**September 2, 2005** Fall 2005

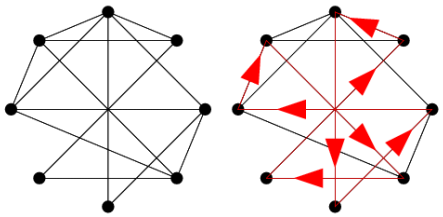
Algorithm Reading Group

*Scribe: Shweta Jain*

# 1 Overview

We discussed a variation of Traveling Salesman problem. This problem had come up in a biology problem and can be solved in polynomial time. We would discuss this algorithm and try to solve this in lower time complexity.

**Definition (Traveling Salesman Problem).** *A problem in graph theory requiring the most efficient (i.e., least total distance) Hamiltonian circuit a salesman can take through each of n cities. No general method of solution is known, and the problem is NP-hard*[1].



We will name our problem *Traveling Salesman Problem with Country* to distinguish it from regular TSP. As the name suggests, the salesman in this problem travels through cities in different countries and tries to minimize the total cost of the entire tour. In the next section, we will describe the problem in details and then explain different algorithms that were suggested to optimize the cost of a complete tour.

# 2 Problem Description

We describe the TSP with country problem here. Consider that there are $C$ countries in the graph that we will traverse and there are $N$ cities such that country $C_i$ has $N_i$ cities in it. The cost metrics for traversing the graph is not defined on cities but on countries i.e the matrix contains the cost of traveling from city $n_k$ of country $C_m$ to city $n_l$ of country $C_n$ where $k \neq l$. Due to different policies in different countries, for some countries it might be

[1] reference http://mathworld.wolfram.com/TravelingSalesmanProblem.html

expensive to travel to the country and cheap to travel within that country but for another set of countries it might be cheap to travel into the country but expensive to travel within. There can be other variations like expensive to travel in but cheap to travel out of a country and vice-versa. Thus the cost matrix describes all such travel costs. Given the the cost matrix we need to find the minimum cost tour that visits all cities in all countries.

We discussed some special cases for this problem which we will discuss in the next subsections.

### 2.1 Case with one country

We observe here that in a special case where there is only one country, the tour could be any permutation of cities since the cost to travel between cities within a country is same for all pairs of cities.

### 2.2 Case with two countries

In a more general case when there are two countries $C_1$ and $C_2$ with $N_1$ and $N_2$ cities respectively, the cost will depend upon the matrix. Thus if it is cheap to travel across countries than to travel within, then the optimum tour will be to visit one city in $C_1$ and then one in $C_2$. In the converse case, the optimum tour will be to visit all $N_1$ cities in $C_1$ and then go to $C_2$ and visit all cities there.

### 2.3 Case one city per country

If there is only one city in every country, the problem reduces to the normal TSP problem and is known to be NP-hard.

### 2.4 Greedy Algorithm

If it is cheap to enter countries but expensive to leave, then the greedy approach of visiting the cheapest unvisited country will probably work.

# 3 Goal

The goal of this session is to think about and discuss approaches to solve the TSP with country problem with a cost of $O(N \times f(N, C))$ where $f$ is an exponential function. Steve's dream algorithm will be to achieve $O(N \times 2^C)$ while Joe would like to solve it with $O(N + 2^C)$ cost. We are not sure if Steve's goal is achievable so Joe's goal is even harder.

# 4 Claims and Observations

**Observation.** *For two countries there are only two optimal traveling patterns with minimum cost, one for each model of the matrix, if the matrix is uniform.*

1. *(12)*,2* (if $N_2 > N_1$)*
2. *(1)* (2)**

**Conjecture.** *The same approach can be extended to C countries.*

**Observation.** *In any tour containing patterns that resemble $ABCDABCD....EFABCD...$ we can move all but one set of repetitive patterns of the same kind and place them adjacent to another pattern of the same kind without changing the cost of the tour.*

**Illustration.** *$AAABABBBCD..$ can be written as $AABAABBBCD...$*

If the costs are uniform then the above observation leads to $O(N \times C!)$ since there are $C!$ choices of patterns in which to visit each country and there are N countries.

**Observation.** *There are $f(C)$ local patterns in any local pattern. Each pattern can have $2^C$ ways of selecting countries. We need to find the number of such patterns.*

**Claim.** *We can find tours with cost $O(N^C)$ by dynamic programming by building the sequence given all previous positions and then make the next visit decision as follows. Let us denote the cost of the optimal tour with $r_i$ visits to country $C_i$ as $F(r_1, r_2, r_3, ....r_i, L]$. Now we need to know the last visit ending at L and the number of visits to each $C_j$ to make the next visit decision. $F(r_1, r_2, r_3, ....r_i, L] = Min(F[...] + C(L', L))$.*

## 4.1 2-factor or Cycle cover algorithm

**Definition (Cycle Cover or 2-factor).** *A cycle cover (or 2-factor) of a graph G consists of a 2-regular spanning subgraph $H \subseteq G$ (i.e a union of cycles in which every vertex is*

*incident with exactly two edges).*

**Definition (Minimum Cost Perfect Matching).** *A MATCHING is a set of edges with no endpoints in common and a perfect matching is a matching that connects all vertices. A minimum cost perfect matching is a perfect matching with the minimum weight or cost in a weighted graph and can be computed in* $O(n^3)$.

Let us form a graph with countries as vertices and duplicate the vertices so that we have two vertices for each country. Let us assign a cost of *infinity* to all edges connecting the same country. Now we can find a minimum cost perfect matching for this graph which we can then subsequently convert into a cycle cover. A cycle cover is a minimum cost tour for the graph and hence has a lower cost than TSP. If the cycle cover can be converted into a TSP tour without increasing the cost then that tour would be the optimal TSP tour. We can achieve this by connecting the subgraphs in the cycle cover by removing edges from one subgraph and adding edges to connect the graph with other subgraphs.

The only counter example to this arguement is if there is a vertex such that the cost of edges connecting this vertex with any other verex is minimum. In such a case any perfect matching will not lead to a minimum cost tour.