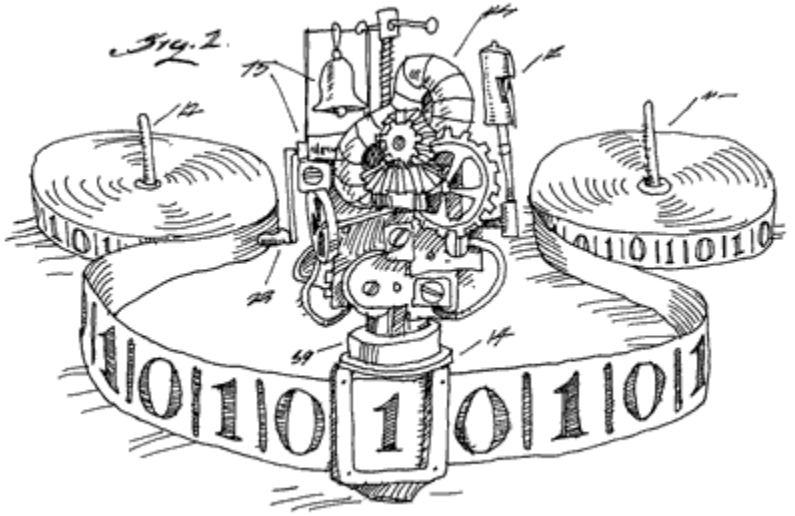
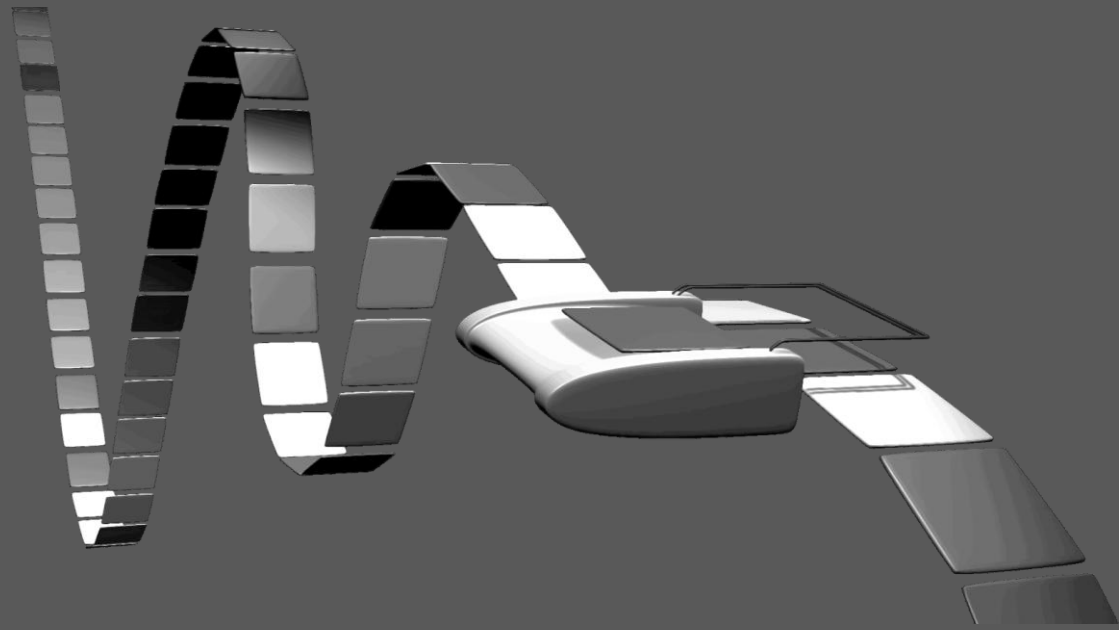
Autômatos com pilha

Introdução à Teoria da Computação

Nicollas Mocelin Sdroievski



Relembrando



Linguagens regulare

- Linguagens L para as quais as seguintes afirmações são verdade
  - Existe um AFD D tal que $L = L(D)$
  - Existe um AFN N tal que $L = L(N)$
  - Existe um ε-AFN E tal que $L = L(E)$
  - Existe uma expressão regular R tal que $L = L(R)$
- Todos estes modelos são equivalentes na sua expressividade

