We will revisit several themes from the course — graphs, induction, greedy algorithms, matchings — by focusing on a classic problem in computer science: the *traveling salesperson problem* (TSP). TSP has no known polynomial time solution. At present, there is an $O(n^2 2^n)$ dynamic programming algorithm for TSP which improved the brute-force $n!$ solution that considers all permutations of the cities. At present, this dynamic programming procedure is the most efficient algorithm for solving TSP *exactly*[1]. However, there are several well-known polynomial-time algorithms for solving TSP *approximately*. That is, finding a solution that is not necessarily optimal, but provably within some constant, say α, of optimal. Here we will develop a 3/2-approximation due to Christofedes. Given any instance of TSP where the distances obey the triangle inequality, Christofedes' algorithm returns a solution with cost at most $3/2 \cdot OPT$ where OPT is the cost of an optimal, minimum cost cycle.

Question 1. *Given a complete, undirected graph $G = (V, E)$ with non-negative and real-valued edges costs $c : E \to \mathbb{R}$, the goal of TSP is to find a minimum cost cycle that visits every vertex in V exactly once. The cost of the cycle is the sum of the costs of the edges in the cycle.* Metric TSP *refers to a class of TSP instances where the edge costs obey the triangle inequality. This means that shortcuts between two cities don't exist; it's always best to take the direct route. Formally the triangle inequality means that*

$$c(u, v) \leq c(u, w) + c(w, v) \qquad \text{for all } u, v, w \in V.$$

We will develop a polynomial-time 3/2-approximation for Metric TSP. In all parts, $G = (V, E)$ is the complete, undirected input graph. V is the set of nodes and E is the set of edges where $|V| = n$ and $|E| = m$. The function c gives the real-valued edge costs that obey the triangle inequality. Sometimes we extend c to a set of edges $E' \subseteq E$ so that $c(E') = \sum_{e \in E'} c(e)$. Please answer the following questions clearly and concisely. If you get stuck on a particular part, move on — you can assume previous results and still make progress.

(a) *Let C^* be the cost of the minimum cost cycle in G. Let T be a minimum spanning tree in G with cost $c(T)$. Show that $c(T) \leq C^*$.*

(b) *Use induction to show that all trees have an even number of odd-degree nodes.*

An undirected multigraph H is an undirected graph with parallel edges. That is, pairs of nodes may have multiple edges between them. An Euler tour *of a connected, undirected multigraph $H = (V', E')$ is a cycle that traverses each edge of H exactly once, although it may visit a vertex more than once. We denote a cycle as a list of vertices, $u_0 \ldots u_{m'}$ where $u_0 = u_{m'}$ and for all $0 \leq i < m'$, (u_i, u_{i+1}) is an edge in E'. Note that an Euler tour has length $m' = |E'|$.*

(c) *Earlier in the semester you showed that a graph has an Euler tour if and only if the degree of every vertex in the graph is even. Quickly argue why your analysis also holds for multi graphs.*

(d) *Similarly, quickly argue that your linear time algorithm for finding Euler tours in graphs extends without alteration to multigraphs.*

A minimum weight perfect matching in a graph G' with n nodes and non-negative edge costs is a matching M of size $n/2$ with minimum cost $c(M)$. Edmonds showed in the 60's how to find a minimum weight perfect matching of a graph in $O(n^4)$ time. Gabow recently improved this running time to $O(n(m + n \log n))$.

(e) *Use parts (a) and (b) along with Gabow's algorithm as a black box, to produce first, a minimum weight perfect matching M, and second, a multigraph $H = (V, E')$ where all the nodes in V have even degree. Note that V refers to the same set of nodes as the input graph.*

(f) *Show that $c(M) \leq 1/2 \cdot C^*$. Using part (a), conclude that $c(E') \leq 3/2 \cdot C^*$.*

(g) *Use parts (c) and (d) along with the multigraph H (and the fact that c obeys the triangle inequality) to produce a cycle $v_0 v_1 \ldots v_n$ that visits every vertex in V exactly once. Conclude that this cycle is a 3/2-approximation for the metric TSP problem.*

Question 2. *I found this homework:*

[1] Actually, four years ago there was a slight improvement which was the first progress in over 20 years