Automata theory

An algorithmic approach

Member(x, X)	:	returns **true** if $x \in X$, **false** otherwise.
Complement(X)	:	returns $U \setminus X$.
Intersection(X, Y)	:	returns $X \cap Y$.
Union(X, Y)	:	returns $X \cup Y$.
Empty(X)	:	returns **true** if $X = \emptyset$, **false** otherwise.
Universal(X)	:	returns **true** if $X = U$, **false** otherwise.
Included(X, Y)	:	returns **true** if $X \subseteq Y$, **false** otherwise.
Equal(X, Y)	:	returns **true** if $X = Y$, **false** otherwise.
Projection_1(R)	:	returns the set $\pi_1(R) = \{x \mid \exists y\ (x, y) \in R\}$.
Projection_2(R)	:	returns the set $\pi_2(R) = \{y \mid \exists y\ (x, y) \in R\}$.
Join(R, S)	:	returns $R \circ S = \{(x, z) \mid \exists y \in X\ (x, y) \in R \wedge (y, z) \in S\}$
Post(X, R)	:	returns $post_R(X) = \{y \in U \mid \exists x \in X : (x, y) \in R\}$.
Pre(X, R)	:	returns $pre_R(X) = \{y \in U \mid \exists x \in X : (y, x) \in R\}$.

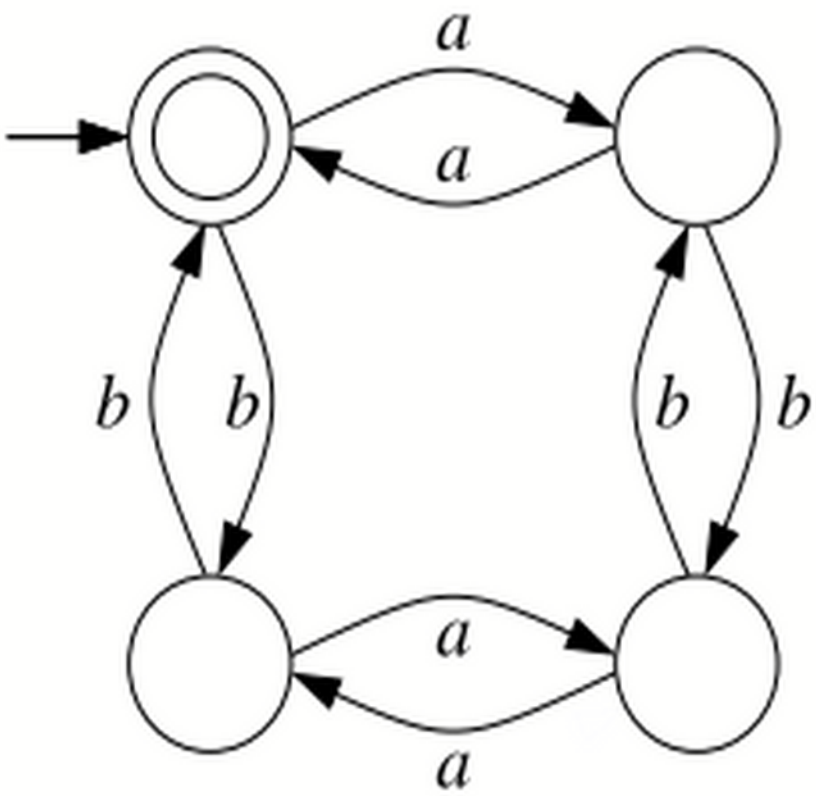
Automata classes and conversions

Regular expressions

$$r ::= \emptyset \mid \varepsilon \mid a \mid r_1 r_2 \mid r_1 + r_2 \mid r^* \qquad \textit{where } a \in \Sigma$$

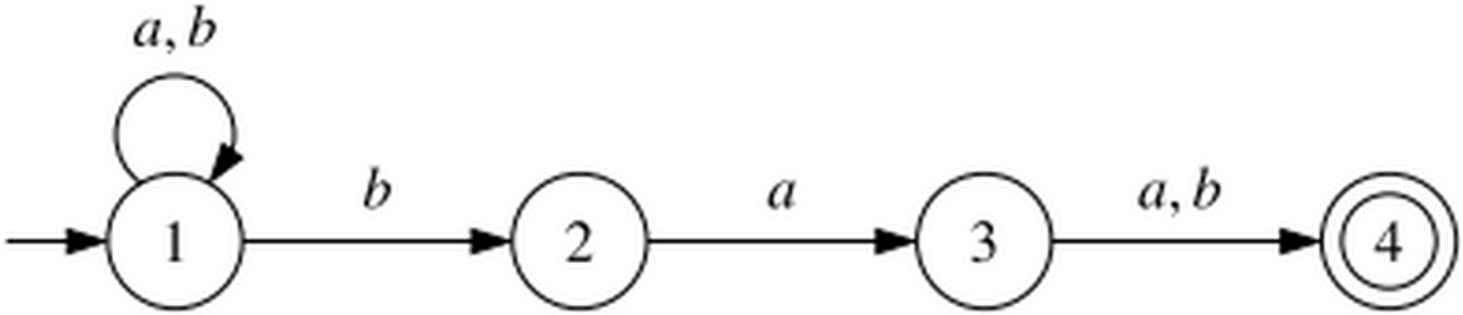
- $L(\emptyset) = \emptyset$,
- $L(\varepsilon) = \{\varepsilon\}$,
- $L(a) = \{a\}$,
- $L(r_1 r_2) = L(r_1) \cdot L(r_2)$, $\qquad L_1 \cdot L_2 = \{w_1 w_2 \in \Sigma^* \mid w_1 \in L_1, w_2 \in L_2\}$.
- $L(r_1 + r_2) = L(r_1) \cup L(r_2)$,
- $L(r^*) = L(r)^*$. $\qquad L^* = \bigcup_{i \geq 0} L^i$, where $L_0 = \{\varepsilon\}$ and $L_{i+1} = L^i \cdot L$

Deterministic finite automata (DFA)



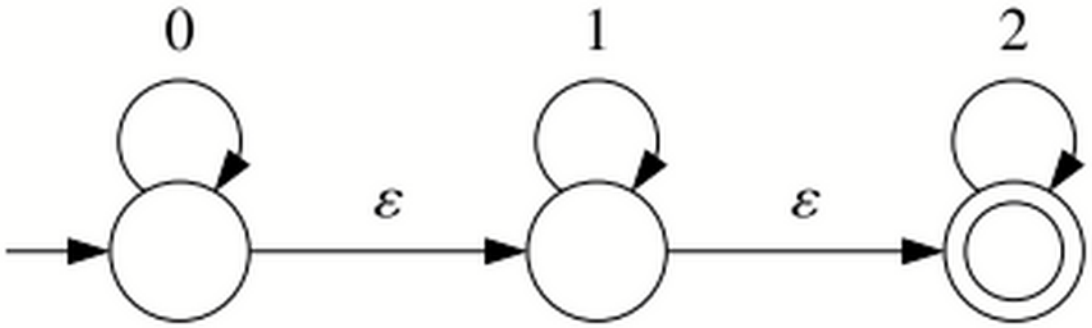
- Q *is a set of* states,
- Σ *is an alphabet,*
- $\delta\colon Q \times \Sigma \to Q$ *is a* transition function,
- $q_0 \in Q$ *is the* initial state, *and*
- $F \subseteq Q$ *is the set of* final states.

Nondeterministic finite automata (NFA)



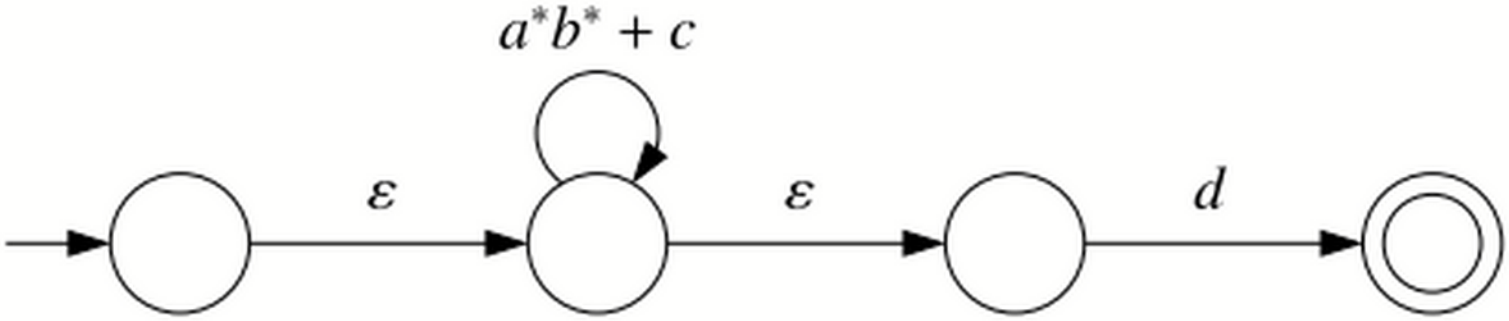
- $\delta\colon Q \times \Sigma \to \mathcal{P}(Q)$ *is a* transition relation.

Nondeterministic finite automata with epsilon-transitions (NFA-e)



- $\delta\colon\ Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$ *is a* transition relation.

