JOURNAL OF NATURAL SCIENCES AND MATHEMATICS RESEARCH

Available online at http://journal.walisongo.ac.id/index.php/jnsmr

Genetic Algorithm for Solving 3 Dimensional Time Table Problem Based on Traveling Salesman Problem (TSP) Method

Dwi Fatul Oktafiani[1], **Muhammad Ardhi Khalif**[1], **Hesti Khuzaimah Nurul Yusufiyah**[1]
[1]Department of Physics, Faculty of Science and Technology, Universitas Islam Walisongo Semarang, Central Java, Indonesia

Corresponding author: coesma@yahoo.com
Received: 23 Januari 2018,
Revised : 25 Maret 2018,
Accepted : 01 Juni 2018.

Abstracts

Scheduling problems are problems that are often faced by educational institutions, especially at the university level. This is because there are several obstacles in the preparation of the schedule, namely first, there should be no duplication of space, day, and hour. Second, there should be no duplication of lecturers on the same day and time, even though in different rooms and in different subjects. Third, there should be no duplication of group classes (study groups). Therefore, the purpose of this study is to obtain a genetic algorithm as a solution in overcoming the three constraints of preparing the schedule by using the Traveling Salesman Problem (TSP) method in the crossover process. To make it easier to organize the schedule, a 3-dimensional matrix is used with the x-axis representing (space, day, hour), the y-axis representing (courses, lecturers, credits) and the z-axis representing (classes). This study simulates the scheduling of 20 courses, 50 credits, 8 lecturers, and 19 classes. Chromosomes in this study are permutations of integers 1-20. Each gene in a chromosome represents a course package. From the scheduling results, the fitness function is 0.96 for 48 schedule slots (2 rooms x 3 days x 8 hours). For schedule slots greater than 50 (3 rooms x 3 days x 8 hours, 2 rooms x 4 days x 8 hours, and 2 rooms x 3 days x 9 hours), this algorithm is successful in getting fitness function 1.

Key words: Genetic Algorithm; 3D *Time Table Problem*; *Traveling Salesman Problem*; *Fitness Function*.

1. Introduction

Planning and formulating a course scheduling strategy is an activity that is routine for the university in every semester. Aspects that need to be considered regarding course scheduling include the name of the lecturer, course, credit load, teaching hours, days and classrooms provided in each faculty. This manual arrangement and arrangement by the academic side causes the final results in the scheduling of these courses to often lead to duplication of space, lecturers, and classes. This resulted in the delay in the input process for the Study Plan Card (KRS).

Therefore, the Pure Physics Study Program at the Faculty of Science and Technology, is trying to make a solution regarding the system using computerized assistance. The scheduling problem will be solved by using a genetic algorithm. Genetic algorithm is classified as an optimization method so that it aims to get the maximum or minimum value of a certain function.

Genetic algorithm, using the mechanism of natural selection and genetic science. In genetics, chromosomes consist of the arrangement of genes. This natural selection process involves gene changes that occur in individuals through the process of reproduction. In genetic algorithms, this breeding process becomes the basic process that is the main concern, with the basic thinking: "How to get better offspring" [1]. The genetic algorithm was first developed by John Holland of the University of Michigan (1975), John Holland said that every problem in the form of adaptation (natural or artificial) can be formulated in genetic terms. The genetic algorithm is a simulation of the Darwinian evolutionary process and genetic operations on chromosomes [2].

Research on scheduling optimization using certain methods has been carried out by several researchers. Lecture scheduling system with genetic algorithm. The advantage of the system is that the result achieved with the best chromosome is worth one [3]. Kumala explained about how to apply genetic algorithms in the production optimization process, and in the results it was found that genetic algorithms are a good method in the optimization process [8]. In the application results can produce output in accordance with rigid limits. Jenna Car, which still uses 2 dimensions in its research [4]. In this study, Jenna Car considers that the room and class are the same, where the simulation is not appropriate with the problem to be studied. From these shortcomings, researchers use genetic algorithms to solve 3-dimensional problems which are expected to be more effective and accurate. It is hoped that this method can be used to solve problems regarding course scheduling at the Walisongo State Islamic University, Semarang.

2. Experiments Procedure

Genetic Algorithm

Genetic algorithm is an adaptive method used to solve a value search in an optimization problem. Genetic Algorithms use a direct analogy of natural behavior known as natural selection. This algorithm works on an initial population consisting of individuals, each of which represents a possible solution to the existing problem. Individuals are represented by a fitness value that will be used to find the best solution.

