LEX10: Regexps directly to DFA*

Lexical Analysis

CMPT 379: Compilers

Instructor: Anoop Sarkar

anoopsarkar.github.io/compilers-class

*This algorithm was first used by Alfred Aho in egrep and later used in awk, lex and flex.

Regexp with Distinct Symbols

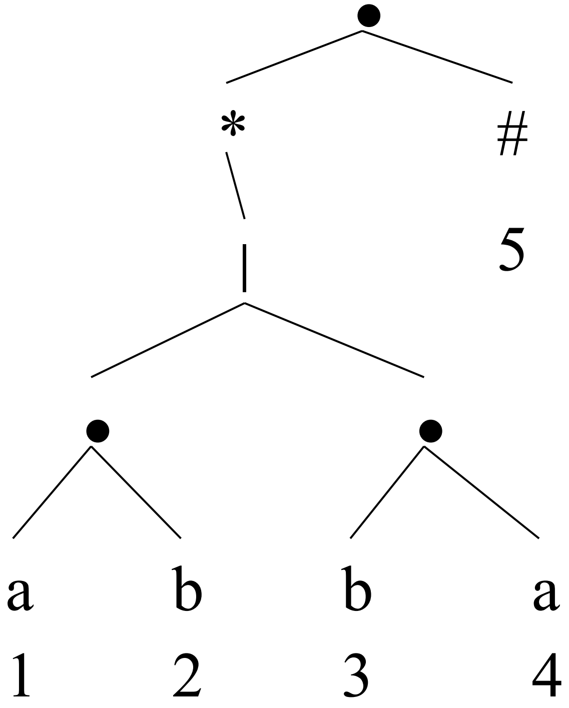
- Associate with each occurrence of a symbol in a regular expression a position
  - Add an end marker ((ab)|(ba))*#
- For example: ((ab)|(ba))*#
  - There are 5 positions:

1:a
2:b
3:b
4:a
5:#

Regexp with Distinct Symbols

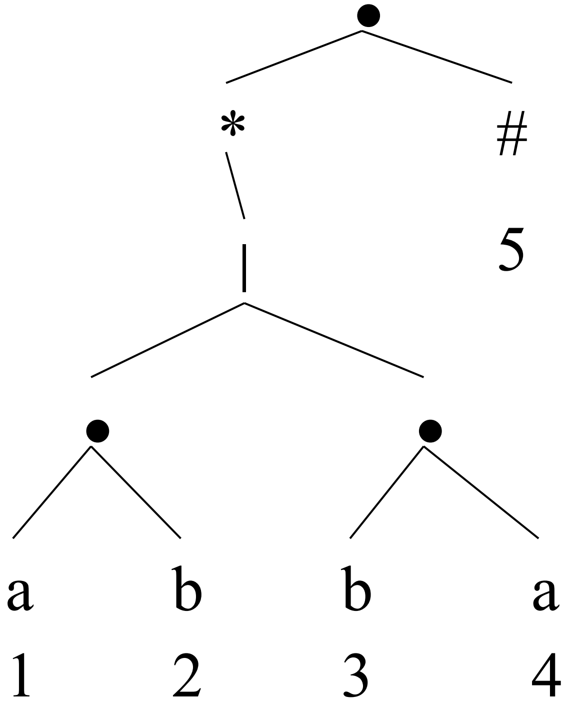
- Associate with each occurrence of a symbol in a regular expression a position
  - Add an end marker ((ab)|(ba))*#

Regexp to DFA: ((ab)|(ba))*#



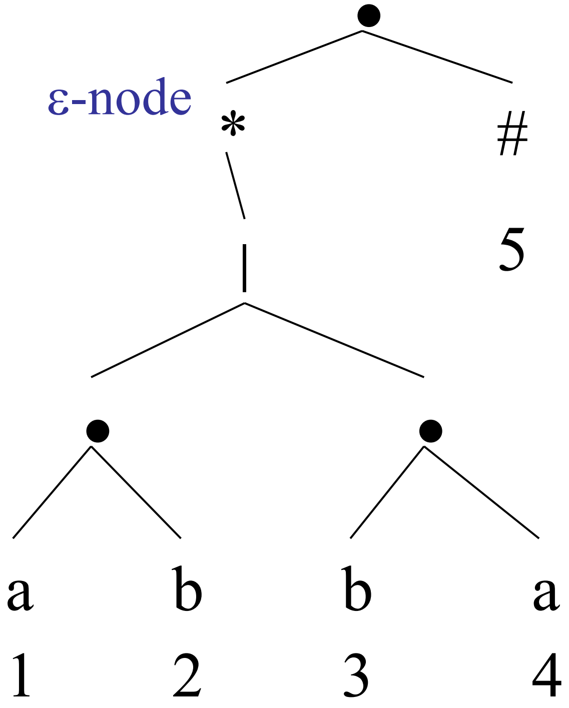
Regexp to DFA: ((ab)|(ba))*#

- ε-node: if the sub-expression has ε in its language



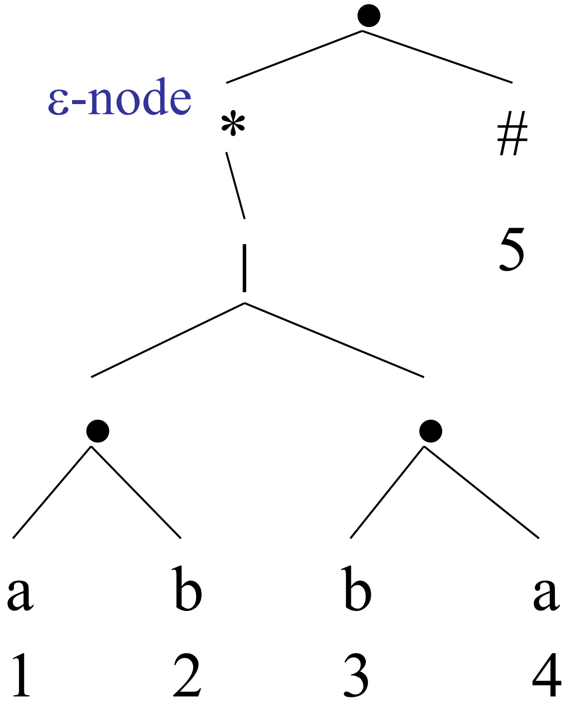
Regexp to DFA: ((ab)|(ba))*#

- ε-node: if the sub-expression has ε in its language



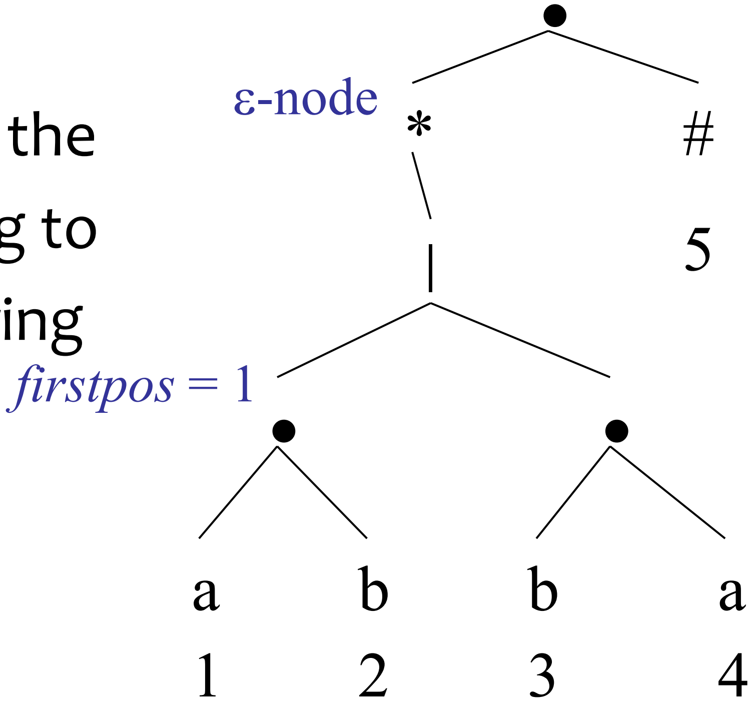
Regexp to DFA: ((ab)|(ba))*#

- ε-node: if the sub-expression has ε in its language
- firstpos(n): the set of positions in the subtree rooted at n corresponding to the first symbol of at least one string



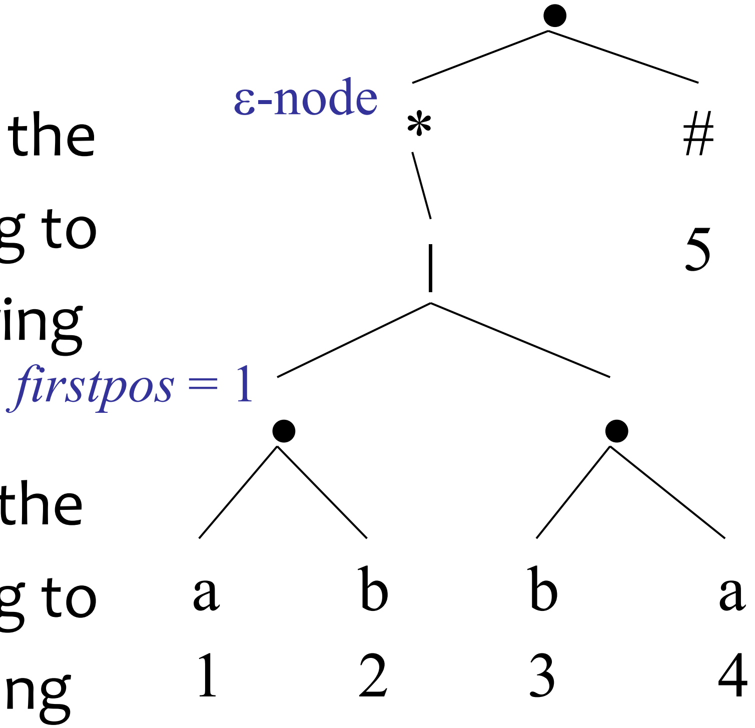
Regexp to DFA: ((ab)|(ba))*#

- ε-node: if the sub-expression has ε in its language
- firstpos(n): the set of positions in the subtree rooted at n corresponding to the first symbol of at least one string



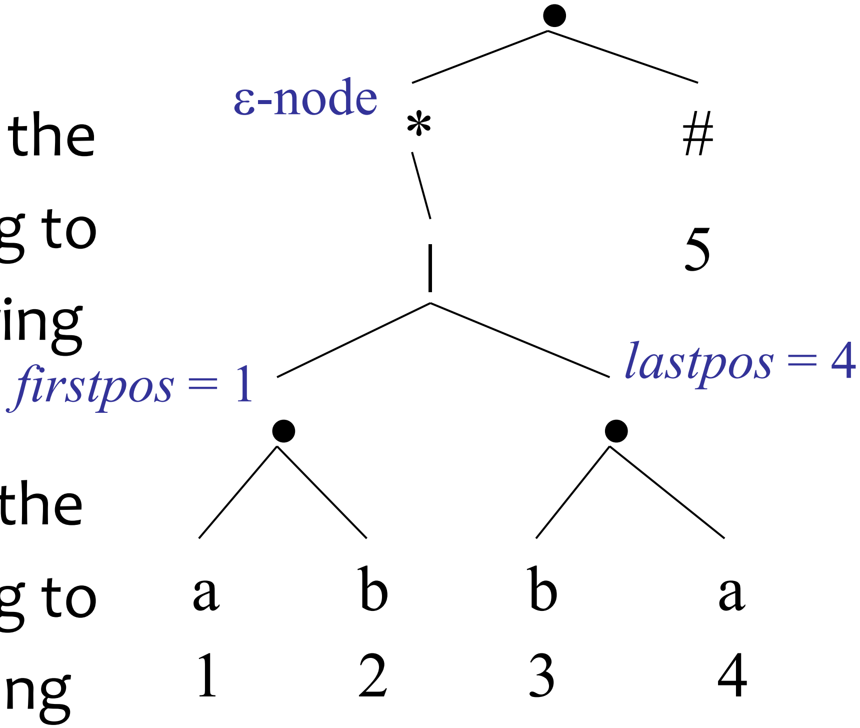
Regexp to DFA: ((ab)|(ba))*#

- ε-node: if the sub-expression has ε in its language
- firstpos(n): the set of positions in the subtree rooted at n corresponding to the first symbol of at least one string
- lastpos(n): the set of positions in the subtree rooted at n corresponding to the last symbol of at least one string

