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 1, 6 februari 2012

Herhaling onderwerpen FI2

5. Pushdown Automata

5.1. Definitions and Examples

- hoorcollege: maandag 6 feb - 14 mei, zaal 403, 13:45–15:30
werkcollege: dinsdag 7 feb - 15 mei, zaal 403, 13:45–15:30
(Wouter Duivesteijn)

- boek: John C. Martin, Introduction to Languages and the Theory of Computation, 4th edition
Er komt verwijslijst naar 3rd edition

- tentamens: maandag 11 juni 2012, 10:00–13:00
maandag 20 augustus 2012, 10:00–13:00

- Drie huiswerkopgaven (individueel)
Niet verplicht, maar . . .
eindcijfer $=$ 0.9 $\times$ tentamencijfer $+$ cijferhuiswerkopgaven

- 6 EC

Fundamentele Informatica 2

2.1. Finite Automata

2.4. The Pumping Lemma

3.1. Regular Languages and Regular Expressions

3.2. Nondeterministic Finite Automata

3.3. The Nondeterminism in an NFA Can Be Eliminated

3.4/3.5. Kleene's Theorem

Fundamentele Informatica 2

4.2. Context-Free Grammars

4.3. Regular Languages and Regular Grammars

4.4. Derivation Trees

6.1. The Pumping Lemma for Context-Free Languages

5. Pushdown Automata

just like FA, PDA accepts strings / language

just like FA, PDA has states

just like FA, PDA reads input one letter at a time

unlike FA, PDA has auxiliary memory: a stack

unlike FA, by default PDA is nondeterministic

unlike FA, by default Λ-transitions are allowed in PDA

Stack in PDA contains symbols from certain alphabet.

Usual stack operations: pop, top, push

Extra possiblity: replace top element X by string α

$\alpha = \Lambda$	pop
$\alpha = X$	top
$\alpha = YX$	push
$\alpha = \beta X$	push*
$\alpha = \ldots$	

Top element X is required to do a move!

Example 5.3. A PDA Accepting the Language *AnBn*

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

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

Q is a finite set of states.
Σ and Γ are finite sets, the *input* and *stack* alphabet.
q_0, the initial state, is an element of Q.
Z_0, the initial stack symbol, is an element of Γ.
A, the set of accepting states, is a subset of Q.
δ, the transition function, is a function from . . . to . . .

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

Q is a finite set of states.
Σ and Γ are finite sets, the *input* and *stack* alphabet.
q_0, the initial state, is an element of Q.
Z_0, the initial stack symbol, is an element of Γ.
A, the set of accepting states, is a subset of Q.
δ, the transition function, is a function from $Q \times (\Sigma \cup \{\Lambda\}) \times \Gamma$
to the set of finite subsets of $Q \times \Gamma^*$.

In principle, Z_0 may be removed from the stack,
but often it isn't.

Example 5.3. A PDA Accepting the Language *AnBn*

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

Transition Table vs Transition Diagram

Move Number	State	Input	Stack Symbol	Move(s)
1	q_0	a	Z_0	(q_1, aZ_0)
2	q_1	a	a	(q_1, aa)
3	q_1	b	a	(q_2, Λ)
4	q_2	b	a	(q_2, Λ)
5	q_2	Λ	Z_0	(q_3, Z_0)
	all other combinations			none

Notation:

configuration (q, x, α)

$(p, x, \alpha) \vdash_M (q, y, \beta)$

$(p, x, \alpha) \vdash_M^n (q, y, \beta)$ $(p, x, \alpha) \vdash_M^* (q, y, \beta)$

$(p, x, \alpha) \vdash (q, y, \beta)$ $(p, x, \alpha) \vdash^n (q, y, \beta)$ $(p, x, \alpha) \vdash^* (q, y, \beta)$

Definition 5.2. Acceptance by a PDA

If $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$ and $x \in \Sigma^*$,
the string x is accepted by M if

$$(q_0, x, Z_0) \vdash_M^* (q, \Lambda, \alpha)$$

for some $\alpha \in \Gamma^*$ and some $q \in A$.

A language $L \subseteq \Sigma^*$ is said to be accepted by M,
if L is precisely the set of strings accepted by M;
in this case, we write $L = L(M)$.

