Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control
Journal homepage: http://kinetik.umm.ac.id
ISSN: 2503-2267
Vol. 4, No. 4, November 2019, Pp. 297-304

Genetic algorithm for teaching distribution based on lecturers' expertise

Rizki Agung Pambudi[1], Wahyuni Lubis[2], Firhad Rinaldi Saputra[3], Hanif Prasetyo Maulidina[4], Vivi Nur Wijayaningrum[*5]
Universitas Brawijaya, Indonesia[1,2,3,4,5]

Article Info
Keywords:
Genetic Algorithm, Optimization, Scheduling, Teaching Distribution

Article history:
Received 4 July 2019
Revised 19 September 2019
Accepted 5 October 2019
Published 30 October 2019

Cite:
Pambudi, R., Lubis, W., Saputra, F., Maulidina, H., & Wijayaningrum, V. (2019). Genetic Algorithm for Teaching Distribution based on Lecturers' Expertise. Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control, 4(4).
doi:http://dx.doi.org/10.22219/kinetik.v4i4.859

*Corresponding author.
Vivi Nur Wijayaningrum
E-mail address:
vivinurw@gmail.com

Abstract

The teaching distribution for lecturers based on their expertise is very important in the teaching and learning process. Lecturers who teach a course that is in accordance with their interests and abilities will make it easier for them to deliver material in class. In addition, students will also be easier to accept the material presented. However, in reality, the teaching distribution is often not in accordance with the expertise of the lecturer so that the lecturers are not optimal in providing material to their students. This problem can be solved using optimization methods such as the genetic algorithm. This study offers a solution for teaching distribution that focuses on the interest of each lecturer by considering the order of priorities. The optimal parameters of the test results are crossover rate (cr) = 0.6, mutation rate (mr) = 0.4, number of generations = 40, and population size = 15. Genetic algorithm is proven to be able to produce teaching distribution solutions with a relatively high fitness value at 4903.3.

1. Introduction

Every educational institution such as a college certainly has a goal to improve the quality and academic abilities of its students. This can be realized by forming a conducive and enjoyable learning environment for students and lecturers in the class. The ability of students to receive material delivered by lecturers in class is not only influenced by the readiness of the students themselves but also influenced by the suitability of the teaching style of the lecturer [1]. This teaching style includes knowledge, self-confidence, performance, and lecturer behavior during the teaching process in the classroom [2]. The ability and expertise of lecturers when teaching in class has a strong influence on student learning outcomes [3]. Without good competence, of course, it will be difficult for lecturers to produce a good performance in the learning process.

In some universities, it is often found that some lecturers do not teach according to their interests, for example, a lecturer who has an interest in the field of intelligent systems is assigned to teach courses in the field of networking. This is usually due to a large number of lecturers accompanied by the number of courses that must be taught by lecturers so that the teaching distribution becomes less in line with the interests of lecturers. The incompatibility of the courses that must be taught by a lecturer with the lecturers' interest will certainly result in the lecturers' performance in teaching becoming less than optimal. As a result, students find it difficult to accept material in class and their academic abilities are not maximal.

Completion of the problem of teaching distribution is very time consuming. In addition, the number of lectures and courses makes this problem more complicated and complex [4]. Some researchers have used various types of methods to solve scheduling problems, one of which is to use integer programming [5]. The study has three stages, the first stage is to set all lecture schedules on a particular day to comply with the rules and limits set in the schedule. Furthermore, in the second stage, a schedule for each day is carried out, namely determining the time allocation according to the day set in the first stage. At this stage, it is necessary to ensure that every student and lecturer does not attend more than one lecture at any given time. At the last stage, each schedule that has been formed is distributed to several types of classes so that it fits the number of students and the needs of their courses. However, using integer programming to solve scheduling problems is considered quite difficult because of the need to consider all goals and boundaries at one time [6].

