CS/ECE 374 ✧ Spring 2017

Homework 10

Due Friday, April 28, 2017 at 10am

1. Given an undirected graph $G = (V, E)$ a *matching* in G is a set of edges $M \subseteq E$ such that no two edges in M share a node. A matching M is *perfect* if $2|M| = |V|$, in other words if every node is incident to some edge of M. PerfectMatching is the following decision problem: does a given graph G have a perfect matching? Describe a polynomial-time reduction from PerfectMatching to SAT. Does this problem that PerfectMatching is a difficult problem?

   Solution: To reduce PerfectMatching to SAT, we need to enforce the requirement that every vertex is adjacent to exactly one edge. We can break this down into the following two conditions:

   - Every vertex is adjacent to at least one edge.
   - Every vertex is adjacent to at most one edge.

   For each edge $(u, v) \in E$, we introduce a variable $y_{u,v}$ that indicates whether (u, v) is included in the perfect matching. Then, to satisfy these conditions, we define the clauses of a CNF formula, ϕ, as follows:

   - For every vertex $v \in V$, include the clause $\left(\bigvee_{u \in N(v)} y_{u,v}\right)$ where $N(v)$ is the set of vertices adjacent to v.
   - For every vertex $v \in V$, define a set $S_v = \{\{(u_1, v), (u_2, v)\} \mid u_1, u_2 \in N(v), u_1 \neq u_2\}$. This is the set of all 2 combinations of edges adjacent to v. Then, for all $\{(u_1, v), (u_2, v)\} \in S_v$, include the clause $\left(\neg y_{u_1,v} \vee \neg y_{u_2,v}\right)$.

   To force every vertex to be adjacent to at least one edge, we introduce a clause with at most $O(|V|)$ literals for each vertex. To force every vertex to be adjacent to at most one edge, we introduce at most $O(|V|^2)$ clauses with 2 literals for each vertex. As finding adjacent vertices can be done in constant time, we can see that this reduction takes polynomial time.

   To prove this reduction is correct, we need to show that G has a perfect matching if and only if ϕ is satisfiable.

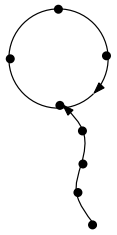
   Suppose G has a perfect matching. Fix an arbitrary perfect matching. As the matching is perfect, every vertex is adjacent to at least one edge. Thus, the first set of clauses will be satisfied. The second set of clauses is the disjunction of the negation of two literals representing adjacent edges. As a matching is a set of edges such that no two edges share a vertex, each of these clauses must be satisfied.

   On the other hand, suppose ϕ is satisfiable. Fix an arbitrary satisfying assignment. By definition, each clause in ϕ contains at least one TRUE literal. Therefore, an assignment that satisfies the first set of clauses indicates that every vertex is adjacent to at least one edge ("perfect"). An assignment that satisfies the second set of clauses indicates that no two adjacent edges are TRUE at the same time. Thus, the assignment must represent a valid matching.

   This reduction is not sufficient to show that PerfectMatching is a difficult problem. ■

   Rubric: 10 points - Standard Reduction rubric.

2. A balloon is a directed graph on an even number of nodes, say $2n$, in which n of the nodes form a directed cycle and the remaining n vertices are connected in a "tail" that consists of a directed path joined to one of the nodes in the cycle. See figure below for a balloon with 8 nodes.



Given a directed graph G and an integer k, the BALLOON problem asks whether or not there exists a subgraph which is a baloon that contains $2k$ nodes. Prove that BALLOON is NP-Complete.

Solution: An instance of DIRECTEDHAMILTONIANCYCLE consists of a directed graph G and the problem is to determine whether there is a spanning cycle in G. We know that DIRECTEDHAMILTONIANCYCLE is NP-hard. To prove that BALLOON is NP-hard, we reduce DIRECTEDHAMILTONIANCYCLE problem to BALLOON.

Let $< G >$ be an instance of DIRECTEDHAMILTONIANCYCLE, where G is a directed graph on n vertices. Construct graph G' by adding a directed path of n vertices to G and connecting its head to an arbitrary vertex in G. We denote the path subgraph of G' consisting of these $n+1$ vertices by P. We claim that G has a Hamiltonian cycle if and only if G' has a balloon on $2n$ vertices.

Suppose G has a Hamiltonian cycle C. Then the subgraph of G' obtained by the union of C and P is a balloon on $2n$ vertices.

