Análise Sintática Determinista

Linguagens Formais e Autómatos

Francisco Coelho
fc@di.uevora.pt

Departamento de Informática
Escola de Ciências e Tecnologia
Universidade de Évora

UNIVERSIDADE DE ÉVORA

UNIVERSIDADE DE ÉVORA

Análise Sintática Determinista

Gramáticas $\mathcal{LL}(1)$

Gramáticas $\mathcal{LR}(0)$

Gramáticas $\mathcal{LR}(1)$

A principal razão da ineficiência dos algoritmos não deterministas é a possibilidade de ocorrerem várias opções quando é aberto um nó na árvore de pesquisa.

- na análise descendente um não terminal pode ter várias produções;
- na análise ascendente podem ser aplicadas várias reduções a uma palavra;

Um algoritmo de análise sintática é determinista se, em cada passo, há informação suficiente para escolher uma única acção.

Essa informação pode inclui os próximos k símbolos (de avanço) da palavra...

UNIVERSIDADE DE ÉVORA

Análise Sintática Determinista

Gramáticas $\mathcal{LL}(1)$

Gramáticas $\mathcal{LR}(0)$

Gramáticas $\mathcal{LR}(1)$

Gramáticas $\mathcal{LL}(k)$ e $\mathcal{LL}(1)$

As gramáticas $\mathcal{LL}(k)$ são gramáticas independentes do contexto que admitem análise sintática descendente determinista com k símbolos de avanço.

Portanto, na análise sintática descendente de uma gramática $\mathcal{LL}(k)$, conhecidos os "próximos k símbolos" da palavra, não é necessário pesquisar produções alternativas.

NB. $\mathcal{LL}$ abrevia "*Left-to-right Leftmost derivation*".

A derivação da palavra $p = acbb$ pela gramática

$$S \to aS|cA$$
$$A \to bA|cB|\lambda$$
$$B \to cB|a|\lambda$$

começa pelo não terminal S e pelo símbolo de avanço a.

A produção $S \to cA$ não pode ser usada neste passo da derivação de p pois produz uma palavra começada por c. Portanto a primeira produção desta derivação só tem um candidato: $S \to aS$. Continuando...

prefixo	avançoresto	não terminal	produção aplicada	derivação
λ	$acbb$	S	$S \to aS$	$S \Rightarrow aS$
a	cbb	S	$S \to cA$	$\Rightarrow acA$
ac	bb	A	$A \to bA$	$\Rightarrow acbA$
acb	b	A	$A \to bA$	$\Rightarrow acbbA$
$acbb$	λ	A	$A \to \lambda$	$\Rightarrow acbb$

UNIVERSIDADE DE ÉVORA

A GIC $G = (V, \Sigma, P, S)$ com terminador $\#$ é $\mathcal{LL}(1)$ se dadas duas derivações esquerdas

$$S \Rightarrow^* u_1 A v_1 \Rightarrow u_1 x v_1 \Rightarrow^* u_1 a w_1$$
$$S \Rightarrow^* u_2 A v_2 \Rightarrow u_2 y v_2 \Rightarrow^* u_2 a w_2$$

com $u_i, w_i \in \Sigma^*$ e $a \in \Sigma$ então $x = y$.

Intuitivamente, não há produções de A distintas que produzam sufixos terminais que comecem pelo mesmo símbolo (aw_1 e aw_2).

Ou seja, a aplicação de produções A distintas resulta em sufixos terminais que diferem logo no primeiro terminal.

NB. O terminador $\#$ garante que há sempre um símbolo de avanço na palavra analisada.

Teorema

Uma gramática $\mathcal{LL}(1)$ é não ambígua.

Teorema

Se algum símbolo não terminal de G é recursivo à esquerda então G não é $\mathcal{LL}(1)$.

NB. As gramáticas $\mathcal{LL}(k)$, além de não ambíguas, também não contêm símbolos inúteis. Pode ser necessário aplicar as técnicas para a construção da forma normal de Greibach.

Seja $G = (V, \Sigma, P, S)$ uma GIC.

Se as produções de $A \in V$ forem

$$A \to wu_1 \mid wu_2 \mid \cdots \mid wu_j \mid v_1 \mid v_2 \mid \cdots \mid v_k \quad u_i, v_i \in (V \cup \Sigma)^*$$

com $w \in (V \cup \Sigma)^*$

então a gramática G' obtida de G

- acrescentado um novo símbolo não terminal Z
- e substituindo as produções de A por

$$A \to wZ \mid v_1 \mid v_2 \mid \cdots \mid v_k$$

- e acrescentado as produções

$$Z \to u_1 \mid u_2 \mid \cdots \mid u_j$$

é equivalente a G.

Primeiros e Seguintes

Os primeiros de $u \in (V \cup \Sigma)^*$ são os símbolos do alfabeto que podem aparecer na primeira posição de uma palavra derivada a partir de u

$$\text{PRIMEIROS}(u) = \{a \in \Sigma \ : \ u \Rightarrow^* ax \in \Sigma^*\}$$

Os seguintes de $A \in V$ são os símbolos do alfabeto que podem aparecer a seguir a A nalguma derivação

$$\text{SEGUINTES}(A) = \{a \in \Sigma \ : \ S \Rightarrow^* uAv \text{ e } a \in \text{PRIMEIROS}(v)\}$$

NB. Os seguintes vão ser necessários para produções vazias.

O conjunto dos símbolos directores da produção $A \to w \in P$ é

$$\text{DIR}(A \to w) = \begin{cases} \text{PRIMEIROS}(w) & \text{se } w \not\Rightarrow^* \lambda \\ \text{PRIMEIROS}(w) \cup \text{SEGUINTES}(A) & \text{se } w \Rightarrow^* \lambda \end{cases}$$

Teorema

Se para qualquer $A \in V$ e quaisquer produções distintas $A \to w$ e $A \to v$

$$\mathit{DIR}(A \to w) \cap \mathit{DIR}(A \to v) = \emptyset$$

então a gramática é $\mathcal{LL}(1)$.

Construção do grafo dos primeiros:

1. os vértices são os elementos de V e de Σ;
2. para cada produção $A \to s_1 s_2 \cdots s_n$ com $s_i \in V \cup \Sigma$:
   2.1 acrescenta-se um arco de A para s_1
   2.2 se $s_1 \in \Lambda$ acrescenta-se também um arco de A para s_2
   2.3 se $s_1, s_2 \in \Lambda$ acrescenta-se também um arco de A para s_3
   2.4 assim sucessivamente até se esgotarem os s_i ou haver um $s_i \notin \Lambda$

O grafo dos primeiros contém um caminho de $A \in V$ para $a \in \Sigma$ se e só se $a \in \text{PRIMEIROS}(A)$.

