CSC373— Algorithm Design, Analysis, and Complexity — Spring 2018

Solutions for Tutorial Exercise 8: Self reduction of Vertex-Cover

---

A polynomial reduction of the VERTEX-COVER search problem to the decision problem is as follows.

```
def S = findMinVC(G)
    # Given undirected graph G = (V, E) find a minimum sized vertex cover S.
    # A polynomial reduction using VERTEX-COVER.
    # Input size: |s| = |V| + |E|.

    # Step 1: Use binary search to solve for minimum size k* of a vertex cover.
    k* = min{k ∈ [0, |V| ] | VERTEX-COVER(G, k) is true}
    # Requires O(log(|V|)) calls to VERTEX-COVER.

    S ← {}
    if k* == 0: # Input graph G must have no edges.
        return S

    V_r ← V; E_r ← E # "Remaining" graph to find a min cover for.
    for v in V:
        # Loop Invariant: All edges in E\E_r are covered by vertices in S, and
        #     G_r = (V_r, E_r) has a VERTEX-COVER of minimum size k* − |S|.

        k ← k* − |S|
        # We must cover all edges in the remaining graph G_r with only k vertices.

        # Build subgraph G' of G_r, with v and all edges terminating at v removed.
        V' ← V_r\{v}
        E_{r,v} ← {e ∈ E_r | e has endpoint v }
        E' ← E_r\E_{r,v}
        G' ← (V', E')

        if VERTEX-COVER(G', k − 1):
            # We can safely add v to S, since G' can be covered in k − 1 vertices
            S ← S ∪ {v}
            if |S| == k*
                break
            V_r ← V'
            E_r ← E'
    Assertion: |S| == k*
    return S
```

Proof of correctness. A simple inductive proof shows the loop invariant must be true. Details omitted. The remaining issue is that to show that when the loop terminates, the assertion at the end is true. That is, is it true that the loop can only exit with the break statement, rather than simply having the for loop exhausting the list of vertices? Note the loop cannot terminate with $|S| > k^*$ (since $|S|$ grows by only one each time the body of the if statement is executed).

1. Question: Complete the proof sketch. Either complete the sketch of this correctness proof or provide a counter-example that shows the above algorithm is incorrect. **Hint:** Consider enhancing the loop invariant. **Soln.** The key to proving this is to enhance the loop invariant. Suppose $(v_1, v_2, \ldots, v_n)$ is the order the vertices in V are iterated over in the for loop. That is, we could replace the for loop with a loop over $j = 1, 2, \ldots, n$, where $n = |V|$ and, at each stage, $v = v_j$.

Loop Invariant v2 $LI(j)$: All edges in $E\backslash E_r$ are covered by vertices in S, $G_r = (V_r, E_r)$ has a VERTEX-COVER of minimum size $k^* - |S|$. Moreover, on iteration j where $v = v_j$, there exists a minimum sized cover C of G such that, for all $1 \le i \le j$, $v_i \in S$ iff $v_i \in C$.

We will next prove this loop invariant $LI(j-1)$ is true at the beginning of each iteration $j = 1, \ldots, n$, and after the n^{th} iteration $LI(n)$ is true, where in all cases here we are assuming the loop has not yet reached the break statement.

Base Case. Upon initialization, $S = \emptyset$, $j = 1$, and it follows that $LI(0)$ is true.

Induction step. Suppose $j \in \{1, 2, \ldots, n\}$ and suppose $LI(j-1)$ is true. Let C be as in the loop invariant (v.2). Note that here we know C and therefore we know S up to this point $j-1$, so we can reconstruct the graph $G_r = (V_r, E_r)$ from these choices for vertices in S (i.e., delete all vertices in S from the original graph and all edges terminating at any vertex in S).

Then there are two major cases. Either $v_j \in C$ or not.

Case 1. Assume $v_j \in C$. Let G' be the graph formed from G_r, in the body of the loop. Consider the test VERTEX-COVER$(G', k-1)$. Since $v_j \in C$ it follows that this test must be true. That is, the remaining elements in C, namely $C \cap V'$, must cover all the edges in G'. Moreover, $|C| = k^*$ and $|S \cup \{v_j\}| = |S| + 1$, so $|C \cap V'| = k - 1$. Therefore the algorithm must add v_j to S and the loop invariant $LI(j)$ follows.

Case 2. Assume $v_j \notin C$. Then there are two sub-cases, both of which involve the G' and k computed in the beginning of the j^{th} execution of the loop body. The two cases are then, either VERTEX-COVER$(G', k-1)$ is false (Case 2a below) or it is true (Case 2b).

Case 2a. Suppose $v_j \notin C$ and VERTEX-COVER$(G', k-1)$ is false. Then, according to the algorithm, v_j is not added to S at this step. And the loop invariant $LI(j)$ must be true.

Case 2b. Suppose $v_j \notin C$ and VERTEX-COVER$(G', k-1)$ is true. Then, according to the algorithm, v_j will be added to S. In particular, the updated S will not be a subset of C, so here we need to use an exchange argument. Specifically we need to argue that there is a another minimum cover, say W, that will satisfy the loop invariant.

We construct a suitable minimum-sized vertex cover W as follows. Since VERTEX-COVER$(G', k-1)$ is true, let W_2 be a vertex cover for G' of size $k-1$. From $LI(j-1)$ it follows that all the edges in $E\backslash(E' \cup E_{r,v})$ are covered by S, and all the edges in $E_{r,v}$ are covered by v_j. Therefore $W = S \cup \{v_j\} \cup W_2$ is a vertex cover for G. Moreover, since these three sets of vertices are disjoint (since they come from vertices v_i with $i < j$, $i = j$, and $i > j$, respectively),

$$\begin{aligned} |W| &= |S| + 1 + (k-1), \\ &= |S| + (k^* - |S|), \quad \text{since the algorithm defines } k = k^* - |S|, \\ &= k^*. \end{aligned}$$

Finally, note that with this definition of W, the loop invariant $LI(j-1)$ ensures that, for each i such that $1 \le i < j$, $v_i \in W$ iff $v_i \in S$. Since W contains v_j, it follows that, for each i such that $1 \le i \le j$, $v_i \in W$ iff $v_i \in S \cup \{v_j\}$, where the latter is just the updated S.

Therefore, if we use this minimum-sized vertex cover W as C in $LI(j)$, it follows that $LI(j)$ is true for this case as well. ■