Optimizing Practical Room Scheduling Using Genetic Algorithms: A Timetabling Case Study at Telkom University

1st Auliya Nur Pratama
Department of Information System
Faculty of Industrial and System Engineering, Telkom University
Bandung, Indonesia
auliyapratama@student.telkomuniversity.ac.id

2nd Ekky Novriza Alam
Department of Information System
Faculty of Industrial and System Engineering, Telkom University
Bandung, Indonesia
ekkynovrizalam@telkomuniversity.ac.id

3rd Sinung Suakanto
Department of Information System
Faculty of Industrial and System Engineering, Telkom University
Bandung, Indonesia
sinung@telkomuniversity.ac.id

2024 4th International Conference on Electronic and Electrical Engineering and Intelligent System (ICE3IS) | 979-8-3503-7836-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICE3IS62977.2024.10776011

***Abstract*— Timetable scheduling is a complex process involving the fulfillment of hard and soft constraints. Many universities face challenges in scheduling practical rooms due to the high number of student practical classes and the limited availability of rooms. Often, scheduling is still done manually with tools like Office Spreadsheets. This research proposes the development of a room scheduling website with an optimization module based on genetic algorithms. The genetic algorithm is used to enhance the flexibility and effectiveness of scheduling by considering room availability, capacity, and equipment compatibility. The implementation results show increased efficiency and more balanced room usage distribution. Scenario testing demonstrates the system's capability to handle various conditions and detect conflicts, making the genetic algorithm an effective solution for managing complex scheduling, saving time and effort compared to conventional methods. This approach can significantly reduce the conventional scheduling time from several days to just a few minutes.**

Keywords*— *Genetic Algorithm, Room Optimization, Practical Room Mapping, Timetable Scheduling

I. INTRODUCTION

Timetable scheduling is a complex process that involves creating a schedule under a set of constraints: hard constraints, which are strict and non-negotiable, and soft constraints, which are desirable but flexible[1]. This complexity is further emphasized by research highlighting the challenges of traditional scheduling methods and the potential benefits of optimization techniques such as genetic algorithms[2]. An example of timetable scheduling in a university setting is the University Course Timetabling Problems, which includes managing lecturer availability, classroom allocation, time slot constraints, student conflicts, room capacity, and special requirements [3].

Several universities in Indonesia have rooms that are shared among multiple users, such as practical rooms. This sharing arrangement poses challenges in scheduling these rooms due to the lack of flexibility and system availability in the scheduling process, which is often still done manually using Office Spreadsheets. For instance, a case study at a faculty in Telkom University revealed that scheduling practical rooms took approximately 7 days. This challenge not only requires a considerable amount of time but also involves risks such as overcrowding in certain rooms and uneven room usage, which can result from unforeseen room utilization.

Timetable scheduling is an NP-hard problem, presenting significant challenges for traditional methods[4]. The Information Systems program at Sepuluh Nopember Institute of Technology faces issues such as time-consuming processes, limited flexibility, and suboptimal schedules with potential conflicts, leading to inefficient room usage[4]. Universidade de Lisboa suggests using a greedy algorithm to optimize scheduling and room occupancy[5]Similarly, Rajarata University of Sri Lanka recommends genetic algorithms to address issues like facility shortages and constraints, aiming for more efficient and error-reduced scheduling[6].

One approach to tackling these challenges is the application of genetic algorithms. Genetic algorithms are an optimization method inspired by biological evolution mechanisms proposed by Charles Darwin[7]. They are used to find optimal solutions in problems involving many variables and constraints. Genetic algorithms employ natural selection mechanisms to improve solutions through processes such as selection, crossover, and mutation.

Therefore, this research proposes the development of a scheduling website with a room allocation optimization module using genetic algorithms.

II. LITERATURE REVIEW

A. Timetable Scheduling

Timetable scheduling is the process of organizing academic schedules in various educational institutions such as universities, high schools, and similar entities. Timetable scheduling involves the effort to map out space and time to meet a series of constraints involving people, available time slots, and space availability [8]. This scheduling problem is considered complex due to the need to satisfy all these criteria while avoiding time and space conflicts. Thus, timetable scheduling is an NP-hard (nondeterministic polynomial time hard) problem, which is challenging to solve using traditional methods or manually [4]. This means finding an optimal solution or a solution that meets the existing constraints is very difficult, and the complexity of computation increases exponentially with the addition of more spaces, times, and other resources[8].

