Solving Multiple Traveling Salesman Problem using K Means Clustering and Mixed Integer Programming

An Integrated Approach

Navneet Verma  
 Computer Science  
Kennesaw State University

Video Link: <https://youtu.be/Dh7Q93TAST0>

ABSTRACT

In this research paper, we explore an efficient algorithm for multiple Traveling Salesman Problem (m-TSP) using an approach which combines K Means clustering algorithm and Mixed Integer Programming (MIP). The Multiple Traveling Salesman problem is an NP hard problem which relates to generation of minimum cost round trip tours for multiple salesmen visiting several cities in their territory. Our novel approach has the promise of producing even workloads for the salesmen while ensuring fast performance.

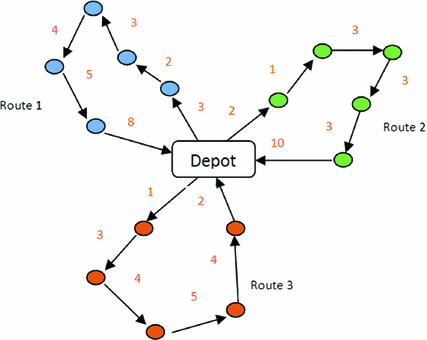
**KEYWORDS**

Traveling Salesman Problem, K Means Clustering, Genetic Algorithm, Mixed Integer Programming

1 INTRODUCTION

The Traveling Salesman Problem (TSP) is the problem of finding the minimum cost route given a list of cities and distances between each pair of cities, such that each city is visited exactly once, and each salesman returns to the origin.

The objective of m-TSP optimization is to assign a tour of disjoint city sets to each of the m salesmen such that the combined tour cost is minimized. The m-TSP is harder problem than TSP since it requires optimal city set assignment and subsequent city traversal order determination for each of the m salesmen as shown in the figure below.



**Figure 1: Routes for 3 salesmen**

TSP is a very challenging optimization problem which falls in the category of NP-hard problems i.e., the set of problems which are at least as hard as the problems in set NP. The associated decision problem, however, is NP-complete. Such problems can be decided by non-deterministic Turing Machine (NDTM) in polynomial time. Such an NDTM guesses various permutations of cities non-deterministically and halts by accepting a string where cost of such a tour is less than a certain cost *C*. Please note, when m=1, the m-TSP is the same as TSP. It is a task of difficult order to achieve efficiency and optimality at the same time. The class of m-TSP solvers can be divided into following two categories:

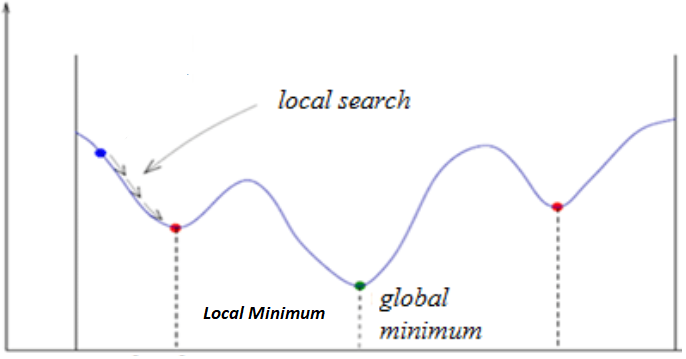
1. Optimization solvers such as mixed integer programming (MIP) which find the minimal tour cost and work efficiently on many (but not all) problem instances.
2. Heuristics techniques such as genetic algorithms, alone or combined with K-means clustering, simulated annealing, ant colony optimization (ACO) etc., which find good enough answers efficiently but are not guaranteed to return minimal tour cost.

The goal of our research is to find more effective solution for m-TSP by combining MIP algorithm with K-means clustering. In this approach, cities will be grouped into m clusters based on distance matrix to first solve the city assignment problem. MIP algorithm will then be applied to these disjoint clusters to find the correct traversal order. This approach is better than using existing approaches using genetic algorithms because MIP solvers do not converge prematurely unlike genetic algorithms. Since territory of a salesman is often defined by cities near each other, K Means clustering can solve the city assignment problem without burdening the MIP solver. We feel such an effective solution will be helpful for many practical applications such as, crew allocation for repair jobs, allocating school buses to various areas in a school district, job scheduling for multiple parallel production lines and routing vehicles for delivery of liquified natural gas (LNG).

