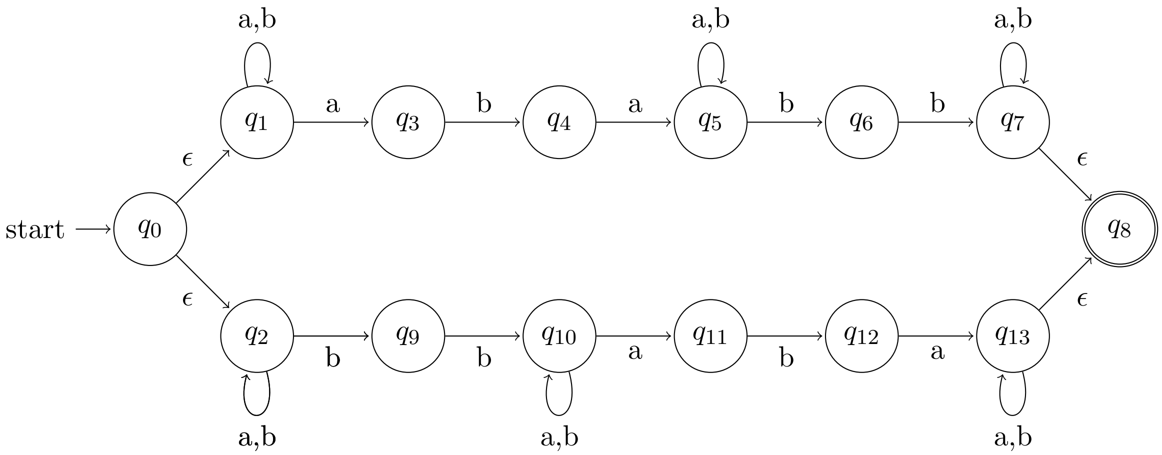
CSE 2001: Introduction to the theory of computation
Assignment 2 (Released Sept 28, 2012)
Solutions

Question 1

Draw a finite automaton that accepts the language of all strings containing *bb* and *aba* as substrings. Note that substrings must be contiguous, so the string *bab* does not contain the word *bb* as a substring.
Solution: The NFA below accepts the language givem.



Question 2

Let $\Sigma = \{a, b\}$. Construct DFA's accepting the following languages:

1. All words not ending in *aab*.

   Solution: We can construct a DFA that accepts the complement of this language – i.e., all strings ending in *aab*, as shown in Figure 1.

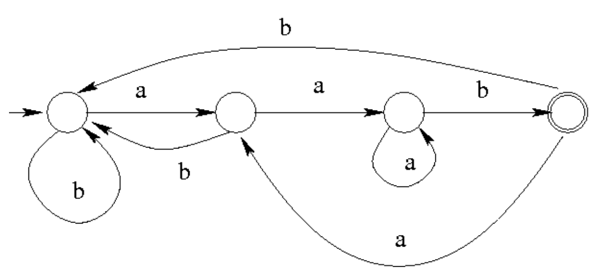
   

   Figure 1: Language that accepts strings ending in *aab*

   Now we simply make the non-accept states accept states and vice versa.

2. Construct a FA that accepts all strings over $\{a, b, c\}$ whose symbols are in alphabetical order. For example, *aaabcc* and *ac* are accepted by the FA, *abca* and *cb* are not.

   Solution: The NFA shown in Figure 2 accepts the language.

   Note: the question had an error - it asked for DFA's, but I mentioned in class that NFA's were acceptable.

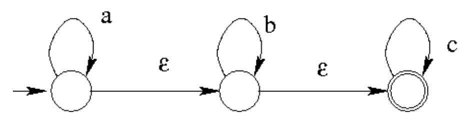


Figure 2: Language that accepts strings in alphabetical order

Question 3

Assume that L_1, L_2 are regular languages. Prove that $L_1 - L_2$ is regular.
Hint: Construct an automaton that accepts $L_1 - L_2$.
Solution: Note that $L_1 - L_2 = L_1 \cap L_2^c = (L_1^c \cup L_2)^c$. So given DFA's D_1, D_2 for L_1, L_2 respectively, we can complement D_1 by swapping the accept and reject states as in part 1 of this question. Then we can form a NFA for $L_1^c \cup L_2$ by using the union construction. Next we convert this to a DFA using the procedure in the text. Finally we complement the DFA as before to get the final DFA.

Question 4

Show that if L is a regular language, the language obtained from L (alphabet Σ) by deleting the last letter in every non-empty word, i.e., the following, is regular.

$$L' = \{w | w\sigma \in L, \sigma \in \Sigma\}$$

Write down a short intuitive argument to show that your answer is correct.
Solution: Consider a DFA for language L. For each accept state s, look at all nodes j such that there are transitions from j to s. Make each such j an accept state and s no longer an accept state. The proof that this works is as follows. Suppose DFA M accepts L and DFA M' accepts L'. For each accept state $s' \in M'$, by construction of M', there is some σ for which $\delta(s', \sigma) = s$ such that s is an accept state of M. This implies that M accepts $w\sigma$ if and only if M' accepts w.

Question 5

Show that if L is a regular language, the language obtained from L (alphabet Σ) by adding a single character σ to each word, i.e., the following, is regular.

$$L' = \{w_1 \sigma w_2 | w_1 w_2 \in L, \sigma \in \Sigma\}$$

Write down a short intuitive argument to show that your answer is correct.
Solution:

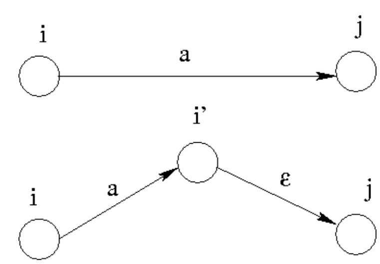


Figure 3: Q5, incorrect attempt

First, an incorrect attempt. As Figure 3 shows, we can replace each edge by two edges (and a new node in each case). This would add new characters σ to existing strings. However, the new language now allows many characters to be added to each string. So we need a new idea.
The procedure for designing the DFA is as follows. Let D be a DFA that accepts L. Then have two copies of D – call them D_1, D_2. The start state of D_1 is the start state of the new automaton and the accept state(s) of D_2 are its accept states. From each node in D_1 add transitions for each $\sigma \in \Sigma$ to its corresponding state in D_2. So given a string $w_1 \sigma w_2$, the automaton follows D_1 on w_1, uses σ to transfer to the same state in D_2 and then proceeds as usual on w_2 in D_2.
The intuitive justification is that the transfer is made on the extra character σ and it can make only one transition between D_1, D_2. The rest of the input is used to travel on D_1, D_2 which are copies of D, so the new automaton accept $w_1 \sigma w_2$ if and only if D accepts $w_1 w_2$.