When performed manually, timetable scheduling can be time-consuming and prone to frequent conflicts. Therefore, in timetable scheduling, there are two types of constraints that must be addressed: hard constraints and soft constraints. Hard constraints are requirements that must be met and cannot be violated during the scheduling process to produce a feasible solution, while soft constraints are conditions that may be violated but should be fulfilled as much as possible to generate an optimal schedule. Thus, these constraints are identified to find a feasible solution[6]. Table 1 summarizes various optimization algorithms used in timetable scheduling, along with their advantages and disadvantages

TABLE I. OPTIMATIZATION ALGORITHMS FOR TIMETABLE SCHEDULING

No	Algorithm Type	Ease to Implement		Reference
		Advantages	*Disadvantages*	
1	Genetic Algorithm	Automates scheduling, adapts easily, integrates well	Requires parameter tuning	[1]
2	Greedy Algorithm	Simple, intuitive, good for large datasets	Often suboptimal, can cause conflicts	[9]
3	Tabu Search Algorithm	Structured approach, avoids previous solutions	Limited scalability, not always optimal	[10]
4	Simulated Annealing Algorithm	Easy to understand, versatile	May not be optimal, depends on parameters	[11]
5	Ant Colony Optimization	Can find optimal solutions, considers constraints	Computationally expensive, slow convergence	[12]
6	Particle Swarm Optimisation	Effective for combinatorial problems	Slow convergence, computationally expensive	[13]

B. *Genetic Algorithm*

Genetic Algorithms (GA) are inspired by biological evolution, as proposed by Charles Darwin, and are used to find optimal solutions in problems with multiple variables. GA operates as a heuristic search algorithm based on evolutionary [14]. In biological evolution, chromosome diversity among individuals affects reproductive success and survival rates. Four key conditions influence the evaluation process in GA[14]. First, reproduction ability is essential for survival, relating to the generation of potential solutions in GA. Second, population presence a diverse or large population enhances genetic variation, which aids in exploring the solution space. Third, genetic diversity within the population is crucial for evolving solutions, as it allows the exploration of different options. Finally, survival differences refer to traits better suited to the environment, which improve survival rates, similar to increasing the fitness values of solutions in GA.

The genetic algorithm helps find optimal solutions by considering factors such as space availability, capacity, and equipment compatibility. GA operates by mimicking the natural selection process, where the best individuals are selected, combined, and modified to produce better solutions[14].

GA involves three basic operators: selection, crossover, and mutation. These operators are used to achieve the best results in the solution search. GA identifies an initial population randomly, evaluates the quality of solutions within the population, and then applies genetic operators to produce the next generation. Figure 1 shows the cycles of the genetic algorithm[15].

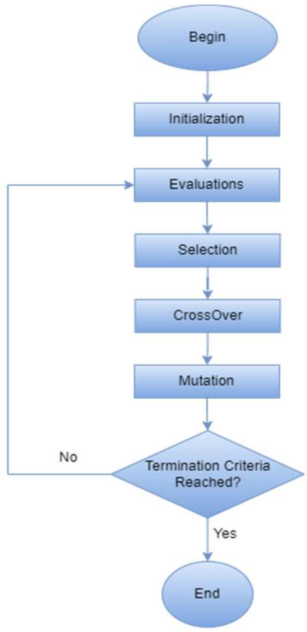

Fig. 1. Genetic Algorithm Cycles

The figure above illustrates the genetic algorithm cycles. In the first step, a number of individuals are selected from the initial population. Next, the selected chromosomes are tested through the application of the fitness function to measure how well the chromosomes meet the desired fitness criteria. In GA, the fitness value indicates how well a solution meets the desired criteria. A fitness value of 0 typically means the solution does not meet the criteria at all, while a fitness value of 1 means the solution perfectly meets the criteria. Chromosomes with fitness values close to the threshold are selected as parents for the next generation. After selecting parents, certain criteria are checked. If the criteria are met, the process stops because a satisfactory solution has been found. If not, the process proceeds to the next step. Crossover operations are performed on two chromosomes, producing offspring. This process continues iteratively until a satisfactory solution that meets the set criteria is found.

