CS4384 : Automata Theory
Homework Assignment 1

Matthew McMillian
mgm160130@utdallas.edu

September 6, 2018

1. Let $B \subset N_1$ (given) be a finite set of positive integers and define $\Sigma = \{a, b\}$. *Mathematically* define a DFA, A, that accepts a string $s \in \Sigma^*$ if and only if $\forall_{i \in B}$, the i^{th} symbol of s is b.

   We immediately define a this new DFA's Σ, Q, Q_0, and F based off the criteria in the question.

   $\Sigma = \{a, b\}$, the alphabet of both sets.

   $Q = [0, n] \cap \mathbb{N}_0$, where $n = \max(\mathrm{B} \cup \{0\})$. We define n in such a that the maximum number of states that we need is the maximum number required in B since we can loop back on the final n state as an accepting state after B's criteria is satisfied. We intersect with $\mathbb{N}_0$ to ensure our set of states are states numbered greater than 0.

   $Q_0 = 0$, our start state is defined to be state 0.

   $F = \{0, n\}$, our set of accepting states. 0 is an accepting state since we must accept the empty set, and n is our final state that satisfies B's requirements.

   The transition function can be broken down into 3 parts.

   – We define an arbitrary set $S_1 = \{((q, b), q + 1) \mid q, q + 1 \in Q\}$. No matter what state we are in, it is OK to transition to the next state $q+1$ if our input is a b, since this will always satisfy B's requirements.
   – We define an arbitrary set $S_2 = \{((q, a), q + 1) \mid q \in Q, q + 1 \in Q - B\}$. We only want to transition to another state with an a iff we are not transitioning to a state that B requires as a b transition.
   – We define an arbitrary set $S_3 = \{((n, \sigma), n) \mid \sigma \in \Sigma\}$. This ensures that our case with the empty set, and our case that we finish looping through B will accept anything after that input.

   $\delta = S_1 \cup S_2 \cup S_3 = \{((q, b), q+1) \mid q, q+1 \in Q\} \cup \{((q, a), q+1) \mid q \in Q, q+1 \in Q - B\} \cup \{((n, \sigma), n) \mid \sigma \in \Sigma\}$, which satisfies all of the cases in the DFA.

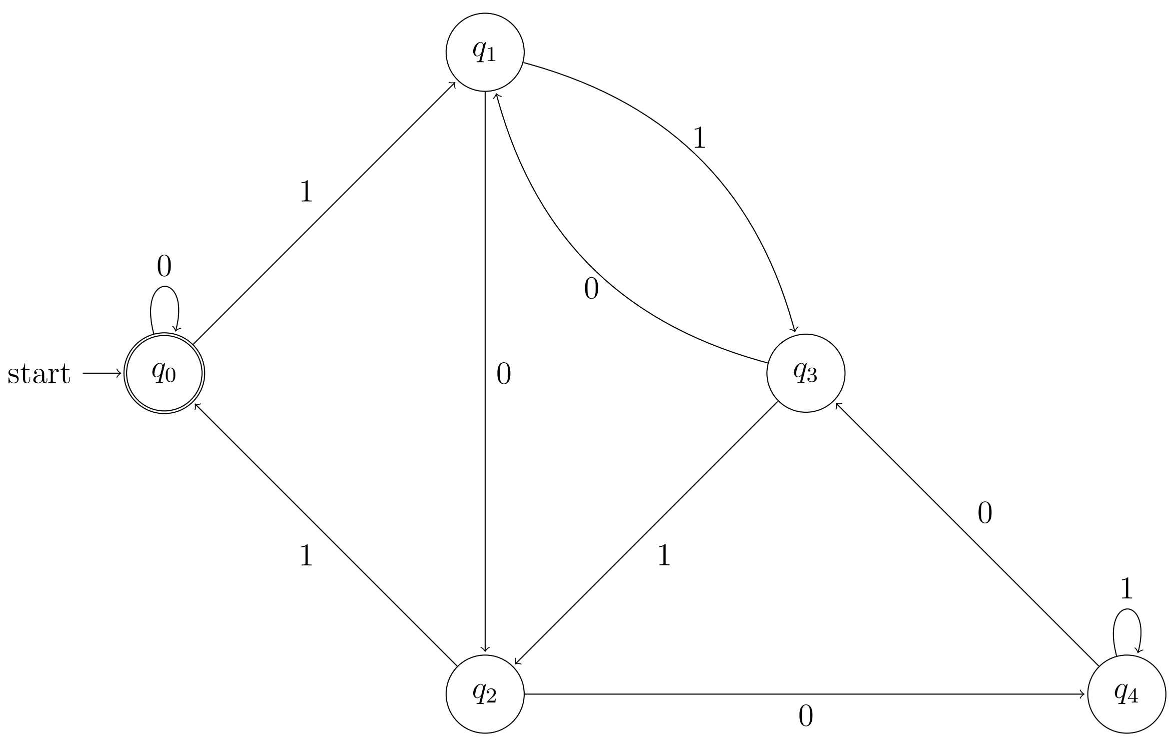
   Thus, we define $\boxed{\mathrm{A} = (Q,\, \Sigma,\, \delta,\, Q_0,\, F)}$ respectively.

