Linguagens Formais

Autômatos de Pilha (AP) – Parte 2

Diego Costa
Heverson Ataíde
Marcio Junior
Werisson Ernesto

Dia 03/11/2017 – Equipe B

AGENDA

- Autômato de Pilha Determinístico;
- Autômato de Pilha não-Determinístico;
- Equivalência do autômato de pilha e Gramática live de Contexto;
  - APNP → GLC
  - GLC → APND
- Questionário
- Referências Bibliográficas

1

Autômato de pilha Determinístico

Autômatos com pilha, como funcionam?

Definição Formal:
Seja “R” um autômato com pilha
Temos então:

$R=(Q,\Sigma,\Gamma,q0,\$,F,\delta);$

1. Autômato de Pilha Determinístico

- **Q** é o conjunto finito de estados;
- **Σ** é o conjunto finito de símbolos de entrada (ou alfabeto de entrada)
- **Γ** é o conjunto finito de símbolos da pilha (ou alfabeto da pilha)
- **q0** é o estado inicial.

1. Autômato de Pilha Determinístico

- **Q** é o conjunto finito de estados;
- **$** é o símbolo inicial da pilha.
- **F** é o conjunto de estados de aceitação.
- **δ** é uma função de transição.

1. Autômato de Pilha Determinístico

- Para que R seja determinístico temos as seguintes condições:
  - (1) $\delta(q,a,b)$ contém ao menos um elemento;
  - (2) Se $\delta(q,\lambda,b)$ não é vazio, então $\delta(q,c,b)$ deve ser vazio para todo $c \in \Sigma$.

1. Autômato de Pilha Determinístico

- Seja :$R=(Q,\Sigma,\Gamma,q0,\$,F,\delta)$ então a linguagem reconhecida é:

- $L(R)=\{w \in \Sigma^*|[q0,w,\epsilon] \vdash^*R\ [qi,\epsilon,\epsilon],qi \in F\};$

1. Autômato de Pilha Determinístico

- Seja $L=\{w \in \Sigma=(0+1)^*|w=xx^h\}$ onde x^h é a cadeia reversa de cadeia x, sabe-se que L é uma LLC.

- Consideremos um marcador c como um marcador de meio de cadeia, portanto a linguagem poderá ser reconhecida por um APD.

Existe não-determinismo em automatos com pilha?

2

Autômato de pilha não-Determinístico

Definição Formal:
Seja “R” um autômato com pilha
Temos então:

$$P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F);$$

2. Autômato de Pilha não-Deteminístico

- Notou alguma diferença?
  - A definição de APD e APND é bem parecida, a diferença está na função δ, que assim como nos automatos finitos define um conjunto de operações

2. Autômato de Pilha não-Deteminístico

- Definindo a função δ:
  - $\delta : Q \times \Sigma \cup \{\epsilon\} \times \Gamma \cup \{\epsilon\} \rightarrow P(Q \times \Gamma \cup \{\epsilon\})$
  - Exemplo:
    - $\delta : (q, a, X) \rightarrow (q_x, \Gamma\{\epsilon\})$

2. Autômato de Pilha não-Deteminístico

- Operações na Pilha:
  - Extração: $X \rightarrow \epsilon$
  - Não alterar: $X \rightarrow X$
  - Alterar o topo: $X \rightarrow Y$
  - Empilhar: $X \rightarrow YX$

2. Autômato de Pilha não-Deteminístico

- Projetando um APND:
- Entrada a ser reconhecida:
  - $E = \{ww^r \mid w=\{0,1\}^*\}$
- Definindo o APND:
  - $P = (\{q0,q1,q2\},\{0,1\},\{0,1,\$\},\delta,q0,\$,\{q2\})$

2. Autômato de Pilha não-Deteminístico

- Definindo a função δ:

  - $\delta(q0,0,\$) = \{(q0,0\$)\}$
  - $\delta(q0,1,\$) = \{(q0,1\$)\}$
  - $\delta(q0,0,0) = \{(q0,00)\}$
  - $\delta(q0,1,0) = \{(q0,10)\}$
  - $\delta(q0,0,1) = \{(q0,01)\}$
  - $\delta(q0,1,1) = \{(q0,11)\}$

  - $\delta(q0,\epsilon,\$) = \{(q1,\$)\}$
  - $\delta(q0,\epsilon,1) = \{(q1,1)\}$
  - $\delta(q0,\epsilon,0) = \{(q1,0)\}$

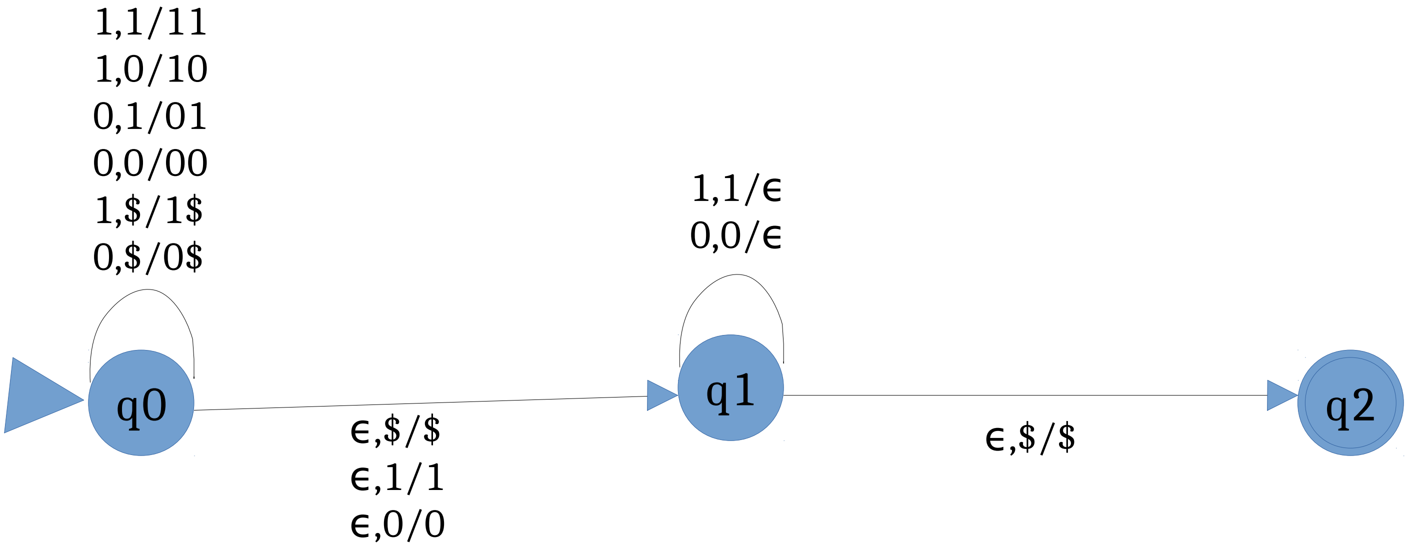
  - $\delta(q1,0,0) = \{(q1,\epsilon)\}$
  - $\delta(q1,1,1) = \{(q1,\epsilon)\}$

2. Autômato de Pilha não-Deteminístico

- Por fim definimos o estado de aceitação, ou seja, a função que faz o movimento para o estado de aceitação:
  - $\delta(q1,\epsilon,\$) = \{(q2,\$)\}$

2. Autômato de Pilha não-Deteminístico

- Representado graficamente:



2. Autômato de Pilha não-Deteminístico

- Analisando um automato:
  - Em APND, como a função de transição não define apenas um único movimento, a representação da execução se faz mais fácil quando em forma de arvore, onde cada bifurcação representa os multiplos movimentos no automato.

2. Autômato de Pilha não-Deteminístico

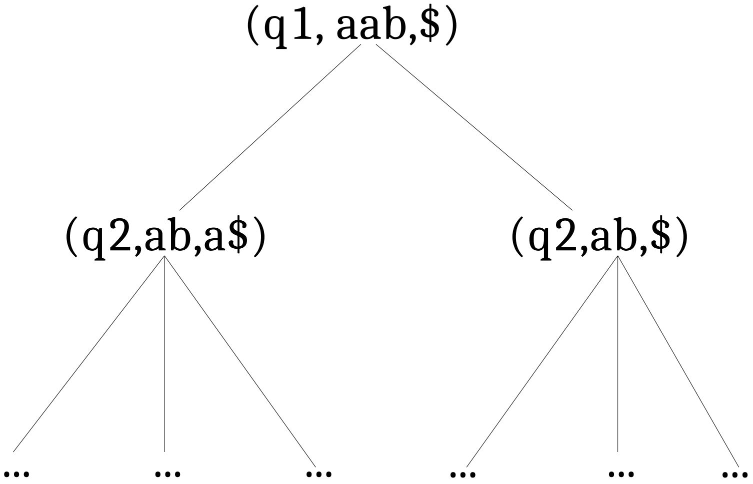
- Analisando um automato:
  - Dizemos que um automato aceitou (ou reconheceu) a entrada quando um de seus galhos atinge o estado de aceitação.

2. Autômato de Pilha não-Deteminístico

- Representando:
  - (q, w, x)
    - Onde q é estado atual
    - W é entrada que não foi processada ainda
    - X é o estado da pilha

2. Autômato de Pilha não-Deteministístico

- Exemplo:



3

Equivalência entre Autômatos de Pilha e Gramáticas Livres de Contexto

“

Temos que para qualquer GLC (Gramática Livre de Contexto) é possível definir um APND (Autômato de Pilha Não-Determinístico) que reconhece exatamente a mesma linguagem gerada pela gramática.

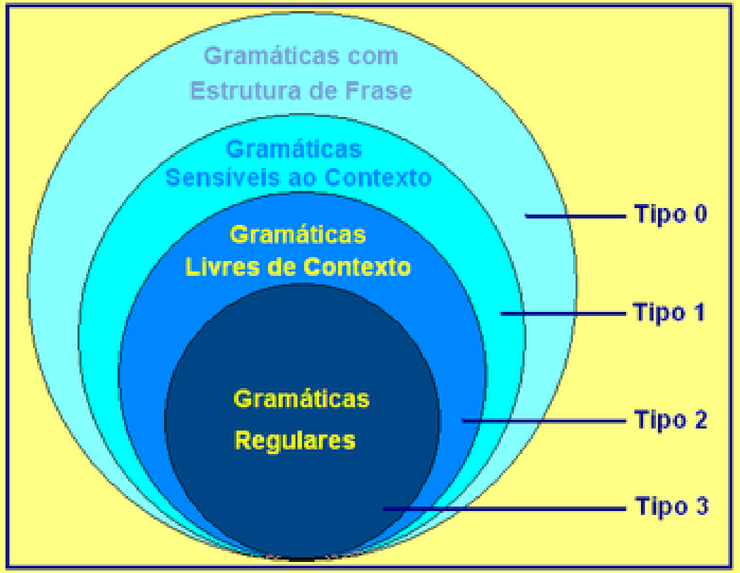
3.1. GLC → APND

- Aonde estão situadas as LLCs (Linguagens Livres de Contexto)?

3.1. GLC → APND

- Aonde estão situadas as LLCs (Linguagens Livres de Contexto)?

- Pela Hierarquia de Chomsky obeservamos:



3.1. GLC → APND

- Breve revisão sobre as GLCs:

  →Uma GLC pode ser definida pela tupla G=(V, ∑, R, S), na qual:

  – V = Conj. Finito de Variáveis Não Terminais
  – ∑ = Conj. Finito de Variáveis Terminais
  – R = Conj. Finito de Regras de Produção
  – S = Variável inicial (S ε V)

3.1. GLC → APND

- Breve revisão sobre as GLCs:

  → $G=(V, \Sigma, R, S)$

  – Para R podemos definir como:

  $x \to w$

  – x = cabeça da regra

  – w = corpo da regra

→ x é uma *única* variável não terminal

→ w é a união de todas as possíveis variáveis terminais e não terminais

3.1. GLC → APND

- Se existe uma GLC então existe um AP capaz de reconhecer a linguagem dessa gramática.

3.1. GLC → APND

- Se existe uma GLC então existe um AP capaz de reconhecer a linguagem dessa gramática.
- Usamos o APND com uma memória adicional em forma de pilha.

3.1. GLC → APND

- Se existe uma GLC então existe um AP capaz de reconhecer a linguagem dessa gramática.
- Usamos o APND com uma memória adicional em forma de pilha.
- *Como gerar esse Autômato?*

  GLC → AP

3.1. GLC → APND

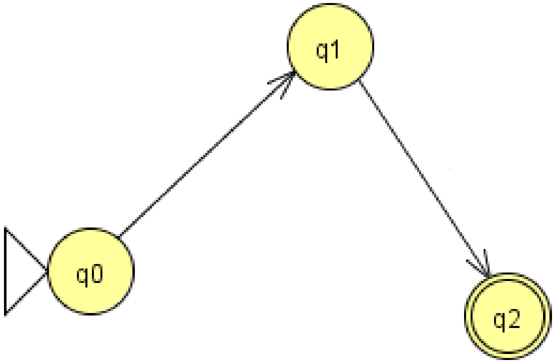
- Definimos G=(V, ∑, R, S):
  - V = (E , I)
  - ∑ = (0 , 1 , + , (,))
  - R =

Production
E→I
E→E+E
E→(E)
I→0
I→1

  - S = (E)

3.1. GLC → APND

- 1º Passo: criar um autômato com 3 estados iniciais.



- Sendo q0 estado inicial e q2 estado final

- 2º Passo: Adicionar transições para empilhar o elemento inicial. S = (E)

- 1ª: $\delta(q0, \lambda, \$) \rightarrow \{(q1, E\$)\}$

3.1. GLC → APND

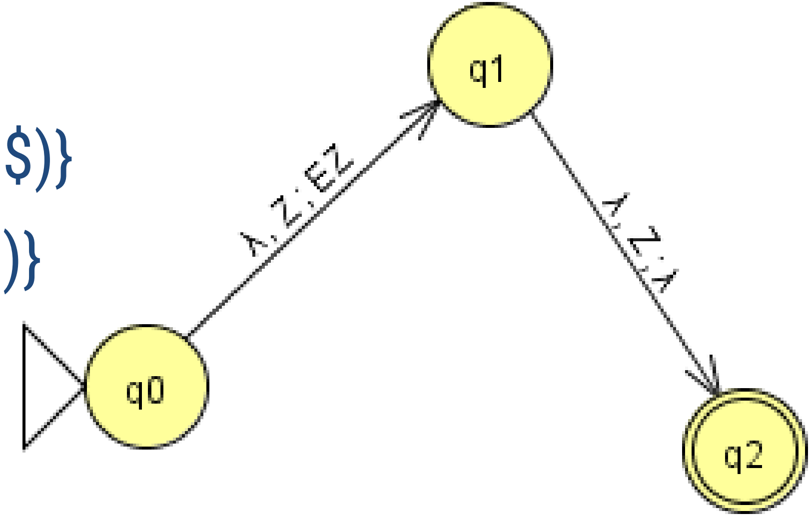
- 2º Passo: Adicionar transições para empilhar o elemento inicial. S = (E)

- 1ª: $\delta(q0, \lambda , \$) \rightarrow \{(q1 , E\$)\}$
- 2ª: $\delta(q1, \lambda , \$) \rightarrow \{(q2 , \$)\}$

3.1. GLC → APND

- 2º Passo: Adicionar transições para empilhar o elemento inicial. S = (E)

- 1ª: $\delta(q0, \lambda, \$) \rightarrow \{(q1, E\$)\}$
- 2ª: $\delta(q1, \lambda, \$) \rightarrow \{(q2, \$)\}$



3.1. GLC → APND

- 3º Passo: Para cada regra existente na gramática criamos, no autômato, uma transição equivalente, do tipo:

  → $(q1, \lambda, A) \rightarrow (q1, w)$

3.1. GLC → APND

- 3º Passo: Para cada regra existente na gramática criamos, no autômato, uma transição equivalente, do tipo:

→ (q1 , λ , A) → (q1 , w)

Production
E→I
E→E+E
E→(E)
I→0
I→1

3.1. GLC → APND

- Assim teremos:

→ (q1 , λ , E) → (q1 , I)

→ (q1 , λ , E) → (q1 , E+E)

→ (q1 , λ , E) → (q1 , (E))

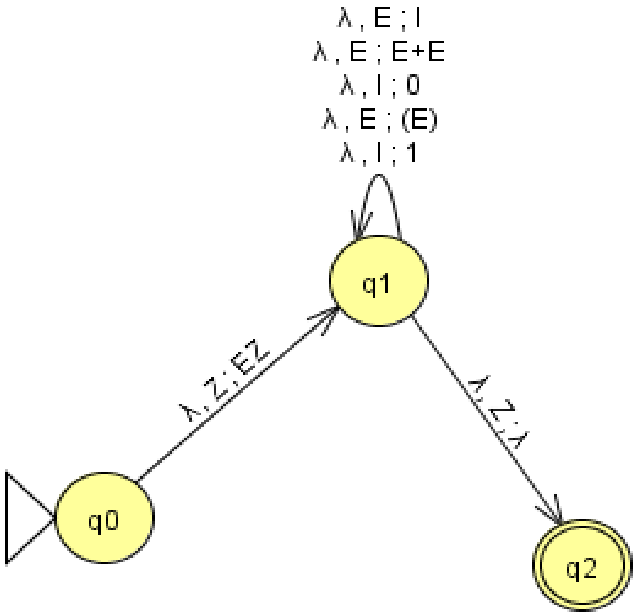
→ (q1 , λ , I) → (q1 , 0)

