



# Compiladores

## Análise sintática ascendente

Artur Pereira <artur@ua.pt>,  
Miguel Oliveira e Silva <mos@ua.pt>

DETI, Universidade de Aveiro

Ano letivo de 2022-2023

## Sumário

- ① Introdução
- ② Conflitos
- ③ Construção de um reconhecedor
- ④ Conjunto de itens
- ⑤ Tabela de decisão de um reconhecedor ascendente

## Análise sintática ascendente

Ilustração por um exemplo

- Considere a gramática

$$D \rightarrow T L ;$$

$$T \rightarrow i \mid r$$

$$L \rightarrow v \mid L , v$$

que representa uma declaração de variáveis a la C

- Como reconhecer a palavra " $u = i v , v ;$ " como pertencente à linguagem definida pela gramática dada?
- Se  $u$  pertence à linguagem definida pela gramática, então  $D \Rightarrow^+ u$
- Gerando uma derivação à direita, tem-se  
$$D \Rightarrow TL ; \Rightarrow TL , v ; \Rightarrow T v , v ; \Rightarrow i v , v ; \checkmark$$
- Tente-se agora fazer a derivação no sentido contrário, isto é, indo de  $u$  para  $D$

## Análise sintática ascendente

Ilustração por um exemplo (cont.)

- Considere a gramática

$$D \rightarrow T L ;$$

$$T \rightarrow i \mid r$$

$$L \rightarrow v \mid L , v$$

e reduza-se a palavra " $u = i v , v ;$ " ao símbolo inicial  $D$

- 

$$i v , v ;$$

$\Leftarrow T v , v ;$  (por aplicação da produção  $T \rightarrow i$ )

*Andarmos ao contrário!*

$\Leftarrow T L , v ;$  (por aplicação da produção  $L \rightarrow v$ )

$\Leftarrow T L ;$  (por aplicação da produção  $L \rightarrow L , v$ )

$\Leftarrow D$  (por aplicação da produção  $D \rightarrow T L ;$ )

*Pensar ascendente*

- Colocando ao contrário, tem-se

$$D \Rightarrow TL ; \Rightarrow TL , v ; \Rightarrow T v , v ; \Rightarrow i v , v ;$$

que corresponde à derivação à direita da palavra " $u = i v , v ;$ "

## Análise sintática ascendente

Ilustração por um exemplo (cont.)

- A tabela seguinte mostra como, na prática, se realiza esta (retro)derivação

| pilha      | entrada      | próxima ação                          |
|------------|--------------|---------------------------------------|
| $i; v, v;$ | $i v, v; \$$ | deslocamento                          |
| $i$        | $i v, v; \$$ | redução por $T \rightarrow i$ Redução |
| $T$        | $v, v; \$$   | deslocamento                          |
| $T v$      | $, v; \$$    | redução por $L \rightarrow v$         |
| $T L$      | $, v; \$$    | deslocamento                          |
| $T L ,$    | $v; \$$      | deslocamento                          |
| $T L , v$  | $; \$$       | redução por $L \rightarrow L, v$      |
| $T L$      | $; \$$       | deslocamento                          |
| $T L ;$    | $\$$         | redução por $D \rightarrow T L ;$     |
| $D$        | $\$$         | deslocamento / aceitação              |
| $D \$$     |              | aceitação                             |

*Para reconhecer:*  
 $D \neq \Rightarrow^+ i v, v; \$$

- A palavra à entrada foi reduzida ao símbolo inicial pelo que é aceite como pertencendo à linguagem

- A aceitação pode ser feita antes de consumir o \$ ou depois

## Análise sintática ascendente

Ilustração de um erro sintático

- Veja-se a reação deste procedimento a uma entrada errada, por exemplo a palavra  $i v v ;$ .

$$\begin{aligned} D &\rightarrow T L ; \\ T &\rightarrow i \mid r \\ L &\rightarrow v \mid L, v \end{aligned}$$

| pilha   | entrada      | próxima ação                  |
|---------|--------------|-------------------------------|
|         | $i v v ; \$$ | deslocamento                  |
| $i$     | $v v ; \$$   | redução por $T \rightarrow i$ |
| $T$     | $v v ; \$$   | deslocamento                  |
| $T v$   | $(v); \$$    | redução por $L \rightarrow v$ |
| $T L$   | $v; \$$      | deslocamento                  |
| $T L v$ | $; \$$       | rejeição                      |

*Podia rejeitar aqui pois  $follow(L) \neq v$*   
*Não evoluí ...*

- Rejeita porque  $L v$  não corresponde ao prefixo de uma produção da gramática
- Na realidade, o erro poderia ter sido detetado dois passos antes, aquando da segunda redução, porque  $v \notin \text{follow}(L)$ 
  - $v$  corresponde ao símbolo à entrada
  - $L$  é o símbolo que iria aparecer no topo da pilha se se fizesse a redução por  $L \rightarrow v$

# Análise sintática ascendente

Ilustração de conflito entre deslocamento e redução

- Considera a gramática

$$S \rightarrow i \underset{c}{\overset{if}{|}} c S \xrightarrow{\text{des}} S$$

$$S \rightarrow i \underset{c}{\overset{if}{|}} c S \xrightarrow{\text{e}} S$$

$$S \rightarrow \underset{a}{\vdots} \underset{a}{\vdots}$$

isto também pode ser S

e aplique-se o procedimento anterior à palavra `icicaea`

| pilha               | entrada                | próxima ação   |
|---------------------|------------------------|--|
|                     | <code>icicaea\$</code> | deslocamento   |
| <code>i</code>      | <code>cicaea\$</code>  | deslocamento   |
| <code>ic</code>     | <code>icaea\$</code>   | deslocamento   |
| <code>ici</code>    | <code>aea\$</code>     | deslocamento   |
| <code>icic</code>   | <code>a ea\$</code>    | deslocamento   |
| <code>icica</code>  | <code>e a\$</code>     | redução por $S \rightarrow a$  |
| <code>icicaS</code> | <code>e a\$</code>     | <b>conflito:</b> <ul style="list-style-type: none"> <li>- redução por <math>S \rightarrow i c S</math></li> <li>- deslocamento para tentar <math>S \rightarrow i c S e S</math></li> </ul> |

Shift - Reduce  
erro!

normalmente este tem prioridade !

↳ Associa o else ao último shift ?

- Esta gramática representa uma estrutura típica em linguagens de programação.  
Qual?

# Análise sintática ascendente

Ilustração de conflito entre reduções

- Considera a gramática

$$S \rightarrow A$$

$$A \rightarrow c$$

$$B \rightarrow c$$

$$A \rightarrow a$$

$$B \rightarrow b$$

e aplique-se o procedimento anterior à palavra `c`

| pilha          | entrada           | próxima ação  |
|----------------|-------------------|---|
| <code>c</code> | <code>c \$</code> | deslocamento  |
| <code>c</code> | <code>\$</code>   | <b>conflito:</b> <ul style="list-style-type: none"> <li>- redução usando <math>A \rightarrow c</math></li> <li>- redução usando <math>B \rightarrow c</math></li> </ul> |

Reduce - Reduce  
erro!

