DD2371 Automata Theory

Examination Problems
with solutions
19 May 2008

Dilian Gurov
KTH CSC
08-790 81 98

---

1. Consider the language over the alphabet $\{a, b\}$ consisting of all strings that do not contain 2 or more consecutive a's and do not end with b. E

   (a) Construct a *deterministic finite automaton* (DFA) that accepts this language. Draw its graph.

   Solution: For example, see solution to problem 2(c).

   (b) Suggest a *regular expression* that generates this language.

   Solution: For example, the regular expression $(a + \epsilon)(bb^*a)^*$.

---

2. For the deterministic automaton given below, apply the *minimization* algorithm of textbook Lecture 14 to compute the equvalence classes of the collapsing relation $\approx$ defined in textbook Lecture 13. E,D

	a	b
$\rightarrow$ q_0 F	q_1	q_3
q_1 F	q_2	q_3
q_2	q_2	q_5
q_3	q_4	q_6
q_4 F	q_2	q_6
q_5	q_2	q_5
q_6	q_7	q_3
q_7 F	q_5	q_6

   (a) Show clearly the computation steps (use tables).

   (b) List the computed equivalence classes.

   Solution: The equivalence classes are: $\{q_0\}$, $\{q_1, q_4, q_7\}$, $\{q_2, q_5\}$ and $\{q_3, q_6\}$.

   (c) Apply the *quotient construction* of textbook Lecture 13 to derive the minimized automaton. Draw its graph.

   Solution: (as a table)

	a	b
$\rightarrow$ $\{q_0\}$ F	$\{q_1, q_4, q_7\}$	$\{q_3, q_6\}$
$\{q_1, q_4, q_7\}$ F	$\{q_2, q_5\}$	$\{q_3, q_6\}$
$\{q_2, q_5\}$	$\{q_2, q_5\}$	$\{q_2, q_5\}$
$\{q_3, q_6\}$	$\{q_1, q_4, q_7\}$	$\{q_3, q_6\}$

---

3. Show that the class of regular languages is *closed* under the following unary operation on languages:

$$\mathbf{min}\ A \stackrel{\text{def}}{=} \{x \in A \mid \text{no proper prefix of } x \text{ is in } A\}$$

(a) Define a construction on finite automata that has the corresponding effect on the accepted language. D,C

Solution: An important result about a *deterministic* finite automaton $M = (Q, \Sigma, \delta, s, F)$ is that $\hat{\delta}(s, x \cdot y) = \hat{\delta}(\hat{\delta}(s, x), y)$. So, if both x and y are accepted by M, the unique path from s to $\hat{\delta}(s, x \cdot y) \in F$ has to pass state $\hat{\delta}(s, x) \in F$. Then, to elimiminate all suffixes of words in $\mathcal{L}(M)$, one has to eliminate all paths starting (and ending) in accept states of M. This is easily achieved by removing all outgoing edges from all accept states. Note that this results in a *nondeterminstic* finite automaton! So, we define:

$$\begin{aligned} N &\stackrel{\text{def}}{=} (Q, \Sigma, \Delta, \{s\}, F) \\ \Delta(q, a) &\stackrel{\text{def}}{=} \{\delta(q, a) \mid q \notin F\} \end{aligned}$$

(b) Prove the construction correct. B,A

Solution: (Sketch) The important result that is needed here is that $\hat{\Delta}(\{q\}, x)$ equals $\left\{\hat{\delta}(q, x)\right\}$ exactly when the unique path from q to $\hat{\delta}(q, x)$ does not pass through an accepting state (since we removed all their outgoing edges), and is the empty set otherwise. Formally:

$$\hat{\Delta}(\{q\}, x) = \begin{cases} \left\{\hat{\delta}(q, x)\right\} & \text{if for no proper prefix } y \text{ of } x,\ \hat{\delta}(q, y) \in F \\ \emptyset & \text{otherwise} \end{cases}$$

After proving this Lemma, it is straightforward to prove that $x \in \mathcal{L}(N) \Leftrightarrow x \in \mathbf{min}\ \mathcal{L}(M)$.

4. Consider the regular language A over alphabet $\Sigma = \{a, b\}$ defined through the regular expression $(ab + b)^*$. Recall the *Myhill–Nerode Theorem*, textbook Lecture 16, with the equivalence relation $\equiv_A$ on strings over Σ defined by: (cf. equation (16.1) on page 97) D,C

$$x_1 \equiv_A x_2 \stackrel{\text{def}}{\Longleftrightarrow} \forall y \in \Sigma^*.(x_1 \cdot y \in A \Leftrightarrow x_2 \cdot y \in A)$$

(a) Show the equivalence classes of Σ^* w.r.t. equivalence $\equiv_A$, represented as regular expressions.

Solution: There are three equivalence classes. They can be represented by the regular expressions: $(ab + b)^*$, $(ab + b)^* a$ and $(ab + b)^* aa(a + b)^*$.

