Tutorial 3

Gidon Rosalki

2025-04-09

1 Determinisation

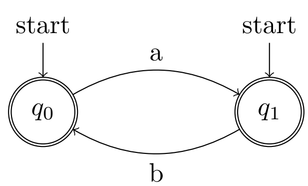
We will demonstrate how to construct a DFA that is equivalent to an NFA.

1.1 Subset construction

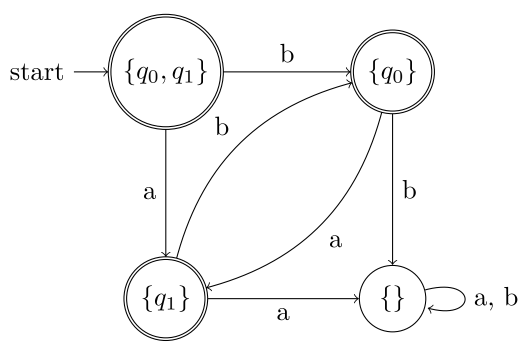
Let there be the NFA $A = (Q, \Sigma, \delta, Q_0, F)$. We define an equivalent DFA $A_D = (P, \Sigma, \eta, p_0, G)$. In this case:

- $P = 2^Q$
- $p_0 = Q_0$
- $G = \{S \subseteq Q : S \cap F \neq \emptyset\}$
- The transition function includes every state in the NFA reachable from here: $\eta(S, \sigma) = \bigcup_{q \in S} \delta(q, \sigma)$

  For example:



  So after subset construction we get:



1.2 Simulation on an NFA

Let there be an NFA A with n states. How do we determine the acceptance of the word w? We have a few options:

1. Determinisation and running the word w on the DFA: this will take $O(2^n \cdot |\Sigma|)$, both in terms of runtime, and space. Running a word takes an additional $O(|w|)$.
2. Running all the available options - brute force: This will take $O\left(n^{|w|}\right)$ to go over all the possible runs of the NFA over w.
3. **Subset construction simulation**: Here we iteratively evaluate all possible transition states for the letter $\sigma_1 \dots \sigma_n$ of the word w. We accept if the final set contains a final state.

This will take $O\left(n^2 \cdot |w|\right)$.

Subset construction simulation 1

w *input*

bool *output*

$S \leftarrow Q_0$

for i=1 in $[|w|]$ **do**

  $S \leftarrow \bigcup_{q \in S} \delta(q, \sigma_i)$

end for

if $F \cap S \neq \emptyset$ **then**

  return accept

else

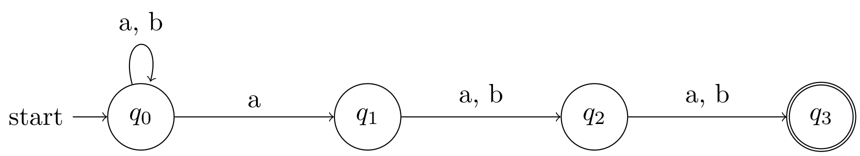
  return reject

end if

Theorem 1. *After every step i, $S = \delta^*(Q_0, w_{\leq i})$ where $w_{\leq i} = w_1 \dots w_i$*

Proof . Guess what? It's by induction. Again. Feel free to do it yourself. $\square$

Example:



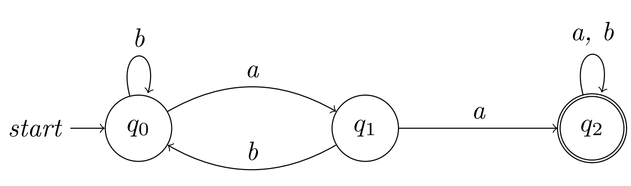
This is the NFA that recognises $L_k = \left\{w \in \{a,b\}^* : \text{The } k \text{ letter from the end is } a\right\}$, so in this case L_3. So, if we simulate for the word *baabbb*, we get the following:

1. $S = Q_0 = \{q_0\}$
2. $S = \{q_0\}$
3. $S = \delta(q_0, a) = \{q_0, q_1\}$
4. $S = \delta(q_0, a) \cup \delta(q_1, a) = \{q_0, q_1, q_2\}$
5. $S = \delta(q_0, q_b) \cup \delta(q_1, b) \cup \delta(q_2, b) = \{q_0, q_2, q_3\}$
6. $S = \{q_0, q_3\}$
7. $S = \{q_0\}$

2 Myhill-Nerode

Definition 2.1 (Separating suffix)**.** *Let there be $L \subseteq \Sigma^*$, and $x, y \in \Sigma^*$. $z \in \Sigma^*$ will be called the Separating suffix if $xz \in L \wedge yz \notin L$, or the opposite.*

Example 1. *Consider the language $L = \left\{w \in \{a,b\}^* : aa \subseteq w\right\}$.*