In other studies, the graph coloring algorithm is used to complete course timetable scheduling [7]. The problem is formulated in the form of a graph with a course as a node. Edge is drawn according to the type of graph created. At conflicting graph, edges are drawn between conflicting courses that have the same students. Whereas on non-conflicting graphs, edges are drawn between mutually exclusive courses that do not have the same students. Conflict graph acts as an input from the graph coloring algorithm, while the resulting output is the minimum number of non-conflicting time slots needed for scheduling courses.

Meta-heuristic algorithms are currently widely used to solve various NP-hard problems such as scheduling problems. Harmony Search was successfully used to solve the problem of scheduling courses at the university by representing problems into a matrix where each matrix element shows the lecture room and time slot for the course [8]. Another meta-heuristic algorithm, Ant Colony Optimization, is also used for the formation of lecture schedules [9]. The problem is represented in the form of construction graph, where each node and edge describe the time and place for lectures. Furthermore, the Ant Colony Optimization algorithm will search for optimal solutions.

The genetic algorithm as one of the meta-heuristic algorithms can also be used to solve combinatorial problems effectively such as determining the shortest path for navigation systems [10] and determination of river water quality [11]. The genetic algorithm can provide several optimal solutions, produce good solutions for various complex problems, and can work well for analytical functions and numerical data [12][13]. Therefore, this study uses the genetic algorithm to solve the problem of teaching distribution because the genetic algorithm is simple and easy to use [14]. This study focuses on the prevalence of teaching distribution in accordance with the field of interest of each lecturer. The solution offered is the formation of teaching distribution by minimizing the existence of discrepancies in the field of lecturer interest with the courses that must be taught.

2. Research Method

The teaching distribution can be considered an assignment problem. The assignment problem is the arrangement of objects or individuals to perform certain tasks that aim to minimize the cost of doing the task. In short, assignments make the right person do the right job [15]. In this study, lecturers represented workers and courses represented assignments. Thus, each lecturer will be assigned to teach in certain courses according to his/her interests and abilities. Table 1 shows the course tables provided along with the number of opened classes in this semester.

Table 1. Course Details

Course Code	Course Name	Credit	Number of Classes
1	Web Programming	4	3
2	System Analysis and Design	5	4
3	Computer Networks	4	3
4	Human-Computer Interaction	3	2
5	Design and Analysis of Algorithms	3	2
6	Basic Programming	5	4
7	Advanced Programming	5	4
8	Pattern Recognition	3	2
9	Calculus	4	3
10	Computer Architecture and Organization	3	3

In Table 1, there are 10 courses that have different classes. The number of classes in each subject shows the number of classes provided for the course, so teaching distribution refers to this number of classes. Each of these courses also has a course code and credit. Furthermore, the field of interest of each lecturer is shown in Table 2.

Table 2. Lecturer Expertise

Lecturer Code	Course Code	
	Major Expertise	Minor Expertise
1	1, 5	4
2	2, 5, 3	10
3	3, 1	7, 9
4	2, 4	8
5	5, 4	10, 2
6	8, 7	3
7	2, 5, 9	3
8	3, 6, 10	5
9	2, 6	3, 4
10	1, 7, 3	5

Then, in Table 2 there is a list of lecturers who will teach courses. Each lecturer has major and minor interests. For example, a lecturer with code 1 has a major interest consisting of course code 1 and course code 5 (Web Programming + Design and Analysis of Algorithms) and minor interest in course code 4 (Human-Computer Interaction). In this case, course code 1 and course code 5 have different priority levels. Course code 1 has a higher priority level than course code 5, so to determine the teaching distribution of lecturers, course code 1 should be prioritized for lecturer with code 1.

2.1 Encoding

The first step in solving a problem using the genetic algorithm is to translate the problem into biological terminology. This is done by forming a chromosome that describes the solution to the problem, called encoding. The performance of the genetic algorithm will be greatly influenced by the type of chromosome representation used during the solution search process [16].

In this study, the type of encoding used is an integer representation. A chromosome consists of several genes containing lecturer code, which are integer numbers 1 to 10 according to the number of lecturers in Table 2. There are 10 courses as shown in Table 1, with 30 classes provided in total, so the number of genes for each chromosome is 30. The chromosome example is illustrated in Figure 1.