# Análise sintática ascendente

Ilustração de falso conflito

- Considere a gramática

$$\begin{array}{l} S \rightarrow a \\ | \\ < S > \\ | \\ a P \\ | \\ < S > S \\ P \rightarrow < S > \\ | \\ < S > S \end{array}$$

e aplique-se o procedimento de reconhecimento à palavra “a < a > a”

| pilha | entrada                  | próxima ação  |
|-------|--------------------------|---|
| a     | a < a > a \$<br><u>P</u> | deslocamento<br>falso conflito:<br>– redução usando $S \rightarrow a$<br>– deslocamento para tentar $S \rightarrow a P$ |

- Deslocamento, porque se se optasse pela redução no topo da pilha ficaria um  $S$  e  $< \notin \text{follow}(S)$

*não é um conflito! follow é essencial*

*Apenas preciso do follow e first!*

# Análise sintática ascendente

Ilustração de falso conflito (cont.)

- Optando pelo deslocamento e continuando...

| pilha     | entrada      | próxima ação                                     |
|-----------|--------------|--|
|           | a < a > a \$ | deslocamento                                     |
| a         | < a > a \$   | deslocamento, porque $< \notin \text{follow}(S)$ |
| a <       | a > a \$     | deslocamento                                     |
| a < a     | > a \$       | redução por $S \rightarrow a$                    |
| a < S     | > a \$       | deslocamento                                     |
| a < S >   | a \$         | deslocamento, porque $a \notin \text{follow}(P)$ |
| a < S > a | \$           | redução por $S \rightarrow a$                    |
| a < S > S | \$           | redução por $P \rightarrow < S > S$              |
| a P       | \$           | redução por $S \rightarrow a P$                  |
| S         | \$           | deslocamento                                     |
| S \$      |              | aceitação  |

## Análise sintática ascendente

### Eliminação de conflito

*Resolver a ambiguidade dos if...else...*

- Pode ser possível alterar uma gramática de modo a eliminar a fonte de conflito
- Considerando que se pretendia optar pelo deslocamento, a gramática da esquerda gera a mesma linguagem que a da direita e está isenta de conflitos.

*Na gramática resolvo a ambiguidade! //*

$$\begin{array}{l}
 S \rightarrow a \\
 | \quad i \ c \ S \\
 | \quad i \ c \ S' \ e \ S \\
 \\ 
 S' \rightarrow a \\
 | \quad i \ c \ S' \ e \ S' \\
 \end{array}$$

*estou a falar que  
os if's de dentro têm else*

## Análise sintática ascendente

### if..then..else sem conflitos

- Considere a gramática seguinte e processe-se a palavra "i c i c a e a"

$$\begin{array}{l}
 S \rightarrow a \mid i \ c \ S \mid i \ c \ S' \ e \ S \\
 S' \rightarrow a \mid i \ c \ S' \ e \ S' \\
 \end{array}$$

| pilha          | entrada          | próxima ação   |
|----------------|------------------|--|
|                | i c i c a e a \$ | deslocamento   |
| i              | c i c a e a \$   | deslocamento   |
| i c            | i c a e a \$     | deslocamento   |
| i c i          | c a e a \$       | deslocamento   |
| i c i c        | a e a \$         | deslocamento   |
| i c i c a      | e a \$           | redução por $S' \rightarrow a$ // $e \in \text{follow}(S')$ , $e \notin \text{follow}(S)$  |
| i c i c S'     | e a \$           | deslocamento   |
| i c i c S' e   | a \$             | deslocamento   |
| i c i c S' e a | \$               | redução por $S \rightarrow a$ // $\$ \in \text{follow}(S)$ , $\$ \notin \text{follow}(S')$ |
| i c i c S' e S | \$               | redução por $S \rightarrow i \ c \ S' \ e \ S$   |
| i c S          | \$               | redução por $S \rightarrow i \ c \ S$  |
| S              | \$               | deslocamento e aceitação   |

- Considere a gramática seguinte e processe-se a palavra "icicaea"

$$S \rightarrow a \mid i \text{ c } S \mid i \text{ c } S' \in S$$

$$S' \rightarrow a \mid i \text{ c } S' \in S'$$

| Pilha      | Entrada    | Próxima ação   |
|------------|------------|--|
| i          | icicaea \$ | Deslocamento   |
| ie         | icae a \$  | Desl.  |
| iei        | cae a \$   | Desl.  |
| ieic       | ea a \$    | Desl.  |
| icica      | ea \$      | Redução por $S' \rightarrow a$ , pois $e \in \text{follow}(s)$ e $e \in \text{follow}(s')$     |
| icics'     | ea \$      | Desl.  |
| icic s'e   | a \$       | Desl.  |
| icic s'ea  | \$         | Redução por $S \rightarrow a$ , pois $\$ \notin \text{follow}(s')$ e $\$ \in \text{follow}(s)$ |
| icic s'e s | \$         | Redução por $S \rightarrow ic\ s'e\ s$   |
| ics        | \$         | Redução por $S \rightarrow ics$  |
| s          | \$         | Desl.  |
| s \$       |            | Aceitação  |

# Construção de um reconhecedor ascendente

## Abordagem

Quero apenas ver o último símbolo da pilha



- Como determinar de forma sistemática a ação a realizar (deslocamento, redução, aceitação, rejeição)?

Peek()  
Pop()  
Push()

| pilha | entrada  | próxima ação                  |
|-------|----------|-------------------------------|
| i     | v v ; \$ | deslocamento                  |
| T     | v v ; \$ | redução por $T \rightarrow i$ |
| T v   | v ; \$   | deslocamento                  |
| T v   | ;        | rejeição                      |

Rejeição antecipada  
follow(L) ≠ ∅

Depende do lookahead!

A pilha + entrada é o que nos faz tomar decisão

- A ação a realizar em cada passo do procedimento de reconhecimento – deslocamento, redução, aceitação ou rejeição – depende da configuração em cada momento
- Uma **configuração** é formada pelo conteúdo da pilha mais a parte da entrada ainda não processada
- A pilha é conhecida – na realidade, é preenchida pelo procedimento de reconhecimento
- Da entrada, em cada momento, apenas se conhece o *lookahead*

# Construção de um reconhecedor ascendente

## Abordagem (cont.)

| pilha | entrada  | próxima ação                  |
|-------|----------|-------------------------------|
| i     | v v ; \$ | deslocamento                  |
| T     | v v ; \$ | redução por $T \rightarrow i$ |
| T v   | v ; \$   | deslocamento                  |
| T v   | ;        | rejeição                      |

- Quantos símbolos da pilha usar?
- Poder-se-á usar apenas um?
- Se se quiser e puder construir um reconhecedor que apenas use o símbolo no topo, uma pilha onde se guardam os símbolos terminais e não terminais tem pouco interesse
- Mas pode definir-se um alfabeto adequado para a pilha
- Os símbolos a colocar na pilha devem representar estados no processo de deslocamento/redução/aceitação
- Por exemplo, um dado símbolo pode significar que, na produção " $D \rightarrow T L ;$ ", já se processou algo que corresponde ao " $T L$ ", faltando o ";"

Pedimos por um símbolo na pilha com um significado mais abrangente  
Que me diga mais do que o topo da pilha



