

# CS 6520: Computational Complexity

## Problem Set 1

Due March 4, 2008

### Problem 1

Give a Karp reduction from CLIQUE to SAT.

### Problem 2

Let QUADRATIC be the problem of deciding whether a given system of quadratic multivariate polynomial equations with integer coefficients has a solution modulo 2. Prove that QUADRATIC is **NP-Complete**.

### Problem 3

An **NP minimization problem** is defined by an objective function  $\text{Obj} : \Sigma^* \times \Sigma^* \rightarrow \mathbb{N}$  for which there is a constant  $c$  and an algorithm that for every  $x, y \in \{0, 1\}^*$ , computes  $\text{Obj}(x, y)$  in time  $O(|x|^c)$ . For such an objective function  $\text{Obj}$ , the corresponding **NP minimization problem** is: Given  $x \in \Sigma^*$ , find a  $y \in \Sigma^*$  such that  $\text{Obj}(x, y)$  is minimized. Prove that  $\mathbf{P} = \mathbf{NP}$  if and only if every **NP minimization problem** has a polynomial-time algorithm.<sup>1</sup>

### Problem 4

Prove that for each  $i \in \mathbb{N}$ ,  $\Sigma_i$ -SAT is  $\Sigma_i^P$ -**Complete**.

### Problem 5

Consider the FACTORING problem: Given a natural number  $N$ , express  $N$  as a product of its prime factors. To date no polynomial-time algorithm for FACTORING is known, and the conjectured hardness of FACTORING has been the basis of several public-key cryptosystems.

1. Prove that if  $\mathbf{P} = \mathbf{NP} \cap \mathbf{coNP}$ , then FACTORING can be solved by a polynomial-time algorithm.

You may find the following facts useful:

---

<sup>1</sup>An **NP maximization problem** can be defined similarly, for which an analogous result holds.

- The Fundamental Theorem of Arithmetics: Every integer greater than one can be decomposed into a product of prime factors, and moreover such a decomposition is *unique* up to the order of the prime factors.
  - There is a polynomial-time algorithm for deciding whether a given integer is a prime.
2. Prove that unless  $\mathbf{NP} = \mathbf{coNP}$ , FACTORING is *not* **NP-Hard** under Cook reduction. Conclude that unless  $\mathbf{NP} = \mathbf{PH}$ , FACTORING is *not* **NP-Hard** under Cook reduction.

### Problem 6

A language  $L$  is *self-reducible* if there is a polynomial-time oracle machine  $M$  such that (a)  $M^L$  decides  $L$ , and (b) on every input  $x$ ,  $M$  only makes queries of length strictly smaller than  $|x|$ . For instance, as shown in class, SAT and TQBF are self-reducible. Prove that every self-reducible language is in **PSPACE**.

### Problem 7

1. A directed graph  $G = (V, E)$  is *strongly connected* if for every pair of vertices  $u, v \in V$ , there is a path from  $u$  to  $v$  in  $G$ . Prove that the problem of deciding whether a given directed graph is strongly connected is **NL-Complete**.
2. Prove that 2-SAT is **NL-Complete**.
3. Let BIPARTITE be the language consisting of bipartite graphs. Prove that BIPARTITE  $\in$  **NL**.
4. Let USTCONN be the following problem: Given an *undirected* graph  $G = (V, E)$  and two vertices  $s, t \in V$ , decide whether there is a path from  $s$  to  $t$  in  $G$ . Prove that USTCONN and BIPARTITE are computationally equivalent under log-space reductions.

### Problem 8

1. Let  $s(n)$  be a space-constructible function. Prove that a language  $L \in \mathbf{NSPACE}(s(n))$  if and only if there is a deterministic Turing machine  $M$  with a read-once<sup>2</sup> certificate tape such that for every  $x \in \{0, 1\}^*$ ,  $x \in L$  if and only if there is a certificate  $y \in \{0, 1\}^{2^{O(s(|x|))}}$  such that  $M(x, y) = 1$ , where  $x$  is placed on  $M$ 's input tape,  $y$  is placed on  $M$ 's read-once certificate tape, and  $M$  uses  $O(s(|x|))$  space.
2. Does the above result hold if the head of the certificate tape of  $M$  is allowed to move in both directions? Justify your answer.

### Problem 9

Prove that  $\mathbf{P} \neq \mathbf{SPACE}(n)$ .

### Problem 10

Prove that there exist oracles  $A$  and  $B$  such that  $\mathbf{NP}^A = \mathbf{coNP}^A$  and  $\mathbf{NP}^B \neq \mathbf{coNP}^B$ .

### Problem 11

Prove that for every  $k \in \mathbb{N}$ , there is a language in  $\mathbf{PH}$  with circuit complexity  $\Omega(n^k)$ . For extra credit, prove that for every  $k \in \mathbb{N}$ , there is a language in  $\Sigma_2^P$  with circuit complexity  $\Omega(n^k)$ .

---

<sup>2</sup>That is, the head of the tape moves only from left to right.