COMP 360: Algorithm Design Instructor: Yang Cai

Assignment 3

Due: 5pm, Nov 13, 2017

General Rules: In solving these questions you may consult books but you may not search on the web for solutions. You must write up your final solution yourself. You should drop your solutions in the assignment drop-off box located on the 3rd floor of the McConnell Engineering Building. No late assignments accepted.

1. (15 pts) Let $\Pi(G, k)$ denote the decision problem of whether the undirected graph $G = (V, E)$ has a subset of vertices $V' \subseteq V$ such that $|V'| = k$ and there is an edge connecting every pair of vertices in V'. Assume that the vertices are labeled as integers $V = \{1, ..., n\}$.

   (a) (5 pts) Prove that $\Pi(G, k)$ is in NP.

   (b) (10 pts) Prove that $\Pi(G, k)$ is NP-Complete by showing 3-SAT $\leq_P \Pi(G, k)$. Hint: Start with a given 3CNF formula with k clauses, $\phi = (a_1 \vee b_1 \vee c_1) \wedge ... \wedge (a_k \vee b_k \vee c_k)$ for the 3-SAT instance. Each of the three literals in each clause is of the form x_i or $\neg x_i$, where x_i are the available boolean variables of ϕ.

   Solution:

   (a) In order to show that $\Pi(G, k) \in$ NP, we must show that there exists a polynomial time verifier algorithm that takes an instance of the problem and a certificate as parameters, and verifies that the certificate is a *yes* instance of $\Pi(G, k)$. Thus, our instance is a graph G, and our certificate is a set of vertices V' of size k. Our verifier algorithm does the following: it checks that there are edges between every two vertices in V'. This algorithm runs in $\binom{k}{2} = O(n^2)$, and is thus clearly polynomial time. Thus, $\Pi(G, k) \in$ NP.

   (b) 3-SAT $\leq_p \Pi(G, k)$.

   Suppose we are given 3-CNF formula with k clauses, $\phi = (a_1 \vee b_1 \vee c_1) \wedge ... \wedge (a_k \vee b_k \vee c_k)$ for the 3-SAT instance. Each of the three literals in each clause is of the form x_i or $\neg x_i$, where x_i are the available boolean variables of ϕ.

   We create the graph G as follows: For each literal in each clause, create a node, which we will label with the name of the literal. The nodes corresponding to each clause will form an independent set of size 3, i.e. we will not have an edges between those nodes. Also we will not have an edge between the conflicting nodes, i.e. between the nodes labeled x_i and $\neg x_i$. We will connect two nodes with an edge in all the other cases. (Note this graph is the complement of the graph which was constructed to show that 3-SAT $\leq_p$ Independent Set.)

   We claim that the original 3-SAT instance is satisfiable if and only if the graph G we have constructed has a subset of size k in G such that there is an edge connecting every pair of vertices in that subset.

   First, if the 3-SAT is satisfiable, then for each clause there exists a literal which evaluates to 1. Take one literal from each clause and look at their corresponding nodes in G. Denote that set by V'. Note it has size exactly k. We claim that every pair of nodes from V' has an edge between them. Indeed, if there exists a pair of nodes which are not connected by an edge, it means they are conflicting nodes, but this cannot happen since the labels of those nodes both evaluate to 1 (it was the condition to be included in V').

Conversely, suppose our graph G has a set V' of size k in which every pair of nodes is connected by an edge. Note that this set contains exactly one node from each of the 3 nodes corresponding to each clause. Now, the truth assignment of the 3-SAT instance will be constructed in the following way: For each variable x_i, if neither x_i nor $\neg x_i$ appears as label of a node in V', then we set $x_i = 1$. Otherwise, exactly one of the x_i or $\neg x_i$ appears as a label of a node in V'; otherwise there would be an edge between two conflicting nodes. Thus, if x_i appears as a label of a node in V', we set $x_i = 1$, and otherwise we set $x_i = 0$. In this way all the labels of nodes in V' will evaluate to 1 and since V' contains one node from the nodes corresponding to each clause, this will be a satisfying assignment for the 3-SAT instance.

2. (15 pts) *Alice needs your help!*
Alice is solving an assignment on Graph Theory, and she observed that cycles in a graph is what makes the questions difficult. She is determined to make all the graphs acyclic with the minimum loss of vertices. She thinks solving this question is easy, but is it?
VERTEX DELETION: Given a directed graph $G = (V, E)$, and an integer k is there a set $V' \subseteq V$ with $|V'| \leq k$ such that V' contains at least one vertex from every directed cycle in G.

