Lecture notes for Mar 29, 2023 Minimum weighted T-joins and Travelling Salesman Problem

Chun-Hung Liu

March 29, 2023

1 Finding a minimum T-join

In this section, we will describe how to find a minimum weighted T-join, for any subset T of $V(G)$ for which a T-join exists. It is easy to characterize the existence of a T-join.

Recall that we show the following proposition last time.

Proposition 1 *Let G be a graph. Let $T \subseteq V(G)$. Then there exists a T-join of G if and only if for every component C of G, $|T \cap V(C)|$ is even.*

Now we study the structure of a T-join.

Lemma 2 *Let G be a graph. Let $w : E(G) \to \mathbb{R}_{\geq 0}$. Let $T \subseteq V(G)$. Let J be a T-join of G with $\sum_{e \in J} w(e)$ minimum. Then J can be partitioned into sets, where each of them is either the edge-set of a path in G connecting two vertices in T or the edge-set of an Eulerian subgraph of G with zero weight.*

Proof. Let $\mathcal{P}$ be a maximal set such that each member of $\mathcal{P}$ is a subset of J and is also the edge-set of a path in G connecting two vertices in T, and members of $\mathcal{P}$ are pairwise disjoint. For every $v \in V(G)$, let k_v be the number of members of $\mathcal{P}$ corresponding to a path having v as an end. Let $H_{\mathcal{P}}$ be the graph with $V(H_{\mathcal{P}}) = V(G)$ and $E(H_{\mathcal{P}}) = \bigcup_{M \in \mathcal{P}} M$. So for every $v \in V(G)$, $\deg_{H_{\mathcal{P}}}(v) \equiv k_v \pmod 2$.

Let H_J be the graph with $V(H_J) = V(G)$ and $E(H_J) = J$. So for every $v \in V(G)$, $\deg_{H_J}(v)$ is odd if and only if $v \in T$.

Note that for every $v \in V(G)$, $\deg_{H_J - E(H_\mathcal{P})}(v) = \deg_{H_J}(v) - \deg_{H_\mathcal{P}}(v) \equiv \deg_{H_J}(v) - k_v \pmod 2$. For every vertex v in $V(G) - T$, v cannot be an end of a path corresponding to a member of $\mathcal{P}$, so $k_v = 0$, and hence $\deg_{H_J - E(H_\mathcal{P})}(v)$ is even. So every vertex with odd degree in $H_J - E(H_\mathcal{P})$ is in T.

If $H_J - E(H_\mathcal{P})$ is Eulerian, then $J \cap E(H_\mathcal{P})$ is a T-join, and since $\sum_{e \in J} w(e)$ is minimum among all T-joins, we know $\sum_{e \in J - E(H_\mathcal{P})} w(e) = 0$, so we obtain the desired partition by adding $E(H_J) - E(H_\mathcal{P})$ into $\mathcal{P}$.

So we may assume $H_J - E(H_\mathcal{P})$ is not Eulerian. By the hand-shake lemma, there are at least two vertices in T having odd degree in $H_J - E(H_\mathcal{P})$.

Let O be the set of odd degree vertices in $H_J - E(H_\mathcal{P})$. So $O \subseteq T$, $|O| \geq 2$ and $J - E(H_\mathcal{P})$ is an O-join of $H_J - E(H_\mathcal{P})$. By the maximality of $\mathcal{P}$, there exists no path in $H_J - E(H_\mathcal{P})$ between two vertices in O. So there exists a component C of $H_J - E(H_\mathcal{P})$ such that $|O \cap V(C)| = 1$. That is, C has only one odd-degree vertex, a contradiction. ■

Lemma 3 *Let G be a graph. Let $w : E(G) \to \mathbb{R}_{\geq 0}$. Let $T \subseteq V(G)$ with $T \neq \emptyset$. If there exists a T-join, then there exists a minimum weighted T-join J such that $J = \bigcup_{i=1}^{|V(T)|/2} E(P_i)$, where $P_1, P_2, ..., P_{|V(T)|/2}$ are edge-disjoint paths each connecting two vertices in T, and every vertex in T is an end of exactly one of those P_i's.*

Proof. Assume that there exists a T-join. Since every edge has nonnegative weight, there exists a minimum weighted T-join. Let J be a minimum weighted T-join with $|J|$ minimum. Then $J = \bigcup_{i=1}^{k} E(P_i)$ for some edge-disjoint paths $P_1, P_2, ..., P_k$ each connecting two vertices in T by Lemma 2, and for some positive integer k. We choose those paths P_i's such that k is minimum.

Suppose to the contrary that some vertex v is an end of two of those P_i's, say P_1 and P_2. Let a_1 and a_2 be the ends of P_1 and P_2 other than v, respectively. If $a_1 = a_2$, then $P_1 \cup P_2$ is Eulerian, so $J - (E(P_1 \cup P_2))$ is a T-join, contradicting the minimality of J. So $a_1 \neq a_2$, and hence $P_1 \cup P_2$ contains a path P from a_1 and a_2. Note that $J - (E(P_1 \cup P_2) - E(P))$ is a T-join. By the minimality of J, $P_1 \cup P_2 = P$. Hence we can reduce the number of paths in the decomposition, a contradiction.

