CS30053 Foundations of Computing, Autumn 2005

Mid-semester examination : Solutions

---

1. *Consider the language*

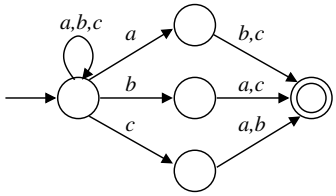
$$L_1 = \{\alpha \in \{a, b, c\}^* \mid \textit{the last two symbols of } \alpha \textit{ are different}\}.$$

(a) *Write a regular expression which generates the language L_1.*

Solution $(a \cup b \cup c)^*((ab) \cup (ac) \cup (bc) \cup (ba) \cup (ca) \cup (cb))$.

(b) *Design an NFA with five states to recognize L_1.*

Solution



2. *Let L be a regular language and n a pumping lemma constant for L. Clearly, any integer $\geqslant n$ can also be used as a pumping lemma constant for L. The smallest positive integer which is a pumping lemma constant for L is called the minimum pumping lemma constant for L. Determine the minimum pumping lemma constants for the languages over $\{a, b\}^*$ defined by the following regular expressions.*

(a) $(ab) \cup (ba)$.

Solution This language is finite. So there must not be any non-trivial pumping, i.e., the pumping lemma must be vacuously true. For that we require the minimum pumping lemma constant to be 3 (just larger than the lengths of all the strings in the language).

(b) $((ab) \cup (ba))^*$.

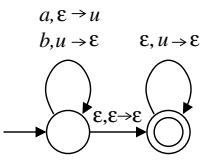
Solution The minimum pumping lemma constant in this case is 2. Given any string α of length $\geqslant 2$ in this language, we can pump the first two symbols any number of times. Moreover, pumping (in/out) a string of length 1 once (or any odd number of times) does not leave the string in the language.

3. *Let $\alpha = a_1 a_2 \ldots a_n$ be a string of length n. A string β is called a prefix of α if $\beta = a_1 a_2 \ldots a_i$ for some $i \in \{0, 1, 2, \ldots, n\}$ (the case $i = 0$ corresponds to $\beta = \epsilon$). Consider the language*

$$L_3 = \{\alpha \in \{a, b\}^* \mid \textit{no prefix of } \alpha \textit{ contains less } a\textit{'s than } b\textit{'s}\}.$$

(a) *Design a PDA to recognize L_3.*

Solution



(b) *Design a CFG G with $\mathcal{L}(G) = L_3$.*

Solution Imagine a as a left parenthesis and b as a right parenthesis. Then strings of L_3 can be obtained from strings of balanced parentheses after dropping zero or more right parentheses. So the following rules generate L_3:

$$S \rightarrow \epsilon \mid aSb \mid aS \mid SS.$$

4. *Consider the language*

$$L_4 = \{\alpha \in \{a,b\}^* \mid |\alpha| = n^2 \text{ for some integer } n \geqslant 0\},$$

where $|\alpha|$ denotes the length of α.

(a) *Prove that L_4 is not context-free.*

Solution Assume that L_4 is context-free, and let n be a PL constant for L_4. Take $\alpha = a^{n^2}$. The PL gives us a decomposition $\alpha = \alpha_1\alpha_2\alpha_3\alpha_4\alpha_5$ with $0 < |\alpha_2\alpha_4| = k \leqslant n$ and with $\alpha' = \alpha_1\alpha_2^2\alpha_3\alpha_4^2\alpha_5 \in L_4$. But $n^2 < |\alpha'| = n^2 + k < (n+1)^2$, a contradiction.

(b) *Prove that the complement $\overline{L_4} = \{a,b\}^* \setminus L_4$ is also not context-free.*

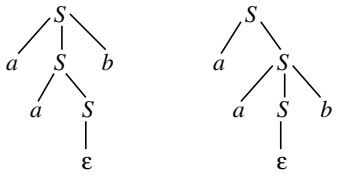
Solution Again assume that $\overline{L_4}$ is context-free, and let n be a PL constant for $\overline{L_4}$. Assume $n \geqslant 2$, so that $n!$ is not a perfect square. Take $\alpha = a^{n!}$. The PL gives a decomposition $\alpha = \alpha_1\alpha_2\alpha_3\alpha_4\alpha_5$ with $0 < |\alpha_2\alpha_4| = k \leqslant n$ and with $\beta_r = \alpha_1\alpha_2^r\alpha_3\alpha_4^r\alpha_5 \in \overline{L_4}$ for all $r \in \mathbb{N}_0$. Since $k \leqslant n$, $n!$ is a multiple of k, i.e., $s = [(n!)^2 - (n!)]/k$ is an integer. Taking $r = s+1$ implies that $\overline{L_4}$ contains a string of length $(n!)^2$, a contradiction.

5. *Let G be the context-free grammar $G = (\{S\}, \{a,b\}, S, R)$ with R consisting of the following rules:*

$$S \to \epsilon \mid aS \mid aSb\,.$$

(a) *Prove that G is ambiguous.*

Solution The following figure shows two different parse trees for the string aab.



(b) *Provide an unambiguous grammar for $\mathcal{L}(G)$.*

Solution

$$\begin{array}{lcll} S & \to & aSb \mid T & \text{[first generate the matching } a\text{'s and } b\text{'s]} \\ T & \to & aT \mid \epsilon & \text{[then generate the excess } a\text{'s]} \end{array}$$

6. **(a)** *Let L and L' be context-free languages. Demonstrate by an example that the language $L \setminus L'$ is not necessarily context-free.*

Solution Let $\Sigma = \{a,b\}$. Take $L = \Sigma^*$. L is regular (generated by the regular expression $(a \cup b)^*$) and so context-free. Finally, take L' to be the complement of the language $\{\alpha\alpha \mid \alpha \in \{a,b\}^*\}$.

(b) *Prove that if L is a context-free language and R a regular language, then $L \setminus R$ is context-free.*

Solution $L \setminus R = L \cap \bar{R}$. R is regular and so closed under complementation, i.e., $\bar{R}$ is regular too. Let P be a PDA recognizing L and D a DFA recognizing R. We run P and D in parallel and use a single stack to be used by P. Thus the set of states of the PDA for $L \cap \bar{R}$ will be $Q_P \times Q_D$. There is only a small problem here: P allows ϵ transitions, whereas D does not. In order to cope up with this difficulty, we augment the transition function of D by $\delta_D(q,\epsilon) = q$ for all states q of D. The remaining details are easy and left to the students.