Traveling Salesman Problem

Namki Kim
Department of Robotics Engineeing
Daegu Gyeongbuk Institute of
Science and Technology
Daegu, Korea
ttorangs3@dgist.ac.kr

Jinmin Kim
Department of Robotics Engineeing
Daegu Gyeongbuk Institute of
Science and Technology
Daegu, Korea
rlawlsals@dgist.ac.kr

***Abstract*—Traveling salesman problem (TSP) is one of the problems which had studied in optimization and used as a typical example among the problems of computational complexity theory. In this project, we defined the traveling salesman problem (TSP) with the Miller-Tucker-Zemlin (MTZ) method among subtour removal methods for solving TSP. In the problem, there exist variable u, which make the ranking of the point. If $x_{ij} = 1$, the new u_j is greater than the previous ranking of u_i. By the MTZ constraints, we can exclude the point we selected before. To implement the optimization problem, we used cvxpy module with python language. We picked 7 cities and input them to our optimization problem written by python. As the result of our setting, the optimal path and minimum distance were derived. To validate the result, we compared them with exhaustive search method. The comparison shows that the two results were same and that the exhaustive search method has much computing time than using cvxpy module.**

Keywords—transportation; discrete optimization; NP-hard

I. INTRODUCTION

Traveling salesman problem is one of the classical NP-hard problems. The task of the problem is to find the shortest route for a traveling salesman to visit all the cities once and once, and return to the starting city. This type of problem and its solution has applications in other areas, such as the drone station [1], automated terrain navigation [2], pickups and deliveries [3] and so on.

As the simplest way to solve a problem, there is a way to solve it with the exhaustive search. The exhaustive search method is to start from each city and visit all cities to select the lowest value. If the number of cities is small, solving it in this method will find the most accurate solution. However, even if the number of cities is over 10, it will take a lot of computational time to calculate all combinations. The exhaustive search method is inefficient because it calculates all solutions that are unlikely to be optimal solutions. There are various optimization techniques for combinatorial search, all of which aim to reduce the number of solutions by preventing them from searching for solutions that are unlikely to be optimized.

Because of the complexity of the problem, many techniques for finding optimal solutions have been proposed. Chisman [4] proposed a method of improving the computational speed through the process of clustering points and obtaining solutions between the clusters. Visiting order restriction theorem, proved by Flood [5], gives a necessary condition for the shortest tour of TSP on the Euclidean plane by using the convex hull. Recently, learning-based methods have been proposed. Shoma [6] proposed algorithms using CNN for 2D Euclidean TSP and a method to apply reinforcement learning for this model. In addition to these studies, various institutions are studying TSP. In this project, we solve the problem by selecting one of the methods to solve the TSP. Using the computer programming, we define and solve the problem to find optimal solutions. Then, it is confirmed that the solution obtained by the method is the same as the solution by exhaustive search. Finally the computation speed was compared by measuring the execution time of the two methods.

II. METHODS

The TSP problem can be modeled with the integral linear program. n is the number of cities, and c_{ij} is the distance or cost from node i to j. The objective function minimizes the distance of each salesman's routes. However, when we solve it, all nodes are visited once, but we will get a solution that has the subtour. If the subtour exists, it will never be possible to visit all nodes due to the closed loop of the subtour.

Here we will apply the most compact subtour elimination method, called Miller-Tucker-Zemlin (MTZ) method [7]. The optimization problem is completed by adding the MTZ constraint that removes subtour. The MTZ constraint is as follows:

$$u_i - u_j + 1 \le (n-1)\left(1 - x_{i,j}\right) \qquad 2 \le i \ne j \le n$$

$$u_1 = 1, \qquad 2 \le u_i \le n \qquad 2 \le i \le n$$

u_i is an integer variable that can have a value from 1 to the number of nodes n. The meaning of this variable is (the ranking of visits to the i-th node) - (the number of visits to the i-th node) . $u_1 = 1$ means visiting node 1 first. u_i functions as a constraint only when $x_{ij} = 1$. If $x_{ij} = 1$, then $u_i + 1 \le u_j$, making the ranking of j greater than the ranking of i — e.g., if the next node is added to the tour, u_i is increased and all nodes are inserted into one tour.