Lema do bombeamento

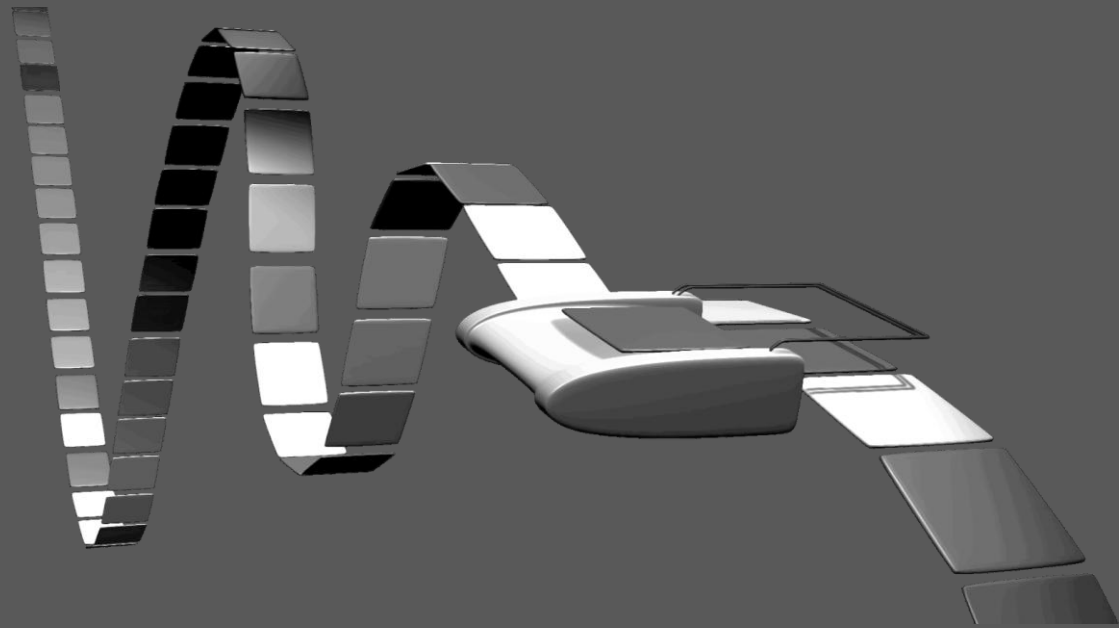
- **Lema (do Bombeamento para linguagens regulares)**. Seja L uma linguagem regular. Então existe uma constante inteira $t \geq 1$ tal que para toda string $w \in L$ satisfazendo $|w| \geq t$, existem strings $x, y, z \in \Sigma^*$ satisfazendo o seguinte:
  - $w = xyz$ e as condições abaixo são satisfeitas:
  1. $y \neq \varepsilon$
  2. $|xy| \leq t$
  3. Para todo $k \geq 0$, $xy^k z \in L$

Exercícios

Exercício 4.1 Prove que as seguintes linguagens não são regulares:

- A linguagem L_{WR} das strings binárias da forma ww^R
- A linguagem L_{WW} das strings binárias da forma ww
- A linguagem L_{EQ} das strings binárias que tem a mesma quantidade de 0's e 1's.
- A linguagem $L_{\mathrm{DB}} = \{w : \text{o número de 0's em } w \text{ é o dobro do número de 1's}\}$.
- A linguagem $L_{+0} = \{0^i 1^j \ : \ i > j\}$
- A linguagem $L_{+1} = \{0^i 1^j \ : \ i < j\}$
- A linguagem $L_{\mathrm{SQ}} = \{1^s : s \text{ é um quadrado perfeito}\}$.