# Construção de um reconhecedor ascendente

Itens de uma gramática

- O alfabeto da pilha representa assim o conjunto de possíveis estados nesse processo de reconhecimento
- Cada estado representa um conjunto de itens
- Cada item representa o quanto de uma produção já foi processado e o quanto ainda falta processar
  - Usa-se um ponto (·) ao longo dos símbolos de uma produção para o representar
- A produção  $A \rightarrow B_1 \ B_2 \ B_3$  produz 4 itens:
  - $A \rightarrow \cdot B_1 \ B_2 \ B_3$
  - $A \rightarrow B_1 \cdot B_2 \ B_3$
  - $A \rightarrow B_1 \ B_2 \cdot B_3$
  - $A \rightarrow B_1 \ B_2 \ B_3 \cdot$

*Resumo se lookahead = follow(A)*
- A produção  $A \rightarrow \epsilon$  produz um único item:

$$A \rightarrow \cdot \rightarrow \text{Não temha nada mas já temo tudo"}$$

*usar os itens para construir um alfabeto da pilha !!!*

- Um item com um ponto (·) à direita representa uma **ação de redução**

## Conjunto dos conjuntos de itens

Ilustração com um exemplo

- Considere a gramática

$$\begin{aligned} S &\rightarrow E \\ E &\rightarrow a \mid ( \ E \ ) \end{aligned}$$

- Reconhecer a palavra  $u = u_1 u_2 \cdots u_n$ , significa reduzir  $u\$$  a  $S\$$ , então, o estado inicial no processo de reconhecimento pode ser definido por

$$Z_0 = \{S \rightarrow \cdot E \$\}$$

*quero um símbolo no lado direito*

- O facto de o ponto (·) se encontrar imediatamente à esquerda de um símbolo significa que para se avançar no processo de reconhecimento é preciso obter esse símbolo

*Símbolo não terminal*

- Se o símbolo é terminal, isso corresponde a uma ação de deslocamento
- Se o símbolo é não terminal, é preciso dar-se a redução de uma produção que o produza

- Isso é considerado juntando ao conjunto os itens iniciais das produções cuja cabeça é o símbolo pretendido

*Resultado é um único conjunto*  $\rightarrow Z_0 = \{S \rightarrow \cdot E \$\} \cup \{E \rightarrow \cdot a, E \rightarrow \cdot ( E )\}$

*Se o ponto se encontra IMEDIATAMENTE à esquerda de um símbolo não terminal → Consumir!*

- Se aparecerem novos símbolos não terminais imediatamente à direita de um ponto (·), repete-se o processo. Faz-se o **fecho (closure)**

## Conjunto dos conjuntos de itens

Ilustração com um exemplo (cont.)

- Evolução de  $Z_0$ : *Pode nuclear* *3 evoluções possíveis!*

$$Z_0 = \{ S \rightarrow \cdot E \$ \} \cup \{ E \rightarrow \cdot a, E \rightarrow \cdot ( E ) \}$$

- O estado  $Z_0$  pode evoluir por ocorrência de um  $E$ , um  $a$  ou um  $($ , que correspondem aos símbolos que aparecem imediatamente à direita do ponto ( $\cdot$ ) *Isto funciona? Quais são os transições que têm  $E$  à direita?*

$$\delta(Z_0, E) = \{ S \rightarrow E \cdot \$ \} = Z_1 \quad \text{um estado novo}$$

$$\delta(Z_0, a) = \{ E \rightarrow a \cdot \} = Z_2 \quad \text{um estado novo}$$

$$\delta(Z_0, ()) = \{ E \rightarrow (\cdot E) \} = Z_3 \quad \text{um estado novo}$$

*Compreendo que não é o que já tinha, apenas nos pontos nucleares*

- $Z_3$  tem de ser estendido pela função de fecho, uma vez que o ponto ( $\cdot$ ) ficou imediatamente à esquerda de um símbolo não terminal ( $E$ )

$$Z_3 = \delta(Z_0, ()) = \{ E \rightarrow (\cdot E) \} \cup \{ E \rightarrow \cdot a, E \rightarrow \cdot ( E ) \} \quad z_3 \neq z_0$$

- $Z_2$ , apenas possui um item terminal (com o ponto ( $\cdot$ ) à direita), que representa uma situação passível de redução, neste caso pela produção  $E \rightarrow a$

## Conjunto dos conjuntos de itens

Ilustração com um exemplo (cont.)

- Evolução de  $Z_1$ :

$$Z_1 = \{ S \rightarrow E \cdot \$ \}$$

- Apenas evolui por ocorrência de um  $\$$

$$\delta(Z_1, \$) = \{ S \rightarrow E \$ \cdot \} \quad \Rightarrow \text{ACCEPT}$$

*Como temos lookahead  
este estado podia  
ser ignorado //*

que corresponde à situação de aceitação

- Se o símbolo inicial da gramática não aparecer no corpo de qualquer produção (como acontece aqui), Pode-se considerar  $Z_1$  como uma situação de aceitação se o lookahead for  $\$$

- Evolução de  $Z_3$ :

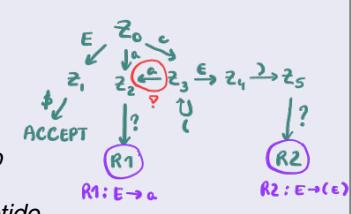
$$Z_3 = \{ E \rightarrow (\cdot E) \} \cup \{ E \rightarrow \cdot a, E \rightarrow \cdot ( E ) \}$$

- Pode evoluir por ocorrência de um  $E$ , um  $a$  ou um  $($

$$\delta(Z_3, E) = \{ E \rightarrow (E \cdot) \} = Z_4 \quad \text{um estado novo}$$

$$\delta(Z_3, a) = \{ E \rightarrow a \cdot \} = Z_2 \quad \text{um estado repetido}$$

$$\delta(Z_3, ()) = \{ E \rightarrow (\cdot E) \} = Z_3 \quad \text{um estado repetido}$$



## Conjunto dos conjuntos de itens

Ilustração com um exemplo (cont.)

- Evolução de  $Z_4$

$$Z_4 = \{ E \rightarrow (\cdot E) \}$$

- Apenas evolui por ocorrência de  $\cdot$

$$\delta(Z_4, \cdot) = \{ E \rightarrow (\cdot E) \} = Z_5 \quad \text{um estado novo}$$

- $Z_5$  apenas possui um item terminal, que representa uma situação passível de redução pela regra  $E \rightarrow (\cdot E)$

- Pode acontecer que um dado elemento (conjunto de itens) possua itens terminais (associados a reduções) e não terminais

Pode acontecer

No mesmo estado pode acontecer redução e deslocamento para outros

## Conjunto dos conjuntos de itens

Ilustração com um exemplo (cont.)

