2G1505 Automata Theory

SOLUTIONS TO EXAM OF:
16 October 2002, 14.oo - 19.oo

Dilian Gurov
KTH/IMIT/LECS

---

1. Give a DFA for the language defined by the regular expression a^*a, and another one for $a(ba)^*$. The union of the two DFAs defines a non-deterministic FA for the language $a^*a + a(ba)^*$. $\boxed{5\text{p}}$

   (a) Apply the subset construction to this NFA to produce a DFA for this language. Omit the inaccessible states. Draw the graph of the resulting DFA.

   (b) Is this DFA minimal? If not, which states are equivalent?

   Solution: The graphs to appear later. $\smile$

---

2. Show that regular languages are closed under doubling: if language L is regular, then so is also the language $L_2 \triangleq \{\textbf{two}\ x \mid x \in L\}$, where string doubling is defined inductively by $\textbf{two}\ \epsilon \triangleq \epsilon$ and $\textbf{two}\ ax \triangleq aa \cdot (\textbf{two}\ x)$. $\boxed{5\text{p}}$

   Solution: Here is a standard solution using finite automata; alternative solutions exist using regular expressions or homomorphisms.

   Let L be regular. Then there is an NFA $N = (Q_N, \Sigma, \Delta_N, S_N, F_N)$ such that $L(N) = L$. Define another NFA, N_2, as follows: start with N and replace every edge $q \xrightarrow{a} q'$ with two edges $q \xrightarrow{a} q''$ and $q'' \xrightarrow{a} q'$ by inserting (for each edge!) a new state q''. We can formalize this idea by taking as states of N_2 the states of N plus the edges of N, the latter represented for example as the set of triples $\{(q_N, a, q'_N) \mid q_N \in Q_N,\ q'_N \in \Delta_N(q_N, a)\}$. So, we can define N_2 as follows:

   - $Q_{N_2} \triangleq Q_N \cup \{(q_N, a, q'_N) \mid q_N \in Q_N,\ q'_N \in \Delta_N(q_N, a)\}$
   - Δ_{N_2} is given by the two defining equations:
     $\Delta_{N_2}(q_N, a) \triangleq Q_N \cup \{(q_N, a, q'_N) \mid q'_N \in \Delta_N(q_N, a)\}$ and
     $\Delta_{N_2}((q_N, b, q'_N), a) \triangleq$ if $a = b$ then $\{q'_N\}$ else $\emptyset$
   - $S_{N_2} \triangleq S_N$
   - $F_{N_2} \triangleq F_N$

   It is straightforward to show that, for the so constructed NFA, $L(N_2) = L_2$, thus implying that L_2 is regular. Hence, regular languages are closed under doubling.

---

3. Consider the language L defined by the regular expression $(a^* + ba)^*$. Describe the equivalence classes of $\{a, b\}^*$ w.r.t. the Myhill–Nerode relation $\equiv_L$ defined by: (cf. equation (16.1) on page 97) $\boxed{5\text{p}}$

   $$x_1 \equiv_L x_2 \overset{\Delta}{\iff} \forall y \in \Sigma^*.(x_1 \cdot y \in L \Leftrightarrow x_2 \cdot y \in L)$$

   Present these equivalence classes through regular expressions. Use $\equiv_L$ to construct a minimal automaton $M_{\equiv_L}$ (cf. page 91) for the language L, and draw the graph of the automaton.

   Solution: The states of the quotient automaton are:

   $$\Sigma^*/_{\equiv_L} = \{L((a + ba)^*),\ L((a + ba)^*b),\ L((a + ba)^*bb(a + b)^*)\}$$

   The graph to appear later.

---

4. Consider the language: **15p**

$$L \triangleq \left\{ x \cdot y \in \{a,b\}^+ \mid y = \textbf{rev}\ x \right\}$$

where **rev** x denotes the reverse string of x (cf. HW 2.2, page 302).

(a) Use the Pumping Lemma to prove that L is *not* regular.

Solution: As a game with the Demon (cf. Lecture 11):

- Demon picks $k > 0$.
- We pick for example $x = \epsilon$, $y = a^k$, $z = bba^k$, and we have $xyz = a^k bba^k \in L$ and $|y| \geq k$.
- Demon picks $uvw = y = a^k$, $v \neq \epsilon$.
- We pick for example $i = 0$.

Then $xuv^iwz = uwz = a^j bba^k$ for some $j < k$, and hence $xuv^iwz \notin A$. We have a winning strategy, and L is therefore not regular.

(b) Give a context–free grammar G for L.

Solution: $S \to aa \mid bb \mid aSa \mid bSb$

(c) Prove your grammar correct (cf. Lecture 20): that is, prove $L = L(G)$.

Solution: We have to prove:

$$\forall x \in \{a,b\}^+.\ (S \xrightarrow{*}_G x \Leftrightarrow x \in L)$$

We show the two directions of the equivalence separately.

