Travelling sales man problem

Siddharth Kashyap
210122058
Chemical Science and Technology
Indian Institute of Technology,
Guwahati, Assam, India

Shreya Anand
210122056
Chemical Science and Technology
Indian Institute of Technology,
Guwahati, Assam, India

***Abstract*— This study presents a method to solve the Traveling Salesman Problem (TSP) and its variation employing GAMS and MATLAB. This study emphasizes the use of metaheuristic technique on NP-hard problem to obtain feasible suboptimal answers for large testcases. It also sheds light on comparison between different metaheuristic technique in context to this problem and its variation. It also talks employing mathematical programming technique on single Travelling salesman problem in search of global optimum solution. The study will also adhere to two more variation which will be explored in detail.**

Keywords—travelling salesman problem, minimum distance tour.

I. INTRODUCTION

The Traveling Salesman Problem (TSP) holds notable significance in various fields, offering valuable solutions to optimize diverse processes.

- In drug discovery, TSP algorithms streamline compound screening, determining the most efficient order for testing candidates. Additionally, TSP aids in molecular conformation studies, guiding the exploration of energetically favourable arrangements of atoms within molecules. Whether optimizing synthesis routes in combinatorial chemistry, planning NMR experiments for structural elucidation, or facilitating cluster analysis in mass spectrometry, TSP serves as a versatile computational tool, contributing to the efficiency and precision of various chemical applications.

- The Traveling Salesman Problem (TSP) finds a practical application in the context of drilling printed circuit boards (PCBs). When connecting a conductor on one layer to a conductor on another layer or positioning integrated circuit pins, holes must be drilled through the board. These holes can vary in size. However, drilling holes with different diameters consecutively requires the machine's head to move to a tool box and switch drilling equipment, which is time-consuming. Therefore, a more efficient approach involves selecting a specific diameter, drilling all holes with that diameter, changing the drill, and repeating the process for the next diameter. Consequently, this drilling challenge can be conceptualized as a series of TSPs.

- TSPs are also used in vehicle routing where a city has n mailboxes that need to be emptied daily within a specified time frame. The objective is to determine the minimum number of trucks required for this task and the most efficient schedule for collections using this optimal number of trucks. As another illustration, envision a situation where n customers have specific demands for certain commodities, and a supplier needs to fulfill all these demands using a fleet of trucks. The challenge is to devise a plan for assigning customers to trucks and establishing a delivery schedule for each truck. This plan must ensure that the capacity of each truck is not exceeded while minimizing the overall travel distance.

Past-work

The problem was first devised in 1930 and was considered mathematically to solve a school bus routing problem. Ray Fulkerson and Selmer M. Johnson from the RAND Corporation, who expressed the problem as an integer linear program and used the cutting plane method find the solution.

II. PROBLEM STATEMENT

The traveling salesman problem is defined within the framework of a network of cities where every pair of cities is connected. We are given N (number of cities) and an adjacency matrix of N*N dimension containing information about distance between each pair of cities. This is due to the assumption each pair of cities is connected to each other. There is a salesmen start his journey from depot city and will end at depot city. Here we have assumed the depot city will be the city indexed at 1. The salesman has allotted N+1 day to complete its tour where he has to visit each city exactly 1 and return to the depot city on (N+1)th day. Due to time constraint every city will be visited exactly once and all the N city has to be visited. Adhering to condition at any day from the current city he can move to any other city. Total distance travelled by the salesmen will be sum of length of all path taken. The main Objective is to minimize the total distance travelled to complete the journey.

III. SOLUTION STRATEGY

1) SINGLE TRAVELLING SALESMAN PROBLEM

A. Matlab – single objective

Matlab is used to metaheuristically minimize the travelling distance in solving the single objective of the Traveling Salesman Problem. Four algorithms, TLBO, PSO, GA,ABC, and DE, were utilized for this purpose. A comprehensive comparison of these different algorithms was conducted to evaluate their respective performances for this problem.

- Decision Variable- A row matrix of size N (number of cities) where each ith element denotes the city visited on ith day. Example for N=4 one feasible solution can be [1 2 4 3]. This denotes starting from 1(the depot city) we visit 2 on the

2nd day 4 on the 3rd day and 3 on the 4th day. Note- we have not included the return to depot city in the decision variable as it will be calculated explicitly and the first element of the decision variable will always be 1. Number of decision variable is N.

