Introduction to the Theory of Computation

XU Ming

School of Software Engineering, East China Normal University

March 10, 2025

OUTLINE

- Finite Automata with ϵ-Transition
- Regular Expressions

Finite Automata with ϵ-Transition

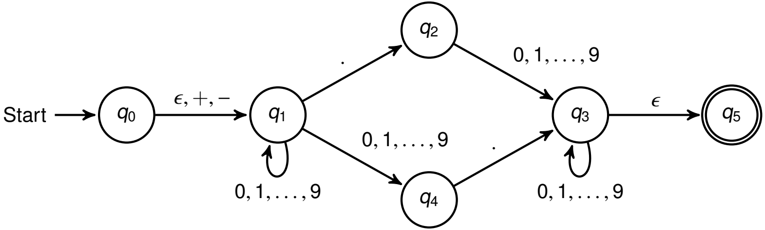
Use of ϵ-Transition

In order to add “programming convenience”, we will extend NFA to ϵ-NFA. In transition diagrams of such NFA, the empty string ϵ is allowed as a label.

However, this new capability does not expand the class of language that can be accepted by finite automata.

Example

Design an automaton A_3 accepting decimal numbers consisting of (1) an optional $+$ or – sign, (2) a string of digits, (3) a decimal point, and (4) another strings of digits. One of the strings (2) and (4) is optional, but not both empty.



Definition of ϵ-NFA

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

- Q is a finite set of states,
- Σ is a finite set of input symbols (an alphabet),
- δ is a transition function from $Q \times (\Sigma \cup \{\epsilon\})$ to the powerset of Q,
- $q_0 \in Q$ is a start state,
- $F \subseteq Q$ is a set of final or accepting states.

Example

The NFA A_3 is an ϵ-NFA, which can be represented as

$$A_3 = (\{q_0, q_1, q_2, q_3, q_4, q_5\}, \{., +, -, 0, \ldots, 9\}, \delta, q_0, \{q_5\})$$

where the transition table for δ is

	ϵ	+,-	.	$1, \ldots, 9$
$\rightarrow q_0$	$\{q_1\}$	$\{q_1\}$	$\emptyset$	$\emptyset$
q_1	$\emptyset$	$\emptyset$	$\{q_2\}$	$\{q_1, q_4\}$
q_2	$\emptyset$	$\emptyset$	$\emptyset$	$\{q_3\}$
q_3	$\{q_5\}$	$\emptyset$	$\emptyset$	$\{q_3\}$
q_4	$\emptyset$	$\emptyset$	$\{q_3\}$	$\emptyset$
$\bigstar q_5$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

ϵ-Closures

Informally, the ϵ-closures of a state q is the set consisting of q and other states by following all transitions out of q that are labeled ϵ.

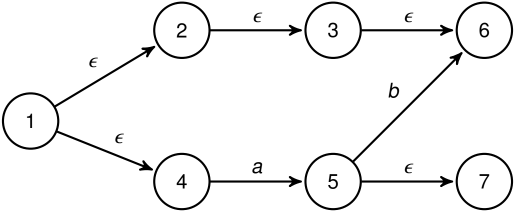
The recursive definition of the ϵ-closures $\mathrm{ECLOSE}(q)$ is given below.

Basis step: $q \in \mathrm{ECLOSE}(q)$.

Inductive step: If $p \in \mathrm{ECLOSE}(q)$, then $\delta(p, \epsilon) \subseteq \mathrm{ECLOSE}(q)$.

Example

For the collection of states, which may be part of some ϵ-NFA, construct ECLOSE(1) and ECLOSE(5).



Solution $\quad \mathrm{ECLOSE}(1) = \{1, 2, 3, 4, 6\}$, $\mathrm{ECLOSE}(5) = \{5, 7\}$.

Extended Transition Function

The transition function δ of an ϵ-NFA can be extended to $\hat{\delta}$:

Basis step: $\hat{\delta}(q, \epsilon) = \text{ECLOSE}(q)$.

