CS4317/9317 Tutorial 10

In start_element

- What is same:

Do child transition from the parent node. (black arrow)

- What is different:

Do sibling transition from the last preceding-sibling.(red arrow)

Union the two state sets, one from child transition, the other from sibling transition.

CS4317/9317 Tutorial 10

In end_element

Query: //a/b/following-sibling::c

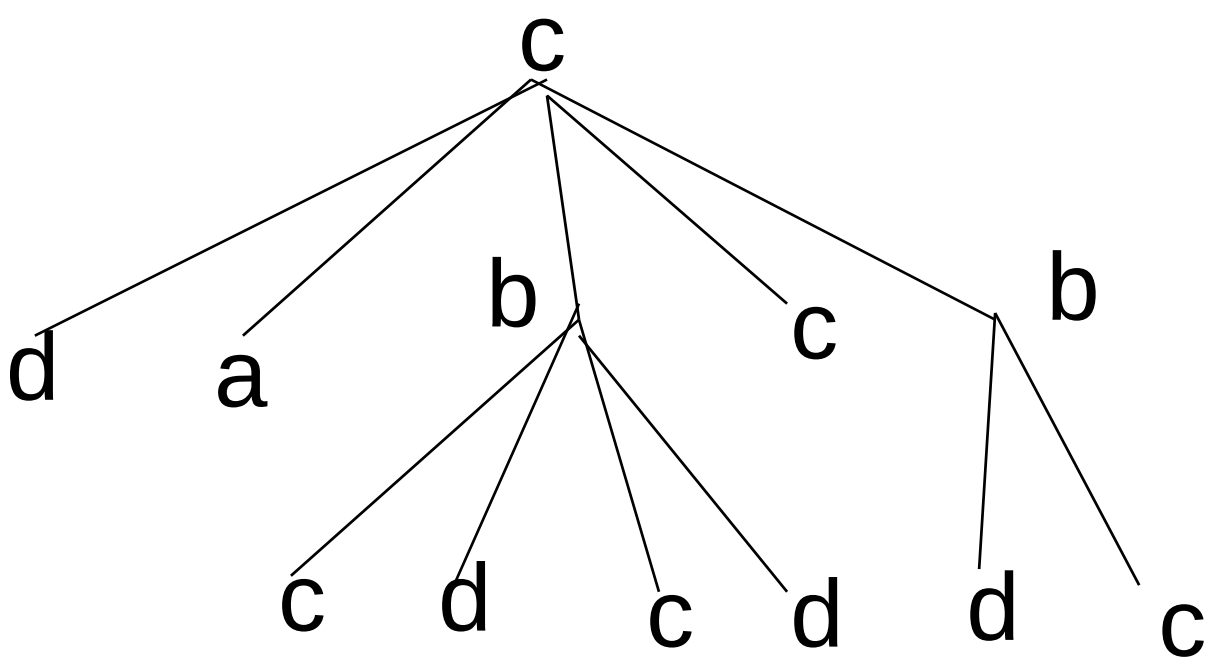
<a>

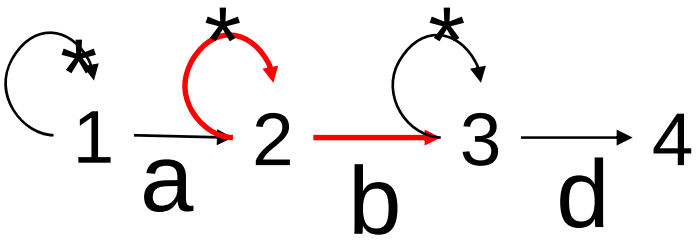
  we cannot pop the state set of here.

<c> we need to use the state set of to do sibling transition here.

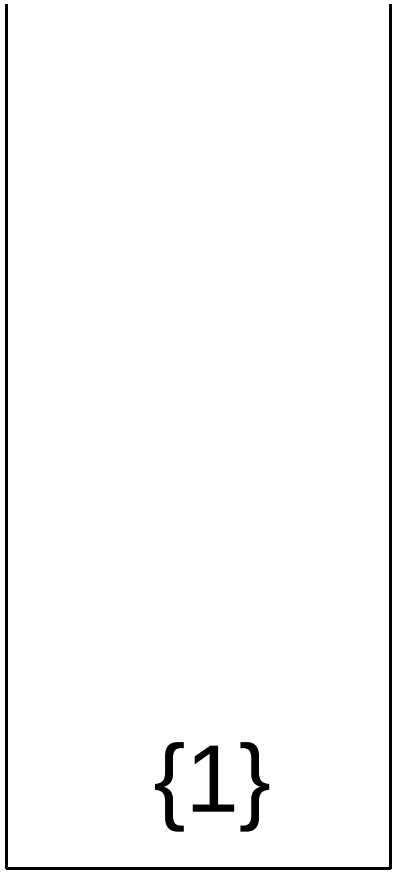
</c>we cannot pop the state set of <c> here.

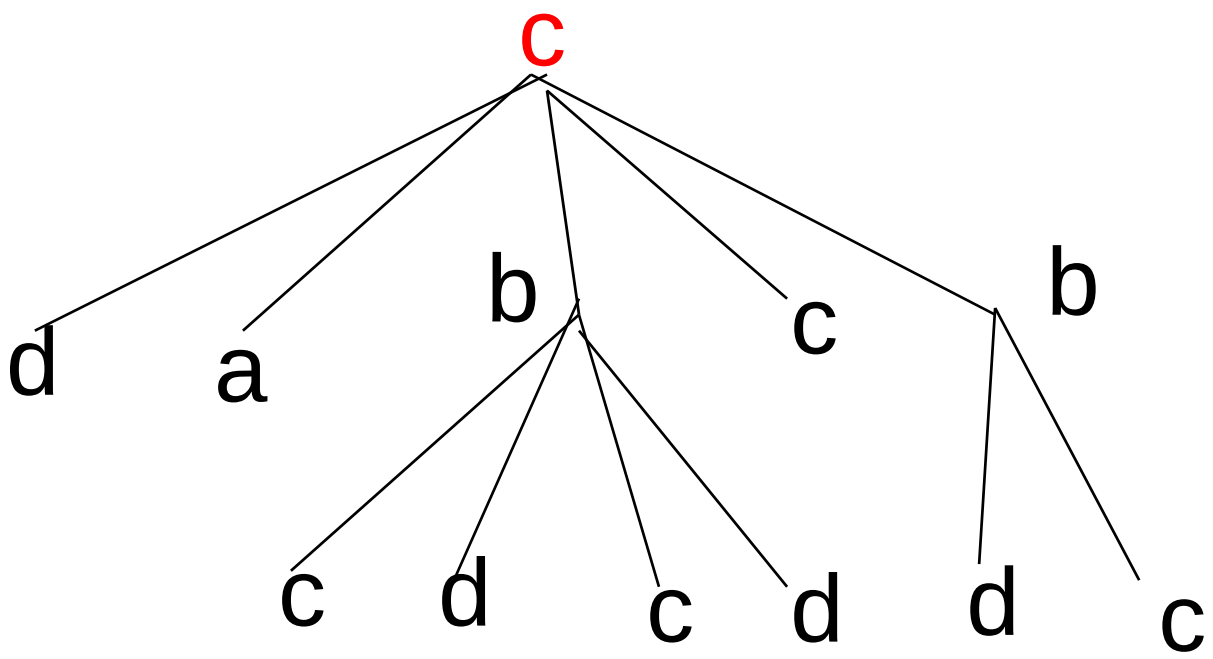
 ok, we can pop the state set of <c> here. All the tags appears after here are not <c>’s sibling.

