AI 201 Mini-Project
A Genetic Algorithm Model for the School Timetabling Problem at the University of the Philippines Rural High School

L. Julongbayan

Date of Submission: December 28, 2024

1. INTRODUCTION

Timetabling is a complex problem that involves scheduling a set of events or activities within specific time slots and resources while satisfying various constraints. In the context of educational institutions, such as schools or universities, timetabling typically refers to the assignment of classes, exams, or other activities to available resources like classrooms, set of students, and teachers [2]. The goal is to create an efficient and feasible schedule that meets the educational needs of students and the operational constraints of the institution.

One common approach is to utilize traditional optimization techniques like mixed integer programming (MIP) [7]. The mixed integer programming utilizes decision variablesthat are set to integral values in finding optimal solutions to the problem. In this method, binary variables are set such that constraints are defined as equations or inequalities that arises from the combination of other variables, and that the binary variables serve to represent yes or no decisions involving the constraints. All equations and inequalities from the constraints must be met by all the decision variables to achieve a valid solution.

A more advanced approach, and the focus of this paper, is the use of genetic algorithms (GAs) [5] to solve timetabling problems. Genetic algorithms are inspired by the process of natural selection, utilizing mechanisms such as mutation, crossover, and selection to iteratively improve candidate solutions. GAs excel at exploring large solution spaces efficiently, making them highly suitable for complex and constrained problems like timetabling. By encoding schedules as chromosomes and defining fitness functions to evaluate their quality, GAs can optimize various factors such as resource utilization, conflict minimization, and user-defined preferences.

Compared to MIP, genetic algorithms offer significant advantages in flexibility and scalability. While MIP systematically explores the solution space to guarantee optimality, it can become computationally expensive and impractical for large-scale or highly complex timetabling problems. Genetic algorithms, on the other hand, provide near-optimal solutions within a reasonable computational time, making them particularly well-suited for real-world applications where calculation cost and adaptability are important considerations.

This paper proposes the use of a genetic algorithm to solve the problem of school timetabling at the University of the Philippines Rural High School (UPRHS), the laboratory high school of the University of the Philippines Los Baños (UPLB). The objective is to generate a weekly class schedule that avoids conflicts in the schedules of teachers, students, and classrooms. In this specific timetabling problem, considerations include the number of classrooms, the suitability of a classroom for particular subjects, the number of teachers, and the number of available time slots. By employing a genetic algorithm, the study seeks to develop an efficient and effective scheduling solution that balances the constraints and requirements of the school's operations while maintaining a manageable computational cost.

2. OBJECTIVES

The general objectives of this project is to create a genetic algorithm model based on a collection of constraints that generates a schedule with little to no violation of any constraints imposed.

The specific objectives of this project are 1) to identify all combination of subjects, sections, teachers, and class instances as single events with a certain duration; 2) to model all the constraints that defines all undesirable conflicts in the school schedule; and 3) to generate an algorithm based on a genetic algorithm model to solve all constraints and generate schedule.

3. METHODOLOGY

The first step in solving the timetabling problem is data acquisition. This involves identifying and collecting relevant data such as subjects, sections, type of room used by each subject, rooms available, teachers teaching the subjects, and timeslots available for the scheduling process. The acquired data is coded so that the data privacy rights of the teachers and students in the school are upheld.

Once the data is acquired, the next step is pre-processing of data. During this stage, the focus is on cleansing the data by transforming all data to the a desired and standardized format which is easier to be encoded in the variable lists of values. This may involve encoding categorical variables and extracting any qualitative data embedded in the notation.

Following pre-processing, the modeling phase begins. The subsequent phase involves constructing the Genetic Algorithm model, where mathematical representations of timetabling constraints and criteria are developed.

Once an optimized solution is obtained, the validation phase ensures the reliability of the model. The generated timetable is compared against the original requirements and constraints, and the quality of the solution is assessed using conflict checking functions. The schedule acquired is plotted and checked for any possible conflict in terms of sections, teachers, or classrooms in a given timeslot.

