2019 ADA miniHW 10

b07902064 資工二 蔡銘軒

January 3, 2020

From the pseudo code, we can see the algorithm runs in $O(N^2)$, which is polynomial time.
As for the approximation ratio, we first show that for a given graph, the the optimal cost of TSP is larger than or equal to the minimum spanning tree of the graph.

Proof. Let C be the cycle that gives the optimal cost of TSP. If we remove an arbitrary edge in C, the remaining edges form a spanning tree, M, of the graph. Since the cost of every edge is nonnegative, we have $cost(M) \leq cost(C)$. And since the minimum spanning tree, M', is the spanning tree with the least cost, we have $cost(M') \leq cost(M)$, and thus $cost(M') \leq cost(C)$. □

Let $S = \{(s_1, t_1), (s_2, t_2), \ldots, (s_{n-1}, t_{n-1})\}$ be the set of edges such that (s_i, t_i) is the edge discovered in the *i-th* iteration. Particulary, (s_1, t_1) is the edge with minimum cost as required by the problem. It is clear that S form a minimum spanning tree, as the algorithm discovers edges in the same manner as the Prim's algorithm. Let T^* be the optimal cycle for the TSP problem, then we have

$$\sum_{i=1}^{n-1} cost(s_i, t_i) \leq cost(T^*)$$

Next we consider the tour T returned by the algorithm. When $n = 2$, the cost of the cycle found by the algorithm is $2 \cdot cost(s_1, t_1)$. In each iteration, we add edges $(v_i, v_j), (v_j, v_k)$ and remove edge (v_i, v_k) for some v_i, v_j, v_k. The cost is changed by $cost(v_i, v_j) + cost(v_j, v_k) - cost(v_i, v_k)$. By triangle inequality, we have $cost(v_i, v_j) + cost(v_i, v_k) \geq cost(v_j, v_k) \Leftrightarrow cost(v_i, v_j) \geq cost(v_j, v_k) - cost(v_i, v_k)$, which indicates the change of cost is at most $2 \cdot cost(v_i, v_j)$. Then we have

$$cost(T) \leq \sum_{i=1}^{n-1} 2 \cdot cost(s_i, t_i) \leq 2 \cdot cost(T^*)$$

The proof is complete.

Reference

The Design of Approximation Algorithms by David P. Williamson, David B. Shmoys. Page 38