Teoria da Computação

Prof. Maicon R. Zatelli

Aula 3 - Linguagens Livres de Contexto

Universidade Federal de Santa Catarina

Florianópolis - Brasil

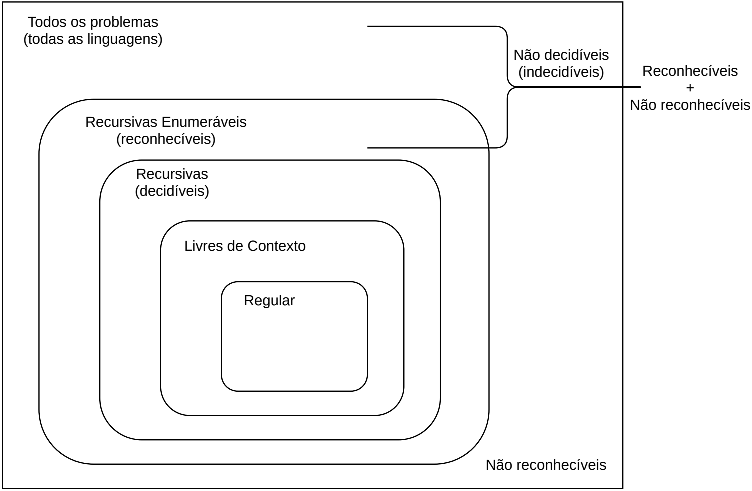
Introdução

- Linguagens livres de contexto
- Gramáticas livres de contexto
- Forma normal de Chomsky
- Autômatos com pilha
- Linguagens não livres de contexto e lema do bombeamento

Material de apoio

- Livro Sipser, Capítulo 2
- Livro Hopcroft, Capítulos 5,6,7

Hierarquia de Chomsky



Indecidíveis - são todas as linguagens Turing-reconhecíveis mas não decidíveis e também as linguagens não Turing-reconhecíveis

Linguagens Livres de Contexto (LLC)

São linguagens mais poderosas que as linguagens regulares.

Exemplos de Uso

- Analisador sintático
- Verificar balanceamentos (ex: palavras com mesma quantidade de a's e b's)
- Avaliar expressões aritméticas
- Processadores de texto
- Construção de linguagens de programação

Gramáticas Livres de Contexto (GLC)

É uma forma de descrever linguagens livres de contexto

Por exemplo, em uma linguagem de programação temos a gramática desta linguagem, que permite escrever todas as construções possíveis com aquela linguagem

- Assim, toda linguagem L é livre de contexto se e somente se ela pode ser descrita por uma GLC G

Note que para mostrar que $L(G) = L$ temos que mostrar que $L \subseteq L(G)$ e $L(G) \subseteq L$

Gramáticas Livres de Contexto (GLC)

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

- V é um conjunto finito de símbolos não terminais (variáveis)
- T é um conjunto finito de símbolos terminais (alfabeto)
- $S \in V$ é o símbolo inicial
- P é um conjunto de regras de produção na forma
  - $A \rightarrow \alpha$, onde $A \in V$ e $\alpha \in (V \cup T)^*$

- Há apenas uma única variável do lado esquerdo da regra
- Há qualquer combinação de variáveis e terminais do lado direito da regra

Gramáticas Livres de Contexto (GLC)

$$L = \{w | w \in \{a, b\}^+ \text{ e } w = a^n b^n \text{ e } n \geq 1\}$$

Gramáticas Livres de Contexto (GLC)

$$L = \{w|w \in \{a, b\}^+ \text{ e } w = a^n b^n \text{ e } n \geq 1\}$$

$S \rightarrow aSb$
$S \rightarrow ab$

$S \rightarrow aSb|ab$

Gramáticas Livres de Contexto (GLC)

$$L = \{w|w \in \{a,b\}^+ \text{ e } w = a^n b^n \text{ e } n \geq 1\}$$

$S \rightarrow aSb$
$S \rightarrow ab$

$S \rightarrow aSb|ab$

- $S = S$
- $T = \{a, b\}$
- $V = \{S\}$
- $P = \{S \rightarrow aSb, S \rightarrow ab\}$

Gramáticas Livres de Contexto (GLC)

Derivação é o processo de gerar cadeias pertencentes à linguagem através da aplicação sucessiva das regras de produção, ou seja, $\alpha_1 X \alpha_2$, usando $X \rightarrow Y$, obtemos $\alpha_1 Y \alpha_2$ e escrevemos $\alpha_1 X \alpha_2 \rightarrow \alpha_1 Y \alpha_2$, onde

- $X \in V$,
- $\alpha_1, \alpha_2 \in (V \cup T)^*$,
- $X \rightarrow Y \in P$, e
- $Y \in (V \cup T)^*$

Gramáticas Livres de Contexto (GLC)

Derivação é o processo de gerar cadeias pertencentes à linguagem através da aplicação sucessiva das regras de produção, ou seja, $\alpha_1 X \alpha_2$, usando $X \rightarrow Y$, obtemos $\alpha_1 Y \alpha_2$ e escrevemos $\alpha_1 X \alpha_2 \rightarrow \alpha_1 Y \alpha_2$, onde

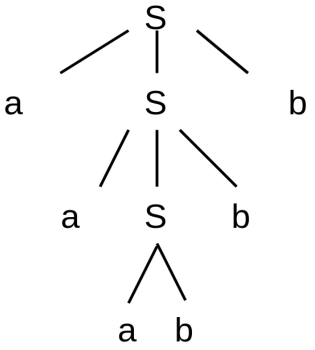
- $X \in V$,
- $\alpha_1, \alpha_2 \in (V \cup T)^*$,
- $X \rightarrow Y \in P$, e
- $Y \in (V \cup T)^*$

1. $S \rightarrow aSb$
2. $S \rightarrow ab$

$S \xrightarrow{(1)} a\underline{S}b \xrightarrow{(1)} aa\underline{S}bb \xrightarrow{(2)} aaabbb$ (derivação de um em um passo)
$S \xrightarrow{*} aaabbb$ (derivação em 0 ou mais passos)

Gramáticas Livres de Contexto (GLC)

Árvore de derivação



- Folhas são sempre terminais
- Nós internos sempre são não terminais
- Filhos dos nós internos formam as regras de produção

$$L(G) = \{w | w \in T^* \text{ e } S \xrightarrow{*} w\}$$

Gramáticas Livres de Contexto (GLC)

$L = \{w | w \in \{(,)\}^+$ e w possui os parênteses balanceados $\}$

- $L = \{(), ()()(), (()()())(), ...\}$

Gramáticas Livres de Contexto (GLC)

$L = \{w | w \in \{(,)\}^+$ e w possui os parênteses balanceados $\}$

- $L = \{(), ()()(), (()()())(), ...\}$

$S \rightarrow ASB | AB | SS$
$A \rightarrow ($
$B \rightarrow)$

Gramáticas Livres de Contexto (GLC)

$L = \{w | w \in \{(,)\}^+$ e w possui os parênteses balanceados $\}$

- $L = \{(), ()()(), (()()())(), ...\}$

$S \rightarrow ASB|AB|SS$
$A \rightarrow ($
$B \rightarrow)$

Derivando ()(()) (derivação mais à esquerda)

- $S \rightarrow \underline{S}S \rightarrow \underline{A}BS \rightarrow (\underline{B}S \rightarrow ()\underline{S} \rightarrow ()\underline{A}SB \rightarrow ()(\underline{S}B \rightarrow ()(\underline{A}BB \rightarrow ()((\underline{B}B \rightarrow ()(()\underline{B} \rightarrow ()(())$

A derivação mais à esquerda sempre substitui a variável mais à esquerda durante o processo de derivação.