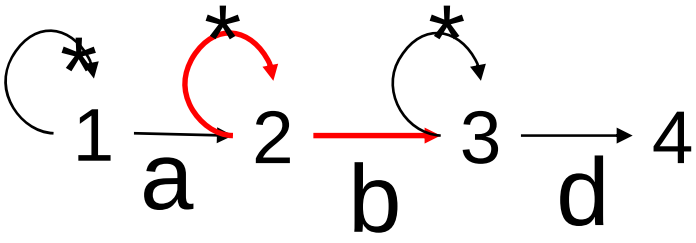




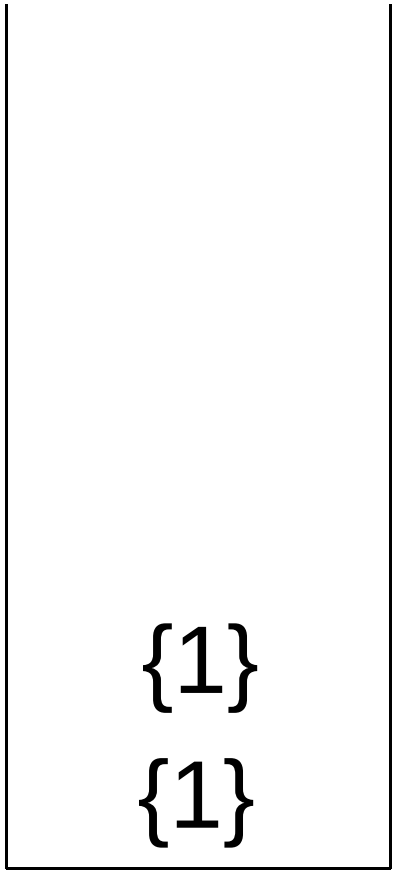
//a/following-sibling::b//d







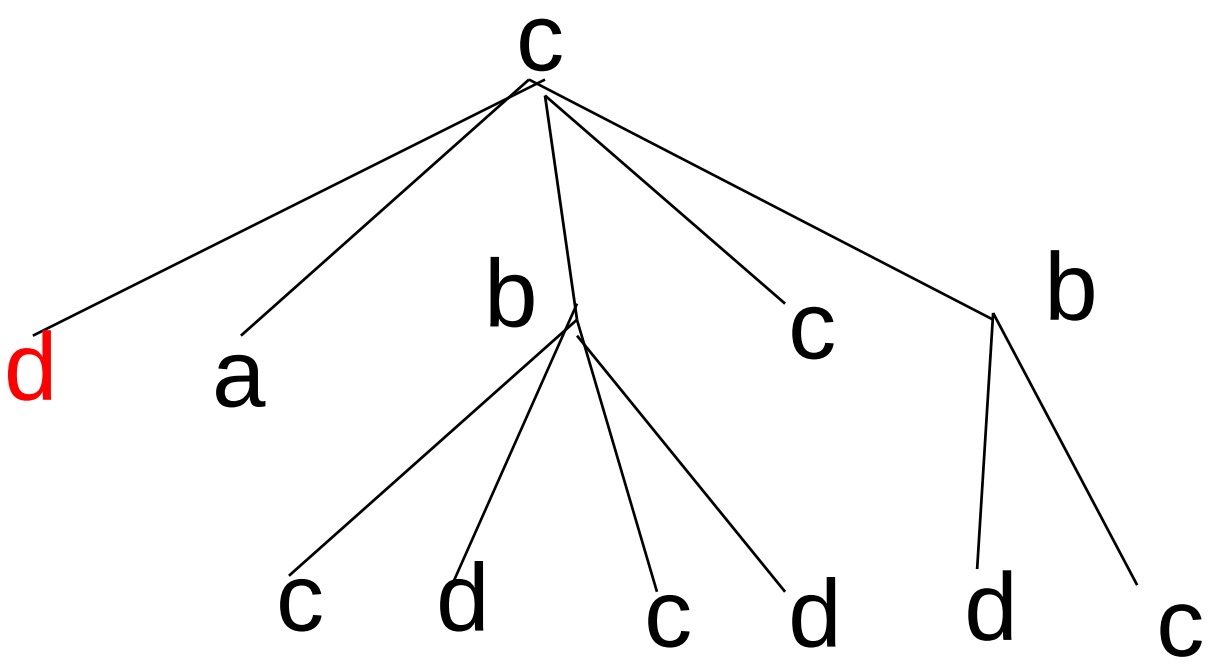
//a/following-sibling::b//d

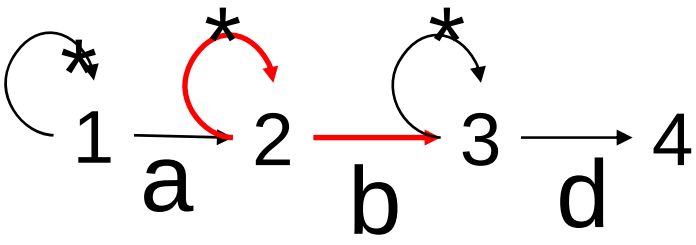


<c>

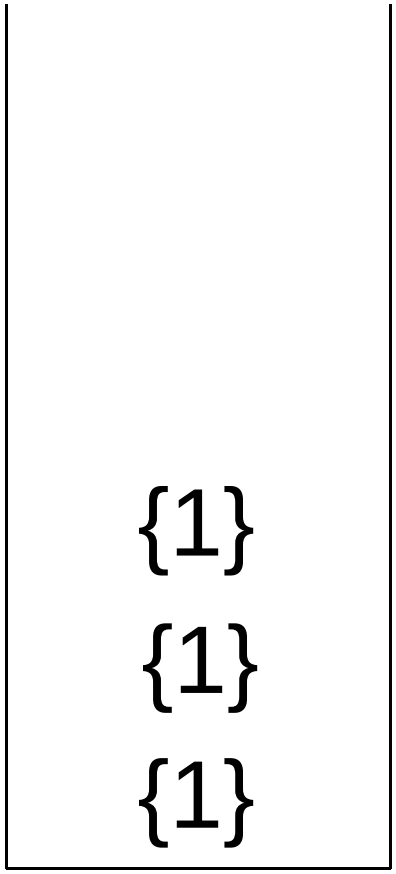
Child: 1->1

Sibling: none





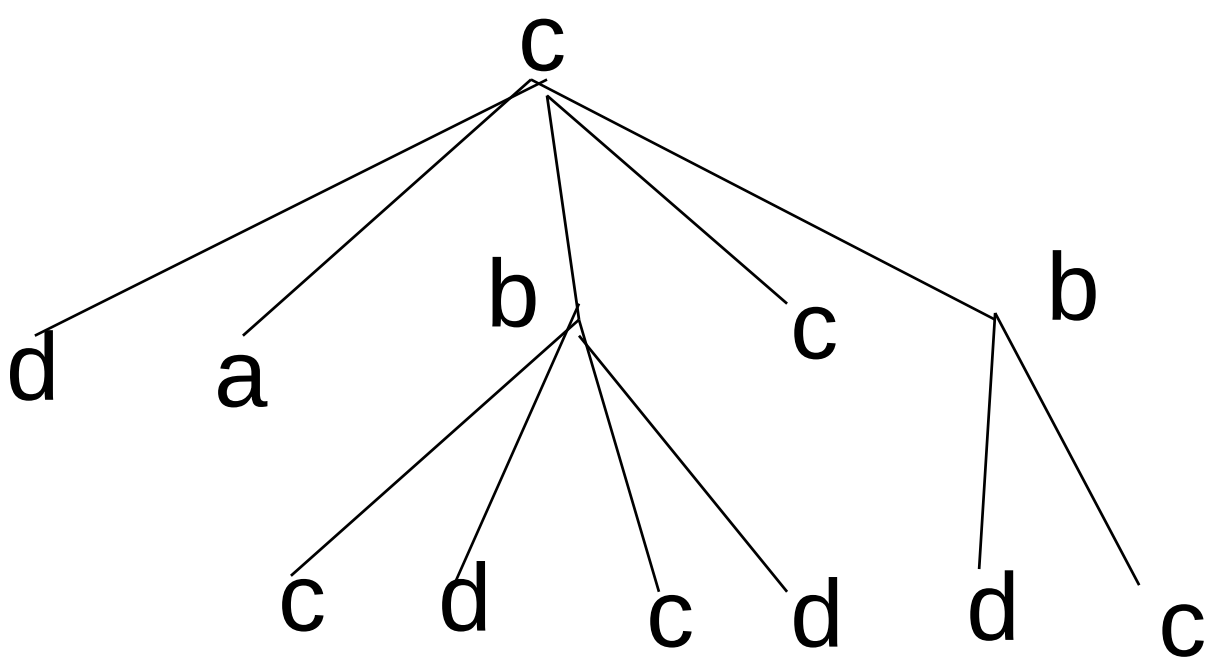
//a/following-sibling::b//d

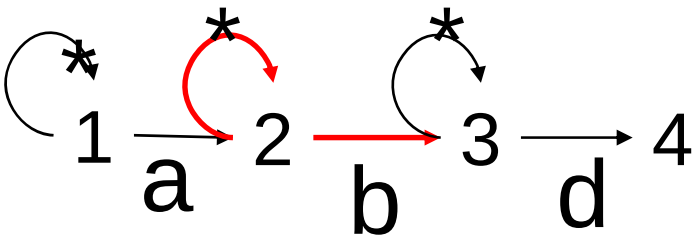