- Pondo tudo junto

$$Z_0 = \{ S \rightarrow \cdot E \$ \} \cup \{ E \rightarrow \cdot a, E \rightarrow \cdot (E) \}$$

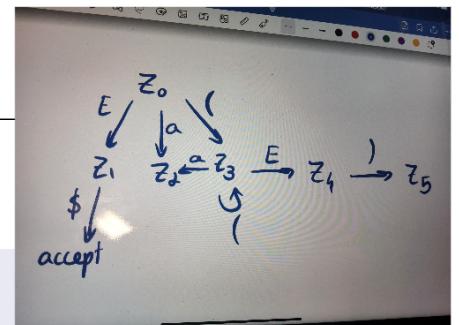
$$Z_1 = \delta(Z_0, E) = \{ S \rightarrow E \cdot \$ \}$$

$$Z_2 = \delta(Z_0, a) = \{ E \rightarrow a \cdot \}$$

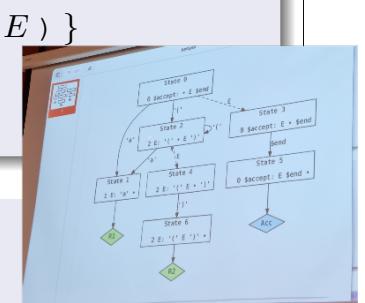
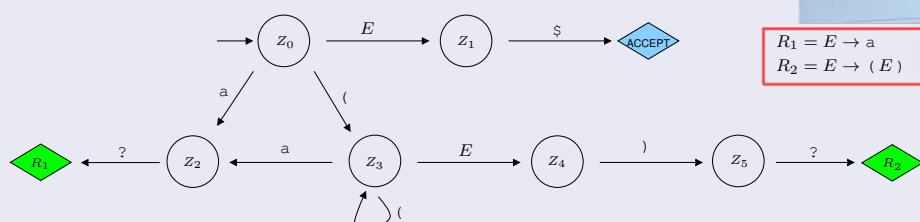
$$Z_3 = \delta(Z_0, ()) = \{ E \rightarrow (\cdot E) \} \cup \{ E \rightarrow \cdot a, E \rightarrow \cdot (E) \}$$

$$Z_4 = \delta(Z_3, E) = \{ E \rightarrow (\cdot E) \}$$

$$Z_5 = \delta(Z_4, ()) = \{ E \rightarrow (\cdot E) \}$$

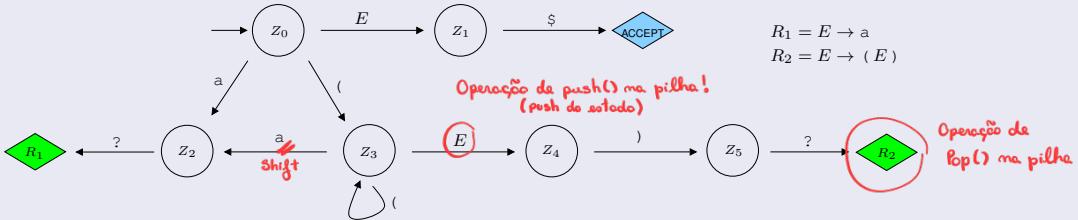


- Representando na forma de um autómato, tem-se



# Conjunto dos conjuntos de itens

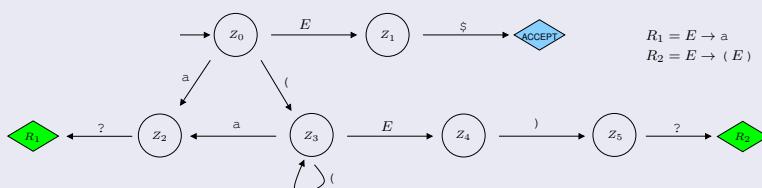
## Ilustração com um exemplo (cont.)



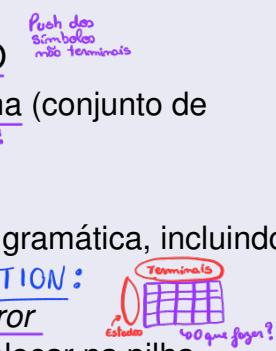
- Neste autómato, os estados representam o alfabeto da pilha
- As transições representam operações de *push*
- As transições etiquetadas com símbolos terminais representam adicionalmente ações de deslocamento (*shift*)
- As ações de redução provocam operações de *pop*, em número igual ao número de elementos do corpo da produção
- As transições etiquetadas com símbolos não terminais ocorrem após as ações de redução
- Tudo isto representa o funcionamento de um autómato de pilha que permite fazer o reconhecimento da linguagem

## Tabela de decisão de um reconhecedor ascendente

### Introdução



- O autómato de pilha pode ser implementado usando uma tabela de decisão *Shift / Reduce*
- Esta tabela contém duas matrizes, ACTION e GOTO
  - as linhas de ambas são indexadas pelo alfabeto da pilha (conjunto de conjuntos de itens) *ou seja, pelo número de estados!*
- A matriz ACTION representa ações
  - as colunas são indexadas pelos símbolos terminais da gramática, incluindo o marcador de fim de entrada (\$)
  - As células contêm as ações *shift*, *reduce*, *accept* ou *error*
  - No caso de *shift*, também inclui o próximo símbolo a colocar na pilha
- A matriz GOTO representa a operação após uma redução
  - as colunas são indexadas pelos símbolos não terminais da gramática !
  - As células indicam que valor colocar na *stack* após uma ação de redução



# Tabela de decisão de um reconhecedor ascendente

## Exemplo

- Considerese o conjunto de conjunto de itens obtido anteriormente

$$Z_0 = \{ S \rightarrow \cdot E \$ \} \cup \{ E \rightarrow \cdot a, E \rightarrow \cdot ( E ) \}$$

$$Z_1 = \delta(Z_0, E) = \{ S \rightarrow E \cdot \$ \}$$

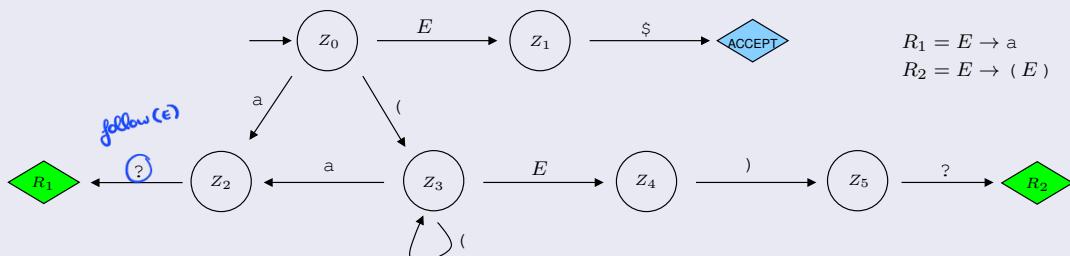
$$Z_2 = \delta(Z_0, a) = \{ E \rightarrow a \cdot \}$$

$$Z_3 = \delta(Z_0, ()) = \{ E \rightarrow (\cdot E) \} \cup \{ E \rightarrow \cdot a, E \rightarrow \cdot ( E ) \}$$

