Compiladores – ACTION e GOTO

Fabio Mascarenhas – 2017.2

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

Otimizando o analisador SLR

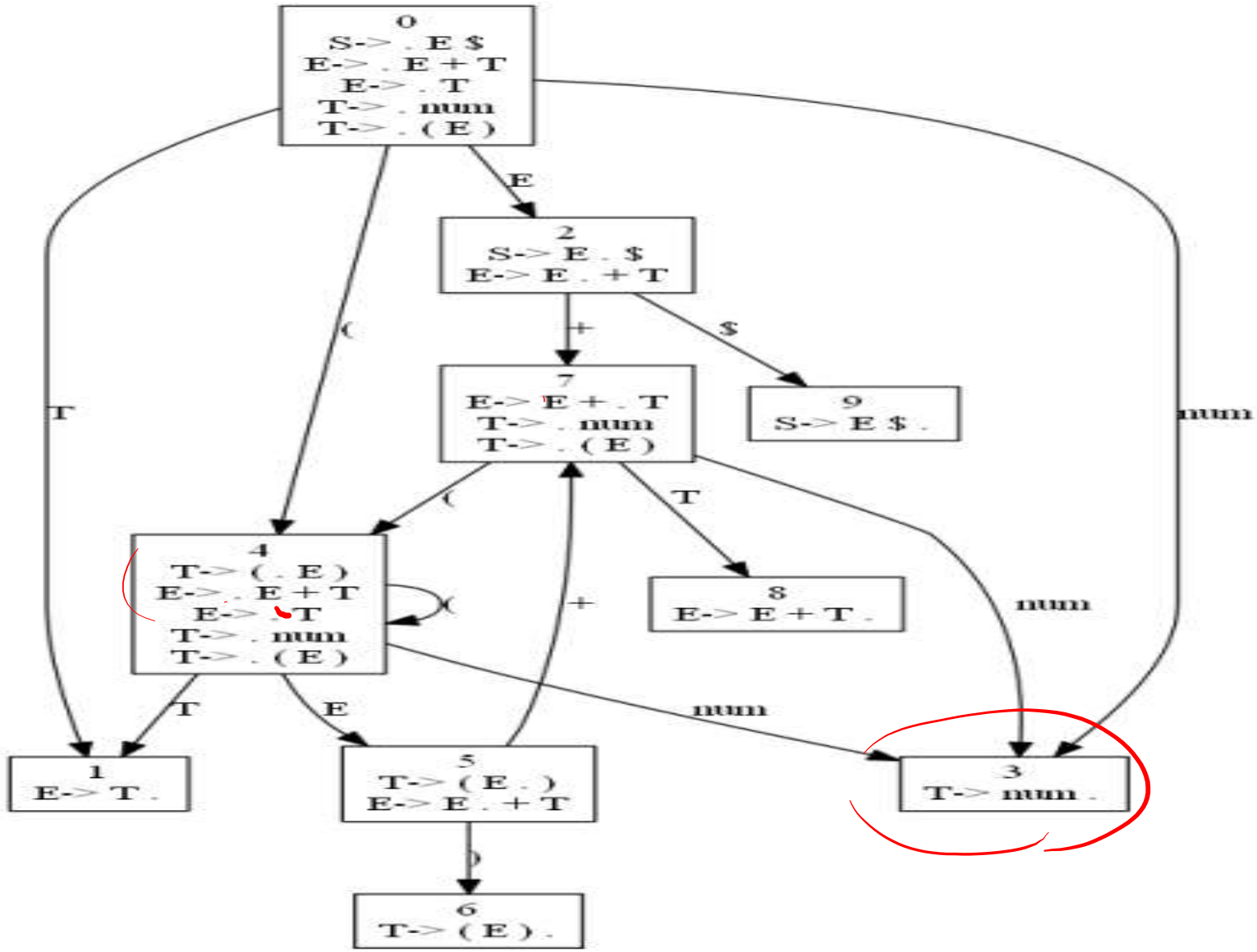
- A implementação do analisador SLR não precisa executar o autômato em toda a pilha sempre

