Solutions to Homework 3
CS 770: Formal Language Theory

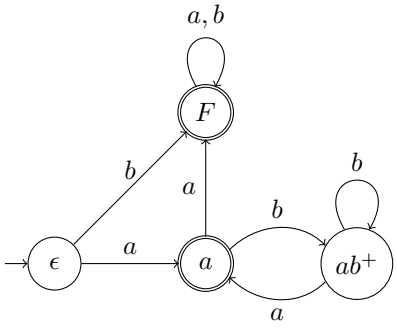
Problem 1. [Category: Comprehension+Design] Consider the language $L = \overline{\mathbf{L}((abb^*)^*)}$.

1. Construct a DFA recognizing L. You need not prove that your construction is correct. If you construct it using the algorithms described in class then you should show all your steps. If you construct the automaton directly then you should explain the intuition behind your construction clearly. **[5 points]**
2. Construct a regular expression for the language L. Again you don't need to prove your regular expression to be correct, but you should show all the steps in the construction. **[5 points]**

Solution:

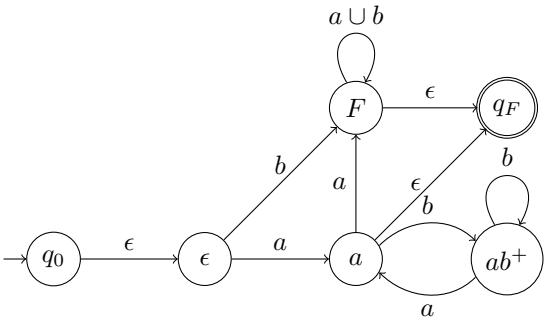
1. For this part, it is very useful to come up with the smallest DFA recognizing L, so that the solution in the next part is easy. One possibility is to start from the regular expression $(abb^*)^*$, construct an NFA for it, convert that NFA to a DFA, and then complement that and "minimize" it. However, we instead will find it convenient to construct a DFA for it directly.

   In order to construct the DFA for L, it is convenient to think about designing a DFA for $(abb^*)^*$ and then "complementing" it. To check if a string follows the pattern $(abb^*)^*$ we will have a state that remembers that we have seen the first a, and state that remembers that we have seen an a followed by at least one b. In addition, we will have an initial state, and a state we go to whenever we discover that the string does not have the pattern $(abb^*)^*$. We pick the final states based on the fact that we want to recognize L. Putting it all together we get the DFA shown below.

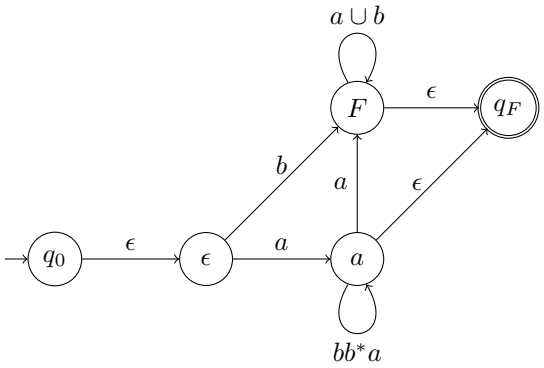


2. To construct a regular expression for the language L, we will use the algorithm described in class. We will convert the DFA above to a GNFA, and then remove one state at a time until we get a two state GNFA. The series of steps is shown below. In the pictures below, we do not draw the transitions labelled $\emptyset$ to avoid clutter.

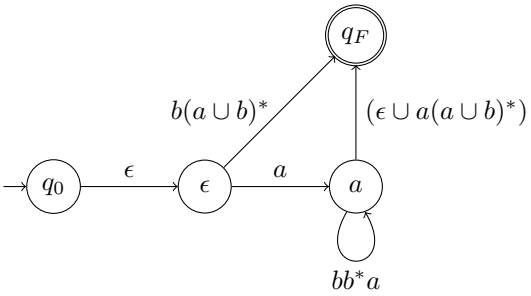
   The regular expression for L is $R = b(a \cup b)^* \cup a(bb^*a)^*(\epsilon \cup a(a \cup b)^*)$.



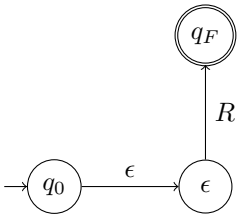
Step 1: Initial GNFA



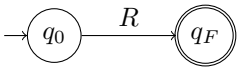
Step 2: GNFA after ripping ab^+



Step 3: GNFA after ripping F



Step 4: GNFA after ripping a



Step 5: GNFA after ripping ϵ

Figure 1: In step 4 and 5, $R = b(a \cup b)^* \cup a(bb^*a)^*(\epsilon \cup a(a \cup b)^*)$. R is the regular expression describing L.

■

Problem 2. [Category: Comprehension+Design+Proof] For a language $A \subseteq \Sigma^*$ define

$$\text{left}(A) = \{w \in \Sigma^* \mid ww^R \in A\}$$

where w^R denotes the reverse of w.

