Compiler Construction

Lecture 6: Syntactic Analysis II ($LL(k)$ Grammars)

Thomas Noll

Lehrstuhl für Informatik 2
(Software Modeling and Verification)

RWTH Aachen University

noll@cs.rwth-aachen.de

http://www-i2.informatik.rwth-aachen.de/i2/cc08/

Summer semester 2008

Outline

Parsing Context-Free Languages

Problem (Word problem for context-free languages)

Given $G \in CFG_\Sigma$ and $w \in \Sigma^*$, decide whether $w \in L(G)$ (and determine a corresponding syntax tree).

- Decidable for arbitrary CFGs (in Chomsky Normal Form) using the tabular method by Cocke, Younger, and Kasami ("CYK Algorithm"; space/time complexity $\mathcal{O}(|w|^2)/\mathcal{O}(|w|^3)$)
- **Goal:** exploit the special syntactic structures as present in programming languages (usually: no ambiguities) to devise parsing methods which are based on deterministic pushdown automata with linear space and time complexity

Two approaches:

Top-down parsing: construction of syntax tree from the root towards the leafs, representation as leftmost derivation

Bottom-up parsing: construction of syntax tree from the leafs towards the root, representation as (reversed) rightmost derivation

Top-Down Parsing

Approach:

1. Given $G \in CFG_\Sigma$, construct a nondeterministic pushdown automaton (PDA) which accepts $L(G)$ and which additionally computes corresponding leftmost derivations (similar to the proof of "$L(CFG_\Sigma) \subseteq L(PDA_\Sigma)$")
   - input alphabet: Σ
   - pushdown alphabet: X
   - output alphabet: $[p]$
   - state set: omitted
2. Remove nondeterminism by allowing lookahead on the input: $G \in LL(k)$ iff $L(G)$ recognizable by deterministic PDA with lookahead of k symbols

The Nondeterministic Top-Down Automaton I

Definition (Nondeterministic top-down parsing automaton)

Let $G = \langle N, \Sigma, P, S \rangle \in CFG_\Sigma$. The nondeterministic top-down parsing automaton of G, NTA(G), is defined by the following components.

- Input alphabet: Σ
- Pushdown alphabet: X
- Output alphabet: $[p]$
- Configurations: $\Sigma^* \times X^* \times [p]^*$ (top of pushdown to the left)
- Transitions for $w \in \Sigma^*$, $\alpha \in X^*$, and $z \in [p]^*$:
  expansion steps: if $\pi(i) = A \to \beta$, then $(w, A\alpha, z) \vdash (w, \beta\alpha, zi)$
  matching steps: for every $a \in \Sigma$, $(aw, a\alpha, z) \vdash (w, \alpha, z)$
- Initial configuration for $w \in \Sigma^*$: (w, S, ε)
- Final configurations: $\{\varepsilon\} \times \{\varepsilon\} \times [p]^*$

Remark: NTA(G) is nondeterministic iff G contains $A \to \beta \mid \gamma$

The Nondeterministic Top-Down Automaton II

Theorem (Correctness of NTA(G))

Let $G = \langle N, \Sigma, P, S \rangle \in CFG_\Sigma$ *and* NTA(G) *as before. Then, for every* $w \in \Sigma^*$ *and* $z \in [p]^*$,

$$(w, S, \varepsilon) \vdash^* (\varepsilon, \varepsilon, z) \quad \textit{iff} \quad z \textit{ is a leftmost analysis of } w$$

Proof.

$\Longrightarrow$ (soundness): see exercises

$\Longleftarrow$ (completeness): on the board

□

Outline

1. Repetition: Top-Down Parsing of Context-Free Languages

2. Adding Lookahead

3. $LL(k)$ Grammars

Adding Lookahead

Goal: resolve nondeterminism of NTA(G) by supporting lookahead of $k \in \mathbb{N}$ symbols on the input

$\Longrightarrow$ determination of expanding A-production by next k symbols

Adding Lookahead

Goal: resolve nondeterminism of NTA(G) by supporting lookahead of $k \in \mathbb{N}$ symbols on the input

$\implies$ determination of expanding A-production by next k symbols

Definition 6.1 (first_k set)

Let $G = \langle N, \Sigma, P, S \rangle \in CFG_\Sigma$, $\alpha \in X^*$, and $k \in \mathbb{N}$. Then the first_k set of α, $\text{first}_k(\alpha) \subseteq \Sigma^*$, is given by

$$\begin{aligned}\text{first}_k(\alpha) := &\{v \in \Sigma^k \mid \text{ex. } w \in \Sigma^* \text{ such that } \alpha \Rightarrow^* vw\} \cup \\ &\{v \in \Sigma^{<k} \mid \alpha \Rightarrow^* v\}\end{aligned}$$

Properties of first_k Sets

Lemma 6.2

Let $G = \langle N, \Sigma, P, S\rangle \in CFG_\Sigma$, $\alpha, \beta \in X^*$, *and* $k \in \mathbb{N}$.