<d>

Child: 1->1

Sibling: none





//a/following-sibling::b//d

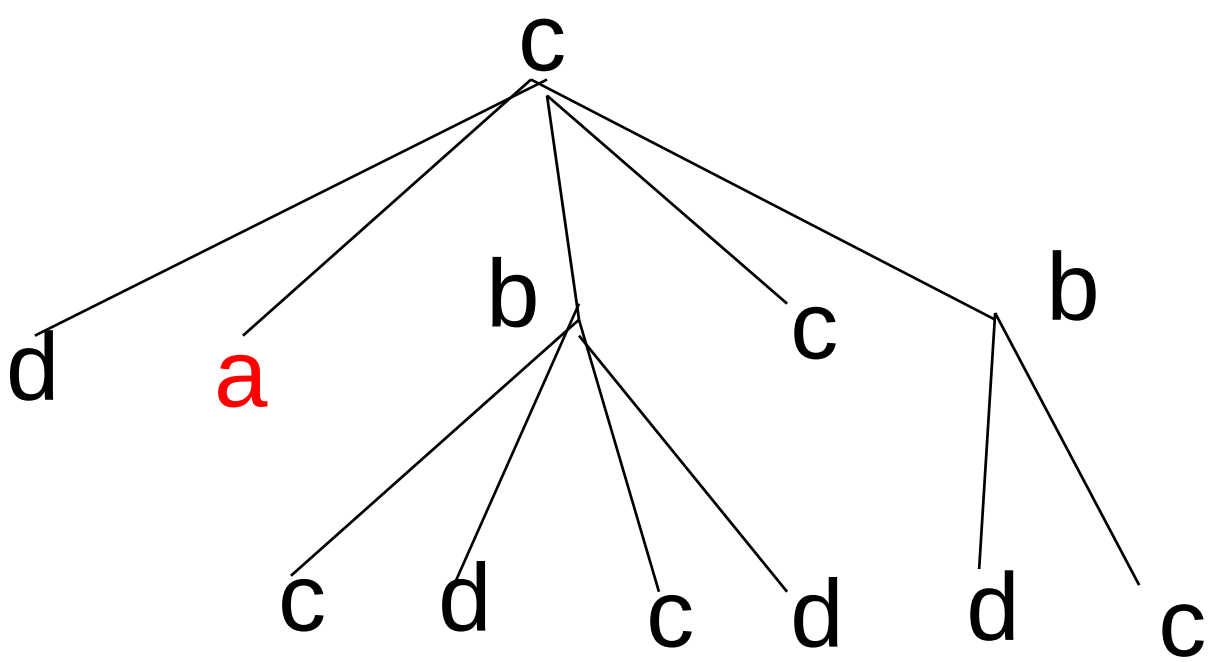
</d>

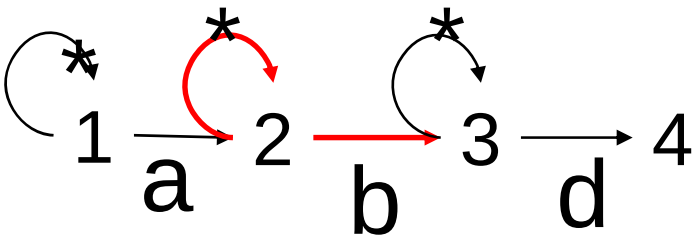
Do nothing

{1}

{1}

{1}

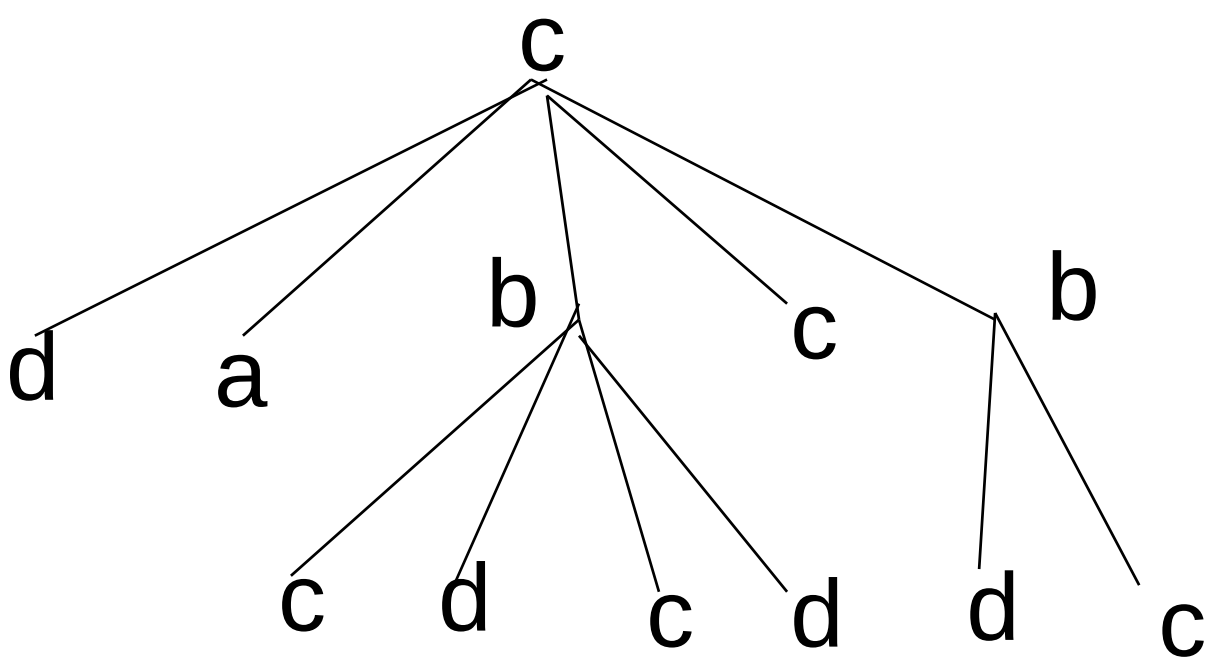


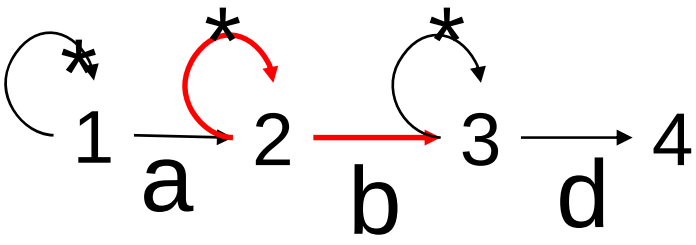


//a/following-sibling::b//d

{1,2}
{1}
{1}

<a>
Child: 1->{1,2}
Sibling: 1->{}





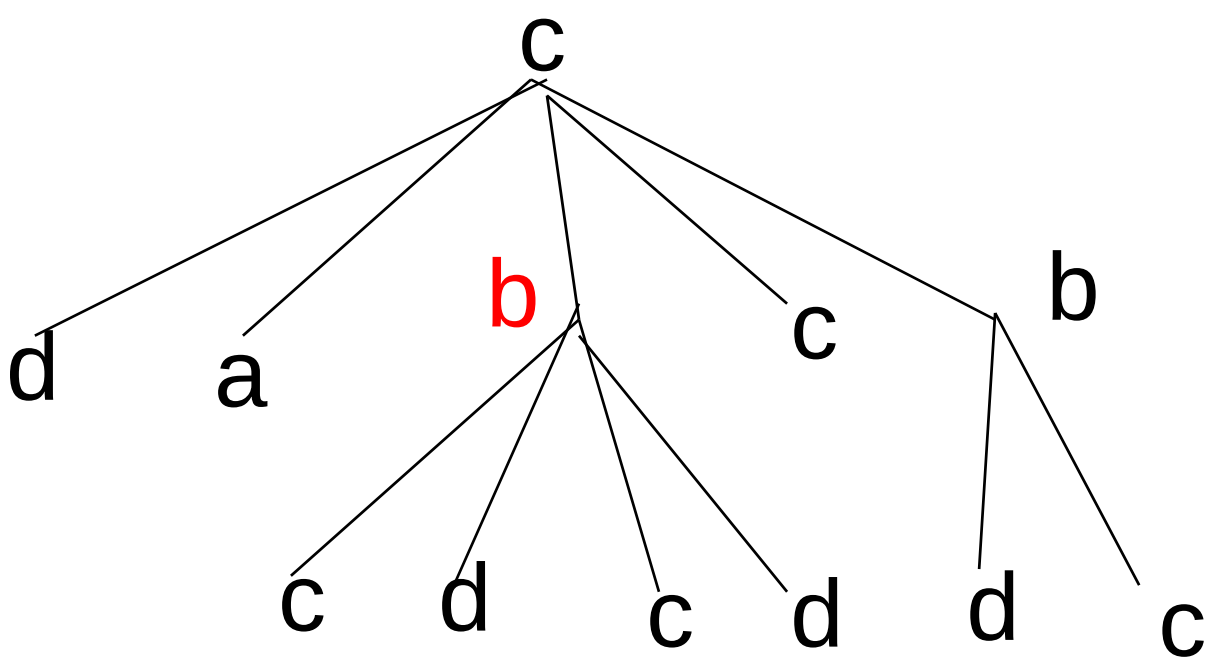