Cálculo dos Primeiros – 2

UNIVERSIDADE DE ÉVORA

Calcula-se indutivamente PRIMEIROS(w) com $w \in (V \cup \Sigma)^*$:

$$\begin{aligned}
\text{PRIMEIROS}(\lambda) &= \emptyset \\
\text{PRIMEIROS}(a) &= \{a\} & a \in \Sigma \\
\text{PRIMEIROS}(A) &= (\textit{usando o grafo}) & A \in V \\
\text{PRIMEIROS}(uv) &= \begin{cases} \text{PRIMEIROS}(u) \text{ se } u \not\Rightarrow^* \lambda \\ \text{PRIMEIROS}(u) \cup \text{PRIMEIROS}(v) \text{ se } u \Rightarrow^* \lambda \end{cases}
\end{aligned}$$

Cálculo dos Seguintes

Construção do grafo dos seguintes:

- os vértices do grafo são os elementos de V e de Σ;
- para cada produção $A \to uBv$ com $B \in V, u, v \in (V \cup \Sigma)^*$
  1. acrescenta-se um arco de B para cada $a \in \mathsf{PRIMEIROS}(v)$
  2. se $v \Rightarrow^* \lambda$ acrescenta-se um arco de B para A

O grafo dos seguintes contém um caminho de $A \in V$ para $a \in \Sigma$ se e só se $a \in \mathsf{SEGUINTES}(A)$.

UNIVERSIDADE DE ÉVORA

A gramática $S \to aSa|bSb|\lambda$ não é $\mathcal{LL}(1)$:
Os conjuntos PRIMEIROS e SEGUINTES são

	S
PRIMEIROS	a, b
SEGUINTES	a, b

enquanto que os conjuntos DIR são

	DIR
$S \to aSa$	a
$S \to bSb$	b
$S \to \lambda$	a, b

Portanto DIR$(S \to aSa) \cap$ DIR$(S \to \lambda) \neq \emptyset$.

Uma Gramática $\mathcal{LL}(1)$ para Expressões Aritméticas

$$
\begin{aligned}
S &\to E\# \\
E &\to E + T \mid T && \text{recursiva!} \\
T &\to (E) \mid a
\end{aligned}
$$

$$
\begin{aligned}
S &\to E\# \\
E &\to TZ \mid T && \text{prefixos comuns!} \\
Z &\to +TZ \mid +T && \text{prefixos comuns!} \\
T &\to (E) \mid a
\end{aligned}
$$

$$
\begin{aligned}
S &\to E\# \\
E &\to TF \\
F &\to Z \mid \lambda && \text{iguais} \ldots \\
Z &\to +TW \\
W &\to Z \mid \lambda && \text{iguais} \ldots \\
T &\to (E) \mid a
\end{aligned}
$$

$$
\begin{aligned}
S &\to E\# \\
E &\to TX \\
Z &\to +TX \\
X &\to Z \mid \lambda \\
T &\to (E) \mid a
\end{aligned}
$$

UNIVERSIDADE DE ÉVORA

produções

$$
\begin{aligned}
S &\to E\# \\
E &\to TX \\
Z &\to +TX \\
X &\to Z \mid \lambda \\
T &\to (E) \mid a
\end{aligned}
$$

primeiros e seguintes

$\Lambda = \{X\}$

	S	E	T	X	Z
PRIMEIROS	$a, ($	$a, ($	$a, ($	$+$	$+$
SEGUINTES		$), \#$	$+,), \#$	$), \#$	$), \#$

directores

$$\text{DIR}(S \to E\#) = \{a, (\}$$
$$\text{DIR}(E \to TX) = \{a, (\}$$
$$\text{DIR}(T \to a) = \{a\}$$
$$\text{DIR}(T \to (E)) = \{(\}$$
$$\text{DIR}(X \to Z) = \{+\}$$
$$\text{DIR}(X \to \lambda) = \{), \#\}$$
$$\text{DIR}(Z \to +TX) = \{+\}$$

```
proc S( )
    exercício
proc E( )
    se avanço ∈ {a, (} então
        T( )
        X( )
    senão
        erro( )
proc T( )
    se avanço ∈ {a} então
        consome(a)
    senão se avanço ∈ {(} então
        consome(()
        E( )
        consome())
    senão
        erro( )
```

```
proc X( )
    se avanço ∈ {+} então
        Z( )
    senão se avanço ∈ {), #}
então
        ;
    senão
        erro( )
proc Z( )
    se avanço ∈ {+} então
        consome(+)
        T( )
        X( )
    senão
        erro( )
```

pilha	avanço resto
$S()$	$a + a\#$
$E()$ consome$(\#)$	$a + a\#$
$T()$ $X()$ consome$(\#)$	$a + a\#$
consome(a) $X()$ consome$(\#)$	$a + a\#$
$X()$ consome$(\#)$	$+a\#$
$Z()$ consome$(\#)$	$+a\#$

pilha	avanço resto
consome$(+)$ $T()$ $X()$ consome$(\#)$	$+a\#$
$T()$ $X()$ consome$(\#)$	$a\#$
consome(a) $X()$ consome$(\#)$	$a\#$
$X()$ consome$(\#)$	$\#\lambda$
consome$(\#)$	$\#\lambda$
λ	$\lambda\lambda$

- Com o exemplo anterior apenas ficamos as saber se a palavra é ou não aceite... o que é pouco, depois de tanto trabalho;
- Idealmente também iríamos obter "algo" que ajude a processar a palavra — uma representação intermédia da palavra, que possa ser "calculada" por um programa;

  uma representação intermédia da palavra "2+3" *seria, por exemplo, a estrutura* `soma(2,3)`

- Os atributos são argumentos das rotinas do analisador sintáctico que vão coleccionando a representação intermédia...