3.1 Data Acquisition

Data from UPRHS were acquired on January 10, 2024. The data obtained includes subjects each with coded sections and teachers, as well as the type of room required by the subject. The duration of each classes were also obtained, as well as their number of instances within the week.

Considering all instances of all subjects taught in all sections, there are a total of 688 unique classes with varying duration. Classes of the same subject and section are typically held twice or thrice each week. The unit of duration is standardized to 30 minutes. The duration of classes in UPRHS ranges from duration 1 (30 minutes) to duration 4 (2 hours).

There are a 8 room types and total of 24 physical classrooms (15 regular rooms, 4 life skills rooms, one science laboratory, one computer laboratory, and 3 commonly-used physical education facility) and one placeholder classroom for swimming pool and 24 placeholder classrooms for IST (online classes) assigned to each Sections. The placeholder classrooms are added to allot a value to the room variable swimming class, and independent study time (IST) events.

The available timeslots for scheduling are all the 30-minutes interval duration from 7:00 AM to 6:00 PM of a given day, from Monday to Friday. There are a total of 110 timeslots available for all the classes.

Course	**Section**	**Teacher**	**Room**	**Duration**
HRO 7	S1	T1	Y9	2
HRO 7	S2	T2	Y9	2
HRO 7	S3	T3	Y9	2
HRO 7	S4	T4	Y9	2
MATH 7	S1	T5	Y1	2
MATH 7	S1	T5	Y9	3
MATH 7	S2	T5	Y1	2
MATH 7	S2	T5	Y9	3
MATH 7	S3	T6	Y1	2
MATH 7	S3	T6	Y9	3
MATH 7	S4	T6	Y1	2
MATH 7	S4	T6	Y9	3

Table 1: Sample of Course and Room Data

The objects used for the School Timetabling Problem are described in Table 2.

3.2 Constraints

Teachers	The object Teachers has an ID and name.
Rooms	The object Rooms has an ID, name and and room type.
Sections	The object Sections has an ID and name.
Subjects	The object Subjects has an ID and name.
Class	The object Class contains Teacher who teaches, Section of students, the Subject to be teached, the Duration of class, and type of room to be used.

Table 2: Description of Objects

For this paper, only hard constraints are implemented listed below. These constraints when not satisfied will lead to infeasible solution.

- C1: Room has no overlapping classes.
- C2: Class must be in correct room type.
- C3: Teachers have no overlapping classes.
- C4: Sections have no overlapping classes.
- C5: No class must be assigned a timeslot corresponding to the lunch time (12:00 PM - 1:00 PM) of each day.
- C6: Online class must be held on Monday while the other classes must be held on days except Monday.
- C7: No class must be assigned a timeslot corresponding to the weekly flag ceremony (Wednesday 7:00 AM - 7:30 AM) during the specified timeslot.

3.3 Genetic Algorithm Model

The genetic algorithm model used in this study is inspired by the method proposed by R. Lakshmi et al.[8] in their paper, "A New Biological Operator in Genetic Algorithm for Class Scheduling Problem" [Lakshmi et al.]. The population generation phase adopts a hybrid approach combining random selection and a greedy algorithm. A random starting point is chosen initially, and entities are then iteratively added to the schedule. Each addition is evaluated to ensure it satisfies the predefined constraints and fits within the structure of the solution. This approach guarantees that all hard constraints are strictly adhered to during the population generation phase, ensuring the validity of the initial solutions.

The evaluation phase determines the fitness of each chromosome, which serves as a measure of its quality relative to the desired solution. Fitness quantifies how well a chromosome meets the objective criteria and constraints. Similar to the approach described by Lakshmi et al., the fitness evaluation step in this model ensures that constraints are treated with uniform weight, thereby promoting balanced assessment. This step also triggers the termination condition when a specified fitness threshold is achieved. To calculate fitness, each chromosome is assessed against the constraints outlined in Table 1. By incorporating insights from R. Lakshmi et al., this phase emphasizes the importance of constraint satisfaction and balanced optimization in the evaluation process.

