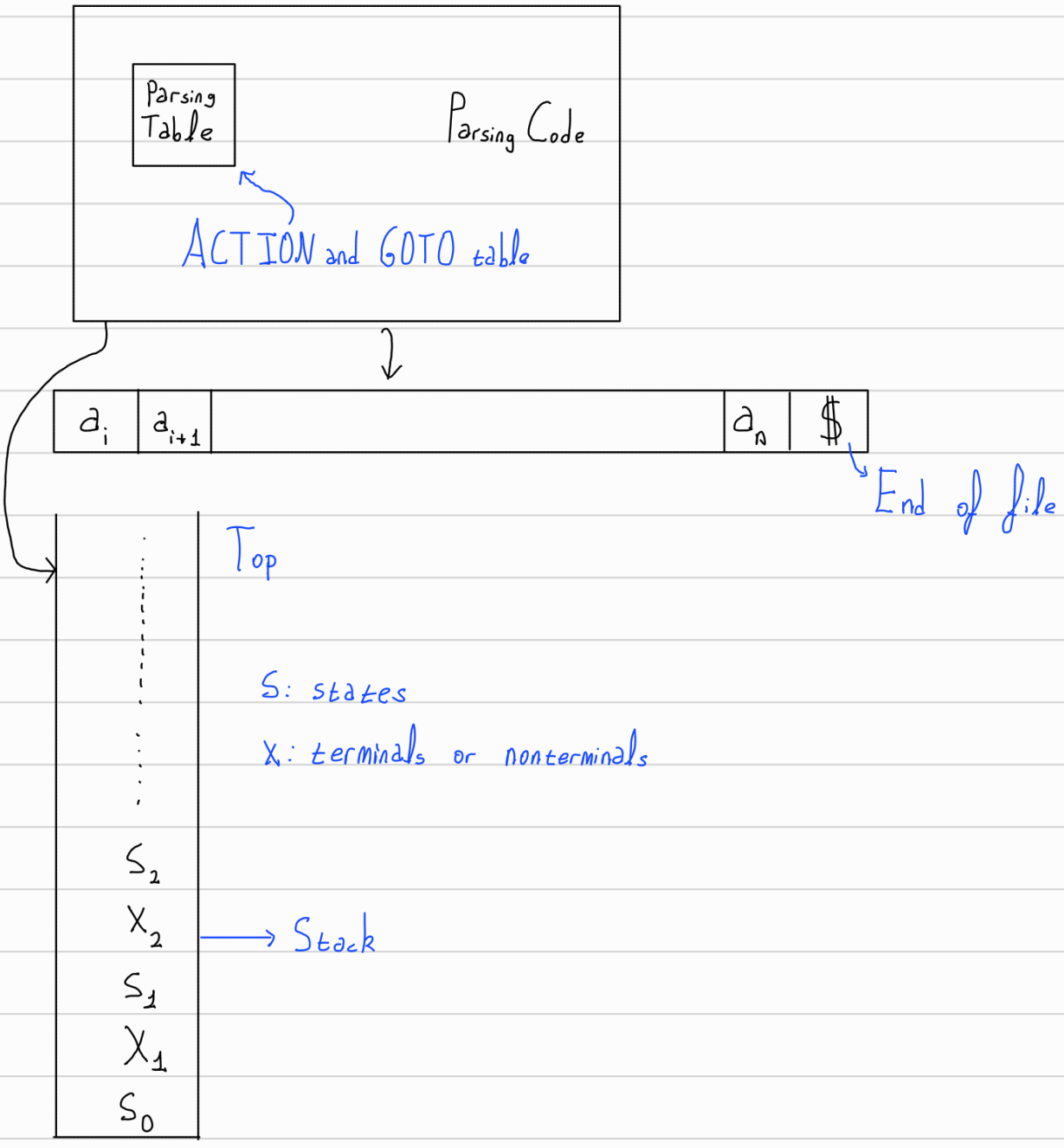
Bottom Up Parsing (LR Parsers) (O(n) de çalışıyor)

Advantages of LR parsers over LL parsers.