- Tente fazer a derivação mais à direita
- Tente criar a árvore de derivação
- Faça a derivação da cadeia (()()())

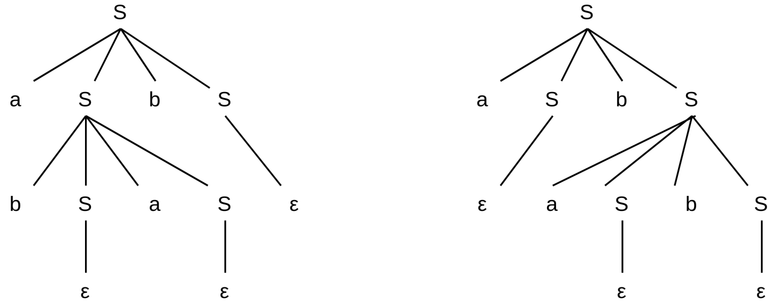
Gramáticas Livres de Contexto (GLC)

Uma gramática é dita ambígua se é possível construir duas ou mais árvores de derivação diferentes para uma mesma palavra

$S \rightarrow aSbS | bSaS | \varepsilon$

Para mostrar que a gramática acima é ambígua basta mostrar duas árvores de derivação diferentes para uma mesma palavra

Seja a palavra $w = abab$



Uma linguagem é inerentemente ambígua se qualquer gramática que a gera é ambígua.

- $L = \{w|w \in \{0, 1, 2\}^* \text{ e } w = 0^n 1^m 2^k \text{ e } (n = m \text{ ou } m = k)\}$
- Ver Lema de Odgen para mostrar que algumas linguagens são inerentemente ambíguas.

Para mostrar que uma gramática livre de contexto não é ambígua basta mostrar que o número de árvores de derivação da gramática G para gerar palavras de tamanho n, para todo n, é o mesmo que o número de palavras de tamanho n na linguagem gerada por G

Linguagens Regulares x Linguagens Livres de Contexto

Teorema: toda linguagem regular é uma linguagem livre de contexto

Linguagens Regulares x Linguagens Livres de Contexto

Teorema: toda linguagem regular é uma linguagem livre de contexto

Prova: por construção

Linguagens Regulares x Linguagens Livres de Contexto

Teorema: toda linguagem regular é uma linguagem livre de contexto

Prova: por construção

- Seja L uma LR. Então há um AFD $M = (Q, \Sigma, \delta, q0, F)$ que a reconhece.

Linguagens Regulares x Linguagens Livres de Contexto

Teorema: toda linguagem regular é uma linguagem livre de contexto

Prova: por construção

- Seja L uma LR. Então há um AFD $M = (Q, \Sigma, \delta, q0, F)$ que a reconhece.
- Construiremos agora uma gramática livre de contexto G_M tal que $L(M) = L(G_M)$

Linguagens Regulares x Linguagens Livres de Contexto

Teorema: toda linguagem regular é uma linguagem livre de contexto

Prova: por construção

- Seja L uma LR. Então há um AFD $M = (Q, \Sigma, \delta, q0, F)$ que a reconhece.
- Construiremos agora uma gramática livre de contexto G_M tal que $L(M) = L(G_M)$
- Cada estado do AFD M é associado a um símbolo não terminal da GLC G_M, sendo o estado inicial $q0$ associado ao símbolo inicial S

Linguagens Regulares x Linguagens Livres de Contexto

Teorema: toda linguagem regular é uma linguagem livre de contexto

Prova: por construção

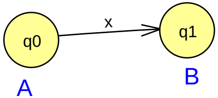
- Seja L uma LR. Então há um AFD $M = (Q, \Sigma, \delta, q0, F)$ que a reconhece.
- Construiremos agora uma gramática livre de contexto G_M tal que $L(M) = L(G_M)$
- Cada estado do AFD M é associado a um símbolo não terminal da GLC G_M, sendo o estado inicial $q0$ associado ao símbolo inicial S
- Para cada transição de M, cria-se uma regra de produção em G_M
  - o estado de origem torna-se o não terminal à esquerda da regra
  - o estado de destino torna-se um não terminal do lado direito da regra, após o terminal lido na transição

Linguagens Regulares x Linguagens Livres de Contexto

Teorema: toda linguagem regular é uma linguagem livre de contexto

Prova: por construção

- Seja L uma LR. Então há um AFD $M = (Q, \Sigma, \delta, q0, F)$ que a reconhece.
- Construiremos agora uma gramática livre de contexto G_M tal que $L(M) = L(G_M)$
- ...
- Para cada transição de M, cria-se uma regra de produção em G_M
  - o estado de origem torna-se o não terminal à esquerda da regra
  - o estado de destino torna-se um não terminal do lado direito da regra, após o terminal lido na transição



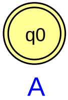
$A \rightarrow xB$

Linguagens Regulares x Linguagens Livres de Contexto

Teorema: toda linguagem regular é uma linguagem livre de contexto

Prova: por construção

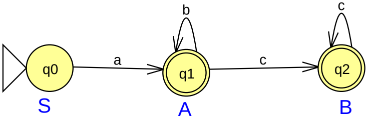
- Seja L uma LR. Então há um AFD $M = (Q, \Sigma, \delta, q0, F)$ que a reconhece.
- Construiremos agora uma gramática livre de contexto G_M tal que $L(M) = L(G_M)$
- ...
- Cria-se uma regra para cada não terminal associado a um estado final, onde o lado direito da regra é formado apenas pela palavra vazia ε



$A \rightarrow \varepsilon$

LR x LLC - Exemplo

M

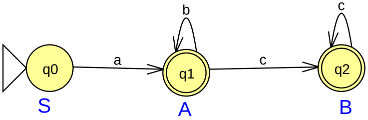


G_M

$S \rightarrow$
$A \rightarrow$
$B \rightarrow$

LR x LLC - Exemplo

M

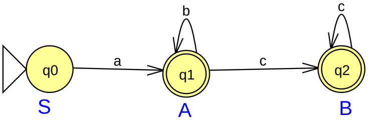


G_M

$$S \rightarrow aA$$
$$A \rightarrow bA|cB$$
$$B \rightarrow cB$$

LR x LLC - Exemplo

M



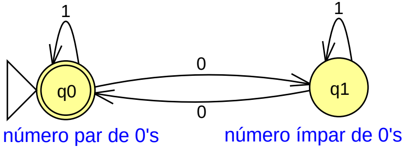
G_M

$$S \rightarrow aA$$
$$A \rightarrow bA|cB|\varepsilon$$
$$B \rightarrow cB|\varepsilon$$

Note que $L(M) = L(G_M)$ e é simples de visualizar que qualquer palavra irá produzir a mesma sequência de estados (transições) e regras de produção utilizadas

LR x LLC - Exemplo

Tente converter o autômato abaixo em uma gramática livre de contexto



GLC - Formas Normais

Formas normais são representações de GLC com restrições rígidas na forma das produções, mas sem reduzir o poder de geração das GLC, exceto pela geração da palavra vazia ε

Duas formas normais são mais conhecidas

- Forma Normal de Chomsky (FNC)
  - Produções são da forma $A \rightarrow BC$ ou $A \rightarrow a$, onde $A, B, C \in V$ e $a \in T$