7	7	3	5	10	4	6	3	2	9	8	8	10	6	6	5	8	6	2	1	4	5	1	2	4	8	3	9	7	10
Course 1			Course 2				Course 3			Course 4		Course 5		Course 6				Course 7				Course 8		Course 9			Course 10		

Figure 1. Example of Chromosome Representation Using Integer

In Figure 1, there are 10 courses, each consisting of 2-4 classes. Each class is represented by a gene containing the lecturer code. For example, course 1 provides 3 classes that will be taught by lecturers with code 7 (2 classes) and code 3 (1 class).

2.2 Fitness Function

A chromosome that represents a solution to a problem is measured for its quality using a fitness function. The better the solution is given by a chromosome, the greater the fitness value [17]. Calculation of fitness function considers the interest of lecturers in certain subjects. In Table 2, each lecturer has a major and minor interest in the courses. In each major/minor interest there is also a priority sequence of the desired courses.

Based on these data, the priority weight tables shown in Table 3 are formed by considering the priorities, major expertise, and minor expertise of lecturers in certain courses.

Table 3. Priority Weight of the Courses

Expertise	Priority				
	1st	2nd	3rd	4th	Not a Priority
Major	1	3	5	7	100
Minor	2	4	6	8	100

Table 3 contains priority weights for courses assigned to lecturers. If the course assigned to the lecturer is a major interest course and is in the first place, the priority weight is 1. If the courses assigned to the lecturer is a minor interest course and is in the first place, then the priority weight is 2. Then, if the courses assigned to the lecturer is not a major and minor interest course, then the priority weight is 100. These values are then used to form the cost matrix as shown in Table 4.

Table 4. Cost Matrix

Lecture Code	Course Code									
	1	2	3	4	5	6	7	8	9	10
1	1	100	100	2	3	100	100	100	100	100
2	100	1	5	100	3	100	100	100	100	2
3	3	100	1	100	100	100	2	100	4	100
4	100	1	100	3	100	100	100	2	100	100
5	100	4	100	3	1	100	100	100	100	2
6	100	100	2	100	100	100	3	1	100	100
7	100	1	2	100	3	100	100	100	5	100
8	100	100	1	100	2	3	100	100	100	5
9	100	1	2	4	100	3	100	100	100	100
10	1	100	5	100	2	100	3	100	100	100

The values in Table 4 are adjusted to the data of lecturer interest in Table 2 and priority weights in Table 3. For example, Table 2 shows that lecturer code 1 has a major interest in code 1 (the first order), major code 5 (the second order), and minor code 4 (the first order). Thus, according to priority weights in Table 3, course code 1 is weighted 1, course code 5 is weighted 3, and course code 4 is weighted 2, while the other course codes are filled with 100 because they are not is on the list of lecturer interest.

By using the cost matrix in Table 4, the fitness value can be calculated using Equation 1, with C = 5000 and n is the number of courses. Examples of cost calculation for chromosome in Figure 1 is shown in Table 5.

$$\text{Fitness} = C - \sum_{i=1}^{n} \text{cost} \quad (1)$$

Based on the results of the cost calculation in Table 5, the fitness value for the chromosomes in Figure 1 can be calculated using Equation 1, so that the fitness value is 2875, which is obtained from the reduction of the constant value C = 5000 with a total cost of 2125.

Table 5. Example of Calculating the Cost

Lecture Code	Assigned Courses	
	Course Code	Cost
1	7	100
	8	100
2	3	5
	7	100
	8	100
3	1	3
	3	1
	9	4
4	2	1
	7	100
	9	100
5	2	4
	6	100
	7	100
6	2	100
	5	100
	6	100
	6	100
7	1	100
	1	100
	10	100
8	4	100
	4	100
	6	3
	9	100
9	3	2
	10	100
10	2	100
	5	2
	10	100
Total Cost		2125

2.3 Crossover

The crossover process aims to produce offspring from two selected chromosomes. The offspring chromosome formed is a combination of the two parent chromosome genes. The crossover method used in this study is a one-cut point crossover. The one-cut point crossover method is a simple crossover method and is often used on various problems [18]. The crossover mechanism starts from randomly selecting two parent chromosomes in the population. A cut point will also be selected on the first parent chromosome. The offspring chromosome will have the same gene as the first parent, that is, from the first gene to the cut point. While the rest of the child's chromosome genes will be filled by genes from the second parent [19]. Examples of crossover processes using a one-cut point method are shown in Figure 2.

