1 Nondeterministic Finite Automata (1.2, Continued)

This continues from the notes from Monday, January 12.

1.1 Equivalence of NFAs and DFAs

Deterministic and nondeterministic finite automata both recognize the same class of languages.

Theorem 1.1

Every nondeterministic finite automaton has an equivalent deterministic finite automaton.

Remark: Here, we say that two machines are equivalent if they recognize the same language.

The proof is as follows[1]:

Proof. Let $N = (Q, \Sigma, \delta, q_0, F)$ be the NFA that recognizes the language L. We want to show that there is a DFA $M = (Q', \Sigma, \delta', q_0', F')$ which recognizes the same L.

1. First, $Q' = \mathcal{P}(Q)$. This is because must have the states in Q' to represents the possible subset of states in Q. In an NFA, we can make multiple copies of the automaton, which may end up at different states over time. We therefore need to account for where these copies can be in our corresponding DFA.

2. The alphabet Σ is the same in both the NFA and DFA.

3. The transition function of the corresponding DFA is defined by:

$$\delta'(X, x) = \{q \in Q \mid q \in \delta(r, x) \text{ for some } r \in X \text{ or accessible via } \epsilon \text{ transitions}\}$$

   Where X is a state of the DFA and $x \in \Sigma$. Because a state in an NFA can have multiple outgoing transition arrows under the same type (e.g. two outgoing arrows for `a`), we need to account for this in the corresponding NFA. This is our first condition in our δ' function; in this sense, if we consider the possible states that we can go to in the NFA, then the corresponding state in our DFA is the union of all of those possible states. We must also consider that, for a given state in an NFA, there may be ϵ transitions. In case there are ϵ transitions, we need to consider where the ϵ transitions put a copy of the machine.

4. The start state in the corresponding DFA is the set $q_0' = \{q_0\} \cup \delta^*(q_0, \epsilon)$. First, we note that the start state in the NFA is q_0; thus, the start state in the corresponding DFA must be *at least* $\{q_0\}$. However, if there are any ϵ transitions from the start state, we must consider those as well since transitioning to another state from the state state via the ϵ transition doesn't consume any input.

5. The set of final states in the corresponding DFA is simply:

$$F' = \{X \mid X \subseteq Q \text{ and } X \cap F \neq \emptyset\}$$

   Here, we're saying that if there are any sets in Q' which contain a final state in F, then said set must be a final set. This is because, in a NFA, we may have multiple copies of the machine running, and if one copy stops at a final state, then the NFA is accepted (despite the other copies not necessarily being at a final state).

The rest of the proof is omitted for now. $\square$

[1] This proof was used in our submission for HW2 Problem 3 (CSE 105, WI22). The group members involved in this submission are (only initials and the last two digits of their PID are shown): CB (67), TT (96), ASRJ (73), and me.

1.1.1 Example: NFA to DFA

Consider the following NFA N:

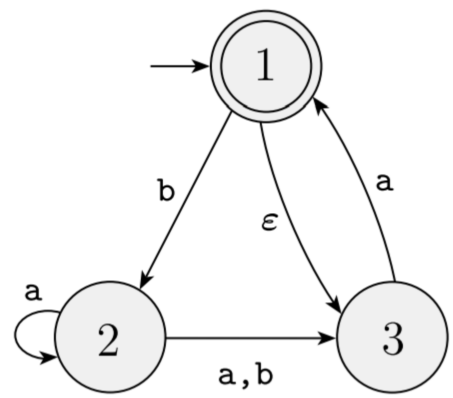


Figure: The NFA N.

We can define $N = (Q, \Sigma, \delta, q_0, F)$ like so:

- $Q = \{1, 2, 3\}$
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$
- δ is defined by

	a	b	ϵ
1	$\emptyset$	$\{2\}$	$\{3\}$
2	$\{2, 3\}$	$\{3\}$	$\emptyset$
3	$\{1\}$	$\emptyset$	$\emptyset$

- $q_0 = 1$
- $F = \{1\}$

We're now being asked to construct a corresponding DFA:

$$D = (Q', \Sigma', \delta', q_0', F')$$

Here, it's trivial to note that:

- $Q' = \mathcal{P}(Q) = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$.
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$
- $q_0 = \{1, 3\}$. This is because we can start at both state 1 and 3 since 3 has an ϵ transition.
- $F' = \{\{1\}, \{1, 2\}, \{1, 3\}, \{1, 2, 3\}\}$. This is because we want all subsets that contain N's accept state.

The hard part is actually "wiring" the DFA up, i.e. the transition function. To do this, we need to analyze how the NFA acts and "translate" it to what the DFA would do. So, let's consider each element in Q' and see how it would relate to the NFA.

