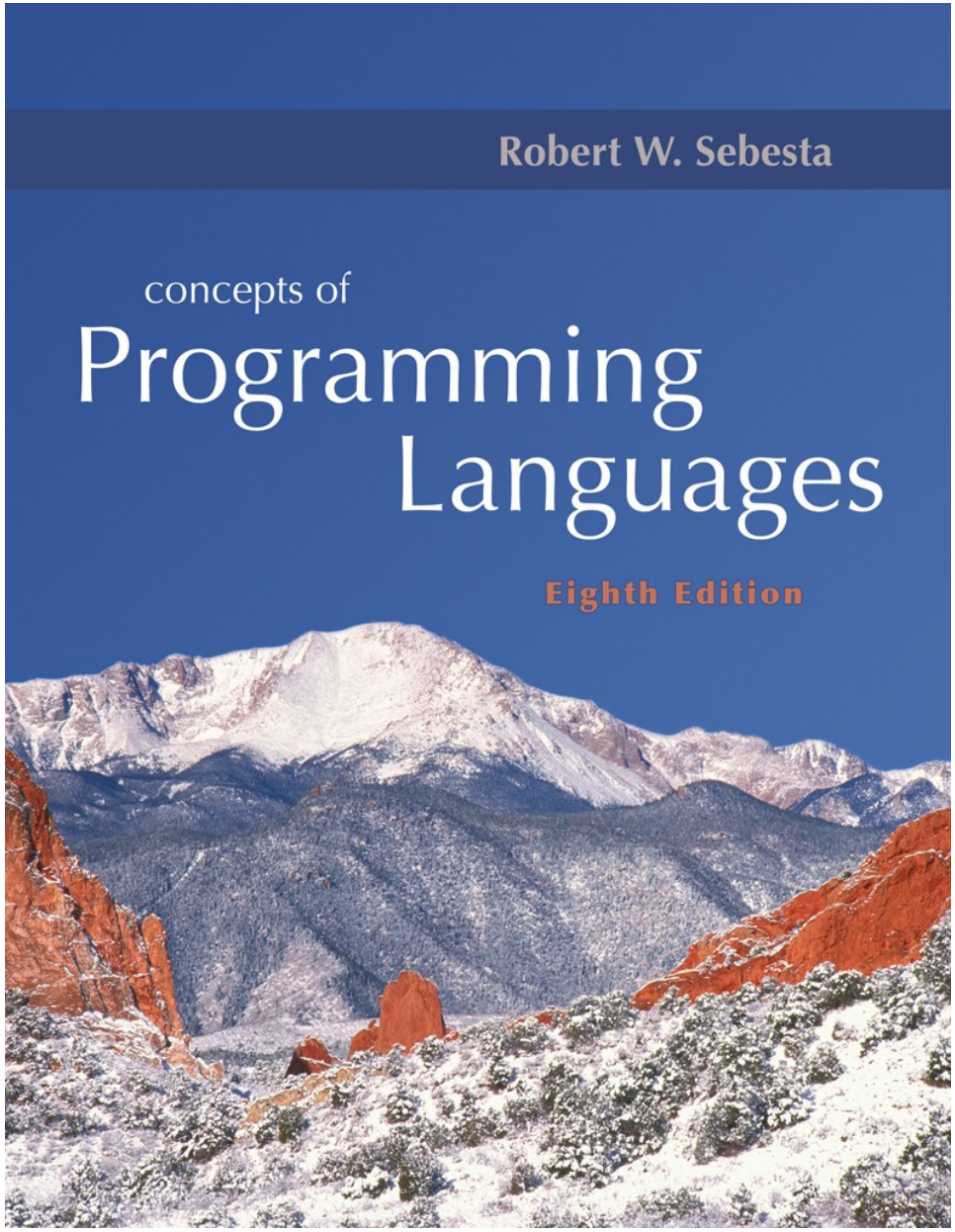
Chapter 4

Lexical and Syntax Analysis



ISBN 0-321-49362-1

Bottom-up Parsing

- The parsing problem is finding the correct RHS in a right-sentential form to reduce to get the previous right-sentential form in the derivation

Bottom-up Parsing

Definition: β is the **handle** of the right sentential form $\gamma = \alpha\beta w$ if and only if $S =>^*_{rm} \alpha A w =>_{rm} \alpha\beta w$

Definition: β is a **phrase** of the right sentential form γ if and only if $S =>^* \gamma = \alpha_1 A \alpha_2 =>^+ \alpha_1 \beta \alpha_2$

Definition: β is a **simple phrase** of the right sentential form γ if and only if $S =>^* \gamma = \alpha_1 A \alpha_2 => \alpha_1 \beta \alpha_2$

Bottom-up Parsing

- Intuición acerca de los manipuladores:
  - El manipulador de una forma sentencial más a la derecha es su frase simple más a la izquierda
  - Dada un árbol de análisis sintáctico, es fácil encontrar el manipulador

Bottom-up Parsing (cont.)