Nondeterministic finite automata with regular-expression transitions (NFA-reg)



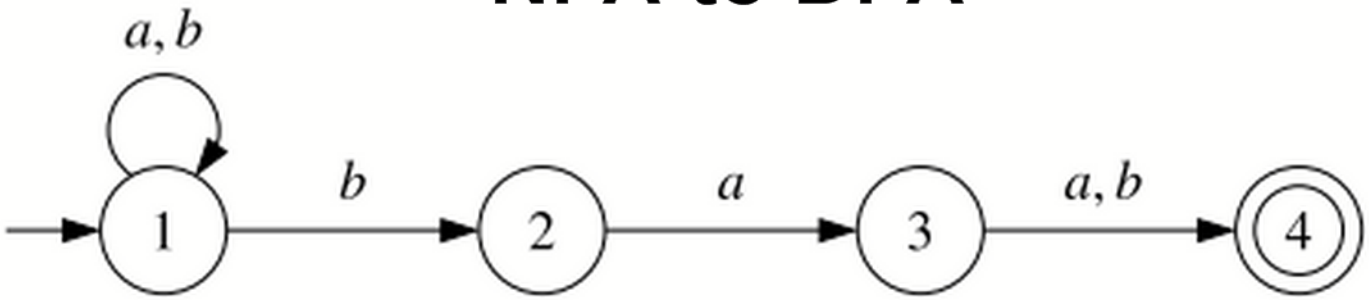
- $\delta\colon Q \times \mathcal{RE}(\Sigma) \to \mathcal{P}(Q)$ *is a relation such that* $\delta(q, r) = \emptyset$ *for all but a finite number of pairs* $(q, r) \in Q \times \mathcal{RE}(\Sigma)$.

Normal form

Definition 2.5 *Let* $A = (Q, \Sigma, \delta, q_0, F)$ *be an automaton. A state* $q \in Q$ *is* reachable from $q' \in Q$ *if* $q = q'$ *or if there exists a run* $q' \xrightarrow{a_1} \ldots \xrightarrow{a_n} q$ *on some input* $a_1 \ldots a_n \in \Sigma^*$. *A is in* normal form *if every state is reachable from the initial state.*

Conversions

NFA to DFA

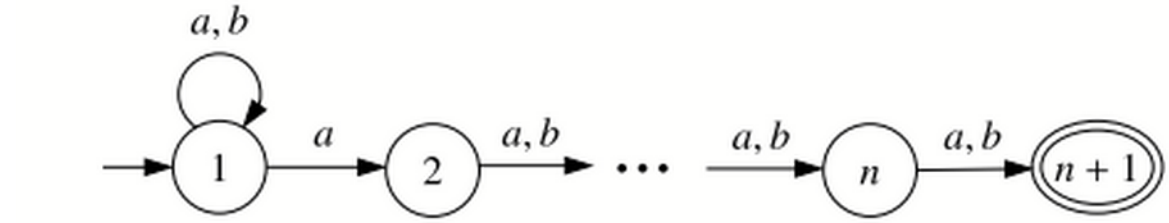


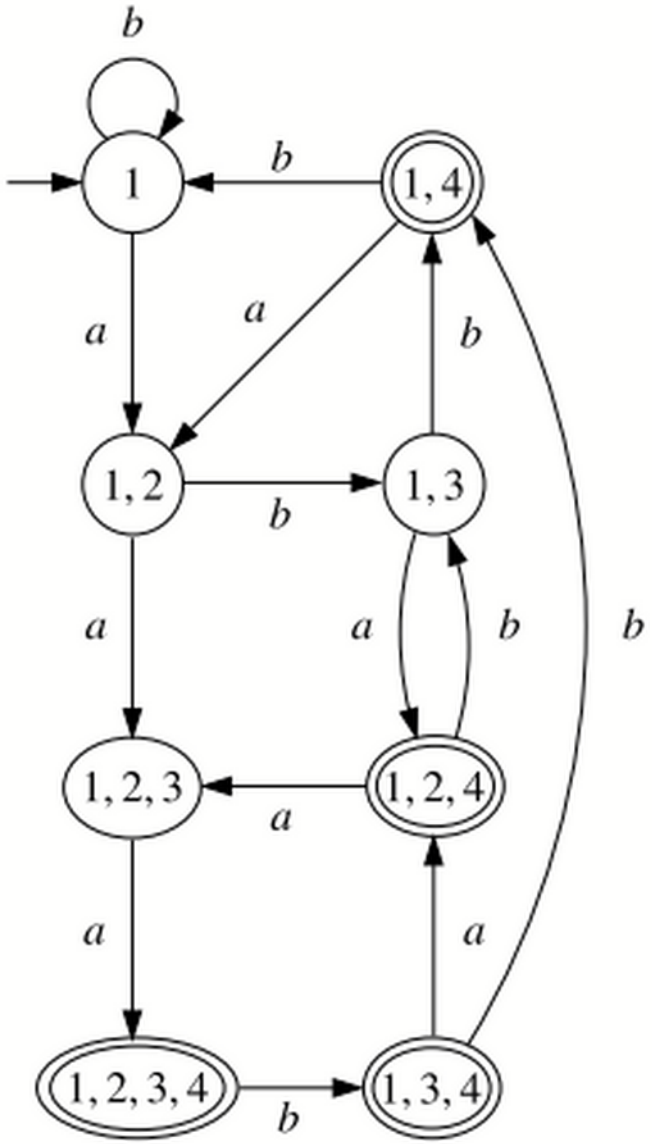
NFAtoDFA(A)

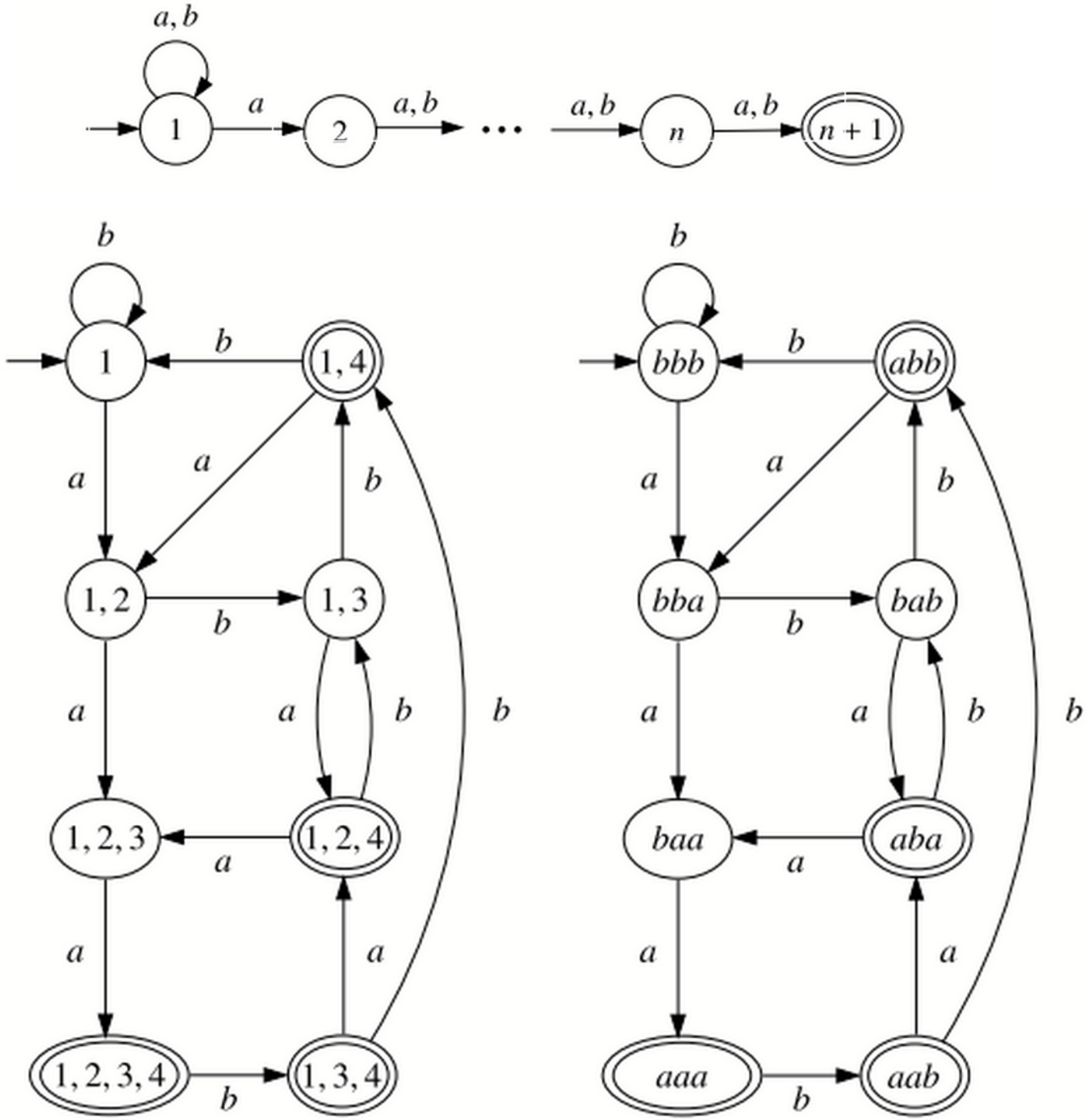
Input: NFA $A = (Q, \Sigma, \delta, q_0, F)$

Output: DFA $B = (\mathcal{Q}, \Sigma, \Delta, Q_0, \mathcal{F})$ with $L(B) = L(A)$

