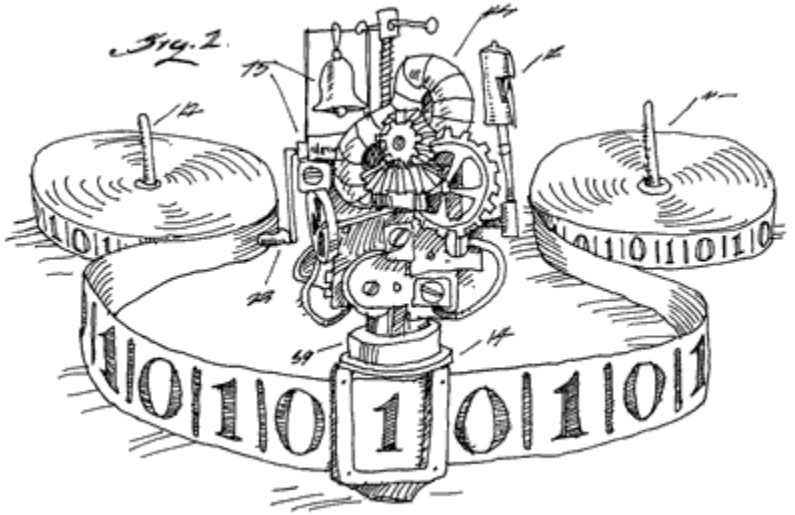
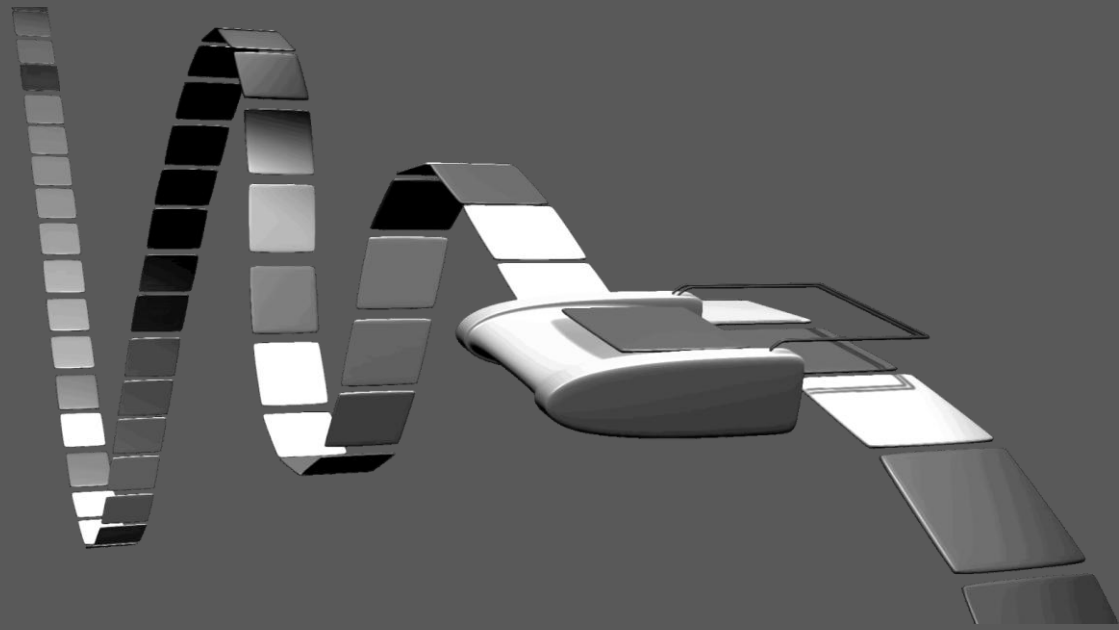
Equivalência entre AFDs e AFNs