1. Taking $A = \{\epsilon, 01, 10, 1001\}$, what is left$(A)$? **[1 points]**
2. Taking $A = \mathbf{L}(0^*110^*)$, what is left(A)? **[1 points]**
3. Prove that if A is regular then left(A) is regular. You can either construct a DFA/NFA/regular expression for left(A) (and then you don't have to prove that your construction is correct) or use previously established closure properties to prove this result. *Hint:* Look at the construction of halving a language in lecture 3 (starting from page 26). **[8 points]**

Solution:

1. left$(A) = \{\epsilon, 10\}$

2. $\text{left}(A) = \mathbf{L}(0^*1)$

3. Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA recognizing A. In order to check if an input w belongs to left(A), we need to check if the input ww^R is accepted by M. We will do this by simultaneously running simultaneously M from the initial state and M in reverse from some final state. Thus, the states of the new machine would be a pair of states — one that is the state of M in the forward simulation and the second that is the state of M in the backward simulation. The new machine accepts if after readin the entire input both the forward and backward simulation are at the same state.

   The formal definition based on the above intuition is as follows. Consider machine $M' = (Q', \Sigma, \delta', q_0', F')$ where

   - $Q' = (Q \times Q) \cup \{q_0'\}$
   - $F' = \{(q, q) \mid q \in Q\}$
   - The transition function δ' is given as follows

$$\delta'(q', a) = \begin{cases} \{q_0\} \times F & \text{if } q' = q_0' \text{ and } a = \epsilon \\ \{(q_1', q_2') \mid \delta(q_1, a) = q_1' \text{ and } \delta(q_2', a) = q_2\} & \text{if } q' = (q_1, q_2) \text{ and } a \in \Sigma \\ \emptyset & \text{otherwise} \end{cases}$$

   The correctness (which was not required to be proved) can be established by proving the following statement by induction on the length of w

$$\forall w. \forall (q_1, q_2) \in Q \times Q.\ q_0' \xrightarrow{w}_{M'} (q_1, q_2) \text{ iff } q_0 \xrightarrow{w}_M q_1 \text{ and } q_2 \xrightarrow{w^R}_M q \text{ for some } q \in F$$

■

Problem 3. [Category: Proof] Let $C = \{1^k x \mid x \in \{0,1\}^*,\ k \geq 1,$ and x contains at most k 1s$\}$. Using the pumping lemma, prove that C is not regular. **[10 points]**

Solution: Let (for contradiction) C be a regular language with p as the pumping lemma constant. Consider the string $w = 1^p 01^p$. Observe that $w \in C$. Let x, y, z be any division of w such that $w = xyz$, $|y| > 0$ and $|xy| \leq p$. Thus, we can assume that $x = 1^i$, $y = 1^j$, and $z = 1^k 01^p$, where $i + j + k = p$, and $j > 0$. Consider $w' = xy^0 z = 1^{i+k} 01^p$. Since w' has $< p$ 1s before the first 0, and so the number of 1s after the first 0 cannot be more that $p - 1$. But w' has p 1s after the first 0 and so $w' \notin C$. Thus, C does not satisfy the pumping lemma and hence is not regular. ■

Problem 4. [Category: Comprehension+Proof] Consider the language $F = \{a^i b^j c^k \mid i, j, k \geq 0$ and if $i = 1$ then $j = k\}$

1. Prove that F is not regular. **[5 points]**

2. Prove that F satisfies the pumping lemma. *Hint:* Take the pumping length to be $p = 3$ and show that F satisfies the pumping lemma for this length. **[5 points]**

Solution:

1. Consider $A = F \cap L(ab^*c^*) = \{ab^n c^n \mid n \geq 0\}$. Define $h : \{a, b, c\}^* \to \{0, 1\}^*$ where $h(a) = \epsilon$, $h(b) = 0$ and $h(c) = 1$. Then, $h(A) = \{0^n 1^n \mid n \geq 0\} = L_{0n1n}$, which is known to be not regular. Thus, F is not regular as L_{0n1n} was obtained from F by applying a series of regularity preserving operations.

2. Take the pumping length $p = 3$. Consider any $w = a^i b^j c^k \in F$, such that $|w| \geq p$.

   If $i \neq 2$, then divide w as follows: Take $x = \epsilon$, y to be the first symbol in w, and z to be the rest of the string. Now, $xyz = w$, $|xy| < 3$ and $|y| > 0$. Observe that the string $xy^t z$, when $t \neq 1$, has the property that the number of as is not 1, and hence $xy^t z \in L$ for any t.

   If $i = 2$, then divide w as follows: Take $x = aa$, y to be the first symbol after that, and z to be the rest of the string. Again, $w = xyz$, $|xy| \leq 3$, and $|y| > 0$. Further, for any t, $xy^t z$ has 2 leading as, and so belongs to F trivially.

   This problem highlights that there are non-regular languages (namely, F) that satisfy the pumping lemma. The pumping lemma says that every regular language satisfies the conditions in the pumping lemma, but does not say that *only* regular languages satisfy the pumping lemma.

■