- Shift-Reduce Algorithms
  - Reduce is the action of replacing the RHS (handle) on the top of the parse stack with its corresponding LHS
  - Shift is the action of moving the next token to the top of the parse stack

Bottom-up Parsing (cont.)

- Ventajas de los analizadores LR:
  - Ellos funcionarán para casi todas las gramáticas que describen lenguajes de programación.
  - Pueden detectar errores de sintaxis lo más pronto posible en una varredura de izquierda para la derecha.
  - La clase de gramáticas LR es un superconjunto de la clase analizable por los analizadores LL (por ejemplo, muchas gramáticas recursivas a la izquierda son LR, pero ninguna es LL).

Bottom-up Parsing (cont.)

- Los analizadores LR deben construirse con una herramienta
- La idea de Knuth: un analizador sintáctico ascendente podría usar todo el historial del análisis, hasta el punto actual, para tomar decisiones de análisis.
  - Habia apenas un número relativamente pequeño de situaciones diferentes que podrían haber ocurrido, entonce la historial podría ser registrada por un estado del analizador, en una pila de análisis

Bottom-up Parsing (cont.)

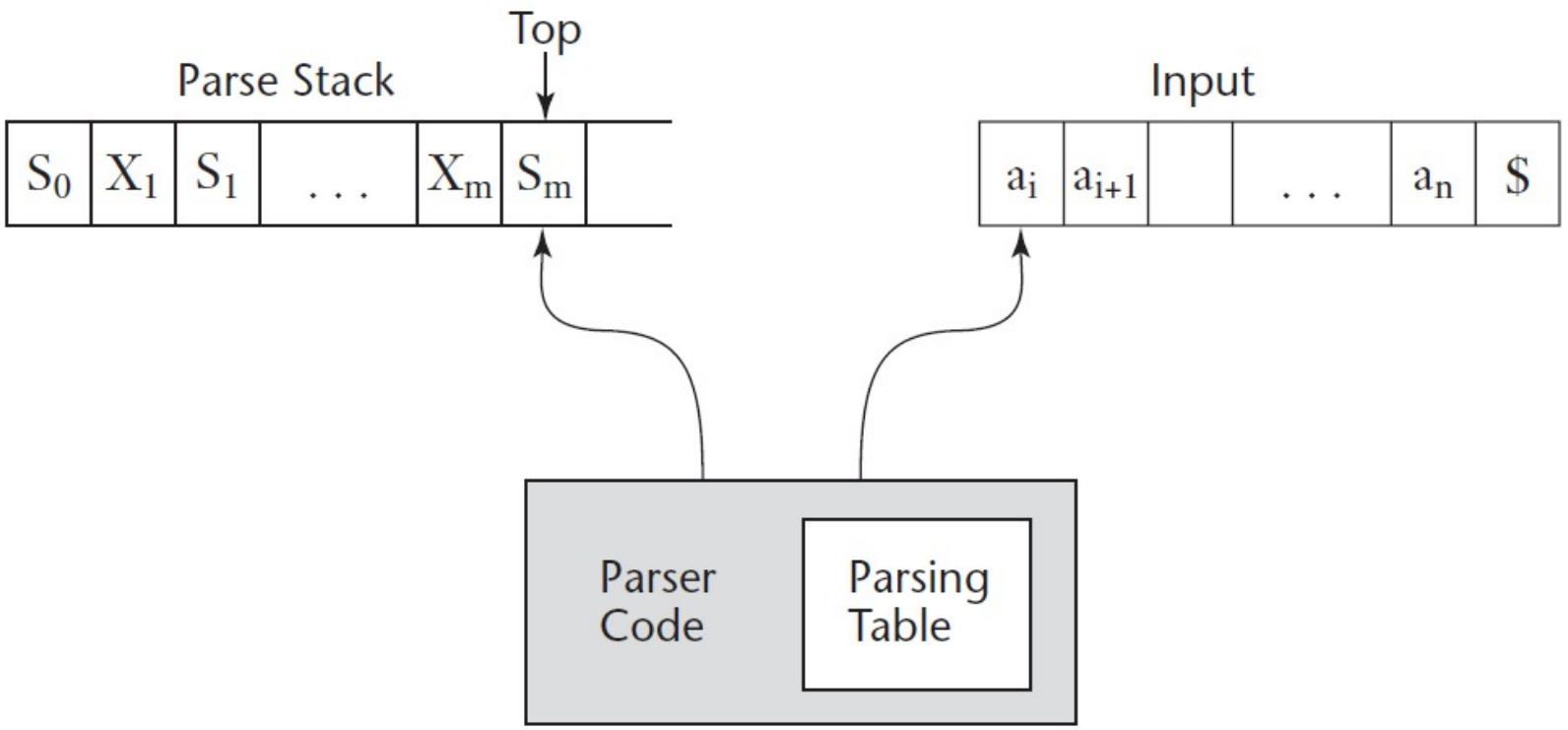
- Una configuración de una analizador sintáctico LR es un par de cadenas (pila, entrada), con la forma detallada