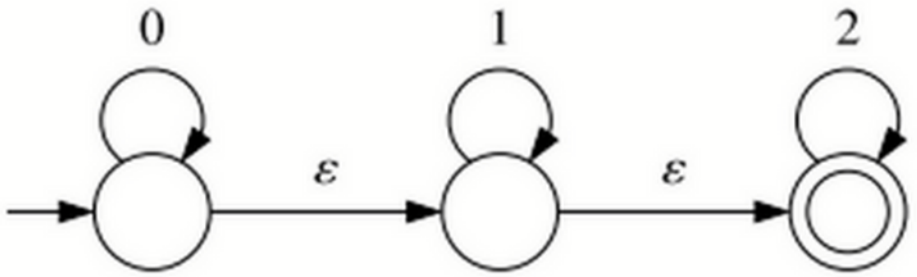
1 $\mathcal{Q}, \Delta, \mathcal{F} \leftarrow \emptyset$; $Q_0 \leftarrow \{q_0\}$
2 $\mathcal{W} = \{Q_0\}$
3 **while** $\mathcal{W} \neq \emptyset$ **do**
4 **pick** Q' **from** $\mathcal{W}$
5 **add** Q' **to** $\mathcal{Q}$
6 **if** $Q' \cap F \neq \emptyset$ **then add** Q' **to** $\mathcal{F}$
7 **for all** $a \in \Sigma$ **do**
8 $Q'' \leftarrow \bigcup_{q \in Q'} \delta(q, a)$
9 **if** $Q'' \notin \mathcal{Q}$ **then add** Q'' **to** $\mathcal{W}$
10 **add** (Q', a, Q'') **to** Δ


a, b
1
a
2
a, b
a, b
n
a, b
n + 1

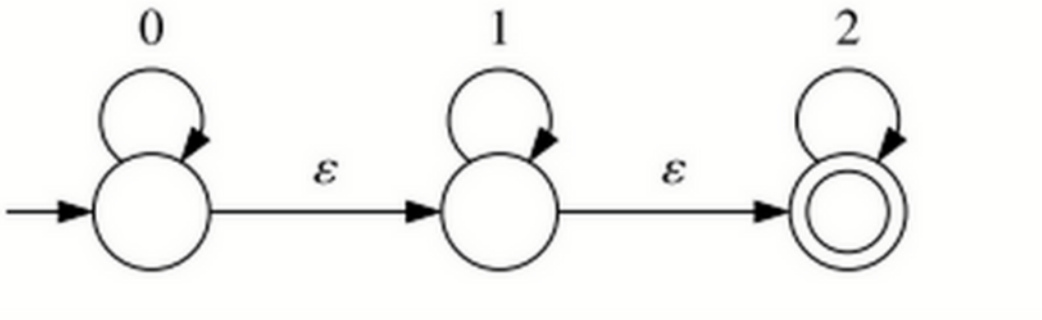

b
1
b
1, 4
a
a
b
1, 2
b
1, 3
a
a
b
b
1, 2, 3
a
1, 2, 4
a
a
1, 2, 3, 4
b
1, 3, 4

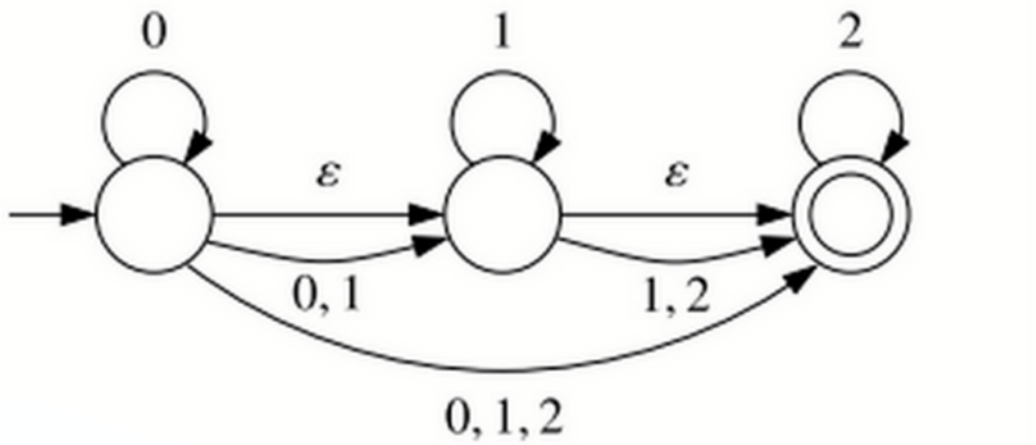

a, b
a
a, b
a, b
a, b
1
2
n
n + 1
b
1
1, 4
1, 2
1, 3
1, 2, 3
1, 2, 4
1, 2, 3, 4
1, 3, 4
bbb
abb
bba
bab
baa
aba
aaa
aab

NFA-e to NFA

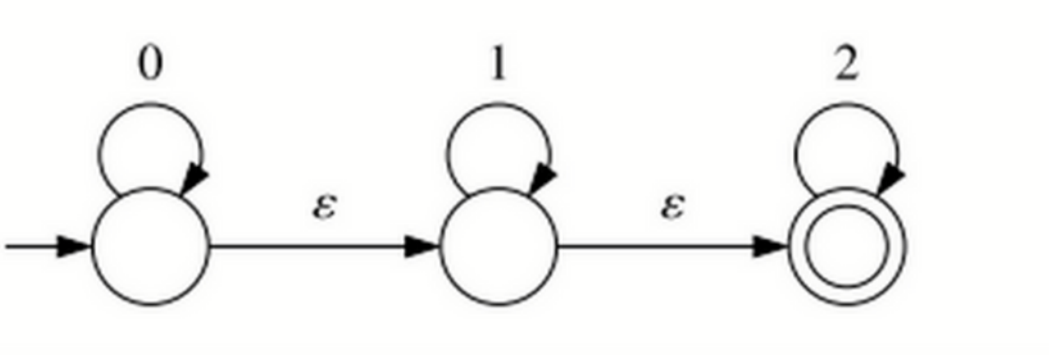


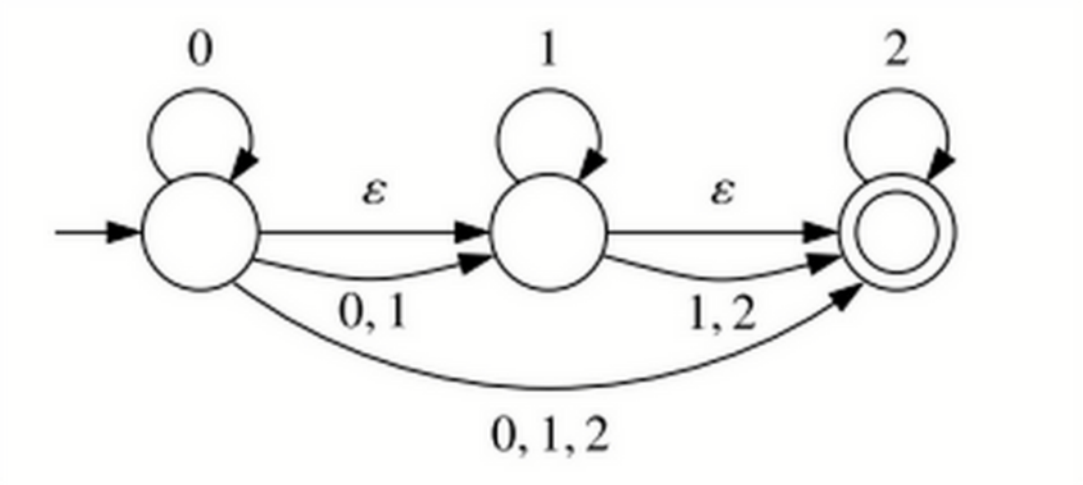
NFA-e to NFA

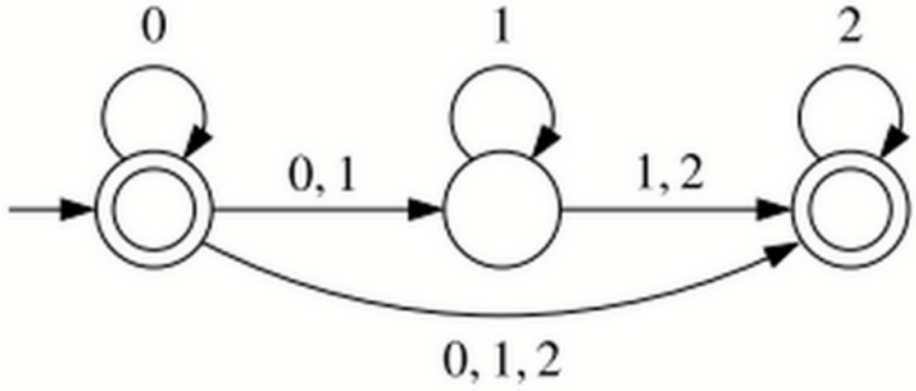




NFA-e to NFA







NFAεtoNFA(A)

Input: NFA-ε $A = (Q, \Sigma, \delta, q_0, F)$

Output: NFA $B = (Q', \Sigma, \delta', q_0', F')$ with $L(B) = L(A)$

```
1   q_0' ← q_0
2   Q' ← {q_0}; δ' ← ∅; F' ← F ∩ {q_0}
3   δ'' ← ∅; W ← {(q, α, q') ∈ δ | q = q_0}
4   while W ≠ ∅ do
5         pick (q_1, α, q_2) from W
6         if α ≠ ε then
7            add q_2 to Q'; add (q_1, α, q_2) to δ'; if q_2 ∈ F then add q_2 to F'
8            for all q_3 ∈ δ(q_2, ε) do
9               if (q_1, α, q_3) ∉ δ' then add (q_1, α, q_3) to W
10           for all a ∈ Σ, q_3 ∈ δ(q_2, a) do
11              if (q_2, a, q_3) ∉ δ' then add (q_2, a, q_3) to W
12        else  /* α = ε */
13           add (q_1, α, q_2) to δ''; if q_2 ∈ F then add q_0 to F'
14           for all β ∈ Σ ∪ {ε}, q_3 ∈ δ(q_2, β) do
15              if (q_1, β, q_3) ∉ δ' ∪ δ'' then add (q_1, β, q_3) to W
```

Proposition 2.7 *Let A be a NFA-ε, and let B = NFAεtoNFA(A). Then B is a NFA and L(A) = L(B).*

Proof:

(1) Termination:

every transition that leaves W is never added to W again

each iteration of the while loop removes a transition from W

(2) B is NFA. Easy.

