Matrik: Jurnal Manajemen, Teknik Informatika, dan Rekayasa Komputer
Vol. 22, No. 1, November 2022, pp. 129∼138
ISSN: 2476-9843, accredited by Kemenristekdikti, Decree No: 200/M/KPT/2020
DOI: 10.30812/matrik.v22i1.2022

Automated University Lecture Schedule Generator based on Evolutionary Algorithm

Yusri Ikhwani[1], Khairan Marzuki[2], As'ary Ramadhan[3]
[1,3]Universitas Islam Kalimantan Muhammad Arsyad Al Banjari, Banjarmasin, Indonesia
[2]Bumigora University, Indonesia

Article Info

Article history:

Received July 01, 2022
Revised October 10, 2022
Accepted November 25, 2022

Keywords:

Automated
Evolutionary Algorithm
Generator
Hard Constraints
Lecture Schedule

ABSTRACT

Lecture schedule is an important element in the learning activity. Creating a lecture schedules for university is a complicated work so in the implementation it have violation of the constraints and it also takes a lot of time since it is created manually. In this paper evolutionary algorithm (EA) is used to create an effective and feasible schedules based on the real data input that is obtained from each department. The objective functions in EA contribute in gaining the fitness function to solve the constraints problem in the schedule by applying weighting for each hard constraints. The objective function is gained from the total of infringement in each soft constraints addition by score weighting. The genetic operator used in EA is stochastic variation Operator. As far as the reproduction operator is concerned, the tournament selection was used with size 3. Crossover operator is conducted after selection process with crossover probability equal to 0.05 and mutation rate is 0.1. The size of population was set to 9 and stopping criteria algorithm was left run for fitness value = 1. The simulation result shows that EA can create lecture schedules efficiently and feasibly. Moreover, it is also faster with the execution time of the proposed EA is less than 30 and easier than creating manually.

Corresponding Author:

Yusri Ikhwani, 082157245971,
Faculty of Information Technology, Study program of Information System,
Universitas Islam Kalimatan Muhammad arsyad al banjari, Banjarmasin, Indonesia,
Email: yusri.uniska@gmail.com.

How to Cite: yusri ikhwani, K. Marzuki, and A. Ramadhan, Automated University Lecture Schedule Generator based on Evolutionary Algorithm, MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer, vol. 22, no. 1, pp. 127-136, Nov. 2022.

1. INTRODUCTION

Lecture schedule is a time allocation process for a certain activity that is already planned in structured and allocating subject to constraints, towards the resource that is given to object to be placed in space time. Scientifically, in the literature, lecture schedule is one of non-deterministic polynomial problems, usually abbreviated as NP-hard problem [1]. One of important components in the learning activity is lecture schedule. The problem of schedule consists of the determination of room and time, teaching learning activity between lectures and students in a certain time that is scheduled before, and fulfilling the constraints. This constraints are divided into two categories, they are hard constraints and soft constraints [2]. When a constraints are fulfilled, developing the lecture schedule can be done easily and feasibly, it means the schedule that is created has been appropriate to be implemented as lecture schedules.

The method that is used to resolve the lecture schedule like an ant colony [3–5] it is succeed to be implemented as a solution in lecture schedule problem which is appropriate with the real condition. Lately, there are a lot of computational intelligence algorithms and techniques including genetic algorithm [6, 7], evolutionary algorithm [8–10], tabu search [11] have been implemented to show the effectivity and reliability of timetable problem. Genetic and Heuristic [12, 13] is a good strategy to solve the complexity by maximizing allocation and minimizing the breaking of constrains as much as possible to determine the best solutions in the scheduling problem.

Current situation in general, lecture schedule consist of entities rooms, courses, lecturers and time periods, that must be scheduled and also consist of a set of constraints that are expected to be satisfied. The term time periods refers to each hour of lecture in 5 working days from monday to friday. The main goal of constructing a lecture schedule is to provide faculty members with a schedule that has good quality. Such an algorithm can construct a quality lecture schedule in only few minutes and in this case the quality of the lecture schedule is formally defined. This has to be done in a way to satisfy a set of constraints. This constrains is focused on the problem of hard constraints. A lecture schedule is called feasible, meaning that it can be used by faculty it was made for, when all hard constraints are satisfied. Hard constraints that must be satisfied in order to keep the lecture schedule feasible are the following Seating capacity and the number of students must fit, The room can only be used to one lecture in the same time, A lecture can only give one lecture at a time, No student can be assigned to more than one course at the same time.

