Huiswerk

- huiswerkopgave 3. . .

Assumptions about a human computer working with a pencil and paper:

1. The only things written on the paper are symbols from some fixed finite alphabet;
2. Each step taken by the computer depends only on the symbol he is currently examining and on his "state of mind" at the time;
3. Although his state of mind may change as a result of his examining different symbols, only a finite number of distinct states of mind are possible.

Actions of a human computer on a sheet of paper:

1. Examining an individual symbol on the paper;
2. Erasing a symbol or replacing it by another;
3. Transferring attention from one symbol to another nearby symbol.

Turing machine

Turing machine has a finite alphabet of symbols.
    (actually two alphabets. . .)
Turing machine has a finite number of states.
Turing machine has a *tape*
    for reading input,
    as workspace,
    and for writing output (if applicable).
Tape is linear, instead of 2-dimensional.
Tape has a left end and is potentially infinite to the right.
Tape is marked off into squares, each of which can hold one symbol.
Tape head is centered on one square of the tape for reading and writing.

A move of a Turing machine consists of:

1. Changing from the current state to another, possibly different state;
2. Replacing the symbol in the current square by another, possibly different symbol;
3. Leaving the tape head on the current square, or moving it one square to the right, or moving it one square to the left if it is not already on the leftmost square.

Just like FA and PDA, Turing machine

- may be used to accept a language
- has a finite number of states

Just like FA, but unlike PDA

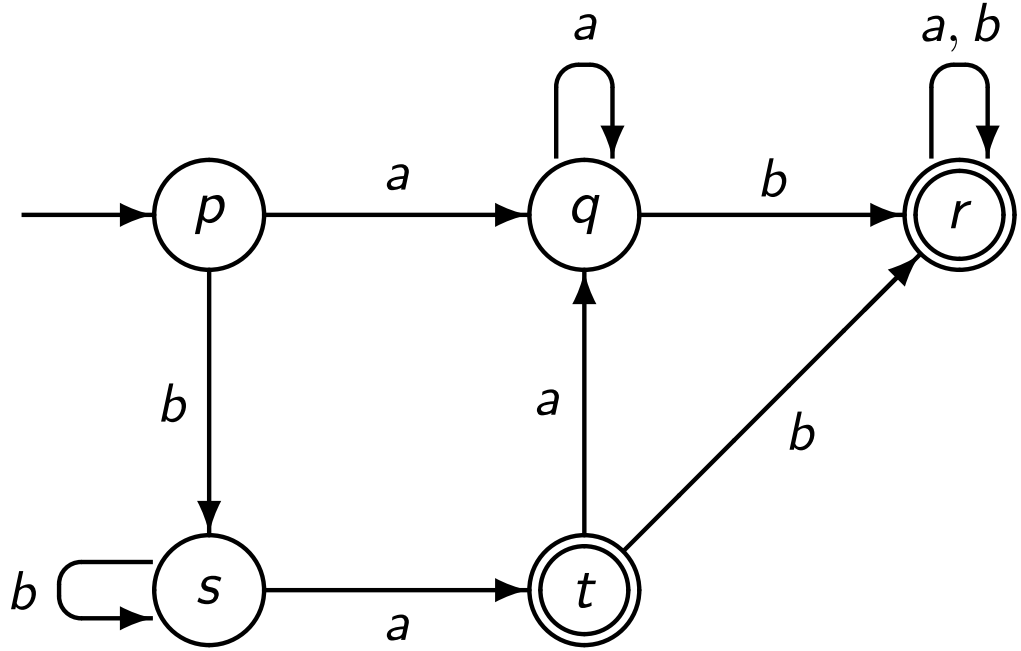
- by default TM is deterministic

Unlike FA and PDA, Turing machine

- may also be used to compute a function *
- is not restricted to reading input left-to-right *
- does not have to read all input *
- does not have a set of accepting states, but has two *halt* states: one for acceptance and one for rejection (in case of computing a function, . . .)
- might not decide to halt

