iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.80451

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

Exam Timetable and Room Allocation Using Genetic Algorithm

Mrs. P. Sarala[1], Shaik Reena[2], Sesham Ravindra Babu[3], Mutyala Rakesh[4], Yarlagadda Sai Venkat[5], Remella Sai Sudha[6]
[1]*Assistant Professor,* [2,3,4,5,6]*Student, Computer Science and Engineering, Dhanekula Institute of Engineering and Technology Vijayawada, India*

Abstract: ***Scheduling examinations at an institution is a very difficult task. Many things must be taken care of at that time. This is a study about Exam Timetable and Room Allocation System. This is a system based on Genetic Algorithm, this is how nature chooses the best ones. It thinks Exam Timetables as a group of codes. Then selection, crossover and mutation are applied to evolve Exam Timetables. There is a way of measuring how good the Exam Timetable is. It penalizes the Exam Timetable in case there is exam conflict or resources not enough capacity, or the exams not evenly spaced out. It rewards the Exam Timetable if it uses resources well. We measure Exam Timetable and Room Allocation System. We find that it greatly decreases the errors in scheduling. It also uses work well and resources well than current way of doing manually. The Exam Timetable and Room Allocation System can be easily adjusted to changes at an institution. It provides a foundation for adding things like assigning people to watch exams automatically and making sure the workload is fair, for the Exam Timetable and Room Allocation System. The Exam Timetable and Room Allocation System works well. Examination scheduling is something that academic institutions have to do. The Genetic Algorithm helps make the scheduling process better. It makes sure that the Exam Timetable and Room Allocation System schedules exams properly. The Exam Timetable and Room Allocation System is good, at what it does. The system is efficient. It helps institutions to manage their resources. The automated system is an improvement, over procedures.***

Index Terms: Exam timetabling, room allocation, genetic algorithm, evolutionary optimization, scheduling automation, fitness function, constraint satisfaction, educational management systems.

I. INTRODUCTION

Scheduling exams in a way that's accurate and does not have any conflicts is very important for any school or university. It is not about logistics a good timetable makes sure that all students are treated fairly that rooms are used well and that the administration can run smoothly. When there are students and more courses making schedules by hand takes a lot of time and effort but it can still have mistakes like having the same student take two exams at the same time or having too many students in one room or not having enough time between exams.

Researchers have found that using computers to make schedules is a way to solve this problem. One way that works well is called the Genetic Algorithm. This method is good because it can look at schedules and find the best one. The Genetic Algorithm works by trying out schedules. It uses rules like how living things evolve to find the schedule. The Genetic Algorithm is good at finding schedules that meet all the rules. It also finds schedules that're good in many ways. This paper is about a system that uses the Genetic Algorithm.

The system makes exam schedules and assigns rooms. The system takes in information, about the school. This includes which students are taking which courses. It also includes which rooms are available and what times exams can be held. Then the system makes a schedule that meets all the rules. The schedule is also fair. The rest of this paper is organized like this:

II. LITERATURE SURVEY

Various researchers have tried to generate exam schedules using various techniques. Burke and his colleagues studied various algorithms including simulated annealing and tabu search and used the ITC 2007 and the Toronto collection to test the techniques. This study was a preliminary one. It had the drawback that changes were difficult to make. It also did not try combining different techniques.

Kaur and Goyal developed a system for generating course schedules. They modeled time slots and room assignment as genes. Their fitness function penalized conflicts. The system performed well on testing. However it did not perform well on large tests. It did not take into account room capacities and the presence of observers.

©IJRASET: All Rights are Reserved

Garg and Singh used scheduling algorithms to generate exam schedules and minimize conflicts. Their model did not take into account fairness. It also did not balance the rules.

Ahmed and Basel used scheduling algorithms to generate exam schedules. Their system achieved fewer conflicts.. However it was slow on large tests. Also there was no notification system or automation.

Ali and Khan developed a system that considered rules such as room size and conflict. It was not as flexible as a system using algorithms. Hence it could not be easily adapted.

All these studies had some problems, like not working with big tests not being fair not being able to notify people and not being able to change and improve. These problems are why we made our system the way we did to deal with exam scheduling in a way. The exam scheduling system is what we focused on and the exam scheduling system is what we tried to improve.

