

Compiladores – Tabelas ACTION e GOTO

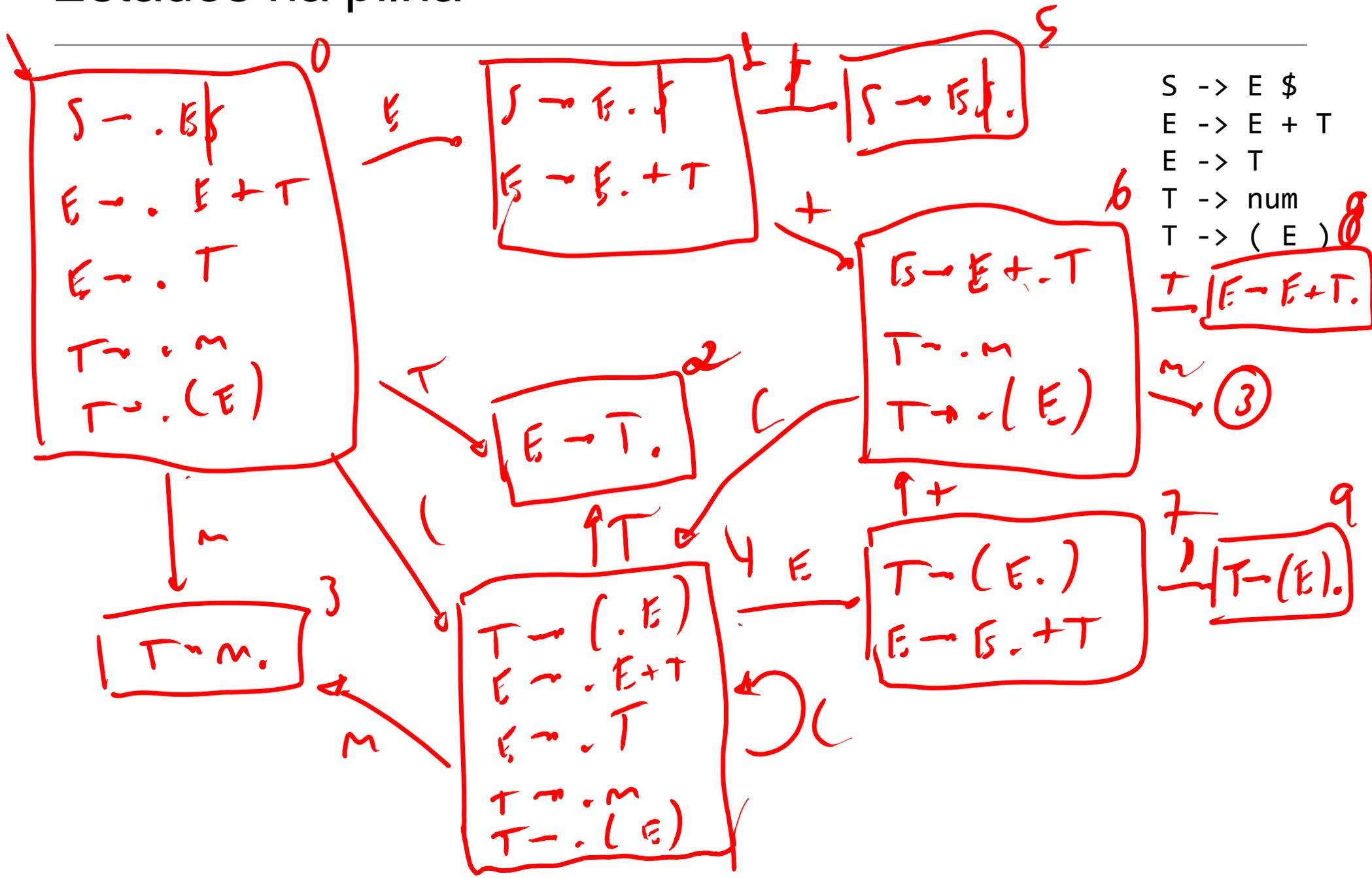
Fabio Mascarenhas – 2015.1

<http://www.dcc.ufrj.br/~fabiom/comp>

Otimizando o analisador SLR

- A implementação do analisador SLR não precisa executar o autômato em toda a pilha sempre
- Podemos associar um número de estado a cada elemento da pilha (com outra pilha, por exemplo), para ser o estado onde o autômato se encontra quando percorreu a pilha até aquele elemento
- Um shift empilha o estado resultante de fazer a transição do estado que estava no topo da pilha antes do shift
- Um reduce empilha o estado resultante de fazer a transição do estado que estava no topo da pilha depois de desempilhar o lado direito

Estados na pilha



\$0 S ✓

Analizando num + (num + num) \$

- 0 S -> E \$
- 1 E -> E + T
- 2 E -> T
- 3 T -> num
- 4 T -> (E)

\$0 | num + (num + num) \$ S

\$0 num3 | + (num + num) \$ R3
 \$0 T2 | + (num + num) \$ R2
 \$0 E2 | + (num + num) \$ S
 \$0 E2 + 6 | (num + num) \$ S
 \$0 E2 + 6 (4 | num + num) \$
 \$0 E2 + 6 (4 num3 | + num) \$ R3
 \$0 E2 + 6 (4 T2 | + num) \$ R2

