CS 170 Dis 10

Released on 2017-04-04

1 Optimization versus Search

Recall the following definition of the Traveling Salesman Problem, which we will call TSP. We are given a complete graph G of whose edges are weighted and a budget b. We want to find a tour (i.e., path) which passes through all the nodes of G and has length $\leq b$, if such a tour exists.

The optimization version of this problem (which we call TSP-OPT) asks directly for the shortest tour.

(a) Show that if TSP can be solved in polynomial time, then so can TSP-OPT.

(b) Do the reverse of (a), namely, show that if TSP-OPT can be solved in polynomial time, then so can TSP.

Solution:

(a) Do a binary search over all possible lengths of the optimal tour, going from 0 to the sum of all distances. Note that binary search is necessary here and we can't just increment the value of b by 1 each time since the sum of all distance is exponential in the size of the input.

(b) Run tsp-opt and return the optimal value. If it is greater than b, then no solution exists, by the definition of optimality.

2 A Faulty Reduction

In the Redrata path problem (AKA the Hamiltonian Path Problem), we are given a graph G and want to find if there is a path in G that uses each vertex exactly once.

What is wrong with the following argument?

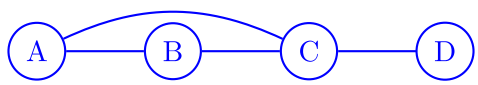
We will show that Undirected Rudrata Path can be reduced to Longest Path in a DAG. Given a graph G, use DFS to find a traversal of G and assign directions to all the edges in G based on this traversal (i.e. edges will point in the same direction they were traversed and back edges will be omitted). This gives a DAG. If the longest path in this DAG has $|V| - 1$ edges then there is a Rudrata path in G since any simple path with $|V| - 1$ edges must visit every vertex.

Solution:

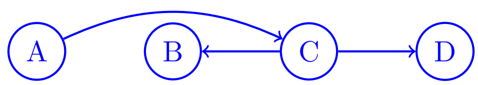
It is true that if the longest path in the DAG has length $|V| - 1$ then there is a Rudrata path in G. However, to prove a reduction correct, **you have to prove both directions**. That is, if you have reduced problem A to problem B by transforming instance I to instance

I' then you should prove that I has a solution **if and only if** I' has a solution. In the above "reduction," one direction doesn't hold. Specifically, if G has a Rudrata path then the DAG that we produce does not necessarily have a path of length $|V| - 1$–it depends on how we choose directions for the edges.

For a concrete counterexample, consider the following graph:



It is possible that when traversing this graph by DFS, node C will be encountered before node B and thus the DAG produced will be



which does not have a path of length 3 even though the original graph did have a Rudrata path.

3 Hitting Set

In the Hitting Set Problem, we are given a family of sets $\{S_1, S_2, \ldots, S_n\}$ and a budget b, and we wish to find a set H of size $\leq b$ which intersects every S_i, if such an H exists. In other words, we want $H \cap S_i \neq \emptyset$ for all i.
Show that the Hitting Set Problem is NP-complete.

Solution: This is a generalization of the Vertex-Cover Problem, which we saw last section. Given a graph G, consider each edge $e = (u, v)$ as a set containing the elements u and v. Then, finding a hitting set of size at most b in this particular family of sets is the same as finding a vertex cover of size at most b for the given graph.