The organization of this paper is as follows. We first present the background material on above solution techniques and review of current research in section 2. We then show that any m-TSP problem can be transformed into an equivalent TSP problem by introduction of m artificial cities. Section 3.1 of this paper presents such a formulation using mixed integer programming (MIP). In section 3.2, we then present an improvement on this formulation by integrating K-means clustering with MIP. Section 4 presents results for formulations presented in section 3.1 and 3.2. Finally, we present our conclusion in section 5 along with directions for further research.

2 LITERATURE REVIEW

The current space for effective solutions for m-TSP abounds in approaches based on genetic algorithms. A genetic algorithm (GA) is a metaheuristic inspired by Darwin’s theory of natural selection. A genetic algorithm works by first randomly generating solution candidates and then improving upon them by mutation, crossover, and selection. The selection is based on a fitness function defined by the algorithm. For a given m-TSP instance, in the first phase, genetic algorithm randomly assigns cities in m groups, thus creating an initial population or seed for the algorithm. In the second phase, it generates candidate permutations of the cities in each assigned group and by the evolutionary process improves upon the solutions. [2] modified this approach by using a constrained optimization solver for the second phase. Although use of constrained optimization solver is guaranteed to find optimal permutation in a group, selection of initial population i.e., initial division of cities in disjoint groups determines the success of this algorithm. Using a large initial population may worsen the run time while using a small population may affect solution quality. To circumvent the issue related to initial population, some researchers in [5] have used K-means clustering to find groups of cities closer to each other to construct an initial population. In the second phase, however, they use GA to construct an optimal tour by using crossover, mutation, and selection within each group (intra group) and among the groups (inter group). The inter group evolution is indeed a step in the direction of integral optimization. However, a GA based solutions have the tendency for premature convergence [4]. Figure 1 shows that once GA based algorithm reaches a local minimum, it may terminate prematurely as it may find its solution value increase before it finds a global minimum. Another important determinant in the success of GA based solution is degree of mutation and crossover. It is a challenging task to determine precise degree of crossover and mutation operators to achieve optimal solution [4].



**Figure 2**: Local vs global minimum

A Mixed Integer Programming (MIP) algorithm uses branch and bound strategy on decision variables whose domain is limited to integers to solve combinatorial optimization problems such as m-TSP. A MIP solver is very similar to goal programming used in languages such as PROLOG. The goal or objective of a MIP solver is often a cost or profit function which needs to be minimized or maximized while respecting the rules which are mathematical inequalities referred to as constraints. Mathematically this idea can be stated as follows.

where:

c is the cost vector; x is the integer decision variable and A is the constraint matrix. All the constraints in A are in the form of inequalities such as

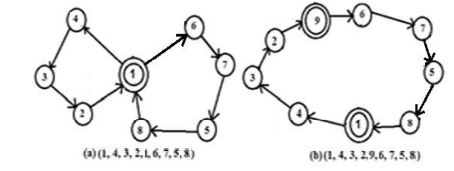
MIP solvers guarantee that algorithm will not get stuck in local minimum. However, this accuracy comes at the expense of exponential run time which is affected by number of constraints in the formulation. To circumvent the increase of run time, iterative approaches are suggested in [1]. In this approach, solver begins by setting up a relaxed version of the problem by omitting some constraints. After a solution to the relaxed problem is obtained, it can be verified in polynomial time for any constraint violations. If omitted constraints were not violated by solution, the solution is accepted as the minimum cost tour. Otherwise, omitted and violated constraints are added to the mix and problem is solved again. This iterative method is called cutting plane method and was proposed by Gomory [6]. In 2004, the Concorde TSP solver, a modern implementation of this idea, was used to find the optimal tour for 24,978 cities in Sweden. To use this method for solving m-TSP, we use the problem transformation technique proposed in [3].

