THE INTERNATIONAL JOURNAL OF SCIENCE & TECHNOLEDGE

Adjustable Knapsack in Travelling Salesman Problem Using Genetic Process

Andysah Putera Utama Siahaan
Faculty, Department of Computer Science,
Universitas Pembangunan Panca Budi, Medan, Sumatera Utara, Indonesia

Abstract:
In the Travelling Salesman Problem, all of the nodes available must be completely passed. The algorithm searches the optimum value that approaches the fitness demand. Usually, fitness is set to "1". It is impossible to reach the highest fitness, but the genetic process can optimize the fitness value based on the total amount of the population included. There are two targets achieved in the Knapsack method, weight, and node count. The users only determine how many nodes included. The computer pick them randomly. The node represents how many stops will be visited while the weight is the total distance of every node passed until it returns to the starting point. The weight of the chosen nodes should approach the target weight.

Keywords: *Artificial Intelligent, TSP, Knapsack Problem*

1. Introduction

In the case of Travelling Salesmen Problem, genetic algorithms have to find the solution of all the nodes that pass and looking for the most optimal value or the shortest path passes on the track. The genetic algorithm can find the optimal solution in each generation were executed but the result is not a correct answer to find the desired distance [7]. Knapsack Problem is a combinatorial optimization problem. For example, given a set of items by weight and value, then do the selection of these items to put in a bag with a limited capacity. Thus, the items included weight must be less than or the same as less capacity of the bag, but the total value as possible [1]. Knapsack problem can specify the desired weight in a living room [2][3]. Knapsack itself has at least two parameters as determining whether the fitness of a population approaching with a predetermined solution. The parameters used in this study is the number of nodes and weight range. The number of nodes is the number of point coordinates that will pass while the weighted distance is the distance between the accumulated number of nodes to go back to the origin node [4][5][8]. In this algorithm is expected to achieve a solution that can generate fitness = 1 or at least closer to that number.

2. Proposed Work

2.1. Coordinate Data

In the genetic algorithm, the determination of mutation carriers who are affected by the value of the movement is very important, because it will provide optimal solutions. The determination of mutations dynamically expected to avoid local optimal and accelerate the discovery of solutions to problems because the change can be controlled. This section explains how the genetic algorithm optimization process took part in the search. In search of the track, there are some important things to include the weight and number of passes. The resulting trajectory is not entirely out of the total trajectory but most track courses depend on the desired input. The number of passes is sought does not exceed the total trajectory.

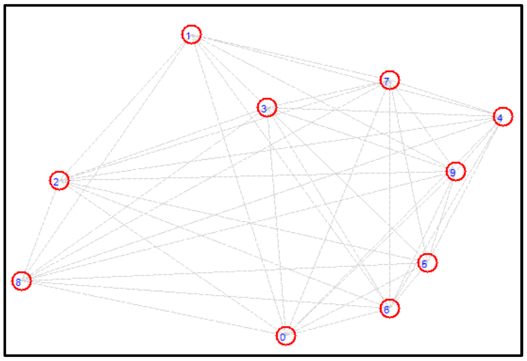

Figure 1: A ten nodes trajectory

The above picture illustrates a track pattern with 10 nodes consisting of the point 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10. Each node will be laid on two-dimensional coordinates of the origin point (0, 0) is at the top left. The following table specifies the information the neighbor coordinate position of each note on the X and Y axis.

Node	X	Y
0	42	40
1	32	7
2	18	23
3	40	15
4	65	16
5	57	32
6	53	37
7	53	12
8	14	34
9	60	22

Table 1: Coordinate data

Table 1 show the coordinate data from the previous figure. It tells the positions of X and Y. The data is generated randomly. Each node has a pair of X and Y axis for example, on the fifth node, it has X-axis = 57 and Y-axis = 32.