proc S(e)
    ...
proc E(e)
    se avanço $\in \{\mathtt{a}, \mathtt{(}\}$ **então**
        T(t)
        X(t,e)
    senão
        erro( )
proc T(e)
    se avanço $\in \{\mathtt{a}\}$ **então**
        consome($\mathtt{a}$)
        e $\leftarrow$ $\mathtt{a}$
    senão se avanço $\in \{\mathtt{(}\}$ **então**
        consome($\mathtt{(}$)
        E(e)
        consome($\mathtt{)}$)
    senão
        erro( )

proc X(e,e1)
    se avanço $\in \{\mathtt{+}\}$ **então**
        Z(e,e1)
    senão se avanço $\in \{\mathtt{)}, \mathtt{\#}\}$ **então**
        e1 $\leftarrow$ e
    senão
        erro( )
proc Z(e,e1)
    se avanço $\in \{\mathtt{+}\}$ **então**
        consome($\mathtt{+}$)
        T(t)
        X(soma(e,t),e1)
    senão
        erro( )

Análise Sintática Determinista

Gramáticas $\mathcal{LL}(1)$

Gramáticas $\mathcal{LR}(0)$

Gramáticas $\mathcal{LR}(1)$

Análise Sintática $\mathcal{LR}$

objectivo gerar (automaticamente) um analisador sintáctico a partir das produções de uma GIC qualquer;

objectivo fazer análise sintática ascendente (mais eficiente?);

problema como escolher deterministicamente as reduções ou transferências?

- $\mathcal{LR}$ abrevia *Left-to-right Rightmost derivation in reverse*;
- assente em tabelas (não é recursiva);
- eficiente: detecta erros sintácticos assim que possível;
- muito trabalhosa feito à mão;

As gramáticas $\mathcal{LR}(k)$ são GIC que admitem análise sintática ascendente determinista com k símbolos de avanço.

problema: quando é que uma produção pode ser aplicada numa derivação “válida”?

Seja $G = (V, \Sigma, P, S)$ uma GIC com S não recursivo e terminador $\#$.

- A palavra $uw \in (V \cup \Sigma)^*$ é um contexto-$\mathcal{LR}(0)$ da produção $A \to w \in P$ se existir uma derivação direita

$$S \Rightarrow_R^* uAv \Rightarrow_R uwv \qquad , v \in \Sigma^*$$

- isto é, há uma redução $uwv \Leftarrow_R^* S$ que começa por $A \to w$;
- Um prefixo viável é um prefixo de um contexto-$\mathcal{LR}(0)$;
- Cada contexto-$\mathcal{LR}(0)$ é uma linguagem regular;

Uma GIC que não é $\mathcal{LR}(0)$

Numa GIC $\mathcal{LR}(0)$ os contextos-$\mathcal{LR}(0)$ são suficientes para escolher uma única acção: contextos de produções diferentes não se intersectam.

A gramática

$$S \to aA|aB$$
$$A \to aAb|b$$
$$B \to bBa|b$$

não é $\mathcal{LR}(0)$ porque...

As derivações direitas têm a forma

$$S \overset{S\to aA}{\Rightarrow_R} aA \overset{A\to aAb}{\Rightarrow^n_R} aa^nAb^n \overset{A\to b}{\Rightarrow_R} aa^nbb^n$$

$$S \overset{S\to aB}{\Rightarrow_R} aB \overset{B\to bBa}{\Rightarrow^n_R} ab^nBa^n \overset{B\to b}{\Rightarrow_R} ab^nba^n$$

...logo os contextos-$\mathcal{LR}(0)$ são

produção	contextos-$\mathcal{LR}(0)$
$S \to aA$	$\{aA\}$
$S \to aB$	$\{aB\}$
$A \to aAb$	$\{aa^nAb : n > 0\}$
$A \to b$	$\{aa^nb : n \geq 0\}$
$B \to bBa$	$\{ab^nBa : n > 0\}$
$B \to b$	$\{ab^n : n > 0\}$

...portanto ab é um contexto-$\mathcal{LR}(0)$ de $B \to b$ e de $A \to b$.

Exemplo — Contextos-$\mathcal{LR}(0)$

$$G_{LR_0} = (\{S, X, Y\}, \{a, b, \#\}, \ldots, S)$$

Sugestão: observar a árvore das derivações direitas.

produção	contextos-$\mathcal{LR}(0)$
$S \to X\#$	$X\#$
$X \to XY$	XY
$X \to \lambda$	λ
$Y \to aYa$	$XaYa, XaaYa, \ldots = Xa^*aYa$
$Y \to b$	$Xb, Xab, Xaab, \ldots = Xa^*b$

problema Os contextos podem ser conjuntos infinitos. Como calculá-los?

Dada uma GIC, após determinados os contextos-$\mathcal{LR}(0)$ de todas as suas produções...

Dada uma palavra de terminais p, após ler o prefixo u de $p = uv$:

1. se $u = xw$ é um contexto-$\mathcal{LR}(0)$ de $A \to w$, reduzir u com $A \to w$;
2. senão, se u é um prefixo viável, transferir;
3. finalmente (se u também não é um prefixo viável) rejeitar p;

Portanto é necessária uma forma eficiente de decidir se uma palavra é um contexto-$\mathcal{LR}(0)$, um prefixo viável ou nenhum.

Itens $\mathcal{LR}(0)$

UNIVERSIDADE DE ÉVORA

problema
: como determinar se uma palavra é um contexto-$\mathcal{LR}(0)$?

resolução
: construindo um AFD a partir das produções da gramática.

Seja $G = (V, \Sigma, P, S)$ uma gramática independente do contexto.

- Os itens $\mathcal{LR}(0)$ de G são as produções que se obtêm de P acrescentado um ponto $\bullet$ em todas as posições possíveis:
  1. Se $A \to \lambda \in P$ então $A \to \bullet$ é um item $\mathcal{LR}(0)$ de G;
  2. Se $A \to w \in P$ então para cada decomposição $w = uv$, $A \to u \bullet v$ é um item $\mathcal{LR}(0)$ de G;
- Um item completo é um item em que o ponto está o mais à direita possível;
- Um item $A \to u \bullet v$ é válido para o prefixo viável xu se xuv é um contexto-$\mathcal{LR}(0)$ *i.e.* $A \to uv$ é "candidato" a reduzir;

Os itens de G_{LR_0} são

$$
\begin{array}{llll}
S \to \bullet X\# & S \to X \bullet \# & S \to X\#\bullet & \\
X \to \bullet XY & X \to X \bullet Y & X \to XY\bullet & \\
X \to \bullet & & & \\
Y \to \bullet aYa & Y \to a \bullet Ya & Y \to aY \bullet a & Y \to aYa\bullet \\
Y \to \bullet b & Y \to b\bullet & &
\end{array}
$$

NB. Os itens completos estão indicados com esta cor.

