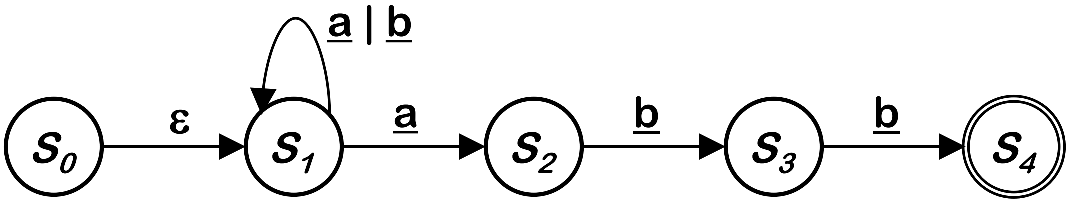
Compiler Design
Fall 2018

Subset Construction Algorithm Animation

Bernd Burgstaller
Yonsei University

Example: Simulating an NFA ``on the fly''

Input = ...



- **On-the-fly simulation** of an NFA is useful in scenarios where a regular expression is used only once (example: in a text editor or IDE).
- In a compiler, the regular expressions for the programming language's tokens are used over and over again → need a more efficient approach → next slide.

Algorithm: NFA $\rightarrow$DFA with Subset Construction

Subset construction works on sets of NFA states.

Each set of NFA states becomes a DFA state.

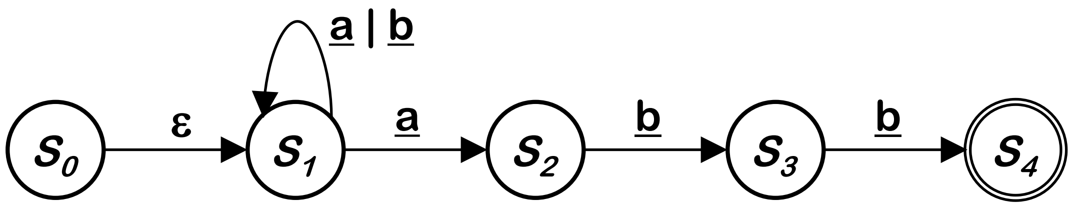
Two key functions

- *Move (S , a)* : set of states reachable from set *S* by a
- *ε-closure (S)* : set of states reachable from set *S* by ε

The algorithm:

- Start state derived from s_0 of the NFA:
  - Take its ε-closure S_0 = ε-closure($\{s_0\}$)
- Compute Move(S_0, α) for each $\alpha \in \Sigma$, and take its ε-closure
- Iterate until no more states are added

Sounds more complex than it is…



Dstates $\leftarrow$ *{};*

add *ε-closure(s_0) as an unmarked state to Dstates;*

while *(there is an unmarked state T in Dstates) {*

  mark *T;*

  for each $\alpha \in \Sigma$ {

    U$\leftarrow$ *ε-closure(Move(T,α))*

    if *(U $\notin$ Dstates)* ***then***

      add *U as an unmarked state to Dstates;*

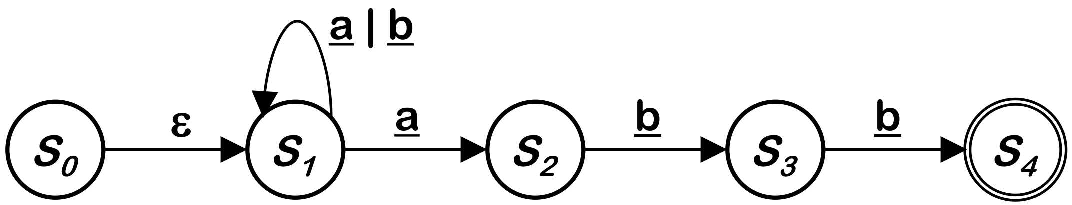
    $\delta[T,\alpha] \leftarrow U$

  }

}

Note:

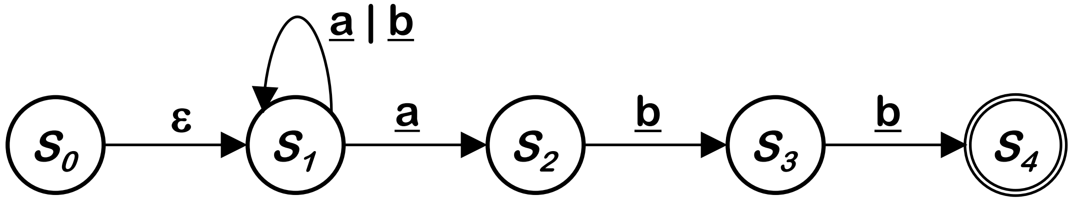
- here δ refers to the transition function of the DFA.
- δ takes a set of NFA-states (= a DFA state) and a symbol α as input, and returns a set of NFA-states (= a DFA state).



***Dstates* = { }**

ε-closure(s_0) = { s_0, s_1 }

Dstates $\leftarrow$ {};
add *ε-closure(s_0) as an unmarked state to Dstates;*
while *(there is an unmarked state T in Dstates) {*
  mark *T;*
  for each $\alpha \in \Sigma$ {
    U $\leftarrow$ *ε-closure(Move(T,α))*
    if *(U $\notin$ Dstates)* ***then***
      add *U as an unmarked state to Dstates;*
    δ*[T,α]* $\leftarrow$ *U*
  }
}



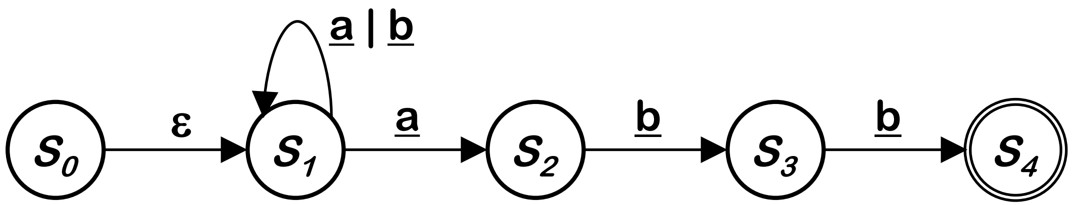
***Dstates* = { { s_0, s_1 } }**

Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an*
unmarked state to Dstates;
while *(there is an unmarked*
state T in Dstates) {
mark *T;*
for each $\alpha \in \Sigma$ *{*
U $\leftarrow$ *ε-closure(Move(T,α))*
if *(U $\notin$ Dstates)* ***then***
add *U as an unmarked*
state to Dstates;
$\delta[T,\alpha] \leftarrow U$
}
}

Dstates contains now one element, i.e., the state { s_0, s_1 } .

Note1: unmarked states are printed in red.

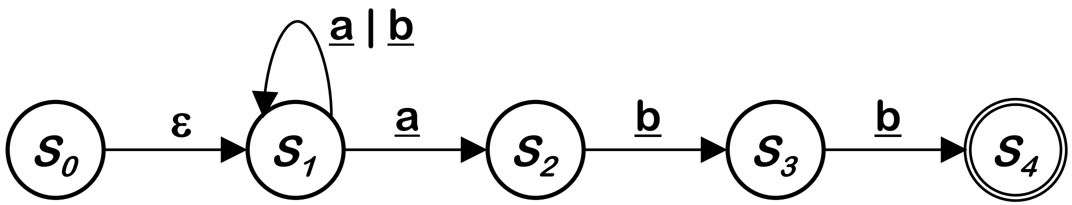
Note2: Dstates means "deterministic states, the states that will make up the DFA".



