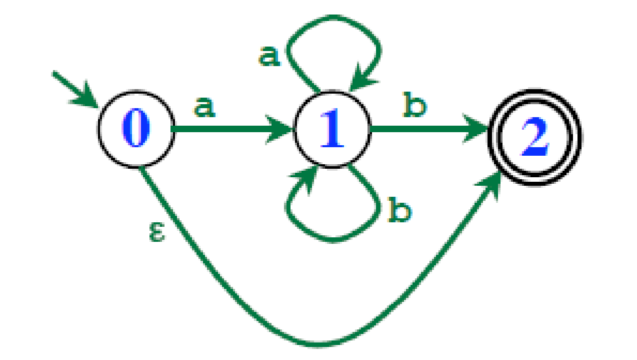
Lexical Analysis: Part 2

Lecture 3

Simulating DFA

```
function Match () returns boolean
   var s: State
      ch: char
s = s₀
ch = nextChar()
while ch ≠ EOF do
   s = Move(s,ch)
   ch = NextChar()
endWhile
if s ∈ FinalStates then
   return true
else
   return false
endIf
endFunction
```



The “Move” function

Perhaps an array **s = Move[s,ch]**

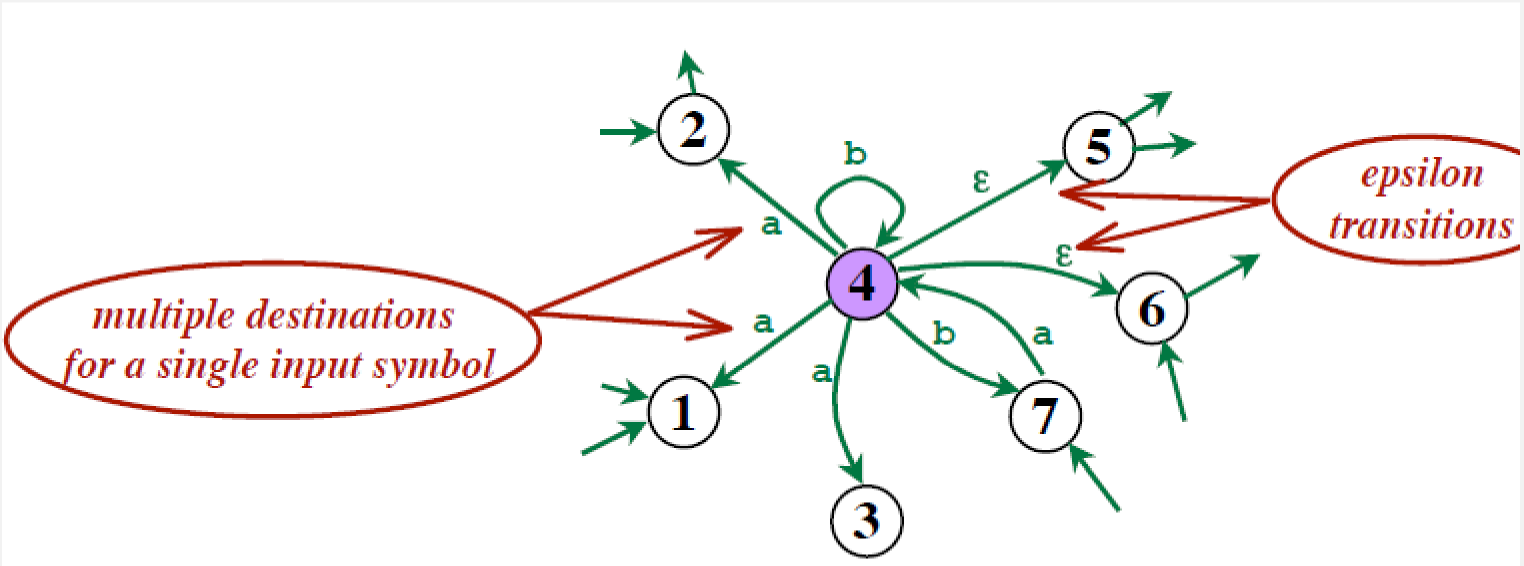
Perhaps a linked list representation, to save space

Is Move always defined?

Use “dead” state to deal with undefined edges.

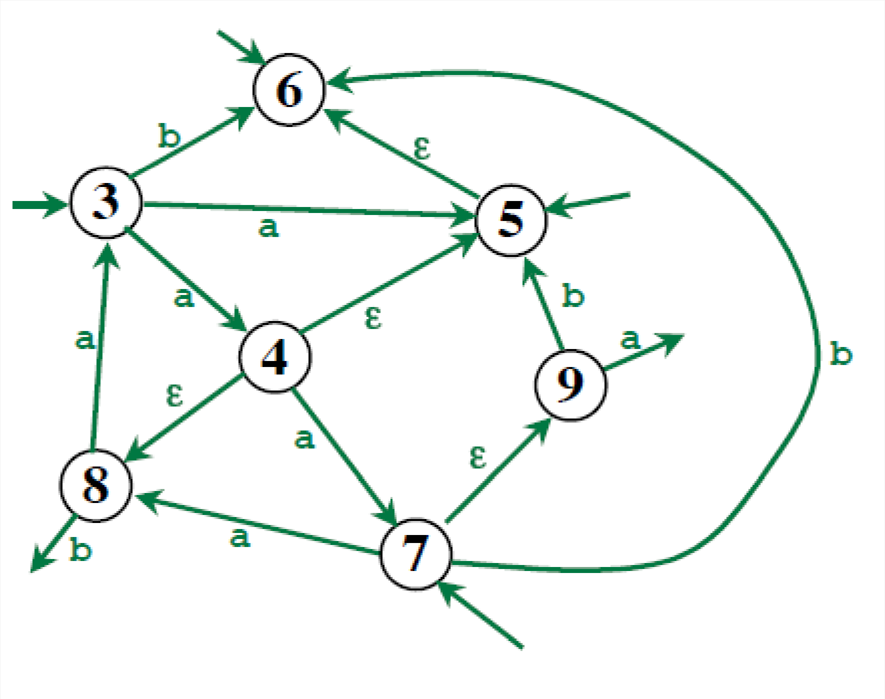
Simulating NFA

```
States: s, t
Sets of states: S, T
Deterministic Machine:
   Move(s,ch) → t
Non-deterministic Machine:
MoveNFA(S,ch) → T
      If s ∊ S and there is an edge...
            then t ∊ T
```