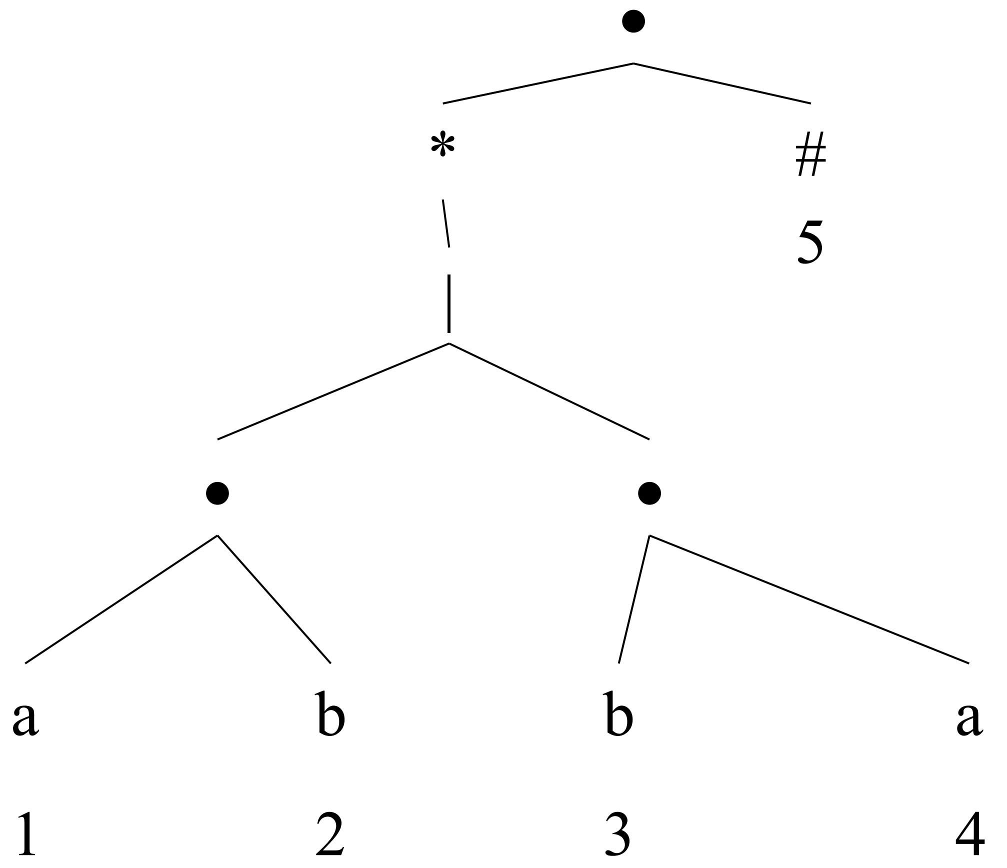


Regexp to DFA: ((ab)|(ba))*#

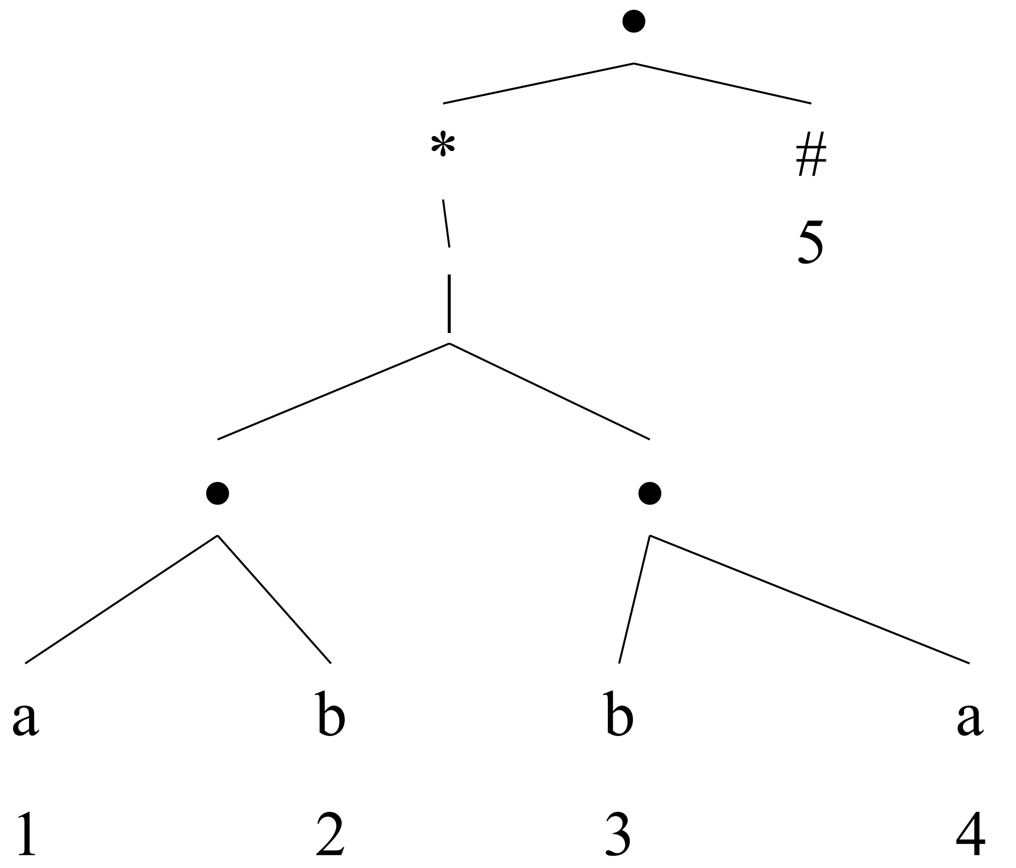
- ε-node: if the sub-expression has ε in its language
- firstpos(n): the set of positions in the subtree rooted at n corresponding to the first symbol of at least one string
- lastpos(n): the set of positions in the subtree rooted at n corresponding to the last symbol of at least one string



Regexp to DFA: ((ab)|(ba))*#

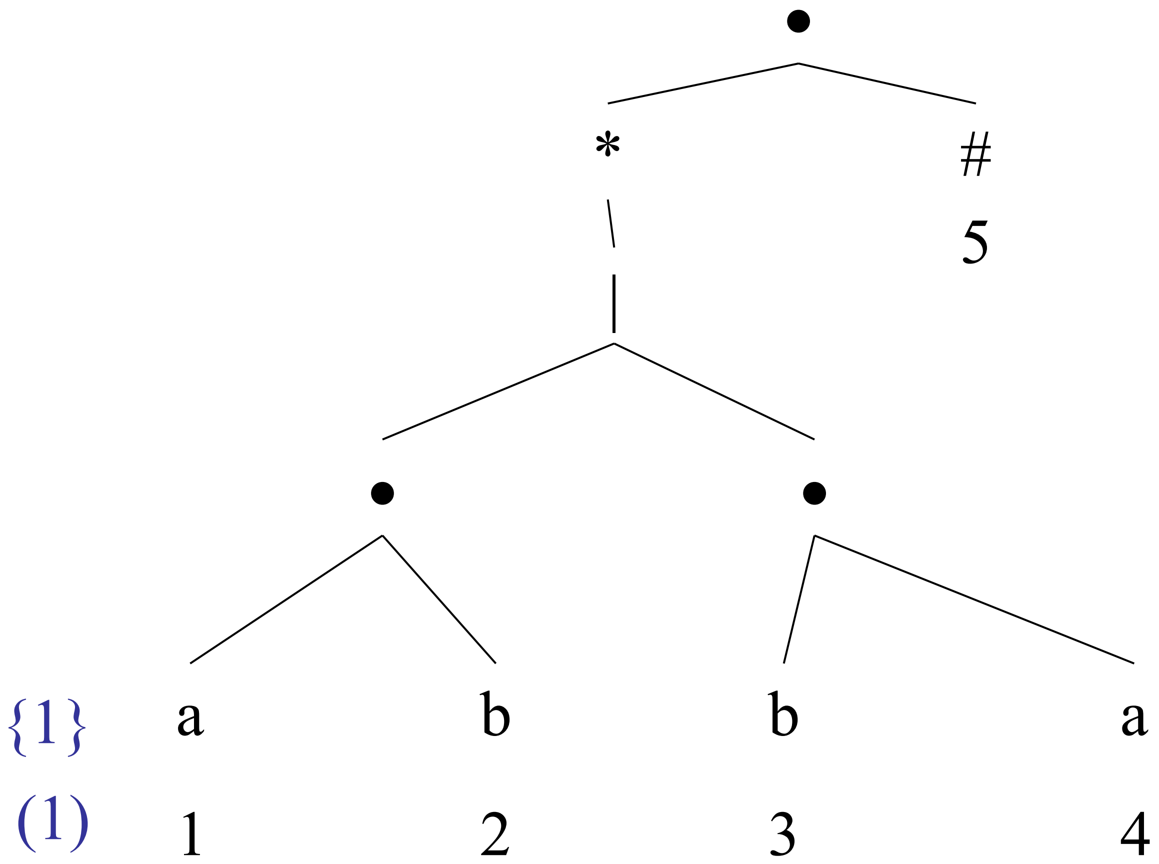


Regexp to DFA: ((ab)|(ba))*#



firstpos = {}
lastpos = ()

