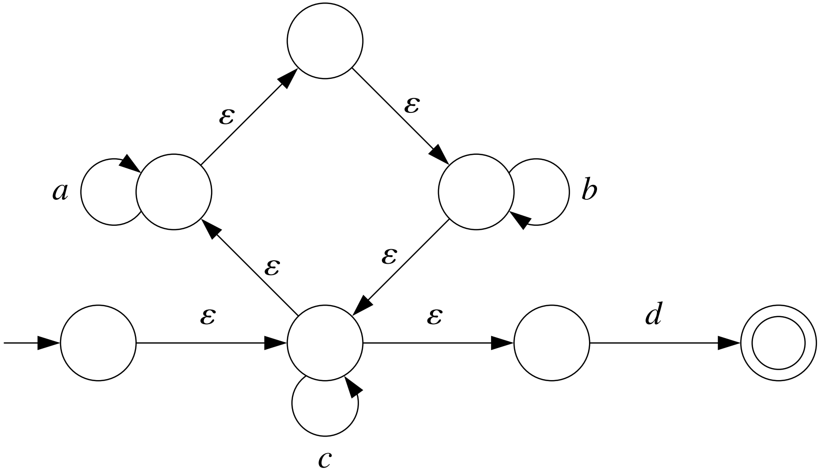
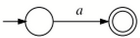
2.3 Regular expressions to NFA-ϵ

For the RE $(a^*b^* + c)^*d$, we intuitively construct the following NFA-ϵ:



Formally, we have the following rules:

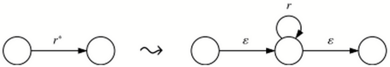
$$
\begin{array}{rcl@{\qquad}rcl}
\emptyset \cdot r & \rightsquigarrow & \emptyset & r \cdot \emptyset & \rightsquigarrow & \emptyset \\
r + \emptyset & \rightsquigarrow & r & \emptyset + r & \rightsquigarrow & r \\
\emptyset^* & \rightsquigarrow & \epsilon & & &
\end{array}
$$



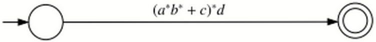
Automaton for the regular expression a, where $a \in \Sigma \cup \{\epsilon\}$

Rule for concatenation

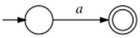
Rule for choice



Rule for Kleene iteration



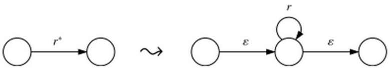
$$\begin{array}{lllllll} \emptyset \cdot r & \rightsquigarrow & \emptyset & \quad & r \cdot \emptyset & \rightsquigarrow & \emptyset \\ r + \emptyset & \rightsquigarrow & r & & \emptyset + r & \rightsquigarrow & r \\ \emptyset^* & \rightsquigarrow & \epsilon & & & & \end{array}$$



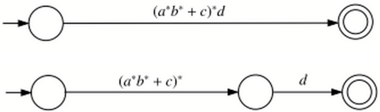
Automaton for the regular expression a, where $a \in \Sigma \cup \{\epsilon\}$

Rule for concatenation

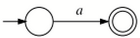
Rule for choice



Rule for Kleene iteration



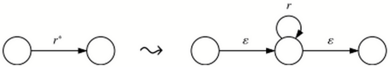
$$
\begin{array}{lcllcl}
\emptyset \cdot r & \rightsquigarrow & \emptyset & r \cdot \emptyset & \rightsquigarrow & \emptyset \\
r + \emptyset & \rightsquigarrow & r & \emptyset + r & \rightsquigarrow & r \\
\emptyset^* & \rightsquigarrow & \epsilon & & &
\end{array}
$$



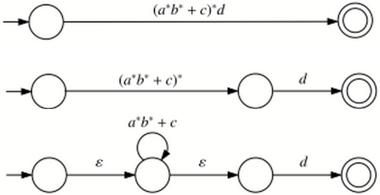
Automaton for the regular expression a, where $a \in \Sigma \cup \{\epsilon\}$