Example

$\text{Move}_{\textbf{NFA}}$ **({3,7}, a) = { 4, 5, 8 }**

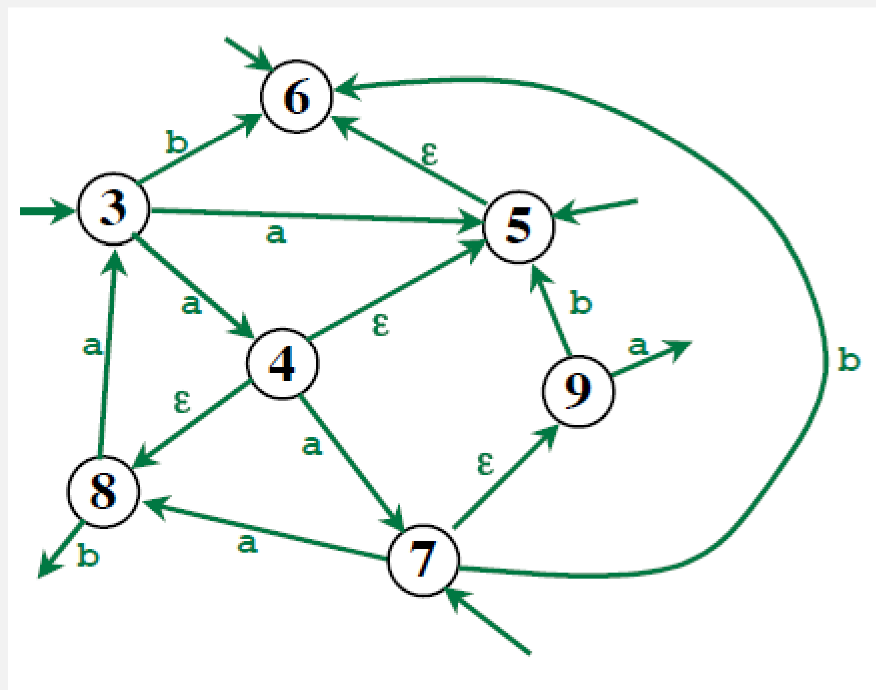


ε-closure

Define **ε**-Closure (s):

The set of states reachable from s on **ε** -transitions.

ε-closure (4) = { 4, 5, 6, 8 }

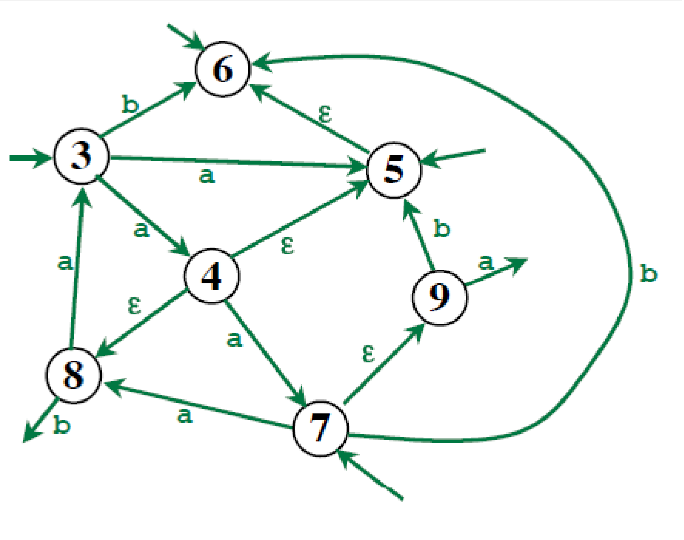


ε-closure

Define **ε**-Closure (s):

The set of states reachable from s on **ε** -transitions.

ε-closure (4) = { 4, 5, 6, 8 }



Define **ε** *-Closure(S):*

{ t | t ϵ **ε**-closure (s) for all s ϵ S }

ε -closure ({4, 7}) = { 4, 5, 6, 7, 8, 9 }

Computation of ε-closure

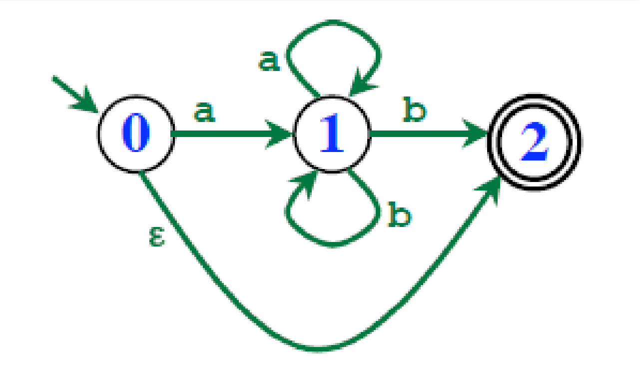
Given: T (= a set of states)
Goal: Compute **ε**-Closure(T)
Approach: Use a stack of states (= the states that we still need to look at)

```
Algorithm:
var
    stack: stack of states
    result: set of states
push all states in T onto stack
result = T
while stack not empty do
    s = pop(stack)
    for each state u
        such that an edge exists do
        if u is not in result then
            add u to result
            push u onto stack
        endIf
     endFor
endWhile
```

Example

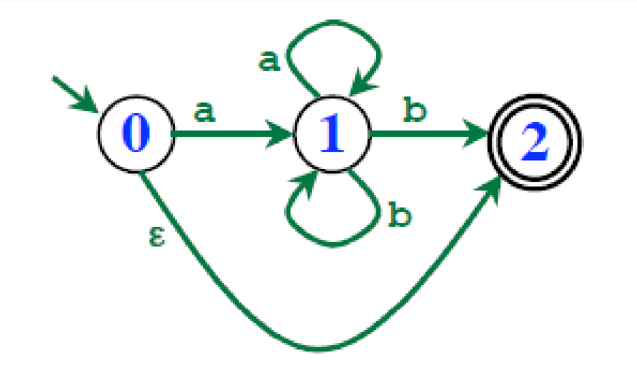
Input String: abab

Let S be the state(s) we are in…



Example

Input String: abab

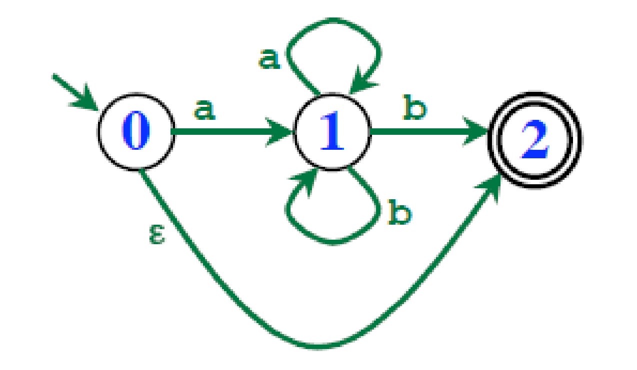


Let S be the state(s) we are in...

```
    S   = ε-Closure ({0})
```

Example

Input String: abab



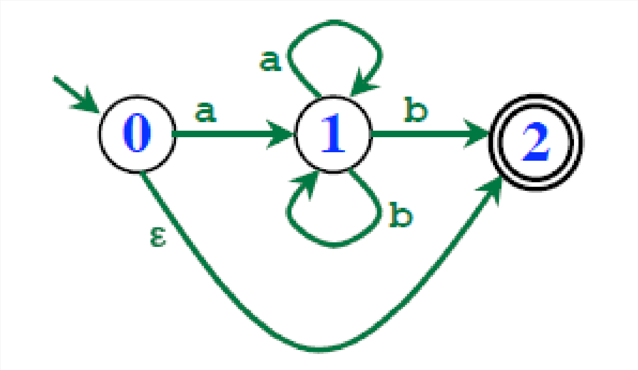
Let S be the state(s) we are in...

```
    S   = ε-Closure ({0})
        = {0,2}
```

Look at next character...   ch = a

Example

Input String: abab



Let S be the state(s) we are in...

