# Automata and Languages


**Prof. Mohamed Hamada**

Software Engineering Lab.
The University of Aizu
Japan


## Nondeterministic Finite Automata with empty moves (λ-NFA)

**Definition**

A nondeterministic finite automaton with empty moves (λ-NFA) M is defined by a 5-tuple $M=(Q,\Sigma,\delta,q_0,F)$, with

- Q: finite set of states
- Σ: finite input alphabet
- δ: transition function $\delta: Q\times(\Sigma \cup \{\lambda\}) \rightarrow P(Q)$
- $q_0 \in Q$: start state
- $F \subseteq Q$: set of final states

## Nondeterministic Finite Automata with empty moves (λ-NFA)

**Definition**

A string *w* is ***accepted*** by a λ-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 λ-NFA *M* is the set of all strings which are accepted by *M* and is denoted by *L* (*M* ).

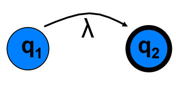
$L(M)=\{w: \delta(q_0,w) \cap F \neq \Phi\}$

## Nondeterministic Finite Automata with empty moves (λ-NFA)

**Notes**

$$\delta : Q \times (\Sigma \cup \{\lambda\}) \rightarrow P(Q)$$

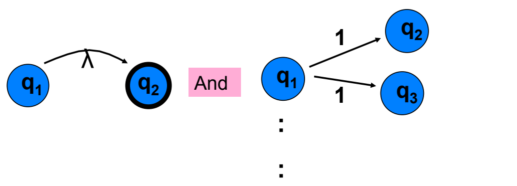
- A *λ-transition* causes the machine to change its state non-deterministically, without consuming any input.



## Nondeterministic Finite Automata with empty moves (λ-NFA)

**Notes**

A λ-NFA has transition rules/possibilities like:




Empty string transition

Nondeterministic transition

## Nondeterministic Finite Automata with empty moves (λ-NFA)

Nondeterminism ~ Parallelism

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

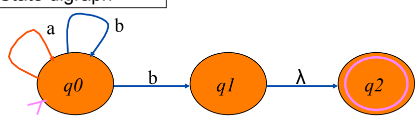
If the final set contains a final state, then the automaton accepts the string.

"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 (λ-NFA)

We can write the NFA in two ways

1. State digraph



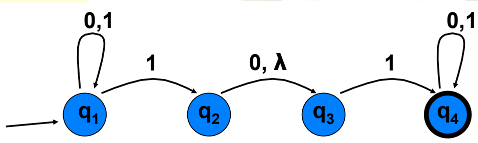

2. Table

$\delta : Q \times (\Sigma \cup \{\lambda\}) \rightarrow P(Q)$

| δ | a | b | λ |
|---|---|---|---|
| *q0* | {*q0*} | {*q0,q1*} | φ |
| *q1* | φ | φ | {*q2*} |
| *q2* | φ | φ | φ |

## Nondeterministic Finite Automata with empty moves (λ-NFA)

**Example**

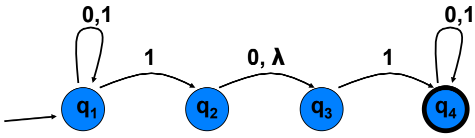



This automaton accepts "0110", because there is a *possible* path that leads to a final state, namely: $q_1 \rightarrow q_1 \rightarrow q_2 \rightarrow q_3 \rightarrow q_4 \rightarrow q_4$

(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 (λ-NFA)

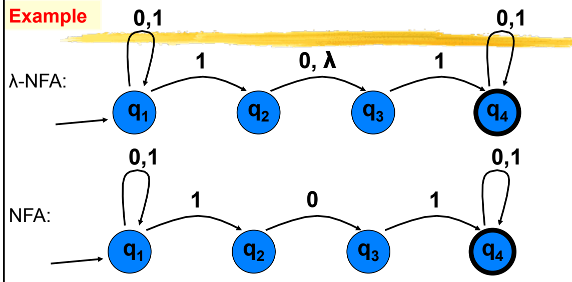
**Example**




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

## Difference between NFA and λ-NFA

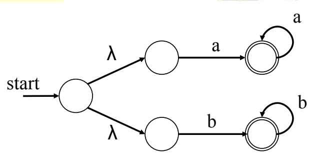
**Example**




The string 11 is accepted by the above λ-NFA
And rejected by the above NFA

## Nondeterministic Finite Automata with empty moves (λ-NFA)

**Example**



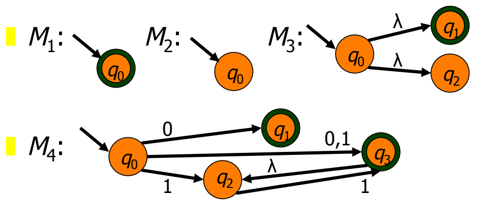

A λ-transition is taken without consuming any character from the input.