Introdução à Teoria da Computação

Nicollas Mocelin Sdroievski

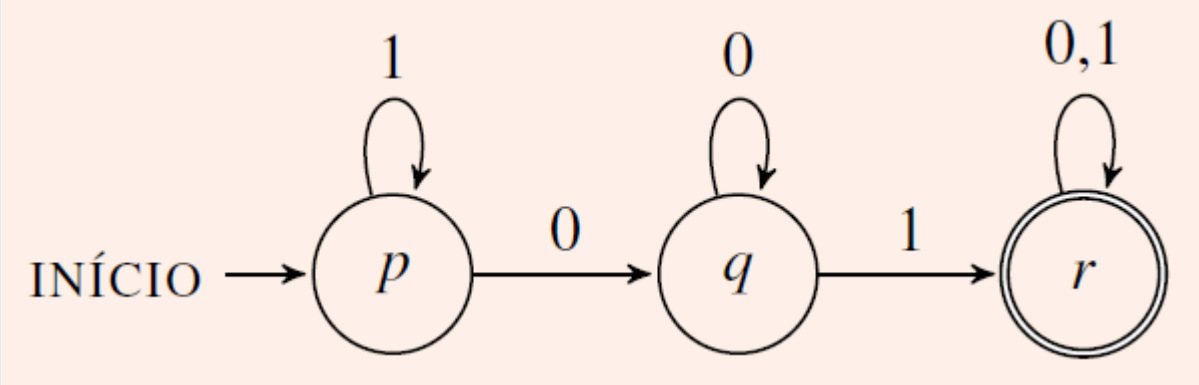


Relembrando

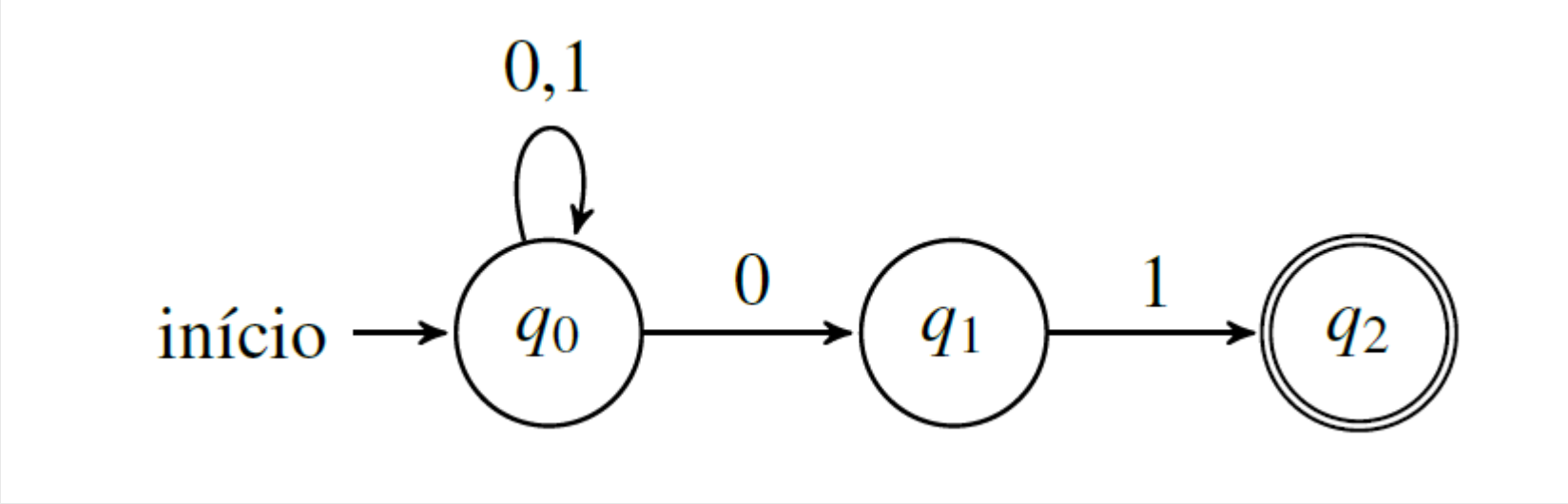


Autômatos Finitos Não Determinísticos

- Definidos por uma quíntupla $D = (Q, \Sigma, \delta, q_0, F)$
  - Q = conjunto de estado
  - Σ = alfabeto
  - δ = função de transição
  - q_0 = estado inicial
  - F = conjunto de estados finais



Definição para o AFN de exemplo

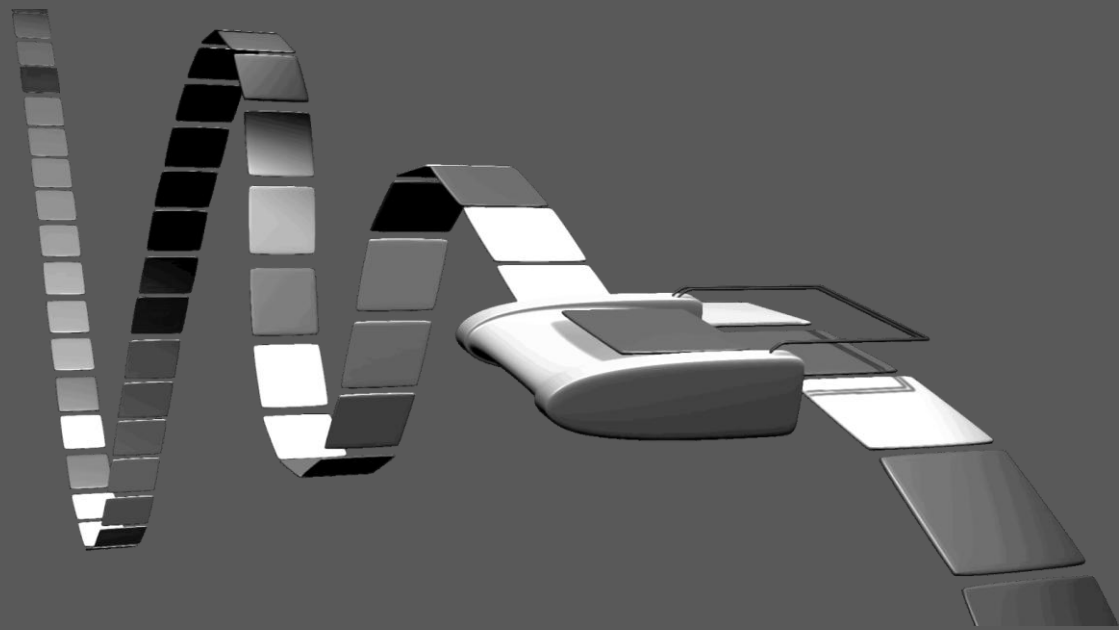


■ **Exemplo 3.2** A definição formal para o diagrama da Figura 3.2 é o AFN $N = (Q, \Sigma, \delta, q_0, F)$, tal que $Q = \{q_0, q_1, q_2\}$, $\Sigma = \{0, 1\}$ e a função δ é definida abaixo:

$$\delta(q_0, 0) = \{q_0, q_1\}$$
$$\delta(q_0, 1) = \{q_0\}$$
$$\delta(q_1, 1) = \{q_2\}$$
$$\delta(q_1, 0) = \delta(q_2, 0) = \delta(q_2, 1) = \emptyset$$

■

Equivalência



Expressividade de modelos de computação

- Quais linguagens cada modelo computacional é capaz de reconhecer?
- Tema recorrente na disciplina
- Mais pra frente, veremos modelos que são, de fato, mais expressivos do que AFDs/AFNs

Algoritmo de construção de conjuntos

- **Entrada:** um AFN $N = (Q_N, \Sigma, \delta_N, q_0, F)$
- **Saída:** um AFD $D = (Q_D, \Sigma, \delta_D, \{q_0\},\ F_D)$ tal que $L(D) = L(N)$
- **Ideia principal:** simular N com estados que são subconjuntos de $\mathcal{P}(Q_N)$, com transições para outros subconjuntos de estados
  - Força bruta, aplicando a função δ_N em todas as combinações possíveis de pares que consistem de um subconjunto de estados de Q_N e um símbolo de Σ

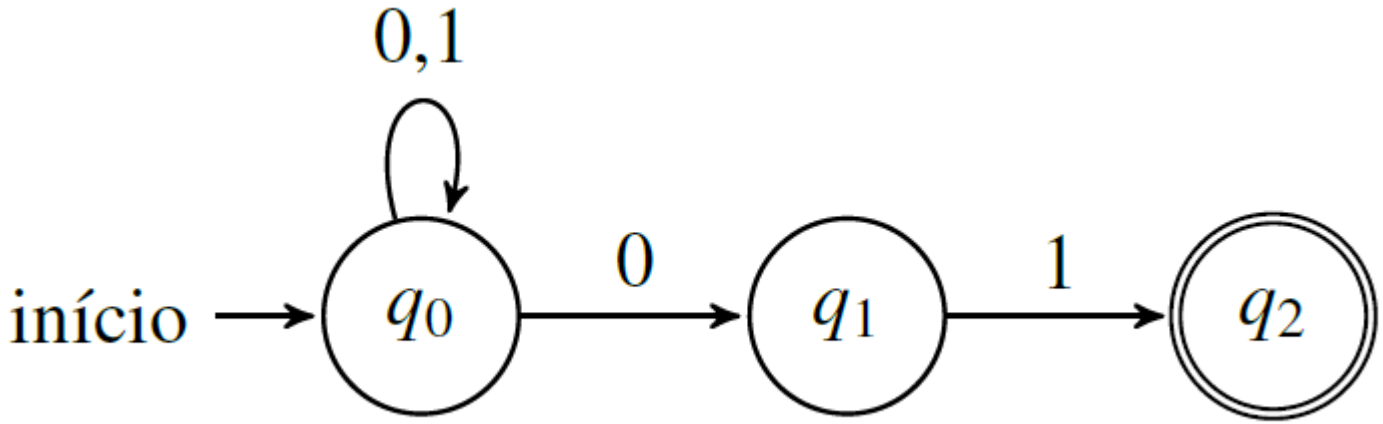
Algoritmo: definição formal

Algorithm 1 Construindo um AFD a partir de um AFN.

AFN_AFD $(Q_N, \Sigma, \delta_N, q_0, F_N)$

1: $Q_D = \mathscr{P}(Q_N)$
2: **for all** $S \subseteq Q_D$ **do**
3: **for all** $a \in \Sigma$ **do**
4: Defina a função δ_D no par (S, a) da seguinte maneira: $\delta_D(S, a) = \bigcup_{p \in S} \delta_N(p, a)$
5: $F_D = \{S \in \mathscr{P}(Q_N) : S \cap F_N \neq \varnothing\}$
6: $D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$
7: **Return** D

