Propriedades de Fecho de Linguagens Regulares.

Gerando Linguagens Regulares

Recorde a seguinte teorema:

THM: Linguagens regulares são aquelas que podem ser geradas a partir de linguagens finitas pela aplicação de operações regulares.

Em particular, o teorema implica que, quando aplicamos uma operação regular a linguagens regulares o resultado é uma linguagem regular. I.e., o conjunto das linguagens regulares é ***fechado*** sob as operações de *união*, *concatenação* e *Kleene-**.

Fecho de Linguagens Regulares

OBJETIVO: Mostrar que o conjunto das linguagens regulares é *fechado* sob operações regulares. I.e., dadas linguagens regulares L_1 e L_2, mostrar:

1. $L_1 \cup L_2$ é regular,
2. $L_1 \bullet L_2$ é regular,
3. L_1^* é regular.

2 e 3 serão provados adiante, depois de vermos NFA's. Já provamos 1.

Outras Construções

A ***diferença*** de dois conjuntos é definida por

$$A - B = \{X \in A \mid X \notin B\}$$

Q: Como modificar a construção da união/interseção para a diferença de duas linguagens?

Diferença

R: Aceite o string apenas quando o primeiro autômato aceita e o segundo não. I.e.

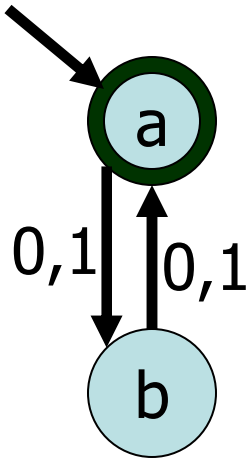
$$M_{-} = (Q_1 \times Q_2, \Sigma, \delta_1 \times \delta_2, (q_{0,1}, q_{0,2}), F_{-})$$

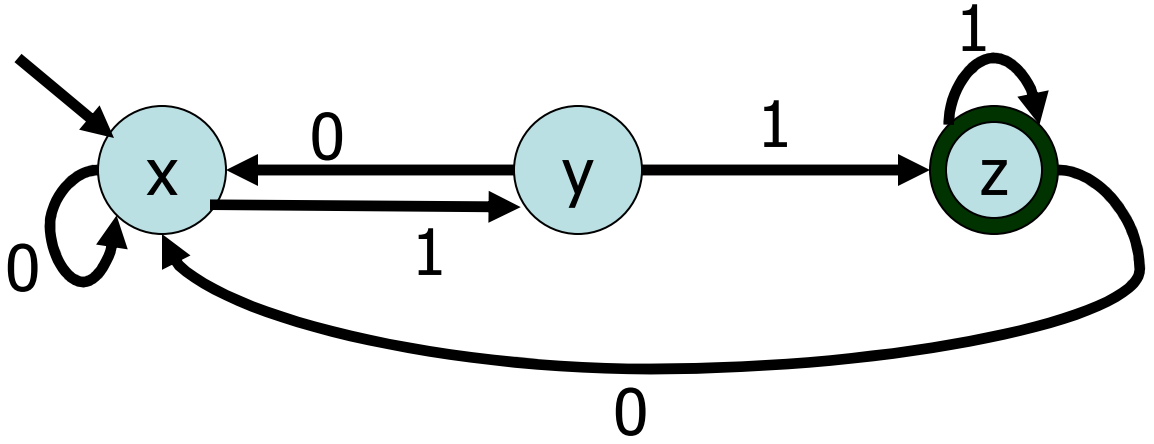
onde $F_{-} = F_1 \times Q_2 - Q_1 \times F_2$

Aplicando ao exemplo:

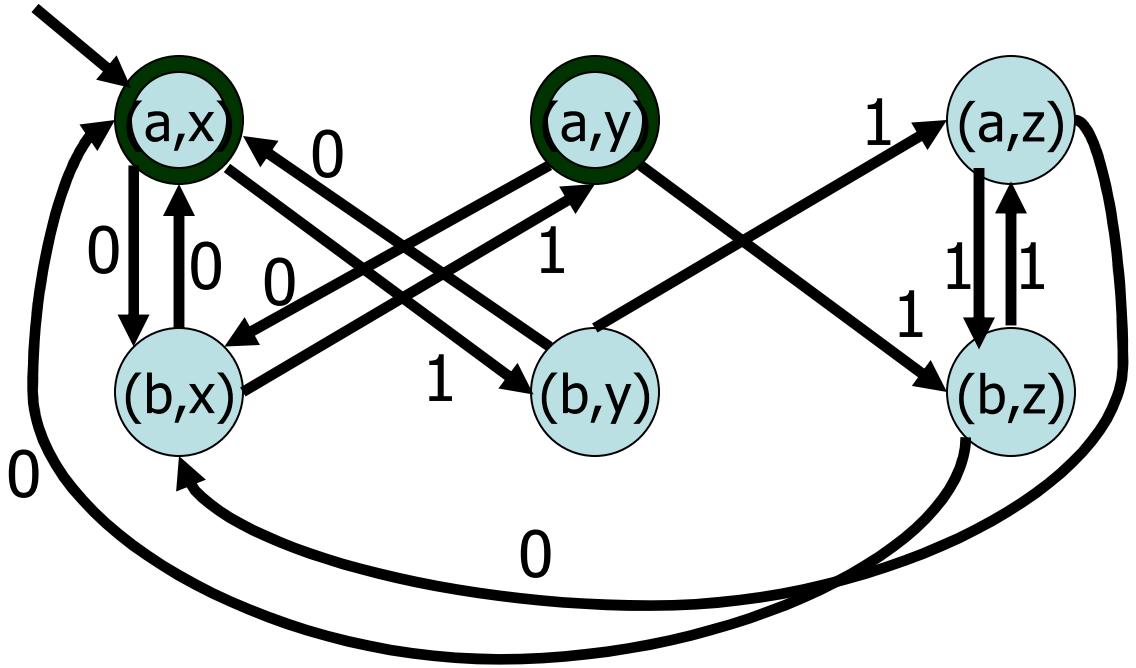
({0,1}{0,1})* - {0,1}*{11}

Diferença: Exemplo





Autômato Diferença:



Outras Construções

A ***diferença simétrica*** de dois conjuntos é

$$A \oplus B = A \cup B - A \cap B$$

Q: Como modificar a construção anteriror para aceitar a diferença simétrica de duas linguagens?

Diferença Simétrica

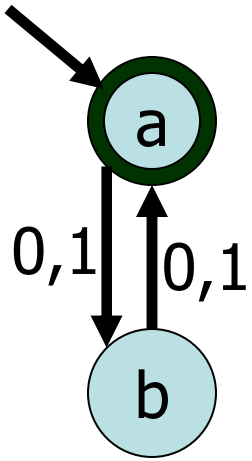
R: Aceite o string quando exatamente um dos automatas originais o aceita. I.e.

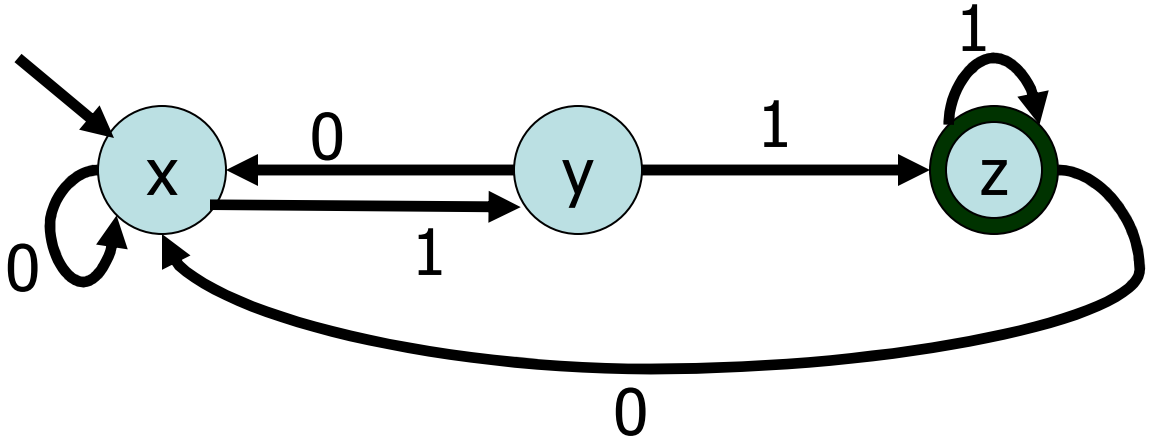
$$M_{\oplus} = (Q_1 \times Q_2, \Sigma, \delta_1 \times \delta_2, (q_{0,1}, q_{0,2}), F_{\oplus})$$