→ (q1 , λ , I) → (q1 , 1)

E → I
E → E + E
E → (E)
I → 0
I → 1

3.1. GLC → APND

- Assim teremos:



3.1. GLC → APND

- 4º Passo: Desempilhar os terminais. Criar, no autômato, uma transição para cada terminal [∑ = (0 , 1 , + , (,))] do tipo:

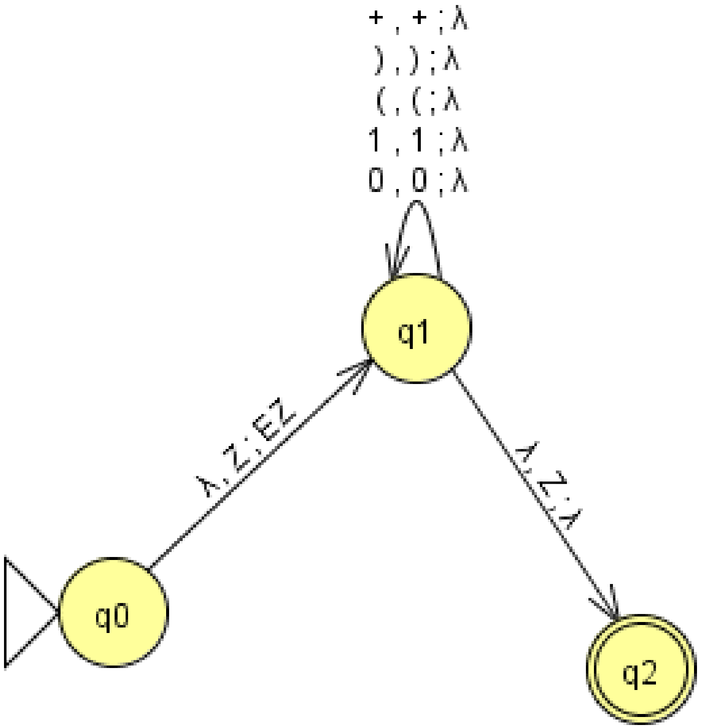
  → (q1 , t , t) -> (q1 , λ)

3.1. GLC → APND

- Assim teremos:
  - $\rightarrow$ (q1 , 0 , 0) $\rightarrow$ (q1 , λ)
  - $\rightarrow$ (q1 , 1 , 1) $\rightarrow$ (q1 , λ)
  - $\rightarrow$ (q1 , + , +) $\rightarrow$ (q1 , λ)
  - $\rightarrow$ (q1 , (, ( ) $\rightarrow$ (q1 , λ)
  - $\rightarrow$ (q1 ,) ,)) $\rightarrow$ (q1 , λ)

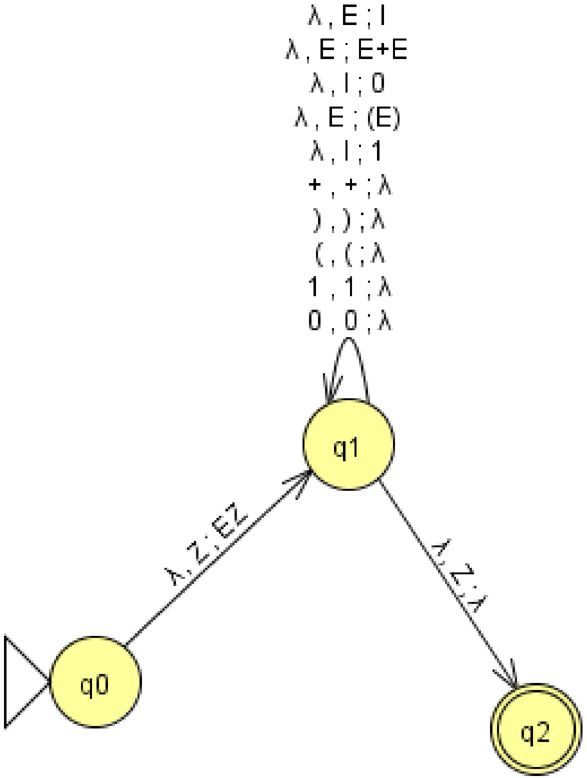
3.1. GLC → APND

- Assim teremos



3.1. GLC → APND

- E finalizamos o automato:



Dado um APD é possível construir um uma GLC que a reconheça?

3.2. APND → GLC

- O teorema da equivalência aceita que os Autômatos com Pilha Não Determinísticos são precisamente as linguagens livres de contexto.

- Podemos especificar um autômato P por pilha vazia

3.2. APND → GLC

- Teorema Seja um L = L(M) para alguma Autômatos com Pilha Não-Determinísticos M. Portanto L é uma linguagem livre de contexto.
- Seja M um APN de um estado. Então existe uma GLC G tal que L(G) = T(M).

3.2. APND → GLC

- L deve estar na forma normal de Greibach, (Sem que haja perda de generalidade). Onde existe a inserção de uma forma normal por vez, assim cada etapa da derivação produz um único terminal.

3.2. APND → GLC

- A→b
- A→bc1c2……cn
- A→bc1c2……cn
  - Onde A, c1,…….,cn são não terminais e b é Terminal.
- Ex: S → (L|λ, L → (LL|)

3.2. APND → GLC

- Seja o autômato de pilha A = (Q, Σ,Γ,δ,q,Z,F = Ø)

- 1- Para explicar uma gramática livre de contexto G = (V, T, P, S) que reconhece L(A).

  – Construir um conjunto de variáveis não terminais tal que V possui um elemento S inedito que seja utilizado como símbolo inicial.

