COSE312: Compilers

Lecture 6 — Syntax Analysis (4): Ambiguous Grammars

Hakjoo Oh
2026 Spring

Parsing with Ambiguous Grammars

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

$$E \rightarrow E + E \mid E * E \mid (E) \mid \mathbf{id} \tag{1}$$

$$\begin{aligned}
E &\rightarrow E + T \mid T \\
T &\rightarrow T * F \mid F \\
F &\rightarrow (E) \mid \mathbf{id}
\end{aligned} \tag{2}$$

- The ambiguous grammar in (1) does not specify the associativity or precedence of the operators $+$ and $*$.
- The unambiguous grammar in (2) gives $+$ lower precedence than $*$, makes both operators left associative.
- In practice, we prefer to use the ambiguous grammar because we can often enforce the associativity and precedence w/o changing grammar.

Conflicts

The augmented grammar:

$$(0)\ E' \to E, \quad E \to (1)\ E + E \mid (2)\ E * E \mid (3)\ (E) \mid (4)\ \text{id}$$

The sets of LR(0) items:

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

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

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

Which states cause conflicts during the SLR parsing?

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 $id + id * id$:

Stack	Symbols	Input	Action
0		$id + id * id\$$	shift to 3
0 3	id	$+id * id\$$	reduce by 4
0 1	E	$+id * id\$$	shift to 4
0 1 4	$E+$	$id * id\$$	shift to 3
0 1 4 3	$E + id$	$*id\$$	reduce by 4
0 1 4 7	$E + E$	$*id\$$	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		$id + id * id\$$	shift to 3
0 3	id	$+id * id\$$	reduce by 4
0 1	E	$+id * id\$$	shift to 4
0 1 4	$E+$	$id * id\$$	shift to 3
0 1 4 3	$E + id$	$*id\$$	reduce by 4
0 1 4 7	$E + E$	$*id\$$	<u>shift to 5</u>, reduce by 1
0 1 4 7 5	$E + E*$	$id\$$	shift to 3
0 1 4 7 5 3	$E + E * id$	$\$$	reduce by 4
0 1 4 7 5 8	$E + E * E$	$\$$	reduce by 2
0 1 4 7	$E + E$	$\$$	reduce by 1
0 1	E	$\$$	accept

Resolving Conflicts with Precedence

When we choose the reduce action:

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

Which is the correct action?

Resolving Conflicts with Associativity

Take the reduce 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		r1	s5		r1	r1	
8		s4, r2	s5, r2		r2	r2	
9		r3	r3		r3	r3	

Exercise

Suppose the parse is at state 8.

- Which is correct when the next input is $+$? Explain with an example.
- Which is correct when the next input is $*$? Explain with an example.

The "Dangling-Else" Ambiguity

Grammar for conditional statements:

$$
\begin{aligned}
stmt \quad \rightarrow \quad & \text{if } expr \text{ then } stmt \\
| \quad & \text{if } expr \text{ then } stmt \text{ else } stmt \\
| \quad & \text{other}
\end{aligned}
$$

Consider the statement:

$$\text{if } E_1 \text{ then if } E_2 \text{ then } S_1 \text{ else } S_2$$

which has two parse trees.

The "Dangling-Else" Ambiguity

Grammar (simplified and augmented):

$$S' \rightarrow S$$
$$S \rightarrow i\ S\ e\ S \mid i\ S \mid a$$

LR(0) states:

$$I_0 = \begin{array}{|l|} \hline S' \rightarrow .S \\ S \rightarrow .iSeS \\ S \rightarrow .iS \\ S \rightarrow .a \\ \hline \end{array} \quad I_1 = \boxed{S' \rightarrow S.} \quad I_2 = \begin{array}{|l|} \hline S \rightarrow i.SeS \\ S \rightarrow i.S \\ S \rightarrow .iSeS \\ S \rightarrow .iS \\ S \rightarrow .a \\ \hline \end{array} \quad I_3 = \boxed{S \rightarrow a.}$$

$$I_4 = \begin{array}{|l|} \hline S \rightarrow iS.eS \\ S \rightarrow iS. \\ \hline \end{array} \quad I_5 = \begin{array}{|l|} \hline S \rightarrow iSe.S \\ S \rightarrow .iSeS \\ S \rightarrow .iS \\ S \rightarrow .a \\ \hline \end{array} \quad I_6 = \boxed{S \rightarrow iSeS.}$$

Which states generate conflicts?

Exercise

- $FOLLOW(S) =$

- Complete the SLR parsing table:

		Action			Goto
STATE	i	e	a	$\$$	S
0	$s2$		$s3$		1
1				acc	
2	$s2$		$s3$		4
3		$r3$		$r3$	
4					
5	$s2$		$s3$		6
6		$r1$		$r1$	

- Which action is correct when conflicts occur? Remove the ambiguity in the parsing table.

Exercise

Describe the parsing actions on input $iiaea$:

Stack	Symbols	Input	Action
0		$iiaea\$$	shift

Exercise

The ambiguity of the grammar

$$
\begin{aligned}
stmt \quad &\rightarrow \quad \text{if } expr \text{ then } stmt \\
&\mid \quad \text{if } expr \text{ then } stmt \text{ else } stmt \\
&\mid \quad \text{other}
\end{aligned}
$$

can be eliminated by introducing auxiliary nonterminals M (*matched statement*) and U (*unmatched statement*):

$$
\begin{aligned}
S \quad &\rightarrow \quad M \\
S \quad &\rightarrow \quad U \\
M \quad &\rightarrow \quad \text{if } expr \text{ then } ? \text{ else } ? \\
M \quad &\rightarrow \quad \text{other} \\
U \quad &\rightarrow \quad \text{if } expr \text{ then } ? \\
U \quad &\rightarrow \quad \text{if } expr \text{ then } ? \text{ else } ?
\end{aligned}
$$

Summary

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