Genetic algorithms have been used to optimize scheduling processes in various studies. At Rajarata University of Sri Lanka, they improved university scheduling by efficiently allocating space and time while addressing facility limitations and availability issues[6]. Other research has developed genetic algorithm-based scheduling systems (tsuGA) to manage university constraints[1] optimized course scheduling to reduce conflicts and enhance space utilization[16], and applied these algorithms to shift scheduling for better employee rotation and allocation[17], as well as intelligent scheduling at universities for effective resource management[18]Research shows that genetic algorithms can effectively address the challenges of university timetable scheduling by optimizing constraints and improving the scheduling process [3].

III. METHODS

This study employs a genetic algorithm approach with stages as illustrated in Figure 2, along with process descriptions and examples of its application as follows.

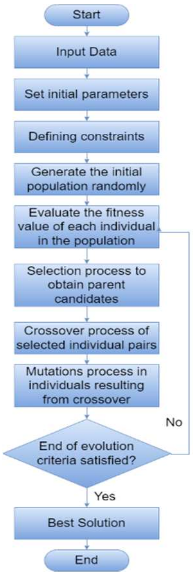

Fig. 2. Flow Diagram of the Genetic Algorithm

A. Input Requirements for the Genetic Algorithm

Data required as initial input before generating the genetic algorithm includes:

1) Users Data: User data contains information about the practical sessions available in the Faculty of Industrial Engineering. This data includes user names, roles, and locations. Table 2 below provides an example of user data.

TABLE II. EXAMPLE USER DATA

User_id	User Name	Role	Location
1	Sisjar Practicum	Practicum	Room
2	RPL Practicum	Practicum	Room

2) Rooms Data: Room data includes information about available rooms, such as capacity, type of PC, and location. Table 3 below provides an example of room data.

TABLE III. EXAMPLE ROOM DATA

Room_id	Room Name	Capacity	PC Type	Location
1	R1	35	High End	Room
2	R2	30	High End	Room
3	R3	15	High End	Room
4	R4	40	No PC	Room
5	R5	25	No PC	Room
6	R6	50	No PC	Workshop

3) Practical Requirements Data: Practical requirements data includes details about the practical sessions that need to be scheduled, such as status (offline/online), number of users, room priority, and required PC type. Table 4 below provides an example of practical requirements data.

TABLE IV. EXAMPLE PRACTICAL REQUIREMENTS DATA

Req_id	Module	Status	Select PC	User Qty	Room Priority
1	Module 2	Offline	High End	15	null
2	Module 1	Offline	High End	30	2
3	Module 1	Online	No PC	30	null

4) Shift Data: Shift data includes information about start time, end time, and descriptions. Table 4 below provides an example of shift data.

TABLE V. EXAMPLE SHIFT DATA

Id	Start_Time	End_Time	Description
1	06:30:00	08:30:00	Shift 1
2	08:30:00	10:30:00	Shift 2

5) Practical Schedule Data: Practical schedule data includes information about shifts, dates, assistant codes, and related practical requirements. Table 6 below provides an example of practical schedule data.

TABLE VI. EXAMPLE PRACTICAL SCHEDULE DATA

Schedule_id	User_id	Requirement_id	Shift_id	Date
1	1	1	1	2024-05-15
2	2	2	2	2024-05-15
3	2	3	1	2024-05-16

B. Defining Constraints

In timetable scheduling, constraints are classified into hard constraints and soft constraints. In applying genetic algorithms, various constraints must be considered in the room scheduling process to ensure that the solutions meet the requirements and limitations. Table 7 below outlines the constraints in the room scheduling process.

TABLE VII. HARD AND SOFT CONSTRAINTS FOR PRACTICUM ROOM SCHEDULING

Hard Constraints	Description
Practicum needs must exist	Ensures that there is a need for the practicum; if not, the schedule is invalid.
Rooms must be available	Ensures that a room has been assigned; if not, the schedule is invalid.
User and room locations must match	User location must match the room location; if not, the schedule is invalid.
No schedule overlaps in the same room, date, and shift	No two schedules should use the same room at the same time.
Room capacity must be sufficient	Ensures that the selected room has enough capacity to accommodate the number of users needed for the scheduled practicum.
Practicum status	If the practicum is 'online', a physical room is not required.
Room must have the required type of PC	Room must be equipped with the type of PC needed for the practicum. If no room with the required PC is available, 'No PC' or 'High End' rooms may be considered if available.
Priority rooms must be assigned first if available	Prioritize assigning rooms that have been designated as priorities for the session.
Soft Constraints	Description
Select alternative rooms if priority rooms are unavailable or conflict arises	Try to find alternative rooms if the priority room is unavailable.
Prioritize schedules with High-End PC needs	Ensure that schedules requiring High-End PCs are allocated to rooms with such PCs first.