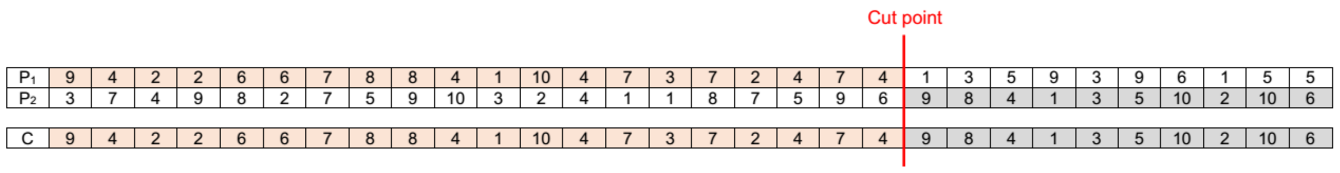

Figure 2. Example of the Crossover Using a One-Cut Point Method

2.4 Mutation

The process of mutation is done by replacing one or several genes on a chromosome. The purpose of this replacement is to increase the chromosome variation in the population so that the search results avoid early convergence. With the existence of mutations, the genetic algorithm is expected to be able to carry out the search process in other areas that have not been reached [20]. The type of mutation used in this study is swap mutation. The mechanism of mutation begins by selecting a parent chromosome and two exchange points. The offspring chromosome is produced by the exchange of these two genes [21]. Examples of mutation processes using the swap mutation method are shown in Figure 3.

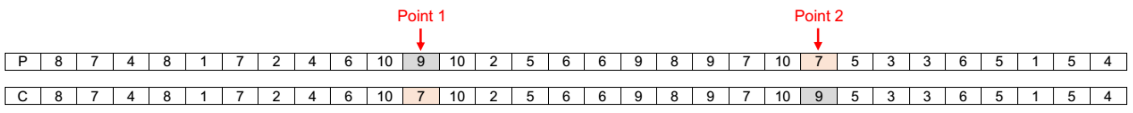

Figure 3. Example of the Mutation Using Swap Mutation

2.5 Selection

The selection process has the responsibility to choose chromosomes that will survive in the population of the next generation. The purpose of selection is to provide opportunities for the chromosomes in the population that have the best fitness. The selection method used in this study is elitism selection, which works by sequencing chromosomes based on their fitness values from the highest to lowest fitness values [20]. The selected chromosomes will survive and can be processed for the next generation.

3. Results and Discussion

There are three types of tests carried out in this study. The three types of testing are used to evaluate the best parameters of the genetic algorithm to get the optimal solution. Each test is carried out using several different values and tested 10 times, and then the average value is calculated in order to get the optimal value.

3.1 Testing of Crossover Rate and Mutation Rate

The aim of testing the crossover rate (cr) and mutation rate (mr) is to find the combination of cr and mr which produces the best average fitness value. The combination of cr and mr in this study was determined by the condition cr + mr = 1 [22]. There are nine combinations of cr and mr values that are tested. The constant parameter value used is the number of generations = 100 and population size = 50. The test results are shown in Table 6.

Table 6. Test Results of Crossover Rate and Mutation Rate

cr	mr	Trial Number										Average Fitness
		1	2	3	4	5	6	7	8	9	10	
0.9	0.1	4412	4591	4609	4596	4491	4404	4492	4427	4594	4523	4513.9
0.8	0.2	4602	4593	4611	4671	4602	4603	4596	4516	4690	4512	4599.6
0.7	0.3	4787	4709	4705	4689	4685	4691	4795	4705	4705	4794	4726.5
0.6	0.4	4768	4796	4793	4784	4803	4794	4789	4800	4791	4798	4791.6
0.5	0.5	4787	4805	4793	4895	4792	4802	4895	4882	4799	4794	4824.4
0.4	0.6	4892	4902	4807	4801	4884	4802	4801	4876	4879	4803	4844.7
0.3	0.7	4894	4889	4890	4891	4869	4899	4894	4891	4896	4888	4890.1
0.2	0.8	4896	4894	4894	4913	4894	4905	4907	4901	4903	4900	4900.7
0.1	0.9	4900	4895	4893	4894	4899	4818	4885	4807	4895	4895	4878.1

Furthermore, the test results in Table 6 are formed into a graph as shown in Figure 4.