Autômatos com pilha



Intuição

- ε-AFNs que possuem uma memória adicional no formato de uma pilha de dados
- ε-AFN
  - Processa um ou zero símbolos da string de entrada
  - Faz uma transição de estado
- Autômato com pilha
  - Consome um ou zero símbolos da string de entrada
  - Desempilha o símbolo do topo da pilha
  - Faz uma transição de estado
  - Empilha uma quantidade finita de símbolos na pilha

Mais comparações

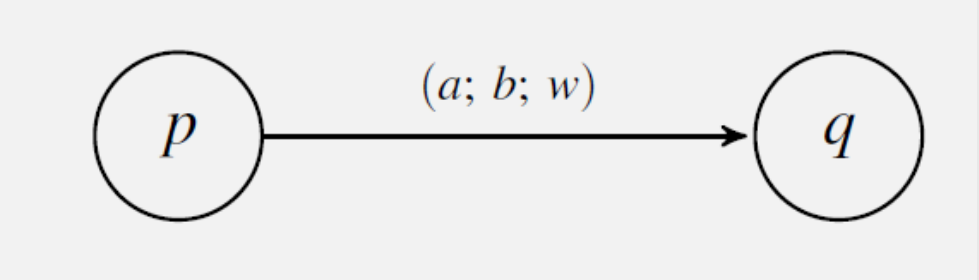
- Em um ε-AFN, a função de transição tem como entrada
  - $Q \times \Sigma \cup \{\varepsilon\}$
- Em um autômato com pilha, a função de transição tem como entrada
  - $Q \times \Sigma \cup \{\varepsilon\} \times \Gamma$
  - Onde Γ é o alfabeto da pilha
- A saída é
  - $\{(q_1, \gamma_1), (q_2, \gamma_2), \ldots, (q_k, \gamma_k)\}$
  - Onde cada q_i é um estado
  - E cada $\gamma_i \in \Gamma^*$ (uma string sobre o alfabeto da pilha)

Definição

- **Definição.** Um autômato com pilha (AP) P é uma 7-tupla $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ tal que:
  - Q, Σ, q_0, F têm a mesma interpretação que em um ε-AFN
  - Γ é o alfabeto da pilha
  - $Z_0 \in \Gamma$ é o símbolo inicial da pilha, de forma que $Z_0 \notin \Sigma$
  - $\delta: Q \times \Sigma \cup \{\varepsilon\} \times \Gamma \to \{(q_1, \gamma_1), (q_2, \gamma_2), \dots, (q_k, \gamma_k)\}$, tal que $q_i \in Q$ e $\gamma_i \in \Gamma^*$
- Por convenção, Z_0 é o único símbolo presente na pilha no início da computação, e marca o “fundo” da pilha

Lendo diagramas

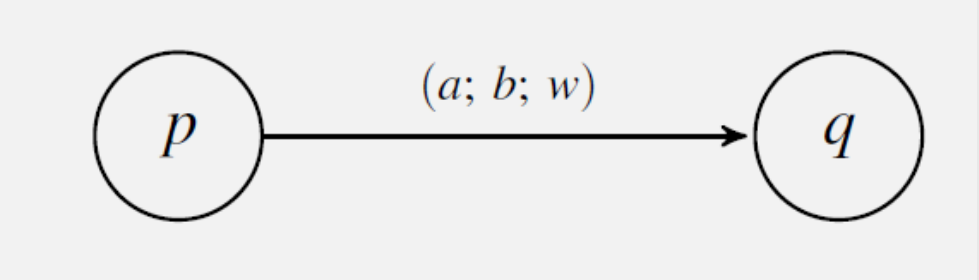
- Os rótulos das transições de um AP são como abaixo



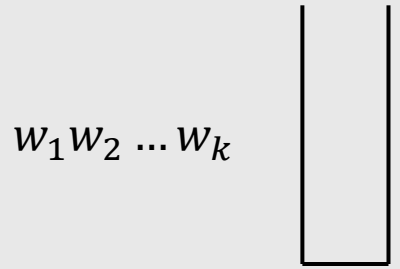
- a = símbolo da string de entrada
- b = topo da pilha (que foi desempilhado)
- w = string para empilhar (de trás pra frente)