- For every PL an LR parser is possible
- Detects syntax errors immadiately
- LR grammers are superset of LL grammers.
- Left recursion is not a problem

Schematic Representation



Parser does (either of)

(Push) — Shift (Move next token to stack top)
— Reduce (Replace stack top)
— Error (syntax error)

There are two tables
— ACTION
— GOTO

Bottom up parser farda grammerde or bellama farklı kural vardır

Example:

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

> Bottom up parsing de kuralları ayrı yaz tablodan dolayı

Table — Action Table — GOTO table

state	id	+	*	(	)	$	_	E	T	F
0	s5			s4				1	2	3
1		s6				ACCEPT				
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	R_1				
10		R3	R3		R3	R_3				
11		R7	R5		R5	R_5				

S_x: shift et "x" nolu state geç ⟶ input değişir.
R_x: x inci kural ile reduce et ⟶ inputu değiştirmez

Ağacın hepsini tutmuyor

input id + id * id $

stack	Input	Action
0	id + id * id $	S5 (shift 5)
0 id5	+ id * id $	R6 (Reduce by 6) GOTO 3
0 F3	+ id * id $	R4, GOTO 2
0 T2	+ id * id $	R2, GOTO 1
0 E1	+ id * id $	S6
0 E1 + 6	id * id $	S5
0 E1 + 6 id5	* id $	R6, GOTO 3
0 E1 + 6 F3	* id $	R4, GOTO 9
0 E1 + 6 T9	* id $	S7 (bunu verdi)
0 E1 + 6 T9 * 7	id $	S5
0 E1 + 6 T9 * 7 id5	$	R6, GOTO 10