- Consider $\{1\} \in Q'$. In the NFA:
  - 1 doesn't go anywhere when `a` is given by itself. **However**, 1 can go to 3 since this is an ϵ transition, and 3 goes to 1 when consuming `a`, so it follows that $\boxed{\{1\} \xrightarrow{\mathtt{a}} \{1, 3\}}$ in the corresponding DFA.

  - 1 goes to 2 when `b` is given, so it follows that $\boxed{\{1\} \xrightarrow{\texttt{b}} \{2\}}$ in the corresponding DFA.

- Consider $\{2\} \in Q'$. In the NFA:
  - 2 goes to 2 *and* 3 when `a` is given, so it follows that $\boxed{\{2\} \xrightarrow{\texttt{a}} \{2,3\}}$ in the corresponding DFA.
  - 2 goes to 3 when `b` is given, so it follows that $\boxed{\{2\} \xrightarrow{\texttt{b}} \{3\}}$ in the corresponding DFA.

- Consider $\{3\} \in Q'$. In the NFA:
  - 3 goes to 1 when `a` is given, but then it can also go to 3 since there is an ϵ transition, so it follows that $\boxed{\{3\} \xrightarrow{\texttt{a}} \{1,3\}}$.
  - 3 doesn't go anywhere when `b` is given, so it follows that $\boxed{\{3\} \xrightarrow{\texttt{b}} \emptyset}$.

We can use the above to build cases for the remaining elements in Q'.

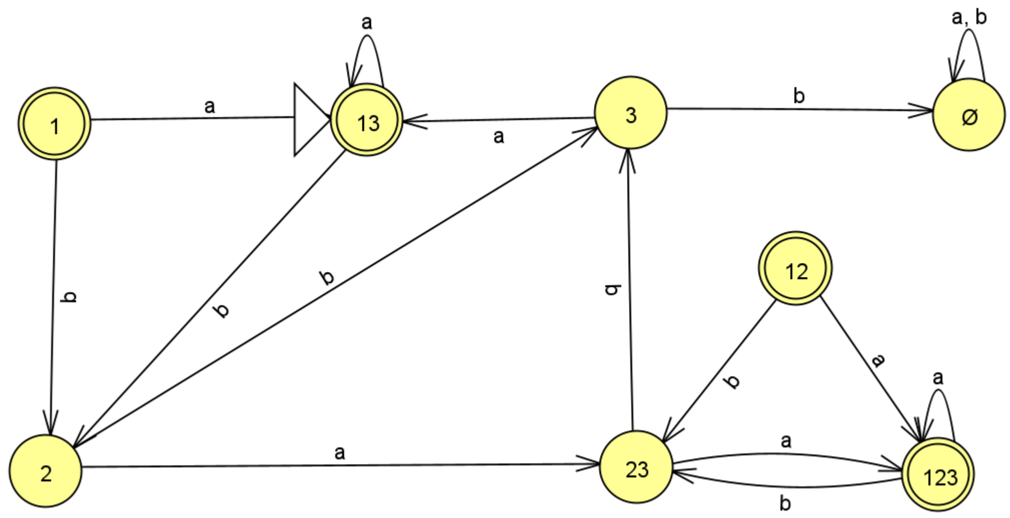
- Consider $\{1,2\} \in Q'$. In the corresponding NFA, this means that there's a copy at state 1 and a copy at state 2. So:
  - Suppose `a` is given. Then, from our previous work, we know that $\{1\} \xrightarrow{\texttt{a}} \{1,3\}$, and $\{2\} \xrightarrow{\texttt{a}} \{2,3\}$. Therefore, $\boxed{\{1,2\} \xrightarrow{\texttt{a}} \{1,2,3\}}$ (recall that we take the union).
  - Suppose `b` is given. Then, we know that $\{1\} \xrightarrow{\texttt{b}} \{2\}$, and $\{2\} \xrightarrow{\texttt{b}} \{3\}$. Therefore, $\boxed{\{1,2\} \xrightarrow{\texttt{b}} \{2,3\}}$.

- Consider $\{1,3\} \in Q'$. In the corresponding NFA, this means that there's a copy at state 1 and a copy at state 3. So:
  - Suppose `a` is given. Then, from our previous work, we know that $\{1\} \xrightarrow{\texttt{a}} \{1,3\}$, and $\{3\} \xrightarrow{\texttt{a}} \{1,3\}$. Therefore, $\boxed{\{1,3\} \xrightarrow{\texttt{a}} \{1,3\}}$.
  - Suppose `b` is given. Then, we know that $\{1\} \xrightarrow{\texttt{b}} \{2\}$, and $\{3\} \xrightarrow{\texttt{b}} \emptyset$. Therefore, $\boxed{\{1,3\} \xrightarrow{\texttt{b}} \{2\}}$.