This contribution is focused on problem of constructing a feasible and efficient lecture schedule for faculty of Information technology at universitas islam kalimatan muhammad arsyad al banjari (UNISKA MAB). Constructing lecture schedule for UNISKA MAB is a complicated work so in the implementation it have violation of the constraints, such as room and time clashes, the lectures have different courses in a time, the students have more than one course in a time, the number of students exceed the room capacity, and creating it also consumes a lot of time since it is created manually. If the problem is not solved, it will affect the academic activities in faculty. To avoid this problematic situation we propose the use evolutionary algorithm (EA) in order to create automated lecture schedule. So, by using computers and automated lecture schedule constructing algorithm, lecture schedule can quickly and easily be created. The objective functions in EA contribute in gaining the fitness function to solve the constraints problem in the lecture schedule by applying weighting for each specific hard constraints that should be satisfied. So, the algorithm can be used each time in order to result lecture schedule satisfying specific constraints.

This research is presented and arranged into several parts. The second part discusses about the structure and operation of EA. The third part explains about the result of methodology investigation. The last part is about the conclusion and suggestion of this reseaech.

2. RESEARCH METHOD

In order to resolve the problem of course schedule in the faculty of Information technology at UNISKA MAB, the construction of automatic timetable has been implemented. The motivations that make us design and implement EA in this problem are described following:

1. EA have been already widely used for solving the lecture schedule problem with very satisfactory result [14, 9].
2. Distribution on lecturers, that is how evently each lecturers hours are distributed among the days of proposed lecture schedule.
3. The course hours of each lecturers in a day should be continues-no idle hours are allowed to exist between courses hours.
4. The faculty does not follow any automated approach when construction lecture schedules. Thus, it was a fresh ground to algorithm.

The data used in this paper are available through https://fti.uniska-bjm.ac.id/jadwal-perkuliahan-semester-ganjil-t-a-2020-2021. The structure and how EA is operated discuss in the following subsections.

2.1. Chromosome encoding

The use of chromosome encoding is one of the significant factors to make EA achieve the convergence and reach optimal solution quickly. The term of encoding is used to explain how to present lecture schedule whether or not it is feasible. The encoding designed for this algorithm is quite different from others already proposed in the literature [10, 15]. Each chromosome is marked to all entities needed such as lectures id (NIDN) and semester that can easily fit with different lecture schedule of faculty problem (see Table 1). A list of all the NIDN is presented in the first column. The first row of the chromosome represents all semester. In Table 2, the intersection of NIDN and semester shows the potential combination where the classes can be assigned. Intersections are filled with NewClass ID. NewClass ID is the attribute composite that is stated three aspects at once: subject ID, room ID, time period ID (date, time). As an example, lecture ID 23H2, in the cell intersection between NIDN 1129039102 and semester 1 shows a subject with ID 2, ID room 3, and ID time periods H2.

Table 1. Representation of Chromosome

NIDN/Semester	1	2	3	4	6	7	8	9
1129039101								
1129039102	23H2							
1129039103								
1129039104								
1129039105								

2.2. Procedure initalization

Procedure initialization is an important issue in every EA because it must make lots of variety of population possibility randomly. It is worth to do because the individuals that establish early generation spread to all search space. It gives opportunity to algorithm to perform a search in all space effectively for the possibility solution and to not being trapped in local optima before the time. Initialization in EA is a part of a process of encoding information input into chromosome representation. Initialization procedure consists of the following steps:

For each chromosome **do**
    For each lecture **do**
        For semester **do**
        1. Choose lecture ID randomly.
        2. If the lecture ID chosen has not been set the semester, so it is set to lectures of the chosen semester. If it is on the other way around, back to the first step.
        end
    end
end

2.3. Fitness function

Fitness function is the calculation of each chromosome that is generated to know the chosen chromosome for the next generation. This function is the specific problem and there is no standard in its calculation. To get feasible solution of lecture schedule problem. Fitness function is a weighted sum calculated based on the violation of hard constraints. When calculating fitness value, we give emphasis on weighting to each hard constraints to calculate objective function towards the problem of this paper. To get objective function, the total of infringement in each hard constraints is adding by weighting value. Table 2 shows the hard constraints that we really consider and its weighting.