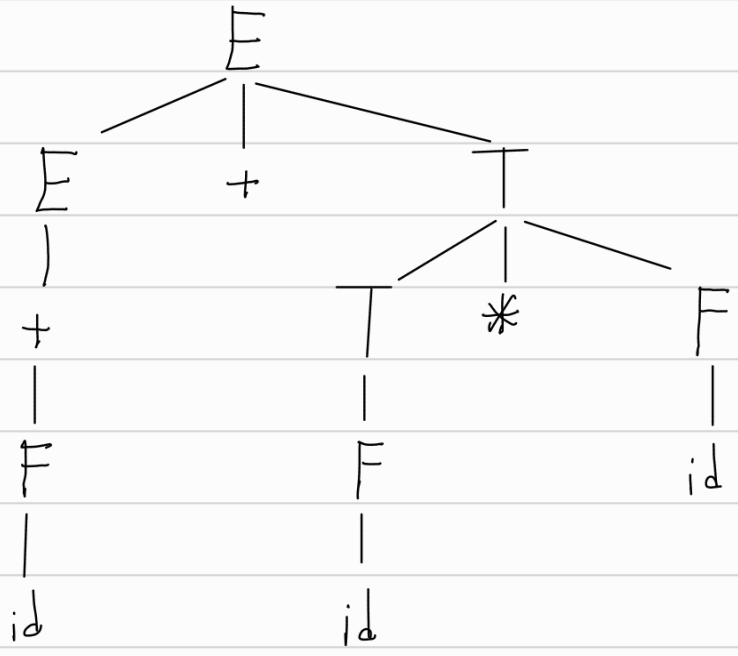
0E1+6T9*7F10 $ R3, Goto 9

rb verdi

0E1+6T9 $ R1, GOTO 1

0E1 $ ACCEPT

Tree Model

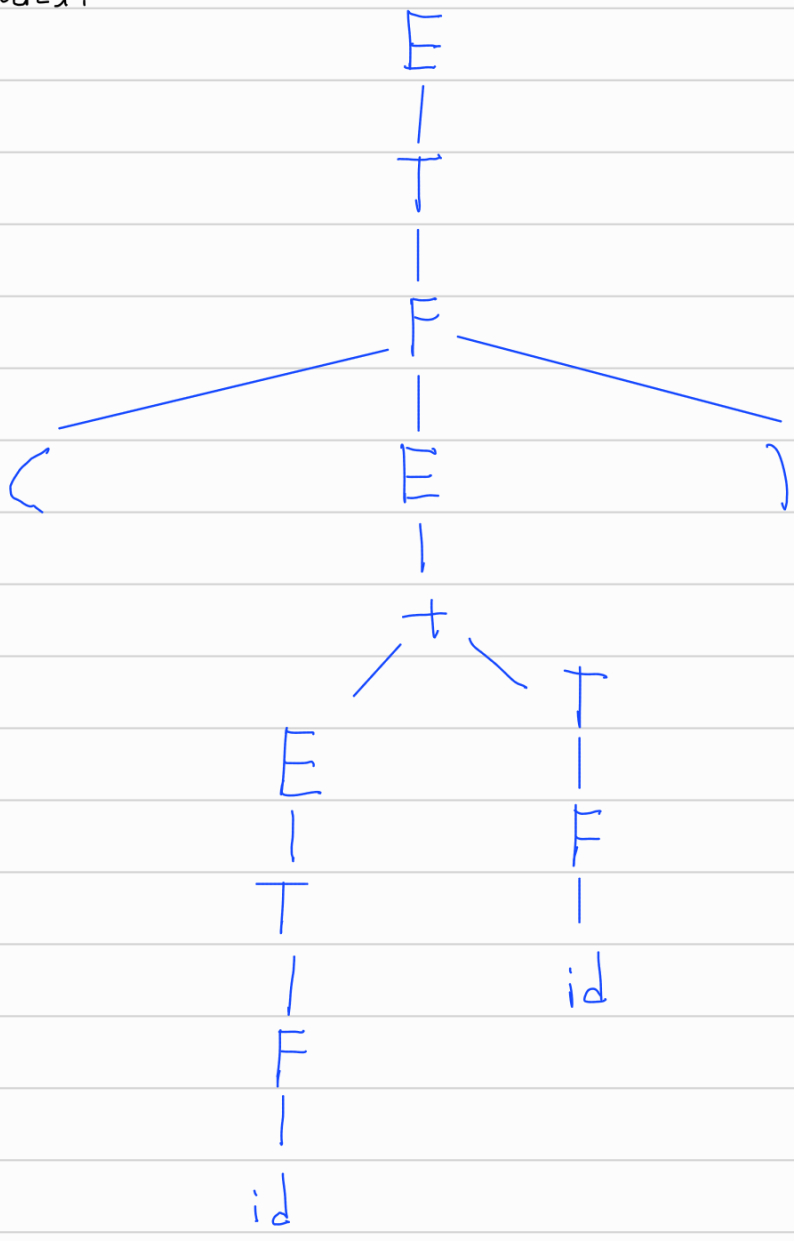


input (id+id)$

Stack	Input	Action
0	(id+id)$	S4
0(4	id+id)$	S5
0(4id5	+id)$	R6, GOTO 3
0(4F3	+id)$	R4, Goto 2
0(4T2	+id)$	R2, GOTO 8
0(4E8	+id)$	S6
0(4E8+6	id)$	S5
0(4E8+6id5	)$	R6, Goto 3
0(4E8+6F3	)$	R4, Goto 9
0(4E8+6T9	)$	R1, Goto 8
0(4E8	)$	S11
0(4E8)12	$	R5, Goto 3
0F3	$	R4, Goto 2

0 T 2	$	R2, Goto 2
0 E 1	$	Accept

Tree Model:



Ex input ()id $

stack	input	Action
0	()id$	S4
0 (4	)id $	ERROR

Modern compiler her error için ayrı bir error kodu vardır

Bottom up parserların en büyük sorunu sütun sayısı artınca tablo büyüyor

LR(2) ise sütunlar karesine çıkar.

Minimum hücre sayılı tablo üretme NP-HARD'dır.

YACC is an LP parser generator.

LR parserlar O(n) de çalışıyor.

When the grammer is ambigous, shift/reduce or reduce/reduce conflicts occurs. (there is no shift/shift conflict)