IME639A. Solving Travelling Salesman Problem as assignment problem

Meena Vikas Gajanand (14384)

Aditya Kumar Singh (13817045)

The **Travelling salesman problem** (**TSP**) is an NP-hard problem. Although an assignment problem can be formulated as a linear programming problem, it is solved by a special method known as Hungarian Method.

**Problem Statement:**

A travelling salesman, plans to visit each of n cities. He wishes to visit each city once and only once, arriving back to city from where he started. The distance between City i and City j is Cij. What is the shortest tour he can take?

* He has to start from any one city and visit each city only once.
* Suppose he starts from the kth city and the last city he visited is m.
* Let the cost of travel from ith city to jth city be Cij.
* Then the objective function is Minimize ΣΣ CijXij

**Algorithm:**

1. Construct a cost [matrix](http://www.ccodechamp.com/category/c-programs/matrix/) to where entry C(a,b) represents total cost required to travel from city 'a' to 'b'. If a=b, C(a,b) = INF since we need to avoid self loops and disconnecting graphs.   
2. Determine the minimum element in each row and subtract the minimum element in each row from all the elements of the respective rows.  
3. Now repeat the same process of step 2 for all the columns i.e. determine the minimum element in each column and subtract that minimum value from all elements of their respective column to obtain new resulting matrix. This is done in update() function in our code.

4. Now after row and column operations, draw minimum number of horizontal and vertical lines to cover all zeros in resulting matrix. Let a be the minimum number of lines and n is the order of matrix. Method for doing this is explained further in check().

Then two possible scenarios can happen :  
4.1 If a=n, then an optimal assignment of paths can be done.  
4.2 If a<n, then proceed.

5. Now find smallest uncovered element in the matrix by “a” lines I.e element which is not crossed by any of the lines.

6. Subtract the minimum element obtained in step 5 from all uncovered elements and add the same elements at the intersection of horizontal and vertical lines to obtain intermediate matrix. Matrix Cross[n][n] can be used to look at elements covered by horizontal and vertical lines.  
7. Repeat step(2) to step (4) until we get the case 4.1 .  
8. Now our problem reduces to finding in optimal path in resulted cost matrix. This is done by traversing rows and selecting entries with zero values and moving to corresponding row that it points .This is done recursively repeated unless a cyclic path is not found. For eg. If we are in ith row and jth column in ith row is zero then our salesperson will move to jth city and we will mark city ith city so that it is not repeated at later stage.

9. If the solutions thus found out are cyclic in nature, then that is the final solution

10. If it is not cyclic, then select the next lowest number in the table (other than zero). Now consider this no. Too during assignment. Check whether cyclic assignment is available.

If not, include the next higher entry in the table and the procedure is repeated until a cyclic assignment is obtained.

**Finding minimum no. of lines :**

1.This is done using backtracking. For any zero we encounter while traversing matrix first check if it is covered by previously assigned lines or not.

2. If not then cut it via a horizontal line and call check() function for further rows(since all zeros will be covered )

3. Similarly cross the element via a vertica line and call check() fucntion for further elements.

4. Now compare no. of lines required in both the cases and assign the one which requires minimum line and bactrack.

5. Do this untill all the zeros are covered by atleast one line.

Finding final cost:

1.Using optimal path check the corresponding entries in our original cost matrix (dup[n][n] is used for this purpose) to find the minimum cost that will be required to cover all the cities by the saesperson.

Let's look at an example which will cover all the sceanrios of our algorithm: