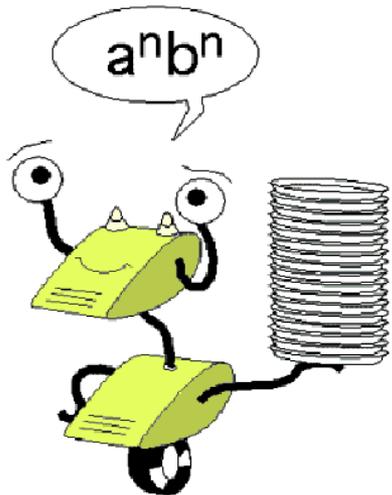


Autômatos com Pilha

a^n



$a^n b^n$



Autômatos com Pilha: Definição Informal e Definição Formal

Linguagem Aceita por um ACP

ACPDet X ACPND

Notação gráfica para ACP

ACP

- Assim como LR tem um autômato equivalente (AF) as LLC tem também uma máquina (ACP).
- A equivalência é menos geral desde que o ACP é **não determinístico**.
- A versão determinística aceita somente um subconjunto das LLC. Isto é, existem, LLC que não são reconhecidas por ACPDet.
 - Exemplo clássico é $L = ww^R$ que é aceito por um ACPND mas não por um ACPDet.
- Entretanto esse conjunto inclui a maioria das Linguagens de Programação.

Hierarquia das Classes de Máquinas e Linguagens

L Recursivamente Enumeráveis/Máquinas de Turing que Reconhecem L

L Recursivas/Máquinas de Turing que Decidem L

L Livres de Contexto/Autômatos a Pilha não Determinísticas

L Livres de Contexto Determinísticas/
Autômatos a Pilha Determinísticas

L Regulares/Autômatos Finitos

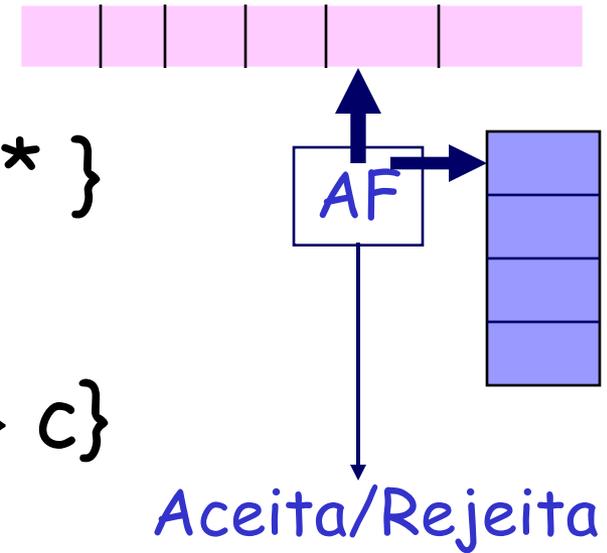
ACP - Definição Informal

- É um AF com uma fita de entrada e uma pilha com símbolos de um dado alfabeto.

Exemplo: $L = \{w c w^R \mid w \in \{0,1\}^*\}$

gerada por $G = (\{S\}, \{0,1,c\}, P, S)$

$P = \{ S \rightarrow 0S0 \mid S \rightarrow 1S1 \mid S \rightarrow c \}$



Exemplos de cadeias aceitas: c , $01c10$, $00c00$

O ACP que reconhece esta linguagem terá:

- Controle Finito com 2 estados: q_1 (empilha) e q_2 (desempilha)
- Pilha com pratos: vermelho, azul (0), verde (1)
- Regras:
 - 1) Começa com prato vermelho na pilha e estado q_1
 - 2) Entrada 0, estado $q_1 \rightarrow$ empilha azul } Permanece em q_1
 - Entrada 1, estado $q_1 \rightarrow$ empilha verde }
 - 3) Entrada c, estado $q_1 \rightarrow$ não altera a pilha muda para q_2
 - 4) Entrada 0, estado q_2 , topo azul \rightarrow desempilha } Permanece em q_2
 - 5) Entrada 1, estado q_2 , topo verde \rightarrow desempilha }
 - 6) Estado q_2 , vermelho \rightarrow desempilha sem ver entrada
 - 7) Para os outros casos não descritos, o dispositivo não faz nenhum movimento

