Comparison of Genetic Algorithm and Hill Climbing for Shortest Path Optimization Mapping

Mona Fronita [1,*], *Rahmat* Gernowo [2], and *Vincencius* Gunawan [2]

[1]Master of Information System, School of Postgraduate Studies, Diponegoro University, Semarang – Indonesia 50242
[2] Department of Physics, Faculty of Science and Mathematics, Diponegoro University, Semarang – Indonesia 50275

Abstract. Traveling Salesman Problem (TSP) is an optimization to find the shortest path to reach several destinations in one trip without passing through the same city and back again to the early departure city, the process is applied to the delivery systems. This comparison is done using two methods, namely optimization genetic algorithm and hill climbing. Hill Climbing works by directly selecting a new path that is exchanged with the neighbour's to get the track distance smaller than the previous track, without testing. Genetic algorithms depend on the input parameters, they are the number of population, the probability of crossover, mutation probability and the number of generations. To simplify the process of determining the shortest path supported by the development of software that uses the google map API. Tests carried out as much as 20 times with the number of city 8, 16, 24 and 32 to see which method is optimal in terms of distance and time computation. Based on experiments conducted with a number of cities 3, 4, 5 and 6 producing the same value and optimal distance for the genetic algorithm and hill climbing, the value of this distance begins to differ with the number of city 7. The overall results shows that these tests, hill climbing are more optimal to number of small cities and the number of cities over 30 optimized using genetic algorithms.

1 Introduction

Traveling Salesman Problem (TSP) is a complex combinatorial optimization problem, originally presented by Dantzig. TSP concept can be illustrated as follows: A salesman depart from a city, visiting all the cities in one trip without visiting the same city more than once and finally returned to the city he left early. The goal is to minimize the total distance and the cost of salesmen who have to travel to visit all the cities [1].

Some methods can be applied to solve the TSP is *Ant Colony Optimization* (ACO) [2], *Annealing Algorithm* [3], *Hill Climbing* [4] *Genetic algorithm* [5], *Greedy algorithm* [6].

From several methods of TSP completion, hill climbing algorithm has good performance in local searching. Starting from defining the initial group, deciding the better search area up to iterating from level to level, the results of each level is taken from best and then compared, so they can get more optimal [7]. This TSP method is used to determine the nodes that has been given the distance among other nodes by comparing the existing node based on selection of the shortest distance from the initial position [8].

Beside hill climbing to solve TSP problems, there is also genetic algorithm. Genetic algorithm is a powerful and flexible metaheuristic as well as the relatively new type of algorithm by adopting the idea of natural selection and genetic changes naturally. This algorithm is known as a tool that can solve combinatorial in optimization problems such as TSP [9]. GA worked by solving the problem of appropriate population selection (known as a chromosome) that will later perform iterations / generations by applying the three basic genetic operators, i.e. selection, crossover and mutation. By combining simple operator as a single operator, to create complex operator, which can accelerate the evolutionary process, so that it can improve the efficiency of GA [10].

In the problems of *Traveling Salesman Problem* is finding a solution to compute the shortest path or minimum travel route of all paths. Genetic algorithm and hill climbing has advantages and disadvantages. An algorithm has a different result, because an algorithm that has a high optimization for a case uncertainly having a high optimization also for other cases. Both of these algorithms depend on the number of cities to determine which is more efficient in calculating the optimum distance. If the number of cities visited by sales fewer so then it is obtained a shorter distance. However, it is still not assured that the total shortest travel distance is obtained when the number of cities which must be visited by sales is the fewest.

Thus, the research it will be carried out a comparison between genetic algorithm and hill climbing based on the calculation of the shortest path and computing time to see which one is more optimal to finish salesman traveling problems.

