CS140: Algorithms

Spring 2001

Solutions to Assignment 11

1. (Trivial) The Largest Common Subgraph problem is defined as follows.

   Input: Graphs G_1 and G_2 and integer k,

   Question: Is there a graph G containing at least k edges that is a subgraph of both G_1 and G_2.

   (a) Give a Yes-instance of LCS.

   (b) Give a No-instance of LCS.

   (c) Cast this problem as a search problem. Show that the search problem is polynomial-time equivalent to the decision problem.

   A search version is to find the largest subgraph (i.e. largest number of edges) that is common to G_1 and G_2. To solve this problem, we'll first find the largest K such that the decision algorithm answers yes on input G_1, G_2, K. Next to find the appropriate subgraphs we remove an unmarked edge from G_1 or G_2. If the decision algorithm answers no to the modified input we restore the edge and mark it necessary. We continue until all remaining edges are marked necessary.

   (d) Show that LCS is in NP.

   Given the subgraph and the mapping of its vertices to the corresponding vertices of G_1 and G_2, we can check that the subgraph exists in G_1 and G_2 and that it has at least K edges.

   (e) Show that LCS is NP-complete by giving a reduction from one of the NP-complete problems we discussed in class.

   We give a reduction from Clique. Given an instance G', K' of Clique we set $G_1 = G'$, G_2 to a complete graph on K' vertices, and $K = K'(K'-1)/2$. G' has a clique of size K if an only if G_1 has a subgraph isomorphic to G_2, which has exactly K' edges.

2. (Easy) Repeat the above exercise for the Minimum Sum of Squares problem defined below.

   Input: A set A of integers and two additional integers K and J.

   Question: Can the elements of A be partitioned into K disjoint sets $A_1, A_2, \ldots, A_K$ such that $\sum_{i=1}^{K}(\sum_{s \in A_i} s)^2 \leq J$?

   Yes-instance: $\{2, 3, 4\}$, $K = 3$, $J = 1000$.
   No-instance: $\{2, 3, 4\}$, $K = 3$, $J = 10$

A search version of the problem is to actually find the partition. To solve this problem we first check if a solution exists using the decision algorithm. If not, we so declare and then halt. If it does, we choose a pair of elements and replace them by their sum. If the resulting instance is now a no-instance, we replace the sum by the individual elements and note that the pair cannot be in the same partition set. We repeat on a new pair until our underlying set has exactly K-elements. We then recover the elements of the underlying set that were summed to produce each such element.

If we are given a proposed parition we can check that it is a valid partition of the underlying set and that the squares of the sums of the partition set are less than J in polynomial time.

To show the problem is NP-hard we do a reduction from Parition. Given an instance S of partition, we set $A = S$, $K = 2$ and $J = 2T^2$ where $T = (\sum_{s \in S} s)/2$.

Suppose S is a yes-instancer of Partition. Then S can be partitioned into sets S_1 and S_2 with equal sums. But then the sum of the squares of S_1 and S_2 is exactly J. So A, K, J is a yes-instance of MSS.

Now suppose that A can be partitioned into A_1 and A_2 so that the sum of their squares is no more than J. Without loss of generality assume that the sum of the elements in A_1 is $T-\alpha$ and the sum of the elements in A_2 is $T+\alpha$. Then the sum of the squares of A_1 and A_2 is $(T-\alpha)^2+(T+\alpha)^2 = 2T^2 + 2\alpha^2$. Since this is no more than $J = 2T^2$, it must be the case that $\alpha = 0$. Hence the sum of the elements in A_1 is the same as for A_2. So S is a yes-instance of Partition.

3. Repeat the previous exercise for the Steiner Tree in Graphs problem defined below. (Hint: Reduce from Vertex Cover.)

Input: A graph G, a subset R of the vertices of G, and an integer K.

Question: Is there a subtree (not necessarily a spanning tree) of G that includes the vertices R and at most K edges?

A search version of this problem asks for the smallest Steiner tree for G, R. To find a solution we first ask the decision problem G, R, K for $K = 1, 2, ...$ until we find the smallest K for which the answer is yes. To actually find the Steiner tree we choose an unmarked edge of G and remove it. If the decision algorithm says no we restore the edge and mark it necessary. We continue until we've marked K edges as necessary.

Given a Steiner tree and a mapping of its vertices to the vertices of G, we can check that the tree is a subgraph of G, that it includes all the vertices of R, and that it has no more than K edges in polynomial time.

To show that the problem is NP-hard we give a reduction from Vertex Cover. Let G', K' be an instance of Vertex Cover. The vertices of G are $\{v | v \in G'\} \cup \{v_e | e \in G'\} \cup \{X\}$. The vertex v_e for some $e \in G'$ is connected to the vertices in G corresponding its endpoints in G'. The vertex X is connected to each vertex that corresponds to a vertex in G'. We set R to be the vertex X and the vertices v_e, for $e \in G'$. We set $K = K' + m$, where m is the number of edges of G'.

Suppose that G', K' is a yes-instance of vertex cover. Let W be a vertex cover of size at most K. Note that W is also a subset of vertices of G. Consider the subgraph of G induced by the vertex set $R \cup W$. Since every edge of G' is incident to some vertex in W every vertex v_e is connected to the corresponding vertex in G. In addition, each vertex of W in G is connected to X. Thus this induced subgraph is connected. Any spanning tree of the subgraph contains exactly K edges and thus is an appropriate Steiner tree. Thus G, R, K is a yes-instance of Steiner Tree.

Now suppose that G, R, K is a yes-instance of Steiner tree. Let W be the vertices of G in the tree that are not in R. Note that W is a subset of the vertices of G' and $||W|| \leq K'$. Let $e = (u, v)$ be an edge of G'. Since $v_e \in R$ and v_e is only connected to vertices u and v in G', either u or v must be in W. Thus W is also a vertex cover of G'. Hence G', K' is a yes-instance of vertex cover.