$$Z_4 = \delta(Z_3, E) = \{ E \rightarrow ( E \cdot ) \}$$

$$Z_5 = \delta(Z_4, ()) = \{ E \rightarrow ( E ) \cdot \}$$

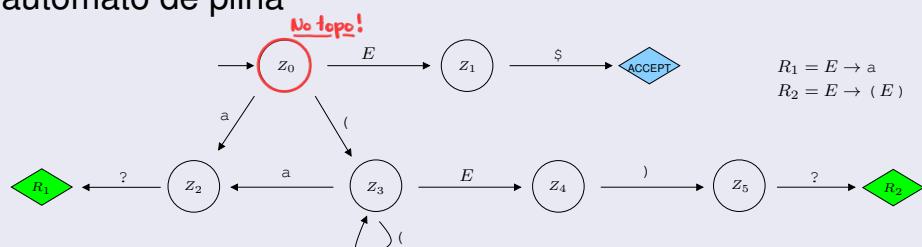
- E o correspondente autómato de pilha



# Tabela de decisão de um reconhecedor ascendente

## Exemplo

- A este autómato de pilha



- Corresponde a tabela de decisão

|       | a            | (            | )                             | \$                            |       |
|-------|--------------|--------------|-------------------------------|-------------------------------|-------|
|       | shift, $Z_2$ | shift, $Z_3$ |                               |                               | GOTO  |
| $Z_0$ |              |              |                               |                               | $E$   |
| $Z_1$ |              |              |                               | ACCEPT                        | $Z_1$ |
| $Z_2$ |              |              | reduce, $E \rightarrow a$     | reduce, $E \rightarrow a$     | $Z_4$ |
| $Z_3$ | shift, $Z_2$ | shift, $Z_3$ |                               |                               |       |
| $Z_4$ |              |              | shift, $Z_5$                  |                               |       |
| $Z_5$ |              |              | reduce, $E \rightarrow ( E )$ | reduce, $E \rightarrow ( E )$ |       |

ACTION

*Faz shift e coloca  $Z_2$  na pilha.*

*No topo!*

*reduce e moveira a redução*

*Não Terminais*

*Acontece após uma redução!*

- Com lookahead de 1, as reduções apenas são colocadas nas colunas correspondentes aos follow.  $\text{follow}(E) = \lambda, \$$

# Reconhecedor ascendente

Algoritmo de reconhecimento

|       | ACTION       |              |                             |                             | GOTO  |
|-------|--------------|--------------|-----------------------------|-----------------------------|-------|
|       | a            | (            | )                           | \$                          | E     |
| $Z_0$ | shift, $Z_2$ | shift, $Z_3$ |                             |                             | $Z_1$ |
| $Z_1$ |              |              |                             | ACCEPT                      |       |
| $Z_2$ |              |              | reduce, $E \rightarrow a$   | reduce, $E \rightarrow a$   |       |
| $Z_3$ | shift, $Z_2$ | shift, $Z_3$ |                             |                             | $Z_4$ |
| $Z_4$ |              |              | shift, $Z_5$                |                             |       |
| $Z_5$ |              |              | reduce, $E \rightarrow (E)$ | reduce, $E \rightarrow (E)$ |       |

- Com base na tabela de decisão, o procedimento de reconhecimento pode ser implementado pelo seguinte algoritmo

```

push( $Z_0$ )
forever
  if top() ==  $Z_1$  and lookahead == $
    ACCEPT
    action = ACTION[top(), lookahead]
    if action is (shift,  $Z_i$ )
      adv(); push( $Z_i$ );
    else if action is (reduce  $A \rightarrow \alpha$ )
      pop | $\alpha$ | símbolos; push(GOTO[_top_(),  $A$ ]);
    else
      !
      REJECT
  
```

- Note que após os  *pops* o símbolo no *top* pode mudar e é o novo símbolo que é usado no GOTO

# Reconhecedor ascendente

Ilustração com o exemplo anterior

|       | ACTION       |              |                             |                             | GOTO  |
|-------|--------------|--------------|-----------------------------|-----------------------------|-------|
|       | a            | (            | )                           | \$                          | E     |
| $Z_0$ | shift, $Z_2$ | shift, $Z_3$ |                             |                             | $Z_1$ |
| $Z_1$ |              |              |                             | ACCEPT                      |       |
| $Z_2$ |              |              | reduce, $E \rightarrow a$   | reduce, $E \rightarrow a$   |       |
| $Z_3$ | shift, $Z_2$ | shift, $Z_3$ |                             |                             | $Z_4$ |
| $Z_4$ |              |              | shift, $Z_5$                |                             |       |
| $Z_5$ |              |              | reduce, $E \rightarrow (E)$ | reduce, $E \rightarrow (E)$ |       |

- Aplique-se este algoritmo à palavra  $((a))$

Após a redução vais para? //

| pilha   | entrada   | próxima ação   |
|---|-----------|--|
| <del>Topo da pilha!</del> $\rightarrow [Z_0]$ | $((a))\$$ | ACTION( $Z_0, ()$ ) = (shift, $Z_3$ )  |
| $Z_0 Z_3$                                     | $(a))\$$  | ACTION( $Z_3, ()$ ) = (shift, $Z_3$ )  |
| $Z_0 Z_3 Z_3$                                 | $a))\$$   | ACTION( $Z_3, a$ ) = (shift, $Z_2$ )   |
| $Z_0 Z_3 Z_3 Z_2$                             | $)\$$     | ACTION( $Z_2, ()$ ) = (reduce $E \rightarrow a$ ) (1 pop)                    |
| $Z_0 Z_3 Z_3 Z_2$                             | $[E]$     | GOTO( $Z_3, E$ ) = $Z_4$ (push $Z_4$ ) <i>1 símbolo</i>                      |
| $Z_0 Z_3 Z_3 Z_4$                             | $)\$$     | ACTION( $Z_4, ()$ ) = (shift, $Z_5$ )  |
| $Z_0 Z_3 Z_3 Z_4 Z_5$                         | $\$$      | ACTION( $Z_5, ()$ ) = (reduce $E \rightarrow (E)$ ) (3 pops) $Z_5, Z_4, Z_3$ |
| $Z_0 Z_3 Z_4$                                 | $[E]$     | GOTO( $Z_3, E$ ) = $Z_4$ (push $Z_4$ )                                       |
| $Z_0 Z_3 Z_4$                                 | $\$$      | ACTION( $Z_4, ()$ ) = (shift, $Z_5$ )  |
| $Z_0 Z_3 Z_4 Z_5$                             | $[E]$     | ACTION( $Z_5, \$$ ) = (reduce $E \rightarrow (E)$ ) (3 pops) $Z_5, Z_4, Z_3$ |
| $Z_0 Z_1$                                     | $\$$      | ACTION( $Z_1, \$$ ) = ACCEPT   |

## Tabela de decisão de um reconhecedor ascendente

### Exemplo #3

- Q Determine-se a tabela de decisão para um reconhecedor ascendente com lookahead 1 da gramática seguinte

$$S \rightarrow a \mid (S) \mid aP \mid (S)S$$

$$P \rightarrow (S) \mid (S)S$$

- O primeiro passo corresponde a alterar a gramática de modo ao símbolo inicial não aparecer do lado direito

$$S_0 \rightarrow S$$

$$S \rightarrow a \mid (S) \mid aP \mid (S)S$$

$$P \rightarrow (S) \mid (S)S$$

## Tabela de decisão de um reconhecedor ascendente

### Exemplo #3 (cont.)

- O passo seguinte corresponde a calcular o conjunto de conjunto de itens

$$Z_0 = \{S_0 \rightarrow \cdot S \$\}$$

$$\cup \{S \rightarrow \cdot a, S \rightarrow \cdot (S), S \rightarrow \cdot aP, S \rightarrow \cdot (S)S\} \quad \text{fecho}$$

$$\delta(Z_0, S) = \{S_0 \rightarrow S \cdot \$\} = Z_1 \quad \text{um estado novo}$$

$$\delta(Z_0, a) = \{S \rightarrow a \cdot, S \rightarrow a \cdot P\} \quad \text{um estado novo}$$