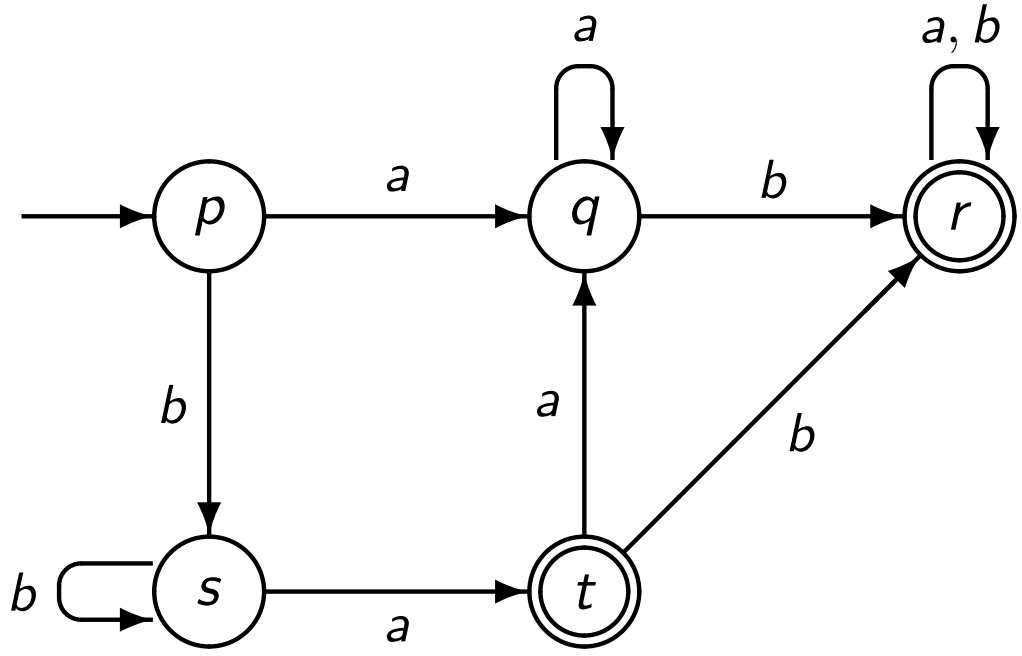
* = just like human computer

Example.
An FA Accepting $L = \ldots$



Example.

An FA Accepting $L = \{a, b\}^*\{ab\}\{a, b\}^* \cup \{a, b\}^*\{ba\}$



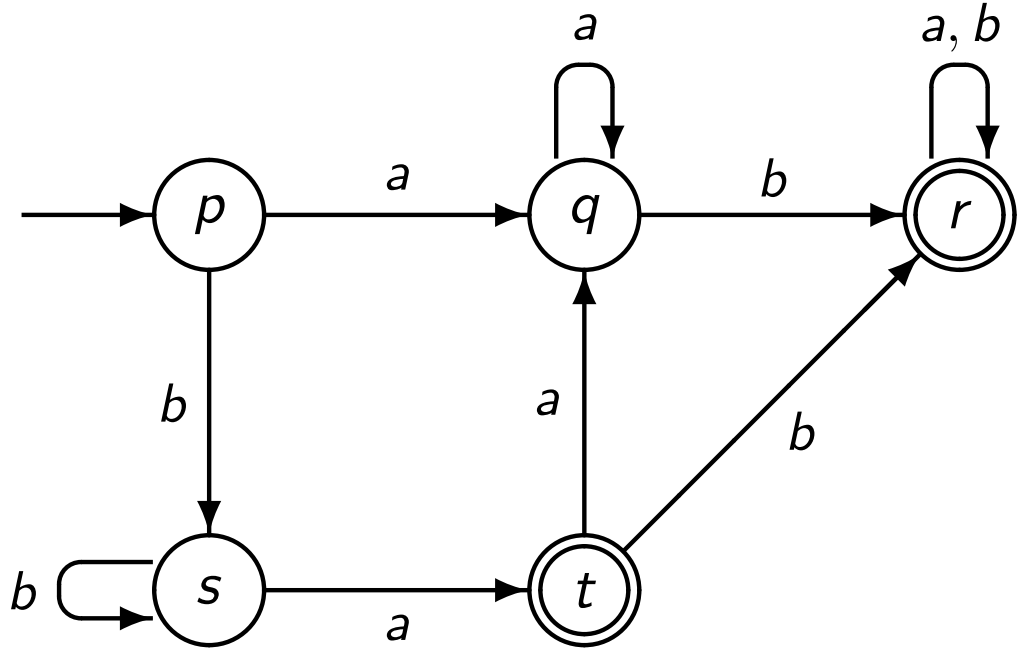
Why two accepting states?

