CSE 105: Introduction to the Theory of Computation Winter 2000

Problem Set 2 Solution

Instructor: Daniele Micciancio Jan. 24, 2000

In this homework set, we always use the alphabet $\Sigma = \{a, b\}$.

Problem 1

Use the construction shown in class (or other equivalent construction) to convert the following NFA N to DFA. Give your answer as a state transition diagram. (You do not need to include in your answer the states that cannot be reached from the start state). **Solution:** The DFA obtained applying the construction in the book has 16

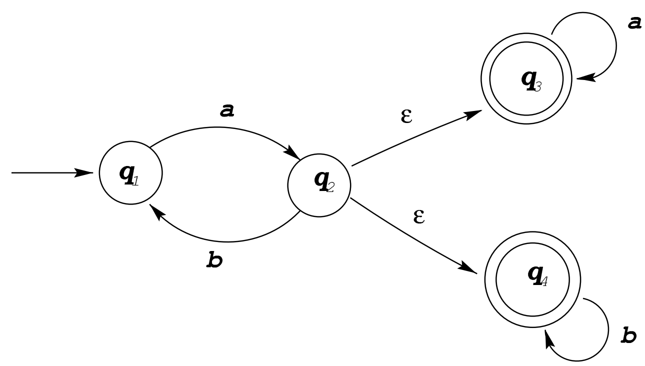

Figure 1: A nondeterministic finite automaton N

states, but some of them cannot be reached from the start state on any input. The following diagram shows only the states that can be reached from the start state:

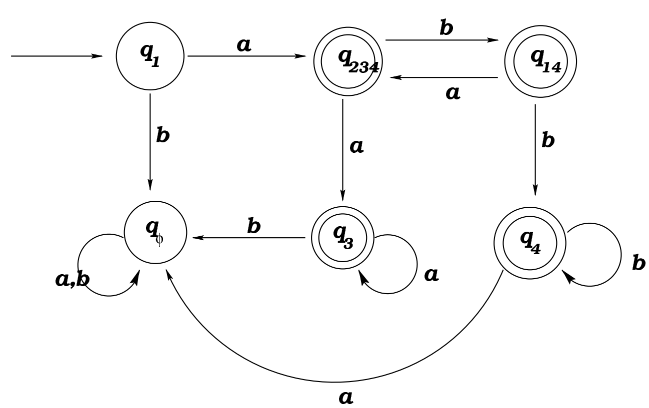

Figure 2: The deterministic finite automaton of N

Problem 2

Use the method discussed in class (or equivalent method) to find NFAs that accept the the languages corresponding to the following regular expressions. Show intermediate steps in the construction. (You can omit a few of the minor steps.)

a. $(aaa)^* \cup b(ab)^*$

Solution: Here we show only the final result. If you are not sure about the intermediate steps, please consult the book or your TA/instructor. The final NFA is given by:

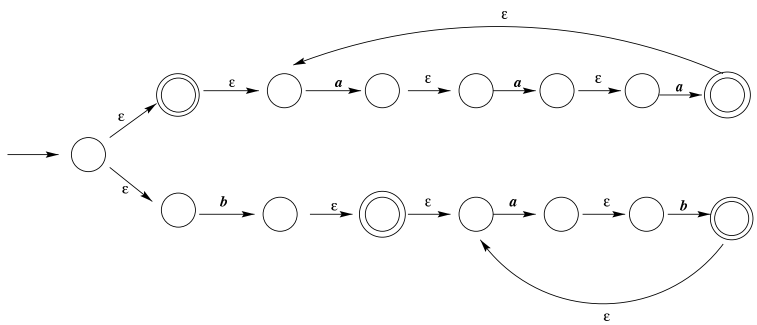

Figure 3: The nondeterministic finite automaton accepting $(ab \cup ba)^*$.

b. $(ab \cup ba)^*$

Solution: The final NFA is given by:

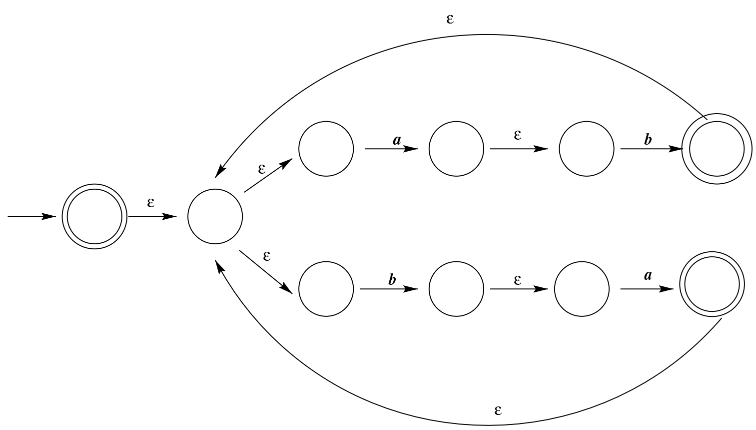

Figure 4: The nondeterministic finite automaton accepting $(ab \cup ba)^*$.

Problem 3

Use the method discussed in class (or the method from the book) to find a regular expression that describes the language accepted by the NFA from problem 1. Show the intermediate steps of the computation. (You can simplify the intermediate expressions as well as the final answer using simple identities like $\epsilon \cdot a = a$ and $a^* \cup a = a^*$.)

Solution: We first add a new start state q_0 with ϵ transition to q_1, and a single accepting state q_5 with ϵ transitions from q_3 and q_4. Then, if we remove states q_1, q_2, q_3 and q_4 one after the other, we get a GNFA with a single transition from q_0 to q_5 with label:

$$a(ba)^*a^* \cup a(ba)^*b^*$$

This regular expression is equivalent to the original NFA from Problem 1.

