Proceedings of Science and Mathematics
Faculty of Science, Universiti Teknologi Malaysia

Vol. 11, 2022, page 1 - 11

Simulation of Solving Visiting Route in Johor Based on Traveling Salesman Problem

Nor Syasya Aqilah Mohd Esa, Wan Rohaizad Wan Ibrahim*
Department of Mathematical Sciences, Faculty of Science, Universiti Teknologi Malaysia
*Corresponding author: wrohaizad@utm.my

Abstract
The purpose of this study is to investigate the application of Traveling Salesman Problem (TSP). TSP is a classical combinatorial optimization problem, which is simple to state but very difficult to solve. No current algorithms are available which can solve these problems in polynomial time, whichis the number of steps grow as a polynomial according to the size of the input. The traveling salesmanproblem involves a salesman who must make a tour of several cities using the shortest path available.The main objective of this research is to determine the best route for traveler should take to traverse through a list of locations and return to the start place. Therefore, a solution has been provided to solvethis TSP problem by using simulation of Tabu Search. Tabu Search enable traveler to find the optimal solution by visiting exactly once for each node. The optimization problem for this study is generated through a software of Microsoft Visual C++ programming to solve the TSP. Hence, the results indicate the shortest and the best route for travelerto traverse through a list of locations and return to the start place in Johor.

Keywords: Tabu Search; Optimization, Tabu List; Optimal solution; Shortest path.

1. Introduction

The travelling salesman problem (TSP) is an algorithmic problem tasked with finding the shortest route between a set of points and locations that must be visited. It is often used in computer science to find the most efficient route for data to travel between various nodes. Applications include identifying network or hardware optimization methods. It was first described by Irish mathematician W.R. Hamiltonand British mathematician Thomas Kirkman in the 1800s through the creation of a game that was solvable by finding a Hamilton cycle, which is a non-overlapping path between all nodes.

TSP has been studied for decades and several solutions have been theorized. The simplest solution is to try all possibilities, but this is also the most time consuming and expensive method. Manysolutions use heuristics, which provides probability outcomes. However, the results are approximate and not always optimal. Other solutions include branch and bound, Monte Carlo and Las Vegas algorithms.

Consider planning a route to visit multiple locations in Johor. This will result in a lot of distanceand time consuming without a proper planning. According to the number of target locations, planning a reasonably short path can be very complex and computationally expensive task. Given several target places in Johor along with the distance and the time taken for traveling from one to the other, what is the shortest path that a traveler should take to traverse through a list of locations in Johor andreturn to the start place.

In this research, the path planning will be evaluate using Tabu Search algorithm. The result of the best route for traveler to use to minimize the distance and the time taken to visit all locations in Johor will be simulated by computer programming.

This research aims to (1) determine the best route for traveler to use to minimize thedistance and the time taken to selected locations in Johor and (2) to simulate Tabu Search model in selecting the route by Microsoft Visual C++.

2. Literature Review

2.1. Travelling Salesman Problem

2.1.1. Travelling Salesman Problem

Algorithm for solving the TSP can be divided into two classes, which are heuristic algorithms and exact algorithms. Furthermore, TSP is an optimization problem that has various applications such asmachine sequencing, vehicle routing, scheduling, planning and logistics. These problems can be solved by using various approaches such as Tabu Search and Simulated Annealing.

2.1.2. Heuristic Algorithm

Heuristic is an algorithm that are problem-dependent technique that use a systematic procedure derived from relatively simple idea towards finding a good and better solution. Heuristic is a techniquethat find a solution not necessary to be optimal at a reasonable computational cost [1]. There is no guarantee on the global optimum although most of the solution found by heuristic algorithm resulted ingood promising result as the algorithm refers to find solution by 'trial and error' [2].

The following are known as heuristic algorithm examples that have been used in solvingTraveling Salesman Problem.

2.1.3. Simulated Annealing

Simulated Annealing (SA) is a probabilistic technique for approximating the global optimum of a given function and a metaheuristic to approximate global optimization in a large search space for an optimization problem.

Annealing procedure defines the optimal molecular arrangements of metal particles where the potential energy of the mass is minimized and refers cooling the metals gradually after subjected to high heat.