7.2. Turing Machines as Language Acceptors

Example 7.3. A TM Accepting a Regular Language

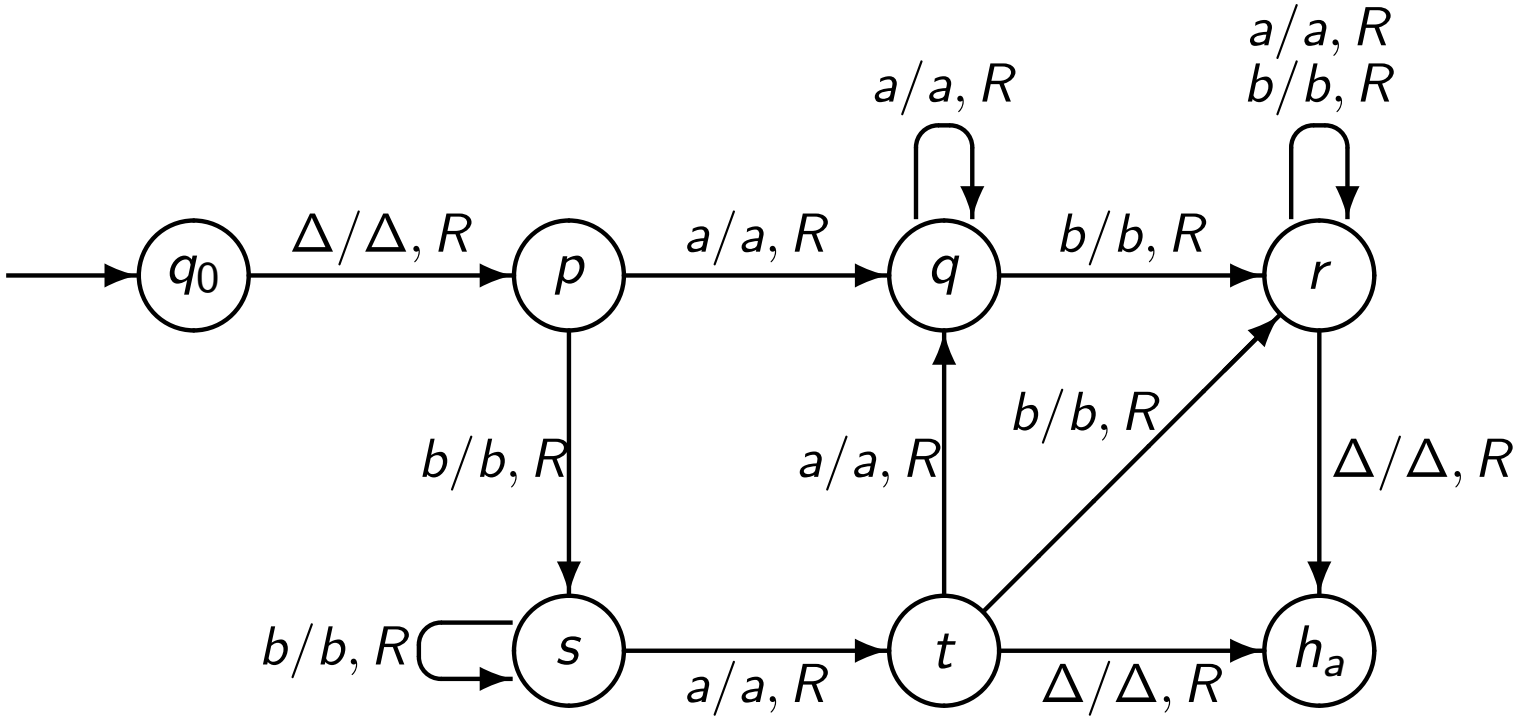
$L = \{a, b\}^*\{ab\}\{a, b\}^* \cup \{a, b\}^*\{ba\}$