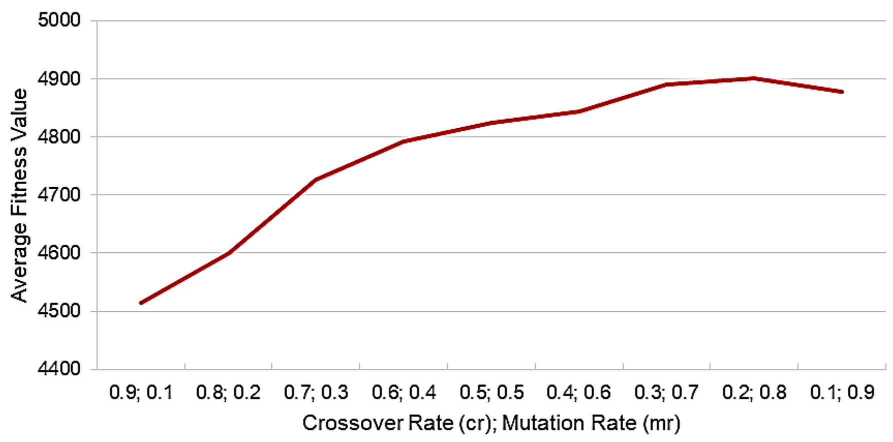

Figure 4. The Test Result Graph of the Value of Crossover Rate and Mutation Rate

Based on the graph in Figure 4, the highest average fitness value is shown on the combination of cr = 0.2 and mr = 0.8. Thus, both of these values are used as optimal parameters for the combination of cr and mr. This shows that the mutation process made a major contribution to get a better solution. However, the value of mr that is too large can cause the solution to be poor because the genetic algorithm will look for solutions such as the random search method [23].

3.2 Testing of the Number of Generations

The purpose of testing the number of generations is to find the number of generations that produce the best average fitness value. The tested value is a multiple of 25 from 25 to 200. The constant parameter value used is population size = 50, and the combination of optimal cr and mr from the results of previous tests, namely cr = 0.2 and mr = 0.8. The test results are shown in Table 7.

Table 7. Test Results of the Number of Generations

Generation	Trial Number										Average
Number	1	2	3	4	5	6	7	8	9	10	Fitness
25	4389	4223	4393	4302	4225	4225	4216	4313	4226	4210	4272.2
50	4684	4697	4619	4706	4592	4625	4702	4612	4595	4692	4652.4
75	4815	4801	4800	4793	4802	4787	4794	4801	4801	4800	4799.4
100	4890	4895	4894	4898	4877	4894	4898	4892	4898	4895	4893.1
125	4900	4897	4893	4912	4900	4901	4896	4909	4913	4911	4903.2
150	4907	4900	4908	4907	4908	4905	4903	4908	4900	4899	4904.5
175	4912	4903	4901	4910	4897	4913	4914	4905	4913	4903	4907.1
200	4906	4912	4902	4910	4905	4908	4912	4898	4908	4906	4906.7

Furthermore, the test results in Table 7 are formed into a graph as shown in Figure 5.

