Chapter 9

Traveling Salesman

Here we study the classical Traveling Salesman problem: Given a complete graph $G = (V, E)$ on n vertices with non-negative edge cost $c : E \to \mathbb{R}^+$ find a *tour* T, i.e., a cycle in G which visits each vertex $v \in V$ exactly once, having minimum $\text{cost}(T) = \sum_{e \in T} c(e)$.

9.1 Hardness of Approximation

The disappointing first fact about Traveling Salesman is that, without assumptions on the edge-cost, the problem can not be approximated, unless $\mathsf{P} = \mathsf{NP}$.

Theorem 9.1. *Let $\alpha(n)$ be any polynomial time computable function. Then there is no $\alpha(n)$-approximation algorithm for* Traveling Salesman, *unless* $\mathsf{P} = \mathsf{NP}$.

Proof. For sake of contradiction, assume that there is a polynomial time $\alpha(n)$-approximation algorithm Alg. We show that such an algorithm can be used to decide the NP-complete problem of deciding the Hamiltionian Cycle problem: Given a graph $H = (V, E)$ on n vertices, decide if H has a tour.

We transform any input H for the Hamiltionian Cycle problem into a graph G for the Traveling Salesman problem as follows: $V(G) = V(H)$, $E(G) = \{uv : u, v \in V\}$, and

$$c(e) = \begin{cases} 1 & \text{if } e \in E(H), \\ \alpha(n) \cdot n & \text{otherwise.} \end{cases}$$

If H has no tour, then any tour T of G has cost $\text{cost}(T) > \alpha(n) \cdot n$. This includes the tour found by Alg.

If H has a tour, then G has a tour T^* with $\text{cost}(T^*) = n$. Since Alg is a $\alpha(n)$-approximation algorithm, it produces a tour T with $\text{cost}(T) \leq \alpha(n) \cdot n$. Clearly T is also a tour in H, since it can not traverse any edge with cost $\alpha(n) \cdot n$ in G.

Therefore, Alg is a polynomial time algorithm which can be used to decide the Hamilton Cycle problem contradicting $\mathsf{P} \neq \mathsf{NP}$. □

9.2 Metric Traveling Salesman

As we have seen that the general Traveling Salesman problem can not be approximated, unless $\mathsf{P} = \mathsf{NP}$, we introduce assumptions on the edge-cost. A natural choice, called Metric Traveling Salesman is that the cost satisfy the triangle inequality $c(uv) \leq c(uw) + c(wv)$ for all $u, v, w \in V$. The problem is still NP-hard but allows constant factor approximations.

Spanning Tree Heuristic

Observe that the cost of any minimum spanning tree S of G is a lower bound for the optimal tour T^*, i.e., $\text{cost}(T^*) \geq \text{cost}(S)$. This is because the removal of any edge in any tour T, including T^*, yields a spanning tree of G.

A graph G is called *Eulerian*, if all its degrees are even. In this case it has an *Euler tour*, i.e., is possible to traverse the edges of G in a cycle that visits each edge exactly once. A respective algorithm can be implemented to run in $O(n+m)$ time.

Algorithm 9.1 Spanning Tree Heuristic

Input. Complete graph $G = (V, E)$, $c : E \to \mathbb{R}^+$

Output. Tour T in G

Step 1. Compute minimum spanning tree S of G.

Step 2. Double the edges of S to obtain Eulerian graph D.

Step 3. Compute Euler tour Q in D.

Step 4. Compute tour T in G that traveres the vertices V in the order of their *first* appearance in Q.

Step 5. Return T.

Theorem 9.2. *The algorithm* Spanning Tree Heuristic *is a* 2*-approximation for* Metric Traveling Salesman.