2. Construct a DFA that accepts all binary strings divisible by 5.

   For this problem, $\Sigma = \{0, 1\}$, and $Q = \{0, 1, 2, 3, 4\}$. Our δ, transition function, is defined as $Q \times \Sigma \rightarrow Q$, which means we should have $Q \times \Sigma$, 2×5, transitions in our diagram. Our $F = \{0\}$. We can build a table modeling the relationships that our states should have until we have $Q \times \Sigma$ transitions. Then we can build our DFA out of the table's transitions.

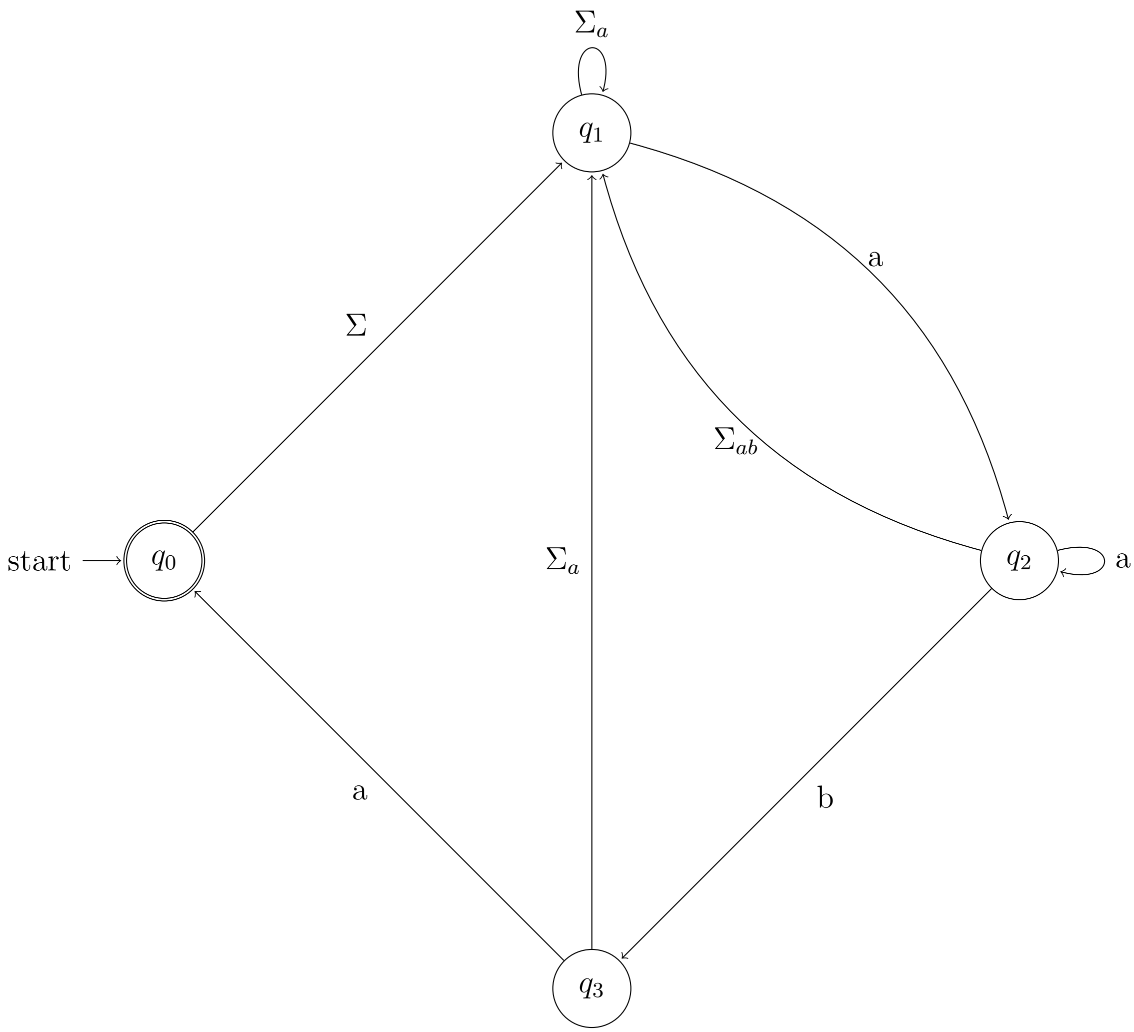
Number	Mod5	Binary	State
1	1	1	1
2	2	10	2
3	3	11	3
4	4	100	4
5	0	101	0
6	1	110	1
7	2	111	2
8	3	1000	3
9	4	1001	4
10	0	1010	5

   Now that we have our table, we can begin to build our DFA from it's transitions.



