Computational Linguistics II: Parsing

Formal Languages: Context Free Languages I

Frank Richter & Jan-Philipp Söhn

fr@sfs.uni-tuebingen.de, jp.soehn@uni-tuebingen.de

November 6th, 2006

Once Again: The Big Picture

hierarchy	grammar	machine	other
type 3	reg. grammar	DFA	reg. expressions
		NFA	
det. cf.	LR(k) grammar	DPDA	
type 2	CFG	PDA	
type 1	CSG	LBA	
type 0	unrestricted grammar	Turing machine	

DFA: Deterministic finite state automaton
(D)PDA: (Deterministic) Pushdown automaton
CFG: Context-free grammar
CSG: Context-sensitive grammar

LBA: Linear bounded automaton

Once Again: The Big Picture

hierarchy	grammar	machine	other
type 3	reg. grammar	DFA	reg. expressions
		NFA	
det. cf.	LR(k) grammar	DPDA	
type 2	CFG	PDA	
type 1	CSG	LBA	
type 0	unrestricted grammar	Turing machine	

DFA: Deterministic finite state automaton
(D)PDA: (Deterministic) Pushdown automaton
CFG: Context-free grammar
CSG: Context-sensitive grammar
LBA: Linear bounded automaton

Form of Grammars of Type 0–3

For $i \in \{0, 1, 2, 3\}$, a grammar $\langle N, T, P, S\rangle$ of Type i, with N the set of non-terminal symbols, T the set of terminal symbols (N and T disjoint, $\Sigma = N \cup T$), P the set of productions, and S the start symbol ($S \in N$), obeys the following restrictions:

T3: Every production in P is of the form $A \rightarrow aB$ or $A \rightarrow \epsilon$, with $B, A \in N$, $a \in T$.

T2: Every production in P is of the form $A \rightarrow x$, with $A \in N$ and $x \in \Sigma^*$.

T1: Every production in P is of the form $x_1 A x_2 \rightarrow x_1 y x_2$, with $x_1, x_2 \in \Sigma^*$, $y \in \Sigma^+$, $A \in N$ and the possible exception of $C \rightarrow \epsilon$ in case C does not occur on the righthand side of a rule in P.

T0: No restrictions.

From Regular to Context-Free

- The language $L_1 = \{a^c b^c \mid c=3\}$ is regular.
- Draw an FSA for it!
- The language $L_2 = \{a^n b^n \mid n \geq 1\}$ is not regular.
- Why not? What is required?

From Regular to Context-Free

- The language $L_1 = \{a^c b^c \mid c{=}3\}$ is regular.
- Draw an FSA for it!
- The language $L_2 = \{a^n b^n \mid n \geq 1\}$ is not regular.
- Why not? What is required?

From Regular to Context-Free

- The language $L_1 = \{a^c b^c \mid c{=}3\}$ is regular.
- Draw an FSA for it!
- The language $L_2 = \{a^n b^n \mid n{\geq}1\}$ is not regular.
- Why not? What is required?

From Regular to Context-Free

- The language $L_1 = \{a^c b^c \mid c{=}3\}$ is regular.
- Draw an FSA for it!
- The language $L_2 = \{a^n b^n \mid n{\geq}1\}$ is not regular.
- Why not? What is required?

Introducing the Pushdown-Automaton

- Enhancement of an FSA
- Stack can store a string of any length. Functions PUSH and POP are only allowed at the top of the stack.
- By definition nondeterministic
- Acceptance by final state or by empty stack (equivalence!)
- A language that is recognized by an NPDA is context-free.

Introducing the Pushdown-Automaton

- Enhancement of an FSA
- Stack can store a string of any length. Functions PUSH and POP are only allowed at the top of the stack.
- By definition nondeterministic
- Acceptance by final state or by empty stack (equivalence!)
- A language that is recognized by an NPDA is context-free.

Introducing the Pushdown-Automaton

- Enhancement of an FSA
- Stack can store a string of any length. Functions PUSH and POP are only allowed at the top of the stack.
- By definition nondeterministic
- Acceptance by final state or by empty stack (equivalence!)
- A language that is recognized by an NPDA is context-free.

Introducing the Pushdown-Automaton

- Enhancement of an FSA
- Stack can store a string of any length. Functions PUSH and POP are only allowed at the top of the stack.
- By definition nondeterministic
- Acceptance by final state or by empty stack (equivalence!)
- A language that is recognized by an NPDA is context-free.

Introducing the Pushdown-Automaton

- Enhancement of an FSA
- Stack can store a string of any length. Functions PUSH and POP are only allowed at the top of the stack.
- By definition nondeterministic
- Acceptance by final state or by empty stack (equivalence!)
- A language that is recognized by an NPDA is context-free.

Defining the Pushdown-Automaton

Definition 1 (NPDA) A nondeterministic pushdown-automaton is a septuple $(\Sigma, Q, \Gamma, q_0, Z, F, \delta)$ where

Σ is a finite set called *the input alphabet*,

Q is a finite set of *states*,

Γ is a finite set called *the stack alphabet*,

$q_0 \in Q$ is the *initial state*,

$Z \in \Gamma$ is the *start symbol* on the stack,

$F \subseteq Q$ the set of *final states*, and

δ is the transition function from $Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma$ to $Pow_e(Q \times \Gamma^*)$.

States of an NPDA

Example of δ:
$\delta(q, a, A) \ni (q', B_1 ... B_n)$

A possible state:
$(q_0, abbaa, AZ)$

$\Rightarrow$ Example of an NPDA for
$L_3 = \{a_1 a_2 ... a_n a_n ... a_2 a_1 | a_i \in \{a, b\}\}$

From a CFG to an NPDA

- Write a grammar for $L_3 = \{a^n b^n c^m | n, m \geq 1\}$
- Together, we build an NPDA...

From a CFG to an NPDA

- Write a grammar for $L_3 = \{a^n b^n c^m | n, m \geq 1\}$
- Together, we build an NPDA...

Defining the DPDA

Definition 2 (DPDA) A deterministic pushdown-automaton is a septuple $(\Sigma, Q, \Gamma, q_0, Z, F, \delta)$ as the NPDI where

$$\text{for all } q \in Q,\ a \in \Sigma \text{ and } A \in \Gamma \text{ holds:}$$
$$|\delta(q, a, A)| + |\delta(q, \epsilon, A)| \leq 1$$

i.e. for a given state, input symbol and topmost element of a stack the DPDA never has a choice of move.

- DPDAs accept per final state and not per empty stack.
- A language that is recognized by an DPDA is deterministically context-free (i.e. all context-free languages with unambiguous grammars).
- DPDA languages lie strictly between regular and context-free languages

States of a DPDA

$\Rightarrow$ Example of a DPDA for
$L_4 = \{a_1 a_2 ... a_n \$ a_n ... a_2 a_1 | a_i \in \{a, b\}\}$