

Introduction to the Theory of Computation

Jean Gallier

Homework 6

April 14, 2009;

Revised Due Date: April 30, by 11:00am, 2009

“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:

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

“B problems” must be turned in.

Problem B1 (20 pts). Let us consider a version of a PDA (shift-reduce PDA) in which the transition function δ is defined as follows: We can have *push* moves

$$(q, YZ) \in \delta(p, a, Z),$$

where $Y, Z \in \Gamma$ (where Γ is the stack alphabet) $a \in (\Sigma \cup \{\epsilon\})$, $p, q \in K$, or *extended pop moves*

$$(q, \epsilon) \in \delta(p, a, \gamma),$$

$a \in (\Sigma \cup \{\epsilon\})$, $p, q \in K$, where $|\gamma| \geq 1$ ($\gamma \in \Gamma^+$). In an extended pop move, several stack symbols can be popped at once in a single move (when $|\gamma| \geq 2$).

Show that every shift-reduce PDA can be converted to an equivalent standard PDA.

Show that every standard PDA can be converted to an equivalent shift-reduce PDA.

Hint. First, convert the standard PDA to an equivalent one which contains a bottom-marker during any computation until the very end, and such that $(q, \gamma) \in \delta(p, a, Z)$ implies $|\gamma| \leq 2$. Simulate a move $(q, Y) \in \delta(p, a, Z)$ by a pop during which you remember that Z was on top,

followed by pushing YU , whatever U is currently on top. A move $(q, XY) \in \delta(p, a, Z)$ is simulated in a similar fashion, but you will use two push moves after the initial pop move.

Problem B2 (10 pts). Prove that the function, $f: \Sigma^* \rightarrow \Sigma^*$, given by

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

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

Problem B3 (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 B4 (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

$$A(4, x) = 2^{2^{\dots^{2^{16}}}} \}^x - 3,$$

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

$$A(4, x) = 2^{2^{\dots^{2^2}}} \}^{x+3} - 3.$$

Problem B5 (40 pts). Prove that a RAM program with $p \geq 2$ registers can be simulated by a RAM program with a single register, by encoding the contents r_1, \dots, r_p of the p registers as the string

$$r_1 \# r_2 \# \dots \# r_p,$$

using a new marker $\#$.

Hint: Begin by simulating a RAM program with two registers using a single register. Another (painful) solution is to simulate a Turing machine using a RAM with a single variable. In this case, the variable will represent the tape uav as $av\#u$. Some care must be exercised at both ends of the tape. Then, go from RAM to TM to RAM with a single register!

Conclude that the halting problem for RAM programs with one register is undecidable.

Problem B6 (20 pts). Prove that it is undecidable whether a context-free grammar generates a regular language.

Problem B7 (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 , and assume that s is given in binary.

(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: 190 points.