Lecture 21, Nov 27, 2023

The Travelling Salesman Problem and Approximate Algorithms

- Travelling Salesman Problem: Given an undirected complete graph $G = (V, E)$ (i.e. edge between every pair of nodes) with non-negative weights, find a *tour* of G of minimum total edge cost
  - A *tour* is a cycle that visits every node of G exactly once
  - Such a minimum cost tour is known as a *TSP tour*
- The brute force algorithm is to find all tours of G and select the minimum cost one, but this takes exponential time
- The TSP problem is NP-hard, i.e. it is at least as hard as all NP-complete problems, so it's very unlikely that we can find a polynomial time algorithm
- The Δ-TSP (also known as triangle-TSP or metric-TSP) is a special case of the TSP problem where the edge costs satisfy the triangle inequality, i.e. for all nodes u, v, w of G, we have $c(u, w) \leq c(u, v) + c(v, w)$
  - e.g. if nodes are locations in space and the edge costs are Euclidean distances, then we have a Δ-TSP problem
  - This is easier to solve than TSP in general, but it is still NP-hard
  - However, we can find a polynomial-time approximate algorithm, which finds a tour whose cost is within a certain constant factor of the optimal tour
- Lemma: For any complete undirected G, the cost of an MST of the graph is always less than or equal to a minimum cost tour
  - Consider any TSP tour of G, which will be a cycle; remove one edge, and now we have a spanning tree
  - Therefore there will always be spanning trees that cost less than the TSP tour of G, so the MST will also cost less than the TSP tour
- The approximate Δ-TSP algorithm is as follows:
  1. Find an MST of G (this takes $O(m \log n)$ time with e.g. Kruskal's algorithm)
  2. Do a full walk of the MST (i.e. like a DFS); this gives a sequence of nodes C
     - C is a cycle that uses each MST edge twice, so it has twice the cost of the MST
     - Note that C is a cycle but not a tour, since it visits some nodes multiple times
  3. Transform C into a tour C^* by using shortcuts to remove repeated nodes
     - Whenever we come back to a node that we have previously visited, we instead skip it and move to the next node directly
     - Using the triangle inequality assumption, we know that this does not increase the cost
     - Since we never increased the cost by using shortcuts, the cost C^* is less than the cost of C, which is twice the cost of the MST
     - By the lemma above, we know the MST cost $<$ TSP tour cost, so the cost of C^* is less than twice the TSP tour cost
- Better algorithms can improve this factor of 2 to 1.5