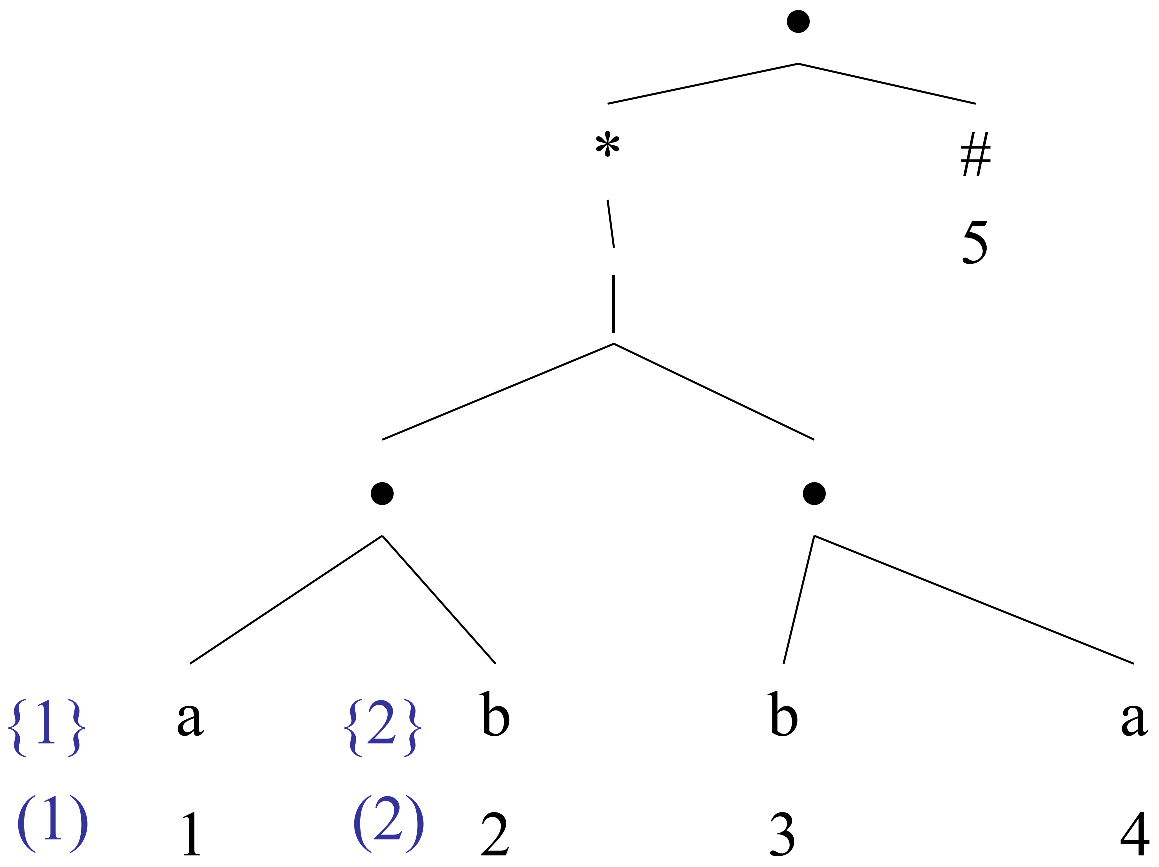
Regexp to DFA: ((ab)|(ba))*#



firstpos = {}

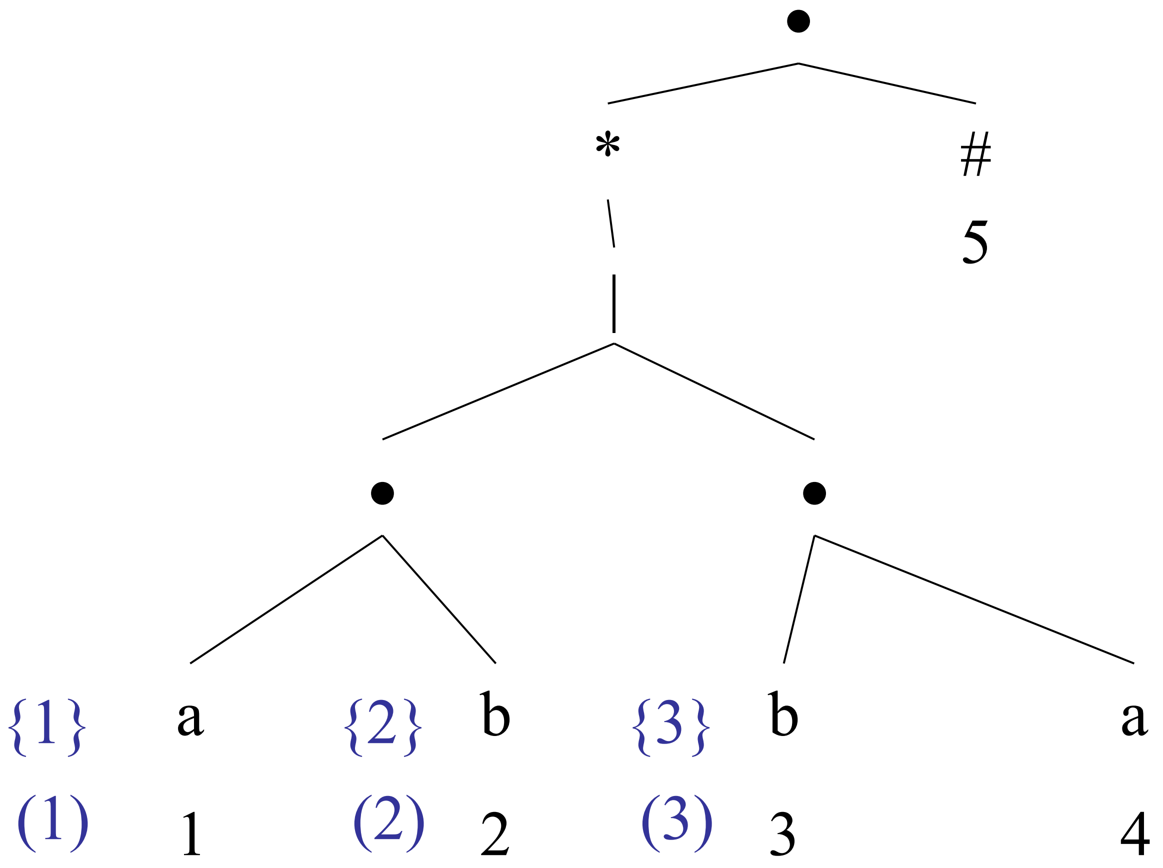
lastpos = ()

Regexp to DFA: ((ab)|(ba))*#



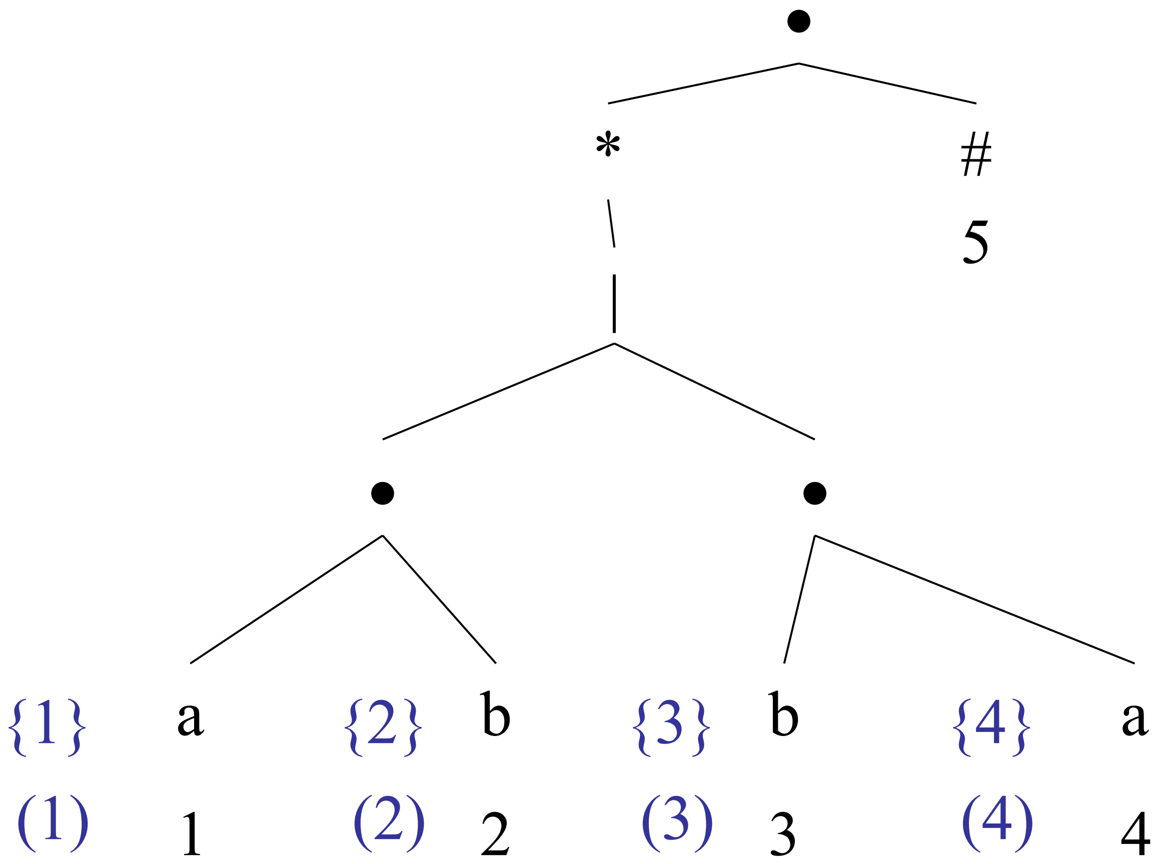
firstpos = {}
lastpos = ()

Regexp to DFA: ((ab)|(ba))*#



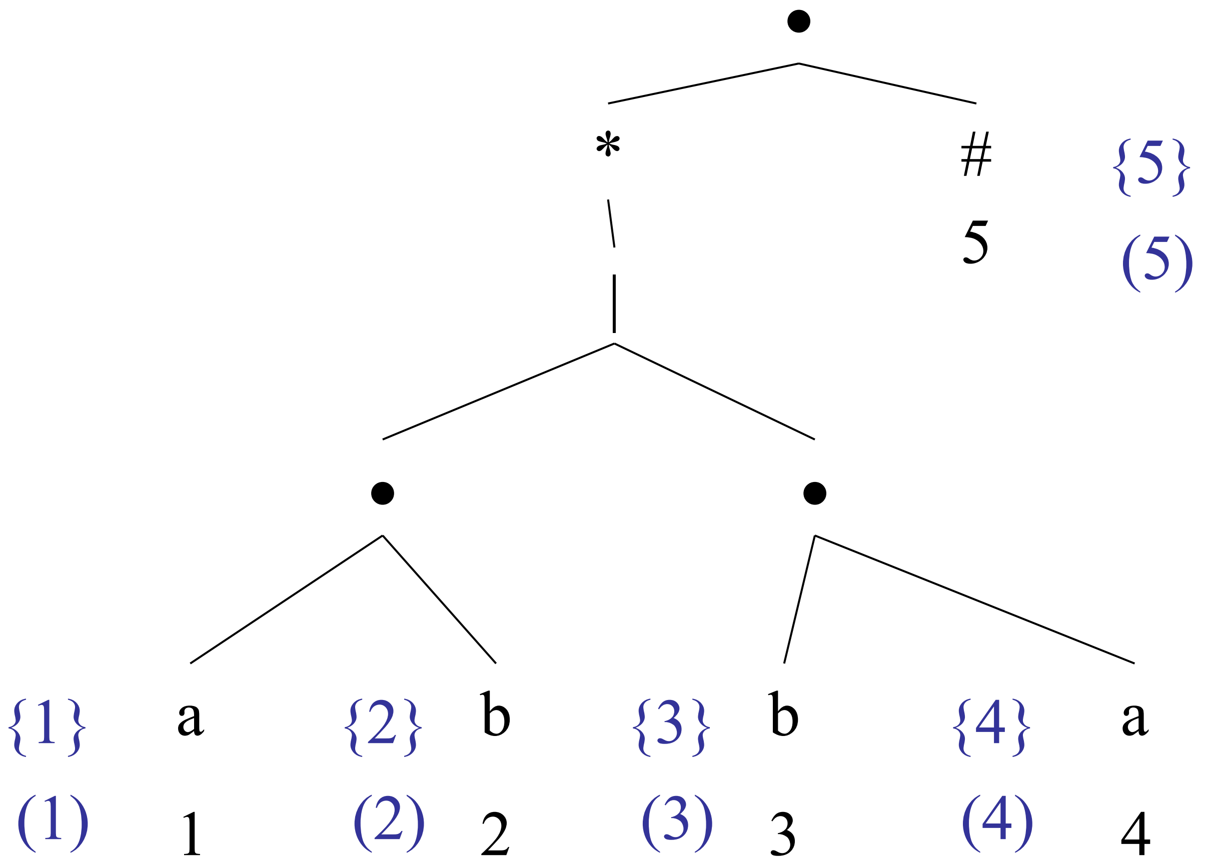
firstpos = {}
lastpos = ()