- 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

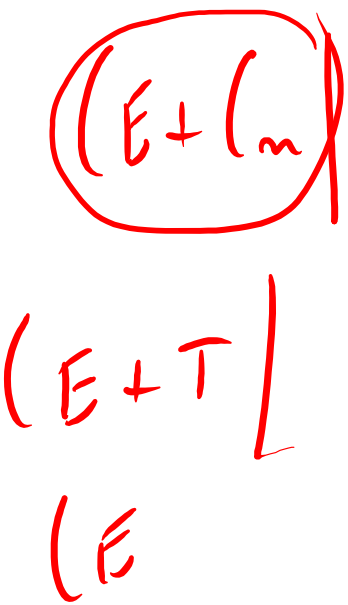
- Um shift empilha o estado resultante de fazer a transição do estado que estava no topo da pilha antes do shift

- Um reduce empilha o estado resultante de fazer a transição do estado que estava no topo da pilha depois de desempilhar o lado direito

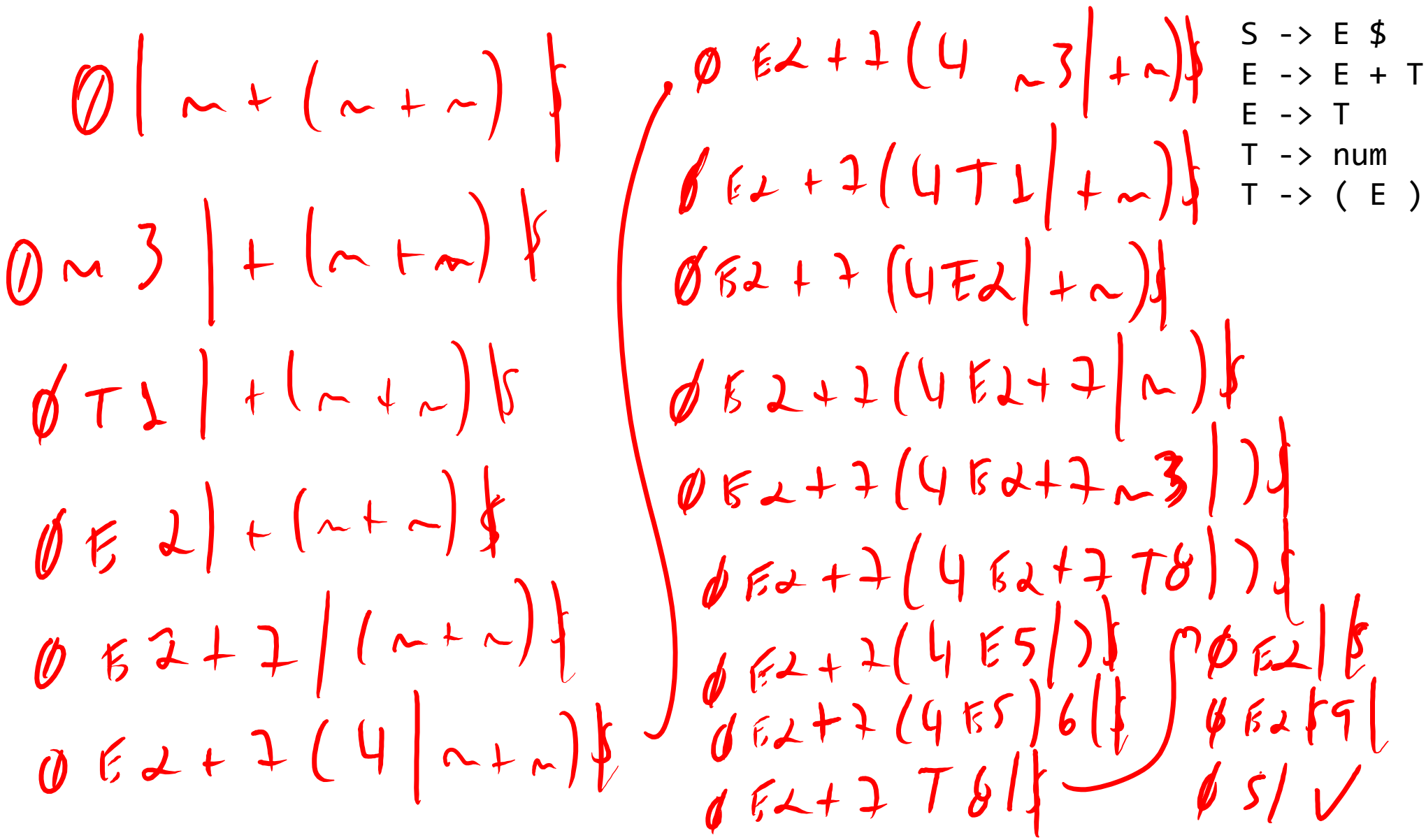
Estados na pilha




```
S -> E $
E -> E + T
E -> T
T -> num
T -> ( E )
```



