2G1505 Automata Theory

SOLUTIONS TO EXAM OF:
17 December 2003

Dilian Gurov
KTH/IMIT/LECS

---

1. Convert the nondeterministic automaton given below to an equivalent deterministic one using the subset construction. Omit inaccessible states. Draw the graph of the resulting DFA.

		a	b	
$\rightarrow$	q_1	q_1, q_2	q_3	F
	q_2	$-$	q_4	F
$\rightarrow$	q_3	q_4	$-$	
	q_4	$-$	q_1, q_4	

Solution: (given as a table)

		a	b	
$\rightarrow$	$\{q_1, q_3\}$	$\{q_1, q_2, q_4\}$	$\{q_3\}$	F
	$\{q_1, q_2, q_4\}$	$\{q_1, q_2\}$	$\{q_1, q_3, q_4\}$	F
	$\{q_1, q_3, q_4\}$	$\{q_1, q_2, q_4\}$	$\{q_1, q_3, q_4\}$	F
	$\{q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_3, q_4\}$	F
	$\{q_3, q_4\}$	$\{q_4\}$	$\{q_1, q_4\}$	
	$\{q_1, q_4\}$	$\{q_1, q_2\}$	$\{q_1, q_3, q_4\}$	F
	$\{q_4\}$	$\{\}$	$\{q_1, q_4\}$	
	$\{q_3\}$	$\{q_4\}$	$\{\}$	
	$\{\}$	$\{\}$	$\{\}$	

---

2. For the deterministic automaton given below, apply the minimization algorithm of Lecture 14 to compute the equvalence classes of the collapsing relation $\approx$ defined in Lecture 13. Show clearly the computation steps. List the equivalence classes, and apply the quotient construction to derive a minimized automaton. Draw its graph.

		a	b	
$\rightarrow$	q_0	q_0	q_1	
	q_1	q_2	q_3	
	q_2	q_2	q_3	
	q_3	q_2	q_4	F
	q_4	q_0	q_1	

Solution: (incomplete) After applying the minimization algorithm we obtain that $q_0 \approx q_4$ and $q_1 \approx q_2$. Thus we obtain the following minimized automaton (given as a table).

		a	b	
$\rightarrow$	$\{q_0, q_4\}$	$\{q_0, q_4\}$	$\{q_1, q_2\}$	
	$\{q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_3\}$	
	$\{q_3\}$	$\{q_1, q_2\}$	$\{q_0, q_4\}$	F

---

3. Let $A \subseteq \Sigma^*$ be a language. We define its prefix closure as:

$$\textbf{pref}\ A = \{x \in \Sigma^* \mid \exists y \in \Sigma^*.\ x \cdot y \in A\}$$

Prove that regular languages are closed under prefixing: if language A is regular, then so is **pref** A.

Solution: Let $A \subseteq \Sigma^*$ be regular. Then there must be a DFA $M_A = (Q, \Sigma, \delta, s, F)$ such that $L(M_A) = A$. Now, based on M_A, define another automaton $M = (Q, \Sigma, \delta, s, F')$ so that:

$$F' = \left\{q \in Q \mid \exists y \in \Sigma^*.\ \hat{\delta}(q, y) \in F\right\}$$

We will show that this automaton accepts the language **pref** A, and hence that **pref** A is regular.

Claim. $L(M) = \textbf{pref}\ A$

Proof.

$$\begin{array}{rcll} x \in L(M) & \Leftrightarrow & \hat{\delta}(s, x) \in F' & \{\text{def. } L(M)\} \\ & \Leftrightarrow & \exists y \in \Sigma^*.\ \hat{\delta}(\hat{\delta}(s, x), y) \in F & \{\text{def. } F'\} \\ & \Leftrightarrow & \exists y \in \Sigma^*.\ \hat{\delta}(s, x \cdot y) \in F & \{\text{HW 1.3, page 301}\} \\ & \Leftrightarrow & \exists y \in \Sigma^*.\ x \cdot y \in A & \{L(M_A) = A\} \\ & \Leftrightarrow & x \in \textbf{pref}\ A & \{\text{def. } \textbf{pref}\ A\} \end{array}$$

4. Consider the language:

$$A = \{x \in \{a, b\}^* \mid \sharp a(x) < \sharp b(x)\}$$

(a) Give a context–free grammar G for A. Explain your choice of productions.