[*] Corresponding author: mona_fron@yahoo.co.id

2 Theoretical Framework

2.1. Traveling Salesman Problem (TSP)

The problem of Traveling Salesman Problem (TSP) is one of the most studied examples in combinatorial optimization. This problem is easy to declare but very difficult to solve. When completed in exact time the required computation will increase exponentially as the problem increases. TSP can be expressed as a problem in finding the minimum distance of a closed journey to a number of n cities where the cities are only visited once. TSP is represented by using a complete and weighted graph G = (V, E) with V set of vertices representing a set of points, and E is the set of edges. Each edge (r, s) ∈ E is the value (distance) rs which is the distance from city r to city s, with (r, s) ∈ V. In a symmetric TSP (distance from city r to point s is equal to distance from point S to point r), drs = drs for all edges (r, s) ∈ E . In a graph, the TSP is drawn as shown in Figure 1 below:

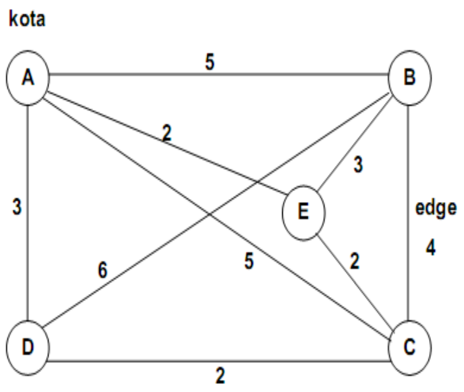

Fig 1 The illustration of TSP

2.2 Hill Climbing Method

The Simple Hill Climbing method is an offshoot of Hill Climbing method. The Simple Hill Climbing search movement starts from the leftmost position after the initial and pointed points are determined by comparing the current point state with a single point regardless of the next point at the same level, and a better first point being selected to the next. The move is done continuously until the final goal is obtained. Hill Climbing algorithm is as follows:

1. Determine the initial random trajectory and calculate the distance of the initial path, then tested by swapping each city. After testing if the initial path is the destination city, stop, and if the initial path is not a destination city continue with the current state as the initial path.
2. Do the following steps until the solution is found, or until no new point will be applied to the current state.
   A) Look for a point that has not been used in the current state, and use an exchange point to get a new state.
   B) Evaluation of the new situation;

If the new state is a destination city, then the search process stops,

If it is not a destination city, but its value is better or smaller than it is now, then make that new state into the present state.

If the new situation is no better than the current situation then continue the search, as in the previous step to get the destination city with the smallest weight. Hill climbing process flow can be seen on fig. 3 Hill Climbing flowchart.

There are several problems that may occur in Hill Climbing method:

A. The algorithm will stop if it has reached the local maximum value i.e. a current path that is considered to have the shortest distance from all previous trajectories, but not necessarily the current path is considered to be the shortest distance is the destination because there is still a track that has not been visited in the predefined iteration.
B. The sequence of points used to swap positions will greatly affect the discovery of solutions.
C. Hill Climbing passes a point of exchange that has been used in the previous step [11].

2.3 Genetic algorithms

Main components of Genetic Algorithm are Encoding, Function Evaluation, Selection, Crossover and Mutation. Encoding is a difficult process in the genetic algorithm. It because the encoding process for each problem is different because not all coding techniques suitable for each problem. The encoding process results a row is called chromosomes. Chromosomes consist of a set of bits known as genes. There are several kinds of coding techniques can be done in a genetic algorithm [12], including binary encoding, permutation encoding, value encoding, and tree encoding.

In TSP, used coding techniques are permutation encoding, where a chromosome represents a permutation of the city by serial numbering 1, 2, 3, ..., n. The evaluation process is a process of calculating fitness values expresses chromosomes quality level as a representation of problem solving. In this case, fitness function *eval (v)* cannot be defined directly from the objective function *f (v)* because TSP is the problem of finding the minimum value. The fitness function must be mapped from the objective function. The objective function is a calculation of all sides formed from existing genes on a chromosome. For example there are as many as *n* genes on a chromosome then *f (v)* of chromosomes is:

$$f(v) = D_{1:2} + D_{2:3} + \ldots + D_{n-1:n} \qquad (1)$$

With $D_{1:2}$ is the distance between city 1 to city 2, to city *n-1* to town *n*. The simplest way to determine the fitness function is mapping every addition value of objective function as a declining of the fitness function value with the following equation:

$$eval\,(v) = \frac{1}{\sum_{i=1}^{n} f(v)} \qquad (2)$$

Selection will determine which individuals are chosen to be recombined and how the *offspring* is made up from those elected individuals. Each individual in a selection forum will receive the reproduction probability depends on the objective value itself against the objective value of all individuals in those selection forum. A selection methods commonly used is the *roulette-wheel*. As the name implies, this method imitates the roulette-wheel game where each chromosome occupies a piece of the circle on the roulette wheel proportionally based on its fitness value. The chromosome which has greater fitness value occupies a piece of the larger circle than the low fitness valuable chromosome, How the roulette-wheel method works [15]:

1. Calculating the probability value of each chromosome

$$P[i] = \frac{eval\ (v)}{\sum_{i=1}^{n} eval\ (v)} \tag{3}$$

2. Generating random number with interval (0-1) for the value of R
3. Calculate the cumulative probability of each chromosome, defined

   K[0]=0 K[i]=K[i-1]+P[i] (4)

Crossover (crossing) is carried on 2 chromosomes to produce children chromosome (*offspring*). Children chromosomes formed will inherit some of their parents chromosomes nature. In the crossover there is a very important parameter, namely the crossover opportunities (P_c). Crossover opportunities shows the ratio of children produced from each generation by the population size. For example, population size (pop size = 100), while the crossover opportunities ($P_c = 0.25$), it means that there are 25 chromosomes of 100 chromosomes which are hoped to be exist in the population and undergoing *crossover*. Crossing over of TSP can be implemented by *order crossover* scheme.

Mutation is the process of modifying the children chromosomes at random. Mutation will create new individual by altering one or more genes in one chromosome. Mutation role is to replace the missing genes from the population as a result of selection process and allow the emergence of genes that are not present in the initial population [13]. In TSP, the mutation operator is usually implemented by exchanging mutated genes with other genes selected randomly. For example, chromosome {2, 3, 4, 1, 5} can be mutated into a chromosome {4, 3, 2, 1, 5}. In this case, gene 1 and gene 3 are exchanged. This mutation scheme known as *swapping mutation*. Genetic algorithm process flow can be seen on fig. 2 Genetic algorithm flowchart.

3 Methodology

3.1 Analysis and Design of Information Systems

The shape of system design can be modeled as a chromosome made up from a number of genes in which each of these genes represent each city that will be passed by a salesman. The number of genes on one chromosome is the number of cities plus one city that will be the similar initial city with the final city which will be passed by the salesman so that the chromosome is a trip route that may be passed by the salesman with the arrangement of input and output process. In fig 2 and 3 there is a flow of performance from the algorithm and hill climbing. In this arrangement, it consists of administrator and user (salesman).

1. *Input* is the process of data input in the form of a series of activities performed by the administrator and salesman.
   Web Administrator, facilitating the entry of data master or data admin and performing management of items data input, city data in the form of latitude and longitude coordinate points, initialization parameters used in the calculation of genetic algorithm and hill climbing
   User, enter data cities to be visited to make delivery for determining the shortest track.

