| | |
|---|---|
| **COMPSCI 638: Graph Algorithms** | October 4, 2019 |
| Lecture 12 | |
| *Lecturer: Debmalya Panigrahi* | *Scribe: Kevin Sun* |

# 1 Overview

In this lecture, we begin studying network design problems. In particular, we give a 2-approximation for the metric Traveling Salesman problem, and an improved version known as Christofides algorithm. For the Steiner tree problem, we give a similar 2-approximation algorithm before turning our attention to the Steiner forest.

# 2 Traveling Salesman

In the Traveling Salesman Problem (TSP), we are given an undirected graph $G = (V, E)$ and each edge $e = \{u, v\}$ has length $d(u, v) \geq 0$. The goal is to compute a tour $T$ that minimizes $d(T) = \sum_{e \in T} d(e)$, where a *tour* (also known as a *Hamiltonian cycle*) is a cycle that visits each vertex exactly once. In the metric version, we assume $d$ satisfies the triangle inequality.

**Theorem 1.** *It is NP-hard to approximate TSP within any constant factor.*

*Proof.* For contradiction, assume we have an $\alpha$-approximation, where $\alpha$ is a constant, for the TSP problem. We claim that this allows us to solve the Hamiltonian Cycle problem, which simply asks if a graph contains a Hamiltonian cycle and is known to be NP-hard.

Let $G = (V, E)$ be an instance of Hamiltonian Cycle on $n$ vertices and $m$ edges. We construct an instance $G' = (V, E')$ of TSP as follows: for every $x, y \in V$, add edge $e = \{x, y\}$ to $E'$ with length $d(x, y) = 1$ if $e \in E$, and $d(x, y) = \alpha n$ otherwise.

Now let $C^*$ denote the optimal tour in $G'$; our $\alpha$-approximation on $G'$ gives a tour $C'$ such that $w(C') \leq \alpha w(C^*)$. If $G$ has a Hamiltonian Cycle, then $w(C') \leq \alpha n = \alpha w(C^*)$; otherwise, $w(C') > \alpha n$ because we must use an edge in $E' \setminus E$ to complete a tour. □

In light of Theorem 1, we relax our problem by assuming that the distance function is a metric.

**A 2-approximation for metric TSP:** Let $T$ be the minimum spanning tree (MST) of the graph, and let $R$ denote the same tree, with two copies of every edge. Note that $R$ is Eulerian because every vertex in $R$ has even degree, so there exists a walk $W$ that traverses every edge exactly once, i.e., $d(W) = d(R) = 2d(T)$. However, $W$ may contain repeated vertices. To circumvent this, we apply "shortcutting" to create $W'$: at each step in $W$, visit the next (in order of $W$) unvisited vertex.

By the triangle inequality, we have $d(W') \leq d(W)$. Furthermore, if $C^*$ denotes an optimal tour, then $d(T) \leq d(C^*)$ because removing any edge of $C^*$ results in a spanning tree. Putting this together, we have $d(W') \leq 2d(T) \leq 2d(C^*)$.

### 2.1 Christofides algorithm

In the MST-based 2-approximation for metric TSP given above, we constructed the walk $W$ in the MST $T$ by simply doubling every edge. In this section, we apply a different technique to improve our approximation ratio from 2 to $3/2$.

The idea is the following: given a minimum spanning tree $T$, we can ensure that each vertex has even degree by doubling every edge. However, we can instead add a matching of the vertices with odd degree in $T$: by the handshaking lemma, there exists an even number of such vertices. This allows us do add fewer edges while ensuring that the resulting graph is still Eulerian.

---

**Algorithm 1** Christofides Algorithm for metric TSP

---

Compute a minimum spanning tree $T$ of $G$.
Let $V_{\text{odd}}$ denote the vertices whose degree in $T$ is odd; notice $|V_{\text{odd}}|$ is even.
Compute a minimum perfect matching $M$ on $V_{\text{odd}}$ in $G$.
Construct the graph $R = T + M$; notice that $R$ is Eulerian.
Let $W$ be an Eulerian circuit in $R$, i.e., a walk that traverses each edge of $R$ exactly once.
**return** the tour $W'$, which is $W$ with shortcuts applied.