Paper (Author, Year)	Methodology	Gaps
Burke et al., 2020	Survey: GA, SA, Tabu Search on benchmark datasets	No real-time adaptability; slow on large instances
D. Kaur, 2021	GA for course timetabling; fitness checks clashes	Scalability issues; no room capacity or invigilator
R. Singh, 2022	Meta-heuristic GA; reduced clashes	No fairness metric; lacks multi-objective handling
Ahmed & Basel, 2023	GA with selection–crossover–mutation on benchmarks	High compute time; no notification module
F. Ali, 2022	Constraint satisfaction; capacity & clash checking	Cannot evolve; no dynamic update support

Table I. Summary of Related Work

III. PROBLEM STATEMENT

Making a schedule for exams is really hard. We have to think about subjects, rooms, students and time slots. The goal is to give each subject a time and a room so that:

1) No student has two exams at the time.
2) We don't put many students in a room for one subject.
3) We spread out exams so students don't get too tired.
4) We use rooms much as possible for all exams.

The problem is that there are ways to make a schedule and all the rules are connected. If we try to fix one thing we might make another thing worse. Usually people try to make a schedule by filling in times one by one and changing it if it doesn't work. This way is not good, for problems and we might not get the best schedule.

IV. EXISTING SYSTEM AND LIMITATIONS

Most institutions still use semi-automated ways to do things. The people in charge usually use spreadsheet tools like Microsoft Excel to make timetables by hand or by using drag-and-drop interfaces. This way of doing things has some problems.

1) Students who are taking courses often have exams at the same time and this is usually only found out after the timetable is published.
2) The school does not make sure that there is room in the classrooms so they get too crowded.
3) If something changes, like a new subject is added or a room is not available the whole schedule has to be redone.
4) Making the schedule takes a lot of time it takes administrators weeks to do it before each exam period.
5) It is also hard to make sure that students have time between exams, which is not fair to them and it is impossible to guarantee this without checking everything manually.

These problems together make the schedules not as good as they could be. They take away, from the important things that the institution should be focusing on like the students education.

V. PROPOSED SYSTEM

The new system gets rid of scheduling and uses a computer program to do the job. This computer program is based on the Genetic Algorithm. The Genetic Algorithm looks at each schedule as a kind of map. This map shows what subject is taught at what time and in what room. The Genetic Algorithm system does things in steps.

©IJRASET: All Rights are Reserved

A. *Initialization*

We start with one hundred people lets call them chromosomes and we pick them randomly. This way every person gets a time and a room. At the beginning it is okay if some rules are not followed. We want all these people to be different from each other at this point. This is, on purpose because it gives us a lot of options to work with and then we can slowly narrow it down to what we're looking for. The chromosomes are the key and the chromosomes are what we are focusing on so we have one hundred chromosomes to begin with.

B. *Fitness Evaluation*

Each chromosome is looked at by a fitness function that has parts. The first weight times the number of times the chromosome breaks the rules about clash plus, the second weight times the number of times it breaks the rules about capacity plus, the third weight times the number of times it breaks the rules about being fair plus, the fourth weight times the number of times it breaks the rules about distribution. The weights can be adjusted by the people in charge. The chromosome with a F value is a better solution.

C. *Selection*

Tournament selection is used with a group of 5 chromosomes. These chromosomes are picked randomly. The one with the fitness score gets to move on to the next step. This helps keep the selection pressure and also keeps the group of chromosomes diverse. The tournament size is 5, in this case.

D. *Crossover*

Uniform crossover happens with a chance of 85%. For each subject the offspring gets the allele from one parent with a 50% chance. This helps mix parts of solutions. The offspring can get parts from each parent that do not cause conflicts. Uniform crossover helps combine solutions. It allows getting sub-assignments, from each parent.

E. *Mutation*

With a chance of 2% a gene that is picked randomly gets moved to a time and room that is allowed. This process helps avoid getting stuck in a solution that's not the best, by making small changes. It does this by changing a gene to a time slot and room pair. The change happens with a probability of 0.02. Mutation is used to make sure the solution does not get stuck in an optimum.