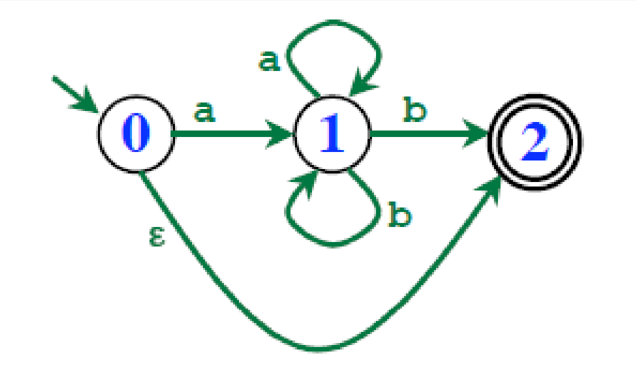
```
    S   = ε -Closure ({0})
        = {0,2}
```

Look at next character...   ch = a

Move to next state(s)…

Example

Input String: abab



Let S be the state(s) we are in...

$S = \varepsilon\text{-Closure}(\{0\})$

$= \{0,2\}$

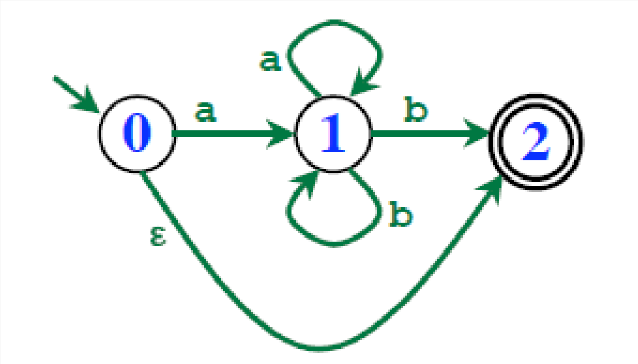
Look at next character... ch = a

Move to next state(s)...

$S = \varepsilon\text{-Closure}(\text{Move}_{NFA}(\{0,2\}, a))$

Example

Input String: abab



Let S be the state(s) we are in...

```
    S   = ε-Closure ({0})
        = {0,2}
```

Look at next character... ch = a

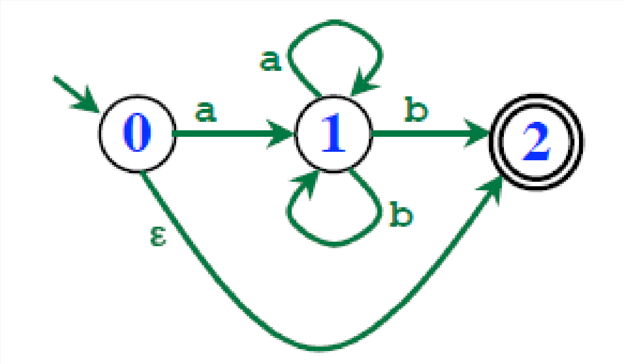
Move to next state(s)...

$S = \boldsymbol{\varepsilon}\text{-Closure}\ (\text{Move}_{NFA}(\{0,2\},\ a))$

$\quad = \boldsymbol{\varepsilon}\text{-Closure}\ (\{1\})$

Example

Input String: abab



Let S be the state(s) we are in...

```
    S   = ε -Closure ({0})
        = {0,2}
```

Look at next character... ch = a

Move to next state(s)...

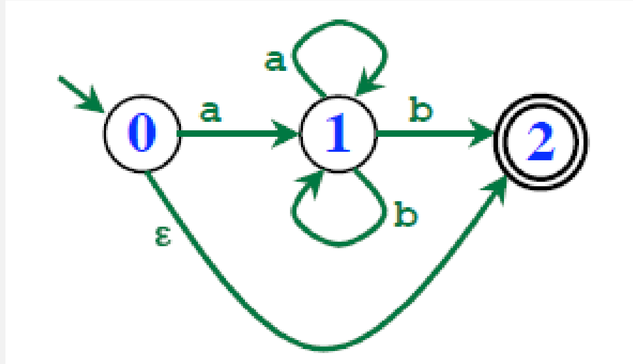
$S = \varepsilon\text{-Closure }(\text{Move}_{NFA}(\{0,2\}, a))$

$\quad = \varepsilon\text{ -Closure }(\{1\})$

$\quad = \{1\}$

Example

Input String: abab



Let S be the state(s) we are in...

$S = \boldsymbol{\varepsilon}\text{-Closure}(\{0\})$

$= \{0,2\}$

Look at next character... ch = a

Move to next state(s)...

$S = \boldsymbol{\varepsilon}\text{-Closure}(\text{Move}_{NFA}(\{0,2\}, a))$

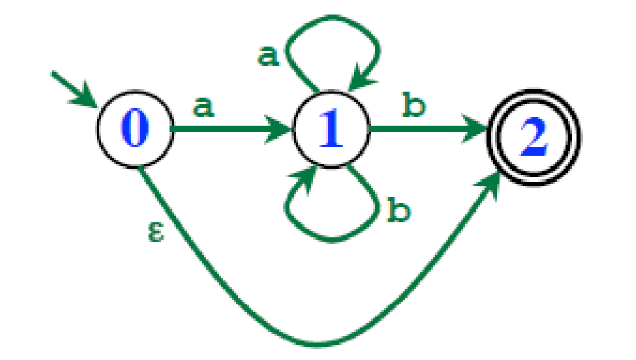
$= \boldsymbol{\varepsilon}\text{-Closure}(\{1\})$

$= \{1\}$

Look at next character... ch = b

Example

Input String: abab



Let S be the state(s) we are in...

$S = \varepsilon\text{-Closure}(\{0\})$

$= \{0,2\}$

Look at next character... ch = a

Move to next state(s)...

$S = \varepsilon\text{-Closure}(\text{Move}_{NFA}(\{0,2\}, a))$

$= \varepsilon\text{-Closure}(\{1\})$

$= \{1\}$

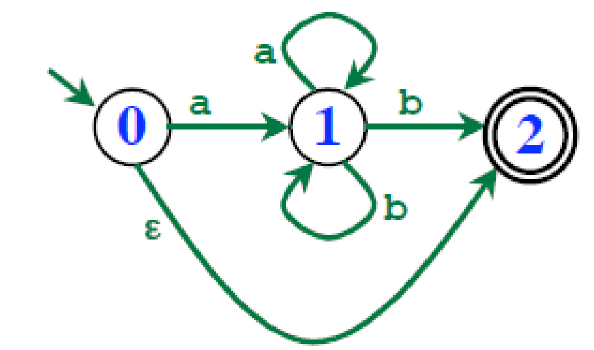
Look at next character... ch = b

Move to next state(s)...

$S = \varepsilon\text{-Closure}(\text{Move}_{NFA}(\{1\}, b))$

Example

Input String: abab



Let S be the state(s) we are in...

```
    S   = ε-Closure ({0})
        = {0,2}
```

Look at next character... ch = a

Move to next state(s)...

$S = \varepsilon\text{-Closure}(\text{Move}_{NFA}(\{0,2\}, a))$

$\quad = \varepsilon\text{-Closure}(\{1\})$

$\quad = \{1\}$

Look at next character... ch = b

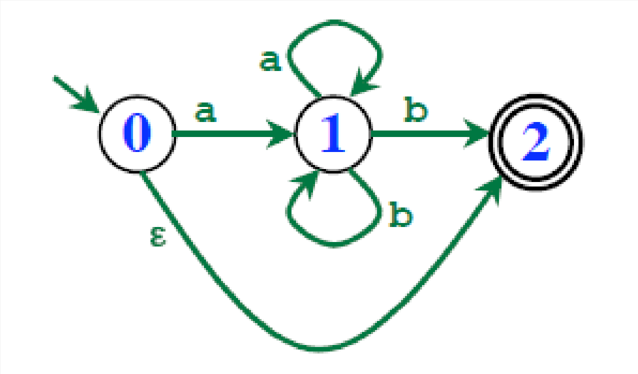
Move to next state(s)...

