CS 21 Decidability and Tractability	**Winter 2018**

Solution Set 6

Posted: February 28

If you have not yet turned in the Problem Set, you should not consult these solutions.

1. First, observe that SUBGROUP ISOMORPHISM is in NP, because if we are given a specification of the subgraph of G and the mapping between its vertices and the vertices of H, we can verify in polynomial time that H is indeed isomorphic to the specified subgraph of G.

   To show SUBGRAPH ISOMORPHISM is NP-hard, we reduce from CLIQUE. Given an instance (G, k) of CLIQUE, where G has n vertices, we produce the following instance of SUBGRAPH ISOMORPHISM: $(G, H = K_\ell)$, where K_ℓ is the complete graph on ℓ vertices and $\ell = \min(k, n+1)$. This reduction runs in polynomial time.

   If $k > n$, then (G, k) is a NO instance of CLIQUE and by our choice of ℓ we produce a NO instance of SUBGRAPH ISOMORPHISM, since there can be no K_{n+1} graph within G which has only n vertices.

   Otherwise, it is clear that G contains a clique of size $\ell = k$ if and only if G contains a subgraph isomorphic to H (these are just two ways of saying the same thing). Thus we have shown that SUBGRAPH ISOMORPHISM is NP-hard, as desired.

2. The problem is in NP because given T, it is easy to check whether each element of the universe U is in at least one set $S_i \in T$. We reduce from VERTEX COVER. Let $\langle G = (V, E), k\rangle$ be an instance of VERTEX COVER. Our reduction produces an instance of SET COVER as follows: the universe is the set of edges E and for each vertex $v \in V$, we have a set S_v consisting of the edges incident to v in G. Clearly this reduction runs in polynomial time.

   Now, we argue that "yes maps to yes": if there is a vertex cover $V' \subseteq V$ with $|V'| \leq k$, then there is a vertex cover $T = \{S_v : v \in V'\}$ of size at most k by definition (every edge $e = (u, v)$ has either u or v in V' and either S_v or S_u – both of which contain e – is therefore in T).

   We now argue that "no maps to no": if there is a set cover T with $|T| \leq k$, then taking V' to be the set of vertices v such that $S_v \in T$, we obtain a vertex cover of size at most k (every edge $e = (u, v)$ occurs in exactly two sets S_v and S_u, and so one of them must be in T, and therefore one of u or v is in V').

3. MINIMUM BISECTION is in NP because given a set $S \subseteq V$ (V are the vertices of the n node input graph) it is easy to verify that $|S| = n/2$ and count the number of edges crossing the cut, making sure there are at least k.

   All of the graphs we discuss below are multigraphs (parallel edges allowed).

   We reduce from MAX CUT. Given an instance $\langle G = (V, E), k\rangle$ of MAX CUT, we perform the following sequence of transformations. Let G_1 be the graph G with an additional n isolated nodes. Observe that if there is a cut $S \subseteq V$ in G with exactly ℓ edges crossing it, there is a *bisection* in G_1 with exactly ℓ edges crossing it, obtained by including $n - |S|$ isolated nodes in the old cut. Also, if there is a bisection in G_1 with exactly ℓ edges crossing it, then by discarding the isolated nodes, we obtain a cut in G with exactly ℓ edges crossing it.

Now let p be the maximum number of parallel edges occurring in G_1. Define G_2 to be the graph that has for each pair $u \neq v$ a number of parallel edges equal to p minus the number of parallel edges between u and v in G_1. Observe that a bisection in G_1 with exactly ℓ edges crossing it, has exactly $pn^2 - \ell$ edges crossing it in G_2. Similarly, a bisection in G_2 with exactly ℓ edges crossing it, has exactly $pn^2 - \ell$ edges crossing it in G_1.

Finally, let G_3 be the graph G_2 with a clique on all of its $2|V|$ nodes added to the existing edges. This is clearly connected. Our reduction produces $\langle G_3, pn^2 + n^2 - k\rangle$ and an instance of MIN BISECTION. Clearly this reduction runs in polynomial time.