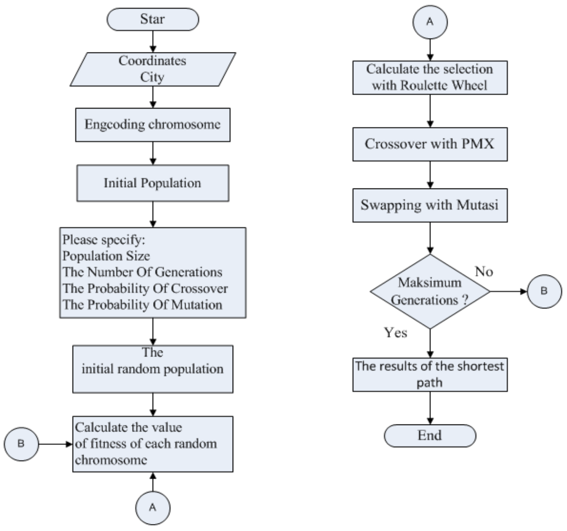
Star
Coordinates City
Engcoding chromosome
Initial Population
Please specify:
Population Size
The Number Of Generations
The Probability Of Crossover
The Probability Of Mutation
The initial random population
Calculate the value of fitness of each random chromosome
B
A
A
Calculate the selection with Roulette Wheel
Crossover with PMX
Swapping with Mutasi
Maksimum Generations ?
No
B
Yes
The results of the shortest path
End

Fig.2 Genetic algorithm flowchart

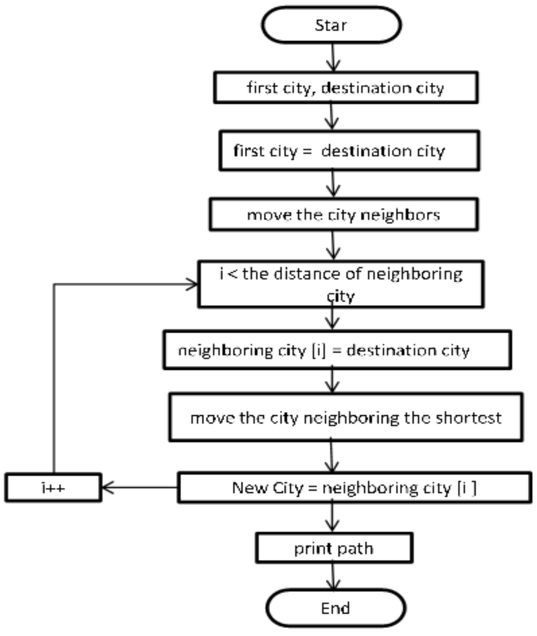
Star
first city, destination city
first city = destination city
move the city neighbors
i < the distance of neighboring city
neighboring city [i] = destination city
move the city neighboring the shortest
i++
New City = neighboring city [i]
print path
End

Fig. 3 Hill Climbing flowchart

2. *Process* is by creating coordinate location point by performing coordinate input in the form of latitude and longitude of JNE locations throughout the territory of Central Java which later became the master data that is stored in the database. Creating a track using Google Map API that connects coordinate points using the marker of inter-city destinations by salesmen that would be seeing a path to be passed by graph using a polyline, and then it will be sorted to obtain the optimal shortest track.The search process using a genetic algorithm for TSP and Hill Climbin.

4 Finding and Discussion

In the TSP process genetic algorithm and hill climbing is compared with some aspects such as the ratio of distance, time of execution, the generation / level and the passed track. Comparison of generation at the level of genetic algorithm and hill climbing, the genetic algorithm parameters to generation, the greater the number of generations, then the more optimal result will be got. While on the *hill climbing*, the repetition process will continue at the next level as far as finding the most optimum distance then the repetition stops. But, in both of these methods the optimal results are not necessarily in the last generation or level. Based on the calculation of the research finding, it is obtained the same and optimal distance by the testing 4, 5 and 6 cities either using genetic algorithm or *hill climbing*. If the number of cities inserted more than 7 cities producing a different city distance but optimal for distance and computing time such as shown in Table 1 and table 2

Table 1

Comparison of distance of the trajectory of the GA and HC

No	Number of cities	GA			HC		
		Average	The shortest time	The longest time	Average	The shortest time	The longest time
1	8	11112.1	553.5	559.9	11450	572.5	572.5
2	16	24217	1038.6	1319.9	26718	1335.9	1335.9
3	24	38650.6	1736.6	2087.2	43356	2167.8	2167.8
4	32	53539.04	115.136	2993.8	2889.1	2889.1	2889.1

Distance on the genetic algorithm varies while on his hill climbing, the distance is more consistent. Further different distance can be seen from the image of a distance testing with the number of different cities.

