm. This heuristic-based mutation operator will concentrate on mutating the infeasible genes. The proposed method was tested using a dataset gathered from Caraga State University. Lastly, a web-based timetabling system suitable for higher educational institutions in the Philippines is developed based on the proposed approach. The system can generate optimal and feasible course schedules in a short amount of time. Program administrators and coordinators can view and analyze timetables in the web-based interface and update them.

2. Problem Definition

Based on [20], curriculum-based course timetabling is defined as a problem that involves arranging n events (classes, courses) $E = \{e_1, e_2 \dots e_n\}$ is assigned to any of n rooms $R = \{r_1, r_2 \dots r_n\}$, and h periods $P = \{p_1, p_2 \dots p_h\}$, according to a set of constraints. Each event e has a set number of lectures l_i and a teacher. A period p is a day-timeslot pair. The total number of scheduling periods is the sum of days and timeslots each day. There are also q curricula $C = \{c_1, c_2 \dots c_q\}$, where the curriculum c_i will enroll a group of courses such that any pair of courses in the group have the same students, thus, events must not conflict.

2.1. Constraints

A timetable is feasible or valid if it schedules all classes and satisfies all the hard constraints. In this work, the hard constraints considered are the following:

1. All classes of a course must be scheduled in distinct periods.
2. Two classes cannot be assigned in the same room simultaneously.

3. All classes belonging to the same curriculum or taught by the same teacher must be scheduled in distinct periods.

Soft constraints, on the other hand, are useful criteria that do not necessarily need to be satisfied. Because these are preferences and not physical conflicts, the more these parameters are met, the better the timetable. These constraints may be violated if no other viable solution exists. If a timetable violates a soft constraint, a penalty is applied. The following soft constraints are shown in this work:

1. Rooms might not be available during certain periods.
2. Faculty might not be available during certain periods.
3. A group of students might not be available during certain periods.
4. The day's first and last time slots cannot be assigned to a faculty on the same day.
5. Classes cannot be scheduled on Saturdays for a group of students at the first-year level.

Additionally, the soft constraints discussed in this work include the teachers' preferred timeslots, the unavailability of students because certain student groups are occasionally not permitted to have a class at a particular time, and the unbalanced distribution of the schedule of the faulty load in a week, to avoid that a faculty can have a schedule of the first timeslots and the last, meaning they will be the first to enter the university and be the last to leave.

2.2. Objective Function

The course timetabling problem aims to find a feasible timetable by satisfying all hard constraints while minimizing the cost of violated soft constraints. The objective function $f(s)$ for a timetable s is the weighted sum of the number of hard-constraint violations $\#hcv$ and soft-constraint violations $\#scv$, which was used in [21], as defined in Equation (1), where W is the weight of the penalty cost for every hard constraint.

$$f(s) := \#hcv(s) * W + \#scv(s) \tag{1}$$

Consequently, on each soft constraint's violation, a penalty of 1 is given, and one also for hard constraints multiplied by W, where W has a value greater than 1. Hence, the hard constraints must be penalized more than soft constraints so that the Algorithm will prioritize solving the hard constraints because it represents the validity of the solution.

3. Proposed Method

Fig. 1 shows how a course scheduling system is contableured. The system has three components: a database management system, an automatic schedule generator, and a user interface module.

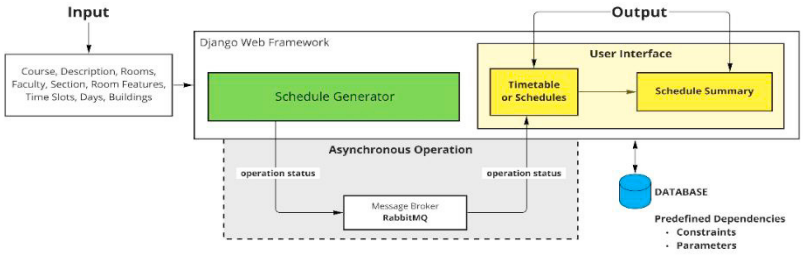

Fig. 1 Configuration of a Course Scheduling System

The first component is the Database Management System (DBMS), which contains the constraints and parameters that reflect subjects, teachers, student groups, and resources such as classrooms and laboratories. Typically, the DBMS also acts as a constraint store containing the constraints that must be satisfied by the schedules. The second component

is the schedule generator based on the enhanced Genetic Algorithm, which generates the timetables. The schedule generator used a message broker to keep the user updated on the progress of the creation of the timetable, which can take some time depending on the number of classes, classrooms, instructors, and other preferences. The third component is a web-based interface module allowing the user to upload courses to schedule, view, and modify the system-generated schedule. The administrator or decision-maker can then conduct what-if analysis and update the timetable.

