

Assignment 1

COMP 531: Theory of Computation (Winter 06)

Due Feb 2 (Thu), before class

Instructions. Apart from correctness, clarity, and conciseness of your solutions are important. Please write your solutions legibly.

You benefit most from doing the problems if you do them alone, however there is also some benefit in collaboration [after all, a good number of research papers are written collaboratively!]. you may collaborate in groups of two or three, but you must write your solution on your own and should list the name of the persons with whom you collaborated. If you need to, you are welcome to come to my office hours to discuss the problems; I may give you some hints. Solutions to a number of problems here can be found in books, research papers, or on the web. You should not refer to these to solve the problems. If you happen to have inadvertently seen a solution please mention this in your work.

All problems are worth 10 points except the first one which is worth 30 pts (each part is 10 pts).

If you think there's an error in some problem, please let me know asap (navin@cs.mcgill.ca).

Problem 1. (Search vs decision) We showed in class that if the decision version of SAT (or in general any NP-complete problem) can be solved in polynomial time, then in fact a satisfying assignment can be found in polynomial time (if it exists). In other words, search reduces to decision for NP-complete problems. This problem asks you questions along similar lines.

1. Weighted clique problem (WCLIQUE). This is just like the usual CLIQUE problem, except that now the edges are wighted and the weight of a clique is the sum of the weights of the edges of the clique.

$$\text{WCLIQUE} = \{\langle G, t \rangle \mid \text{graph } G \text{ contains a clique with weight } \geq t\}.$$

Show that if WCLIQUE admits a polytime algorithm, then the problem of finding the *maximum weight clique* in a graph can also be solved on polynomial time.

2. Graph isomorphism problem (GI). We say that two undirected graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are *isomorphic* if $|V_1| = |V_2|$ and there's a permutation $\sigma : V_1 \rightarrow V_2$ such that for all $u, v \in V_1$ we have $(u, v) \in E_1$ iff $(\sigma(u), \sigma(v)) \in E_2$.

GI is the language

$$\{\langle G_1, G_2 \rangle \mid \text{graph } G_1 \text{ is isomorphic to graph } G_2\}.$$

The search problem for GI is to find a permutation σ as above if G_1 and G_2 are isomorphic. Show that if there is a polytime algorithm for GI then there's also a polytime algorithm for the search problem.

Remark. GI is one of the few natural problems in NP that are neither known to be in P nor are they known to be NP-complete. Thus the fact that search reduces to decision for NP-complete problems does not seem to be applicable here.

3. Factorization. Show that if SAT admits a polynomial time algorithm then there's a polytime algorithm for factoring integers.

Remark. As we mentioned in class, a natural decision problem associated with factorization is PRIMES which is known to be in P, but the search problem is not known to reduce to decision in this case.

Problem 2. By definition, if $P = NP$ then there's a polynomial time algorithm for SAT, and if the formula is satisfiable then it finds a satisfying assignment. It turns out that we can say a little more: Design an *explicit* algorithm which, given a satisfiable boolean formula, finds a satisfying assignment and furthermore is guaranteed to work in polynomial time if $P = NP$. If the formula is not satisfiable then the algorithm may run for exponential time before answering that the formula is not satisfiable.

Problem 3. Integer Linear Programming (ILP): Given an $m \times n$ matrix A with integer entries and a column vector \mathbf{b} of m integers, does there exist a column vector \mathbf{x} of n integers such that $A\mathbf{x} \geq \mathbf{b}$.

(1) Show that ILP is NP-hard by reducing 3SAT to ILP. (2) Is it obvious that $ILP \in NP$ as is usually the case when showing a problem to be NP-complete? If yes, then give a short proof, if no then indicate the difficulty.

Problem 4. For a language A , define

$$A_{<n} = \{x \mid x \in A, |x| < n\}.$$

Say that a language A is *nice* if whether $x \in A$ can be determined by a polynomial time algorithm allowed to ask polynomially many queries of the form " $y \in A_{<|x|}$?". Prove that if A is nice then $A \in PSPACE$.

Problem 5. Show that the classes P, L, PSPACE, NP, EXP are closed under logspace reductions.

Problem 6. We defined the language $Dyck_1$ in class. One can similarly define $Dyck_i$ for $i > 1$ by allowing i types of brackets; here we consider the case $i = 2$. Let $Dyck_2$ be the language of properly nested parantheses and brackets. For example, $(([[[]]]) \in Dyck_2$ but $(([])) \notin Dyck_2$.

Show that $Dyck_2 \in L$.

Problem 7. ShortestPATH problem is given by

$$\text{ShortestPATH} = \{\langle G, s, t, k \rangle \mid \text{where the shortest path from node } s \text{ to node } t \text{ is of length } k\}.$$

Prove that ShortestPATH $\in NL$.

Problem 8. In this problem we consider a special case of QBF problem in which the underlying boolean function is promised to be *monotone*. We say that a boolean function $f(x_1, \dots, x_n)$ is monotone if changing the value of a variable from 0 to 1 does not change the value of f from 1 to 0. In other words, if you increase the value of a variable then the value of f cannot decrease.

We define Monotone QBF to be the decision problem

$$\exists x_1 \forall x_2 \dots \Phi(x_1, \dots, x_n)?$$

where the function represented by formula Φ is monotone.

Do one of the following two things: (a) prove that Monotone QBF is PSPACE-complete; or (b) give an algorithm to solve arbitrary instances of Monotone QBF that runs in *time* polynomial in n .