- Adhering to the condition of visiting each city once we have to keep a visited matrix of N*1 dimension initially containing all 0 and as we visit the city we increment the value at that index. This will help to keep which cities have been visited to avoid them visiting again.
- Due the significance of decision variable every element has to be and integer varying from 1 to N. It is possible set of decision variable generated by metaheuristic technique does not adhere to above rule. So we employ correctness, every time we first round the decision variable and bound it. Here we use corner bounding technique where we bound the elements if needed to corner points that is 1 and N if needed.

```
x(i)=round(x(i));
x(i)=min(numCities,x(i));
x(i)=max(1,x(i));
```

- If any element in the decision variable is repeated we employ correctness by randomly checking any city that is not bounded and assigning to it. This ensures every city appears in the decision variable exactly once.

```
while visited(x(i))==1
    x(i)=randi(numCities,1);
end
```

- Evaluation of Objective function- Due to the nature of the decision variable denoting the city travelled in sequential order. There are a total of N+1 trips distance travelled I each trip is looked up with help of adj matrix and added to total.

```
curVisCity=x(i);prevVisCity=x(i-1);
TotalDistance=TotalDistance+adj_mat(curVisCity,prevVisCity);
```

- The last trip would be from city visited at Nth day back to the depot city.

```
TotalDistance=TotalDistance+adj_mat(x(numCities),x(1));
```

Results

- Now we have out file to calculate fitness function ready we employ 5 major metaheuristic techniques – TLBO, DE, ABC, PSO and GA. Parameter for the above algorithm can be

```
Np=50;T=50;  %setting problem parameters for TLBO
Pc=0.8;F=0.85; %setting problem parameters for DE
w=1.7;c1=1.5;c2=1.5; %setting problem parameters for PSO
limit=100;%setting problem parameters for ABC
etac=20;etam=21;pc=0.8;pm=0.3;%setting problem parameters for GA
```

tweaked but for simplicity we have it at general values.

- We kept Population size Np=50, number of iterations T=100 and number of runs NRuns=5. The best fitness values obtained by respective techniques are shown below.

3434.00	3447.00	3536.00	3064.00	3161.00
3363.00	3430.00	3474.00	3184.00	3067.00
3291.00	3296.00	3490.00	3305.00	2956.00
3458.00	3446.00	3587.00	3158.00	3051.00
3381.00	3497.00	3581.00	3299.00	3161.00

- All the techniques were then compared using statistical table which include minimum, maximum, mean, standard deviation and median

3291.00	3296.00	3474.00	3064.00	2956.00
3458.00	3497.00	3587.00	3305.00	3161.00
3385.40	3423.20	3533.60	3202.00	3079.20
65.32	75.43	51.37	101.64	85.88
3381.00	3446.00	3536.00	3184.00	3067.00

of values obtained by each in 5 runs respectively.

- On observation it is inferred that Genetic algorithm produced the lowest minimum and maximum values among all techniques in 5 runs. So the minimum travelling distance using metaheuristic techniques is 2956 units.

Conclusion and Discussion

The best value obtained by the metaheuristic is 2956, however, it is not guaranteed it is the global optimum. Note- If we increase the population size Np, and number of iterations T we might obtain a better value. Though metaheuristic are simple to use it has one of the major drawbacks of uncertainty.

B. GAMS – single objective

The problem is also solved using mathematical programming techniques with the help of GAMS. Mixed integer programming has been solved to get the global optimal solution. This solution will be used to compare the efficiency of metaheuristic techniques applied earlier. To solve a problem using GAMS mathematical models are developed and then these models are incorporated to GAMS as code.

Miller–Tucker–Zemlin formulation-

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

$$x_{ij} \in \{0,1\} \qquad i,j = 1,\ldots,n;$$

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

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

$$u_i - u_j + n x_{ij} \leq n-1 \qquad 2 \leq i \neq j \leq n;$$

$$1 \leq u_i \leq n \qquad 1 \leq i \leq n.$$

- C(i,j) is adjacency table containing distance between every pair of cities.
- n is total number of cities.
- u(i) a variable keep track of order of visit for all n. if u(i) < u(j) this implies city i is visited before city j.
- x(i,j) is a binary variable of N*N dimension
- F to store the total distance travelled along the journey.

$$x_{ij} = \begin{cases} 1 & \text{the path goes from city } i \text{ to city } j \\ 0 & \text{otherwise} \end{cases}$$

Number of decision variable is N+N*N+1.

Results

```
Proven optimal solution
MIP Solution:          1387.000000     (45237 iterations, 3299 nodes)
Final Solve:           1387.000000     (0 iterations)

Best possible:         1387.000000
Absolute gap:             0.000000
Relative gap:             0.000000
```

On running the above formulation we global optimum is obtained at 1387. This distance is significantly lower than what obtained in the metaheuristics technique about 53% lower. This lays down the importance of mathematical programming technique though metaheuristics are best for black box problem it might not be able to search for global optimum for large test cases.

C. MATLAB – multi-objective

The above problem we discussed was single objective involving only minimization of travelling distance. Now we introduce a slight variation that every city the salesman visit he will sell his product to the buyer. The number of products sold will be equal to the number of residents living in that city at that particular day. This problems also supports the possibility that number of buyers in that city will change with each day. So here we introduce population matrix of N*N dimension stating the number of people are there in on ith city on jth day. If salesman visits that city he will sell the product to all those present in that city.