2.2. Distance

To determine which path has the best optimal value. The distance used to determine the position of the track which has the most optimal value that is by determining the number of tracks which are closest to the desired weight. Search this distance is different from the distance in the case of TSP in general. In the general case, the distance will calculate the weight of each node, but the problem Knapsack sought distance based on the number of nodes that cool only and specific weights revolution. The distance can be calculated by the following formula.

$$Distance = \sum_{i=0}^{n} \sqrt{(x2 - x1)^2 + (y2 - y1)^2} \qquad (1)$$

2.3. Fitness Calculation

The search of fitness in the genetic processes should be able to make a difference in the search for a value of 0. If the difference value is 0 then the fitness of the population is 1. The difference in value between the length of the path and the target weight is an Error Code. If the Error Code 0 means automatically produces fitness would be worth 1. It can be seen through the following formula.

$$F = \frac{1}{1+BobotTarget-Jarak} \qquad (2)$$

2.4. Selection

At the selection process, the genetic processes will duplicate the population considered either to be used as an opportunity to the next generation. To find out more about the selection process in this method can be seen in the following explanation.

2.5. Mutation

Mutation process used is a dynamic mutation in which the value of mutation rate is dependent on the state of one generation. If in one generation there are many fitness values closer to the target, then the percentage of mutations will be getting less and less and vice versa if the values of fitness on subsequent generations away from the target, the percentage of mutations will be greater.

For example, there are 10 populations is divided into three parts, namely Low, Medium and High. On the results of fitness before the mutation will do fitness testing which is closer to the target. The output value of the calculation into the value used as the mutation rate.

In the calculation of the following will explain how the calculations dynamically to determine the value of the process of genetic mutation rate. This example having ten pieces of the population after the selection process where each population has five nodes were visited. The data below is taken from the calculation of the sum of the distance the path traversed. Every population has a different value. These values will be compared to predetermined targets. A value which is closer to the target has a great chance to be a solution. Values in excess of 1 cannot be validated because it has exceeded its target. The solution should only be between a target value and the value that was on at the bottom.

3. Result and Discussion

3.1. Coordinate Formation

In the genetic This section includes a discussion of the methods that have been discussed in previous chapters. Here will be tested multiple samples of coordinates TSP. The following table is an early test data to implement the method Knapsack on the TSP.

Coordinate	X	Y
0	23	38
1	26	28
2	68	16
3	30	21
4	41	14
5	15	41
6	56	6
7	57	30
8	42	17
9	25	22
10	11	36
11	35	7
12	27	13
13	6	37
14	7	23
15	30	12
16	59	8
17	36	30
18	58	39
19	46	22
20	40	17
21	41	39
22	39	9
23	58	22
24	13	13
25	31	27
26	32	5
27	32	20
28	32	10
29	24	29

Table 2: Tested coordinate data

In Table 2, there are 30 pieces of coordinates which run on the TSP. These coordinates starting from the coordinates of 0 to 29. Coordinates creation is done randomly. The test is not to save the coordinates in a static database. It is intended that this method can be applied to a wide range of issues.

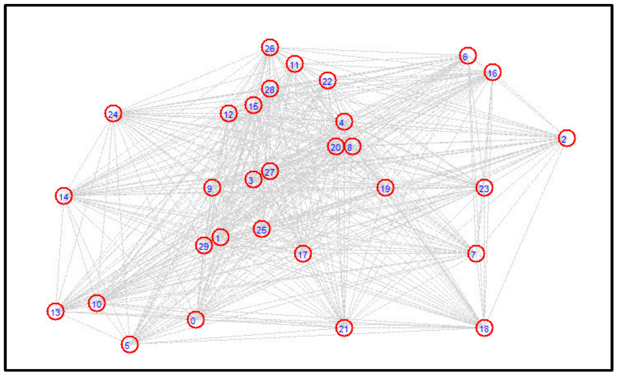

Figure 2: Trajectory establishment of the coordinates

Figure 2 is the nodes that have been formed based on the coordinate data that has been generated at random. Values X axis and Y axis can be seen in Table 4.1. This image forming nodes that are within the range of X <= 75 and Y <= 45. This value can be customized according to user needs. So that the calculation does not have a value that is too large, then the reference value does not exceed 100. In the manufacture of TSP coordinates, must not have value X axis and Y axis simultaneously, in other words, there should not be exactly the same coordinate value. Problems faced if there are two nodes that have the same coordinates is not a movement in the nodes. But it has a very low probability because the value is raised is a random value. Meanwhile, if there are some nodes that have the same value on one of the axes X or Y axis, this will not affect the course of the calculation process TSP Knapsack.

