NFA with ϵ-transitions

null string

Extension of NFAs where transitions labelled with symbol ϵ are allowed; this means that the automaton can change state **without consuming** any of its input

They accept all and only the regular languages

Easier to design than NFAs

ϵ-NFA widely used in compilers and for search of patterns in a text

Example

A fractional number consists of

- $+$ or - sign, optional
- a first string of digits
- one decimal point
- a second string of digits

(XOR)

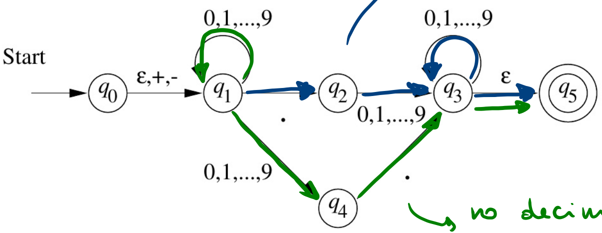
with the first or the second strings **optional**, but not both

This example comes from a lexical analyser in compiler theory

Example

ϵ-NFA accepting fractional numbers

no integer part

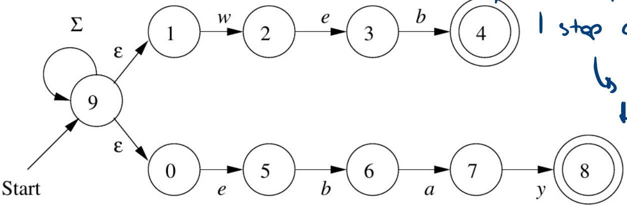


no decimal part

The ϵ-transition makes operators + and - optional

Example

ϵ-NFA accepting set of keywords $\{\texttt{ebay}, \texttt{web}\}$



once i get to this final state I stop computation
↳ not used in this course

The ϵ-transition makes it easy to combine several automata

NFA with ϵ-transitions

A **nondeterministic finite automaton with ϵ-transitions** (ϵ-NFA) is a 5-tuple

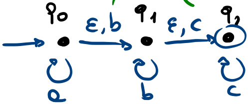
$$A = (Q, \Sigma, \delta, q_0, F)$$

where

- Q, Σ, q_0, and F are defined as for NFAs
- δ is a **transition** function $Q \times (\Sigma \cup \{\epsilon\}) \to 2^Q$, with 2^Q denoting the class of subsets of Q

Test

Specify an ϵ-NFA accepting the language of strings over $\{a, b, c\}$ with zero or more a's, followed by zero or more b's, followed by zero or more c's



b and c are not needed here, it just makes it faster, not "more right"

ϵ-closure

Let us compute the **ϵ-closure** of a state q, written $\text{ECLOSE}(q)$, adding all the states reachable from q itself through a sequence of one or more symbols ϵ

Needed later in the definition of $\hat{\delta}$ function

Base $q \in \text{ECLOSE}(q)$

Induction $(p \in \text{ECLOSE}(q) \wedge r \in \delta(p, \epsilon)) \Rightarrow r \in \text{ECLOSE}(q)$

Extension to set of states S

$$\text{ECLOSE}(S) = \bigcup_{q \in S} \text{ECLOSE}(q)$$

Extended transition function $\hat{\delta}$

Base $\hat{\delta}(q, \epsilon) = \text{ECLOSE}(q)$

Induction $\hat{\delta}(q, xa)$ is computed as

- $\{p_1, \ldots, p_k\} = \hat{\delta}(q, x)$ → states I am in after reading x
- $\{r_1, \ldots, r_m\} = \bigcup_{i=1}^{k} \delta(p_i, a)$ → states I am in after reading xa
- $\hat{\delta}(q, xa) = \text{ECLOSE}(\{r_1, \ldots, r_m\})$ → close (ε) all those states

Note that processing of ϵ symbols is accounted for after the processing of each symbol in Σ

Accepted language for ϵ-NFA

The language **accepted** by ϵ-NFA $E = (Q, \Sigma, \delta, q_0, F)$ is

$$L(E) = \{w \mid \hat{\delta}(q_0, w) \cap F \neq \emptyset\}$$

From ϵ-NFA to DFA

Given the ϵ-NFA

$$E = (Q_E, \Sigma, \delta_E, q_0, F_E)$$

we construct a DFA

$$D = (Q_D, \Sigma, \delta_D, q_D, F_D)$$

such that $L(D) = L(E)$

Construction details :

- $Q_D = \{S \mid S \subseteq Q_E,\ S = \mathrm{ECLOSE}(S)\}$
- $q_D = \mathrm{ECLOSE}(q_0)$
- $F_D = \{S \mid S \in Q_D,\ S \cap F_E \neq \emptyset\}$

From ϵ-NFA to DFA

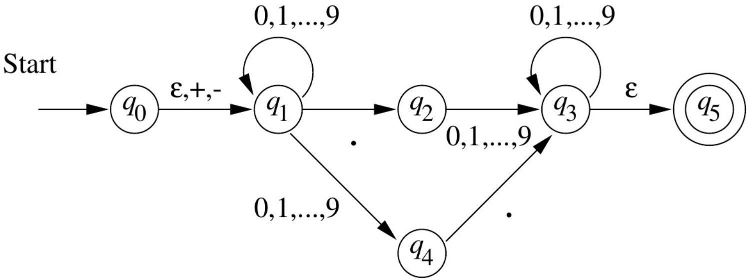
Construction details (cont'd)

Computation of $\delta_D(S, a)$, $a \in \Sigma$ and $S \in Q_D$

- $S = \{p_1, \ldots, p_k\}$
- $\{r_1, \ldots, r_m\} = \bigcup_{i=1}^{k} \delta_E(p_i, a)$
- $\delta_D(S, a) = \text{ECLOSE}(\{r_1, \ldots, r_m\})$

Example

ϵ-NFA E



Example

Computation of some of the values of δ_D

- $\delta_D(\{q_0, q_1\}, +) = \text{ECLOSE}(\delta_E(q_0, +) \cup \delta_E(q_1, +)) = \text{ECLOSE}(\{q_1\}) = \{q_1\}$
- $\delta_D(\{q_1\}, 0) = \text{ECLOSE}(\delta_E(q_1, 0)) = \text{ECLOSE}(\{q_1, q_4\}) = \{q_1, q_4\}$
- $\delta_D(\{q_1, q_4\}, .) = \text{ECLOSE}(\delta_E(q_1, .) \cup \delta_E(q_4, .)) = \text{ECLOSE}(\{q_2, q_3\}) = \{q_2, q_3, q_5\}$
- $\delta_D(\{q_2, q_3, q_5\}, 0) = \text{ECLOSE}(\delta_E(q_2, 0) \cup \delta_E(q_3, 0) \cup \delta_E(q_5, 0)) = \text{ECLOSE}(\{q_3\} \cup \{q_3\} \cup \emptyset) = \text{ECLOSE}(\{q_3\}) = \{q_3, q_5\}$
- $\cdots$

Example

DFA D constructed from E; the DFA has been further simplified, omitting the trap state and all transitions leading to that state