

Regular Expressions (RE)

Empty set	\mathbf{F}	A RE denotes the empty set
Empty string	$\mathbf{?}$	A RE denotes the set $\{?\}$
Symbol	\mathbf{a}	A RE denotes the set $\{a\}$
Alternation	$\mathbf{M + N}$	If M is a RE for the set M and N is a RE for the set N, then M+N is a RE for the set $M \cup N$
Concatenation	$\mathbf{M \cdot N}$	If M is a RE for the set M and N is a RE for the set N, then M.N is a RE for the set $M \cdot N$
Kleene-*	$\mathbf{M^*}$	If M is a RE for the set M, then M^* is a RE for the set M^*

Regular Expressions (RE)

Operation	Notation	Language	UNIX
Alternation	R_1+r_2	$L(r_1) \cup L(r_2)$	$r_1 r_2$
Concatenation	$r_1 \bullet r_2$	$L(r_1) \bullet L(r_2)$	$(r_1)(r_2)$
Kleene-*	r^*	$L(r)^*$	$(r)^*$
Kleene-+	r^+	$L(r)^+$	$(r)^+$
Exponentiation	r^n	$L(r)^n$	$(r)\{n\}$

Regular Expressions (RE)

Example

For the alphabet $S=\{0, 1\}$

$0+1$ is a RE denote the set $\{0\} \cup \{1\}$

0^* is a RE denote the set $\{0\}^* = \{?, 0, 00, \dots\}$

0.1^* is a RE denote the set $\{0\} \cdot \{?, 1, 11, \dots\} = \{0, 01, 011, \dots\}$

Regular Expressions (RE)

Notes

For a RE r , $r^i = r.r \dots r$ i -times

Operations precedence: $* > . > +$

So we can omit many parentheses, for example: the RE $((0(1^*)) + 0)$ can be written as 01^*+0

We me abbreviate rr^* to r^+

The corresponding set (language) denoted by a RE r will be expressed as $L(r)$

Nondeterministic Finite Automata (NFA)

Definition

A nondeterministic finite automaton (NFA) M is defined by a 5-tuple $M=(Q,S,d,q_0,F)$, with

- ☑ Q : finite set of states
- ☑ S : finite input alphabet
- ☑ d : transition function $d:Q \times S \rightarrow P(Q)$
- ☑ $q_0 \in Q$: start state
- ☑ $F \subseteq Q$: set of final states

Nondeterministic Finite Automata (NFA)

Definition

A string w is **accepted** by an NFA M if and only if *there exists* a path starting at q_0 which is labeled by w and ends in a final state.

The **language accepted by** an NFA M is the set of all strings which are accepted by M and is denoted by $L(M)$.
 $L(M) = \{w: d(q_0, w) \cap F \neq \emptyset\}$

Nondeterministic Finite Automata (NFA)

Definition

A nondeterministic finite automaton has transition rules like:

Nondeterministic transition

Nondeterministic Finite Automata (NFA)

Nondeterminism - Parallelism

For any string w , the nondeterministic automaton can be in a subset $\subseteq Q$ of several possible states.

If the final set contains a final state, then the automaton accepts the string.

"The automaton processes the input in a parallel fashion; its computational path is no longer a line, but more like a tree".

Nondeterministic Finite Automata (NFA)

We can write the NFA in two ways

1. State digraph

2. Table

δ	a	b
q0	{q0}	{q0, q1}
q1	ϕ	{q2}
q2	ϕ	ϕ

Nondeterministic Finite Automata (NFA)

Example 1

Write an NFA for the language, over $S=\{a,b\}$, ending in bb

$(a \cup b)^* bb$

$Q = \{q0, q1, q2\}$
 $\Sigma = \{a, b\}$
 $F = \{q2\}$

Check the input abb ?

Nondeterministic Finite Automata (NFA)

Quiz

Check the input abb ?