F. *Termination*

The algorithm stops after it has done 500 generations or when the best score gets really bad. This bad score is the point where the schedule's totally okay, with all the rules. The schedule that does the job across all 500 generations is the one that is chosen as the final schedule. The final schedule is the one that is picked because it has the score and that is the timetable that we want.

VI. SYSTEM ARCHITECTURE AND DESIGN

A. *System Architecture*

The Data Input part is where we get information about students, their enrollments, rooms with capacities and allowed time slots for students. We store this information in a MySQL database or in CSV files for setups. The Processing Engine checks the Data Input to make sure the Data Input is correct. It also builds a representation of the Data Input.

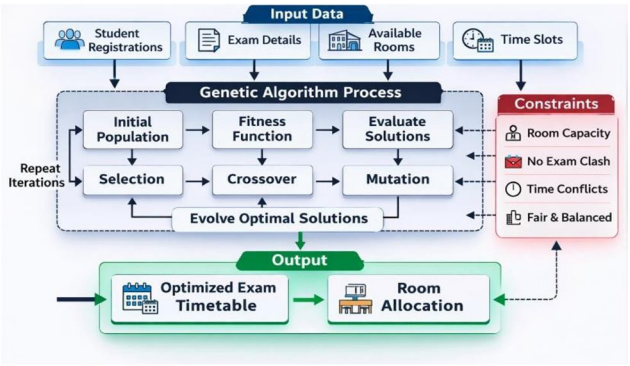


Fig. 1. System Architecture: Layered processing pipeline from data input through GA optimization to timetable output.

The Output Interface shows the timetable for students on a web dashboard. This web dashboard is accessible to administrators, students and invigilators so they can see the timetable, for students.

A. UML Use Case Diagram

There are three people who use the system: the Administrator, Students and Invigilators. The Administrator is the one who puts information into the system makes the timetable, checks and changes the schedules of the candidates and then shares the result. The Students and Invigilators can only look at the timetable after it has been shared. They cannot make any changes to it. The Administrator is the one who starts the process that uses the GA engine and the room allocation module. These are parts of the system that work automatically when the Administrator asks to generate something.

B. UML Class Diagram

Six core classes make up the domain model. Admin. This class helps with system operations. Student. This class keeps track of enrollment records. Students can also view timetables. Subject. This class has information about courses. Room. This class has details about room capacity and availability. Timetable. This class has the schedule. GeneticAlgorithm. This class does selection, crossover, mutation and fitness evaluation. The Timetable class connects to Subject and Room in a one-, to-many way. Student uses Timetable to see published schedules.

C. System Flowchart

Upon starting the Administrator puts in the exam details. The system then checks the inputs to make sure they are correct creates a group of solutions and checks how good they are. If the best solution is better, than the limit set the system uses a method called GA to try and find a better solution. It does this by picking the solutions combining them and making small changes to create a new group of solutions then checks how good they are again. This keeps happening until the system decides it has an enough solution. When it does it assigns rooms. Makes a final timetable that is shared with everyone.

VII. RESULTS AND DISCUSSION

We did an experiment to test something on a group of students. This group was like an engineering department with 12 teachers 5 rooms to take tests and 3 times a day to take these tests over 6 days. There were 240 students, in this group. We used a computer program called the Genetic Algorithm for this test. The computer program ran for 500 rounds using the rules we talked about in Section V. We wanted to see how well the Genetic Algorithm worked for this test so we used it with the 12 teachers, 5 rooms and 240 students.

