

Non-deterministic Finite Automata (NFA)

NFA versus DFA

- In a DFA, at every state q , for every symbol a , there is a unique a -transition i.e. there is a unique q' such that $q \xrightarrow{a} q'$.
This is not necessarily so in an NFA. At any state, an NFA may have multiple a -transitions, or none.
- In a DFA, transition arrows are labelled by symbols from Σ ; in an NFA, they are labelled by symbols from $\Sigma \cup \{ \epsilon \}$. I.e. an NFA may have ϵ -transitions.
- We may think of the non-determinism as a kind of parallel computation wherein several processes can be running concurrently.
When the NFA splits to follow several choices, that corresponds to a process “forking” into several children, each proceeding separately. If at least one of these accepts, then the entire computation accepts.

Definition: NFA

A *nondeterministic finite automaton* (NFA) is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where

- (i) Q is a finite set of states
- (ii) Σ is a finite alphabet
- (iii) $q_0 \in Q$ is the start state
- (iv) $\delta : Q \times (\Sigma \cup \{\epsilon\}) \rightarrow \mathcal{P}(Q)$ is the transition function
- (v) $F \subseteq Q$ is the set of final states.

Note: $\mathcal{P}(Q) \stackrel{\text{def}}{=} \{X : X \subseteq Q\}$ is the *power set* of Q . Equivalently δ can be presented as a relation, i.e. a subset of $(Q \times (\Sigma \cup \{\epsilon\})) \times Q$.

For $a \in \Sigma \cup \{\epsilon\}$ we define $q \xrightarrow{a} q' \stackrel{\text{def}}{=} q' \in \delta(q, a)$.

Some definitions and notations

Fix an NFA $N = (Q, \Sigma, \delta, q_0, F)$.

$L(N)$, the *language accepted by N* , consists of all strings w over Σ satisfying

$q_0 \xRightarrow{w} q$ where q is a final state. Here $\cdot \xRightarrow{\quad} \cdot$ is defined by:

- $q \xRightarrow{\epsilon} q'$ iff $q = q'$ or there is a sequence $q \xrightarrow{\epsilon} \dots \xrightarrow{\epsilon} q'$ of one or more ϵ -transitions in N from q to q' .
- For $w = a_1 \dots a_{n+1}$ where each $a_i \in \Sigma$, $q \xRightarrow{w} q'$ iff there are $q_1, q'_1, \dots, q_{n+1}, q'_{n+1}$ (not necessarily all distinct) such that

$$q \xRightarrow{\epsilon} q_1 \xrightarrow{a_1} q'_1 \xRightarrow{\epsilon} q_2 \xrightarrow{a_2} q'_2 \xRightarrow{\epsilon} \dots q'_n \xRightarrow{\epsilon} q_{n+1} \xrightarrow{a_{n+1}} q'_{n+1} \xRightarrow{\epsilon} q'$$

Intuitively $q \xRightarrow{w} q'$ means:

“There is a sequence of transitions from q to q' in N in which the symbols in w occur in the correct order, but with 0 or more ϵ -transitions before or after each one”.

We shall sometimes write $\hat{\delta}(q, w) = \{q' \in Q : q \xRightarrow{w} q'\}$, for $w \in \Sigma^*$.

Note: In case N is a DFA, for any $q \in Q$ and $w \in \Sigma^*$, there is a *unique* q' such that $q \xRightarrow{w} q'$ (thus, by abuse of notation, we write $\hat{\delta}(q, w) = q'$).

Exercise. Writing $w = a_1 \cdots a_{n+1}$, we have $q \xRightarrow{w} q'$ is equivalent to: there exist q_1, \dots, q_n such that

$$q \xRightarrow{a_1} q_1 \xRightarrow{a_2} \cdots \xRightarrow{a_{n+1}} q'$$

Equivalence of NFAs and DFAs: The Subset Construction

Observation. Every DFA is an NFA!

