Predictive Parsing

Lecture 9
Section 4.4

Robb T. Koether

Hampden-Sydney College

Wed, Feb 11, 2009

Outline

Nullability and FIRST for Strings

- The parse table has
  - One row for each nonterminal,
  - One column for each terminal and `$`.
- `$` is the end-of-file marker.
- Each cell in the table represents a combination (A, a) of a nonterminal A and a terminal a.

Nullability and FIRST for Strings

- We need to extend the definitions of nullable and FIRST to *strings* of grammar symbols.
  - ε is nullable.
  - Let $\alpha = X_1 X_2 \ldots X_n$ be a string of grammar symbols. Then α is nullable if each X_i is nullable.

Nullability and FIRST for Strings

- By definition, FIRST$(\varepsilon) = \varnothing$.
- For the string $\alpha = X_1 X_2 \ldots X_n$,
  - Let k be the largest integer for which X_1, X_2, ..., and X_k are nullable.
  - Then

$$\text{FIRST}(\alpha) = \text{FIRST}(X_1) \cup \ldots \cup \text{FIRST}(X_{k+1}).$$

- We will not need FOLLOW for strings.

Parse Table Entries

- Each table entry is a production $A \to \alpha$.
- Rules for entering $A \to \alpha$ in the table:
  - For every rule $A \to \alpha$ and for every $a \in \text{FIRST}(\alpha)$, write $A \to \alpha$ in cell (A, a).
  - For every α that is nullable and for every rule $A \to \alpha$ and for every $a \in \text{FOLLOW}(A)$, write $A \to \alpha$ in cell (A, a).
- Write "error" in all cells that do not contain a production.

Parse Table Entries

- The interpretation of $A \to \alpha$ in cell (A, a) is that if A is on top of the stack and we encounter a in the input, then we should replace A by α on the stack.
- Push the symbols of α onto the stack in *reverse order*, from right to left, so that the first (leftmost) symbol is on the top of the stack.

Example

Example (Parse Table)

- Let the grammar be

$$
\begin{aligned}
E &\rightarrow T\ E' \\
E' &\rightarrow +\ T\ E' \mid \varepsilon \\
T &\rightarrow F\ T' \\
T' &\rightarrow *\ F\ T' \mid \varepsilon \\
F &\rightarrow (E) \mid \mathbf{id} \mid \mathbf{num}
\end{aligned}
$$

Example

Example (Parse Table)

- Recall

Nonterminal	Nullable	FIRST	FOLLOW
E	No	$\{(, \mathbf{id}, \mathbf{num}\}$	$\{\$,)\}$
E'	Yes	$\{+\}$	$\{\$,)\}$
T	No	$\{(, \mathbf{id}, \mathbf{num}\}$	$\{\$,), +\}$
T'	Yes	$\{*\}$	$\{\$,), +\}$
F	No	$\{(, \mathbf{id}, \mathbf{num}\}$	$\{*, \$,), +\}$

Example

Example (Parse Table)

- Consider the production $E \to T\ E'$.
  - $FIRST(TE') = FIRST(T) = \{\ (, \mathbf{num}, \mathbf{id}\}$.
  - Therefore, enter $E \to T\ E'$ in cells $(E,\ (\ )$, $(E, \mathbf{num})$, and $(E, \mathbf{id})$.
  - $T\ E'$ is not nullable.

Example

Example (Parse Table)

- Consider the production $E' \longrightarrow \varepsilon$.
- $FIRST(\varepsilon) = \varnothing$.
- ε is nullable and $FOLLOW(E') = \{\$,)\}$.
- Therefore, enter $E' \longrightarrow \varepsilon$ in cells $(E', \$)$ and $(E',)\,)$.

Example

Example (Parse Table)

- Handle the other productions similarly.
- In which cells do we enter $E' \longrightarrow + \ T \ E'$?
- In which cells do we enter $T \longrightarrow F \ T'$?
- In which cells do we enter $T' \longrightarrow * \ F \ T'$?
- In which cells do we enter $T' \longrightarrow \varepsilon$?
- In which cells do we enter $F \longrightarrow \ (E)$?
- In which cells do we enter $F \longrightarrow \mathbf{id}$?
- In which cells do we enter $F \longrightarrow \mathbf{num}$?