Exemplo

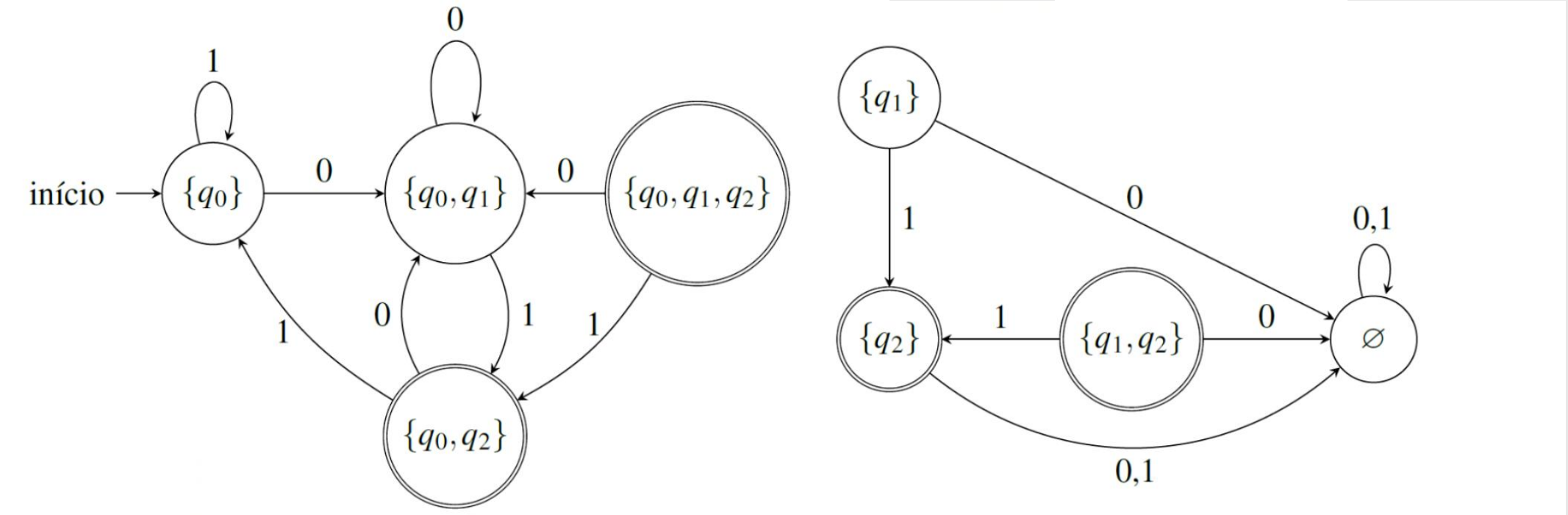


	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\varnothing$	$\{q_2\}$
$^*q_2$	$\varnothing$	$\varnothing$

Resultado: tabela de transições

	0	1
$\varnothing$	$\varnothing$	$\varnothing$
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_1\}$	$\varnothing$	$\{q_2\}$
$*\{q_2\}$	$\varnothing$	$\varnothing$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$*\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
$*\{q_1, q_2\}$	$\varnothing$	$\{q_2\}$
$*\{q_0, q_1, q_2\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$