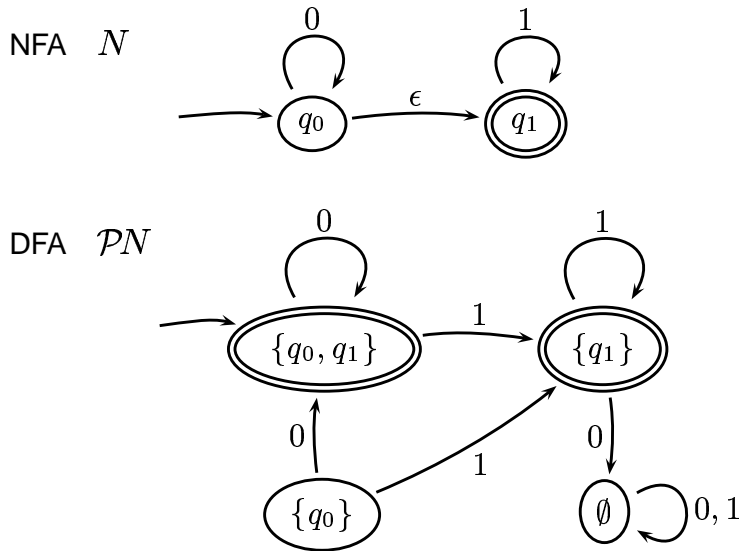
Say two automata are *equivalent* if they accept the same language.

Theorem(Determinization). Every NFA has an equivalent DFA.

Proof. Fix an NFA $N = (Q_N, \Sigma_N, \delta_N, q_N, F_N)$, we construct an equivalent DFA $\mathcal{P}N = (Q_{\mathcal{P}N}, \Sigma_{\mathcal{P}N}, \delta_{\mathcal{P}N}, q_{\mathcal{P}N}, F_{\mathcal{P}N})$ such that $L(N) = L(\mathcal{P}N)$:

- $Q_{\mathcal{P}N} \stackrel{\text{def}}{=} \{S : S \subseteq Q_N\}$
- $\Sigma_{\mathcal{P}N} \stackrel{\text{def}}{=} \Sigma_N$
- $S \xrightarrow{a} S'$ in $\mathcal{P}N$ iff $S' = \{q' : \exists q \in S. (q \xrightarrow{a} q' \text{ in } N)\}$
- $q_{\mathcal{P}N} \stackrel{\text{def}}{=} \{q : q_N \xrightarrow{\epsilon} q\}$
- $F_{\mathcal{P}N} \stackrel{\text{def}}{=} \{S \in Q_{\mathcal{P}N} : F_N \cap S \neq \emptyset\}$

Example. All words that begin with a string of 0's followed by a string of 1's.



Note. State $\{q_0\}$ is redundant.

Proof of " $L(N) \subseteq L(\mathcal{PN})$ ":

Suppose $\epsilon \in L(N)$. Then $q_N \xrightarrow{\epsilon} q'$ for some $q' \in F_N$. Hence $q' \in q_{\mathcal{PN}}$, and so, $q_{\mathcal{PN}} = \{q'' : q_N \xrightarrow{\epsilon} q''\} \in F_{\mathcal{PN}}$ i.e. $\epsilon \in L(\mathcal{PN})$.

Now take any non-null $u = a_1 \cdots a_n$. Suppose $u \in L(N)$. Then there is a sequence of N -transitions

$$q_N \xrightarrow{a_1} q_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} q_n \in F_N \quad (1)$$

Since \mathcal{PN} is deterministic, feeding a_1, \cdots, a_n to it results in the sequence of \mathcal{PN} -transitions

$$q_{\mathcal{PN}} \xrightarrow{a_1} S_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} S_n \quad (2)$$

where

$$\begin{aligned} S_1 &= \{q' : \exists q \in q_{\mathcal{PN}}.(q \xrightarrow{a_1} q' \text{ in } N)\} \\ S_2 &= \{q' : \exists q \in S_1.(q \xrightarrow{a_2} q' \text{ in } N)\} \\ &\vdots \end{aligned}$$

By definition of $\delta_{\mathcal{PN}}$, from (1), we have $q_1 \in S_1$, and so $q_2 \in S_2, \cdots$, and so

$q_n \in S_n$, and hence $S_n \in F_{\mathcal{PN}}$ because $q_n \in F_N$. Thus (2) shows that $u \in L(\mathcal{PN})$.

