2 Automata

[Linz, Chapter 1, p.26 –, Chapter 2]

Definition: An **automaton**, A, is an **abstract model** of a digital computer.

It contains:

- An **input file** (over a given alphabet)
- (temporary) **storage device**: a set of cells, each holding 1 symbol (maybe)
- **control** unit, in one of a number of **internal states**

Discrete time frame:

At any moment, A has a **configuration**
(**state**, **input symbol**, **storage info**)

Transition function:

configurations $\rightarrow$ **configurations** (moves)

Our **automata** are **special simple** versions of these.

They may be **deterministic** or **nondeterministic**.

Deterministic: each move is **uniquely** determined.

Acceptor A

Given an input string, A makes a series of moves, and **if it halts**, its output is ”**Yes**” or ”**No**”

(i.e. 0 or 1).

i.e. it **accepts** or **rejects** the input string.

A language, $L \subseteq \Sigma^*$, is **accepted** by A if

$L = \{u \in \Sigma^* \mid A \textbf{ accepts } u\}$.

Definition: **Deterministic finite acceptor (dfa)** is a quintuple

$$M = (Q,\ \Sigma,\ \delta,\ q_0,\ F) \qquad \text{[Linz, § 2.1]}$$

where

Q is a finite set of **states**

Σ is a finite **input alphabet** (input symbols)

δ: $Q \times \Sigma \to Q$ is the **transition function**

q_0 is the **initial state**

$F \subseteq Q$ is the set of **final states**

dfa's can be represented by **transition tables** or **transition graphs**

Example: [Linz, Example 2.1]

$M = (Q,\ \Sigma,\ \delta,\ q_0,\ F)$, where

$Q = \{q_0, q_1, q_2\}$

$\Sigma = \{0, 1\}$

$F = \{q_1\}$

and δ is given by:

	0	1
$\longrightarrow q_0$	q_0	q_1
$F : q_1$	q_0	q_2
q_2	q_2	q_1

Corresponding **transition graph**: See Linz, Fig. 2.1.

Note:

In transition graph,

show **initial state** by '$\longrightarrow$'

and **final state** by **double circle**.

Extended transition function

Given dfa: $M = (Q,\ \Sigma,\ \delta,\ q_0,\ F)$

Define the **extended transition function**

$$\delta^* : Q \times \Sigma^* \to Q$$

$\delta^*(q, u) =$ **state** of M starting in q after reading u.

E.g. $\delta^*(q, abc) = \underline{\delta(\delta(\delta(q, a), b), c)}$

Recursive definition of δ^*:

$\delta^*(q, \lambda) = \underline{q}$

$\delta^*(q, ua) = \underline{\delta(\delta^*(q, u), a)}$

This definition is by **structural recursion** on $u \in \Sigma^*$

or by **recursion** on $|u|$ (see p.).

Now we can define

Definition: The language accepted by a dfa, $M = (Q,\ \Sigma,\ \delta,\ q_0,\ F)$ is:

$$L(M) = \{u \in \Sigma^* \mid \delta^*(q_0, u) \in F\}$$

This connects δ^* with the transition graph of M:

Theorem [L, Theorem 2.1]

If $M = (Q,\ \Sigma,\ \delta,\ q_0,\ F)$ is a dfa with transition graph, G_M, then:

$\forall q_i, q_j \in Q, u \in \Sigma^*$,

$\delta^*(q_i, u) = q_j \iff$ there is a **path** in G_n with label u from q_i to q_j.
[Linz uses "**walk**" instead of "**path**"]

Proof By structural induction on u.

Examples **of transition graphs for dfa's**:

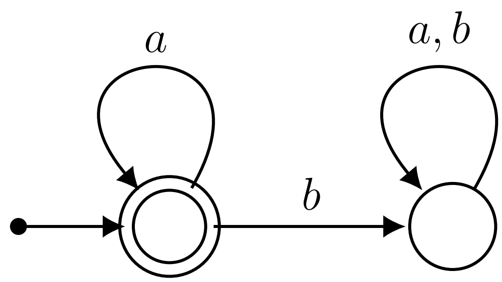
see Linz:
Example 2.2: $L = \{a^n b \mid n \geq 0\}$
Example 2.3: $L = \{abu \mid u \in \Sigma^*\}$ for $\Sigma = \{a, b\}$

Examples of dfa's

Assume $\Sigma = \{a, b\}$.

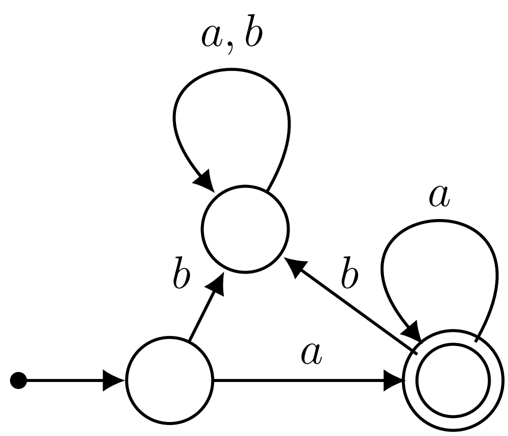
Construct dfa's for the following languages over Σ:

(1) The set of all words containing only 'a' $= \{a\}^*$:



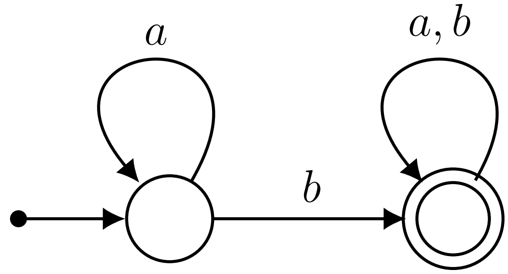
The second state is a **trap state** (not a final state).

(2) The set of all **non-empty** words containing only 'a' $= \{a\}^+$:

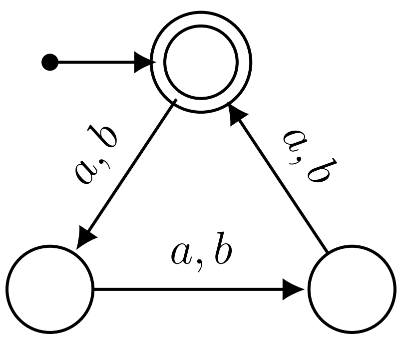


(3) The set of all words containing at least one '***b***':

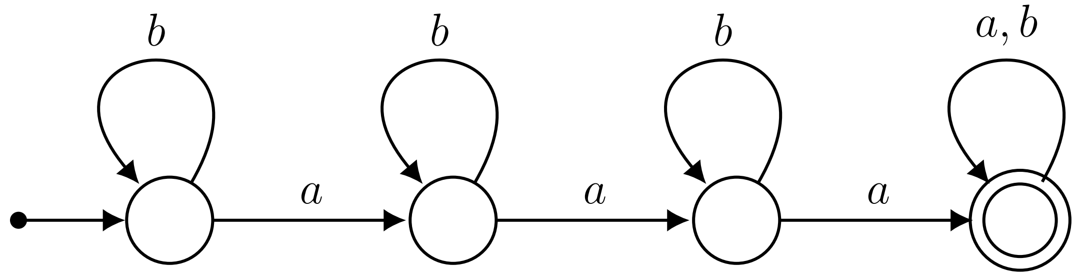
This is the **complement** of language (1)! That means we can interchange the final and non-final states of (1) to get the appropriate dfa:



(4) The set of all words of lengths that are multiples of 3:



(5) The set of words with at least 3 '***a***''s:



(6) The set of words containing '***aaa***':