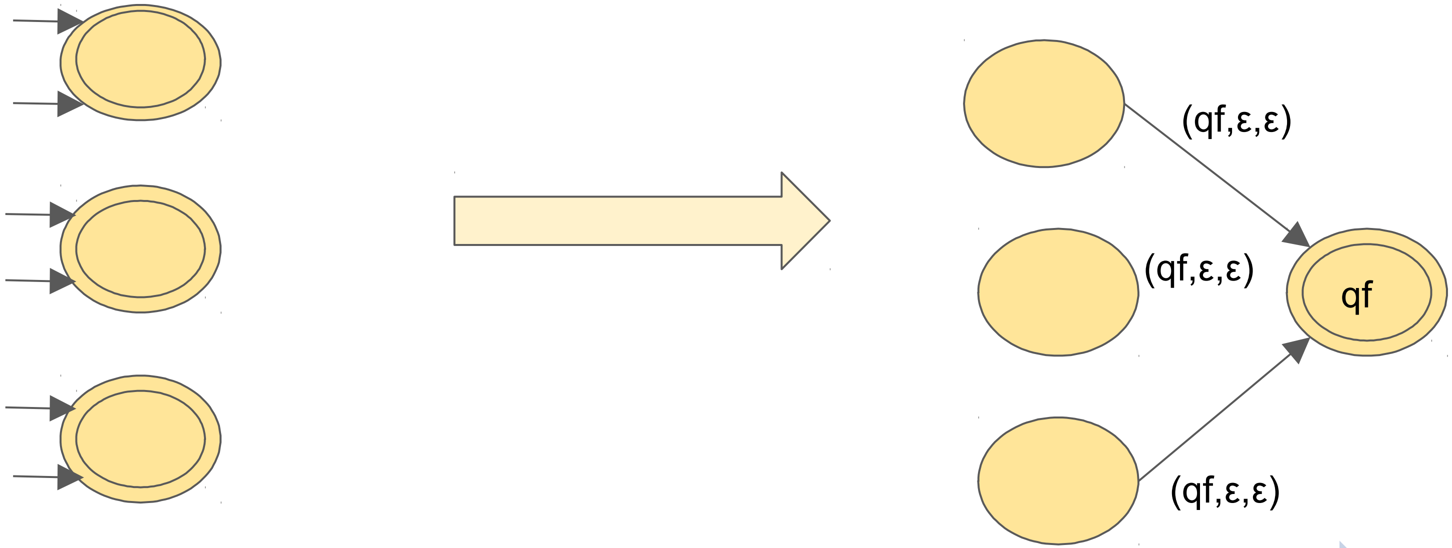
  – Para representar as variáveis temos: [q0,X, q1], onde q0 e q1 ∈ Q são estados do autômato e X ∈ Γ um símbolos da fita.

3.2. APND → GLC

- 2) Construir um conjunto de regras de substituição.
  - Para todo $X \in \Gamma$ teremos uma produção S→[q0,Z0,q1]
  - Se o estado δ(q0,a,X) contendo (r, Y1Y2Y3..Yk) todas as listas de estado serão gerado novos estados de dados por combinações [q0,Z0,q1] →a[rY1r1][r1Y1r2]...[rk-1Ykrk],

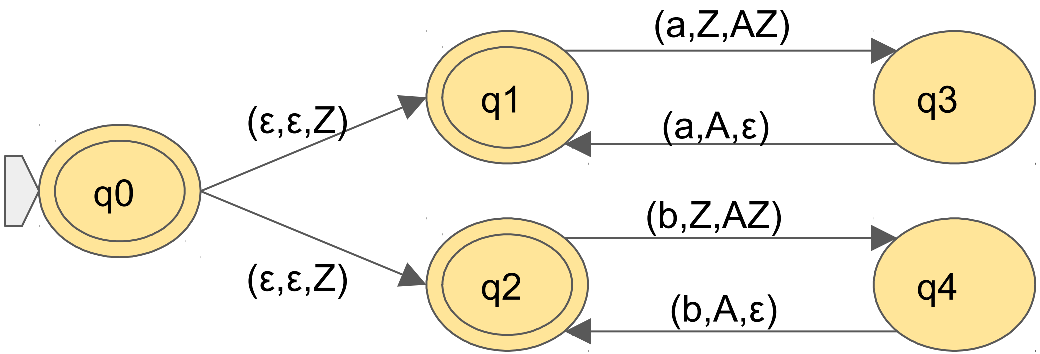
3.2. APND → GLC

No PDA só existirá apenas um estado final aceitável (qf).