  $(S_0X_1S_1X_2S_2 \ldots X_mS_m, a_ia_{i+1} \ldots a_n\$)$

Bottom-up Parsing (cont.)

- Una tabla de análisis sintactico LR tiene dos partes, una tabla ACTION y una tabla GOTO
  - La tabla ACTION especifica la acción del analizador, dado el estado del analizador y el siguiente token
    - Las filas son nombres de estados; las columnas son terminales
  - La tabla GOTO indica cual estado colocar en la parte superior de la pila de análisis después de realizar una acción de reducción
    - Las filas son nombres de estado; las columnas son no terminales

La estructura de un analizador LR



Bottom-up Parsing (cont.)

- Configuración inicial: $(S_0, a_1 \ldots a_n\$)$
- Acciones del analizador:
  - Si ACTION$[S_m,a_i]$= Desplazar S, la siguiente configuración es: $(S_0X_1S_1X_2S_2 \ldots X_mS_ma_iS, a_{i+1}\ldots a_n\$)$
  - Si ACTION$[S_m,a_i]$= Reducir $A\rightarrow\beta$ y S=GOTO$[S_{m-r}, A]$, donde r=tamaño de β, la proxima configuración es $(S_0X_1S_1X_2S_2 \ldots X_{m-r}S_{m-r}AS, a_{i+1}\ldots a_n\$)$

Bottom-up Parsing (cont.)

- Parser actions (continued):
  - If ACTION[S_m, a_i] = Accept, the parse is complete and no errors were found.
  - If ACTION[S_m, a_i] = Error, the parser calls an error-handling routine.

The next slide shows the parsing table for the grammar

1. $E \rightarrow E + T$
2. $E \rightarrow T$
3. $T \rightarrow T * F$
4. $T \rightarrow F$
5. $F \rightarrow (E)$
6. $F \rightarrow id$

LR Parsing Table

	Action						Goto		
Estado	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				aceitar			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

1. $E \rightarrow E + T$
2. $E \rightarrow T$
3. $T \rightarrow T * F$
4. $T \rightarrow F$
5. $F \rightarrow (E)$
6. $F \rightarrow id$

Stack	*Input*	*Action*
0	id + id * id $	Shift 5
0id5	+ id * id $	Reduce 6 (use GOTO[0, F])
0F3	+ id * id $	Reduce 4 (use GOTO[0, T])
0T2	+ id * id $	Reduce 2 (use GOTO[0, E])
0E1	+ id * id $	Shift 6
0E1+6	id * id $	Shift 5
0E1+6id5	* id $	Reduce 6 (use GOTO[6, F])
0E1+6F3	* id $	Reduce 4 (use GOTO[6, T])
0E1+6T9	* id $	Shift 7
0E1+6T9*7	id $	Shift 5
0E1+6T9*7id5	$	Reduce 6 (use GOTO[7, F])
0E1+6T9*7F10	$	Reduce 3 (use GOTO[6, T])
0E1+6T9	$	Reduce 1 (use GOTO[0, E])
0E1	$	Accept

Bottom-up Parsing (cont.)

- A parser table can be generated from a given grammar with a tool, e.g., **`yacc`**

Resumen

- El análisis sintaxis es una parte común de la implementación del lenguaje.
- Un analizador léxico es un cazador de patrones que aísla pequeñas partes de un programa
  - Detectar errores de sintaxis
- El analizador sintáctico construye un árbol de análisis sintáctico
- Un analizador descendente recursivo es un analizador sintáctico LL
  - EBNF
- Problema del análisis sintáctico para analizadores sintácticos ascendentes: encontrar la subcadena de la forma de sentencia actual
- La familia LR de analizadores shift-reduce es el enfoque de análisis sintáctico ascendente más común