So every vertex is an end of exactly one of those P_i's. Hence $k = |V(T)|/2$.

■

==========

Finding a minimum T-join in a nonnegative weighted graph
Input: A graph G, a function $w : E(G) \to \mathbb{R}_{\geq 0}$, and a subset T of $V(G)$.
Output: Either a T-join J with minimum $\sum_{e \in J} w(e)$, or output "no T-join exists".
Procedure:

Step 1: If there exists a component C of G such that $|V(C) \cap T|$ is odd, then output "no T-join exists".

Step 2: Construct a complete graph H with $V(H) = T$. For each pair of distinct vertices x, y in T, compute a shortest path $P_{x,y}$ in (G, w) from x to y, and define $f(xy)$ to be the length of $P_{x,y}$. (Note that if $P_{x,y}$ does not exist, we let $f(xy) = |V(G)| \sum_{e \in E(G)} w(e) + 1$.)

Step 3: Find a minimum weighted perfect matching M of the weighted graph (H, f). Output $\Delta_{xy \in M} E(P_{x,y})$, where Δ is the operator that denotes symmetric difference.

==========

Theorem 4 *Given a graph G, a function $w : E(G) \to \mathbb{R}_{\geq 0}$ and a subset T of $V(G)$, the above algorithm correctly outputs a minimum weighted T-join of G or concludes that there exists no T-joins, in time $O(|V(G)|^3)$.*

Proof. By Proposition 1, Step 1 correctly decides whether G has a T-join.

Let J be the output of the algorithm. It implies that Step 2 is executed. So for every component C of G, $|V(C) \cap T|$ is even. Hence there exists a perfect matching M_0 of H such that for every $e \in M_0$, $f(e)$ is the length of a path in (G, w), so $f(e) \leq \sum_{e \in E(G)} w(e)$. So the weight of M_0 is at most $|V(H)| \cdot \sum_{e \in E(G)} w(e)$. Since M is a minimum weighted perfect matching of (H, f), the weight of M is at most the weight of M_0, so M does not contain any edge e with $f(e) = |V(G)| \sum_{e \in E(G)} w(e) + 1$. That is, for every edge $e = xy \in M$, $P_{x,y}$ exists. So J is well-defined. Moreover, since each vertex in T is an end of exactly one of the path in $\{P_{x,y} : xy \in M\}$, and no vertex in $V(G) - T$ is an end of a path in $\{P_{x,y} : xy \in M\}$, we know that J is a T-join.

Let J^* be a minimum weighted T-join. By Lemma 3, we may assume that J^* is a union of edge-disjoint paths $P_1, P_2, ..., P_{|V(T)|/2}$ each connecting

two vertices in T, and every vertex in T is an end of exactly one of those P_i's. For each i, let a_i, b_i be the ends of P_i. So $\{a_i b_i : i \in [|V(T)|/2]\}$ is a perfect matching of H, and $J^* = \bigcup_{xy \in M} E(P_{x,y})$. Hence the weight of J^* is at least the weight of J. Therefore, J is a minimum weighted T-join.

Now we consider the time complexity. Step 1 can be done in linear time. To implement Step 2, before we find the shortest paths, we can make the graph simple without changing the distance between any pair of vertices by first removing all loops and for every pair of vertices x, y, only keeping one edge between x, y with the smallest weight. So for every pair x, y, we can find a shortest path between x, y in time $O(|V(G)|^2 + |E(G)|) = O(|V(G)|^2)$ by Dijkstra's algorithm. Hence Step 2 can be done in time $O(|V(G)|^4)$. In fact, it can be improved to $O(|V(G)|^3)$ by using a more efficient algorithm that computes the shortest paths for each pair of vertices in $V(G)$ in time $O(|V(G)||E(G)| + |V(G)|^2 \log |V(G)|)$ by Bazaraa and Langley and by Johnson. And a minimum weighted perfect matching of (H, f) can be found in time $O(|T|^3) = O(|V(G)|^3)$. Finally, J can be constructed in time $O(|V(G)|^2)$. Therefore, the total running time is $O(|V(G)|^3)$. ∎

1.1 Solving Chinese Postman Problem

Recall that we proved the following results.

Theorem 5 *Let G be a graph. Let $w : E(G) \to \mathbb{R}_{\geq 0}$. Let $T \subseteq V(G)$. Let J be a T-join of G with $\sum_{e \in J} w(e)$ minimum. Let G' be the graph obtained from G by duplicating each edge in J once. Then every Eulerian circuit of G' is a minimum weighted Chinese postman tour of G.*

Theorem 6 *Given a connected graph G whose every vertex has even degree, one can find an Eulerian circuit of G in linear time.*

Now we describe an algorithm for solving the Chinese Postman Problem.