First a finite automaton:



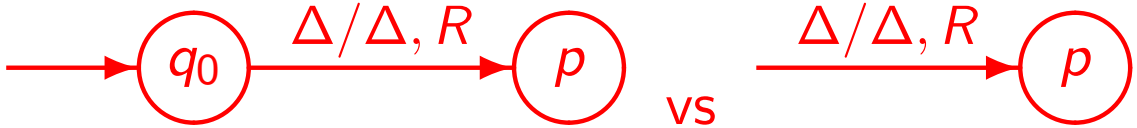
Example 7.3. A TM Accepting a Regular Language

$L = \{a, b\}^*\{ab\}\{a, b\}^* \cup \{a, b\}^*\{ba\}$



Δ vs Λ

Λ vs $\{\Lambda\}$ vs $\emptyset$



Example 7.3. A TM Accepting a Regular Language

$L = \{a, b\}^*\{ab\}\{a, b\}^* \cup \{a, b\}^*\{ba\}$

First a finite automaton, then a Turing machine

This conversion works in general for FAs.

As a result,

- only moves to the right,
- no modifications of symbols on tape,
- no moves to the reject state, but ...

In this case,

- we could modify TM, so that it does not always read entire input.

Example 7.5. A TM Accepting $XX = \{xx \mid x \in \{a, b\}^*\}$

$$\underline{\triangle} aabaab$$

$$\ldots$$

Example 7.5. A TM Accepting $XX = \{xx \mid x \in \{a, b\}^*\}$

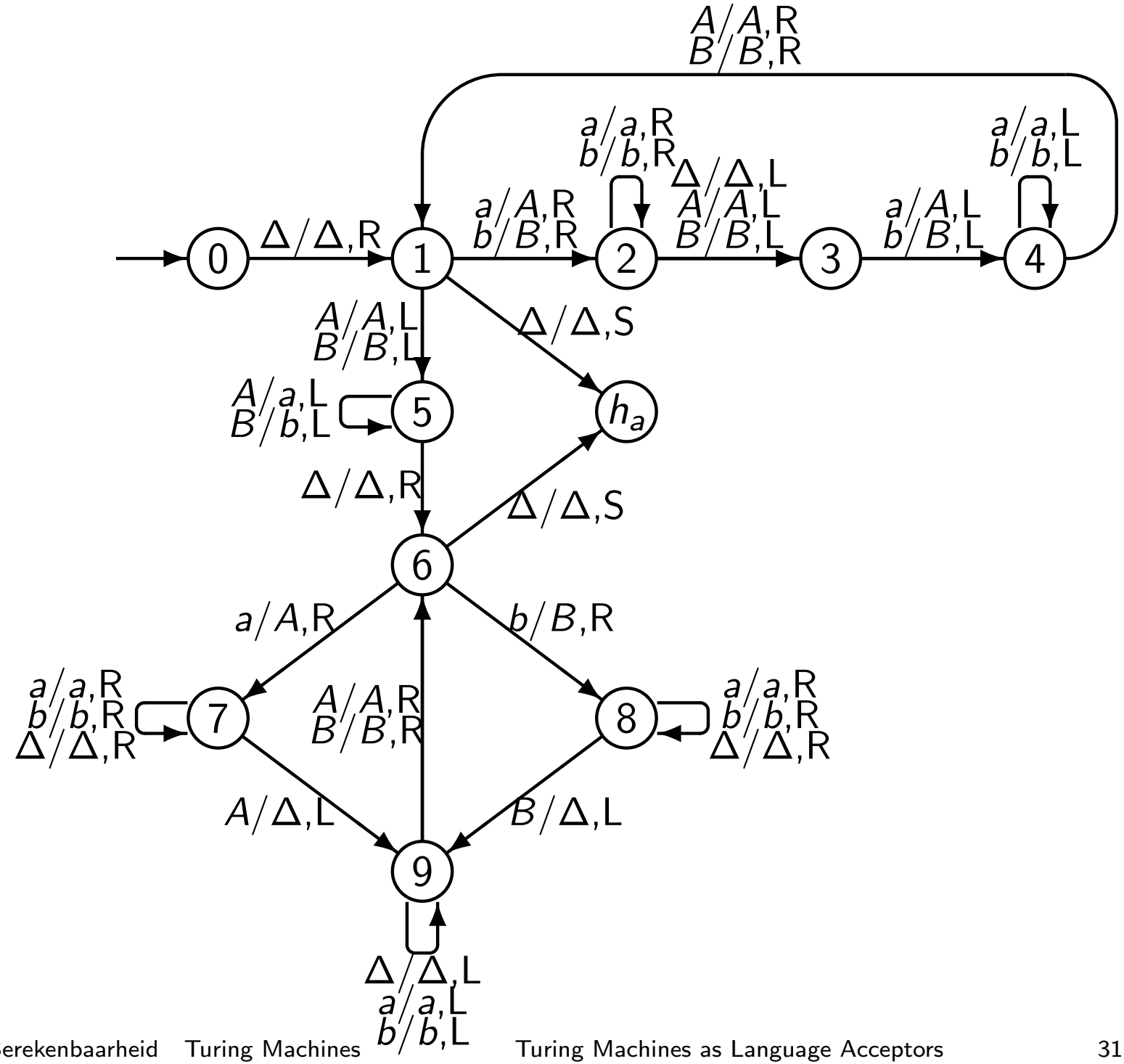
$$\underline{\Delta}\, a\, a\, b\, a\, a\, b$$
$$\Delta A\, a\, b\, a\, a\, b$$
$$\Delta A\, a\, b\, a\, a B$$
$$\Delta A A\, b\, a\, a B$$