Table 2. Weighting of hard constraints

Hard constraints	Wheight
The first, room capacity and the number of students must fit (set as 0)	1
Second, the room can only be used to one lecture in the same time (set as 0)	1
Third, lectures only give same course at different time (set as 0)	1

So, the objective function that can be used is (2) where W_i refers to weighting value the violation hard constraints and C_i^hard refers to hard constraints values.

$$\text{Objective Function} = \Sigma_{i=0}^{\text{total class}} W_i + C_i^{hard} \tag{1}$$

Therefore, fitness function can be used where it refers to a chromosome (solution candidate). Fitness values needs to be added by 1 to avoid program error that is caused by divided by 0.

$$\text{Fitness value (C)} = 1/(1 + \text{Objective Function}) \tag{2}$$

2.4. Genetic operator

Genetic operator that is used in EA is selection and stochastic variation operator. Stochastic variation operators can be divided into two classes, they are crossover operator and mutation operator that will be explained on the following subsections.

1. Selection operator

Selection is used to select best chromosome form the population according to their fitness. Some of the individuals in the current population that have lower fitness are chosen as elite. These elite individuals are passed to the next population. For the selection process we use tournament selections. It consists of the following steps:

For each elites size **do**
pop = elite
For pop < population size **do**
    While population size < tournament zise **do**
        Choose two chromosome Schedule from population
        Get chromosome Schedules fitness value
        Sort two chromosome Schedule fitness value and reverse
    end
  end
end.

2.5. Crossover operator

Crossover Operator is conducted after selection process. Crossover operator is used to recombine genetic information between two selected chromosomes and generate new offspring with predefined crossover rate (cr). For the election process of crossover, it consists of the following steps:

For eliteNumber in classes
  If random value [0,1] > cr
    Select chromosome Schedule 1 as new offspring, otherwise, select chromosome Schedule 2 as new offspring
  end If
end for

Value that is put to cr parameter is 0.5, so from one generation there is 50 percent chromosome that has crossover process. After process of that, the next step is making offsprings from the chosen chromosomes. The Chromosome with predefined cr less than and greater than random value will be chosen as new offsprings, so there is 50/50 offsprings in each crossover process. Table 3, 4, 5.

Table 3. Chromosome schedule 1

NIDN/Semester	1	2	3	...
1129039101	401	352	541	...
1129039102				...
1129039103	501			...
1129039104		300		...
1129039105	1	2	3	...

Table 4. Chromosome schedule 2

NIDN/Semester	1	2	3	...
1129039101	401	352	541	...
1129039102				...
1129039103	501			...
1129039104		300		...
1129039105		523	402	...

Table 5. New Offspring

NIDN/Semester	1	2	3	...
1129039101	401	352	541	...
1129039102				...
1129039103	501			...
1129039104		300		...
1129039105		523	402	...

1. Mutation operator

Our goal in using mutation operator is to add the quality of chromosome. Mutation is close to crossover process if there is no change in its chromosome. The number of chromosomes having mutation in one population is determined by mutation rate parameter (mr). Value that is put to mr parameter is 0.1 . Algorithm will generate random number, if the random number is smaller than mr value so the mutation process is conducted Table 6 and 7:

Table 6. Offspring before mutation

NIDN/Semester	1	2	3	...
1129039101	401	352	541	...
1129039102				...
1129039103	501			...
1129039104		300		...
1129039105		523	402	...

Table 7. Offspring after mutation

NIDN/Semester	1	2	3	...
1129039101	501	300	401	...
1129039102				...
1129039103	325			...
1129039104		532		...
1129039105		541	402	...