GLC - Formas Normais

Formas normais são representações de GLC com restrições rígidas na forma das produções, mas sem reduzir o poder de geração das GLC, exceto pela geração da palavra vazia ε

Duas formas normais são mais conhecidas

- Forma Normal de Chomsky (FNC)
  - Produções são da forma $A \rightarrow BC$ ou $A \rightarrow a$, onde $A, B, C \in V$ e $a \in T$
  - Árvore de derivação é sempre binária

GLC - Formas Normais

Formas normais são representações de GLC com restrições rígidas na forma das produções, mas sem reduzir o poder de geração das GLC, exceto pela geração da palavra vazia ε

Duas formas normais são mais conhecidas

- Forma Normal de Chomsky (FNC)
  - Produções são da forma $A \rightarrow BC$ ou $A \rightarrow a$, onde $A, B, C \in V$ e $a \in T$
  - Árvore de derivação é sempre binária
- Forma Normal de Greibach (FNG)
  - Produções são da forma $A \rightarrow a\alpha$, onde α é uma cadeia de V^*, $a \in T$ e $A \in V$

GLC - Formas Normais

Formas normais são representações de GLC com restrições rígidas na forma das produções, mas sem reduzir o poder de geração das GLC, exceto pela geração da palavra vazia ε

Duas formas normais são mais conhecidas

- Forma Normal de Chomsky (FNC)
  - Produções são da forma $A \rightarrow BC$ ou $A \rightarrow a$, onde $A, B, C \in V$ e $a \in T$
  - Árvore de derivação é sempre binária
- Forma Normal de Greibach (FNG)
  - Produções são da forma $A \rightarrow a\alpha$, onde α é uma cadeia de V^*, $a \in T$ e $A \in V$
  - A cada passo de derivação um terminal é consumido

GLC - Formas Normais

Formas normais são representações de GLC com restrições rígidas na forma das produções, mas sem reduzir o poder de geração das GLC, exceto pela geração da palavra vazia ε

Duas formas normais são mais conhecidas

- Forma Normal de Chomsky (FNC)
  - Produções são da forma $A \rightarrow BC$ ou $A \rightarrow a$, onde $A, B, C \in V$ e $a \in T$
  - Árvore de derivação é sempre binária

- Forma Normal de Greibach (FNG)
  - Produções são da forma $A \rightarrow a\alpha$, onde α é uma cadeia de V^*, $a \in T$ e $A \in V$
  - A cada passo de derivação um terminal é consumido

Ambas (FNC e FNG) não possuem ε em suas linguagens

GLC - Formas Normais

Teorema: qualquer linguagem livre de contexto é gerada por uma gramática livre de contexto na Forma Normal de Chomsky

GLC - Formas Normais

Teorema: qualquer linguagem livre de contexto é gerada por uma gramática livre de contexto na Forma Normal de Chomsky

Prova: por construção

GLC - Formas Normais

Teorema: qualquer linguagem livre de contexto é gerada por uma gramática livre de contexto na Forma Normal de Chomsky

Prova: por construção

- Vamos converter qualquer GLC para a FNC seguindo os passos abaixo

GLC - Formas Normais

Teorema: qualquer linguagem livre de contexto é gerada por uma gramática livre de contexto na Forma Normal de Chomsky

Prova: por construção

- Vamos converter qualquer GLC para a FNC seguindo os passos abaixo

1. Criamos uma nova variável como símbolo inicial

GLC - Formas Normais

Teorema: qualquer linguagem livre de contexto é gerada por uma gramática livre de contexto na Forma Normal de Chomsky

Prova: por construção

- Vamos converter qualquer GLC para a FNC seguindo os passos abaixo

1. Criamos uma nova variável como símbolo inicial
2. Eliminamos as produções vazias, na forma $A \rightarrow \varepsilon$

GLC - Formas Normais

Teorema: qualquer linguagem livre de contexto é gerada por uma gramática livre de contexto na Forma Normal de Chomsky

Prova: por construção

- Vamos converter qualquer GLC para a FNC seguindo os passos abaixo

1. Criamos uma nova variável como símbolo inicial
2. Eliminamos as produções vazias, na forma $A \rightarrow \varepsilon$
3. Eliminamos as produções unitárias, na forma $A \rightarrow B$

GLC - Formas Normais

Teorema: qualquer linguagem livre de contexto é gerada por uma gramática livre de contexto na Forma Normal de Chomsky

Prova: por construção

- Vamos converter qualquer GLC para a FNC seguindo os passos abaixo

1. Criamos uma nova variável como símbolo inicial
2. Eliminamos as produções vazias, na forma $A \rightarrow \varepsilon$
3. Eliminamos as produções unitárias, na forma $A \rightarrow B$
4. Convertemos as produções restantes para a forma apropriada
    - $A \rightarrow BC$ ou $A \rightarrow a$

GLC - Formas Normais

Teorema: qualquer linguagem livre de contexto é gerada por uma gramática livre de contexto na Forma Normal de Chomsky

Prova: por construção

- Vamos converter qualquer GLC para a FNC seguindo os passos abaixo

1. Criamos uma nova variável como símbolo inicial
2. Eliminamos as produções vazias, na forma $A \to \varepsilon$
3. Eliminamos as produções unitárias, na forma $A \to B$
4. Convertemos as produções restantes para a forma apropriada
    - $A \to BC$ ou $A \to a$
5. Possível também eliminar os símbolos inúteis (inalcançáveis ou improdutivos)

GLC - Formas Normais - Exemplo

$S \rightarrow ASA|aB$
$A \rightarrow B|S$
$B \rightarrow b|\varepsilon$

GLC - Formas Normais - Exemplo

$S \rightarrow ASA|aB$
$A \rightarrow B|S$
$B \rightarrow b|\varepsilon$

- Criamos uma nova variável como símbolo inicial

GLC - Formas Normais - Exemplo

$S \to ASA|aB$
$A \to B|S$
$B \to b|\varepsilon$

- Criamos uma nova variável como símbolo inicial

$S' \to S$
$S \to ASA|aB$
$A \to B|S$
$B \to b|\varepsilon$

GLC - Formas Normais - Exemplo

$S' \rightarrow S$
$S \rightarrow ASA|aB$
$A \rightarrow B|S$
$B \rightarrow b|\varepsilon$

GLC - Formas Normais - Exemplo

$S' \to S$
$S \to ASA|aB$
$A \to B|S$
$B \to b|\varepsilon$

- Eliminamos as produções vazias, na forma $A \to \varepsilon$

GLC - Formas Normais - Exemplo

$S' \to S$
$S \to ASA|aB$
$A \to B|S$
$B \to b|\varepsilon$

- Eliminamos as produções vazias, na forma $A \to \varepsilon$
- Identificar as variáveis anuláveis
  - Variáveis que geram ε ($B \to \varepsilon$) ou que apenas possuem variáveis anuláveis à direita da produção ($A \to B$)
  - $\{B, A\}$
- Reescrever cada regra com e sem as variáveis anuláveis

$S' \to S$
$S \to ASA|aB|AS|SA|S|a$
$A \to B|S$
$B \to b$

GLC - Formas Normais - Exemplo

$S' \rightarrow S$
$S \rightarrow ASA|aB|AS|SA|S|a$
$A \rightarrow B|S$
$B \rightarrow b$