Lendo diagramas

- Os rótulos das transições de um AP são como abaixo

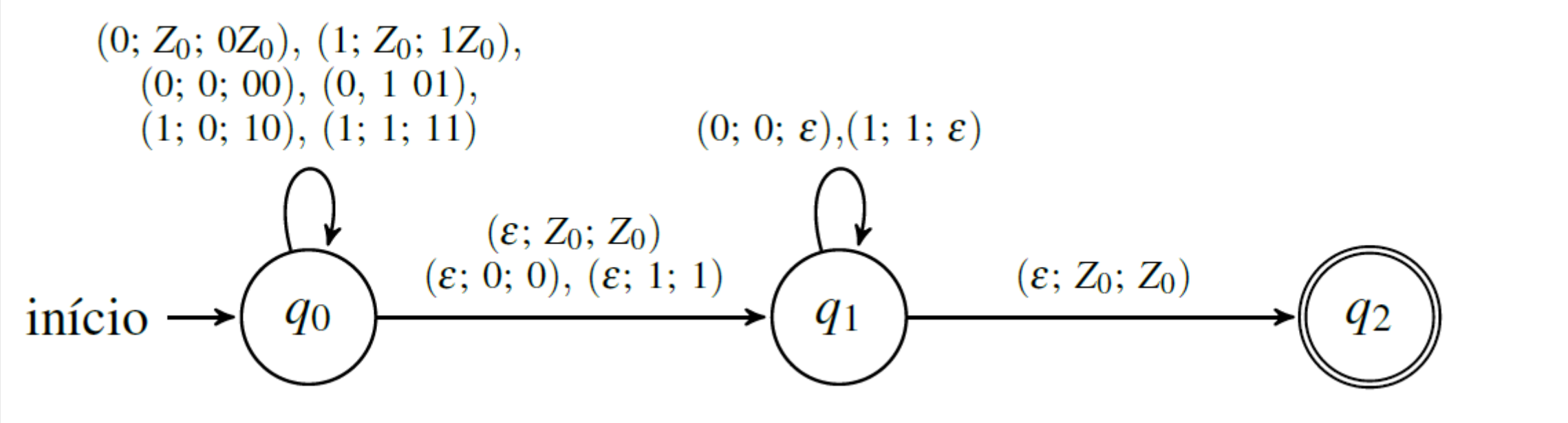


- a = símbolo da string de entrada
- b = topo da pilha (que foi desempilhado)
- w = string para empilhar (de trás pra frente)