$S = \varepsilon\text{-Closure}(\text{Move}_{NFA}(\{1\}, b))$

$\quad = \varepsilon\text{-Closure}(\{1,2\})$

Example

Input String: abab



Let S be the state(s) we are in...

$S = \boldsymbol{\varepsilon}\text{-Closure}(\{0\})$

$= \{0,2\}$

Look at next c

Look at next character... ch = a

Move to next state(s)...

$S = \boldsymbol{\varepsilon}\text{-Closure}(\text{Move}_{\text{NFA}}(\{0,2\}, a))$

$= \boldsymbol{\varepsilon}\text{-Closure}(\{1\})$

$= \{1\}$

Look at next character... ch = b

Move to next state(s)...

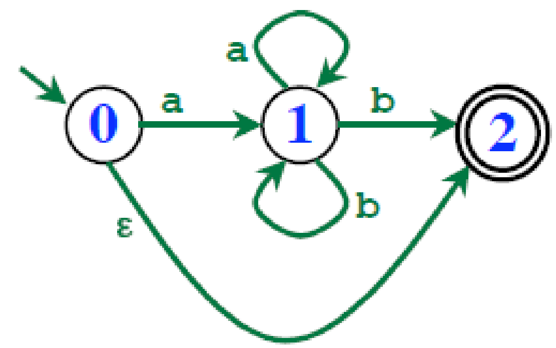
$S = \boldsymbol{\varepsilon}\text{-Closure}(\text{Move}_{\text{NFA}}(\{1\}, b))$

$= \boldsymbol{\varepsilon}\text{-Closure}(\{1,2\})$

$= \{1,2\}$

Example

Input String: abab



Let S be the state(s) we are in...

S = ε-Closure ({0})

= {0,2}

Look at next character... ch = a

Move to next state(s)...

S = ε-Closure ($Move_{NFA}$({0,2}, a))

= ε-Closure ({1})

= {1}

Look at next character... ch = b

Move to next state(s)...

S = ε-Closure ($Move_{NFA}$({1}, b))

= ε-Closure ({1,2})

= {1,2}

Look at next character... ch = a

Move to next state(s)...

S = ε-Closure ($Move_{NFA}$({1,2},a)

Example

Input String: abab

Let S be the state(s) we are in...

$S = \boldsymbol{\varepsilon}\text{-Closure}(\{0\})$
$\quad = \{0,2\}$

Look at next character... ch = a

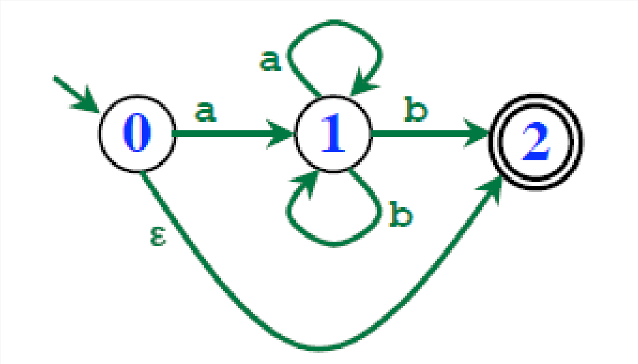
Move to next state(s)...

$S = \boldsymbol{\varepsilon}\text{-Closure}(\text{Move}_{NFA}(\{0,2\}, a))$
$\quad = \boldsymbol{\varepsilon}\text{-Closure}(\{1\})$
$\quad = \{1\}$

Look at next character... ch = b

Move to next state(s)...

$S = \boldsymbol{\varepsilon}\text{-Closure}(\text{Move}_{NFA}(\{1\}, b))$
$\quad = \boldsymbol{\varepsilon}\text{-Closure}(\{1,2\})$
$\quad = \{1,2\}$



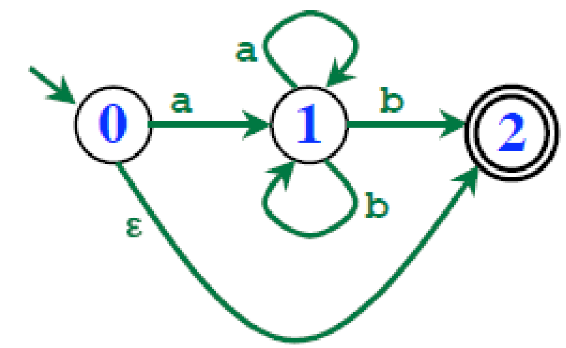
Look at next character... ch = a

Move to next state(s)...

$S = \boldsymbol{\varepsilon}\text{-Closure}(\text{Move}_{NFA}(\{1,2\},a))$
$\quad = \boldsymbol{\varepsilon}\text{-Closure}(\{1\})$
$\quad = \{1\}$

Look at next character... ch = b

Example



Input String: **abab**

Let S be the state(s) we are in...

S = **ε**-Closure ({0})
  = {0,2}

Look at next character... ch = a

Move to next state(s)...

S = **ε**-Closure ($Move_{NFA}$({0,2}, a))
  = **ε**-Closure ({1})
  = {1}

Look at next character... ch = b

Move to next state(s)...

S = **ε**-Closure ($Move_{NFA}$({1}, b))
  = **ε**-Closure ({1,2})
  = {1,2}

Look at next character... ch = a

Move to next state(s)...

S = **ε**-Closure ($Move_{NFA}$({1,2},a))
  = **ε**-Closure ({1})
  = {1}

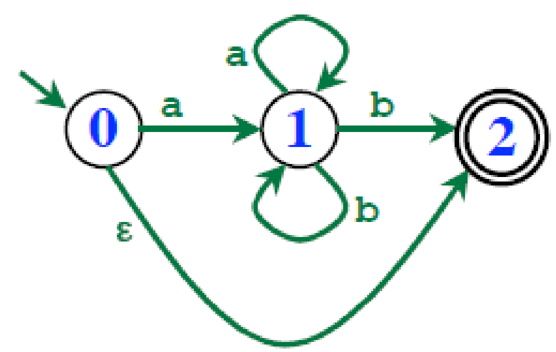
Look at next character... ch = b

Move to next state(s)...

S = **ε**-Closure ($Move_{NFA}$({1}, b))

Example

Input String: abab



Let S be the state(s) we are in...

S = **ε**-Closure ({0})
= {0,2}

Look at next character... ch = a

Move to next state(s)...

S = **ε**-Closure (Move$_{NFA}$({0,2}, a)
= **ε**-Closure ({1})
= {1}

Look at next character... ch = b

Move to next state(s)...

S = **ε**-Closure (Move$_{NFA}$({1}, b)
= **ε**-Closure ({1,2})
= {1,2}

Look at next character... ch = a

Move to next state(s)...