GLC - Formas Normais - Exemplo

$S' \rightarrow S$
$S \rightarrow ASA|aB|AS|SA|S|a$
$A \rightarrow B|S$
$B \rightarrow b$

- Eliminamos as produções unitárias, na forma $A \rightarrow B$

GLC - Formas Normais - Exemplo

$S' \rightarrow S$
$S \rightarrow ASA|aB|AS|SA|S|a$
$A \rightarrow B|S$
$B \rightarrow b$

- Eliminamos as produções unitárias, na forma $A \rightarrow B$
- Escrever a gramática sem as produções unitárias

GLC - Formas Normais - Exemplo

$S' \rightarrow S$
$S \rightarrow ASA|aB|AS|SA|S|a$
$A \rightarrow B|S$
$B \rightarrow b$

- Eliminamos as produções unitárias, na forma $A \rightarrow B$
- Escrever a gramática sem as produções unitárias

$S' \rightarrow$
$S \rightarrow ASA|aB|AS|SA|a$
$A \rightarrow$
$B \rightarrow b$

GLC - Formas Normais - Exemplo

$S' \to S$
$S \to ASA|aB|AS|SA|S|a$
$A \to B|S$
$B \to b$

- Eliminamos as produções unitárias, na forma $A \to B$

- Escrever a gramática sem as produções unitárias
- Incluir o que pode ser gerado com as produções unitárias

GLC - Formas Normais - Exemplo

$S' \rightarrow S$
$S \rightarrow ASA|aB|AS|SA|S|a$
$A \rightarrow B|S$
$B \rightarrow b$

- Eliminamos as produções unitárias, na forma $A \rightarrow B$

- Escrever a gramática sem as produções unitárias
- Incluir o que pode ser gerado com as produções unitárias

$S' \rightarrow$ **ASA | aB | AS | SA | a**
$S \rightarrow ASA|aB|AS|SA|a$
$A \rightarrow$ **b | ASA | aB | AS | SA | a**
$B \rightarrow b$

GLC - Formas Normais - Exemplo

$S' \rightarrow$ **ASA | aB | AS | SA | a**
$S \rightarrow ASA|aB|AS|SA|a$
$A \rightarrow$ **b | ASA | aB | AS | SA | a**
$B \rightarrow b$

GLC - Formas Normais - Exemplo

$S' \rightarrow$ **ASA | aB | AS | SA | a**
$S \rightarrow ASA|aB|AS|SA|a$
$A \rightarrow$ **b | ASA | aB | AS | SA | a**
$B \rightarrow b$

- Convertemos as produções restantes para a forma apropriada
  - $A \rightarrow BC$ ou $A \rightarrow a$

GLC - Formas Normais - Exemplo

$S' \rightarrow$ **ASA | aB | AS | SA | a**
$S \rightarrow ASA|aB|AS|SA|a$
$A \rightarrow$ **b | ASA | aB | AS | SA | a**
$B \rightarrow b$

- Convertemos as produções restantes para a forma apropriada
  - $A \rightarrow BC$ ou $A \rightarrow a$
- Assegurar que no lado direito possui apenas variáveis ou um único terminal

GLC - Formas Normais - Exemplo

$S' \to$ **ASA | aB | AS | SA | a**
$S \to ASA|aB|AS|SA|a$
$A \to$ **b | ASA | aB | AS | SA | a**
$B \to b$

- Convertemos as produções restantes para a forma apropriada
  - $A \to BC$ ou $A \to a$
- Assegurar que no lado direito possui apenas variáveis ou um único terminal

$S \to aB$

$S \to A'B$
$A' \to a$

GLC - Formas Normais - Exemplo

$S' \rightarrow \textbf{ASA | aB | AS | SA | a}$
$S \rightarrow ASA|aB|AS|SA|a$
$A \rightarrow \textbf{b | ASA | aB | AS | SA | a}$
$B \rightarrow b$

- Convertemos as produções restantes para a forma apropriada
  - $A \rightarrow BC$ ou $A \rightarrow a$
- Assegurar que o lado direito possui sempre duas variáveis (no máximo)

$S' \rightarrow ASA$

$S' \rightarrow CA$
$C \rightarrow AS$

GLC - Formas Normais - Exemplo

$S' \rightarrow$ **ASA | aB | AS | SA | a**
$S \rightarrow ASA|aB|AS|SA|a$
$A \rightarrow$ **b | ASA | aB | AS | SA | a**
$B \rightarrow b$

- Convertemos as produções restantes para a forma apropriada

GLC - Formas Normais - Exemplo

$S' \to$ **ASA | aB | AS | SA | a**
$S \to ASA|aB|AS|SA|a$
$A \to$ **b | ASA | aB | AS | SA | a**
$B \to b$

- Convertemos as produções restantes para a forma apropriada

$S \to aB$

$S \to A'B$
$A' \to a$

$S' \to ASA$

$S' \to CA$
$C \to AS$

Resultado

$S' \to CA|A'B|AS|SA|a$
$C \to AS$
$A' \to a$
$S \to CA|A'B|AS|SA|a$
$A \to b|CA|A'B|AS|SA|a$
$B \to b$

GLC - Formas Normais

Tente colocar as gramáticas a seguir na forma normal de Chomsky

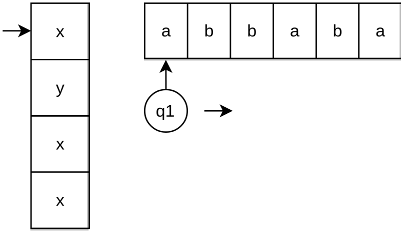
$S \to ABC|aB$
$B \to aB|a$
$A \to bA|\varepsilon$
$C \to AB|\varepsilon$

$S \to AbaC$
$A \to BC$
$B \to b|\varepsilon$
$C \to D|\varepsilon$
$D \to d$

$S \to Aa|B$
$B \to A|bb$
$A \to a|bc|B$

Autômatos com Pilha (AP)

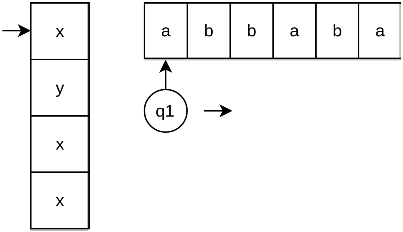
Um autômato com pilha (AP) é um autômato finito (AF) com uma memória auxiliar (uma pilha). Um AP é um reconhecedor de linguagens livres de contexto.