Distance from *hill climbing* is always similar proven from 20 times testing, while the genetic algorithm to get various distance. The test based on this distance can be seen from the optimal value of the shortest track on genetic algorithm and *hill climbing*. The results of distance comparison among the eight cities shows that the use of *hill climbing* is more optimal and always similar in fig 4.

Table 2

Comparison of the time path of GA dan HC

No	Number of cities	GA			HC		
		Average	The shortest time	The longest time	Average	The shortest time	The longest time
1	8	850.575	41.791	44.766	3.518	0.185	0.251
2	16	3101.754	131.99	236.338	47.833	2.013	3.292
3	24	3570.397	149.786	282.84	73.326	3.715	4.298
4	32	9845.27	108.23	2720.4	139.559	7.165	7.502

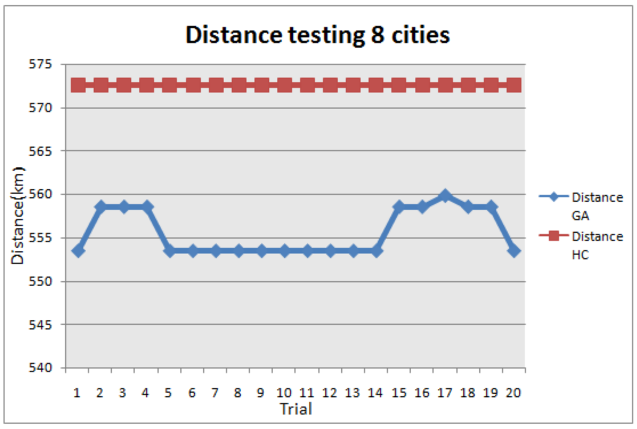

Fig. 4 Distance testing 8 cities

In fig 5, The distance using *hill climbing* with 16 testing of the city is similar, while the genetic algorithm produces various distances. The distance resulting from *hill climbing* is more optimal than the genetic algorithm. However, the genetic algorithm has a shorter distance.

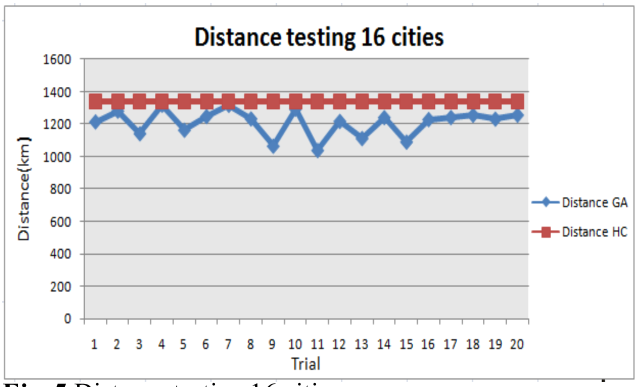

Fig. 5 Distance testing 16 cities

By testing within the 24 cities, in fig 6 hill climbing results are still more optimal than the genetic algorithm, but the results of the genetic algorithm distance does not differ much from *hill climbing*. On testing to 4 and 9 to produce the same distance.

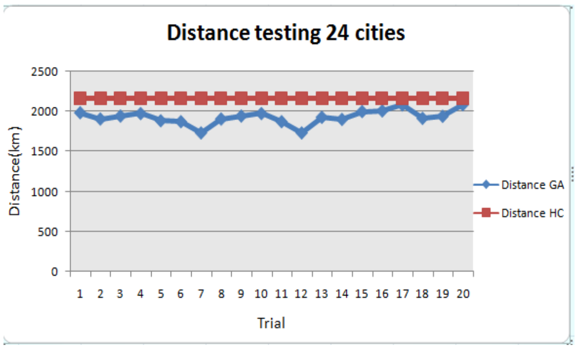

Fig. 6 Distance testing 24 cities

By testing of 32 cities, it is produced an optimal distance using genetic algorithm, even it is lower than the *hill climbing*, although the most optimal average distance is *hill climbing*. The optimal distance obtained from testing 1, 6 and 9.

