Recursive Descent Parsers

Lecture 6

Robb T. Koether

Hampden-Sydney College

Mon, Feb 2, 2009

Outline

Recursive Descent Parser

- Each nonterminal in the grammar is implemented as a function.
- Begin with the start symbol S of the grammar by calling the function `S()`.
- Based on the first token received, apply the appropriate grammar rule for S.
- Continue in this manner until S is "satisfied." That is, all nonterminals have been replaced with terminals.

Recursive Descent Parsers

- The first Pascal compiler used a recursive descent parser.
- Recursive descent parsers have the benefit of being very simple to implement.
- However,
  - Error-recovery is difficult.
  - They are not able to handle as large a set of grammars as other parsing methods.

Error Recovery

- When a syntax error occurs, in order for the compiler to recover, it usually has to discard the last few tokens, move to the end of the line, and resume.
- In a recursive descent parser, discarding tokens involves returning from several nested function calls.
- In a table-driven parser, discarding tokens requires simply clearing part of the stack.

Example

Example (Recursive Descent)

- Write a parser for the following grammar.

$$\begin{aligned} S &\rightarrow \textbf{if } C \textbf{ then } S \textbf{ ;} \\ &\quad\mid \textbf{while } C \textbf{ do } S \textbf{ ;} \\ &\quad\mid \textbf{id} = \textbf{num} \\ &\quad\mid \textbf{id} \; {++} \\ C &\rightarrow \textbf{id} == \textbf{num} \mid \textbf{id} \; != \textbf{num} \end{aligned}$$

where S represents a statement and C represents a condition.

Example

Example (Recursive Descent)

- Run the example `SimpleParser.java`.

Example

Example (Recursive Descent)

- Modify the previous example by adding the production

$$S' \rightarrow SS' \mid \varepsilon$$

where S' represents a sequence of statements.

Example

Example (Recursive Descent)

- Modify the previous example by adding the productions

$$\begin{aligned} S &\rightarrow \textbf{do}\ S\ \textbf{while}\ C\ \textbf{;} \\ C &\rightarrow \textbf{id} < \textbf{num} \end{aligned}$$