Fecho de um Conjunto de Itens $\mathcal{LR}(0)$

O fecho de um conjunto I de itens define-se recursivamente

base $I \subseteq \text{fecho}(I)$

passo se $A \to u \bullet Bv \in \text{fecho}(I)$ com $B \in V$ então para cada produção $B \to w$ da gramática, $B \to \bullet w \in \text{fecho}(I)$

fecho nada mais pertence a $\text{fecho}(I)$

exemplo $\text{fecho}(\{X \to X \bullet Y\}) = \{X \to X \bullet Y, Y \to \bullet aYa, Y \to \bullet b\}$

Este fecho proporciona a construção directa de um AFD para os itens válidos. A alternativa implica um AFND... que teria de ser convertido num AFD.

Autómato Finito dos Itens $\mathcal{LR}(0)$ Válidos

Seja $G = (V, \Sigma, P, S)$ uma gramática independente do contexto

O autómato dos itens válidos de G, que reconhece os prefixos viáveis de G, é o autómato finito determinista

$$A = (Q, V \cup \Sigma, \delta, q_I, Q \setminus \{\emptyset\})$$

onde

estado inicial $q_I = \text{fecho}(\{S \to \bullet w \ : \ S \to w \in P\})$

transição para cada $q \in Q$ e $x \in V \cup \Sigma$

$$\delta(q, x) = \text{fecho}(\{A \to ux \bullet v \ : \ A \to u \bullet xv \in q\})$$

Exemplo – Autómato dos Itens $\mathcal{LR}(0)$ Válidos de $G_{\mathcal{LR}_0}$

Sem representar o estado vazio:

	q		
0	$q_I = \{S \to \bullet X\#, X \to \bullet XY, X \to \bullet\}$	X	q_1
1	$q_1 = \{S \to X \bullet \#, X \to X \bullet Y, Y \to \bullet aYa, Y \to \bullet b\}$	$\#$	q_2
		Y	q_3
		a	q_4
		b	q_5
2	$q_2 = \{S \to X\#\bullet\}$		
3	$q_3 = \{X \to XY\bullet\}$		
4	$q_4 = \{Y \to a \bullet Ya, Y \to \bullet aYa, Y \to \bullet b\}$	Y	q_6
		a	q_4
		b	q_5
5	$q_5 = \{Y \to b\bullet\}$		
6	$q_6 = \{Y \to aY \bullet a\}$	a	q_7
7	$q_7 = \{Y \to aYa\bullet\}$		

NB. Os itens completos estão indicados com esta cor.

UNIVERSIDADE DE ÉVORA

entrada: $G = (V, \Sigma, P, S)$ uma GIC $\mathcal{LR}(0)$
entrada: $A = (Q, V \cup \Sigma, \delta, q_I, Q \setminus \{\emptyset\})$ o AFD dos itens válidos de G
entrada: $p \in \Sigma^*$

```
u ← λ                                          ▷ "prefixo"
v ← p                                          ▷ sufixo
erro ← falso
repete
    q ← δ̂(q_I, u)
    se q contém A → w• então                   ▷ e portanto u = xw
        u ← xA                                 ▷ REDUÇÃO
    senão se q ≠ ∅ então                       ▷ e portanto u é um prefixo viável
        TRANSFERÊNCIA(u, v)                    ▷ muda ambos u e v
    senão
        erro ← verdade
até u = S ou erro
se u = S então
    ACEITA
senão
    REJEITA
```

u	v	q	item completo?	acção
λ	$aba\#$	$\hat{\delta}(q_I, \lambda) = q_I$	✓	$X \to \lambda$
X	$aba\#$	$\hat{\delta}(q_I, X) = q_1$		$TRANSF$
Xa	$ba\#$	$\hat{\delta}(q_I, Xa) = q_4$		$TRANSF$
Xab	$a\#$	$\hat{\delta}(q_I, Xab) = q_5$	✓	$Y \to b$
XaY	$a\#$	$\hat{\delta}(q_I, XaY) = q_6$		$TRANSF$
$XaYa$	$\#$	$\hat{\delta}(q_I, XaYa) = q_7$	✓	$Y \to aYa$
XY	$\#$	$\hat{\delta}(q_I, XY) = q_3$	✓	$X \to XY$
X	$\#$	$\hat{\delta}(q_I, X) = q_1$		$TRANSF$
$X\#$	λ	$\hat{\delta}(q_I, X\#) = q_2$	✓	$S \to X\#$
S	λ			$ACEITA$

UNIVERSIDADE DE ÉVORA

Teorema

Uma GIC é $\mathcal{LR}(0)$ se e só se o seu autómato dos itens válidos satifaz as seguintes condições:

- *Nenhum estado contém dois itens completos (caso contrário há um conflito redução/redução: que produção usar para reduzir?);*
- *Se um estado contém um item completo, em todos os outros itens desse estado o ponto é imediatamente seguido por um não terminal (caso contrário há um conflito transferência/redução).*

NB. Este teorema pode ser aplicado para confirmar que uma gramática é $\mathcal{LR}(0)$ ou que não é $\mathcal{LR}(0)$!

Tabela de Análise Sintática $\mathcal{LR}(0)$

- Uma linha por estado do AFD dos itens válidos, excepto o estado $\emptyset$;
- Uma coluna por cada símbolo de $(V \cup \Sigma) \setminus \{S\}$ cujo conteúdo corresponde à função de transição do autómato;
- Uma coluna ACÇÃO que na linha q contém
  - **ACEITA** se q contém um item completo de uma produção de S
  - **TRANSF** se q contém um item $A \to u \bullet av$ com $a \in \Sigma$
  - $A \to w$ indicando uma **redução** se q contém o item completo $A \to w \bullet$ e $A \neq S$
- As posições vazias da tabela indicam a **rejeição** da palavra;

Exemplo — Tabela de Análise Sintática $\mathcal{LR}(0)$ para $G_{\mathcal{LR}_0}$

UNIVERSIDADE DE ÉVORA

$$S \to X\#$$

$$X \to XY \mid \lambda$$

$$Y \to aYa \mid b$$

q	X	Y	a	b	$\#$	ACÇÃO
0	1					$X \to \lambda$
1		3	4	5	2	TRANSF
2						ACEITA
3						$X \to XY$
4		6	4	5		TRANSF
5						$Y \to b$
6			7			TRANSF
7						$Y \to aYa$

Autómato de Pilha Reconhecedor $\mathcal{LR}(0)$

Sejam $G = (V, \Sigma, P, S)$ uma gramática $\mathcal{LR}(0)$ e
$A = (Q, V \cup \Sigma, \delta_A, q_I, Q \setminus \{\emptyset\})$ o seu AFD dos itens válidos

O autómato de pilha que reconhece a linguagem gerada por G é

$$R = (\{p_I, p\}, \Sigma, V \cup \Sigma \cup Q \setminus \{\emptyset\}, \delta, p_I, \{p\})$$

em que a transição δ...

Transição do AP reconhecedor $\mathcal{LR}(0)$

UNIVERSIDADE DE ÉVORA

inicialização $[p, q_I] \in \delta(p_I, \lambda, \lambda)$

aceitação $[p, \lambda] \in \delta(p, \lambda, q_n a_n \cdots q_2 a_2 q_1 a_1 q_I)$ para todo o $q_n \in Q$ que contém um item completo $S \to a_1 a_2 \cdots a_n \bullet$ onde

$$[q_I, a_1 a_2 \cdots a_n] \vdash_A [q_1, a_2 \cdots a_n] \vdash_A^* [q_n, \lambda]$$

redução $[p, q' A q_0] \in \delta(p, \lambda, q_n a_n \cdots q_2 a_2 q_1 a_1 q_0)$ para todo o $q_n \in Q$ que contém um item completo $A \to a_1 a_2 \cdots a_n \bullet$ com $A \neq S$ e onde

$$q' = \delta_A(q_0, A)$$

$$[q_0, a_1 a_2 \cdots a_n] \vdash_A [q_1, a_2 \cdots a_n] \vdash_A^* [q_n, \lambda]$$

transferência $[p, q' a q] \in \delta(p, a, q)$ se $\delta_A(q, a) = q'$ e $a \in \Sigma$

A Transição $\mathcal{LR}(0)$ por outros termos

inicialização $p_I \xrightarrow{\lambda,\ \lambda/q_I} p$

aceitação transições da forma $p \xrightarrow{\lambda,\ X/\lambda} p$ — para calcular X

1. seja $q \in Q$ com o item completo $S \to a_1 a_2 \cdots a_n \bullet$
2. *que caminhos (no AFD)* $q_I \overset{a_1}{\vdash} q_2 \overset{a_2}{\vdash} \cdots \overset{a_n}{\vdash} q$ *levam de* q_I *a* q *lendo* $a_1 a_2 \cdots a_n$*?* cada X é um desses caminhos **invertido**;

redução transições da forma $p \xrightarrow{\lambda,\ X/Y} p$ — para calcular X e Y

1. seja $q \in Q$ com o item completo $B \to a_1 a_2 \cdots a_n \bullet$ e $B \neq S$;
2. X é como acima, mas o caminho não tem de começar em q_I;
3. $Y = q' B q_1$ sendo q_1 o estado inicial do caminho e $q' = \delta_A(q_1, B)$

transferência $p \xrightarrow{a,\ q/q'aq} p$ com $q' = \delta_A(q, a)$

Transição do AP para G_{LR_0}

inicialização $(p_I, \lambda) \xrightarrow{\lambda} (p, \mathbf{0})$

aceitação

$$(p, \mathbf{2}\#\mathbf{1}X\mathbf{0}) \xrightarrow{\lambda} (p, \lambda) \qquad S \to X\#\bullet \in \mathbf{2}$$

redução

$$(p, \mathbf{0}) \xrightarrow{\lambda} (p, \mathbf{1}X\mathbf{0}) \qquad X \to \bullet \in \mathbf{0}$$
$$(p, \mathbf{3}Y\mathbf{1}X\mathbf{0}) \xrightarrow{\lambda} (p, \mathbf{1}X\mathbf{0}) \qquad X \to XY\bullet \in \mathbf{3}$$
$$(p, \mathbf{5}b\mathbf{1}) \xrightarrow{\lambda} (p, \mathbf{3}Y\mathbf{1}) \qquad Y \to b\bullet \in \mathbf{5}$$
$$(p, \mathbf{5}b\mathbf{4}) \xrightarrow{\lambda} (p, \mathbf{6}Y\mathbf{4}) \qquad Y \to b\bullet \in \mathbf{5}$$
$$(p, \mathbf{7}a\mathbf{6}Y\mathbf{4}a\mathbf{1}) \xrightarrow{\lambda} (p, \mathbf{3}Y\mathbf{1}) \qquad Y \to aYa\bullet \in \mathbf{7}$$
$$(p, \mathbf{7}a\mathbf{6}Y\mathbf{4}a\mathbf{4}) \xrightarrow{\lambda} (p, \mathbf{6}Y\mathbf{4}) \qquad Y \to aYa\bullet \in \mathbf{7}$$

transferência

$$(p, \mathbf{1}) \xrightarrow{a} (p, \mathbf{4}a\mathbf{1})$$
$$(p, \mathbf{1}) \xrightarrow{b} (p, \mathbf{5}b\mathbf{1})$$
$$(p, \mathbf{1}) \xrightarrow{\#} (p, \mathbf{2}\#\mathbf{1})$$
$$(p, \mathbf{4}) \xrightarrow{a} (p, \mathbf{4}a\mathbf{4})$$
$$(p, \mathbf{4}) \xrightarrow{b} (p, \mathbf{5}b\mathbf{4})$$
$$(p, \mathbf{6}) \xrightarrow{a} (p, \mathbf{7}a\mathbf{6})$$

Exemplo — Simulação do AP para G_{LR_0} de $aba\#$

controlo	pilha	palavra	ACÇÃO
p_I	λ	$aba\#$	
p	$\mathbf{0}$	$aba\#$	$X \to \lambda$
p	$\mathbf{1}X\mathbf{0}$	$aba\#$	TRANSF
p	$\mathbf{4}a\mathbf{1}X\mathbf{0}$	$ba\#$	TRANSF
p	$\mathbf{5}b\mathbf{4}a\mathbf{1}X\mathbf{0}$	$a\#$	$Y \to b$
p	$\mathbf{6}Y\mathbf{4}a\mathbf{1}X\mathbf{0}$	$a\#$	TRANSF
p	$\mathbf{7}a\mathbf{6}Y\mathbf{4}a\mathbf{1}X\mathbf{0}$	$\#$	$Y \to aYa$
p	$\mathbf{3}Y\mathbf{1}X\mathbf{0}$	$\#$	$X \to XY$
p	$\mathbf{1}X\mathbf{0}$	$\#$	TRANSF
p	$\mathbf{2}\#\mathbf{1}X\mathbf{0}$	λ	$S \to X\#$
p	λ	λ	ACEITA

Análise Sintática Determinista

Gramáticas $\mathcal{LL}(1)$

Gramáticas $\mathcal{LR}(0)$

Gramáticas $\mathcal{LR}(1)$

A gramática das expressões aritméticas simples,

$$
\begin{aligned}
S &\to E\# \\
E &\to E + T \mid T \\
T &\to T \times F \mid F \\
F &\to n \mid (E)
\end{aligned}
$$

não é $\mathcal{LR}(0)$.