Some things that must be done in the Genetic Algorithm are [6]:

a) Define the individual, where the individual states one possible solution (solution) of the problems raised.
b) Define the fitness value, which is a measure of whether or not an individual or solution is obtained.
c) Determine the initial population generation process. This is usually done using random generation such as random – walk.
d) Determine the selection process to be used.
e) Determine the process of crossover and gene mutation to be used.

Method of Traveling Salesman Problem

In this problem, we assume the salesman travels by visiting n cities via the shortest possible route. He visited each city exactly once, then returned to the city where he started. Therefore a candidate solution would be to list all the cities in the order he visited them [5]. We denote cities such as city 1, city 2, city 3, . . . , city n ; where city 1 is the starting point. Thus the chromosomes for the genetic algorithm permutation differ from the integers 1 through n. There are many variations of the traveling salesman problem, we assume that the distance d from city ci to city cj is d (ci ; c j) = d (c j ; c i) for all i 2 [1 ; n] and j 2 [1 ; n]. We can see how in the real world this would not be the case, the highway in one direction may be a little more narrow than when we go directly in the opposite direction.

The case is called the "unsymmetrical" traveling salesman problem [9]. To modify the crossing to accommodate permutations, which is discussed in Goldberg [6] and Haupt [5]. First we choose to create offspring. Unless swapping integers to equal value, each one random location of the chromosome. The two parent integers are exchanged for the offspring to have one duplicated integer. The offspring are not the same in each integer, then it is a valid permutation. The mutation is modified to run randomly by selecting two integers on one chromosome from the new generation and swapping them.

In this study, the researchers assumed several things, namely:

1) The population in this study was 20 chromosomes that were randomly generated. One chromosome contains 20 serial number of course packages, so it is a permutation of integers 1-20.
2) Fitness function is a comparison of the number of credits that have been successfully scheduled with the number of credits that must be scheduled.
3) The selection rule will select 10 chromosomes with the highest fitness function for crossover.
4) Crossover is done between the 10 best chromosomes to produce 10 new chromosomes. The crossover method used is the same as the crossover method used in the Traveling Salesman Problem.
5) Mutation is done by swapping 2 genes at random on a randomly selected chromosome from the population.

The preparation of the course schedule is carried out using a 3-dimensional matrix:

- Axis x = (space, day, hour)
- y axis = (courses, lecturers, credits)
- z axis = (class)

The amount of data :

- Axis x = nSpace x nDays x nHours
- y axis = nCourse
- z axis = nClass

For this research sample using:

- nCourse = 20
- nSKS = 50
- nLecturer = 8
- nClass = 19

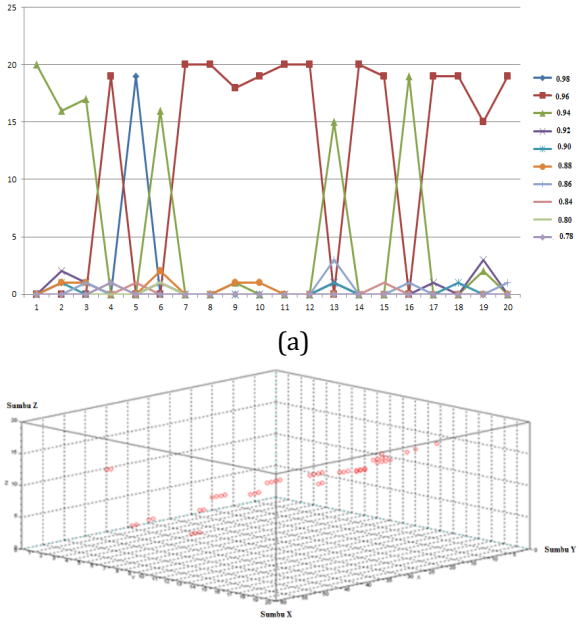
3. Result and Discussion

This research is done by varying the number of rooms, days and hours to see the ability of the algorithm in scheduling course packages according to the sample.

1. *NTotal = 48 consisting of nroom = 2 , nday = 3 , nhour = 8.*

The results of this first study get a fitness function value of 0.96 where the value is obtained from (48/50) or a comparison of scheduled credits with credits that must be scheduled so that the results are not able to reach the fitness function value 1. Although the fitness function value does not reach 1, the algorithm This has been able to organize the schedule according to the needs without any duplication. If the results of the fitness function reach 1, it means that in this case there is a

duplication, because Ntotal < nSKS so it is not possible for the fitness function to get a value of 1. In this study, 20 (Running) were carried out to organize the schedule and the results obtained were an increase in the fitness function where the fitness value function 0.96 the longer the number of genes the more. Evidenced by the research chart below.