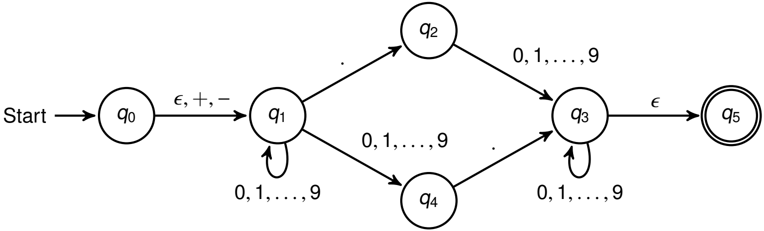
Inductive step: Suppose $w = xa$, then

$$\hat{\delta}(q, w) = \bigcup_{r \in \delta(\hat{\delta}(q,x),a)} \text{ECLOSE}(r)$$

where $\delta(S, a) = \bigcup_{p \in S} \delta(p, a)$.

Example

Compute $\hat{\delta}(q_0, 5.6)$ for the ϵ-NFA A_3.



- $\hat{\delta}(q_0, \epsilon) = \text{ECLOSE}(q_0) = \{q_0, q_1\}$.
- Since $\delta(q_0, 5) \cup \delta(q_1, 5) = \emptyset \cup \{q_1, q_4\} = \{q_1, q_4\}$,
  $\hat{\delta}(q_0, 5) = \text{ECLOSE}(q_1) \cup \text{ECLOSE}(q_4) = \{q_1, q_4\}$.
- Since $\delta(q_1, .) \cup \delta(q_4, .) = \{q_2, q_3\}$,
  $\hat{\delta}(q_0, 5.) = \text{ECLOSE}(q_2) \cup \text{ECLOSE}(q_3) = \{q_2, q_3, q_5\}$.
- Since $\delta(q_2, 6) \cup \delta(q_3, 6) \cup \delta(q_5, 6) = \{q_3\}$,
  $\hat{\delta}(q_0, 5.6) = \text{ECLOSE}(q_3) = \{q_3, q_5\}$.

Language of ϵ-NFA

The language of an ϵ-NFA $A = (Q, \Sigma, \delta, q_0, F)$ is defined by

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

Example

For $A_3 = (\{q_0, q_1, \ldots, q_5\}, \{., +, -, 0, \ldots, 9\}, \delta, q_0, \{q_5\})$, the string 5.6 is in the language of A_3, since $\hat{\delta}(q_0, 5.6)$ contains the accepting state q_5.

Equivalence of DFA and ϵ-NFA

Given any ϵ-NFA E, we can find a DFA D that accepts the same language as E. The construction is very close to the subset construction, as the states of D are subsets of the states of E.

Let ϵ-NFA $E = (Q_E, \Sigma, \delta_E, q_0, F_E)$, we will construct an equivalent DFA $D = (Q_D, \Sigma, \delta_D, q_D, F_D)$.

Here is the detail of the construction.

- $Q_D = \{S \mid S \subseteq Q_E, S = \text{ECLOSE}(S)\}$
- $q_D = \text{ECLOSE}(q_0)$
- $F_D = \{S \mid S \in Q_D, S \cap F_E \neq \emptyset\}$
- For every $S \in Q_D$ and $a \in \Sigma$,

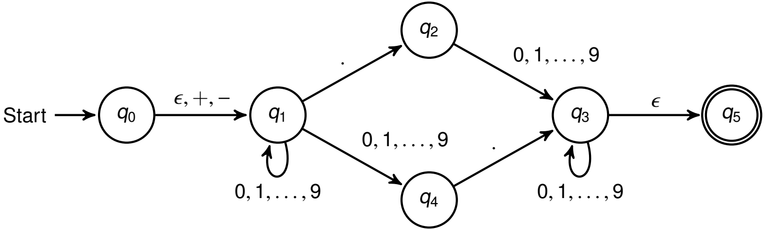
$$\delta_D(S, a) = \bigcup_{r \in \delta_E(S,a)} \text{ECLOSE}(r)$$

where $\delta_E(S, a) = \bigcup_{p \in S} \delta_E(p, a)$.