$$\Delta A A\, b\, a A B$$
$$\Delta A A B\, a A B$$
$$\Delta A A B A A B$$
$$\ldots$$

Example 7.5. A TM Accepting $XX = \{xx \mid x \in \{a, b\}^*\}$

$$\underline{\Delta}\, a\, a\, b\, a\, a\, b$$
$$\Delta A\, a\, b\, a\, a\, b$$
$$\Delta A\, a\, b\, a\, a B$$
$$\Delta A A\, b\, a\, a B$$
$$\Delta A A\, b\, a A B$$
$$\Delta A A B\, a A B$$
$$\Delta A A B A A B$$
$$\Delta\, a\, a\, b A A B$$
$$\Delta A\, a\, b A A B$$
$$\ldots$$

Example 7.5. A TM Accepting $XX = \{xx \mid x \in \{a, b\}^*\}$

$$\underline{\Delta}\, a\, a\, b\, a\, a\, b$$
$$\Delta A\, a\, b\, a\, a\, b$$
$$\Delta A\, a\, b\, a\, a\, B$$
$$\Delta AA\, b\, a\, a\, B$$
$$\Delta AA\, b\, a\, A\, B$$
$$\Delta AAB\, a\, A\, B$$
$$\Delta AAB\, A\, A\, B$$
$$\Delta\, a\, a\, b\, A\, A\, B$$
$$\Delta A\, a\, b\, A\, A\, B$$
$$\Delta A\, a\, b\, \Delta\, A\, B$$
$$\Delta AA\, b\, \Delta\, A\, B$$
$$\Delta AA\, b\, \Delta\Delta\, B$$
$$\Delta AAB\, \Delta\Delta\, B$$
$$\Delta AAB\, \Delta\Delta\Delta$$


