IOSR Journal of Mathematics (IOSR-JM)
e-ISSN: 2278-5728, p-ISSN: 2319-765X. Volume 12, Issue 5 Ver. V (Sep. - Oct.2016), PP 80-84
www.iosrjournals.org

A Tabu Search Approximation for finding the Shortest distance using Traveling Salesman Problem

Md. Mamun-Ur-Rashid Khan[1], Md. Asadujjaman[2]
Department of Mathematics, Dhaka University, Dhaka – 1000, Bangladesh

Abstract: *In this research, we present the basic concepts of Tabu search method for optimizing problem such as Traveling Salesman Problem (TSP). The main purpose of this work is to understand the symmetric TSP, solve this problem by using Tabu search method to find the shortest distance with small search space and computational needs and then using a computer program (using MATLAB software) to solve large scale Symmetric TSP. Finally, we find the shortest distance to visit 64 district headquarters of Bangladesh.*
Keywords *Traveling Salesman Problem (TSP), Hamiltonian Cycle, Tabu Search*

I. Introduction

In 1986 *Fred Glover*[5] proposed a new approach called Tabu search. The basic idea of Tabu search is to do local search whenever it counts a local optimum value by allowing non- improving moves, cycling back to previously visited solutions is prevented by the use of memories, called Tabu lists (short-term memory), that record the recent history of the search.

Tabu search doesn't stop at the first local optimum when no improvement is possible; the possible solution in the neighborhood is always selected, even if it is worse than the current solution. This allows it to search more solutions from the feasible region. To avoid cycling, a Tabu list is used to store solutions that cannot be considered at next iteration. Typically, the Tabu list cannot contain all of the previously visited solutions, because of the computational requirements.

In this research symmetric TSP is solved by Tabu search method to find an optimal solution with small search space and computational requirements with shortest distance.

II. Traveling Salesman Problem (TSP)

The ***Traveling salesman problem (TSP)*** [2] seeks to minimize the cost or distance or time of the route for a salesman to visit all the cities exactly one and return where he starts. It is very easy to describe but difficult to solve because it requires finding a ***Hamiltonian cycle*** (Figure 2) of minimum cost, distance or time. This problem perhaps has been the most well-studied combinatorial optimization problem.

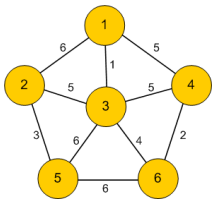

Figure 1: A weighted graph

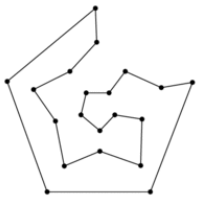

Figure 2: Hamiltonian cycle

TSP can be represented by a complete weighted graph g = (v, e) (Figure 1) with v being vertices, representing the cities and e is the set of edges fully connecting the vertices. Each assigned a value d_{ij} which is the length of the edge (i, j) is the distance between cities i and j. In the symmetric problem d(i, j) = d(j, i). Where the pair (i, j) is the same as the pair (j, i).

Symmetric TSP

The symmetric traveling salesman problem[1, 4] is defined on an undirected network in which travel is allowed in either direction of each undirected arc or edge. The problem is to find an undirected tour with a minimal traveling distance. To formulate the symmetric TSP, one must note that $c_{ij} = c_{ji}$ and $y_{ij} = y_{ji}$. Thus, each edge can be identified by a single index k, each decision variable by y_k, and each distance by c_k. As a result, the symmetric problem can be formulated with only one-half the number of 0-1 variables. To find a tour in an undirected network g (e, a), one must select a subset of undirected arcs such that every node y is connected to exactly two of the undirected arcs selected. The symmetric problem requires additional constraints to eliminate all sub tours, but not any tour.

Formulation of the symmetric TSP as a 0-1 integer program:

Step 1

Input parameters: a list of undirected arcs indexed by k and their associated distances, c_k

Decision variables: whether or not each undirected arc k is in the tour (y_k= 0 or 1)

Constraints: each city in the tour must have exactly two undirected arcs incident to it, and all sub tours must be eliminated

Objective: total distance traveled in a tour must be minimal

Step 2

Let E be the set of all undirected arcs, E_j be the set of all undirected arcs connected to city j, and E_S be the set of all undirected arcs connecting the cities in any proper subset S. Also let y_k = 1 if undirected arc k∈E is in the tour, and y_k = 0 otherwise. Then the symmetric TSP can be formulated as a 0-1 integer program[2]:

Minimize $\frac{1}{2}\sum_{j=1}^{n}\sum_{k\in E_j} c_k\, y_k$

Such that $\sum_{k\in E_j} y_k = 2$ for all cities j=1, 2, … , n

$\sum_{j\in E_S} y_j = |S| - 1$ for all $|S|$= 2, 3…,n-2

$y_j = 0 \text{ or } 1$ for all $j \in E$

The Cost matrix is of the form

$$\begin{pmatrix} \infty & c12 & c13 & & \dots & \dots & c1k \\ c21 & \infty & c23 & \cdots & \dots & \dots & c2k \\ \dots & \dots & \infty & & \dots & \dots & \dots \\ \dots & \vdots & & \ddots & & \vdots & \\ \dots & \dots & \dots & & \infty & \dots & \dots \\ \dots & \dots & \dots & \cdots & \dots & \infty & \dots \\ ck1 & ck2 & ck3 & & \dots & \dots & \infty \end{pmatrix}$$

where, c_{ij}= c_{ji}, c_{ii}= ∞, i =1, 2, … , k

III. The Tabu Search Method

Tabu search[8, 10] is an iterative technique which uses local or neighborhood search procedure. The iteration continues until some restriction or stopping criteria is satisfied. To explore regions of the search space that would be left unexplored by the local search procedure and by doing this escape local optimality, Tabu search modifies the neighborhood structure of each solution as the search progresses. The basic form of the Tabu search algorithm consists of the following[8]:

1. Generating an initial solution.

2. Generating a neighboring solution of the current solution.

3. A function that measures each neighboring solution.

4. A Tabu list in order to prevent cycling and leads the search to unexplored regions of the solution space.

5. An aspiration criterion.

Algorithm of Tabu Search for finding Shortest Distance[8]:

Step 1: Initialization:

Input the number of cities n, and their pair wise distances, the stopping criteria (maximum number of iterations etc), Set tabu tenure t, Initialize tabu list is empty.

Step 2:

Find best solution x* with objective function z* at the same time is considering as current Solution x as follows:

Let, x be the current solution. x* = x the best solution.

z* = z(x), where z* = summation of distance for x solution.

Step 3: Iteration:

While stop criterion is not satisfied Continue.

* Identify all the neighborhood solution by using the move operation between two cities randomly (swapping two cities) of the current solution. Each x ∈ X associated with a neighborhood N(X) ⊂ X.

Step 3.1:

Select best acceptable move from N(x) by choosing the solution has minimum z(x) from this is transforming x into x' ∈ N(X) and objective function value transforms to z(x') and add its mark to the tabu list.

Check x' is Tabu:

IF No Move to step 3.2

Else step 3.3

Step 3.2: Perform exchange:

x = x', z(x) = z(x').

If z(x) < z* then

z* = z(x), x* = x.
End if
Go to step 4

Step 3.3: Check x' is the goal
IF Yes Do step 3.2
Else
Check Neighbor set.

Step 4:
Record tabu for the current move in tabu list, i.e. update the tabu list by change the contents of tabu list as following:
* If solution not tabu add solution to tabu list by change the position of city from zero to t. For example if the problem consist of 5 cities and solution that resulted from (swap city 3 & city 2) is not tabu the tabu list must be [0 0 t 0 0] in the next iteration the tabu tenure decrease by one.
End

Step 5: Result: solution x* has the length of tour that must be minimum of all determined solutions, with objective function value z*.

Initial Solution[8]
The initial configuration specifies where the search begins in the search space. To generate the initial solution we used Nearest Neighborhood algorithm as follows:
1. Sort all edges.
2. Select the shortest edge and make the Hamiltonian cycle without creating a cycle.

Neighborhood Structure[8]
The entire feasible route that can be configured from the current solution is called Neighborhood solution. In neighborhood search, each solution $x \in X$ has an associated set of neighbors, $N(x) \subset X$, called the neighborhood of x, each solution $x' \in (x)$ can be reached directly from x by an operation called Move. And x is said to move to x' when such an operation is performed. Normally Neighborhood solutions are made by small changes to the current solution. There are many possible ways of selecting the best neighbor including improving of objective function, considering a subset of the neighbors and selecting the best of the set, evaluating the whole set of neighboring solutions and selecting the best in terms of the objective functions. In our problem we use second method for selecting the best neighbor.

Swapping
Exchange two cities in the problem.