Proof. Let T^* be an optimal tour in G. We clearly have $\text{cost}(T^*) \geq \text{cost}(S) = \text{cost}(Q)/2$. In the construction of T, consider a situation, where T traverses an edge uv, while Q traverses a path $uw_1, w_1w_2, \ldots, w_{k-1}w_k, w_kv$. By the triangle inequality (applied multiple times if necessary), we have

$$c(uv) \leq c(uw_1) + \cdots + c(w_kv).$$

Therefore $\text{cost}(T) \leq \text{cost}(Q)$, which yields

$$\text{cost}(T) \leq 2 \cdot \text{cost}(T^*).$$

It remains to show that T is indeed a tour in G. Since T visits each vertex in the order of first appearance in Q, i.e., at most once, and since Q visits each vertex at least once as S is a spanning tree, T visits each vertex exactly once. $\square$

It is an exercise to give a tight example for this algorithm.

Christofides Algorithm

In the above heuristic, we doubled all the edges of the spanning tree S in order to obtain an Eulerian graph D. Maybe there is a smarter way of finding such a graph. Recall that a graph is Eulerian if all its degrees are even. Thus we do not have to be concerned about the vertices with even degree in the spanning tree S. Also recall that the number of vertices with odd degree in any graph is even k, say. Our goal will be to start with the spanning tree S and obtain a graph D by adding a collection of edges (a matching) $e_1, \ldots, e_{k/2}$ between the vertices of odd degree in S. Observe that the even degrees in S remain even in D and that the odd degrees in S become also even in D. Thus D is an Eulerian graph. We want to find the cheapest possible matching of such kind.

Algorithm 9.2 CHRISTOFIDES

Input. Complete graph $G = (V, E)$, $c : E \to \mathbb{R}^+$

Output. Tour T in G

Step 1. Compute minimum spanning tree S of G.

Step 2. Let $W \subseteq V$ be the odd-degree vertices in S. Let $H = (W, F)$, where $F = \{vw : v, w \in W\}$.

Step 3. Compute minimum cost perfect matching M in H (using the cost function c).

Step 4. Let $D = S \cup M$ and compute an Euler tour Q in D.

Step 5. Compute tour T in G that traveres the vertices V in the order of their *first* appearance in Q.

Step 6. Return T.

Lemma 9.3. *Let $W \subseteq V$ such that $|W|$ is even, let $H = (W, F)$, where $F = \{vw : v, w \in W\}$, and let M be a minimum cost perfect matching in H. Then*

$$\mathrm{cost}(T^*) \geq 2 \cdot \mathrm{cost}(M).$$

Proof. First observe that H has a perfect matching since the graph is complete and has an even number of vertices. Let T^* be an optimal tour in G and let T be the tour in H which visits the vertices W in the same order as in T^*. For every edge $uv \in T$ there is a path $uw_1, \ldots, w_k v \in T^*$ and by the triangle inequality we have $c(uv) \leq c(uw_1) + \cdots + c(w_k v)$. Therefore $\mathrm{cost}(T^*) \geq \mathrm{cost}(T)$. On the other hand, T is a cycle with even number of edges. Thus, by considering the edges alternatingly, T can be decomposed into two matchings M_1 and M_2. Clearly $\mathrm{cost}(M_1) \geq \mathrm{cost}(M)$ and $\mathrm{cost}(M_2) \geq \mathrm{cost}(M)$, which yields

$$\mathrm{cost}(T^*) \geq \mathrm{cost}(T) = \mathrm{cost}(M_1) + \mathrm{cost}(M_2) \geq 2 \cdot \mathrm{cost}(M)$$

as claimed. □

Theorem 9.4. *The algorithm* CHRISTOFIDES *is a 3/2-approximation for* METRIC TRAVELING SALESMAN.

Proof. We have already argued that the graph D constructed is an Eulerian graph and, by the triangle inequality, the constructed tour T has $\mathrm{cost}(T) \leq \mathrm{cost}(D)$. Then we have

$$\mathrm{cost}(T) \leq \mathrm{cost}(D) = \mathrm{cost}(S) + \mathrm{cost}(M) \leq \mathrm{cost}(T^*) + \frac{1}{2} \cdot \mathrm{cost}(T^*) = \frac{3}{2} \cdot \mathrm{cost}(T^*)$$

as claimed. □