C. Determining Initial Parameters

Parameters in the genetic algorithm are control parameters for the calculations to be performed. The genetic algorithm parameters are as follows:

1) Population Size: Determines the number of individuals in each generation.

2) Max Generations: Determines how many iterations the genetic algorithm will run.

3) Mutation Rate: The probability of mutation occurring in offspring.

D. Initial Population Initialization

The initial population consists of a number randomly initialized individuals. Each individual represents a scheduling solution. The initial population is created by selecting random combinations of practicum requirements and available rooms. It is important to ensure that the population has sufficient diversity to avoid premature convergence. Below is the initialization process for “Schedule_id 1”.

TABLE VIII. EXAMPLE INITIAL POPULATION FOR SCHEDULE_ID 1

Individual	User_id	Room_id	Requirement_id	Shift_id
1	1	1	1	1
2	1	2	1	1
3	1	3	1	1
4	1	4	1	1

E. Fitness Evaluation

Evaluate each individual based on criteria such as room availability, capacity, and conflicts. For example, an individual with "Room_id 1" has the highest fitness value because the room is available, capacity is sufficient, and there are no conflicts.

TABLE IX. EXAMPLE FITNESS EVALUATION

Ind.	Room	Capacity	Room Priority	PC	Conflict	Fit
1	R1 available	R1 (35) >= User QTY (15)	No	R1 (High End) fits	No conflict with other schedules	1
2	R2 available	R2 (30) >= User QTY (15)	No	R2 (High End) fits	No conflict with other schedules	1
3	R3 available	R3 (15) = User QTY (15)	No	R3 (High End) fits	No conflict with other schedules	1
4	R4 available	R4 (40) >= User QTY (15)	No	R4 (No PC) does not fit	No conflict with other schedules	0

F. Selection

In the selection stage, individuals with the highest fitness values are chosen as parents for the next stage. Individuals with a fitness of 1 are selected (Individuals 1, 2, and 3).

TABLE X. EXAMPLE SELECTION STAGE

Individu	User_id	Room_id	Requirement_id	Shift_id
1	1	1	1	1
2	1	2	1	1
3	1	3	1	1

G. Crossover

Perform crossover between selected pairs to produce more diverse offspring. For example, a crossover between parents A and B produces offspring D and E.

TABLE XI. EXAMPLE CROSSOVER STAGE

Parent	User_id	Room_id	Requirement_id	Shift_id
A	1	1	1	1
B	1	2	1	1
Offspring	**User_id**	**Room_id**	**Requirement_id**	**Shift_id**
D	1	1	1	1
E	1	2	1	1

H. Mutation

Each offspring has a 10% probability of mutation to maintain genetic diversity. For example, offspring G experiences a mutation in "Room_id".

TABLE XII. EXAMPLE MUTATION

Offspring	User_id	Room_id	Requirement_id	Shift_id
G	1	4	1	1

I. Iteration and Replacement

The iteration process involves repeating steps 2-6 until the maximum number of generations is reached. The new population, after selection, crossover, and mutation, replaces the old population.

TABLE XIII. NEW POPULATION AFTER ITERATION

Individual	User_id	Room_id	Requirement_id	Shift_id	Fitness
1	1	1	1	1	1
2	1	2	1	1	1
3	1	1	1	1	1
4	1	2	1	1	1

J. Final Results

After several generations, the best solution found from the final population is taken as the final solution.

TABLE XIV. FINAL RESULTS EXAMPLE OF GENETIC ALGORITHM APPLICATION

Schedule_id	User_id	Room_id	Require_id	Shift_id
1	1	1	1	1
2	2	2	2	2
3	2	null	3	1

IV. RESULT AND DISCUSSION

For scheduling practicum rooms at the Faculty of Industrial Engineering, Telkom University, a genetic algorithm was developed as an optimization solution. Key features of the practicum room mapping optimization module include generating room mappings, adjusting schedules before finalization, saving results to the database, adjusting schedules after finalization, and viewing the practicum room schedule. The implementation results, shown in Figures 3 to 7, demonstrate these features in action, from generating room mappings to visualizing the practicum schedule.

