CSE 531: Computability and Complexity — Autumn 2004

Problem Set #4 — Instructor: Venkatesan Guruswami

Due on Tuesday, **November 30, 2004** in class.

Instructions: Same as Problem Set 1.
Readings: Chapter 8.

---

1. An *unrestricted grammar* (or a *rewriting system*) is a 4-tuple $G = (V, \Sigma, R, S)$ where
   - V is an alphabet;
   - $\Sigma \subset V$ is the set of *terminal* symbols, and $V - \Sigma$ is called the set of *nonterminal* symbols;
   - $S \in V - \Sigma$ is the *start* symbol; and
   - R, the set of *rules*, is a finite subset of $(V^*(V - \Sigma)V^*) \times V^*$.

   (Thus the "only" difference from context-free grammars is that the left-hand sides of rules need not consist of single nonterminals.) Let us write $\alpha \to \beta$ if $(\alpha, \beta) \in R$; and let's define $u \Rightarrow_G v$ iff, for some $w_1, w_2 \in V^*$ and some rule $\alpha \to \beta \in R$, $u = w_1 \alpha w_2$ and $v = w_1 \beta w_2$. Let $\overset{*}{\Rightarrow}_G$ denote the reflexive, transitive closure of $\Rightarrow_G$. We say that a string $w \in \Sigma^*$ is generated by G if and only if $S \overset{*}{\Rightarrow}_G w$. Finally, $L(G) \subseteq \Sigma^*$, the *language generated by* G, is defined to be the set of all strings in Σ^* generated by G.

   Now, define a context-sensitive grammar to be one for which whenever $(x, y) \in R$ we have $|x| \leq |y|$ (i.e., the right hand side of rules are at least as long as the left hand size). Now to your exercises.

   (a) Prove that the class of languages generated by context-sensitive grammars is precisely NSPACE(n).
   (Hint: The harder direction is to prove that languages in NSPACE(n) are context-sensitive. For this, it might help to construct a grammar whose rules simulate backward moves of M (for an arbitrary TM M), and whose derivations will consequently simulate backward computations of M. This will show that unrestricted grammars can generate any Turing recognizable language. Now see how the space restriction can be used to argue that the grammar may be assumed to be context-sensitive.)

   (b) Use the above to show that the acceptance problem for context-sensitive languages, i.e., the language

   $$A_{\text{CSG}} = \{\langle G, w \rangle \mid G \text{ is a context-sensitive grammar that generates } w\}$$

   is PSPACE-complete. (Recall that, in contrast, we showed that the corresponding language A_{CFG} for context-free grammars is in P. It can be shown that A_{CFG} is in fact complete for the class P, under logspace reductions.)

2. Given an integer matrix $A \in \mathbb{Z}^{m \times n}$ and an integer vector $b \in \mathbb{Z}^m$, the pair (A, b) is said to be feasible if the system of linear inequalities $Ax \leq b$ has a solution $x = \langle x_1, x_2, \ldots, x_n \rangle \in \mathbb{R}^n$ over **reals**. Prove that the language

   $$\text{LINEAR-PROGRAMMING} = \{(A, b) \mid \text{ the pair } (A, b) \text{ is feasible}\} ,$$

is P-hard under logspace reductions, i.e., show that for every language $B \in \mathrm{P}$, we have $B \leq_L$ LINEAR-PROGRAMMING.

<u>Hint</u>: First prove that the language

$$\text{CIRCUIT-VAL} = \{(C, a) \mid \text{ circuit } C \text{ evaluates to } \mathsf{True} \text{ on assignment } a\}$$

is P-hard under logspace reductions. (Our proof of Cook-Levin theorem from class should be handy here.) Then give a logspace reduction from CIRCUIT-VAL to LINEAR-PROGRAMMING.

3. Problem 8.12, Sipser's book. (Properly nested parantheses and brackets is in L)

4. We know that 3SAT is NP-complete, but 2SAT (satisfiability of 2CNF formulae) has a polynomial time algorithm. In this exercise, your task is to pinpoint the complexity of 2SAT by proving that 2SAT is NL-complete.

5. Problem 8.15, Sipser's book (HAPPY-CAT is in P)
<u>Hint</u>: You can think about this problem directly, but if you are stuck, looking at Lemma 10.21 in the textbook might help.

6. This problem concerns ***branching programs*** which are described in Section 10.2 of Sipser's book. We briefly repeat the definition here. A branching program is a directed acyclic graph where all nodes are labeled by variables, except for two output nodes labeled 0 or 1. The nodes that are labeled by variables are called query nodes, each of which has two outgoing edges, one labeled 0 and the other labeled 1. Both output nodes have no outgoing edges, and one of the nodes of the branching program is designated the start node. A branching program determines a Boolean function as follows. Take any assignment to the variables appearing on its query nodes and, beginning at the start node, follow the path determined by taking the outgoing edge from each query node according to the value assigned to the indicated variable (i.e. take the 0-edge is the variable is 0 and 1-node if it is 1). Do this until one of the output nodes is reached. The label of this output node is the output of the branching program on that input.

   Define a ***family of branching programs*** $\mathcal{B} = (B_1, B_2, B_3, \dots)$ to be an infinite list of branching programs. The n'th member B_n of the list is a branching program that has n input variables $x_1, \dots, x_n$. Say that a family of branching programs ***decides a language*** $A \subseteq \{0,1\}^*$ if for every string a of some length j, $a \in A$ iff $B_j(a) = 1$. Here $B_j(a)$ denotes the output of the branching program B_j when its j input variables $x_1, \dots, x_j$ are set to the values $a_1, \dots, a_j$.

   Define the size of a branching program to be the number of nodes in it.

   (a) Give a diagram representing the n'th branching program for even n in a family deciding the language $\{w \mid w \in \{0,1\}^* \text{ and } w \text{ has an odd number of 1's}\}$. Your branching program should have size $O(n)$ to receive full credit.
   (b) Show that if A is a language in LOGSPACE, then A is decided by a family of branching programs where the n'th member of the family has at most $\mathrm{poly}(n)$ nodes. (Essentially, this shows that branching programs are to the class L what circuits are to the class P.)

7. * **(Optional Problem for Extra Credit)** Define the language

$$\text{CYCLE} = \{\langle G \rangle \mid G \text{ is an undirected graph that has some cycle}\}\ .$$

   Prove that CYCLE $\in$ LOGSPACE.