Problem 4

Prove that for every regular expression R there exists another regular expression R' such that the language recognized by R' is the complement of the language recognized by R, i.e., $L(R') = \Sigma^* - L(R)$. (You can use the results proved in class and in the first problem set to solve this problem.)

Proof First notice that given a regular expression R, one can build a NFA N recognizing the same language (see Lemma 1.29 in the book). This automaton, can be transformed into an equivalent DFA M (See theorem 1.19). At this point we have a DFA M recognizing the language $L(R)$, and we want to transform it into a DFA (or NFA) recognizing the complement of $L(R)$, i.e. $\Sigma^* - L(R)$. Notice that the language Σ^* is regular because it is accepted by the DFA $U = (\{q\}, q, \{q\}, \Sigma, \delta : (q, x) \mapsto q)$ with a single (accepting) state q that loops on every input symbol. Now you can use the construction from Problem 3 Problem Set 1, to build a DFA accepting the language $\Sigma^* - L(R)$. Notice that this construction is equivalent to changing the set of accepting states on M from F to $Q - F$ (i.e., we are exchanging final and non-final states). This automaton M' recognizes the regular language $\bar{L}$ which is the complement of $L(R)$. Finally we can use the construction from Lemma 1.32 in the book to transform D' into an equivalent regular expression R' that correspond to the complement of the language $L(R)$.

Give regular expressions corresponding to the complement of the following regular expressions:

a. $(a \cup ab)^*$

Solution: First we find the NFA and then convert it to DFA.

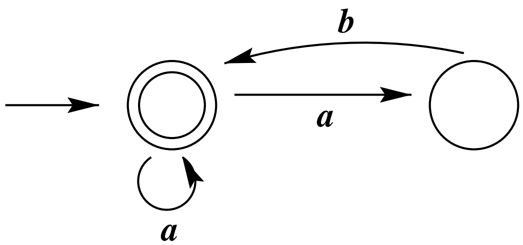

Figure 5: NFA accepting $(a \cup ab)^*$.

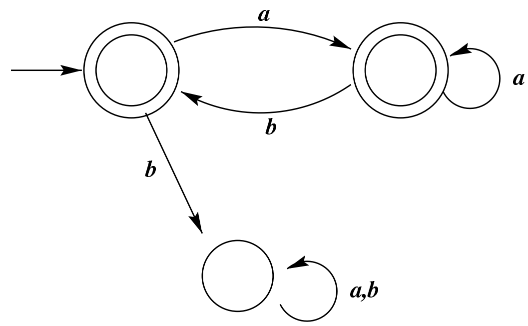

Figure 6: DFA accepting $(a \cup ab)^*$.

By changing final states, we get the DFA for $\overline{(a \cup ab)^*}$.

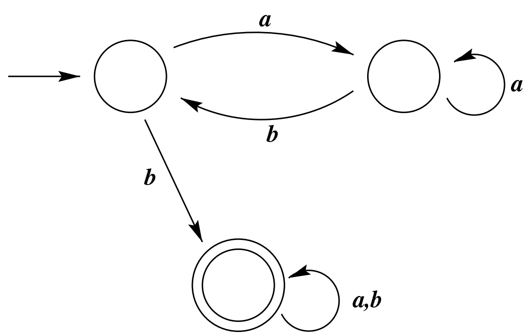

Figure 7: DFA accepting $\overline{(ab \cup ba)^*}$.

The regular expression for language $\overline{(ab \cup ba)^*}$ is

$$(aa^*b)^*b(a \cup b)^*$$

b. $(aa)^* \cup b^*$

Solution: First we find the NFA and then convert it to DFA.

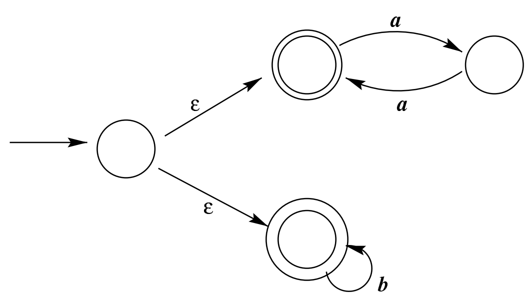

Figure 8: NFA accepting $(aa)^* \cup b^*$.

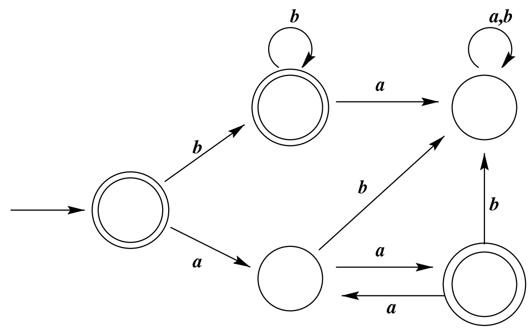

Figure 9: DFA accepting $(aa)^* \cup b^*$.

By changing final states, we get the DFA for $\overline{(aa)^* \cup b^*}$.

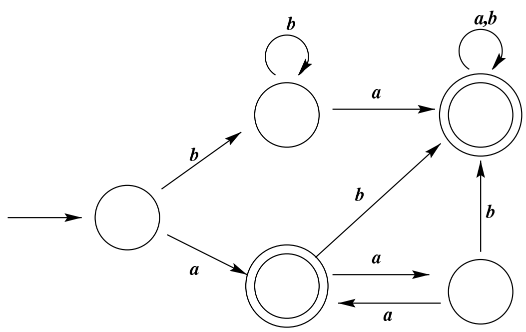

Figure 10: DFA accepting $\overline{(aa)^* \cup b^*}$.

The regular expression for language $\overline{(aa)^* \cup b^*}$ is

$$a(aa)^* \cup (bb^*a \cup a(aa)^*(b \cup ab))(a \cup b)^*$$