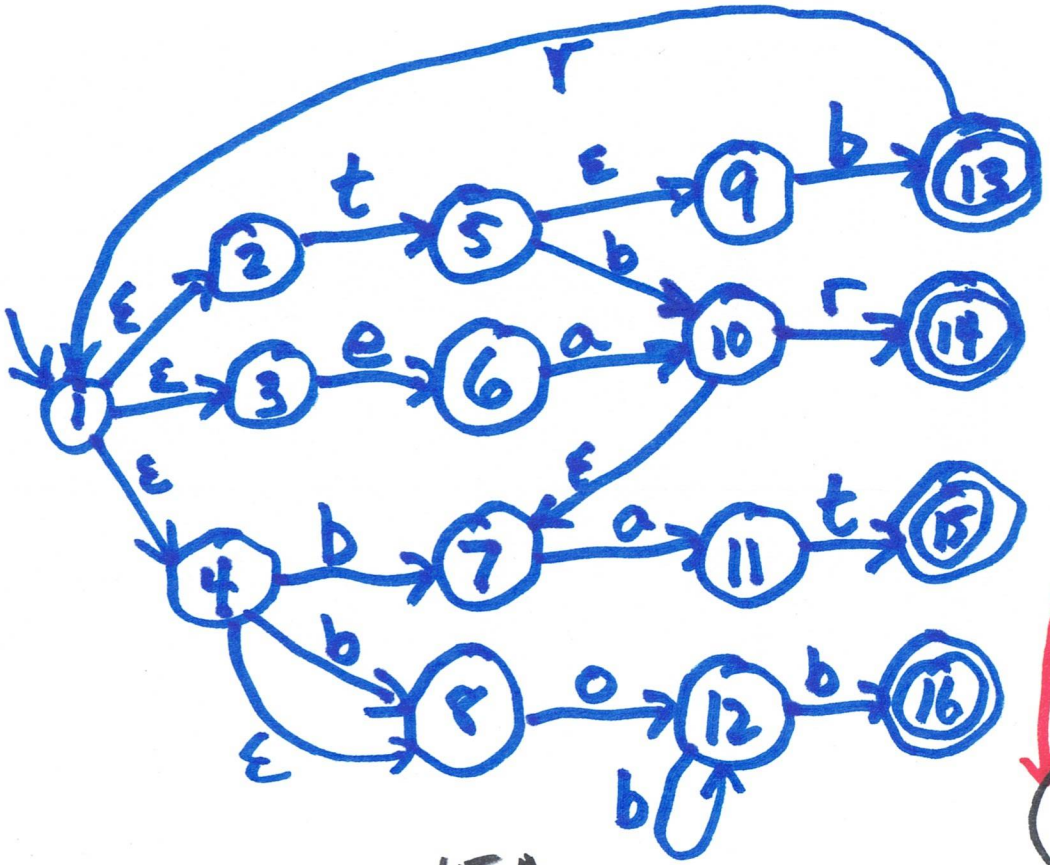
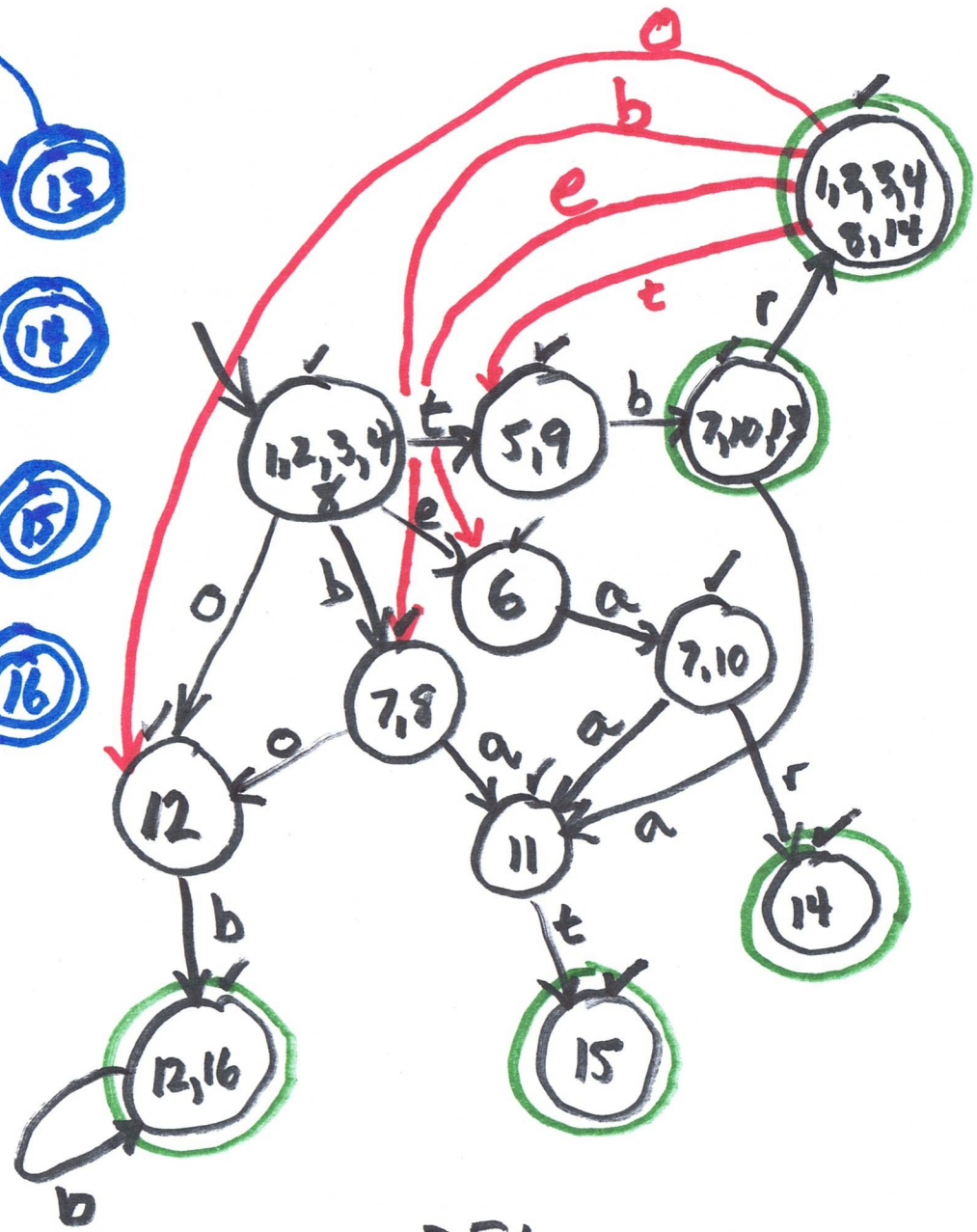
```
d0 = ε-closure(n0)
D = {d0}                    // set of all states of DFA
worklist = {d0}             // states discovered but not expanded

while (! worklist.isEmpty()) {
    remove a state d from worklist.

    for each character c ∈ Σ
        d' = ε-closure(δN(d,c))
        δD[d,c] = d'
        if d' ∉ D then
            D = D ∪ {d'}
            add d' to worklist
}
```