$(a \cup b)^* bb$

Input:

a	b	b
---	---	---

$q2$ is a final state hence the input abb is accepted

Nondeterministic Finite Automata (NFA)

Example 2

Write an NFA for the language, over $S=\{a,b\}$, $L=(a \cup b)^* bb(a \cup b)^*$

Nondeterministic Finite Automata (NFA)

Example 3

Write an NFA for the language, over $S=(a,b)$,
 $L=(a \cup b)^* (aa \cup bb) (a \cup b)^*$

Nondeterministic Finite Automata (NFA)

Example 4

What language is accepted by this NFA?

Answer: (a+b)*abb

Nondeterministic Finite Automata (NFA)

Example 5

For example, consider the following NFA which reads the input 11000.

Accepted!

NFA → DFA

⚡ **Theorem:** For every language L that is accepted by a nondeterministic finite automaton, there is a (deterministic) finite automaton that accepts L as well. FA and NFA are equivalent computational models.

⚡ **Proof idea:** When keeping track of a nondeterministic computation of an NFA N we use many 'fingers' to point at the subset $\subseteq Q$ of states of N that can be reached on a given input string. We can simulate this computation with a deterministic automaton M with state space $P(Q)$.

NFA → DFA

Proof

Let L be the language recognized by the NFA $N = (Q, S, d, q_0, F)$. Define the DFA $M = (Q', S, d', q'_0, F')$ by

- $Q' = P(Q)$
- $d'(R, a) = \{q \in Q \mid q \in d(r, a) \text{ for an } r \in R\}$
- $q'_0 = \{q_0\}$
- $F' = \{R \in Q' \mid R \text{ contains a 'final state' of } N\}$

⚡ It is easy to see that the previously described deterministic finite automaton M accepts the same language as N.

NFA → DFA

Example 1

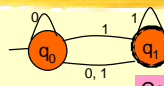
Convert the NFA: into a DFA?

Given NFA	⇒	Constructed DFA
$Q = \{q_0, q_1\}$	⇒	$Q' = P(Q) = \{F, \{q_0\}, \{q_1\}, \{q_0, q_1\}\}$
q_0	⇒	$q'_0 = \{q_0\}$
$F = \{q_1\}$	⇒	$F' = \{\{q_1\}, \{q_0, q_1\}\}$

For d' see the next slide

NFA → DFA

Example 1

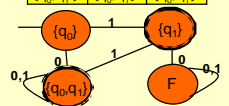
Convert the NFA:  into a DFA?

Given NFA

- $d(q_0, 0) = \{q_0, q_1\}$ $\implies d'(\{q_0\}, 0) = \{q_0, q_1\}$
- $d(q_0, 1) = \{q_1\}$ $\implies d'(\{q_0\}, 1) = \{q_1\}$
- $d(q_1, 0) = F$ $\implies d'(\{q_1\}, 0) = F$
- $d(q_1, 1) = \{q_0, q_1\}$ $\implies d'(\{q_1\}, 1) = \{q_0, q_1\}$

Constructed DFA

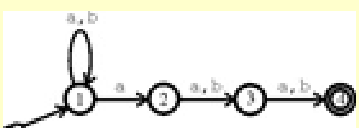
d'	0	1
F	F	F
$\{q_0\}$	$\{q_0, q_1\}$	$\{q_1\}$
$\{q_1\}$	F	$\{q_0, q_1\}$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_1\}$



NFA → DFA

Example 2

Start with the NFA:

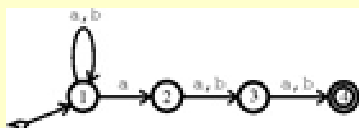


Q1: What's the accepted language?
Q2: How many states does the subset construction create in this case?

NFA → DFA

Example 2

A1: $L = \{x \in \{a,b\}^* \mid 3^{\text{rd}} \text{ bit of } x \text{ from right is } a\}$




A2: $16 = 2^4$ states.
That's a lot of states. Would be nice if only had to construct useful states, i.e. those that can be reached from start state.

NFA → DFA

Example 2

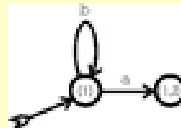
Start with $\{1\}$:



NFA → DFA

Example 2

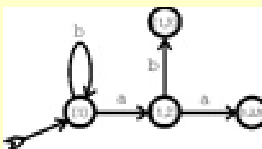
Branch out. Notice that $\delta(1, a) = \{1, 2\}$.