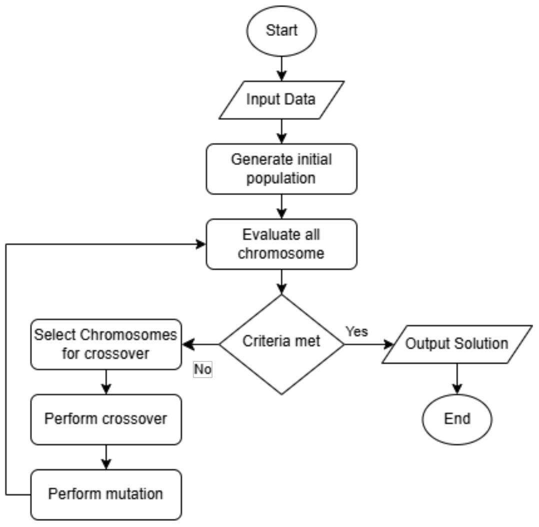


Figure 1: Flowchart of Genetic Algorithm.

3.3.1 Chromosome

Genetic algorithms work by evolving a set of solutions into better ones. Each solution is often referred to as a chromosome, and a group of chromosomes creates a population. In the context of class scheduling, a chromosome represents a potential schedule. This schedule must encode which teacher will teach which section, for which course, in which room, and at what time slot.

3.3.2 Selection

Selection identifies parent chromosomes for reproduction. Rank-based selection is used, where random subsets of chromosomes are compared, and the best-performing ones are chosen as parents. This method efficiently balances constraints like avoiding teacher conflicts and ensuring room availability. However, its high selection pressure requires careful tuning to maintain population diversity and prevent premature convergence.

3.3.3 Crossover

Crossover combines genetic material from parent chromosomes to produce offspring. Order-based crossover, as suggested by Lakshmi et al., is used for class scheduling due to its ability to preserve logical structures, such as grouping a teacher's classes to minimize idle time. Offspring are evaluated to ensure compliance with all hard constraints, like room and teacher availability, while improving the overall schedule.

3.3.4 Mutation

Mutation introduces variability into the genetic algorithm by modifying one or more genes in a chromosome. As noted in R. Lakshmi et al.'s study, mutation is crucial for maintaining diversity in the population and preventing premature convergence.

In the context of class scheduling, mutation involves small changes to an existing schedule. Examples include swapping two time slots, reassigning a teacher to a different class, or rearranging the order of classes within a schedule. These mutations are constrained to ensure that no hard constraints, such as room availability or teacher conflicts, are violated.

3.3.5 Fitness

Fitness evaluation measures the quality of a chromosome relative to the desired solution. Inspired by R. Lakshmi et al., this study evaluates fitness by assessing how well a schedule satisfies the defined constraints (Table 2). Each constraint is assigned equal weight, ensuring a balanced evaluation. Constraints include avoiding teacher and room conflicts, adhering to room capacities, and ensuring proper allocation of time slots.

A fitness score is calculated for each chromosome, where higher scores indicate better alignment with constraints. This phase also triggers termination when the fitness of a chromosome meets or exceeds a predefined threshold, signaling that an optimal or near-optimal solution has been found. Fitness evaluation plays a crucial role in guiding the genetic algorithm toward high-quality solutions while maintaining adherence to hard constraints.

4. RESULTS AND DISCUSSION

The utilization of Genetic algorithm model to address a complex timetabling problem revealed notable challenges and successes in achieving optimal scheduling solutions. Initial implementation failed to generate a solution that satisfies all constraints or fitness criteria, but the subsequent refinement of the algorithm ensured a solution generated with a reasonable fitness score. This approach led to important breakthroughs, producing solutions aligned with the timetabling requirements that had eluded the initial implementation.

To speed up the execution, we have tried to improve the algorithm by directly restricting the domains of the classroom for online classes versus regular classes. Online classes were hard-coded to assign on Monday, while regular classes are assigned on other days. Lunchtime and flag ceremony time have been removed in the initialization of chromosomes to give the initial population a good fitness start. However, mutation step might still assign the classes to said constrained time slots.