Analisando num + (num + num) $



```
S -> E $
E -> E + T
E -> T
T -> num
T -> ( E )
```

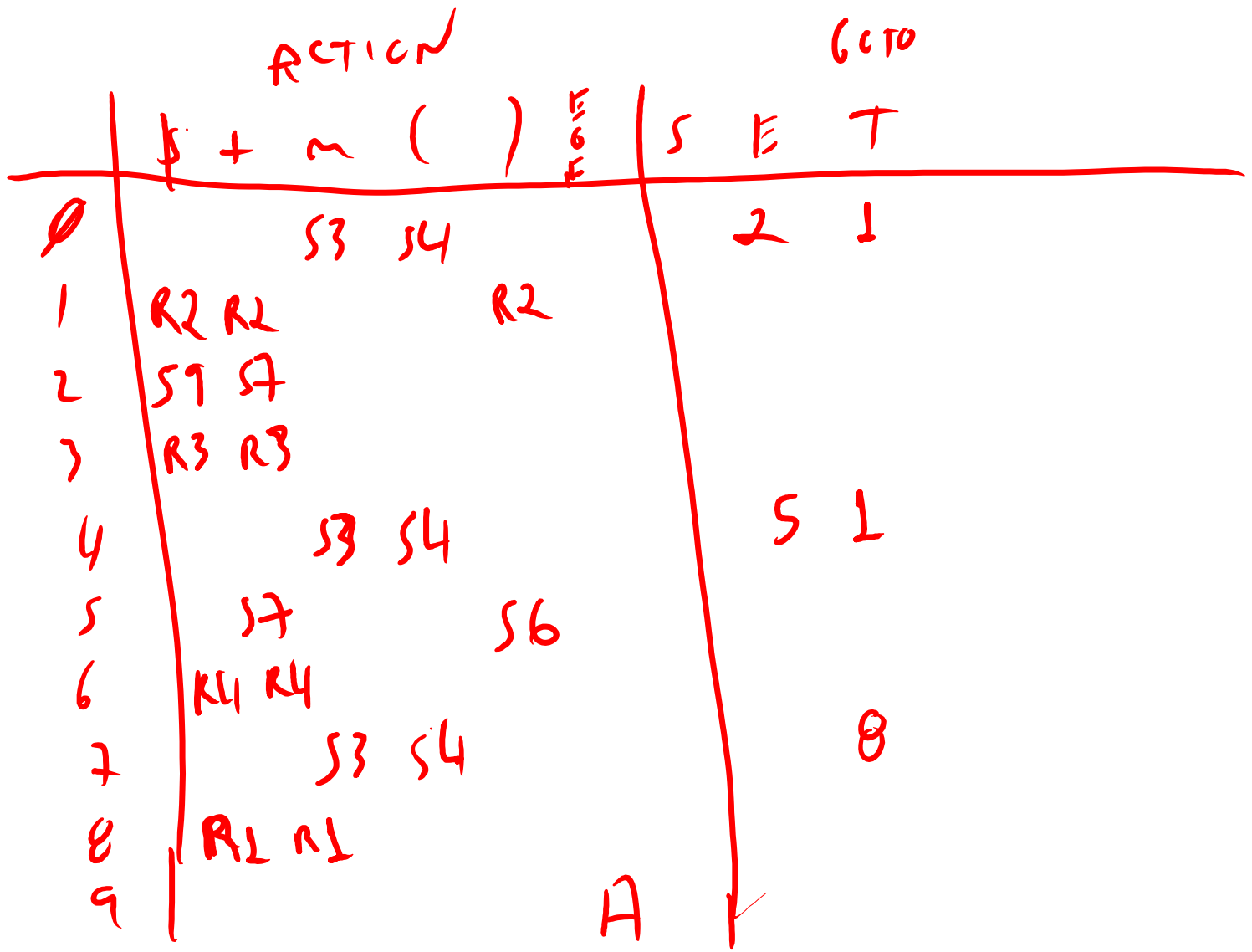
Tabelas ACTION e GOTO

- 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 *Sn* 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 *Rn* 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