Diferença para os autômatos finitos

- Um AF escolhe uma transição analisando o símbolo atual da cadeia de entrada e o estado atual
- Um AP analisa o símbolo atual da cadeia de entrada, o estado atual, e o topo da pilha

Autômatos com Pilha (AP)



Um AP manipula uma pilha como resultado de uma transição

- Desempilha um símbolo da pilha
- Empilha um símbolo da pilha
- Ignora a pilha (empilha/desempilha topo da pilha)

$AP = (Q, \Sigma, \Delta, \delta, q0, Z_0, F)$

- Q é um conjunto finito de estados
- Σ é um conjunto finito de símbolos de entrada (alfabeto de entrada)
- Δ é um conjunto finito de símbolos da pilha (alfabeto de pilha)
- $\delta : Q \times \Sigma_\varepsilon \times \Delta_\varepsilon \rightarrow P(Q \times \Delta_\varepsilon)$ é uma função de transição, onde $\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$ e $\Delta_\varepsilon = \Delta \cup \{\varepsilon\}$
  - δ(estado atual,símbolo lido entrada,topo da pilha) $\rightarrow$ {(novo estado,novo topo da pilha)}
- $q0 \in Q$ é um estado inicial
- Z_0 é o símbolo de fundo da pilha
- $F \subseteq Q$ é um conjunto de estados finais

Autômatos com Pilha (AP)

δ(estado atual,símbolo lido entrada,topo da pilha) $\rightarrow$ $\{$novo estado,novo topo da pilha)$\}$ topo da pilha

- desempilha sempre algum símbolo
- se ε na transição, não desempilha nada

novo topo da pilha

- empilha sempre algum símbolo
- se ε na transição, não empilha nada

Autômatos com Pilha (AP)

Um AP pode aceitar por pilha vazia ou estado final

Pilha vazia

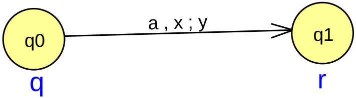
- A pilha inicia com o símbolo de fundo de pilha
- Aceitação ocorre quando a pilha fica vazia (quando o símbolo de fundo da pilha é removido). Não é necessário possuir estado final.
- $L(M) = \{w \in \Sigma^* | (q0, w, Z_0) \overset{*}{\rightarrow} (r, \varepsilon, \varepsilon) \text{ e } r \in Q\}$

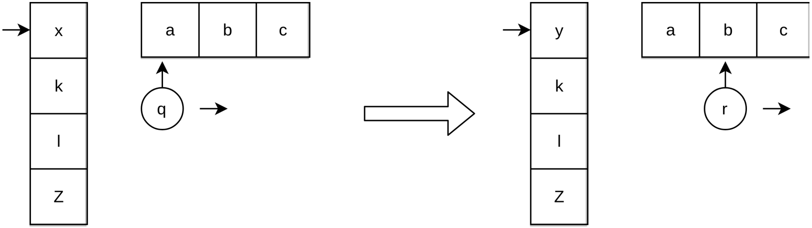
Estado final

- Aceitação ocorre quando um estado final é alcançado e a entrada foi toda consumida. Não é necessário que a pilha esteja vazia.
- $L(M) = \{w \in \Sigma^* | (q0, w, Z_0) \overset{*}{\rightarrow} (r, \varepsilon, \gamma) \text{ e } r \in F \text{ e } \gamma \in \Delta^*\}$

Autômatos com Pilha (AP)

$$(q, a, x) \rightarrow (r, y)$$





Autômatos com Pilha (AP)

Vamos também permitir que possa ser empilhada uma cadeia de símbolos a cada transição disparada. Assim, as regras de transição seguem a forma abaixo:

$\delta : Q \times \Sigma_\varepsilon \times \Delta_\varepsilon \to P(Q \times \Delta_\varepsilon^*)$ é uma função de transição, onde $\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$ e $\Delta_\varepsilon = \Delta \cup \{\varepsilon\}$

Notações:

- $a, x \to xyz$
- a, x, xyz
- $a, x; xyz$
- $a, x/xyz$

Autômatos com Pilha (AP)

O instantâneo (snapshot) de um AP é dado por (q, w, γ), onde q é o estado atual, w é a parte da entrada ainda não lida, e γ é o conteúdo da pilha.

Desta forma, ao disparar uma transição temos que $(q, aw, x\gamma) \rightarrow (r, w, y\gamma)$, onde a é o símbolo da entrada consumida, x é o símbolo desempilhado, w é o que ainda deve ser consumido da entrada, y é o símbolo empilhado, e o AP moveu do estado q para o estado r

Você pode assumir (para esta disciplina) que a pilha já está inicializada contendo o símbolo de fundo de pilha (Z)

Autômatos com Pilha (AP)

$L = \{w | w \in \{0,1\}^* \text{ e } w = 0^n1^n \text{ e } n \geq 1\}$

$L = \{w | w \in \{0,1\}^* \text{ e } w = 0^n 1^n \text{ e } n \geq 1\}$

Intuição:

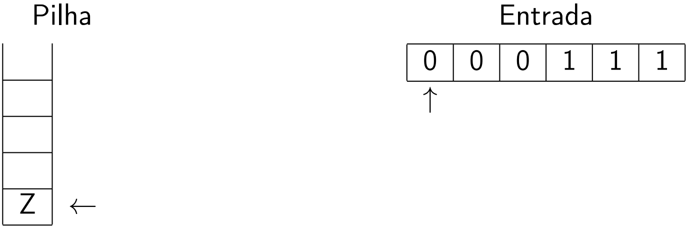
Vamos simular o comportamento do AP para a entrada $w = 000111$

Autômatos com Pilha (AP)

$L = \{w | w \in \{0,1\}^* \text{ e } w = 0^n1^n \text{ e } n \geq 1\}$

Intuição:

Vamos simular o comportamento do AP para a entrada $w = 000111$



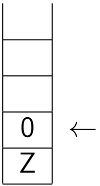
Autômatos com Pilha (AP)

$L = \{w|w \in \{0,1\}^* \text{ e } w = 0^n1^n \text{ e } n \geq 1\}$

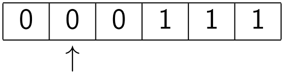
Intuição:

Vamos simular o comportamento do AP para a entrada $w = 000111$

Pilha



Entrada



Autômatos com Pilha (AP)

$L = \{w | w \in \{0, 1\}^* \text{ e } w = 0^n 1^n \text{ e } n \geq 1\}$

Intuição:

Vamos simular o comportamento do AP para a entrada $w = 000111$

Pilha

0	$\leftarrow$
0	
Z	

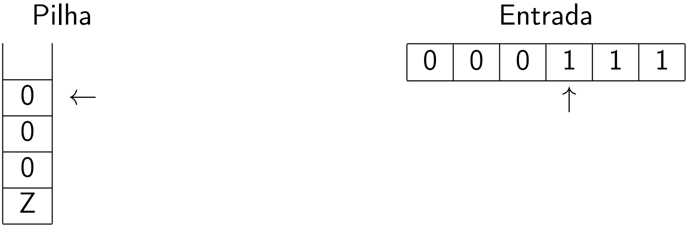
Entrada

0	0	0	1	1	1
		$\uparrow$			

$L = \{w | w \in \{0,1\}^* \text{ e } w = 0^n1^n \text{ e } n \geq 1\}$

Intuição:

Vamos simular o comportamento do AP para a entrada $w = 000111$

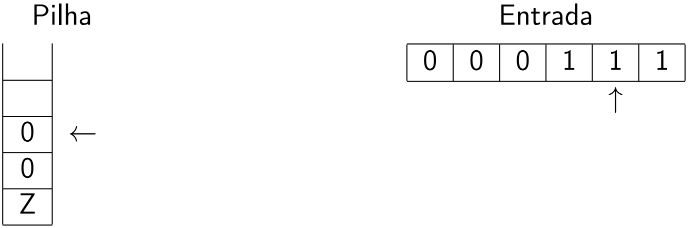


Autômatos com Pilha (AP)

$L = \{w|w \in \{0,1\}^* \text{ e } w = 0^n1^n \text{ e } n \geq 1\}$

Intuição:

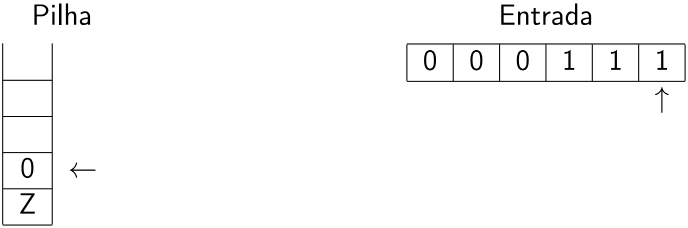
Vamos simular o comportamento do AP para a entrada $w = 000111$