- Este dispositivo aceita uma cadeia de entrada se: **ao processar o último símbolo a pilha fica vazia.**
- Exemplos:
 - 1) $w = 01c10$

TopoPilha	Estado	Entrada	Pilha
Verm	q1	0	
Azul	q1	1	
Verde	q1	c	verde
Verde	q2	1	azul
Azul	q2	0	verm
Verm	q2	-	

2) $w = 101c100$

TopoPilha	Estado	Entrada	Pilha
Verm	q_1	1	
Verde	q_1	0	verde
Azul	q_1	1	azul
Verde	q_1	c	verde
Verde	q_2	1	verm
Azul	q_2	0	
Verde	q_2	0 ação ???	

ACP - Definição Formal

- OACP terá uma fita de entrada, um controle finito e uma pilha que contém uma cadeia de símbolos de algum alfabeto.
- O símbolo mais à esquerda será considerado estar no TOPO.
- O dispositivo será **não determinístico**, pois terá algum número finito de escolhas de movimentos

2 Tipos de Movimentos

1. Um símbolo de entrada é lido

e, dependendo (do símbolo de entrada, topo da pilha, estado) realiza 1 escolha dentre as possíveis.

Cada escolha consiste (estado seguinte, cadeia de símbolos (pode ser vazia) para substituir o topo).

Depois da escolha, a cabeça avança.

2. Chamado movimento- λ

similar ao primeiro, exceto que o símbolo de entrada não é usado e a cabeça não avança depois do movimento.

Esse tipo de movimento permite o ACP manipular a pilha sem ler símbolos de entrada.

Linguagem aceita por um ACP

1. Conjunto de todas as entradas para as quais alguma seqüência de movimentos faz com que a pilha fique vazia → Linguagem aceita por Pilha vazia
2. Similar a AF: definimos alguns estados como finais e definimos a linguagem aceita como o conjunto de todas as entradas para as quais alguma escolha de movimento faz o ACP entrar num estado final → Linguagem aceita por estados finais.

As duas formas acima são equivalentes.

Ver (Menezes, 2002) pg 112

Um ACP M é uma sétupla $(K, \Sigma, \Gamma, \delta, q_0, z_0, F)$ onde:

(H,M,U, 2001)

1. K é um conjunto finito de estados
2. Σ (sigma) é um alfabeto finito chamado alfabeto de entrada
3. Γ (gama) é um alfabeto finito chamado alfabeto da pilha
4. $q_0 \in K$ é o estado inicial. A máquina começa nele.
5. $z_0 \in \Gamma$ é o símbolo inicial da pilha. Aparece inicialmente na pilha.
6. F é o conjunto de estados finais $F \subseteq K$
7. δ (delta) é um mapeamento de

$$K \times (\Sigma \cup \{\lambda\}) \times \Gamma \rightarrow \text{subconjuntos de } K \times \Gamma^* \\ \text{(topo)}$$

OBS: quando o ACP aceita por pilha vazia $F = \emptyset$

Interpretação de δ

$$1. \quad \delta (q, a, z) \rightarrow \{(p_1, \gamma_1), (p_2, \gamma_2), \dots (p_m, \gamma_m)\}$$

$q, p_i \in K, \quad a \in \Sigma, \quad z \in \Gamma^*$

ACP no estado q , com símbolo de entrada a e z no topo da pilha pode, para qualquer i , mudar para o estado p_i , substituir z por γ_i (**substitui por uma cadeia de símbolos da pilha**) e avançar a cabeça de leitura.

$$2. \quad \delta (q, \lambda, z) \rightarrow \{(p_1, \gamma_1), (p_2, \gamma_2), \dots (p_m, \gamma_m)\}$$

Semelhante a anterior, só que não espera símbolo de entrada e não avança a cabeça de leitura.

