ISSN: 2706-9087

Journal of Humanitarian and Applied Sciences
Issue 13 – Volume 7

مجلـة العلــوم الإنســانيــة والتطبيـقيـة
العدد 13– المجلــد 7

An automated study schedule via genetic algorithm

Ahmed Almantsri[1] Mohamed Alhamrouni[2] Aisha Basheer E'atiwa[3]

[1][2][3] Computer Science Dept. Faculty of Arts & Science Kasr Khiar, Elmergib University-Libya
Correspondence to: [1] aaalmontasery@elmergib.edu.ly - [2] maalhamroni@elmergib.edu.ly

الملخص

الجدول الدراسي هو واحد من الاحتياجات التعليمية الأساسية. ومع ذلك، عندما يتطلب أتمتة الجداول، سيكون توزيع المحاضرات الدراسية أكثر تعقيدا وصعوبة استناد على القيود المفروضة لإنشائها، وتختلف هذه القيود وفقا لأهمية كل قيد ومدى إمكانية تطبيقه للوصول إلى الحل الأمثل.

هناك العديد من الطرق والخوارزميات المستخدمة في عملية أتمتة الجداول الدراسية، وتختلف هذه الطرق باختلاف نتائجها. في هذا البحث، قمنا باستخدام الخوارزمية الجينية وحققنا نتائج مميزة يمكن الوثوق كحل لمؤسسة حقيقية لإنشاء جداولها الدراسة بشكل مؤتمت لمنع أي تعارض بين مكونات الجدول بناء على قائمة القيود التي وضعناها لهذا النهج.

الكلمات المفتاحية: الجدول الدراسي، الأتمتة، الخوارزمية الجينية، النظرية التطورية، NP-complete.

Abstract

A study schedule is one of the basic educational needs. When it, however, becomes an automated requirement, it will be more complicated and harder to solve and distribute study lectures without considering the restrictions and limitations imposed to create study schedules as these constraints differ according to the importance of each constraint and the extent to which it can be applied to reach the optimal solution.

There is a dozen of methods and algorithms used in the process of study schedules, and these methods differ according to their results. In this research, we used the genetic algorithm and we achieved distinctive results that can be trusted for a real institution to generate an automated study schedule with free-conflict lectures based on the constraints list we set for this approach.

Keywords: study schedule, automation, genetic algorithm, evolutionary, NP-complete.

1. Introduction

One of the main aspects of organizing the study process in higher education institutions with its various specializations and degrees is the issue of the study schedule. The study schedule is considered one of the most common problems for educational institutions, and it gets more complicated when the number of rules increases for these institutions (Rochman, Syakur, Rachmad, & Imamah, 2019) (Premasiril.D.M, 2018). There have been many studies conducted on this issue, and the results obtained from these studies explain the importance of finding solutions for timetabling, and the reason is the difference in data and restrictions between the educational institutions to which the study was applied. The process of creating study schedules before using any technologies was difficult and requires a tremendous

amount of work during every new academic year, to get started smoothly and efficiently. This problem periodically occurs at the beginning of each semester (Wen-jing, 2018). Therefore, the goal is to automate the process of finding the best distribution for a limited number of study lectures with A specific number of classrooms to solve all constraints related to the study schedule for the targeted institution (Chaudhari, Dmello, Shah, & Bhangale, 2022).

GA clears up the limitations to improve the study schedule without any restrictions or conditions, also there are some other problems such as traveling salesman which is counted as NP-Complete (Britannica, 2021) and requires more effort to solve (Adewole Philip, 2011) (Reinelt, 1991). TSP describes calculating or choosing the shortest route from one point to another. The best example for TSP is Google Maps which provides the end-user with the road to reach the destination. Open-Pit Production Scheduling (OPPS) is another NP-complete problem and it can be solved using GA to encode the solutions for it (Alipour, Khodaiari, Jafari, & Tavakkuli-Moghaddam, 2020).

However, time consumption is the main concern about solving this issue, as the problem becomes more complicated and sometimes becomes impossible to solve because of the lack in one of the basic requirements for study schedule. On the other hand, the more constraints we have, the harder it gets to find the optimal solution much faster (Dener & Calp, 2018).