3 METHODOLOGY

In our approach, we utilize the idea of reduction used in complexity proofs to first reduce the m-TSP to TSP. Mathematically, a mapping reduction R from language L1 to language L­2 is defined as follows:

)

To reduce the m-TSP to TSP, we assume a single salesman, but we add m-1 additional cities located at distance 0 from the main depot. [3] presents the following example for two salesmen with 8 cities. In the left diagram we see the optimal tours of 2 salesmen. The right diagram represents the same tour with artificial city 9 added to the mix. Please note that main depot 1 and artificial city 9 are located at distance 0 from each other. The tour strings {1,4,3,2,1} and {1,6,7,5,8,1} in language L1 get transformed to string {1,4,3,2,9,6,7,5,8,1} in language L2.



**Figure 3:** Problem reduction

The advantage of this reduction is that greatly simplifies the implementation of the solution since we do not need to add any extra constraints to make distinction between m tours. In the next section, we show how we apply this reduction algorithm using Mixed Integer Program.

3.1 MIXED INTEGER PROGRAM

The mixed integer program is characterized by objective function and constraints. In this formulation, our goal is to minimize overall cost of all the m tours in terms of distance travelled. Let us assume cij represents the distance between city i and j and let xij be a binary variable which assumes value 1 when city j immediately follows city i in the tour and otherwise stays 0. We can state the formulation of mixed integer program as provided in [7] as follows:

Minimize

is the cost of travel between cities i and j

Exit Constraint: Each city i must exit at exactly one j

Entry Constraint: Each city j must have exactly one entry point at some i

This formulation, however, results in sub-tours. For example, in the transformed problem, we can get tours {1,4,3,1} and {2,9,6,7,5,8,2} as shown.Diagram

Description automatically generatedDiagram

Description automatically generated

**Figure 4**: Sub tours

Sub tour elimination constraints can be added to the original formulation to avoid such solutions. However, there can be a lot of subtours and adding elimination constraints beforehand can significantly affect the problem size and performance. Hence, we follow the lazy approach in which we detect the subtours and iterate again by adding subtour elimination constraints. To eliminate above subtours there must be one arc from set {1,4,3} to set {2,9,6,7,5,8,2} and another arc from set {2,9,6,7,5,8,2} to set {1,4,3}. This requirement can be stated as follows:

, salesman should enter at least one city in set 1 and exit in set 2

, salesman should enter at least one city in set 2 and exit in set 1

These constraints are added as needed to eliminate the subtours.

MIP solver uses branch and bound search algorithms to explore correct combinations of binary variables xij to find the optimal tour. As the number of cities get large, search space could become prohibitively expensive. To prune the search space, we present an improvement in the next section.

3.2 INTEGRATED APPROACH

The formulation detailed in previous section can be improved if we are able to rule out some entry and exit points. In that case, xij = 0 and MIP solver can prune the search space greatly, thus improving the runtime substantially. Since each salesman is normally assigned a territory consisting of closely existing locations, K means clustering can be used to group cites in m groups. The steps to form clusters are:

Step 1: Choose m random points as cluster centers called centroids.

Step 2: Assign each city to the closest cluster by calculating Euclidean distance to the centroid.

Step 3: Identify new centroids by taking average of the (x, y) coordinates assigned to each cluster as shown below:

, N is the number of points in the cluster

, N is the number of points in the cluster

Step 4: Keep repeating step 2 and 3 till convergence is achieved.

After creating the clusters, we mandate that cities in clusters P and Q do not enter or exit each other. Mathematically, we can state this constraint as follows:

,

,

-Coding for integrated approach using K-means algorithm.

One key characteristic of our implementation is that we cluster by angles instead of Euclidean distance. We first transform all the co-ordinates to angles and then project them on an imaginary circle of radius R and then use the K Means algorithm outlined above. Since K Means algorithm is very sensitive to the initial choice of centroids, we used furthest point heuristic and repeats to obtain even distribution of points using approach outlined in [9].

To test our algorithms, we used the 48-city dataset provided by [8].

4 RESULTS

The iterative sub-tour elimination algorithm mentioned in section 3.1 converges quickly as the graph below shows. Initially, MIP solver produces roughly half the number of tours as the total number of cities. This indicates that round trip tours between each pair of cities are preferred initially. However, sub tour elimination constraints work efficiently to eliminate such unacceptable tours.

**Figure 5**: Sub tour convergence

To verify the correctness of our approach, we compared the results produced by our algorithm for 1 salesman with the dataset provided on website [8] and found that our results match with the published results on the website. The following graph shows the tour produced by our algorithm for 1 salesman.

