## 3.2 Construction of Minimal DFAs

### Theorem 21

*For a given regular language $L$, let $A$ be the DFA constructed according to the Myhill-Nerode theorem. Then $A$ has, among all DFAs for $L$, a minimal number of states.*

### Proof.

Let $A = (Q, \Sigma, \delta, q_0, F)$ mit $L(A) = L$. Then

$$x \equiv_A y :\Leftrightarrow \hat{\delta}(q_0, x) = \hat{\delta}(q_0, y)$$

defines an equivalence relation which refines $\equiv_L$.
Thus: $|Q| = \text{index}(\equiv_A) \geq \text{index}(\equiv_L) =$ number of states of the Myhill-Nerode automaton. $\square$

# Algorithm for Constructing a Minimal DFA

Input: $A(Q, \Sigma, \delta, q_0, F)$ DFA $\quad (L = L(A))$

Output: equivalence relation on $Q$.

0. ensure that $A$ is in normal form
1. mark all pairs $\{q_i, q_j\} \in Q^2$ with

$$q_i \in F \text{ and } q_j \notin F \text{ resp. } q_i \notin F \text{ and } q_j \in F\ .$$

2. **for** all unmarked pairs $\{q_i, q_j\} \in Q^2$, $q_i \neq q_j$ **do**

    **if** $(\exists a \in \Sigma)[\{\delta(q_i, a), \delta(q_j, a)\}$ is marked$]$ **then**

        mark $\{q_i, q_j\}$;

        **for** all $\{q, q'\}$ in $\{q_i, q_j\}$'s list **do**

            mark $\{q, q'\}$ and remove it from list;

            do this recursively for all pairs in the list of $\{q, q'\}$, and so on.

        **od**

    **else**

        **for** all $a \in \Sigma$ **do**

            **if** $\delta(q_i, a) \neq \delta(q_j, a)$ **then**

                enter $\{q_i, q_j\}$ into the list of $\{\delta(q_i, a), \delta(q_j, a)\}$

            **fi**

        **od**

    **fi**

**od**

3. Output: $q$ equivalent to $q' \Leftrightarrow \{q, q'\}$ *not* marked.

## Theorem 22

*The above algorithm constructs a minimal DFA for $L(A)$.*

## Proof.

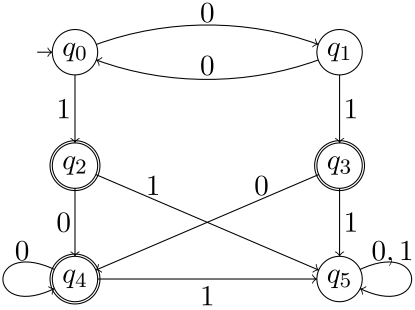
Let $A' = (Q', \Sigma', \delta', q'_0, F')$ be the DFA constructed using the equivalence classes determined by the algorithm.
Obviously $L(A) = L(A')$.
We have: $\{q, q'\}$ becomes marked iff

$$(\exists\, w \in \Sigma^*)[\hat{\delta}(q, w) \in F \wedge \hat{\delta}(q', w) \notin F \text{ or vice versa}],$$

as can be seen by a simple induction on $|w|$.
Thus: The number of states of $A'$ (*viz.*, $|Q'|$) equals the index of $\equiv_L$. □

## Example 23

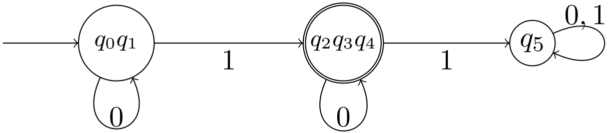
automaton $A$:




| | $q_0$ | $q_1$ | $q_2$ | $q_3$ | $q_4$ | $q_5$ |
|---|---|---|---|---|---|---|
| $q_0$ | / | / | / | / | / | / |
| $q_1$ | | / | / | / | / | / |
| $q_2$ | × | × | / | / | / | / |
| $q_3$ | × | × | | / | / | / |
| $q_4$ | × | × | | | / | / |
| $q_5$ | × | × | × | × | × | / |

automaton $A'$:

$L(A') = 0^*10^*$



## Theorem 24

*Let* $A = (Q, \Sigma, \delta, q_0, F)$ *be a DFA. Then the running time for the above minimization algorithm is* $O(|Q|^2|\Sigma|)$.

## Proof.

For each $a \in \Sigma$, each position in the table is visited only a constant number of times. □

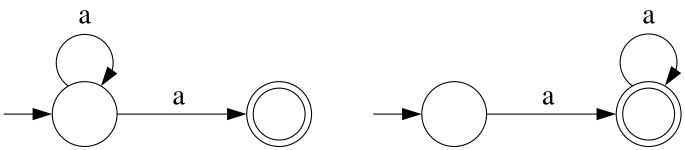
**Remark:**

The above minimization algorithm

- starts with a very coarse partition of the state set $Q$, containing $\equiv_L$
- splits a class of the partition whenever it has to
- does this as long as any further splitting might be possible
- finally forms the quotient automaton defined by the final partition of $Q$ (which is a coarsening of $\equiv_A$)

# 3.3 Minimizing NFAs

We first observe that a minimal NFA need not be unique (unlike the situation for DFAs):