Therefore, this study aims to create a system that can find a solution and generate a study schedule and obtain an automated system that provides flexibility for the process of creating study schedules compared to some previous works on the same issue.

2. Literature review

Before, some studies presented some methods to solve the problem of an automated study schedule, taking into account the requirements and constraints imposed by the scheduling process. (Adewole Philip, 2011) used the Genetic algorithm to find and solve the self-scheduling problem, but they discovered that the study schedules are among the difficult problems that are complicated to solve by increasing the restrictions imposed on the study schedule. GA is one of the best-used algorithms to solve timetabling problems (Zhu, Li, & Li, 2021), even though, it does not guarantee finding the optimal solution in every single test. It is all about the complexity of constraints and time consumption.

This can explain the effectiveness of solving the problems of study schedules. It has shown great potential in scheduling formation as a directly applicable solution to a very wide range of scheduling problems. Compared to other real problems, more research still needs to be done to measure the performance of this evolutionary algorithm.

(K. Abeywardena, 2021) proposed GA to solve the study schedule, and they solve hard constraints only to reduce the required time taken to initiate the timetable. This concept is logically acceptable because soft constraints will affect the process by enforcing new rules which might be incompatible with each other.

Time consumption is the main concern when it comes to solving NP-Complete problems. (BALAN, 2021) focused on the required time to solve this issue regarding the number of limitations for study schedule.

3. The Proposed Approach.

In this paper, we chose the genetic algorithm to solve the study schedule problem. Based on the previous study, this algorithm showed great results in finding the optimal solution.

3.1 Genetic Algorithm.

The concept of a genetic algorithm was discovered by John Holland, professor of psychology at the University of Michigan, in the early 1970s (Holland, 1992). It was based on Darwin's theory of genetic evolution. According to Darwin's theory, everyone needs to struggle to survive, and for those who have the genetic features of "fitness" they will have a greater chance of survival, which is called natural selection.

Natural selection passes genetic traits on to the next generation. after a few generations, the genetic traits will become dominant in the population, thus evolving the population based on the "survival of the fittest".

The previous process can be described as follows:

1. Population initialization.
2. Generation assessment (Fitness Calculation).
3. Parents' Selection.
4. Crossover.
5. Mutation.

3.2Genetic Algorithm stages.

Before we start explaining these five phases, we need to demonstrate the essential requirements.

1- Chromosome representation

In this step, we specify the representation method for the chromosome. It is a set of values, and every value represents a certain feature of the chromosome. There are some methods to represent chromosomes, but before we mention them, we need to identify the traits that make up the structure of the chromosome.

Course	Timeslot	Room	Level

whereas:

- The course represents the course code.
- The time Slots represent lecture time for courses.
- The Room represents the room number for the lecture.
- The Level represents the semester number.

2- Representation Methods

After specifying the chromosome structure, now we will list all methods that can be used in displaying the previous features.

- **Integer Method.**

In this method, each gene represents a series of non-repeating numbers arranged in a sequence, for example:

Data Mining Course	22
Lecture Time	12
Room Number	3
Level (Semester)	8

Example 1: represents chromosome structure using integers.

- **Binary method.**

This method is frequently used because, you can use zeros and ones to present features for chromosome (Nguyen, Duong, & Nguyen, 2021).

Chromosome 1	101100 100010 110011 000101
Chromosome 2	101101 011100 010011 000101

Example 2: represents chromosome structure using binary and first six digits in the structure represent the course.

- **Floating point method.**

It is a series of four float values to represent the chromosome.

Chromosome 1	1.2344 4.6792 3.5205 2.5704
Chromosome 2	4.3230 3.1012 2.0023 1.2333

For example, the third float value of the structure represents the Room.

Chromosome structure is very important. The longer chromosome is, the more time is required to solve the issue. Therefore, we decreased the size of the chromosome as much as possible.

In this paper, we used the integer method to represent the chromosome structure. This structure consists of as the follows:

n. represents the subjects, where n = the subject code.
m represents time, where m = the period of the lecture.
z represents the rooms, where z = the room number.
k represents the level (semester), where k = the level.

