Equivalência entre AP e GLC

Luís Felipe

UFF

31 de Maio de 2023

Luís Felipe

31/05/23

GLC $\rightarrow$ AP

Lema. Se uma linguagem é LLC, então existe um AP que a reconheça.

Ideia da prova: A partir de uma GLC, $G = (V, T, S, R)$, vamos construir um AP equivalente P que aceita toda palavra w derivada por G.

P deve ser capaz de determinar se há derivação em G que leva S para w. O não-determinismo é fundamental, pois permite ao autômato que adivinhe as regras corretamente.

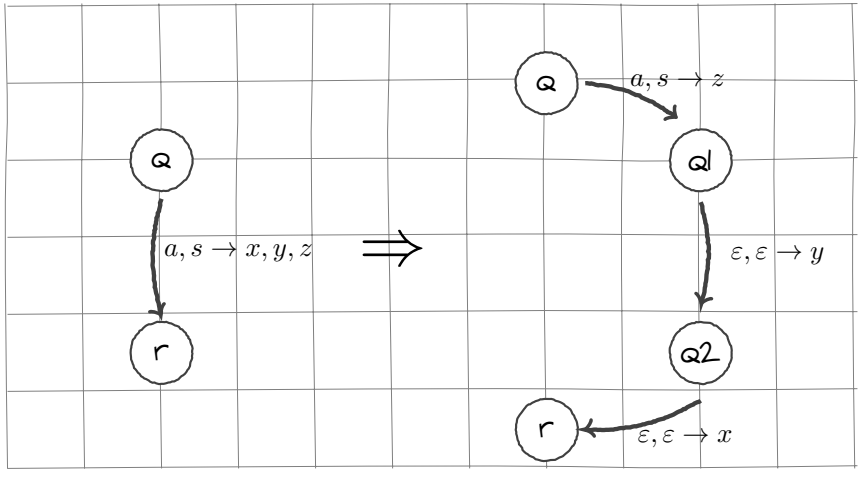
O autômato empilha S e simula a derivação. Se em algum momento ele chegar em uma cadeia de terminais, então G deriva tal palavra e, se ela for a mesma da entrada do AP, então deve ser aceita por P.

Luís Felipe

31/05/23

Continuação - ideia adicional

- Como o AP armazenaria as cadeias intermediárias de variáveis ou terminais obtidas nas derivações?
  - Vamos usar a pilha.
  - O AP precisa ter acesso às variáveis para fazer as substituições.
- Mas como escrever uma cadeia inteira na pilha?
  - Vamos usar uma notação simplificada. Por exemplo: $(r, xyz) \in \delta(q, a, s)$
  - Vamos usar esta abreviação para o seguinte esquema:



Luís Felipe

31/05/23

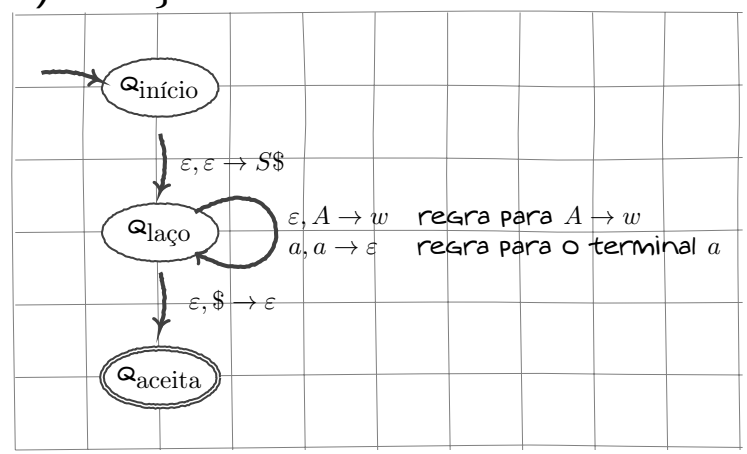
Construção formal do AP

Vamos construir o AP, $P = (\Sigma, \Gamma, Q, q_0, F, \delta)$ da seguinte forma:

- $Q = \{q_{\text{início}}, q_{\text{laço}}, q_{\text{aceita}}\} \cup E$, onde E é o conjunto de estados necessários para implantar a abreviação descrita anteriormente.