S = **ε**-Closure (Move$_{NFA}$({1,2},a)
= **ε**-Closure ({1})
= {1}

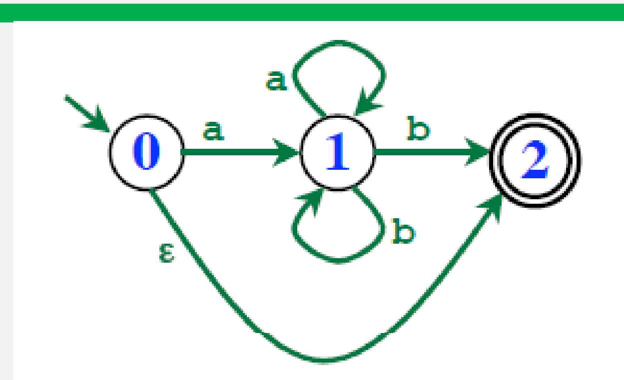
Look at next character... ch = b

Move to next state(s)...

S = **ε**-Closure (Move$_{NFA}$({1}, b)
= {1,2}

Example

Input String: abab



Let S be the state(s) we are in...

S = **ε**-Closure ({0})

= {0,2}

Look at next character... ch = a

Move to next state(s)...

S = **ε**-Closure (Move_{NFA}({0,2}, a))

= **ε**-Closure ({1})

= {1}

Look at next character... ch = b

Move to next state(s)...

S = **ε**-Closure (Move_{NFA}({1}, b))

= **ε**-Closure ({1,2})

= {1,2}

Look at next character... ch = a

Move to next state(s)...

S = **ε**-Closure (Move_{NFA}({1,2},a))

= **ε**-Closure ({1})

= {1}

Look at next character... ch = b

Move to next state(s)...

S = **ε**-Closure (Move_{NFA}({1}, b))

= {1,2}

Look at next character... ch = EOF

Does S contain a Final State?

This string is accepted!!!

Simulating a NFA

```
function Match () returns boolean
var S: set of states
   ch: char
S = ε-Closure({s0})
ch = nextChar()
while ch ! EOF do
   S = ε-Closure(MoveNFA(S,ch))
   ch = NextChar()
endWhile
if S∩FinalStates ≠ {} then
      return true
else
      return false
endIf
endFunction
```

Thompson’s construction

Build an NFA for: **ab*c|d*e***

Thompson's construction

Build an NFA for: **ab*c|d*e***

Break the expression into sub-expressions: **(ab*c)** | **(d*e*)**

Thompson’s construction

Build an NFA for: **ab*c|d*e***

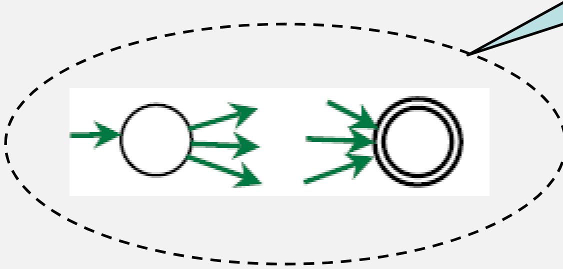
Break the expression into sub-expressions: **(ab*c)** | **(d*e*)**
Glue the two NFA together

Thompson's construction

Build an NFA for: **ab*c|d*e***

Break the expression into sub-expressions: **(ab*c) | (d*e*)**
Glue the two NFA together

NFA for **ab*c**

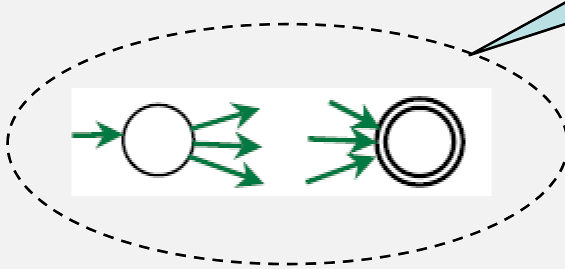


Thompson's construction

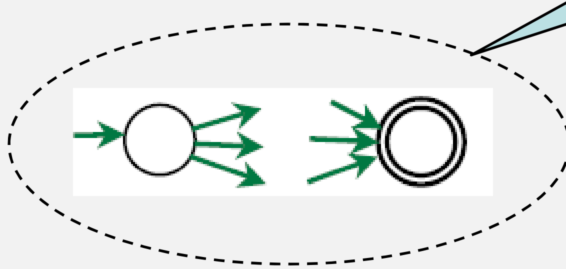
Build an NFA for: **ab*c|d*e***

Break the expression into sub-expressions: **(ab*c)** | **(d*e*)**
Glue the two NFA together

NFA for **ab*c**



NFA for **d*e***

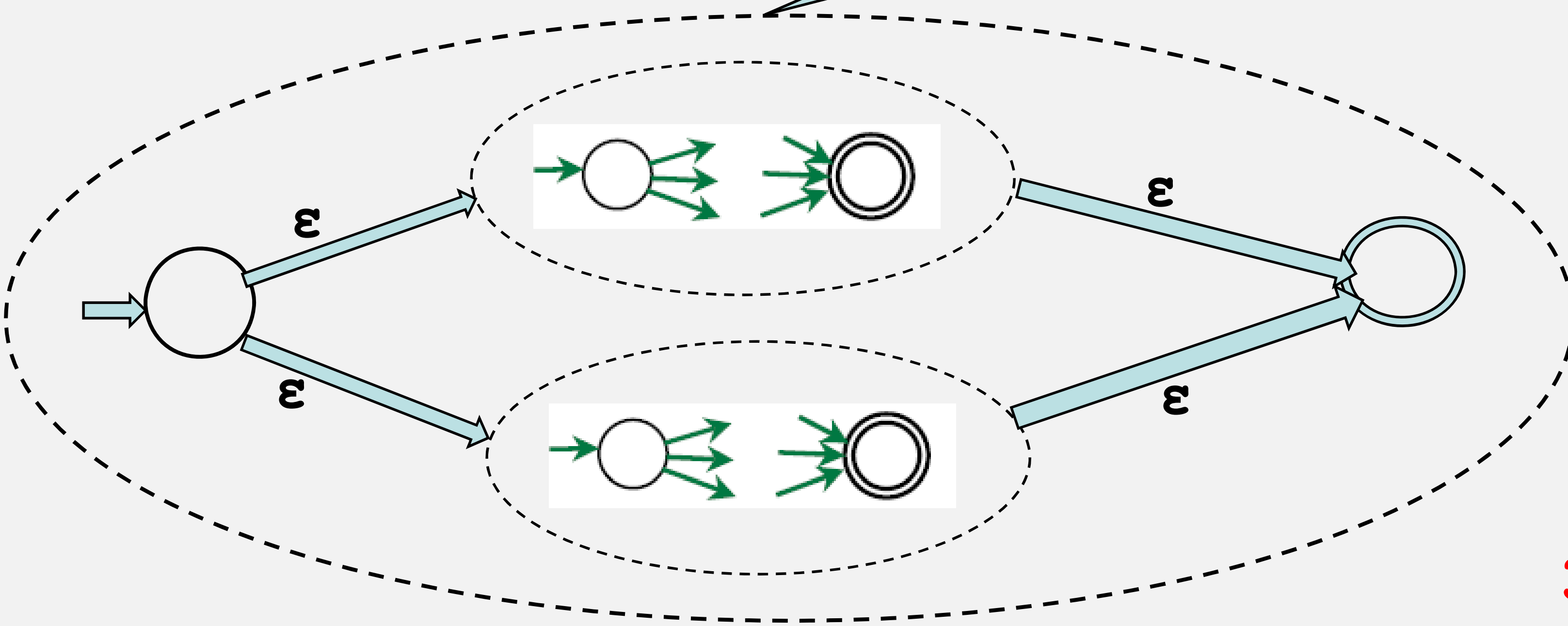


Thompson’s construction

Build an NFA for: **ab*c|d*e***

Break the expression into sub-expressions: **(ab*c) | (d*e*)**
Glue the two NFA together

NFA for **ab*c | (d*e*)**



Types of Regular Expression

case 1: **a** **where a$\epsilon\Sigma$**
case 2: **r1|r2**
case 3: **r1r2**
case 4: **r1***
case 5: **ε**
case 6: **(r1)**

For every NFA we construct...

