1999 CST Part IA Reg. Lang. & Finite Automata, paper 2, q 7

AMP

(i) Assume L regular – so there is a DFA M accepting L. Define a NFA M′ as follows:

- states of M′ are $Q \times \{0,1\}$, where Q = states of M.
- start state of M′ is $(q_0, 0)$, where q_0 = start state of M
- (q,i) is accepting in M′ iff q is accepting in M
- Transitions of M′ are of three types:

(a) $(q,0) \xrightarrow{b} (q',0)$, where $q \xrightarrow{b} q'$ in M

(b) $(q,1) \xrightarrow{b} (q',1)$, where $q \xrightarrow{b} q'$ in M

(c) $(q,0) \xrightarrow{b} (q',1)$, where $q \xrightarrow{\bar{b}} q'$ in M

where $\bar{b} = \begin{cases} 1 & \text{if } b=0 \\ 0 & \text{if } b=1 \end{cases}$

Thus for any transition sequence in M′

$$(q_0,0) \xrightarrow{b_1} \cdots \xrightarrow{b_n} (q,i) \text{ accepting}$$

<u>either</u> $i=0$, the sequence contains no type-(c) transition, $q_0 \xrightarrow{b_1} \cdots \xrightarrow{b_n} q \in Accept_M$ in M and $b_1 \cdots b_n \in L(M)$.

<u>or</u> $i=1$, the sequence contains exactly one type-(c) transition, at i^{th} place say, and $b_1 \cdots b_n$ differs at i^{th} place from a string accepted by M

Thus $L(M') \subseteq L'$, and conversely $L' \subseteq L(M')$ for similar reasons. So $L' = L(M')$ is regular.

Suppose M has ℓ states (so $\ell \geq 1$). If $L(M) \neq \emptyset$, then we can find a string in $L(M)$ of <u>shortest length</u>, $a_1 a_2 \cdots a_n$ say with $n \geq 0$. Thus there is a transition sequence in M

$$q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} q_n$$

with q_0 the start state & q_n accepting.

If $n \geq \ell$, then not all the $n+1$ states $q_0, \ldots, q_n$ can be distinct: in that case choosing the left-most pair $i<j$ of repeated states $q_i = q_j$ we get

$$q_0 \xrightarrow{a_1 \cdots a_i}{}^{*} q_i \overset{a_{i+1} \cdots a_j}{=} q_j \xrightarrow{a_{j+1} \cdots a_n}{}^{*} q_n$$

and hence $a_1 \cdots a_i a_{j+1} \cdots a_n$ is a strictly shorter string in $L(M)$ – contradiction.

So $n < \ell$, i.e. M accepts a string of length

5 $<$ # states.

<u>Kleene's Theorem</u>: The collection of regular languages, i.e. those accepted by some deterministic finite automaton, is the same as the collection of languages determined

2 by regular expressions

Given a regular expression r, by Kleene's Theorem we can construct a DFA M with $L(M) = L(r)$. So it suffices to decide whether $L(M)$ is empty. From above, if $L(M)$ contains any string, it contains one of length less than ℓ = # states of M. So we just have to check each of the finitely many strings over the alphabet of length less than ℓ and see whether they are accepted by M or not.

3

20