$$\cup \{P \rightarrow \cdot (S), P \rightarrow \cdot (S)S\} = Z_2 \quad \text{fecho}$$

$$\delta(Z_0, ()) = \{S \rightarrow (\cdot S), S \rightarrow (\cdot S)S\} \quad \text{um estado novo}$$

$$\cup \{S \rightarrow \cdot a, S \rightarrow \cdot (S), S \rightarrow \cdot aP, S \rightarrow \cdot (S)S\} = Z_3 \quad \text{fecho}$$

$$\delta(Z_2, P) = \{S \rightarrow aP \cdot\} = Z_4 \quad \text{um estado novo}$$

$$\delta(Z_2, ()) = \{P \rightarrow (\cdot S), P \rightarrow (\cdot S)S\} \quad \text{um estado novo}$$

$$\cup \{S \rightarrow \cdot a, S \rightarrow \cdot (S), S \rightarrow \cdot aP, S \rightarrow \cdot (S)S\} = Z_5 \quad \text{fecho}$$

$$\delta(Z_3, S) = \{S \rightarrow (S \cdot), S \rightarrow (S \cdot)S\} = Z_6 \quad \text{um estado novo}$$

$$\delta(Z_3, a) = \{S \rightarrow a \cdot, S \rightarrow a \cdot P\} = Z_2 \quad \text{um estado repetido}$$

$$\delta(Z_3, ()) = \{S \rightarrow (\cdot S), S \rightarrow (\cdot S)S\} = Z_3 \quad \text{um estado repetido}$$

$$S_0 \rightarrow S$$

$$S \rightarrow a \mid (S) \mid aP \mid (S)S$$

$$P \rightarrow (S) \mid (S)S$$

Q Determine-se a tabela de decisão para um reconhecedor ascendente com lookahead 1 da gramática seguinte

$$S \rightarrow a \mid (S) \mid aP \mid (S)S$$

$$P \rightarrow (S) \mid (S)S$$

- O primeiro passo corresponde a alterar a gramática de modo ao símbolo inicial não aparecer do lado direito

$$S_0 \rightarrow S$$

$$S \rightarrow a \mid (S) \mid aP \mid (S)S$$

$$P \rightarrow (S) \mid (S)S$$

|                      |
|----------------------|
| $S_0 \rightarrow S$  |
| $S \rightarrow a$    |
| $\quad   \quad (S)$  |
| $\quad   \quad aP$   |
| $\quad   \quad (S)S$ |
| $P \rightarrow (S)$  |
| $\quad   \quad (S)S$ |

$$Z_0 = \{ S_0 \rightarrow \cdot S \$ \} \cup \{ S \rightarrow \cdot a, S \rightarrow \cdot (S), S \rightarrow \cdot aP, S \rightarrow \cdot (S)S \}$$

ACCEPT

$$\delta(Z_0, S) = \{ S_0 \rightarrow S \cdot \$ \} = Z_1$$

Estado Novo

$$\delta(Z_0, a) = \{ S \rightarrow a \cdot \} \text{ acentuo com follow}(S)$$

Fecho + Estado Novo

$$\delta(Z_0, ( ) ) = \{ S \rightarrow (\cdot S), S \rightarrow (\cdot S)S \} \cup \{ S \rightarrow \cdot a, S \rightarrow \cdot (S), S \rightarrow \cdot aP, S \rightarrow \cdot (S)S \}$$

Fecho + Estado Novo

$$\delta(Z_0, P) = \{ S \rightarrow aP \cdot \} = Z_4$$

Estado Novo

$$\delta(Z_0, C) = \{ P \rightarrow (\cdot S), P \rightarrow (\cdot S)S \} \cup \{ S \rightarrow \cdot a, S \rightarrow \cdot (S), S \rightarrow \cdot aP, S \rightarrow \cdot (S)S \} = Z_5$$

Fecho + Estado Novo

$$\delta(Z_0, S) = \{ S \rightarrow (S \cdot ), S \rightarrow (S \cdot )S \} = Z_6$$

Estado Novo

Olhemos apenas para a parte nuclear

$$\delta(Z_3, a) = \{ S \rightarrow a \cdot, S \rightarrow a \cdot P \} \cup \dots = Z_2$$

Estado Repetido

$$\delta(Z_3, ( ) ) = \{ S \rightarrow (\cdot S), S \rightarrow (\cdot S)S \} \cup \dots = Z_3$$

Estado Repetido

$$\delta(Z_5, S) = \{ P \rightarrow (S \cdot ), P \rightarrow (S \cdot )S \} = Z_7$$

Estado Novo

$$\delta(Z_5, a) = \{ S \rightarrow a \cdot, S \rightarrow a \cdot P \} \cup \dots = Z_2$$

Estado Repetido

$$\delta(Z_5, C) = \{ S \rightarrow (\cdot S), S \rightarrow (\cdot S)S \} \cup \dots = Z_3$$

Estado Repetido

$$\delta(Z_6, >) = \{ S \rightarrow (S \cdot ), S \rightarrow (S \cdot )S \} \cup \{ S \rightarrow \cdot a, \dots \} = Z_8$$

Fecho + Estado Novo

$$\delta(Z_7, >) = \{ P \rightarrow (S \cdot ), P \rightarrow (S \cdot )S \} \cup \{ S \rightarrow \cdot a, \dots \} = Z_9$$

Fecho + Estado Novo

$$\delta(Z_8, S) = \{ S \rightarrow (S \cdot ), S \cdot \} = Z_{10}$$

Estado Novo

$$\delta(Z_8, a) = \{ S \rightarrow a \cdot, S \rightarrow a \cdot P \} \cup \dots = Z_2$$

Estado Repetido

$$\delta(Z_8, C) = \{ S \rightarrow (\cdot S), (\cdot S)S \} \cup \dots = Z_3$$

Estado Repetido

$$\delta(Z_9, S) = \{ P \rightarrow (S \cdot ), S \cdot \} = Z_{11}$$

Estado Novo

$$\delta(Z_9, a) = \{ S \rightarrow a \cdot, S \rightarrow a \cdot P \} \cup \dots = Z_2$$

Estado Repetido

$$\delta(Z_9, ( ) ) = \{ S \rightarrow (\cdot S), S \rightarrow (\cdot S)S \} \cup \dots = Z_3$$

Estado Repetido

### ACTION

|          | a            | (            | ) | \$ |                                | S     | P |
|----------|--------------|--------------|---|----|--------------------------------|-------|---|
| $Z_0$    | Shift, $Z_2$ | Shift, $Z_3$ |   |    | ACCEPT                         | $Z_1$ |   |
| $Z_1$    |              |              |   |    |                                |       |   |
| $Z_2$    | Shift, $Z_5$ |              |   |    | Reduce, $S \rightarrow a$      |       |   |
| $Z_3$    |              |              |   |    | Reduce, $S \rightarrow a$      |       |   |
| $Z_4$    |              |              |   |    | follow( $S$ ) = $\{ \}, \$ \}$ |       |   |
| $Z_5$    |              |              |   |    | CUIDADO                        |       |   |
| $Z_6$    |              |              |   |    |                                |       |   |
| $Z_7$    |              |              |   |    | Mais de Kermo<br>( $\dots$ )   |       |   |
| $Z_8$    |              |              |   |    |                                |       |   |
| $Z_9$    |              |              |   |    |                                |       |   |
| $Z_{10}$ |              |              |   |    |                                |       |   |
| $Z_{11}$ |              |              |   |    |                                |       |   |

