Tree Traveling Salesman Problem

S.N. Kabadi*

The University of New Brunswick

R. Chandrasekaran

University of Texas at Dallas

* This research was partially supported by N.S.E.R.C. (Canada) research grant # A8085.

Given a directed graph $G = [N, A]$ and arc lengths $c_{i,j}$, the problem of finding a closed directed path of minimum length that passes through each node precisely once is known in the literature as the *Traveling Salesman Problem* (TSP) [LLRS] . Such paths are also called *Hamiltonian Cycles or Tours.* If for each node i, we can select a state k from a set of possible states S_i, and the length of the arc from nodes i to j is given by $c_{i,j}^{k\,l}$ where k is the state of i and l is the state of j, then we get a *generalized traveling salesman problem* (GTSP) [KC]. Here we have to find not only the optimum tour but also the states for each node. There is a way to formulate the GTSP as a TSP on a larger graph. Hence, the two problems are equivalent.

We consider the following special case of GTSP which generalizes some work of Gilmore and Gomory [GG] on a special case of TSP.

Problem I: Given an undirected tree $T = [V, E]$; nonnegative edge lengths $d_{p,q}$ for $(p, q) \in E$; a set of unordered pairs of nodes (a_i, b_i), with $a_i \in V$ and $b_i \in V$, for $1 \leq i \leq n$. For $x, y \in V$, let $d_{x,y}$ be the length of the unique path between x and y in T. The problem we consider is a special case of GTSP on another graph $G = [N, A]$ with $|N| = n$; where each node has two states α and β. $c_{i,j}^{\alpha\,\alpha} = d_{a_j,b_i}$; $c_{i,j}^{\alpha\,\beta} = d_{b_j,b_i}$; $c_{i,j}^{\beta\,\alpha} = d_{a_j,a_i}$; $c_{i,j}^{\beta\,\beta} = d_{b_j,a_i}$.

Assumption: All a_i, b_i are distinct and the set of nodes in T with degree 2 is a subset of the set of these nodes.

Let $S = \bigcup_{i=1}^{n} \{a_i, b_i\} = \{s_1, s_2, \ldots, s_{2n}\}$. The algorithm presented below is a generalization of an algorithm found in [KC] for the case T is a path.

Algorithm TTSP:

Step 1: Let $H = K_S$ be the complete graph on S. For s_i, s_j in S, let d_{s_i, s_j} be the length of the unique path in T between s_i and s_j. Using Algorithm M (described later), find a perfect matching φ^*, in H with the minimum value of $\left(\sum_{i=1}^{2n} d(s_i, \varphi(s_i))/2 = c_{\varphi^*}\right.$.

Definition: For any perfect matching φ in H, let $G_\varphi = [N, E_\varphi]$ be an undirected graph, where

$N = \{i: 1 \leq i \leq 2n\}$ and

$E_\varphi = [(i,j)$: either $\varphi(s_i) = s_j$ or $\{s_i, s_j\} \equiv \{a_i, b_i\}$ for some $i]$.

Step 2: If G_{φ^*} is connected, then stop; we have the required tour. Else, let the connected components of G_{φ^*} be $G^t = [V^t, E^t_{\varphi^*}]$ for $1 \leq t \leq k$ with $k > 1$.

Go to step 3.

Step 3: For $1 \leq t \leq k$, contract nodes in V^t into a single node x_t in T to obtain a multigraph $Q = [X, F]$. $\nabla = \bigcup_{t=1}^{k} \{x_t\} \subset X$ for $1 \leq t \leq k$; $X - \nabla$ are the nodes in T that do not correspond to any a_i or b_i. Find the minimum Steiner tree F^* in Q containing the set ∇.

Step 4: Modify φ^* using edges in F^* to obtain the tour τ^* as follows.

Let $\tau^0 = \varphi^*$; $F^1 = F^*$; $i = 1$.

4(a): $e = (u, v) \in F^i$, where s is a tip node of F^i. Clearly, $s \in \nabla$. If $t \notin \nabla$, contract s and t and let the new node be s; modify steiner tree to F^{i+1} and go to step 4(b). If $t \in \nabla$, then let

$\tau^i(s) = \tau^{i-1}(t)$; $\tau^i(t) = \tau^{i-1}(s)$;

$\tau^i(\tau^{i-1}(t)) = s$; $\tau^i(\tau^{i-1}(s)) = t$;

$\tau^i(x) = \tau^{i-1}(x)$ otherwise.