1. $\mathrm{first}_k(\alpha) \neq \emptyset$
2. $\varepsilon \in \mathrm{first}_k(\alpha)$ *iff* $k = 0$ *or* $\alpha \Rightarrow^* \varepsilon$
3. $\alpha \Rightarrow^* \beta \implies \mathrm{first}_k(\beta) \subseteq \mathrm{first}_k(\alpha)$
4. $v \in \mathrm{first}_k(\alpha)$ *iff ex.* $w \in \Sigma^*$ *such that* $\alpha \Rightarrow^* w$ *and* $\mathrm{first}_k(w) = \{v\}$

Properties of first_k Sets

Lemma 6.2

Let $G = \langle N, \Sigma, P, S\rangle \in CFG_\Sigma$, $\alpha, \beta \in X^*$, *and* $k \in \mathbb{N}$.

1. $\text{first}_k(\alpha) \neq \emptyset$
2. $\varepsilon \in \text{first}_k(\alpha)$ *iff* $k = 0$ *or* $\alpha \Rightarrow^* \varepsilon$
3. $\alpha \Rightarrow^* \beta \implies \text{first}_k(\beta) \subseteq \text{first}_k(\alpha)$
4. $v \in \text{first}_k(\alpha)$ *iff ex.* $w \in \Sigma^*$ *such that* $\alpha \Rightarrow^* w$ *and* $\text{first}_k(w) = \{v\}$

Proof.

on the board □

Outline

1. Repetition: Top-Down Parsing of Context-Free Languages
2. Adding Lookahead
3. $LL(k)$ Grammars

$LL(k)$ Grammars I

$LL(k)$: reading of input from left to right with k-lookahead, computing a leftmost analysis

$LL(k)$ Grammars I

$LL(k)$: reading of input from left to right with k-lookahead, computing a leftmost analysis

Definition 6.3 ($LL(k)$ grammar)

Let $G = \langle N, \Sigma, P, S \rangle \in CFG_\Sigma$ and $k \in \mathbb{N}$. Then G has the $LL(k)$ property (notation: $G \in LL(k)$) if for all leftmost derivations of the form

$$S \Rightarrow_l^* wA\alpha \begin{cases} \Rightarrow_l w\beta\alpha \Rightarrow_l^* wx \\ \Rightarrow_l w\gamma\alpha \Rightarrow_l^* wy \end{cases}$$

such that $\text{first}_k(x) = \text{first}_k(y)$, it follows that $\beta = \gamma$ (i.e., the same production is applied to A).

$LL(k)$ Grammars I

$LL(k)$: reading of input from left to right with k-lookahead, computing a leftmost analysis

Definition 6.3 ($LL(k)$ grammar)

Let $G = \langle N, \Sigma, P, S \rangle \in CFG_\Sigma$ and $k \in \mathbb{N}$. Then G has the $LL(k)$ property (notation: $G \in LL(k)$) if for all leftmost derivations of the form

$$S \Rightarrow_l^* wA\alpha \begin{cases} \Rightarrow_l w\beta\alpha \Rightarrow_l^* wx \\ \Rightarrow_l w\gamma\alpha \Rightarrow_l^* wy \end{cases}$$

such that $\text{first}_k(x) = \text{first}_k(y)$, it follows that $\beta = \gamma$ (i.e., the same production is applied to A).

Remarks:

- If $G \in LL(k)$, then the leftmost derivation step for $wA\alpha$ in the above diagram is determined by the next k symbols following w.
- Problem: how to determine the A-production from the lookahead (potentially infinitely many derivations to wx/wy)?

$LL(k)$ Grammars II

Lemma 6.4 (Characterization of $LL(k)$)

$G \in LL(k)$ iff for all leftmost derivations of the form

$$S \Rightarrow_l^* wA\alpha \begin{cases} \Rightarrow_l w\beta\alpha \\ \Rightarrow_l w\gamma\alpha \end{cases}$$

such that $\beta \neq \gamma$, it follows that $\text{first}_k(\beta\alpha) \cap \text{first}_k(\gamma\alpha) = \emptyset$.

$LL(k)$ Grammars II

Lemma 6.4 (Characterization of $LL(k)$)

$G \in LL(k)$ iff for all leftmost derivations of the form

$$S \Rightarrow_l^* wA\alpha \begin{cases} \Rightarrow_l w\beta\alpha \\ \Rightarrow_l w\gamma\alpha \end{cases}$$

such that $\beta \neq \gamma$, it follows that $\mathrm{first}_k(\beta\alpha) \cap \mathrm{first}_k(\gamma\alpha) = \emptyset$.

Proof.

on the board □

Lemma 6.4 (Characterization of $LL(k)$)

$G \in LL(k)$ iff for all leftmost derivations of the form

$$S \Rightarrow_l^* wA\alpha \begin{cases} \Rightarrow_l w\beta\alpha \\ \Rightarrow_l w\gamma\alpha \end{cases}$$

such that $\beta \neq \gamma$, it follows that $\text{first}_k(\beta\alpha) \cap \text{first}_k(\gamma\alpha) = \emptyset$.

Proof.

on the board □

Remarks:

- If $G \in LL(k)$, then the A-production is determined by the lookahead sets $\text{first}_k(\beta\alpha)$ (for every $A \to \beta \in P$).
- Problem: still infinitely many rightmost contexts α to be considered (if β "too short", i.e., $\text{first}_k(\beta\alpha) \neq \text{first}_k(\beta)$).