Solution. So here a is the separating suffix for the words *bbba* and b, since $bbbaa \in L$ but $ba \notin L$. However, *babab* and b do not have a separating suffix. $\square$

Theorem 2. *Let $L \in REG$, and let there be A a DFA that decides the language L. Let there be $x, y \in \Sigma^*$. If when running on A, x, y both arrive at the same state, as in $\delta^*(q_0, x) = \delta^*(q_0, y)$, then there does not exist a separating suffix.*

Proof . Let there be $z \in \Sigma^*$.

$$\begin{aligned}\delta^*(q_0, xz) &= \delta^*(\delta^*(q_0, x), z) \\ &= \delta^*(\delta^*(q_0, y), z) \\ &= \delta^*(q_0, yz)\end{aligned}$$

Therefore, xz and yz reach the same state, and therefore both belong to L, and do not belong to L together. □

However, if x, y have a separating suffix, then they will arrive at different states in A. Going back to our example, we can use this to show that there is no smaller DFA that recognises L. Consider a, b, aa: Every pair of two words have a separating suffix:

- a, b have the separating suffix a
- a, aa have the separating suffix ε
- b, aa have the separating suffix ε

We conclude that in every DFA A that recognises L, each of the words a, b, aa must reach a different state, so A has at least 3 states. Therefore, the DFA we saw above is minimal.

Definition 2.2 (Myhill-Nerode)**.** *Let there be $L \subseteq \Sigma^*$. $x, y \in \Sigma^*$ will be called MN equivalent with respect to L if there does not exist a separating suffix between x and y. In this case we will write $x \sim_L y$.*

Theorem 3 (Equivalence classes)**.** *$\sim_L$ is an equivalence class if*

- *Reflexive: $x \sim_L x$*
- *Symmetry: $x \sim_L y \Leftrightarrow y \sim_L x$*
- *Transitivity: $x \sim_L y \wedge y \sim_L z \implies x \sim_L z$*

Proof . The full proof is left as an exercise, but let us prove transitivity: Suppose that $x \sim_L y \wedge y \sim_L w$ then for every $z \in \Sigma^*$:

$$xz \in L \Leftrightarrow yz \in L \Leftrightarrow wz \in L$$

So $xz \in L \Leftrightarrow wz \in L \implies x \sim_L w$ □

So going back to the previous example, L has 3 different MN equivalence classes.

1. Words that contain aa
2. Words that don't contain aa, and end in a
3. Words that don't contain aa, and do not end in a

Theorem 4. *Given a regular language L, every MN equivalence class of L corresponds to a single state in the minimal DFA that decides L. In particular, the number of states in the minimal DFA is the number of MN equivalence classes.*

Proof . In the lecture. □

Theorem 5. *The language $L = \left\{w \in \{a, b\}^* : \#_a(w) \geq \#_b(w)\right\}$ is not regular.*

Proof . Let us assume the contradiction that it is regular, and therefore tehre exists an automaton that decides L, with k states. Consider the following $k+1$ words: $\varepsilon, a, aa, \ldots, a^k$. Since there are $k+1$, by the pigeonhole principle, 2 must reach the same state. Let there be such a pair a^i, a^j, and we will assume without loss of generality that $i < j$. We will choose $z = b^j$. So z is a separating suffix for a^i and a^j, thus contradicting the proposition. □

Theorem 6. *The language $L = \left\{q^{n^2} : n \geq 0\right\}$ (over $\Sigma = \{1\}$) is not regular.*

Proof . Let there be, without loss of generality, $i < j$ such that $x = 1^{i^2}, y = 1^{j^2} \in L$. We will show that x, y belong to different classes. Let $z = 1^{2i+1}$. So

$$|xz| = |x| + |z| = i^2 + 2i + 1 = (i+1)^2$$

So $xz \in L$. On the other hand

$$|yz| = |y| + |z| = j^2 + 2i + 1 < j^2 + 2j + 1 = (j+1)^2$$

So $j^2 < |yz| < (j+1)^2$, so its length is not a square number, and therefore $yz \notin L$. □

Theorem 7. *The language $L = \{1^p : p \text{ is a prime number}\}$ is not regular*

Proof Sketch. Every infinite unary regular language contains an arithmetic sequence. As there is a finite number of states, it must be that there is a loop of size k. If $w \in L$, then also $w \cdot \sigma \ldots \sigma \in L$ where σ appears k times, or $2k$ times, and so on. The prime numbers do not contain an arithmetic sequence. For every $n \in \mathbb{N}$, there is a sequence of size $n-1$, which is not prime. For example, $n!, n!+2, n!+3, \ldots, n!+n$. Since $n!$ is not prime, and is a multiplication by 2 by definition, then $n!+2$ is not prime as it is divisible by 2. Similarly, $n!+3$ is not prime since it is divisible by 3, and so on. $\square$