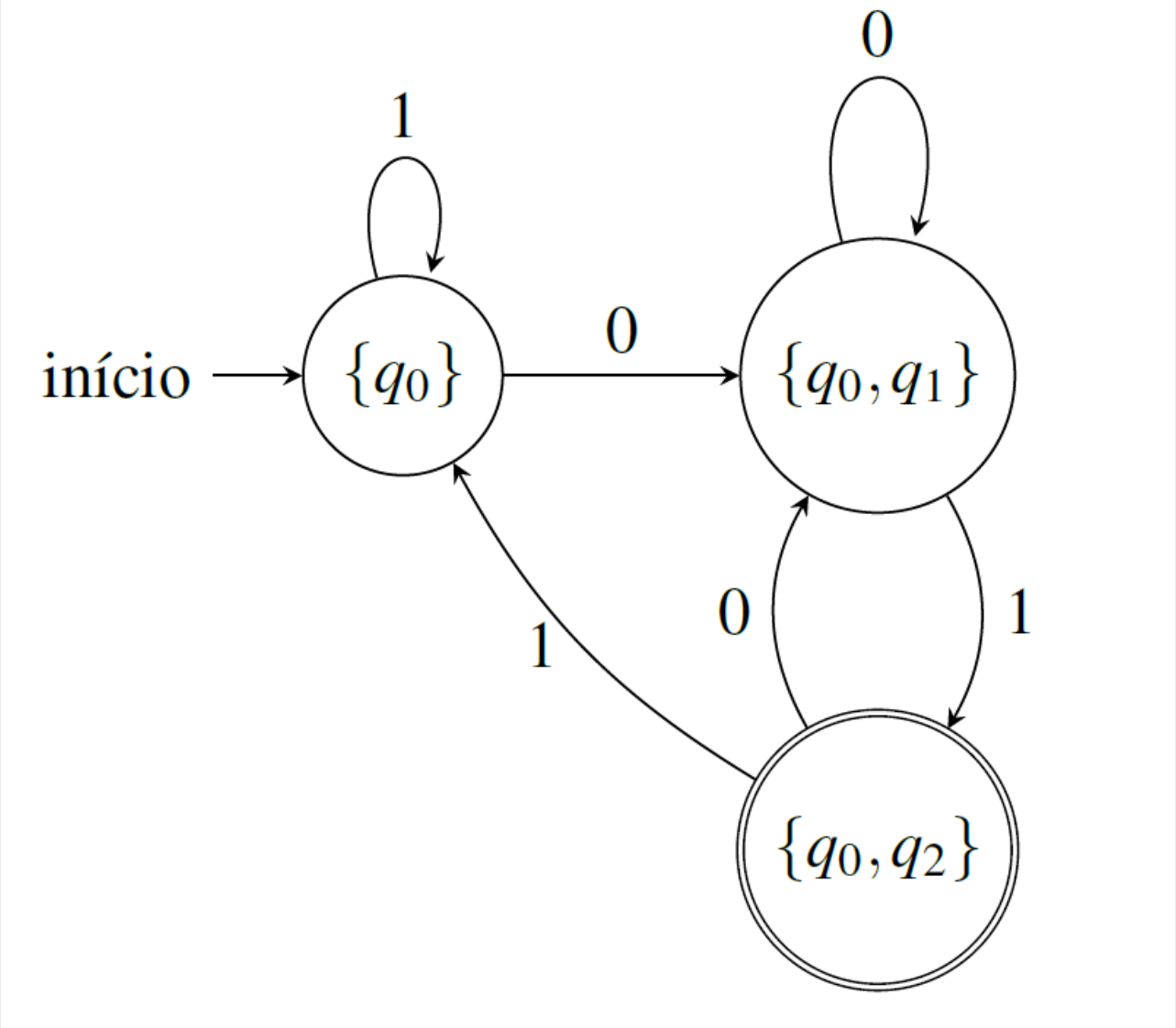
Resultado: diagrama



Algoritmo: versão melhorada

- O algoritmo anterior produz AFDs que potencialmente possuem estados inalcançáveis
- **Solução:** executar o algoritmo e eliminar os estados inalcançáveis
- **Solução mais simples:** executar o algoritmo passo a passo, iniciando em $\{q_0\}$ e apenas adicionando os conjuntos de estados que surgirem
  - Faremos isso daqui pra frente

AFD sem estados inalcançáveis



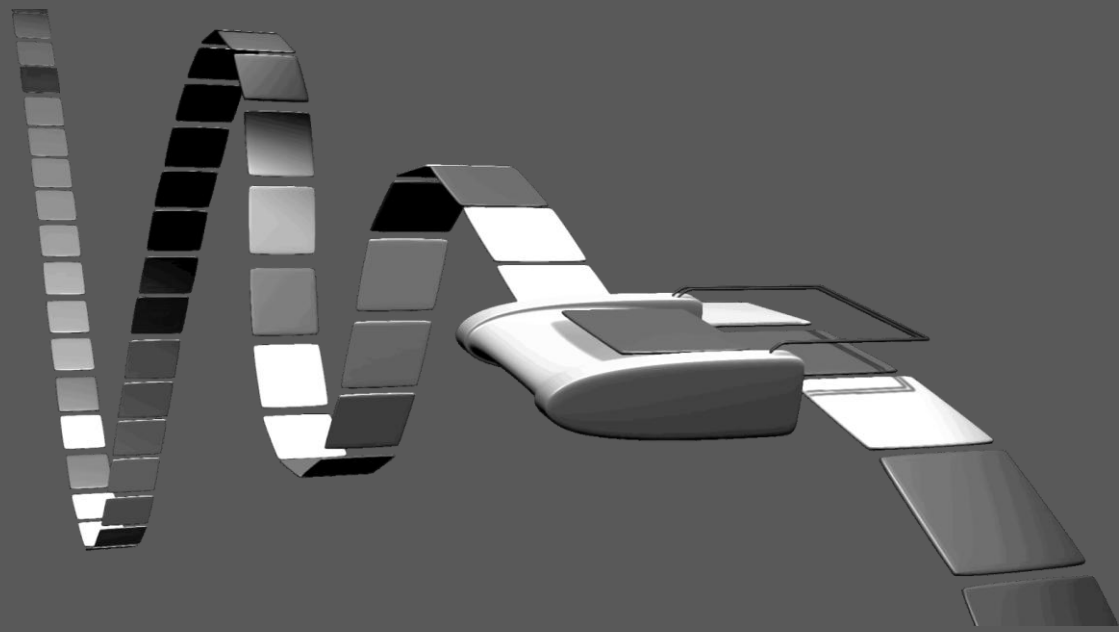
Teoremas

- **Teorema.** Se o AFD $D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$ é obtido pelo Algoritmo 1 a partir de um AFN $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$, então $L(N) = L(D)$

Exercício 3.17 (Opcional) Prove o Teorema 3.3.1. Dica: prove por indução que $\forall w \in \Sigma^*$, $\hat{\delta}_D(q_0, w) \in F_N \Leftrightarrow \hat{\delta}_N(\{q_0\}, w) \cap F_N$. ■

- Minha sugestão: provar por indução que $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$ (veja que os estados de D são conjuntos de estados de N, assim como a imagem de $\hat{\delta}_N$)
- **Teorema.** Uma linguagem L é aceita por um AFD se e somente se L é aceita por um AFN.