(b) For every pair of (different) equivalence classes A_1 and A_2, give the shortest *distinguishing experiment*, by means of a string $y \in \Sigma^*$ such that $x_1 \cdot y \in A \Leftrightarrow x_2 \cdot y \notin A$ for any $x_1 \in A_1$ and $x_2 \in A_2$.

Solution: $(ab + b)^*$ is distinguished from $(ab + b)^* a$ by ϵ, $(ab + b)^*$ is distinguished from $(ab + b)^* aa(a + b)^*$ by ϵ, and $(ab + b)^* a$ is distinguished from $(ab + b)^* aa(a + b)^*$ by b.

5. Consider the language family $\boxed{\text{B,A}}$

$$A_n \stackrel{\text{def}}{=} \{x \in \{a,b\}^* \mid \text{for every prefix } y \text{ of } x,\ 0 \leq \sharp a(y) - \sharp b(y) \leq n\}$$

Prove formally that for every n, the minimal DFA accepting A_n has exactly $n+2$ states.

Solution: We know that the minimal DFA accepting A_n is unique up to isomorphism. Then, one can prove the above result by exhibiting a DFA for A_n that has exactly $n+2$ states, and is minimal, in the sense that no two of its states are equivalent.

For a given n, define the DFA

$$M_n \stackrel{\text{def}}{=} (\{q_0, \ldots, q_{n+1}\}, \{a,b\}, \delta, q_0, \{q_0, \ldots, q_n\})$$

where $\delta(q_i, a) \stackrel{\text{def}}{=} q_{i+1}$ for $0 \leq i \leq n$, $\delta(q_{i+1}, b) \stackrel{\text{def}}{=} q_i$ for $0 \leq i < n$, $\delta(q_0, b) \stackrel{\text{def}}{=} q_{n+1}$, $\delta(q_{n+1}, a) \stackrel{\text{def}}{=} q_{n+1}$, and $\delta(q_{n+1}, b) \stackrel{\text{def}}{=} q_{n+1}$. M_n has $n+2$ states, and it is easy to see that M_n accepts A_n.

We now show that M_n is minimal. For $0 \leq i < j \leq n$, we have $q_i \not\approx q_j$ with witness b^{i+1} (since $\hat{\delta}(q_i, b^{i+1}) = q_{n+1} \notin F$ while $\hat{\delta}(q_j, b^{i+1}) = q_{j-(i+1)} \in F$). And for $0 \leq i \leq n$, we have $q_i \not\approx q_{n+1}$ with the obvious witness ϵ.

---

6. Consider the context–free grammar:

$$S \rightarrow \epsilon \mid aS \mid Sb$$

(a) Which language does this grammar generate? $\boxed{\text{D,C}}$

Solution: It generates the language $\mathcal{L}(a^*b^*)$.

(b) Prove your answer correct. $\boxed{\text{B,A}}$

Solution: The proof of $S \rightarrow^+_G x \Leftrightarrow x \in \mathcal{L}(a^*b^*)$ is standard, as discussed in class.

---

7. Consider the following language:

$$L \stackrel{\text{def}}{=} \{a^m b^n \mid m \neq n\}$$

over the alphabet $\Sigma = \{a, b\}$.

(a) Refer to the *closure properties* of context–free languages to show that L is context–free. $\boxed{\text{E}}$

Solution: We have $L = A \cdot C \cup C \cdot B$ for languages $A \stackrel{\text{def}}{=} \mathcal{L}(aa^*)$, $B \stackrel{\text{def}}{=} \mathcal{L}(bb^*)$, and $C \stackrel{\text{def}}{=} \{a^n b^n \mid n \geq 0\}$, all of which are context–free. Since CFLs are closed under concatenation and union, L must also be context–free.

(b) Guided by your answer, give a context–free grammar G generating L. $\boxed{\text{E}}$

Solution: Using the constructions used for proving the corresponding closure properties, we easily obtain the grammar:

$$\begin{array}{rcl} S & \rightarrow & S_A S_C \mid S_C S_B \\ S_A & \rightarrow & a \mid aS_A \\ S_B & \rightarrow & b \mid bS_B \\ S_C & \rightarrow & \epsilon \mid aS_C b \end{array}$$

(c) Construct a *deterministic pushdown automaton* (DPDA) that accepts L on *final states.* (Recall that DPDAs rewrite $\perp$ only to strings of shape $\gamma\perp$, so they never halt because of an empty stack.) Draw its graph and explain its workings. E,D

Solution: (Sketch) It is not difficult to come up with a DPDA for this language having 6 control states. The key idea is to push onto the stack a letter A for the first a of the input word, but push a different letter B for all following a's. This allows to detect the b "matching" the first a, and to move to a non-final control state.