The following is the steps that we applied:

For mutation Schedule i = to numb_elitism, population_size
  If mr > random value [0,1]
    Select offspringMutation
  end If
end for

3. RESULT AND ANALYSIS

In order to demonstrate the efficiency and performance of the proposed algorithm, firstly the algorithm was applied to real-world input data coming from two study program in faculty of information technology UNISKA MAB, they are Engineering (TI) and Information System (SI). The data of a typical study program lecture schedule problem is presented in Table 8 and 9. The first column of Table 8 contains a list of the available lecturers, each having a different identification code (ID). In the given problem, there exist 28 course subjects. The required teaching hours per week for every lecturer (total hours column) and their breakdown to the various course subjects (each having a different identification code) is presented in the second set of columns, For example, S1 that shows the identification code of subject ALGO 1. The Days column presents the weekdays that each lecturer is available at faculty.

Table 8. The initial data of lecture schedule problem

ID.	Lecturers	Course subjects identification code (S)								Total hours	Days				
	NIDN	S1	S2	S3	S4	S5	S6	…	S28		Mo	Tu	We	Th	Fr
L1	1129039102	7.5						…		7.5	√	√	√	√	√
L2	1129039103		7.5					…		7.5	√	√	√	√	√
L3	1129039104			4.5				…		4.5	√	√	√	√	√
L4	1129039105					3	1.5	…		4.5	√	√	√	√	√
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
L74	1129039100			7.5				…		7.5	√	√	√	√	√

Table 9 contains the weekly lecture schedule for each course subjects and equivalent view of the lecture schedule for use by the lecturers The first column of Table 9 refers to course subject indentifcation code. The second and third columns show weekdays and hours of time periods and classrooms. For example, hours H1 that shows time priod 07.30 09.00. Time periods for typical faculty of Information technology at UNISKA MAB is shown in Table 10. The detail description of the data can be found in fti.uniska-bjm.ac.id/jadwal-perkuliahan-semester-ganjil-t-a-2020-2021.

Table 9. Time periods

ID	Days Lecturer	Monday								Tuesday								…
	NIDN	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	H11	H12	H13	H14	H15	H16	…
1	1129039101	S1	S1											S1	S1			…
2	1129039102	S2	S2			S2				S2				S2	S2			…
3	1129039103	S3	S3											S3	S3			…
4	1129039104	S4	S4											S4	S4			…
5	1129039105	S5	S5															…
6	1129039106			S6			S6			S6				S5	S5			…
7	1129039107			S6	S6													…
8	1129039108																	…
9	1129039009	S8																…
10	1129039110	S9						S9	S9									…

Table 10. Time periods

ID	Monday	ID	Tuesday	ID	Wednesday	ID	Thursday	ID	Friday
H1	07.30 09.00	H9	07.30 09.00	H17	07.30 09.00	H25	07.30 09.00	H33	08.00 09.30
H2	09.10 10.40	H10	09.10 10.40	H18	09.10 10.40	H26	09.10 10.40	H34	09.45 11.15
H3	10.50 12.20	H11	10.50 12.20	H19	10.50 12.20	H27	10.50 12.20	H35	14.00 15.30
H4	13.00 14.30	H12	13.00 14.30	H20	13.00 14.30	H28	13.00 14.30	H36	15.40 17.10
H5	14.40 16.10	H13	14.40 16.10	H21	14.40 16.10	H29	14.40 16.10	H37	19.15 20.30
H6	16.20 17.50	H14	16.20 17.50	H22	16.20 17.50	H30	16.20 17.50	H38	20.40 -22.05
H7	19.15 20.30	H15	19.15 20.30	H23	19.15 20.30	H31	19.15 20.30		
H8	20.40 22.05	H16	20.40 22.05	H24	20.40 22.05	H32	20.40 22.05		

The comparison of the lecture schedule constructed by proposed EA and the real lecture schedule used at faculty is based on the following two criteria:

1. Distribution of lectuers, that is, how evenly each lecturers hours are distributed among the days of the proposed lecture schedule. For example, if a lecturer giving a lesson 15 hours per week and is available at faculty for five days, the ideal distribution of lecturers would be (15/5) 3 hours of teaching per days.
2. Distribution of class rooms, It is how equal the distribution of class room during the courses. The ideal distribution of class rooms is only be used to one lecture in the same time.

Genetic operator parameter used in this experiments is presented in Table 11.

Table 11. Parameter genetic operator

Population Size	Numb.Of Elite	Tournament selection size	(cr)	(mr)
9	1	3	0.5	0.1

Comparing lecture schedule constructed by proposed EA with real-world lecture schedule used at faculty is presented in Table 12.

