Solutions for Problem Set 1
CIS 770: Formal Language Theory

Problem 1. [Category: Design] Design a DFA for the language $L_{A1} = \{w \in \{a,b\}^* |$ number of a's in w is not divisible by $3\}$.

Solution: The DFA recognizing L_{A1} will remember how many a's modulo 3 it has seen so far in the input. The states will be $0, 1, 2$, where state i denotes that number of a's modulo 3 seen so far is i. Based on this intuition, the DFA is shown in Figure 1. ■

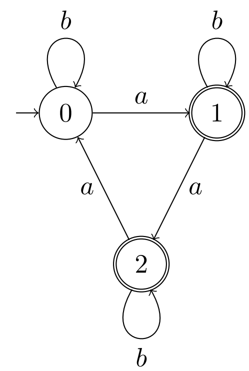


Figure 1: DFA for language L_{A1}.

Problem 2. [Category: Design] Design a DFA for the language $L_{A3} = \{w \in \{a,b\}^* \mid$ if w starts with an a then it does not end with a $b\}$.

Solution: The DFA will remember what the first symbol of the input is, and if the first symbol is an a, it will also remember the last symbol read. Thus the states will be ϵ (initial state), b (input began with b), aa (input began with a and the last symbol read is a), and ab (input began with a and the last symbol read is b). Based on this intuition the transition diagram of the DFA is shown in Figure 2. ■

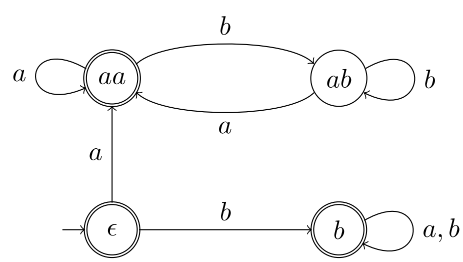


Figure 2: DFA for language L_{A3}.

Problem 3. [Category: Design] Design a DFA for the language $L_{A4} = \{w \in \{a,b\}^* \mid ba$ appears exactly twice as a substring}.

Solution: The DFA will remember how many times the substring ba has appeared and what the last symbol read is; remembering the last symbol read will help the DFA recognize whether it has seen a ba such string. The states are of the form ia (i ba substrings and ends in a) or ib (i ba substrings and ends in b), where i is $0, 1$, or 2. In addition, we will have a "dead state" D, where we remember that we have seen ba more than twice. The DFA is shown in Figure 3. ■

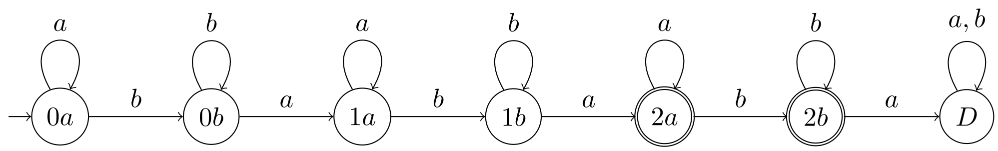


Figure 3: DFA for language L_{A4}.

Problem 4. [Category: Design+Proof] Let $A_k \subseteq \{a, b\}^*$ be the collection of strings w where there is a position i in w such that the symbol at position i (in w) is a, and the symbol at position $i + k$ is b. For example, consider A_2 (when $k = 2$). $baab \in A_2$ because the second position ($i = 2$) has an a and the fourth position has a b. On the other hand, $bb \notin A_2$ (because there are no as) and $aba \notin A_2$ (because none of the as are followed by a b 2 positions away).

1. Design a DFA for language A_k. Your formal description (by listing states, transitions, etc. and not "drawing the DFA") will depend on the parameter k but should work no matter what k is; see lecture 2, last page for such an example. **[5 points]**
2. Prove that your DFA is correct when $k = 2$. **[5 points]**

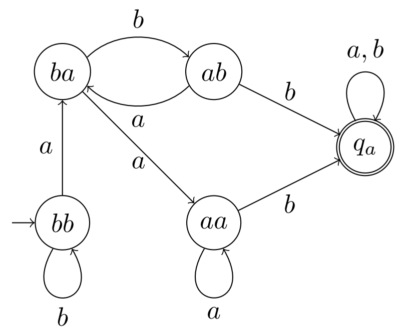
Solution:

1. The DFA for A_k will remember the last k symbols read from the input. When a b is read, if the symbol k positions before was an a then the DFA will move to an accept state, where it will stay no matter what the remaining symbols in the input are; this is because once we find a pair of a and b that are k positions apart, the input is in the language no matter what the other symbols are. This intuition is formalized in the following construct of a DFA $M_k = (Q_k, \{a, b\}, \delta_k, q_k, F_k)$ where

   - The set of states $Q_k = \{a, b\}^k \cup \{q_a\}$. That is, a state either remembers the last k symbols read (is a member of $\{a, b\}^k$) or is the state q_a which remembers that the input must be accepted.
   - The initial state is $q_k = bb \cdots b = b^k$
   - The set of final states is $F_k = \{q_a\}$
   - The transition function δ_k is given by

   $$\delta_k(q, c) = \begin{cases} w_2 w_3 \cdots w_k c & \text{if } q = w_1 w_2 \cdots w_k \text{ and either } c = a \text{ or } w_1 \neq a \\ q_a & \text{if } q = a w_2 \cdots w_k \text{ and } c = b \\ q_a & \text{if } q = q_a \end{cases}$$

   Thus, the DFA M_k has $2^k + 1$ states.