Exercício



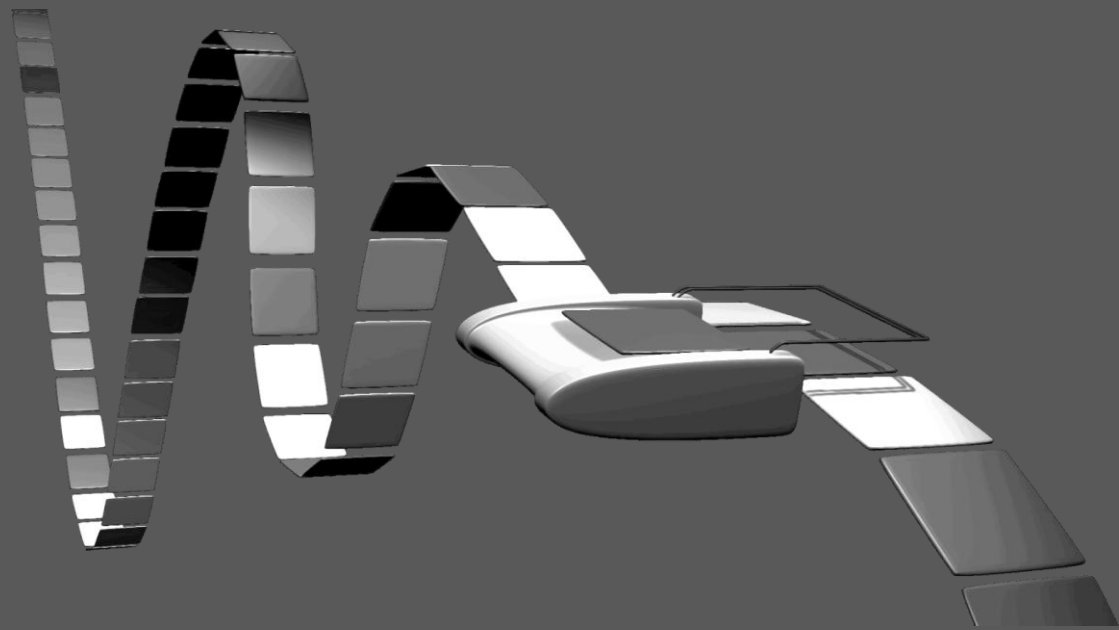
Exercício

Exercício 3.18 Considere o o AFN dado pela tabela abaixo:

	0	1
$\rightarrow p$	$\{p, q\}$	$\{p\}$
q	$\{r\}$	$\{r\}$
r	$\{s\}$	$\varnothing$
$*s$	$\{s\}$	$\{s\}$

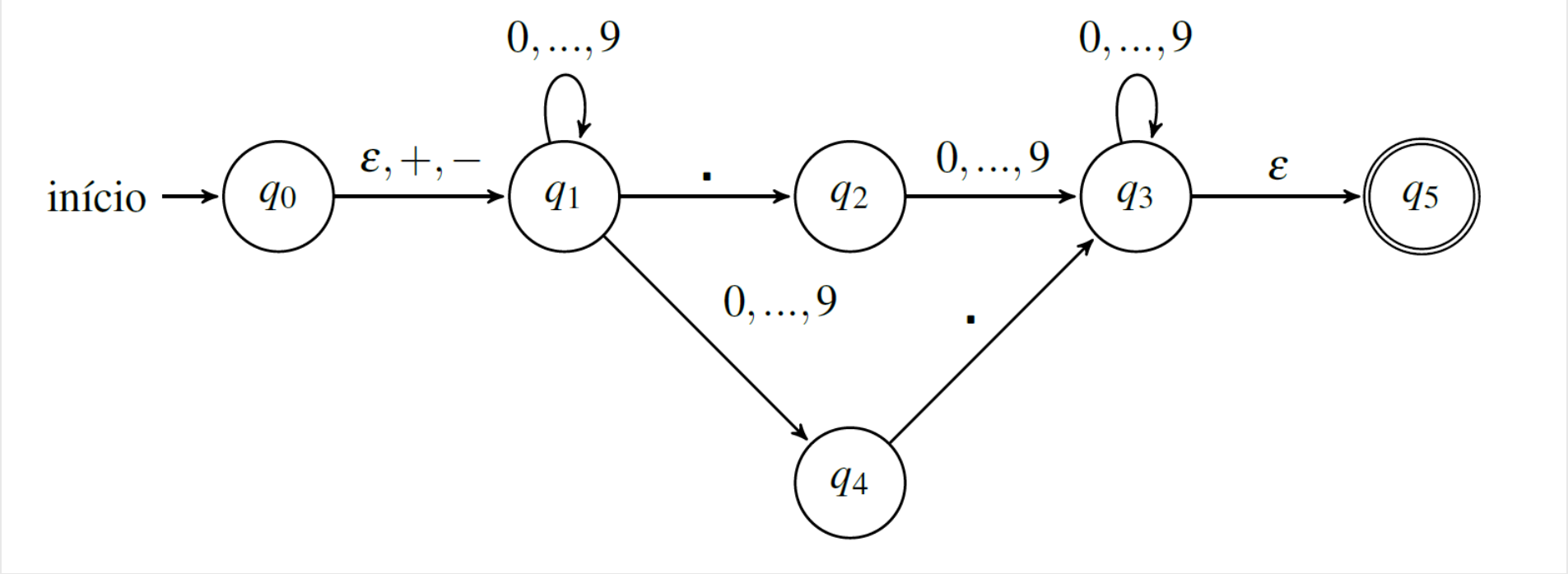
Apresente um AFD que aceite a mesma linguagem do AFN acima. ■