Rule for concatenation

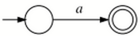
Rule for choice



Rule for Kleene iteration



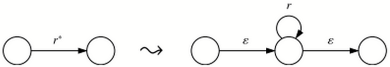
$$\emptyset \cdot r \rightsquigarrow \emptyset \qquad r \cdot \emptyset \rightsquigarrow \emptyset$$
$$r + \emptyset \rightsquigarrow r \qquad \emptyset + r \rightsquigarrow r$$
$$\emptyset^* \rightsquigarrow \epsilon$$



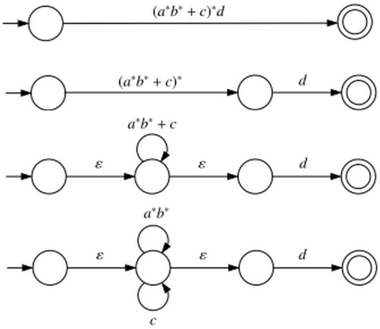
Automaton for the regular expression a, where $a \in \Sigma \cup \{\epsilon\}$

Rule for concatenation

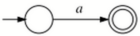
Rule for choice



Rule for Kleene iteration



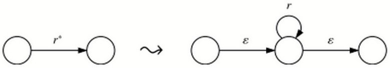
$$\emptyset \cdot r \rightsquigarrow \emptyset \qquad r \cdot \emptyset \rightsquigarrow \emptyset$$
$$r + \emptyset \rightsquigarrow r \qquad \emptyset + r \rightsquigarrow r$$
$$\emptyset^* \rightsquigarrow \epsilon$$



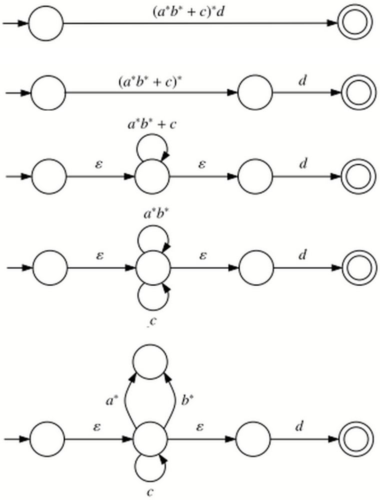
Automaton for the regular expression a, where $a \in \Sigma \cup \{\epsilon\}$

