

Introduction to the Theory of Computation

Homework 5b

April 3, 2003; Due April 17, beginning of class

Important Note: You have the option to either write up the solution of Problem B6 or to do the extra credit of HW5a. If you do both, you will get extra points.

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

Problem A1. Give constructions proving that for any PDA M , the acceptance modes $T(M)$, $N(M)$, and $L(M)$, are equivalent.

Problem A2. (i) Given any set X , for any two subsets $A, B \subseteq X$, prove that $A \subseteq B$ iff $A \cap \overline{B} = \emptyset$, where \overline{B} denotes the complement of B in X .

(ii) Sketch an algorithm to test whether $L(G) \subseteq L(D)$, where G is a context-free grammar and D is a DFA.

Problem A3. Given a homomorphism $h: \Sigma^* \rightarrow \Delta^*$, for any any context-free language $L \subseteq \Delta^*$, prove that $h^{-1}(L)$ is context-free.

Hint. Find a construction using a PDA.

“B problems” must be turned in.

Problem B1 (40 pts). Use the pumping lemma (or Ogden’s lemma) to show that the following languages are not context-free:

$$L_1 = \{a^m b^n c^p \mid 1 \leq m < n < p\}$$

$$L_2 = \{a^n b^n c^p \mid n, p \geq 1, p \neq n\}$$

Hint. For L_1 , consider $a^m b^n c^p$ with m large, and let the a ’s be distinguished. For L_2 , let k be the integer of Ogden’s lemma. Let $p = k!$, and consider $a^{2p} b^{2p} c^p$, with the c ’s distinguished.

Problem B2 (30 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 B3 (50 pts). Give pushdown automata for the following languages:

- (a) $L_5 = \{ww^R \mid w \in \{a, b\}^*\}$ (w^R denotes the reversal of w)
- (b) $L_6 = \{a^m b^n \mid 1 \leq m \leq n \leq 3m\}$
- (c) $L_7 = \{a^n b^n \mid n \geq 1\} \cup \{a^{2n} b^n \mid n \geq 1\}$
- (d) $L_8 = \{a^{2m} b^n a^{2m} b^p \mid m, n, p \geq 1\} \cup \{a^m b^{3n} a^p b^{3n} \mid m, n, p \geq 1\}$
- (e) $L_9 = \{xcy \mid |x| = |y|, x, y \in \{a, b\}^*\}$

In each case, give a brief justification of the fact that your PDA generates the desired language.

Problem B4 (50 pts). Let $L \subseteq \{a\}^*$ be a context-free language. Prove that L is actually a regular language. Proceed as follows. If L is finite, this is obvious, thus, assume that L is infinite. Let $L = L(G)$, for some CFG G .

(i) Let $K > 1$ be the constant of the pumping lemma for G , and let $r = K!$. Prove the following fact: for every $w \in L$, if $|w| \geq K$, then

$$\{wa^{rn} \mid n \geq 0\} \subseteq L.$$

(ii) For every i such that $0 \leq i < r$, let

$$L_i = \{a^n \mid a^n \in L, n \geq K, n \equiv i \pmod{r}\}.$$

Clearly,

$$L = \{a^n \mid a^n \in L, n < K\} \cup \bigcup_{i=0}^{r-1} L_i.$$

If $L_i \neq \emptyset$, let z_i be the shortest string in L_i . Prove that

$$L_i = \{z_i a^{rm} \mid m \geq 0\}.$$

Conclude that L is regular.

(iii) Prove that it is decidable whether $L_i = \emptyset$.

(iv) Given a context-free language L over $\{a, b\}$, prove that it is decidable whether $\{a\}^* \subseteq L$.

Problem B5 (30 pts). (1) Given the alphabet $\Sigma_2 = \{a, b, \bar{a}, \bar{b}\}$, define the relation \simeq on Σ_2^* as follows: For all $u, v \in \Sigma_2^*$,

$$u \simeq v \quad \text{iff} \quad \exists x, y \in \Sigma_2^*, \quad u = xa\bar{a}y, \quad v = xy \quad \text{or} \quad u = xb\bar{b}y, \quad v = xy.$$

Let \simeq^* be the reflexive and transitive closure of \simeq , and let $D_2 = \{w \in \Sigma_2^* \mid w \simeq^* \epsilon\}$. Give a context-free grammar for D_2 , and justify your answer.

Note: Strings such as $a\bar{a}b\bar{b}$ and $ab\bar{b}\bar{a}$ are in D_2 .

(2) Given the alphabet $\Sigma_m = \{a_1, \dots, a_m, \bar{a}_1, \dots, \bar{a}_m\}$, define the relation \simeq on Σ_m^* as follows: For all $u, v \in \Sigma_m^*$,

$$u \simeq v \quad \text{iff} \quad \exists x, y \in \Sigma_m^*, \quad u = xa_i\bar{a}_iy, \quad v = xy, \quad \text{for some } i, 1 \leq i \leq m.$$

Let \simeq^* be the reflexive and transitive closure of \simeq , and let $D_m = \{w \in \Sigma_m^* \mid w \simeq^* \epsilon\}$. Give a context-free grammar for D_m , and justify your answer.