3.1. Genetic Algorithm Using Heuristic Mutation

In this work, the initial population is generated randomly, where each individual or chromosome is arranged in a 2d-array form, where each chromosome represents a timetable. The chromosomes comprise genes consisting of a course, teacher, section, classroom, and period (day and timeslot). The genes are in integer formats and used to generate and search for feasible and optimal solutions.

A uniform crossover operator of two individuals is used to create two offspring by randomly selecting two parents and will swap genes using equal probability. Accordingly, the uniform crossover is effective for many problems numerical optimization problems [22].

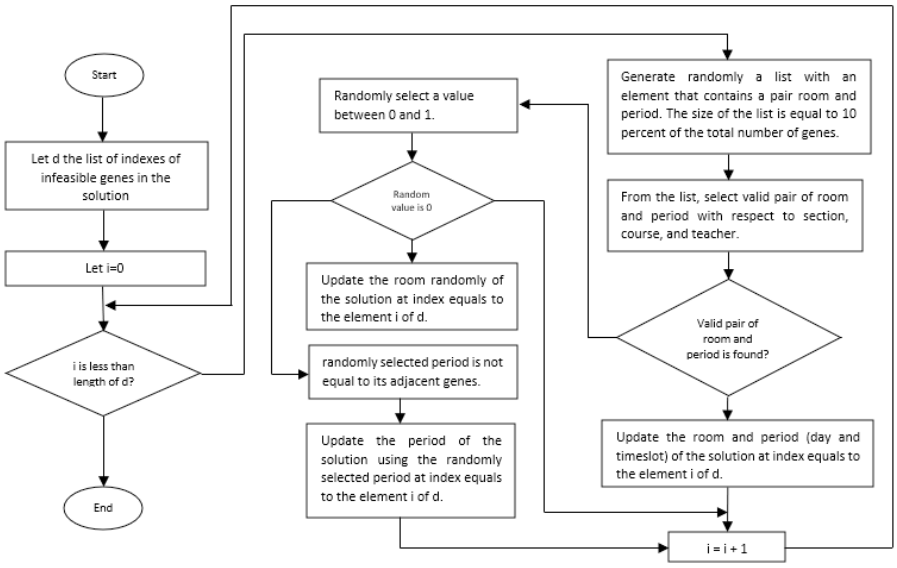

Fig. 2 Flow of the Heuristic Mutation

In mutating each candidate solution in the population (see Fig. 2), the heuristic mutation operator will generate random genes equivalent to 10 percent of the total genes or classes. Out of these randomly generated genes, the operator will select a feasible or valid gene that satisfies all hard constraints. If no valid gene is found, it will randomly change the room or its periods (day and timeslots). However, before updating these genes, the operator will ensure that the randomly selected periods do not equal their adjacent genes or classes.

Finally, the new offspring are evaluated using the fitness function shown in Equation (2) and will replace the weak individuals in the population. The fitness function indicates how near a specific solution satisfies the objectives. Each chromosome contains a set of classes, and each class that causes the constraint violation penalizes the fitness function.

$$fitness = 1/(1 + f(s)) \tag{2}$$

Moreover, the solution's fitness is based on the sum of penalties from violated hard and soft constraints, as defined in Equation (1), where f(s) is the cost of violating hard and soft constraints.

4. Results and Discussion

The system was tested using actual datasets from the Caraga State University-Main Campus in Butuan City, Philippines, composed of 1-semester curriculum-based enrollment data from six colleges and departments with 27 academic programs, excluding graduate program offerings. The dataset was separated into four groups to evaluate the algorithms' performance on varying data sizes, and the classes will be dispersed over 48 timeslots per week, Monday through Saturday. The administrator manually predetermines teachers in every class, and the automated system's task is to find feasible classrooms and timeslots for every course or class.

Table 1. Performance of the approaches in terms of the number of iterations and generation time based on different datasets.