Exemplo

- AP para a linguagem L_{RR} das strings binárias da forma ww^R

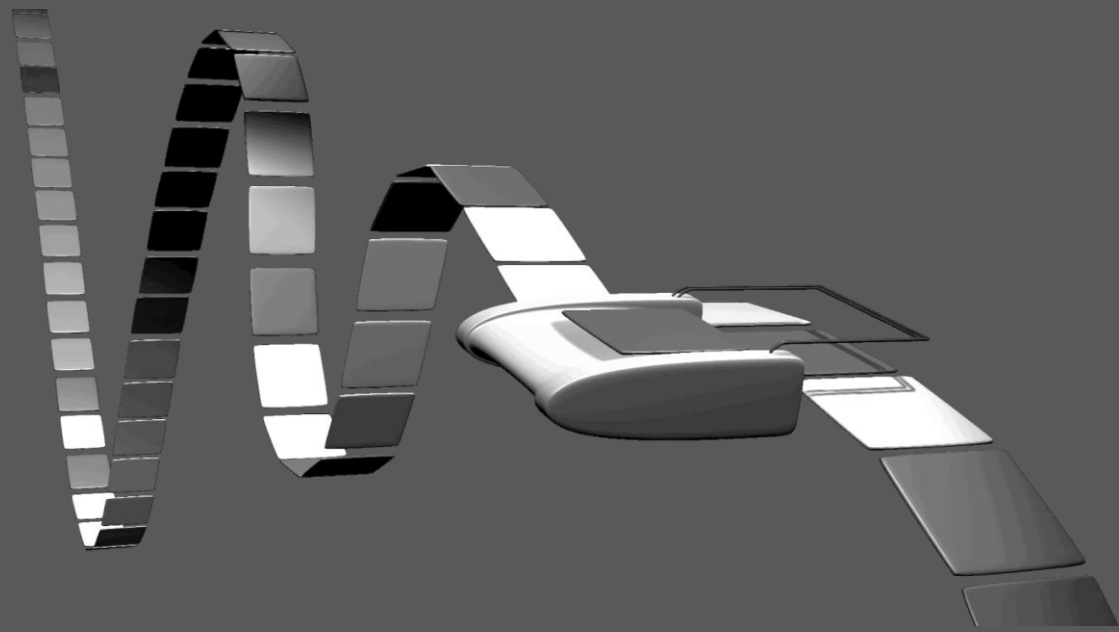


Projetando APs

- É comum sempre manter Z_0 no fundo da pilha
  - Ajuda a saber quando a pilha está vazia
- **Observação**. Um AP *morre* se tenta realizar uma transição com a pilha vazia (sem nem mesmo Z_0)

Exercício 4.8 Apresente a definição formal e o diagrmaa de um AP que aceite a linguagem $L = \{0^n 1^n \ : \ n \geq 1\}$. ■

Definição formal da computação



Configuração

- Considere um AFD (ou um caminho de computação de um ε-AFN) processando uma string $w = w_1 w_2 \ldots w_k$
- Depois de processar $w_1 w_2 \ldots w_i$, que informação é necessária para continuar a computação com $w_{i+1} w_{i+2} \ldots w_k$?
  - O estado em que o AFD se encontra
  - A própria string $w_{i+1} w_{i+2} \ldots w_k$
- E um AP?
  - O estado em que o AP se encontra
  - A própria string $w_{i+1} w_{i+2} \ldots w_k$
  - Os símbolos que estão na pilha de dados (uma string)

