CS 170 Final Review: Complexity

1 To NP or not to NP, that is the question

For the following questions, circle the (unique) condition that would make the statement true.

(a) If B is **NP**-complete, then for any problem $A \in \mathbf{NP}$, there exists a polynomial-time reduction from A to B.

Always true   True iff $\mathbf{P} = \mathbf{NP}$ True iff $\mathbf{P} \neq \mathbf{NP}$ Always false

Solution: Always true: this is the definition of **NP**-hard, and all **NP**-complete problems are **NP**-hard

(b) If B is in **NP**, then for any problem $A \in \mathbf{P}$, there exists a polynomial-time reduction from A to B.

Always true   True iff $\mathbf{P} = \mathbf{NP}$ True iff $\mathbf{P} \neq \mathbf{NP}$ Always false

Solution: Always true: since we have polynomial time for our reduction, we have enough time to simply solve any instance of A during the reduction.

(c) Horn SAT is **NP**-complete.

Always true   True iff $\mathbf{P} = \mathbf{NP}$ True iff $\mathbf{P} \neq \mathbf{NP}$ Always false

Solution: True iff $\mathbf{P} = \mathbf{NP}$: Horn SAT is in **P**.

2 $\frac{1}{3}$ Independent Set

In the $\frac{1}{3}$-INDEPENDENT SET problem you are given a graph (V, E) and you are asked to find an independent set of the graph of size exactly $\frac{|V|}{3}$. In other words, the target size g is not part of the input, but it is always $\frac{|V|}{3}$. Prove this special case of INDEPENDENT SET is **NP**-complete.

Solution: A solution to $\frac{1}{3}$ Independent Set can be verified in polynomial time by checking that there are no edges between the nodes of the independent set and checking its size.

Use a reduction from Independent Set to $\frac{1}{3}$-Independent Set. Consider an instance of Independent Set with a graph G and target size g. If $g \geq |V|/3$, then we can add $3g - |V|$ vertices to G such that the new vertices all have edges to each other and the original vertices. We can then solve our modified G with $\frac{1}{3}$-Independent Set.

If $g < |V|/3$, then we add $(|V| - 3g)/2$ isolated vertices to G. We then solve our modified G with $\frac{1}{3}$-Independent Set and receive an independent set as an answer. From that independent set, we return only the vertices that were in our original graph.

3 Bounded Clique and Fake Reductions

Consider the CLIQUE problem restricted to graphs in which every vertex has degree at most 3. Call this problem CLIQUE-3. Recall that the CLIQUE problem is given a graph and a goal g, find a set of g vertices such that all possible edges between them are present.

(a) Prove that CLIQUE-3 is in **NP**.

(b) What is wrong with the following proof of **NP**-completeness for CLIQUE-3?

We know that the CLIQUE problem in general graphs is **NP**-complete, so it is enough to present a reduction from CLIQUE-3 to CLIQUE. Given a graph G with vertices of degree ≤ 3, and a parameter g, the reduction leaves the graph and the parameter unchanged: clearly the output of the reduction is a possible input for the CLIQUE problem. Furthermore, the answer to both problems is identical. This proves the correctness of the reduction and, therefore, the **NP**-completeness of CLIQUE-3.

(c) It is true that the VERTEX COVER problem remains **NP**-complete even when restricted to graphs in which every vertex has degree at most 3. Call this problem VC-3. What is wrong with the following proof of **NP**-completeness for CLIQUE-3?

We present a reduction from VC-3 to CLIQUE-3. Given a graph $G = (V, E)$ with node degrees bounded by 3, and a parameter b, we create an instance of CLIQUE-3 by leaving the graph unchanged and switching the parameter to $|V| - b$. Now, a subset $C \subseteq V$ is a vertex cover in G if and only if the complementary set $V - C$ is a clique in G. Therefore G has a vertex cover of size $\leq b$ if and only if it has a clique of size $\geq |V| - b$. This proves the correctness of the reduction and, consequently, the **NP**-completeness of CLIQUE-3.

(d) Describe an $O(|V|^4)$ algorithm for CLIQUE-3.

Solution:

(a) Given a clique in the graph, it is easy to verify in polynomial time that there is an edge between every pair of vertices. Hence a solution to CLIQUE-3 can be checked in polynomial time.