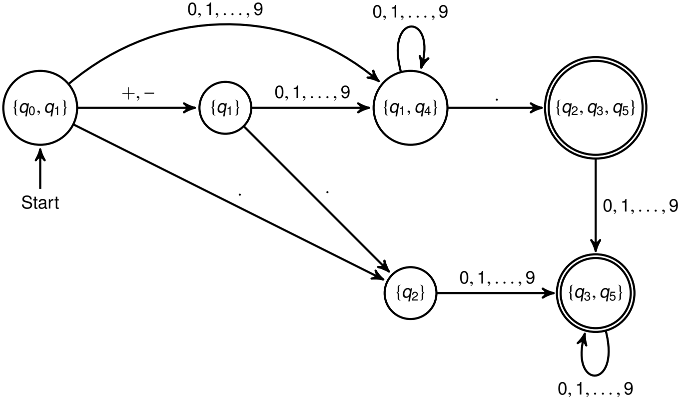
In general, $Q_D \neq 2^{Q_E}$.

Example

Eliminate ϵ-transition and construct an DFA from the ϵ-NFA A_3



The dead state $\emptyset$ and all transitions to it are omitted.



Theorem 2.3

A language L is accepted by some ϵ-NFA if and only if L is accepted by some DFA.

Proof (if) This direction is easy. Suppose $L = L(D)$ for some DFA. Turn D into an ϵ-NFA E by adding transition $\delta_E(q, \epsilon) = \emptyset$ for all states q of D.
(only-if) Let $E = (Q_E, \Sigma, \delta_E, q_0, F_E)$ is an ϵ-NFA. Apply the modified subset construction described above to produce the DFA $D = (Q_D, \Sigma, \delta_D, q_D, F_D)$. We show

$$\hat{\delta}_E(q_0, w) = \hat{\delta}_D(q_D, w)$$

by induction on $|w|$.

Basis step: $\hat{\delta}_E(q_0, \epsilon) = \text{ECLOSE}(q_0) = q_D = \hat{\delta}_D(q_D, \epsilon)$.

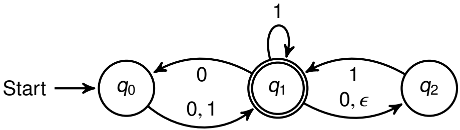
Inductive step:

$$
\begin{aligned}
\hat{\delta}_E(q_0, xa) &= \bigcup_{p \in \delta_E(\hat{\delta}_E(q_0, x), a)} \text{ECLOSE}(p) && \text{(definition of } \hat{\delta}_E\text{)} \\
&= \bigcup_{p \in \delta_E(\hat{\delta}_D(q_D, x), a)} \text{ECLOSE}(p) && \text{(induction hypothesis)} \\
&= \delta_D(\hat{\delta}_D(q_D, x), a) && \text{(modified subset construction)} \\
&= \hat{\delta}_D(q_D, xa) && \text{(definition of } \hat{\delta}_D\text{)}
\end{aligned}
$$

□

Example

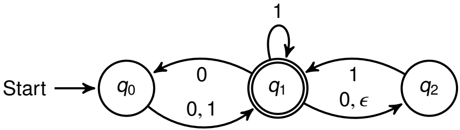
Convert the following ϵ-NFA into an equivalent DFA.



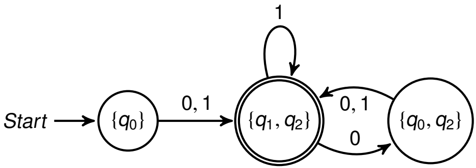
Solution

Example

Convert the following ϵ-NFA into an equivalent DFA.



Solution



Regular Expressions

Regular expressions denote languages, e.g.

$$\mathbf{01}^* + \mathbf{10}^*$$

- A regular expression is a "user-friendly", declarative way of describing a regular language.
- A FA (DFA or NFA) is a "blueprint" for constructing a machine recognizing a regular language.