//a/following-sibling::b//d

Do nothing

{1,2}

{1}

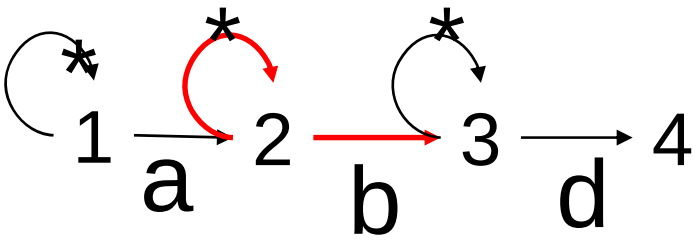
{1}



{1,2,3}

{1}

{1}



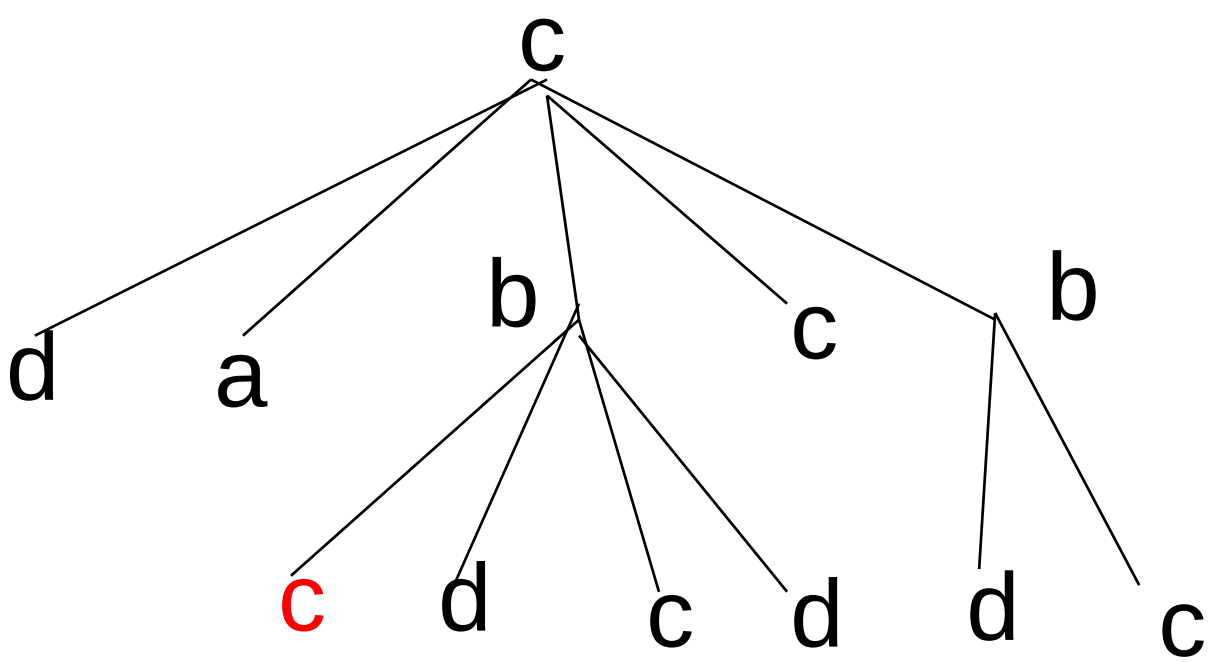
//a/following-sibling::b//d

<b>

Child: 1->{1}

Sibling: 1->{}

2->{2,3}

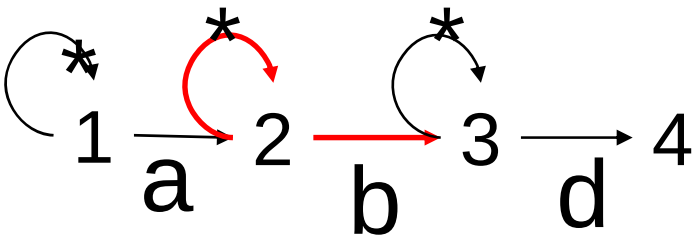


{1, 3}

{1,2,3}

{1}

{1}



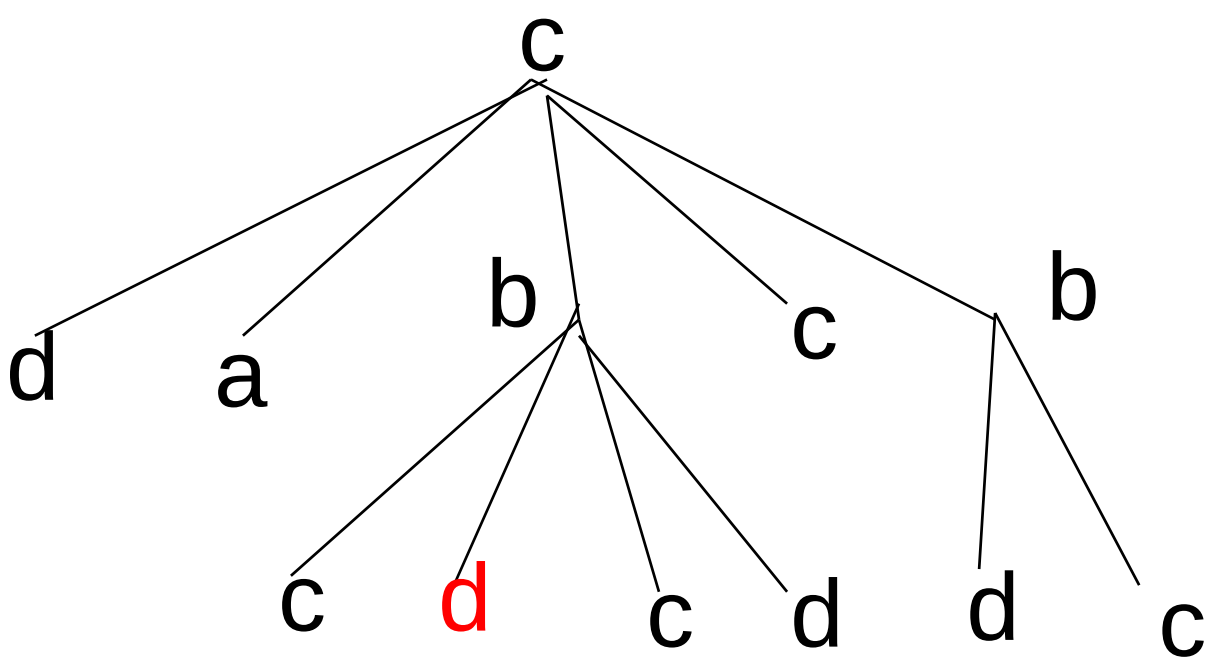
//a/following-sibling::b//d

<c>

Child: 1->{1} 2->{} 3->{3