Mas a análise sintática $\mathcal{LR}(0)$ pode ser modificada de forma a considerar um símbolo de avanço...

Item $\mathcal{LR}(1)$

Seja $G = (V, \Sigma, P, S)$ uma gramática independente do contexto.

Os itens-$\mathcal{LR}(1)$ de G têm a forma

$$[A \to u \bullet b \; ; \; L]$$

onde

- $A \to u \bullet v$ é um item $\mathcal{LR}(0)$, o núcleo e
- $L \subseteq \Sigma \cup \{\#\}$ é o conjunto dos símbolos de avanço

Um item $[A \to u \bullet v \; ; \; L]$ é válido para xu se para cada $a \in L$ existe uma derivação

$$S \Rightarrow_R^* xAy$$

com $a \in \text{PRIMEIROS}(y\#)$.

Fecho $\mathcal{LR}(1)$

O fecho$_1$ de um conjunto X de itens $\mathcal{LR}(1)$ define-se recursivamente

base $X \subseteq \text{fecho}_1(X)$

passo se $[A \to u \bullet Bv \ ; \ L] \in \text{fecho}_1(X)$ (e $B \in V$) então para cada produção $B \to w$ também $[B \to \bullet w \ ; \ K] \in \text{fecho}_1(X)$ onde

$$K = \begin{cases} \text{PRIMEIROS}(v)\,, & v \not\Rightarrow^* \lambda \\ \text{PRIMEIROS}(v) \cup L, & v \Rightarrow^* \lambda \end{cases}$$

fecho nada mais pertence a fecho$_1(X)$

Exemplo

$$G_{LR_1} = (\{S, A\}, \{a, b\}, \ldots, S)$$
$$S \to AbA$$
$$A \to Aa \mid \lambda$$

$$\begin{aligned}
\mathsf{fecho}_1(\{[S \to Ab\underline{\bullet A}\ ;\ \{\#\}]\}) = {} & \{[S \to Ab \bullet A\ ;\ \{\#\}]\} \cup \\
& \{[A \to \underline{\bullet A}a\ ;\ \{\#\}], [A \to \bullet\ ;\ \{\#\}]\} \cup \\
& \{[A \to \bullet Aa\ ;\ \{a\}], [A \to \bullet\ ;\ \{a\}]\} \cup \\
& \{[A \to \bullet Aa\ ;\ \{a\}], [A \to \bullet\ ;\ \{a\}]\} \\
& \ldots \\
& = \Big\{ [S \to Ab \bullet A\ ;\ \{\#\}], \\
& [A \to \bullet Aa\ ;\ \{\#\}], [A \to \bullet Aa\ ;\ \{a\}], \\
& [A \to \bullet\ ;\ \{\#\}], [A \to \bullet\ ;\ \{a\}] \Big\}
\end{aligned}$$

Autómato Finito dos Itens $\mathcal{LR}(1)$ Válidos

Seja $G = (V, \Sigma, P, S)$ uma gramática independente do contexto e $G' = (V \cup \{S'\}, \Sigma, P \cup \{S' \to S\}, S')$.

O autómato dos itens $\mathcal{LR}(1)$ válidos de G' é o AFD

$$A = (Q, V \cup \Sigma, \delta, q_I, Q \setminus \{\emptyset\})$$

onde

estado inicial $q_I = \mathsf{fecho}_1(\{[S' \to \bullet S \ ; \ \{\#\}]\})$

transição para cada $q \in Q$ e $x \in V \cup \Sigma$

$$\delta(q, x) = \mathsf{fecho}_1(\{[A \to ux \bullet v \ ; \ L] \ : \ [A \to u \bullet xv \ ; \ L] \in q\})$$

Condições $\mathcal{LR}(1)$

Uma GIC é $\mathcal{LR}(1)$ se o seu autómato dos itens $\mathcal{LR}(1)$ válidos satisfaz as seguintes condições:

- se um estado contém dois itens completos distintos $[A \to w\bullet\ ;\ L]$ e $[B \to u\bullet\ ;\ K]$ então $L \cap K = \emptyset$;
- se um estado contém um item completo $[A \to w\bullet\ ;\ L]$ e um item $[B \to u \bullet av\ ;\ K]$ então $a \notin L$;

Tabela de Análise Sintática $\mathcal{LR}(1)$

- uma linha por estado do AFD dos itens válidos, excepto o estado $\emptyset$
- uma coluna por símbolo de $(V \cup \Sigma) \setminus \{S\}$ cujo conteúdo corresponde à transição do autómato
- uma coluna por símbolo $a \in \Sigma \cup \{\#\}$ que, na linha q contém a **acção**

ACEITA	se q contém um item completo de uma produção de S e $a = \#$
TRANSF	se q contém um item $A \to u \bullet av; L$
$A \to w$	indicando uma REDUÇÃO se q contém o item completo $[A \to w\bullet\ ;\ L]$ e $A \neq S$ e $a \in L$

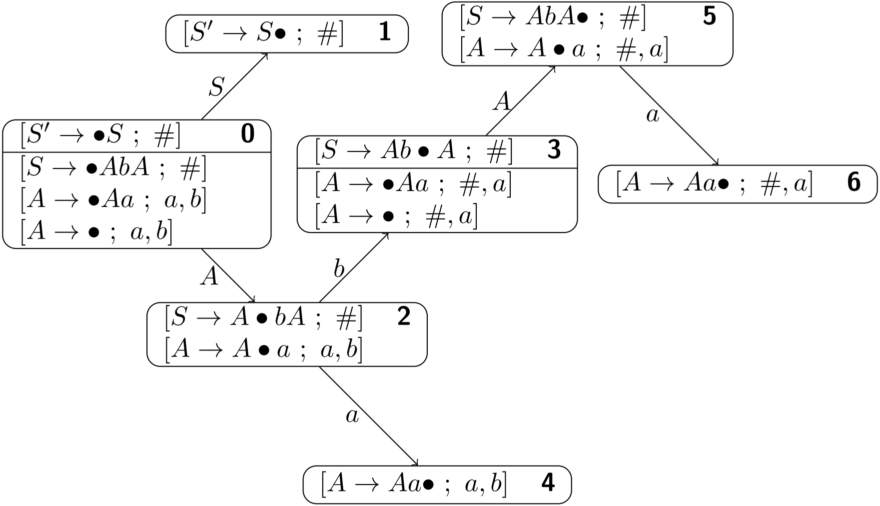
As posições vazias indicam a REJEIÇÃO da palavra.