NFA → DFA

Example 2

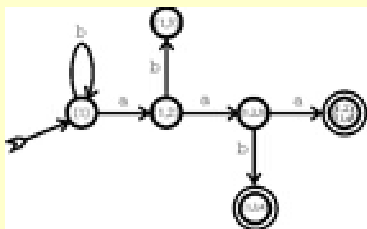
Branch out. Notice that $\delta'(\{1, 2\}, a) = \{1, 2, 3\}$.



NFA → DFA

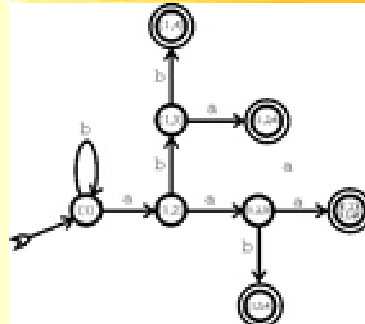
Example 2

Branch out. Note that $\delta^*(\{1,2,3\}, a) = \{1,2,3,4\}$



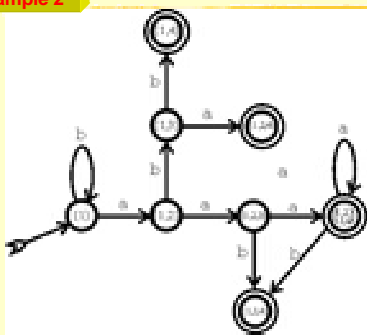
NFA = DFA

Example 2



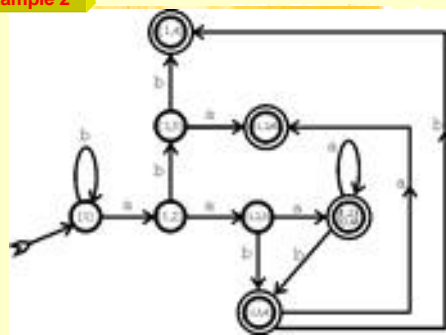
NFA → DFA

Example 2



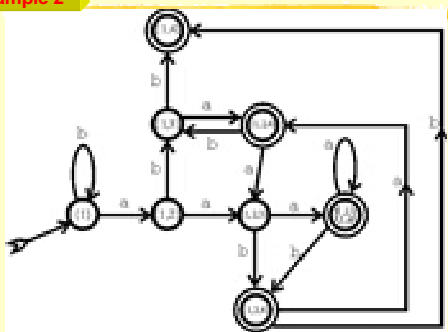
NFA → DFA

Example 2



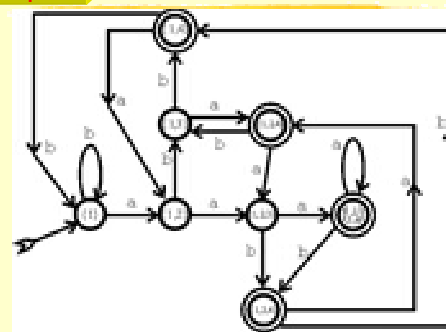
NFA → DFA

Example 2



NFA → DFA

Example 2



NFA → DFA

Example 2

Summarizing:

Therefore, we saved 50% effort by not constructing all possible states unthinkingly.

NFA → DFA

Exercise

Convert the following NFA into an equivalent DFA?

Nondeterministic Finite Automata with empty moves (?-NFA)

Definition

A nondeterministic finite automaton with empty moves (?-NFA) M is defined by a 5-tuple $M = (Q, S, d, q_0, F)$, with

- ☑ Q : finite set of states
- ☑ S : finite input alphabet
- ☑ d : transition function $d: Q \times (S \cup \{\epsilon\}) \rightarrow P(Q)$
- ☑ $q_0 \in Q$: start state
- ☑ $F \subseteq Q$: set of final states

Nondeterministic Finite Automata with empty moves (?-NFA)

Definition

A string w is **accepted** by a ?-NFA M if and only if *there exists* a path starting at q_0 which is labeled by w and ends in a final state.