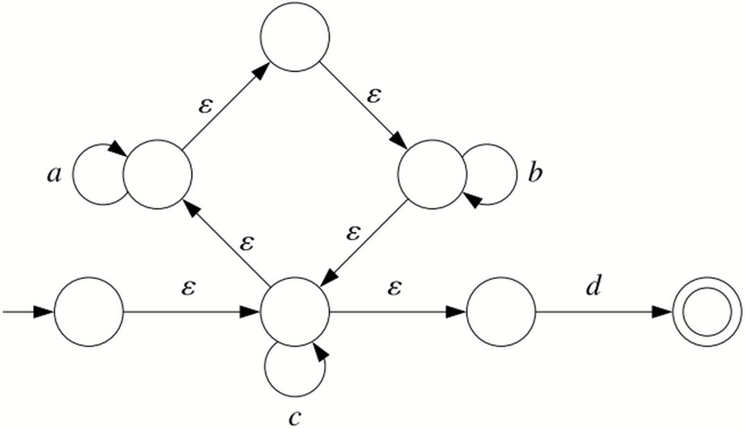
Rule for concatenation

Rule for choice

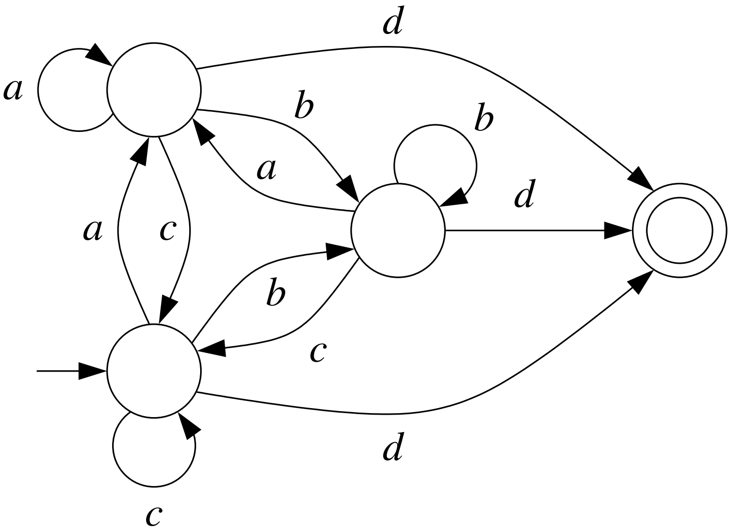


Rule for Kleene iteration



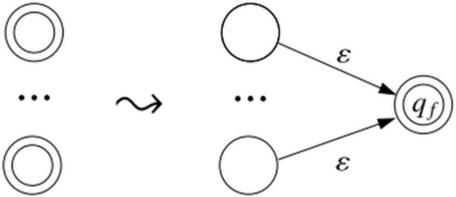

ε
ε
a
b
ε
ε
ε
ε
d
c

And finally, removing ϵ-transitions, we obtain:

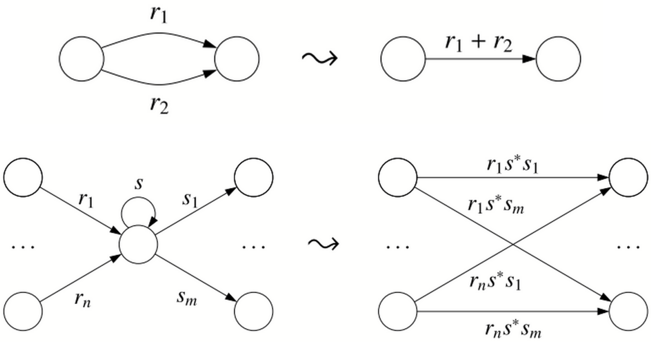


2.4 NFA-ϵ to regular expressions

Preprocessing:



Processing:

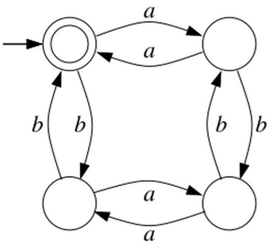


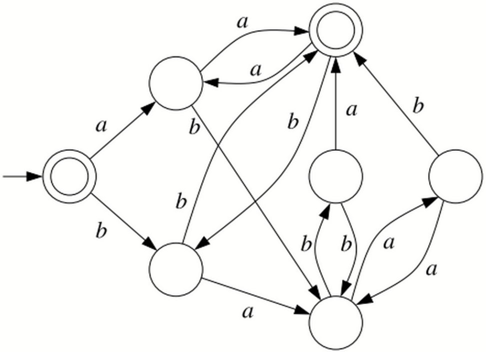
Postprocessing (if necessary):

3. Minimization and Reduction

In this section, we are going to look at the problem of constructing minimal size DFA's for a given regular language, or reducing the size of an NFA without changing the language it accepts.

Example 13





3.1 Residual

Definition 14

Let $L \subseteq \Sigma^*$ be a language, and $w \in \Sigma^*$ a word. The w-residual of L is the language

$$L^w := \{u \in \Sigma^*;\ wu \in L\}\,.$$

A language $L' \subseteq \Sigma^*$ is a residual of L if $L' = L^w$ for at least one $w \in \Sigma^*$.

We note that:

$$(L^w)^u = L^{wu}\,.$$

Relation between residuals and states:

Let A be a DFA and q a state of A.