What does the NFA above accepts?

aa*+bb*

## Nondeterministic Finite Automata with empty moves (λ-NFA)

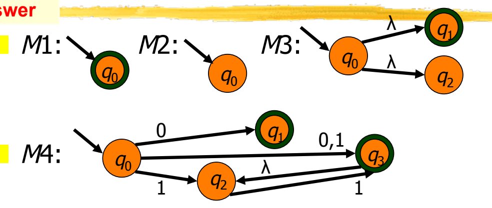
**Quiz**




What are $\delta(q_0,0)$, $\delta(q_0,1)$, $\delta(q_0, \lambda)$ in each of $M_1$, $M_2$, $M_3$ and in $M_4$.?

## Nondeterministic Finite Automata with empty moves (λ-NFA)

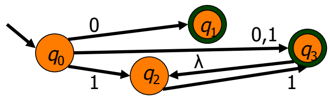
### Answer




$M1$: $\delta(q_0,0) = \delta(q_0,1) = \delta(q_0, \lambda) = \varnothing$
$M2$: Same
$M3$: $\delta(q_0,0) = \delta(q_0,1) = \varnothing$, $\delta(q_0, \lambda) = \{q_1,q_2\}$
$M4$: $\delta(q_0,0)=\{q_1,q_3\}$, $\delta(q_0,1)=\{q_2,q_3\}$, $\delta(q_0, \lambda)=\varnothing$

## Nondeterministic Finite Automata with empty moves (λ-NFA)

### Quiz

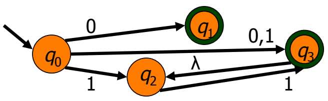



Which of the following strings is accepted?

1. λ
2. 0
3. 1
4. 0111

## Nondeterministic Finite Automata with empty moves (λ-NFA)

### Answer




1. λ is rejected. No path labeled by empty string from start state to an accept state.
2. 0 is accepted. EG the path $q_0 \underset{0}{\rightarrow} q_1$
3. 1 is accepted. EG the path $q_0 \underset{1}{\rightarrow} q_3$
4. 0111 is accepted. There is only one accepted path:

$$q_0 \underset{0}{\rightarrow} q_3 \underset{\lambda}{\rightarrow} q_2 \underset{1}{\rightarrow} q_3 \underset{\lambda}{\rightarrow} q_2 \underset{1}{\rightarrow} q_3 \underset{\lambda}{\rightarrow} q_2 \underset{1}{\rightarrow} q_3$$

## Nondeterministic Finite Automata with empty moves (λ-NFA)

### Definition

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

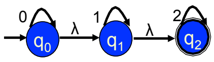
**λ-closure(s)={q: there is a path from s to q labeled λ}**

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

**λ-closure(T)=** $\bigcup_{s\in T}$ **λ-closure(s)**

## Nondeterministic Finite Automata with empty moves (λ-NFA)

**Example 1:**



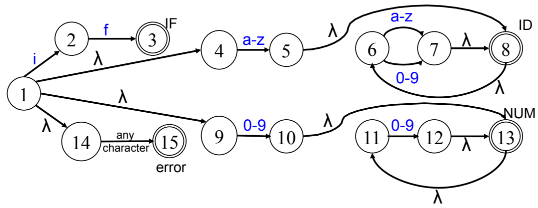

**λ-closure($q_0$) = {$q_0$,$q_1$, $q_2$}**

**λ-closure($q_1$) = {$q_1$, $q_2$}**

**λ-closure($q_2$) = {$q_2$}**

## Nondeterministic Finite Automata with empty moves (λ-NFA)

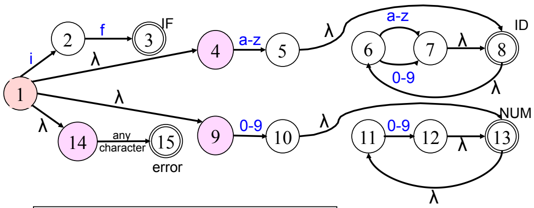
**Example 2:**




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

## Nondeterministic Finite Automata with empty moves (λ-NFA)

**Example**

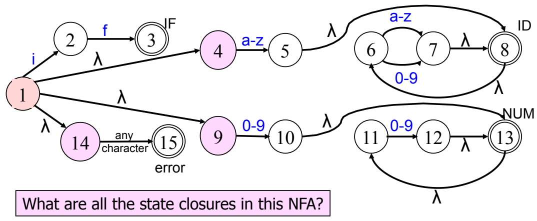



What states can be reached from state 1 without consuming a character?