   (a) (5 pts) Show that VERTEX DELETION is in NP.

   (b) (10 pts) Prove that VERTEX DELETION is NP-Complete by showing VERTEX COVER $\leq_P$ VERTEX DELETION.

   Solution:

   (a) This problem is in NP. To check a proposed solution $V' \subseteq V$, we just verify that the graph G with $V - V'$ vertices is an acyclic graph. This verification can be done in polynomial time.

   (b) We will use a reduction from Vertex Cover to show that the Vertex Deletion problem is in NP-Complete. Given an instance of Vertex Cover that is an undirected graph G and a positive integer k, we will build a new graph G'' with the same vertices as G but replace every undirected edge (u,v) in G with two directed edges (u,v) and (v,u) in G''. Our instance of VERTEX DELETION would be the graph G'' with the same integer k. Clearly this reduction takes polynomial time since it simply copies G but replaces each edge with two edges.
   Now, we show that there is a vertex cover of size k in G if an only if there is a subset of k vertices in G'' whose removal breaks all cycles. First assume that there is a vertex cover of size k in G. Now remove these k vertices from G'' along with the edges incident upon them. Notice that for any directed edge(u,v) ϵ G'' at least one of u or v must have been removed, because one of u or v must have been in our vertex cover. Thus, after removing these k vertices and their incident edges from G'', no two vertices have an edge between them, and consequently there can be no cycles.
   Conversely, assume that there exists a set S of k vertices in G'' whose removal breaks all cycles. By construction every pair of vertices (u,v) that had an edge between them in G have a cycle (u,v) (v,u) in G''. Since the removal of set S broke all cycles in G'' for each edge $(u, v)\epsilon G$ the set S must contain either u or v (or both). Thus S is a vertex cover of G

3. (15 pts) Consider the TRIANGLE REMOVAL problem. We are given an undirected graph $G = (V, E)$, and an integer k, is it possible to find a set of vertices $U \subseteq V$ where $|U| \leq k$ such that deleting these vertices removes all the triangles (i.e. cycles of length 3) from the graph.

   (a) (5 pts) Prove that TRIANGLE REMOVAL is in NP.

   (b) (10 pts) Prove that TRIANGLE REMOVAL is NP-complete by showing VERTEX COVER $\leq_P$ TRIANGLE REMOVAL.

   Solution:
   (a) Given a graph $G = (V, E)$, integer k and certificate $U \subseteq V$, we want to output True if and only if $|U| \leq k$ and $(V - U, E)$ contains no triangles.
   There are many ways to check for triangles in polynomial time with respect to $|V| + |E|$.

Brute force: For every set of 3 edges, $\{e_1, e_2, e_3\}$, check if this is a triangle.
Adjacency matrix: Let A be the adjacency matrix of some graph. Inductively, we see that $A^n[i,j]$ gives the number of distinct walks from vertices i to j of length n. Thus the graph contains no triangles if and only if $trace(A^3) = 0$. By optimizing matrix multiplication, this gives a better running time.

(b) Given $G = (V, E)$, let $G' = (V', E')$ be G with the following additions: for every edge $(u, v) \in G$, add new vertex $w_{u,v}$ and edges (u, w) and (v, w) to G'.
Notation: For any graph $G = (V, E)$ and any $U \subseteq V$, we will use $(V - U, E)$ to denote the graph $(V - U, E'')$ where $E'' = \{e = (u, v) : e \in E, u \in V - U \text{ and } v \in V - U\}$.

Claim: Given some graph $G = (V, E)$ and corresponding $G' = (V', E')$ as described above, then there exists vertex cover U for G, $|U| \leq k$, if and only if there exists triangle removal set U' for G', $|U'| \leq k$.

Proof: ($\Leftarrow$) Suppose we have U' such that $(V' - U', E')$ does not contain any triangles and $|U'| \leq k$. Define U'': $\forall v' \in U'$, if $v' \in V$, then $v' \in U''$. If $v' \in (U' - V)$, then by definition of G', v' has two neighbors, $w' \in V$ and $u' \in V$. If $u' \notin U'$ and $w' \notin U'$ we define $w' \in U''$. Note that v' is only a vertex of one triangle, (v', u', w'), by construction. Hence U'' is a triangle removal set for G'. Also $|U''| \leq |U'| \leq k$ and $U'' \subseteq V$.
For all $e = (u, v) \in E$, there is triangle $(u, v, w_{u,v})$ in G' with $w_{u,v} \notin V$ and U'' is a triangle removal set for G' with $w_{u,v} \notin U''$, hence $u \in U''$ or $v \in U''$. That is, U'' is a vertex cover for G.