Passo computacional

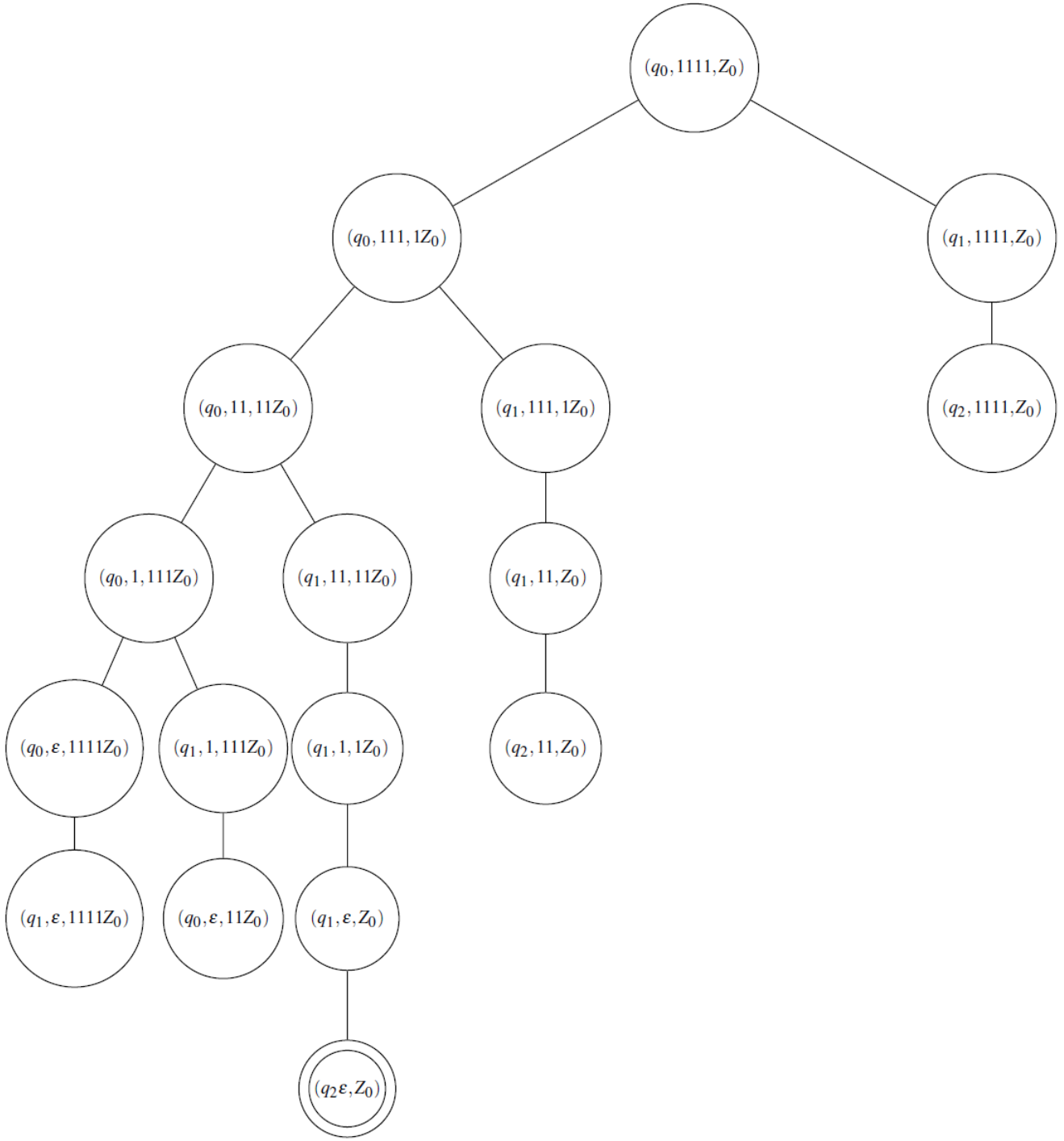
- **Definição (Configuração de um AP).** Seja $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ um AP e $q \in Q, w \in \Sigma^*, \gamma \in \Gamma^*$. Uma tripla (q, w, γ) é chamada de uma *configuração* de P.
- **Definição (Um passo computacional).** Seja Seja $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ um AP. Escrevemos $(q, aw, X\beta) \vdash_P (p, w, \alpha\beta)$ se $(p, \alpha) \in \delta(q, a, X)$, onde $w \in \Sigma^*$ e $\beta \in \Gamma^*$.
- **Exemplo.** O AP P_{RR} do slide X com a entrada 010010 no seu primeiro passo sai da configuração $(q_0, 010010, Z_0)$ e atinge a configuração$(q_0, 10010, 0Z_0)$

Vários passos computacionais

- **Definição (Sequência de passos computacionais).** O símbolo $\vdash_P^*$, usado para representar um sequência de passos que leva a computação de uma configuração C_i para outra configuração C_j é definido recursivamente
  - **Base.** Se C_i é uma configuração de um AP, então $C_i \vdash_P^* C_i$
  - **Indução.** $C_i \vdash_P^* C_j$ se existe C_k tal que $C_i \vdash_P C_k$ e $C_k \vdash_P^* C_j$
- **Definição (Aceitação e rejeição).** Seja $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ um AP. Dado $w \in \Sigma^*$, dizemos que P aceita w se $(q_0, w, Z_0) \vdash_P^* (q, \varepsilon, \alpha)$ para $q \in F$ e $\alpha \in \Gamma^*$. Caso contrário, dizemos que P rejeita w.

Árvore de computações

- **Definição.** Dada um AP $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ e uma string w. A árvore de computações possíveis de P com w é definida indutivamente:
  - **Base:** Inclua na árvore o nó raiz correspondente à configuração $C_0 = (q_o, w, Z_0)$
  - **Indução:** Seja C_i um nó da árvore. Se $C_i \vdash_P C_j$, então inclua o nó C_j como filho de C_i na árvore
- No próximo slide, exemplo de árvore de computação do autômato P_{RR} com a string 1111