(3) L(B) is included in L(A). Easy (transitions of B are shortcuts).

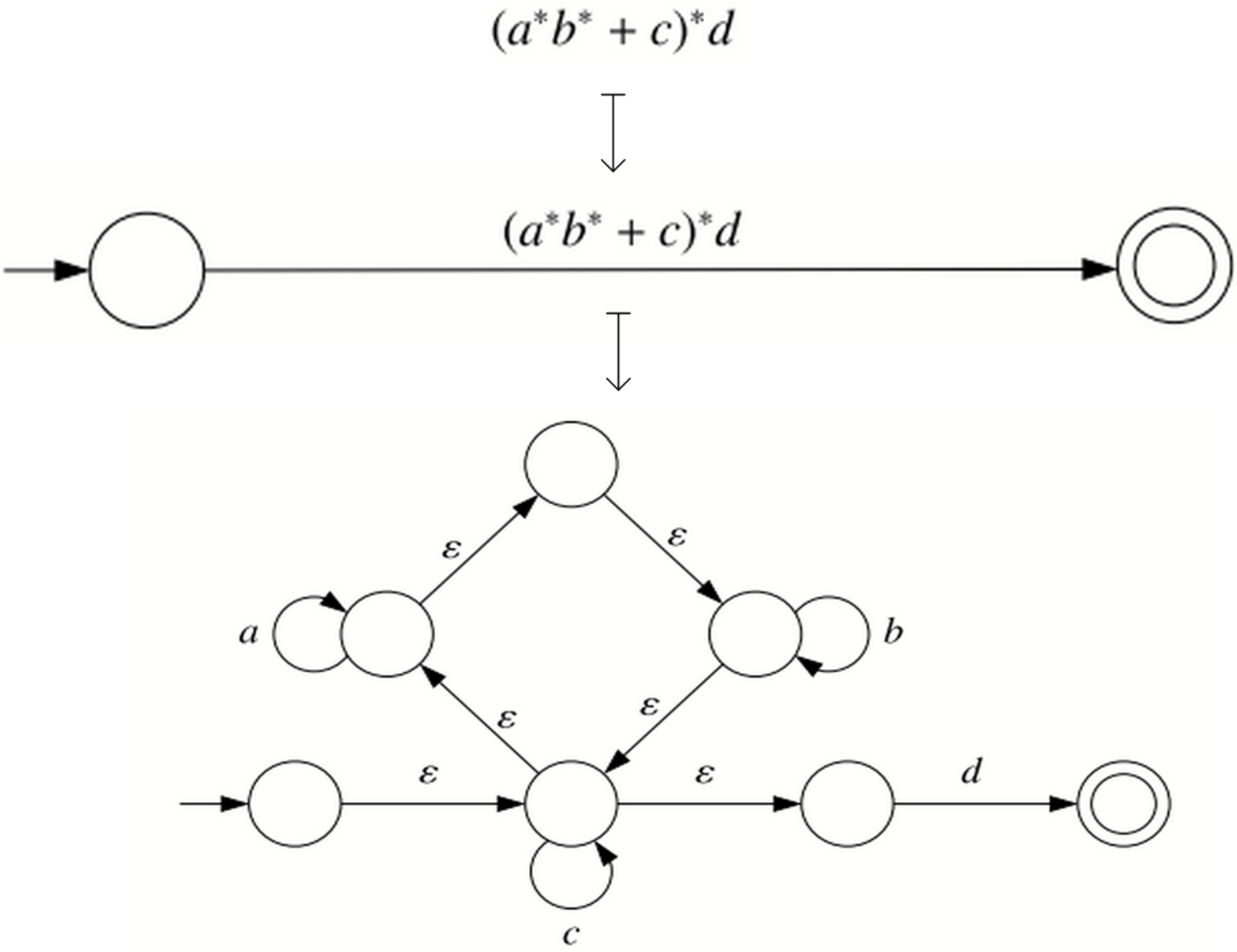
(4) L(A) is included in L(B).

(4.1) If e belongs to L(A), then e belongs to L(B)

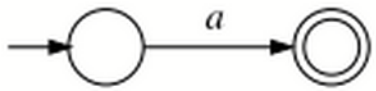
(4.2) If a nonempty word belongs to L(A), then it also belongs to L(B)

$$q_0 \xrightarrow{\varepsilon} \cdots \xrightarrow{\varepsilon} q_{m_1} \xrightarrow{a_1} q_{m_1+1} \xrightarrow{\varepsilon} \cdots \xrightarrow{\varepsilon} q_{m_n} \xrightarrow{a_n} q_{m_n+1} \xrightarrow{\varepsilon} \cdots \xrightarrow{\varepsilon} q_m$$

Regular expressions to NFA-e



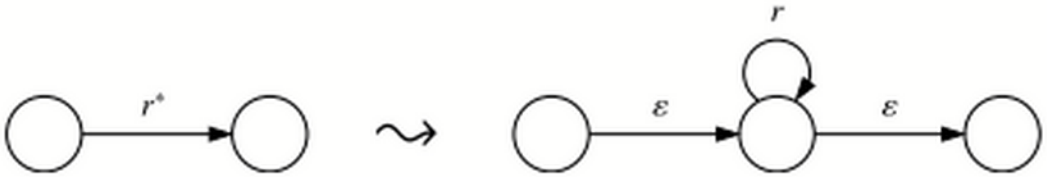
$$\emptyset \cdot r \rightsquigarrow \emptyset \qquad r \cdot \emptyset \rightsquigarrow \emptyset$$
$$r + \emptyset \rightsquigarrow r \qquad \emptyset + r \rightsquigarrow r$$
$$\emptyset^* \rightsquigarrow \epsilon$$



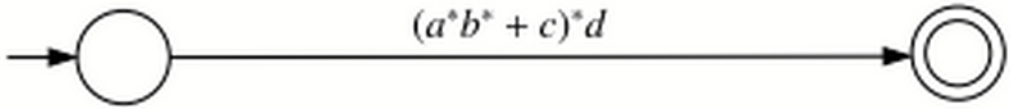
Automaton for the regular expression a, where $a \in \Sigma \cup \{\epsilon\}$

Rule for concatenation

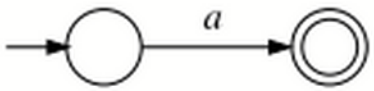
Rule for choice



Rule for Kleene iteration



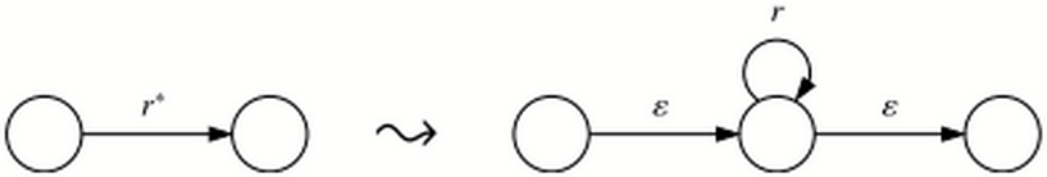
$$\emptyset \cdot r \rightsquigarrow \emptyset \qquad r \cdot \emptyset \rightsquigarrow \emptyset$$
$$r + \emptyset \rightsquigarrow r \qquad \emptyset + r \rightsquigarrow r$$
$$\emptyset^* \rightsquigarrow \epsilon$$



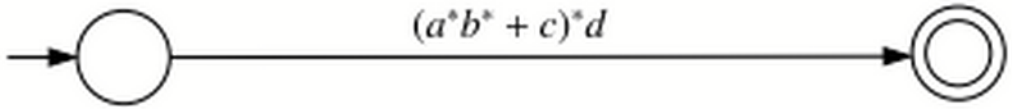
Automaton for the regular expression a, where $a \in \Sigma \cup \{\epsilon\}$