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

## Nondeterministic Finite Automata with empty moves (λ-NFA)

**Example**




What are all the state closures in this NFA?

closure(1) = {1,4,9,14}
closure(5) = {5,6,8}
closure(8) = {6,8}
closure(7) = {6,7,8}

closure(10) = {10,11,13}
closure(13) = {11,13}
closure(12) = {11,12,13}

## Nondeterministic Finite Automata with empty moves (λ-NFA)

**Definition: Extension of δ**

$$\delta : Q \times (\sum \cup \{\lambda\}) \rightarrow P(Q) \implies \hat{\delta} : Q \times \sum^{*} \rightarrow P(Q)$$

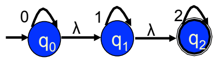
$\hat{\delta}$ **is defined as follows:**

1. $\hat{\delta}$(q,λ)=λ-closure(q)
2. $\hat{\delta}$ (q,wa)=λ-closure(T) where

T={p: p∈δ(r,a) and r∈ $\hat{\delta}$(q,w)}, a∈**Σ**, w∈Σ*

## Nondeterministic Finite Automata with empty moves (λ-NFA)

**Example: Extension of δ**




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

## λ-NFA → NFA

- Theorem: For every language L that is accepted by a λ-NFA, there is an NFA that accepts L as well.
- λ-NFA and NFA are equivalent computational models.

## λ-NFA → NFA

- Proof:

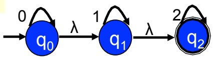
Let M=(Q,**Σ**,δ,$q_0$,F) be a λ-NFA, an equivalent NFA, M' =(Q,**Σ**,δ' ,$q_0$,F' ) can be constructed as follows:

1. $F' = \begin{cases} F \cup \{q_0\} & \text{If } \lambda\text{-closure}(q_0) \cap F \neq \Phi \\ F & \text{Otherwise} \end{cases}$

2. $\delta'(q,a) = \hat{\delta}(q,a)$

## λ-NFA → NFA

**Example:**

For the λ-NFA: 

Construct the equivalent NFA?

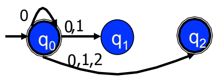
**Answer:**

| Given λ-NFA | | Constructed NFA |
|---|---|---|
| $Q=\{q_0,q_1,q_2\}$ and $\Sigma=\{0,1,2\}$ | ⇒ | $Q=\{q_0,q_1,q_2\}$ and $\Sigma=\{0,1,2\}$ |
| λ-closure($q_0$)=$\{q_0,q_1,q_2\}$ ∩ F≠Φ | ⇒ | F' = $\{q_0,q_2\}$ |

$\delta'(q_0,0) = \hat{\delta}(q_0,0) = \{q_0,q_1,q_2\}$

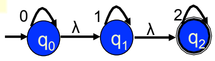
$\delta'(q_0,1) = \hat{\delta}(q_0,1) = \{q_1,q_2\}$

$\delta'(q_0,2) = \hat{\delta}(q_0,2) = \{q_2\}$




## λ-NFA → NFA

**Example:**

For the λ-NFA: 

Construct the equivalent NFA?

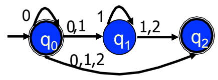
**Answer:**

| Given λ-NFA | | Constructed NFA |
|---|---|---|
| $Q=\{q_0,q_1,q_2\}$ and $\Sigma=\{0,1,2\}$ | ⇒ | $Q=\{q_0,q_1,q_2\}$ and $\Sigma=\{0,1,2\}$ |
| λ-closure($q_0$)=$\{q_0,q_1,q_2\}$ ∩ F≠Φ | ⇒ | F' = $\{q_0,q_1\}$ |

$\delta'(q_1,0) = \hat{\delta}(q_1,0) = \Phi$

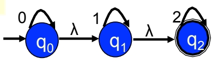
$\delta'(q_1,1) = \hat{\delta}(q_1,1) = \{q_1,q_2\}$

$\delta'(q_1,2) = \hat{\delta}(q_1,2) = \{q_2\}$




## λ-NFA → NFA

**Example:**

For the λ-NFA: 

Construct the equivalent NFA?

