

CM0081 Formal Languages and Automata

§ 2.5 Finite Automata with Epsilon-Transitions

Andrés Sicard-Ramírez

Universidad EAFIT

Semester 2024-1

Preliminaries

Conventions

- ▶ The number and page numbers assigned to chapters, examples, exercises, figures, quotes, sections and theorems on these slides correspond to the numbers assigned in the textbook [Hopcroft, Motwani and Ullman (1979) 2007].
- ▶ The natural numbers include the zero, that is, $\mathbb{N} = \{0, 1, 2, \dots\}$.
- ▶ The power set of a set A , that is, the set of its subsets, is denoted by $\mathcal{P}A$.

Finite Automata with Epsilon-Transitions

Introduction

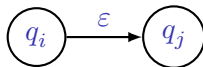
- ▶ An automaton with ε -transitions is a NFA where transition with the empty string are allowed.



Finite Automata with Epsilon-Transitions

Introduction

- ▶ An automaton with ϵ -transitions is a NFA where transition with the empty string are allowed.

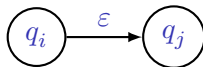


- ▶ The ϵ -transitions do **not** increase the computational power (or expressive power) of finite automata.

Finite Automata with Epsilon-Transitions

Introduction

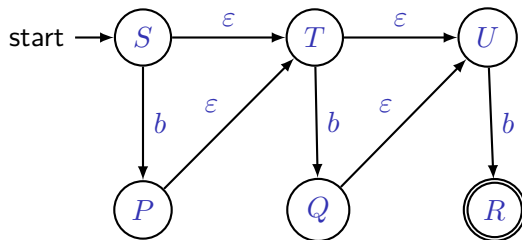
- ▶ An automaton with ϵ -transitions is a NFA where transition with the empty string are allowed.



- ▶ The ϵ -transitions do **not** increase the computational power (or expressive power) of finite automata.
- ▶ The ϵ -transitions facilitate the design of the automata.

Finite Automata with Epsilon-Transitions

Example ([Kozen (1997) 2012])



The automaton accepts the language $\{b, bb, bbb\}$.

Finite Automata with Epsilon-Transitions

Example

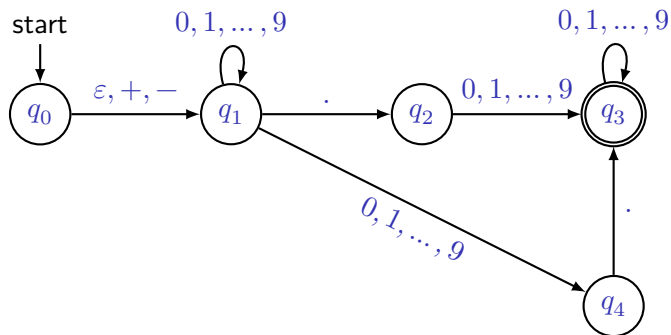
A finite automaton with epsilon-transitions that accepts decimal numbers consisting of:

- (i) an optional $+$ or $-$ sign,
- (ii) a string of digits,
- (iii) a decimal point and
- (iv) another string of digits. Either this string of digits, or the string (ii) can be empty, but at least one of the two strings must be non-empty.

(continued on next slide)

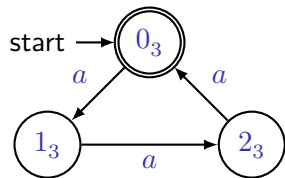
Finite Automata with Epsilon-Transitions

Example (continuation)

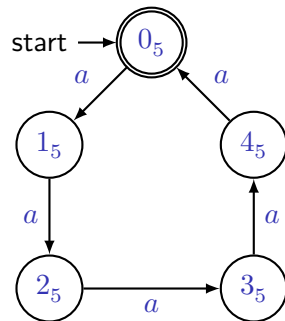


Finite Automata with Epsilon-Transitions

Example



$\{ w \in \{a\}^* : |w| \text{ is divisible by } 3 \}$

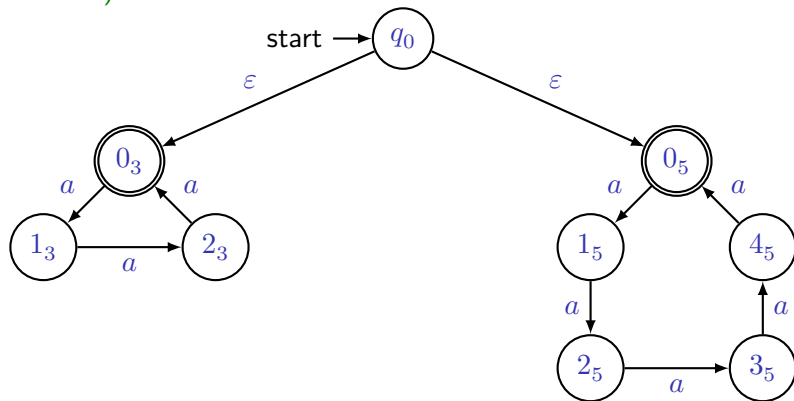


$\{ w \in \{a\}^* : |w| \text{ is divisible by } 5 \}$

(continued on next slide)