Methods	No. of Classes/ Courses	Actual Classrooms Utilized	Best Utilized Classrooms	Best Iteration	Average Iteration	Std	Best Time	Average Time (seconds)	Std
GA Using Heuristic Mutation	*400*	88	48	**2.000**	**3.400**	0.843	**0.590**	**1.061s**	0.353
	800	104	66	**3.000**	**5.200**	**1.229**	**2.370**	**3.455s**	**0.650**
	1200	117	88	**6.000**	**8.000**	**1.414**	**6.430**	**8.554s**	**1.402**
	1700	132	103	**12.000**	**15.500**	**3.028**	**16.670**	**21.670s**	**4.247**
GA using Heuristic Mutation to Random Selected Genes with a probability of 0.01	*400*	88	55	212.000	358.600	77.547	43.100	74.632s	17.364
	800	104	77	1326.000	2080.889	1207.758	751.490	1185.929s	667.650
	1200	117		-	-		-	-	-
	1700	132		-	-		-	-	-
GA Using Invalid Genes Focused Random Resetting Mutation	*400*	88	50	3.000	4.100	**0.632**	1.280	1.595s	**0.198**
	800	104	71	5.000	7.000	1.449	5.145	7.608s	1.475
	1200	117	92	10.000	14.000	**2.539**	16.670	23.015s	3.979
	1700	132	114	24.000	33.900	6.420	55.700	81.355s	16.402

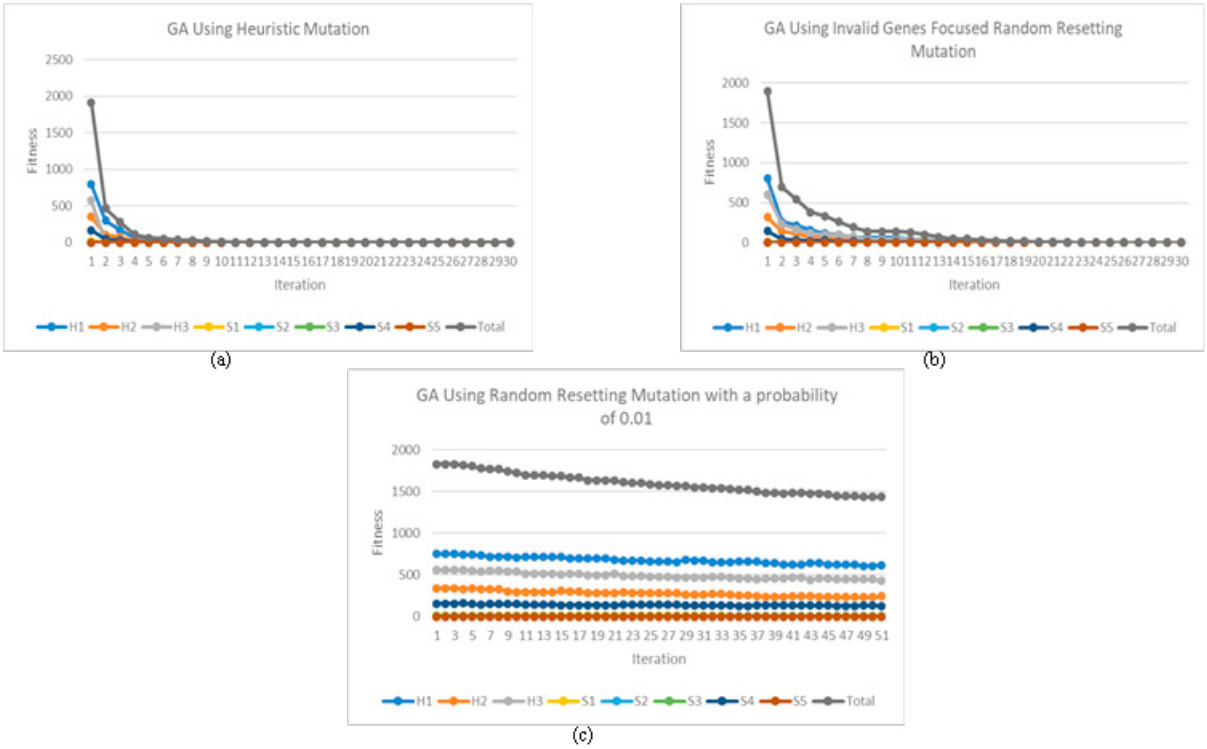

Fig. 3 Fitness values and the number of violations per constraint in every generation

4.1. Performance analysis

The results shown in Table 1 demonstrate that the GA that uses the Heuristic Mutation is preferable to the other mutation method. The optimal and average times and the number of iterations required to generate a solution are in bold font. The methods also use fewer classrooms than the actual schedule generated manually, and GA using heuristic mutation, utilized fewer classrooms. However, in Table 1, the penalty cost is not reflected since all constraints are satisfied by the method since the datasets came from the previous semester, which contain only a few preferences, especially for the teachers. Furthermore, the automated scheduling system provides timetables that utilize fewer classrooms than the manual process, maximizing the utilization of resources like power and other consumables.