Regexp to DFA: ((ab)|(ba))*#



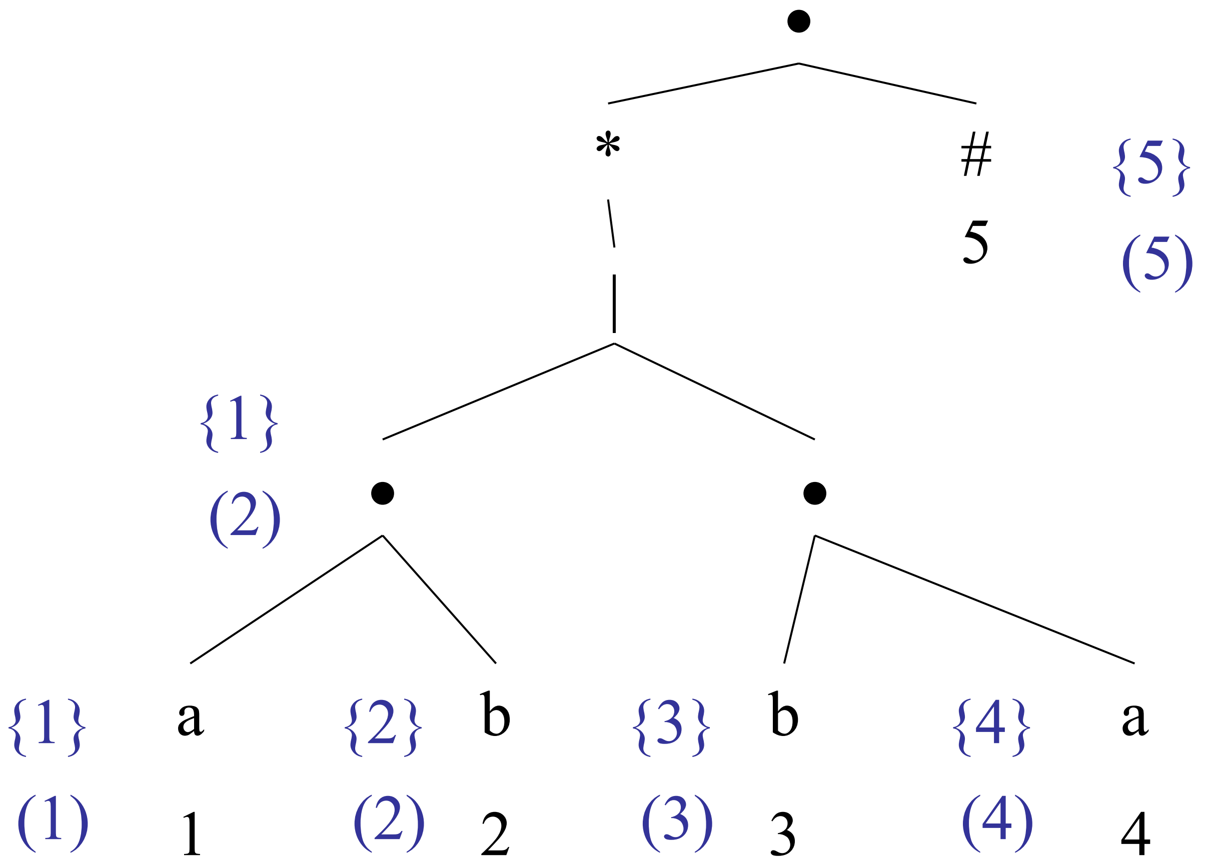
firstpos = {}
lastpos = ()

Regexp to DFA: ((ab)|(ba))*#



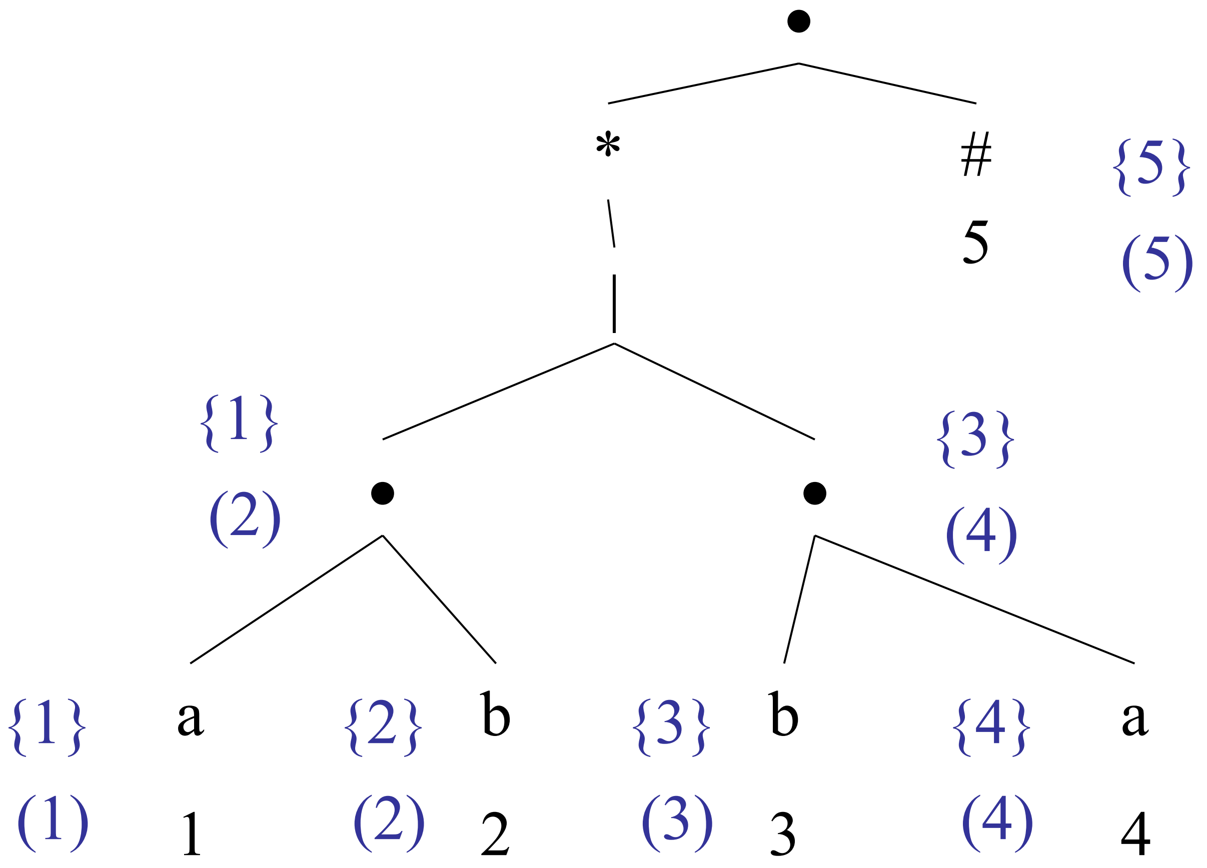
firstpos = { }
lastpos = ()

Regexp to DFA: ((ab)|(ba))*#



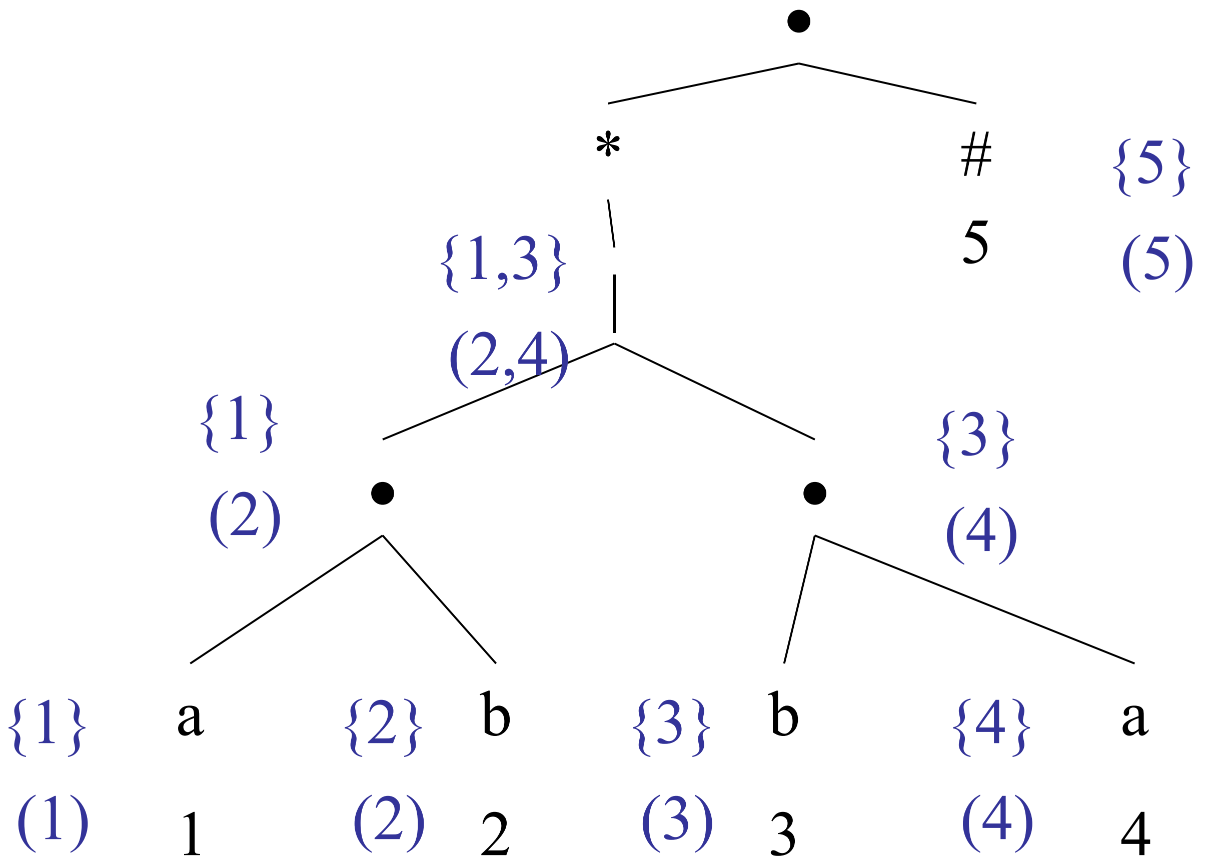
firstpos = { }
lastpos = ( )

