Fundamentele Informatica 3

voorjaar 2012

`http://www.liacs.nl/home/rvvliet/fi3/`

Rudy van Vliet
kamer 124 Snellius, tel. 071-527 5777
rvvliet(at)liacs.nl

college 6, 12 maart 2012

6. Context-Free and Non-Context-Free Languages

6.2. Intersections and Complements of CFLs

7. Turing Machines

6.2. Intersections and Complements of CFLs

Excercise.

Show that if L is accepted by a DPDA M, then there is a PDA M' accepting the language $\{x\#y \mid\ x \in L$ and $xy \in L\}$.
(The symbol # is assumed not to be in any of the strings of L.)

Excercise 5.20.

Show that if L is accepted by a DPDA M, then there is a DPDA M' accepting the language $\{x\#y \mid \ x \in L \text{ and } xy \in L\}$.
(The symbol # is assumed not to be in any of the strings of L.)

Exercise. (from last week)

Let M be a DPDA.

Construct a DPDA M' such that $L(M') = L(M)$
and M' has no Λ-transitions from an accepting state.

Theorem 6.13.

If L_1 is a context-free language and L_2 is a regular language, then $L_1 \cap L_2$ is a CFL.

Exercise 6.8.

Show that if L_1 is a DCFL and L_2 is regular, then $L_1 \cap L_2$ is a DCFL.

Exercise 6.21. Use Exercises 5.20 and 6.8 to show that the following languages are not DCFLs.
This technique is used in Floyd and Beigel (1994), where the language in Exercise 5.20 is referred to as Double-Duty(L).

a. *Pal*, the language of palindromes over $\{a, b\}$. (Hint: Consider the regular language corresponding to $a^+b^+a^+\#b^+a^+$.)

b. $\{x \in \{a, b\}^* \mid \ n_b(x) = n_a(x)$ or $n_b(x) = 2n_a(x)\}$

reg. languages	reg. grammar	FA	reg. expression
cf. languages	cf. grammar	PDA	
		TM	

7. Turing Machines

7.1. A General Model of Computation

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

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

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, then a Turing machine

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

Huiswerkopgave 2. Toepassen van stelling 5.29.
Inleverdatum dinsdag 27 maart 2012, 13:45 uur

Theorem 5.29.

If $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$ is a pushdown automaton accepting L by empty stack,
then there is a context-free grammar G such that $L = L(G)$.

Proof.

We define $G = (V, \Sigma, S, P)$ as follows:
V contains S as well as all possible variables of the form $[p, A, q]$, where $A \in \Gamma$ and $p, q \in Q$.

etc.