$F^{i+1} = F^i - \{e\}$; go to step 4(b)

4(b): If $F^{i+1} \neq \phi$, $i = i+1$, go to step 4(a). Else, let $\tau^* = \tau^i$; go to step 5.

Step 5: Construct G_{T^*}. *Such a modification of* φ^* *is called a patching of* φ^* *and it is well known that this results in a tour* [KC]. A traversal of the tour also yields the states for the GTSP. This is the required solution of the GTSP.

Algorithm M:

Step 0: Let $T^0 = T$; $j = 0$; $S^0 = S$.

Step 1: Let u be a tip node of T^j and $e = (u, v)$. If $u \notin S^j$, then delete u and e to get the new tree T^{j+1}. Let $S^{j+1} = S^j$ and go to step 2. If $u \in S^j$ and if $v \notin S^j$, contract e and let the new node be u; new tree be T^{j+1}; $S^{j+1} = S^j$; go to step 2. If $u \in S^j$, and $v \in S^j$, then (u, v) in the matching; hence $\varphi^*(u) = v$ and $\varphi^*(v) = u$; delete e from the tree to get T^{j+1}; $S^{j+1} = S^j - \{u, v\}$; go to step 2.

Step 2: $j = j+1$; if $S^j = \phi$, stop. Else go to step 1.

Validity of Algorithm M:

A perfect matching φ in S is *noncrossing* iff $\forall\ [i \neq j \neq \varphi(i), i, j, \varphi(i), \in S]$ the path $P(i, \varphi(i))$ and $P(j, \varphi(j))$ in T do not have a common node.

Lemma 1: A perfect matching φ in S is optimal iff it is noncrossing.

Lemma 2: Algorithm M produces a noncrossing matching in S.

Validity of Algorithm TTSP:

The validity of algorithm TTSP follows from the following two lemmas.

If we used step 4, starting with any steiner tree F containing $\bar{V}$ in Q yields a feasible solution to GTSP.

Lemma 3: For any steiner tree F containing ∇ in Q, $c(\varphi^*(F)) = c(\varphi^*) + 2 \sum_{e \in F} d_e$ where $c(\varphi(F))$ is the cost of the solution to GTSP obtained by doing step 4 to φ using F and $c(\varphi)$ is the cost of the matching φ in K_S.

Lemma 4: For any feasible solution τ to GTSP $\exists$ a steiner tree F in Q $\ni$ $c(\varphi^*(F)) \leq c(\tau)$ where $c(\tau)$ is the cost of τ.

The proof of lemma 4 *actually provides a polynomial algorithm that produces such an* F *given* τ. Thus, Problem I is polynomially equivalent to step 3 of algorithm TTSP (that of finding the required steiner tree in Q).

Theorem 1: The problem of determining minimum cost steiner tree in step 3 of algorithm TTSP is NP–hard.

Remark: This might look like a special case of the steiner tree problem, but it is still NP–hard.

Corollary 1: Problem I is NP–hard.

Remark 2: *However, if every node of T corresponds to one of the* a_i or b_i, *then in step 3 of algorithm* TTSP, $X - \nabla = \phi$. *In this case, the steiner tree problem is a minimum spanning tree problem which is nicely solvable* [K]. We, therefore, have the following generalization of the result in [KC] (which itself is a generalization of the result in [GG]):

Theorem 2: If every node in T is an a_i or b_i for some i, then algorithm TTSP is O(n log n), and it solves the problem.

References:

[GG]
P.C. Gilmore: R.E. Gomory — Sequencing a One-state Variable Machine: A Solvable case of the Traveling Salesman Problem, *Operations Research*, 12, (1964), 655–679.

[K]
J.B. Kruskal: — On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem, *Proc. Amer. Math. Soc.*, 7, (1956), 48–50.

[KC]
S.N. Kabadi : R. Chandrasekaran — Solvable Classes of Generalized Traveling Salesman Problem, 1985.

[LLRS]
E.L. Lawler, J.K. Lenstra, A.H.G. Rinooy Kahn, D.G. Shmoys — The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization, J.Wiley, (1985).