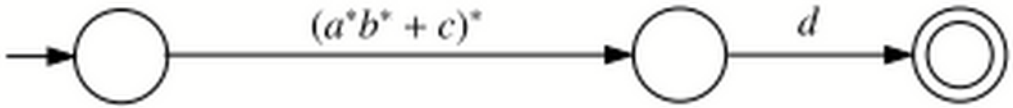
Rule for concatenation

Rule for choice

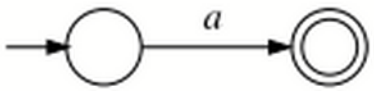


Rule for Kleene iteration





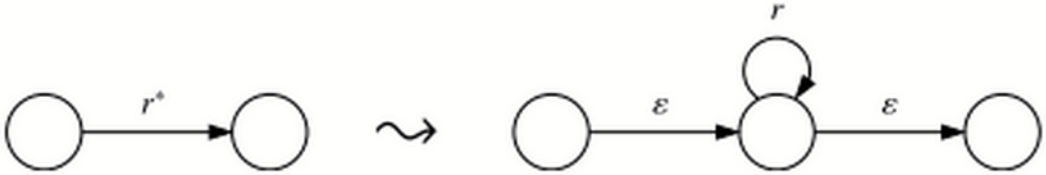
$$\emptyset \cdot r \rightsquigarrow \emptyset \qquad r \cdot \emptyset \rightsquigarrow \emptyset$$
$$r + \emptyset \rightsquigarrow r \qquad \emptyset + r \rightsquigarrow r$$
$$\emptyset^* \rightsquigarrow \epsilon$$



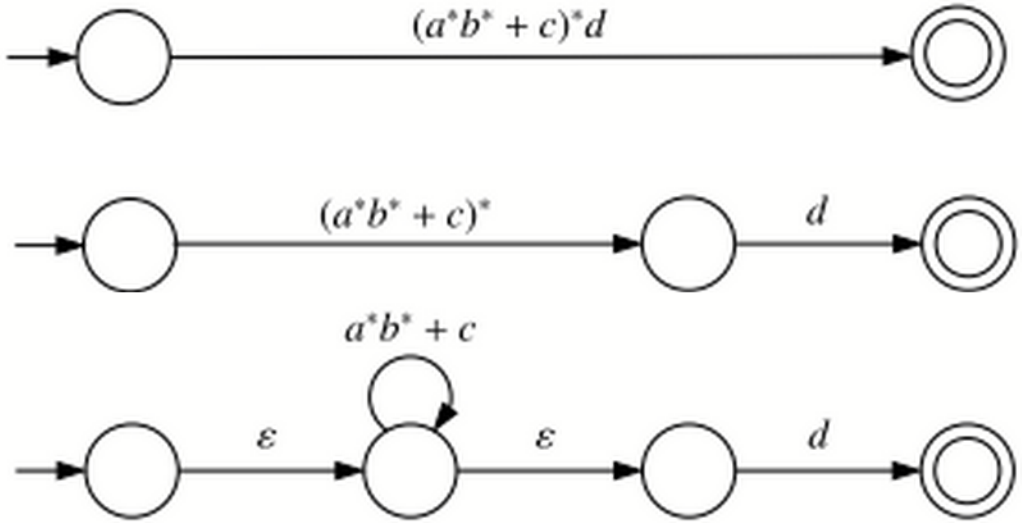
Automaton for the regular expression a, where $a \in \Sigma \cup \{\epsilon\}$

Rule for concatenation

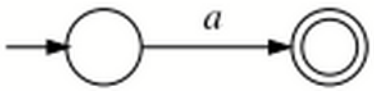
Rule for choice



Rule for Kleene iteration



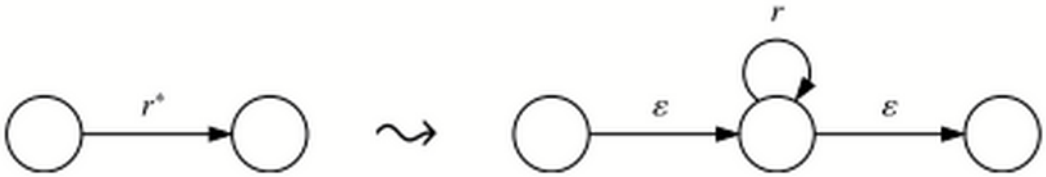
$$\emptyset \cdot r \rightsquigarrow \emptyset \qquad r \cdot \emptyset \rightsquigarrow \emptyset$$
$$r + \emptyset \rightsquigarrow r \qquad \emptyset + r \rightsquigarrow r$$
$$\emptyset^* \rightsquigarrow \epsilon$$



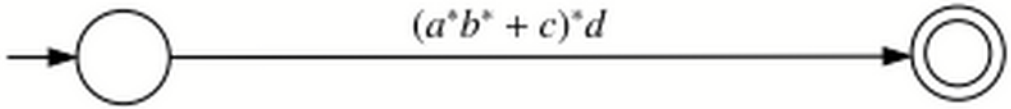
Automaton for the regular expression a, where $a \in \Sigma \cup \{\epsilon\}$

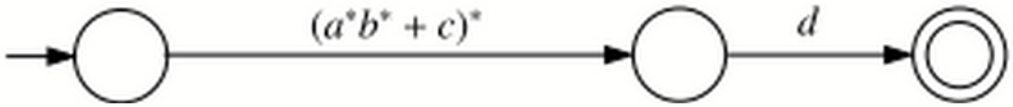
Rule for concatenation

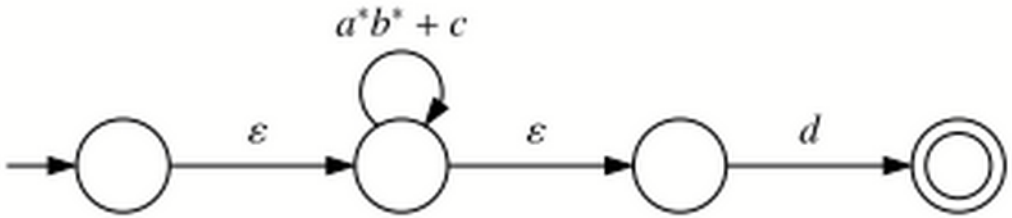
Rule for choice

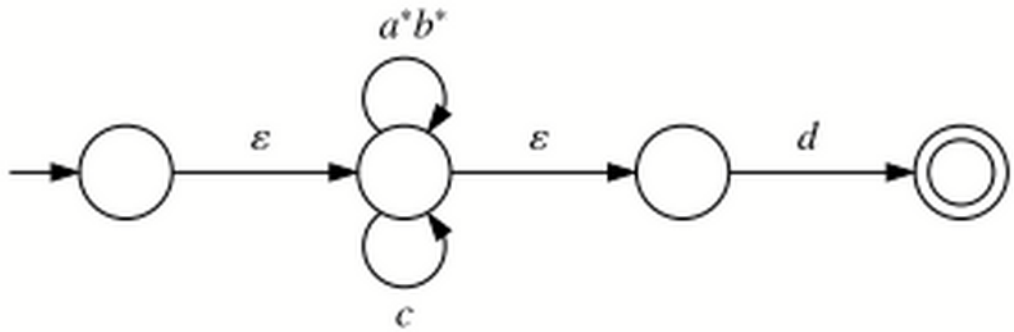


Rule for Kleene iteration

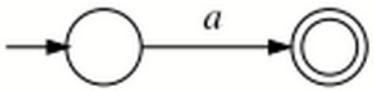








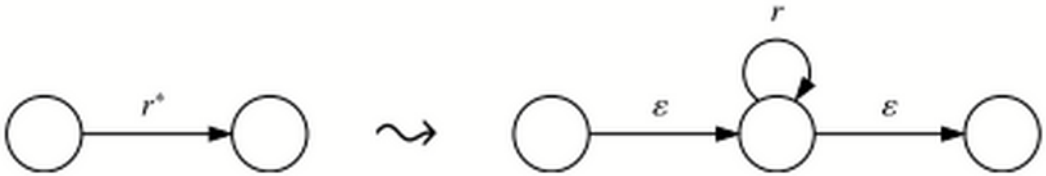
$$\emptyset \cdot r \rightsquigarrow \emptyset \qquad r \cdot \emptyset \rightsquigarrow \emptyset$$
$$r + \emptyset \rightsquigarrow r \qquad \emptyset + r \rightsquigarrow r$$
$$\emptyset^* \rightsquigarrow \epsilon$$



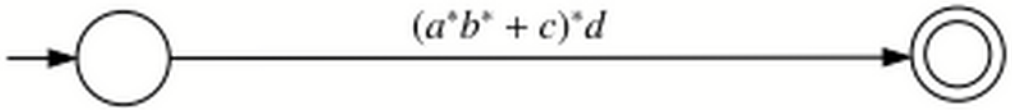
Automaton for the regular expression a, where $a \in \Sigma \cup \{\epsilon\}$