- $q_0 = q_{\text{início}}, F = \{q_{\text{aceita}}\}$

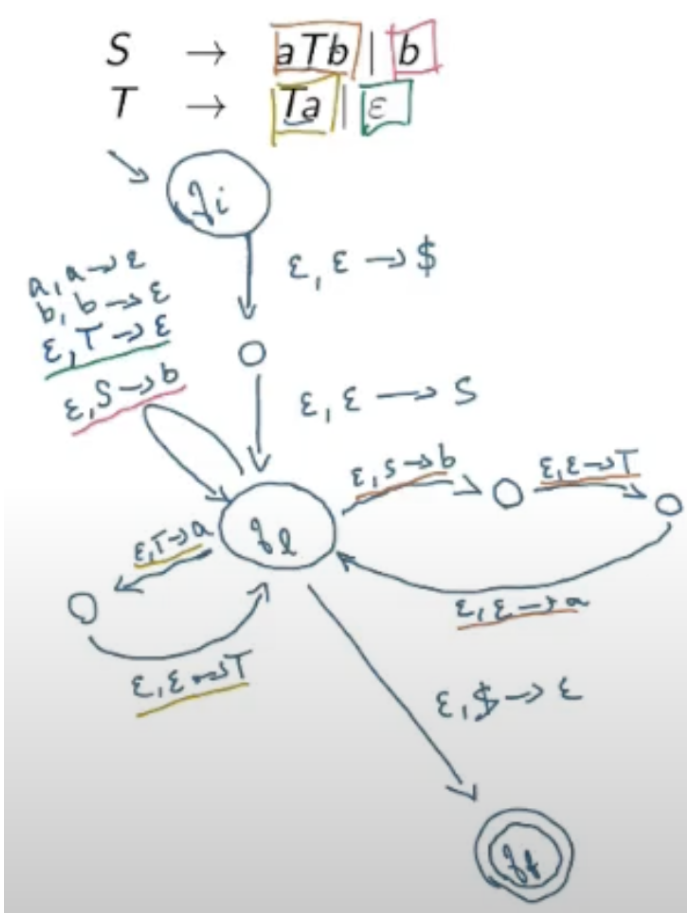
- δ é definida da seguinte forma:
  - $\delta(q_{\text{início}}, \varepsilon, \varepsilon) = \{(q_{\text{laço}}, S\$)\}$
  - $\delta(q_{\text{laço}}, \varepsilon, A) = \{(q_{\text{laço}}, w) \ : \ (A \to w) \in R\}$
  - $\delta(q_{\text{laço}}, a, a) = \{(q_{\text{laço}}, \varepsilon)\}$
  - $\delta(q_{\text{laço}}, \varepsilon, \$) = \{(q_{\text{aceita}}, \varepsilon)\}$