$$S \to AbA$$
$$A \to Aa \mid \lambda$$

$$\Lambda = \{A\}$$

não terminal	S	A
PRIMEIROS	$\{a, b\}$	$\{a\}$

Exemplo (cont.) — Autómato dos Itens Válidos



Exemplo (cont.) — Tabela de Análise Sintática

q	S	A	a	b	a	b	$\#$
0	1	2			$A \to \lambda$	$A \to \lambda$	
1							ACEITA
2			4	3	TRANSF	TRANSF	
3		5			$A \to \lambda$		$A \to \lambda$
4					$A \to Aa$	$A \to Aa$	
5			6		TRANSF		$S \to AbA$
6					$A \to Aa$		$A \to Aa$

Autómato de Pilha Reconhecedor $\mathcal{LR}(1)$

UNIVERSIDADE DE ÉVORA

Sejam $G = (V, \Sigma, P, S)$ uma gramática $\mathcal{LR}(1)$ e
$A = (Q, V \cup \Sigma, \delta_A, q_I, Q \setminus \{\emptyset\})$ o seu AFD dos itens válidos

O autómato de pilha que reconhece a linguagem gerada por G é

$$R = (Q_R, \Sigma \cup \{\#\}, V \cup \Sigma \cup Q \setminus \{\emptyset\}, \delta_R, p_I, F_R)$$

com

estados $Q_R = \{p_I, p\} \cup \{p_a \; : \; a \in \Sigma \cup \{\#\}\}$

aceitação $F_R = \{p_\#\}$

e a transição $\delta_R \ldots$

Transição do AP Reconhecedor $\mathcal{LR}(1)$

inicialização $[p, q_I] \in \delta_R(p_I, \lambda, \lambda)$

avanço $[p_a, \lambda] \in \delta_R(p, a, \lambda)$ para cada $a \in \Sigma \cup \{\#\}$

aceitação $[p_\#, \lambda] \in \delta_R(p_\#, \lambda, q_n a_n \cdots q_2 a_2 q_1 a_1 q_I)$ para cada $q_n \in Q$ que contém um item completo $[S \to a_1 a_2 \cdots a_n \bullet \ ; \ L]$ com $\# \in L$ e

$$[q_I, a_1 a_2 \cdots a_n] \vdash_A [q_1, a_2 \cdots a_n] \vdash_A \cdots \vdash_A [q_n, \lambda]$$

redução $[p_a, q' A q_0] \in \delta_R(p_a, \lambda, q_n a_n \cdots q_2 a_2 q_1 a_1 q_0)$ para cada $q_n \in Q$ que contém um item completo $[A \to a_1 a_2 \cdots a_n \bullet \ ; \ L]$ com $a \in L$, $A \neq S$ e

$$q' = \delta_A(q_0, A)$$

$$[q_0, a_1 a_2 \cdots a_n] \vdash_A [q_1, a_2 \cdots a_n] \vdash_A \cdots \vdash_A [q_n, \lambda]$$

transferência $[p, q' a q] \in \delta_R(p_a, \lambda, q)$ para cada $q \in Q$ que contém um item $[A \to u \bullet av \ ; \ L]$ com $a \in \Sigma$ e $q' = \delta_A(q, a)$

A Transição $\mathcal{LR}(1)$ por outros termos

inicialização $p_I \xrightarrow{\lambda,\ \lambda/q_I} p$

avanço $p \xrightarrow{a,\ \lambda/\lambda} p_a$

aceitação transições da forma $p_\# \xrightarrow{\lambda,\ X/\lambda} p_\#$ — para calcular X

1. seja $q \in Q$ com o item completo $[S \to a_1 a_2 \cdots a_n \bullet\ ;\ L]$ com $\# \in L$
2. *que caminhos (no AFD)* $q_I \overset{a_1}{\vdash} q_2 \overset{a_2}{\vdash} \cdots \overset{a_n}{\vdash} q$ *levam de* q_I *a* q *lendo* $a_1 a_2 \cdots a_n$*?* cada X é um desses caminhos **invertido**;

redução transições da forma $p_a \xrightarrow{\lambda,\ X/Y} p_a$ — para calcular X e Y

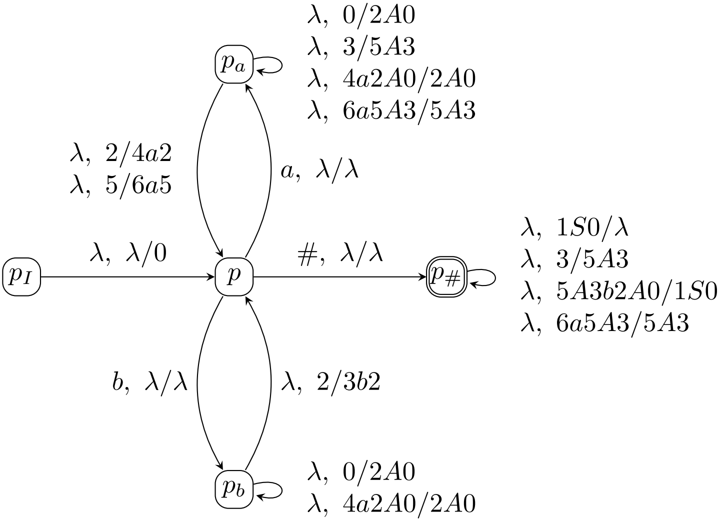
1. seja $q \in Q$ com o item completo $[B \to a_1 a_2 \cdots a_n \bullet\ ;\ L]$, $a \in L$ e $B \neq S$
2. X é como acima, mas o caminho não tem de começar em q_I
3. e $Y = q' B q_1$ sendo q_1 o estado inicial do caminho e $q' = \delta_A(q_1, B)$