Sibling:none

</c> do nothing

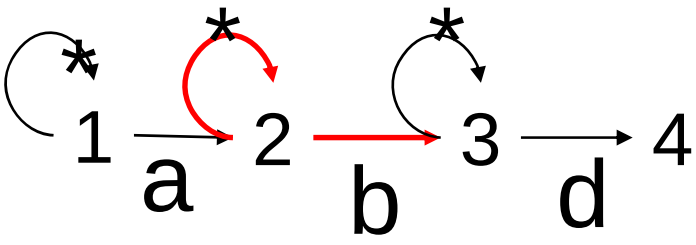


{1, 3,4}

{1,2,3}

{1}

{1}



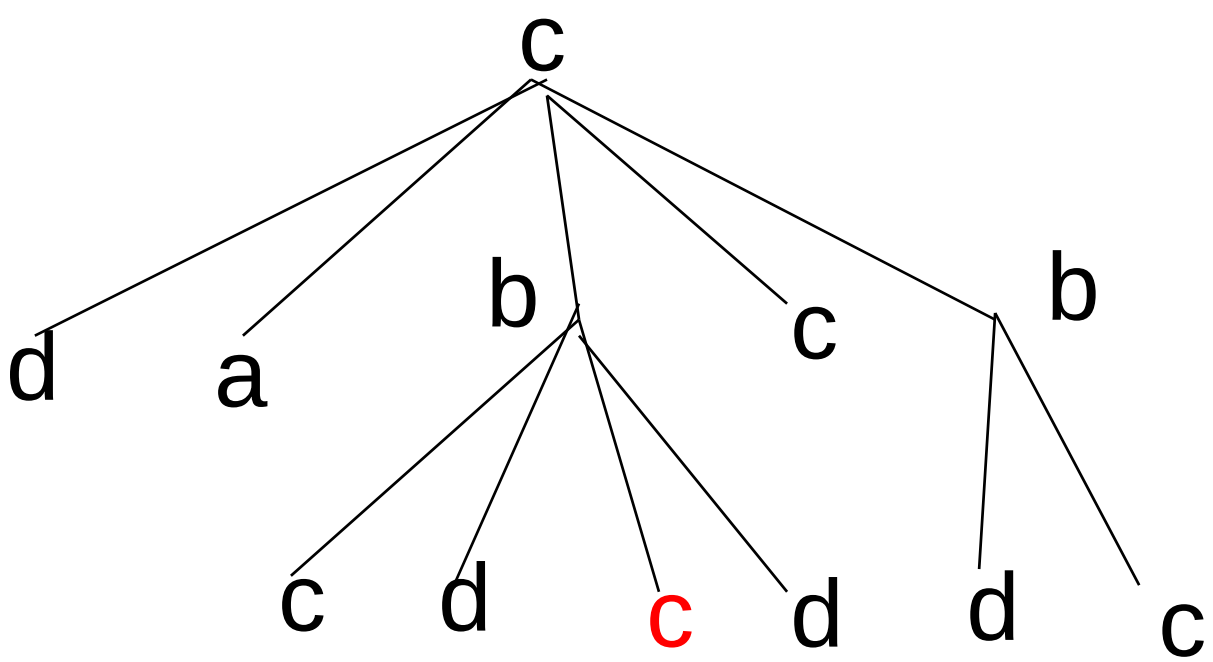
//a/following-sibling::b//d

<d>

Child: 1->{1} 2->{} 3->{3,4}

Sibling:1,3->{}

</d> do nothing

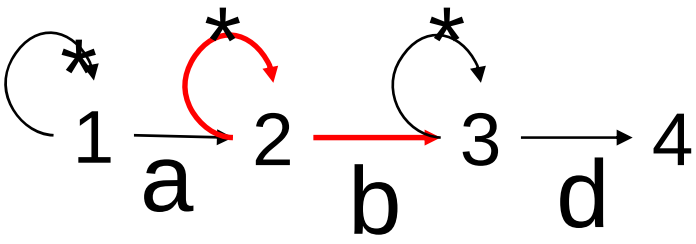


{1, 3}

{1,2,3}

{1}

{1}



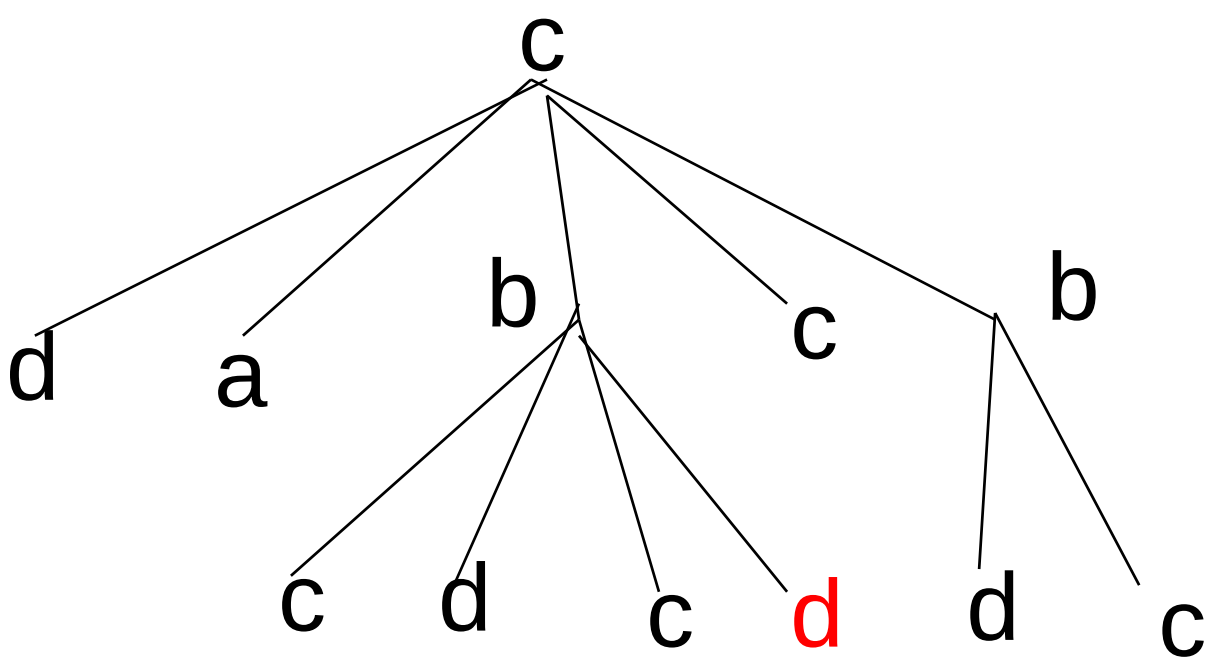
//a/following-sibling::b//d

<c>

Child: 1->{1} 2->{} 3->{3}