Building Regular Expressions

Inductive definition of regular expression (RE) and its language.

Basis step:

- ϵ and $\emptyset$ are regular expression

$$L(\epsilon) = \{\epsilon\}, \quad L(\emptyset) = \emptyset.$$

- If $a \in \Sigma$, then **a** is a regular expression

$$L(\mathbf{a}) = \{a\}.$$

Inductive step:

- If E is a regular expression, then (E) is a regular expression

$$L((E)) = L(E).$$

- If E and F are regular expression, then $E + F$ is a regular expression

$$L(E + F) = L(E) \cup L(F).$$

- If E and F are regular expression, then $E.F$ is a regular expression

$$L(E.F) = L(E).L(F).$$

- If E is a regular expression, then E^* is a regular expression

$$L(E^*) = (L(E))^*.$$

Precedence of Regular Expression Operators

1. The star operator is of highest precedence.
2. Next in precedence comes the concatenation or dot operator.
3. Finally, all unions or plus operators are grouped with their operands.

Example

$\mathbf{01}^* + \mathbf{1}$ is grouped as $(\mathbf{0}(\mathbf{1}^*)) + \mathbf{1}$.

Languages Associated with Regular Expressions

Example

The expression $R = (\mathbf{aa})^*(\mathbf{bb})^*\mathbf{b}$ denoted the set of all strings with an even number of a's followed by an odd number of b's; that is

$$L(R) = \{a^{2n}b^{2m+1} \mid n \geq 0, m \geq 0\}.$$

Example

Exhibit the language $L(\mathbf{a}^*(\mathbf{a}+\mathbf{b}))$ in set notation.

Solution

Since $L(\mathbf{a}^*(\mathbf{a}+\mathbf{b})) = L(\mathbf{a}^*)(L(\mathbf{a}+\mathbf{b})) = (L(\mathbf{a}))^*(L(\mathbf{a}) \cup L(\mathbf{b}))$. *So the answer is* $\{a, aa, aaa, \ldots; b, ab, aab, \ldots\}$.

Languages Associated with Regular Expressions

Example

The expression $R = (\mathbf{aa})^*(\mathbf{bb})^*\mathbf{b}$ denoted the set of all strings with an even number of a's followed by an odd number of b's; that is

$$L(R) = \{a^{2n}b^{2m+1} \mid n \geq 0, m \geq 0\}.$$

Example

Exhibit the language $L(\mathbf{a}^*(\mathbf{a}+\mathbf{b}))$ in set notation.

Solution

Since $L(\mathbf{a}^*(\mathbf{a}+\mathbf{b})) = L(\mathbf{a}^*)(L(\mathbf{a}+\mathbf{b})) = (L(\mathbf{a}))^*(L(\mathbf{a}) \cup L(\mathbf{b}))$. *So the answer is* $\{a, aa, aaa, \ldots; b, ab, aab, \ldots\}$.

Languages Associated with Regular Expressions

Example

The expression $R = (\mathbf{aa})^*(\mathbf{bb})^*\mathbf{b}$ denoted the set of all strings with an even number of a's followed by an odd number of b's; that is

$$L(R) = \{a^{2n}b^{2m+1} \mid n \geq 0, m \geq 0\}.$$

Example

Exhibit the language $L(\mathbf{a}^*(\mathbf{a} + \mathbf{b}))$ in set notation.

Solution

Since $L(\mathbf{a}^*(\mathbf{a} + \mathbf{b})) = L(\mathbf{a}^*)(L(\mathbf{a} + \mathbf{b})) = (L(\mathbf{a}))^*(L(\mathbf{a}) \cup L(\mathbf{b}))$. *So the answer is* $\{a, aa, aaa, \ldots; b, ab, aab, \ldots\}$.

Going from an informal description or set notation to a regular expression tends to be a little harder.

Example

Write a regular expression for the set of strings that consist of alternating 0's and 1's.