Similarly, Fig. 3 shows that in generating a timetable for 1700 classes, the method in (a) outperforms both methods (b) and (c) in terms of the number of iterations. Method (a) can reduce the number of violations on three hard (H1-H3) and five soft (S1-S5) constraints while requiring fewer iterations. Moreover, the results are based on the average of 10 generations/runs using four different data sizes.

4.2. Graphical User Interface

Fig. 4 illustrates an interface with system-generated timetables, displaying information such as the date created, the number of courses/classes, and the number of conflicting schedules. To generate timetables, the users can upload a file in a comma-separated values (CSV) format containing constraints such as sections, teachers, rooms, and other preferences. After the timetable is generated, the user can update the class schedule manually.

Fig. 4 Generated Timetables

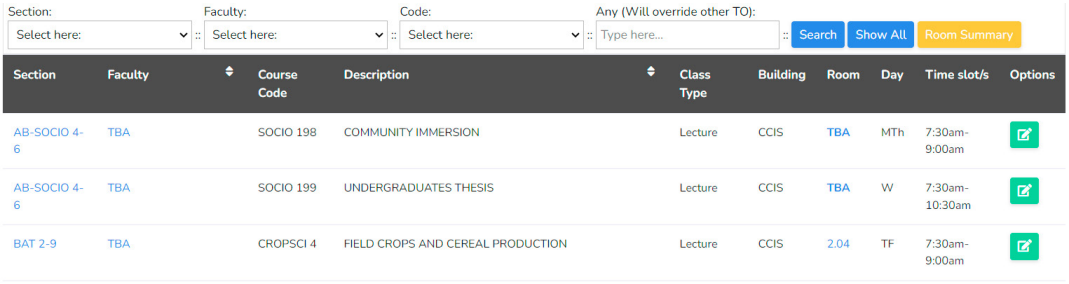

Fig. 5 List of Auto-Generated Class Schedule

Fig. 5 shows the class schedule generated, which displays the class section, teachers/faculty, course, type of class (lecture or laboratory), room and building, and timeslots. The list also has an edit button, allowing users to change and update the class schedule manually. The changes, however, are not saved if the manually entered values violate one or more hard constraints. Shown also in Fig. 6 were workloads (a) produced automatically by the system and (b) produced manually using spreadsheets. Although in (b), the consultation hours and official working are reflected, (b) presented an example of a faculty schedule that could violate some of the faculty preferences. The workload includes a timetable where the faculty teaches the first and final periods, meaning they arrive at school first and leave last. In addition, the Saturday work schedule is one that many employees wish to avoid. These violations occur regularly each semester due to the challenges of manual scheduling.

Time	Monday	Tuesday	Wednesday
7:30am-9:00am			
9:00am-10:30am			
10:30am-12:00nn			AB-SOCIO 2-5, SOCIO 103 CCIS , Room: 6.08 , LEC
12:00nn-1:30pm	BSAF 2-45, SOCIO 116 NSB , Room: 10.09 , LEC	AB-SOCIO 4-6, SOCIO 124 CCIS , Room: 6.06 , LEC	AB-SOCIO 2-5, SOCIO 103 CCIS , Room: 6.08 , LEC
1:30pm-3:00pm	AB-SOCIO 2-4, SOCIO 103 CCIS , Room: 6.05 , LEC		
3:00pm-4:30pm	AB-SOCIO 4-6, SOCIO 114 CCIS , Room: 6.05 , LEC		
4:30pm-6:00pm			BSAF 2-44, SOCIO 116 NSB , Room: 10.1 , LEC
6:00pm-7:30pm			BSAF 2-44, SOCIO 116 NSB , Room: 10.1 , LEC

(a)