3- **Constraints**:

Constraints are divided according to importance into two main types:

- **Hard Constraint**

There are some constraints that cannot be bypassed and must be resolved to obtain a usable schedule (Wen-jing, 2018). For example, if a faculty member is not present at the same time twice at the same period of time.

- **Soft Constraint**

These constraints can be ignored because study schedule depends on preventing all mandatory conflicts (Sakaliuk & Trishyn, 2021), while soft constraints are desirable which can be explained as "if it is possible, then do it". In addition, these limitations will require more time to proceed, and it means a high number of generations (Tung, Jaafar, Abdul-Aziz, Nguyen, & Boi, 2021). Here are some examples:

1) Professor prefers a certain time
2) Course is preferred to be in the morning
3) Students prefer specific lectures be continuous at the same day.

Based on these constraints, the process of finding a solution for study schedules is quite difficult. Our case study has some obstacles including the following:

1) Lack of rooms or labs compared to the number of courses.
2) Three different durations.
3) Lab requirement.

Because of the mentioned obstacles, we have set a list of basic constraints, which are hard constraints to prevent the above-mentioned issues.

Table 1. introduces the hard constraints.

Features	**Constraint**
Levels	Each constraint includes some subjects
	Each level cannot reserve two lectures at the same time
Rooms	Each room can handle one lecture based on timeslot
	All subjects cannot occupy any room at the same time
Courses	Every subject can show only one or two times in the generated table
	Each subject cannot reserve two rooms in the entire schedule
Timeslot	Each time slot can be assigned based on the number rooms.

We excluded all soft constraints due to the mentioned obstacles because it will require more time to generate the study schedule and it may cause an infinity loop, and thus, we will not be unable to find the optimal solution quickly (East, 2019).

3.2.1 Population Initialization

The process of determining population density is one of the most important things to make the algorithm work correctly to obtain sound results. Among the characteristics of the study table that help determine population density is the number of subjects.

The length of the chromosome in this project was determined in 4 characters or features, and the size of the population is "number of courses = n", and then a set of random numbers is generated to represent the initial values of these chromosomes as individuals in the population.

1,2,3,4,5,6,7,……..,n

3.2.2 Fitness Calculation

It is the function that evaluates the gene of the current generation to select the best chromosome to be used as Offsprings.

We calculated the fitness function based on: (K. Abeywardena, 2021)

$$Fitness = (\sum_{i=0}^{k} (C_i * w_i)/(n * m))$$

whereas: -

C_i represents the constraints
W_i represents the weights of the constraints
n is the sum of the time periods
m represents the number of rooms

The evaluation of every generation is done based on the value of the fitness function, where it is possible to judge the development of future generations if this function exceeds a certain value of the gene (Nurminen, 2022).

If the value of the fitness function exceeds 0, it means we have a conflict and it has to be resolved.

3.2.3 Parents' Selection

Based on the evaluation of the fitness function of each gene, a percentage of the overall evaluation of the generation is assigned to each gene and we need to set the offspring percent to produce the next generation of chromosomes. Parents of the next generation are chosen by one of the following methods:

- Roulette wheel algorithm
- Competition tournament
- Random selection
- Best selection
- Top percent selection

We chose the roulette wheel as a selection method, as this method depends on choosing the best in the population density of individuals after calculating the total fitness for the population, and the values are used as a probability to choose the best genes in a subset of parents.

Table 2. shows the process of selecting parents using the Roulette Wheel.

Chromosome	1	2	3	4	5	6	7	8	9	10
Fitness	7	1	15	2	10	11	6	9	4	6
Accu. Fitness	7	8	23	25	35	46	52	61	65	71

3.2.4 Crossover

This process helps generate new chromosomes as a new offspring from the selected parents to generate a new child (gene) with healthier features, so it can be used in the next to generate to find the optimal solution.

We use a single-point random selection within the two parents and then start swapping values from the two selected parent genes to mix them and get new genes with new features.

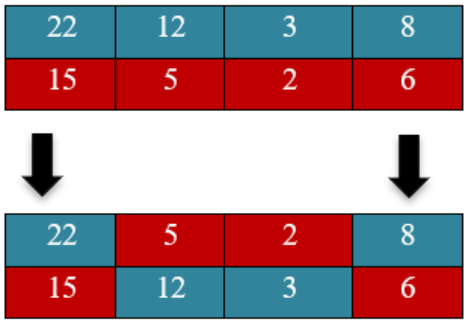

Figure1: shows the crossover process.