Regexp to DFA: ((ab)|(ba))*#



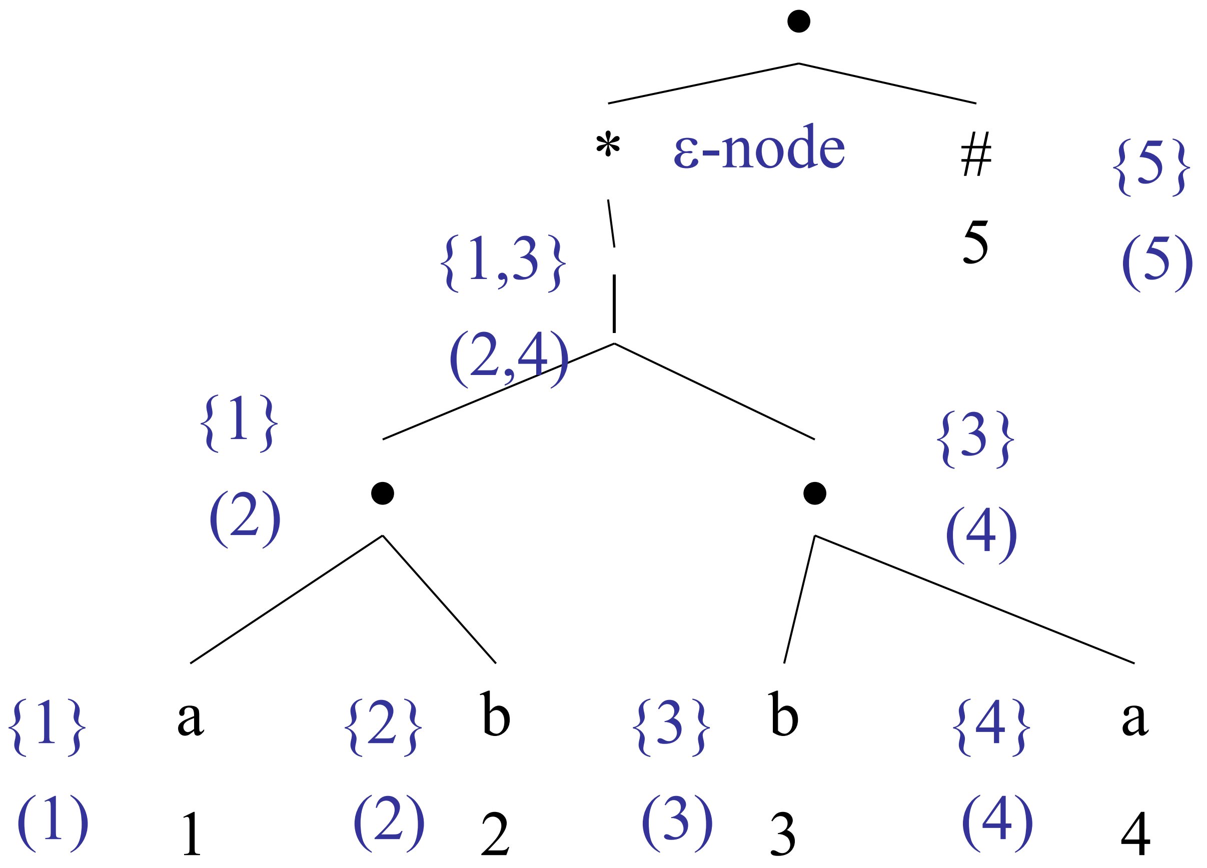
firstpos = {}
lastpos = ()

Regexp to DFA: ((ab)|(ba))*#



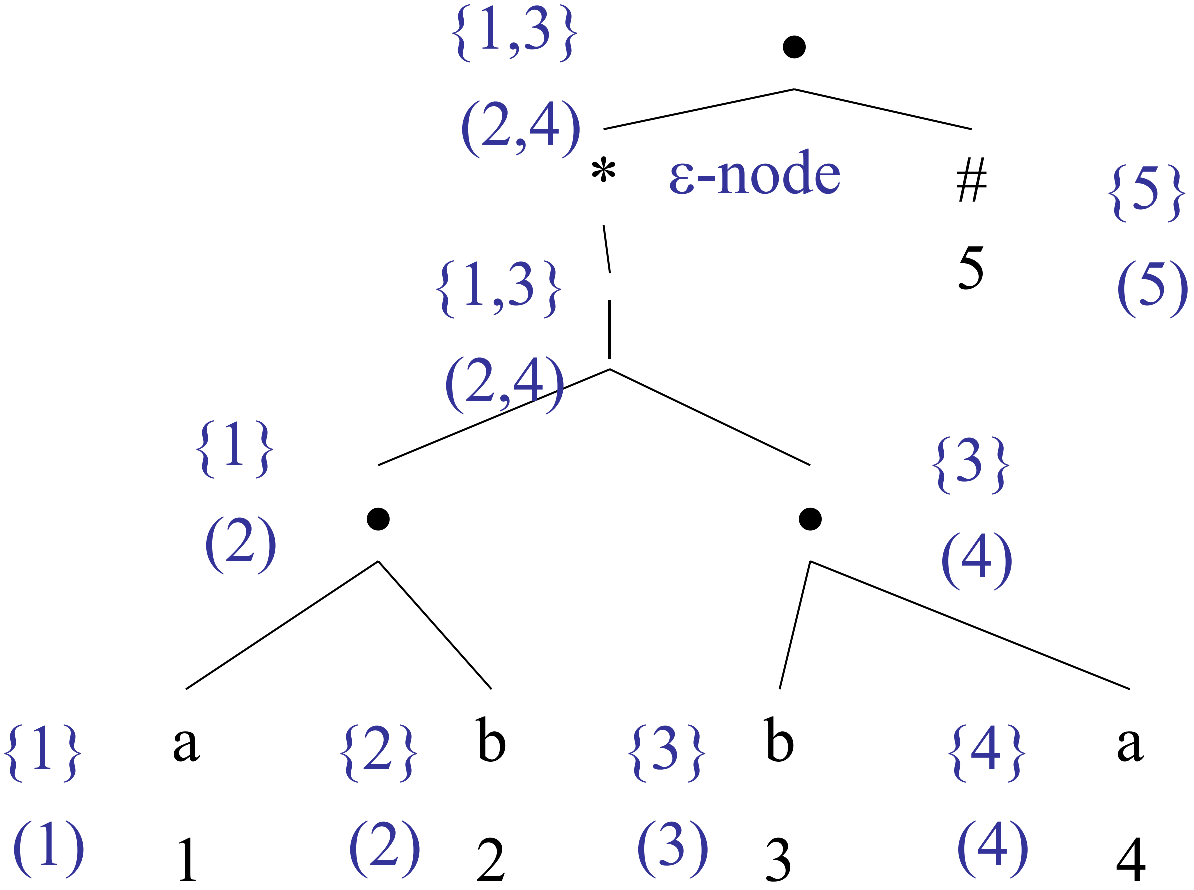
firstpos = {}
lastpos = ()

Regexp to DFA: ((ab)|(ba))*#



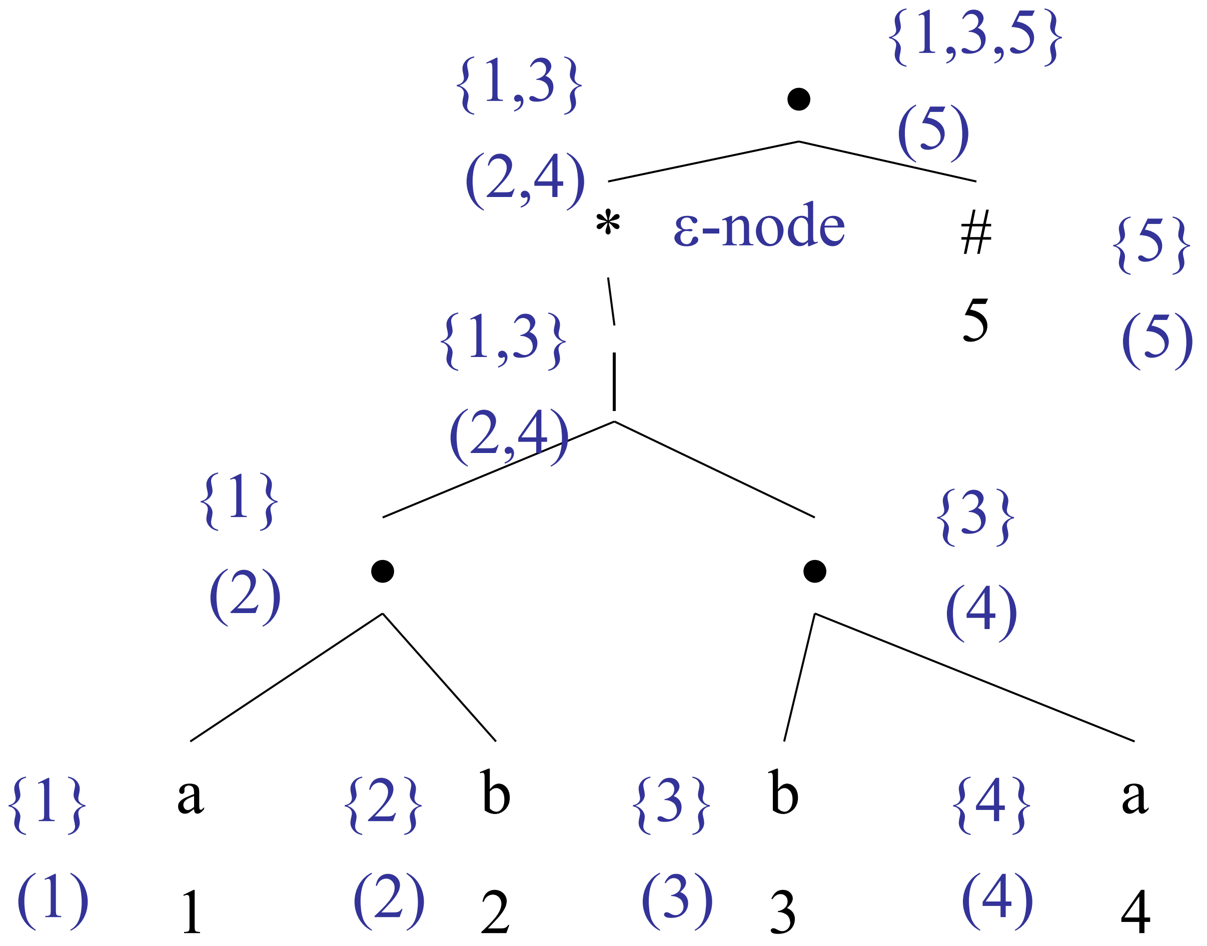
firstpos = {}
lastpos = ()

Regexp to DFA: ((ab)|(ba))*#



firstpos = {}
lastpos = ()

Regexp to DFA: ((ab)|(ba))*#



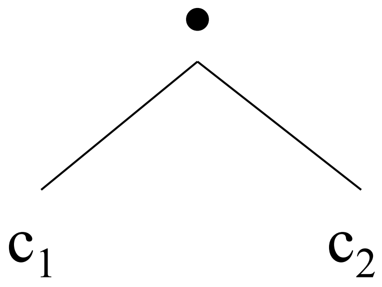
firstpos = {}
lastpos = ()

Regexp to DFA: *followpos*

- *followpos(p):* tells us which positions can follow a position *p*
- There are two rules that use the *firstpos* {} and *lastpos* () information

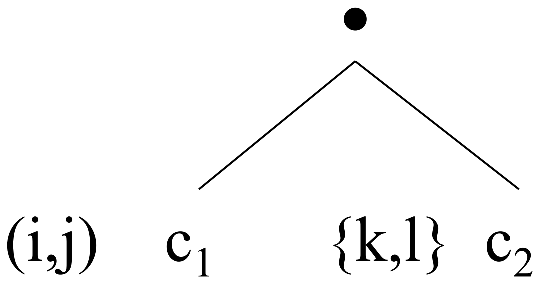
Regexp to DFA: *followpos*

- *followpos(p):* tells us which positions can follow a position *p*
- There are two rules that use the *firstpos* {} and *lastpos* () information



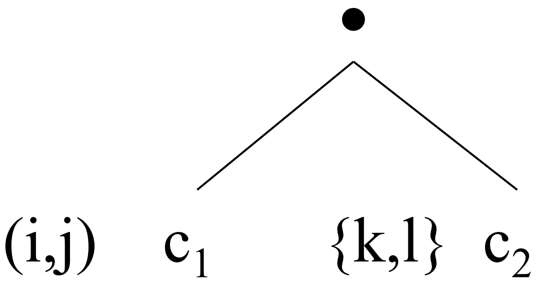
Regexp to DFA: *followpos*

- *followpos(p):* tells us which positions can follow a position *p*
- There are two rules that use the *firstpos* {} and *lastpos* () information



Regexp to DFA: *followpos*

- *followpos(p):* tells us which positions can follow a position *p*
- There are two rules that use the *firstpos* {} and *lastpos* () information

