COSE312: Compilers

Lecture 9 — LR Parsing with Ambiguous Grammars

Hakjoo Oh
2017 Spring

Parsing with Ambiguous Grammars

In programming languages, ambiguous grammars provide more natural and concise specification:

$$E \rightarrow E + E \mid E * E \mid (E) \mid \mathbf{id} \quad \textit{vs.} \quad \begin{array}{lcl} E & \rightarrow & E + T \mid T \\ T & \rightarrow & T * F \mid F \\ F & \rightarrow & (E) \mid \mathbf{id} \end{array}$$

Conflicts

$$I_0: \boxed{\begin{array}{l} E' \to .E \\ E \to .E + E \\ E \to .E * E \\ E \to .(E) \\ E \to .\mathrm{id} \end{array}} \qquad I_1: \boxed{\begin{array}{l} E' \to E. \\ E \to E. + E \\ E \to E. * E \end{array}} \qquad I_2: \boxed{\begin{array}{l} E \to (.E) \\ E \to .E + E \\ E \to .E * E \\ E \to .(E) \\ E \to .\mathrm{id} \end{array}}$$

$$I_3: \boxed{E \to \mathrm{id}.} \qquad I_4: \boxed{\begin{array}{l} E \to E + .E \\ E \to .E + E \\ E \to .E * E \\ E \to .(E) \\ E \to .\mathrm{id} \end{array}} \qquad I_5: \boxed{\begin{array}{l} E \to E * .E \\ E \to .E + E \\ E \to .E * E \\ E \to .(E) \\ E \to .\mathrm{id} \end{array}}$$

$$I_6: \boxed{\begin{array}{l} E \to (E.) \\ E \to E. + E \\ E \to E. * E \end{array}} \qquad I_7: \boxed{\begin{array}{l} E \to E + E. \\ E \to E. + E \\ E \to E. * E \end{array}} \qquad I_8: \boxed{\begin{array}{l} E \to E * E. \\ E \to E. + E \\ E \to E. * E \end{array}} \qquad I_9: \boxed{E \to (E).}$$

SLR Parsing Table

STATE	id	+	*	(	)	$	E
0	$s3$			$s2$			$g1$
1		$s4$	$s5$			acc	
2	$s3$			$s2$			$g6$
3		$r4$	$r4$		$r4$	$r4$	
4	$s3$			$s2$			$g7$
5	$s3$			$s2$			$g8$
6		$s4$	$s5$		$s9$		
7		$s4, r1$	$s5, r1$		$r1$	$r1$	
8		$s4, r2$	$s5, r2$		$r2$	$r2$	
9		$r3$	$r3$		$r3$	$r3$	

Resolving Conflicts with Precedence and Associativity

Conflicts are resolved by assuming that

- $*$ takes precedence over $+$, and
- $+$ and $*$ are left-associative.

Resolving Conflicts with Precedence

The parsing process has shift/reduce conflicts for input $\mathbf{id} + \mathbf{id} * \mathbf{id}$:

Stack	Symbols	Input	Action
0		$\mathbf{id} + \mathbf{id} * \mathbf{id\$}$	shift to 3
0 3	$\mathbf{id}$	$+\mathbf{id} * \mathbf{id\$}$	reduce by 4
0 1	$\boldsymbol{E}$	$+\mathbf{id} * \mathbf{id\$}$	shift to 4
0 1 4	$\boldsymbol{E}+$	$\mathbf{id} * \mathbf{id\$}$	shift to 3
0 1 4 3	$\boldsymbol{E} + \mathbf{id}$	$*\mathbf{id\$}$	reduce by 4
0 1 4 7	$\boldsymbol{E} + \boldsymbol{E}$	$*\mathbf{id\$}$	shift to 5, reduce by 1

Which is the correct action?

Resolving Conflicts with Precedence

When we choose the shift action:

Stack	Symbols	Input	Action
0		$\mathbf{id} + \mathbf{id} * \mathbf{id}\$$	shift to 3
0 3	$\mathbf{id}$	$+\mathbf{id} * \mathbf{id}\$$	reduce by 4
0 1	$\boldsymbol{E}$	$+\mathbf{id} * \mathbf{id}\$$	shift to 4
0 1 4	$\boldsymbol{E}+$	$\mathbf{id} * \mathbf{id}\$$	shift to 3
0 1 4 3	$\boldsymbol{E} + \mathbf{id}$	$*\mathbf{id}\$$	reduce by 4
0 1 4 7	$\boldsymbol{E} + \boldsymbol{E}$	$*\mathbf{id}\$$	shift to 5, reduce by 1
0 1 4 7 5	$\boldsymbol{E} + \boldsymbol{E}*$	$\mathbf{id}\$$	shift to 3
0 1 4 7 5 3	$\boldsymbol{E} + \boldsymbol{E} * \mathbf{id}$	$\$$	reduce by 4
0 1 4 7 5 8	$\boldsymbol{E} + \boldsymbol{E} * \boldsymbol{E}$	$\$$	reduce by 2
0 1 4 7	$\boldsymbol{E} + \boldsymbol{E}$	$\$$	reduce by 1
0 1	$\boldsymbol{E}$	$\$$	accept