***Dstates* = { { s_0, s_1 } }**

Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an*
unmarked state to Dstates;
while *(there is an unmarked*
state T in Dstates) {
mark *T;*
for each $\alpha \in \Sigma$ {
U $\leftarrow$ *ε-closure(Move(T,α))*
if *(U $\not\in$ Dstates)* ***then***
add *U as an unmarked*
state to Dstates;
$\delta[T,\alpha] \leftarrow U$
}
}

Now we pick state { s_0, s_1 } from Dstates and mark it.
Note: marked states are printed in green.



Dstates = **{** { s_0, s_1 } **}**

T = { s_0, s_1 }

Σ ***=*** { a, b }

Dstates $\leftarrow$ *{};*

add *ε-closure(s_0) as an*

unmarked state to Dstates;

while *(there is an unmarked*

state T in Dstates) {

mark *T;*

for each $\alpha \in \Sigma$ {

U $\leftarrow$ *ε-closure(Move(T,α))*

if *(U $\notin$ Dstates)* ***then***

add *U as an unmarked*

state to Dstates;

δ*[T,α]* $\leftarrow$ *U*

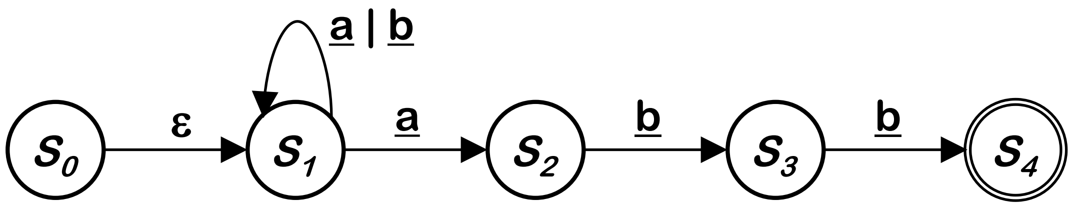
}

}

The **for** loop is going to iterate over all symbols in Σ.

First iteration: a

Second iteration: b.



Dstates ← {};
add ε-closure(s_0) as an
unmarked state to Dstates;
while (there is an unmarked
state T in Dstates) {
mark T;
for each $\alpha \in \Sigma$ {
U ← ε-closure(Move(T,α))
if (U ∉ Dstates) **then**
add U as an unmarked
state to Dstates;
δ[T,α] ← U
}
}

***Dstates* = { { s_0, s_1 } }**

***T* =** { s_0, s_1 }

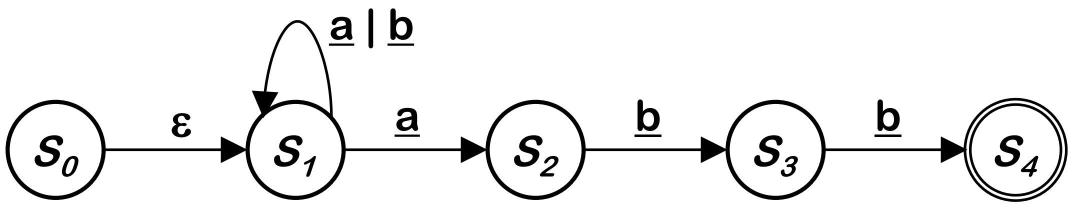
Σ = { a, b }

The **for** loop is going to iterate over all symbols in Σ.
First iteration: α = a
Second iteration: α = b.

Move (T, a) = { s_1, s_2 }

ε-closure({ s1, s2 }) = { s_1, s_2 }

U= { s_1, s_2 }



Dstates ← {};

add *ε-closure(s_0) as an unmarked state to Dstates;*

while *(there is an unmarked state T in Dstates) {*

  mark *T;*

  for each $\alpha \in \Sigma$ {

    U← ε-closure(Move(T,α))

    if *(U ∉ Dstates)* ***then***

      add *U as an unmarked state to Dstates;*

    δ[T,α] ← U

  }

}

***Dstates* = {** { s_0, s_1 } **}**

***T* =** { s_0, s_1 }

Σ = { a, b }

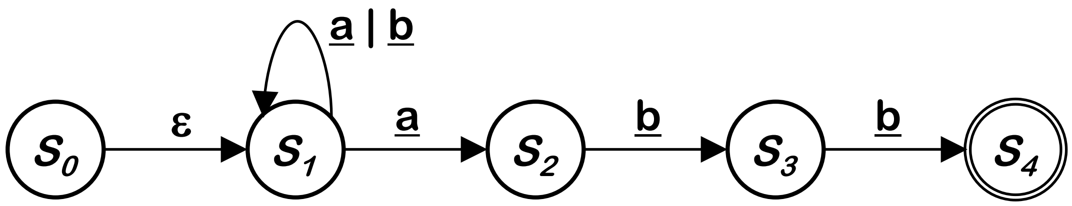
The **for** loop is going to iterate over all symbols in Σ.

First iteration: α =a

Second iteration: α =b.

U= { s_1, s_2 }

State U is not contained in Dstates yet, so we add it as an unmarked state.



***Dstates* = { { s_0, s_1 }, { s_1, s_2 } }**

***T* =** { s_0, s_1 }

Σ **=** { a, b }

U= { s_1, s_2 }

Dstates $\leftarrow$ {};

add ε*-closure(s_0) as an unmarked state to Dstates;*

while *(there is an unmarked state T in Dstates) {*

mark *T;*

for each $\alpha \in \Sigma$ {

$U \leftarrow \varepsilon$*-closure(Move(T,*α*))*

if *(U* $\notin$ *Dstates)* ***then***

add *U as an unmarked state to Dstates;*

$\delta[T,\alpha] \leftarrow U$

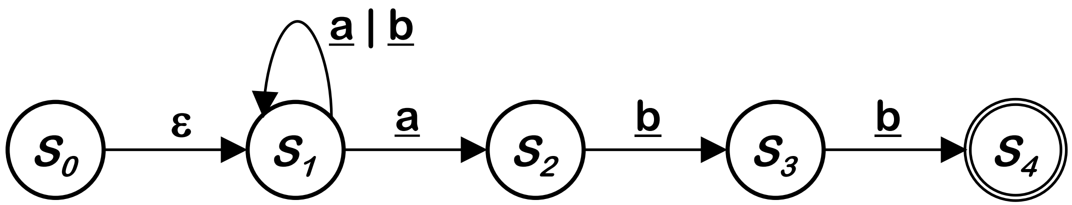
}

}

Here we start assembling the transition function δ for our DFA.

If we read an a in state T, then the DFA will move to state {s_1, s_2} :

δ	a	b
{s_0 ,s_1}	{s_1, s_2}	



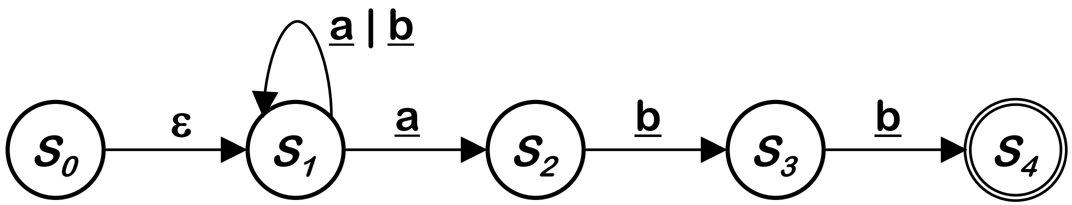
Dstates **= {** { s_0, s_1 }, { s_1, s_2 } **}**

T **=** { s_0, s_1 } Σ **=** { a, b } U= { s_1, s_2 }

Second iteration of the **for** loop: α =b.

Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an*
unmarked state to Dstates;
while *(there is an unmarked*
state T in Dstates) {
mark *T;*
for each $\alpha \in \Sigma$ {
U $\leftarrow$ *ε-closure(Move(T,α))*
if *(U $\notin$ Dstates)* ***then***
add *U as an unmarked*
state to Dstates;
δ[T,α] $\leftarrow$ *U*
}
}

δ	a	b
{s_0 ,s_1}	{s_1, s_2}	



Dstates $\leftarrow$ *{};*

add *ε-closure(s_0) as an*
unmarked state to Dstates;

while *(there is an unmarked*
state T in Dstates) {

mark *T;*

for each $\alpha \in \Sigma$ {

U $\leftarrow$ *ε-closure(Move(T,α))*

if *(U $\notin$ Dstates)* ***then***

add *U as an unmarked*
state to Dstates;

δ[T,α] $\leftarrow$ *U*

}

}

***Dstates* = {** { s_0, s_1 }, { s_1, s_2 } **}**

***T* =** { s_0, s_1 }

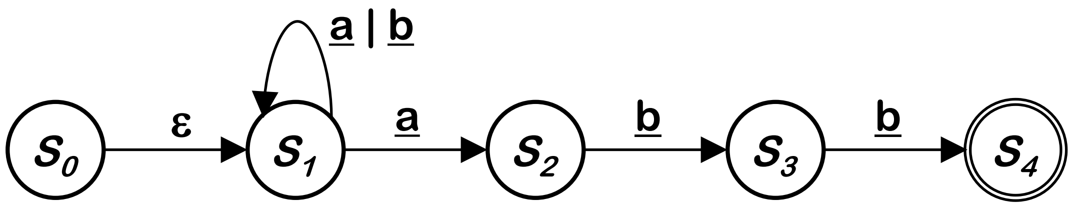
Σ **=** { a, b }

Second iteration of the **for** loop: α = b.

Move (T, b) = { s_1 }

ε-closure({ s1 }) = { s_1 }

U= { s_1 }



Dstates $\leftarrow$ *{};*

add *ε-closure(s_0) as an unmarked state to Dstates;*

while *(there is an unmarked state T in Dstates) {*

  mark *T;*

  for each $\alpha \in \Sigma$ {

    U $\leftarrow$ *ε-closure(Move(T,α))*

    if *(U $\notin$ Dstates)* ***then***

      add *U as an unmarked state to Dstates;*

    δ[T,α] $\leftarrow$ *U*

  }

}

Dstates = { { s_0, s_1 }, { s_1, s_2 } }

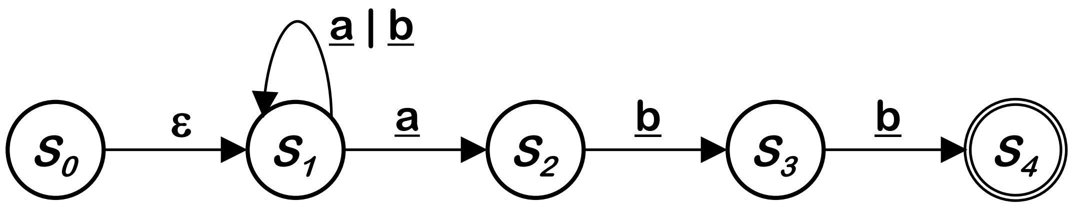
T = { s_0, s_1 }

Σ **=** { a, b }

Second iteration of the **for** loop: α =b.

U= { s_1 }

State U is not contained in Dstates yet, so we add it as an unmarked state.



Dstates $= \{ \{ s_0, s_1 \}, \{ s_1, s_2 \}, \{ s_1 \} \}$

$T = \{ s_0, s_1 \}$

$\Sigma = \{ \underline{a}, \underline{b} \}$

$U = \{ s_1 \}$

Dstates $\leftarrow$ *{};*

add *ε-closure(s_0) as an unmarked state to Dstates;*

while *(there is an unmarked state T in Dstates) {*

mark *T;*

for each $\alpha \in \Sigma$ {

$U \leftarrow \varepsilon$*-closure(Move(T,α))*

if *(U $\notin$ Dstates)* ***then***

add *U as an unmarked state to Dstates;*

$\delta[T,\alpha] \leftarrow U$

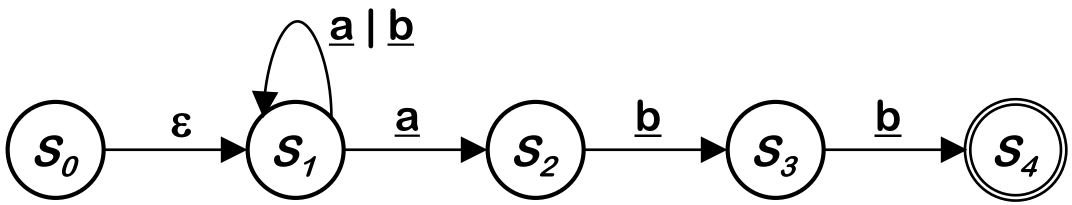
}

}

Second iteration of the **for** loop: $\alpha = \underline{b}$.

If we read a b in state T, then the DFA will move to state $\{s_1\}$:

δ	a	b
$\{s_0, s_1\}$	$\{s_1, s_2\}$	$\{ s_1 \}$



Σ = { $\underline{a}$, $\underline{b}$ }

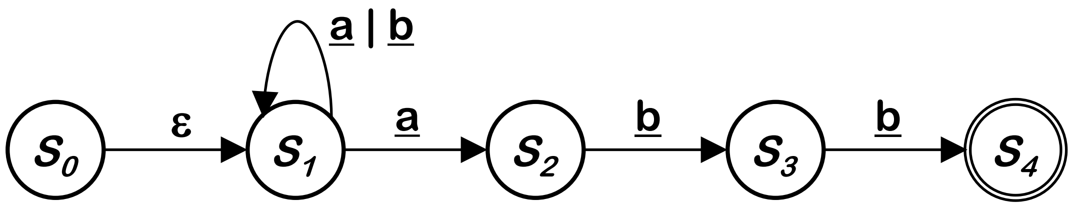
***Dstates* = { { s_0, s_1 }, { s_1, s_2 }, { s_1 } }**

Dstates $\leftarrow$ *{};*
add ε-closure(s_0) *as an*
unmarked state to Dstates;
while *(there is an unmarked*
state T in Dstates) {
mark *T;*
for each $\alpha \in \Sigma$ {
$U \leftarrow \varepsilon$-closure(Move(T,α))
if ($U \notin$ Dstates) **then**
add *U as an unmarked*
state to Dstates;
$\delta[T,\alpha] \leftarrow U$
}
}

Dstates contains now two unmarked states, { $s1$, $s2$ } and { $s1$ }.

We pick state { $s1$, $s2$ } next.

