ORIE 6300 Mathematical Programming I December 3, 2014

Recitation 12

Lecturer: Chaoxu Tong *Topic: Approximation Algorithms for the Traveling Salesman Problem*

1 Approximation Algorithms

There are several approaches for dealing with NP-complete problems[1]:

1. Identify special cases that are easy to solve: for example, the INDEPENDENT SET problem is NP-complete on arbitrary graph, but easy on trees: recursively pick all leaves and delete its neighbor give the largest possible independent set.
2. Heuristics: often using local search technique
3. Approximation Algorithm: algorithms that runs in polynomial time and always produce a solution close to the optimal.

We called an algorithm an **α-approximation algorithm** if it runs in polynomial time, and always outputs a solution that is at most $\alpha \cdot OPT$ for a minimization problem (or at least $1/\alpha \cdot OPT$ for a maximization problem), where OPT denote the optimal value. Note that we must have $\alpha \geq 1$ and it does not have to be a constant. Here we will discuss approximation algorithms for the Traveling Salesman Problem.

Recall in the Traveling Salesman Problem, we are given a complete graph G with nonnegative edge costs, and the goal is to find a minimum-cost cycle that visits every vertex exactly once.

First we will show that there is no constant factor approximation algorithm for TSP in general. We will do so via a reduction from the **Hamiltonian Cycle Problem**: Given a graph, decide if there is a cycle that go through all vertices exactly once. The Hamiltonian Cycle Problem is NP-complete.

Claim 1 *There is no α-approximation algorithm for Traveling Salesman Problem where α is a constant.*

Proof: Suppose for contradiction that such an α-approximation algorithm exists. We will use this algorithm to solve the Hamiltonian Cycle Problem in polynomial time, giving a contradiction.

Given an instance of Hamiltonian Cycle with graph G, construct an instance of TSP with graph G' that is a complete graph on the same set of vertex. Set the edge costs as follows: if an edge e is in G then let $c_e = 1$, otherwise set $c_e = \alpha n$, where n is the number of vertices in G.

Notice that if G has a Hamiltonian cycle, then the optimal tour in G' has cost n by using only the edges in G that are in the Hamiltonian Cycle.

Otherwise if G does not have a Hamiltonian cycle, then the optimal tour in G' must contain some edge not in G and hence has cost $> \alpha n$.

[1]Based on previous note by Maurice Cheung

Since the cost of the optimal tour differ by a factor of more than α depending on if G has a Hamiltonian cycle or not, the α-approximation will always be able to distinguish between the two cases, hence decide if G has a Hamiltonian Cycle. This give us the desired contradiction. $\square$

This lead us to consider the special case of **Metric TSP**. In the metric TSP, edge costs have to satisfy the triangle inequality, i.e. for any three vertices x, y, z, $c_{xz} \leq c_{xy} + c_{yz}$

We note that the metric TSP is still NP-complete. In the proof of the previous claim, set $c_e = 2$ if $e \notin G$. Then the edge costs satisfy triangle inequality, and using the same argument as in the proof above we will that any algorithm that solves the metric TSP can solve the Hamiltonian cycle problem.

2 A 2-Approximation Algorithm for Metric TSP

The key to designing approximation algorithm is to obtain a bound on the optimal value OPT. In the case of TSP, the minimum spanning tree gives a lower bound on OPT

Claim 2 *The cost of a Minimum Spanning Tree is no greater than the cost of an optimal tour*

Proof: Notice that deleting an edge from a given tour gives us a spanning tree, hence the minimum spanning tree give us a lower bound on OPT, the cost of the optimal tour. $\square$

This gives us a simple 2-approximation algorithm for metric TSP. We need the following definition to define the algorithm.

Definition 1 *A (multi-) graph is Eulerian if it has a closed walk that uses every edge exactly once.*

Fact 1 *A connected graph is Eulerian if and only if every vertex has even degree.*

The algorithm is as follows:

1. Find a minimum spanning tree of G
2. Duplicate each edge in the minimum spanning tree to obtain a Eulerian graph
3. Find a Eulerian tour $\mathcal{J}$ of the Eulerian graph
4. Convert $\mathcal{J}$ to a tour $\mathcal{T}$ by going through the vertices in the same order of $\mathcal{T}$, skipping vertices that were already visited

Claim 3 *The above algorithm is a 2-approximation algorithm for metric TSP*

Proof: As noted above, cost$(MST) \leq OPT$. Since $\mathcal{J}$ contains two copies of each edge in MST, cost$(\mathcal{J})$= $2 \cdot$ cost(MST). Finally, by triangle inequality, shortcutting previously visited vertices does not increase the cost. Hence we have cost$(\mathcal{T}) \leq 2 \cdot$ cost$(MST) \leq OPT$. $\square$

3 Christofides' Algorithm

Christofides' Algorithm is a 3/2-approximation algorithm for metric TSP. It is very similar to the 2-approximation algorithm above. The improvement comes from finding a better way to construct the Eulerian graph. Since a graph is Eulerian if and only if every vertices has even degree, we only need to be concerned about the vertices that have odd degree in the MST. Now if we add to the MST a minimum-cost perfect matching on V', every vertex will have an even degree, and we get an Eulerian graph. To bound the approximation ratio of this algorithm, it suffices to bound the cost of the minimum cost perfect matching in terms of OPT.

Claim 4 *Let $V' \subseteq V$ such that $|V'|$ is even, and let M be a minimum cost perfect matching on V'. Then cost(M) $\leq OPT/2$.*

Proof: Consider an optimal TSP tour. Let $\mathcal{T}'$ be the tour on V' by shortcutting vertices in $V\ V'$. By triangle inequality, cost$(\mathcal{T}') \leq OPT$. Now $\mathcal{T}'$ is the union of two perfect matchings on V', each consisting of alternate edges on $\mathcal{T}'$. Since M is a minimum cost perfect matching, we have 2cost$(M) \leq OPT$ $\square$

Christofides' Algorithm is as follows:

1. Find a minimum spanning tree of G
2. Compute a minimum cost perfect matching M on the set of odd-degree vertices of MST. Add M to MST to obtain an Eulerian graph
3. Find a Eulerian tour $\mathcal{J}$ of the Eulearian graph
4. Convert $\mathcal{J}$ to a tour $\mathcal{T}$ by going through the vertices in the same order of $\mathcal{T}$, skipping vertices that were already visited.