The **language accepted by** a ?-NFA M is the set of all strings which are accepted by M and is denoted by $L(M)$.

$L(M) = \{w : d(q_0, w) \cap F \neq \emptyset\}$

Nondeterministic Finite Automata with empty moves (?-NFA)

Notes

$d : Q \times (\Sigma \cup \{\epsilon\}) \rightarrow P(Q)$

☞ A ϵ -transition causes the machine to change its state non-deterministically, without consuming any input.

Nondeterministic Finite Automata with empty moves (?-NFA)

Notes

A ?-NFA has transition rules/possibilities like:

Empty string transition Nondeterministic transition

Nondeterministic Finite Automata with empty moves (?-NFA)

Nondeterminism - Parallelism

For any string w , the nondeterministic automaton can be in a subset $\subseteq Q$ of several possible states.

If the final set contains a final state, then the automaton accepts the string.

"The automaton processes the input in a parallel fashion; its computational path is no longer a line, but more like a tree".

Nondeterministic Finite Automata with empty moves (?-NFA)

We can write the NFA in two ways

1. State digraph

2. Table

δ	a	b	?
q0	{q0}	{q0, q1}	\emptyset
q1	\emptyset	\emptyset	{q2}
q2	\emptyset	\emptyset	\emptyset

$d : Q \times (\Sigma \cup \{I\}) \rightarrow P(Q)$

Nondeterministic Finite Automata with empty moves (?-NFA)

Example

This automaton accepts "0110", because there is a *possible* path that leads to a final state, namely: $q_1 \rightarrow q_1 \rightarrow q_2 \rightarrow q_3 \rightarrow q_4$
(note that $q_1 \rightarrow q_1 \rightarrow q_1 \rightarrow q_1 \rightarrow q_1$ is *not* accepting)

Nondeterministic Finite Automata with empty moves (?-NFA)

Example

The string 1 gets rejected: on "1" the automaton can only reach: $\{q_1, q_2, q_3\}$.

Nondeterministic Finite Automata with empty moves (?-NFA)

Example

A ?-transition is taken without consuming any character from the input.

What does the NFA above accepts?

aa^*+bb^*

Nondeterministic Finite Automata with empty moves (?-NFA)

Quiz

What are $d(q_0, 0)$, $d(q_0, 1)$, $d(q_0, ?)$ in each of M_1, M_2, M_3 and in M_4 ?

Nondeterministic Finite Automata with empty moves (?-NFA)

Answer

$\text{M1: } \begin{array}{c} \text{?} \\ \text{?} \\ \text{?} \end{array} \begin{array}{c} q_0 \\ \text{?} \\ \text{?} \end{array} \begin{array}{c} q_1 \\ \text{?} \\ \text{?} \end{array}$
 $\text{M2: } \begin{array}{c} \text{?} \\ \text{?} \\ \text{?} \end{array} \begin{array}{c} q_0 \\ \text{?} \\ \text{?} \end{array}$
 $\text{M3: } \begin{array}{c} \text{?} \\ \text{?} \\ \text{?} \end{array} \begin{array}{c} q_0 \\ \text{?} \\ \text{?} \end{array} \begin{array}{c} q_1 \\ \text{?} \\ \text{?} \end{array}$
 $\text{M4: } \begin{array}{c} \text{?} \\ \text{?} \\ \text{?} \end{array} \begin{array}{c} q_0 \\ \text{?} \\ \text{?} \end{array} \begin{array}{c} q_1 \\ \text{?} \\ \text{?} \end{array} \begin{array}{c} q_2 \\ \text{?} \\ \text{?} \end{array} \begin{array}{c} q_3 \\ \text{?} \\ \text{?} \end{array}$

$M1: d(q_0, 0) = d(q_0, 1) = d(q_0, ?) = \emptyset$
 $M2: \text{ Same}$
 $M3: d(q_0, 0) = d(q_0, 1) = \emptyset, d(q_0, ?) = \{q_1, q_2\}$
 $M4: d(q_0, 0) = \{q_1, q_3\}, d(q_0, 1) = \{q_2, q_3\}, d(q_0, ?) = \emptyset$

Nondeterministic Finite Automata with empty moves (?-NFA)

Quiz

Which of the following strings is accepted?