(a)

(b)

Figure 1. The result of (a) Graphs, and (b) Matrix 3 Dimensions Sample 1

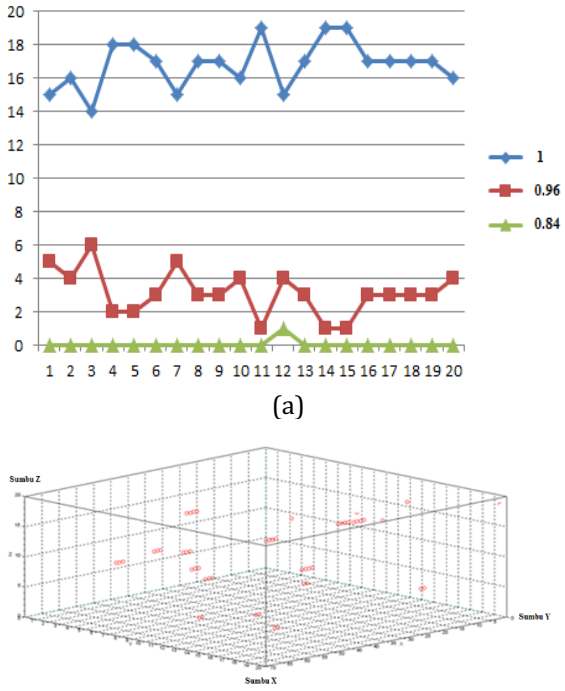
This graph uses the x-axis (running) and the y-axis (fitness function value). It can be seen that every time it runs it will produce 20 courses. For example, for the 1st run there are (0.94) as many as 20. Meanwhile for the 19th running the fitness function value is 0.96 as many as 15 chromosomes, 0.92 as many as 3 chromosomes and 0.94 as many as 2 chromosomes.

From the graph above, it can also be seen that the fitness function value of 0.96 experienced an increase in the number of chromosomes in the arrangement of the course schedule. This sample requires a TSP crossover and permutation process because the fitness function results < 1.

The results of the 3-dimensional matrix show that there is no duplication. There are red circles that show the credit load. So that in this sample the algorithm created has been able to schedule courses according to needs.

2. *NTotal = 64 which consists of nroom = 2 , nday = 4 , nhour = 8*

The results showed that the value of the fitness function had reached 1 with relatively the same number of chromosomes each time it was running. This algorithm has been able to schedule courses without going through a crossover and permutation process because the value of the fitness function has reached 1. With the number of Ntotal > nSKS it allows the algorithm to produce a fitness function of 1 even though the number of chromosomes is not yet fully worth 1. The results can be seen from the graph below



(a)

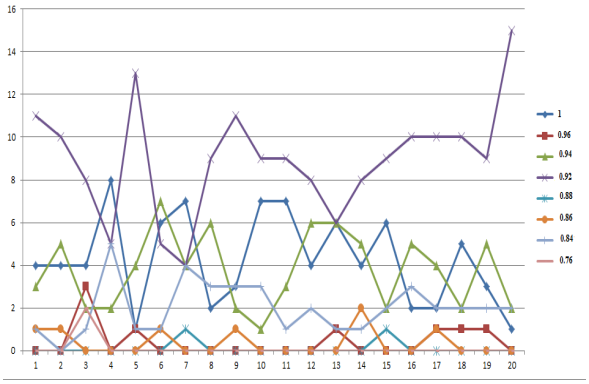
(b)

Figure 2. The result of (a) Graphs, and (b) Matrix 3 Dimensions Sample 2

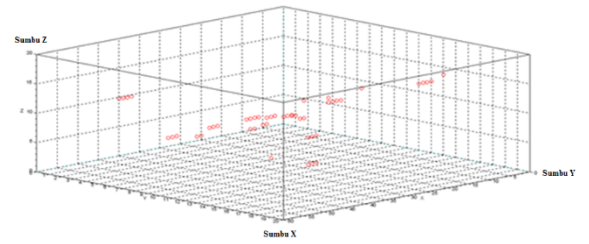
The resulting 3-dimensional matrix shows that there is no duplication of courses, lecturers and rooms. With Ntotal, which makes it easier for the algorithm to place course packages according to need.