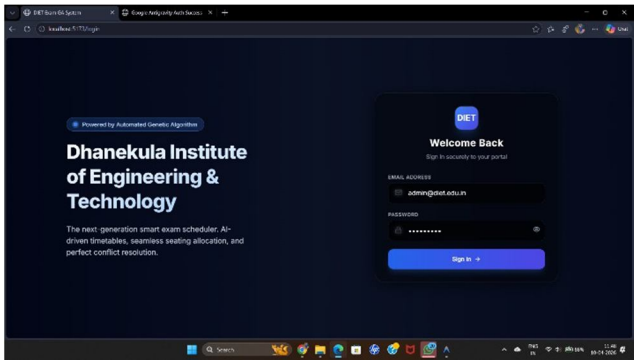


Fig 2: Landing Page

A. Convergence Performance

Figure 2 shows us the fitness score for 500 generations. At the start the average fitness score was around 87 penalty units, which's really bad because it means a lot of rules were broken.. By the 150th generation the fitness score went down to below 30 which is good because it means most of the rules were being followed. The fitness score of the Genetic Algorithm kept getting better and better until it was zero by the generation. This means the Genetic Algorithm is really good, at finding the solution. The Genetic Algorithm can look at all the solutions. Pick the Genetic Algorithm solution that works best. This is what Figure 2 is telling us about the fitness score of the Genetic Algorithm over time.

B. Conflict Reduction

Table II shows how the manual baseline and the GA-based system compare when it comes to conflict metrics. The GA-based system got rid of all the constraint problems, such as students having clashes and rooms being over capacity. It also cut down on constraint problems like unequal gaps between exams by 91 percent compared to the manual scheduling method. The GA-based system really made a difference, in reducing these issues.

Metric	Manual Method	GA System
Student Exam Clashes	14	0
Room Capacity Violations	7	0
Unequal Gap Violations	22	2
Avg. Schedule Build Time	48 hrs	2.3 sec
Room Utilization Rate	61%	88%

Table II. Performance Comparison: Manual vs. GA-Based Scheduling

C. Room Utilization

Figure 3 shows a bar chart of how roomsre used in five different places for two methods. When people did the scheduling by hand the rooms were not used evenly with some used little as 42 percent of the time and others used as much as 79 percent of the time. The schedule that was optimized with the Genetic Algorithm method did a job of using the rooms with usage ranging from 81 percent, to 94 percent, which means the Genetic Algorithm method did a better job of distributing the rooms.

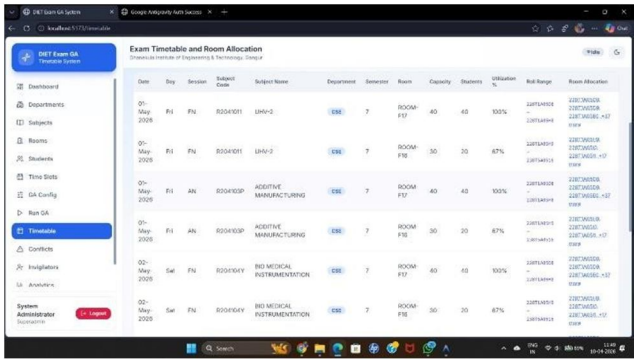


Fig 3: Room Allocation

D. Fitness Score Distribution

The population fitness distribution at points in time (1, 100 250 500) is shown in Figure 4 as box plots. We can see that the range of fitness levels gets smaller and smaller. This means that the population fitness distribution is getting closer to a group of good solutions with low penalties. By the time we reach generation 500 the middle value of the population fitness distribution is very close to zero. The population fitness distribution is clearly moving towards these solutions and the population fitness distribution is getting better, over time.

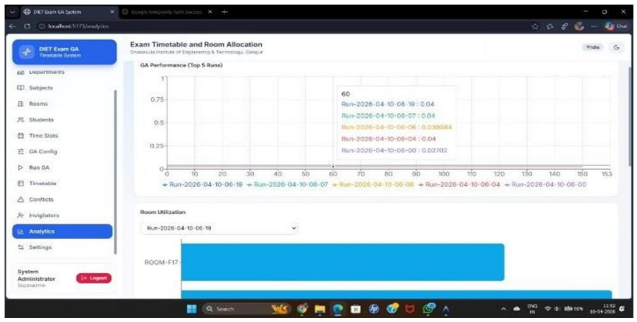


Fig 4: GA Performance

E. Test Cases and Validation

Four structured test cases were executed to validate functional correctness:

TC#	Scenario	Expected Output	Actual Output	Result
1	Upload valid exam dataset	File accepted; records parsed	File parsed correctly; 12 subjects loaded	Pass
2	Display timetable with all required fields	Subject, date, day, time, room, in-charge visible	All fields displayed; date–day alignment correct	Pass
3	Validate date-day alignment	Date and day match for each subject	e.g., 14-May-21 → Friday confirmed	Pass
4	GA optimization on full dataset	Zero conflicts; constraints satisfied	Optimized timetable produced; 0 clashes, 0 capacity violations	Pass

Table III. Test Case Summary

VIII. CONCLUSION AND FUTURE WORK

This paper is about a system that uses a Genetic Algorithm to make exam timetables and assign rooms for schools. The system obtains information from schools. Generates schedules that are equitable and satisfy all requirements of the schools. It does the job without errors. Utilizes rooms well about 88% of the time which is significantly better than happens when done by hand.