transferência $p_a \xrightarrow{\lambda,\ q/q'aq} p$ com $q' = \delta_A(q, a)$

UNIVERSIDADE DE ÉVORA

inicialização	$(p_I, \lambda) \xrightarrow{\lambda} (p, 0)$	
avanço	$(p, \lambda) \xrightarrow{a} (p_a, \lambda)$	
	$(p, \lambda) \xrightarrow{b} (p_b, \lambda)$	
	$(p, \lambda) \xrightarrow{\#} (p_\#, \lambda)$	
aceitação	$(p_\#, 1S0) \xrightarrow{\lambda} (p_\#, \lambda)$	
redução	$(p_a, 0) \xrightarrow{\lambda} (p_a, 2A0)$	**de** $[A \to \bullet, ba] \in 0$
	$(p_b, 0) \xrightarrow{\lambda} (p_b, 2A0)$	
	$(p_\#, 3) \xrightarrow{\lambda} (p_\#, 5A3)$	**de** $[A \to \bullet, \#a] \in 3$
	$(p_a, 3) \xrightarrow{\lambda} (p_a, 5A3)$	
	$(p_b, 4a2A0) \xrightarrow{\lambda} (p_b, 2A0)$	**de** $[A \to Aa\bullet, ba] \in 4$
	$(p_a, 4a2A0) \xrightarrow{\lambda} (p_a, 2A0)$	
	$(p_\#, 5A3b2A0) \xrightarrow{\lambda} (p_\#, 1S0)$	**de** $[S \to AbA\bullet, \#] \in 5$
	$(p_\#, 6a5A3) \xrightarrow{\lambda} (p_\#, 5A3)$	**de** $[A \to Aa\bullet, \#a] \in 6$
	$(p_a, 6a5A3) \xrightarrow{\lambda} (p_a, 5A3)$	
transferência	$(p_a, 2) \xrightarrow{\lambda} (p, 4a2)$	
	$(p_b, 2) \xrightarrow{\lambda} (p, 3b2)$	
	$(p_a, 5) \xrightarrow{\lambda} (p, 6a5)$	

Transição do AP Reconhecedor $\mathcal{LR}(1)$ para G_{LR_1}



Computação de $aaba$ no AP Reconhecedor $\mathcal{LR}1$ para $G_{\mathcal{LR}_1}$

estado	pilha	palavra
p_I	λ	$aaba\#$
p	0	$aaba\#$
p_a	0	$aba\#$
p_a	$2A0$	$aba\#$
p	$4a2A0$	$aba\#$
p_a	$4a2A0$	$ba\#$
p_a	$2A0$	$ba\#$
p	$4a2A0$	$ba\#$
p_b	$4a2A0$	$a\#$
p_b	$2A0$	$a\#$
p	$3b2A0$	$a\#$
...		

estado	pilha	palavra
...		
p_a	$3b2A0$	$\#$
p_a	$5A3b2A0$	$\#$
p	$6a5A3b2A0$	$\#$
$p_\#$	$6a5A3b2A0$	λ
$p_\#$	$5A3b2A0$	λ
$p_\#$	$1S0$	λ
$p_\#$	λ	λ

ACEITA

Seja $A = (Q, \Sigma, \delta, q_I, F)$ um autómato dos itens $\mathcal{LR}(1)$ válidos

O autómato amalgamado A' é o autómato que resulta de fundir num só os estados de A com o mesmo conjunto de núcleos $\mathcal{LR}(0)$.

Se $X = \{q_1, q_2, \ldots, q_m\}$ for um conjunto de estados de A com o mesmo núcleo $\mathcal{LR}(0)$ o resultado da fusão dos estados de X é o estado $\widehat{X}$ que contém os itens $A \to u \bullet v; L_1 \cup L_2 \cup \cdots \cup L_m$ tais que $A \to u \bullet v; L_i \in q_i$

Seja $\{Q_1, Q_2, \cdots, Q_n\}$ uma partição de Q tal que todos os estados de Q_i têm o mesmo núcleo e, se $i \neq j$, os núcleos dos estados de Q_i e Q_j são diferentes. Então

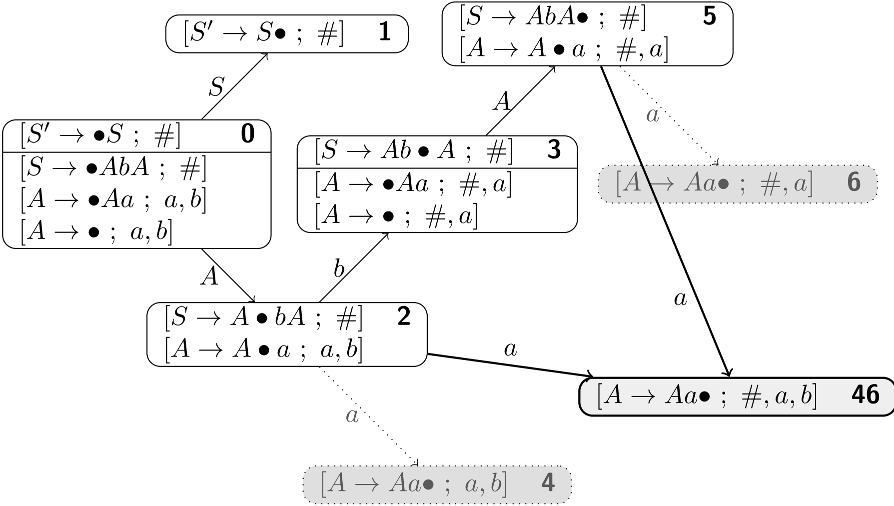
$$A' = \left(Q_A, \Sigma, \delta', \widehat{\{q_I\}}, Q_A \setminus \left\{\widehat{\{\emptyset\}}\right\}\right)$$

com

estados $Q_A = \left\{\widehat{Q_1}, \widehat{Q_2}, \ldots, \widehat{Q_n}\right\}$

transição $\delta'\left(\widehat{Q_i}, a\right) = \widehat{Q_j}$ se existe $q \in Q_i$ tal que $\delta(q, a) \in Q_j$

Exemplo — Autómato Amalgamado



UNIVERSIDADE DE ÉVORA

Uma gramática independente do contexto é $\mathcal{LALR}(1)$ se o seu autómato amalgamado satisfaz as condições $\mathcal{LR}(1)$.

O interesse das gramáticas $\mathcal{LALR}(1)$ ...

... consiste em reduzir o número de estados do autómato dos itens válidos que, além das gramáticas "simples" usadas para ilustrar estes métodos, facilmente atingem um número (muito) elevado.