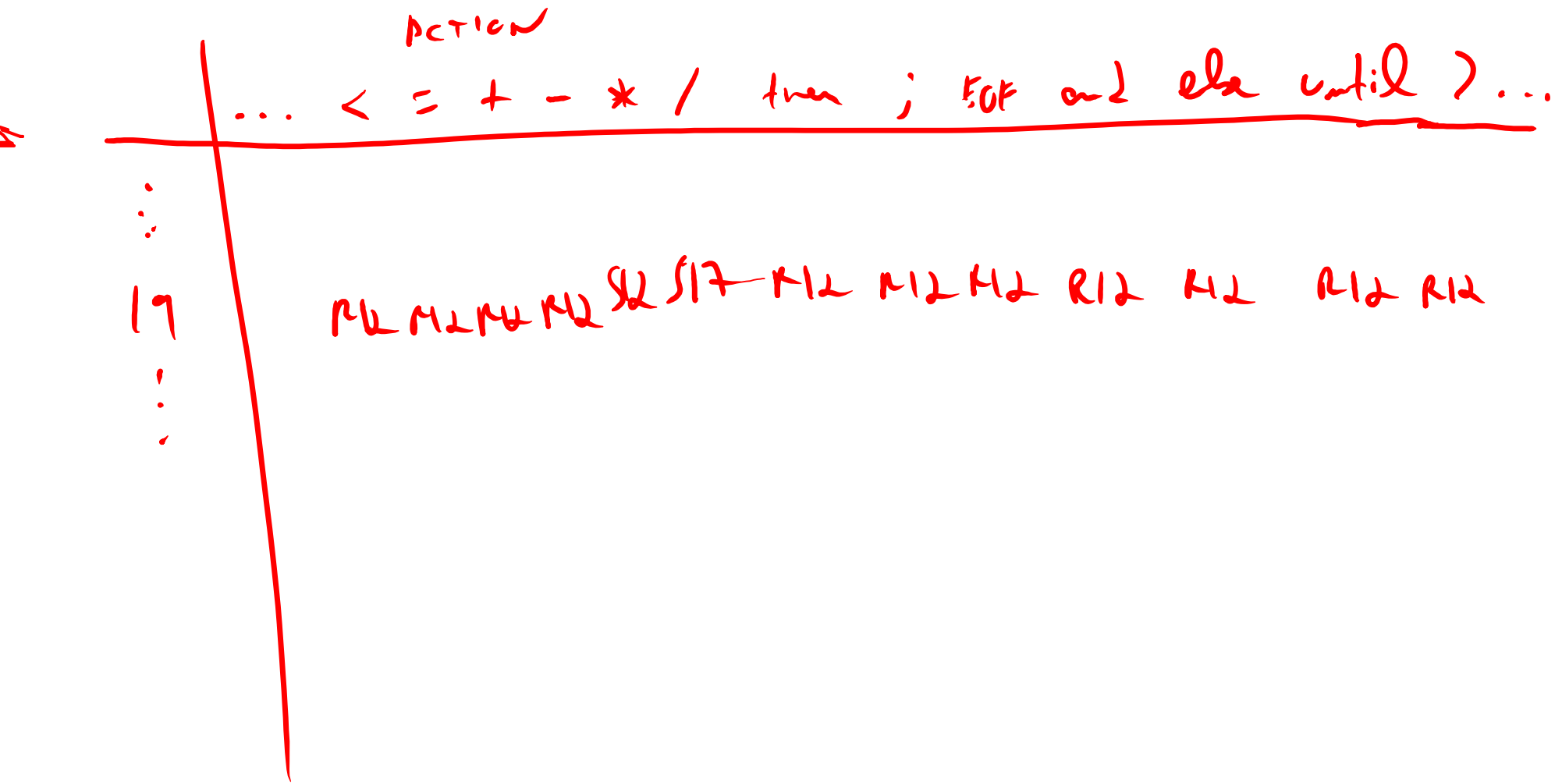


0 S -> E $
1 E -> E + T
2 E -> T
3 T -> num
4 T -> (E)

Analisadores 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

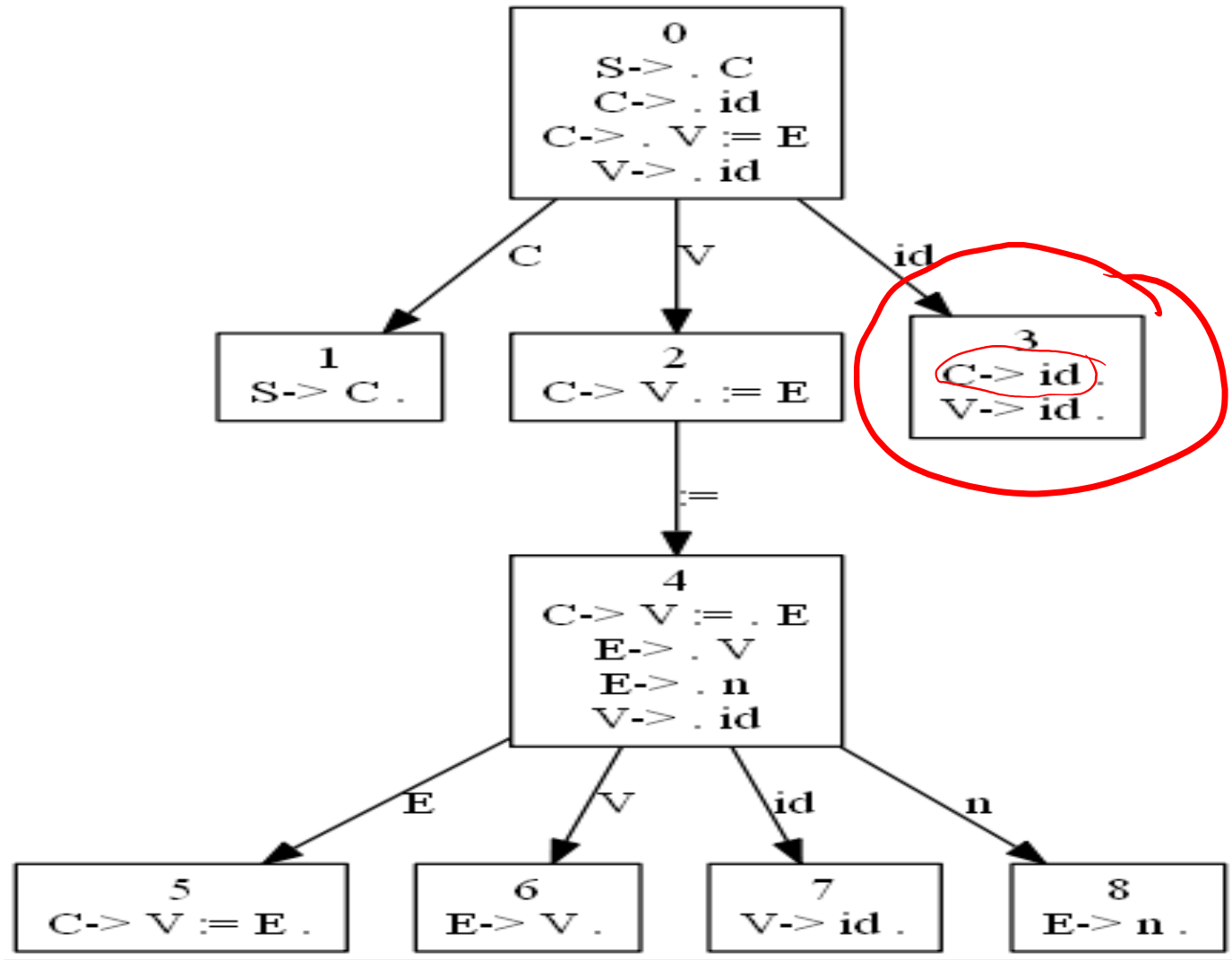
Trecho da tabela de TINY



Limitações do método SLR

- Existem gramáticas que não são SLR:

```
S -> C
C -> id
C -> V := E
V -> id
E -> V
E -> n
```



follow(C) = {e.f}

follow(V) = {:=, e.f}

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

LR(0) < SLR < LALR < LR(1) < LR(k)

∨

LL(1)