ε-AFNs



AFNs com transições ε

- AFNs que podem conter transições com o rótulo ε
- Permite mudanças de estado (não determinísticas) sem consumir nenhum símbolo de entrada



Definição

- **Definição.** Um ε-AFN é uma 5-tupla $E = (Q, \Sigma, \delta, q_0, F)$ ta; que cada um dos componentes tem a mesma interpretação que um AFN, exceto pelo fato que δ é uma função do tipo $\delta: Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow \mathcal{P}(Q)$, sendo que $\varepsilon \notin \Sigma$.
- ε-NFA do exemplo:

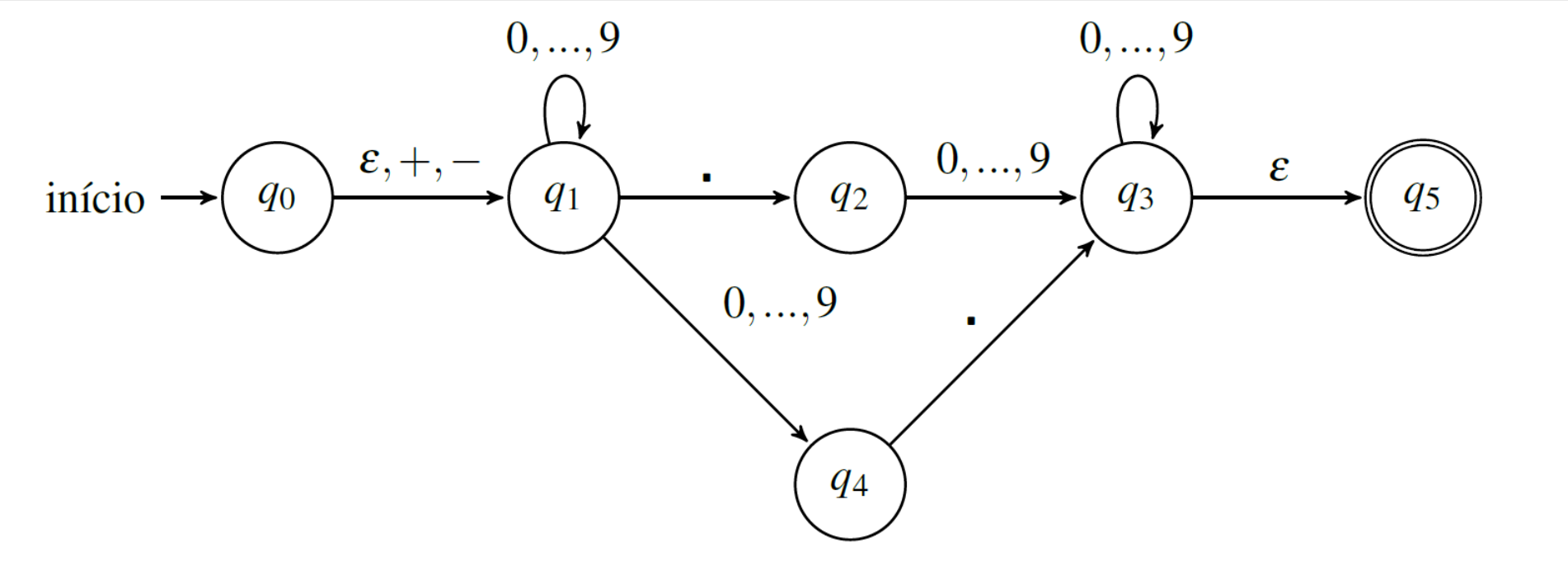
	ε	*sinal*	.	dígito
q_0	$\{q_1\}$	$\{q_1\}$	$\varnothing$	$\varnothing$
q_1	$\varnothing$	$\varnothing$	$\{q_2\}$	$\{q_1, q_4\}$
q_2	$\varnothing$	$\varnothing$	$\varnothing$	$\{q_3\}$
q_3	$\{q_5\}$	$\varnothing$	$\varnothing$	$\{q_3\}$
q_4	$\varnothing$	$\varnothing$	$\{q_3\}$	$\varnothing$
$^*q_5$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$

ε-fecho

- **Ideia.** O ε-Fecho de um estado q é o conjunto de todos os possíveis estados que podem ser atingidos a partir de q (incluindo o próprio q) utilizando-se uma quantidade arbitrária de transições ε

- **Definição.** Seja um ε-AFN com conjunto de estados Q e seja $q \in Q$. O conjunto ε-Fecho(q) é definido de maneira indutiva:
  - **Base:** $q \in \varepsilon$-Fecho(q)
  - **Indução:** se $p \in \varepsilon$-Fecho(q) e $r \in \delta(p, \varepsilon)$, então $r \in \varepsilon$-Fecho(q)