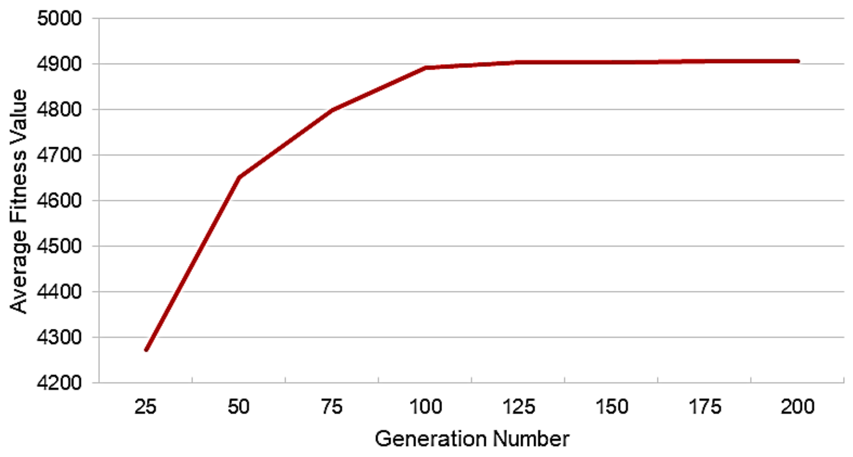

Figure 5. The Test Result Graph of Generation Number

Based on the graph in Figure 5, the average fitness value increases from generation 25 to 100, and then the average fitness value tends to be stable when the generation reaches 125. This shows that 125 is the optimal parameter value for the number of generations.

3.3 Testing of Population Size

The purpose of testing population size is to find the population size that produces the best average fitness value. The value tested is a multiple of 10 from 10 to 70. The value of the constant parameter used is the optimal parameter from the results of previous tests, namely the number of generations = 125, cr = 0.2, and mr = 0.8. The test results are shown in Table 8.

Table 8. Test Results of Population Size

Population	Trial Number										Average
Size	1	2	3	4	5	6	7	8	9	10	Fitness
10	4794	4797	4804	4801	4799	4804	4699	4790	4805	4728	4782.1
20	4805	4803	4807	4812	4809	4792	4808	4803	4806	4804	4804.9
30	4882	4896	4889	4902	4899	4807	4816	4807	4898	4875	4867.1
40	4899	4897	4894	4899	4891	4892	4895	4896	4898	4900	4896.1
50	4905	4903	4904	4910	4898	4901	4904	4899	4905	4904	4903.3
60	4911	4907	4900	4902	4906	4911	4903	4913	4908	4915	4907.6
70	4910	4909	4905	4904	4920	4906	4909	4903	4902	4906	4907.4

Furthermore, the test results in Table 8 are formed into a graph as shown in Figure 6.

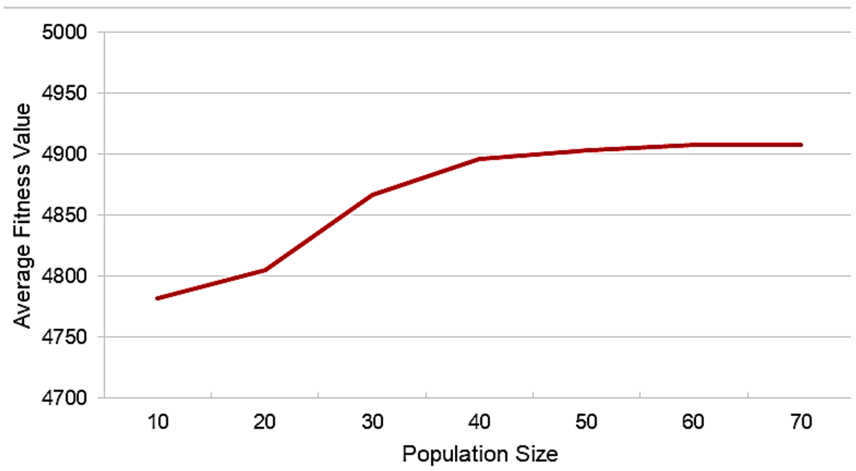

Figure 6. The Test Result Graph of Population Size

Based on the graph in Figure 6, the average fitness value increases from population size 10 to 40 and then tends to be stable when the population size is 50. This shows that 50 is the optimal parameter value for population size.

By using the parameters that have been obtained from the results of these tests, the algorithm can provide optimal results, which means that the solutions provided are good enough without requiring much computational time. Figure 7 shows the solution provided by the system when the algorithm is run using the optimal parameters.