$L = \{w | w \in \{0,1\}^* \text{ e } w = 0^n 1^n \text{ e } n \geq 1\}$

Intuição:

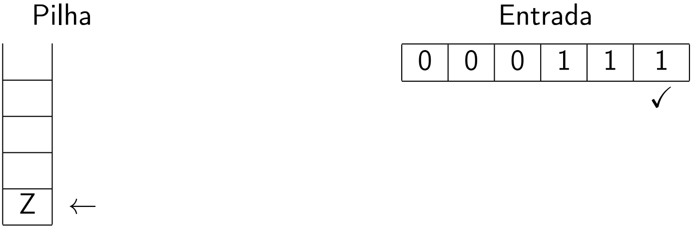
Vamos simular o comportamento do AP para a entrada
$w = 000111$



$L = \{w | w \in \{0, 1\}^* \text{ e } w = 0^n 1^n \text{ e } n \geq 1\}$

Intuição:

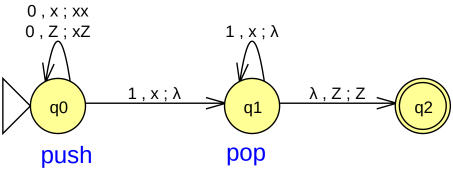
Vamos simular o comportamento do AP para a entrada $w = 000111$



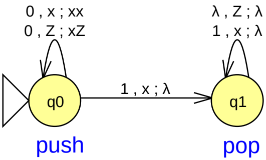
Autômatos com Pilha (AP)

$$L = \{w | w \in \{0,1\}^* \text{ e } w = 0^n 1^n \text{ e } n \geq 1\}$$

Estado Final



Pilha Vazia



$Q = \{q0, q1, q2\}$, $q0 = q0$, $F = \{q2\}$
$\Sigma = \{0, 1\}$, $\Delta = \{Z, x\}$, $Z_0 = Z$
$\delta(q0, 0, Z) \to (q0, xZ)$, $\delta(q0, 0, x) \to (q0, xx)$,
$\delta(q0, 1, x) \to (q1, \lambda)$, $\delta(q1, 1, x) \to (q1, \lambda)$, $\delta(q1, \lambda, Z) \to (q2, Z)$

Autômatos com Pilha (AP)

$L = \{ww^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$

Autômatos com Pilha (AP)

$L = \{ww^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par }\}$

Antes de tentar a linguagem acima, vamos tentar:
$L = \{w2w^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par }\}$

$L = \{ww^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$

Antes de tentar a linguagem acima, vamos tentar:
$L = \{w2w^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$

Intuição:

Vamos simular o comportamento do AP para a entrada
$w = 1002001$

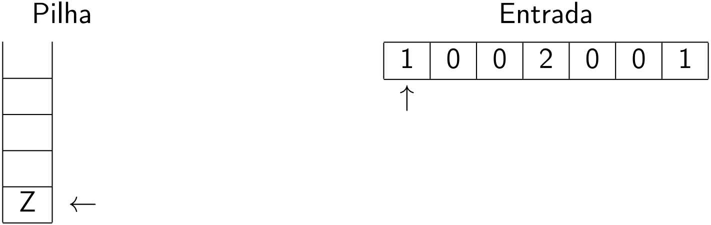
Autômatos com Pilha (AP)

$L = \{ww^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$

Antes de tentar a linguagem acima, vamos tentar:
$L = \{w2w^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$

Intuição:

Vamos simular o comportamento do AP para a entrada
$w = 1002001$



$L = \{ww^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$

Antes de tentar a linguagem acima, vamos tentar:
$L = \{w2w^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$

Intuição:

Vamos simular o comportamento do AP para a entrada
$w = 1002001$

Pilha

1	$\leftarrow$
Z	

Entrada

1	0	0	2	0	0	1
	$\uparrow$					

Autômatos com Pilha (AP)

$L = \{ww^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$

Antes de tentar a linguagem acima, vamos tentar:
$L = \{w2w^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$

Intuição:

Vamos simular o comportamento do AP para a entrada
$w = 1002001$

Pilha

0	←
1	
Z	

Entrada

1	0	0	2	0	0	1
		↑				

Autômatos com Pilha (AP)

$L = \{ww^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par }\}$

Antes de tentar a linguagem acima, vamos tentar:
$L = \{w2w^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par }\}$

Intuição:

Vamos simular o comportamento do AP para a entrada
$w = 1002001$

Pilha

0	←
0	
1	
Z	

Entrada

1	0	0	2	0	0	1
			↑			

Autômatos com Pilha (AP)

$L = \{ww^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par }\}$

Antes de tentar a linguagem acima, vamos tentar:
$L = \{w2w^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par }\}$

Intuição:

Vamos simular o comportamento do AP para a entrada
$w = 1002001$

Pilha

0	←
0	
1	
Z	

Entrada

1	0	0	2	0	0	1
				↑		

Autômatos com Pilha (AP)

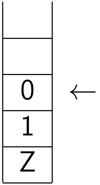
$L = \{ww^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$

Antes de tentar a linguagem acima, vamos tentar:
$L = \{w2w^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$

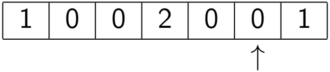
Intuição:

Vamos simular o comportamento do AP para a entrada
$w = 1002001$

Pilha



Entrada



Autômatos com Pilha (AP)

$L = \{ww^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$

Antes de tentar a linguagem acima, vamos tentar:
$L = \{w2w^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$

Intuição:

Vamos simular o comportamento do AP para a entrada
$w = 1002001$

Pilha

1
Z

Entrada

1	0	0	2	0	0	1

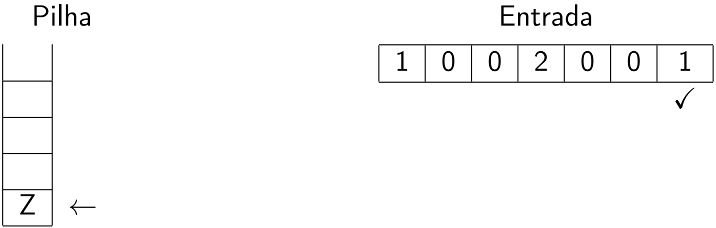
Autômatos com Pilha (AP)

$L = \{ww^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$

Antes de tentar a linguagem acima, vamos tentar:
$L = \{w2w^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$

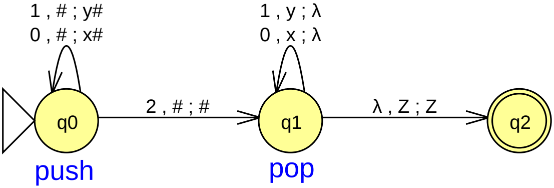
Intuição:

Vamos simular o comportamento do AP para a entrada $w = 1002001$



Autômatos com Pilha (AP)

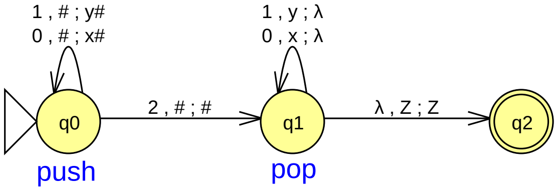
$$L = \{w2w^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$$