A/A,R
B/B,R
a/a,R
b/b,R
Δ/Δ,L
a/a,L
b/b,L
a/A,R
b/B,R
A/A,L
B/B,L
a/A,L
b/B,L
0
Δ/Δ,R
1
2
3
4
A/A,L
B/B,L
Δ/Δ,S
A/a,L
B/b,L
5
h_a
Δ/Δ,R
Δ/Δ,S
6
a/A,R
b/B,R
a/a,R
b/b,R
Δ/Δ,R
7
A/A,R
B/B,R
8
a/a,R
b/b,R
Δ/Δ,R
A/Δ,L
B/Δ,L
9
Δ/Δ,L
a/a,L
b/b,L

Example 7.5. A TM Accepting $XX = \{xx \mid x \in \{a, b\}^*\}$

$$
\begin{array}{l}
\underline{\Delta}\, a\, a\, b\, a\, a\, b \\
\Delta A\, a\, b\, a\, a\, b \\
\Delta A\, a\, b\, a\, a\, B \\
\Delta A A\, b\, a\, a\, B \\
\Delta A A\, b\, a\, A\, B \\
\Delta A A B\, a\, A\, B \\
\Delta A A B\, A\, A\, B \\
\Delta\, a\, a\, b\, A\, A\, B \\
\Delta A\, a\, b\, A\, A\, B \\
\Delta A\, a\, b\, \Delta\, A\, B \\
\Delta A A\, b\, \Delta\, A\, B \\
\Delta A A\, b\, \Delta \Delta\, B \\
\Delta A A B\, \Delta \Delta\, B \\
\Delta A A B\, \Delta \Delta \Delta
\end{array}
$$

What if input $w \notin XX$?