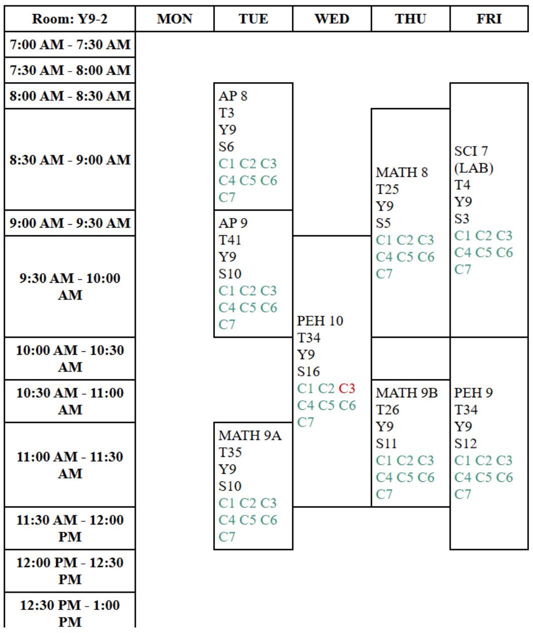
The following are the parameters used for the Genetic algorithm for this paper.
No. of chromosomes: 100 (Population Size)
Selection Method : Rank-based Selection
Selection Rate: 10%
Crossover: Two Point Crossover
Crossover%: 80%
Mutation Rate: 3%

The fitness score criterion used for the algorithm is 0.99. As shown in Table 3, first 2500 generations had a fitness score of around 0.90 took 1.3 hours. At 10,000 generations, the fitness score is only around 0.96 with run time of 5 hours. The program was forcibly stopped without hitting the stopping criteria at 0.985 fitness score in 100,000 generations as no significant increase in score observed in last 50,000 for 26 hours. It can be observed that as the fitness score approaches to 1, the convergence slows down. Proper mutation rate scheduler rate is recommended to hasten the convergence of the solution.

Generation	Ftiness	Runtime (hrs)
2500	0.900	1.3
10,000	0.959	5.2
50,000	0.981	26.0
100,000	0.985	52.0

Table 3: Fitness Score and runtime

The final schedule output is plotted in an html file for simplicity of checking the solution. Since the stopping criterion is met, the solution has few constraints violated. Shown in Figure below is the sample schedule for Room Y9-2. For subject AP 8, it can be observed that all constraints are satisfied as indicated in the green font notations (C1 to C7). Meanwhile, for subject PEH 10, C3 is violated as indicated in red font.



Room: Y9-2	MON	TUE	WED	THU	FRI
7:00 AM - 7:30 AM					
7:30 AM - 8:00 AM					
8:00 AM - 8:30 AM		AP 8 T3 Y9 S6 C1 C2 C3 C4 C5 C6 C7			SCI 7 (LAB) T4 Y9 S3 C1 C2 C3 C4 C5 C6 C7
8:30 AM - 9:00 AM				MATH 8 T25 Y9 S5 C1 C2 C3 C4 C5 C6 C7	
9:00 AM - 9:30 AM		AP 9 T41 Y9 S10 C1 C2 C3 C4 C5 C6 C7			
9:30 AM - 10:00 AM			PEH 10 T34 Y9 S16 C1 C2 C3 C4 C5 C6 C7		
10:00 AM - 10:30 AM					PEH 9 T34 Y9 S12 C1 C2 C3 C4 C5 C6 C7
10:30 AM - 11:00 AM				MATH 9B T26 Y9 S11 C1 C2 C3 C4 C5 C6 C7	
11:00 AM - 11:30 AM		MATH 9A T35 Y9 S10 C1 C2 C3 C4 C5 C6 C7			
11:30 AM - 12:00 PM					
12:00 PM - 12:30 PM					
12:30 PM - 1:00 PM					

Figure 2: Plotted Timetable Solution.