Sibling:1,3,4->{}

</c> do nothing

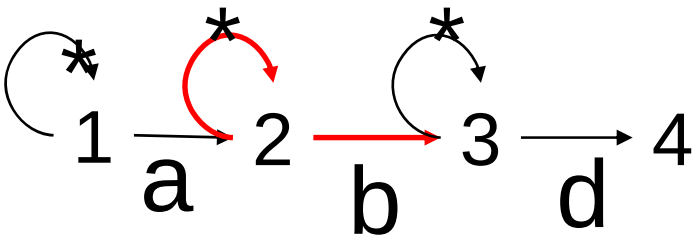


{1, 3,4}

{1,2,3}

{1}

{1}



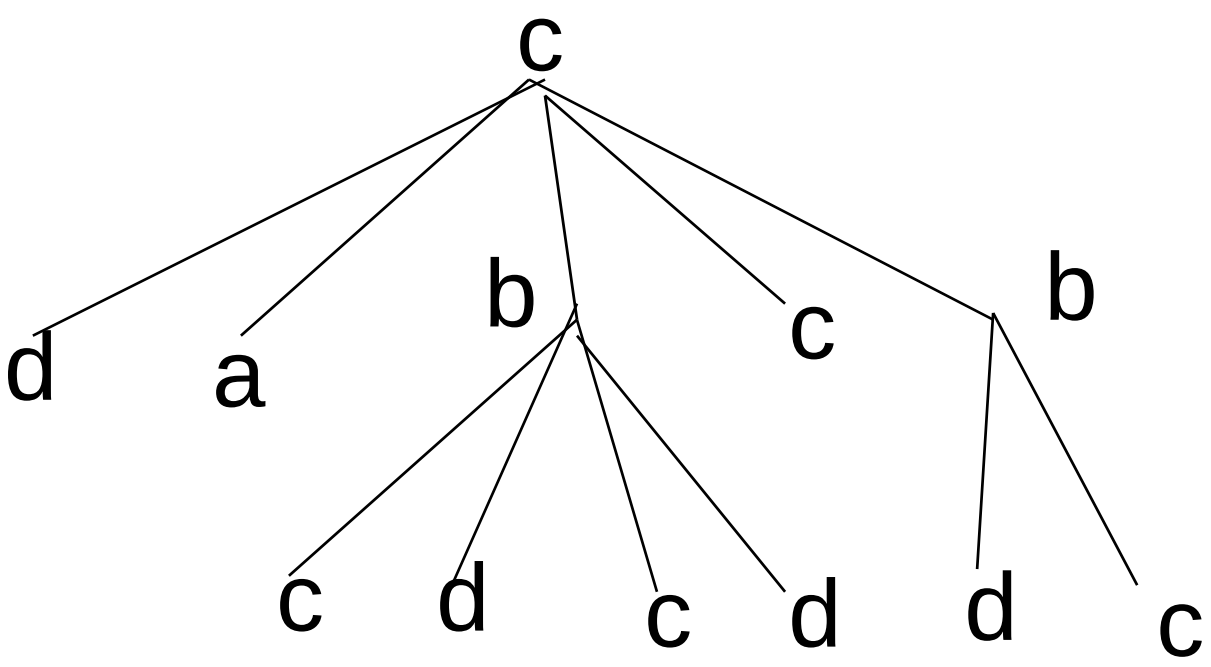
//a/following-sibling::b//d

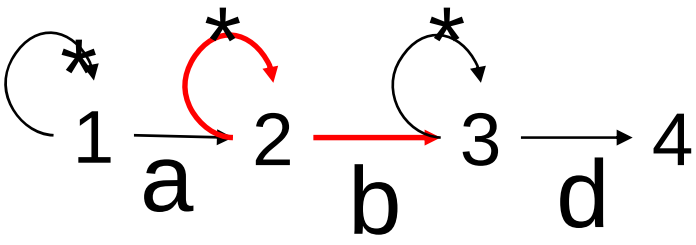
<d>

Child: 1->{1} 2->{} 3->{3,4}

Sibling:1,3->{}

</d> do nothing





//a/following-sibling::b//d

{1,2,3}

{1}

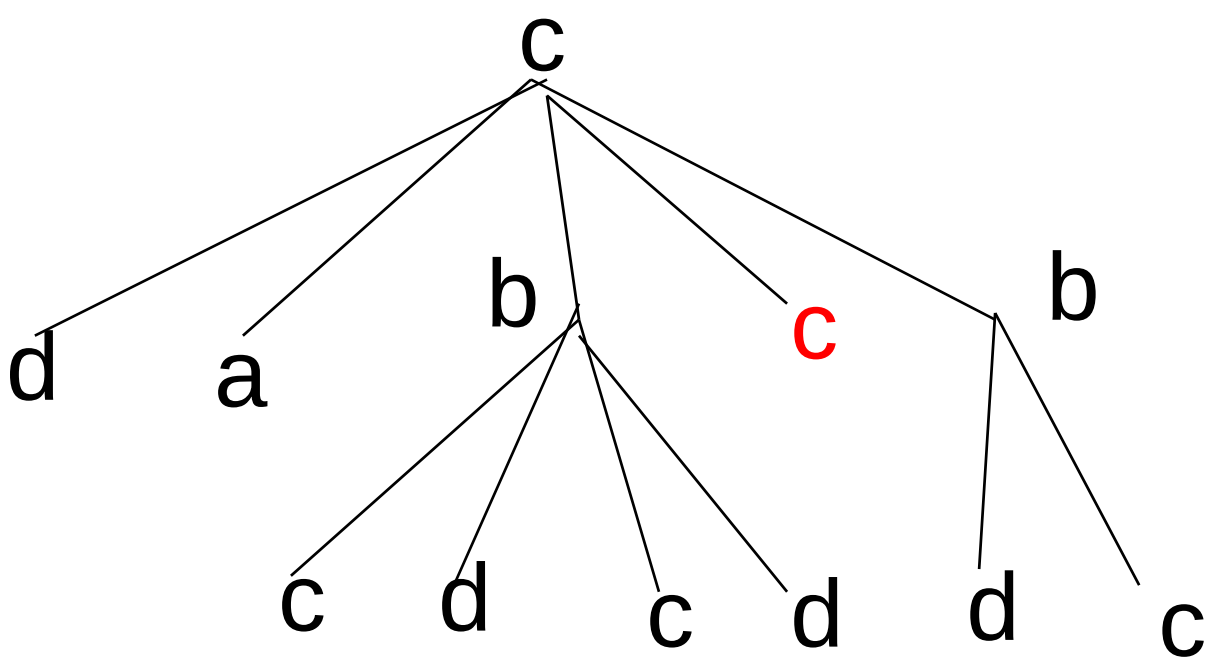
{1}

Pop stack

Why?