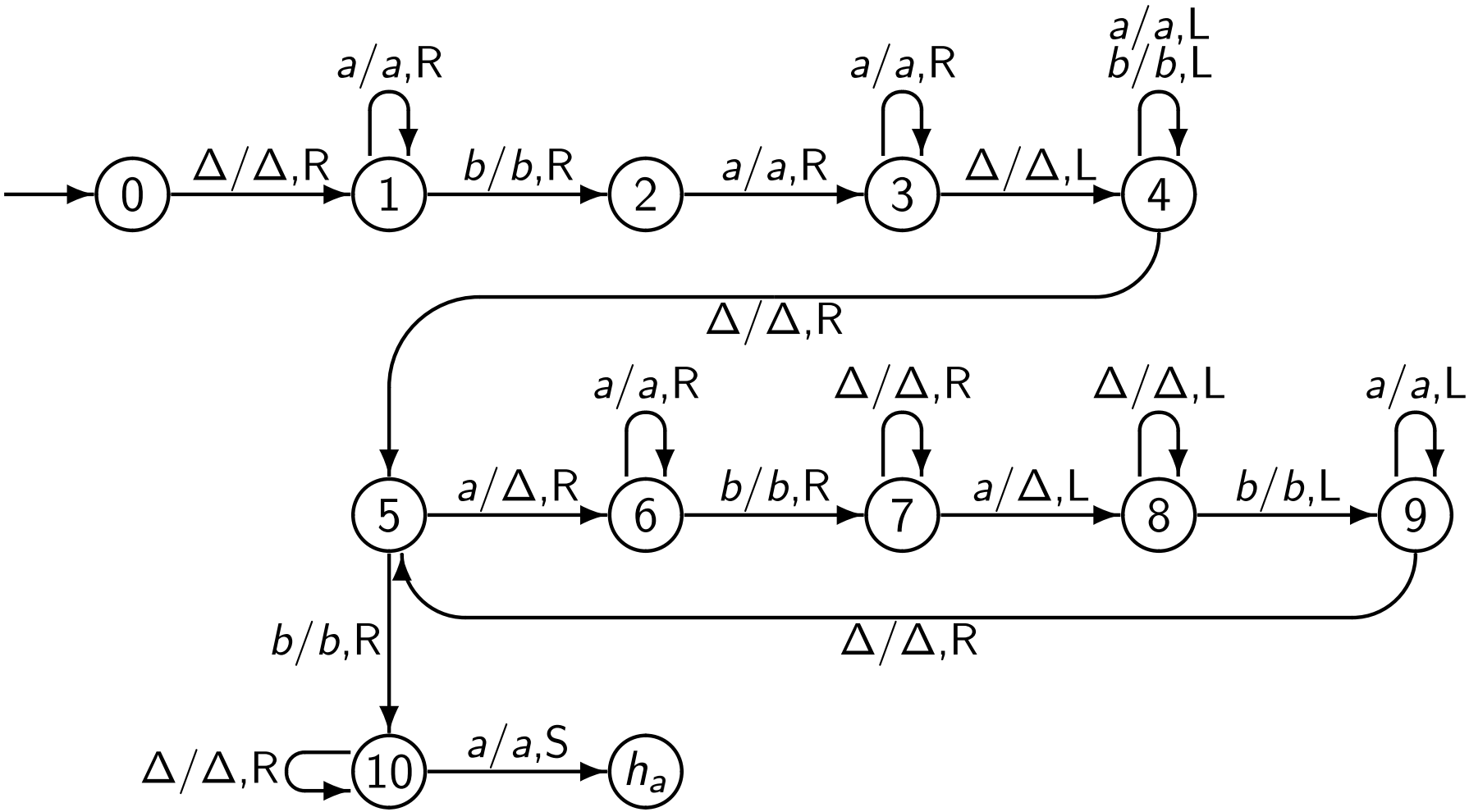
T decides on membership of XX

Example 7.7. Accepting $L = \{a^i b a^j \mid 0 \leq i < j\}$

Example 7.7. Accepting $L = \{a^i b a^j \mid 0 \leq i < j\}$

$$\underline{\Delta}\, a\ a\ b\ a\ a\ a\ a$$
$$\Delta\Delta\, a\ b\ a\ a\ a\ a$$
$$\Delta\Delta\, a\ b\,\Delta\ a\ a\ a$$
$$\Delta\Delta\Delta\, b\,\Delta\ a\ a\ a$$
$$\Delta\Delta\Delta\, b\,\Delta\Delta\ a\ a$$

Example 7.7. Accepting $L = \{a^i b a^j \mid 0 \leq i < j\}$



What if $x \notin L$?

Example 7.7. Accepting $L = \{a^i b a^j \mid 0 \leq i < j\}$
To illustrate that a Turing machine T may run forever for an input that is not in $L(T)$. No problem!
No problem?

Part of an exercise from exercise class 9

Exercise 7.4.

For each of the following languages, draw a transition diagram for a Turing machine that accepts that language.

a. ...

b. $\{a^i b^j \mid i < j\}$

c. ...

d. ...

Definition 2.11. A Finite Automaton
A *finite automaton* (FA) is a 5-tuple $(Q, \Sigma, q_0, A, \delta)$, where ...

Definition 5.1. A Pushdown Automaton
A *pushdown automaton* (PDA)
is a 7-tuple $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$, where ...

Definition 7.1. Turing machines
A Turing machine (TM) is ...

Turing machine

Turing machine has a finite alphabet of symbols.

(actually two alphabets. . .)

Turing machine has a finite number of states.

7.1. A General Model of Computation

Definition 7.1. Turing machines
A Turing machine (TM) is a 5-tuple $T = (Q, \Sigma, \Gamma, q_0, \delta)$, where
Q is a finite set of states. The two *halt* states h_a and h_r are not elements of Q.
Σ, the input alphabet, and Γ, the tape alphabet, are both finite sets, with $\Sigma \subseteq \Gamma$. The *blank* symbol Δ is not an element of Γ.
q_0, the initial state, is an element of Q.
δ is the transition function: ...

