NFAs and Properties of Regular Languages

Prof. Shrisha Rao
IIIT Bangalore

srao@iiitb.ac.in

2025-08-12

More NFAs

Give NFAs to accept the following languages:

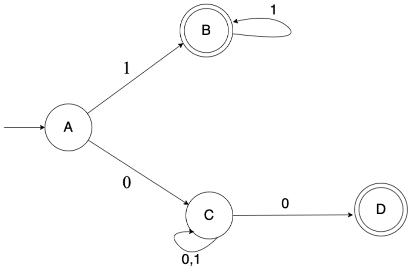
(1) The set of all strings with only an even number of 0s, or only exactly two 1s.

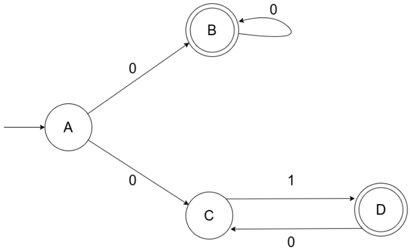
(2) The language 0^*.

(3) The set of all strings where every odd position is a 1.

Recap: Language of an NFA

What languages do these NFAs accept?

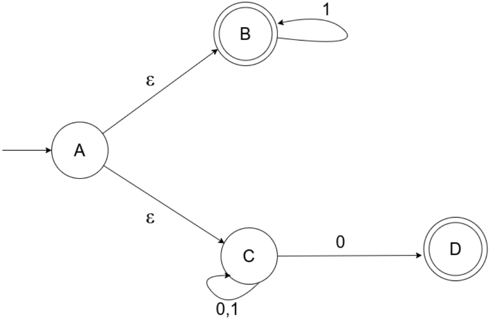




ϵ-TRANSITIONS

- An ϵ-transition happens when an NFA moves from one state to another on an ϵ, i.e., with no input symbol.
- Obviously, ϵ-transitions are only possible with NFAs, not with DFAs.
- ϵ-transitions are particularly useful in case of NFAs that have to accept REs of the form $A + B$.
- For instance, consider an NFA to accept all binary strings which have only 1s, or where the last symbol is a 0. (What is the RE for this language?)

NFA FOR $1^* + (0+1)^*0$



5-tuple definition of NFAs

- An NFA M is also given by the 5-tuple

$$M = \langle Q, \Sigma, \delta, q_o, F \rangle$$

- $Q, \Sigma, q_0,$ and F have the same meanings as before.
- The transition function δ has to be changed to allow for non-determinism.
- Problem: for any function, $f(x) = a$ and $f(y) = a$, with $x \neq y$ is quite possible; however, $f(x) = a$ and $f(x) = b$ with $a \neq b$ is impossible. (A function can map different values in the domain to the same value in the range, but it can never map a single value in the domain to different values in the range.)

Transition Function for NFAs

- The solution is to define $\delta : Q \times \Sigma \longrightarrow 2^Q$, i.e., δ maps from $Q \times \Sigma$ to the powerset of Q.
- In this way, the problem is averted. If an NFA can transition from state A to states B and C on some input a, then instead of saying $\delta(\mathsf{A}, a) = \mathsf{B}$ and $\delta(\mathsf{A}, a) = \mathsf{C}$, we say $\delta(\mathsf{A}, a) = \{\mathsf{B}, \mathsf{C}\}$.
- There is a further enhancement possible, given that an NFA can also transition on ϵ. So the best way to specify δ is by saying

$$\delta : Q \times \left(\Sigma \cup \{\epsilon\}\right) \longrightarrow 2^Q.$$

Exercise

(4) Give 5-tuple definitions for each of the three NFAs so far discussed (on slides 3 and 5).

Kleene's Theorem

The following are identical properties for a language L.

- L is a regular language (given by a regular expression).
- L is accepted by a DFA.
- L is accepted by an NFA.

Therefore, non-determinism does not add to the power of a FA, but it makes the construction easier.

Regularity is Closed Under Union

- The class of regular languages is closed under union; this means that if L_1 and L_2 are regular languages, then so is $L_1 \cup L_2$.
- We can prove this and similar properties using Kleene's theorem.
- In such cases, the approach is to assume that L_1 and L_2 are accepted by DFAs (or NFAs) M_1 and M_2, and then to construct new automata that accept $L_1 \cup L_2$.

Showing Regularity to be Closed Under Union

- Let $M_1 \equiv \langle Q_1, \Sigma, \delta_1, q_1, F_1 \rangle$ accept L_1, and $M_2 \equiv \langle Q_2, \Sigma, \delta_2, q_2, F_2 \rangle$ accept L_2. NB: Without loss of generality, we can assume that the alphabet Σ is common to M_1 and M_2.
- We need a machine M that accepts $L_1 \cup L_2$.
- M will also of course be of the form $\langle Q, \Sigma, \delta, q_0, F \rangle$.
- Have to specify the individual elements Q, δ, etc., in terms of Q_1, Q_2, δ_1, δ_2, etc.

The DFA for the Union Language

Solution idea:
Construct a compound machine M which runs M_1 and M_2 in parallel with their respective state changes for every input, and accepts if either M_1 or M_2 accepts.

- $Q \equiv Q_1 \times Q_2$
- $\delta : (Q_1 \times Q_2) \times \Sigma \longrightarrow Q_1 \times Q_2$, given by $\delta\big((r_1, r_2), a\big) = \big(\delta_1(r_1, a), \delta_2(r_2, a)\big)$, where $r_1 \in Q_1$ and $r_2 \in Q_2$.
- $q_0 \equiv (q_1, q_2)$, i.e., the starting state of M is the ordered pair of the starting states of M_1 and M_2.
- $F \equiv (F_1 \times Q_2) \cup (Q_1 \times F_2)$. Note that $F \equiv F_1 \times F_2$ is wrong! F is $\{(r_1, r_2) \mid r_1 \in F_1$ or $r_2 \in F_2\}$.

Regularity is Closed Under Intersection

- The intersection $L_1 \cap L_2$ of two regular languages L_1 and L_2 is also regular.
- As before, the proof proceeds by constructing a machine M to accept $L_1 \cap L_2$, given machines M_1 and M_2 that accept L_1 and L_2.

Regularity is Closed Under Complementation

- If L is a regular language, then so is the complement language $\overline{L}$, where $\overline{L} \equiv \Sigma^* \setminus L$.
- The proof is as before, considering the existence of a machine M that accepts L.
- Show the construction of a machine $\overline{M}$ that accepts $\overline{L}$, given a machine M that accepts L.

Exercises

Give NFAs to accept the following languages:

(5) The set of all strings with an even number of 0s (and possibly 1s as well), or exactly two 1s only.

(6) The language 0^*.

(7) The language given by the RE $0^*1 + 1^*0$.