---

**Theorem 2.** *Algorithm 1 returns a* $3/2$*-approximation for metric TSP.*

*Proof.* Let $C^*$ denote an optimal tour. By the triangle inequality, we have

$$d(W') \le d(W) = d(T) + d(M) \le d(C^*) + d(M),$$

where the second inequality again follows because removing any edge of $C^*$ yields a spanning tree. Thus, it suffices to prove $d(M) \le d(C^*)/2$.

Recall that $V_{\text{odd}}$ denotes the vertices whose degree in $T$ is odd. Consider the cycle $C^*_{\text{odd}}$ on $V_{\text{odd}}$ obtained by traversing $C^*$ while shortcutting all vertices in $V \setminus V_{\text{odd}}$. By the triangle inequality, $d(C^*_{\text{odd}}) \le d(C^*)$. Furthermore, since $|V_{\text{odd}}|$ is even, $C^*_{\text{odd}}$ can be partitioned into two perfect matchings $M_1, M_2$ on $V_{\text{odd}}$ such that $d(C^*_{\text{odd}}) = d(M_1) + d(M_2)$. Since $M$ (from the algorithm) is a minimum perfect matching on $V_{\text{odd}}$, we have

$$d(M) \le \min\{d(M_1), d(M_2)\} \le \frac{d(C^*_{\text{odd}})}{2} \le \frac{d(C^*)}{2}. \qquad \square$$

**Remark:** Despite the simplicity in both the algorithm and its analysis, Algorithm 1 provides the best-known approximation guarantee for metric TSP in general graphs. Improving upon this $3/2$ is a notorious open problem in theoretical computer science; it is conjectured that $4/3$ is attainable.

In asymmetric TSP (ATSP), the problem is the same but we are now given a *directed* graph. The first approximation algorithm for this problem was a $O(\log n)$-approximation given by Frieze et al. [FGM82]. Almost thirty years later, this bound was improved to $O(\log n / \log\log n)$ by Asadpour et al. [AGM+10]. Most recently, Svensson et al. [STV18] gave a constant approximation for ATSP.

## 3 Steiner Tree and Forest

In the Steiner tree problem, we are given a graph $G = (V, E)$, a subset $S \subseteq V$ known as *terminals*, and every edge $e = \{u, v\}$ has a length $d(u, v) \ge 0$. Our goal is to find a tree $T$ in $G$ with minimum cost $d(T) = \sum_{e \in T} d(e)$ whose vertex set contains all of $S$. (Vertices of $T$ in $V \setminus S$ are known as *Steiner vertices*). In the metric version, we assume $d$ satisfies the triangle inequality.

This problem is a generalization of both the $s$-$t$ shortest path problem ($|S| = 2$) and the minimum spanning tree problem ($S = V$). However, unlike these two special cases, the Steiner Tree problem is NP-hard. In fact, Chlebík and Chlebíková [CC08] showed that even obtaining a 96/95-approximation is NP-hard.

**A 2-approximation for metric Steiner tree:** Let $G'$ be a complete graph on $S$; the length of edge $\{s_i, s_j\}$ is the length of the shortest path between $s_i$ and $s_j$ in $G$. Let $T'$ be a minimum spanning tree (MST) of $G'$, and let $T$ be the corresponding subgraph of $G$, with edges removed as necessary so that $T$ is a Steiner tree and $d(T) \leq d(T')$.

We now bound the cost of $d(T')$. Let $T^*$ denote the optimal Steiner tree, and let $P^*$ denote a path through the terminals in $G$ obtained by shortcutting through $T^*$. By the triangle inequality, $d(P^*) \leq 2d(T^*)$. Since $P^*$ corresponds to a spanning tree in $G'$, we have $d(T') \leq d(P^*) \leq 2d(T^*)$.