Example

Example (Parse Table)

	+	*	(	)	**num**	**id**	$
E			$E \to T\ E'$		$E \to T\ E'$	$E \to T\ E'$	
E'	$E' \to +\ T\ E'$			$E' \to \varepsilon$			$E' \to \varepsilon$
T			$T \to F\ T'$		$T \to F\ T'$	$T \to F\ T'$	
T'	$T' \to \varepsilon$	$T' \to *\ F\ T'$		$T' \to \varepsilon$			$T' \to \varepsilon$
F			$F \to (E)$		$F \to$ **num**	$F \to$ **id**	

Predictive Parsing

Definition (LL(1) Grammar)

An LL(1) grammar is a grammar whose predictive parse table does not contain any multiple entries.

- A multiple entry would indicate that we could not decide which production to apply.

Predictive Parsing Algorithm

- The predictive parsing algorithm uses
  - The parse table,
  - An input buffer containing a sequence of tokens,
  - A stack of grammar symbols.

Predictive Parsing Algorithm

The Parsing Algorithm

- Initialize the input buffer to the input followed by `$`.
- Initialize the stack to `$` and S, with S on the top.

Predictive Parsing Algorithm

The Parsing Algorithm

- Consider the top stack symbol X.
- There are three possibilities.
  - X is a terminal.
  - X is a nonterminal.
  - X is \$.

Predictive Parsing Algorithm

The Parsing Algorithm

- If X is a terminal, then
  - If X matches the current token,
    - Pop X from the stack.
    - Advance to the next token.
  - Otherwise, it is an error.

Predictive Parsing Algorithm

The Parsing Algorithm

- If X is a nonterminal, then
  - Use X together with the current token to get the entry from the parse table.
  - If the entry is a production,
    - Pop X from the stack.
    - Push the symbols on the right-hand side of the production, from right to left, onto the stack.
  - Otherwise, it is an error.

Predictive Parsing Algorithm

The Parsing Algorithm

- If X is `$`, then
  - If the current token is also `$`,
    - Accept the input.
  - Otherwise, it is an error.

Example

Example (Predictive Parsing)

- Parse the string $(\mathbf{id} + \mathbf{num}) * \mathbf{id}$.

Example

Example (Predictive Parsing)

Stack	Input
$ E	(**id** + **num**) $\star$ **id** $
$ $E'\ T$	(**id** + **num**) $\star$ **id** $
$ $E'\ T'\ F$	(**id** + **num**) $\star$ **id** $
$ $E'\ T'\ )\ E\ ($	(**id** + **num**) $\star$ **id** $
$ $E'\ T'\ )\ E$	**id** + **num**) $\star$ **id** $
$ $E'\ T'\ )\ E'\ T$	**id** + **num**) $\star$ **id** $
$ $E'\ T'\ )\ E'\ T'\ F$	**id** + **num**) $\star$ **id** $
$ $E'\ T'\ )\ E'\ T'$ **id**	**id** + **num**) $\star$ **id** $
$ $E'\ T'\ )\ E'\ T'$	+ **num**) $\star$ **id** $
$ $E'\ T'\ )\ E'$	+ **num**) $\star$ **id** $
$ $E'\ T'\ )\ E'\ T\ +$	+ **num**) $\star$ **id** $
$ $E'\ T'\ )\ E'\ T$	**num**) $\star$ **id** $

Example

Example (Predictive Parsing)

Stack	Input
$ $E'\ T'\ )\ E'\ T'\ F$	**num**) ⋆ **id** $
$ $E'\ T'\ )\ E'\ T'$ **num**	**num**) ⋆ **id** $
$ $E'\ T'\ )\ E'\ T'$	) ⋆ **id** $
$ $E'\ T'\ )\ E'$	) ⋆ **id** $
$ $E'\ T'\ )$	) ⋆ **id** $
$ $E'\ T'$	⋆ **id** $
$ $E'\ T'\ F$ ⋆	⋆ **id** $
$ $E'\ T'\ F$	**id** $
$ $E'\ T'$ **id**	**id** $
$ $E'\ T'$	$
$ E'	$
$	$

Assignment

Homework

- The grammar

$$R \quad \longrightarrow \quad R \cup R \mid RR \mid R^* \mid (R) \mid \mathbf{a} \mid \mathbf{b}$$

  generates all regular expressions on the alphabet $\Sigma = \{\mathbf{a}, \mathbf{b}\}$.
- Using FIRST and FOLLOW from the exercise of the previous lecture, construct a parse table for this grammar.
- Parse the expression $\mathbf{ab}^* (\mathbf{a} \cup \mathbf{b})$.