onde $F_{\oplus} = F_{\cup} - F_{\cap}$

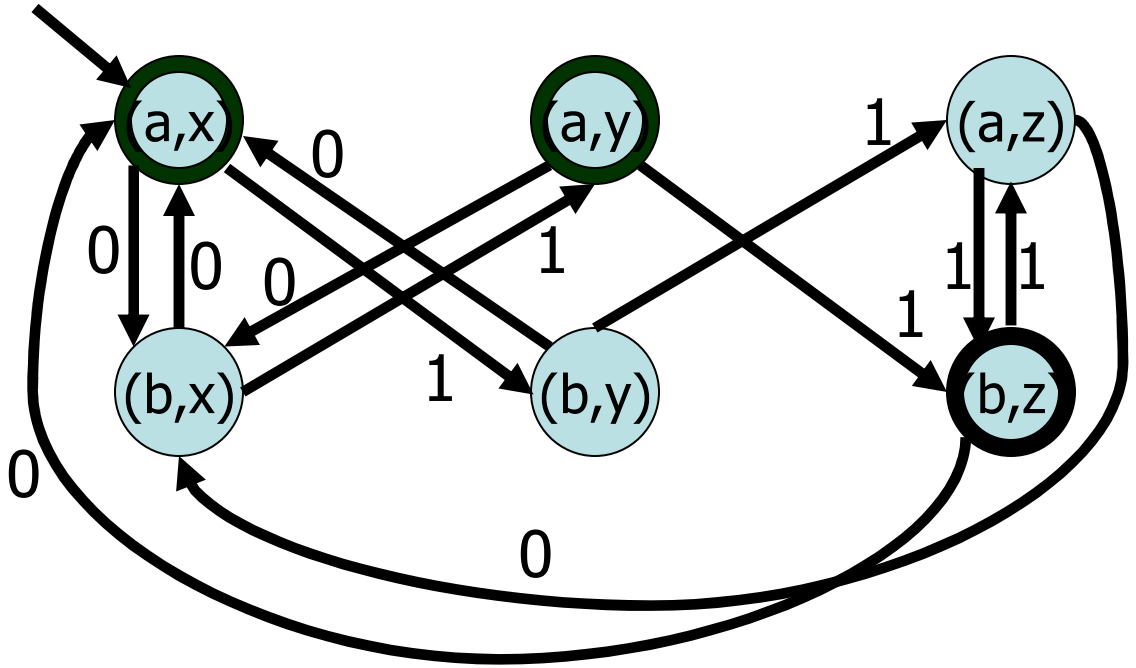
Aplicando ao nosso exemplo:

Diferença Simétrica: Exemplo





Autômato Diferença Simétrica:



Fecho de Linguagens Regulares - Resumo

Mostramos *construtivamente* que o conjunto das linguagens regulares é fechado sob as operations booleanas. I.e., dadas linguagens regulares L_1 e L_2 vimo que:

1. $L_1 \cup L_2$ is regular,
2. $L_1 \cap L_2$ is regular,
3. $L_1 - L_2$ is regular,
4. $L_1 \oplus L_2$ is regular,
5. $\overline{L_1}$ is regular.

No. 1 é também uma operação regular. Ainda precisamos mostrar que linguagens regulares são fechadas sob concatenação e Kleene-*.