### GOTO

|                      |
|----------------------|
| $S_0 \rightarrow S$  |
| $S \rightarrow a$    |
| $\quad   \quad (S)$  |
| $\quad   \quad aP$   |
| $\quad   \quad (S)S$ |
| $P \rightarrow (S)$  |
| $\quad   \quad (S)S$ |

## Tabela de decisão de um reconhecedor ascendente

### Exemplo #3 (cont.)

- continuando, apenas mostrando os elementos envolvidos em processamento

|   |                    |
|---|--------------------|
| $Z_2 = \{S \rightarrow a \cdot, S \rightarrow a \cdot P\} \cup \dots$   |                    |
| $Z_3 = \{S \rightarrow (\cdot S), S \rightarrow (\cdot S) S\} \cup \dots$   |                    |
| $Z_5 = \{P \rightarrow (\cdot S), P \rightarrow (\cdot S) S\}$  |                    |
| $\cup \{S \rightarrow \cdot a, S \rightarrow \cdot (S), S \rightarrow \cdot a P, S \rightarrow \cdot (S) S\}$       |                    |
| $Z_6 = \{S \rightarrow (S \cdot), S \rightarrow (S \cdot) S\}$  |                    |
| $\delta(Z_5, S) = \{P \rightarrow (S \cdot), P \rightarrow (S \cdot) S\} = Z_7$                                     | um estado novo     |
| $\delta(Z_5, a) = \{S \rightarrow a \cdot, S \rightarrow a \cdot P\} = Z_2$   | um estado repetido |
| $\delta(Z_5, ()) = \{S \rightarrow (\cdot S), S \rightarrow (\cdot S) S\} = Z_3$                                    | um estado repetido |
| $\delta(Z_6, ()) = \{S \rightarrow (S \cdot), S \rightarrow (S \cdot) S\}$  | um estado novo     |
| $\cup \{S \rightarrow \cdot a, S \rightarrow \cdot (S), S \rightarrow \cdot a P, S \rightarrow \cdot (S) S\} = Z_8$ |                    |
| $\delta(Z_7, ()) = \{P \rightarrow (S \cdot), P \rightarrow (S \cdot) S\}$  | um estado novo     |
| $\cup \{S \rightarrow \cdot a, S \rightarrow \cdot (S), S \rightarrow \cdot a P, S \rightarrow \cdot (S) S\} = Z_9$ |                    |

---

$S_0 \rightarrow S$   
 $S \rightarrow a \mid (S) \mid a P \mid (S) S$   
 $P \rightarrow (S) \mid (S) S$

## Tabela de decisão de um reconhecedor ascendente

### Exemplo #3 (cont.)

- continuando...

|   |                    |
|---|--------------------|
| $Z_2 = \{S \rightarrow a \cdot, S \rightarrow a \cdot P\} \cup \dots$   |                    |
| $Z_3 = \{S \rightarrow (\cdot S), S \rightarrow (\cdot S) S\} \cup \dots$                                     |                    |
| $Z_8 = \{S \rightarrow (S \cdot), S \rightarrow (S \cdot) S\}$  |                    |
| $\cup \{S \rightarrow \cdot a, S \rightarrow \cdot (S), S \rightarrow \cdot a P, S \rightarrow \cdot (S) S\}$ |                    |
| $Z_9 = \{P \rightarrow (S \cdot), P \rightarrow (S \cdot) S\}$  |                    |
| $\cup \{S \rightarrow \cdot a, S \rightarrow \cdot (S), S \rightarrow \cdot a P, S \rightarrow \cdot (S) S\}$ |                    |
| $\delta(Z_8, S) = S \rightarrow (S \cdot) S \cdot \} = Z_{10}$  | um estado novo     |
| $\delta(Z_8, a) = \{S \rightarrow a \cdot, S \rightarrow a \cdot P\} = Z_2$                                   | um estado repetido |
| $\delta(Z_8, ()) = \{S \rightarrow (\cdot S), S \rightarrow (\cdot S) S\} = Z_3$                              | um estado repetido |
| $\delta(Z_9, S) = \{P \rightarrow (S \cdot) S \cdot\} = Z_{11}$   | um estado novo     |
| $\delta(Z_9, a) = \{S \rightarrow a \cdot, S \rightarrow a \cdot P\} = Z_2$                                   | um estado repetido |
| $\delta(Z_9, ()) = \{S \rightarrow (\cdot S), S \rightarrow (\cdot S) S\} = Z_3$                              | um estado repetido |

---

$S_0 \rightarrow S$   
 $S \rightarrow a \mid (S) \mid a P \mid (S) S$   
 $P \rightarrow (S) \mid (S) S$

## Tabela de decisão de um reconhecedor ascendente

### Exemplo #3 (cont.)

- O que resulta em

|   |                           |                        |                         |
|---|---------------------------|------------------------|-------------------------|
| $Z_0 = \{S_0 \rightarrow \cdot S \$\} \cup \dots$                         | $\delta(Z_0, S) = Z_1$    | $\delta(Z_0, a) = Z_2$ | $\delta(Z_0, ()) = Z_3$ |
| $Z_1 = \{S_0 \rightarrow S \cdot \$\}$                                    |                           |                        |                         |
| $Z_2 = \{S \rightarrow a \cdot, S \rightarrow a \cdot P\} \cup \dots$     |                           | $\delta(Z_2, P) = Z_4$ | $\delta(Z_2, ()) = Z_5$ |
| $Z_3 = \{S \rightarrow (\cdot S), S \rightarrow (\cdot S) S\} \cup \dots$ | $\delta(Z_3, S) = Z_6$    | $\delta(Z_3, a) = Z_2$ | $\delta(Z_3, ()) = Z_3$ |
| $Z_4 = \{S \rightarrow a P \cdot\}$                                       |                           |                        |                         |
| $Z_5 = \{P \rightarrow (\cdot S), P \rightarrow (\cdot S) S\} \cup \dots$ | $\delta(Z_5, S) = Z_7$    | $\delta(Z_5, a) = Z_2$ | $\delta(Z_5, ()) = Z_3$ |
| $Z_6 = \{S \rightarrow (S \cdot), S \rightarrow (S \cdot) S\}$            |                           |                        | $\delta(Z_6, ()) = Z_8$ |
| $Z_7 = \{P \rightarrow (S \cdot), P \rightarrow (S \cdot) S\}$            |                           |                        | $\delta(Z_7, ()) = Z_9$ |
| $Z_8 = \{S \rightarrow (S) \cdot, S \rightarrow (S) \cdot S\} \cup \dots$ | $\delta(Z_8, S) = Z_{10}$ | $\delta(Z_8, a) = Z_2$ | $\delta(Z_8, ()) = Z_3$ |
| $Z_9 = \{P \rightarrow (S) \cdot, P \rightarrow (S) \cdot S\} \cup \dots$ | $\delta(Z_9, S) = Z_{11}$ | $\delta(Z_9, a) = Z_2$ | $\delta(Z_9, ()) = Z_3$ |
| $Z_{10} = \{S \rightarrow (S) S \cdot\}$                                  |                           |                        |                         |
| $Z_{11} = \{P \rightarrow (S) S \cdot\}$                                  |                           |                        |                         |