- Consider $\{2,3\} \in Q'$. In the corresponding NFA, this means that there's a copy at state 2 and a copy at state 3. So:
  - Suppose `a` is given. Then, from our previous work, we know that $\{2\} \xrightarrow{\texttt{a}} \{2,3\}$, and $\{3\} \xrightarrow{\texttt{a}} \{1,3\}$. Therefore, $\boxed{\{2,3\} \xrightarrow{\texttt{a}} \{1,2,3\}}$.
  - Suppose `b` is given. Then, we know that $\{2\} \xrightarrow{\texttt{b}} \{3\}$, and $\{3\} \xrightarrow{\texttt{b}} \emptyset$. Therefore, $\boxed{\{2,3\} \xrightarrow{\texttt{b}} \{3\}}$.

- Consider $\{1,2,3\} \in Q'$. In the corresponding NFA, this means that there's a copy at state 1, 2, and 3. So:
  - Suppose `a` is given. From our previous work, we know that $\boxed{\{1,2,3\} \xrightarrow{\texttt{a}} \{1,2,3\}}$.
  - Suppose `b` is given. From our previous work, we know that $\boxed{\{1,2,3\} \xrightarrow{\texttt{b}} \{2,3\}}$.

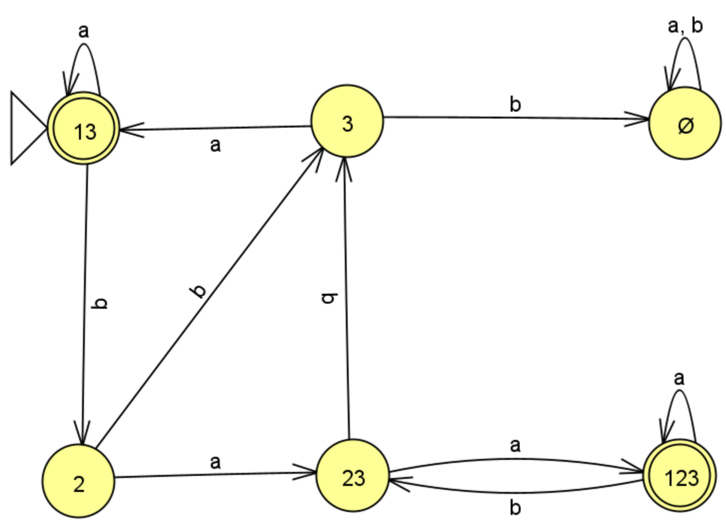
This gives us the following DFA:



However, we note a few things.

- State 1 doesn't have anything coming into it. Therefore, we can remove it.
- State 12 doesn't have anything coming into it. Therefore, we can remove it.

This gives us the simplified DFA:



1.2 Applications of Theorem

There are several applications of this theorem.

Corollary 1.1

A language is regular if and only if some nondeterministic finite automaton recognizes it.

Theorem 1.2

The class of regular languages is closed under the union operation.

Proof. (Sketch.) Suppose A_1 and A_2 are regular languages. We want to show that $A_1 \cup A_2$ is regular. We can take two NFAs, N_1 for A_1 and N_2 for A_2, and combine them to make one new NFA N. The idea is that N must accept its input if either N_1 and N_2 accepts. So, essentially, we want to run both N_1 and N_2 in parallel. To simulate this behavior, we can create a new start state q_0 with two ϵ transitions pointing to the original start states of N_1 and N_2 (everything else about N_1 and N_2 are left unchanged). □

Theorem 1.3

The class of regular languages is closed under the concatenation operation.

Proof. (Sketch.) Suppose A_1 and A_2 are regular languages. We want to show that $A_1 \circ A_2$ is regular. We can take two NFAs, N_1 for A_1 and N_2 for A_2, and combine them to make one new NFA N. The idea for N is as follows:

- Start at the starting state for N_1 and remove the starting state for N_2.
- Connect each accept state in N_1 to the original start state in N_2. The accept states in N_1 will no longer be accept states.

By starting at the N_1 part of N, we guarantee that we will recognize some language A_1. Then, once we hit the original accept state in N_1, we can evaluate the rest of the string in N_2. If we hit an accept state in N_2, then we have recognized $A_1 \circ A_2$. □

Theorem 1.4

The class of regular languages is closed under the union operation.

Proof. Suppose A_1 is a regular language. We want to show that A_1^* is also regular. Consider the NFA N_1 for A_1. We want to modify N_1 so it recognizes A_1^*. Thus, our idea for the new NFA N is as follows:

- Because ϵ (the empty string) is valid under A_1^*, we must make a new start state that goes to the original start state; then, we can make the transition from the new start state to the original start state ϵ.
- We can connect the accept states in N_1 back to the original start state (not the new start state) with the labels being ϵ.
- The accept states in N_1 is the same for N.

By starting at the new start state, we can guarantee that ϵ will be accepted if it is the only thing to be read. Processing the string is as expected. However, once we reach the accept state, we need to *go back* to the original start state to process the next "word." This process keeps going until we no longer have any words to process. In this case, if we end off at any accept state with nothing left to read, then we accept. □