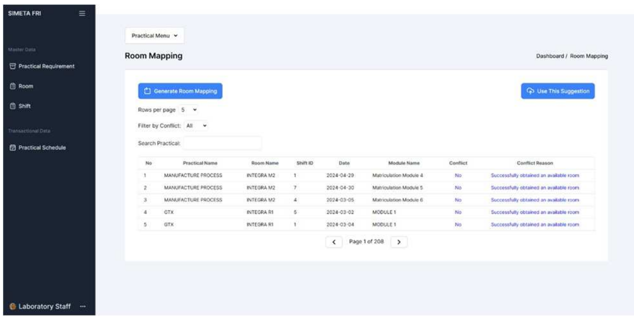

Fig. 3. Generate Room Mapping

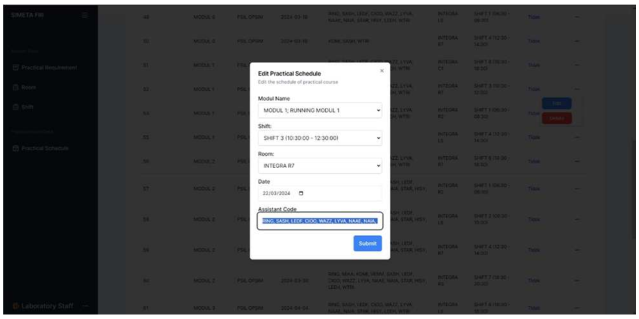

Fig. 4. Adjust Mapping Schedule (Before Final Generate)

Fig. 5. Save Mapping Results to Database

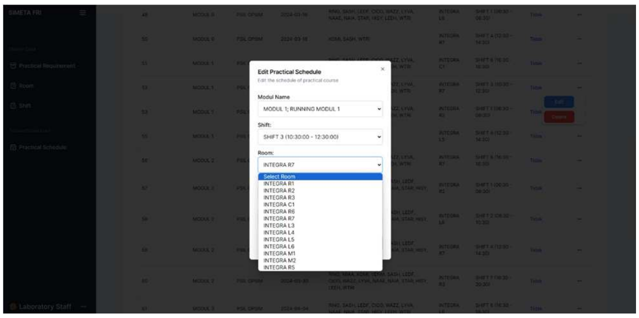

Fig. 6. Adjust Mapping Schedule (After Final Generate)

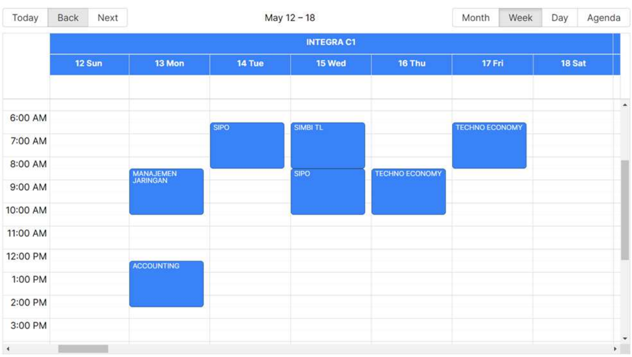

Fig. 7. View Practicum Room Schedule

A. Testing

The system testing phase aims to ensure that the developed system functions well and meets the established requirements. Testing is conducted in the main stage, namely scenario testing.

1) Scenario Testing: Scenario testing evaluates the system's performance under realistic conditions. The results show that the system handles various conditions and constraints effectively and detects conflicts when they occur.

TABLE XV. SCENARIO TESTING OF GENETIC ALGORITHM

No	Scenario	Description	Actual Result
1	No Conflict	No conflict with optimal conditions	fitness: 1, conflict: No, conflictReason: Room allocated
2	Room Capacity Conflict	Conflict due to insufficient room capacity	fitness: 0, conflict: Yes, conflictReason: Room unavailable
3	User Location Conflict	Conflict due to incompatible user location	fitness: 0, conflict: Yes, conflictReason: Room unavailable
4	Priority Room Conflict	Conflict due to unavailable priority room	fitness: 0, conflict: Yes, conflictReason: No rooms available
5	No Conflict for Online Practicum	No conflict for online practicum	fitness: 1, conflict: No, conflictReason: No conflict
6	Room Availability	Room available for practicum needs	fitness: 1, conflict: No, conflictReason: Room allocated
7	Room Priority	Selected room matches priority	fitness: 1, conflict: No, conflictReason: Room allocated
8	Room Capacity	Room capacity sufficient	fitness: 1, conflict: No, conflictReason: Room allocated
9	PC Availability	Room has required PC category	fitness: 1, conflict: No, conflictReason: Room allocated
10	User Location Compatibility	User location compatible with room location	fitness: 1, conflict: No, conflictReason: Room allocated
11	Schedule Overlap Conflict	Conflict due to overlapping schedules	fitness: 0, conflict: Yes, conflictReason: No rooms available

