Automata & Formal Languages

Homework 4 – Non-Deterministic Finite Automata

Abraham Murciano

April 21, 2020

1. (a) Figure 1 shows an NFA without ε transitions or multiple start states which identifies the language $\{\varepsilon\}$.

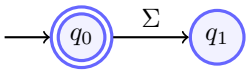

Figure 1: NFA for question 1a

(b) Figure 2 shows an NFA which accepts the language of words over {a,b} that end in "abb".

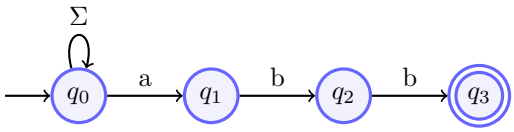

Figure 2: NFA for question 1b

(c) Figure 3 shows an NFA which accepts the language of words over $\Sigma = \{a, b\}$ that contain "aa" or that have an odd number of "b"s.

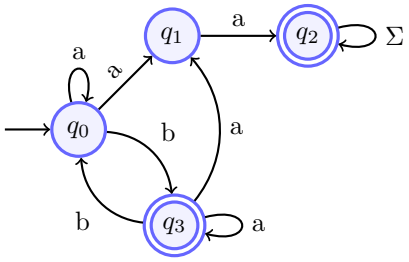

Figure 3: NFA for question 1c

2. Figure 4 shows a conversion of the NFA in figure 3 into a DFA. Table 1 shows a table which we can use to aid us in the construction of the DFA.

New label	Current state	Transition on a	Transition on b
r_0	$\{q_0\}$	$\{q_0, q_1\}$	$\{q_3\}$
r_1	$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$	$\{q_3\}$
r_2	$\{q_3\}$	$\{q_1, q_3\}$	$\{q_0\}$
r_3	$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_2, q_3\}$
r_4	$\{q_1, q_3\}$	$\{q_1, q_2, q_3\}$	$\{q_0\}$
r_5	$\{q_2, q_3\}$	$\{q_1, q_2, q_3\}$	$\{q_0, q_2\}$
r_6	$\{q_1, q_2, q_3\}$	$\{q_1, q_2, q_3\}$	$\{q_0, q_2\}$
r_7	$\{q_0, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_2, q_3\}$

Table 1: A table to translate the NFA in figure 3 to the DFA in figure 4

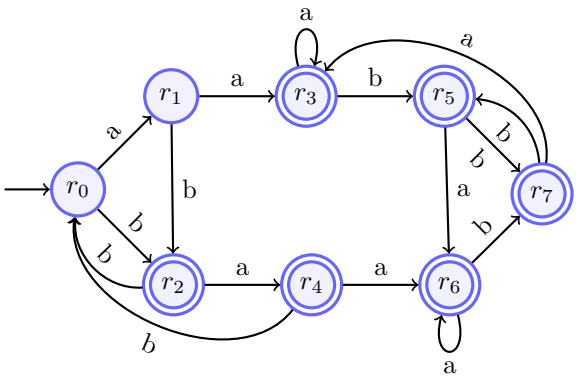

Figure 4: A DFA equivalent to the automata in figure 3

3. (a) Let A be the following NFA without ε transitions,

$$A = (Q_A, \Sigma, \delta_A, q_0, F_A)$$

where Q_A is a finite set of states, Σ is the alphabet, $\delta_A : Q_A \times \Sigma \to P(Q_A)$ is the transition function, q_0 is the only start state, and F_A is the set of final states.

We must show that an NFA, B, exists which has only one final state and accepts the same language as A with the exception of ε. Specifically,

$$\mathcal{L}(B) = \mathcal{L}(A) - \{\varepsilon\}$$

Let

$$B = (Q_B, \Sigma, \delta_B, q_0, \{f_B\})$$

where $Q_B = Q_A \cup \{f_B\}$ is the set of states, $\delta_B : Q_B \times \Sigma \to P(Q_B)$ is the transition function, q_0 is once again the only start state, and f_B is the only final state of B which was not in Q_A but is in Q_B.

To properly define B given that A is properly defined, all that remains is to provide a definition for δ_B in terms of elements of A.

$$\delta_B(q, \sigma) = \begin{cases} \delta_A(q, \sigma) & \text{if } \delta_A(q, \sigma) \cap F_A = \phi \\ \delta_A(q, \sigma) \cup \{f_B\} & \text{if } \delta_A(q, \sigma) \cap F_A \neq \phi \end{cases}$$

In simpler terms, what all of the above means is that given the NFA A, we can construct another NFA which we label B which has only one final state f_B by adding a new state f_B and transitioning to it as well whenever a transition takes us to what would have been a final state in NFA A.

