Finite Automata With \( \varepsilon \)Transitions

- Allow \( \varepsilon \) to be a label on arcs.
- Nothing else changes: acceptance of \( w \) is still the existence of a path from the start state to an accepting state with label \( w \).
- But \( \varepsilon \) can appear on arcs, and means the empty string (i.e., no visible contribution to \( w \)).
- 001 is accepted by the path \( q, s, r, q, r, s \), with label \( 0\varepsilon01\varepsilon = 001 \).

Elimination of \( \varepsilon \)Transitions

- \( \varepsilon \) transitions are a convenience, but do not increase the power of FA’s. To eliminate \( \varepsilon \) transitions:
  1. Compute the transitive closure of the \( \varepsilon \) arcs only.
  2. If a state \( p \) can reach state \( q \) by \( \varepsilon \) arcs, and there is a transition from \( q \) to \( r \) on input \( a \) (not \( \varepsilon \)), then add a transition from \( p \) to \( r \) on input \( a \).
  3. Make state \( p \) an accepting state if \( p \) can reach some accepting state \( q \) by \( \varepsilon \) arcs.
  4. Remove all \( \varepsilon \) transitions.
We have proved by construction that

**Theorem.** Every NFA has an equivalent DFA.

**Definition.** A language is regular if some DFA recognizes it.

**Corollary.** A language is regular if and only if some NFA recognizes it.

---

We have seen that for any NFA there exists an equivalent DFA. Hence

**A language is regular if and only if some NFA recognizes it.**

We will show today that regular languages are closed under regular operations.

### Regular Operations (again)

- Let $L_1$ and $L_2$ be languages. We defined the regular operations **union**, **concatenation**, and **star** as follows.

  - **Union:** $L_1 \cup L_2 = \{ w : w \in L_1 \text{ or } w \in L_2 \}$.
  - **Concatenation:** $L_1 \cdot L_2 = \{ vw : w \in L_1 \text{ and } v \in L_2 \}$.
  - **Star:** $L_1^* = \{ w_1 w_2 \ldots w_k : k \geq 0 \text{ and each } w_i \in L_1 \}$.

- Example: Let the alphabet $\Sigma$ be the standard 26 letters {a,b,…,z}.
  - If $L_1=$\{good, bad\} and $L_2=$\{boy, girl\}, then
    
    - $L_1 \cup L_2 =$\{good, bad, boy, girl\}.
    - $L_1 \cdot L_2 =$\{goodboy, badboy, goodgirl, badgirl\}.
    - $L_1^* =$\{ε, good, bad, goodgood, badgood, badbad, goodbad, goodgoodgood, goodgoodbad, goodbadbad, …\}
Th.1 The class of regular languages is closed under the union operation.

- We have regular languages $L_1$ and $L_2$ and want to prove that $L_1 \cup L_2$ is regular.
- The idea is to take two NFAs $N_1$ and $N_2$ for $L_1$ and $L_2$, and combine them into one new NFA $N$.
- $N$ must accept its input if either $N_1$ or $N_2$ accepts this input.
- $N$ will have a new state that branches to the start states of the old machines $N_1, N_2$ with $\varepsilon$ arrows.
- In this way $N$ non-deterministically guesses which of the two machines accepts the input.
- If one of them accepts the input then $N$ will accept it, too.

$$N_1 = (Q_1, \Sigma, \delta_1, q_{10}, F_1) \text{ recognizes } L_1$$

$$N_2 = (Q_2, \Sigma, \delta_2, q_{20}, F_2) \text{ recognizes } L_2$$

$$Q = (Q_1 \cup Q_2 \cup \{q_0\}, \delta, q_0, F) \text{ is the start state}$$

$$\delta(q_0, a) = \begin{cases} \delta_1(q, a) & q \in Q_1 \\ \delta_2(q, a) & q \in Q_2 \\ \{q_1, q_2\} & q = q_0 \text{ and } a = \varepsilon \\ \emptyset & q = q_0 \text{ and } a \neq \varepsilon \end{cases}$$

$$N = (Q, \Sigma, \delta, q_0, F) \text{ recognizes } L_1 \cup L_2$$

Th.2 The class of regular languages is closed under the concatenation operation.

- We have regular languages $L_1$ and $L_2$ and want to prove that $L_1 \cdot L_2$ is regular.
- The idea is to take two NFAs $N_1$ and $N_2$ for $L_1$ and $L_2$, and combine them into a new NFA $N$.
- $N$ accepts when the input can be split into two parts, the first accepted by $N_1$ and the second by $N_2$.
- We can think of $N$ as non-deterministically guessing where to make the split.

$$N_1 = (Q_1, \Sigma, \delta_1, q_{10}, F_1) \text{ recognizes } L_1$$

$$N_2 = (Q_2, \Sigma, \delta_2, q_{20}, F_2) \text{ recognizes } L_2$$

$$Q = (Q_1 \cup Q_2, \delta, q_{10}, F) \text{ is the set of final states}$$

$$q_{10} \text{ is the start state}$$

$$\delta(q, a) = \begin{cases} \delta_1(q, a) & q \in Q_1, q \notin F_1 \\ \delta_1(q, a) & q \in F_1, a \neq \varepsilon \\ \delta_2(q, a) \cup \{q_2\} & q \in F_1, a = \varepsilon \\ \delta_2(q, a) & q \in Q_2 \end{cases}$$

$$N = (Q, \Sigma, \delta, q_{10}, F) \text{ recognizes } L_1 \cdot L_2$$
Th.3 The class of regular languages is closed under the star operation.

- We have regular language $L_1$ and want to prove that $L_1^*$ is regular.
- We take an NFA $N_1$ for $L_1$, and modify it to recognize $L_1^*$.
- The resulting NFA $N$ accepts its input if it can be broken into several pieces and $N_1$ accepts each piece.
- $N$ is like $N_1$ with additional $\varepsilon$ arrows returning to the start state from the accept state.
- In addition we must modify $N$ so that it accepts $\varepsilon$, which always is a member of $L_1^*$.

$N_1 = (Q_1, \Sigma, \delta_1, q_0, F_1)$ recognizes $L_1$

$N = (Q, \Sigma, \delta, q_0, F)$ recognizes $L_1^*$

$Q = \{q_0\} \cup Q_1$

$F = \{q_0\} \cup F_1$

$q_0$ is the start state

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