OBS 1: Se $\gamma_i = \lambda$ então há desempilhamento; se $\gamma_i = z$ então a pilha fica inalterada; se $\gamma_i = yz$ então y é empilhado.

OBS 2: Um ACP não pode fazer um movimento se a pilha estiver vazia.

- Formalize o ACP do exemplo anterior.
Deve aceitar por pilha vazia

- Controle Finito com 2 estados: $q1$ (empilha) e $q2$ (desempilha)
- Pilha com pratos: vermelho, azul (0), verde (1)
- Regras:
- Começa com prato vermelho na pilha e estado $q1$
- Entrada 0, estado $q1 \rightarrow$ empilha azul
- Entrada 1, estado $q1 \rightarrow$ empilha verde
- Entrada c , estado $q1 \rightarrow$ não altera a pilha muda para $q2$
- Entrada 0, estado $q2$, topo azul \rightarrow desempilha
- Entrada 1, estado $q2$, topo verde \rightarrow desempilha
- Estado $q2$, vermelho \rightarrow desempilha sem ver entrada
- Para os outros casos não descritos, o dispositivo não faz nenhum movimento

Permanece em $q1$

Permanece em $q2$

- $M = (\{q1, q2\}, \{0, 1, c\}, \{Vm, Az, Vd\}, \delta, q1, Vm, \emptyset)$

- | | |
|---|---|
| 1. $\delta(q1, 0, Vm) = \{(q1, AzVm)\}$ | 7. $\delta(q1, c, Vm) = \{(q2, Vm)\}$ |
| 2. $\delta(q1, 0, Az) = \{(q1, AzAz)\}$ | 8. $\delta(q1, c, Az) = \{(q2, Az)\}$ |
| 3. $\delta(q1, 0, Vd) = \{(q1, AzVd)\}$ | 9. $\delta(q1, c, Vd) = \{(q2, Vd)\}$ |
| 4. $\delta(q1, 1, Vm) = \{(q1, VdVm)\}$ | 10. $\delta(q2, 0, Az) = \{(q2, \lambda)\}$ |
| 5. $\delta(q1, 1, Az) = \{(q1, VdAz)\}$ | 11. $\delta(q2, 1, Vd) = \{(q2, \lambda)\}$ |
| 6. $\delta(q1, 1, Vd) = \{(q1, VdVd)\}$ | 12. $\delta(q2, \lambda, Vm) = \{(q2, \lambda)\}$ |

Todos os conjuntos só possuem 1 elemento e δ não é definida para $\delta(q2, 0, Vm)$ e $\delta(q2, 1, Vm)$, pois é definida para $\delta(q2, \lambda, Vm)$

ACPDet

- O ACP do exemplo é **determinístico** no sentido que, no máximo um movimento é possível de qualquer configuração. Podemos tirar o $\{e\}$
- **Formalmente, dizemos que um ACP $M = (K, \Sigma, \Gamma, \delta, q_0, z_0, F)$ é determinístico se:**

1. Para cada $q \in K$ e $z \in \Gamma$ se $\delta(q, \lambda, z)$ é não vazio então $\delta(q, a, z)$ é vazio para $\forall a \in \Sigma$

(impede a possibilidade de escolha entre um mov- λ e um envolvendo um símbolo de entrada)

2. $\delta(q, a, z)$ contém um só elemento para $\forall q \in K, a \in \Sigma \cup \{\lambda\}, z \in \Gamma$

(impede a escolha de movimento tanto para (q, a, z) como para (q, λ, z))

- Façam um ACP não-determinístico que reconhece $L = \{ww^R \mid w \in \{0,1\}^*\}$ por pilha vazia
- Vejam que aqui o ACP deve aceitar a cadeia vazia.

$$M = (\{q1, q2\}, \{0, 1, c\}, \{Vm, Az, Vd\}, \delta, q1, Vm, \emptyset)$$

1. $\delta(q1, 0, Vm) = \{(q1, AzVm)\}$
2. $\delta(q1, 1, Vm) = \{(q1, VdVm)\}$
3. $\delta(q1, 0, Az) = \{(q1, AzAz), (q2, \lambda)\}$
4. $\delta(q1, 0, Vd) = \{(q1, AzVd)\}$
5. $\delta(q1, 1, Az) = \{(q1, VdAz)\}$
6. $\delta(q1, 1, Vd) = \{(q1, VdVd), (q2, \lambda)\}$
7. $\delta(q2, 0, Az) = \{(q2, \lambda)\}$
8. $\delta(q2, 1, Vd) = \{(q2, \lambda)\}$
9. $\delta(q1, \lambda, Vm) = \{(q2, \lambda)\}$
10. $\delta(q2, \lambda, Vm) = \{(q2, \lambda)\}$

- M aceita a cadeia vazia pela regra 9
- O ACP é **não determinístico** pois:
 - As regras 3 e 6 possuem uma escolha de movimento.
 - Se M decide que o meio da cadeia de entrada foi alcançado então escolhe a segunda opção do conjunto e vai para o estado q2.
 - δ é definida para $\delta(q1, \lambda, Vm)$ e também para $\delta(q1, 0, Vm)$ e $\delta(q1, 1, Vm)$

Configuração de um ACP e Mudança de configuração

- Uma configuração de um ACP M é um par (q, γ) onde $q \in K$ e $\gamma \in \Gamma^*$
- Mudança de configuração:

Seja $a \in \Sigma \cup \{\lambda\}$, γ e $\beta \in \Gamma^*$, $z \in \Gamma$ e $\delta(q, a, z) = (p, \beta)$ então:

$$a: (q, z\gamma) \underset{M}{|--} (p, \beta\gamma)$$

Significa que, de acordo com as regras do ACP, a entrada a faz com que M vá da configuração $(q, z\gamma)$ para $(p, \beta\gamma)$.

Estendendo para cadeia:

$$a_1 a_2 \dots a_n: (q_1, \gamma_1) \underset{M}{|--}^* (q_{n+1}, \gamma_{n+1})$$

$$\text{Por convenção: } \lambda: (q, \gamma) \underset{M}{|--}^* (q, \gamma)$$

(H, M, U, 2001) pg 224 define configuração como uma tripla pois embute a cadeia de entrada no par acima. Difere somente a notação!

Linguagem Aceita

- Para um ACP M , definimos $L(M)$, a linguagem aceita por **estado final** como:

$$L(M) = \{w \mid w:(q_0, z_0) \xrightarrow[M]{*} (q, \gamma) \text{ para } \forall \gamma \in \Gamma^* \text{ e } q \in F\}$$

- Para um ACP M nós definimos $N(M)$, a linguagem aceita por **pilha vazia** como:

$$N(M) = \{w \mid w:(q_0, z_0) \xrightarrow[M]{*} (q, \lambda) \text{ para } \forall q \in K\}$$

O N em $N(M)$ significa null stack = empty stack

OBS 1: quando aceitamos por pilha vazia o conjunto de estados finais é irrelevante.

OBS 2: As duas definições são equivalentes: ver Teo 6.9 e Teo 6.11 em (H,M,U, 2001)

Exercício 1

- Encontre um ACP M que reconheça o conjunto $L = \{ w \mid w \in \{0,1\}^* \text{ e } w \text{ consiste de } nro(1) = nro(0) \}$ por pilha vazia.
- Diga se o ACP é ACPDET ou ACPND
- Mostrar as configurações do ACP com a entrada 0101

Dica:

- associar 0 com Az e 1 com Vd
- "matar" 0 com 1 e 1 com 0
- fazer regra para aceitar cadeia vazia

- Qual seria a *GLC* que reconhece esta linguagem?

