ECS120 Introduction to the Theory of Computation
Fall Quarter 2007

Homework 9 Help

Due Friday December 7, 2007

Since this homework is due on the last day of class, you'll not be getting these homeworks back from me with comments. You'll be able to see your score on my.ucdavis.edu once it is graded. If you want to get your homework back once it is graded, please see Professor Filkov.

Problem 9.1

Let k be a positive integer. We define

$$k\text{-}CLIQUE = \{ \langle G \rangle \mid G \text{ is an undirected graph with a } k\text{-clique} \}.$$

Show that for any constant k, $k\text{-}CLIQUE \in \text{P}$.

Theorem 7.24 (p268)
The language $CLIQUE = \{ \langle G, k \rangle \mid G \text{ is an undirected graph with a } k\text{-clique} \}$ is in NP.

The proof for this theorem gives a polynomial-time verifier for $CLIQUE$. You can use this polynomial-time verifier in your solution.

Extra Hint: How many k-node subgraphs are in G?

Problem 9.2

Suppose that there is a polynomial-time algorithm which, on input of a Boolean formula ϕ, decides if ϕ is satisfiable. Describe a polynomial-time algorithm which finds a satisfying assignment for ϕ.

Name the polynomial-time algorithm so that you can use it in your solution. For example, let $\mathcal{A}_{\text{sat}}(\phi)$ be the polynomial-time algorithm that decides if ϕ is satisfiable.

Make sure to include an argument that the algorithm you provide is indeed polynomial-time.

Extra Hint: Try to find a satisfying assignment one variable at a time.

Problem 9.3

Show that if P = NP, then every language $A \in P$, except $A = \emptyset$ and $A = \Sigma^*$, is NP-complete.
Sipser Problem 7.17

Why do we need the part "$A \neq \emptyset$ and $A \neq \Sigma^*$" to solve this problem? Notice that if this is true, then we are guaranteed that there exists two different elements x and y such that $x \in A$ and $y \notin A$.

To show A is NP-complete, you need to show it satisfies two conditions:

1. $A \in$ NP, and
2. Every L in NP is polynomial-time reducible to A.

The first part should be simple, since you are assuming that P = NP. The second part is a reduction, specifically $L \leq_{\mathrm{p}} A$ for all $L \in NP$. The assumption that P = NP will help you with the second part as well.

Problem 9.4

A ***coloring*** of a graph is an assignment of colors to its nodes so that no two adjacent nodes are assigned the same color. Let

$$3COLOR = \{\langle G \rangle \mid \text{the nodes of } G \text{ can be colored with three colors such that no two nodes joined by an edge have the same color}\}.$$

Show that $3COLOR$ is NP-complete. (Hint: Use the following three subgraphs.)
Sipser Problem 7.27

Don't forget that there are two parts to showing a language is NP-complete. You have to show:

1. $3COLOR \in$ NP
2. Every $L \in NP$ is polynomial-time reducible to $3COLOR$.

Try showing the reduction $3SAT \leq_{\mathrm{p}} 3COLOR$!

Corollary 7.42 (page 282) states that $3SAT$ is NP-complete. This means all languages are polynomial-time reducible to $3SAT$. Therefore, if we can show that $3SAT \leq_{\mathrm{p}} 3COLOR$, then:

$$\forall L \in \text{NP}, \quad L \leq_{\mathrm{p}} 3SAT \leq_{\mathrm{p}} 3COLOR$$

Therefore, if you show $3SAT \leq_{\mathrm{p}} 3COLOR$, you satisfy the second requirement of NP-completeness.

Why $3SAT$? It might be easier to find a mapping, since it has three elements just like $3COLOR$.

Tidbits: This is related to the **four color theorem**. From Wikipedia:

> "The four color theorem (also known as the four color map theorem) states that given any plane separated into regions, such as a political map of the states of a country, the regions may be colored using no more than four colors in such a way that no two adjacent regions receive the same color....
>
> ...The conjecture was first proposed in 1852 when Francis Guthrie, while trying to color the map of counties of England, noticed that only four different colors were needed."

Just in case you were wondering where all of these weird "problems" were coming from. Many of them are based on real-world problems. (There are other applications of the four-color theorem, this is just where it actually came from.)

Problem 9.5

Define the *SET-PARTITION* problem as follows:

$$\begin{aligned} \textit{SET-PARTITION} \;=\; & \{ \langle S \rangle \mid S = \{ x_1, \ldots, x_k \} \text{ each } x_i \text{ is a positive integer,} \\ & \text{and for some } A \subseteq S, \sum_{x_i \in A} x_i = \sum_{x_i \neq A} x_i \} \end{aligned}$$

Show that *SET-PARTITION* is NP-complete.

Don't forget that there are two parts to showing a language is NP-complete. You have to show:

1. *SET-PARTITION* $\in$ NP
2. Every $L \in NP$ is polynomial-time reducible to *SET-PARTITION* .

Try showing the reduction *SUBSET-SUM* $\leq_{\text{p}}$ *SET-PARTITION*!

Theorem 7.56 (page 292) states that *SUBSET-SUM* is NP-complete. This means all languages are polynomial-time reducible to *SUBSET-SUM* . Therefore, if we can show that *SUBSET-SUM* $\leq_{\text{p}}$ *SET-PARTITION*, then:

$$\forall L \in \text{NP}, \quad L \leq_{\text{p}} \textit{SUBSET-SUM} \leq_{\text{p}} \textit{SET-PARTITION}$$

Therefore, if you show *SUBSET-SUM* $\leq_{\text{p}}$ *SET-PARTITION*, you satisfy the second requirement of NP-completeness.