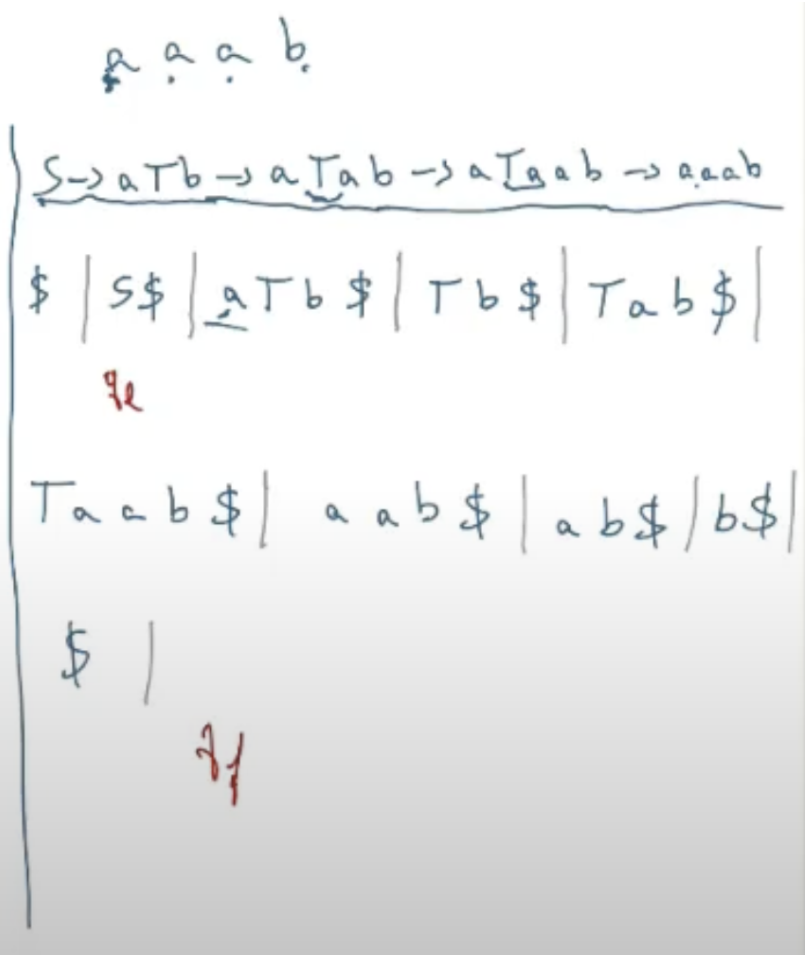


Luís Felipe

31/05/23

Exemplo





Luís Felipe

31/05/23

AP $\rightarrow$ GLC

Lema. Se um AP reconhece uma linguagem, então ela é livre de contexto.

Ideia da prova: A partir de um AP P que reconhece L, vamos construir uma GLC que gere todas as palavras que P aceita.

Assim, para cada par de estados p, q em P, vamos criar uma variável A_{pq}.

Esta variável deve gerar todas as cadeias que levam p a q, tendo na pilha em q os mesmos elementos que havia em p. Assim, observe que, se a pilha em p era vazia, então em que q também será.

Vamos considerar que o AP está simplificado:

1. apenas um estado final, q_{final}
2. esvazia a pilha para aceitar
3. cada transição ou empilha ou desempilha um símbolo (ou exclusivo)

Luís Felipe

31/05/23

Simplificando *P*

- Para a característica 3:

  Vamos substituir uma transição que desempilha e empilha por duas em sequência, i.e., uma desempilha e a outra empilha.

  Cada transição que nem empilha nem desempilha será substituída por uma sequência de duas transições: uma que empilha um símbolo arbitrário e outra que desempilha tal símbolo.

Luís Felipe

31/05/23

Computação de P

- Pensemos como P computa uma palavra w. O primeiro movimento de P tem que ser o de empilhar e o último tem que ser o de desempilhar (dado P simplificado).

- Há duas possibilidades: Caso 1: o primeiro símbolo empilhado é o último desempilhado (a pilha começa vazia e só fica vazia no fim); ou Caso 2: primeiro símbolo empilhado é desempilhado em algum momento intermediário da computação.

- Caso 1: Quando a é lido ao incluir t na pilha e b é lido ao remover t da pilha, $(r, t) \in \delta(p, a, \varepsilon)$ e $(q, \varepsilon) \in \delta(s, b, t)$, temos a regra $A_{pq} \to aA_{rs}b$.

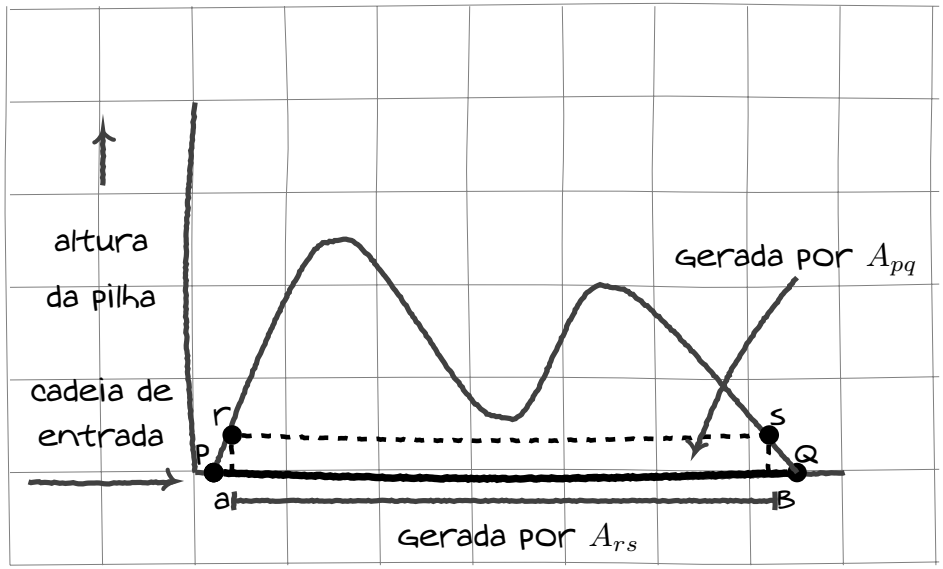
- Caso 2: Se r é o estado no qual a pilha fica vazia, temos a regra $A_{pq} \to A_{pr}A_{rq}$.