Exemplo



Função de transição estendida

- **Definição.** Dado um ε-AFN $N = (Q, \Sigma, \delta, q_0, F)$, definimos $\hat{\delta} : Q \times \Sigma^* \rightarrow \mathcal{P}(Q)$ indutivamente:
  - Base: $\hat{\delta}(q, \varepsilon) =$ ε-Fecho(q)
  - Indução: Seja $w \in \Sigma^*$ da forma xa, onde $x \in \Sigma^*$ e $a \in \Sigma$. Suponha que $\hat{\delta}(q, x) = \{p_1, p_2, \ldots, p_k\}$ e que

$$\bigcup_{i=1}^{k} \delta(p_i, a) = \{r_1, r_2, \ldots, r_m\}$$

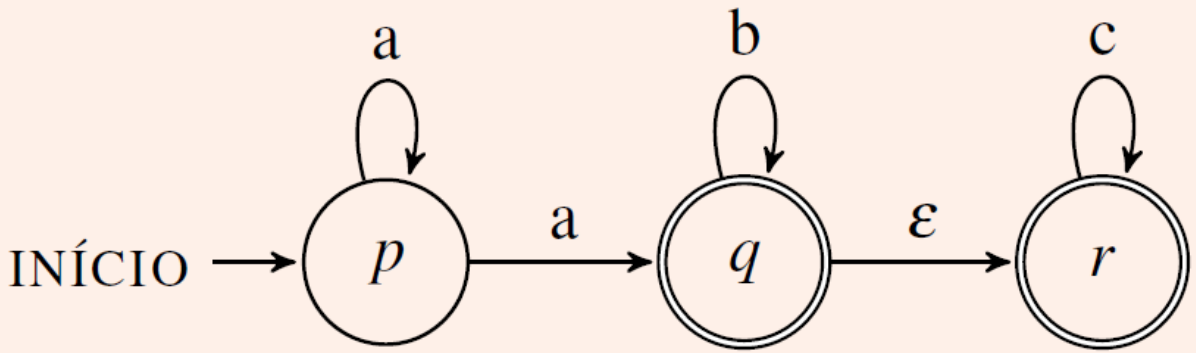
Então

$$\hat{\delta}(q, w) = \bigcup_{j=1}^{m} \varepsilon\text{-Fecho}(r_j)$$

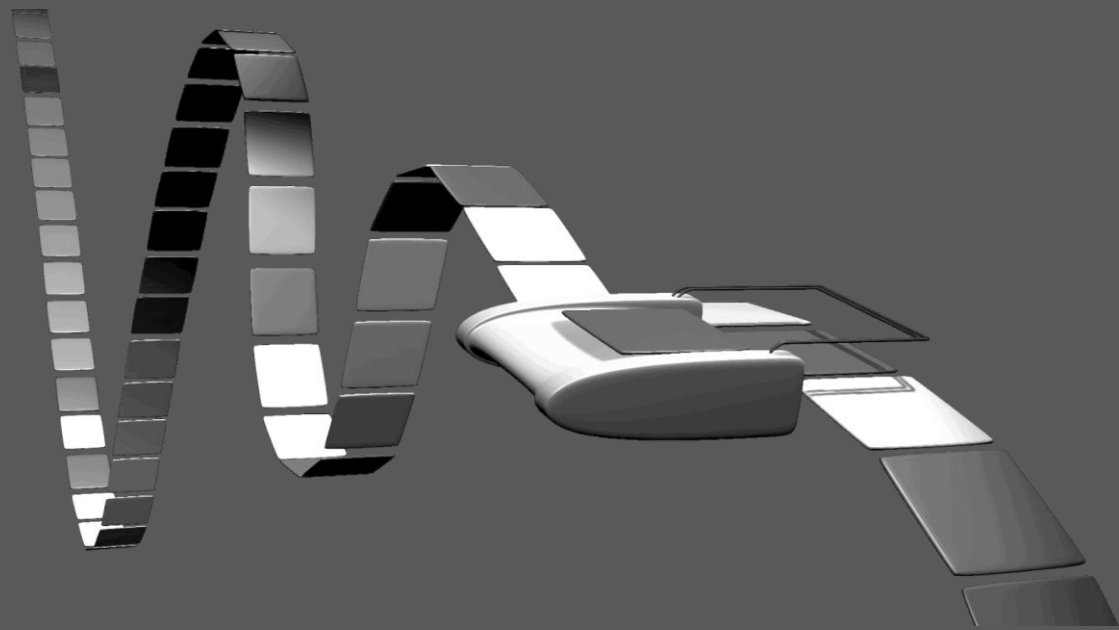
Exercício

Exercício 3.21 (Resolvido) Seja $\Sigma = \{a, b, c\}$. Apresente um ε-AFN para a linguagem das strings sobre Σ que têm a forma $a^n b^m c^l$, tal que $n > 0$, $m \geq 0$, $l \geq 0$.

Solução:



Para fechar



Para fechar

- Hoje
  - Equivalência entre AFNs e AFDs
    - Algoritmo de conversão AFN → AFD
  - ε-AFNs
    - AFNs com transições ε
- Próxima aula
  - Equivalência entre ε-AFNs e AFDs
    - Algoritmo similar ao visto hoje
  - Introdução à expressões regulares

Referências

- [SIL25] – SILVA, M.V.G. **Autômatos, Computabilidade e Complexidade Computacional** , 2025.