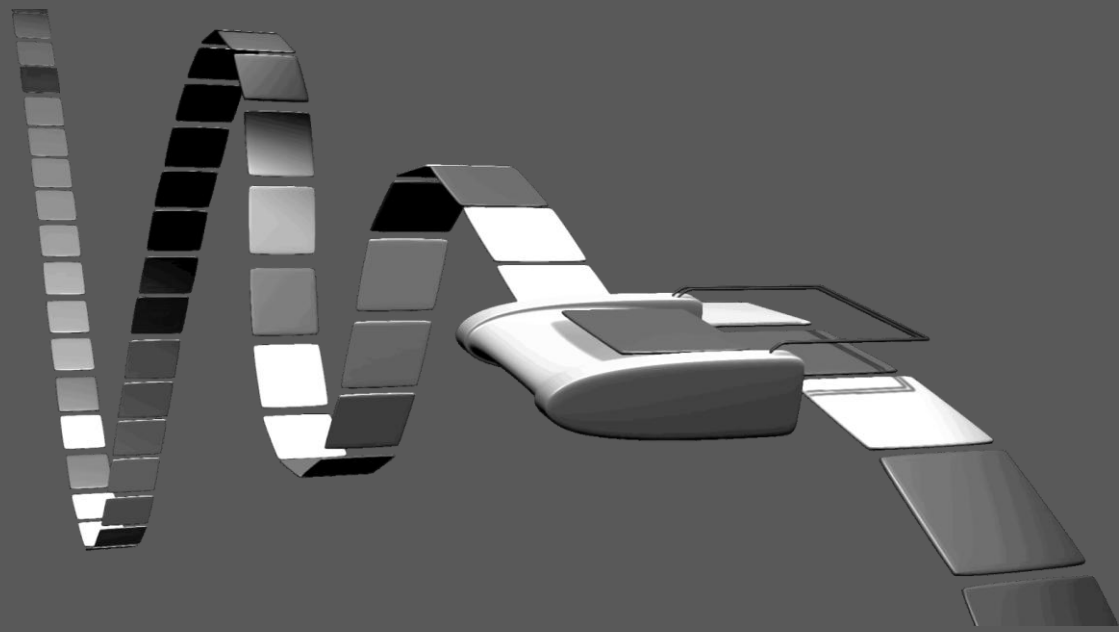

$(q_0, 1111, Z_0)$
$(q_0, 111, 1Z_0)$
$(q_1, 1111, Z_0)$
$(q_0, 11, 11Z_0)$
$(q_1, 111, 1Z_0)$
$(q_2, 1111, Z_0)$
$(q_0, 1, 111Z_0)$
$(q_1, 11, 11Z_0)$
$(q_1, 11, Z_0)$
$(q_0, \varepsilon, 1111Z_0)$
$(q_1, 1, 111Z_0)$
$(q_1, 1, 1Z_0)$
$(q_2, 11, Z_0)$
$(q_1, \varepsilon, 1111Z_0)$
$(q_0, \varepsilon, 11Z_0)$
(q_1, ε, Z_0)
$(q_2 \varepsilon, Z_0)$

Linguagens aceitas por APs

- **Definição.** Seja $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ um AP. Definimos a linguagem de P como $L(P) = \{w \in \Sigma^* \mid w \text{ é aceita por } P\}$.

- **Definição.** Dada uma linguagem L, se existe um AP P tal que $L = L(P)$, então L é dita uma **linguagem livre de contexto**.

Variações de APs



Autômatos que esvaziam a pilha

- **Pergunta chave:** quais são as strings que fazem um determinado AP esvaziar completamente a sua pilha (e então morra na próxima transição)?
- **Definição.** Seja $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ um AP. Então $N(P) = \{ w \in \Sigma^* \mid (q_0, w, Z_0) \vdash_P^* (q, \varepsilon, \varepsilon) \}$ para $q \in Q$ qualquer.
- **Exemplo.** O AP P_{RR} nunca esvazia sua pilha, portanto $N(P_{RR}) = \emptyset$.
- **Duas possibilidades:** ou P esvazia a pilha ou não
  - Permite usar $N(P)$ para definir uma certa “linguagem” do autômato
  - Que pode ser diferente de $L(P)$