3. Construct an NFA for the language $L = \{w \mid w \text{ contains the substring } aba\}$.

For this NFA, $\Sigma = \{w \mid w \text{ is any character}\}$, and $Q = \{0, 1, 2, 3\}$. Let $\Sigma_a = \Sigma - \{a\}$ and $\Sigma_{ab} = \Sigma - \{a, b\}$. The following NFA satisfies the language L.



4. Convert the following NFAs to DFAs.

a.) We construct 2 tables, 1 NFA table, and 1 DFA table. We get the values from the NFA table, and iteratively through filling out the DFA table.

Table 1: NFA to DFA

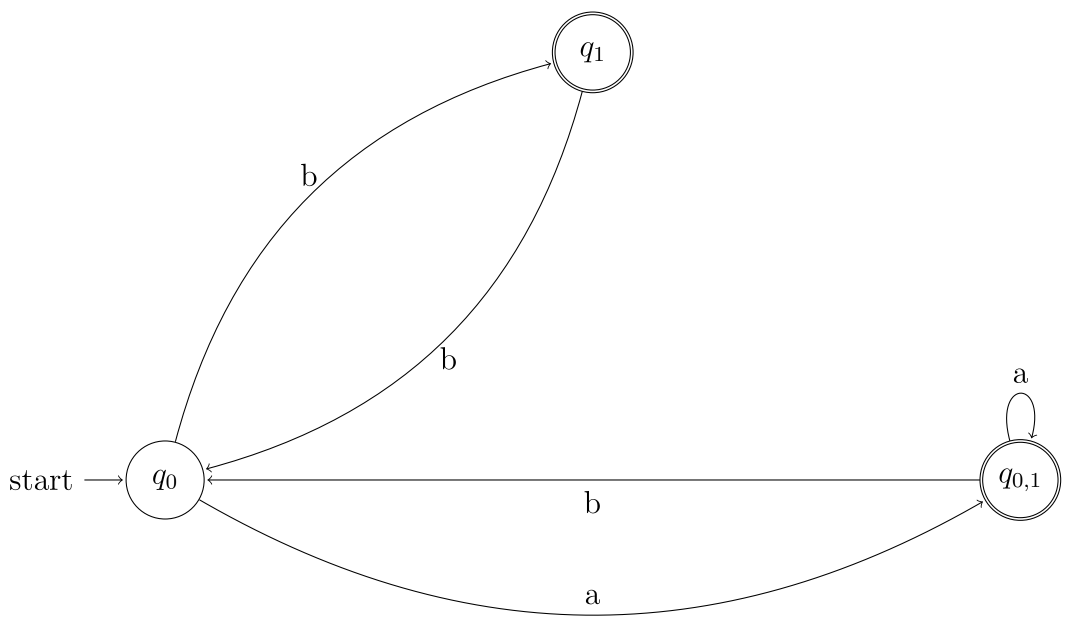
Table 2: NFA

q	a	b
0	0,1	-
1	-	0

Table 3: DFA

q	a	b
0	0,1	-
1	-	0
0,1	0,1	0

And we can then construct our DFA from the prior table.



b.) Similar to part a.), we construct 2 tables, 1 NFA table, and 1 DFA table. We get the values from the NFA table, and iteratively through filling out the DFA table.

And we can then construct our DFA from the prior table.