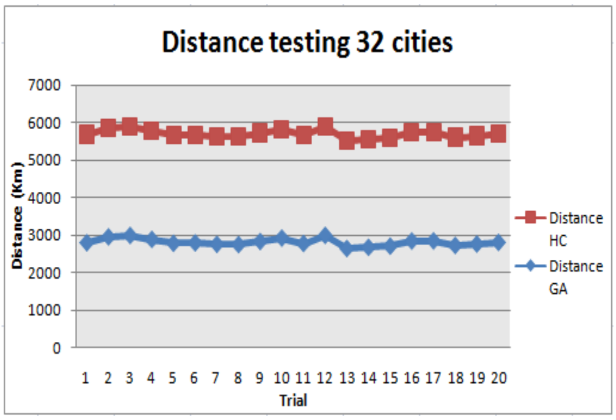

Fig. 7 Distance testing 32 cities

After testing the entire distance with the number 8, 16, 24 and 32 cities, it can be seen that the genetic algorithm has the long enough average distance in testing the amount of 8, 16 and 24 cities, in fig 7 with 32 cities which has relatively shorter distance. While *hill climbing* on testing the amount of 8, 16 and 24, the city which has shorter track distance, with the number of 32 cities which has longer track

5 Conclusion

Web-based system built by mapping optimization the shortest track with genetic comparisons and *hill climbing*. In a system of genetic algorithm, parameters to a total population of 10, 60% of the crossover probability value, mutation probability value of 50% and the number of generations are 200. Meanwhile, on *hill climbing* there is no input of certain parameters. The result of the system that will be compared based on distance, computation time, generation / level and track sequence. The results can be seen after 20 times of testing with the amount of city 8, 16, 24 and 32. Based on the overall computation of the fastest time using *hill climbing* is the shortest time, 57.753 seconds. Based on the relatively optimal distance of *hill climbing* with the number of cities which are relatively fewer, for cities over 30, it will be more optimal to use Genetic Algorithms.

References

1. Gutin, abd, G., and A. P. Punnen, *The traveling salesman problem and its varia-tions*. NewYork: Springer, (2007)
2. M. Mavrovouniotis, and Yang., Applied Soft Computin **13** 4023–4037, (2013)
3. X. Zheng, Z. Li, and M. Li, *Applying Simulated Annealing Algorithm*, (2010)
4. H. S. Jacobson, A. L. McLay, N. S. Hall, D. Henderson, E. D. Vaughan, International Journal of Mathematical And Computer Modelling. Vol **43** page 1061-1073, (2006)
5. Lim, European of operational reaserch **5**, 1459-1478, (2005)
6. S. Maity, A. Roy, and M. Maity, Journal Computer and Industrial Engineering Vol. **83***, 273-296*, (2015)
7. J. Brecklinghaus, and S. Hougardy, Operations Research Letters, **43** 259–261, (2015)
8. H. Junliang, M. Weijian, Y. Wei, *Journal of Shanghai Maritime University*, **28** (4),11, (2007)
9. J. Majumdar, and A,K. Bhunia, Journal of Computational and Applied Mathematics **235** page 3063–3078,(2011)
10. S. Kusumadewi, and H. Purnomo, *Penyelesaian Masalah Optimasi Dengan Teknik-Teknik Heuristik*. GRAHA ILMU : Yogyakarta, (2005)
11. Z. Zukhri, *Algoritma Genetika : Metode komputasi Evolusioner untuk Menyelesaikan Masalah Optimasi,* Penerbit Andi Yosgyakarta, (2014)
12. Suyanto, *Artificial Intelligence.* Informatika: Bandung, (2014)