AFN: Exemplo

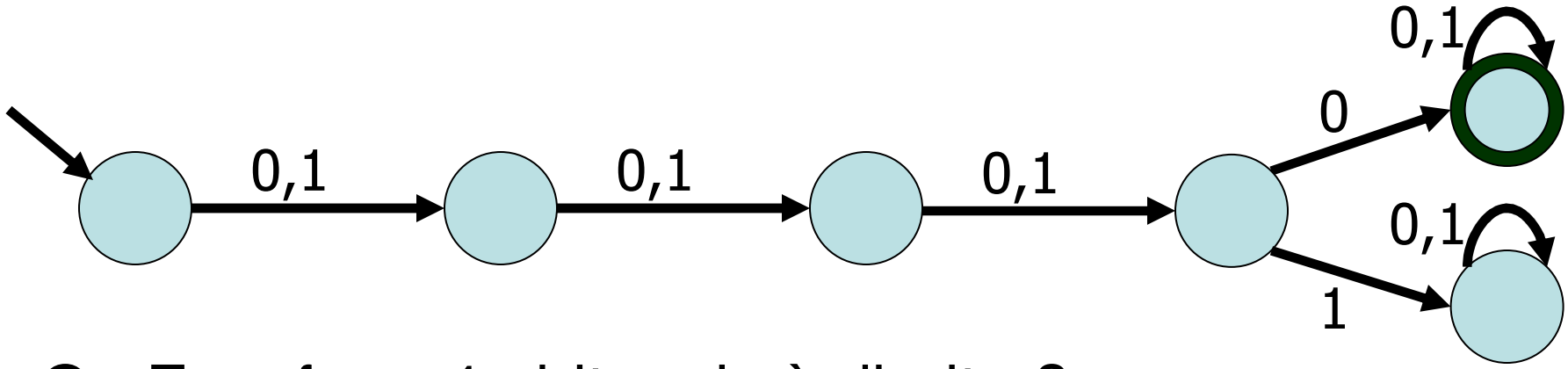
Q: Desenhe um AF que aceita a linguagem

$L_1 = \{ x \in \{0,1\}^* \mid 4^{o.}$ bit mais à esq. de x é 0 $\}$

AFN: Exemplo

R: AFD para

$$L_1 = \{\, x \in \{0,1\}^* \mid 4^{o\cdot} \text{ bit mais à esq. de } x \text{ é } 0 \,\}$$



Q: E se for o $4^{o\cdot}$ bit mais à direita ?

$$L_2 = \{\, x \in \{0,1\}^* \mid 4^{o\cdot} \text{ bit mais à dir. de } x \text{ é } 0 \,\}$$

AFN: Exemplo

R: Note que L_2 é o reverso de L_1. I.e. dizer que o 4º. bit mais à esquerda é 0 é o mesmo que dizer que o 4º. bit mais à direita do reverso é 0. Formalmente

$$L_2 = L_1^R = \{ x^R \mid x \in L_1 \}$$

A idéia é inverter a figura. Por exemplo, se uma parte da computação do autômato é

START:0→1→0→0→1:ACCEPT

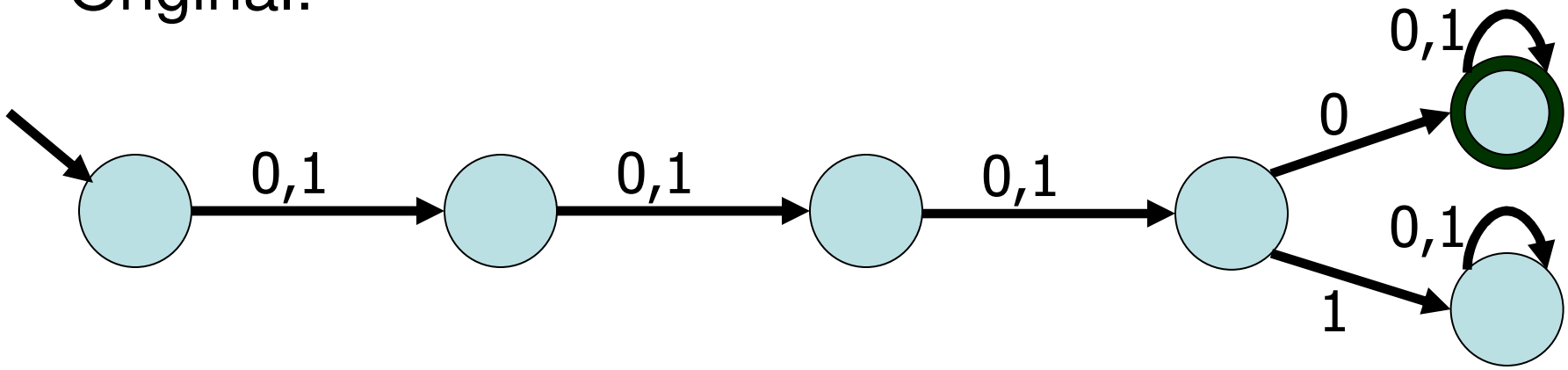
a computação a seguir deveria ser parte do autômato que aceita a linguagem reversa:

ACCEPT:0←1←0←0←1:START

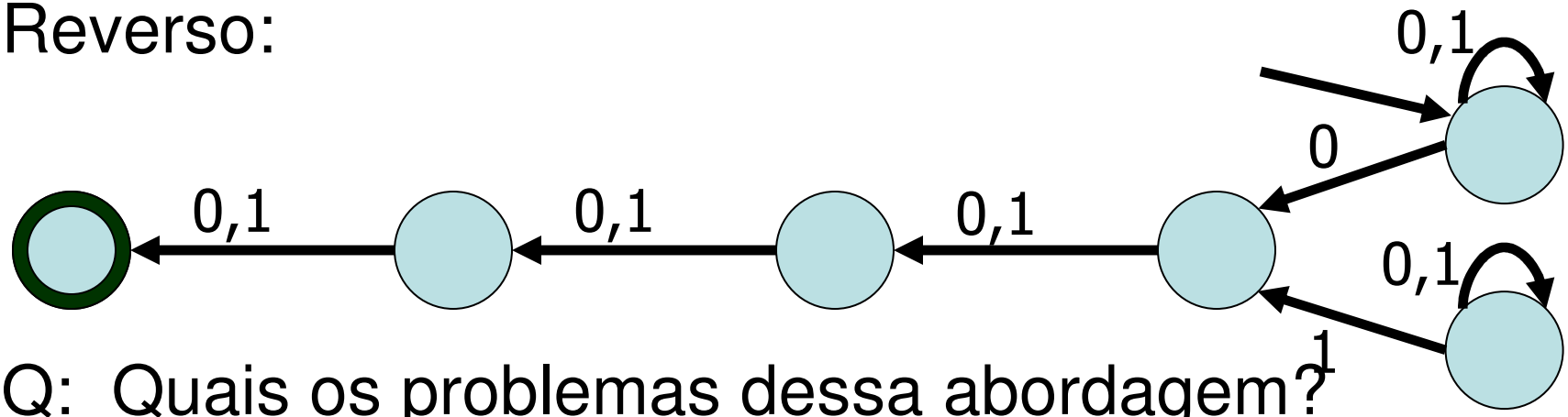
Tentemos então trocar *estados iniciais* com *estados de aceitação* e inverter todas setas.

AFN: Exemplo

Original:



Reverso:



Q: Quais os problemas dessa abordagem?

AFN: Exemplo

Resposta:

1. Estados inátingíveis. *Inútil, mas não fere as regras do nosso modelo.*
2. O estado final é agora um estado de aceitação e de *erro*. *Pode ser corrigido adicionando-se um estado de falha.*
3. No estado inicial agora não sabemos o que fazer ao ler 0. *Não determinismo. Problema.*

AFN: Exemplo

Obtemos um AF *não determinístico*. E agora?

De fato, AFNs (= AF não determinísticos) são muito úteis em diversas aplicações.

Idéia, manter informação sobre mais de um estado corrente, se necessário..

For exemplo, você pode ver como JFLAP lida com o não determinismo

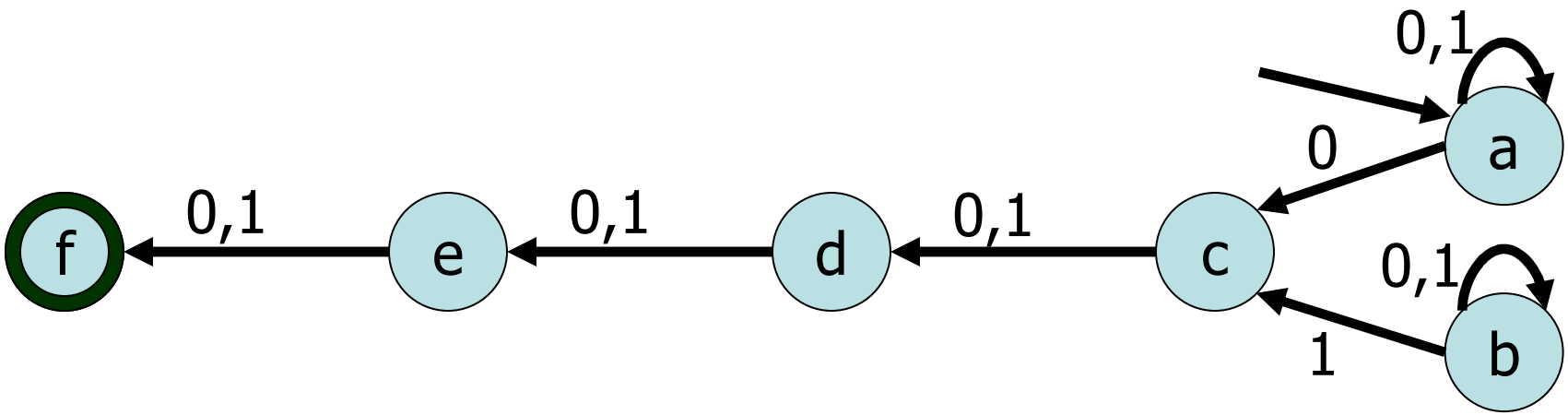
O jogo a seguir ilustra essa idéia.