Time	Monday	Tuesday	Wednesday
07:00AM-07:30AM			
07:30AM-08:00AM	CSC 104-AE1 HIRAYA-CCIS CL1 LAB	ITE 12-IK2 BH-BH 123 LAB	
08:00AM-08:30AM			
08:30AM-09:00AM			
09:00AM-09:30AM	OFFICIAL WORKING HOURS	ITE 12-JK2 BH-BH 123 LAB	
09:30AM-10:00AM			
10:00AM-10:30AM			
10:30AM-11:00AM	CONSULTATION- -	ITE 12-IK2 BH-CCIS ST 1 LEC	
11:00AM-11:30AM			
11:30AM-12:00PM		OFFICIAL WORKING HOURS	
12:00PM-12:30PM	CSC 104-DE1 HIRAYA-CCIS CL1 LAB		
12:30PM-01:00PM			
01:00PM-01:30PM	OFFICIAL WORKING HOURS	OFFICIAL WORKING HOURS	
01:30PM-02:00PM			
02:00PM-02:30PM	CSC 104-AE1 BH-BH 126 LEC		
02:30PM-03:00PM			
03:00PM-03:30PM	CONSULTATION- -		
03:30PM-04:00PM			
04:00PM-04:30PM	OFFICIAL WORKING HOURS		
04:30PM-05:00PM	IT 111-GY2Y1 HIRAYA-CCIS CL2 LAB		
05:00PM-05:30PM			
05:30PM-06:00PM			
06:00PM-06:30PM	IT 111-HY2Y1 HIRAYA-CCIS CL2 LAB		
06:30PM-07:00PM			
07:00PM-07:30PM			

(b)

Fig. 6 Sample faculty teaching load and schedules; (a) automated, (b) manual

4.3. Implications

Proper course scheduling is vital when providing a high level of service to an educational institution's most important clients: students. Due to limited classroom space and human resources, manual course timetabling against faculty and room schedules is inefficient [23]. Universities and colleges strive to decrease academic inefficiencies by automating scheduling [24].

Automated class scheduling allows for more effective and time-saving production of class schedules compared to the manual process [25]. Because of the automated approach, the person in charge of the program or the people assigned to the course schedule no longer needs to plot class schedules manually. The system will automatically generate balanced timetables that satisfy the hard constraints and minimize soft constraints violation, saving time and other resources and contributing favorably to the academic institutions and student performance.

With the course scheduling system's efficiency, the academic program administrator can do other duties while providing class schedules that are balanced and conflict-free. Faculty members with balanced and conflict-free class schedules can effectively manage their time for other equally important responsibilities. Because of how easily class schedules can be generated for students and teachers, both have plenty of time to prepare for other tasks. The teaching staff can easily check their class schedule at least two to three weeks before the beginning of the semester. During this time, they can also prepare the necessary course prerequisites, such as course syllabi and other learning materials. This approach helps improve education delivery, which enhances students' academic performance.

Using an automated system for course scheduling helps reduce the time and effort in creating class schedules. Most colleges and universities manually plot schedules that take at least two (2) weeks. Various factors, including room availability, faculty schedules, existing workloads, and student schedules, are considered during this time. With automated course scheduling, the development of timetables is streamlined, resulting in schedules that are well-balanced and free of scheduling conflicts for both students and faculty members.

5. Conclusion and future work

The work presented in this paper is the implementation of an enhanced genetic algorithm that utilizes a heuristic mutation to solve the course timetabling problem. Based on the results, the performance of enhanced GA in solving curriculum-based timetabling using actual datasets produced a practical timetable with no violation of hard constraints while minimizing the soft constraints. Although the traditional GA and its operators can create feasible timetables, it takes significantly longer than when the GA employs the heuristic mutation strategy. Moreover, the method can

produce timetables with fewer classrooms than those created manually, allowing the institution to save cost, space, and resources.

Future research directions include testing the approach with additional or alternative soft constraints, and the method needs to be tested using larger datasets from other higher academic institutions and integrate different applicable rules and policies.

References

[1] Arratia-Martinez NM, Avila-Torres PA, Trujillo-Reyes JC. (2021) "Solving a university course timetabling problem based on aacsb policies," *Mathematics* **9(19)**.

[2] Babaei H, Karimpour J, Hadidi A. (2015) "A survey of approaches for university course timetabling problem," *Computers & Industrial Engineering* **86**:43-59.

[3] Obit JH, Ouelhadj D, Landa-Silva D, Vun TK, Alfred R. (2011) "Designing a multi-agent approach system for distributed course timetabling Evolutionary Non-Linear Great Deluge for University Course Timetabling," in *Proceedings of the 2011 IEEE Hybrid Intelligent Systems Conference*.