(b) The reduction is in the wrong direction. We must reduce CLIQUE to CLIQUE-3 if we intend to show that CLIQUE-3 is at least as hard as CLIQUE.

(c) The statement "a subset $C \subseteq V$ is a vertex cover in G if and only if the complementary set $V - C$ is a clique in G" used in the reduction is false. C is a vertex cover if and only if $V - C$ is an *independent set* in G.

(d) The largest clique in the graph can be of size at most 4, since every vertex in a clique of size k must have degree at least $k - 1$. Thus, there is no solution for $k > 4$, and for $k \leq 4$ we can check every k-tuple of vertices, which takes $O(|V|^k) = O(|V|^4)$ time.

4 Connected Spanning Subgraph [Fa13 Final]

Show that given a graph G and a positive integer k, the problem of finding a connected spanning subgraph where every node has degree at most k is NP-complete by reducing a known NP-Complete problem to Connected Spanning Subgraph. A spanning subgraph of $G = (V, E)$ is another graph $G' = (V, E')$ where $E' \subseteq E$ (notice that the set of vertices is the same)
(Note: the other requirement for np-completeness is that the problem can be verified in polynomial time, which is trivial for this problem).
Solution: The main takeaway is to find similar problems to reduce from. (Ex: reduce graph to graph problems)
This is a generalization of Rudrata Path. If we set $k = 2$, we will get either a Rudrata Path or a Rudrata Cycle, and it is easy to get a Rudrata Path from a Rudrata Cycle (delete one of the edges). Conversely, if there exists a Rudrata Path, that will be a connected spanning subgraph where every node has degree at most 2.

5 Reductions Potpourri

List of hints

- The RUDRATA PATH problem is known to be NP-Complete. It finds a simple path in a graph G which goes through all vertices.
- The SET COVER problem is known to be NP-Complete. Given a universe of elements U, a list of sets $S = \{S_1, S_2, \ldots, S_k\}$, and an integer m, it finds a subset of S of size less than or equal to m, such that the union of these subsets is U.
- The INDEPENDENT SET problem takes in a graph G, and a lower bound k. It aims to find a set of vertices of size at least k, such that no two vertices in the set are adjacent.

(a) Show that the LONGEST PATH problem is NP-Complete. The LONGEST PATH problem takes in an unweighted Graph, G, and an integer g. It finds a simple path of length g in G. Remember that a simple path does not reuse vertices.
To prove that a problem is NP-Complete, show:
1. that given a potential solution you can verify its correctness in polynomial time
2. that a known NP-Complete problem can be reduced to the problem (reduce one of the 3 problems above to Longest Path).

Solution: First, we prove that LONGEST PATH is in NP. Given a path, we just count the number of edges in the path. This takes at most $O(V + E)$ time. Therefore LONGEST PATH is in NP.

Next, we reduce RUDRATA PATH to LONGEST PATH. Given an instance to RUDRATA PATH G, we create an instance of LONGEST PATH, where G remains the same, and $g = |V| - 1$.

Now, we use our black box to solve LONGEST PATH. If a path of length g exists, then this path must be using all $|V|$ vertices, and is a RUDRATA PATH. IF a path does not exist, then there is no path using all $|V|$ vertices, and there is no corresponding RUDRATA PATH.

This is a polynomial time reduction. We are done, LONGEST PATH is NP-Complete.

(b) Suppose you work for the United Nations. Let L be the set of all languages spoken by people across the world. The United Nations also has a set of translators, all of whom speak English, and some other languages from L. Due to budget cuts, you can only afford to keep k translators on your payroll. Can you do this, while still ensuring that there is someone who speaks every language in L? If you could solve this problem efficiently (in polynomial time), then what would that imply?

Solution:

We can reduce Set Cover to this problem. Set $U = L$. For every subset in S, say that you have a person who speaks the languages inside their corresponding set. Let $k = m$.

Note that reducing Set Cover to UN Problem implies that the UN Problem is at least as difficult as the Set Cover Problem. If there is a polynomial time solver for the UN Problem, then there is a polynomial time solver for the Set Cover problem!

If we can solve this problem, we can solve Set Cover in polynomial time. This would mean that $P = NP$.