Qual seria a *GLC* que reconhece esta linguagem?

$$G = (\{S, A\}, \{0, 1\}, P, S)$$

$$P = \{S \rightarrow \lambda \mid A$$

$$A \rightarrow 0A1 \mid 1A0 \mid 01 \mid 10 \mid A01 \mid A10 \}$$

OU

$$P = \{S \rightarrow \lambda \mid A$$

$$A \rightarrow 0A1 \mid 1A0 \mid 01 \mid 10 \mid 01A \mid 10A \}$$

As regras 7 e 8 precisam estar lá para reconhecer, p.e., 0110 e 1001

$$M = (\{q1\}, \{0,1\}, \{Vm, Az, Vd\}, \delta, q1, Vm, \emptyset)$$

$$1. \quad \delta(q1, 0, Vm) = \{(q1, AzVm)\}$$

$$2. \quad \delta(q1, 1, Vm) = \{(q1, VdVm)\}$$

$$3. \quad \delta(q1, 0, Az) = \{(q1, AzAz)\}$$

$$4. \quad \delta(q1, 1, Vd) = \{(q1, VdVd)\}$$

$$5. \quad \delta(q1, 1, Az) = \{(q1, \lambda)\}$$

$$6. \quad \delta(q1, 0, Vd) = \{(q1, \lambda)\}$$

$$7. \quad \delta(q1, \lambda, Vm) = \{(q1, \lambda)\}$$

ACPND pelas regras 1, 2 e 7 pois temos a chance de ler um símbolo da entrada ou usar um mov- λ .

Configurações para 0101

Entrada	Configuração
	$(q1, Vm)$
0	$(q1, AzVm)$
01	$(q1, Vm)$
010	$(q1, AzVm)$
0101	$(q1, Vm)$
λ	$(q1, \lambda)$

Exercício 2

- Encontre um ACP M que reconheça o conjunto $L = \{ 0^n 1^n \mid n > 0 \}$ por pilha vazia.
- Diga se o ACP é ACPDET ou ACPND

Notação gráfica para ACP

- O diagrama de transição para ACP que aceitam a linguagem por estado final segue:
- Os nós correspondem aos estados do ACP
- Existem o estado inicial e os finais
- Um arco rotulado com $a, X; \alpha$ do estado q para p significa que $\delta(q, a, X)$ contém o par (p, α) entre os pares.
- Convencionalmente, Z_0 é o símbolo da pilha (no JFLAP é Z).
- Leia-se: simbolo, topo_antigo;novo_topo

JFLAP mostrando a Configuração Inicial

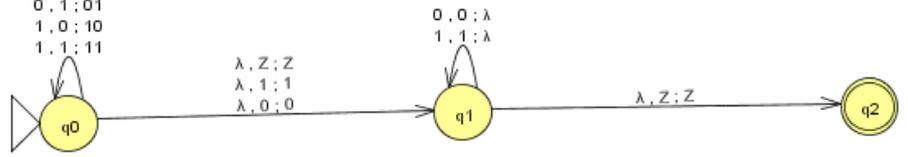
The screenshot displays the JFLAP software interface. The main window shows a finite automaton with three states: q_0 (start state, green circle), q_1 (yellow circle), and q_2 (final state, double yellow circle). The transitions are as follows:

- From q_0 to q_0 : $0, Z; 0Z$, $1, Z; 1Z$, $0, 0; 00$, $0, 1; 01$, $1, 0; 10$, $1, 1; 11$
- From q_0 to q_1 : $\lambda, Z; Z$, $\lambda, 1; 1$, $\lambda, 0; 0$
- From q_1 to q_1 : $0, 0; \lambda$, $1, 1; \lambda$
- From q_1 to q_2 : $\lambda, Z; Z$

The bottom panel shows the initial configuration: state q_0 , input string 0110 , and stack Z .