Conversely, suppose that G' has a balloon subgraph B on $2n$ vertices. Since G' has exactly $2n$ vertices, B has to span the entire graph, i.e. every vertex of G' belongs to B. By construction, the n vertices of $V(G') \setminus V(G)$ are not on a cycle. Therefore, in B, these vertices can only be on the tail of the balloon. Thus, the n vertices of G are on the body of the balloon. In particular, there is a cycle spanning G. Hence, G has a Hamiltonian cycle.

To complete the proof, we show that BALLOON is also NP. To see this point, observe that given a graph G, an integer k, and a graph H one can verify in polynomial time whether H is a subgraph of G and if H is balloon on $2k$ vertices. Let the certificate be a sequence of $2k$ distinct vertices $v_1, \ldots, v_{2k}$ such that $v_1, \ldots, v_{k+1}$ form a directed path in G an $v_{k+1}, \ldots, v_{2k}$ form a directed cycle in G. There is a polynomial time Algorithm for BALLOON that can verify the validity of this certificate. Therefore, BALLOON $\in$ NP. ■

Rubric: 10 points:

- 2 points for proving Balloon $\in$ NP.
- 8 points for proving Balloon is NP-hard:
  - + 3 points for the reduction itself
    - * For an NP-hardness proof, the reduction must be from a known NP-hard problem. You can use any of the NP-hard problems listed in the lecture notes (except the one you are trying to prove NP-hard, of course).
  - + 2 points for the "if" proof of correctness
  - + 2 points for the "only if" proof of correctness
  - + 1 point for writing "polynomial time"
  - – An incorrect polynomial-time reduction that still satisfies half of the correctness proof is worth at most 3/8.
  - – A reduction in the wrong direction is worth 0/8.

3. Consider the following problem. You are managing a communication network, modeled by a directed graph $G = (V, E)$. There are c *users* who are interested in making use of this network. User i (for each $i = 1, 2, \ldots, c$) issues a *request* to reserve a specific path P_i in G on which to transmit data.

   You are interested in accepting as many of these path requests as possible, subject to the following restriction: if you accept both P_i and P_j, then P_i and P_j can not share any modes.

   Thus the *Path Selection Problem* asks: Given a directed graph $G = (V, E)$, a set of requests $P_1, \ldots, P_c$-each of which must be a path in G- and a number k, is it possible to select at least k of the paths so that no two of the selected paths share any nodes?

   Prove that the Path Selection is NP-Complete.

   Solution:

   - *Path selection* is in NP: Given a set of k paths from among P_1, ...,P_c, we can check if no two paths share any nodes in polynomial time.
   - *Path selection* is NP hard: We show this by reducing a known NP-hard problem, *Independent Set* to *Path Selection* as follows:
     - Let (G, k) with $|V| = n$ and $|E| = m$ be an instance of the *Independent Set* problem. For the sake of simplicity, lets assume that the edges are labelled $1..m$.
     - We construct a new directed graph $G' = (V', E')$ where V' = $\{e | e \in E\}$ and $E' = \{(e_i, e_j) | e_i, e_j \in V'\}$. It is to be noted that G' is a complete directed graph.
     - We now construct n paths in G' corresponding to the n vertices in G as follows: $\forall i \in V$, let $e_{i_1} \ldots e_{i_{deg(i)}}$ be the edges incident on i. Correspondingly, we construct P_i to be $e_{i_1} \rightarrow e_{i_2} \ldots \rightarrow e_{i_{deg(i)}}$. Again, it is to be noted that these are all valid paths since G' is a complete graph.
     - It can be seen that G' and the aforementioned paths can be constructed in polynomial time from G .
     - Now we are left to prove that this reduction is valid for which we need to prove that *G has an independent set of size k if and only if there are k paths in $P_1, P_2, \ldots, P_n$ that are node-disjoint in G'*.
     - This follows almost immediately since if $e = (i, j)$ is an edge in E, then both P_i and P_j contain the vertex e in G' and thus cannot both be chosen for the node-disjoint paths selection. Conversely, if P_i and P_j share a vertex e in G', then i and j cannot both be chosen to construct an independent set in G.

   ■

   Rubric: 10 points - Standard Reduction rubric.

Rubric (for all polynomial-time reductions): 10 points =

- \+ 3 points for the reduction itself
  - – For an NP-hardness proof, the reduction must be from a known NP-hard problem. You can use any of the NP-hard problems listed in the lecture notes (except the one you are trying to prove NP-hard, of course).
- \+ 3 points for the "if" proof of correctness
- \+ 3 points for the "only if" proof of correctness
- \+ 1 point for writing "polynomial time"

- An incorrect polynomial-time reduction that still satisfies half of the correctness proof is worth at most 4/10.
- A reduction in the wrong direction is worth 0/10.