

Homework I  
CS6520  
Computational Complexity

January 31, 2007

Due on Thu, Feb 8th.

1. An instance of 2-SAT consists of  $n$  boolean variables  $x_1, x_2, \dots, x_n$  and  $m$  clauses, and each clause contains at most 2 literals. We say that the instance is satisfiable if there is a {TRUE, FALSE}-assignment to  $x_1, x_2, \dots, x_n$  that satisfies every clause.
  - Show that deciding if an instance of 2-SAT is satisfiable is in P. *Hint: Given two clauses, one involving  $x_i$  and the other involving  $\bar{x}_i$  try and replace them with a single clause.*
  - Show that deciding if there is an assignment that satisfies at least  $k$  out of  $m$  clauses is NP-complete. Note that the parameter  $k$  is now part of the input. *Hint : Use a reduction from Vertex Cover.*
2. The class EXP is defined as

$$\text{EXP} = \bigcup_k \text{DTIME}(2^{n^k})$$

Show that  $\text{NP} \subseteq \text{EXP}$  and  $\text{co-NP} \subseteq \text{EXP}$ .

3. Show that if  $\text{P} = \text{NP}$ , there is a polynomial time algorithm to find a satisfying assignment to a 3-SAT formula if such an assignment exists.
4. Let BIPARTITE denote the language of all graphs which are bipartite. Show that  $\text{BIPARTITE} \in \text{NL}$ .
5. A directed graph is *strongly connected* if for every pair of vertices  $(s, t)$  there is a directed path from  $s$  to  $t$  in  $G$ . Show that the problem of deciding whether a graph is strongly connected is NL-complete.

6. The 0-1 knapsack problem is defined as follows: Let  $\{a_i\}_{i=1}^n, b$  be positive integers (represented in binary). The knapsack problem asks whether there is an integer solution to

$$\sum_{i=1}^n a_i X_i = b \quad \forall i \quad 0 \leq X_i \leq 1$$

It is well-known that this problem is NP-complete. Show that if we remove the constraints that  $X_i$ 's are 0-1 and allow them to be arbitrary integers, then the problem is in P. In other words, deciding if the following equation has an integer solution is in P.

$$\sum_{i=1}^n a_i X_i = b$$

7. A problem  $A$  is NP-hard if there is a polynomial time reduction to it from some NP-complete problem ( $A$  itself need not be in NP).
- Show that the following problem is NP-hard. Given a polynomial  $P(X_1, \dots, X_n)$  with integer coefficients, the problem is to decide whether the following equation has an integer solution :

$$P(X_1, \dots, X_n) = 0$$

*Hint : Show that in fact the problem is NP-hard for polynomials of degree 2, using a reduction from knapsack.*

- Can't we simply guess a solution if it exists and verify it ? Doesn't this mean that the problem is in NP ?
8. For a language  $L \subseteq \{0, 1\}^*$ , and a function  $f(n)$  (assume that  $f(n)$  is computable in time  $O(f(n))$ ), let  $L_f \subseteq \{0, 1, \#\}^*$  denote the following language :

$$L_f := \{x\#^{f(|x|)} \mid x \in L\}$$

- Suppose that  $L \in \text{DTIME}(f(n))$ . Then show that  $L_f \in \text{DTIME}(O(n))$ . Show similar results for non-deterministic time classes and deterministic space classes.
- Show that if  $f(n)$  is a polynomial function, then  $L \in \text{P}$  iff  $L_f \in \text{P}$ .
- Show that  $\text{P} \neq \text{DSpace}(O(n))$ . *Hint : Assume an equality and arrive at a contradiction via the Deterministic Space Hierarchy Theorem.*

- Define the class NEXP as

$$\text{NEXP} := \cup_k \text{NTIME}(2^{n^k})$$

Prove that if  $\text{P} = \text{NP}$  then  $\text{EXP} = \text{NEXP}$ .