δ	$\underline{a}$	$\underline{b}$
{s_0 ,s_1}	{s_1, s_2}	{ s_1 }



Σ **=** { a, b }

***Dstates =* {** { s_0, s_1 }, { s_1, s_2 }, { s_1 } **}**

T = { s_1, s_2 }

Dstates $\leftarrow$ *{};*

add *ε-closure(s_0) as an*

unmarked state to Dstates;

while *(there is an unmarked*

state T in Dstates) {

mark *T;*

for each $\alpha \in \Sigma$ {

U $\leftarrow$ *ε-closure(Move(T,α))*

if *(U* $\notin$ *Dstates)* ***then***

add *U as an unmarked*

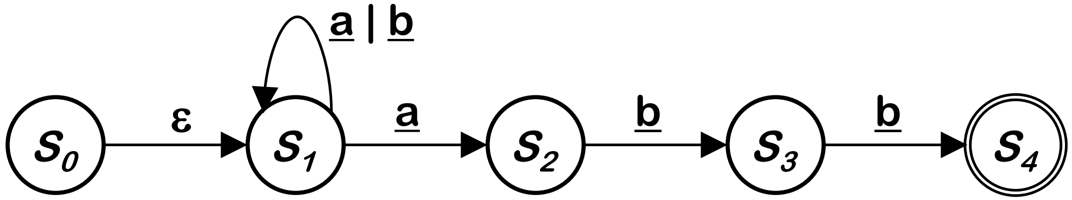
state to Dstates;

δ *[T,α]* $\leftarrow$ *U*

}

}

δ	a	b
{s_0 ,s_1}	{s_1, s_2}	{ s_1 }



Σ **=** { a, b }

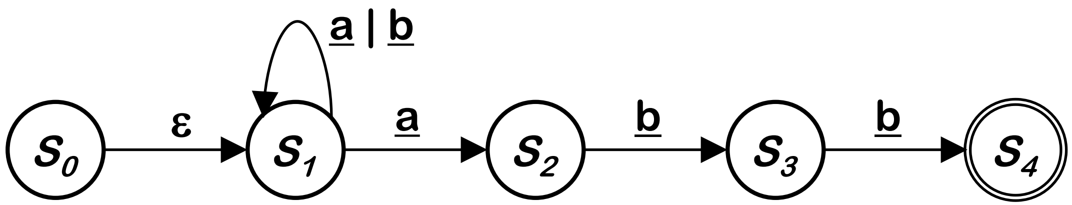
***Dstates =* {** { s_0, s_1 }, { s_1, s_2 }, { s_1 } **}**

***T* =** { s_1, s_2 }

α **=** a

Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an*
unmarked state to Dstates;
while *(there is an unmarked*
state T in Dstates) {
mark *T;*
for each $\alpha \in \Sigma$ {
U $\leftarrow$ *ε-closure(Move(T,α))*
if *(U* $\notin$ *Dstates)* ***then***
add *U as an unmarked*
state to Dstates;
$\delta[T,\alpha] \leftarrow U$
}
}

δ	a	b
{s_0 ,s_1}	{s_1, s_2}	{ s_1 }



Σ = { a, b }

Dstates = { { s_0, s_1 }, { s_1, s_2 }, { s_1 } }

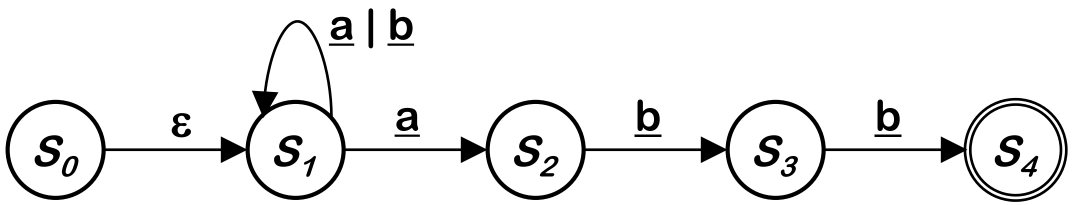
T = { s_1, s_2 }

α = a

Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an*
unmarked state to Dstates;
while *(there is an unmarked*
state T in Dstates) {
mark *T;*
for each $\alpha \in \Sigma$ *{*
U $\leftarrow$ *ε-closure(Move(T,α))*
if *(U $\notin$ Dstates)* ***then***
add *U as an unmarked*
state to Dstates;
δ[T,α] $\leftarrow$ U
}
}

U= { s_1 , s_2 }

δ	a	b
{s_0 ,s_1}	{s_1, s_2}	{ s_1 }



Σ **=** { a, b }

***Dstates* =** { { s_0, s_1 }, { s_1, s_2 }, { s_1 } }

***T* =** { s_1, s_2 }

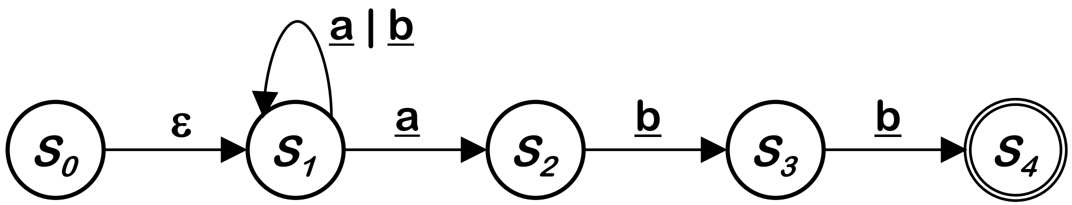
α **=** a

U= { s_1 , s_2 }

Dstates $\leftarrow$ {};
add ε-closure(s_0) as an
unmarked state to Dstates;
while (there is an unmarked
state *T* in Dstates) {
mark *T*;
for each $\alpha \in \Sigma$ {
$U \leftarrow \varepsilon$-closure(Move(T,α))
if ($U \notin$ Dstates) **then**
add *U* as an unmarked
state to Dstates;
$\delta[T,\alpha] \leftarrow U$
}
}

State U is already contained in Dstates (we are currently working on it!). Nothing to be done here...

δ	a	b
{s_0 ,s_1}	{s_1, s_2}	{ s_1 }



Σ = { a, b }

Dstates = { { s_0, s_1 }, { s_1, s_2 }, { s_1 } }

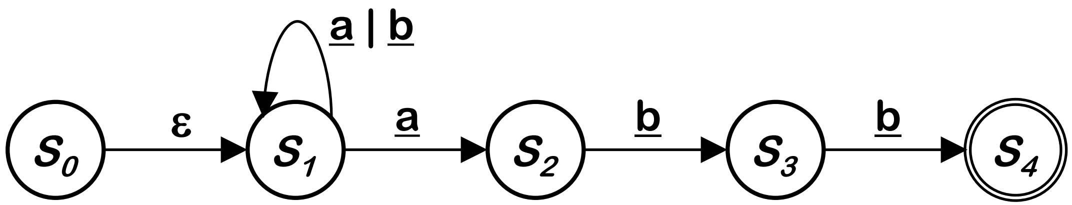
T = { s_1, s_2 }

α = a

U= { s_1 , s_2 }

Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an unmarked state to Dstates;*
while *(there is an unmarked state T in Dstates) {*
  mark *T;*
  for each $\alpha \in \Sigma$ {
    U$\leftarrow$ ε-closure(Move(T,α))
    if *(U $\notin$ Dstates)* ***then***
      add *U as an unmarked state to Dstates;*
    δ[T,α] $\leftarrow$ U
  }
}

δ	a	b
$\{s_0, s_1\}$	$\{s_1, s_2\}$	$\{ s_1 \}$
$\{ s_1, s_2 \}$	$\{ s_1, s_2 \}$	



Σ **=** { a, b }

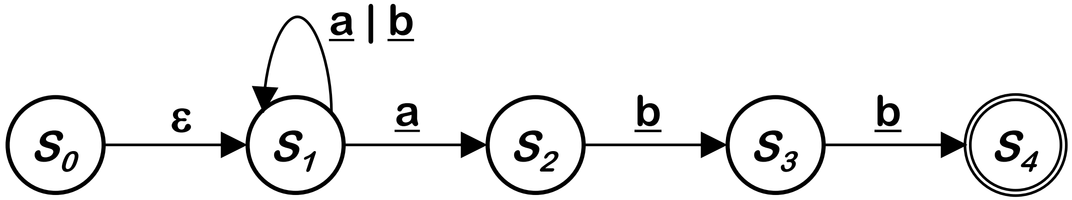
Dstates = **{** { s_0, s_1 }, { s_1, s_2 }, { s_1 } **}**