- ?
- 0
- 1
- 0111

Nondeterministic Finite Automata with empty moves (?-NFA)

Answer

- ? is rejected. No path labeled by empty string from start state to an accept state.
- 0 is accepted. EG the path $q_0 \xrightarrow{0} q_1$
- 1 is accepted. EG the path $q_0 \xrightarrow{1} q_2$
- 0111 is accepted. There is only one accepted path:
 $q_0 \xrightarrow{0} q_1 \xrightarrow{1} q_2 \xrightarrow{1} q_3 \xrightarrow{1} q_3 \xrightarrow{1} q_3$

Nondeterministic Finite Automata with empty moves (?-NFA)

Definition

Given a ?-NFA state s , the **?-closure(s)** is the set of states that are reachable through ?-transition from s .

?-closure(s) = {q: there is a path from s to q labeled ?}

Given a set of ?-NFA states T , the **?-closure(T)** is the set of states that are reachable through ?-transition from any state $s \in T$.

?-closure(T) = $\bigcup_{s \in T} \text{?-closure}(s)$

Nondeterministic Finite Automata with empty moves (?-NFA)

Example 1:

?-closure(q₀) = {q₀, q₁, q₂}
?-closure(q₁) = {q₁, q₂}
?-closure(q₂) = {q₂}

Nondeterministic Finite Automata with empty moves (?-NFA)

Example 2:

What states can be reached from state 1 without consuming a character?

Nondeterministic Finite Automata with empty moves (?-NFA)

Example

What states can be reached from state 1 without consuming a character?

{1,4,9,14} form the **?-closure** of state 1

Nondeterministic Finite Automata with empty moves (?-NFA)

Example

What are all the state closures in this NFA?

closure(1) = {1,4,9,14}	closure(10) = {10,11,13}
closure(5) = {5,6,8}	closure(13) = {11,13}
closure(8) = {6,8}	closure(12) = {12,13}
closure(7) = {7,8}	

Nondeterministic Finite Automata with empty moves (?-NFA)

Definition: Extension of d

$$d : Q \times (\sum U\{I\}) \rightarrow P(Q) \implies \hat{d} : Q \times \sum^* \rightarrow P(Q)$$

\hat{d} is defined as follows:

- $\hat{d}(q, ?) = \text{?-closure}(q)$
- $\hat{d}(q, wa) = \text{?-closure}(T)$ where $T = \{p : p \in d(r, a) \text{ and } r \in \hat{d}(q, w)\}, a \in S, w \in S^*$

Nondeterministic Finite Automata with empty moves (?-NFA)

Example: Extension of d

$\hat{d}(q_0, 01) = \{q_1, q_2\}$

?-NFA \rightarrow NFA

⌘ **Theorem:** For every language L that is accepted by a ?-NFA, there is an NFA that accepts L as well.

⌘ ?-NFA and NFA are equivalent computational models.

?-NFA \rightarrow NFA

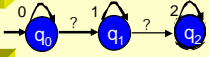
⌘ **Proof:**

Let $M = (Q, S, d, q_0, F)$ be a ?-NFA, an equivalent NFA, $M' = (Q, S, d', q_0, F')$ can be constructed as follows:

- $F' = \begin{cases} F \cup \{q_0\} & \text{If ?-closure}(q_0) \cap F \neq \emptyset \\ F & \text{Otherwise} \end{cases}$
- $d'(q, a) = \hat{d}(q, a)$

?-NFA → NFA

Example:

For the ?-NFA:  Construct the equivalent NFA?