### 3.1 A primal-dual approach to Steiner Forest

The Steiner forest problem is the following generalization of Steiner tree: we are given a graph $G = (V, E)$ where every edge $e$ has cost $d(e)$ and a set of $k$ terminal *pairs* $(s_i, t_i)$ for $i = 1, \ldots, k$. Our goal is to compute a forest $F$ in $G$ such that connects $s_i$ to $t_i$ for every $i$ and minimizes $d(F) = \sum_{e \in F} d(e)$. (The Steiner tree problem is the special case where every $t_i$ is a fixed terminal.)

In this section, we describe a primal-dual algorithm proposed by Agarwal et al. [AKR95] and simplified and generalized by Goemans and Williamson [GW95]. Note that the term "primal-dual" was coined by Goemans and Williamson, who pioneered the usage of this technique in the design of approximation algorithms. We describe our algorithm in two separate phases.

**Balls and moats:** The key idea behind this algorithm is the notion of balls and moats. For any vertex $u$ and number $r \geq 0$, let $B_r(u)$ denote the region of $G$ within radius $r$ from $u$. To connect terminals $s$ and $t$, we will increase $r$ until $B_r(s)$ and $B_r(t)$ intersect i.e., when $r = d(s,t)/2$.

A *moat* is formed when two balls that do not correspond to a terminal pair, say $B_r(s_1)$ and $B_r(s_2)$ collide. In this case, the moat is the union of these two balls, and the growth of the moat is dictated by the growth of $B_r(s_1) \cup B_r(s_2)$. Thus, a ball is essentially a special case of a moat.

**Inactivate moats:** A moat $M$ is *inactive* if every terminal in the moat is satisfied, that is, $|M \cap \{s_i, t_i\}|$ is either 0 or 2 for every $i \in \{1, \ldots, k\}$. As we will see, this determines the stopping condition of our algorithm. (Note that an activate moat can become active if an activate moat expands into it.)

**The algorithm:** The first phase of the algorithm, known as the *forward phase*, is the following: starting with a ball with radius 0 around every terminal vertex, uniformly expand *active* balls/moats until every moat is inactive. At this point, let $F$ be the set of edges that are completely contained in a moat. (Although $F$ is a feasible Steiner forest, its cost can be very high.)

The second phase of the algorithm is known as *reverse delete*. It is a simple greedy heuristic that reduces the size of $F$: for each $e \in F$ (considered in reverse order of addition to $F$), remove $e$ from $F$ if $F \setminus \{e\}$ is feasible. In the next lecture, we'll show how to interpret the radii of the balls as dual values in the corresponding linear program.

## 4 Summary

In this lecture, we saw an MST-based 2-approximation for both metric TSP and the metric Steiner tree problem. We improved the former to a 3/2-approximation using Christofides algorithm, and we began looking at a primal-dual algorithm for the Steiner forest problem. In the next lecture, we will continue studying the primal-dual algorithm for Steiner forest.

## References


[AGM$^+$10] Arash Asadpour, Michel X Goemans, Aleksander Mądry, Shayan Oveis Gharan, and Amin Saberi. An o (log n/log log n)-approximation algorithm for the asymmetric traveling salesman problem. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 379–389. SIAM, 2010.

[AKR95] Ajit Agrawal, Philip Klein, and R Ravi. When trees collide: An approximation algorithm for the generalized steiner problem on networks. *SIAM Journal on Computing*, 24(3):440–456, 1995.

[CC08] Miroslav Chlebík and Janka Chlebíková. The steiner tree problem on graphs: Inapproximability results. *Theoretical Computer Science*, 406(3):207–214, 2008.

[FGM82] Alan M Frieze, Giulia Galbiati, and Francesco Maffioli. On the worst-case performance of some algorithms for the asymmetric traveling salesman problem. *Networks*, 12(1):23–39, 1982.

[GW95] Michel X Goemans and David P Williamson. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24(2):296–317, 1995.

[STV18] Ola Svensson, Jakub Tarnawski, and László A. Végh. A constant-factor approximation algorithm for the asymmetric traveling salesman problem. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, pages 204–213, New York, NY, USA, 2018. ACM.