In general manner, SA algorithm adopts an iterative movement according to the variable temperature parameter which imitates the annealing transaction of the metals.

2.1.4. Tabu Search

The Tabu Search algorithm is to force an embedded heuristic from returning to recently visited areas of the search space, which is called as cycling. The plan of this approach is to maintain a temporary memory for all the changes of recent moves within the search space and preventing future moves fromundoing those changes.

2.1.5. Genetic Algorithms

A Genetic Algorithm is a randomized global search technique that solves problems by reproducing thenatural evolution of organism, generation after generation, by depending on the phenomena's heredity and law survives stated by Charles Darwin [3]. It generates a sequence of populations of candidate solutions to the underlying optimization problem by using a set of genetically inspired stochastic solution transition operators to transform each population of candidate solutions into a descendent population.

2.1.6 Nearest Neighbour Algorithm

The Nearest Neighbor algorithm (NN) is the first algorithm that has been introduced to solve TSP [2]It is also a simple heuristic algorithm based on greedy procedure. It starts tour by selecting random city and adds the nearest unvisited city to the last city in the tour until all cities are visited.

2.1.7 Ant Colony Optimization

This approach is driven by ant behavior searching for food. Ants will wander arbitrary and when the food is found, the ants will return to the colony by placing the pheromone trace. Basically, the ACO replicates the way ants promptly establish the shortest path between the nest and a food source [4].

2.2. C++ Programming

C++ Programming was created by Bjarne Stroustrup and this language was designed for the purposeof upgrading existing capabilities of C Programming with object-oriented programming features SIMULA-67. [5] However, C++ is a program that has more advanced features than C programming. C++ is a strong, efficient, and fast language and allows exception of handling and overloading functionality in C. [6] Therefore, C++ programming is used in solving TSP to have clear and feasible solution.

2.3. Terminology in Tabu Search

Table 1: Terminologies in Tabu Search

TERMINOLOGY	DEFINITION
A move	A transition from a current solution to its neighbouring solution.
An attribute	The elements that constitute the move.
Tabu list	A list of moves that are currently tabu (a list of forbidden exchanges to avoid cycling between thesame solutions endlessly).
Tabu list size	The number of iterations for which a recently accepted move is not allowed to be reserved.
Aspiration criterion	Criterion used to identify tabu restrictions that may be overridden,
Neighbourhood	The set if all possible neighbour solution that can be reached with one move.
Neighbourhood solution	One move from the current solution.
Forbidding strategy	The tabu condition that forbid a move from being reserved.
Freeing strategy	The conditions that allow a move to become nontaboo because either its tabu status has become not tabu or such a move satisfies an aspiration criterion.

3. Research Methodology

3.1. Traveling Salesman Problem Model

We need to find a route with minimum total distance travelled for salesman who need to visit variousof cities. However, the city travelled must be visited only once which started and ended form a homelocation. Suppose that there are 25 places in Johor that a traveler needs to visit and move in the shortest path and must cover all the selected places, visiting each location only one and back at the place of origin (depot). Therefore, our goal is to find the best route with minimal distance.

3.2. Basic Form of Tabu Search

The most basic form of the tabu search algorithm consists of the following:

1) Generating an initial solution.
2) Generating neighboring solution of the current solution.
3) A function that measures each neighboring solution.
4) A tabu list – to prevent cycling and leads the search to unexplored regions of the solutionspace.
5) An aspiration criterion.

3.3. The Data

The data can be collected by listing all the locations so that the coordinate can be read to be plot onthe map.

3.4. Plotting the map

The location that has been collected will be plot onto the map to get clear coordinates. Figure 1shows the location of the places to visit in Johor.