We must now show that indeed, $\mathcal{L}(B) = \mathcal{L}(A) - \{\varepsilon\}$. To do so we will show that given a word $w = u\sigma$ which ends is the symbol σ, if A accepts it then so does B, and vice versa.

$$\begin{aligned}
w \in \mathcal{L}(A) &\Rightarrow \hat{\delta}_A(q_0, w) \cap F_A \neq \phi \\
&\Rightarrow \delta_A(\hat{\delta}_A(q_0, u), \sigma) \cap F_A \neq \phi \\
&\Rightarrow \delta_B(\hat{\delta}_A(q_0, u), \sigma) = \delta_A(\hat{\delta}_A(q_0, u), \sigma) \cup \{f_B\} \\
&\Rightarrow \hat{\delta}_B(q_0, w) \cap \{f_B\} = \{f_B\} \neq \phi && \text{because } \hat{\delta}_A(q_0, u) \subseteq \hat{\delta}_B(q_0, u) \\
&\Rightarrow w \in \mathcal{L}(B)
\end{aligned}$$

The above proves that if A accepts w then so does B. To prove that if B accepts w then A accepts it, we can use the following proof.

$$\begin{aligned}
w \in \mathcal{L}(B) &\Rightarrow \hat{\delta}_B(q_0, w) \cap \{f_B\} \neq \phi \\
&\Rightarrow f_B \in \hat{\delta}_B(q_0, w) \\
&\Rightarrow f_B \in \delta_B(\hat{\delta}_B(q_0, u), \sigma) \\
&\Rightarrow f_B \in \delta_B(\hat{\delta}_A(q_0, u), \sigma) && \text{because } \delta_B(f_B, \sigma) = \phi \\
&\Rightarrow \delta_A(\hat{\delta}_A(q_0, u), \sigma) \cap F_A \neq \phi \\
&\Rightarrow \hat{\delta}_A(q_0, w) \cap F_A \neq \phi \\
&\Rightarrow w \in \mathcal{L}(A)
\end{aligned}$$

This proves that any word which is accepted by one is also accepted by the other, with the exception of the word ε. This proof is invalid for ε because we assumes that $w = u\sigma$, so w must have at least one symbol (σ).

(b) Since all regular languages can be identified by an NFA with a single start state and without ε transitions, then any regular language which does not contain ε can be identified by an NFA with a single start state, without ε transitions, and with only one final state. This NFA can be constructed using the definition specified in part a.

4. Let $\mathcal{L}$ be any regular language. We must show that $\mathcal{L}^R$ – that is, the language of all the words which can be obtained from $\mathcal{L}$ by reversing the order of the symbols – is also regular.

   Since $\mathcal{L}$ is a regular language, there exists an NFA which accepts it. That NFA can be modified to accept $\mathcal{L}^R$. Firstly, the final states of the new NFA will be only the old start state. Then a new state q_0 will be added. This new state is the start state of the new NFA. If $\varepsilon \in \mathcal{L}$, then q_0 will also be accepted.

   Next, all the transitions from a state q_a to another state q_b on a symbol σ will be reversed, so that q_b will transition to q_a on that same symbol σ. And finally, for all transitions (after the reversal of the transitions) coming out of a state q_f which was a final state in the old NFA, to any other state q_g on a symbol σ will be copied so that there is a transition coming out of q_0 leading to q_g on the symbol σ.

   This is all that is needed to construct an NFA which accepts $\mathcal{L}^R$ given an NFA that accepts $\mathcal{L}$. Therefore $\mathcal{L}^R$ must be regular.

5. Given any finite automaton A, if we reverse every acceptance state in A to a non-acceptance state, and vice versa, we will not necessarily receive an automaton which identifies the complement of $\mathcal{L}(A)$.

   We will prove this with a counter-example. Consider the NFA in figure 5. It accepts the language $\{\varepsilon\}$. If we make all accepted states not accepted and all not accepted states into accepted ones, then we end up with the NFA in figure 6. The accepted language fot this NFA is ϕ. This is clearly not the complement of $\{\varepsilon\}$, so we can conclude that toggling the states' acceptance will not necessarily yield an automaton which accepts the complement of the language.

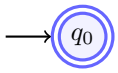

Figure 5: An NFA which accepts the language $\{\varepsilon\}$

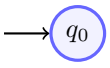

Figure 6: An NFA which accepts the language ϕ