III. SYSTEM MODELS

The objective function and constraints of the problem are defined. The non-convex optimization formulation of a general tsp is defined as follows:

Objective function:

$$\min \sum_{i=1}^{n} \sum_{j \neq i, j=1}^{n} c_{i,j} x_{i,j}$$

Constraints:

$$\sum_{i=1, i \neq j}^{n} x_{ij} = 1 \qquad j = 1, \ldots, n$$

$$\sum_{j=1, j \neq i}^{n} x_{ij} = 1 \qquad i = 1, \ldots, n$$

To eliminate the error caused by subtour, Miller-Tucker-Zemlin (MTZ) formation was additionally set as a constraint as described in the method. Therefore, the system model of TSP finally defined is as follows:

Objective function:

$$\min \sum_{i=1}^{n} \sum_{j \neq i, j=1}^{n} c_{i,j} x_{i,j}$$

Constraints:

$$\sum_{i=1, i \neq j}^{n} x_{ij} = 1 \qquad j = 1, \ldots, n$$

$$\sum_{j=1, j \neq i}^{n} x_{ij} = 1 \qquad i = 1, \ldots, n$$

+ Miller-Tucker-Zemlin (MTZ) formulation

$$u_i - u_j + 1 \leq (n-1)\left(1 - x_{i,j}\right) \qquad 2 \leq i \neq j \leq n$$

$$0 \leq u_i \leq n \qquad 2 \leq i \leq n$$

IV. Implementation

A. Using cvxpy

The library, cvxpy is an open-source Python library for convex optimization problems. We programmed using this along the flow chart.

1) Flow chart

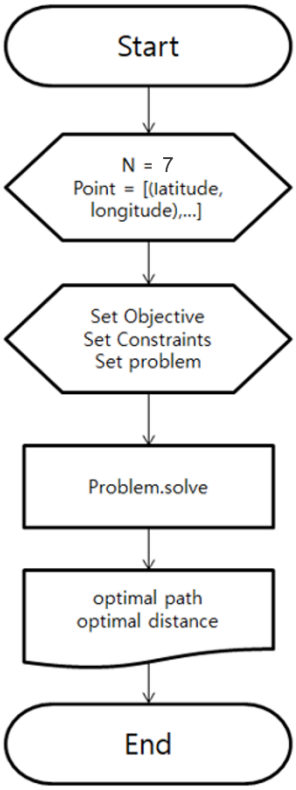

Fig. 1 Flow chart of our optimization problem

2) Result

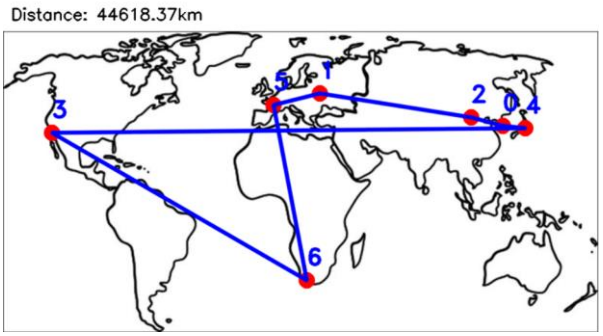

Fig. 2. Visual representation of the result. Optimal distance is shown at the top.

Figure 2 is a visual representation of the optimal path and distance obtained using cvxpy. The results obtained by cvxpy are as follows:

(cities = 0, 1, 2, 3, 4, 5, 6)
Optimal path: 0 => 2 => 1 => 5 => 6 => 3 => 4 => 0
Optimal distance: 44618.37 km

B. Validation

Ground-truth was obtained using the exhaustive search algorithm to confirm that the answer obtained using cvxpy was true, and the two results were compared. In addition, to compare the computational efficiency of the cvxpy and exhaustive search method, the computing time was compared with the increase of the number of cities.

1) Result

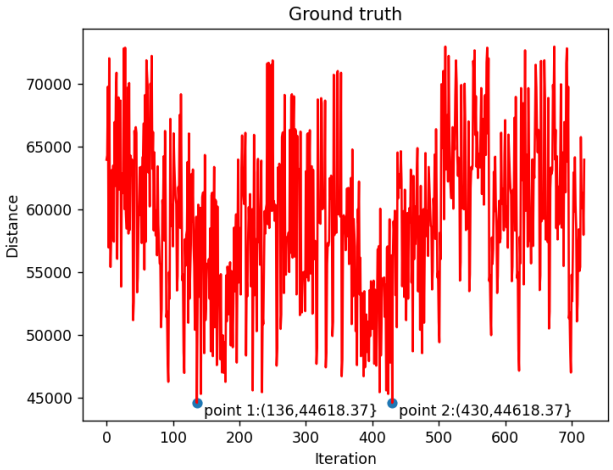

Fig. 3. The distance of all paths calculated in exhaustive search method.