3.2.5 Mutation

The process comes after applying the crossover step. First, we specify the mutation rate of targeted genes, then we apply a random swapping for these genes to produce new features in the best chromosomes based on the fitness value. In this process consider the mutation rate and the degree of probability of a mutation, if the mutations are likely the best chromosome loss. However, if too few mutations, the chromosome will long to find the optimal solution (Utama, 2016).

3.3 Flowchart of GA Algorithm:

We used the Computer Department of Arts and Sciences faculty at Elmergib University as our main case study. We aim to generate an automated study schedule using a

genetic algorithm without any conflict among chromosome features to gain a user-friendly schedule.

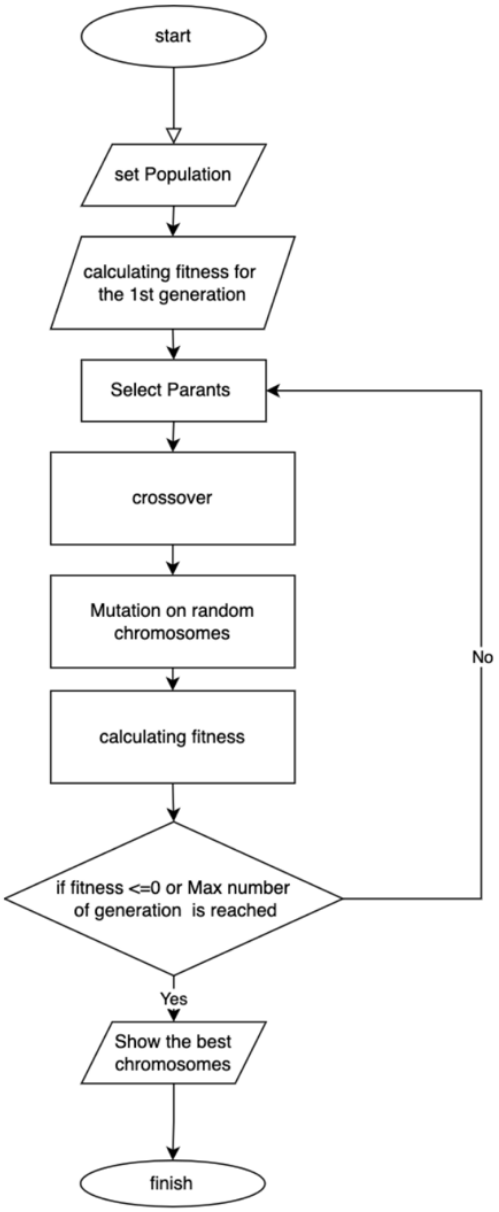

Figure 2. shows the flowchart of how the genetic algorithm works

4. Test and results

Our experiment consists of ten epochs to verify the efficiency of the method and to find the optimal solution by measuring the time consumption for each experiment. It is clear from the results that the genetic algorithm needs several generations to get the optimal solution. The difference in results is due to some reasons, the most important of which is the mutation process which leads to improving genes and obtaining better generations based on

the fitness value but it may affect the result because of the randomness in adjusting the select chromosomes.

The table (2) summarizes the settings for our model:

Population	45 choromosome
Representation Method	Integer
Parent Selection	50%
Mutation Rate	5%
Max No. of generations	3000
No. Of Constraints	7

4.1 Output Design

We designed the user interface for the system to look the same as in the case study institution. Our result appears when we click on generate schedule option on the main window for the system. It will create a conflict-free schedule using the genetic algorithm. The following figure shows this result:

Figure 3. shows the result of generating new table using the proposed technique.

This main window includes the following options:

1- **Courses**: to manage courses that will be used to generate the study schedule.
2- **Rooms**: to manage rooms that will be used to generate the study schedule.
3- **Levels**: to manage eight semesters and connect them with courses.
4- **Time slot**: to add time slots for the lectures
5- **Generate Schedule**: it is the main option to start generating the study schedule
6- **Show Schedule**: to check the generated schedule which will be saved in our database.
7- **Export Schedule**: to export the schedule to department's template.
8- **Delete Schedule**: to delete the last generated schedule.

