Spring, 2010 CIS 511

Introduction to the Theory of Computation
Jean Gallier

Homework 6

April 8, 2010; Due April 22, 2010

"A problems" are for practice only, and should not be turned in.

Problem A1. Prove that every context-free language is a recursive set.

Problem A2. Consider the definition of the Kleene T-predicate given in the notes in Definition 5.4.1.

(i) Verify that $T(x, y, z)$ holds iff x codes a RAM program, y is an input, and z codes a halting computation of P_x on input y.

(ii) Verify that the Kleene normal form holds:

$$\varphi_x(y) = \mathrm{Res}[\min z(T(x, y, z))].$$

"B problems" must be turned in.

Problem B1 (40 pts). A *linear context-free grammar* is a context-free grammar whose productions are of the form either

$$\begin{aligned} A &\longrightarrow uBv, \quad \text{or} \\ A &\longrightarrow u, \end{aligned}$$

where A, B are nonterminals and $u, v \in \Sigma^*$. A language is *linear context-free* iff it is generated by some linear context-free grammar.

(a) Prove that every regular language is linear context-free. Prove that if L is a linear context-free language, then for every $a \in \Sigma$, the language $L/a = \{w \in \Sigma^* \mid wa \in L\}$ is also linear context-free.

Hint. Construct a grammar using some new nonterminals, $[A/a]$, and new productions

$$[A/a] \longrightarrow \alpha, \quad \text{if} \quad A \longrightarrow \alpha a \in P \quad \text{or} \quad A \longrightarrow \alpha a B \in P \quad \text{with} \quad B \stackrel{+}{\Longrightarrow} \epsilon$$

and

$$[A/a] \longrightarrow u[B/a], \quad \text{if} \quad A \longrightarrow uB \in P,$$

(b) Prove that it is undecidable whether a context-free language, L, is linear context-free.

Hint. To prove part (b), you will need the fact that a certain property P is nontrivial, where P is defined so that for every context-free language, L, $P(L)$ holds iff L is linear-context-free. For this, you will need to prove that there is some context-free language that is not linear context-free. We claim that

$$L = \{a^m b^m c^n d^n \mid m, n \geq 1\}$$

is such a language, although this is not so easy to prove rigorously. One way to do so is to prove a special pumping lemma for the linear context-free languages (which you may use without proof).

Pumping Lemma for the linear context-free languages:

For every linear context-free grammar, $G = (V, \Sigma, P, S)$, there is some integer, $K \geq 1$, so that, for every $w \in \Sigma^*$, if $w \in L(G)$ and $|w| \geq K$, then there is some decomposition, u, v, x, y, z, of w so that

(1) $w = uvxyz$.

(2) $uv^n xy^n z \in L(G)$, *for all* $n \geq 0$.

(3) $v \neq \epsilon$ *or* $y \neq \epsilon$.

(4) $|uvyz| \leq K$.

The new ingredient in this pumping lemma is that $|uvyz| \leq K$. Then, you can use this pumping lemma to prove that $L = \{a^m b^m c^n d^n \mid m, n \geq 1\}$ is not linear context-free.

Problem B2 (30 pts). Given any set, X, for any subset, $A \subseteq X$, recall that the *characteristic function*, χ_A, of A is the function defined so that

$$\chi_A(x) = \begin{cases} 1 & \text{iff } x \in A \\ 0 & \text{iff } x \in X - A. \end{cases}$$

(i) Prove that, for any two subsets, $A, B \subseteq X$,

$$\begin{aligned} \chi_{A \cap B} &= \chi_A \cdot \chi_B \\ \chi_{A \cup B} &= \chi_A + \chi_B - \chi_A \cdot \chi_B. \end{aligned}$$

(ii) Given any $n \geq 2$ subsets, $A_1, A_2, \ldots, A_n \subseteq X$, prove that

$$\begin{aligned} \chi_{A_1 \cap \cdots \cap A_n} &= \chi_{A_1} \cdots \chi_{A_n} \\ \chi_{A_1 \cup \cdots \cup A_n} &= \sum_{\substack{I \subseteq \{1, \ldots, n\} \\ I \neq \emptyset}} (-1)^{|I|-1} \prod_{i \in I} \chi_{A_i} \end{aligned}$$

(iii) Prove that the union and the intersection of any two *r.e* sets, $A, B \subseteq \mathbb{N}$, is also an *r.e.* set. Prove that the union and the intersection of any two recursive sets, $A, B \subseteq \mathbb{N}$, is also a recursive set.

Problem B3 (30 pts). Given $\Sigma = \{a\}$, give a Turing machine accepting

$$L = \{a^{2^n} \mid n \geq 1\}.$$

Remember that your Turing machine must terminate either with a tape containing a single 1 and possibly some blanks iff the input is correct else with a tape containing a single 0 and possibly some blanks. You should run your Turing Machine interpreter (from HW5) on the TM recognizing L to make sure that it is correct! Make sure that you test all possible cases of wrong input!

Problem B4 (20 pts). Prove that the following properties of partial recursive functions are undecidable:

(a) A partial recursive function is a constant function.

(b) Two partial recursive functions φ_x and φ_y are identical.

(c) A partial recursive function φ_x is equal to a given partial recursive function φ_a.

(d) A partial recursive function diverges for all input.

Problem B5 (40 pts). Let A be any $p \times q$ matrix with integer coefficients and let $b \in \mathbb{Z}^p$ be any vector with integer coefficients. The 0-1 *integer programming problem* is to find whether the system

$$Ax = b$$

has any solution, $x \in \{0, 1\}^q$.

(i) Prove that the 0-1 integer programming problem is in $\mathcal{NP}$.

(ii) Prove that the 0-1 integer programming problem is $\mathcal{NP}$-complete by providing a polynomial-time reduction from the bounded-tiling problem. **Do not try to reduce any other problem to the** 0-1 **integer programming problem**.

Hint. Given a tiling problem, $((\mathcal{T}, V, H), \widehat{s}, \sigma_0)$, create a 0-1-valued variable, x_{mnt}, such that $x_{mnt} = 1$ iff tile t occurs in position (m, n) in some tiling. Write equations or inequalities expressing that a tiling exists and then use "slack variables" to convert inequalities to equations. For example, to express the fact that every position is tiled by a single tile, use the equation

$$\sum_{t \in \mathcal{T}} x_{mnt} = 1,$$

for all m, n with $m \neq 0$, $-s \leq m \leq s$ and $1 \leq n \leq s$.

(iii) Prove that the restricted 0-1 integer programming problem in which the coefficients of A are 0 or 1 and all entries in b are equal to 1 is also $\mathcal{NP}$-complete.

Problem B6 (50 pts). Let $\mathcal{NEXP}$ be the class of languages accepted in time bounded by $2^{p(n)}$ by a nondeterministic Turing machine, where $p(n)$ is a polynomial. Consider the problem of tiling a $2s \times s$ rectangle, described in Section 7.5 of the notes (slides on the web), but only with a single initial tile σ_0. Formally, an instance of this problem is of the form $((\mathcal{T}, V, H), \widehat{s}, \sigma_0)$, where s is the binary representation of $s \geq 2$ in binary and σ_0 is the initial tile in position $(1, 1)$.

(i) Prove that the above tiling problem is $\mathcal{NEXP}$-complete.

(ii) Now, consider the problem of tiling the *entire upper half-plane*, starting with a single tile σ_0 (of course, the set of tile patterns, $\mathcal{T}$, is finite). More precisely, this problem is said to have a solution if for every $s > 1$, there a function σ_s tiling the $2s \times s$-rectangle.

Prove that this problem is undecidable.

TOTAL: 210 points.