

CS 273, Fall 2007

Exam 1 Solutions

Problem 1: Short Answer (12 points, about 10 min)

The answers to these problems should be short and not complicated.

(a) List the elements of $\{2, 3\} \times \{a, b\}$.

$\{(2, a), (2, b), (3, a), (3, b)\}$. Given the phrasing of the question, the set brackets are optional. However, it's important that the individual elements (e.g. $(3, a)$) are pairs, not sets.

(b) We've seen six operations that regular languages are closed under. List five of them. (Short names are sufficient.)

Any five out of: union, intersection, set complement, string reversal, star, concatenation. (Set subtraction is also an acceptable answer.)

(c) $\mathbb{P}(\emptyset) = \{\emptyset\}$

(d) Is it true that any DFA can be modified to create an equivalent DFA with only a single accept state?

No. We showed this only for NFAs. Our construction used ϵ -transitions to connect the old final states to a new single final state. DFAs don't have ϵ -transitions, so this construction won't work and, in fact, you can find DFAs that can't be restructured to have only one final state.

(e) Suppose you have a DFA $M = (Q, \Sigma, \delta, q_0, F)$ recognizing a language L . Explain how to build a DFA M' which recognizes \overline{L} (i.e. the set complement of L). One sentence should be sufficient.

Switch all the accepting states to be non-accepting, and all the non-accepting states to be accepting.

(f) Is $(ab^* + ba^*)^* = (a + b)^*$?

If your answer is NO, give a string that is in one language but not in the other.

Yes. The righthand expression generates all strings of a's and b's. Notice that one option for the star operator is to generate the empty string. So the lefthand side can generate anything of the form $(a\epsilon + b\epsilon)^*$, but this is just $(a + b)^*$.

Problem 2: DFA design (8 points, about 12 min)

Let $\Sigma = \{a, b\}$. Let L be the set of strings that *do not* have the word aab as a (contiguous) substring

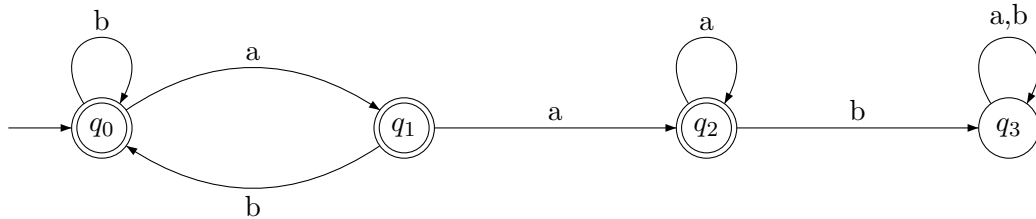
$$L = \{w \mid w \neq w_1 aab w_2 \text{ for any } w_1, w_2 \in \Sigma^*\}$$

For example, $baa \in L$ and $abab \in L$, but $aab \notin L$, $babaabab \notin L$, and $ababaab \notin L$. Note that L contains all strings shorter than 3 characters.

Construct a DFA that accepts L and give a state diagram showing **all** states in the DFA. Your DFA should not contain more than 10 states.

Explain briefly how your DFA works. You do not need to prove formally that your DFA is correct. *You will receive **zero** credit if your DFA uses more than **10** states or makes significant use of non-determinism.*

Notice that the DFAs for a language and its set complement are the same, except for flipping the accept/reject state markings. So one easy solution to this problem is to design a DFA which accepts strings containing aab , and then reverse the accept/reject markings.



The rightmost state of this machine is a “fail” state. Strings reach the fail state if they contain aab . Until/unless they reach the fail state, they are accepted.

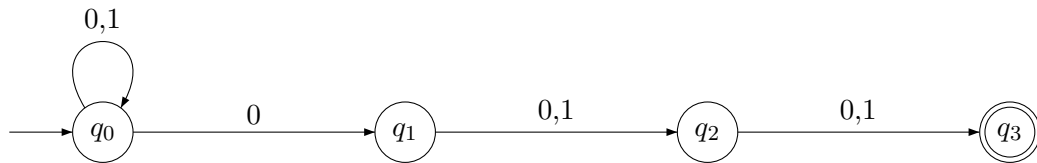
Problem 3: NFA design (8 points, about 10 min)

Let $\Sigma = \{0, 1\}$. Let L be the language containing all strings over Σ that have length at least three and have a 0 in the third position from the end.

That is, $L = \{w_1 0 c_1 c_2 \mid w_1 \in \Sigma^*; c_1, c_2 \in \Sigma\}$

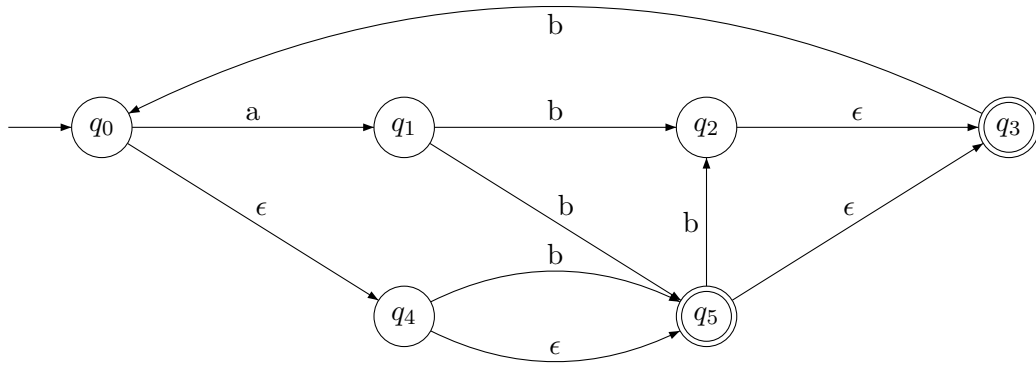
For example, L contains 001000 and 11011, but L does not contain 00 nor 0100.