- 1 start state
- 1 accepting state
- No edge enters the start state
- No edge leaves the accepting state

Types of Regular Expression

case 1: **a where aϵΣ**

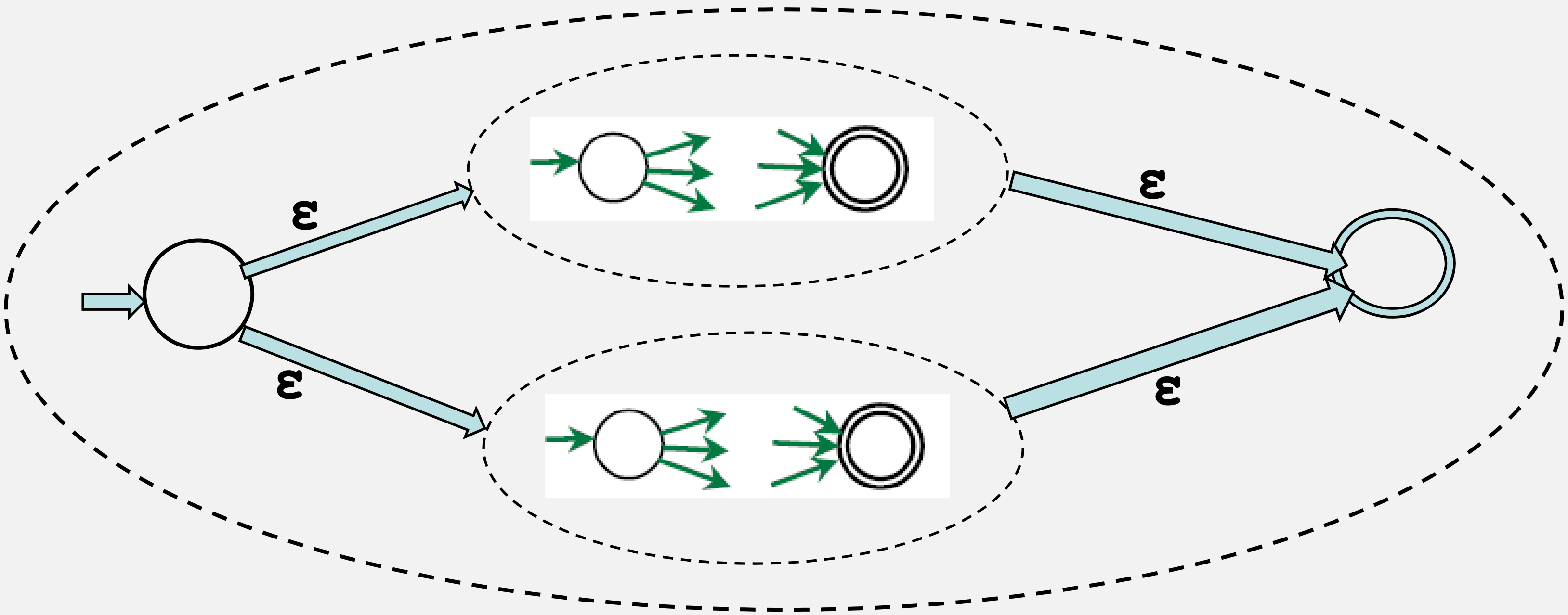
For a regular expression consisting of only **a (for any aϵΣ)**
Construct