2. When $k = 2$, the automaton M_2 can be drawn as follows. Given that the initial state is bb and the unique accept state is q_a, to prove correctness we need to show that

$$\forall w \in \{a,b\}^*.\ bb \xrightarrow{w}_{M_2} q_a \text{ iff } w \in A_2$$

However, as in other examples we have seen in the lecture notes, this statement needs to be strengthened if the standard proof by induction (on the length of w) is to succeed. The way to strengthen this proof is by characterizing the collection of strings that are accepted from *each* state, and not just the initial state.

Thus what we will prove by induction on the length of w is the following (stronger) statement.

$$\begin{array}{ll} \forall w \in \{a,b\}^*. & bb \xrightarrow{w}_{M_2} q_a \text{ iff } w \in A_2 \\ & ba \xrightarrow{w}_{M_2} q_a \text{ iff } baw \in A_2 \\ & ab \xrightarrow{w}_{M_2} q_a \text{ iff } abw \in A_2 \\ & aa \xrightarrow{w}_{M_2} q_a \text{ iff } aaw \in A_2 \\ & q_a \xrightarrow{w}_{M_2} q_a \text{ iff } w \in \{a,b\}^* \end{array}$$

We will prove the above statement by induction on the length of w.

Base Case: When $w = \epsilon$, $q \xrightarrow{w}_{M_2} q$ for each $q \in Q_2$. Also, $\epsilon \notin A_2$, $ba\epsilon \notin A_2$, $ab\epsilon \notin A_2$, and $aa\epsilon \notin A$. Thus, when $w = \epsilon$, we have established each of the 5 conditions above.

Ind. Hyp.: We will assume that the statement we are trying to prove holds for all w, such that $|w| < k$. That is, we will assume that

$$\begin{array}{ll} \forall w \in \{a,b\}^*.|w| < k, & bb \xrightarrow{w}_{M_2} q_a \text{ iff } w \in A_2 \\ & ba \xrightarrow{w}_{M_2} q_a \text{ iff } baw \in A_2 \\ & ab \xrightarrow{w}_{M_2} q_a \text{ iff } abw \in A_2 \\ & aa \xrightarrow{w}_{M_2} q_a \text{ iff } aaw \in A_2 \\ & q_a \xrightarrow{w}_{M_2} q_a \text{ iff } w \in \{a,b\}^* \end{array}$$

Ind. Step: Consider w such that $|w| = k$. Now, w can be in one of two forms: either $w = au$, or $w = bu$, where $|u| = k - 1$. We will consider the various cases, and show that the correctness statement we are trying to prove holds in the induction step.

- **Case** bb**:** First consider $w = au$. Then we have $bb \overset{w=au}{\longrightarrow}_{M_2} q_a$ iff $ba \xrightarrow{u}_{M_2} q_a$ (because $\delta_2(bb, a) = ba$) iff $bau \in A_2$ (ind. hyp.) iff $au \in A_2$ (definition of A_2) iff $w \in A_2$. Similarly, if $w = bu$ then $bb \overset{w=bu}{\longrightarrow}_{M_2} q_a$ iff $bb \xrightarrow{u}_{M_2} q_a$ (because $\delta_2(bb, b) = bb$) iff $u \in A_2$ (ind. hyp.) iff $bu \in A_2$ (definition of A_2) iff $w \in A_2$.