“b” has child node,

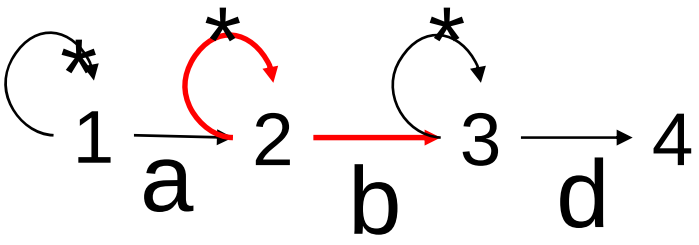
pop the child node’s StateSet



{1,2}

{1}

{1}



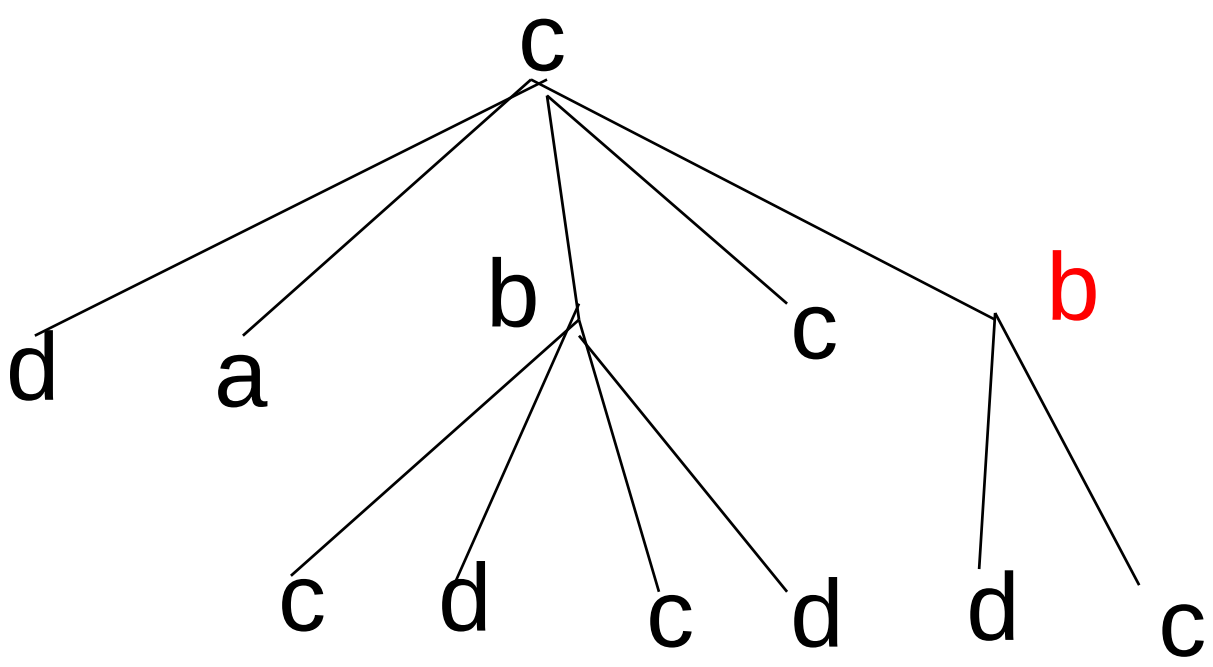
//a/following-sibling::b//d

<c>

Child: 1->{1}

Sibling:1,3->{} 2->{2}

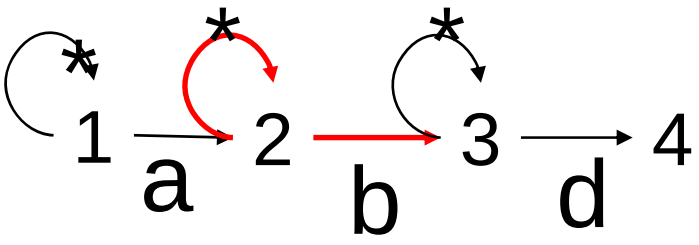
</c> do nothing



{1,2,3}

{1}

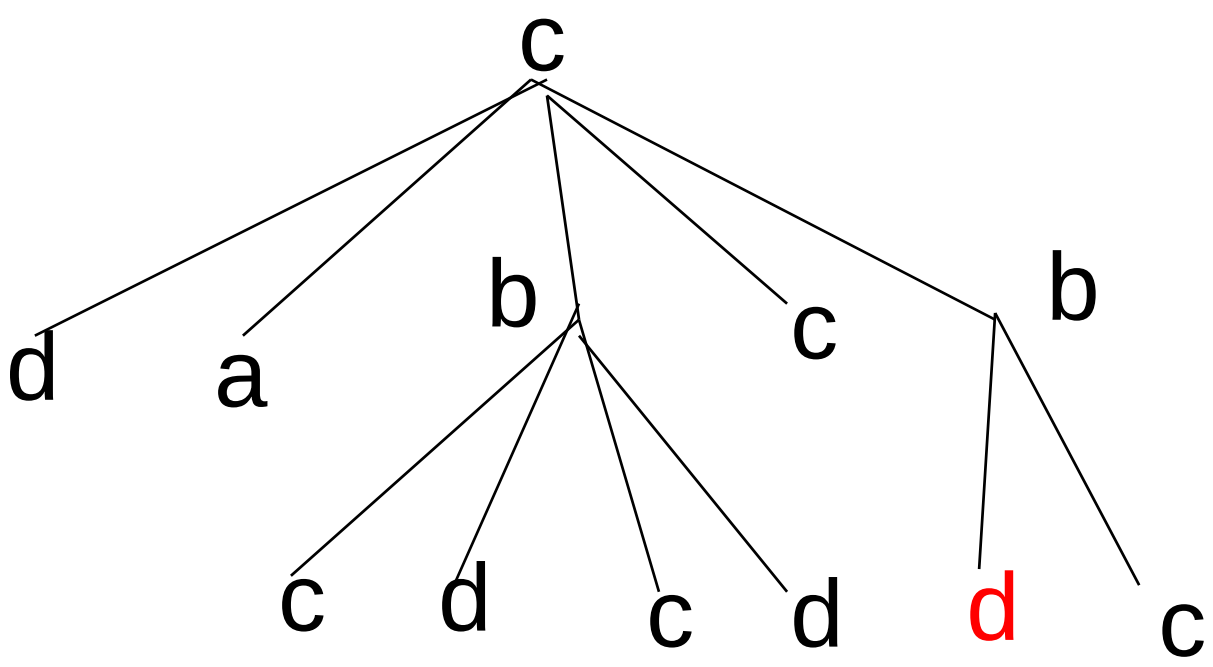
{1}



Child: 1->{1}

Sibling:1->{} 2->{2,3}

//a/following-sibling::b//d

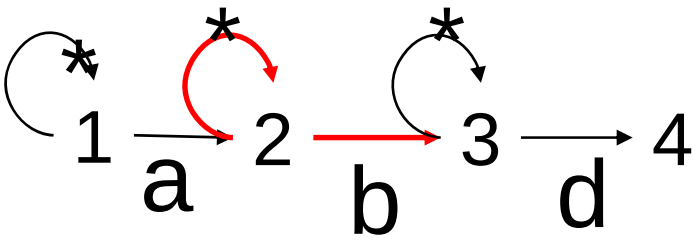


{1,3,4}

{1,2,3}

{1}

{1}



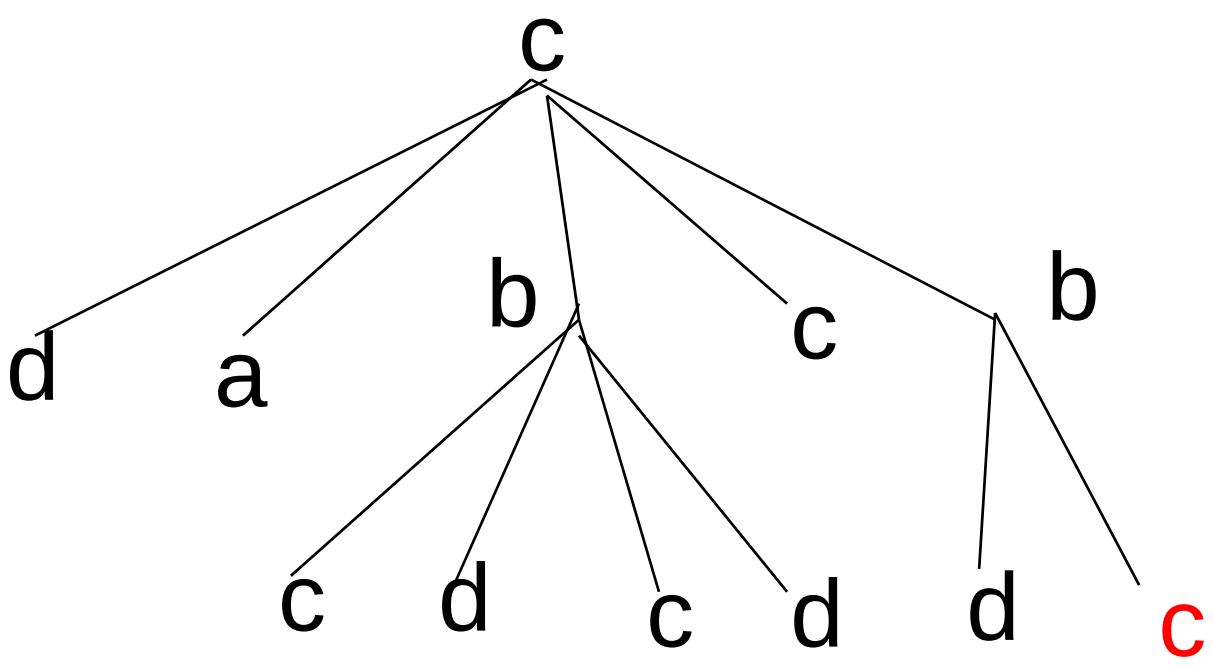
//a/following-sibling::b//d

<d>

Child: 1->{1} 2->{} 3->{3,4}

Sibling:none

</d> do nothing

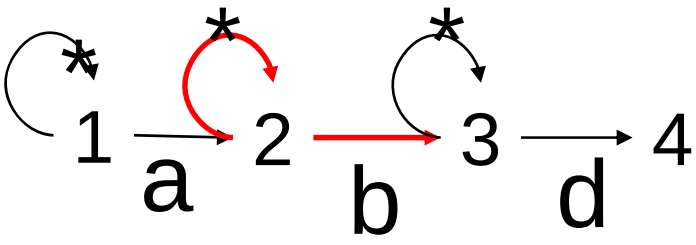


{1,3}

{1,2,3}

{1}

{1}