So here we first evaluate the maximum profit that the salesman can make and then we minimize travelling distance keeping the profit intact.

Evaluation of maximum profit-

Due to the nature of the decision variable denoting the city travelled in sequential order.

```
numPeople=pop_mat(visitingCity,i);  %number of buyer in that city on ith perticular day
Revenue(i)=SP*numPeople;            %revenue numbe rof people buying the product
visited(visitingCity)=1;            %mark that city as visited
```

After maximum profit calculation, it was passed in another round of metaheuristic technique to minimize the travelling distance. The whole process was automated.

```
maxProfit=-min(bestfitness); % determining max profit and using it calucate for multi OBJ
probMult=@TSP_Mult;          % function handle for maxPofit with min distance to travel
for i=1:NRuns

    rng(i,"twister");
    [bestSolC(i,:),~,bestfitnessC(i,1),~,~]=TLBO_MultiObj(Np,T,lb,ub,probMult,maxProfit);
end
```

During the calculation of travelling distance if those set of decision variables produced lower profit than maximum profit a hefty penalty was levied which eliminated the that solution.

```
profit=sum(Revenue);
profit_violation=0;
if profit < maxProfit
    profit_violation=10^15;
end

f=TotalDistance+profit_violation;
```

Result

The minimum travelling distance with maximum profit was 3446 greater than the distance calculated as single objective. Using metaheuristics such problems can be solved where one objective is of more importance as illustrated.

D. C++ single Objective

Using language such as C++ we use the Backtracking algorithm which is simply trying all possible ways. This algorithm is computationally expensive and won't work for large test cases. So to counter this problem we use Backtracking with Memoization.
Memoization is a process by which we store the already computed results in cache memory which we may require later to compute other subproblems. Since it involves trying all possible combinations this solution guarantees global optimum still it is computationally very expensive for large testcases.

2) *Multiple travelling salesmen problem*

IV. Problem Statement

The M-traveling salesman problem is defined within the framework of a network of cities where every pair of cities is connected. We are given N (number of cities) , M (number of Salesmen), maxVisit (maximum number of cities a salesman can visit), and an adjacency matrix of N*N dimension containing information about the distance between each pair of cities. This is due to the assumption each pair of cities is connected to each other. Every salesman will start his journey from Depot City and will end at Depot City. Here we have assumed the depot city will be the city indexed at 1. It is not mandatory for a salesman to visit all cities but all cities have to be visited by exactly one salesman. If a salesman returns back to the depot city it means the journey of that salesman has been completed and he cannot visit any other city. So for correct inputs, we need to make sure maxVisit* M >=N. The total distance travelled is the sum of the distance travelled by each salesman in their journey. Our main objective is to minimize the total distance travelled by a salesman.

V. Solution Strategy

A. MATLAB- single Objective

- Decision Variable- A row matrix of size M*N (number of cities) where each ith row denotes the ith salesmen and jth column denotes the city visited by ith salesman on jth day. The number of decision variables is N*M.
- Adhering to the condition of visiting each city once we have to keep a visited matrix of N*1 dimension initially containing all 0 and as we visit the city we increment the value at that index. This will help to keep which cities have been visited to avoid them visiting again. It also helps to keep track of whether all city are visited or not by any salesmen just once.
- All corrections which were employed in single TSP are employed here, in addition, if a salesman returns to depot city, he cannot visit any other city anymore.

```
if x(i+(j-1)*numDays)==1
    TotalDistance=TotalDistance+adj_mat(x(i-1+(j-1)*numDays),1);

    for k=i+1:numDays
        x(k+(j-1)*numDays)=1;
    end
    break;
end
```

- It may happen all salesman completes their tour early without all city being visited for that penalty is imposed.

```
for i=1:numCities
    %all city has to be visited exaclty once
    if visited(i)==0
        visitViolation=10^5;
    end
end
```

- It may happen a salesman visits more cities than he was allowed to, for that penalty is imposed.

```
if prevVisCity~=curVisCity
    numVis=numVis+1;
end
if numVis>maxVis
    salesM_violation=10^5;
end
```

- Evaluation of Objective function- Due to the nature of the decision variable denoting the city travelled in sequential order of days, distance can be calculated as

```
curVisCity=x(i);prevVisCity=x(i-1);
TotalDistance=TotalDistance+adj_mat(curVisCity,prevVisCity);
```

- Total distance will be sum of distance travelled by each salesman.