3.2. Knapsack Target Value

Target is a goal that will be achieved in the case of TSP. The fundamental difference between conventional TSP with TSP Knapsack is situated in the determination of the target. In the TSP Knapsack, the number of nodes to be skipped is determined based on the wishes of the user. Weights or distances achieved are also to be determined in advance. While the conventional TSP, all nodes that are created must be passed with a minimum distance of most genetic algorithm. At this time the calculation of the target for the number of nodes that is passed is 10 nodes while the target desired distance of 400. It is expected that TSP Knapsack calculation can determine which nodes that have the same opportunities to the target or close to the target with as close.

3.3. Population

Intake of population taken at random from the arrangement coordinate initial TSP. At the coordinates previously stated there are 30 pieces of coordinates that each have a value of x-axis and y-axis. The population should not exceed the amount of the existing coordinates. For example, would have taken a population consisting of 10 nodes.

Coordinate	X	Y
22	39	9
8	42	17
12	27	13
4	41	14
13	6	37
3	30	21
14	7	23
11	35	7
26	32	5
10	11	36
22	39	9

Table 3: Population generated

Table 3 represents the nodes that are selected from the formation of 10 nodes which have become the target of Knapsack TSP. The formation process will last until the population size of the population is met. In this experiment will be carried out as many as 10 pieces' populations. The data shown above is a data capture as much of the population only.

P	N 0	N 1	N 2	N 3	N 4	N 5	N 6	N 7	N 8	N 9
0	22	8	12	4	13	3	14	11	26	10
1	4	29	0	18	22	26	12	14	23	27
2	1	4	22	16	10	8	19	6	20	28
3	19	16	1	27	13	14	9	21	8	7
4	24	7	4	26	15	16	3	1	20	17
5	15	28	21	8	18	19	14	25	24	3
6	15	5	23	27	19	16	28	3	2	8
7	15	24	3	14	2	25	28	26	0	21
8	3	16	15	2	12	23	26	21	7	25
9	28	14	23	5	21	17	12	7	3	27

Table 4: A ten populations generated

The table above is the formation of a population of 10 pieces were randomly obtained from population generation process. Each population of population 0 to 9 will be calculated to obtain the fitness value. The fitness, approaching the target has a greater opportunity to be a parent in the process of selection and mutation.

3.4. Fitness Value

The previous population is the temporary form of the results of the formation of the track corresponding to the desired target. N0 - N9 declared node that will be passed on the 10 tracks selected. Population 0 through 9 have a fitness value of the population respectively. Calculation of fitness in the first generation is only a test of whether the value of fitness of the population in line with expectations or Fitness = 1. But in general, something that is not possible on the first generation fitness value is achieved perfectly. In the calculation of the genetic algorithm is not always the target of fitness is worth one. In Knapsack problem, fitness can be obtained by way of the best fitness value in the amount of the existing generation.

Node 22 to Node 8

Distance 1.X1	=	39
Distance 1.Y1	=	9
Distance 1.X2	=	42
Distance 1.Y2	=	17
Distance 1	=	$\sqrt{(42-39)^2+(17-9)^2}$
	=	$\sqrt{3^2+8^2}$
	=	$\sqrt{9+64}$
	=	$\sqrt{73}$
	=	8.544
	=	9

Node 8 to Node 12

Distance 1.X1	=	42
Distance 1.Y1	=	17
Distance 1.X2	=	27
Distance 1.Y2	=	13
Distance 1	=	$\sqrt{(27-42)^2+(13-17)^2}$
	=	$\sqrt{-15^2+-4^2}$
	=	$\sqrt{225+16}$
	=	15.524
	=	16