B. Evaluation

The evaluation stage ensures that the developed and optimized system works as expected. This phase evaluates the effectiveness of the genetic algorithm used for practicum room mapping, with an in-depth analysis of optimization results. Room occupancy before and after applying the genetic algorithm is compared. The table below shows the room occupancy results for the even semester practicum data of 2024 at the Faculty of Industrial Engineering.

TABLE XVI. ROOM OCCUPANCY RESULTS

Room	Timeslots Used		Occupancy	
	Before GA	***After GA***	***Before GA***	***After GA***
R1	161	143	34.85	30.95
R2	87	77	18.77	16.67
R3	71	74	15.43	16.02
C1	72	92	15.53	19.91
R5	74	175	16.02	37.88
R6	75	72	16.32	15.58
R7	14	84	2.95	18.18
L3	14	7	2.95	1.52
L4	137	6	29.69	1.30
L5	137	82	29.69	17.75
L6	44	74	9.53	16.02
M1	131	60	28.35	12.99

Room	Timeslots Used		Occupancy	
	Before GA	***After GA***	***Before GA***	***After GA***
M2	3	74	0.65	16.02
Total Capacity Utilization	1020	1020	220.73	220.78
Rata-rata	78.45	78.46	16.98	16.98
Standar Deviasi	49.57	43.42	10.73	9.40
Range	158.00	169.00	34.20	36.58

Based on the analysis results in the table, the application of the genetic algorithm has proven successful in improving the efficiency and effectiveness of practicum room usage at the Faculty of Industrial Engineering. The distribution of room usage becomes more even and optimal with significantly reduced usage variation, indicating that the genetic algorithm can save time and effort compared to using Office Spreadsheet. The standard deviation of room occupancy decreased from 10.73 to 9.40, indicating more consistent and balanced room usage. Additionally, the range of room occupancy decreased from 34.20 to 36.58, reflecting more stable usage variation. By using the genetic algorithm, the scheduling process for practicum rooms, which previously took days, now only takes approximately 5 minutes for the given data case.

V. Conclusion

The application of the genetic algorithm in scheduling practicum room mapping at the Laboratory of the Faculty of Industrial Engineering, Telkom University, has shown significant improvements in the efficiency and effectiveness of room usage. Scenario testing revealed that the system is capable of managing various conditions and constraints, successfully detecting conflicts when they arise. The genetic algorithm optimizes room usage distribution, resulting in a more balanced and optimal allocation, with a notable reduction in usage variation. This demonstrates that the genetic algorithm can effectively address complex scheduling challenges, offering substantial time, effort, and resource savings compared to traditional methods such as using Office Spreadsheets. These findings underscore the importance of advanced optimization technologies and methods in managing academic scheduling and enhancing the operational performance of educational institutions. Future research should delve deeper into the genetic algorithm, exploring various performance-affecting parameters and considering the integration of machine learning or other optimization algorithms to compare results and identify the most effective solutions.

References

[1] H. Kutty Mammi and L. Ying Ying, "Timetable Scheduling System using Genetic Algorithm for School of Computing (tsuGA)," *International Journal of Innovative Computing*, vol. 11, no. 2, pp. 67–72, Oct. 2021, doi: 10.11113/ijic.v11n2.342.

[2] M. Lindahl, A. J. Mason, T. Stidsen, and M. Sørensen, "A strategic view of University timetabling," *Eur J Oper Res*, vol. 266, no. 1, pp. 35–45, Apr. 2018, doi: 10.1016/j.ejor.2017.09.022.

[3] T. Ardi Nugraha, K. Trinanda Putra, and N. Hayati, "University Course Timetabling with Genetic Algorithm: A Case Study," *Journal of Electrical Technology UMY*, vol. 1, no. 2, pp. 100–105, 2017, doi: 10.18196/jet.1213.