Answer:

Given ?-NFA

$Q = \{q_0, q_1, q_2\}$ and $S = \{0, 1\}$

?-closure(q_0) = $\{q_0, q_1, q_2\} \cap F? F$

$d^*(q_0, 0) = \hat{d}(q_0, 0) = \{q_0, q_1, q_2\}$

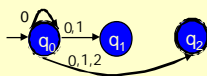
$d^*(q_0, 1) = \hat{d}(q_0, 1) = \{q_1, q_2\}$

$d^*(q_0, 2) = \hat{d}(q_0, 2) = \{q_2\}$

Constructed NFA

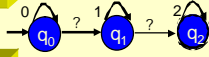
$Q = \{q_0, q_1, q_2\}$ and $S = \{0, 1\}$

$F' = \{q_0, q_1\}$



?-NFA → NFA

Example:

For the ?-NFA:  Construct the equivalent NFA?

Answer:

Given ?-NFA

$Q = \{q_0, q_1, q_2\}$ and $S = \{0, 1\}$

?-closure(q_0) = $\{q_0, q_1, q_2\} \cap F? F$

$d^*(q_1, 0) = \hat{d}(q_1, 0) = F$

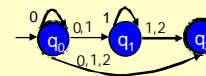
$d^*(q_1, 1) = \hat{d}(q_1, 1) = \{q_1, q_2\}$

$d^*(q_1, 2) = \hat{d}(q_1, 2) = \{q_2\}$

Constructed NFA

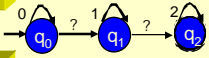
$Q = \{q_0, q_1, q_2\}$ and $S = \{0, 1\}$

$F' = \{q_0, q_1\}$



?-NFA → NFA

Example:

For the ?-NFA:  Construct the equivalent NFA?

Answer:

Given ?-NFA

$Q = \{q_0, q_1, q_2\}$ and $S = \{0, 1\}$

?-closure(q_0) = $\{q_0, q_1, q_2\} \cap F? F$

$d^*(q_2, 0) = \hat{d}(q_2, 0) = F$

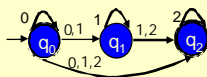
$d^*(q_2, 1) = \hat{d}(q_2, 1) = F$

$d^*(q_2, 2) = \hat{d}(q_2, 2) = \{q_2\}$

Constructed NFA

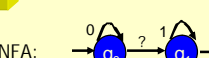
$Q = \{q_0, q_1, q_2\}$ and $S = \{0, 1\}$

$F' = \{q_0, q_1\}$



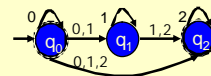
?-NFA → NFA

Example:

For the ?-NFA: 

Construct the equivalent NFA?

Answer:





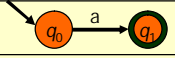
RE → ?-NFA

⚡ **Theorem:** Let r be RE, there exist a ?-NFA that accepts $L(r)$.

RE → ?-NFA

Proof:

The proof works by induction, using the recursive definition of regular expressions.

RE	?-NFA
F	
$?$	
a	

RE → ?-NFA

Proof:

$r1+r2 \rightarrow L(M1) \cup L(M2)$

$r^* \rightarrow L(M)^*$

$r1.r2 \rightarrow L(M1) L(M2)$

RE → ?-NFA

Example 1

For the regular expression $r=if$ we build the ?-NFA as follows:

The ?-NFA for a symbol i is:

The ?-NFA for a symbol f is:

The ?-NFA for the regular expression if is:

RE → ?-NFA

Example 2

For the regular expression $r=0+1^*$ build the equivalent ?-NFA ?

The ?-NFA for a symbol 0 is:

The ?-NFA for a symbol 1 is:

The ?-NFA for a symbol 1^* is:

The ?-NFA for the regular expression $0+1^*$ is:

RE → ?-NFA

Example 2

For the regular expression $r=0+1^*$ build the equivalent ?-NFA ?

The ?-NFA for the regular expression $0+1^*$ is:

RE → ?-NFA

Example 3

Q: Find an NFA for the regular expression $(0\cup 1)^*(0000000\cup 111(0\cup 1)^*111)(0\cup 1)^*$

RE → ?-NFA

Example 3

$(0\cup 1)^*(0000000\cup 111(0\cup 1)^*111)(0\cup 1)^*$

Note that: in this example $e = ?$

Exercise

RE → ?-NFA

Construct a ?-NFA for the regular expression:

$010^*1+(1+0)^*+101^*$