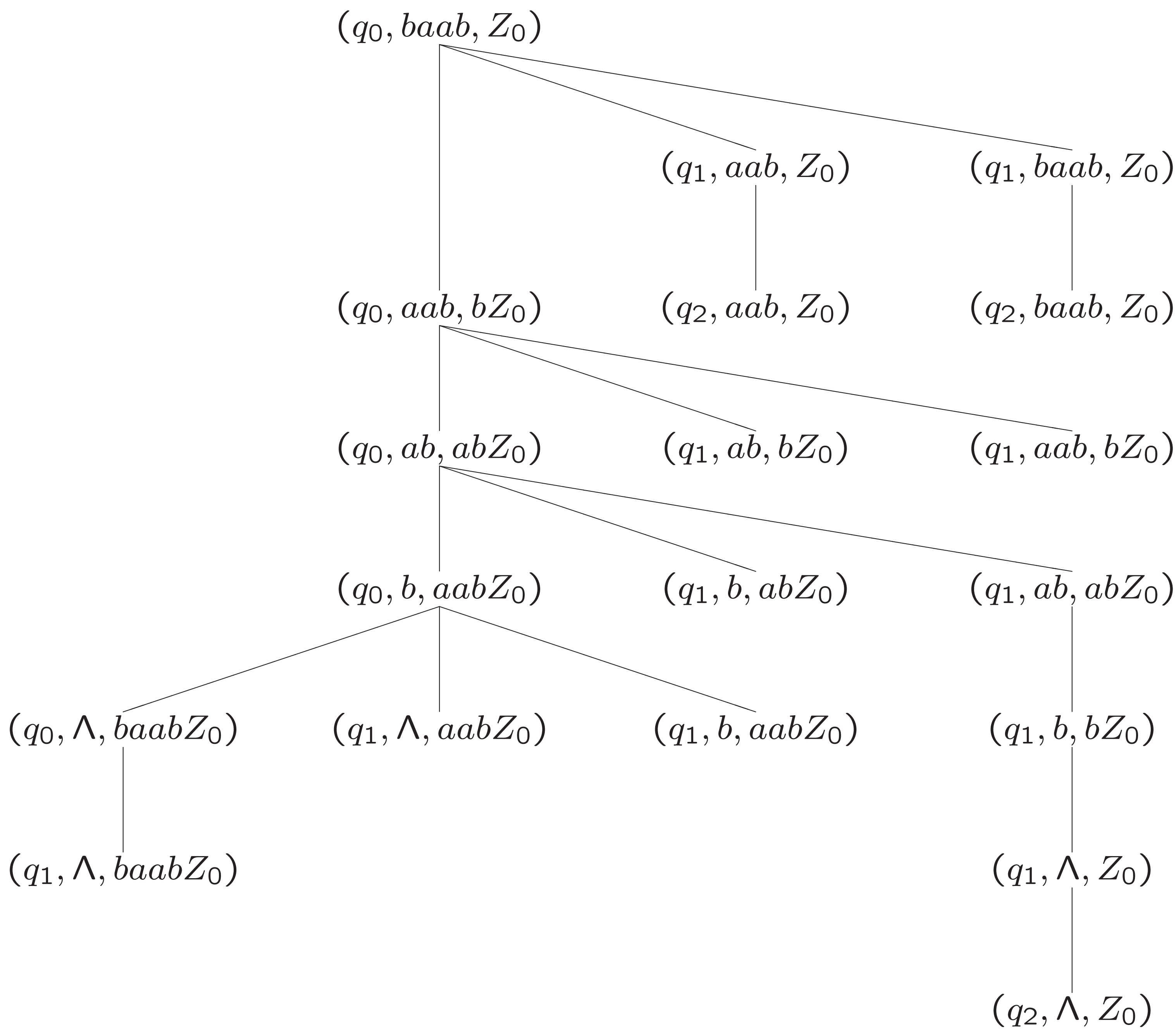
Sometimes a string accepted by M, or a language accepted by M, is said to be accepted *by final state*.

Example 5.3. PDAs Accepting the Languages *AnBn* and *SimplePal*

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

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

Example 5.7. A Pushdown Automaton Accepting *Pal*


$(q_0, baab, Z_0)$
(q_1, aab, Z_0)
$(q_1, baab, Z_0)$
(q_0, aab, bZ_0)
(q_2, aab, Z_0)
$(q_2, baab, Z_0)$
(q_0, ab, abZ_0)
(q_1, ab, bZ_0)
(q_1, aab, bZ_0)
$(q_0, b, aabZ_0)$
(q_1, b, abZ_0)
(q_1, ab, abZ_0)
$(q_0, \Lambda, baabZ_0)$
$(q_1, \Lambda, aabZ_0)$
$(q_1, b, aabZ_0)$
(q_1, b, bZ_0)
$(q_1, \Lambda, baabZ_0)$
(q_1, Λ, Z_0)
(q_2, Λ, Z_0)

5.2. Deterministic Pushdown Automata

Definition 5.10. A Deterministic Pushdown Automaton

A pushdown automaton $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$ is *deterministic* if it satisfies both of the following conditions.

1. For every $q \in Q$, every $\sigma \in \Sigma \cup \{\Lambda\}$, and every $X \in \Gamma$, the set $\delta(q, \sigma, X)$ has at most one element.

2. For every $q \in Q$, every $\sigma \in \Sigma$, and every $X \in \Gamma$, the two sets $\delta(q, \sigma, X)$ and $\delta(q, \Lambda, X)$ cannot both be nonempty.

A language L is a *deterministic context-free language* (DCFL) if there is a deterministic PDA (DPDA) accepting L.

2. (in other words): For every $q \in Q$ and every $X \in \Gamma$, if $\delta(q, \Lambda, X)$ is not empty, then $\delta(q, \sigma, X)$ is empty for every $\sigma \in \Sigma$.