1	1	10	2	7	9	2	3	8	8	4	4	1	5	9	9	9	9	10	10	6	6	6	6	7	7	7	8	5	5
Course 1			Course 2				Course 3			Course 4		Course 5		Course 6				Course 7				Course 8		Course 9			Course 10		

Figure 7. Example Solution Using Optimal Parameters

In Figure 7, it can be seen that all lecturers get courses according to their interests. Lecturer with code 1 teaches course 1 (priority 1) and course 5 (priority 3). Lecturer with code 2 teaches course 2 (priority 1). Lecturer with code 3 teaches course 3 (priority 1). Lecturer with code 4 teaches course 4 (priority 3). Lecturer with code 5 teaches course 5 (priority 1) and course 10 (priority 2). Lecturer with code 6 teaches course 7 (priority 3) and course 8 (priority 1). Lecturer with code 7 teaches course 2 (priority 1) and course 9 (priority 5). Lecturer with code 8 teaches course 3 (priority 1). Lecturer with code 9 teaches course 2 (priority 1) and course 6 (priority 3). Lecturer with code 10 teaches course 1 (priority 1) and course 7 (priority 3). Thus, there are no lecturers who teach courses that are not in accordance with their interests. In addition, based on Table 3, it is known that most lecturers get major interest courses.

4. Conclusion

Based on the results of testing and analysis that has been done, it can be concluded that the genetic algorithm can be used to solve the problem of teaching distribution. The parameters used to produce the optimal solution consist of crossover rate (cr) = 0.6, mutation rate (mr) = 0.4, number of generations = 40, and population size = 15. Using these optimal parameters, the genetic algorithm is able to provide very good solutions that are the composition of the teaching distribution in accordance with the interest of each lecturer. This is evidenced by the high average fitness value given.

The next study is done by modifying the fitness function, so that the fitness value is not only calculated based on the priority of the courses chosen by the lecturer, but also by considering the group of courses according to their fields, such as networking, software engineering, intelligent systems, and others. In addition, the number of classes obtained by lecturers is also calculated so as to maintain the balance of teaching distribution. To improve the solutions provided by genetic algorithm, a mechanism for hybridizing genetic algorithm with other algorithms can also be done.

References

[1] R. M. Felder and E. R. Henriques, “Learning and Teaching Styles In Foreign and Second Language Education,” *Foreign Lang. Ann.*, Vol. 28, No. 1, Pp. 21–31, 1995. https://doi.org/10.1111/j.1944-9720.1995.tb00767.x

[2] A. F. Grasha, "Teaching With Style: A Practical Guide to Enhancing Learning by Understanding Teaching and Learning Styles," *Pittsburgh: Alliance Publishers,* 1996.

[3] M. A. Mawoli and A. Y. Babandako, “An Evaluation of Staff Motivation, Dissatisfaction and Job Performance in an Academic Setting,” *Aust. J. Bus. Manag. Res.*, Vol. 1, No. 9, Pp. 1–13, 2011.

[4] A. R. Komijan and M. N. Koupaei, “A Mathematical Model for University Course Scheduling: A Case Study,” *Int. J. Tech. Res. Appl.*, No. 19, Pp. 20–25, 2017.

[5] N. A. H. Aizam and C. Y. Liong, “Mathematical Modelling of University Timetabling : A Mathematical Programming Approach,” *Int. J. Appl. Math. Stat.*, Vol. 37, No. 7, Pp. 110–122, 2013.

[6] L. Urbanucci, “Limits and Potentials of Mixed Integer Linear Programming Methods for Optimization of Polygeneration Energy Systems,” *Energy Procedia*, Vol. 148, Pp. 1199–1205, 2018. https://doi.org/10.1016/j.egypro.2018.08.021

[7] R. Ganguli and S. Roy, “A Study on Course Timetable Scheduling Using Graph Coloring Approach,” *Int. J. Comput. Appl. Math.*, Vol. 12, No. 2, Pp. 469–485, 2017.