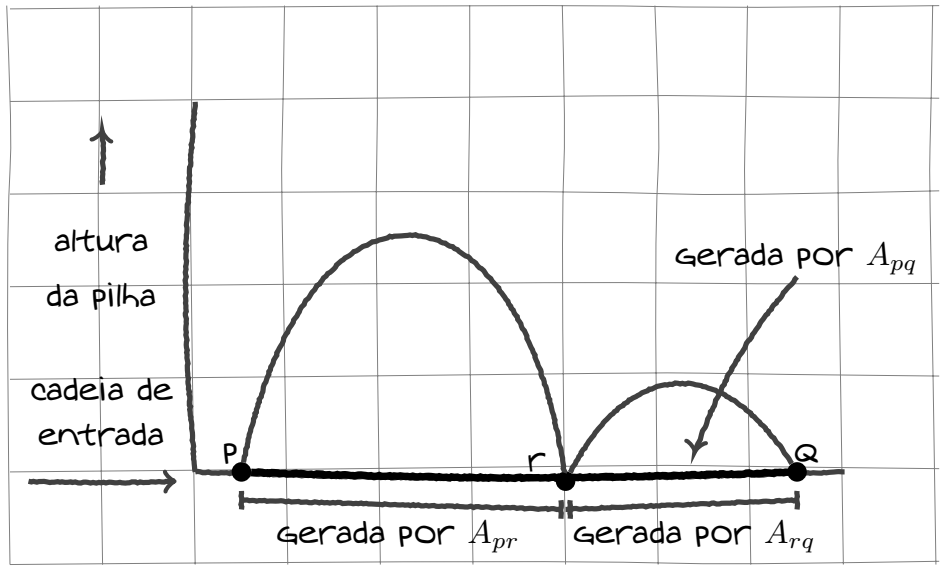
- Finalmente, para cada estado p temos a regra: $A_{pp} \to \varepsilon$.

Luís Felipe

31/05/23

Esquemas





Luís Felipe

31/05/23

Formalmente

Seja $M = (Q, \Sigma, \Gamma, \delta, q_0, \{q_{final}\})$. Definimos $G = (T, V, S, R)$ da seguinte forma:

- $T = \Sigma$
- $V = \{A_{pq} \ : \ p, q \in Q\}$
- $S = A_{q_0, q_{final}}$
- Regras:
  1. $p, q, r, s \in Q$, $t \in \Gamma$, $a, b \in \Sigma \cup \{\varepsilon\}$. Se $(r, t) \in \delta(p, a, \varepsilon)$ e $(q, \varepsilon) \in \delta(s, b, t)$, então: $A_{pq} \rightarrow aA_{rs}b$.
  2. $p, q, r \in Q$, então: $A_{pq} \rightarrow A_{pr}A_{rq}$
  3. $p \in Q$, então: $A_{pp} \rightarrow \varepsilon$

Luís Felipe

31/05/23

Para finalizar...

- Falta mostrar que essa construção de fato funciona, ou seja:

  Afirmação. Se A_{pq} gera w, então w pode levar P de p com pilha vazia para q com pilha vazia.

  Ideia da Prova: Indução no número de passos na derivação de w a partir de A_{pq}. Obs.: Observe a recursividade que há nas regras impostas.

  Afirmação. Se w pode levar P de p com pilha vazia para q com pilha vazia, então A_{pq} gera w.

  Ideia da Prova: Indução no número de passos na computação de P que vai de p para q com pilhas vazias sobre a entrada w.