The Genetic Algorithm was run for 500 generations to determine how effective the Genetic Algorithm was. The data demonstrates the Genetic Algorithm is quite efficient in obtaining solutions and does so quickly.

There are things that can be done to make the Genetic Algorithm system better in the future. One thing to try is to use the Genetic Algorithm with search methods. This might help the Genetic Algorithm find answers faster when there is a lot of data from schools. Add a feature that automatically assigns people to watch the exams. Use the Genetic Algorithm with methods to improve it. Improve the Genetic Algorithm by using something called Reinforcement Learning to adjust its settings while the Genetic Algorithm is running.

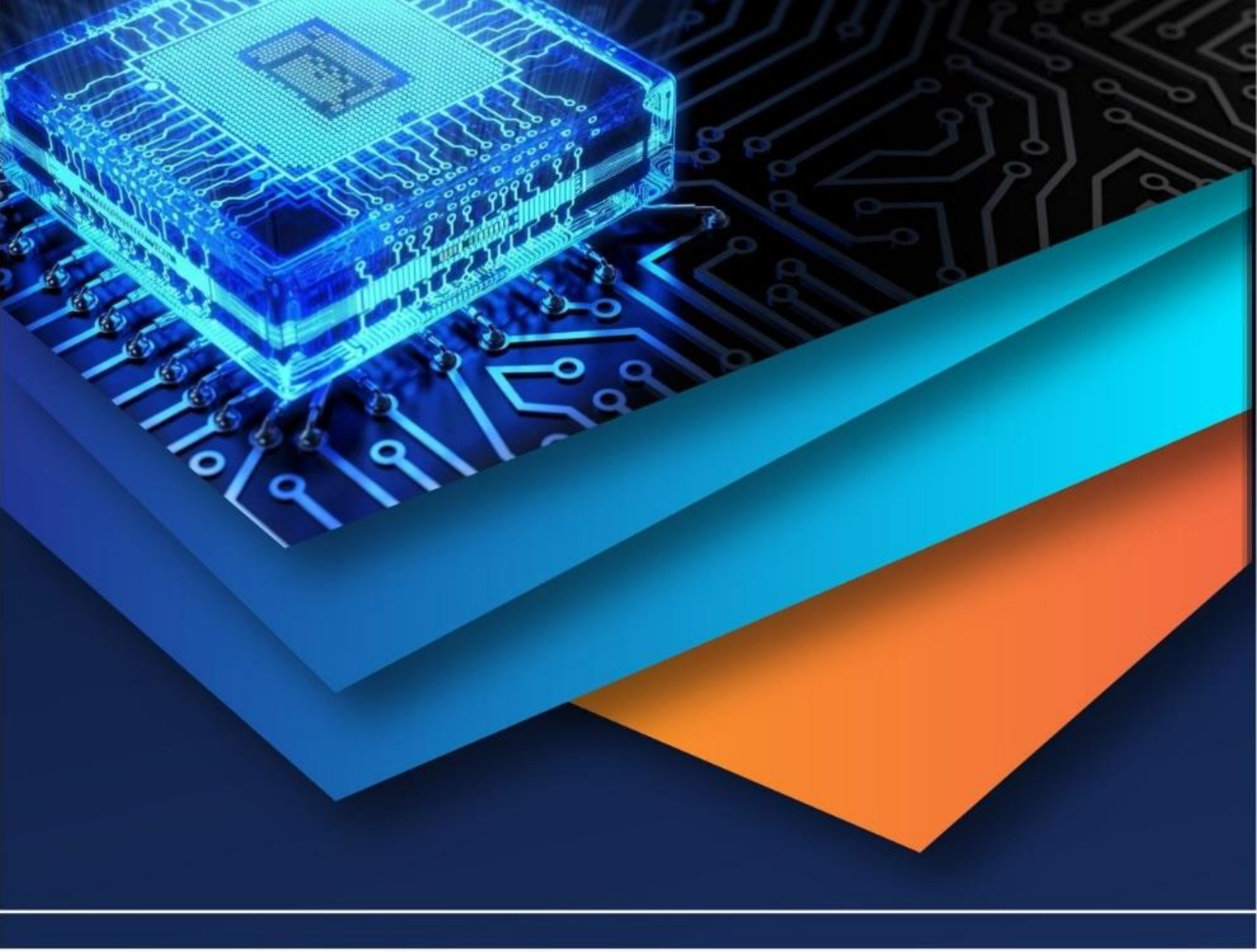
This could make the solutions the Genetic Algorithm finds better for schools. If the Genetic Algorithm system was available on the internet and could send notifications in time it would be easier for people to use the Genetic Algorithm system and would help them communicate better with schools. The Genetic Algorithm system is a solution to a problem that schools face, which is making exam timetables and assigning rooms. The results also show that the Genetic Algorithm system can handle the work of making exam timetables and assigning rooms for schools. The Genetic Algorithm system is good, at solving these problems for schools.

