CS140: Algorithms

Fall 2001

Solutions to Assignment 10

1. In the study of NP-completeness we focus on "decision problems," which differ from standard search problems in that we are only required to answer a yes/no question about the input. In practice, of course, we usually want to solve the search variations of these decision problems. In this exercise you'll show that the search variations of several decision problems are really no harder than the decision version. You'll do this by reducing the search problem to the decision problem. Your reductions should be polynomial-time.

   Dominating Set: A dominating set of a graph G is a subset of vertices W such that every vertex of G is either in W or is adjacent to a vertex in W.

   Input: A graph G and integer k
   Question: Does G have a dominating set of size at most k/
   Input: A graph G
   Search Output: A smallest dominating set of G

   Use the decision algorithm on inputs G, K for $K = 1, 2, ...$ to determine the size, M, of the smallest dominating set of G. Now choose an unmarked vertex v of G. (Initially all vertices are unmarked.) Add a new vertex X attached by an edge to v. Ask the decision algorithm if the modified graph has a dominating set of size M. If yes, then mark v as a member of the dominating set and leave X in the graph. If no, mark v as a non-member and remove the vertex X. Repeat until M vertices are marked as members of the dominating set.

   Clique: A clique in a graph G is a subset of vertices W such that each pair of vertices in W is connected by an edge in G.

   Input: A graph G and an integer k
   Question: Does G have a clique of size at least k?
   Input: A graph G
   Search Output: A largest clique of G

   Use the decision algorithm on inputs G, K for $K = 1, 2, ...$ to determine the size, M, of the largest clique of G. Now choose an unmarked vertex v of G and eliminate it. Ask the decision algorithm if the modified graph has a clique of size M. If no, then reinstate v and mark it as a member of the clique. Repeat until M vertices are marked.

   Hamiltonian Cycle: A Hamiltonian cycle of a graph G is a simple cycle that includes every vertex.

Input: A graph G

Question: Does G contain a Hamiltonian cycle?

Input: A graph G

Search Output: A Hamiltonian cycle of G if one exists and otherwise "No Hamiltonian cycle"

Use the decision algorithm to determine whether or not a Hamiltonian cycle exists. If not, then answer accordingly. If so, then choose an unmarked edge e of G and eliminate it. Ask the decision algorithm if the resulting graph has a Hamiltonian cycle. If not, reinstate e and mark it as a member of the Hamiltonian cycle. Repeat until n edges are marked, where $n = ||V||$.

3-coloring: A 3-coloring of graph G is an assignment of a unique color in {red, blue, green } to each vertex of G so that the endpoints of each edge are colored differently.

Input: A graph G

Question: Can G be 3-colored?

Input: A graph G

Search Output: A 3-coloring of G if one exists and otherwise "No 3-coloring"

Use the decision algorithm to determine if the graph is 3-colorable. If no,t then answer accordingly. If so, then add three new vertices, named R, B, and G,, connected in a triangle. We'll begin by coloring these vertices, as their names suggest, red, blue, and green. Next choose an uncolored vertex v. Add edges between v and two vertices from $\{R, B, G\}$. Ask the decision algorithm if the modified graph is 3-colorable. If it is, then color v the color of the unused vertex in $\{R, B, G\}$. If the graph is no longer 3-colorable, remove the added edges and repeat using a different pair from $\{R, G, B\}$. Repeat until all vertices are colored. (Note – the partial coloring is not input to the decision algorithm.)

3 Dimensional Matching: Let U, V, and W be disjoint sets such that $||U|| = ||V|| = ||W|| = q$. Let C be a subset of $U \times V \times W$; in other words C is a collection of triples (u, v, w), where $u \in U$, $v \in V$, $w \in W$. A 3 dimensional matching is a subset of C such that each element of U, V and W occurs in exactly one triple of C.

Input: Disjoint sets U, V, and W, where $||U|| = ||V|| = ||W|| = q$ and a subset C of $U \times V \times W$.

Question: Does C contain a three-dimensional matching?

Input: Disjoint sets U, V, and W, where $||U|| = ||V|| = ||W|| = q$ and a subset C of $U \times V \times W$.

Search Output: A three-dimensional matching from C, if one exists, and otherwise a statement that one does not exit.

Use the decision algorithm to determine if there is a three-dimensional matching. If not, answer accordingly. Otherwise, remove an unmarked triple from C. Ask the decision algorithm if the modified instance has a 3DM. If not, restore the triple and mark it as a member of the matching. Repeat until q triples are marked.

2. The Isomorphic Subtree problem is defined as follows.

   Input: A graph G and a tree T.

   Question: Is there a subset W of vertices of G such that the subgraph of G induced by W is exactly T?

   (a) Show that the Isomorphic Subtree problem is in NP.
   Solution left to the student.

   (b) Give a yes-instance of the problem.
   Solution left to the student.

   (c) Give a no-instance of the problem.
   Solution left to the student.