Note: D_m is known as the *Dyck set* on m letters.

Problem B6 (80 pts). In this problem, the fundamental property of LR-parsing (due to Knuth) is established.

For simplicity, let us consider context-free grammars without ϵ -rules. Given a reduced context-free grammar $G = (V, \Sigma, P, S')$ augmented with start production $S' \rightarrow S$, where S' does not appear in any other productions, the set C_G of *characteristic strings of G* is the following subset of V^* (watch out, not Σ^*):

$$C_G = \{\alpha\beta \in V^* \mid S' \xRightarrow{rm}^* \alpha B v \xRightarrow{rm} \alpha\beta v, \\ \alpha, \beta \in V^*, v \in \Sigma^*, B \rightarrow \beta \in P\}.$$

In words, C_G is a certain set of prefixes of sentential forms obtained in rightmost derivations: those obtained by truncating the part of the sentential form immediately following the rightmost symbol in the righthand side of the production applied at the last step.

The fundamental property of LR-parsing is that C_G is a *regular language*. A nondeterministic automaton N_{C_G} accepting C_G can be constructed according to the method described in Section 1 of the handout *A Survey of LR-Parsing Methods, etc.*. Please, review this construction.

(i) Let G be the following grammar:

$$\begin{aligned} S' &\rightarrow E \\ E &\rightarrow E + T \mid T \\ T &\rightarrow T * F \mid F \\ F &\rightarrow (E) \mid a \end{aligned}$$

with $\Sigma = \{+, *, (,), a\}$.

Give the automaton N_{C_G} for the grammar G .

(ii) Using the standard algorithms, give a deterministic finite automaton equivalent to N_{C_G} . Do not include the “dead state”.

(iii) You shall now prove that $L(N_{C_G}) = C_G!$ This proves the correctness of the method that you are going to implement in the programming project!

(1) Prove the following claim by induction on the length of rightmost derivations:

Claim 1: For any nonterminal A , for every rightmost derivation

$$A \xRightarrow{rm}^* \alpha B v \xRightarrow{rm} \alpha \beta v,$$

where $v \in \Sigma^*$, $B \in N$, and $\alpha, \beta \in V^*$, if we denote the first production in the above rightmost derivation as $A \rightarrow \delta$, then there is a computation on input $\alpha\beta$ from state $A \rightarrow \delta$ to the final state $B \rightarrow \beta$.

To prove this claim, you will have to show the following (think about it in terms of parse trees). For any nonterminal A , every rightmost derivation from A is either of the form

- (i) $A \xRightarrow{rm} \delta$, for some production $A \rightarrow \delta$, in which case $A = B$ and $\delta = \beta$, or of the form
- (ii) $A \xRightarrow{rm} \lambda B_i \rho \xRightarrow{rm}^* \lambda B_i w \xRightarrow{rm}^* \lambda \alpha_i B w_i w \xRightarrow{rm} \lambda \alpha_i \beta w_i w$, with $w, w_i \in \Sigma^*$, $A, B, B_i \in N$, $\lambda, \rho, \alpha_i, \beta \in V^*$, and where

$$B_i \xRightarrow{rm}^* \alpha_i B w_i \xRightarrow{rm} \alpha_i \beta w_i \quad \text{and} \quad \rho \xRightarrow{rm}^* w.$$

Let $B_i \rightarrow \delta_i$ be the first production applied in the rightmost derivation from B_i . In the first case, there is a computation in N_{C_G} from state $A \rightarrow \delta$ to the final state $A \rightarrow \delta$ (where again, $A \rightarrow \delta = B \rightarrow \beta$), and in the second case, there is a computation in N_{C_G} from state $A \rightarrow \delta$ to $B_i \rightarrow \delta_i$ on input λ , and a computation from state $B_i \rightarrow \delta_i$ to the final state $B \rightarrow \beta$ on input $\alpha_i \beta$.

Conclude that C_G is a subset of $L(N_{C_G})$.

(2) Prove the following claim by induction on the number of ϵ -transitions in a computation in N_{C_G} :

Claim 2: For any state $A \rightarrow \delta$, if there is a computation on input γ to some final state $B \rightarrow \beta$, then there is some rightmost derivation $A \xRightarrow{rm}^* \alpha B v \xRightarrow{rm} \alpha \beta v$, such that, the production applied in the first rightmost derivation step is $A \rightarrow \delta$, and $\gamma = \alpha \beta$.

For this, prove the following:

- (i) Either $\gamma = \delta$ and the computation is from state $A \rightarrow \cdot \delta$ to state $A \rightarrow \delta \cdot$, or
- (ii) δ is of the form $\lambda B_i \rho$, γ is of the form $\lambda \alpha_i \beta$, and there is a computation on input $\alpha_i \beta$ from some state of the form $B_i \rightarrow \cdot \delta_i$ to the final state $B_i \rightarrow \delta_i \cdot$, and a rightmost derivation as in Claim 1.

Conclude that $L(N_{C_G})$ is a subset of C_G , thus establishing that $C_G = L(N_{C_G})$.

TOTAL: 200 + 80 points.