Section 9 Handout

CS 121

November 29, 2010

Today's Topics

- Polynomial-time reductions
- NP-hard and NP-complete

1 Polynomial-time reductions

Recall: We defined A to be *mapping reducible* to B, or $A \leq_m B$, iff there exists a computable function f such that for any string w, $f(w) \in B$ iff $w \in A$.
This type of reduction was useful when we were concerned with whether things were computable or decidable at all. However...

Exercise 1.1. *Show that if $A \leq_m B$ and $B \in$ P, it is NOT necessarily true that $A \in$ P. (Assume that there is at least one decidable language L that is not in* P*)*

The problem is that f is too "powerful". With an appropriate computable f, we can reduce any decidable language to pretty much any other decidable language. So, to talk about the relative difficulty of problems in P and NP, we need a new type of reduction:

We say that A is *polynomial-time reducible* to B, or $A \leq_P B$, if the reducing function is computable *in polynomial time*. That is, if there exists an f computable in polynomial time such that for any string w, $f(w) \in B$ iff $w \in A$.

Properties:

- If $A \leq_P B$ and $B \in$ P, then $A \in$ P (why?)
- If $A \leq_P B$ and $B \in$ NP, then $A \in$ NP (why?)
- If $A \leq_P B$ and $B \leq_P C$, then $A \leq_P C$ (why?)

2 NP-hard and NP-complete

A language L is said to be NP-hard iff for all $L' \in$ NP, $L' \leq_P L$.
A language L is said to be NP-complete iff $L \in$ NP and L is NP-hard.

Properties:

- If $A \leq_P B$ and A is NP-hard, then B is NP-hard (why?)
- If A is NP-complete and $A \in$ P, then P = NP (why?)

To show that L is NP-complete,

- Show that $L \in$ NP.
- Show that L is NP-hard by choosing a known NP-complete language L', and reducing: $L' \leq_P L$.

Some known NP-complete languages:
SAT, 3-SAT, Vertex Cover, Clique.

3 Exercises

Exercise 3.1. *Say that two boolean formulas are* ***equivalent*** *if on any possible truth assignment, they both evaluate to the same value (true or false).*

Let $L = \{\langle \phi_1, \phi_2 \rangle : \phi_1$ *and* ϕ_2 *are not equivalent*$\}$*. Show that* L *is* NP*-complete.*

Exercise 3.2. *Why do we believe that* a^*b^* *is not* NP*-complete? Show that if* a^*b^* *is not* NP*-complete, then* P $\neq$ NP*.*

Exercise 3.3. *Define* IntegerLinearProgram *to be the problem:*

Given variables $x_1, ..., x_n$ *and one or more linear inequalities with integer coefficients, such as* $2x_1 - x_3 > 5$ *and* $x_1 + x_2 + 4x_5 \leq -2$*, is there an integer solution that satisfies all inequalities?*

Show that IntegerLinearProgram *is* NP*-complete.*
*(Hint: reduce from 3-*SAT*)*

Exercise 3.4. *A **clique** of size k in a graph is a set of k vertices such that there is an edge between every pair of vertices in the clique.*
Recall that CLIQUE $= \{\langle G, k\rangle : G$ *has a clique of size* $k\}$ *is* NP*-complete.*

Now define HALF-CLIQUE :

$$\{\langle G\rangle : G \text{ has a clique of size at least } |V|/2 \text{ where } |V| \text{ is the number of vertices in } G\}$$

Show that HALF-CLIQUE *is* NP*-complete.*

Exercise 3.5. *Say that a formula is **minimal** if no shorter boolean formula is equivalent to it. Let* MIN-FORMULA *be the language of all minimal boolean formulas.*

Show that if P = NP*, then* MIN-FORMULA $\in$ P*.*
(Note: it is currently believed that MIN-FORMULA $\notin$ NP*.)*