Solution

$$(\mathbf{01})^* + (\mathbf{10})^* + \mathbf{1}(\mathbf{01})^* + \mathbf{0}(\mathbf{10})^*,$$

or equivalently

$$(\epsilon + \mathbf{1})(\mathbf{01})^*(\epsilon + \mathbf{0}).$$

Going from an informal description or set notation to a regular expression tends to be a little harder.

Example

Write a regular expression for the set of strings that consist of alternating 0's and 1's.

Solution

$$(\mathbf{01})^* + (\mathbf{10})^* + \mathbf{1}(\mathbf{01})^* + \mathbf{0}(\mathbf{10})^*,$$

or equivalently

$$(\epsilon + \mathbf{1})(\mathbf{01})^*(\epsilon + \mathbf{0}).$$

Example

Find a regular expression for the language
$L = \{w \in \{0,1\}^* \mid w$ has no pair of consecutive zeros$\}$.

Solution

One observation is that whenever a 0 occurs, it must be followed immediately by a 1. This suggests $(\mathbf{1}^*\mathbf{011}^*)^*$. However, the answer is still incomplete, since the strings ending in 0 or consisting of all 1's are unaccounted for.

So the correct answer is $(\mathbf{1}^*\mathbf{011}^*)^*(\mathbf{0}+\epsilon)+\mathbf{1}^*(\mathbf{0}+\epsilon)$.

Another shorter answer is $(\mathbf{1}+\mathbf{01})^*(\mathbf{0}+\epsilon)$.

Example

Find a regular expression for the language
$L = \{w \in \{0,1\}^* \mid w \text{ has no pair of consecutive zeros}\}$.

Solution

One observation is that whenever a 0 *occurs, it must be followed immediately by a* 1*. This suggests* $(\mathbf{1}^*\mathbf{011}^*)^*$*. However, the answer is still incomplete, since the strings ending in* 0 *or consisting of all* 1*'s are unaccounted for.*

So the correct answer is $(\mathbf{1}^*\mathbf{011}^*)^*(\mathbf{0}+\epsilon)+\mathbf{1}^*(\mathbf{0}+\epsilon)$.

Another shorter answer is $(\mathbf{1}+\mathbf{01})^*(\mathbf{0}+\epsilon)$.

Example

Find a regular expression that denotes all bit strings whose value, when interpreted as binary integer, is great than or equal to 40.

Solution

The bit string must be at least 6 bits long. If it is longer than 6 bits, its value is at least 64, so anything will do. If it is exactly 6 bits, then either the second bit from the left (16) or the third bit from the left (8) must be 1. So the solution is

$$(\mathbf{111}+\mathbf{110}+\mathbf{101})(\mathbf{0}+\mathbf{1})(\mathbf{0}+\mathbf{1})(\mathbf{0}+\mathbf{1})+$$
$$\mathbf{1}(\mathbf{0}+\mathbf{1})(\mathbf{0}+\mathbf{1})(\mathbf{0}+\mathbf{1})(\mathbf{0}+\mathbf{1})(\mathbf{0}+\mathbf{1})(\mathbf{0}+\mathbf{1})(\mathbf{0}+\mathbf{1})^*$$

Example

Find a regular expression that denotes all bit strings whose value, when interpreted as binary integer, is great than or equal to 40.

Solution

The bit string must be at least 6 bits long. If it is longer than 6 bits, its value is at least 64, so anything will do. If it is exactly 6 bits, then either the second bit from the left (16) or the third bit from the left (8) must be 1. So the solution is

$$(\mathbf{111} + \mathbf{110} + \mathbf{101})(\mathbf{0} + \mathbf{1})(\mathbf{0} + \mathbf{1})(\mathbf{0} + \mathbf{1}) + \mathbf{1}(\mathbf{0} + \mathbf{1})(\mathbf{0} + \mathbf{1})(\mathbf{0} + \mathbf{1})(\mathbf{0} + \mathbf{1})(\mathbf{0} + \mathbf{1})(\mathbf{0} + \mathbf{1})(\mathbf{0} + \mathbf{1})^*$$

Homework

Exercises 2.5.2 & 3.1.1