Proof of “ $L(\mathcal{PN}) \subseteq L(N)$ ”:

Suppose $\epsilon \in L(\mathcal{PN})$. Then $q_{\mathcal{PN}} \in F_{\mathcal{PN}}$ i.e. $F_N \cap \{q : q_N \xrightarrow{\epsilon} q\} \neq \emptyset$, or equivalently, for some $q' \in F_N$, $q_N \xrightarrow{\epsilon} q'$. Hence $\epsilon \in L(N)$.

Now suppose some non-null $u = a_1 \cdots a_n \in L(\mathcal{PN})$. I.e. there is a sequence of \mathcal{PN} -transitions of the form (2) with $S_n \in F_{\mathcal{PN}}$ i.e. with S_n containing some $q_n \in F_N$. Now since $q_n \in S_n$, by definition of $\delta_{\mathcal{PN}}$, there is some $q_{n-1} \in S_{n-1}$ with $q_{n-1} \xrightarrow{a_{n-1}} q_n$ in N . Working backwards in this way, we can build up a sequence of N -transitions like (1), until we deduce that $q_N \xrightarrow{a_1} q_1$. Thus we get a sequence of N -transitions with $q_n \in F_N$, and hence $u \in L(N)$.

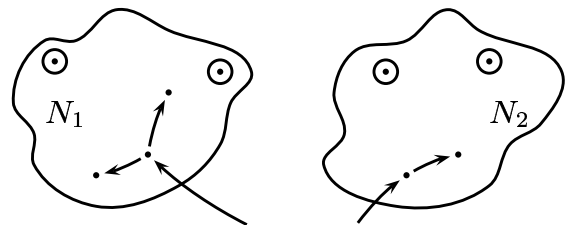
□

Closure under regular operations revisited

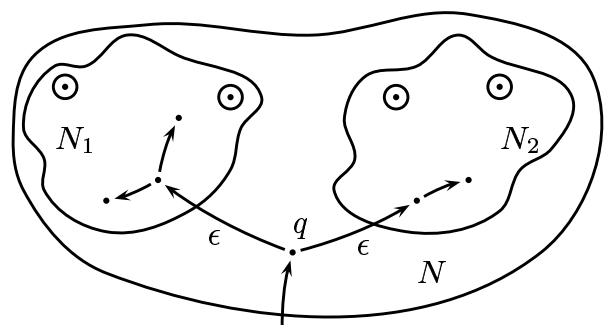
Using nondeterminism makes *some* proofs much easier.

Theorem. Regular languages are closed under union.

Take NFAs N_1 and N_2 .

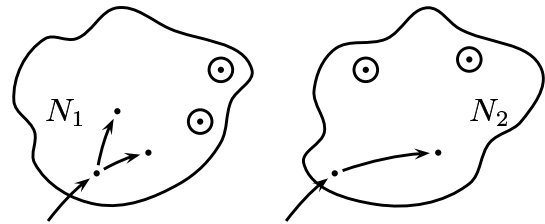


Define N that accepts $L(N_1) \cup L(N_2)$ by adding a new start state q to the disjoint union of (the respective state transition graphs of) N_1 and N_2 , and a ϵ -transition from q to each start state of N_1 and N_2 .