**Minimal NFAs are hard to compute:**

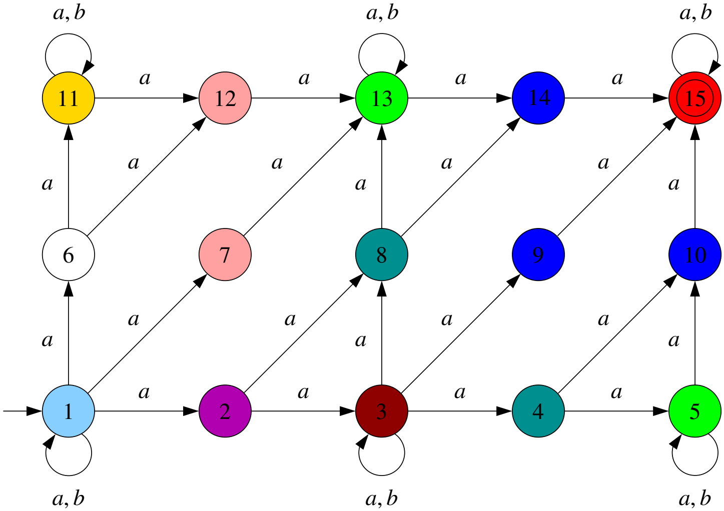
Theorem 25

*The following decision problem is PSPACE-complete: given an NFA $A$ and a number $k \geq 1$, is there an NFA with at most $k$ states which is equivalent to $A$.*

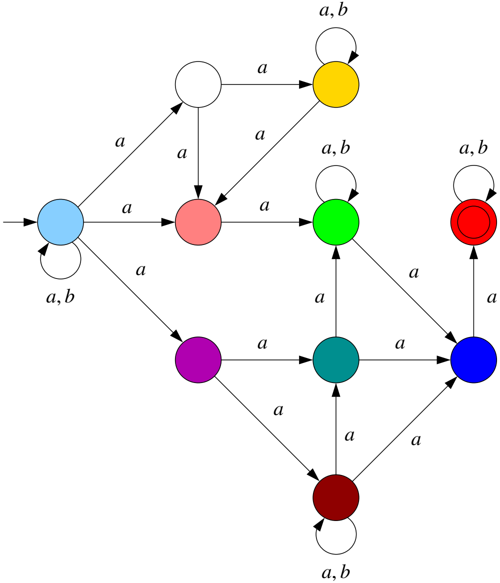
No proof.

However, quite often we can still compute a partition of the state set $Q$ of a given NFA which leads to a reduction of the number of states.

Example 26



Constructing the quotient automaton, we obtain



# What is a „suitable" partition?

- The quotient w.r.t. the partition must recognize the same language as the original NFA.
- So, by the Lemma, we can take any partition that refines the language partition.
- A partition refines the language partition iff states in the same block recognize the same language (states in different blocks may not recognize different langauges, though!).
- Such partitions necessarily refine the partition $\{F, Q \setminus F\}$.

## Computing a suitable partition

- Idea: use the same algorithm as for DFA, but with new notions of unstable block and block splitting.
- We must guarantee:

  after termination, states of a block recognize the same language

  or, equivalently

  after termination, states recognizing different languages belong to different blocks

**Key observation:**

If $L(q_1) \neq L(q_2)$ then either

- one of $q_1, q_2$ is final and the other non-final, or
- one of $q_1, q_2$, say $q_1$, has a transition $q_1 \xrightarrow{a} q_1'$ such that every $a$-transition $q_2 \xrightarrow{a} q_2'$ satisfies: $L(q_1') \neq L(q_2')$ .

This suggests the following definition:

**Definition:** Let $B, B'$ blocks of a partition P, and let $a \in \Sigma$ . The pair $(a, B')$ splits B if there are states $q_1, q_2 \in B$ such that

$$\delta(q_1, a) \cap B' = \emptyset \quad \text{and} \quad \delta(q_2, a) \cap B' \neq \emptyset$$

The result of the split is the partition

$$Ref_P^{NFA}[B, a, B] = (P \setminus \{B\}) \cup \{B_0, B_1\}$$

where

$$B_0 = \{q \in B \mid \delta(q, a) \cap B' = \emptyset\}$$
$$B_1 = \{q \in B \mid \delta(q, a) \cap B' \neq \emptyset\}$$

A partition is unstable if there are $B, a, B'$ such that $(a, B')$ splits $B$ , otherwise it is stable.

$CSR(A)$

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

**Output:** The partition $CSR$.

**if** $F = \emptyset$ or $Q \setminus F = \emptyset$ **then return** $\{Q\}$

**else** $P \leftarrow \{F, Q \setminus F\}$

**while** $P$ is unstable **do**

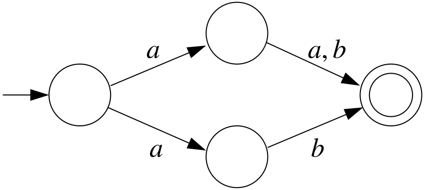
  pick $B, B' \in P$ and $a \in \Sigma$ such that $(a, B')$ splits $B$

  $P \leftarrow Ref_P^{NFA}[B, a, B']$

**return** $P$

It is not hard to see that the construction given above results in an NFA which is equivalent to the original NFA.

However:
**The result might not be minimal:**




or

**The result is finer than the language partition:**