Spring, 2013 CIS 511

Introduction to the Theory of Computation
Jean Gallier

Homework 6

April 10, 2013; Due April 23, 2013, beginning of class

"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 (60 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 (40 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). Let $\Sigma = \{a_1, \ldots, a_k\}$ be some alphabet and suppose $g, h_1, \ldots, h_k$ are some total functions, with $g\colon (\Sigma^*)^{n-1} \to \Sigma^*$, and $h_i\colon (\Sigma^*)^{n+1} \to \Sigma^*$, for $i = 1, \ldots, k$. If we write $\overline{x}$ for $(x_2, \ldots, x_n)$, for any $y \in \Sigma^*$, where $y = a_{i_1} \cdots a_{i_m}$ (with $a_{i_j} \in \Sigma$), define the following sequences, u_j and v_j, for $j = 0, \ldots, m+1$:

$$
\begin{aligned}
u_0 &= \epsilon \\
u_1 &= u_0 a_{i_1} \\
&\vdots \\
u_j &= u_{j-1} a_{i_j} \\
&\vdots \\
u_m &= u_{m-1} a_{i_m} \\
u_{m+1} &= u_m a_i
\end{aligned}
$$

and

$$
\begin{aligned}
v_0 &= g(\overline{x}) \\
v_1 &= h_{i_1}(u_0, v_0, \overline{x}) \\
&\vdots \\
v_j &= h_{i_j}(u_{j-1}, v_{j-1}, \overline{x}) \\
&\vdots \\
v_m &= h_{i_m}(u_{m-1}, v_{m-1}, \overline{x}) \\
v_{m+1} &= h_i(y, v_m, \overline{x}).
\end{aligned}
$$

(i) Prove that

$$v_j = f(u_j, \overline{x}),$$

for $j = 0, \ldots, m+1$, where f is defined by primitive recursion from g and the h_i's, that is

$$
\begin{aligned}
f(\epsilon, \overline{x}) &= g(\overline{x}) \\
f(ya_1, \overline{x}) &= h_1(y, f(y, \overline{x}), \overline{x}) \\
&\vdots \\
f(ya_i, \overline{x}) &= h_i(y, f(y, \overline{x}), \overline{x}) \\
&\vdots \\
f(ya_k, \overline{x}) &= h_k(y, f(y, \overline{x}), \overline{x}),
\end{aligned}
$$

for all $y \in \Sigma^*$ and all $\overline{x} \in (\Sigma^*)^{n-1}$. Conclude that f is a total function.

(ii) Use (i) to prove that if g and the h_i's are RAM computable, then the function, f, defined by primitive recursion from g and the h_i's is also RAM computable.

Problem B4 (10 pts). Prove that the function, $f\colon \Sigma^* \to \Sigma^*$, given by

$$f(w) = a_1^{|w|}$$

is primitive recursive ($\Sigma = \{a_1, \ldots, a_N\}$).

Problem B5 (30 pts). *Ackermann's function* A is defined recursively as follows:

$$\begin{aligned} A(0,\, y) &= y+1, \\ A(x+1,\, 0) &= A(x,\, 1), \\ A(x+1,\, y+1) &= A(x,\, A(x+1, y)). \end{aligned}$$

Prove that

$$\begin{aligned} A(0,\, x) &= x+1, \\ A(1,\, x) &= x+2, \\ A(2,\, x) &= 2x+3, \\ A(3,\, x) &= 2^{x+3} - 3, \end{aligned}$$

and

$$A(4, x) = \left.2^{2^{\cdot^{\cdot^{\cdot^{2^{16}}}}}}\right\}{\scriptstyle x} - 3,$$

with $A(4, 0) = 16 - 3 = 13$. Equivalently (and perhaps less confusing)

$$A(4, x) = \left.2^{2^{\cdot^{\cdot^{\cdot^{2^{2}}}}}}\right\}{\scriptstyle x+3} - 3.$$

Problem B6 (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 B7 (60 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 $1 \leq m \leq 2s$ 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.

TOTAL: 250 points.