Fundamentele Informatica 1 (I&E)

najaar 2014

`http://www.liacs.leidenuniv.nl/~vlietrvan1/fi1ie/`

Rudy van Vliet
rvvliet(at)liacs(dot)nl

college 12, woensdag 3 december 2014

7. Turing Machines

7. Turing Machines

reg. languages	FA	reg. grammar	reg. expression
determ. cf. languages	DPDA		
cf. languages	PDA	cf. grammar	
re. languages	TM	unrestr. grammar	

7.1. A General Model of Computation

$AnBnCn = \{a^i b^i c^i \mid i \geq 0\}$

$L = \{xcx \mid x \in \{a, b\}^*\}$

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

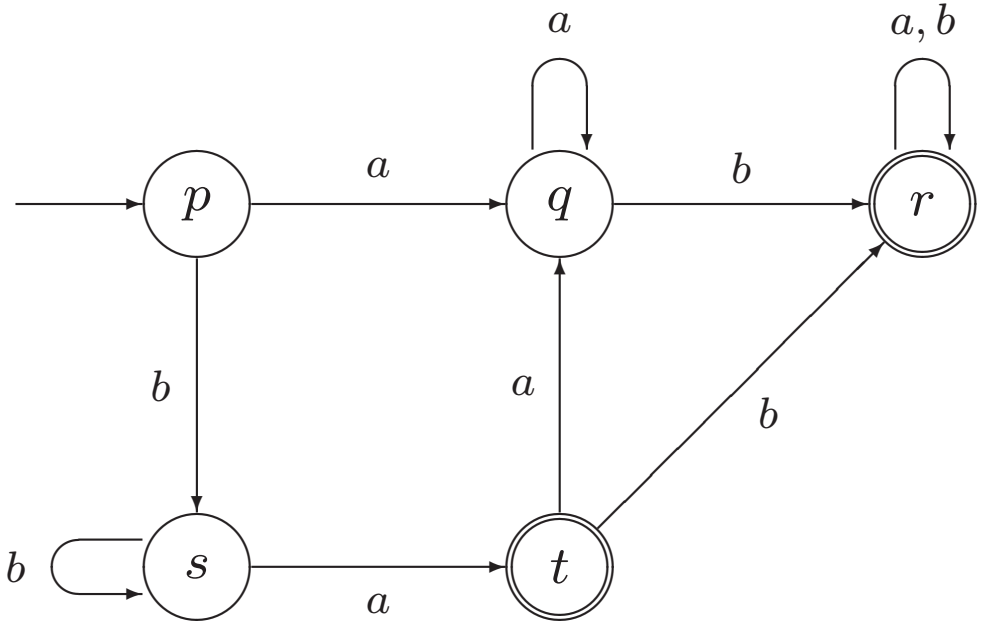
Unlike FA and PDA, Turing machine

- 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, . . .)
- may decide not to halt

* = just like human computer

Example.