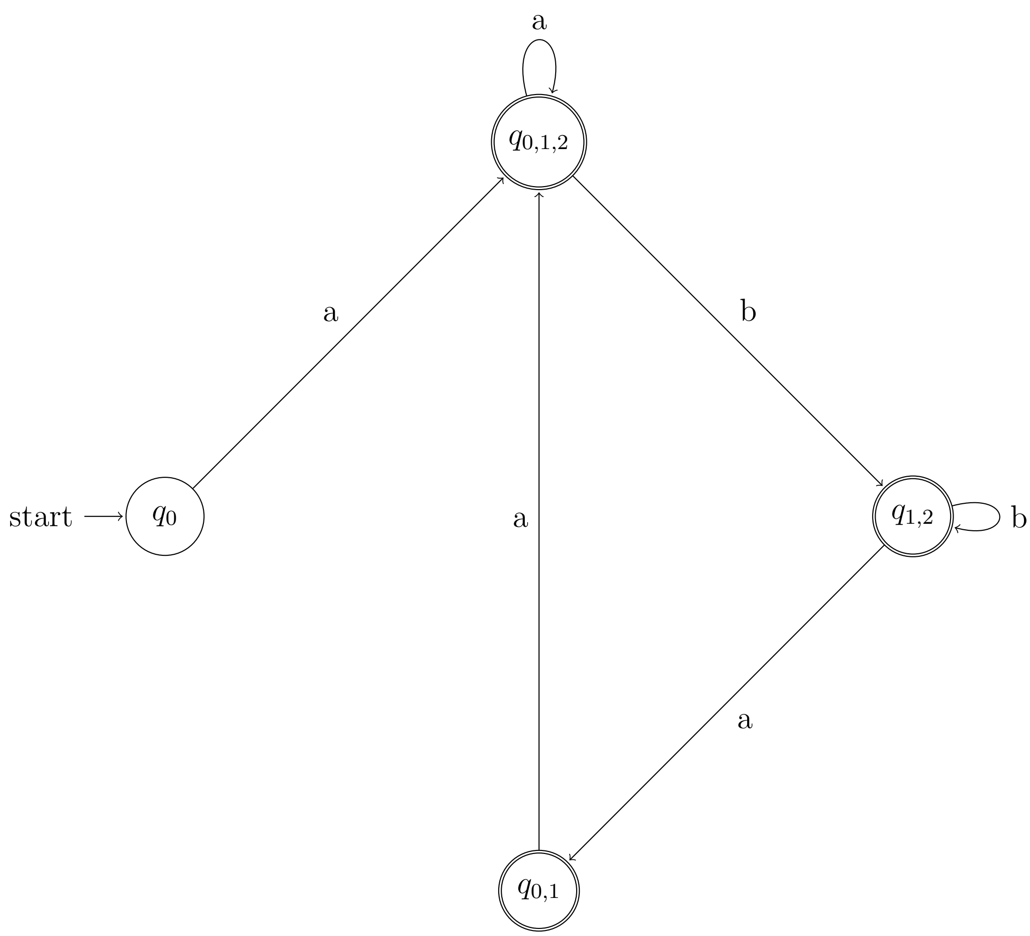
Table 4: NFA to DFA

Table 5: NFA

q	a	b
0	0,1,2	-
1	0	-
2	1	1,2

Table 6: DFA

q	a	b
0	0,1,2	-
0,1,2	0,1,2	1,2
1,2	0,1	1,2
0,1	0,1,2	-



5. Convert the following DFAs to REs.

We define Adren's Theorm below:

Let P,Q be regular expressions. If p $\neq \emptyset$, then R = Q + RP = QP*.

a.) Regular Expression for DFA a.,

$$
\begin{aligned}
q_0 &= aq_0 + bq_1 + \epsilon \\
q_1 &= aq_1 + bq_0 = bq_0a^* && \text{(by Adren's Theorm)} \\
q_0 &= aq_0 + bbq_0a^* + \epsilon \\
q_0 &= q_0(a + bba^*) + \epsilon \\
q_0 &= (a + bba^*)^* && \text{(by Adren's Theorm)}
\end{aligned}
$$

Thus our regular express for part a.) is $\boxed{(a + bba^*)^*}$.

b.) Regular Expression for DFA b.,

$$
\begin{aligned}
q_0 &= aq_1 + bq_1 + \epsilon \\
q_1 &= aq_1 + bq_2 \\
q_2 &= bq_1 + aq_0 \\
q_1 &= aq_1 + b(bq_1 + aq_0) = aq_1 + bbq_1 + aq_0 \\
q_1 &= q_1(a + bb) + aq_0 = (a + bb)^*aq_0 && \text{(by Adren's Theorm)} \\
q_0 &= aa(a + bb)^*q_0 + ba(a + bb)^*q_0 + \epsilon \\
q_0 &= q_0(aa(a + bb)^* + ba(a + bb)^*) + \epsilon \\
q_0 &= (aa(a + bb)^* + ba(a + bb)^*)^* && \text{(by Adren's Theorm)}
\end{aligned}
$$

Thus our regular express for part b.) is $\boxed{(aa(a + bb)^* + ba(a + bb)^*)^*}$.