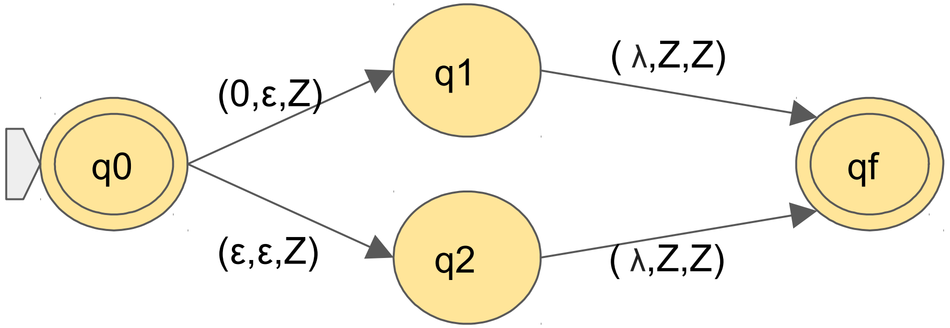
3.2. APND → GLC

- Exemplo: Existe mais de um estado final



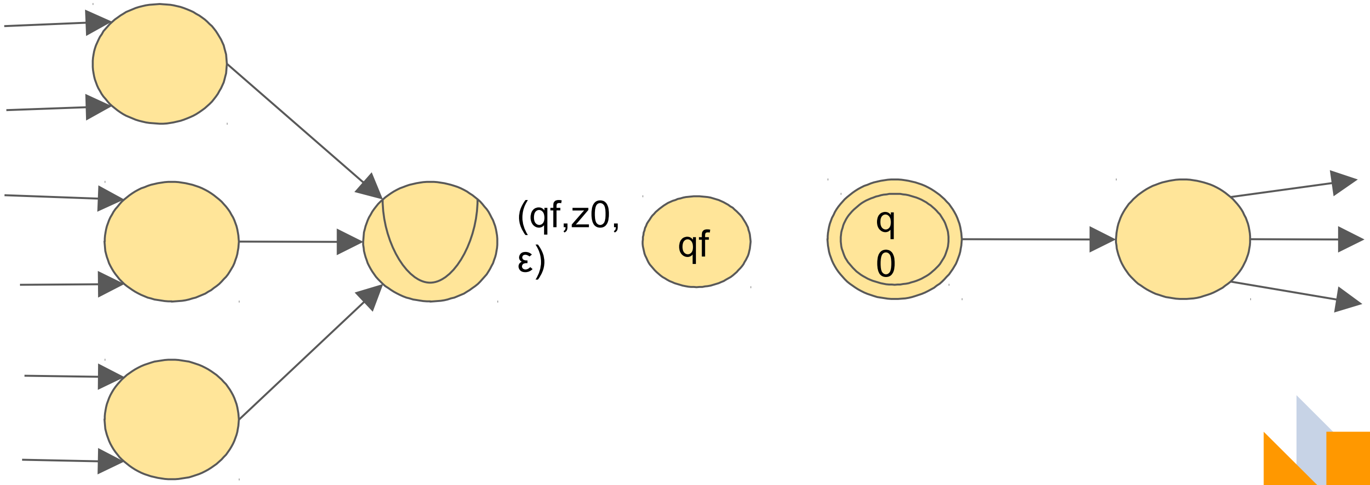
3.2. APND → GLC

- É necessário Criar um único estado final
- Transformar os estados q1, q2 em não finais
- Criar uma transição desempilhar z0 para o estado final.



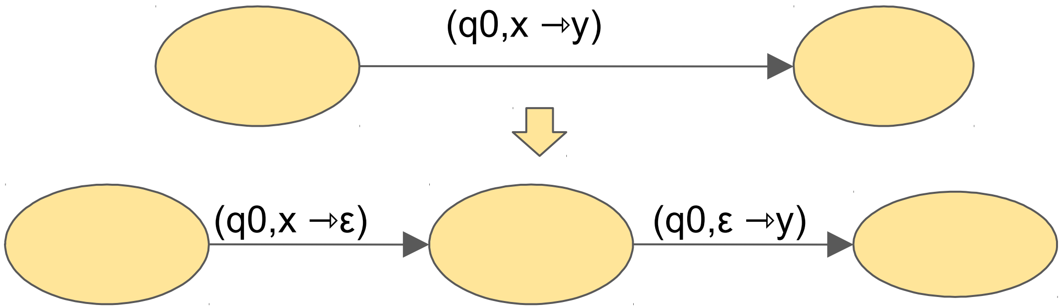
3.2. APND → GLC

2) O estado de Vazio deve se o primeiro a ser inserido na pilha.O estado final z0 só poderá ser acessado se e somente se a pilha estiver vazia.



3.2. APND → GLC

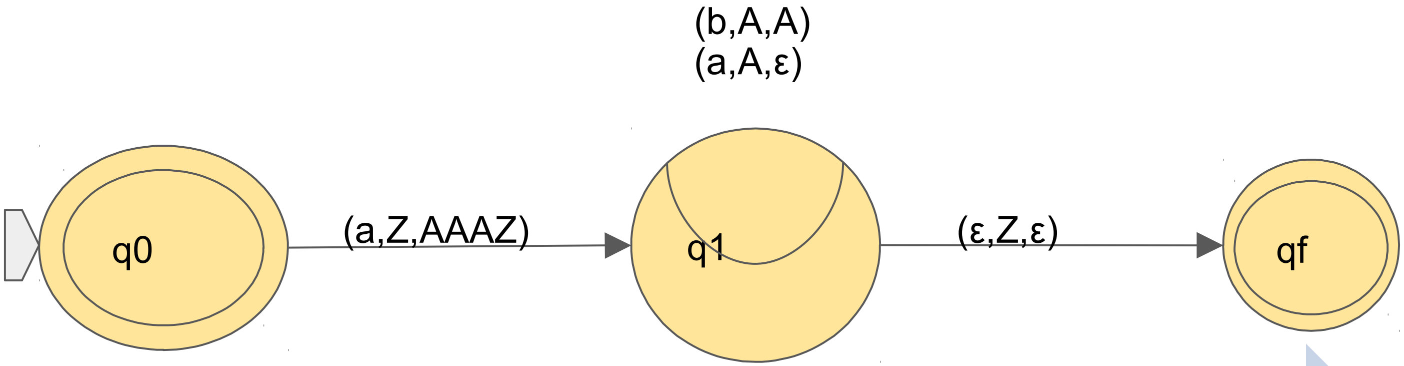
3) Deverá haver apenas acréscimo ou remoção, a cada operação.
 - Não pode haver uma operação dupla, ou que insira mais de um elemento na pilha.