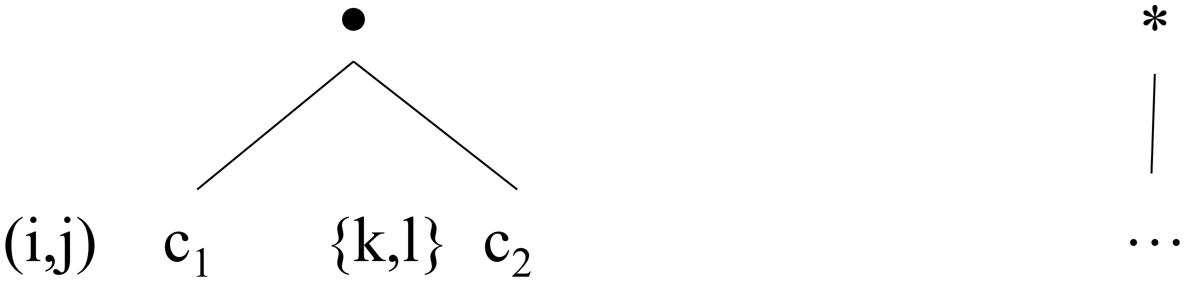


followpos(i)+=k,l

followpos(j)+=k,l

Regexp to DFA: *followpos*

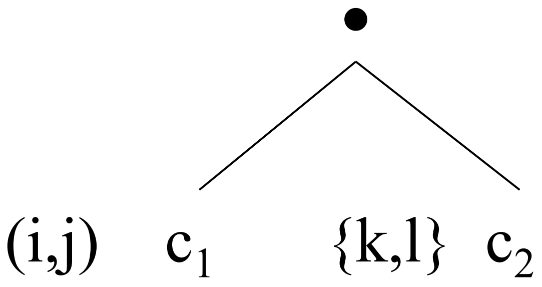
- *followpos(p):* tells us which positions can follow a position *p*
- There are two rules that use the *firstpos* {} and *lastpos* () information



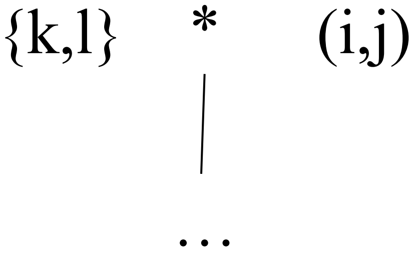
followpos(i)+=k,l
followpos(j)+=k,l

Regexp to DFA: *followpos*

- *followpos(p):* tells us which positions can follow a position *p*
- There are two rules that use the *firstpos* {} and *lastpos* () information

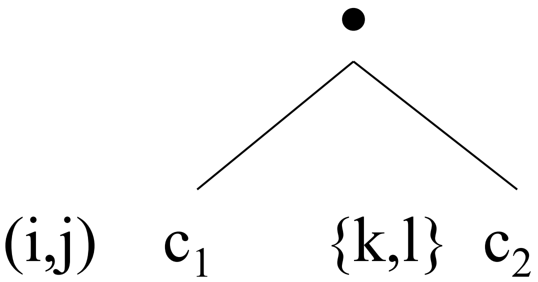


followpos(i)+=k,l
followpos(j)+=k,l



Regexp to DFA: *followpos*

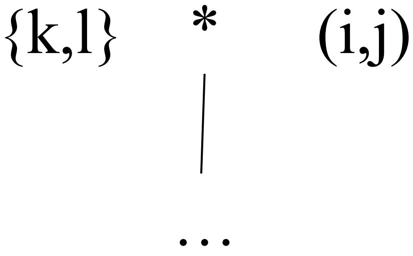
- *followpos(p):* tells us which positions can follow a position *p*
- There are two rules that use the *firstpos* {} and *lastpos* () information



•

(i,j) c_1 {k,l} c_2

followpos(i)+=k,l
followpos(j)+=k,l

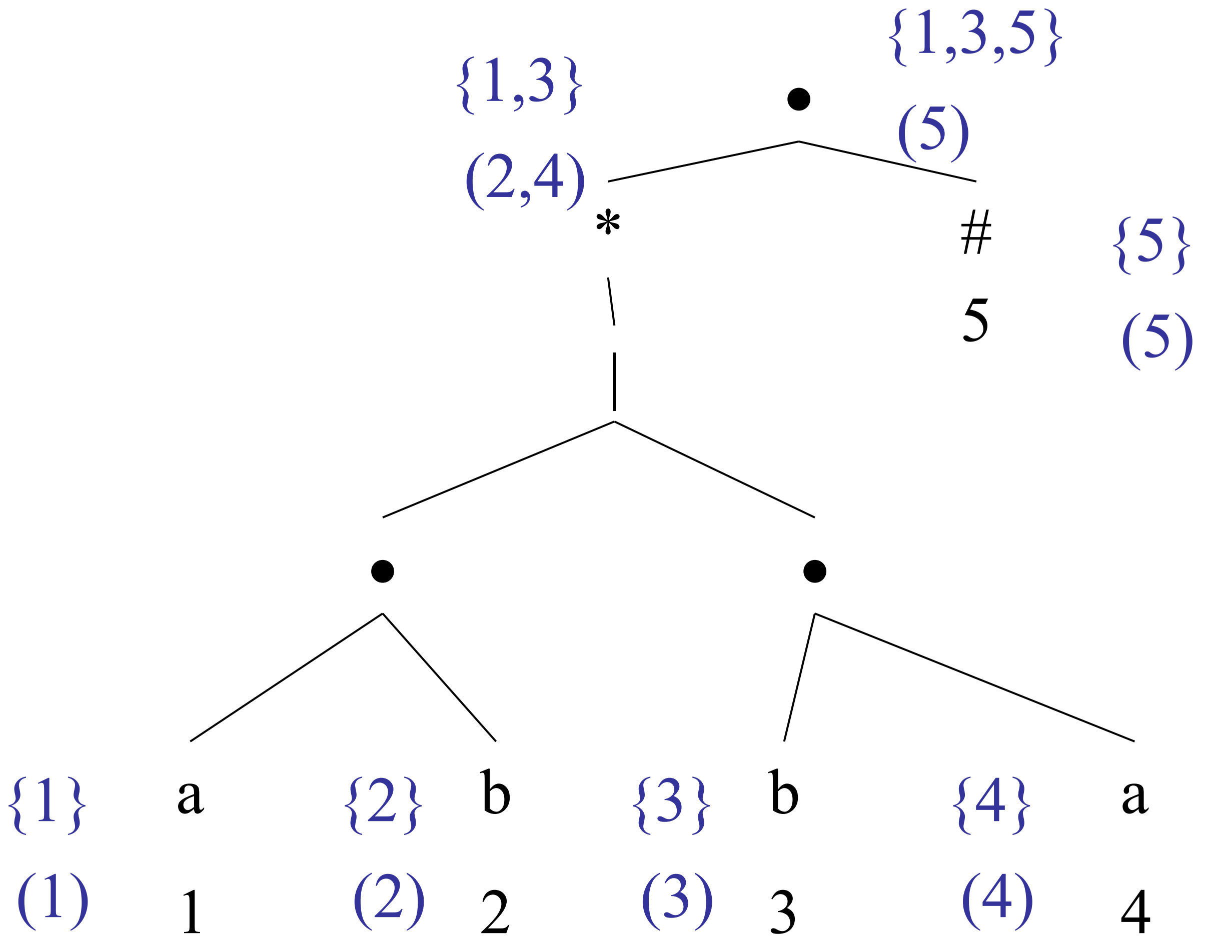


{k,l} * (i,j)