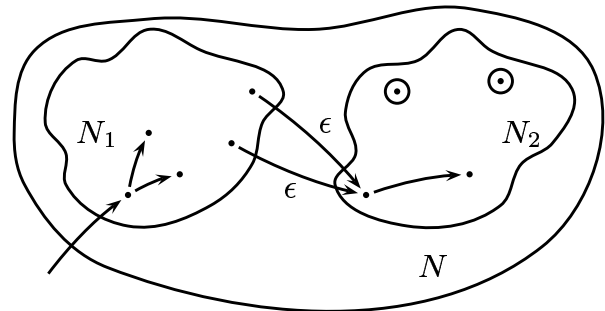


Theorem. Regular languages are closed under concatenation.

Take NFAs N_1 and N_2 .



An NFA N that accepts $L(N_1) \cdot L(N_2)$ can be obtained from the disjoint union of N_1 and N_2 by making the start state of N_1 the start state of N , and by adding an ϵ -transition from each accepting state of N_1 to the start state of N_2 . The accepting states of N are those of N_2 .

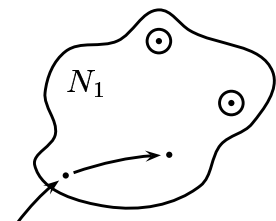


Theorem. Regular languages are closed under star.

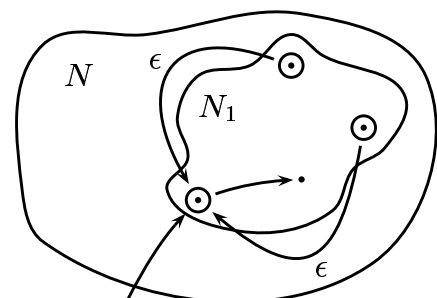
First attempt:

Take an NFA $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ that accepts A_1 . Construct N that accepts

$$A_1^* = \{\epsilon\} \cup A_1 \cup A_1 A_1 \cup \dots$$

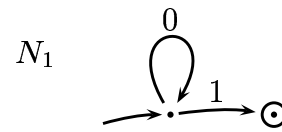


Obtain N from N_1 by making the start state accepting, and by adding a new ϵ -transition from each accepting state to the start state.

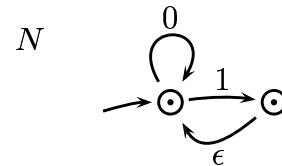


What is wrong with this?

Consider the two-node two-edge NFA N_1
 that accepts $\{0^i 1 : i \geq 0\}$
 (namely the language
 defined by the regular expression 0^*1).

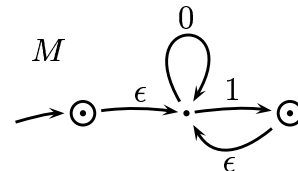


The above construction gives the NFA N



But N accepts $(0 + 0^*1)^* = (0 + 1)^* \neq (0^*1)^*$.
 E.g. N accepts 010 which is not in $L(N_1)^*$

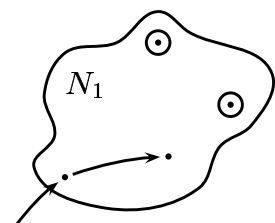
The NFA M accepts $L(N_1)^*$



Proof: Regular languages are closed under star

Second (correct) attempt:

Take an NFA $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ that accepts A_1 .



Define $N = (Q_1 \cup \{q_0\}, \Sigma, \delta, q_0, F_1 \cup \{q_0\})$
 where

$$\delta(q, a) = \begin{cases} \delta_1(q, a) & q \in Q_1 \text{ and } q \notin F_1 \\ \delta_1(q, a) & q \in F_1 \text{ and } a \neq \epsilon \\ \delta_1(q, a) \cup \{q_1\} & q \in F_1 \text{ and } a = \epsilon \\ \{q_1\} & q = q_0 \text{ and } a = \epsilon \\ \emptyset & q = q_0 \text{ and } a \neq \epsilon \end{cases}$$