An FA Accepting $L = \ldots$

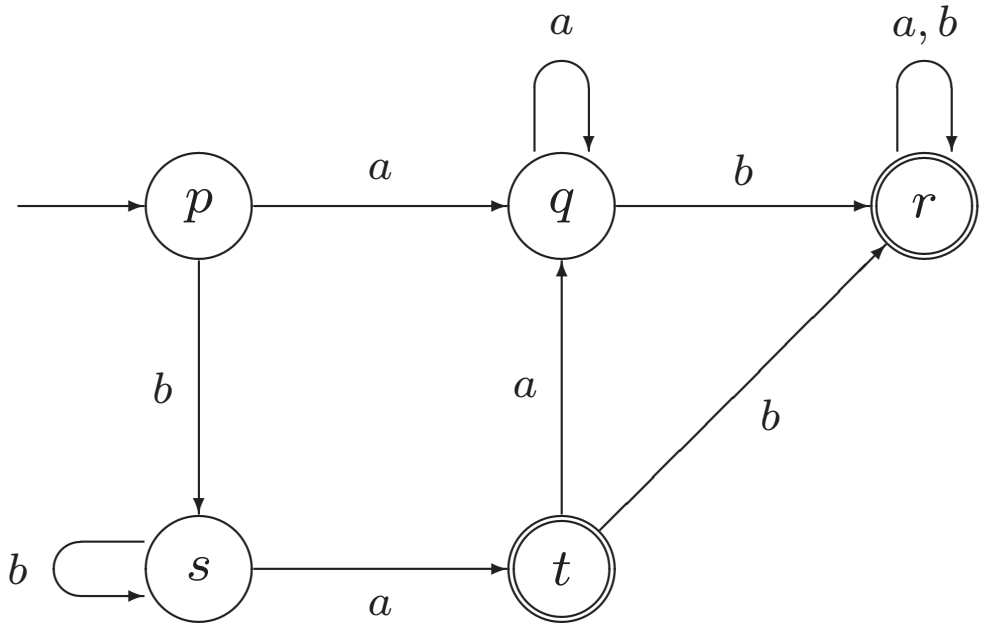


7.2. Turing Machines as Language Acceptors

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

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

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 $XX = \{xx \mid x \in \{a, b\}^*\}$

$$
\begin{array}{c}
\underline{\triangle}\, a\ a\ b\ a\ a\ b \\
\triangle A\, a\ b\ a\ a\ b \\
\triangle A\, a\ b\ a\ a\, B \\
\triangle AA\, b\ a\ a\, B \\
\triangle AA\, b\ a\, AB \\
\triangle AAB\, a\, AB \\
\triangle AABAAB \\
\cdots
\end{array}
$$

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

$$
\begin{array}{c}
\underline{\triangle}\, a\ a\ b\ a\ a\ b \\
\triangle A\, a\ b\ a\ a\ b \\
\triangle A\, a\ b\ a\ a\, B \\
\triangle A A\ b\ a\ a\, B \\
\triangle A A\ b\ a\, A B \\
\triangle A A B\, a\, A B \\
\triangle A A B A A B \\
\triangle\, a\ a\ b\, A A B \\
\triangle A\, a\ b\, A A B \\
\ldots
\end{array}
$$

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

$$
\begin{array}{l}
\underline{\triangle}\, a\ a\ b\ a\ a\ b \\
\triangle A\, a\ b\ a\ a\ b \\
\triangle A\, a\ b\ a\ a\ B \\
\triangle AA\, b\ a\ a\ B \\
\triangle AA\, b\ a\ A\ B \\
\triangle AAB\, a\ A\ B \\
\triangle AAB\, A\ A\ B \\
\triangle a\ a\ b\ A\ A\ B \\
\triangle A\, a\ b\ A\ A\ B \\
\triangle A\, a\ b\ \triangle\ A\ B \\
\triangle AA\, b\ \triangle\ A\ B \\
\triangle AA\, b\ \triangle\triangle\ B \\
\triangle AAB\triangle\triangle\ B \\
\triangle AAB\triangle\triangle\triangle
\end{array}
$$

Turing machine

Turing machine has a finite alphabet of symbols.

(actually two alphabets. . .)

Turing machine has a finite number of states.

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.

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:

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

Interpretation of

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

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 non-blanks.

Notation:

configuration. . .

Notation:

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

Notation:

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

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

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

initial configuration corresponding to input x: ...

Notation:

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

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

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

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

This notation does not have to be used at the exam.
This slide is meant only to understand Definition 7.2 and Definition 7.9.

Definition 7.2. Acceptance by a TM

If $T = (Q, \Sigma, \Gamma, q_0, \delta)$ is a TM and $x \in \Sigma^*$,
x is accepted by T if

$$q_0 \Delta x \vdash_T^* w h_a y$$

for some strings $w, y \in (\Gamma \cup \{\Delta\})^*$
(i.e., if, starting in the initial configuration corresponding to input x, T eventually halts in the accepting state).

N.B.: sequence of moves leading to h_a is unique

A language $L \subseteq \Sigma^*$ is accepted by T if $L = L(T)$, where

$$L(T) = \{x \in \Sigma^* \mid \ x \text{ is accepted by } T \ \}$$

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!

Exercise.

Draw a transition diagram for a Turing machine that accepts

$AnBnCn = \{a^i b^i c^i \mid i \geq 0\}$

En verder. . .

Maandag 8 december 2014:
11:15: inleveren huiswerkopgave
vervolgens: laatste college

Vrijdag 19 december 2014, 13:45–. . . :
vragenuur (in Leiden)

Dinsdag 23 december 2014, 14:00–17:00:
tentamen