...and call it N(**a)**

Types of Regular Expression

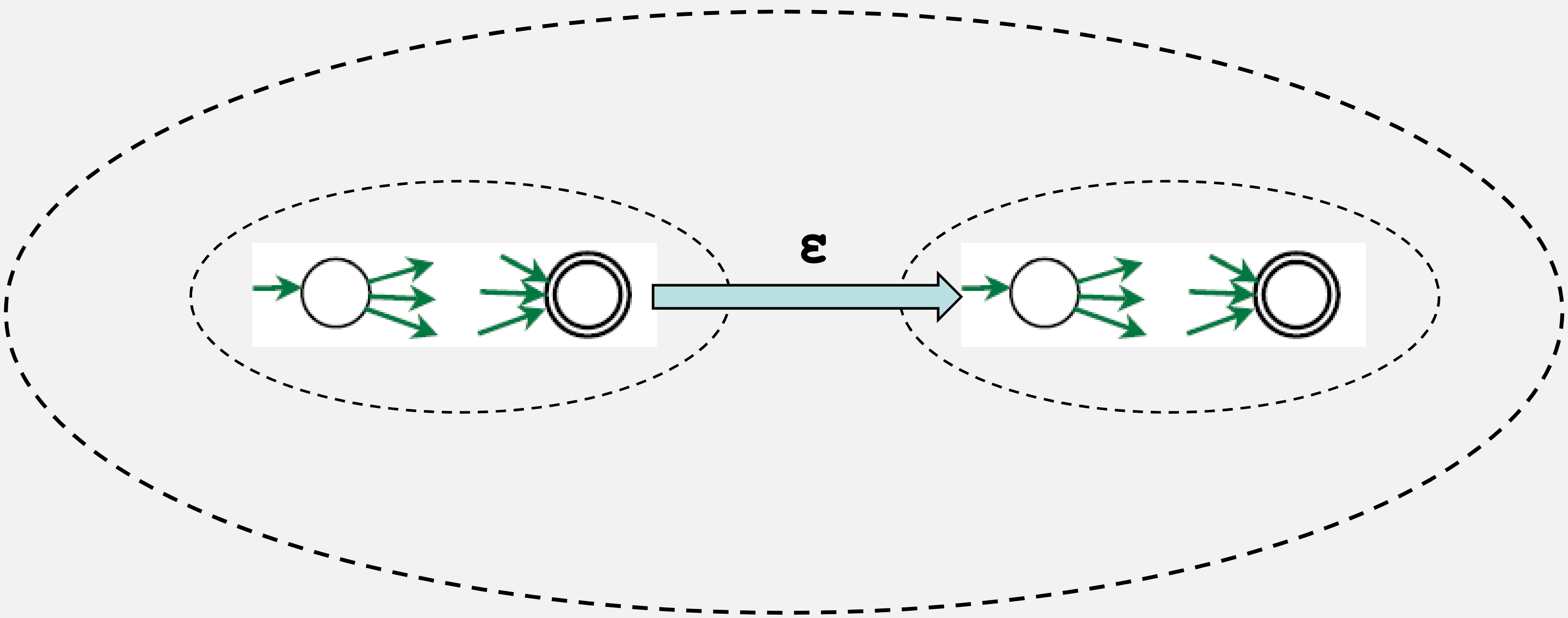
case 2: **r1|r2**

For **r1|r2, construct N(r1|r2)**



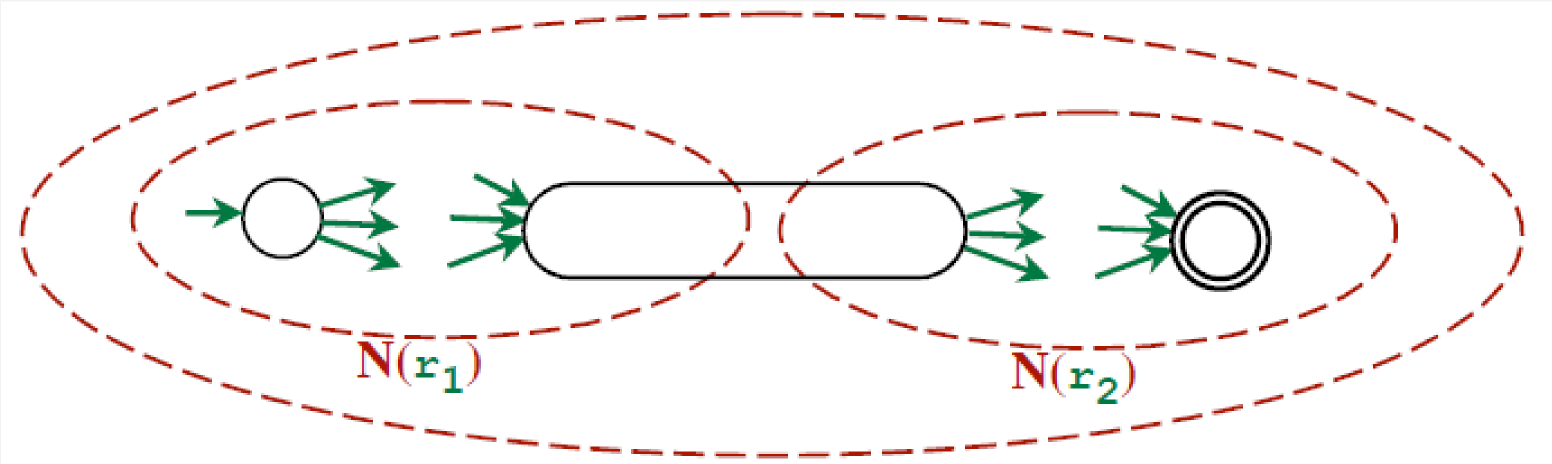
Types of Regular Expression

case 3: **r1r2**



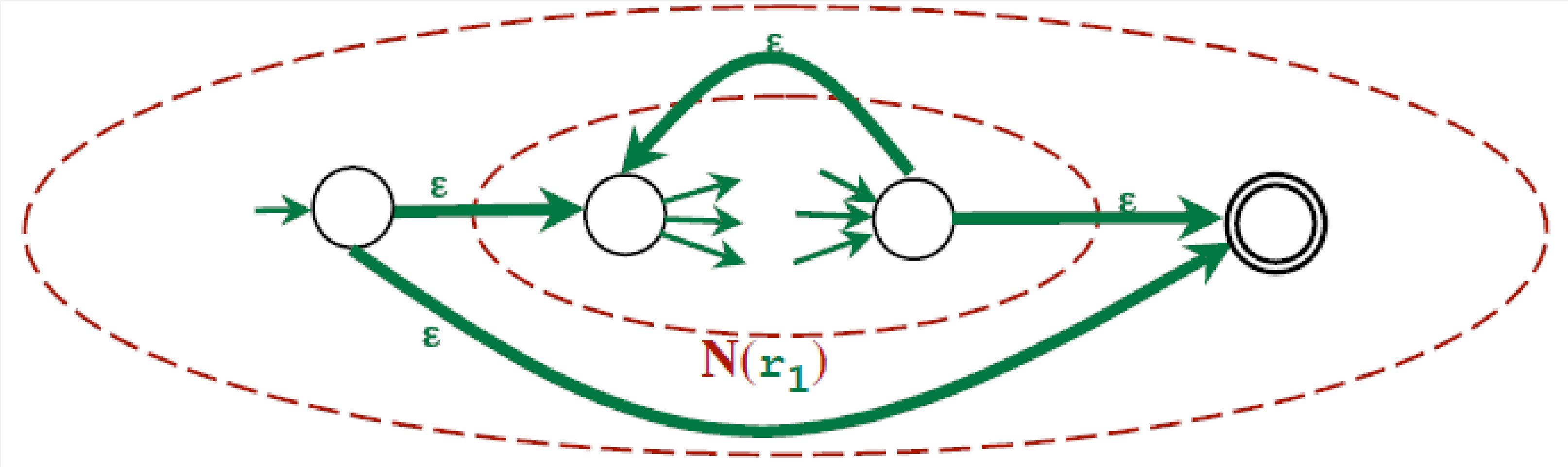
Types of Regular Expression

case 3: **r1r2 (alternative: combine states)**



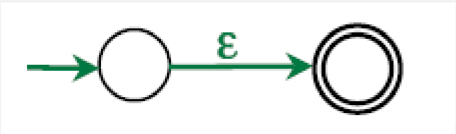