($\Rightarrow$) Let U a vertex cover for G and $|U| \leq k$. Let $G'' = (V' - U, E')$. That is, for every edge $e = (u', v') \in G''$, at least one of u' and v' is in $V' - U$. Thus, any triangle in G'' would contain some edge $e' = (x, y)$, with $x \in V' - U$ and $y \in V' - U$. But no such edge exists in G' and hence also not in G''. That is, U is a triangle removal set for G'.

4. (15 pts) Given a set of positive integers $\mathcal{S}$ containing n elements $\mathcal{S} = \{e_1, e_2, ..., e_n\}$ and a number B. We call S_1, S_2, S_3 a partition of $\mathcal{S}$ if S_1, S_2, S_3 are disjoint and $S_1 \cup S_2 \cup S_3 = \mathcal{S}$. Let $T_j = \sum_{e_i \in S_j} e_i$ for all $j \leq 3$. Decide whether there exists a partition S_1, S_2, S_3 such that

$$\frac{T_1}{T_2} + \frac{T_2}{T_3} + \frac{T_3}{T_1} \leq B$$

(a) (5 pts) Prove this problem is in NP.
(b) (5 pts) 3-Partition: Is there a partition S_1, S_2 and S_3 of S, such that $T_1 = T_2 = T_3$? Prove 3-Partition is NP-Complete. You may assume that the Number Partitioning problem (Question 26 on p. 518 in the textbook) is NP-Complete.
(c) (5 pts) Prove this problem is NP-Complete.

Solution:

(a) Given a 3-partition (S_1, S_2, S_3), it's enough to verify if the inequality holds. The verification can be done in polynomial time.
(b) We will come up with a reduction from the Number Partitioning problem to 3-Partition problem. For any set of natural numbers $S = \{a_1, ..., a_n\}$ for the Number Partitioning problem, let $m = \frac{1}{2}\sum_{i=1}^{n} a_i$. Consider the 3-Partition problem with set of elements $S' = \{a_1, ..., a_n, m\}$. For simplicity, we denote $sum(T)$ the sum of all elements in a set T.
Suppose there is a number partition (S_1, S_2) for S, then clearly $S_1, S_2, \{m\}$ is a 3-partition for S' since $sum(S_1) = sum(S_2) = m$. On the other side, suppose (S_1, S_2, S_3) is a 3-partition for S'. Without loss of generality assume $m \in S_1$. Since $sum(S_1) = \frac{1}{3}sum(S') = m$ and all numbers are natural numbers, S_1 only contains m and some zero numbers. Thus $(S_2 \cup$

$S_1 \backslash \{m\}, S_3)$ is a number partition for S, since $sum(S_2) + sum(S_1 \backslash \{m\}) = sum(S_2) + 0 = sum(S_3)$.

Now since the Number Partitioning problem is NP-Complete, the 3-Partition problem is also NP-Complete.

(c) To prove it's NP-Complete, we will make a reduction from the 3-Partition problem to this problem. For any set of natural numbers $S = \{a_1, ..., a_n\}$, consider the given problem with the same set of elements S and $B = 3$.

Suppose there is a 3-partition (S_1, S_2, S_3) for S, then the same partition will satisfy the inequality since $\frac{T_1}{T_2} + \frac{T_2}{T_3} + \frac{T_3}{T_1} = 1 + 1 + 1 = 3$.

On the other side, if there is a partition (S_1, S_2, S_3) for S such that $\frac{T_1}{T_2} + \frac{T_2}{T_3} + \frac{T_3}{T_1} \leq 3$. By AM-GM inequality,

$$\frac{T_1}{T_2} + \frac{T_2}{T_3} + \frac{T_3}{T_1} \geq 3 \cdot (\frac{T_1}{T_2} \cdot \frac{T_2}{T_3} \cdot \frac{T_3}{T_1})^{\frac{1}{3}} = 3$$

.

and the equality sign holds if and only if $T_1 = T_2 = T_3$. With the two inequalities, we have $\frac{T_1}{T_2} + \frac{T_2}{T_3} + \frac{T_3}{T_1} = 3$ and thus $T_1 = T_2 = T_3$. This implies that (S_1, S_2, S_3) is also a 3-partition for S (with equal sum).