Homework Set 6

Miles Van de Wetering, Charles Hill, Cierra Shawe

March 16, 2017

Problem 3

Let $G = (V, E)$ be a complete undirected graph where the edge lengths $w(e)$ for every $e \in E$ are elements of $\{1, 2\}$. This graph satisfies clearly the triangle inequality. Give a $4/3$ factor approximation algorithm for TSP in this special class of graphs.

Solution

Grab any vertex $s \in V$ and walk along weight 1 edges only visiting unmarked nodes, marking nodes as we go, until either no weight 1 edge is available (in this case walk along one final edge of weight 2 back to s) or we arrive back at s by a weight 1 edge. Repeat until no unmarked vertices exist. If no initial edge of weight 1 exists incident to s, take any weight 2 edge to an unmarked node u and continue as normal. If no proper weight 1 edge exists incident to u, take any weight 2 edge and continue as normal.
When this process is complete, we will have a forest of at most $\frac{|V|}{3}$ cycles. Next merge the cycles one by one, each time deleting the maximum edges $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$ from each cycle and connecting the two cycles as cheaply as possible using $e_3 = (u_1, u_2)$ and $e_4 = (v_1, v_2)$ or $e_3 = (u_1, v_2)$ and $e_4 = (v_1, u_2)$. The optimal solution would be the case where every edge has a weight of one, creating a total weight of V-1. Otherwise, for every sub-cycle there is at most a "waste" of one, by taking a wrong turn (choosing the wrong weight). There is at most $\frac{|V|}{3}$ sub-cycles with this amount of waste, so it will be at worst $\frac{|4V|}{3}$ of the optimal solution.

An imperative description of the algorithm first, followed by an explanation of the steps.
Let $M \leftarrow 1$
Let $V_s \leftarrow \emptyset$
(a) While there exist unmarked vertices:
Let v be any unmarked vertex.
Let E_M be every edge of weight 1 incident to v. We will keep piecing together these edges (and perhaps more that we find) until we have reached every vertex possible using only weight 1 edges.
(b) While $E_M \neq \emptyset$:
Pick any edge $e = (x, y) \in E_M$. If x or y are unmarked (note that one of these two will always already be marked):
Mark x (or y, but I will use x for simplicity), labeling it as part of subset M.
Push all weight 1 edges $e = (x, y)$ where y is unmarked into E_M. If no such edges exist, then mark this vertex x as being the terminus of the M subtree. Mark only one such terminus per subtree, and push this terminus into the
Otherwise, delete e altogether since it w.
(b) ends
Increment M
This will create a minimum forest connecting as many vertices as possible using edges of weight 1. It is important to note that after each iteration of this while loop, the subset M will contain every vertex reachable

Problem 4

The Steiner tree problem is as follows. Given $G = (V, E)$ with positive edge weights, and whose vertices are partitioned into two sets R (required) and S (Steiner), find a minimum cost tree in G that contains all required vertices. Design a 2-approximation algorithm for the Steiner tree problem.

Solution

First compute the $G' = (R \bigcup S, E')$ as the metric closure (a complete graph) of G with any all-source shortest path algorithm. Now, each edge in G' has a weight equivalent to the shortest path between the same two nodes in G, and (if we

are clever and preserve this data) can also contain the information corresponding to the actual edges in that shortest path). Now we may simply run Prim's algorithm to select the minimum spanning tree of $G = (R, E^{'})$. We can demonstrate this is 2-approximation algorithm because; our algorithm will have a "spoke" structure due to the nature of a complete graph, the optimal solution may be better than our solution if some of the paths share sub paths, which reduces the overall path length as we would already have this distance.

Referenced:

https://en.wikipedia.org/wiki/Prim%27s_algorithm
https://en.wikipedia.org/wiki/Minimum_spanning_tree
https://lucatrevisan.wordpress.com/2011/01/13/cs261-lecture-2-steiner-tree-approximation/