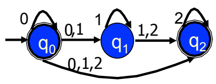
**Answer:**

| Given λ-NFA | | Constructed NFA |
|---|---|---|
| $Q=\{q_0,q_1,q_2\}$ and $\Sigma=\{0,1,2\}$ | ⇒ | $Q=\{q_0,q_1,q_2\}$ and $\Sigma=\{0,1,2\}$ |
| λ-closure($q_0$)=$\{q_0,q_1,q_2\}$ ∩ F≠Φ | ⇒ | F' = $\{q_0,q_1\}$ |

$\delta'(q_2,0) = \hat{\delta}(q_2,0) = \Phi$

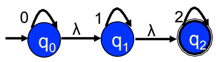
$\delta'(q_2,1) = \hat{\delta}(q_2,1) = \Phi$

$\delta'(q_2,2) = \hat{\delta}(q_2,2) = \{q_2\}$



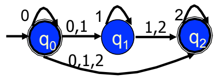

## λ-NFA → NFA

**Example:**

For the λ-NFA: 


Construct the equivalent NFA?

**Answer:**



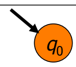
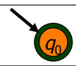
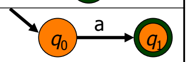
## RE → λ-NFA

Theorem: Let r be RE, there exist a λ-NFA that accepts L(r).

## RE → λ-NFA

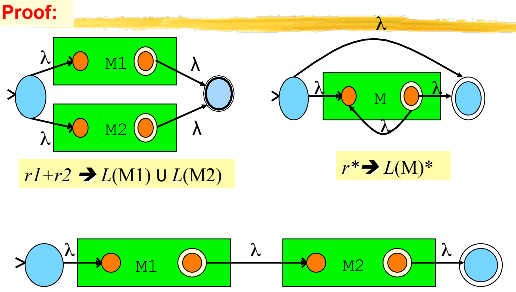
**Proof:**

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

| RE | λ-NFA |
|---|---|
| Φ |  |
| λ |  |
| a |  |

## RE → λ-NFA

**Proof:**




*r1+r2* → *L*(M1) ∪ *L*(M2)

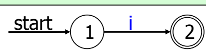
*r** → *L*(M)*

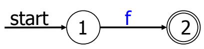
*r1.r2* → *L*(M1) *L*(M2)

## RE → λ-NFA

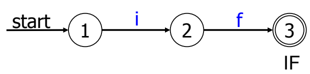
**Example 1**

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

The λ-NFA for a symbol i is: 

The λ-NFA for a symbol f is: 

The λ-NFA for the regular expression if is:



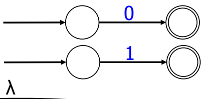
## RE → λ-NFA

**Example 2**

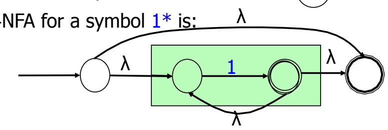
For the regular expression r=0+1* build the equivalent λ-NFA ?

The λ-NFA for a symbol 0 is:

The λ-NFA for a symbol 1 is:




The λ-NFA for a symbol 1* is:



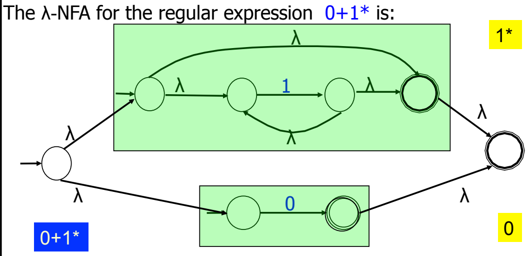

The λ-NFA for the regular expression 0+1* is:

## RE → λ-NFA

**Example 2**

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

The λ-NFA for the regular expression 0+1* is:




## RE → λ-NFA

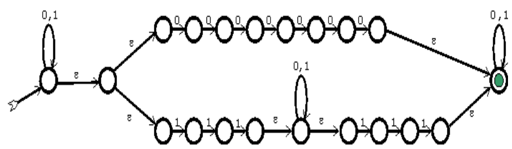
**Example 3**

Q: Find an NFA for the regular expression (0∪1)*(0000000∪111(0∪1)*111)(0∪1)*

## RE → λ-NFA

**Example 3**

(0∪1)*(0000000∪111(0∪1)*111)(0∪1)*



Note that: in this example ε = λ

# RE → λ-NFA

**Exercise**

Construct a λ-NFA for the regular expression:

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