

CS385-Theory of Computation

First Test

October 16, 2006

1. Consider the NFA

$$\mathcal{M} = (\{0, 1, 2\}, \{a, b\}, \delta, 0, \{2\})$$

defined by

$$\delta(0, a) = \{0, 1\}, \delta(0, b) = \{0\}, \delta(1, b) = \{2\},$$

and

$$\delta(q, \sigma) = \emptyset$$

for all pairs (q, σ) not explicitly listed above.

(a) Draw the state-transition diagram of this NFA.

(b) Give examples of two strings that are in $L(\mathcal{M})$, and two that are not in $L(\mathcal{M})$.

(c) Draw the state-transition diagram of a DFA that recognizes the *complement* of $L(\mathcal{M})$. You may be able to just find this by thinking directly about how to recognize strings from this language in a deterministic fashion, but it is reasonably efficient to apply the subset construction to the NFA above, particularly if you keep in mind that you don't need to include all 8 subsets of $\{0, 1, 2\}$, just those that arise in the course of applying the construction.

(d) Use your DFA from part (c) to find a regular expression for the complement of $L(\mathcal{M})$. Here I definitely do want you to apply the algorithm that you learned for doing this. (It is not hard to come up with a much simpler regular expression for this language just by reasoning about what this set of strings is, and you can do this for partial credit if you're unsure about the algorithm.)

2. Let L be a language over a finite alphabet Σ . Let $\kappa(L)$ be the set of all strings over Σ of the form

$$a_1 b_1 \cdots a_n b_n,$$

where

$$a_1 \cdots a_n \in L.$$

That is, $\kappa(L)$ consists of all the strings of even length such that when you erase the second, fourth, sixth, etc. letters, you get a string in L .

(a) When Σ is $\{a, b\}$ and L is $a^*b(a+b)^*$, what is $\kappa(L)$? You can give either a succinct English description, a regular expression, or an automaton.

(b) Show that whenever L is a regular language, $\kappa(L)$ is regular. There are two distinct strategies for doing this—one is to use a construction with automata (i.e., do something with an automaton that recognizes L to obtain one that recognizes $\kappa(L)$), the other is to modify a regular expression for L . Whichever method you use, you can describe your construction informally in words and pictures rather than formal notation, but you must be completely precise, and give a small example. You do not need to prove that your construction actually works—that is, that the automaton or regular expression you arrive at actually defines $\kappa(L)$ —although of course this will be a part of any real proof.

3. For the examples given below, tell whether the language is regular or nonregular, and justify your assertion: If the language is nonregular, you can prove this using the pumping lemma, or by reduction to a language that we have already shown to be nonregular. If the language is regular, you can give an automaton or a regular expression.

(a)

$$\{ubv : u, v \in \{a, b\}^*, |u| < |v|\}.$$

(b)

$$\{ubv : u, v \in \{a, b\}^*, |u| < |v|\} \cup \{uav : u, v \in \{a, b\}^*, |u| < |v|\}.$$