York University — EECS 2001 — May 27, 2015

Finite Automata Review Questions

1. Egbert is designing an web interface to access the Leutonian National Library. The web site will require registered users to choose a password. A password is a string of characters that are either Leutonian letters or Leutonian digits. The set of Leutonian letters is $\{\mathtt{k}, \mathtt{z}, \mathtt{c}, \mathtt{v}\}$. (Since they lack vowels, Leutonian words are notoriously difficult for non-native speakers to pronounce.) Leutonians use the same decimal numbering scheme that we do, so their digits are `0` to `9`.

To make passwords harder to guess, Egbert comes up with the following rules for legal passwords.

- A password must contain at least one Leutonian digit.
- The length of the password must be at least 3.
- No letter can appear after a digit.
- A `v` must never appear immediately after a `c`. (This would spell out a very bad Leutonian swear word.)

For example, `kczv78` and `791` are legal passwords, but `5k`, `vvvvv` and `kcvk99` are not.

(a) Design a deterministic finite automaton that accepts a string if and only if it is a legal password. You may assume the input alphabet is the set of all Leutonian letters and digits. Use as few states as possible.

Hint: it is possible to use fewer than 15 states.

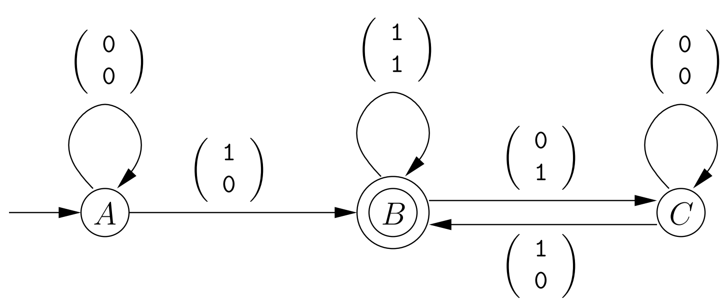
(b) For each state of the automaton you drew in part (a), describe, in English, exactly which strings take the automaton to that state. (You do not have to prove your answer is correct.)

2. In this question, we shall consider a finite automaton that uses the input alphabet

$$\Sigma = \left\{ \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\}.$$

If $x \in \Sigma^*$ is a string, we define $top(x)$ and $bottom(x)$ to be the two numbers represented (in binary) by the top row and the bottom row of bits in x. For example, if $x = \begin{pmatrix} 1 \\ 0 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \end{pmatrix}$, then $top(x) = 43$ (since the top row of x is `101011`, which is the binary representation of 43) and $bottom(x) = 29$ (since the bottom row of x is `011101`, which is the binary representation of 29).

Now, consider the finite automaton shown below. We use the convention that if no transition is shown, the automaton moves to the reject state (not shown) and then stays there forever.



(a) Find four different strings that the finite automaton accepts. For each string x that you find, write down $top(x)$ and $bottom(x)$. At least one of the strings you find should include the character $\begin{pmatrix} \texttt{0} \\ \texttt{1} \end{pmatrix}$.

(b) Fill in the blanks in the following claim with simple statements about the relationship between $top(x)$ and $bottom(x)$.

Claim: For all strings $x \in \Sigma^*$ of length at least 1:

(i) $\delta^*(A, x) = A$ if and only if $top(x) = bottom(x) = 0$.
(ii) $\delta^*(A, x) = B$ if and only if ________________________
(iii) $\delta^*(A, x) = C$ if and only if ________________________

(c) Give a detailed proof of the "only if" part of all three claims in part (b).

Note: it is important that you prove the correct direction. If you prove the "if" direction, you will not get credit for this question.

(d) Complete the following claim with a simple statement about x. For all $x \in \Sigma^*$ of length at least one, the finite automaton accepts x if and only if ________________________. Indicate why your answer follows from your claim in part (b).

3. If L is a language over the alphabet Σ, let $EXTRA(L)$ be the set of all strings obtained by inserting exactly one extra character into any one of the strings in L. More formally, $EXTRA(L) = \{xay : x, y \in \Sigma^* \text{ and } xy \in L \text{ and } a \in \Sigma\}$.

(a) If $\Sigma = \{\texttt{a}, \texttt{b}\}$ and $L_1 = \{\texttt{aa}, \varepsilon, \texttt{b}\}$, what is $EXTRA(L_1)$?

(b) Does there exist a language L_2 such that $EXTRA(L_2) = L_2$? Prove your answer is correct.

(c) Explain why the set of regular languages is closed under the $EXTRA$ operation. (In other words, show that if L is regular, then $EXTRA(L)$ must also be regular.) Your argument should have the same form as the proof of Theorem 1.47 in the textbook: first give a high-level description of your proof idea in English, then give a detailed description of the construction. In addition, you should describe, for each state of your new machine, exactly which strings will take the machine into that state (but you do not need to give a formal proof of this claim).