Now for "yes maps to yes". If there is a cut in G with at least k edges crossing it, then there is a bisection in G_1 with at least k edges crossing it, and there is a bisection in G_2 with at *most* $pn^2 - k$ edges crossing it as we have argued above. In G_3, this cut has an additional n^2 edges coming from the clique we added on top of G_2 (there are n nodes on each side of the cut and all n^2 edges between them are present in that clique). So we have a bisection with $pn^2 + n^2 - k$ edges crossing it in G_3 as required.

Finally we argue that "no maps to no". Suppose there is a bisection in G_3 with at most $pn^2 + n^2 - k$ edges crossing it. We know that the clique we added to G_3 contributes exactly n^2 edges (because there are n nodes on each side of the cut and all n^2 edges between them are present in that clique). So in G_2 the same bisection has at most $pn^2 - k$ edges crossing it. As we argued above, this implies that G_1 has at *least* k edges crossing it, and then (also as argued above) G must have a cut with at least k edges crossing it, as required.

4. (a) First, note that PARTITION is in NP because given subset $T \subseteq \{1, 2, \ldots, n\}$ we can verify in polynomial time that $\sum_{i\in T} a_i = \sum_{i\notin T} a_i$.

To show that PARTITION is NP-hard, we reduce from SUBSET SUM. Given an instance $(a_1, a_2, \ldots, a_n, B)$ of SUBSET SUM, let $M = \sum_i a_i$. Our reduction produces the following instance of PARTITION:

$$a_1, a_2, \ldots, a_n, a_{n+1} = L - B, a_{n+2} = L - (M - B)$$

where $L = M + 1$. Clearly this reduction runs in polynomial time.

If we started with a YES instance of SUBSET SUM, then we claim that the reduction produces a YES instance of PARTITION. Suppose there exists a subset $T \subseteq \{1, 2, \ldots, n\}$ for which $\sum_{i\in T} a_i = B$. Then we have $\sum_{i\notin T} a_i = M - B$, and so we have $a_{n+1} + \sum_{i\in T} a_i = L = a_{n+2} + \sum_{a\notin T} a_i$, which implies that the instance is partitionable.

If the reduction produces a YES instance of PARTITION, then we claim that $(a_1, a_2, \ldots, a_n, B)$ was a YES instance of SUBSET SUM. Let T' specify the partition. Observe that we can't have both a_{n+1} and a_{n+2} in the same part of the partition, because then the sum of the integers in that part would be at least $a_{n+1} + a_{n+2} = 2L - M > M$, and the sum of the integers in the other part would be at most M. The sum of all elements is $2L$, so we must have:

$$\sum_{i\in T'} a_i = L = \sum_{a\notin T'} a_i.$$

If a_{n+1} is in the first part, then $T' - \{a_{n+1}\}$ is a subset of elements of the subset sum instance that sum to B, and if a_{n+1} is in the second part, then $\overline{T'} - \{a_{n+1}\}$ is a subset of elements of S that sum to B. We conclude that we started with a YES instance of SUBSET SUM as required.

(b) First, note that KNAPSACK is in NP because given subset of the n elements, we can verify in polynomial time that the sum of their values is at least V, and the sum of their costs is at most C.

To show that KNAPSACK is NP-hard, we reduce from SUBSET SUM. Given an instance $(S = \{a_1, a_2, \ldots, a_n\}, B$ of SUBSET SUM, our reduction produces the following instance of KNAPSACK: the cost c_i of item i is set to a_i, and the value v_i of item i is set to a_i as well. We set $V = C = B$. Clearly this reduction runs in polynomial time.

If we started with a YES instance of SUBSET SUM, then we claim that the reduction produces a YES instance of KNAPSACK. Suppose there exists a subset $T \subseteq S$ for which $\sum_{a \in T} a = B$. Then packing the element in T into our knapsack costs B and has value B, so the instance of KNAPSACK produced by the reduction is a YES instance.

If the reduction produces a YES instance of KNAPSACK, then we claim that (S, B) is a YES instance of SUBSET SUM. Let $T \subseteq S$ be the items packed into the knapsack, whose total value is at least V and whose total cost is at most C. In other words $\sum_{a \in T} a \geq V = B$ and $\sum_{a \in T} a \leq C$, which implies that $\sum_{a \in T} a = B$. We conclude that (S, B) is a YES instance of SUBSET SUM as required.