

Computational Linguistics II: Parsing

Overview, Left-Recursion, Bottom-up Parsing

Frank Richter & Jan-Philipp Söhn

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

<http://www.english-linguistics.de/fr/teaching/ws06-07/cl2/slides/cl2-lecture16.pdf>

The Big Picture

hierarchy	grammar	machine	other
type 3	reg. grammar	DFA	reg. expressions
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

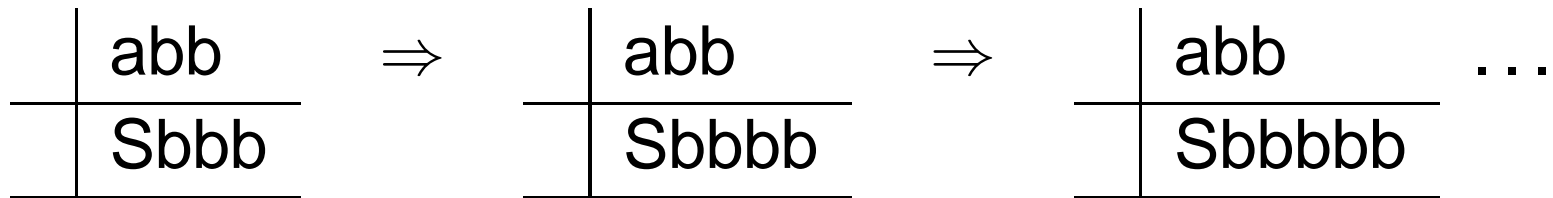
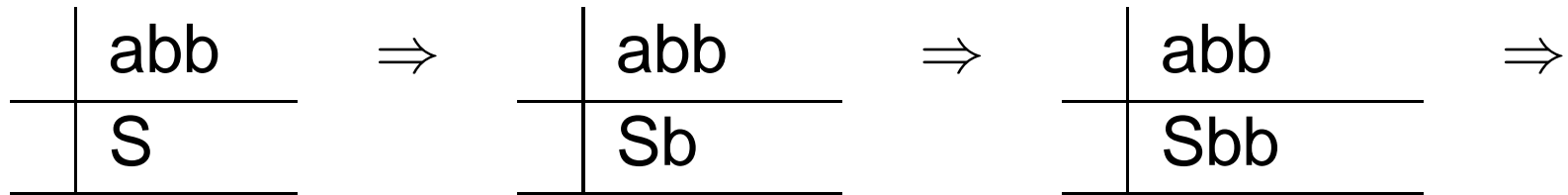
CSG: Context-sensitive grammar

LBA: Linear bounded automaton

Problem: Left-Recursion

- Grammar: $S \rightarrow Sb \mid a$

- Derivation for “abb”:



- Problem: will never get around to finding a terminal in first position which it can match against the input \Rightarrow infinite loop

Left-Recursion

- What exactly is left recursion?
- Two cases:
 1. **immediate left recursion:** $A \rightarrow A \alpha$
lefthand side symbol is the same as **first** righthand side symbol
 2. **indirect left recursion:** $A \rightarrow B \alpha \rightarrow \dots \rightarrow A \beta$
A extends via intermediate steps into another derivation part that **starts** with A.
- Only *first, non-terminal* righthand side symbols are problematic.

Eliminating Immediate Left-Recursion

- rule again: $S \rightarrow Sb \mid a$
 $\Rightarrow a, ab, abb, abbb$
 $\Rightarrow a(b)^*$
- i.e. one part **in front** which ends the recursion, and the recursive part **at the end**
- idea: reformulated rule: $S \rightarrow a X \mid a$
 $X \rightarrow b X \mid b$
- more complicated ex.: $S \rightarrow Sb \mid Sd \mid ac \mid e$
- language: $ac, acb, acbb, acbbb, acbd, acdb, e, eb, ebb, ebd, edb, \dots \Rightarrow (ac \mid e) (b \mid d)^*$
- reformulated:
 $S \rightarrow acX \mid eX \mid ac \mid e; X \rightarrow bX \mid dX \mid b \mid d$