Aspiration Criterion
In order to overturn the tabu list when there is a good tabu move, aspiration criterion is used, the tabu move is accepted if it produces better solution than the best obtained so far.

Termination criterion
The most commonly used stopping criteria in tabu search are:
1. Terminate the search after number of iterations.
2. After some number of iterations without an improvement in the objective function's value.
Here we use the second criteria are used for stopping criteria.

Example:
There are C1, C2, C3, C4, C5 the set of all cities[8].
The distance between each city to another in table (1)
The solution can be written as 2→5→1→4→3→2. An itinerary that begins
And ends at the same city and visits each city once is called a tour.

	C1	**C2**	**C3**	**C4**	**C5**
C1	∞	132	217	164	158
C2	132	∞	290	201	79
C3	217	290	∞	113	303
C4	164	201	113	∞	196
C5	158	79	303	196	∞

Table: 1

Suppose the initial solution to our problem is the shown in figure (1): Begin from first city return to first city

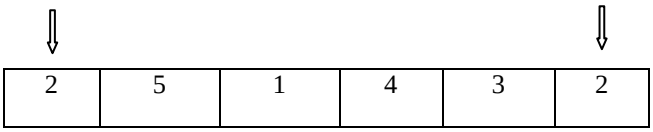

Figure (1)

The ordering in figure (1) specifies that city 2 is placed in the first position, followed by city 5, etc. Tabu search methods operate to identify the neighborhood that can be reached from any current solution. Identifying moves that lead from one solution to the next. In our problem, a swap exchanges the position of two cities as illustrated in figure (2):

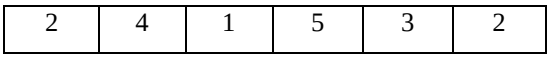

Figure (2) swap of cities 4 & 5

Therefore, the complete neighborhood of a given current solution consists of 6 adjacent solutions that can be obtained by such swaps. In here the tabu tenure t was taken equal 3. The number of iterations tabu move is considered to remain tabu.
Table 2 shows the iteration:

It.	Solution Path	Swap	Current Solution	Best Solution
0	2→5→1→4→3→2	1↔5	804	804
1	2→1→5→4→3→2	1↔5	889	804
2	2→5→1→4→3→2	1↔4	804	804
3	2→5→4→1→3→2	3↔5	928	804
4	2→3→4→1→5→2	3↔4	786	786
5	2→4→3→1→5→2	1↔4	768	768
6	2→1→3→4→5→2	3↔4	737	737
7	2→1→4→3→5→2	-	791	737

Table (2)

Optimal solution of the problem is 737.

IV. Solution of Large Scale Problem Using MATLAB Program:

Given the distances between every pair of 64 districts[11] of Bangladesh. A salesman wants to travel the all 64 cities starting from Dhaka and return to Dhaka. Here we need to find the minimum distance for Traveling.

	Dhaka (1)	Bagerhat (2)	Bandarban (3)	..	..	Thakurgaon (64)
Dhaka (1)	**0**	178	316	..	..	407
Bagerhat (2)	178	**0**	437	..	..	536
Bandarban (3)	316	437	**0**	..	..	720
.....	.....	.....	.....	**0**	..	.....
.....	.....	.....	.....	..	**0**	.....
Thakurgaon (64)	407	536	720	..	..	**0**

Initial solution:
Current Tour =

1 → 42 → 38 → 18 → 43 → 8 → 20 → 37 → 62 → 61 → 28 → 46 → 39 → 21 → 59 → 63 → 60 → 7 → 40 → 25 → 17 → 56 → 32 → 29 → 47 → 14 → 64 → 50 → 54 → 44 → 49 → 30 → 24 → 34 → 22 → 41 → 19 → 2 → 27 → 52 → 23 → 5 → 6 → 51 → 4 → 33 → 58 → 9 → 31 → 48 → 16 → 12 → 10 → 55 → 26 → 3 → 13 → 35 → 53 → 15 → 11 → 36→ 57 → 45 → 1

Initial tour distance = 5189 (Km.).