Figure 3 is a graph that the distance of all paths calculated in exhaustive. Point 1 and point 2 is indicate the point when it is a minimum distance. The results are as follows:

(cities = 0, 1, 2, 3, 4, 5, 6)

Minimum path : 0 => 2 => 1 => 5 => 6 => 3 => 4 => 0

Minimum distance : 44618.37 km

The results from exhaustive search and cvxpy derive the same path and distance.

2) Computing time

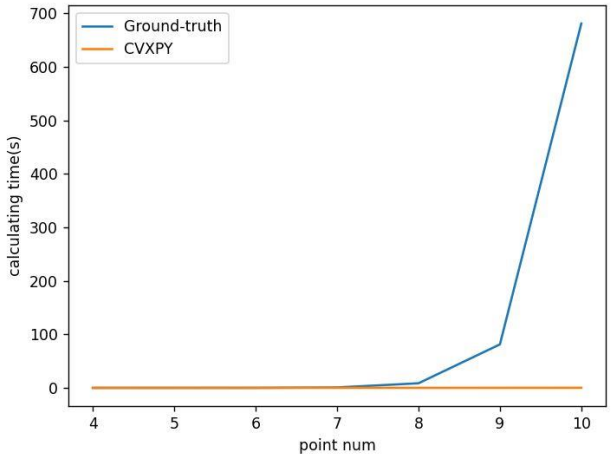

Fig. 4. Comparing the computing time between cvxpy and exhaustive search method.

As shown in the graph, when the number of cities is 8, the computing time of the two is similar. However, from the number of cities is 9, the exhaustive search method starts to increase sharply. Therefore, from the time N is 9, it can be seen that it is efficient to use a solution algorithm different from CVXPY other than a full search method.

V. Conclusion

In this project, we defined the traveling salesman problem (TSP) with the Miller-Tucker-Zemlin (MTZ) method among subtour removal methods for solving TSP. In the problem, there exist variable u, which make the ranking of the point. If $x_{ij} = 1$, the new uj is greater than the previous ranking of ui. By the Miller-Tucker constraints, we can exclude the point we selected before. To implement the optimization problem, we used cvxpy module with python language. We picked 7 cities and input them to our optimization problem written by python. As the result of our setting, the optimal path and minimum distance were printed. To validate the result, we compared them with exhaustive search method. The comparison shows that the two results were same and that the exhaustive search method has much computing time than using cvxpy module.

We picked 7 cities, and if the number of cities increases, the results of cvxpy and exhaustive search method may be different. To validate this, we have to compute with the exhaustive search method. But it will take too much time to calculate all those combinations. If there is a heuristic method that can solve an optimal solution in a short time, if not optical, it is more reasonable to use that method.

Through this project, we have experienced defining optimization problems and obtaining the optimal solution through computer programming. If we research the optimization problem in the future, this project will help us solve the problem.

References

[1] S. Kim and I. Moon, “Traveling salesman problem with a drone station,” IEEE Trans. Syst. Man, Cybern. Syst., vol. 49, no. 1, pp. 42–52, 2019.

[2] E. Serin, S. Adali, and S. Balcisoy, “Entropy assisted automated terrain navigation using Traveling Salesman problem,” Proc. VRCAI 2011 ACM SIGGRAPH Conf. Virtual-Reality Contin. its Appl. to Ind., vol. 1, no. 212, pp. 41–48, 2011.

[3] A. S. Elgesem, E. S. Skogen, X. Wang, and K. Fagerholt, “A traveling salesman problem with pickups and deliveries and stochastic travel times: An application from chemical shipping,” Eur. J. Oper. Res., vol. 269, no. 3, pp. 844–859, 2018.

[4] J. A. Chisman, “The clustered traveling salesman problem,” Comput. Oper. Res., vol. 2, no. 2, pp. 115–119, 1975.

[5] M. M. Flood, “The traveling salesman problem”, Operations Research, Vol. 4, pp. 61-75, 1956.

[6] S. Miki, D. Yamamoto, and H. Ebara, “Applying Deep Learning and Reinforcement Learning to Traveling Salesman Problem,” Proc. - 2018 Int. Conf. Comput. Electron. Commun. Eng. iCCECE 2018, pp. 65–70, 2019.

[7] C. E. Miller, R. A. Zemlin, and A. W. Tucker, “Integer Programming Formulation of Traveling Salesman Problems,” J. ACM, vol. 7, no. 4, pp. 326–329, 1960.