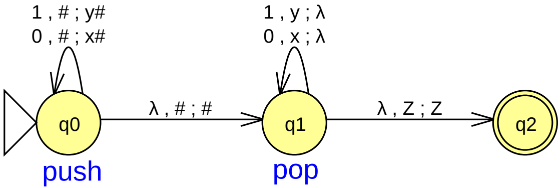
$$L = \{ww^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$$

Autômatos com Pilha (AP)

$$L = \{w2w^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$$



$$L = \{ww^R | w \in \{0,1\}^* \text{ e } |ww^R| \text{ é par } \}$$

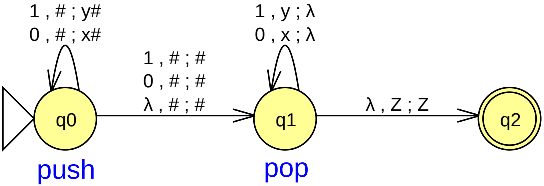


Autômatos com Pilha (AP)

$$L = \{w | w \in \{0,1\}^* \text{ e } w = w^R\}$$

Autômatos com Pilha (AP)

$$L = \{w | w \in \{0,1\}^* \text{ e } w = w^R\}$$



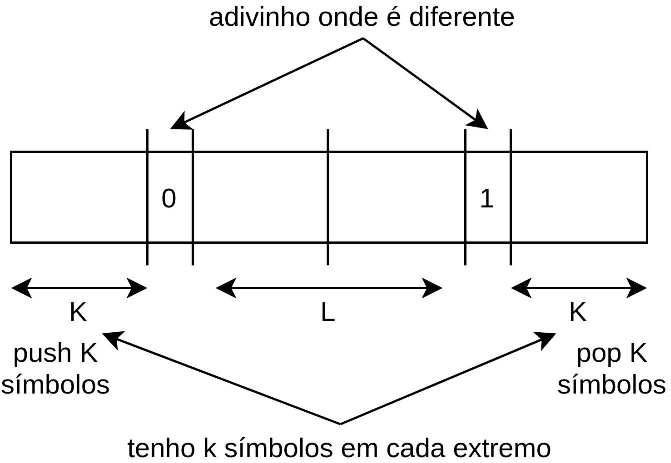
Desafios

Tente construir autômatos com pilha para as seguintes linguagens:

- $L = \{\overline{ww^R} | w \in \{0,1\}^*\}$
- $L = \{\overline{ww} | w \in \{0,1\}^*\}$
- $L = \{ww | w \in \{0,1\}^*\}$
- $L = \{w2w | w \in \{0,1\}^*\}$

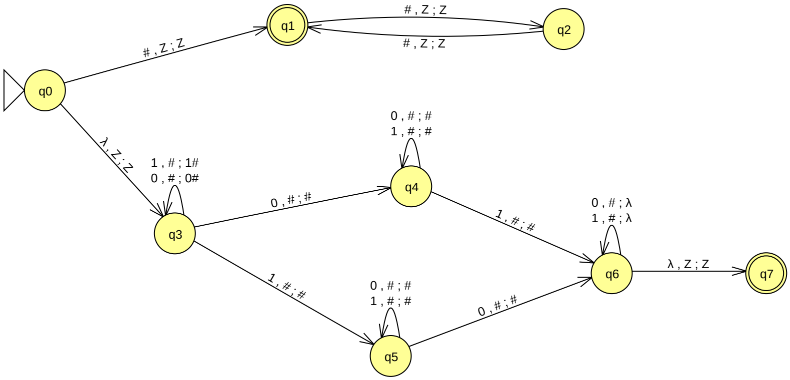
Autômatos com Pilha (AP)

$L = \{\overline{ww^R} | w \in \{0,1\}^*\}$



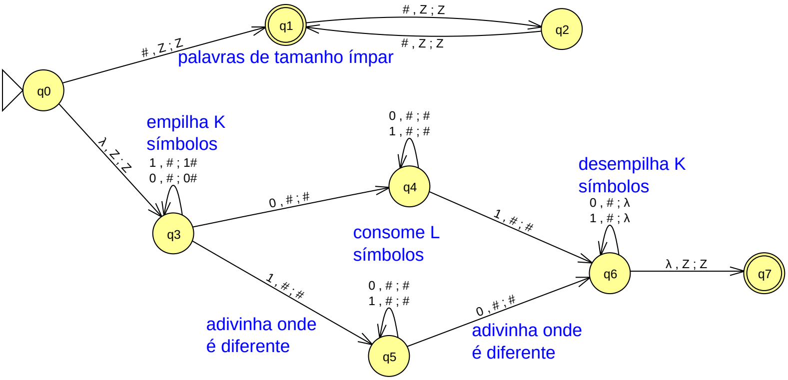
Autômatos com Pilha (AP)

$$L = \{\overline{ww^R} | w \in \{0,1\}^*\}$$



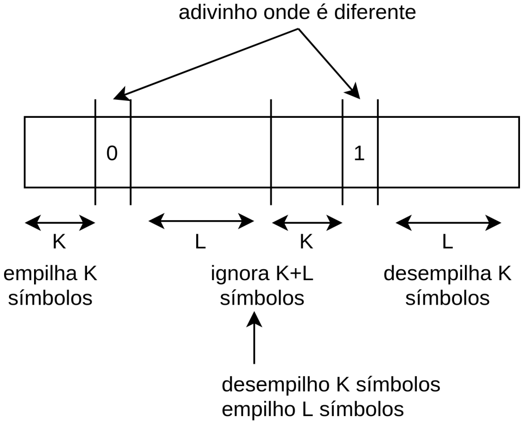
Autômatos com Pilha (AP)

$$L = \{\overline{ww^R} | w \in \{0,1\}^*\}$$



Autômatos com Pilha (AP)

$L = \{\overline{ww} | w \in \{0, 1\}^*\}$



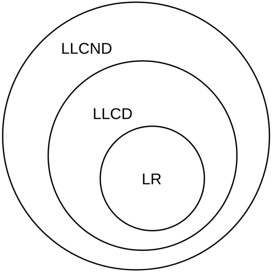
A sequência de operações deve ser: push K, pop K, push L, pop L

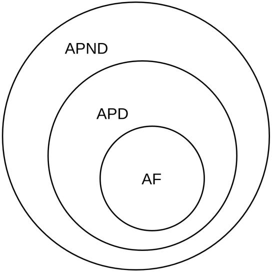
Autômatos com Pilha (AP)

Um autômato com pilha determinístico (APD) reconhece todas as linguagens regulares (LR), mas nem todas as linguagens livres de contexto (LLC), apenas as determinísticas (LLCD)

Um autômato com pilha não determinístico (APND) reconhece todas as linguagens reconhecidas por um APD e também todas as LLC não determinísticas (LLCND)

Os APND são os reconhecedores da classe de linguagens livres de contexto





Autômatos com Pilha (AP)

- APDs são muito utilizados na confecção de parsers
- Qualquer LLC pode ser reconhecida por um AP com um único estado
- Um AP com duas pilhas tem o mesmo poder computacional que uma Máquina de Turing e também é uma máquina universal, isto é, se há um algoritmo para resolver um problema, então ele pode ser expresso por um AP com duas pilhas

Tente construir um AP para a linguagem abaixo:

- $L = \{w | w \in \{(,)\}^+$ e w possui os parênteses balanceados $\}$
  - $L = \{(), ()()(), (()()())(), ...\}$

Como mostrar que uma linguagem é livre de contexto?