Assumptions about a human computer working with a pencil and paper:

1. ...
2. Each step taken by the computer depends only on the symbol he is currently examining and on his "state of mind" at the time;
3. ...

A move of a Turing machine consists of:

1. Changing from the current state to another, possibly different state;
2. Replacing the symbol in the current square by another, possibly different symbol;
3. Leaving the tape head on the current square, or moving it one square to the right, or moving it one square to the left if it is not already on the leftmost square.

$$\delta(p, X) = (q, Y, D)$$

$$f(n) = n!$$

$$f(x, y) = x * y$$

Definition 7.1. Turing machines
A Turing machine (TM) is a 5-tuple $T = (Q, \Sigma, \Gamma, q_0, \delta)$, where
Q is a finite set of states. The two *halt* states h_a and h_r are not elements of Q.
Σ, the input alphabet, and Γ, the tape alphabet, are both finite sets, with $\Sigma \subseteq \Gamma$. The *blank* symbol Δ is not an element of Γ.
q_0, the initial state, is an element of Q.
δ is the transition function:

$$\delta : Q \times (\Gamma \cup \{\Delta\}) \to (Q \cup \{h_a, h_r\}) \times (\Gamma \cup \{\Delta\}) \times \{R, L, S\}$$

If q is h_a or h_r, the move causes T to halt
What if $D = L$ and T is on square 0?

Normally, TM starts with

- input string starting in square 1 and all other squares blank,
- and its tape head on square 0.

Tape always contains finite number of nonblanks.

Notation:
configuration. . .

Notation:

description of tape contents: $x\underline{\sigma}y$ or $x\underline{y}$

$x\underline{y} = x\underline{y}\Delta = x\underline{y}\Delta\Delta$

if $y = \Lambda$, then $x\underline{\Delta}$

Notation:

description of tape contents: $x\underline{\sigma}y$ or $x\underline{y}$

$x\underline{y} = x\underline{y}\Delta = x\underline{y}\Delta\Delta$

if $y = \Lambda$, then $x\underline{\Delta}$

configuration $xqy = xqy\Delta = xqy\Delta\Delta$

if $y = \Lambda$, then $xq\Delta$

$(Q \cup \{h_a, h_r\}) \cap (\Gamma \cup \{\Delta\}) = \emptyset$

Notation:

description of tape contents: $x\underline{\sigma}y$ or $x\underline{y}$

$x\underline{y} = x\underline{y}\Delta = x\underline{y}\Delta\Delta$

if $y = \Lambda$, then $x\underline{\Delta}$

configuration $xqy = xqy\Delta = xqy\Delta\Delta$

if $y = \Lambda$, then $xq\Delta$

move: $xqy \vdash_T zrw$ $xqy \vdash_T^* zrw$

$xqy \vdash zrw$ $xqy \vdash^* zrw$

example: configuration $aabqa\Delta a$ and $\delta(q, a) = (r, \Delta, L)$ ...

initial configuration corresponding to input x: ...

Notation:

description of tape contents: $x\underline{\sigma}y$ or $x\underline{y}$

$x\underline{y} = x\underline{y}\Delta = x\underline{y}\Delta\Delta$

if $y = \Lambda$, then $x\underline{\Delta}$

configuration $xqy = xqy\Delta = xqy\Delta\Delta$

if $y = \Lambda$, then $xq\Delta$

move: $xqy \vdash_T zrw$ $xqy \vdash_T^* zrw$

$xqy \vdash zrw$ $xqy \vdash^* zrw$

example: configuration $aabqa\Delta a$ and $\delta(q, a) = (r, \Delta, L)$

$aabqa\Delta a \vdash aarb\Delta\Delta a$

initial configuration corresponding to input x: $q_0\Delta x$