3. *NTotal = 54 consisting of nroom = 2, nday = 3 , nhour = 9*

This algorithm in this third experiment has been able to schedule courses without going through the crossover and permutation process because the fitness function value has reached 1. The results show that the fitness function value has reached 1 with a different number of chromosomes each time it is running. With the number of Ntotal > nSKS allows the algorithm to produce a fitness function 1 even though the number of chromosomes is small. This is due to the Ntotal value which is almost equal to nSKS (50) so that when arranging course packages there are very few ways to arrange them. The results of the study can be seen from the graph below:



(a)



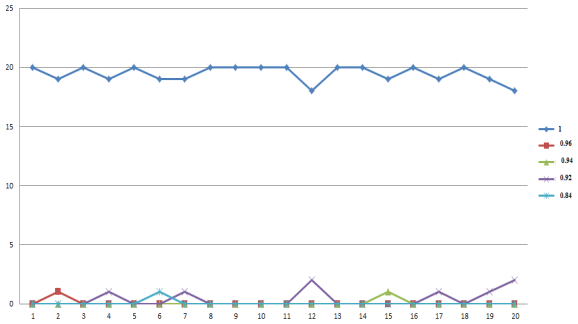
(b)

Figure 3. The result of (a) Graphs, and (b) Matrix 3 Dimensions Sample 3

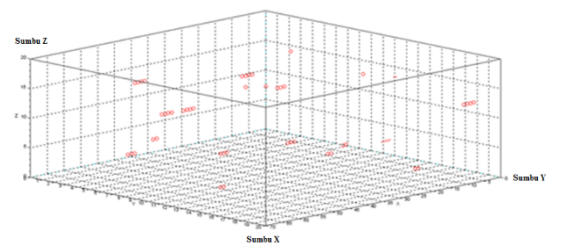
From the graph above, it can be seen that the value that is often repeated every time it is running is 0.92 which increases until it is running to 20. The resulting 3-dimensional matrix for this sample has been able to arrange the course packages according to their needs.

4. *NTotal = 72 consisting of nroom = 3 , nday = 3 , nhour = 8*

This algorithm has been able to schedule courses without going through a crossover and permutation process because the fitness function value has reached 1. The results show that the fitness function value is relatively the same every time it is running. With the number of Ntotal > nSKS allows the algorithm to produce a fitness function 1 with more chromosome values. Because with the same day and hour when adding one more space, it means that the algorithm is more flexible to arrange the schedule, so this experiment is declared the best experiment than the others. The matrix in this 4th experiment has no duplication and is arranged quickly. The results of the study can be seen from the graph below:



(a)



(b)

Figure 4. The result of (a) Graphs, and (b) Matrix 3 Dimensions Sample 4

4. Conclusion

The Genetic Algorithm that has been made has been able to schedule courses according to the specified limits so as to avoid duplication of lecturers, rooms and classes. In the first

experiment with a total of 48, the program has succeeded in scheduling 48 credits as evidenced by the fitness function 0.96 (48/50). Experiments 2 to 4 with a total of more than 50, the program has succeeded in scheduling courses with a fitness function value = 1. Even in just one loop. This algorithm has succeeded in increasing the fitness function through crossover and mutation processes.

Acknowledgment

The author wish to thank to the Universitas Islam Negeri Walisongo Semarang for the support.

References

[1] Rusdiana.2004.Judul Skripsi :*Optimalisasi Penjadwalan Mata Kuliah menggunakan Metode Algoritma Genetika.* Jakarta : Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Syarif Hidayatullah.

[2] A. Jain, D.S. Jain, dan D.P. Chande, "*Formulation of Genetic Algorithm to Generate Good Quality Course Timetable*". *International Journal of Innovation, Management and Technology 1*, (2010) 248-251.

[3] Fitri, R. (2004). *Penjadwalan Perkuliahan Dengan Pengujian Tabel Waktu (Time-Table) Menggunakan Algoritma Genetika.* 2004: Universitas Komputer Indonesia.

[4] Jenna Carr. 2014. *An Introduction to Genetic Algorithms*.

[5] Haupt, R. L., & Haupt, S. E. (2004). *Practical Genetic Algorithms* (2nd ed.). Hobo-ken: Wiley.

[6] Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading: Addison-Wesley.

[7] Zheng, Yingsong., Kiyooka, Sumio.(1999),"*Genetic Algorithm Applications*", Asia-Pacific Journal of Operational Research, 7 172-189.

[8] Saifullah, Shoffan., Arief Hermawan.2016. *Pengembangan Sistem Penjadwalan Kuliah Menggunakan Algoritma Steepest Ascent Hill Climbing*

[9] Simon, D. (2013). *Evolutionary Optimization Algorithms: Biologically-Inspired and Population-Based Approaches to Computer Intelligence*. Hoboken: Wiley.

[10] Fachrudin, A. (2010). *Penerapan Algoritma Genetika Untuk Masalah Penjadwalan Job Shop Pada Lingkungan Industri Pakaian.* Surabaya: Institut Teknologi Sepuluh Nopember.