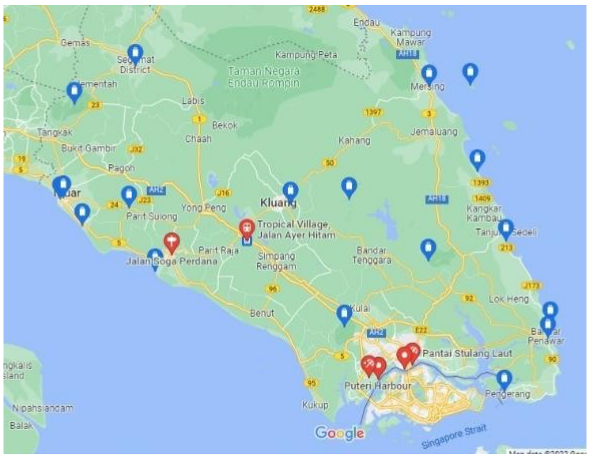


Figure 1 The location of the places to visit in Johor

3.5. Initial Solution

Following the procedure of Tabu Search in solving the Traveling Salesman Problem, the traveler mustbegin their journey from the depot, which is located at Pantai Stulang Laut, Johor Bahru and then move to the next location. An initial solution is generated by choosing the nearest location, for the purpose of solving TSP.

3.6. Initial Solution with Euclidean Distance

The simplest TSP which is symmetric, involving the traveler and the distance between two places.This type of TSP is known as Euclidean Distance, where the total cost for theproblem is given by the total distance travelled by the traveler.

To calculate the distance travelled, the formula of the ordinary Euclidean distance between twoplaces as.

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \quad i,j = 0,1,2,3,4,\ldots n \qquad (1)$$

where (x_i, y_i) is the coordinate for the location number one and (x_j, y_j) is the coordinate for locationnumber two with n is the total number of locations in the system.

3.7. Searching Method

The movement of the traveler is defined by randomly selecting two paths and swap it. The differencebetween each method is on how to accept the new solution after changing path or move has been done. After changing the path, the route will be different, and we need to calculate the new distance.

3.8. Microsoft Visual Studio C++

To obtain the solution for TSP using Tabu Search as the basic heuristic requires thousands of iterations and it is impossible to be calculated manually. Hence, this software is implemented as oneof the tools. Figure 2 shows the interface of the program that has been used to solve the problem.

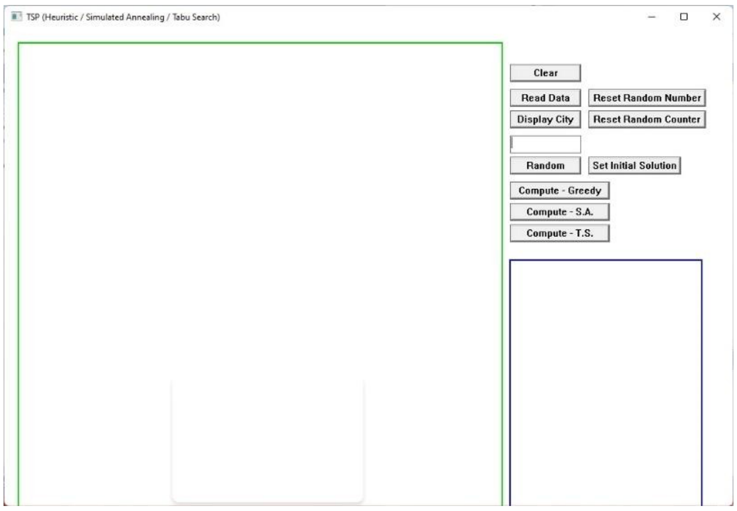


Figure 2 The interface of the program

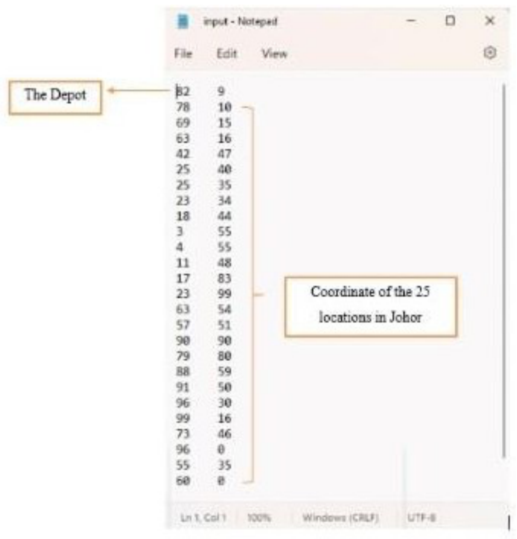
Based on Figure 2, the field mark (1) shows the space for displaying the locations and route travelled by the traveler starting from the depot. Next, field mark (2) displays the information of calculations such as the cost and number of iterations involved. Field mark (3) is the control buttons. Apart from that, field mark (4) displays compute button for Greedy Method, Simulated Annealing as wellas Tabu Search. The function for each button is described in the Table 2 below:

Table 2: Button Function

Button	Function
Clear	To reset the program, clear screen and all variables involved.
Read Data	To read the data from the input file.
Display City	To reset the route and display only thelocation of the cities
Reset Random Number	To reset the random number
Reset Random Counter	To reset the number counter
Set Initial Solution	To compute an initial solution
Compute - Greedy	Compute the solution for Greedy Method
Compute – S.A	Compute the solution using Simulated Annealing
Compute – T.S	Compute the solution using Tabu Search

3.9. Input Data

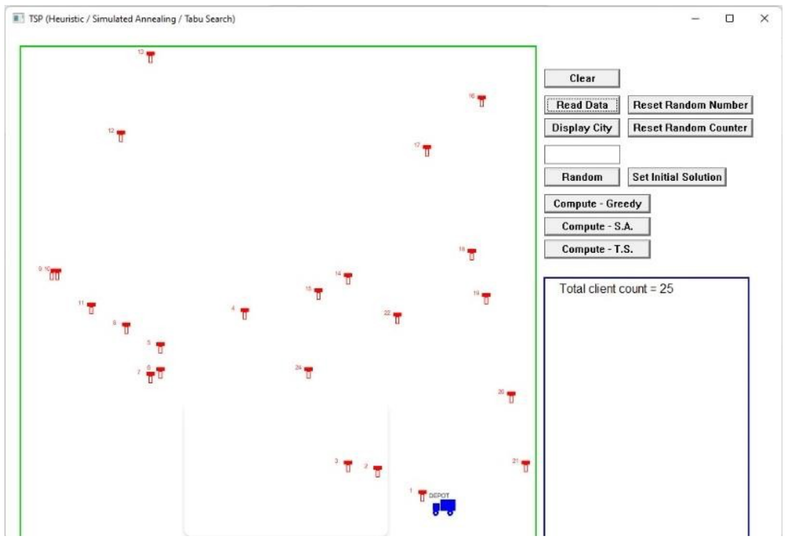
This program read the data from a text file named “input.txt”. The text file consists of the coordinate for each location selected in Johor with range number between 0 to 100 for both x-axis and y-axis. Each coordinate for x-axis and y-axis must entered by pair in one line and separated by using spaceor tab. Figure 3 displays the coordinates of all the locations in Johor. Figure 3 shows the coordinate for each location in input file.


The Depot
Coordinate of the 25 locations in Johor
input - Notepad
File Edit View
Ln 1, Col 1 100% Windows (CRLF) UTF-8

```
82	9
78	10
69	15
63	16
42	47
25	40
25	35
23	34
18	44
3	55
4	55
11	48
17	83
23	99
63	54
57	51
90	90
79	80
88	59
91	50
96	30
99	16
73	46
96	0
55	35
60	0
```

Figure 3 The coordinate for each location in input file

The program will read and display all the coordinates by clicking “Read Data” button asshown in Figure 4.