Table 12. Timetable comparison that is generated by EA with the timetable that is used in FTI

Number of lecturer	Number of available classroom	Teaching hours	Lecture schedule used at faculty		Lecture schedule created by EA	
			Distribution of lecturers	Classroom used	Distribution of lecturers	Classroom used
10	11	107	40(19)	5	29(8)	7

Regarding the distribution of teachers, two results (numbers) are presented in the respective column of Table 2. The first number is the number of teachers whose teaching hours do not have an even distribution among the days of the lecture schedule, while the second number (between the parentheses) is the number of days that this uneven distribution occurs. For example, from Table 10 with ID 2, if a lecturer is available at faculty for five days, has 15 teaching hours ($H1 + H2 + H13 + H14 + H16 + H18 + H19 + H20 + H22 + H33$ = 15 teaching hours) and their distribution among these five days is the following: $34.5601.5$, then the number of days in which uneven distribution for this lecturer occurs is four (4.5, 6, 0 and 1.5), the ideal distribution of lecturers is (: 33333) for five days or two course subjects teaching per days.

In Table 12 the performance and efficiency of the proposed EA is shown by comaparing the lecture schedule constructed by it with the real lecture schedule used at the faculty information technology in UNIKA MAB. These data can be found in https://fti.uniska-bjm.ac.id/jadwal-perkuliahan-semester-ganjil-t-a-2020-2021. We also analyze the convergence evolution of proposed EA. The EA increses sharply when generation number is between zero and a thousand, however after 10000 generations nearly the same fitness value is observed Figure 1.

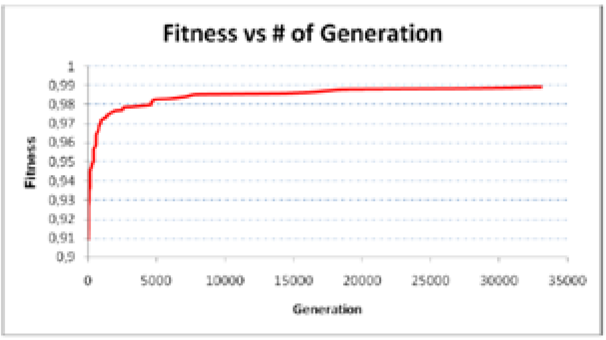

Figure 1. Fitness and number generation

4. CONCLUSION

Lecture schedule at University is a hard to solve problem, especially, in large universities. Constructing timetables is an important task that consumes time and effort of the involved personnel. In this study, we focus on the university lecture schedule problem whose solution can be obtained with a evolutionary algorithm. The problem is regarded as scheduling a set of courses into a set of time periods and a set of classrooms without violating hard constraints. It is be done by gaining the fitness function to solve the constraints problem in the schedule by applying weighting for each hard constraints. The objective function is gained from the total of infringement in each hard constraints addition by score weighting. Simulation results showed that the algorithm is able to construct a feasible and very efficient timetable quickly and easily. Both crossover and mutation operators are needed in order to prevent the search to be trapped in local minima. Higher probability values for these operators increase the efficiency of the algorithm. And also we realize that increasing the size of the population increases the convergence of the algorithm with respect to the generation number with the cost of high computation. Determination of the best parameter values, such as the population size, mutation rate, etc. It is also very hard to theoretically determine how the genetic operations affect the macroscopic behavior of genetic algorithms such as convergence of solutions. Thus, we try to determine sensitivity of the method to parameters experimentally.

5. ACKNOWLEDGEMENTS

We thank all the authors for their contribution.

6. DECLARATIONS

AUTHOR CONTIBUTION

Fisr Author conceptualized of this study, conducted experiments, wrote the original draft, and revised the manuscript. First Author wrote the manuscript and performed the experiments. AR made the experimental plan, supervised the work and revised the manuscript. Third Author performed the data analysis and reevised the manuscript. First Author made the experimental plan and revised the manuscript. Second Author evaluated the developed technique and revised the manuscript. Third Author designed the experimental plan, supervised the work andrevised the manuscript. All authors have read and agreed to the published version of the manuscript

FUNDING STATEMENT

This work was supported LP2M UNISKA MAB (APBU Ta.2021/2022).