$L = \{ww^R \mid w \text{ pertence a } \{0,1\}^*\}$
Cadeia submetida 0110
 Z é o símbolo inicial da pilha

0, Z : 0Z
 1, Z : 1Z
 0, 0 : 00
 0, 1 : 01
 1, 0 : 10
 1, 1 : 11



Input	Result
10	Reject
01	Reject
1001	Accept
11000011	Accept
	Accept

$$L = \{ww^R \mid w \text{ pertence a } \{0,1\}^*\}$$

JFLAP mostrando o tracing de aceitação

The screenshot shows the JFLAP interface with a Turing machine diagram and an acceptance tracing window. The Turing machine has three states: q_0 , q_1 , and q_2 . The transitions are:

- q_0 to q_0 : $0, Z; 0Z$, $1, Z; 1Z$, $0, 0; 00$, $0, 1; 01$, $1, 0; 10$, $1, 1; 11$
- q_0 to q_1 : $\lambda, Z; Z$, $\lambda, 1; 1$, $\lambda, 0; 0$
- q_1 to q_1 : $0, 0; \lambda$, $1, 1; \lambda$
- q_1 to q_2 : $\lambda, Z; Z$

The acceptance tracing window, titled "Accepting configuration found!", shows the sequence of configurations for the input string "11":

- Configuration 1: q_0 | 11 | Z
- Configuration 2: q_0 | 11 | 1Z
- Configuration 3: q_1 | 11 | 1Z
- Configuration 4: q_1 | 11 | Z
- Configuration 5: q_2 | 11 | Z

The text "Cadeia submetida: 11" is displayed in the center, and the regular expression $L = \{ww^R \mid w \text{ pertence a } \{0,1\}^*\}$ is shown at the bottom.

Cadeia submetida: 11

$$L = \{ww^R \mid w \text{ pertence a } \{0,1\}^*\}$$

JFlap mostrando a aceitação de cadeia

JFLAP : (acp2.jff)

File Input Test Convert Help

Editor Simulate: 01c10

0, Z ; 0Z
1, Z ; 1Z
0, 0 ; 00
0, 1 ; 01
1, 0 ; 10
1, 1 ; 11

c, 1 ; 1
c, 0 ; 0
c, Z ; Z

0, 0 ; λ
1, 1 ; λ

λ, Z ; Z

q0 q1 q2

q2 01c10

z

$L = \{wcw^R \mid w \text{ pertence a } \{0,1\}^*\}$

Cadeia submetida 01c10

Z é o símbolo inicial da pilha

Step Reset Freeze Thaw Trace Remove

Iniciar | Sce185_2005_A_B | Microsoft PowerPoint - [...] | JFLAP : (acp2.jff) | 17:14

Jflap mostrando rejeição de cadeia

The screenshot shows the JFLAP software interface. At the top, the title bar reads "JFLAP : (acp2.jff)". Below it is a menu bar with "File", "Input", "Test", "Convert", and "Help". The main window is titled "Editor" and "Simulate: 11".

The main area displays a finite automaton with three states: q_0 (start state, green circle), q_1 (yellow circle), and q_2 (final state, double yellow circle). The transitions are as follows:

- q_0 to q_0 : $0, Z ; 0Z$, $1, Z ; 1Z$, $0, 0 ; 00$, $0, 1 ; 01$, $1, 0 ; 10$, $1, 1 ; 11$
- q_0 to q_1 : $c, 1 ; 1$, $c, 0 ; 0$, $c, Z ; Z$
- q_1 to q_1 : $0, 0 ; \lambda$, $1, 1 ; \lambda$
- q_1 to q_2 : $\lambda, Z ; Z$

Below the diagram is a simulation window. On the left, a stack is shown with the top element being q_0 and the string $11Z$. On the right, the text reads:

$L = \{wcw^R \mid w \text{ pertence a } \{0,1\}^*\}$
Cadeia submetida 11
 Z é o símbolo inicial da pilha

At the bottom of the JFLAP window are buttons for "Step", "Reset", "Freeze", "Thaw", "Trace", and "Remove". The Windows taskbar at the very bottom shows the "Iniciar" button and several open applications, including "Microsoft PowerPoint - [...]" and "JFLAP : (acp2.jff)". The system clock shows "17:15".