Automata & Formal Languages

Homework 2 – Regular Languages

Abraham Murciano

March 18, 2020

Part 1

1. Given a finite automaton $A = (\Sigma, Q, q_0, \delta, F)$ with $Q = F$, it follows that $\mathcal{L}_A = \Sigma^*$. This is because a word w is accepted if and only if $\hat{\delta}(q_0, w) \in F$. And we know that $\hat{\delta} : Q \times \Sigma^* \to Q$, therefore in this case we can say that $\hat{\delta} : Q \times \Sigma^* \to F$, because $Q = F$.

   Therefore, $\forall w \in \Sigma^*, \hat{\delta}(q_0, w) \in F$, meaning every word in Σ^* is accepted. Since there are no words which aren't in Σ^*, we can say $\mathcal{L}_A = \Sigma^*$.

2. If $\mathcal{L}$ is an infinite language, it may still be regular. For example, if $\mathcal{L} = \Sigma^*$, it is an infinite language, but it is regular since an automata can easily be built for it.

3.

4. In all deterministic finite automata (DFA), the number of non-cyclic paths from the start state to any state in the automaton must be finite.

   A non-cyclic path is a path that does not contain the same state more than once. Since there is a finite number of states in the DFA, the total number of distinct non-cyclic paths from the start state is finite, since this number is bounded from above by $\sum_{i=0}^{|Q|} |\Sigma|^i$.

Part 2

1. Let $\mathcal{L}_1$ be a language over $\{a, b\}$ whose words are of even length, and let $\mathcal{L}_2$ be a language over the same alphabet whose words end in "bbb". Figure 1 shows DFAs accepting $\mathcal{L}_1$ and $\mathcal{L}_2$, and Figure 2 shows a DFA accepting only words which $\mathcal{L}_1$ accepts but $\mathcal{L}_2$ does not.

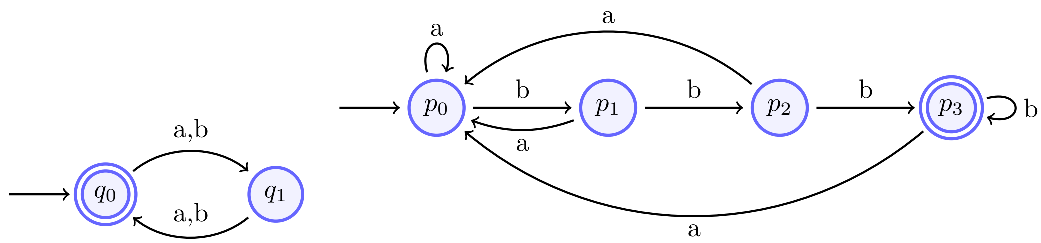

Figure 1: A DFA accepting $\mathcal{L}_1$ (left) and another accepting $\mathcal{L}_2$ (right)

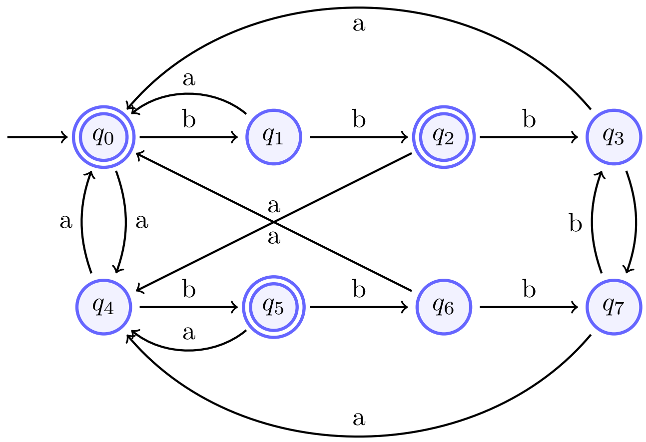

Figure 2: A DFA accepting $\mathcal{L}_1 - \mathcal{L}_2$

2. Let $\mathcal{L}$ be language over the alphabet $\{\mathrm{a}, \mathrm{b}\}$ whose words contain an equal number of sub-strings "ab" and "ba". For example, ε, "a", "b", "abba", "aba", and "abaaba" belong to the language, while "ab", "ba", and "baaaba" do not.

   This is a regular language as it is possible to construct a DFA to compute this as shown in Figure 3.

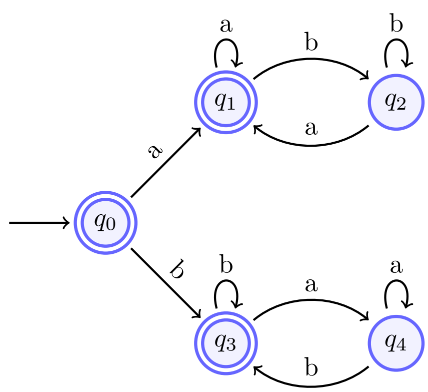

Figure 3: A DFA which computes if a word contains "ab" and "ba" the same number of times