Resolving Conflicts with Precedence

When we choose the reduce action:

Stack	Symbols	Input	Action
0		$\mathbf{id} + \mathbf{id} * \mathbf{id}\$$	shift to 3
0 3	$\mathbf{id}$	$+\mathbf{id} * \mathbf{id}\$$	reduce by 4
0 1	$\boldsymbol{E}$	$+\mathbf{id} * \mathbf{id}\$$	shift to 4
0 1 4	$\boldsymbol{E}+$	$\mathbf{id} * \mathbf{id}\$$	shift to 3
0 1 4 3	$\boldsymbol{E} + \mathbf{id}$	$*\mathbf{id}\$$	reduce by 4
0 1 4 7	$\boldsymbol{E} + \boldsymbol{E}$	$*\mathbf{id}\$$	shift to 5, reduce by 1
0 1	$\boldsymbol{E}$	$*\mathbf{id}\$$	shift to 5
0 1 5	$\boldsymbol{E}*$	$\mathbf{id}\$$	shift to 3
0 1 5 3	$\boldsymbol{E} * \mathbf{id}$	$\$$	reduce by 4
0 1 5 8	$\boldsymbol{E} * \boldsymbol{E}$	$\$$	reduce by 2
0 1	$\boldsymbol{E}$	$\$$	accept

Resolving Conflicts with Precedence

Take the shift action when the parser is at state 7 and the next input symbol is $*$:

STATE	id	+	*	(	)	$	E
0	$s3$			$s2$			$g1$
1		$s4$	$s5$			acc	
2	$s3$			$s2$			$g6$
3		$r4$	$r4$		$r4$	$r4$	
4	$s3$			$s2$			$g7$
5	$s3$			$s2$			$g8$
6		$s4$	$s5$		$s9$		
7		$s4, r1$	$s5$		$r1$	$r1$	
8		$s4, r2$	$s5, r2$		$r2$	$r2$	
9		$r3$	$r3$		$r3$	$r3$	

Resolving Conflicts with Associativity

The parsing goes into a shift/reduce conflict for input $\mathbf{id} + \mathbf{id} + \mathbf{id}$:

Stack	Symbols	Input	Action
0 1 4 7	$\boldsymbol{E} + \boldsymbol{E}$	$+\mathbf{id\$}$	shift to 4, reduce by 1

Which is the correct action?

Resolving Conflicts with Associativity

STATE	id	+	*	(	)	$	E
0	$s3$			$s2$			$g1$
1		$s4$	$s5$			acc	
2	$s3$			$s2$			$g6$
3		$r4$	$r4$		$r4$	$r4$	
4	$s3$			$s2$			$g7$
5	$s3$			$s2$			$g8$
6		$s4$	$s5$		$s9$		
7		$r1$	$s5$		$r1$	$r1$	
8		$r2$	$r2$		$r2$	$r2$	
9		$r3$	$r3$		$r3$	$r3$	

The "Dangling-Else" Ambiguity

$$
\begin{array}{lcl}
\textit{stmt} & \rightarrow & \textbf{if } \textit{expr} \textbf{ then } \textit{stmt} \\
& | & \textbf{if } \textit{expr} \textbf{ then } \textit{stmt} \textbf{ else } \textit{stmt} \\
& | & \textbf{other}
\end{array}
$$

The "Dangling-Else" Ambiguity

$$
\begin{array}{rcl}
\textit{stmt} & \rightarrow & \textbf{if } \textit{expr} \textbf{ then } \textit{stmt} \\
& | & \textbf{if } \textit{expr} \textbf{ then } \textit{stmt} \textbf{ else } \textit{stmt} \\
& | & \textbf{other}
\end{array}
$$

Simplified grammar:

$$
\begin{array}{rcl}
S' & \rightarrow & S \\
S & \rightarrow & i\ S\ e\ S \mid i\ S \mid a
\end{array}
$$

Conflicts

LR(0) states include the state:

$$I_4 = \boxed{\begin{array}{l} S \rightarrow iS.eS \\ S \rightarrow iS. \end{array}}$$

Conflicts

LR(0) states include the state:

$$I_4 = \boxed{\begin{array}{l} S \rightarrow iS.eS \\ S \rightarrow iS. \end{array}}$$

The conflict in the SLR parsing table:

STATE	i	e	a	\$	S
4		$s5, r2$		$r2$	

Which is the correct action?

Summary

- Ambiguous grammar is useful for programming languages.
- We can use the ambiguous grammar in LR parsing by specifying precedence and associativity rules.