[4] A. Muklason, R. G. Irianti, and A. Marom, "Automated Course Timetabling Optimization Using Tabu-Variable Neighborhood Search Based Hyper-Heuristic Algorithm," *Procedia Comput Sci*, vol. 161, pp. 656–664, 2019, doi: 10.1016/j.procs.2019.11.169.

[5] A. Lemos, F. S. Melo, P. T. Monteiro, and I. Lynce, "Room usage optimization in timetabling: A case study at Universidade de Lisboa," *Operations Research Perspectives*, vol. 6, p. 100092, 2019, doi: 10.1016/j.orp.2018.100092.

[6] PremasirilDM, "University Timetable Scheduling Using Genetic Algorithm Approach Case Study: Rajarata University OF Sri Lanka," *Journal of Engineering Research and Application www.ijera.com*, vol. 8, no. 12, pp. 30–35, 2018, doi: 10.9790/9622-0812023035.

[7] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. 1989.

[8] P. G. Daniel, Dr. A. O. Maruf, and Dr. B. Modi, "Paperless Master Timetable Scheduling System," *Int J Appl Sci Technol*, vol. 8, no. 2, 2018, doi: 10.30845/ijast.v8n2a7.

[9] J. B. C. Legaspi, R. M. De Angel, A. Lagman, and J. H. J. Ortega, "Web Based Course Scheduling System Using Greedy Algorithm," *International journal of simulation: systems, science & technology*, pp. 1–7, 2019, doi: 10.5013/ijssst.a.20.s2.14.

[10] Zaeniah and Salman, "Designing Class Schedule Information System By Using Taboo-Search Method," *Jurnal PILAR Nusa Mandiri*, vol. 16, pp. 241–248, 2020, [Online]. Available: https://utmmataram.ac.id

[11] W. Sari and J. E. Suseno, "Metode Simulated Annealing untuk Optimasi Penjadwalan Perkuliahan Perguruan Tinggi," *Jurnal Sistem Informasi Bisnis*, vol. 6, no. 2, p. 133, 2016, doi: 10.21456/vol6iss2pp133-143.

[12] T. Herianto, "Implementation of the Ant Colony System Algorithm in the Lecture Scheduling Process," *Instal : Jurnal Komputer*, vol. 12, no. 02, pp. 54–60, 2021, doi: 10.54209/jurnalkomputer.v12i02.23.

[13] S. Imran Hossain, M. A. H. Akhand, M. I. R. Shuvo, N. Siddique, and H. Adeli, "Optimization of University Course Scheduling Problem using Particle Swarm Optimization with Selective Search," *Expert Syst Appl*, vol. 127, pp. 9–24, 2019, doi: 10.1016/j.eswa.2019.02.026.

[14] T. Suratno, N. Rarasati, and G. Z`, "Optimization of Genetic Algorithm for Implementation Designing and Modeling in Academic Scheduling," *EKSAKTA: Berkala Ilmiah Bidang MIPA*, vol. 20, no. 1, pp. 17–24, 2019, doi: 10.24036/eksakta/vol20-iss1/166.

[15] A. Lambora, K. Gupta, and K. Chopra, "Genetic Algorithm- A Literature Review," *Proceedings of the International Conference on Machine Learning, Big Data, Cloud and Parallel Computing: Trends, Prespectives and Prospects, COMITCon 2019*, no. 1998, pp. 380–384, 2019, doi: 10.1109/COMITCon.2019.8862255.

[16] D. Nasien and A. Andi, "Optimization of Genetic Algorithm in Courses Scheduling," *IT Journal Research and Development*, vol. 6, no. 2, pp. 151–161, 2022, doi: 10.25299/itjrd.2022.7896.

[17] W. Sardjono, W. Priatna, D. S. Nugroho, A. Rahmasari, and E. Lusia, "Genetic algorithm implementation for application of shifting work scheduling system," *ICIC Express Letters*, vol. 15, no. 7, pp. 791–802, 2021, doi: 10.24507/icicel.15.07.791.

[18] O. Alhuniti, R. Ghnemat, and M. S. A. El-Seoud, "Smart University Scheduling Using Genetic Algorithms," *ACM International Conference Proceeding Series*, pp. 235–239, 2020, doi: 10.1145/3436829.3436873.