Solution: There are many possible solutions. One way of looking at the strings of the language is to devide these into the ones which have exactly one occurrence of b more than occurrences of a, and those that have more. A string is in the first group exactly when it can be represented as a string of the shape $e_1 \cdot b \cdot e_2$, where e_1 and e_2 are strings with an equal number of a's and b's. Thus, e_1 and e_2 can be produced by the grammar $E \to \epsilon \mid aEb \mid bEa \mid EE$. A string is in the second group exactly when it is the concatenation of two strings of A. So we arrive at the following grammar:

$$\begin{array}{rcl} S & \to & EbE \mid SS \\ E & \to & \epsilon \mid aEb \mid bEa \mid EE \end{array}$$

(b) Construct an NPDA accepting A on empty stack. Explain its workings.

Solution: Again, there is a number of good solutions. One elegant solution using ε-transitions (proposed by one of the students at the exam) is based on the observation that if we use the "standard" productions for comparing occurrences:

$$\begin{array}{ll} \langle q, \bot\rangle \overset{a}{\hookrightarrow} \langle q, A\bot\rangle & \langle q, \bot\rangle \overset{b}{\hookrightarrow} \langle q, B\bot\rangle \\ \langle q, A\rangle \overset{a}{\hookrightarrow} \langle q, AA\rangle & \langle q, B\rangle \overset{b}{\hookrightarrow} \langle q, BB\rangle \\ \langle q, B\rangle \overset{a}{\hookrightarrow} \langle q, \varepsilon\rangle & \langle q, A\rangle \overset{b}{\hookrightarrow} \langle q, \varepsilon\rangle \end{array}$$

then a string is in A exactly when after reading it the stack contains only B's (on top of $\bot$). Note that there must be at least one such B. So, we can obtain the desired behaviour by adding two more productions:

$$\begin{array}{l} \langle q, \bot\rangle \overset{b}{\hookrightarrow} \langle q, B\rangle \\ \langle q, B\rangle \overset{\varepsilon}{\hookrightarrow} \langle q, \varepsilon\rangle \end{array}$$

5. Apply the Pumping Lemma for context–free languages (as a game with the Demon) to show that the language:

$$A = \left\{ a^n b^n a^j \mid j \leq n \right\}$$

is not context–free.

Solution:

– Demon picks an arbitrary $k \geq 0$.

\+ We pick $z = a^k b^k a^k$ which is in A.

– Demon picks u, v, w, x, y such that $z = uvwxy$, $|vx| > 0$, $|vwx| \leq k$.

\+ If $vwx = a^l b^m$ for some $l, m \geq 0$, we pick $i = 0$. Otherwise we pick $i = 2$.

Since $|vwx| \leq k$, vwx has either the shape $a^l b^m$ or $b^m a^l$ for some l, m such that $l + m \leq k$. In the first case xv^0wx^0y must be of the shape $a^p b^q a^k$ for some p, q such that $p + q < 2k$, and thus is not in A. In the second case, xv^2wx^2y will either not be in $L(a^*b^*a^*)$ at all, and thus not in A, or else xv^2wx^2y must be of the shape $a^k b^p a^q$ for some p, q such that $p + q > 2k$, and thus not in A. Hence, we win the game in all cases, which shows that A is not context–free.

---

6. Give a detailed description (preferably as a graph) of a total Turing machine accepting the language:

$$A = \left\{ a^n b^{\frac{n(n+1)}{2}} \mid n \geq 0 \right\}$$

Explain the underlying algorithm.

Solution (just a very brief sketch for now) We use the well–known equation $\sum_{i=1}^{n} = \frac{n(n+1)}{2}$. We proceed in rounds, in each round marking an a and then marking as many b's as there are marked a's (which equals the number of the current round).