(d) Recall the *Chomsky–Schützenberger Theorem* (textbook Supplementary Lecture G). Show how this theorem applies to the above language L, by identifying: D,C

- a suitable natural number n,
- a regular language R over the alphabet Σ_n of the n–th balanced parentheses language PAREN_n, and
- a homomorphism $h : \Sigma_n \to \Sigma^*$,

so that you can argue that $L = h(\mathrm{PAREN}_n \cap R)$ holds.

Solution: Again guided by the decomposition in (a), one can take $n = 3$, regular language $R = \mathcal{L}([_1[_1{}^*]_1{}^*[_2{}^*]_2{}^* + [_2{}^*]_2{}^*[_3{}^*]_3{}^*[_3)$ and homomorphism h defined by $h([_1) = h([_2) = a$, $h(]_2) = h(]_3) = b$, and $h(]_1) = h([_3) = \epsilon$.

---

8. Consider the language: D,C

$$A = \left\{a^l b^m a^n \mid l < m < n\right\}$$

Use the *Pumping Lemma* for context–free languages, as a game with a Deamon, to prove that A is not context–free.

Solution: (Sketch) By picking $z = a^k b^{k+1} a^{k+2}$ it is easy to win the game, by pumping out (i.e. picking $i = 0$) or in (e.g. picking $i = 2$) depending on whether $v \cdot x$ overlaps with the last block (in which case it cannot overlap with the first block) or not, respectively.

---

9. Give a detailed description, preferably as a graph, of a *total Turing machine* accepting the language: E,D

$$A = \left\{a^{n^2} \mid n \geq 0\right\}$$

Explain the underlying algorithm.

Solution: (Sketch) One idea is to procede in rounds, by marking the letters on the tape with single or double dots, so that at the end of round k the tape contents are:

$$\vdash \underbrace{\overbrace{\ddot{a}\ddot{a}\ldots\ddot{a}}^{k}\dot{a}\dot{a}\ldots\dot{a}}_{k^2}aa\ldots a$$

Noticing that $(k+1)^2 = k^2 + 2k + 1$ and that a block of k^2 a's and another one of k a's are readily present after round k, it is not difficult to compute the tape contents needed at the end of round $k+1$, namely:

$$\vdash \underbrace{\overbrace{\ddot{a}\ddot{a}\ldots\ddot{a}}^{k+1}\dot{a}\dot{a}\ldots\dot{a}}_{k^2+2k+1}aa\ldots a$$

The machine accepts if after some completed round all a's are marked, and rejects if all a's are marked before completion of the latest round.

---

10. Show that the problem of whether a Turing machine eventually writes a given letter on its tape for exactly 777 input strings is *undecidable*. Or, in other words, show that the set [B,A]

$$P \stackrel{\text{def}}{=} \left\{ \hat{M}\sharp\hat{a} \mid \text{for 777 inputs, } M \text{ eventually writes } a \text{ on its tape} \right\}$$

is not *recursive*.

Hint: Find a suitable *problem P' on recursively enumerable sets*, for which you:

(a) argue that P' is not *trivial* and hence, by *Rice's Theorem*, is undecidable, and

(b) reduce P' to the original problem P, by describing how from a total TM for P you can build a total TM for P'.

Solution: The bottom–line idea in many problems like this one is to reduce acceptance to the given problem, in this case eventual writing of some letter to the tape.

The problem P' of whether a Turing machine M accepts exactly 777 strings (i.e. whether $|\mathcal{L}(M)| = 777$) is obviously a nontrivial problem on recursively enumerable sets, and hence, by Rice's Theorem, is undecidable. We will reduce this problem to the original problem P above.

Assume M_P is a total TM for P. Construct Turing machine N as follows. On input $\hat{M}$, machine N:

- modifies $\hat{M}$ to $\hat{M}'$ by adding: a new symbol a to the input alphabet of M, a new state t_{new} which is made the accept state of M', and transitions from the original accepting state (of M) to t_{new} that on any tape symbol rewrite this symbol to a, and
- overwrites the input with $\hat{M}'\sharp\hat{a}$, rewinds, and continues as M_P.

Then, we have:

$$\begin{aligned}
N \text{ accepts } \hat{M} &\Leftrightarrow M_P \text{ accepts } \hat{M}'\sharp\hat{a} \\
&\Leftrightarrow \text{for 777 inputs, } M' \text{ eventually writes } a \text{ on its tape} \\
&\Leftrightarrow \text{for 777 inputs, } M \text{ accepts} \\
&\Leftrightarrow |\mathcal{L}(M)| = 777
\end{aligned}$$

Since M_P is total, we obtain that N rejects $\hat{M}$ if $|\mathcal{L}(M)| \neq 777$, and hence N is a total TM deciding problem P'. But this problem is undecidable, and so we arrived at a contradiction. Therefore no total TM for P exists, and hence problem P is undecidable.