REFERENCES

[1] E. K. Burke, J. P. Newall, and R. F. Weare, “A memetic algorithm for university exam timetabling,” in Proc. IEEE Congr. Evol. Comput. (CEC), 1996, pp. 241–246.
[2] S. Abdullah, H. Turabieh, B. McCollum, and P. McMullan, “A hybrid metaheuristic approach to the university course timetabling problem,” in Proc. IEEE Congr. Evol. Comput. (CEC), 2012.
[3] R. Qu, E. K. Burke, B. McCollum, L. T. G. Merlot, and S. Y. Lee, “A survey of search methodologies and automated system development for examination timetabling,” IEEE Trans. Evol. Comput., vol. 13, no. 2, pp. 260–283, Apr. 2009.
[4] M. W. Carter and G. Laporte, “Recent developments in practical examination timetabling,” in Proc. Practice and Theory of Automated Timetabling (PATAT), 1996.

[5] A. R. Mahlous and H. Mahlous, "Student timetabling genetic algorithm accounting for student preferences," PeerJ Comput. Sci., vol. 9, 2023.
[6] A. Aslan, "Hybrid genetic algorithms for examination timetabling problem," arXiv preprint arXiv:2306.00534, 2023.
[7] F. Maspiyanti, et al., "Course timetabling using genetic algorithm and fuzzy logic," J. Inf. Syst. Eng., 2025.
[8] A. Muklason, et al., "Generic university examination timetabling system using hill climbing optimization," Procedia Comput. Sci., vol. 231, pp. 1–10, 2024.
[9] R. W. Bello and J. A. Godwill, "Hybrid technologies and genetic algorithms applied to school timetable generation," World Sci. News, vol. 189, pp. 288–310, 2024.
[10] L. M. Cornei and M. E. Breabăn, "Enhancing genetic algorithms with graph neural networks: A timetabling case study," arXiv preprint arXiv:2602.08619, 2026.
[11] E. K. Burke, J. D. Landa Silva, and E. Soubeiga, "A survey of examination timetabling problems," European Journal of Operational Research, vol. 153, no. 1, pp. 3–34, 2020.
[12] D. Kaur and A. Goyal, "University course timetabling using genetic algorithm," in Proc. IEEE Int. Conf. Comput. Commun. Autom. (ICCCA), 2021, pp. 1–5.
[13] R. Singh and S. Garg, "Optimization of exam timetable using meta-heuristic approach," in IEEE Int. Conf. Intelligent Systems and Networks (ICISN), 2022, pp. 234–239.
[14] M. Ahmed and H. Basel, "Genetic algorithm based university examination scheduling system," IEEE Access, vol. 11, pp. 55120–55130, 2023.
[15] F. Ali and M. Khan, "Constraint-based automated exam timetabling system," in IEEE Int. Conf. Adv. Comput. Sci. Eng. (ICACSE), 2022, pp. 78–84.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)