Finite Automata with Epsilon-Transitions

Example (continuation)



$$\{ w \in \{a\}^* : |w| \text{ is divisible by } 3 \text{ or by } 5 \}$$

Finite Automata with Epsilon-Transitions

Definition

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

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

where

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

Epsilon-Closures

Definition

The **ε -closure** of a state q , denoted by $\text{eclose} : Q \rightarrow \mathcal{P}Q$, are all states reachable from q by a sequence $\varepsilon \cdots \varepsilon$. We recursively define eclose by:

$$\frac{}{q \in \text{eclose}(q) \text{ if } q \in Q} \qquad \frac{p \in \text{eclose}(q) \quad \delta(p, \varepsilon) = r}{r \in \text{eclose}(q)}$$

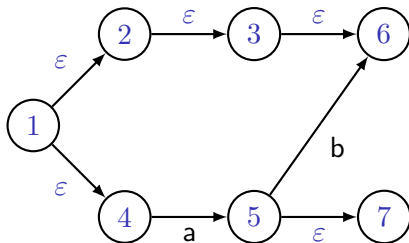
Epsilon-Closures

Definition

The ϵ -**closure** of a state q , denoted by $\text{eclose} : Q \rightarrow \mathcal{P}Q$, are all states reachable from q by a sequence $\epsilon \cdots \epsilon$. We recursively define eclose by:

$$\frac{}{q \in \text{eclose}(q) \text{ if } q \in Q} \qquad \frac{p \in \text{eclose}(q) \quad \delta(p, \epsilon) = r}{r \in \text{eclose}(q)}$$

Example



Epsilon-Closures

Definition

Let S be a set of states. The ϵ -**closure** of S is defined by

$$\text{eclose}(S) := \bigcup_{q \in S} \text{eclose}(q).$$

Extension of the Transition Function for ε -NFAs

Definition

Let $E = (Q, \Sigma, \delta, q_0, F)$ be an ε -NFA. The **extension of the transition function**, denoted by $\hat{\delta}$, is recursively defined by

$$\begin{aligned}\hat{\delta} : Q \times \Sigma^* &\rightarrow \mathcal{P}Q \\ \hat{\delta}(q, \varepsilon) &= \text{eclose}(q), \\ \hat{\delta}(q, xa) &= \text{eclose}(\{r_1, r_2, \dots, r_m\}),\end{aligned}$$

where

$$\begin{aligned}\hat{\delta}(q, x) &= \{p_1, p_2, \dots, p_k\}, \\ \bigcup_{i=1}^k \delta(p_i, a) &= \{r_1, r_2, \dots, r_m\}.\end{aligned}$$

Languages Accepted by ε -NFAs

Recall

Let $D = (Q, \Sigma, \delta, q_0, F)$ and $N = (Q, \Sigma, \delta, q_0, F)$ be a DFA and a NFA, respectively. Recall that the languages accepted by D and N were defined by

$$L(D) := \{ w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in F \},$$

$$L(N) := \{ w \in \Sigma^* \mid \hat{\delta}(q_0, w) \cap F \neq \emptyset \}.$$

Languages Accepted by ε -NFAs

Definitions

Let $E = (Q, \Sigma, \delta, q_0, F)$ be a ε -NFA and let $w \in \Sigma^*$ be a string.

- (i) The string w is **accepted** by E iff $\hat{\delta}(q_0, w) \cap F \neq \emptyset$.

Languages Accepted by ε -NFAs

Definitions

Let $E = (Q, \Sigma, \delta, q_0, F)$ be a ε -NFA and let $w \in \Sigma^*$ be a string.

- (i) The string w is **accepted** by E iff $\hat{\delta}(q_0, w) \cap F \neq \emptyset$.
- (ii) The string w is **rejected** by E iff $\hat{\delta}(q_0, w) \cap F = \emptyset$.

Languages Accepted by ε -NFAs

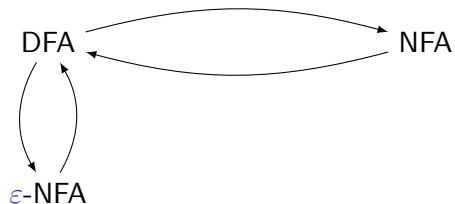
Definitions

Let $E = (Q, \Sigma, \delta, q_0, F)$ be a ε -NFA and let $w \in \Sigma^*$ be a string.

- (i) The string w is **accepted** by E iff $\hat{\delta}(q_0, w) \cap F \neq \emptyset$.
- (ii) The string w is **rejected** by E iff $\hat{\delta}(q_0, w) \cap F = \emptyset$.
- (iii) The **language accepted** by E , denoted $L(E)$, is the set of strings accepted by E , that is,

$$L(E) := \{ w \in \Sigma^* \mid \hat{\delta}(q_0, w) \cap F \neq \emptyset \}.$$

Equivalence of DFAs, NFAs and ϵ -NFAs



Regular Languages

Definition



A language L is **regular** iff exists a finite automaton A (DFA, NFA or ε -NFA) such that $L = L(A)$.

Regular Languages

Exercise 2.5.3.a

Let L be the set of strings consisting of zero or more a 's followed by zero or more b 's, followed by zero or more c 's. Prove that L is a regular language.

References

-  Hopcroft, J. E., Motwani, R. and Ullman, J. D. [1979] (2007). Introduction to Automata Theory, Languages, and Computation. 3rd ed. Pearson Education (cit. on p. 2).
-  Kozen, D. C. [1997] (2012). Automata and Computability. Third printing. Undergraduate Texts in Computer Science. Springer. DOI: [10.1007/978-1-4612-1844-9](https://doi.org/10.1007/978-1-4612-1844-9) (cit. on p. 6).