Total client count = 25

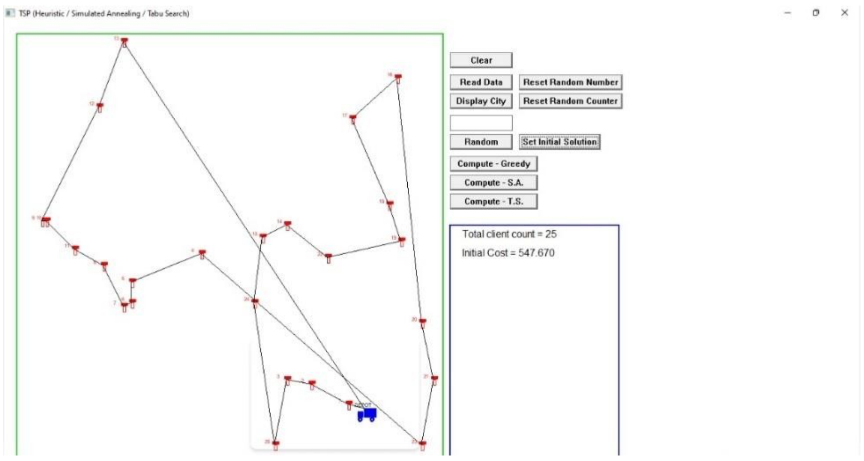
Figure 4 The location of each coordinate on display window

The information about the coordinate will be stored and used to calculate the distance between each location. Figure 5 displays the coding used to calculate the distance between thelocation.

```
double CMainFrame::calcdist(CPoint p1,CPoint p2)

{

        double L;

        L=sqrt(pow((double)p1.x-
p2.x,2)+pow((double)p1.y-p2.y,2));

        return L;
```

Figure 5 C++ coding to calculate distance between locations

Initial solution must be obtained before we can proceed to the next step so that the comparisonbetween initial cost and cost from Tabu Search method can be made. Figure 6 illustrates the initial solution route travelled in Johor by the traveler.



```
Total client count = 25
Initial Cost = 547.670
```

Figure 6 Initial solution route travelled in Johor

The output file content is shown as below:

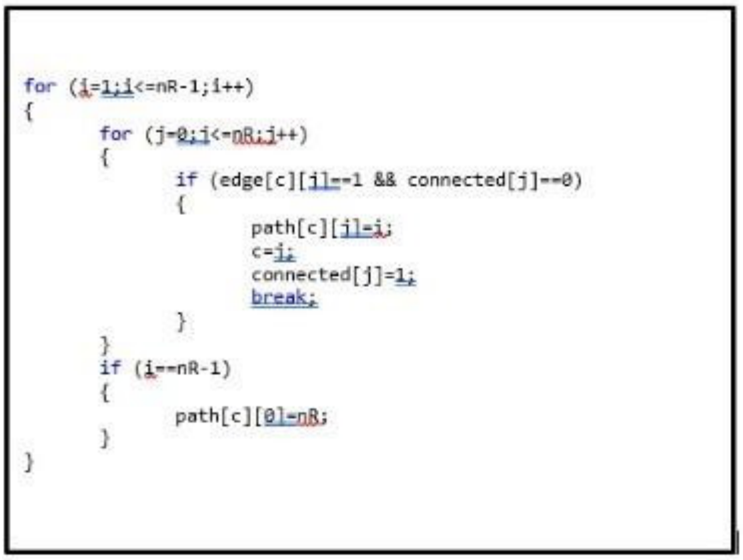
```
DEPOT - 1 - 2 - 3 - 25 - 24 - 15 - 14 - 22 - 19 - 18 - 17 - 16 - 20 - 21 - 23 - 4 - 5 - 6 - 7 - 8 - 11 - 10 - 9 -
12 - 13 - DEPOT

Initial Cost = 547.67
```

Figure 7 shows the coding to calculate the initial cost.

```
for (i=0;i<=nR-1;i++)
        {
                for (j=0;j<=i;j++)
                {

        distance[i][j]=calcdist(node[i],node[j]);
                        distance[j][i]=distance[i][j];
                }
        }


        for (i=0;i<=nR-1;i++)
        {
                visit[i]=0;
                connected[i]=0;
                for (j=0;j<=nR;j++)
                        edge[i][j]=0;
        }


        visit[0]=1;
        double lowest;
        int select;


        //c = currentnode
        c=0;
        totalcost=0.0;
        dc.MoveTo(LPixel[c]);
        connected[c]=1;
        ofp << "DEPOT - ";
```