Node 12 to Node 4

Distance 1.X1	=	27
Distance 1.Y1	=	13
Distance 1.X2	=	41
Distance 1.Y2	=	14
Distance 1	=	$\sqrt{(41-27)^2+(14-13)^2}$
	=	$\sqrt{14^2+1^2}$
	=	$\sqrt{196+1}$
	=	$\sqrt{197}$
	=	14.036
	=	14

Node 4 to Node 13

Distance 1.X1	=	41
Distance 1.Y1	=	14
Distance 1.X2	=	6
Distance 1.Y2	=	37
Distance 1	=	$\sqrt{(6-41)^2+(37-14)^2}$
	=	$\sqrt{-35^2+23^2}$
	=	$\sqrt{1225+529}$
	=	$\sqrt{1754}$
	=	41.881
	=	42

Node 13 to Node 3

Distance 1.X1 = 6
Distance 1.Y1 = 37
Distance 1.X2 = 30
Distance 1.Y2 = 21
Distance 1 =
$\sqrt{(30-6)^2+(21-37)^2}$
= $\sqrt{24^2+-16^2}$
= $\sqrt{576+256}$
= $\sqrt{832}$
= 28.844
= 29

Node 3 to Node 14

Distance 1.X1 = 30
Distance 1.Y1 = 21
Distance 1.X2 = 7
Distance 1.Y2 = 23
Distance 1 =
$\sqrt{(7-30)^2+(23-21)^2}$
= $\sqrt{-23^2+2^2}$
= $\sqrt{529+4}$
= $\sqrt{533}$
= 23.087
= 23

Node 14 to Node 11

Distance 1.X1 = 7
Distance 1.Y1 = 23
Distance 1.X2 = 35
Distance 1.Y2 = 7
Distance 1 =
$\sqrt{(35-7)^2+(7-23)^2}$
= $\sqrt{28^2+-16^2}$
= $\sqrt{784+256}$
= $\sqrt{1040}$
= 32.249
= 32

Node 11 to Node 26

Distance 1.X1 = 35
Distance 1.Y1 = 7
Distance 1.X2 = 32
Distance 1.Y2 = 5
Distance 1 =
$\sqrt{(32-35)^2+(5-7)^2}$
= $\sqrt{-3^2+-2^2}$
= $\sqrt{9+4}$
= $\sqrt{13}$
= 3.6056
= 4

Node 26 to Node 10

Distance 1.X1 = 32
Distance 1.Y1 = 5
Distance 1.X2 = 11
Distance 1.Y2 = 36
Distance 1 =
$\sqrt{(11-32)^2+(36-5)^2}$
= $\sqrt{-21^2+31^2}$
= $\sqrt{441+961}$
= $\sqrt{1402}$

= 37.443
= 37

Node 10 to Node 22

Distance 1.X1 = 11
Distance 1.Y1 = 36
Distance 1.X2 = 39
Distance 1.Y2 = 9
Distance 1 =

$$\sqrt{(39-11)^2+(9-36)^2}$$

$= \sqrt{28^2+-27^2}$
$= \sqrt{784+729}$
$= \sqrt{1513}$
= 38.897
= 39

Total Distance = $\sum_{i=0}^{9} D$
= 9+16+14+42+29+23+32+4+37+39
= 245

Fitness $= \frac{1}{1+Target-Jarak}$
$= \frac{1}{1+400-245}$
$= \frac{1}{156}$
= 0.0064