- **Case** ba**:** When $w = au$, we have $ba \xrightarrow{w=au}_{M_2} q_a$ iff $aa \xrightarrow{u}_{M_2} q_a$ (because $\delta_2(ba, a) = aa$) iff $aau \in A_2$ (ind. hyp.) iff $ba(au) \in A_2$ (definition of A_2) iff $baw \in A_2$. Similarly, if $w = bu$ then $ba \xrightarrow{w=bu}_{M_2} q_a$ iff $ab \xrightarrow{u}_{M_2} q_a$ (because $\delta_2(ba, b) = ab$) iff $abu \in A_2$ (ind. hyp.) iff $ba(bu) \in A_2$ (definition of A_2) iff $baw \in A_2$.
- **Case** ab**:** When $w = au$, we have $ab \xrightarrow{w=au}_{M_2} q_a$ iff $ba \xrightarrow{u}_{M_2} q_a$ (because $\delta_2(ab, a) = ba$) iff $bau \in A_2$ (ind. hyp.) iff $ab(au) \in A_2$ (definition of A_2) iff $abw \in A_2$. Similarly, if $w = bu$ then $ab \xrightarrow{w=bu}_{M_2} q_a$ iff $q_a \xrightarrow{u}_{M_2} q_a$ (because $\delta_2(ab, b) = q_a$) iff $u \in \{a, b\}^*$ (ind. hyp.) iff $ab(bu) \in A_2$ (definition of A_2) iff $abw \in A_2$.
- **Case** aa**:** When $w = au$, we have $aa \xrightarrow{w=au}_{M_2} q_a$ iff $aa \xrightarrow{u}_{M_2} q_a$ (because $\delta_2(aa, a) = aa$) iff $aau \in A_2$ (ind. hyp.) iff $aa(au) \in A_2$ (definition of A_2) iff $aaw \in A_2$. Similarly, if $w = bu$ then $aa \xrightarrow{w=bu}_{M_2} q_a$ iff $q_a \xrightarrow{u}_{M_2} q_a$ (because $\delta_2(aa, b) = q_a$) iff $u \in \{a, b\}^*$ (ind. hyp.) iff $aa(bu) \in A_2$ (definition of A_2) iff $aaw \in A_2$.
- **Case** q_a**:** Consider $w = cu$, where $c = a$ or $c = b$. We have $q_a \xrightarrow{w=cu}_{M_2} q_a$ iff $q_a \xrightarrow{u}_{M_2} q_a$ (because $\delta_2(q_a, a/b) = q_a$) iff $u \in \{a, b\}^*$ (ind. hyp.) iff $(cu) \in \{a, b\}^*$ iff $w \in \{a, b\}^*$.

Thus the correctness has been established by induction.

■

Problem 5. [Category: Design] Design a DFA for the language $L_{A2} = \{w \in \{a, b\}^* |$ number of a's in w is at least 2 or the number of bs is at most $1\}$.

Solution: The DFA will count the number of as seen (either 0, 1 or at least 2) and the number of bs seen (either 0, 1 or at least 2). Thus states of the DFA will be of the form ij, where i is the number of as seen and j is the number of bs seen. Observe, that once we have seen at least 2 as, we don't need to keep track of the number of bs because we should accept no matter what the rest of the string is. Thus, we won't have separate states 20, 21 and 22, but have only one state +2 that just remembers that we have seen at least 2 as. The DFA based on this intuition is shown in Figure 4. ■

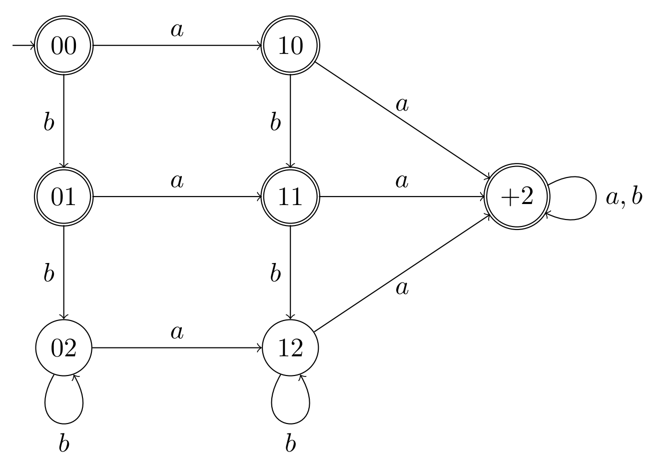


Figure 4: DFA for language L_{A2}.

Problem 6. [Category: Design] Design a DFA for the language $L_B = \{w \in \{a, b\}^* \mid w$ has at least 2 as and ends with $ab\}$

Solution: The DFA will count the number of as to check that there are at least 2 as (states 0 and 1 remembering that 0 as and 1 a have been observed), and if there are at least 2 as, the automaton will remember whether the string ends in an a (state A), ends in an bb (state B), or ends in an ab (state C). Based on this intuition the DFA transition diagram is shown in Figure 5. ■

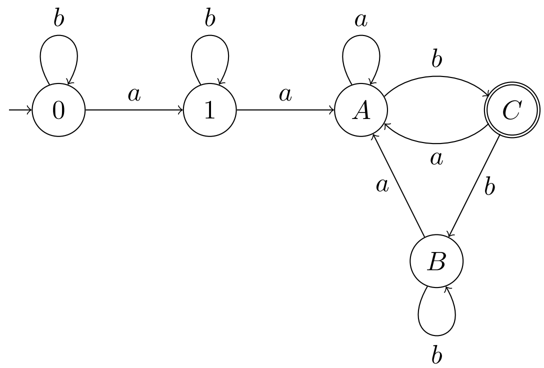


Figure 5: DFA for language L_B.