==========

Solving Chinese Postman Problem for nonnegative weighted graphs
Input: A connected graph G and a function $w : E(G) \to \mathbb{R}_{\geq 0}$.
Output: A Chinese postman tour W with minimum $\sum_{e \in E(W)} w(e)$.
Procedure:

Step 1: Let $T = \{v \in V(G) : \deg_G(v) \text{ is odd}\}$. Find a minimum weighted T-join J of (G, w).

Step 2: Construct the graph G' by copying each edge in J.

Step 3: Find an Eulerian circuit W' of G'. Let W be the walk obtained from W' by replacing each edge in J by its original in G. Output W.

==========

Theorem 7 *Given an nonnegative weighted graph (G, w), the above algorithm outputs a Chinese postman tour W with minimum $\sum_{e \in E(W)} w(e)$ in time $O(|V(G)|^3)$.*

Proof. The correctness immediately follows from Theorem 5. The time complexity immediately follows from Theorems 6 and 4. ∎

2 Travelling Salesman Problem

Recall that the Chinese Postman Problem asks for a tour that visits every edge at least once. Travelling Salesman Problem is similar, but now we ask for a tour that visits every vertex exactly once, and the problem is restricted on weighted complete graphs.

==========
Travelling Salesman Problem (TSP)
Input: A positive integer n and a function $w : E(K_n) \to \mathbb{R}_{\geq 0}$.
Output: A cycle C of K_n containing every vertex exactly once such that $\sum_{e \in E(C)} w(e)$ is minimum.
==========

Travelling Salesman Problem is NP-hard because it is more general than finding a Hamiltonian cycle of a graph, as we will show. A *Hamiltonian cycle* of a graph G is a cycle C with $V(C) = V(G)$.

Theorem 8 *Deciding whether the input graph has a Hamiltonian cycle or not is NP-hard.*

Proof. Recall that deciding whether a graph G has a Hamiltonian path is NP-hard. Note that for every pair of vertices x, y, G has a Hamiltonian path between x and y if and only if $G + xy$ has a Hamiltonian cycle. So if testing

Hamiltonian cycle of any given graph can be done in polynomial time, then we can test whether an input G has a Hamiltonian path in polynomial time by checking $\binom{n}{2}$ graphs has a Hamiltonian cycle or not, where each graph is obtained from G by adding one edge. ∎

Theorem 8 not only implies that TSP is NP-hard, it implies that even finding a constant factor approximation is NP-hard, as shown below.

Corollary 9 *For every real number k, it is NP-hard to, given a weighted complete graph (K_n, w), output a Hamiltonian cycle C such that $\sum_{e\in E(C)} w(e) \leq k \cdot \mathrm{OPT}$, where* OPT *is the weight of the shortest Hamiltonian cycle in (K_n, w).*

Proof. Let G be a graph. Let $n = |V(G)|$. We assume $V(G) = V(K_n)$ for simplicity of notations. Define $w : E(K_n) \to \mathbb{R}_{\geq 0}$ such that $w(e) = 1$ if $e \in E(G)$, and $w(e) = k|V(G)| + 1$ if $e \notin E(G)$.

Note that if G has a Hamiltonian cycle, then $\mathrm{OPT} = n$; if G does not have a Hamiltonian cycle, then the minimum weighted Hamiltonian cycle C of (K_n, w) must uses at least one edge in $E(K_n) - E(G)$, so $\mathrm{OPT} = w(C) \geq k|V(G)| + 1$.

Let A be an algorithm that approximates OPT with factor k. Let ℓ be the weight of the cycle output from A (with input (K_n, w)). So $\mathrm{OPT} \leq \ell \leq k \cdot \mathrm{OPT}$. Hence if $\ell \leq k|V(G)|$, then $\mathrm{OPT} \leq |V(G)|$, so G has a Hamiltonian cycle; if $\ell > k|V(G)|$, then $\mathrm{OPT} \neq |V(G)|$, so $\mathrm{OPT} \geq k|V(G)| + 1$, and hence G does not have a Hamiltonian cycle. That is, $\ell \leq k|V(G)|$ if and only if G has a Hamiltonian cycle. Therefore, if A runs in polynomial time, then we can test whether G has a Hamiltonian cycle in polynomial time. So approximating OPT up to a factor k is NP-hard by Theorem 8. ∎

As TSP is NP-hard, we look for approximation algorithms. Due to the hardness for approximating TSP, we consider a special case of TSP.

==========

Metric TSP

Input: A positive integer n and a function $w : E(K_n) \to \mathbb{R}_{\geq 0}$ that satisfies the triangle inequality (i.e. for any $x, y, z \in V(K_n)$, $w(x, y) + w(y, z) \geq w(x, z)$).

Output: A cycle C of K_n containing every vertex exactly once such that $\sum_{e\in E(C)} w(e)$ is minimum.

==========