3.5. Result of a Ten Generations

The following explanation will be attached on the calculation of fitness to get nodes on a given track. This calculation involves a target=10 and the total distance=400. The explanation below will explain the process steps to obtain the best fitness value which has a global optimum. The following words describe the calculation of the last generation.

```
Fitness After The Previous Generation
======================================
Population [0]   Distance : 400   Fitness : 1
Population [1]   Distance : 402   Fitness : -1
Population [2]   Distance : 402   Fitness : -1
Population [3]   Distance : 402   Fitness : -1
Population [4]   Distance : 402   Fitness : -1
Population [5]   Distance : 402   Fitness : -1
Population [6]   Distance : 402   Fitness : -1
Population [7]   Distance : 402   Fitness : -1
Population [8]   Distance : 402   Fitness : -1
Population [9]   Distance : 402   Fitness : -1

Probability [0]          : -0,125
Probability [1]          : 0,125
Probability [2]          : 0,125
Probability [3]          : 0,125
Probability [4]          : 0,125
Probability [5]          : 0,125
Probability [6]          : 0,125
Probability [7]          : 0,125
Probability [8]          : 0,125
Probability [9]          : 0,125
```

```
Cumulative Probability [0]: -0,125
Cumulative Probability [1]: 0
Cumulative Probability [2]: 0,125
Cumulative Probability [3]: 0,25
Cumulative Probability [4]: 0,375
Cumulative Probability [5]: 0,5
Cumulative Probability [6]: 0,625
Cumulative Probability [7]: 0,75
Cumulative Probability [8]: 0,875
Cumulative Probability [9]: 1
Random Number [0] : 0,95710364168375
Random Number [1] : 0,255675350900588
Random Number [2] : 0,0118650235290942
Random Number [3] : 0,147136231021553
Random Number [4] : 0,735718659467864
Random Number [5] : 0,814073528542218
Random Number [6] : 0,236068050486999
Random Number [7] : 0,234900025760243
Random Number [8] : 0,789310280601173
Random Number [9] : 0,309437703951
```

P	N0	N1	N2	N3	N4	N5	N6	N7	N8	N9
0	22	11	7	2	18	8	24	19	23	16
1	19	18	23	8	2	22	11	24	16	7
2	19	18	23	8	2	22	11	24	16	7
3	19	18	23	8	2	22	11	24	16	7
4	19	18	23	8	2	22	11	24	16	7
5	19	18	23	8	2	22	11	24	16	7
6	19	18	23	8	2	22	11	24	16	7
7	19	18	23	8	2	22	11	24	16	7
8	19	18	23	8	2	22	11	24	16	7
9	19	18	23	8	2	22	11	24	16	7

Table 5: Population after selection

```
Mutation Rate : 0,9
Total Mutasi : 90
```

P	N0	N1	N2	N3	N4	N5	N6	N7	N8	N9
0	22	11	7	2	18	8	24	19	23	16
1	18	19	2	8	23	24	11	16	22	7
2	24	23	8	19	2	22	7	11	16	18
3	8	7	11	23	22	16	18	24	19	2
4	23	18	16	22	2	7	11	8	19	24
5	22	7	19	16	23	2	8	24	18	11
6	8	7	11	23	22	16	18	24	19	2
7	8	7	11	23	22	16	18	24	19	2
8	22	7	19	16	23	2	8	24	18	11
9	18	19	2	8	23	24	11	16	22	7

Table 6: Population after mutation

```
Fitness After Genetic Process
======================================
Population [0]     Distance : 400     Fitness : 1
Population [1]     Distance : 401     Fitness : ∞
Population [2]     Distance : 408     Fitness : -0,14285714
Population [3]     Distance : 402     Fitness : -1
Population [4]     Distance : 399     Fitness : 0,5
```

Population [5]	Distance : 400	Fitness : 1
Population [6]	Distance : 402	Fitness : -1
Population [7]	Distance : 402	Fitness : -1
Population [8]	Distance : 400	Fitness : 1
Population [9]	Distance : 401	Fitness : ∞