[4] Rossi-Doria O, Sampels M, Birattari M, Chiarandini M, Dorigo M, Gambardella LM, Knowles J, Manfrin M, Mastrolilli M, Paechter B, Paquete L. (2002) "A Comparison of the Performance of Different Metaheuristics on the Timetabling Problem," in *International Conference on the Practice and Theory of Automated Timetabling*.

[5] Gonsalves T, Oishi R. (2015) "Artificial Immune Algorithm for exams timetable," *Journal of Information Sciences and Computing Technologies* **4(2)**:287–296,.

[6] Soria-Alcaraz JA, Özcan E, Swan J, Kendall G, Carpio M. (2016) "Iterated local search using an add and delete hyper-heuristic for university course timetabling," *Applied Soft Computing* **40**:581–593.

[7] Abdelhalim EA, El Khayat GA. (2016) "A Utilization-based Genetic Algorithm for Solving the University Timetabling Problem (UGA)," *Alexandria Engineering Journal* **55(2)**:1395–1409.

[8] Agárdi A, Nehéz K, Hornyák O, Kóczy LT. (2021) "A hybrid discrete bacterial memetic algorithm with simulated annealing for optimization of the flow shop scheduling problem," *Symmetry (Basel)* **13(7)**.

[9] Smutnicki C, Pempera J, Rudy J, Żelazny D. (2015) "A new approach for multi-criteria scheduling," *Computers & Industrial Engineering* **90**:212–220.

[10] Alzaqebah M, Abdullah S. (2015) "Hybrid bee colony optimization for examination timetabling problems," *Computers & Operations Research* **54**:142–154.

[11] Oner A, Ozcan S, Dengi D. (2011) "Optimization of university course scheduling problem with a hybrid artificial bee colony algorithm," *Evolutionary Computation (CEC)*.

[12] Tarawneh HY, Ayob M, Ahmad Z. (2013) "A hybrid simulated annealing with solutions memory for curriculum-based course timetabling problem," *Journal of Applied Sciences* **13(2)**:262-269.

[13] Fong CW, Asmuni H, McCollum B, McMullan P, Omatu S. (2014) "A new hybrid imperialist swarm-based optimization algorithm for university timetabling problems," *Information Sciences* **283**:1-21.

[14] Abdelhalim EA, El Khayat GA. (2016) "A Utilization-based Genetic Algorithm for Solving the University Timetabling Problem (UGA)," *Alexandria Engineering Journal* **55(2)**:1395–1409.

[15] Mahmoodabadi MJ, Nemati AR. (2016) "A novel adaptive genetic algorithm for global optimization of mathematical test functions and real-world problems," *Engineering Science and Technology, an International Journal* **19(4)**:2002–2021.

[16] Pillay N. (2014) "A survey of school timetabling research," *Annals of Operations Research* **218(1)**:261–293.

[17] MirHassani SA, Habibi F. (2013) "Solution approaches to the course timetabling problem," *Artificial Intelligence Review* **39(2)**.

[18] Teoh CK, Wibowo A, Ngadiman MS. (2015) "Review of state of the art for metaheuristic techniques in Academic Scheduling Problems," *Artificial Intelligence Review* **44**:1–21.

[19] Juang YS, Lin SS, Kao HP. (2007) "An adaptive scheduling system with genetic algorithms for arranging employee training programs," *Expert Systems with Applications*.

[20] Hao JK, Benlic U. (2011) "Lower Bounds for the ITC-2007 Curriculum-Based Course Timetabling Problem," *European Journal of Operational Research*.

[21] Jat SN, Yang S. (2009) "A hybrid genetic algorithm and tabu search approach for post enrolment course timetabling," *Journal of Scheduling* **14(6):**617–637.

[22] Rezaeipanah A, Matoori SS, Ahmadi G. (2021) "A hybrid algorithm for the university course timetabling problem using the improved parallel genetic algorithm and local search," *Applied Intelligence* **51(1)**:467–492.

[23] Matias JB, Fajardo AC, Medina RP. (2018) "A hybrid genetic algorithm for course scheduling and teaching workload management," in *2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM)*:1–6.

[24] Duan Y, Lu W. (2021) "Automatic Course Scheduling System in Universities Based on Hybrid Genetic-Ant Colony," *Journal of Physics: Conference Series* **2066(1)**.

[25] Stallaert J. (1997) "Automated timetabling improves course scheduling at UCLA," *Interfaces* **27(4)**:67–81, 1997.