COMPETING INTEREST The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest

REFERENCES

[1] A. Rezaeipanah, S. S. Matoori, and G. Ahmadi, “A Hybrid Algorithm for The University Course Timetabling Problem Using The Improved Parallel Genetic Algorithm and Local Search,” *Applied Intelligence*, vol. 51, no. 1, pp. 467–492, 2021.

[2] W. A. Algasm, "Hybrid Algorithm to Solve Timetabling Problem," *IOP Conference Series: Materials Science and Engineering*, vol. 928, no. 3, pp. 1–8, nov 2020. [Online]. Available: https://iopscience.iop.org/article/10.1088/1757-899X/928/3/032053

[3] M. Mazlan, M. Makhtar, A. F. K. Ahmad Khairi, and M. A. Mohamed, "University Course Timetabling Model Using Ant Colony Optimization Algorithm Approach," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 13, no. 1, pp. 72–76, 2019.

[4] S. Aslan and C. Aci, "Solving University Course Timetabling Problem Using Ant Colony Optimization : An Example of Mersin University Engineering Faculty," in *International Conference on Advanced Technologies, Computer Engineering and Science (ICATCES'18)*, vol. May, 2018, pp. 154–157.

[5] A. F. Khair, M. Makhtar, M. Mazlan, M. A. Mohamed, and M. N. A. Rahman, "An Ant Colony Algorithm for Universiti Sultan Zainal Abidin Examination Timetabling Problem," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 13, no. 1, pp. 191–198, 2019.

[6] Z. Qingfu and C. Fei, "Special Issue on The 17th Annual UK Workshop on Computational Intelligence," *Soft Computing*, vol. 22, no. 21, pp. 6965–6966, 2018.

[7] K. Zhu, L. D. Li, and M. Li, "A Survey of Computational Intelligence in Educational Timetabling," *International Journal of Machine Learning and Computing*, vol. 11, no. 1, pp. 40–47, 2021.

[8] M. H. Tayarani-N, "Novel Operators for Quantum Evolutionary Algorithm in Solving Timetabling Problem," *Evolutionary Intelligence*, vol. 14, no. 4, pp. 1869–1893, 2021.

[9] M. K. Kakkar, J. Singla, N. Garg, G. Gupta, P. Srivastava, and A. Kumar, "Class Schedule Generation Using Evolutionary Algorithms," *Journal of Physics: Conference Series*, vol. 1950, no. 1, 2021.

[10] I. A. Abduljabbar and S. M. Abdullah, "An Evolutionary Algorithm for Solving Academic Courses Timetable Scheduling Problem," *Baghdad Science Journal*, vol. 19, no. 2, pp. 399–408, 2022.

[11] A. O., I. E., and I. Kingsley, "A Tabu Search-Based University Lectures Timetable Scheduling Model," *International Journal of Computer Applications*, vol. 181, no. 9, pp. 16–23, 2018.

[12] M. Elliot, F. S. Gbenga, and M. E. J, "Enhanced Heuristic Teaching Timetabling Algorithm Using Genetic algorithm," *International Journal of Scientific & Technology Research*, vol. 9, no. 4, pp. 3804–3814, 2020.

[13] H. Hairani, A. Anggrawan, A. I. Wathan, K. A. Latif, K. Marzuki, and M. Zulfikri, "The Abstract of Thesis Classifier by Using Naive Bayes Method," in *2021 International Conference on Software Engineering & Computer Systems and 4th International Conference on Computational Science and Information Management (ICSECS-ICOCSIM)*, no. August. IEEE, aug 2021, pp. 312–315. [Online]. Available: https://ieeexplore.ieee.org/document/9537006/

[14] A. Slowik and H. Kwasnicka, "Evolutionary Algorithms and Their Applications to Engineering Problems," *Neural Computing and Applications*, vol. 32, no. 16, pp. 12 363–12 379, 2020.

[15] E. A. Abdelhalim and G. A. El Khayat, "A Utilization-Based Genetic Algorithm for Solving The University Timetabling Problem (UGA)," *Alexandria Engineering Journal*, vol. 55, no. 2, pp. 1395–1409, 2016.