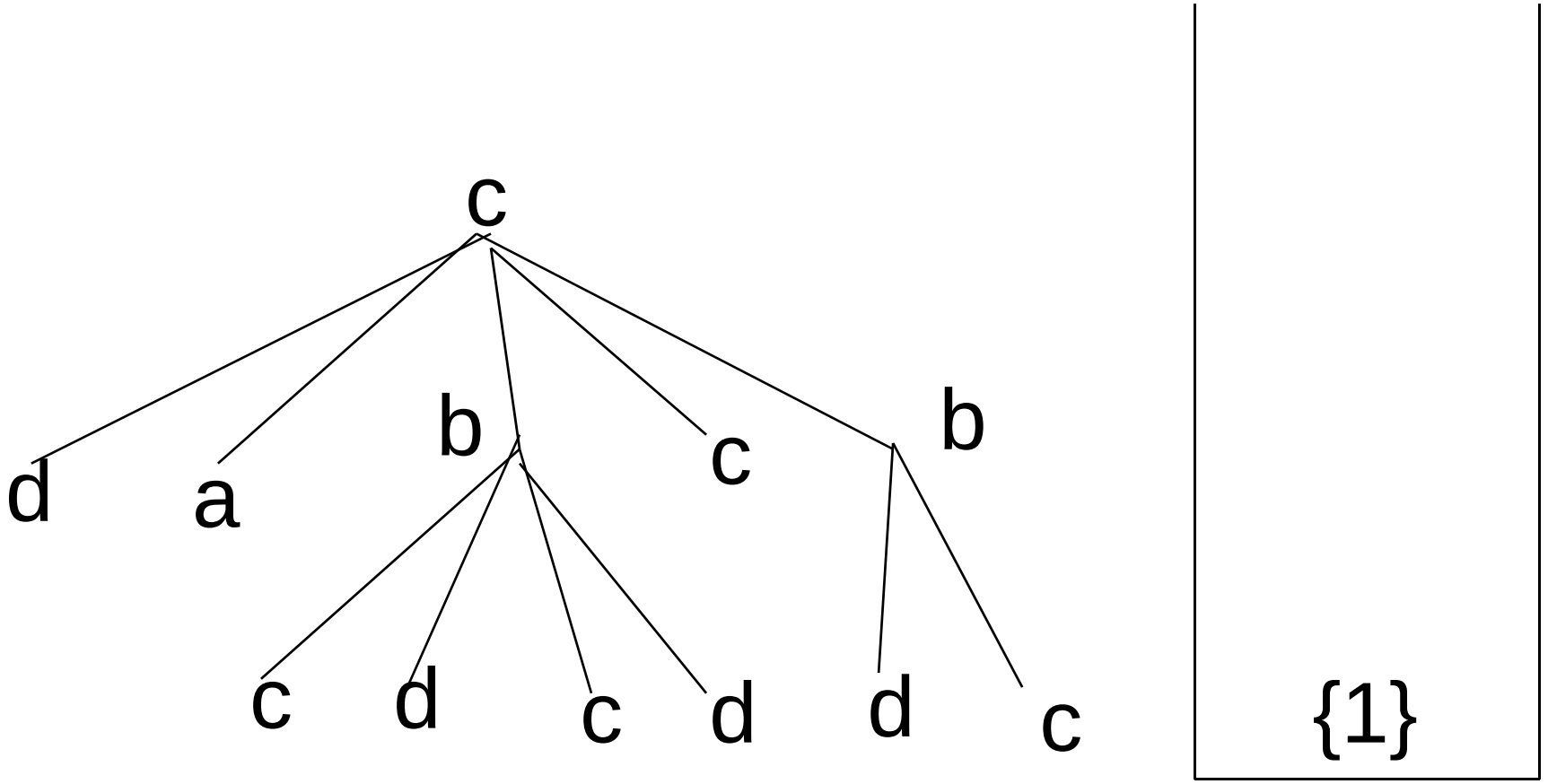
//a/following-sibling::b//d

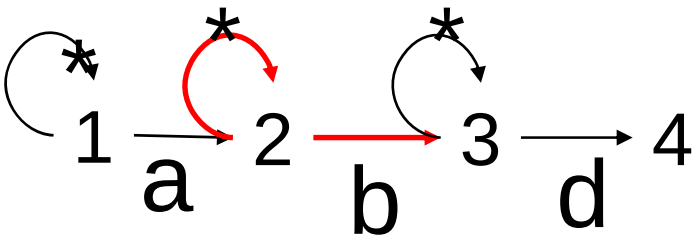
<c>

Child: 1->{1} 2->{} 3->{3}

Sibling:1,3,4->{}

</c> do nothing





//a/following-sibling::b//d

 Pop Stack (leaf node c's StateSet)
</c> Pop Stack (b's StateSet)
endDoc pop stack (root node c's StateSet)

Now StateSet is empty.
The {1} shown in the figure is the initial state.

Summary

- In start_element:

```
If(firstchild){
    States=childTrans(stack[top])
    stack=push(States)
}else{
    StatesA=childTrans(stack[top-1])
    StatesB=siblingTrans(stack[top])
     States=StatesA union StatesB
    stack.pop()
    stack.push(States)
}
```

- In end_element:

```
If(haschild){
    stack.pop()
}
```