Figure 7 C++ code to calculate initial cost

By clicking the “Compute – T.S” button, it will activate the program to solve the problem usingTabu Search algorithm. Figure 8 shows the C++ code for assigning the path number.



```
for (i=1;i<=nR-1;i++)
{
        for (j=0;i<=nR;j++)
        {
                if (edge[c][j]==1 && connected[j]==0)
                {
                        path[c][j]=i;
                        c=j;
                        connected[j]=1;
                        break;
                }
        }
        if (i==nR-1)
        {
                path[c][0]=nR;
        }
}
```

Figure 8 C++ code for assigning path number

4. Results and discussion

4.1. Run Results

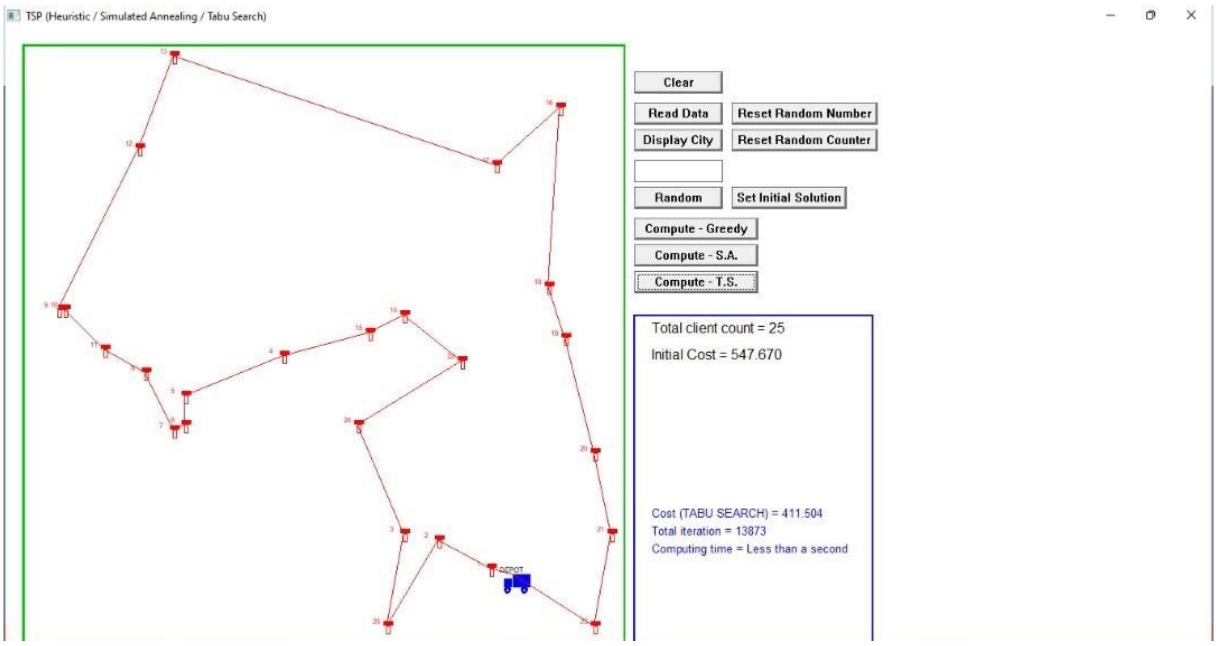
By using Microsoft Visual C++, further runs have been repeated for five times using the same initialsolution and the result has been recorded as shown in Table 3 below.

Table 3: Results of five runs for Tabu Search

Run	Number of Iteration	Cost
Initial		547.670
1st	13964	419.107
2nd	13946	424.944
3rd	13933	411.504
4th	13931	424.313
5th	13873	411.504
Best Solution		411.504

From the table above, the best value obtain is at 13873 iterations. After a few runs, the value generated is optimal. However, it is not the best solution for the problem

4.2. Result



```
Total client count = 25
Initial Cost = 547.670

Cost (TABU SEARCH) = 411.504
Total iteration = 13873
Computing time = Less than a second
```

Figure 9 The optimal solution

DEPOT - 1 - 2 - 25 - 3 - 24 - 22 - 14 - 15 - 4 - 5 - 6 - 7 - 8 - 11 - 10 - 9 - 12 - 13 - 17 - 16 - 18 - 19 - 20 - 21 - 23 - DEPOT