Definition 15

The state-language $L_A(q)$ (or just $L(q)$) is the language recognized by A with q as initial state.

We remark:

- State-languages are residuals. For every state q of A, $L(q)$ is a residual of $L(A)$.
- Residuals are state-languages. For every residual R of $L(A)$, there is a state q such that $R = L(q)$.

Important consequence:

A regular language has finitely many residuals,

and, equivalently,

languages with infinitely many residuals are not regular.

Canonical DFA for a regular language:

Definition 16

Let $L \subseteq \Sigma^*$ be a formal language. The canonical DFA for L is the DFA $C_L := (Q_L, \Sigma, \delta_L, q_{0L}, F_L)$ given by

- Q_L is the set of residuals of L, *i.e.*, $Q_L = \{L^w;\ w \in \Sigma^*\}$
- $\delta(K, a) = K^a$ for every $K \in Q_L$ and $a \in \Sigma$
- $q_{0L} = L$, and
- $F_L = \{K \in Q_L\ ; \epsilon \in K\}$

Theorem 17

The canoncial DFA for L recognizes L.

Proof.

Let $w \in \Sigma^*$. We show by induction on $|w|$ that $w \in L$ iff $w \in L(C_L)$.

$$
\begin{array}{lll}
 & \epsilon \in L & (w = \epsilon) \\
\iff & L \in F_L & (\text{definition of } F_L) \\
\iff & q_{0L} \in F_L & (q_{0L} = L) \\
\iff & \epsilon \in L(C_L) & (q_{0L} \text{ is the initial state of } C_L)
\end{array}
$$

$$
\begin{array}{lll}
 & aw' \in L & \\
\iff & w' \in L^a & (\text{definition of } L^a) \\
\iff & w' \in L(C_{L^a}) & (\text{induction hypothesis}) \\
\iff & aw' \in L(C_L) & (\delta_L(L, a) = L^a)
\end{array}
$$

□

Definition 18

Let $L \subseteq \Sigma^*$ be a formal language. Define the relation $\equiv_L \subseteq \Sigma^* \times \Sigma^*$ by

$$x \equiv_L y \Leftrightarrow (\forall z \in \Sigma^*)[xz \in L \Leftrightarrow yz \in L]$$

Lemma 19

$\equiv_L$ is a right-invariant equivalence relation.

Here right-invariant means:

$$x \equiv_L y \Rightarrow xu \equiv_L yu \text{ for all } u \,.$$

Proof.

Clear! □

Theorem 20 (Myhill-Nerode)

Let $L \subseteq \Sigma^*$. Then the following are equivalent:

1. *L is regular*
2. *$\equiv_L$ has finite index (= number of equivalence classes)*
3. *L is the union of some of the finitely many equivalence classes of $\equiv_L$.*

Proof.

(1)⇒(2):

Let $L = L(A)$ for some DFA $A = (Q, \Sigma, \delta, q_0, F)$.

Then we have

$$\hat{\delta}(q_0, x) = \hat{\delta}(q_0, y) \quad \Rightarrow \quad x \equiv_L y \; .$$

Thus there are at most as many equivalence classes as A has states.

Proof.

(2)⇒(3):

Let $[x]$ be the equivalence class of x, $y \in [x]$ and $x \in L$.

Then, by the definition of $\equiv_L$, we have:

$$y \in L$$

Proof.

(3)⇒(1):

Define $A' = (Q', \Sigma, \delta', q_0', F')$ with

$$\begin{aligned}
Q' &:= \{[x];\ x \in \Sigma^*\} && (Q' \text{ finite!}) \\
q_0' &:= [\epsilon] \\
\delta'([x], a) &:= [xa] \quad \forall x \in \Sigma^*, a \in \Sigma && (\text{consistent!}) \\
F' &:= \{[x];\ x \in L\}
\end{aligned}$$

Then:

$$L(A') = L$$

□