Autômatos que esvaziam a pilha

- **Teorema.** Seja P um AP qualquer. Então existe um AP P' tal que $L(P') = N(P)$

- **Teorema.** Seja P um AP qualquer. Então existe um AP P' tal que $N(P') = L(P)$

APs determinísticos

- **Definição.** Um autômato com pilha determinístico (APD) P é um autômato com pilha $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ cuja função de transição δ possui as seguintes restrições
  - Para quaisquer $q \in Q, a \in \Sigma \cup \{\varepsilon\}$ e $X \in \Gamma$, temos $|\delta(q, a, X)| \leq 1$.
  - Se existe $q \in Q, a \in \Sigma$ e $X \in \Gamma$ tal que $\delta(q, a, X) \neq \emptyset$, então $\delta(q, \varepsilon, X) = \emptyset$.

Exercício 4.16 (Resolvido) Considere o alfabeto $\{0, 1, M\}$ e a seguinte linguagem sobre este alfabeto: $L_{\text{RMR}} = \{wMw^R : \text{tal que } w \in \{0, 1\}^*\}$. Mostre um APD que decida L_{RMR}.

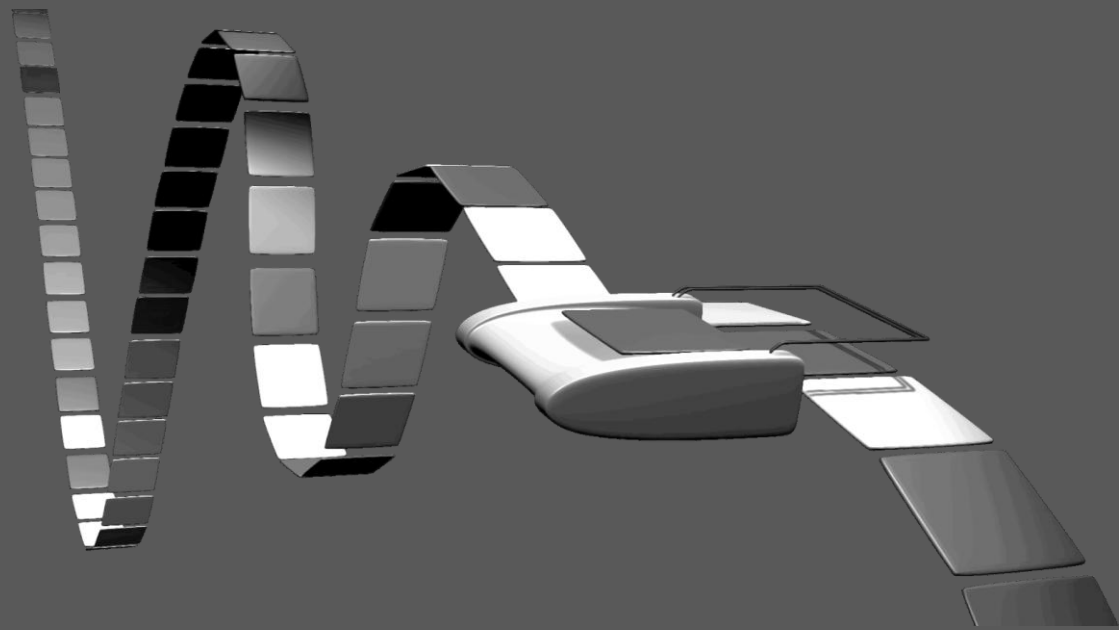
Alguns resultados

- **Teorema.** O conjunto de linguagens decididas por APs é estritamente maior do que o conjunto das linguagens regulares.

- **Teorema.** Toda linguagem decidida por um APD também é decidida por um AP.

- **Teorema.** Não existe APD que decida a linguagem L_{RR}.

Para fechar



Para fechar

- Hoje
  - Autômatos com pilha
- Próxima aula
  - Gramáticas livres de contexto
    - Definem regras para produzir strings
    - Equivalentes a APs

Referências

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