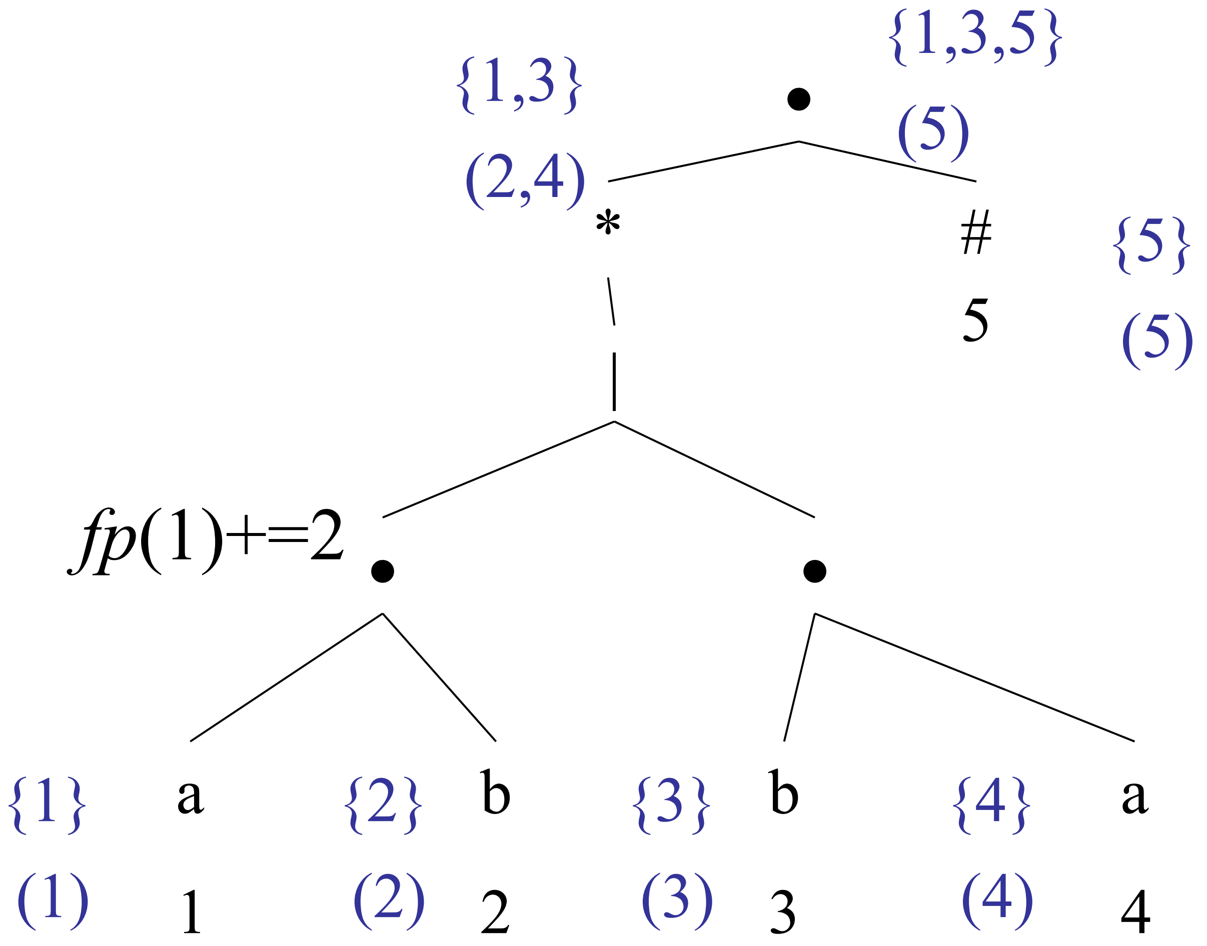
…

followpos(i)+=k,l
followpos(j)+=k,l

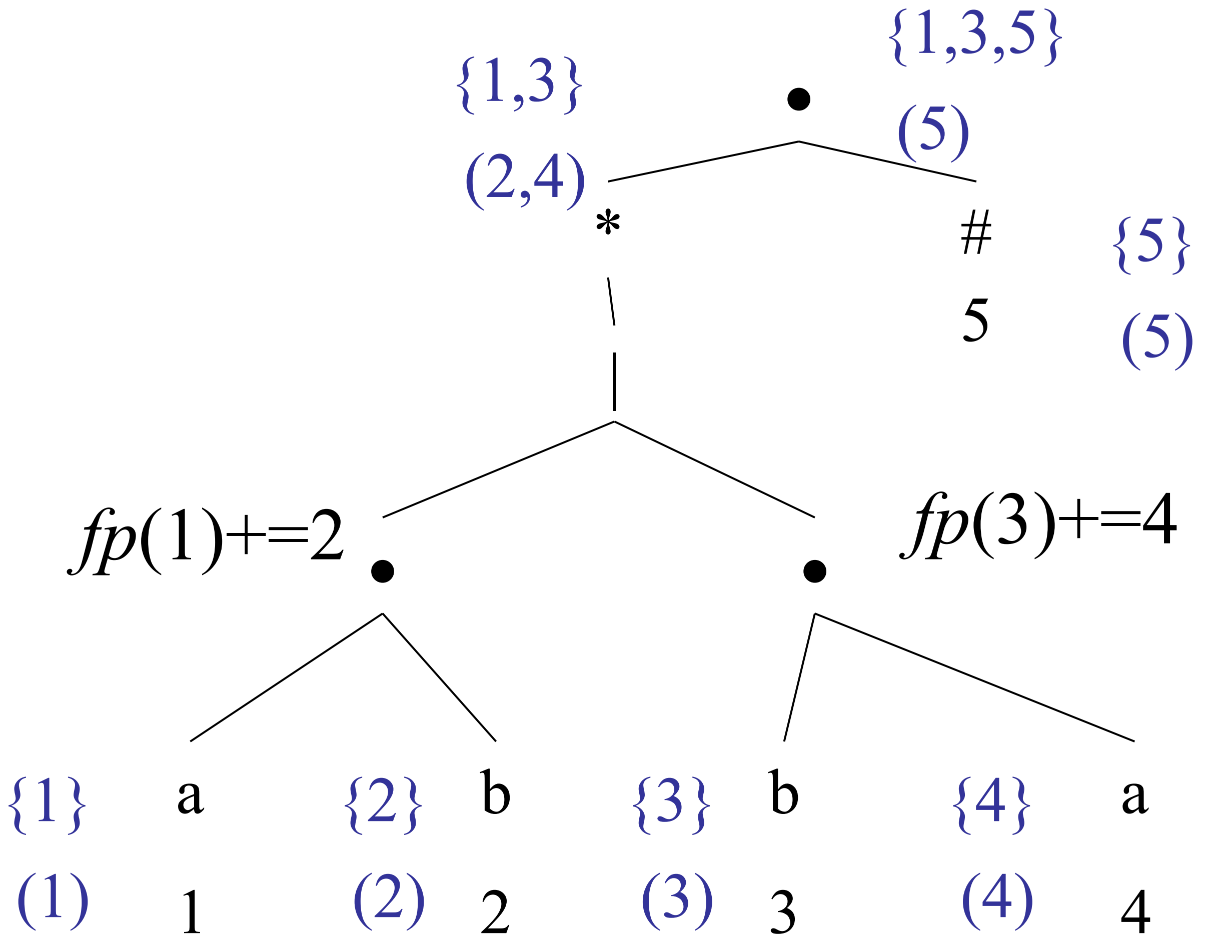
Regexp to DFA: ((ab)|(ba))*#



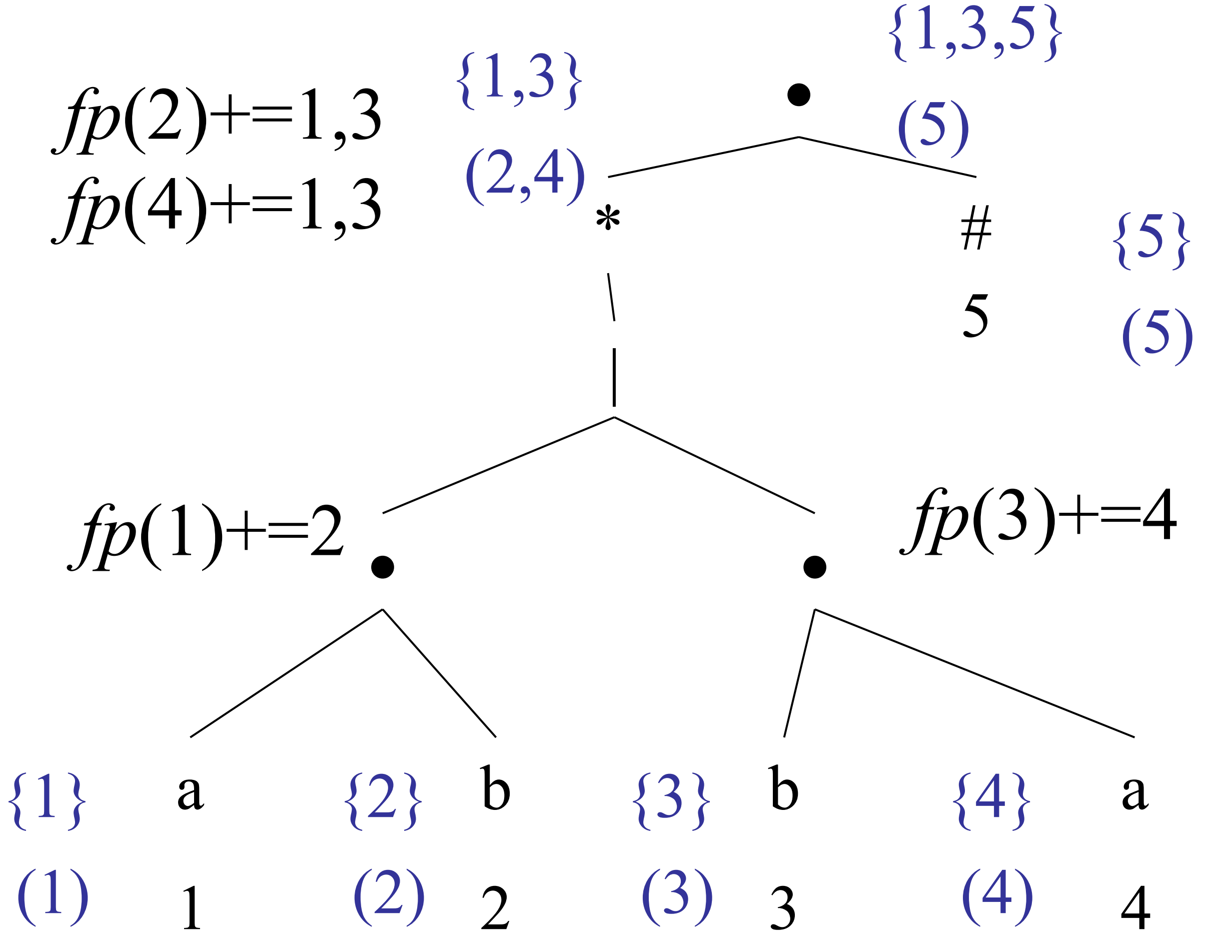
Regexp to DFA: ((ab)|(ba))*#



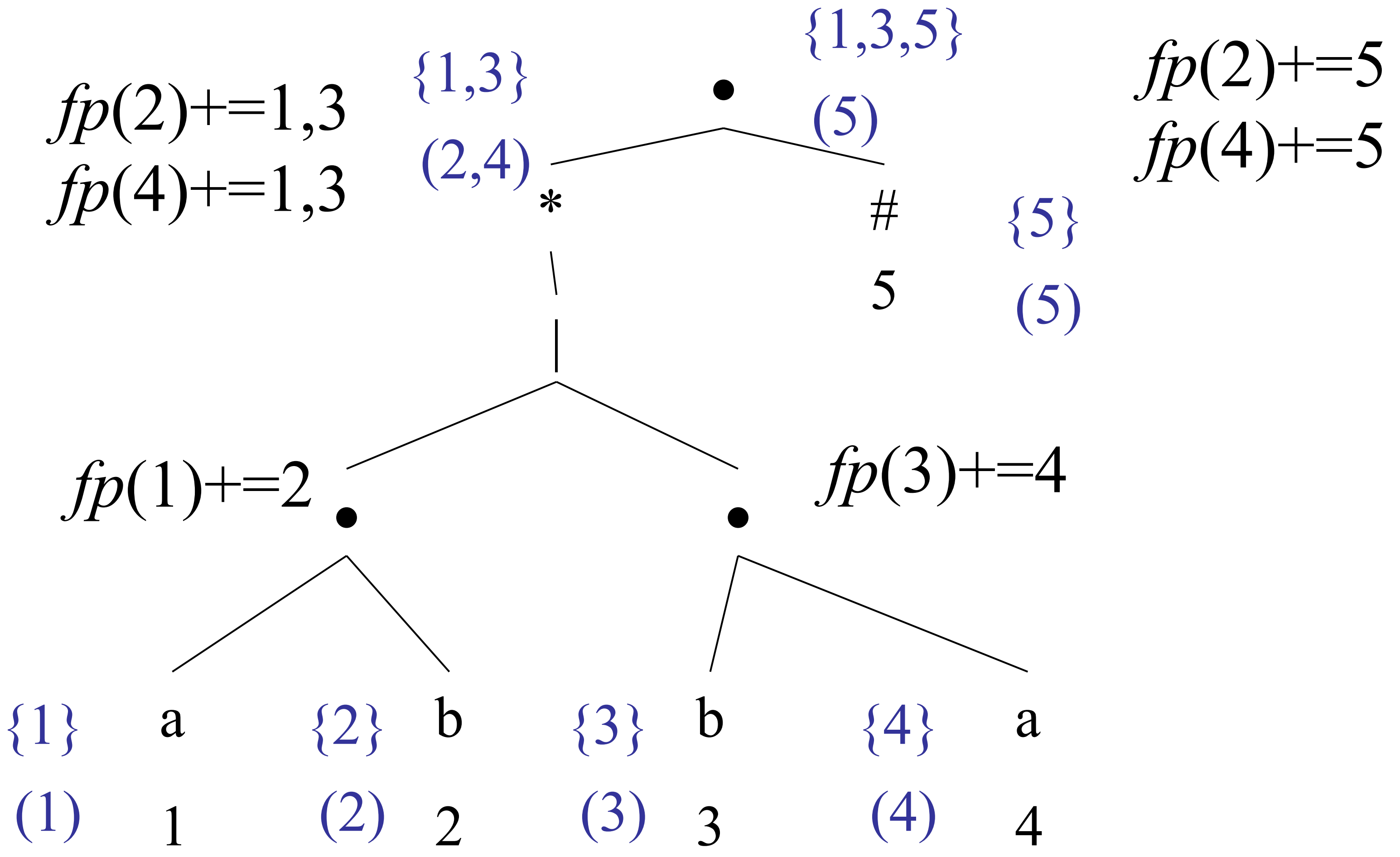
Regexp to DFA: ((ab)|(ba))*#



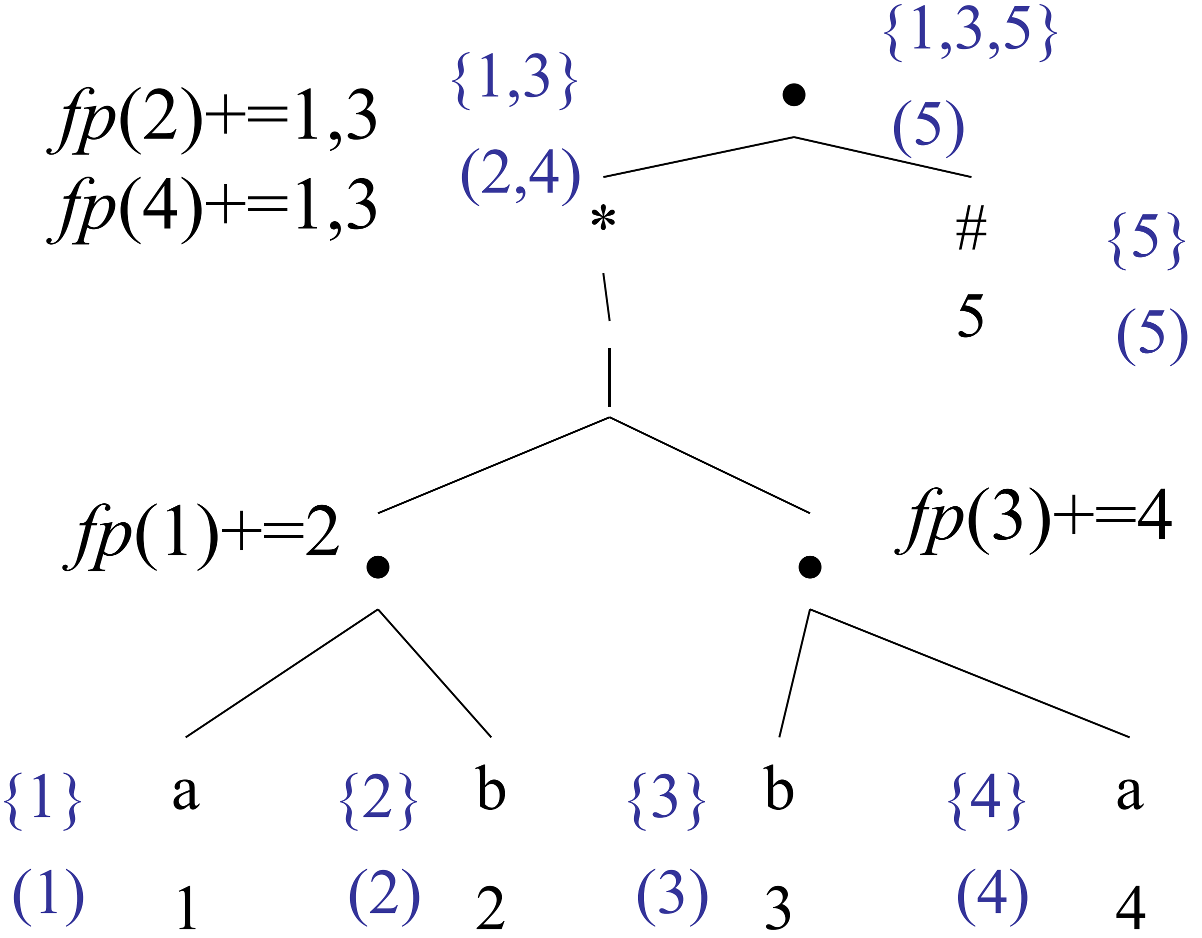
Regexp to DFA: ((ab)|(ba))*#



Regexp to DFA: ((ab)|(ba))*#



Regexp to DFA: ((ab)|(ba))*#



fp(2)+=1,3
fp(4)+=1,3

fp(2)+=5
fp(4)+=5

fp(1)+=2

fp(3)+=4

root={1,3,5}
fp(1)=2
fp(3)=4
fp(2)=1,3,5
fp(4)=1,3,5

Regexp to DFA: ((ab)|(ba))*#

$root = \{1,3,5\}$

$fp(1)=2$

$fp(3)=4$

$fp(2)=1,3,5$

$fp(4)=1,3,5$

1:a

2:b

3:b

4:a

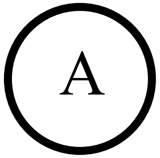
5:a

Regexp to DFA: ((ab)|(ba))*#

root={1,3,5}
fp(1)=2
fp(3)=4
fp(2)=1,3,5
fp(4)=1,3,5

{1,3,5} A

1:a
2:b
3:b
4:a
5:a



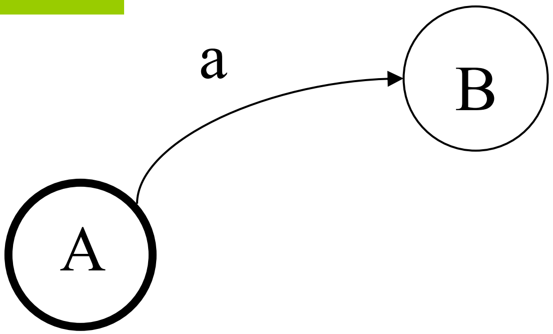
Regexp to DFA: ((ab)|(ba))*#

root={1,3,5}
fp(1)=2
fp(3)=4
fp(2)=1,3,5
fp(4)=1,3,5

{1,3,5} A

A: *fp*(1),a {2},a B,a

1:a
2:b
3:b
4:a
5:a



Regexp to DFA: ((ab)|(ba))*#

root={1,3,5}

fp(1)=2

fp(3)=4

fp(2)=1,3,5

fp(4)=1,3,5

{1,3,5} A

A: fp(1),a {2},a B,a