O Jogo de “DETERMINIZAR”

Regras:

1. O AFN é um time de estados
2. O árbitro lê o string de entrada
3. Cada membro de um time é responsável por saber para quem ‘passar o bastão’. Se necessário, devolva o bastão para o árbitro (no caso de transição indefinida), ou espalhe o bastão para vários membros do time (no caso de múltiplas opções)
4. Cada membro do time é responsável por saber se é ou não um estado de aceitação (em F)
5. Ao final do string de enrada, qualquer estado de aceitação que tenha o bastão deve gritar: “ACEITA”
6. If nehum estado grita ACEITA, o string é rejeitado.

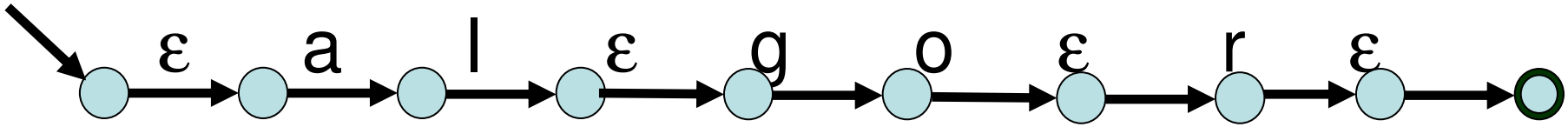
O Jogo de “DETERMINIZAR”



ε–Não determinismo

Existe outra forma de não determinismo. Queremos permitir que o autômato dê uma *pausa* e não leia nenhum próximo símbolo na entrada. Isso é representado por uma ε-transição.

Q: Qual é o string aceito pelo autômato?



Tipos de Não Determinismo

Não determinismo tipo	Análogo Máquina	Function δ	Fácil corrigir?	Formal
Nenhuma opção				
Mais de uma opção				
ε				

Tipos de Não Determinismo

Não determinismo tipo	Análogo Máquina	Function δ	Fácil corrigir?	Formal
Nenhuma opção	Falha			
Mais de uma opção	Escolha aleatória			
ε	Pausa leitura			

Tipos de Não Determinismo

Não determinismo tipo	Análogo Máquina	Function δ	Fácil corrigir?	Formal
Nenhuma opção	Falha	Nenhuma saída		
Mais de uma opção	Escolha aleatória	Múltiplos valores		
ε	Pausa leitura	*Redefine alfabeto*		

Tipos de Não Determinismo

Não determinismo tipo	Análogo Máquina	Function δ	Fácil corrigir?	Formal
Nenhuma opção	Falha	Nenhuma saída	sim, estado de falha	
Mais de uma opção	Escolha aleatória	Múltiplos valores	Não	
ε	Pausa leitura	*Redefine alfabeto*	Não	

Tipos de Não Determinismo

Não determinismo tipo	Análogo Máquina	Function δ	Fácil corrigir?	Formal
Nenhuma opção	Falha	Nenhuma saída	sim, estado de falha	$\|\delta(q,a)\|=0$
Mais de uma opção	Escolha aleatória	Múltiplos valores	Não	$\|\delta(q,a)\|>1$
ε	Pausa leitura	*Redefine alfabeto*	Não	$\|\delta(q,\varepsilon)\|>1$