3.2. APND → GLC

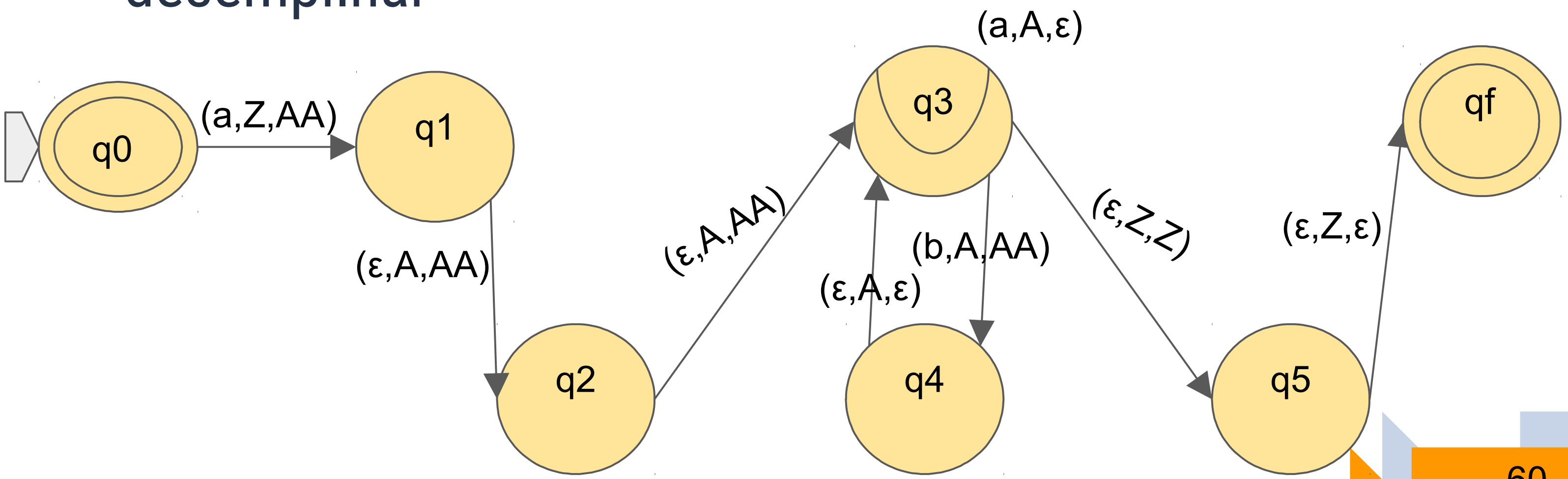
- $A = \{ a^{+}(b+a)^{*} \mid a = 4\}$, $Q = \{q0, q1, qf\}$, $\Sigma = \{a, b\}$, $\Gamma = \{Z, A\}$,

- EX; Empilha-se mais de um elemento durante a primeira transição:



3.2. APND → GLC

- É necessária a criação de estados intermediários, com um elemento por vez também para o caso de empilhar e desempilhar



3.2. APND → GLC

1. Definir variável de início composta de:(qi,Z,qf) [iniciar,fundo de pilha,final]

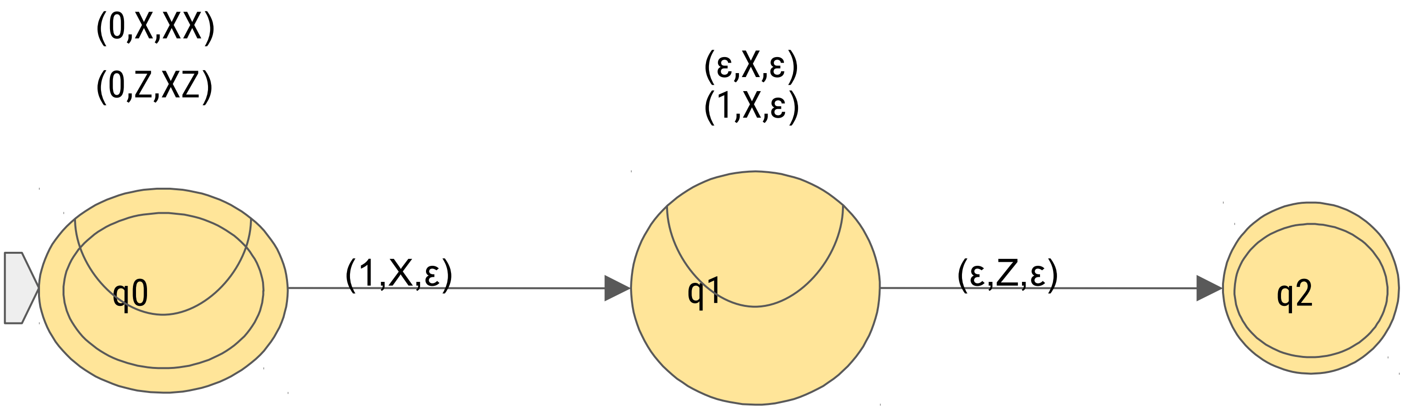
2. Identificar transições que que desempenham elementos: (qi, a, A)→ (qs,XZ), as quais geraram transições: (qi,A,qs) →(a).

3. Identificar as as transições que empilham elementos: (qi, a, A)→ (qs,ε), estas gerarão transições do tipo (qi,A,qs) → a(qs,X,qk)(qk,X,qs).

Percorrer todos os estados possíveis

3.2. APND → GLC

Ex: $A = \{ 0^m 1^n | m >= n \}$, $Q = \{q, r\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{Z, X\}$,



3.2. APND → GLC

$A = \{ 0^m 1^n \mid m >= n\}$, $Q = \{q0, q1\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{Z, X\}$,

Estado	Entrada	Pilha	Novo Estado	Pilha	
q0	0	Z	q0	XZ	empilhar
q0	0	X	q0	XX	empilhar
q0	1	X	q1	ε	desempilhar
q1	1	X	q1	ε	desempilhar
q1	ε	X	q1	ε	desempilhar
q1	ε	Z0	q2	ε	desempilhar

3.2. APND → GLC

$A = \{ 0^m 1^n | m >= n\}$, $Q = \{q0, q1\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{Z, X\}$,

Estado	Entrada	Pilha	Novo Estado	Pilha	
q0	0	Z	q0	XZ	empilhar
q0	0	X	q0	XX	empilhar
q0	1	X	q1	ε	desempilhar
q1	1	X	q1	ε	desempilhar
q1	ε	X	q1	ε	desempilhar
q1	ε	Z0	q2	ε	desempilhar

Estados possíveis:

[q0, X, q0] →	[q0, X, q0] [q0, X, q0]	[q0, X, q0] →	[q0, X, q1] [q1, X, q0]	[q0, Z, q2] →	[q0, X, q1] [q1, Z, q2]
[q0, X, q0] →	[q0, X, q2] [q2, X, q0]	[q0, X, q1] →	[q0, X, q0] [q0, X, q1]	[q0, Z, q2] →	[q0, X, q2] [q2, Z, q2]
[q0, X, q1] →	[q0, X, q1] [q1, X, q1]	[q0, X, q1] →	[q0, X, q2] [q2, X, q1]	[q1, X, q1] →	ε
[q0, X, q2] →	[q0, X, q0] [q0, X, q2]	[q0, X, q2] →	[q0, X, q1] [q1, X, q2]	[q0, X, q0] →	1
[q0, X, q2] →	[q0, X, q2] [q2, X, q2]	[q0, Z, q0] →	[q0, X, q0] [q0, Z, q0]	[q0, X, q1] →	1
[q0, Z, q0] →	[q0, X, q1] [q1, Z, q0]	[q0, Z, q0] →	[q0, X, q2] [q2, Z, q0]	[q1, Z, q2] →	ε
[q0, Z, q1] →	[q0, X, q0] [q0, Z, q1]	[q0, Z, q1] →	[q0, X, q1] [q1, Z, q1]		
[q0, Z, q1] →	[q0, X, q2] [q2, Z, q1]	[q0, Z, q2] →	[q0, X, q0] [q0, Z, q2]		

3.2. APND → GLC

Fazendo a simplificação, temos:

[q0, z0, q0] →	[q0, X, q0] [q0, z0, q0]	[q1, X, q1] →	ε
[q0, z0, q1] →	[q0, X, q0] [q0, z0, q1]	[q1, X, q2] →	ε
[q0, z0, q0] →	[q0, X, q1] [q1, z0, q0]		
[q0, z0, q1] →	[q0, X, q1] [q1, z0, q1]		

4

Questionário

Questão 01

Desenvolva Gramáticas Livre de Contexto (GLC) que gere a seguinte linguagen:

$L = \{a,b\}^*$

Questão 01 – Resolução

$G=(V,T,P,S)$

$V=\{S\}$

$T=\{a,b\}$

$P=\{S\rightarrow a \mid aS \mid b \mid bS \mid \varepsilon\}$

Questão 02

Desenvolva Gramáticas Livre de Contexto (GLC) que gere a seguinte linguagem:

$L = \{a^i b^{2i} \mid i >= 1\}$

Questão 02 – Resolução

G=(V,T,P,S)

V={S}

T={a,b}

P={S→ abb | aSbb}

Questão 03

Qual a diferença entre um autômato de pilha determinístico e um autômato de pilha não determinístico?

Questão 03 – Resolução

O APD e o APND são basicamente o mesmo,mas para o autômato ser determinístico, ele tem que respeitar duas restrições:a não existência de movimentação espontânea e a cada entrada só existir um movimento possível.

Questão 04

Dada a definição do automato abaixo, desenhe um diagrama de trasinções de estado (grafo):

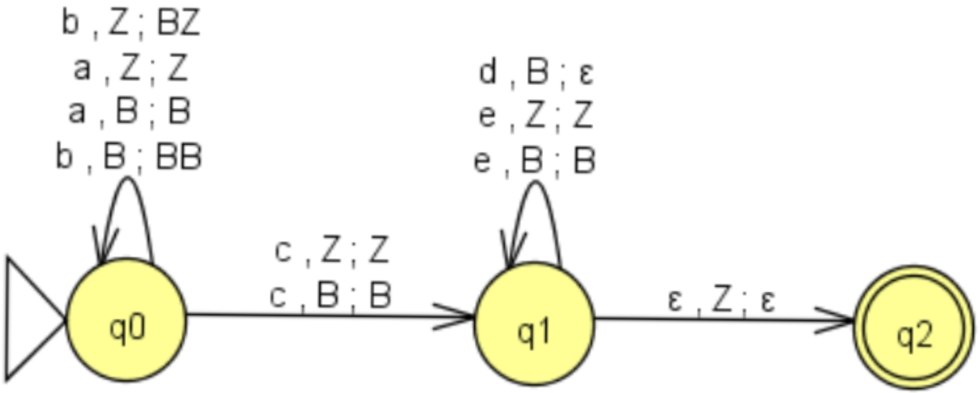
- $(Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

$Q = \{q0,q1,q2\}$; $\Sigma = \{a,b,c,d,e\}$; $\Gamma = \{Z,B\}$; $Z_0 = \varepsilon$; $F = \{q2\}$

Questão 04

- $\delta = \{(q0,a,Z) \rightarrow (q0,Z); (q0,a,B) \rightarrow (q0,B); (q0,b,Z) \rightarrow (q0,BZ); (q0,b,B) \rightarrow (q0,BB); (q0,c,Z) \rightarrow (q1,Z); (q0,c,B) \rightarrow (q1,B); (q1,d,B) \rightarrow (q1,\varepsilon); (q1,e,Z) \rightarrow (q1,Z); (q1,e,B) \rightarrow (q1,B); (q1,\varepsilon,Z) \rightarrow (q2,\varepsilon)\}$

Questão 04 – Resolução



Questão 05

Dado um Autômato de pilha não-determinístico, defina quando este, por sua vez, aceita uma cadeia de simbolos.

Questão 05 – Resolução

Dado que durante a execução de APND, este se ramifica em vários galhos, é dito que uma cadeia é aceita quando pelo menos uma de suas bifucaçoes alcança o estado final de aceitação consumindo assim toda a cadeia de simbolos.

5

Referências Bibliográficas

5. Referências Bibliográficas

- [01] Introdução à teoria de autômatos, linguagens e computação/ Hopcroft, John E. and Motwani, Rajeev. and Ullman, Jeffrey D. -2002 Chapter 5 e 6 - introduction to automata theory, languages, and computation.

5. Referências Bibliográficas

- [02] Sipser, Michael -2007 - Chapter 2 - introduction to theory of computation 2ª E.d.

5. Referências Bibliográficas

- [03] M.V.M. Ramos, J.J. Neto e I.S Linguagens Formais: Teoria,Modelagem e Implementação , Vega Bookman, 2009.

5. Referências Bibliográficas

- [04] RANGEL, José Lucas. Linguagens livres de contexto e autômatos de pilha, maio 1999.