**Figure 6**: Tour for 1 salesman

However, applying the same algorithm for 2 salesmen after problem transformation produces uneven tours. As the dashed tour divider line in the graph below shows that first salesman visits only 1 city with the itinerary 1-8-49, where city 49 is the virtual city created with the same co-ordinates as city 1.

**Figure 7**: Tour for 2 salesmen

Incorporating K Means clustering with the MIP algorithm does increase the tour cost slightly but it produces more even workload between the two salesmen as the dashed tour divider line in the shows in the graph below.

**Figure 7**: Tour for 2 salesmen with K Means

We found that the performance of m-TSP algorithm with K Means clustering is reasonably good for this dataset. The sub-tour elimination constraints worked efficiently to force faster convergence to the optimum. We performed 2 runs of the algorithm and found that convergence time for 48 city 2 salesmen problem was around 3-4 seconds on a 2.60 GHz, 6 core machine with 32 GB of memory.

|  |  |
| --- | --- |
| **Run #** | **Time (seconds)** |
| Run 1 | 4.1 |
| Run 2 | 3.2 |

5 CONCLUSION AND FUTURE WORK

Our conclusion is that integrated approach combining K Means clustering with Mixed Integer Programming is an effective algorithm to solve m-TSP problem with around 50 cities in the dataset. K Means clustering produces even workload for each salesman and addresses the city assignment problem effectively, thus freeing MIP solver to focus on city order traversal. This division of labor produces excellent performance and realistic results. Our approach can be applied with some modifications for capacitated vehicle routing problem (VRP) as well. The problem transformation idea in our approach is an innovative approach to solve m-TSP using the same formulation as TSP. However, for larger datasets, this approach may not be scalable. We can use divide and conquer strategy using K Means clustering and solve the sub-problems in parallel. This approach has the potential to scale for larger datasets. More studies need to be performed with larger datasets to understand the scalability and accuracy of this approach.

REFERENCES

[1] Elaine Rich. 2019. Automata, Computability and Complexity. Pearson Education Inc.

[2] Jessica M Rudd, Andrew M Henshaw, Lauren Staples, Sanjoosh Akkineni, Lin Li and Joe DeMaio.2020. Genetic Algorithm Guidance of a Constraint Programming Solver for the Multiple Traveling Salesman Problem (2020). Published and Grey Literature from PhD Candidates*.23.*<https://digitalcommons.kennesaw.edu/dataphdgreylit/23>

[3] Maha Ata Al-Furhud and Zakir Hussain Ahmed. 2020. Genetic Algorithms for Multiple Traveling Salesman Problem. (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 11, No. 7.

[4] S Katoch, S.S. Chauhan and V Kumar. 2021. A review on genetic algorithm: past, present, and future. *Multimed Tools Appl* **80**, 8091–8126 (2021). <https://doi.org/10.1007/s11042-020-10139-6>

[5] Lizhuang Tan, Yan-Yan Tan, Guo-Xiao Yun and Chao Zhang. (2016). An improved genetic algorithm based on k-means clustering for solving traveling salesman problem. [https://doi.org/10.1142/9789813200449\_0042Conference](https://doi.org/10.1142/9789813200449_0042) Name:ACM Woodstock conference

[6]Ralph E. Gomory. 1958. Outline of an algorithm for integer solutions to linear programs. Bulletin of the American Mathematical Society, 64 (1958), 275-278

[7] Nemhauser and Wolsey. 1999. Integer and combinatorial optimization. Wiley Interscience

[8] John Burkardt.2019. Data for Traveling Salesman Problem. <https://people.sc.fsu.edu/~jburkardt/datasets/tsp/tsp.html>

[9] Pasi Franti, Sami Sieranoja. 2019. How much K Means can be improved by better initialization and repeats. Journal of Pattern Recognition, Elsevier. <https://doi.org/10.1016/j.patcog.2019.04.014>

Conference Short Name:WOODSTOCK’18

Conference Location:El Paso, Texas USA

ISBN:978-1-4503-0000-0/18/06

Year:2018

Date:June

Copyright Year:2018

Copyright Statement:rightsretained

DOI:10.1145/1234567890

RRH: F. Surname et al.

Price:$15.00