T = { s_1, s_2 }

α **=** b

Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an unmarked state to Dstates;*
while *(there is an unmarked state T in Dstates) {*
  mark *T;*
  for each $\alpha \in \Sigma$ {
    U$\leftarrow$ ε-closure(Move(T,α))
    if *(U $\notin$ Dstates)* ***then***
      add *U as an unmarked state to Dstates;*
    δ*[T,α]* $\leftarrow$ *U*
  }
}

δ	a	b
{s_0 ,s_1}	{s_1, s_2}	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	



Σ = { a, b }

Dstates = { { s_0, s_1 }, { s_1, s_2 }, { s_1 } }

T = { s_1, s_2 }

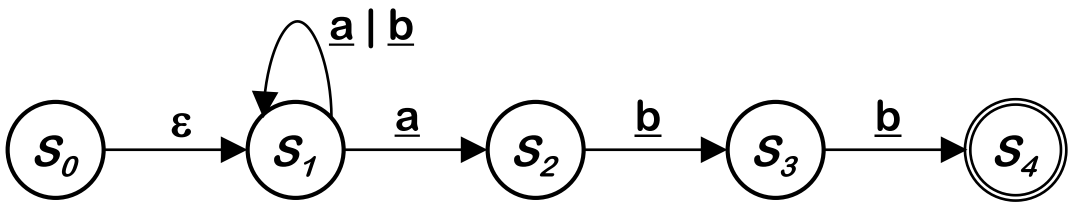
α = b

Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an*
unmarked state to Dstates;

while *(there is an unmarked*
state T in Dstates) {
mark *T;*
for each *$\alpha \in \Sigma$ {*
U $\leftarrow$ ε-closure(Move(T,α))
if *(U $\notin$ Dstates)* ***then***
add *U as an unmarked*
state to Dstates;
δ[T,α] $\leftarrow$ U
}
}

U = { s_1 , s_3 }

δ	a	b
{s_0 ,s_1}	{s_1, s_2}	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	



Σ **=** { a, b }

***Dstates* = {** { s_0, s_1 }, { s_1, s_2 }, { s_1 } **}**

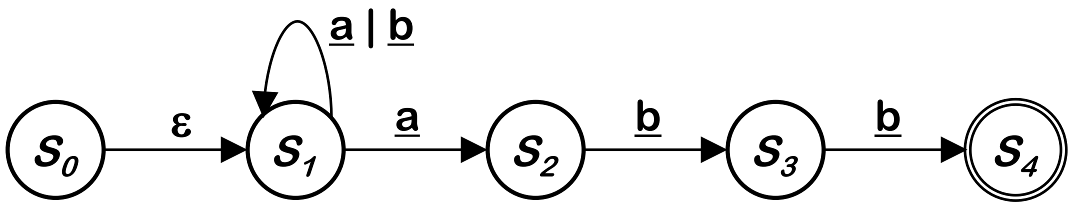
***T* =** { s_1, s_2 }

α **=** b

Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an*
unmarked state to Dstates;
while *(there is an unmarked*
state T in Dstates) {
mark *T;*
for each $\alpha \in \Sigma$ {
U $\leftarrow$ *ε-closure(Move(T,α))*
if *(U $\notin$ Dstates)* ***then***
add *U as an unmarked*
state to Dstates;
δ[T,α] $\leftarrow$ U
}
}

U= { s_1 , s_3 }

δ	a	b
{s_0 ,s_1}	{s_1, s_2}	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	



Σ **=** { a, b }

***Dstates* =** { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 } **}**

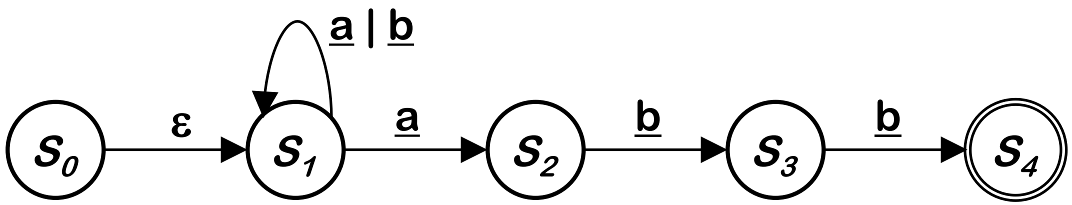
***T* =** { s_1, s_2 }

α **=** b

Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an*
unmarked state to Dstates;
while *(there is an unmarked*
state T in Dstates) {
mark *T;*
for each *$\alpha \in \Sigma$ {*
U $\leftarrow$ ε-closure(Move(T,α))
if *(U $\notin$ Dstates) **then***
add *U as an unmarked*
state to Dstates;
δ[T,α] $\leftarrow$ U
}
}

U= { s_1 , s_3 }

δ	a	b
{s_0 ,s_1}	{s_1, s_2}	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	{ s_1 , s_3 }



Σ **=** { a, b }

***Dstates* =** { { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 } }

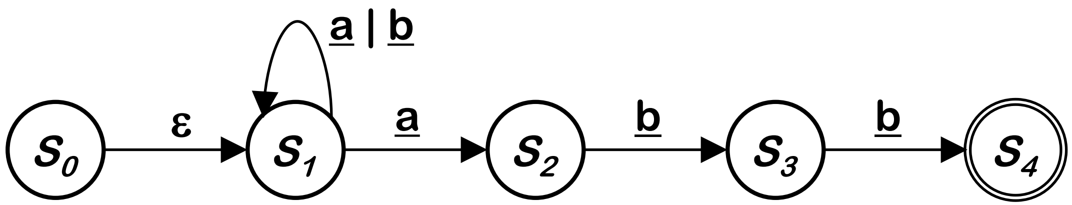
***T* =** { s_1, s_2 }

α **=** b

Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an*
unmarked state to Dstates;
while *(there is an unmarked*
state T in Dstates) {
mark *T;*
for each $\alpha \in \Sigma$ {
U$\leftarrow$ ε-closure(Move(T,α))
if *(U $\notin$ Dstates)* ***then***
add *U as an unmarked*
state to Dstates;
δ[T,α] $\leftarrow$ U
}
}

{ *s1* } is the next unmarked state in Dstates.

δ	a	b
{s_0 ,s_1}	{s_1, s_2}	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	{ s_1 , s_3 }



Σ **=** { a, b }

***Dstates* =** { { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 } }

***T* =** { s_1 }

α **=** a

Dstates $\leftarrow$ *{};*

add *ε-closure(s_0) as an*
unmarked state to Dstates;

while *(there is an unmarked*
state T in Dstates) {

mark *T;*

for each $\alpha \in \Sigma$ {

U $\leftarrow$ ε-closure(Move(T,α))

if *(U $\notin$ Dstates)* ***then***

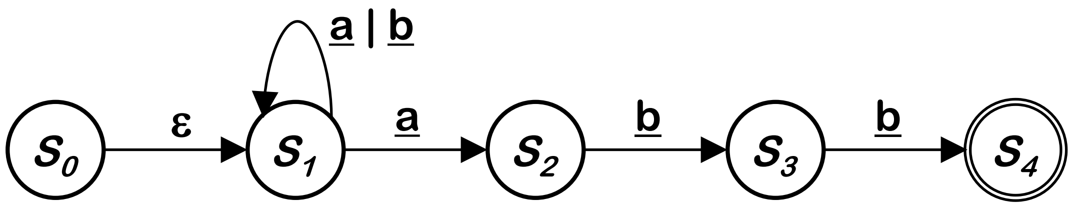
add *U as an unmarked*
state to Dstates;