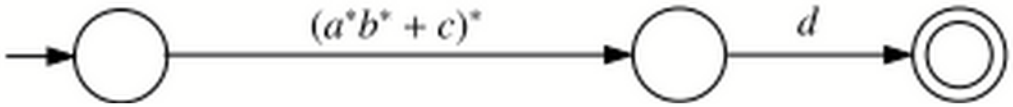
Rule for concatenation

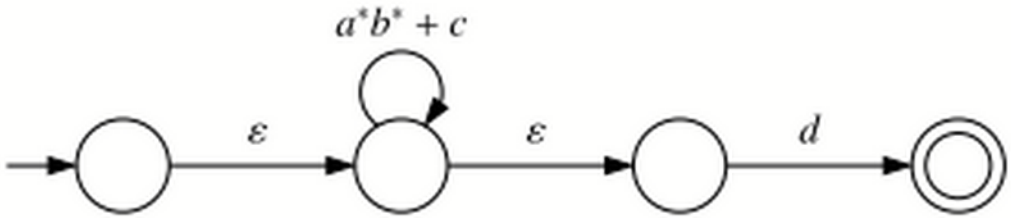
Rule for choice

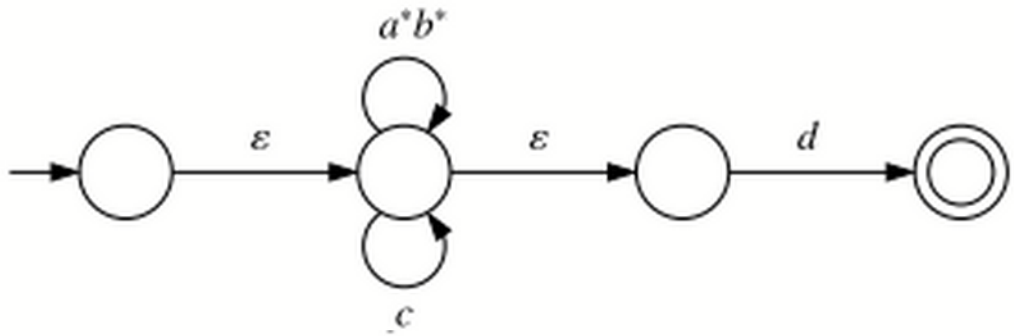


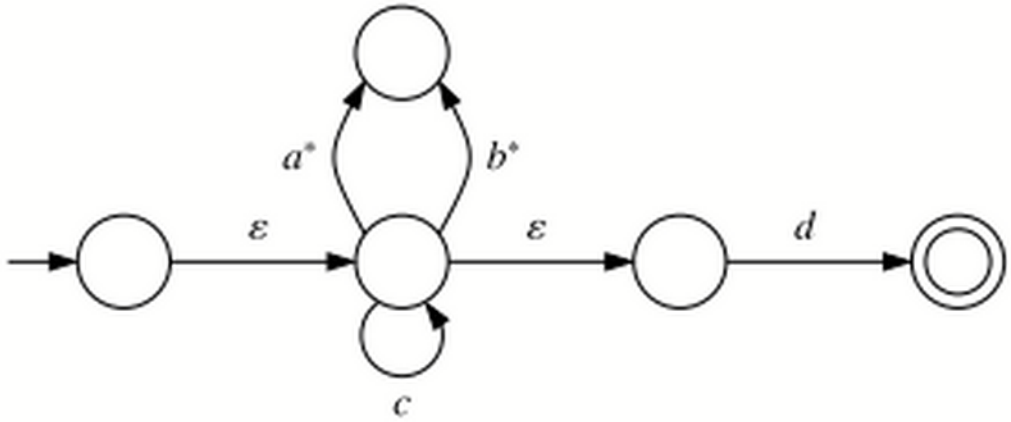
Rule for Kleene iteration

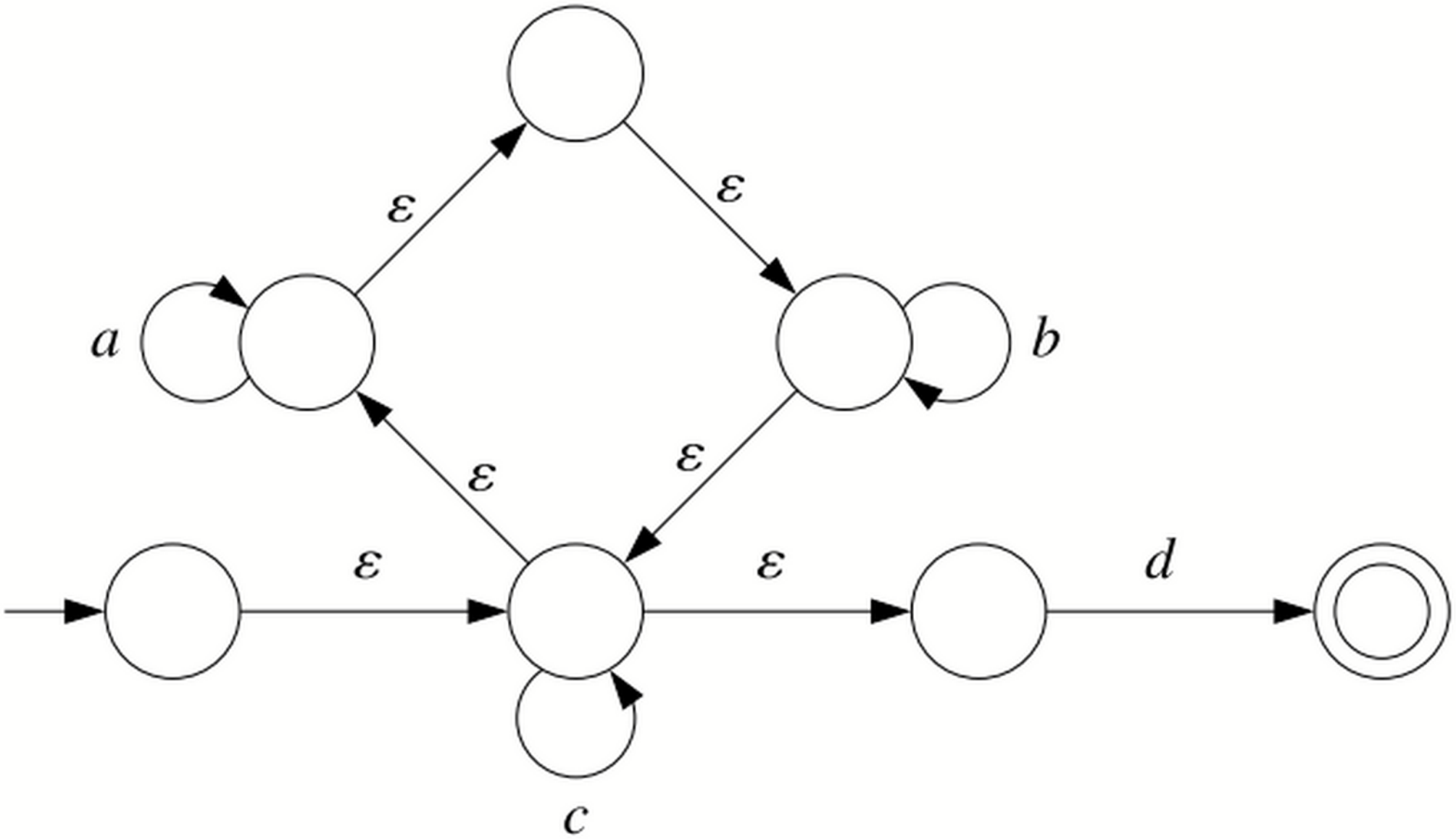