Results

•Now we have out file to calculate fitness function ready we employ 5 major metaheuristic techniques – TLBO, DE, ABC, PSO and GA. Parameter for the above algorithm can be tweaked but for simplicity we have it at general values.

```
Np=50;T=50;  %setting problem parameters for TLBO
Pc=0.8;F=0.85; %setting problem parameters for DE
w=1.7;c1=1.5;c2=1.5; %setting problem parameters for PSO
limit=100;%setting problem parameters for ABC
etac=20;etam=21;pc=0.8;pm=0.3;%setting problem parameters for GA
```

4302.00	4583.00	100000000000000000000.00	3789.00	3598.00
3772.00	4552.00	4039.00	3862.00	3824.00
4022.00	4321.00	4357.00	3869.00	3468.00
3884.00	3928.00	4345.00	3549.00	3525.00
4031.00	4284.00	4115.00	3705.00	3641.00

- We kept Population size Np=50, number of iterations T=100 and number of runs NRuns=5. The best fitness values obtained by respective techniques are shown below.
- Set of decision variable corresponding to best fitness can be obtained.
- All the techniques were then compared using statistical table which include minimum, maximum, mean, standard deviation and median of values obtained by each in 5 runs respectively

3772.00	3928.00	4039.00	3549.00	3468.00
4302.00	4583.00	100000000000000000000.00	3869.00	3824.00
4002.20	4333.60	20000000000000000000.00	3754.80	3611.20
198.79	263.17	44721359549995794432.00	132.79	136.27
4022.00	4321.00	4345.00	3789.00	3598.00

On observation, it is inferred that Genetic Algorithm, produced the lowest minimum and maximum values among all techniques in 5 runs. The minimum travelling distance using metaheuristic techniques is 3468 units.

B. GAMS single Objective

The problem is also solved using mathematical programming techniques with the help of GAMS. Mixed integer programming has been solved to get the global optimal solution. This solution will be used to compare the efficiency of metaheuristic techniques applied earlier. To solve a problem using GAMS mathematical models are developed and then these models are incorporated to GAMS as code.

Formulation-

$$x_{ij} = \begin{cases} 1 & \text{If arc (i, j) is used in the tour,} \\ 0 & \end{cases}$$

$$\text{Minimize} \qquad \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$$

$$\sum_{j=2}^{n} x_{1j} = m$$

$$\sum_{j=2}^{n} x_{j1} = m$$

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 2, \ldots\ldots, n$$

$$\sum_{j=1}^{n} x_{ij} = 1, \; i = 2, \ldots\ldots, n$$

$$u_i - u_j + p x_{ij} \leq p - 1 \text{ for } 2 \leq i \neq j \leq n$$

- C(i,j) is an adjacency table containing the distance between every pair of cities.
- n is the total number of cities.
- u(i) a variable keep track of the order of visit for all n. if u(i) < u(j) this implies city i is visited before city j.
- x(i.j) is a binary variable of N*N dimension
- F to store the total distance travelled along the journey.

Result

Using the above formulation the result was inconclusive as it couldn't reach the optimal solution.

VI. Conclusion and Future Work

In this report, we saw different ways to solve the Travelling salesman problem (TSP) and discussed about their outcomes and shortcomings. It also shed light upon different variations of TSP those variations were inspired by examples illustrated in course CL-643 IIT Guwahati of 2023.
Ultimately, it can be concluded that in the world of optimization, both metaheuristics and mathematical programming technique has their own stands, and their application totally depends on the problem description.

A metaheuristic algorithm is a search method crafted to discover a robust solution to a complex optimization problem that proves challenging to solve optimally or is of black-box type. The essence of these algorithms lies in uncovering near-optimal solutions, relying on imperfect or incomplete information, especially in the practical context of resource constraints in the real world.
Mathematical programming is used where it is possible to derive formulation of the problem and the optimality of the solution cannot be compromised.

Future Work

- Parallel and Distributed Computing: Advancements in parallel and distributed computing resulted in researchers to explore techniques to solve large-scale TSP instances more efficiently by distributing the computation across multiple processors or systems
- Multi-Objective TSP: Extending TSP to handle multiple conflicting objectives simultaneously. This involves optimizing not only for the shortest route but also considering other criteria such as cost, time, or environmental impact to attain to SDG goals.
- Applications in Emerging Technologies: TSP can also be applied in conjunction with emerging technologies like blockchain, Internet of Things (IoT), or quantum computing to solve complex optimization problems efficiently.

Acknowledgment

We are thankful to course Prakash Kotecha Associate Professor IIT Guwahati and TA's for giving successfully directing and conducting course CL-643.

References

- Traveling Salesman Problem: An Overview of Applications, Formulations, and Solution Approaches --Rajesh Matai1, Surya Prakash Singh2 and Murari Lal Mittal (2010) - https://www.researchgate.net/publication/221909777_Traveling_Salesman_Problem_an_Overview_of_Applications_Formulations_and_Solution_Approaches