$\delta[T,\alpha] \leftarrow U$

}

}

δ	a	b
{s_0 ,s_1}	{s_1, s_2}	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	{ s_1 , s_3 }



$\Sigma = \{ \underline{a}, \underline{b} \}$

Dstates = { { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 } }

T = { s_1 }

α = $\underline{a}$

Dstates $\leftarrow$ *{};*

add *ε-closure(s_0) as an*
unmarked state to Dstates;

while *(there is an unmarked*
state T in Dstates) {

mark *T;*

for each $\alpha \in \Sigma$ {

U $\leftarrow$ *ε-closure(Move(T,α))*

if *(U $\notin$ Dstates)* ***then***

add *U as an unmarked*
state to Dstates;

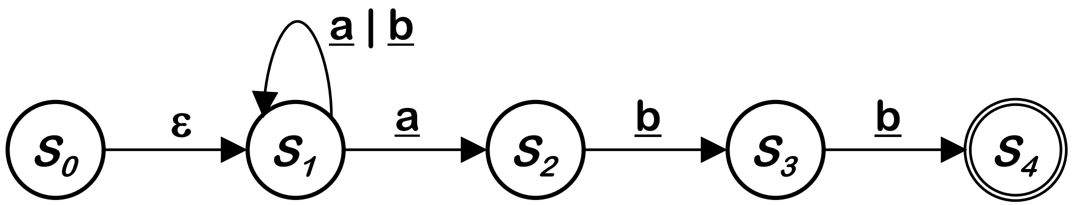
$\delta[T,\alpha] \leftarrow U$

}

}

U= { s_1 , s_2 }

δ	$\underline{a}$	$\underline{b}$
$\{s_0, s_1\}$	$\{s_1, s_2\}$	$\{ s_1 \}$
$\{ s_1, s_2 \}$	$\{ s_1, s_2 \}$	$\{ s_1 , s_3 \}$



Σ **=** { a, b }

Dstates = **{** { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 } **}**

T = { s_1 }

α **=** a

Dstates $\leftarrow$ *{};*

add *ε-closure(s_0) as an*
unmarked state to Dstates;

while *(there is an unmarked*
state T in Dstates) {

mark *T;*

for each $\alpha \in \Sigma$ {

U$\leftarrow$ ε-closure(Move(T,α))

if *(U $\notin$ Dstates)* ***then***

add *U as an unmarked*
state to Dstates;

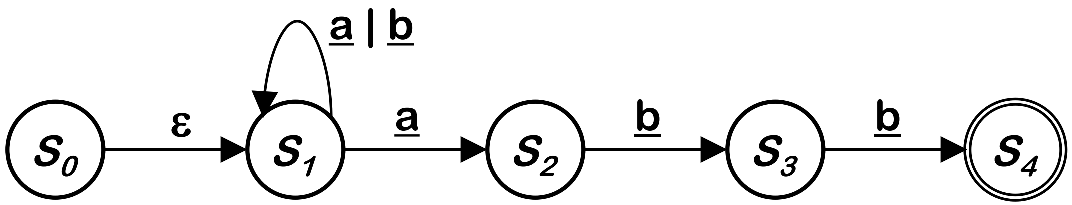
$\delta[T,\alpha] \leftarrow U$

}

}

U= { s_1 , s_2 }

δ	a	b
{s_0 ,s_1}	{s_1, s_2}	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	{ s_1 , s_3 }
{ s_1 }	{ s_1, s_2 }	



Σ **=** { a, b }

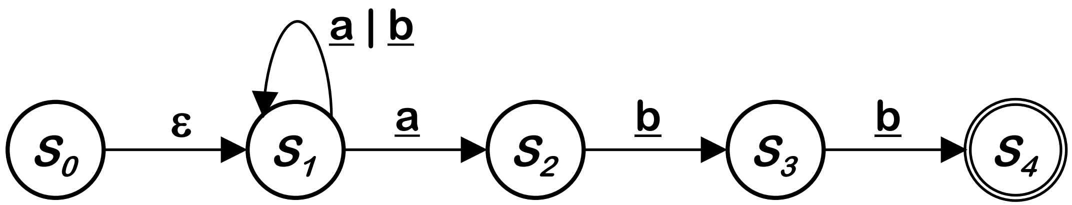
Dstates = **{** { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 } **}**

T = { s_1 }

α **=** b

Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an*
unmarked state to Dstates;
while *(there is an unmarked*
state T in Dstates) {
mark *T;*
for each $\alpha \in \Sigma$ {
U$\leftarrow$ ε-closure(Move(T,α))
if *(U $\notin$ Dstates)* ***then***
add *U as an unmarked*
state to Dstates;
$\delta[T,\alpha] \leftarrow U$
}
}

δ	a	b
{s_0 ,s_1 }	{s_1, s_2 }	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	{ s_1 , s_3 }
{ s_1 }	{ s_1, s_2 }	



Σ **=** { a, b }

Dstates = **{** { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 } **}**

T = { s_1 }

α **=** b

Dstates $\leftarrow$ *{};*

add *ε-closure(s_0) as an*
unmarked state to Dstates;

while *(there is an unmarked*
state T in Dstates) {

mark *T;*

for each $\alpha \in \Sigma$ {

U$\leftarrow$ ε-closure(Move(T,α))

if *(U $\notin$ Dstates)* ***then***

add *U as an unmarked*
state to Dstates;

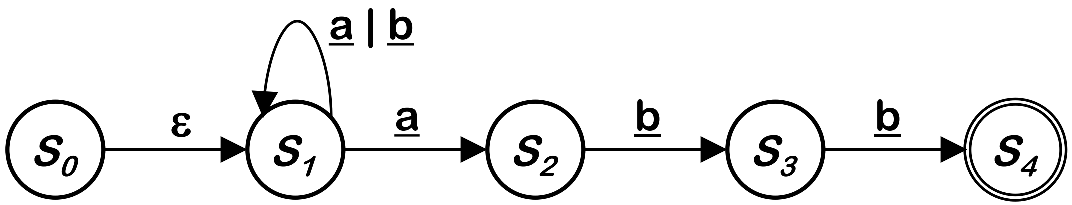
δ[T,α] $\leftarrow$ U

}

}

U= { s_1 }

δ	a	b
{s_0 ,s_1}	{s_1, s_2}	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	{ s_1 , s_3 }
{ s_1 }	{ s_1, s_2 }	



Σ **=** { a, b }

Dstates = **{** { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 } **}**

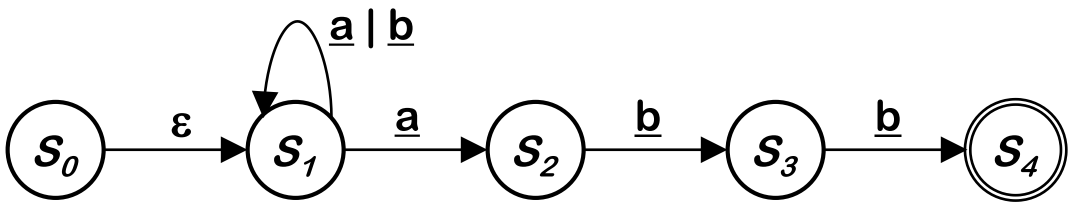
T = { s_1 }

α **=** b

Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an*
unmarked state to Dstates;
while *(there is an unmarked*
state T in Dstates) {
mark *T;*
for each $\alpha \in \Sigma$ {
U $\leftarrow$ ε-closure(Move(T,α))
if *(U $\notin$ Dstates)* ***then***
add *U as an unmarked*
state to Dstates;
δ[T,α] $\leftarrow$ U
}
}