[8] M. A. Al-Betar and A. T. Khader, “A Harmony Search Algorithm for University Course Timetabling,” *Ann. Oper. Res.*, Vol. 194, No. 1, Pp. 3–31, 2012. https://doi.org/10.1007/s10479-010-0769-z

[9] P. Gore, P. Sonawane, and S. Potdar, “Timetable Generation Using Ant Colony Optimization Algorithm,” *Int. J. Innov. Res. Comput. Commun. Eng.*, Vol. 5, No. 3, Pp. 6033–6039, 2017.

[10] A. N. Syahputra, A. S. Kholimi, and L. Husniah, “Genetic Algorithm Application for Navigation System on Virtual Shop,” *KINETIK*, Vol. 3, No. 4, Pp. 311–318, 2018. http://dx.doi.org/10.22219/kinetik.v3i4.668

[11] Q. Kotimah, W. F. Mahmudy, and V. N. Wijayaningrum, “Optimization of Fuzzy Tsukamoto Membership Function Using Genetic Algorithm to Determine the River Water,” *Int. J. Electr. Comput. Eng.*, Vol. 7, No. 5, Pp. 2838–2846, 2017. http://doi.org/10.11591/ijece.v7i5.pp2838-2846

[12] X.-S. Yang, "Nature-Inspired Metaheuristic Algorithms," *Second Edi. Luniver Press,* 2010.

[13] V. N. Wijayaningrum and W. F. Mahmudy, “Fodder composition optimization using modified genetic algorithm,” *Indones. J. Electr. Eng. Informatics*, Vol. 7, No. 1, Pp. 67–74, 2019. http://dx.doi.org/10.11591/ijeei.v7i1.461

[14] S. Malik and S. Wadhwa, “Preventing Premature Convergence in Genetic Algorithm Using DGCA and Elitist Technique,” *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, Vol. 4, No. 6, Pp. 410–418, 2014.

[15] H. A. Taha, "Operations Research: An Introduction," *8th Edi. New Jersey: Prentice Hall*, 2007.

[16] A. Bedboudi, C. Bouras, and M. T. Kimour, “An Heterogeneous Population-Based Genetic Algorithm for Data Clustering,” *Indones. J. Electr. Eng. Informatics*, Vol. 5, No. 3, Pp. 275–284, 2017. http://dx.doi.org/10.11591/ijeei.v5i3.299

[17] V. N. Wijayaningrum, W. F. Mahmudy, and M. H. Natsir, “Optimization of Poultry Feed Composition using Hybrid Adaptive Genetic Algorithm and Simulated Annealing,” *J. Telecommun. Electron. Comput. Eng.*, Vol. 9, No. 2–8, Pp. 183–187, 2017.

[18] J. Magalhães-Mendes, “A Comparative Study of Crossover Operators for Genetic Algorithms to Solve the Job Shop Scheduling Problem,” *WSEAS Trans. Comput.*, Vol. 12, No. 4, Pp. 164–173, 2013.

[19] M. Mitchell, "An Introduction to Genetic Algorithms. Cambridge," *USA: MIT Press,* 1998.

[20] R. L. Haupt and S. E. Haupt, "Practical Genetic Algorithms," *New York, USA: John Willey & Sons, Inc.,* 2004.

[21] A. E. Eiben and J. E. Smith, "Introduction to Evolutionary Computing," *Second Edi. Springer,* 2015.

[22] V. N. Wijayaningrum and W. F. Mahmudy, “Optimization of Ship’s Route Scheduling Using Genetic Algorithm,” *Indones. J. Electr. Eng. Comput. Sci.*, Vol. 2, No. 1, Pp. 180–186, 2016. http://doi.org/10.11591/ijeecs.v2.i1.pp180-186

[23] W. F. Mahmudy, R. M. Marian, and L. H. S. Luong, “Real Coded Genetic Algorithms for Solving Flexible Job-Shop Scheduling Problem - Part II: Optimization,” *Adv. Mater. Res.*, Vol. 701, Pp. 364–369, 2013. http://dx.doi.org/10.4028/www.scientific.net/AMR.701.364