Best On Postition : 8

P	**N 0**	**N 1**	**N 2**	**N 3**	**N 4**	**N 5**	**N 6**	**N 7**	**N 8**	**N 9**
0	22	7	19	16	23	2	8	24	18	11
1	18	19	2	8	23	24	11	16	22	7
2	24	23	8	19	2	22	7	11	16	18
3	8	7	11	23	22	16	18	24	19	2
4	23	18	16	22	2	7	11	8	19	24
5	22	7	19	16	23	2	8	24	18	11
6	8	7	11	23	22	16	18	24	19	2
7	8	7	11	23	22	16	18	24	19	2
8	22	7	19	16	23	2	8	24	18	11
9	18	19	2	8	23	24	11	16	22	7

Table 7: Population after elitism

Calculation of the ten generations that produce the best value Fitness = 1 where the target is obtained in accordance with the desired target. The population that has a value Fitness = 1, there are three among other populations [0], the population [5] and the population [8]. Tracks in this population are as follows:

22-7 - 19-16 - 23-2 - 8 - 24-18 - 11-22

The trails have a total range of 400. The use of dynamic mutations in highly creating the global optimum achievement because the population has a fitness value which is closer to the target fitness is not carried the mutation, so it will have a great opportunity to improve the value of their genes in future generations. The more value near the target fitness value the fewer mutations do. This dynamic mutation process greatly affects the outcome of genetic processes.

4. Conclusion

The process of the genetic algorithm at TSP Knapsack greatly assist in the improvement process of fitting the distance desired. The mutation process helps the generation reach the target value. The target value and the total nodes visited can be determined previously. It is to limit the ability of the subject to pass the trajectory. Knapsack is very important to apply in some specific cases. It helps to maintain the subject abilities.

5. Future Scope

In static mutations, mutations would need to have a fixed value. There is a condition in which a generation does not require a lot of the mutation process. If this happens, it will be many genes that approached the target value of fitness will be replaced with a better value. This process will assess the situation of a generation. Values range or range can be determined based on need. This dynamic mutation process can restrict a generation will be mutated. If the generation has a lot of fitness approaching the target value, the mutation process will be reduced and the possibility of both the population will be less replaceable.

Mutation process does not examine the situation of previous generations. It is desirable that the mutation rate determination for the better. Supervised learning methods can be used as a method of determining the combination of the Mutation Rate. So the value of the mutation can learn from the habits that happened before. In a subsequent study is expected to be no improvement determination of mutations in this TSP Knapsack. If the dynamic mutation value depends on the value of fitness in a generation that is running, the continuation of this study is expected mutation rate value depends on the calculation of the generations that have been missed previously.

6. References

i. A. Zaheed dan I. Younas, “A Dynamic Programming based GA for 0- 1 Modified Knapsack Problem,” International Journal of Computer Applications, vol. 16, no. 7, pp. 1-6, 2011.

ii. P. Chu dan J. Beasley, “A Genetic Algorithm for the Multidimensional Knapsack Problem,” Journal of Heuristics, vol. 4, no. 1, pp. 63-86, 1998.

iii. M. Varnamkhasti, “Overview of the Algorithms for Solving the Multidimensional Knapsack Problems,” Advanced Studies in Biology, vol. 4, no. 1, pp. 37-47, 2012.

iv. S. Khuri, T. Baeck dan J. Heitkötter, “The Zero/One Multiple Knapsack Problem and Genetic Algorithms,” dalam Proc. ACM Symposium on Applied Computing (SAC'94), New York, 1994.

v. J. Beasley, “Distributing test problems by electronic mail,” ournal of the Operational Research Society, vol. 4, no. 1, p. 1069–1072, 1990.

vi. P. Cappanera dan M. Trubian, “A Local-search-based Heuristic for the Demand-constrained Multidimensional Knapsack Problem,” INFORMS Journal on Computing, vol. 17, no. 1, pp. 82-98, 2005.

vii. D. Pisinger, “A Minimal Algorithm for the Multiple-choice Knapsack Problem,” European Journal of Operational Research, vol. 83, no. 2, pp. 394-410, 1995.

viii. A. Freville, “The Multidimensional 0-1 Knapsack Problem: An overview,” European Journal of Operational Research, vol. 155, no. 1, pp. 1-21, 20 November 2004.