U= { s_1 }

δ	a	b
{s_0 ,s_1}	{s_1, s_2}	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	{ s_1 , s_3 }
{ s_1 }	{ s_1, s_2}	{ s_1 }



Σ = { a, b }

Dstates = { { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 } }

α =

Dstates $\leftarrow$ *{};*

add *ε-closure(s_0) as an unmarked state to Dstates;*

while *(there is an unmarked state T in Dstates) {*

mark *T;*

for each $\alpha \in \Sigma$ {

U$\leftarrow$ ε-closure(Move(T,α))

if *(U $\notin$ Dstates)* ***then***

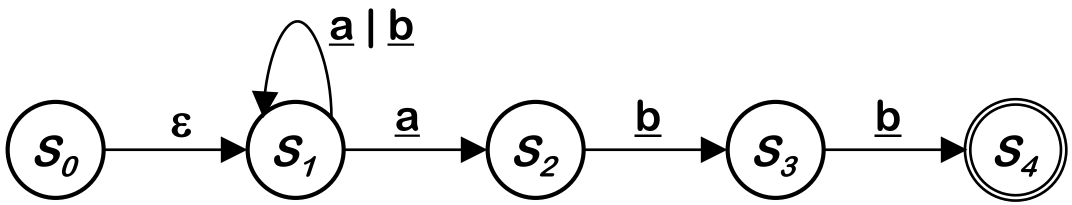
add *U as an unmarked state to Dstates;*

δ[T,α] $\leftarrow$ U

}

}

δ	a	b
{ s_0 , s_1 }	{ s_1, s_2 }	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	{ s_1 , s_3 }
{ s_1 }	{ s_1, s_2 }	{ s_1 }



Σ **=** { a, b }

***Dstates* =** { { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 } }

***T* =** { s1 , s3 }

α **=**

Dstates $\leftarrow$ *{};*

add *ε-closure(s_0) as an*
unmarked state to Dstates;

while *(there is an unmarked*
state T in Dstates) {

mark *T;*

for each $\alpha \in \Sigma$ {

U$\leftarrow$ *ε-closure(Move(T,α))*

if *(U $\notin$ Dstates)* ***then***

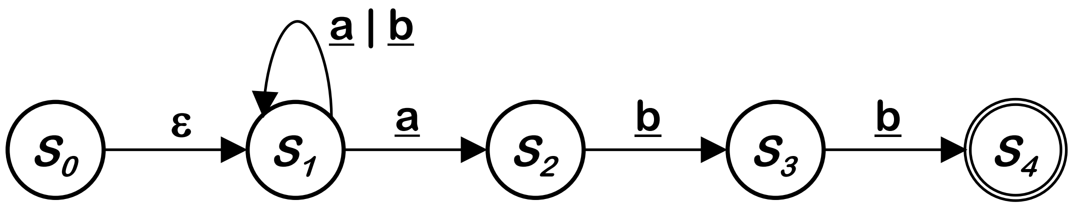
add *U as an unmarked*
state to Dstates;

δ*[T,α]* $\leftarrow$ *U*

}

}

δ	a	b
{s_0 ,s_1 }	{s_1, s_2 }	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	{ s_1 , s_3 }
{ s_1 }	{ s_1, s_2 }	{ s_1 }



Σ **=** { <u>a</u>, <u>b</u> }

Dstates = **{** { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 } **}**

T = { s_1 , s_3 }

α **=** <u>a</u>

Dstates $\leftarrow$ *{};*

add *ε-closure(s_0) as an*

unmarked state to Dstates;

while *(there is an unmarked state T in Dstates) {*

mark *T;*

for each $\alpha \in \Sigma$ {

U$\leftarrow$ ε-closure(Move(T,α))

if *(U $\notin$ Dstates)* ***then***

add *U as an unmarked state to Dstates;*

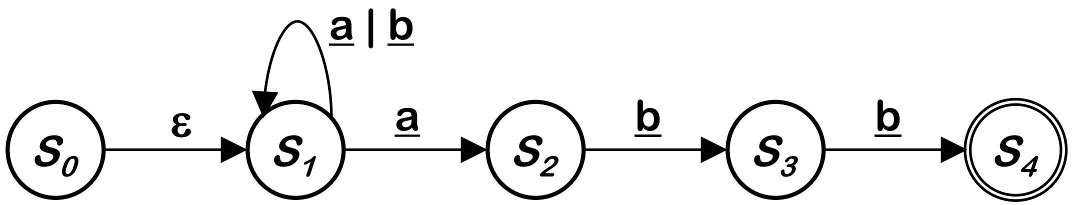
δ [T,α] $\leftarrow$ U

}

}

If we are in state T, reading an <u>a</u>, we get to { s_1, s_2 }. *This transition is added to* δ.

δ	<u>a</u>	<u>b</u>
{s_0 ,s_1 }	{s_1, s_2 }	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	{ s_1 , s_3 }
{ s_1 }	{ s_1, s_2 }	{ s_1 }
{ s_1 , s_3 }	{ s_1, s_2 }	



Σ = { a, b }

Dstates **=** { { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 } }

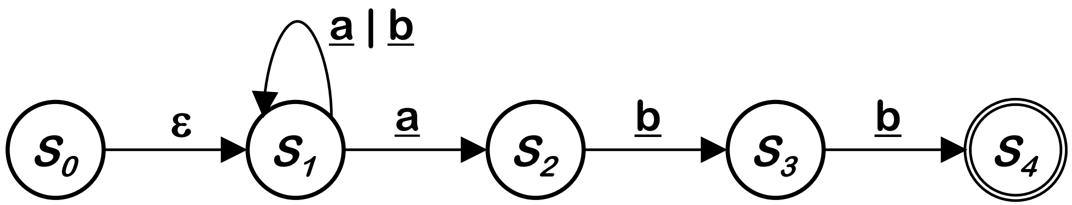
T **=** { s_1 , s_3 }

α = b

Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an*
unmarked state to Dstates;
while *(there is an unmarked*
state T in Dstates) {
mark *T;*
for each *$\alpha \in \Sigma$ {*
$U \leftarrow \varepsilon$-closure(Move(T,α))
if *(U $\notin$ Dstates) **then***
add *U as an unmarked*
state to Dstates;
δ[T,α] $\leftarrow$ U
}
}

If we are in state T, reading a b, we get to { s_1, s_4 } ...