\$0 E2 + 6 (4 E7 | + num) \$ S
 \$0 E2 + 6 (4 E2 + 6 | num) \$ S
 \$0 E2 + 6 (4 E7 + 6 num3 |) \$ R3
 \$0 E2 + 6 (4 E7 + 6 T8 |) \$ R2
 \$0 E2 + 6 (4 E7 |) \$ S
 \$0 E2 + 6 (4 E7) 9 | \$ R4
 \$0 E2 + 6 T8 | \$ R1
 \$0 E2 | \$ S -> \$0 E2 | \$ S | NO

Tabelas ACTION e GOTO

- Podemos construir uma grande tabela a partir do autômato, e guiar o analisador a partir dessa tabela
- As linhas são estados, as colunas símbolos (terminais e não-terminais)
- A parte da tabela dos terminais se chama ACTION
 - Ela diz o que o autômato deve fazer se o próximo token for o terminal
- A parte dos não-terminais se chama GOTO
 - Ela diz para qual estado ir após uma redução para aquele não-terminal

Preenchendo a tabela

- Para cada estado:
 - Transições em terminais viram entradas S_n para aquele terminal, onde n é o estado de destino (ACTION)
 - Transições em não-terminais viram entradas n para aquele não-terminal (GOTO)
 - Itens de redução viram entradas R_n para todos os terminais no FOLLOW do não-terminal da regra, onde n é o número de regra (ACTION)
 - Itens de redução para o símbolo inicial da gramática e o final da entrada geram entradas A , para *accept* (ACTION)

Tabelas ACTION e GOTO

	ACTION						GOTO		
	\$	+	num	()	EOF	E	T	S
0			S3	S4			1	2	
1	S5	S6							
2	R2	R2			R2				
3	R3	R3			R3				
4			S3	S4			7	2	
5						A			
6			S3	S4				8	
7		S6			S9				
8	R2	R2			R2				
9	R4	R4			R4				

- 0 S -> E \$
- 1 E -> E + T
- 2 E -> T
- 3 T -> num
- 4 T -> (E)

Analísadores LR de tabela

- Buracos na tabela indicam erros sintáticos
- Tentar adicionar uma entrada em uma célula já preenchida é um conflito, usar as regras para resolução
- Todos os métodos LR com um token de lookahead usam a mesma estrutura de tabela, o que varia é só o método de preenchimento, e o tamanho da tabela no caso da análise LR(1)
- As tabelas para analisadores LR(0), SLR e LALR de uma dada gramática têm o mesmo tamanho

AUTÔMATO LR(0)

Trecho da tabela de TINY

ACTION

	<	=	+	-	*	/			
23	R13	R13	R13	R13	R13	R13			
19	R12	R12	R12	R12	S12	S17			

id

Limitações do método SLR

$$\text{Follow}(C) = \{ \epsilon, \text{EOF} \}$$

$$\text{Follow}(V) = \{ :=, \text{EOF} \}$$

- Existem gramáticas que não são SLR:

Initial LR(0) items:

- $S \rightarrow \cdot C$
- $C \rightarrow \cdot id$
- $C \rightarrow \cdot V := E$
- $V \rightarrow \cdot id$

Shift on 'id' from initial state:

- $S \rightarrow C \cdot$

LR(0) items for non-terminal V:

- $V \rightarrow \cdot id$
- $V \rightarrow id \cdot$
- $V \rightarrow \cdot := E$
- $V \rightarrow := \cdot V$

- $S \rightarrow C$
- $C \rightarrow id$
- $C \rightarrow V := E$
- $V \rightarrow id$
- $E \rightarrow V$
- $E \rightarrow n$

Shift on 'id' from $C \rightarrow \cdot V := E$:

- $C \rightarrow id \cdot, \text{EOF}$ (conflict)
- $V \rightarrow id \cdot, :=$

conflict
R/R on EOF

Shift on 'V' from initial state:

- $C \rightarrow V \cdot := E$

Shift on 'id' from $C \rightarrow V \cdot := E$:

- $V \rightarrow id \cdot$

Shift on 'id' from $V \rightarrow \cdot := E$:

- $S \rightarrow C \rightarrow V := E \rightarrow V := V \rightarrow$
- $V := id \rightarrow id := id$

LR(0) items for non-terminal E:

- $E \rightarrow \cdot V$
- $E \rightarrow V \cdot$
- $E \rightarrow \cdot := E$
- $E \rightarrow := \cdot V$

Shift on 'id' from $E \rightarrow \cdot V$:

- $S \rightarrow C \rightarrow id$
- $S \rightarrow C \rightarrow V := E \rightarrow V := n \rightarrow id := n$

Limitações do método SLR

- Existem métodos de análise mais poderosos
- LALR associa um conjunto similar ao FOLLOW para cada item, mas mais preciso que o FOLLOW
- LR(1) e LR(k) mudam o conceito de item, gerando um autômato maior e mais preciso