15-453 Formal Languages, Automata, and Computation

Review Problems 3

Mihai Budiu

Wednesday, April 26

Problem 1

Note that in the Cook-Levin theorem the formula we derive for the behavior of a turing machine is in CNF-SAT form. So the Cook-Levin theorem actually proves that CNF-SAT is NP-complete. Prove that SAT is NP-complete by giving a polynomial-time reduction between CNF-SAT and SAT.

Solution sketch: we can use the following reduction: on input ϕ, check if ϕ is in CNF-SAT form. If not, output "false" (a non-satisfiable formula). Else output ϕ. This works because every formula in CNF-SAT form is also in SAT form.

Problem 2

Give a polynomial-time reductions from CNF-SAT to k-CNF-SAT, for $k > 2$.

Solution sketch: We can use the following property: the formulas $(C \vee D)$ and $(x \vee C) \wedge (\neg x \vee D)$ have the same satisfiability. We will illustrate for $k = 3$: we can rewrite

$$\begin{aligned} a_1 \vee a_2 \vee \ldots a_k &= (x_1 \vee a_1 \vee a_2) \wedge (\neg x_1 \vee a_3 \vee \ldots a_n) \\ &= (x_1 \vee a_1 \vee a_2) \wedge (\neg x_1 \vee a_3 \vee x_2) \wedge (\neg x_2 \vee a_4 \vee \ldots a_n) \\ &= (x_1 \vee a_1 \vee a_2) \wedge (\neg x_1 \vee a_3 \vee x_2) \wedge (\neg x_2 \vee a_4 \vee x_3) \ldots \end{aligned}$$

From each clause with k literals we get $k-2$ clauses with 3 literals. This is a polynomial blowup, and the rewriting can be done in linear time. Moreoever, this is a reduction.

Problem 3

Using the result from problem 4, homework 10, show that if P$\neq$NP, there is no polynomial-time reduction between CNF-SAT and 2-CNF-SAT.

Solution sketch: From hw 10 problem 4 we know that 2-CNF-SAT $\in$ P. We also know that CNF-SAT is NP-complete. We use the theorem: "if A is NP-complete, $A \leq_p B$ and $B \in$P, then P=NP" to get a contradiction.

Problem 4

For each of these two problems:

1. Given a graph G, what is the size of a minimum vertex-cover?
2. Given a graph G, find a minimum vertex-cover.

prove the following facts:

(a) If the problem can be solved in polynomial time, then P=NP.

(b) If P=NP, then the problem can be solved in polynomial time.

Solution sketch:

1. Given a graph G, what is the size of a minimum vertex-cover?
   (a) **To prove:** If the problem can be done in poly-time, then P=NP.
   If we can find the size of a minimum vertex-cover, then we can solve k-VERTEX-COVER in polynomial time in the following way:

   > On input $\langle G, k\rangle$, call "size of minimum vertex-cover" on G. The answer we get is r. Compare k to r: if $k \geq r$ then output "yes", else output "no".

   Because k-VERTEX-COVER is NP-complete, we must have P=NP.

   (b) **To prove:** If P=NP, then the problem can be solved in polynomial time.
   If P=NP, we can solve k-VERTEX-COVER in polynomial time. To find the size of the minimum vertex-cover we call successively k-VERTEX-COVER on $\langle G, 1\rangle, \langle G, 2\rangle, \ldots \langle G, n\rangle$ (n is the number of vertices). Whenever k-VERTEX-COVER answers "yes", stop.

2. Given a graph G, find the minimum vertex-cover (a set of nodes).
   (a) **To prove:** If the problem can be solved in polynomial time, then P=NP.
   We can solve k-VERTEX-COVER (with input $\langle G, k\rangle$ in polynomial time in the following way: call "minimum vertex-cover(G)", let the size of the resulting set be r. If $k \geq r$ answer "yes", else answer "no".

   (b) **To prove:** If P=NP, then the problem can be solved in polynomial time.
   If P=NP we can solve k-VERTEX-COVER in polynomial time. We can find a minimum vertex cover set in the following way: on input G

   - For $k = 1, 2, \ldots n$ call k-VERTEX-COVER on input $\langle G, k\rangle$. Break out of the cycle when the answer is "yes". Let this value of k be r.
   - For each node $x \in V$,
     - add one extra edge (x, y_x) to G
     - call r-VERTEX-COVER on $\langle G, r\rangle$.
     - **If** the answer is "yes", x is in the minimum vertex-cover,
     - **else** remove the edge (x, y_x) from G.

Problem 5

Definition: a set of nodes in a graph is called "independent" if no two nodes in the set are adjacent. Give poly-time reductions between k-INDEPENDENT-SET and k-VERTEX-COVER (both ways).

Solution sketch: a set of nodes is independent if and only if the complement of the set is a vertex-cover. Se we can reduce both ways k-VERTEX-COVER and k-INDEPENDENT-SET using the mapping $\langle G, k \rangle \longleftrightarrow \langle G, |V| - k \rangle$ (V being the set of nodes).

Problem 6

Give poly-time reductions between k-INDEPENDENT-SET and k-CLIQUE.

Solution sketch: a set of nodes is an independent set if and only if the same set of nodes is a clique in the graph obtained by complementing the edges. Let the graph be $G = \langle V, E \rangle$, where V is a set of nodes and E is a set of edges. The reduction both ways is the following: $\langle V, E, k \rangle \longleftrightarrow \langle V, E^C, k \rangle$. E^C is the complement of the set of edges $E^C = V \times V - E - I$, where $I = \{(a, a) \mid a \in V\}$.