Give the state diagram of an NFA that recognizes L and has no more than 6 states. Full credit requires a machine that isn't overly complex and that exploits the extra features of NFAs. Don't solve this with a DFA.



Problem 4: NFA transitions (8 points, about 8 min)

Suppose that the NFA $N = (Q, \{a, b\}, \delta, q_0, F)$ is defined by the following state diagram:



Fill in the following values:

- (a) $\delta(q_0, a) = \{q_1\}$
- (b) $\delta(q_1, b) = \{q_2, q_5\}$
- (c) $\delta(q_4, a) = \emptyset$
- (d) $\delta(q_2, b) = \emptyset$
- (e) $F = \{q_3, q_5\}$

If S is any set of states, recall that $E(S)$, the ϵ -closure of S , is defined as

$$E(S) = \{q \in Q \mid \text{there is a state } r \in S \text{ such that } q \text{ is reachable from } r \text{ using zero or more } \epsilon \text{ transitions}\}$$

Fill in the values of the following expressions:

- (f) $E(\{q_0\}) = \{q_0, q_4, q_5, q_3\}$
- (g) $E(\{q_1, q_2\}) = \{q_1, q_2, q_3\}$

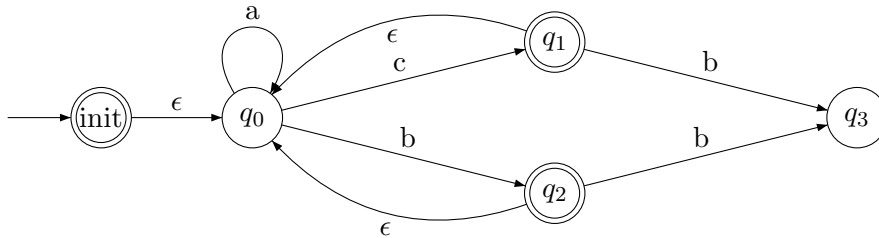
Regarding the language accepted by the NFA:

- (h) Does the NFA accept the word bb ? Yes. Go to q_3 using entirely ϵ -transitions. Take the b arc back to q_0 . Then go to the final state q_5 using an ϵ -transition and a transition on b .

Problem 5: Short Answer II (6 points, about 8 min)

The answers to these problems should be short and not complicated.

(a) Here is the state diagram for an NFA recognizing some language L . Add extra states and/or transitions to this diagram, to produce the state diagram for an NFA recognizing L^* . (Do not remove states or transitions from the input NFA.)



The extra initial state is important, because it allows us to accept the empty string without also accepting strings like aaa.

(b) Suppose you have an NFA $M = (Q, \Sigma, \delta, q_0, F)$ which contains no ϵ transitions. We saw how to simulate M with a DFA $M' = (\mathbb{P}(Q), \Sigma, \delta', q'_0, F')$. Suppose that S is a state of M' (i.e. $S \subseteq Q$), and $a \in \Sigma$. Then

$$\delta'(S, a) = \bigcup_{q \in S} \delta(q, a)$$

Problem 6: NFA modification and tuple notation (8 points, about 12 min)

Given a DFA M that accepts the language L , design a new NFA M' that accepts the language $L^R = \{w \mid w^R \in L\}$, where w^R is the reversed version of the string w . For example, if $w = aab$, then $w^R = baa$.

(a) Briefly explain the idea behind your construction, using English and/or pictures.

Make M have a single accept state, by adding a new accept state q_{acc} and making epsilon transitions from the old accept states to q_{acc} . Then reverse all the transitions in M , swapping the start and accept states.

(b) Suppose that $M = (Q, \Sigma, \delta, q_0, F)$. Give the details of your construction of M' , using tuple notation.

$$M' = (Q \cup \{q_{acc}\}, \Sigma, \delta', q_{acc}, \{q_0\}).$$

The transition function δ' is defined as follows.

$$\delta'(q_{acc}, \epsilon) = F$$

$$\delta'(q, a) = \{r \in Q \mid \delta(r, a) = q\}, \text{ for every } q \in Q$$

$$\delta'(q, a) = \emptyset, \text{ for all other values of } q \text{ and } a$$

Notice that M' cannot be a DFA for two reasons. First, we needed ϵ -transitions to create a single accept state, which could be made into the initial state of the reversed automaton. Second, the output of δ' sometimes needs to contain more than one state, because a state q might have several incoming transitions on the same character a .