   (d) Show that the Isomorphic Subtree problem is NP-hard by giving a polynomial-time reduction from one of the standard NP-hard problems we've used in class.

      i. Which NP-hard problem will you use?
      Hamiltonian Path

      ii. Given an input X of the selected problem, how do you construct the input G, T for the Isomorphic Subtree problem? Show that the reduction can be done in polynomial-time.
      Given a instance $G' = (V', E')$ of Hamiltonian path we let $G = G'$ and set T to be a simple path on $||V'||$ vertices. The construction is linear in the size of G'.

      iii. Suppose the input X of the selected problem is a yes-instance. Prove that the instance G, T of Isomorphic Subtree created by your reduction is also a yes-instance.
      By definition, G' has a Hamiltonian cycle iff it has a subgraph isomorphic to T.

      iv. Suppose that the instance G, T of Isomorphic Subtree created by your reduction is a yes-instance. Prove that the original instance X of the selected problem is also a yes-instance.
      See above.

3. The Bipartite Cover Problem is defined as follows.

   Input: A bipartite graph $G = (U, V, E)$ and an integer K.

   Question: Is there a subset $W \subseteq U$ such that $||W|| \leq K$ and every vertex in V is adjacent to some vertex in W?

   (a) Show that the Bipartite Cover problem is in NP.
   Solution left to the student.

(b) A search version of this problem takes as input a bipartite graph $G = (U, V, E)$ and outputs a smallest subset $W \subset U$ such that every vertex in V is adjacent to some vertex in U. Give a polynomial-time reduction from this search problem to the decision problem.

Solution left to the student.

(c) Show that the Bipartite Cover problem is NP-hard by giving a polynomial-time reduction from one of the standard NP-hard problems we've used in class.

i. Which NP-hard problem will you use?
Vertex Cover

ii. Given an input X of the selected problem, how do you construct the input G, K for the Bipartite Cover problem? Show that the reduction can be done in polynomial-time.
Let $G' = (V', E'), K'$ be an instance of vertex cover. We construct G by setting $U = V'$, $V = \{x_e | e \in E'\}$, and $E = \{(u, x_e) | u \text{ is an endpoint of } e \text{ in } G'\}$. Then we set $K = K'$.

iii. Suppose the input X of the selected problem is a yes-instance. Prove that the instance G, K of Bipartite Cover created by your reduction is also a yes-instance.
The proof is left to the student.

iv. Suppose that the instance G, T of Bipartite Cover created by your reduction is a yes-instance. Prove that the original instance X of the selected problem is also a yes-instance.
The proof is left to the student.

4. The Shortest Common Supersequence problem is defined as follows.

Input: A set of strings $S = \{s_1, s_2, \ldots, s_n\}$ over an alphabet Σ and an integer K.

Question: Is there a string of length at most K that contains each $s \in S$ as a subsequence?

For example *abcd* is a shortest string that contains *ac* and *bd* as subsequences. (Other strings work as well.)

(a) Find a shortest string that contains *aba*, *cb*, and *caa* as subsequences.
Solution left to the student.

(b) Show that the Shortest Common Supersequence problem is in NP.
Solution left to the student.

(c) Show that the Shortest Common Supersequence problem is NP-hard by giving a reduction from Vertex Cover.

i. Given an instance $G = (V, E), K$ of Vertex Cover, show how to create the instance S, K' of Shortest Common Supersequence.
Since isolated vertices need never be included in a vertex cover, we will assume that every edge of G is incident to some edge.

For each edge (u, v) in G we add the strings uv and vu in S. We set $K' = n + K$, where n is the number of vertices of G.

ii. Suppose that G, K is a yes-instance of Vertex Cover. Prove that the instance S, K' of Shortest Common Supersequence created by your reduction is also a yes-instance.

   Let W be a vertex cover of size K. Let α be a string consisting of the vertices in W in any order. Let β be a string consisting of the vertices of G that are not in W, again in any order. We claim that the string $\alpha\beta\alpha$ is a supersequence of the strings in S and has length K'.

   Let uv be an arbitrary string in S. By construction (u, v) is an edge of G. Since W is a vertex cover of G, either u or v or both are in W. If both are in V then α contains the symbols u and v and so $\alpha\beta\alpha$ contains uv as a subsequence. If only one of u and v are in W then assume, WLOG, that u is in W. Then u is in α and v is in β. As before, $\alpha\beta\alpha$ contains uv as a subsequence. Since $|\alpha| = K$ and $|\beta| = n - K$, the length condition is satisfied.

iii. Suppose that the instance S, K' of Shortest Common Supersequence produced by your reduction is a yes-instance. Prove that the input G, K is a yes-instance of Vertex Cover.

   Let γ be a supersequence of S of length $n + k$. Let W be the vertices of G that occur more than once in γ. Since every vertex of G appears as a string in S, $||W|| \leq K$. Now consider an edge (u, v) of G. By construction γ contains uv and vu as subsequences. Hence it contains uvu or vuv as a subsequence. Therefore either u or v are in W. Thus W is a vertex cover.