($\Rightarrow$) By induction on the length of the derivation of x.

<u>Basis</u> Holds vacuously, since $S \xrightarrow{0}_G x$ is false: $x = S$ is impossible since $S \notin \{a,b\}^+$.

<u>Induction Step</u> Assume $x' \in L$ for all x' such that $S \xrightarrow{n}_G x'$ (induction hypothesis).

Let $S \xrightarrow{n+1}_G x$. Then, we must have $S \xrightarrow{1}_G \gamma$ and $\gamma \xrightarrow{n}_G x$ for some $\gamma \in \{a,b,S\}^+$. But then γ can only be aa or bb or aSa or bSb. The first two cases imply $n = 0$ and $x = \gamma$, and then obviously $x \in L$. In the case $\gamma = aSa$, it must be that $x = ax'a$ and $S \xrightarrow{n}_G x'$ for some x'. From the induction hypothesis, we have $x' \in L$. But $x = ax'a$, and therefore also $x \in L$.

The case $\gamma = bSb$ is similar.

($\Leftarrow$) By induction on $|x|$, which is even and positive.

<u>Basis</u> $|x| = 2$, then $x \in L$ implies that x is either aa or bb. In both cases $S \xrightarrow{1}_G x$ and therefore $S \xrightarrow{*}_G x$.

<u>Induction Step</u> Assume $S \xrightarrow{*}_G x'$ for all $x' \in L$ such that $|x'| = n$ (induction hypothesis).

Let $|x| = n + 2$, and let $x \in L$. It must be that either $x = ax'a$ or $x = bx'b$ for some x' such that $x' \in L$ and $|x'| = n$. From the induction hypothesis, $S \xrightarrow{*}_G x'$. Then, in the case $x = ax'a$ we also have $aSa \xrightarrow{*}_G ax'a = x$, and since $S \xrightarrow{1}_G aSa$, then $S \xrightarrow{*}_G x$.

The case $x = bx'b$ is similar.

(d) Give an NPDA for L.

Solution: One possibility is to put G in GNF:

$$
\begin{aligned}
&S \to aA \mid bB \mid aSA \mid bSB \\
&A \to a \\
&B \to b
\end{aligned}
$$

and then construct the NPDA canonically (cf. Lecture 24):

$$
\begin{aligned}
&Q \triangleq \{q\} \\
&\Sigma \triangleq \{a, b\} \\
&\Gamma \triangleq \{S, A, B\} \\
&\delta \triangleq \begin{array}{l}
\langle q, S\rangle \xhookrightarrow{a} \langle q, SA\rangle \\
\langle q, S\rangle \xhookrightarrow{a} \langle q, A\rangle \\
\langle q, S\rangle \xhookrightarrow{b} \langle q, SB\rangle \\
\langle q, S\rangle \xhookrightarrow{b} \langle q, B\rangle \\
\langle q, A\rangle \xhookrightarrow{a} \langle q, \epsilon\rangle \\
\langle q, B\rangle \xhookrightarrow{b} \langle q, \epsilon\rangle
\end{array} \\
&s \triangleq q \\
&\perp \triangleq S
\end{aligned}
$$

---

5. Give a detailed description of a total Turing machine accepting the palindromes over $\{a, b\}$: that is, all strings $x \in \{a, b\}^*$ such that $x = \mathbf{rev}\ x$. $\boxed{\text{5p}}$

Solution: We build a machine which repeatedly scans the tape from left to right, trying to match the first input symbol (which is directly replaced by the blank symbol) with the last input symbol (also directly replaced by the blank symbol).
Graph to appear later.

---

6. Argue that acceptance is not decidable: that is, that there is no total Turing machine M_A accepting the language $L_A \triangleq \left\{\hat{M}\sharp\hat{x} \mid M \text{ accepts } x\right\}$, by reducing from the Halting problem. $\boxed{\text{5p}}$

Solution: A TM halts on input x if it either accepts x or otherwise rejects x. We use this observation to show that the Halting problem (cf. Lecture 31) can be reduced to the above acceptance problem.

Assume that there is a total Turing machine M_A accepting L_A. We can then build a machine M_R which, on any input $\hat{M}\sharp\hat{x}$, first swaps the values of t and r in $\hat{M}$ (that is, swaps the accepting and the rejecting states of M), and then behaves exactly like M_A. Hence M_R is a total Turing machine deciding rejection. We can now combine M_A and M_R to produce a total Turing machine M_H deciding the Halting problem: for example, on any input $\hat{M}\sharp\hat{x}$, let M_H first run as M_A and accept if M_A accepts, but continue as M_R if M_A rejects. M_H will thus accept $\hat{M}\sharp\hat{x}$ if M halts on x, and will reject $\hat{M}\sharp\hat{x}$ otherwise.

But the Halting problem is undecidable, and therefore there is no total Turing machine M_A accepting L_A. The acceptance problem is therefore undecidable.