Eliminating Immediate Left-Recursion

- rule form: $A \rightarrow A \alpha_1 \mid \dots \mid A \alpha_n \mid \beta_1 \mid \dots \mid \beta_m$
- **A_head**: righthand sides that are not left-recursive:
betas
 $A_head \rightarrow \beta_1 \mid \dots \mid \beta_m$
- **A_tail**: righthand sides to be duplicated: alphas
 $A_tail \rightarrow \alpha_1 \mid \dots \mid \alpha_n$
- **A_tails**: recursion over alphas:
 $A_tails \rightarrow A_tail \mid A_tail A_tails$
- **A**: puts it together:
 $A \rightarrow A_head \mid A_head A_tails$

Immediate Left-Recursion

- left-recursive rule:

$$A \rightarrow A \alpha_1 \mid \dots \mid A \alpha_n \mid \beta_1 \mid \dots \mid \beta_m$$

- non-left-recursive rules:

$$A \rightarrow A_head \mid A_head A_tails$$

$$A_head \rightarrow \beta_1 \mid \dots \mid \beta_m$$

$$A_tails \rightarrow A_tail \mid A_tail A_tails$$

$$A_tail \rightarrow \alpha_1 \mid \dots \mid \alpha_n$$

Eliminating Indirect Left-Recursion

• grammar:

$$S \rightarrow A B$$
$$A \rightarrow C B \mid b$$
$$C \rightarrow S a$$
$$B \rightarrow b$$

• derivation: $S \Rightarrow A B \Rightarrow C B B \Rightarrow S a B B \Rightarrow A B a B B$
 $\Rightarrow C B B a B B \Rightarrow S a B B a B B \Rightarrow A B a B B a B B \Rightarrow$
 $\dots \Rightarrow bb(abb)^*$

• idea: move left recursion closer, until it is in the same rule

Indirect Left-Recursion (2)

• steps:

1. enumerate all non-terminals: A_1, A_2, \dots, A_n
2. check for A_1 whether it is immediately left-recursive, eliminate left-recursion
3. check for A_2 whether it has a rule: $A_2 \rightarrow A_1 \alpha$
4. yes? if $A_1 \rightarrow \beta_1 \mid \dots \mid \beta_m$:
 $A_2 \rightarrow \beta_1 \alpha \mid \dots \mid \beta_m \alpha$
5. check for A_n whether it has rules:
 $A_n \rightarrow A_1 \alpha$
...
 $A_n \rightarrow A_n \alpha$
6. yes? eliminate according to step 4

Indirect Left-Recursion – Example

- enumerated grammar:

$S1 \rightarrow A B$

$A2 \rightarrow C B \mid b$

$C3 \rightarrow S a$

$B4 \rightarrow b$

- 1**: is S immediately left-recursive? no.
- 2** \rightarrow **1**: rule $A \rightarrow S \alpha$? no.
- 2** \rightarrow **2**: is A immediately left-recursive? no.
- 3** \rightarrow **1**: rule $C \rightarrow S \alpha$? yes: $C3 \rightarrow S1 a$
- replace by: $C \rightarrow A B a$

Indirect Left-Recursion – Example

- $3 \rightarrow 2$: rule $C \rightarrow A \alpha?$ yes (new rule!):
 $C_3 \rightarrow A_2 B a$
- replace by: $C \rightarrow C B B a \mid b B a$
- $3 \rightarrow 3$ (immediate left recursion): rule $C \rightarrow C \alpha?$ yes (new rule!): $C_3 \rightarrow C_3 B B a \mid b B a$
- replace by:
 $C \rightarrow C_head \mid C_head C_tails$
 $C_head \rightarrow b B a$
 $C_tails \rightarrow C_tail \mid C_tail C_tails$
 $C_tail \rightarrow a B B$