Iteration	Swap	Current Solution	Best Solution
0	-	5189	5189
1	44↔45	5096	5096
2	50↔14	5016	5016
3	44↔15	4951	4951
4	34↔57	4864	4864
5	49↔44	4780	4780
6	49↔30	4715	4715

7	45↔54	4680	4680
8	12↔16	4648	4648
9	26↔10	4617	4617
10	10↔13	4586	4586
11	13↔3	4540	4540
12	37↔61	4520	4520
13	36↔11	4503	4503
14	56↔29	4491	4491
15	21↔59	4487	4487
16	22↔57	4484	4484
17	51↔4	4483	4483
18	2↔3	5954	4483

Solution after 18 iterations:
Best tour =

1 → 42 → 38 → 18 → 43 → 8 → 20 → 61 → 62 → 37 → 28 → 46 → 39 → 59 → 21 → 63 → 60 → 7 → 40 → 25 → 17 → 29 → 32 → 56 → 47 → 50 → 64 → 14 → 45 → 54 → 44 → 49 → 24 → 22 → 57 → 41 → 19 → 2 → 27 → 52 → 23 → 5 → 6 → 4 → 51 → 33 → 58 → 9 → 31 → 48 → 12 → 16 → 26 → 55 → 3 → 13 → 10 → 35 → 53 → 30 → 36 → 11→ 34 → 15 → 1

Best solution = 4483 (Km.).

That is, the Route will be

Dhaka (Zero Point) → Narayanganj → Munshiganj → Gazipur → Narsingdi → Brahmanbaria → Habiganj → Sunamganj → Sylhet → Moulvibazar → Kishoreganj → Netrakona → Mymensingh → Sherpur → Jamalpur → Tangail → Sirajganj → Bogra → Naogaon → Joypurhat → Gaibandha → Kurigram → Lalmonirhat → Rangpur → Nilphamari → Panchagarh→ Thakurgaon → Dinajpur → Nawabganj → Rajshahi → Natore → Pabna → Jhenaidah → Jessore → Satkhira → Narail → Gopalganj → Bagerhat→ Khulna → Pirojpur → Jhalokathi → Barisal→ Bhola →Barguna → Patuakhali→ Madaripur → Shariatpur → Chandpur → Lakshmipur → Noakhali → Comilla→ Feni→ Khagrachhari → Rangamati → Bandarban → Cox's Bazar → Chittagong → Manikganj → Rajbari → Kushtia → Meherpur → Chuadanga → Magura → Faridpur → Dhaka (Zero Point).

V. Conclusion

Tabu search is a powerful algorithm to solve the optimization problems and is now applied to solve many difficult problems. The current work was studied traveling salesman problem because it is very useful to reach the optimal solution for the small size with 64 cities with 18 iterations. We see that the introduction of different neighborhood was necessary to reduce the computation time for tabu search, in which the approach swap between cities are efficient and create better solution, a neighborhood search algorithm searches among the neighbors of a candidate to find a better one. Tabu is useful to help the search procedure move away from previously visited portions of the search space and perform more extensive exploration.

References

[1]. **Applegate David L. , Bixby Robert E., Vasek Chvatal, Cook William J.,** "*The Traveling Salesman Problem A Computational study*", Princeton university press, Princeton and oxford [2006].
[2]. **Der-San Chen, Robert G. Batson, Yu Dang,** "*Applied Integer Programming*", John Wiley & sons, inc, publication [2010].
[3]. **TahaH. A.,** "*Operations Research An Introduction*", 8th edition, Pearson Prentice Hall, Pearson Education Inc. [2007].
[4]. **Donald Davendra,** "*Traveling salesman problem, Theory and Applications*", JanezaTrdine 9, 51000 Rijeka, Croatia, ISBN 978-953-307-426-9.
[5]. **Fred Glover, Manuel Laguna,** "*Tabu search*", Kluwer Academic publishers, 1997.
[6]. **Holland John H.,** "*Adaptation in Natural and Artificial systems*", MIT press edition, 1992.
[7]. **Irina Bryan**, "*The Traveling Salesman Problem*", [2009].
[8]. **Isra Natheer Alkallak, Ruqaya Zedan Sha'ban,** "*Tabu Search Method for Solving the Traveling salesman Problem*", University of Mosul, Raf. J. of Comp. & Math's. , Vol. 5, No. 2, 2008.
[9]. **Mashid Abtahi Froushani, Rosnah Mohd. Yusuff**, "*Development of an Innovative Algorithm for the Traveling salesman problem (TSP)"* , European Journal of Scientific Research, ISSN 1450-216X Vol.29 No.3 (2009), pp.349-359.
[10]. **Sachin Jayaswal, "***A comparative study of Tabu search and Simulated Annealing for Traveling Salesman Problem",* Department of Management Sciences, University of waterloo.
[11]. Roads and Highways Department, Ministry of Communications, Government of the People's Republic of Bangladesh. *www.rhd.gov.bd*.