A: fp(3),b {4},b C,b

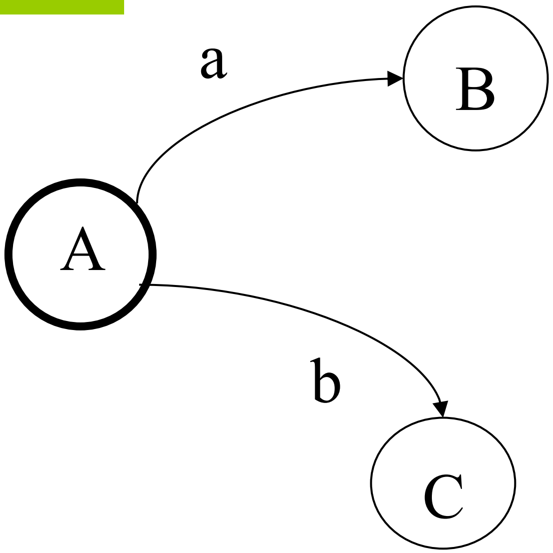
1:a

2:b

3:b

4:a

5:a



Regexp to DFA: ((ab)|(ba))*#

root={1,3,5}
fp(1)=2
fp(3)=4
fp(2)=1,3,5
fp(4)=1,3,5

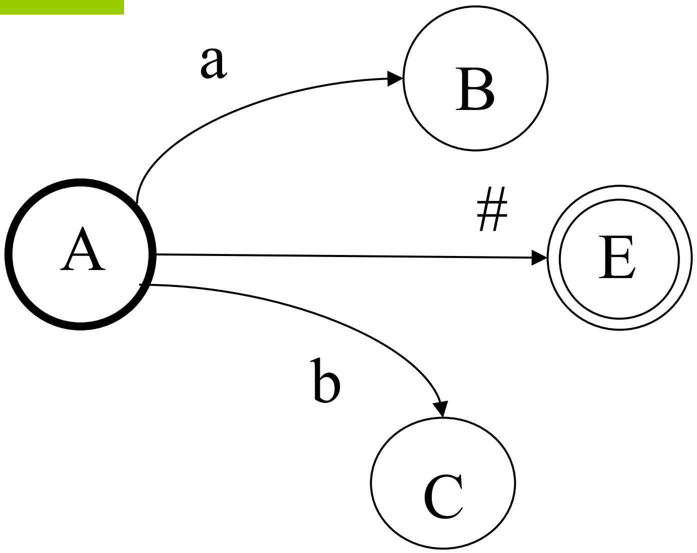
{1,3,5} A

A: fp(1),a {2},a B,a

A: fp(3),b {4},b C,b

A: fp(5),# {},# E,#

1:a
2:b
3:b
4:a
5:a



Regexp to DFA: ((ab)|(ba))*#

root={1,3,5}
fp(1)=2
fp(3)=4
fp(2)=1,3,5
fp(4)=1,3,5

{1,3,5} A

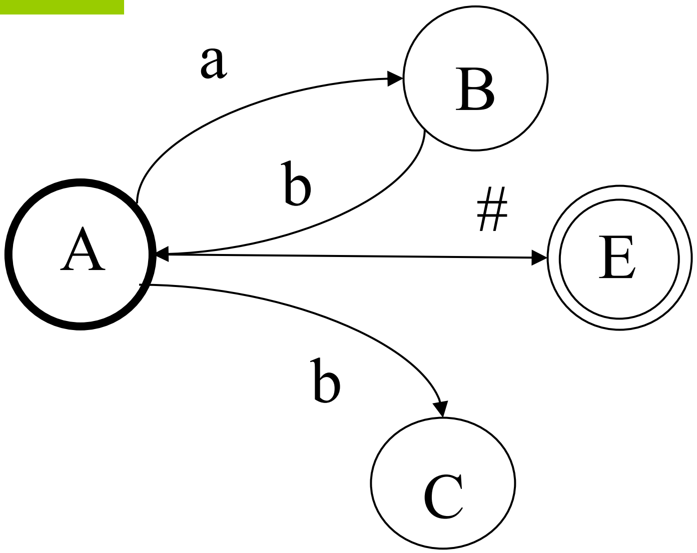
A: *fp*(1),a {2},a B,a

A: *fp*(3),b {4},b C,b

A: *fp*(5),# {},# E,#

B: *fp*(2),b {1,3,5},b A,b

1	a
2	b
3	b
4	a
5	a



Regexp to DFA: ((ab)|(ba))*#

root={1,3,5}

fp(1)=2

fp(3)=4

fp(2)=1,3,5

fp(4)=1,3,5

{1,3,5} A

A: *fp*(1),a {2},a B,a

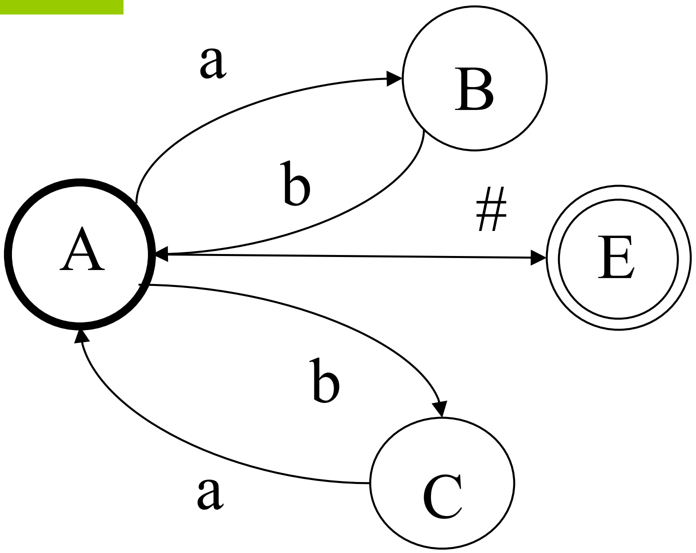
A: *fp*(3),b {4},b C,b

A: *fp*(5),# {},# E,#

B: *fp*(2),b {1,3,5},b A,b

C: *fp*(4),a {1,3,5},a A,a

1	a
2	b
3	b
4	a
5	a



Regexp to DFA: ((ab)|(ba))*#

root={1,3,5}
fp(1)=2
fp(3)=4
fp(2)=1,3,5
fp(4)=1,3,5

1:a
2:b
3:b
4:a
5:a

{1,3,5} A

A: fp(1),a {2},a B,a

A: fp(3),b {4},b C,b

A: fp(5),# {},# E,#

B: fp(2),b {1,3,5},b A,b

C: fp(4),a {1,3,5},a A,a

Any state with a transition on # will be marked as final state