9- **Exit**: to close the main window of the system.

4.2 Result

After generating the study schedule, we can export the final table by clicking on the seventh option on the main window to get an Ms. Excel file and use it as a real solution. There are some common constraints, and if one of them is achieved, the other constraint has already been achieved, which helped speed up access to the final solution, e.g. sorting prerequisite courses in different time slots will prevent conflicts when students start enrolling in the new semester.

Table No. (3) shows the process of testing experiments

Experiment	Generations number
1st	503
2nd	178
3th	1830
4th	577
5th	1536
8th	2288
9th	452
10th	207
11th	296

The above result shows that the genetic algorithm can be affected by the random distribution of genes, and figure (5) shows the results for these experiments.

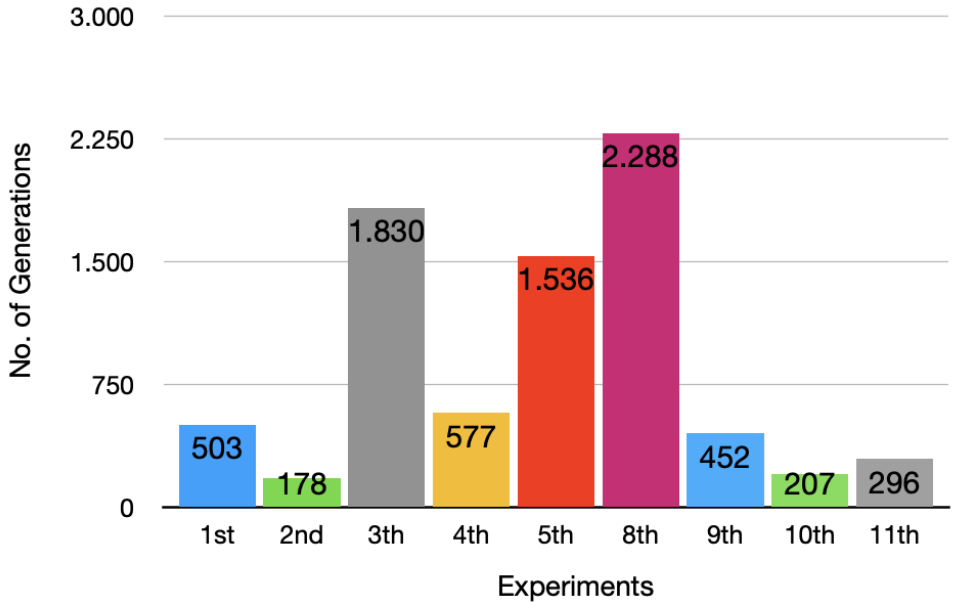

Figure 4. illustrates the results of the proposed method experiments

5. Conclusion

we concluded that the genetic algorithm can automate electronic tables, and try to find solutions to the specific constraints whether it is hard or soft. We can note that the number of generations varies from one experiment to another, which indicates that the solutions are affected by the type of constraints presented and that the result is affected according to the data presented for the solution by calculating the fitness of genes and applying the intersection and mutation processes, which leads to either an increase or a decrease in the number of generations in a direct way to reach the solution optimum.

However, there are some constraints, especially the soft ones, that might increase the number of generations noticeably, which affects the process of obtaining a schedule in a short time. For example, a faculty member prefers certain time and days. Here it will be a little different and it will become difficult to satisfy their desirable lectures.

Finally, we can conclude that the genetic algorithm is one of the best available algorithms for solving NP-complete problems due to the possibility of developing the solution and improving the results.

6. Future Works

Based on our conclusion, we decided to develop the performance of the genetic algorithm by reducing the number of generations and preventing randomness by improving the performance of previous experiments, and thus, increasing the efficiency of the algorithm and helping to find much faster solutions with much better results. The development process can include integrating the algorithm with one of the available learning methods such as neural networks and supervised learning techniques.

7. References

Wen-jing, W. (2018, May). Improved Adaptive Genetic Algorithm for Course Scheduling in Colleges and Universities. *International Journal of Emerging Technologies in Learning (iJET), 13*(06), 29-42.