δ	a	b
{ s_0 , s_1 }	{ s_1, s_2 }	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	{ s_1 , s_3 }
{ s_1 }	{ s_1, s_2 }	{ s_1 }
{ s_1 , s_3 }	{ s_1, s_2 }	



Σ **=** { a, b }

***Dstates* =** { { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 } }

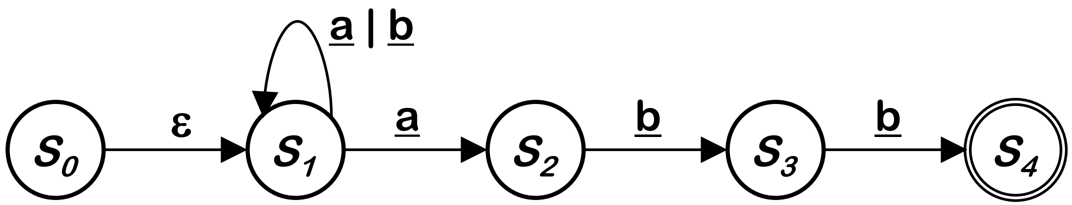
***T* =** { s_1 , s_3 }

α **=** b

Dstates $\leftarrow$ {};
add *ε-closure(s_0) as an*
unmarked state to Dstates;
while *(there is an unmarked*
state T in Dstates) {
mark *T;*
for each $\alpha \in \Sigma$ {
U$\leftarrow$ ε-closure(Move(T,α))
if *(U $\not\in$ Dstates)* ***then***
add *U as an unmarked*
state to Dstates;
δ[T,α] $\leftarrow$ U
}
}

U= { s_1 , s_4 }

δ	a	b
{s_0 ,s_1 }	{s_1, s_2 }	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	{ s_1 , s_3 }
{ s_1 }	{ s_1, s_2 }	{ s_1 }
{ s_1 , s_3 }	{ s_1, s_2 }	



Σ **=** { a, b }

***Dstates* = {** { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 } **}**

***T* =** { s_1 , s_3 }

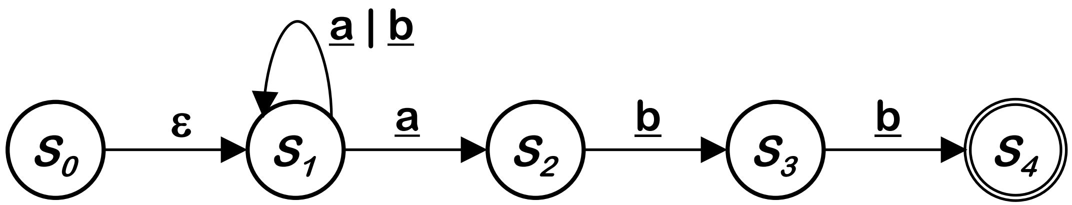
α **=** b

Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an*
unmarked state to Dstates;
while *(there is an unmarked*
state T in Dstates) {
mark *T;*
for each $\alpha \in \Sigma$ {
U$\leftarrow$ ε-closure(Move(T,α))
if *(U $\notin$ Dstates)* ***then***
add *U as an unmarked*
state to Dstates;
δ[T,α] $\leftarrow$ U
}
}

U= { s_1 , s_4 }

U not yet in *Dstates* → add.

δ	a	b
{s_0 ,s_1 }	{s_1, s_2 }	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	{ s_1 , s_3 }
{ s_1 }	{ s_1, s_2 }	{ s_1 }
{ s_1 , s_3 }	{ s_1, s_2 }	



Σ **=** { a, b }

Dstates = **{** { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 }, { s_1 , s_4 } **}**

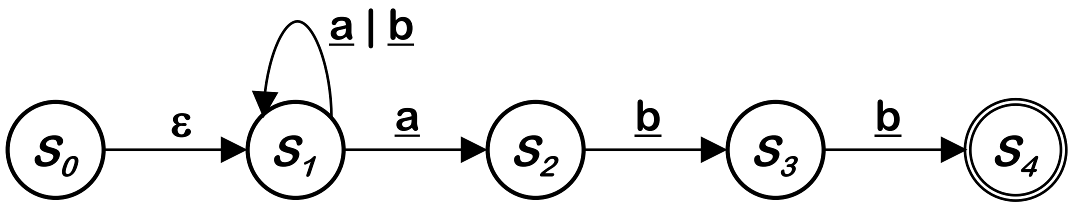
Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an*
unmarked state to Dstates;
while *(there is an unmarked*
state T in Dstates) {
mark *T;*
for each $\alpha \in \Sigma$ {
U $\leftarrow$ *ε-closure(Move(T,α))*
if *(U $\notin$ Dstates)* ***then***
add *U as an unmarked*
state to Dstates;
$\delta[T,\alpha] \leftarrow U$
}
}

T = { s_1 , s_3 }

α **=** b

U= { s_1 , s_4 }

δ	a	b
{ s_0 , s_1 }	{ s_1, s_2 }	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	{ s_1 , s_3 }
{ s_1 }	{ s_1, s_2 }	{ s_1 }
{ s_1 , s_3 }	{ s_1, s_2 }	{ s_1 , s_4 }



Σ = { a, b }

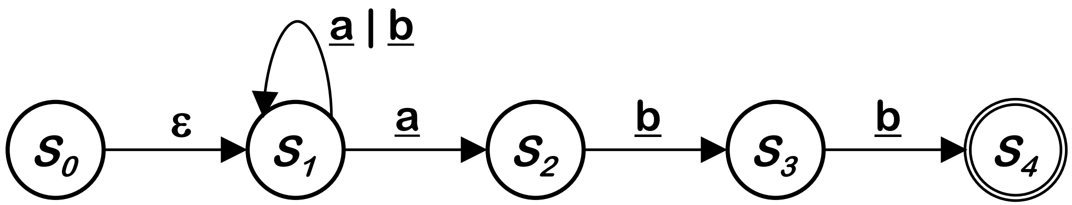
Dstates = { { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 }, { s_1 , s_4 } }

Dstates $\leftarrow$ *{};*
add *ε-closure(s_0) as an unmarked state to Dstates;*
while *(there is an unmarked state T in Dstates) {*
  mark *T;*
  for each $\alpha \in \Sigma$ {
    U $\leftarrow$ *ε-closure(Move(T,α))*
    if *(U $\notin$ Dstates)* ***then***
      add *U as an unmarked state to Dstates;*
    δ *[T,α]* $\leftarrow$ *U*
  }
}

T={ s_1 , s_4 } is the last unmarked state.

Reading an a takes us to { s_1, s_2 }.
Reading a b takes us to { s_1 }.
Those two transitions are added to δ.

δ	a	b
{ s_0 , s_1 }	{ s_1, s_2 }	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	{ s_1 , s_3 }
{ s_1 }	{ s_1, s_2 }	{ s_1 }
{ s_1 , s_3 }	{ s_1, s_2 }	{ s_1 , s_4 }
{ s_1 , s_4 }	{ s_1, s_2 }	{ s_1 }



Σ **=** { a, b }

***Dstates* =** { { s_0, s_1 }, { s_1, s_2 }, { s_1 }, { s_1 , s_3 }, { s_1 , s_4 } }

Dstates ← {};
add *ε-closure(s_0) as an unmarked state to Dstates;*
while *(there is an unmarked state T in Dstates) {*
  mark *T;*
  for each *α ∈ Σ {*
    U← ε-closure(Move(T,α))
    if *(U ∉ Dstates)* ***then***
      add *U as an unmarked state to Dstates;*
    δ [T,α] ← U
  }
}

No more unmarked states in Dstates.

The algorithm stops.

δ	a	b
{s_0 ,s_1 }	{s_1, s_2 }	{ s_1 }
{ s_1, s_2 }	{ s_1, s_2 }	{ s_1 , s_3 }
{ s_1 }	{ s_1, s_2 }	{ s_1 }
{ s_1 , s_3 }	{ s_1, s_2 }	{ s_1 , s_4 }
{ s_1 , s_4 }	{ s_1, s_2 }	{ s_1 }

δ	<u>a</u>	<u>b</u>
$\{s_0, s_1\}$	$\{s_1, s_2\}$	$\{s_1\}$
$\{s_1, s_2\}$	$\{s_1, s_2\}$	$\{s_1, s_3\}$
$\{s_1\}$	$\{s_1, s_2\}$	$\{s_1\}$
$\{s_1, s_3\}$	$\{s_1, s_2\}$	$\{s_1, s_4\}$
$\{s_1, s_4\}$	$\{s_1, s_2\}$	$\{s_1\}$

All DFA states that contain an accepting NFA state become accepting states in the DFA!

The DFA for the above transition function δ :