5. CONCLUSION

Based on our initial implementations, it is crucial that the variables and constraints are defined precisely in the preprocessing and modeling phase of the project. A combination of constraints that completely exhausts the domain of at least one of the variables is sufficient enough to yield a no solution run.

It has been evident that the run time of an optimization method like the Genetic algorithm is dependent on the efficiency of the code, as well as the parameters used in the model. In optimization methods, we are guaranteed to have a solution given enough time to run and an accurate algorithm. However, application of stopping criteria evidently helped to find satisfying solution with the interest of time.

The successful application of Genetic algorithm model in our school timetabling problem is a significant achievement towards the enhancement of the school's scheduling processes in the future. The results underscore the model's flexibility and ease of use in harmonizing diverse constraints, minimizing conflicts, and optimizing resource utilization. The generated timetable not only resolves immediate scheduling challenges but also lays the groundwork for sustainable, long-term planning.

As academic institutions increasingly strive for operational efficiency and pinpoint accuracy, the Genetic algorithm model is a reliable model of the timetabling problem that guarantees us a valid solution, given enough time, if the solution mathematically exists.

In essence, this project contributes to the diverse results in the timetabling problems around the globe, which has been a crucial problem to solve given the global limitations of resources such as time, teachers, and classrooms.

6. RECOMMENDATIONS

Based on what we have achieved in this project, we would recommend several possible continuation to this specific instance of timetabling problem. One of which is the inclusion of teacher as a new variable that can teach on different sections of a subject instead of a pre-determined section to teach. This would give more freedom on the algorithm to assign teachers on more classrooms and timeslots.

Another recommendation is to explore the use of a constraint programming approach or MIP completely or as an auxilliary to the currently used method in this project. The use of soft constraints and penalty systems may also be considered if ever this project be continued.

While the genetic algorithm implemented in this study draws heavily from the work of R. Lakshmi[8], additional tuning of hyperparameters is recommended to further optimize performance for specific class scheduling problems. Hyperparameters such as the number of chromosomes, selection criteria, and crossover strategy can significantly influence the quality and efficiency of the generated solutions specifically, mutation rate update as the model approaches convergence.

Lastly, code optimization is recommended to run the program with multi-threaded implementation to hasten the convergence of genetic algorithm.

7. REFERENCES

[1] S. Abdullah, E. K. Burke, and B. McCollum, "A hybrid evolutionary approach to the university course timetabling problem," *IEEE Congress on Evolutionary Computation*, 1764-1768, 2007

[2] E. Alexa, D. Zmaranda, and E.V. Moisi, "Modeling and solving the university timetabling problem using the constraints satisfaction model: A Case Study," *13th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, 1-6, 2021

[3] O. Chávez-Bosquez, J. Hernández-Torruco, B. Hernández-Ocaña, and J. Canul-Reich, "Modeling and solving a Latin American University course timetabling problem instance," *Mathematics*, 8(10), 833. 2020

[4] M. A. Cruz-Chávez, M. Flores-Pichardo, A. Martínez-Oropeza, P. Moreno-Bernal, and M. H. Cruz-Rosales, "Solving a real constraint satisfaction model for the university course timetabling problem: A case study," *Mathematical Problems in Engineering*, 1–14. 2016

[5] A. K. Herath, "Genetic algorithm for university course timetabling problem," *Electronic Theses and Dissertations.* 443. 2017

[6] V. Pereira, and H. G. Costa, "Linear integer model for the course timetabling problem of a faculty in Rio de Janeiro," *Advances in Operations Research*, 1–9. 2016

[7] E. Rappos, E. Thiémard, S. Robert, and J. Hêche, "A mixed-integer programming approach for solving university course timetabling problems," *Journal of Scheduling*, 25(4), 391–404. 2022

[8] R. Lakshmi, S. Kalpana, and P. Deepalakshmi, "A New Biological Operator in Genetic Algorithm for Class Scheduling Problem," Proceedings of the International Conference on Computational Intelligence and Computing Research, 2014.