Rochman, E. M., Syakur, M. A., Rachmad, A., & Imamah. (2019). Subject Scheduling Using Genetic Algorithms (Case Study: SMK Negeri 1 Labang-Madura-Indonesia). *Journal of Physics: Conference Series*.

Premasiril.D.M. (2018). University Timetable Scheduling Using Genetic Algorithm Approach Case Study: Rajarata University OF Sri Lanka. *International Journal of Engineering Research and Applications (IJERA), 8*(12), 30-35.

Adewole Philip, A. A. (2011). A Genetic Algorithm for Solving Travelling Salesman Problem. *(IJACSA) International Journal of Advanced Computer Science and Applications, 2*(1), 26-29.

Britannica, T. E. (2021, June 14). *NP-complete problem*. (Encyclopedia Britannica) Retrieved June 8, 2022, from https://www.britannica.com/science/NP-complete-problem

Oladipo, W. K., Bamidele, A. O., & Olalekan, A. M. (2019, February). Automatic Timetable Generation using Genetic Algorithm. *International Journal of Applied Information Systems (IJAIS), 12*(19).

K. Abeywardena, G. D. (2021). A Genetic Algorithm for University Timetabling . *the proceedings of International Conference on Multidisciplinary Approaches in Science (ICMAS)*.

BALAN, I. (2021, Jan). A New Genetic Approach for Course Timetabling Problem. *Journal of Applied Computer Science & Mathematics, 15*(31).

Reinelt, G. (1991). TSPLIB—A Traveling Salesman Problem Library. *ORSA Journal on Computing*, 376-384.

Utama, S. A. (2016, 11). Scheduling Courses Using Genetic Algorithms. *International Journal of Computer Applications, 153*, 975-8887. 10.5120/ijca2016911984.

Holland, J. H. (1992). Genetic Algorithm. *Scientific American, 267*(1), 66-73. http://www.jstor.org/stable/24939139. Accessed 24 Jun. 2022.

Chaudhari, Y. S., Dmello, V. W., Shah, S. S., & Bhangale, P. (2022). Autonomous Timetable System Using Genetic Algorithm. In *2022 4th International Conference on Smart Systems and Inventive Technology (ICSSIT)* (pp. 1687-1694).

Nurminen, K. (2022). *Automating construction workforce allocation and scheduling using genetic algorithms*.

Nguyen, T. S., Duong, H. T., & Nguyen, K. D. (2021). *Automatic Generation of Course Schedules Using Genetic Algorithm.* Retrieved from https://doi.org/10.1007/978-3-030-97610-1_4

Dener, M., & Calp, M. H. (2018). SOLVING THE EXAM SCHEDULING PROBLEMS IN CENTRAL EXAMS WITH GENETIC ALGORITHMS. *Mugla Journal of Science and Technology*, 102-115.

Rivera, G., Cisneros, L., Sanches-solis, P., Ranges, V. N., & Rodas, O. G. (2020). *Genetic Algorithm for Scheduling Optimization Considering Heterogeneous Containers: A Real-World Case Study.* Mexico: Department of Electrical and Computer Engineering, Autonomous University of Cd. Juárez, Cd. Juárez 32315, Mexico.

Alipour, A., Khodaiari, A. A., Jafari, A., & Tavakkuli-Moghaddam, R. (2020). Production scheduling of open-pit mines using genetic algorithm: a case study. *International Journal of Management Science and Engineering Managemen*, 176-183.

Tung, S. N., Jaafar, J. B., Abdul-Aziz, I., Nguyen, H. G., & Boi, N. N. (2021). Genetic Algorithm for Solving Multi-Objective Optimization in Examination Timetabling Problem. *International Journal of Emerging Technologies in Learning*.

Zhu, K., Li, L., & Li, M. (2021). A survey of computational intelligence in educational timetabling. *International Journal of Machine Learning and Computing*, 40-47.

East, A. R. (2019). Timetable Scheduling via Genetic Algorithm. *National University of Ireland*.

Sakaliuk, O., & Trishyn, F. (2021). USING A GENETIC ALGORITHM TO SOLVE THE COURSES TIMETABLING CREATION PROBLEM. *Automation of technological and business processes*, 22-28.