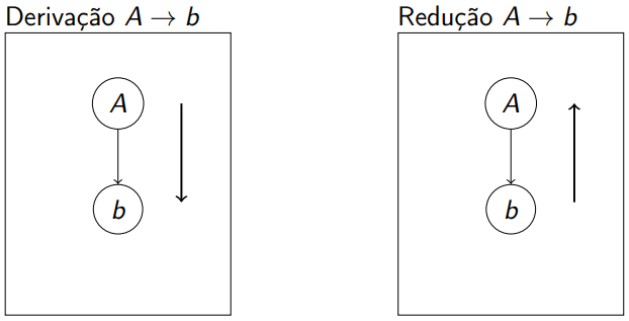
Aula 18 - Análise Sintática Ascendente - Introdução

Conceituação

- Analisadores Sintáticos Ascendentes constróem a árvore de derivação da direita para a esquerda.
- Dentre os analisadores desta categoria, destacam-se os LR (left to right parsing producing rightmost derivation), que serão alvo do estudo.
- Uma das diferenças mais importantes para os descendentes é a forma de usar a produção para construir a árvore. Por exemplo, seja a produção A → b: um analisador descendente usa derivações enquanto um analisador ascendente usa reduções



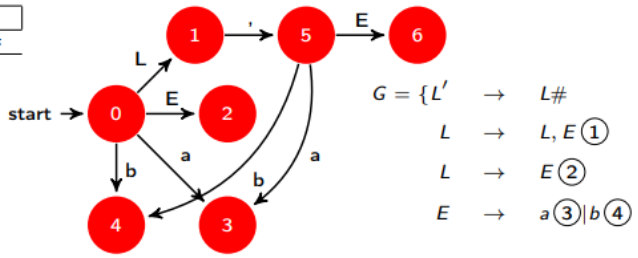
- Analisadores Ascendentes
  - Existem vários analisadores sintáticos nesta categoria: Análise de precedência simples, Análise de precedência de operadores, Análise LR (SLF(0), SLR(1), LALR(1), LR(1))
  - L(SLR(0)) ⊂ L(SLR(1)) ⊂ L(LR(1)) = L(LALR(1))

Parser LR

- Componentes
  - Entrada = contém α
  - Parser = código executável
  - Floresta = estrutura de dados auxiliar
  - Tabela ação = matriz obtida a partir de G
  - Tabela desvios = matriz obtida a partir de G
- Tabela de desvios
  - Representação de um grafo com uma tabela
  - linhas = vértices e colunas = variáveis e terminar da gramática

Tabela de Desvios

	L	E	a	b	,	#
0	1	2	3	4		
1					5	
2						
3						
4						
5		6	3	4		
6						



$G = \{L' \rightarrow L\#$
$L \rightarrow L, E\ (1)$
$L \rightarrow E\ (2)$
$E \rightarrow a\ (3) \mid b\ (4)$

- Tabela de ações
  - A tabela de ações indica o que deve ser feito em um vértice do grafo em função do token da entrada

e (empilha)
R (Reduz)
A (Aceita)
(erro)

Tabela de Ações

	a	b	,	#
0	e	e		
1			e	A
2			R_2	R_2
3			R_3	R_3
4			R_4	R_4
5	e	e		
6			R_1	R_1

  - Empilha: cria uma nova árvore com o token atual e consome o token da entrada
  - Reduz: executa a redução indicada
  - Aceita: indica que a árvore foi construída
- As tabelas são obtidas a partir da gramática

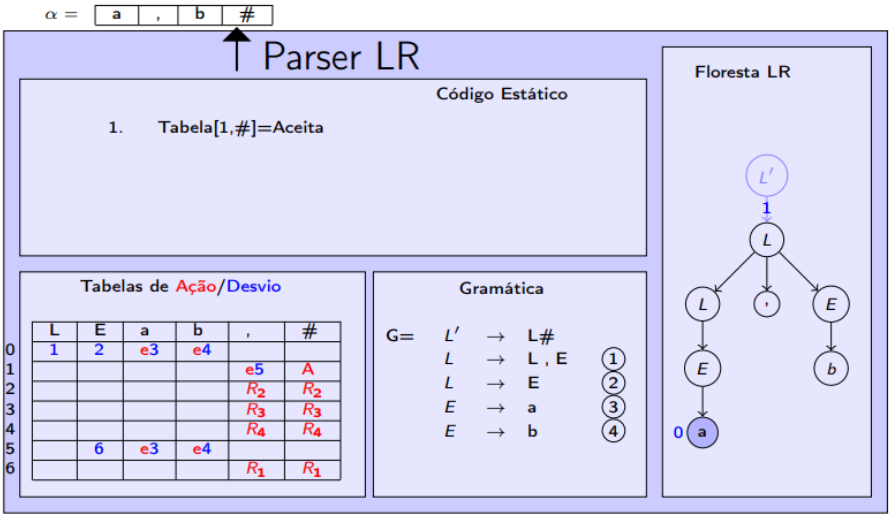
- Normalmente, as tabelas são fundidas
  - vermelho: tabela de ações;
  - azul: tabela de desvios;

$$
\begin{aligned}
G_1 = \{L' &\rightarrow L\# \\
L &\rightarrow L, E\ (1) \\
L &\rightarrow E\ (2) \\
L &\rightarrow a\ (3) \\
L &\rightarrow b\ (4)
\end{aligned}
$$

	Tabela de Ação e Desvios					
	L	E	a	b	,	#
0	1	2	e3	e4		
1					e5	A
2					R_2	R_2
3					R_3	R_3
4					R_4	R_4
5		6	e3	e4		
6					R_1	R_1

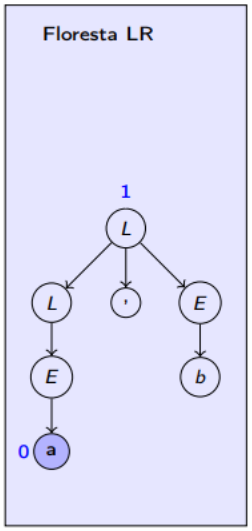
Execução

- Cada passo do parser usa como parâmetros:
  - Token de entrada
  - Floresta de sub-árvores
  - Estados: linha da tabela ou o estado indicado no topo de cada árvore
- Execução final:



Implementação

- Uma implementação prática usa uma pilha, que representa os estados de todas as árvores da floresta



Passo	Pilha	Entrada
Início	0	a,b#
1	0 3	,b#
2	0 2	,b#
3	0 1	,b#
4	0 1 5	b#
5	0 1 5 4	#
6	0 1 5 6	#
7	0 1	#