Types of Regular Expression

case 4: **r1***



Types of Regular Expression

case 5: **ε**

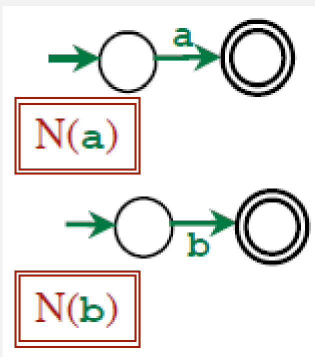


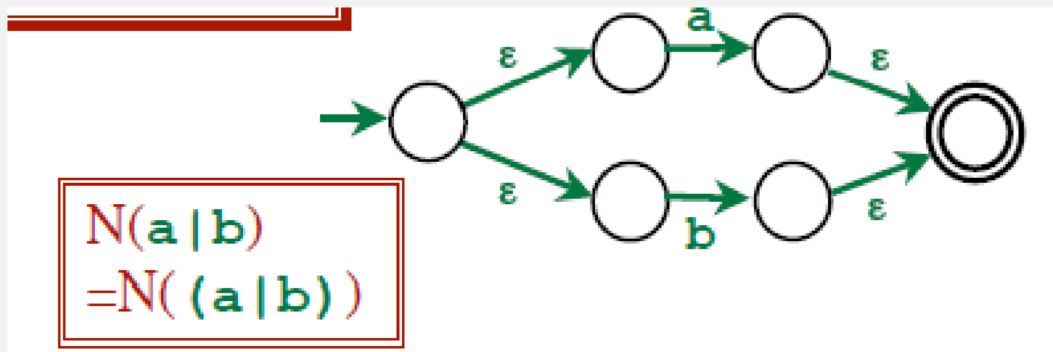
case 6: **(r1)**

Let N(**(r1)) be N(r1) itself.**

Example

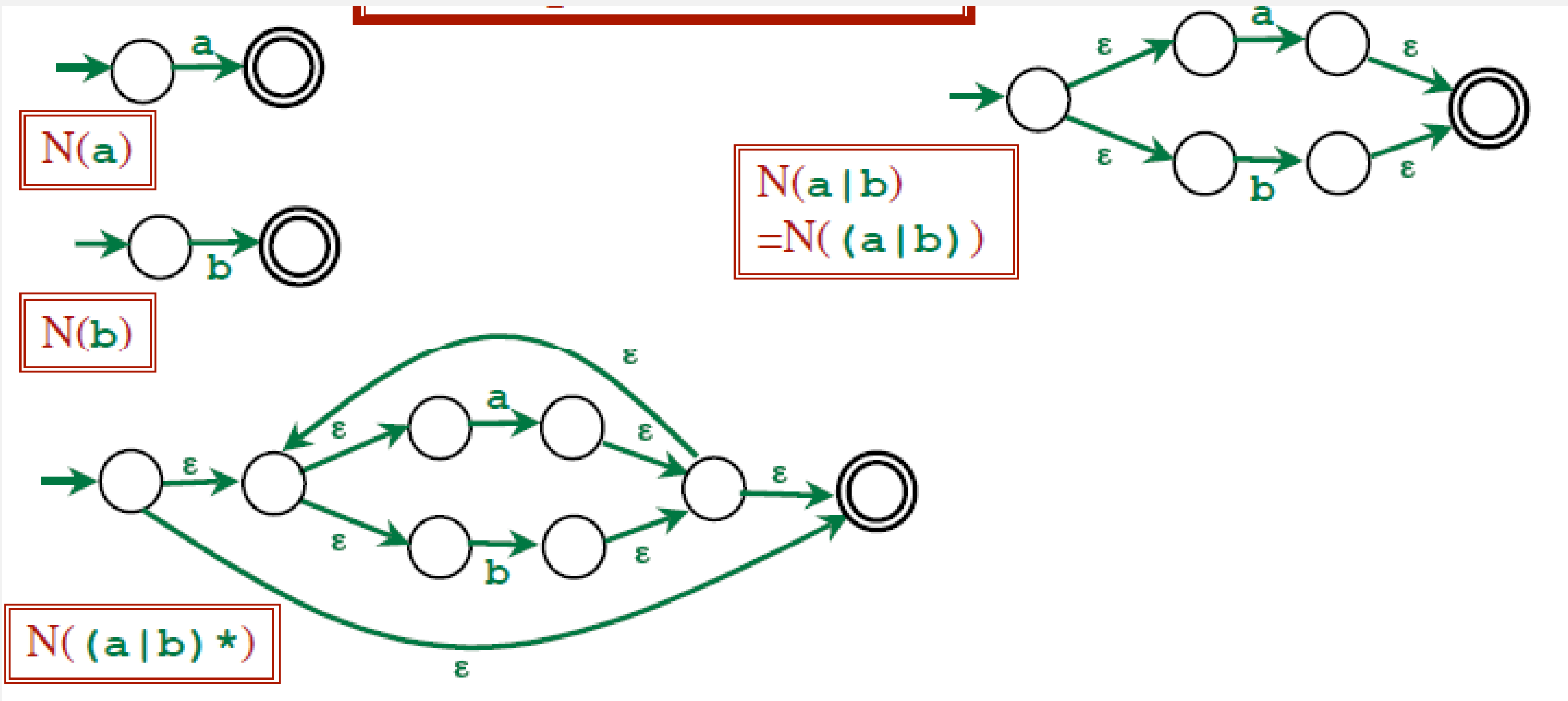
Example: **(a|b)*abb**





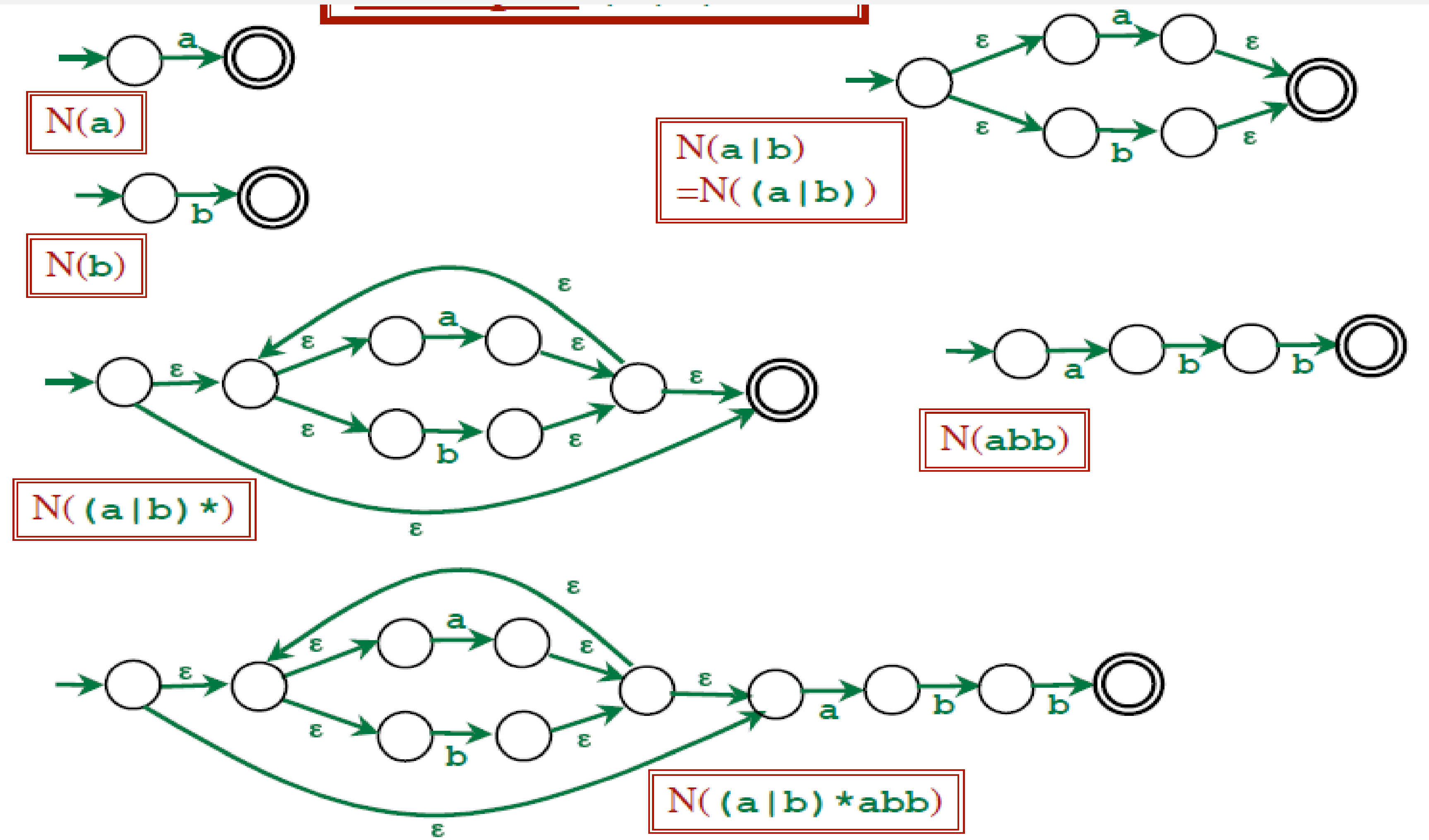
Example

Example: **(a|b)*abb**



Example

Example: **(a|b)*abb**



Thank You