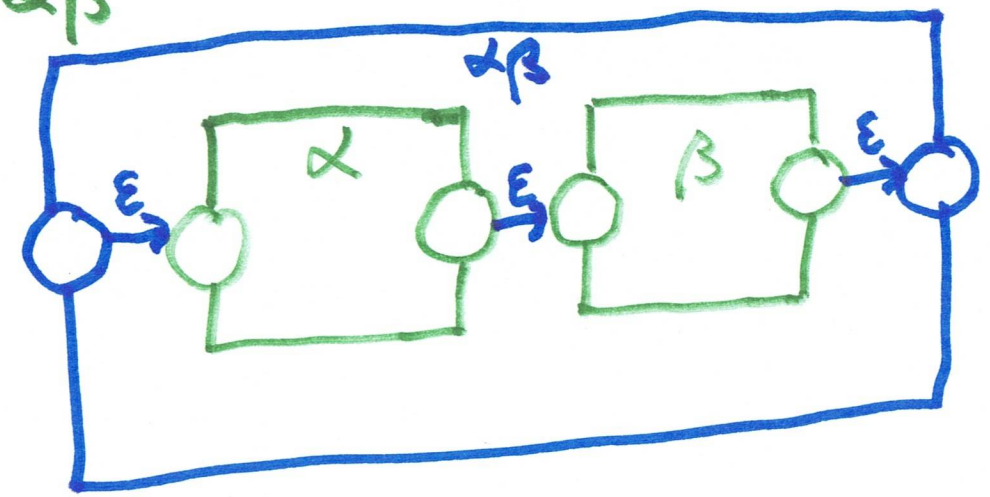

r
t
ε
b
ε
2
5
9
13
ε
e
a
r
1
3
6
10
14
ε
ε
ε
b
a
t
4
7
11
15
b
o
b
8
12
16
ε
b

NFA

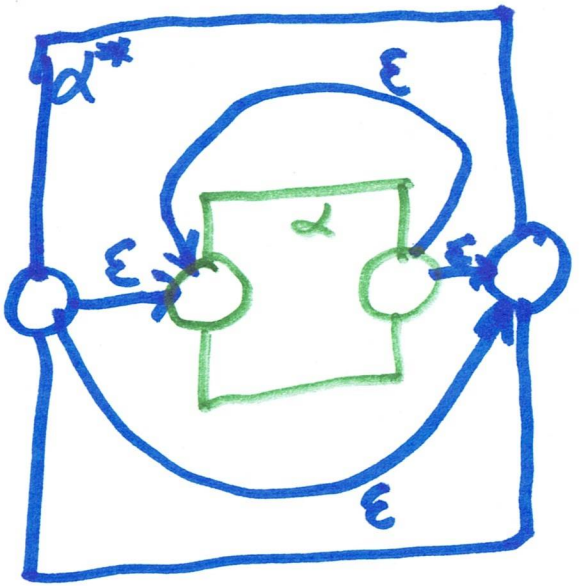

o
b
e
t
r
1,2,3,4
5,9
7,10,13
1,3,3,4
8,14
o
b
e
6
a
7,10
7,8
o
a
a
a
12
11
14
r
b
t
12,16
15
b

DFA

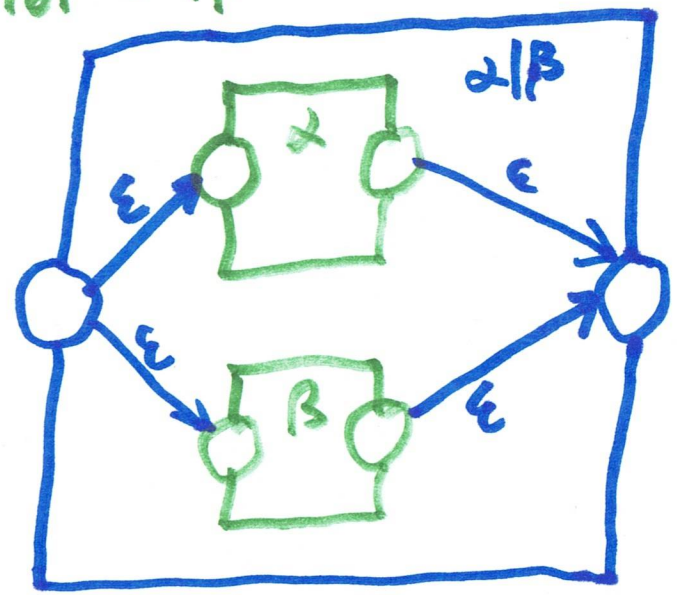
NFA $\alpha\beta$



NFA for α^*



NFA for $\alpha|\beta$



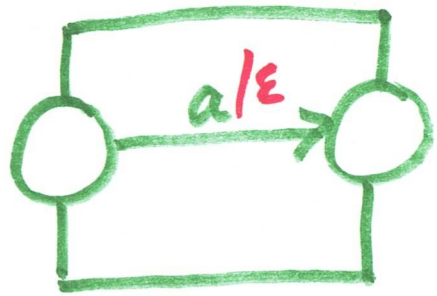
Constructing an NFA from a Regular Expression

(Thompson's construction)

Construct NFA recursively as we construct R.E.

ε	$a \in \Sigma$ }
a	
$\alpha\beta$	α, β are reg. expr.
$\alpha \mid \beta$	" " "
α^*	α is reg. expr.

NFA for $a \in \Sigma$ (or ε)



c*(a|b)