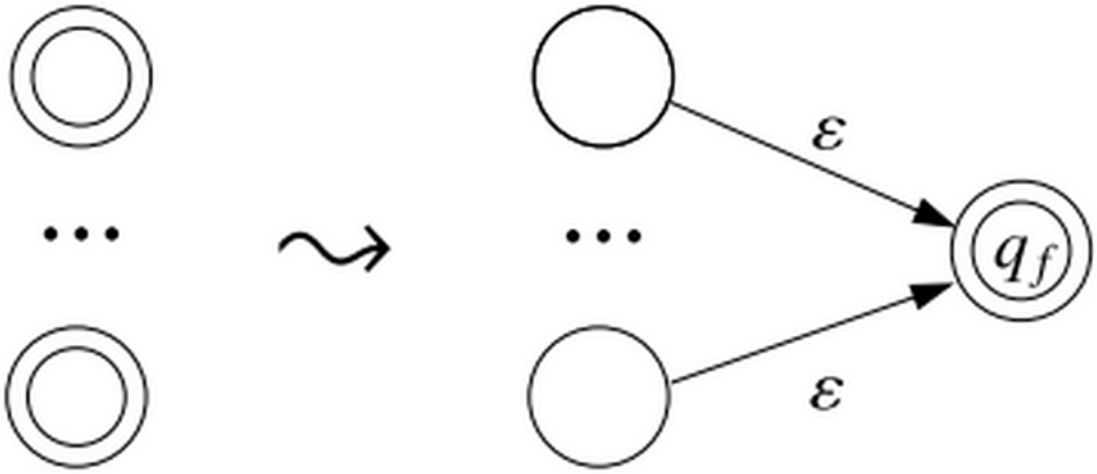




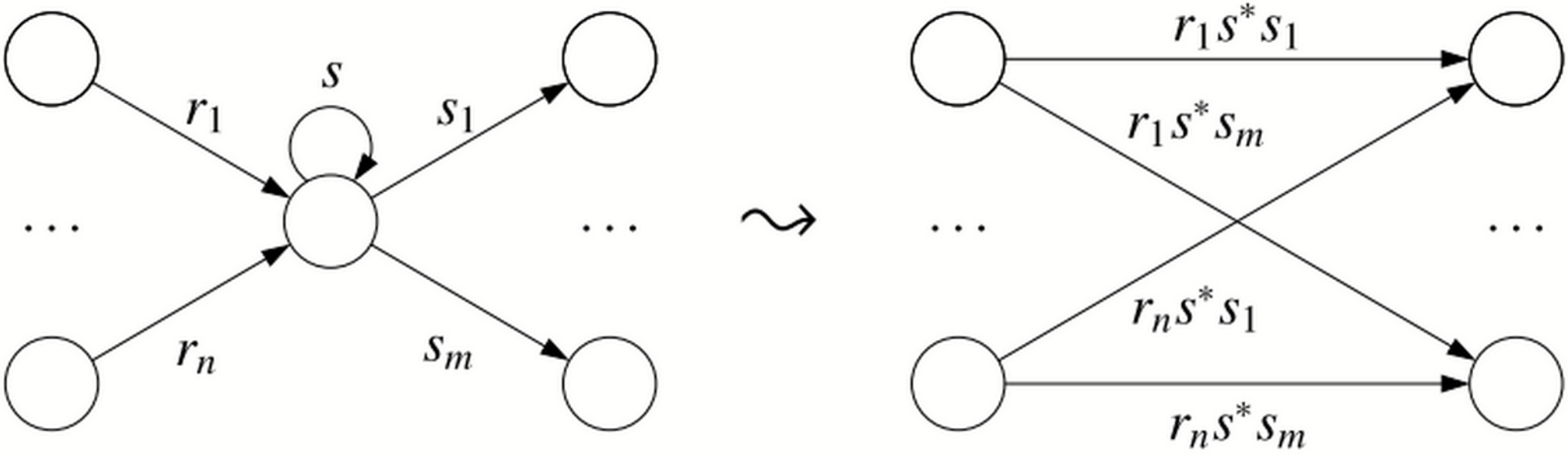

ε
ε
a
b
ε
ε
ε
ε
d
c

NFA-e to regular expressions

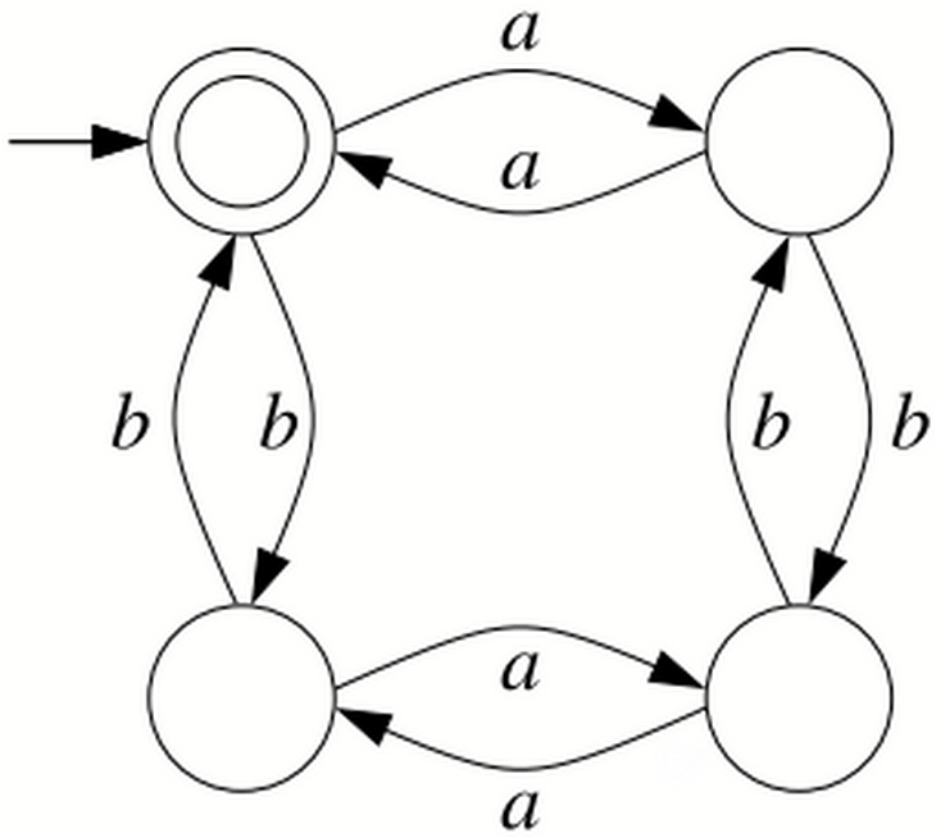
Preprocessing:

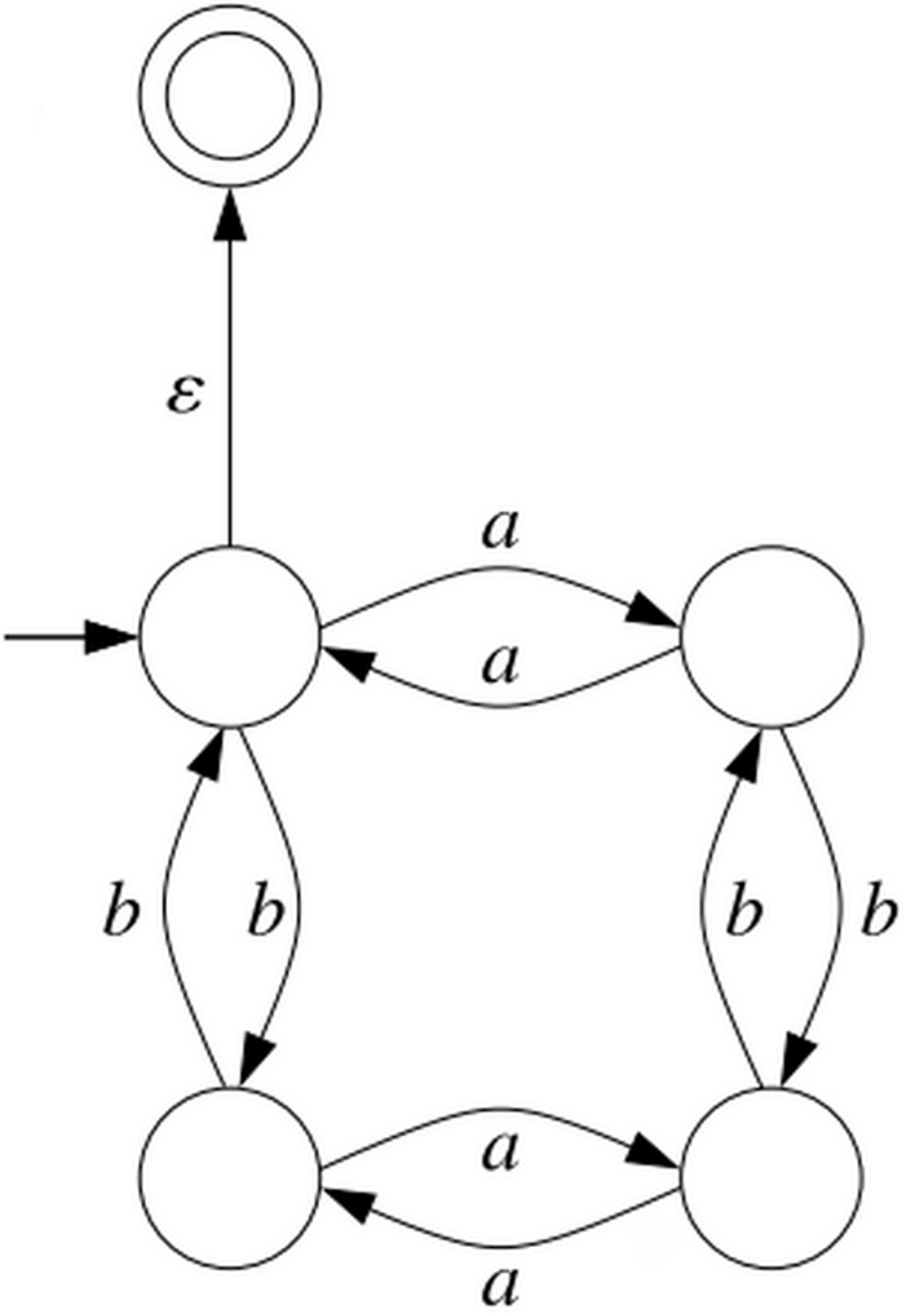


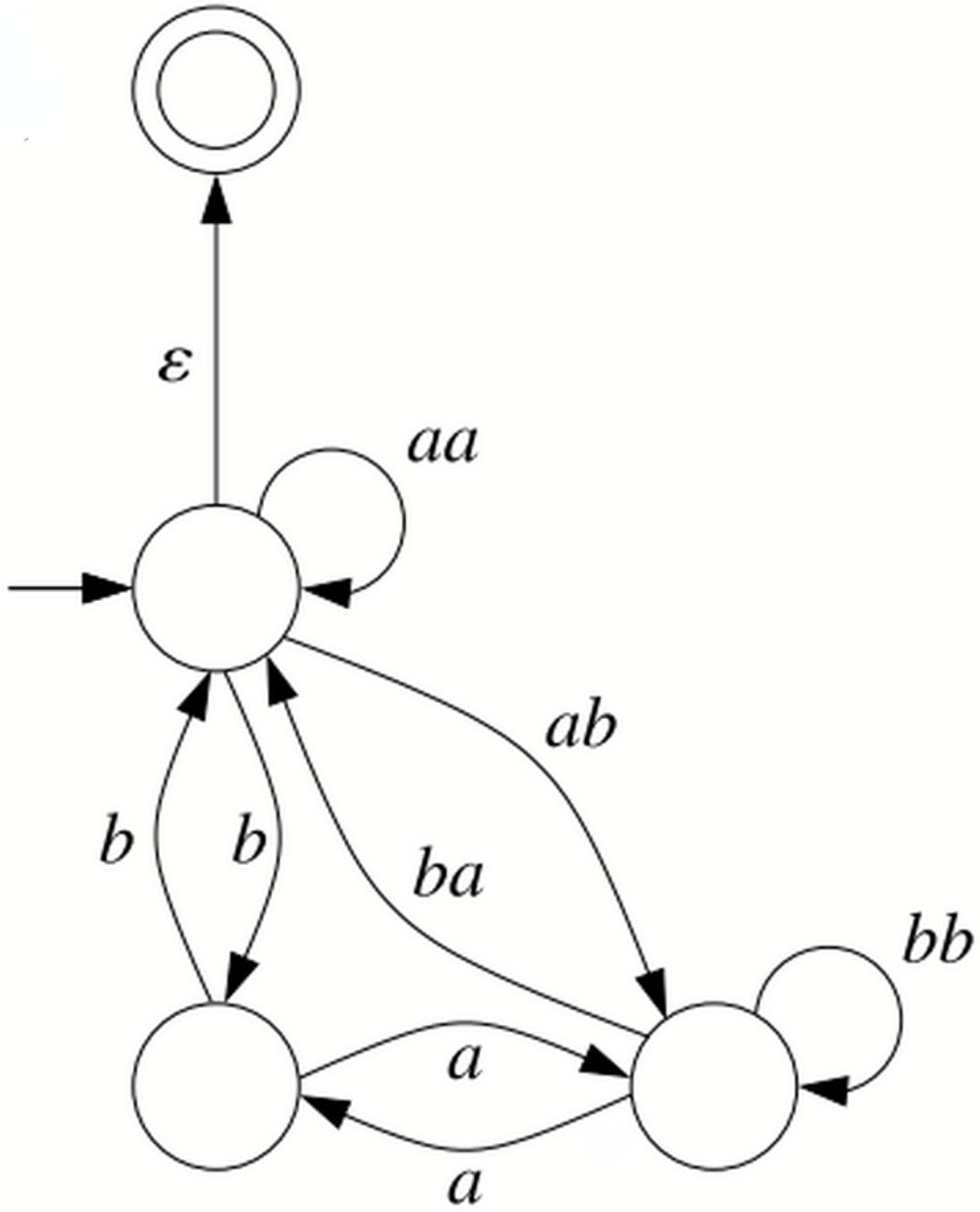
Processing:

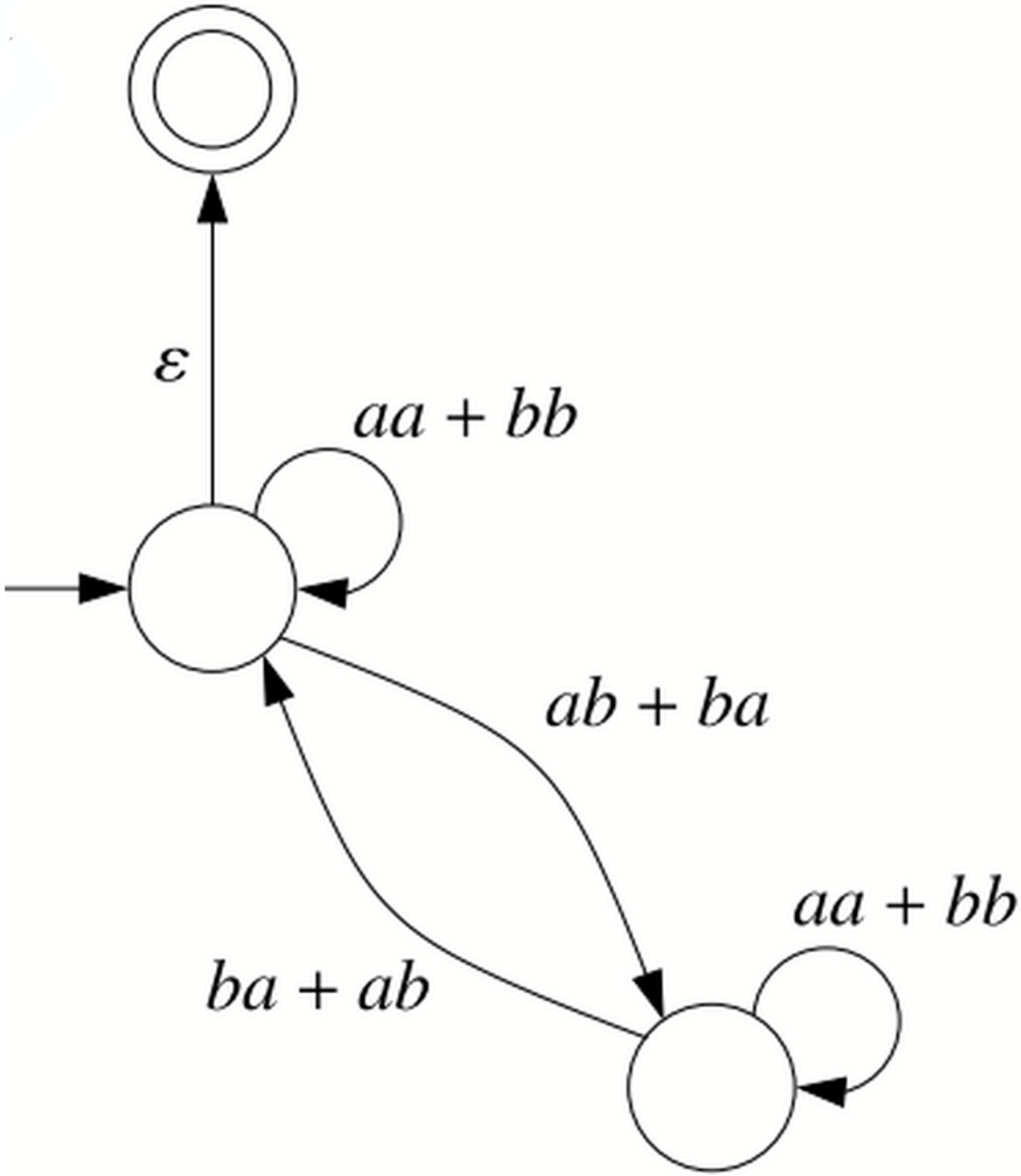


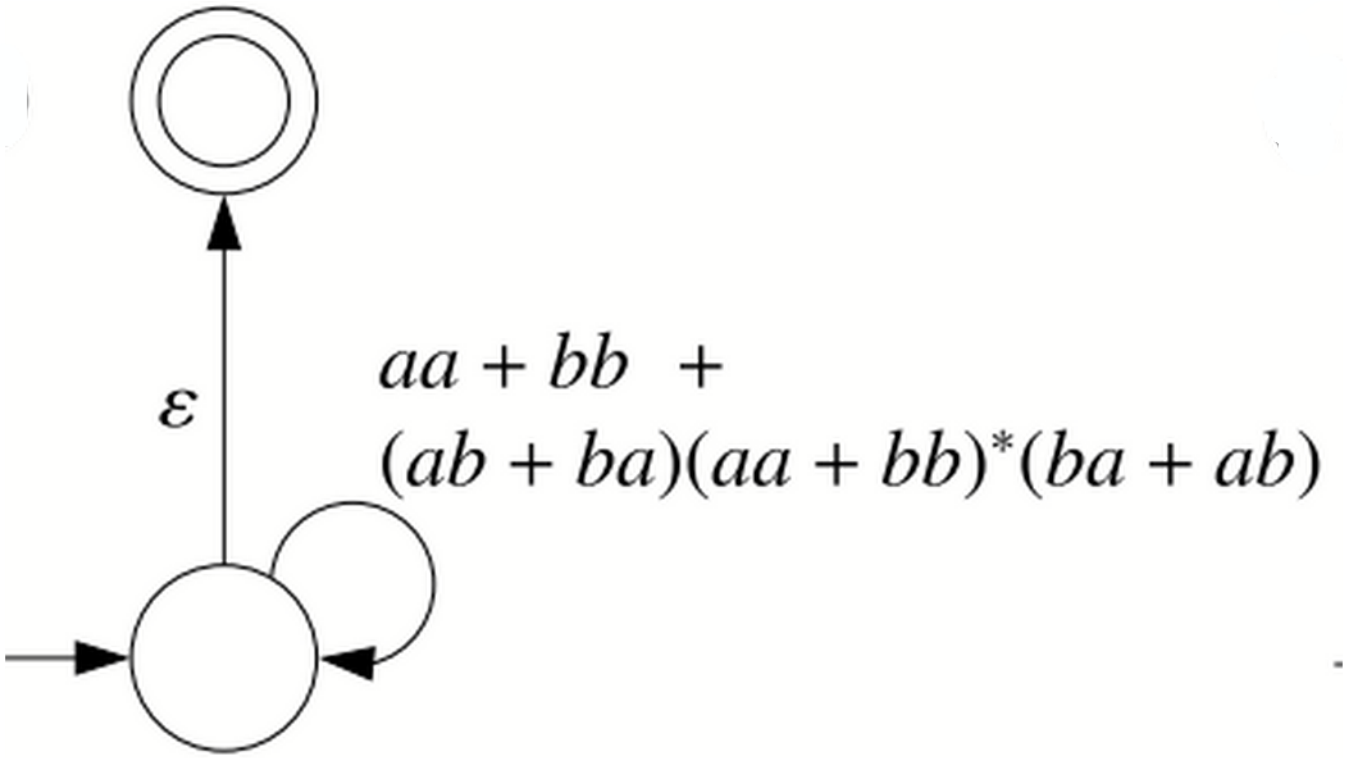
Postprocessing (if necessary):


a
a
b
b
b
b
a
a


ε
a
a
b
b
b
b
a
a


ε
aa
ab
b
b
ba
bb
a
a


ε
aa + bb
ab + ba
aa + bb
ba + ab


ε
aa + bb +
(ab + ba)(aa + bb)*(ba + ab)