$S_0 \rightarrow S$   
 $S \rightarrow a \mid (S) \mid a P \mid (S) S$   
 $P \rightarrow (S) \mid (S) S$

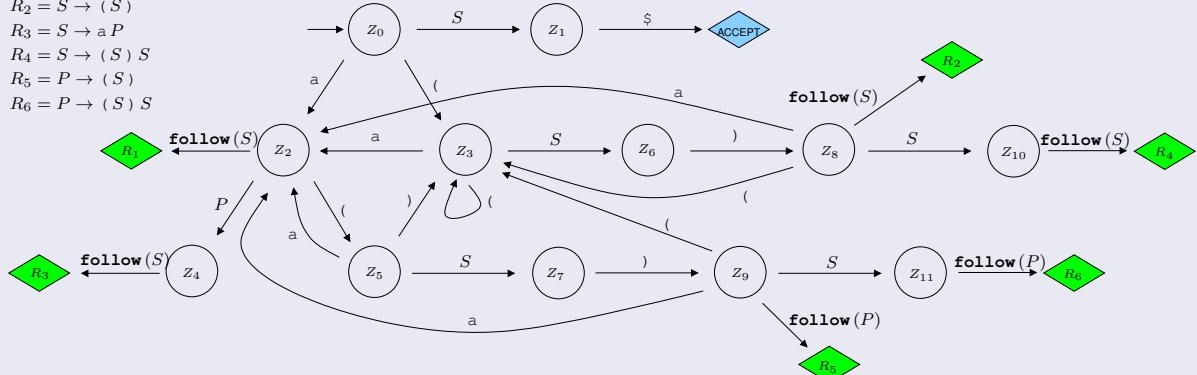
## Tabela de decisão de um reconhecedor ascendente

### Exemplo #3 (cont.)

- O que resulta em

|   |                           |                        |                         |
|---|---------------------------|------------------------|-------------------------|
| $Z_0 = \{S_0 \rightarrow \cdot S \$\} \cup \dots$                         | $\delta(Z_0, S) = Z_1$    | $\delta(Z_0, a) = Z_2$ | $\delta(Z_0, ()) = Z_3$ |
| $Z_1 = \{S_0 \rightarrow S \cdot \$\}$                                    |                           |                        |                         |
| $Z_2 = \{S \rightarrow a \cdot, S \rightarrow a \cdot P\} \cup \dots$     |                           | $\delta(Z_2, P) = Z_4$ | $\delta(Z_2, ()) = Z_5$ |
| $Z_3 = \{S \rightarrow (\cdot S), S \rightarrow (\cdot S) S\} \cup \dots$ | $\delta(Z_3, S) = Z_6$    | $\delta(Z_3, a) = Z_2$ | $\delta(Z_3, ()) = Z_3$ |
| $Z_4 = \{S \rightarrow a P \cdot\}$                                       |                           |                        |                         |
| $Z_5 = \{P \rightarrow (\cdot S), P \rightarrow (\cdot S) S\} \cup \dots$ | $\delta(Z_5, S) = Z_7$    | $\delta(Z_5, a) = Z_2$ | $\delta(Z_5, ()) = Z_3$ |
| $Z_6 = \{S \rightarrow (S \cdot), S \rightarrow (S \cdot) S\}$            |                           |                        | $\delta(Z_6, ()) = Z_8$ |
| $Z_7 = \{P \rightarrow (S \cdot), P \rightarrow (S \cdot) S\}$            |                           |                        | $\delta(Z_7, ()) = Z_9$ |
| $Z_8 = \{S \rightarrow (S) \cdot, S \rightarrow (S) \cdot S\} \cup \dots$ | $\delta(Z_8, S) = Z_{10}$ | $\delta(Z_8, a) = Z_2$ | $\delta(Z_8, ()) = Z_3$ |
| $Z_9 = \{P \rightarrow (S) \cdot, P \rightarrow (S) \cdot S\} \cup \dots$ | $\delta(Z_9, S) = Z_{11}$ | $\delta(Z_9, a) = Z_2$ | $\delta(Z_9, ()) = Z_3$ |
| $Z_{10} = \{S \rightarrow (S) S \cdot\}$                                  |                           |                        |                         |
| $Z_{11} = \{P \rightarrow (S) S \cdot\}$                                  |                           |                        |                         |

$R_1 = S \rightarrow a$   
 $R_2 = S \rightarrow (S)$   
 $R_3 = S \rightarrow a P$   
 $R_4 = S \rightarrow (S) S$   
 $R_5 = P \rightarrow (S)$   
 $R_6 = P \rightarrow (S) S$



## Tabela de decisão de um reconhecedor ascendente

Exemplo #3 (cont.)

- E finalmente a tabela de decisão

|          | a                           | (                           | )                           | \$                        | S        | P     |
|----------|-----------------------------|-----------------------------|-----------------------------|---------------------------|----------|-------|
| $Z_0$    | <i>shift, Z<sub>2</sub></i> | <i>shift, Z<sub>3</sub></i> |                             |                           | $Z_1$    |       |
| $Z_1$    |                             |                             |                             | <i>ACCEPT</i>             |          |       |
| $Z_2$    |                             | <i>shift, Z<sub>5</sub></i> | <i>reduce S → a</i>         | <i>reduce S → a</i>       |          | $Z_4$ |
| $Z_3$    | <i>shift, Z<sub>2</sub></i> | <i>shift, Z<sub>3</sub></i> |                             |                           |          | $Z_6$ |
| $Z_4$    |                             |                             | <i>reduce S → a P</i>       | <i>reduce S → a P</i>     |          |       |
| $Z_5$    | <i>shift, Z<sub>2</sub></i> | <i>shift, Z<sub>3</sub></i> |                             |                           |          | $Z_7$ |
| $Z_6$    |                             |                             | <i>shift, Z<sub>8</sub></i> |                           |          |       |
| $Z_7$    |                             |                             | <i>shift, Z<sub>9</sub></i> |                           |          |       |
| $Z_8$    | <i>shift, Z<sub>2</sub></i> | <i>shift, Z<sub>3</sub></i> | <i>reduce S → ( S )</i>     | <i>reduce S → ( S )</i>   | $Z_{10}$ |       |
| $Z_9$    | <i>shift, Z<sub>2</sub></i> | <i>shift, Z<sub>3</sub></i> | <i>reduce P → ( S )</i>     | <i>reduce P → ( S )</i>   | $Z_{11}$ |       |
| $Z_{10}$ |                             |                             | <i>reduce S → ( S ) S</i>   | <i>reduce S → ( S ) S</i> |          |       |
| $Z_{11}$ |                             |                             | <i>reduce P → ( S ) S</i>   | <i>reduce P → ( S ) S</i> |          |       |

## Tabela de decisão de um reconhecedor ascendente

Exercício

Q Determine-se a tabela de decisão para um reconhecedor ascendente com *lookahead* 1 da gramática seguinte

$$S \rightarrow \epsilon \mid S B \text{ a} \mid S A \text{ b}$$

$$A \rightarrow \text{a} \mid A A \text{ b}$$

$$B \rightarrow B B \text{ a} \mid \text{b}$$