Total distance (cost) = 411.504

Total iteration = 13873

Total computation time = Less than a second

Table 5: The actual route and total distance

Actual Route	**Total Actual Distance**
Pantai Stulang Laut – Zoo Johor – Puteri Harbour – Pulau Kukup – Legoland – Gunung Pulai – Kota Tinggi Waterfall – Gunung Belumut – Gunung Lambak – Tropical Village Ayer Hitam – Hutan Lipur Soga Perdana – Pantai Minyak Beku – Perigi Batu Pahat – Ladang Nanas Kampung Parit Tengah – Pantai Leka – Muar Clock Tower – Masjid Jamek Sultan Ibrahim – Gunung Ledang – Batu Hampar Flower Garden – Mersing Muzeum – Pulau Besar – Tanjung Leman Beach – Jason Bay (Teluk Mahkota) – Tanjung Balau – Desaru Beach – Bukit Pengerang – Pantai Stulang Laut	1112.46 km

Conclusion

After the simulation, the cost (total distance) is 411.504 with total iteration of 13873. Meanwhile the actual route and total distance for this problem is 1112.46km. The objective of this study to determine the best route for traveler to use to minimize the distance and the time taken to visit all locations in Johor is succeeded. Therefore, it is proven that Tabu Search can be used to solve a TravelSalesman Problem.

Acknowledgement

The researcher would like to thank all people who have supported the research and Directorate of Research and Community Service, Directorate General of Strengthening for Research and Development, Ministry of Research, Technology and Higher Education of Malaysia, Universiti Teknologi Malaysia which has funded this research.

References

[1] Reeves, C., 1995. Modern Heuristic Techniques for Combinatorial Problems. Mc-Greenhill

[2] Halim, A. H., and I. Ismail, (2017). Combinatorial Optimization: Comparison of Heuristic Algorithms in Travelling Salesman Problem. Archives of Computational Methods in Engineering.

[3] El-Sherbeny, N. A. (2010). Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods. Journal of King Saud University – Science, 22(3), 123-131. Doi:https://doi.org/10.1016/j.jksus.2010.02.002

[4] Prakasam, A., & Savarimuthu, N. (2016). Metaheuristic algorithms and probabilistic behaviour: a comprehensive analysis of Ant Colony Optimization and its variants. *Artificial Intelligence Review,* 45(1), 97-130.

[5] O'Regan, G. (2018). Introduction to Programming Languages. In *World of Computing* (pp. 155- 178): Springer.

[6] Thornton, K. (2003). Libsequence:a C++ class library for evolutionary genetic analysis. *Bioinformatics,* 19(17), 2325-2327.

[7] Laporte, G. (1991). *The traveling salesman problem: An overview of exact and approximatealgorithms*. Centre de recherche sur les transports, Université de Montréal.

[8] Wu Berbeglia, G., Laporte, G., & Cordeau Jean-Franç ois. (2010). *A hybrid tabu search and constraint programming algorithm for the dynamic dial-a-ride problem*. CIRRELT.

[9] Bortfeldt, A., Gehring, H., & Mack, D. (2002). *A parallel tabu search algorithm for solving thecontainer loading problem*. Fachbereich Wirtschaftswiss., Fernuniv.

[10] Dorigo, M., & Stü tzle Thomas. (2004). *Ant colony optimization*. MIT press.

[11] Fiechter, C.-N. (1994). A parallel tabu search algorithm for large traveling salesman problems. *Discrete Applied Mathematics*, *51*(3), 243–267. https://doi.org/10.1016/0166-218x(92)00033-i

[12] Glover, F., & Laguna, M. (1997). Tabu search background. *Tabu Search*, 1–24. https://doi.org/10.1007/978-1-4615-6089-0_1

[13] Greco, F., & Gerace, I. (2008). The symmetric circulant traveling salesman problem. *Traveling Salesman Problem*. https://doi.org/10.5772/5581

[14] Helshani, L., & Ramollari, E. (2016). Comparative study of genetic algorithm and simulatedannealing algorithm. *Proceedings of The 5th International Virtual Scientific Conference*. https://doi.org/10.18638/ictic.2016.5.1.269