Como mostrar que uma linguagem é livre de contexto?

- Mostrar que ela é regular
- Construir um autômato com pilha, gramática livre de contexto

Ling. Não Livres de Contexto - Lema do Bombeamento

Como mostrar que uma linguagem é livre de contexto?

- Mostrar que ela é regular
- Construir um autômato com pilha, gramática livre de contexto

Como mostrar que uma linguagem não é livre de contexto?

Ling. Não Livres de Contexto - Lema do Bombeamento

Como mostrar que uma linguagem é livre de contexto?

- Mostrar que ela é regular
- Construir um autômato com pilha, gramática livre de contexto

Como mostrar que uma linguagem não é livre de contexto?

- Usar propriedades de fechamento
- Usar o lema do bombeamento (pumping lemma)

Ling. Não Livres de Contexto - Lema do Bombeamento

Se L é uma LLC, então existe uma constante $n \geq 0$ tal que para toda palavra $w \in L$, com $|w| \geq n$, então existe uma forma de dividir $w = uvxyz$ satisfazendo as condições:

1. $|vxy| \leq n$
2. $|vy| \geq 1$
3. $uv^ixy^iz \in L$, para todo $i \geq 0$

Ling. Não Livres de Contexto - Lema do Bombeamento

Se L é uma LLC, então existe uma constante $n \geq 0$ tal que para toda palavra $w \in L$, com $|w| \geq n$, então existe uma forma de dividir $w = uvxyz$ satisfazendo as condições:

1. $|vxy| \leq n$
2. $|vy| \geq 1$
3. $uv^ixy^iz \in L$, para todo $i \geq 0$

É importante ressaltar que o fato de uma linguagem satisfazer o lema do bombeamento não implica que ela é uma linguagem livre de contexto, mas se ela é uma linguagem livre de contexto, certamente ela satisfaz o lema do bombeamento

$L = \{w | w \in \{a, b, c\}^* \text{ e } w = a^n b^n c^n \text{ e } n \geq 1\}$

Vamos mostrar que L não é uma LLC.

$L = \{w | w \in \{a, b, c\}^* \text{ e } w = a^n b^n c^n \text{ e } n \geq 1\}$

Vamos mostrar que L não é uma LLC.

Prova: por contradição

$L = \{w | w \in \{a, b, c\}^* \text{ e } w = a^n b^n c^n \text{ e } n \geq 1\}$

Vamos mostrar que L não é uma LLC.

Prova: por contradição

- Suponha que L seja uma LLC e seja k uma constante inteira qualquer.

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{w | w \in \{a, b, c\}^*$ e $w = a^n b^n c^n$ e $n \geq 1\}$

Vamos mostrar que L não é uma LLC.

Prova: por contradição

- Suponha que L seja uma LLC e seja k uma constante inteira qualquer.
- Considere agora a palavra $w = a^k b^k c^k$ e veja que $w \in L$ e $|w| \geq k$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{w | w \in \{a, b, c\}^* \text{ e } w = a^n b^n c^n \text{ e } n \geq 1\}$

Vamos mostrar que L não é uma LLC.

Prova: por contradição

- Suponha que L seja uma LLC e seja k uma constante inteira qualquer.
- Considere agora a palavra $w = a^k b^k c^k$ e veja que $w \in L$ e $|w| \geq k$
- Pelo lema do bombeamento, poderíamos decompor a palavra $w = uvxyz$, com $|vxy| \leq k$ e $|vy| \geq 1$

$L = \{w | w \in \{a, b, c\}^* \text{ e } w = a^n b^n c^n \text{ e } n \geq 1\}$

Vamos mostrar que L não é uma LLC.

Prova: por contradição

- Suponha que L seja uma LLC e seja k uma constante inteira qualquer.
- Considere agora a palavra $w = a^k b^k c^k$ e veja que $w \in L$ e $|w| \geq k$
- Pelo lema do bombeamento, poderíamos decompor a palavra $w = uvxyz$, com $|vxy| \leq k$ e $|vy| \geq 1$
- Teríamos assim que $uv^i xy^i z \in L$ para todo $i \geq 0$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{w | w \in \{a, b, c\}^* \text{ e } w = a^n b^n c^n \text{ e } n \geq 1\}$

Vamos mostrar que L não é uma LLC.

Prova: por contradição

- Suponha que L seja uma LLC e seja k uma constante inteira qualquer.
- Considere agora a palavra $w = a^k b^k c^k$ e veja que $w \in L$ e $|w| \geq k$
- Pelo lema do bombeamento, poderíamos decompor a palavra $w = uvxyz$, com $|vxy| \leq k$ e $|vy| \geq 1$
- Teríamos assim que $uv^i xy^i z \in L$ para todo $i \geq 0$
- Vamos ver agora as formas de quebrar a palavra w

$L = \{w | w \in \{a, b, c\}^* \text{ e } w = a^n b^n c^n \text{ e } n \geq 1\}$

Vamos mostrar que L não é uma LLC.

Prova: por contradição

- **Caso 1**: como $|vxy| \leq k$, temos que vxy poderia conter apenas um dos símbolos (ou só a, ou só b, ou só c)

$L = \{w | w \in \{a, b, c\}^* \text{ e } w = a^n b^n c^n \text{ e } n \geq 1\}$

Vamos mostrar que L não é uma LLC.

Prova: por contradição

- **Caso 1**: como $|vxy| \leq k$, temos que vxy poderia conter apenas um dos símbolos (ou só a, ou só b, ou só c)
- Assim, note que se $i = 0$ temos que $uv^0xy^0z \notin L$, pois certamente terá um número diferente de a, b e c

$L = \{w | w \in \{a, b, c\}^* \text{ e } w = a^n b^n c^n \text{ e } n \geq 1\}$

Vamos mostrar que L não é uma LLC.

Prova: por contradição

- **Caso 1**: como $|vxy| \leq k$, temos que vxy poderia conter apenas um dos símbolos (ou só a, ou só b, ou só c)
- Assim, note que se $i = 0$ temos que $uv^0xy^0z \notin L$, pois certamente terá um número diferente de a, b e c
- Note que estaremos removendo ao menos um símbolo da palavra, pois ou v ou y certamente não é vazio

$L = \{w | w \in \{a, b, c\}^* \text{ e } w = a^n b^n c^n \text{ e } n \geq 1\}$

Vamos mostrar que L não é uma LLC.

Prova: por contradição

- **Caso 2**: como $|vxy| \leq k$, temos que vxy poderia conter no máximo dois símbolos diferentes (ou só a e b, ou só b e c)

$L = \{w | w \in \{a, b, c\}^* \text{ e } w = a^n b^n c^n \text{ e } n \geq 1\}$

Vamos mostrar que L não é uma LLC.

Prova: por contradição

- **Caso 2**: como $|vxy| \leq k$, temos que vxy poderia conter no máximo dois símbolos diferentes (ou só a e b, ou só b e c)
- Desta forma, note que novamente se $i = 0$ temos que $uv^0xy^0z \notin L$, pois certamente terá um número diferente de a, b e c

$L = \{w | w \in \{a, b, c\}^* \text{ e } w = a^n b^n c^n \text{ e } n \geq 1\}$

Vamos mostrar que L não é uma LLC.

Prova: por contradição

- **Caso 2**: como $|vxy| \leq k$, temos que vxy poderia conter no máximo dois símbolos diferentes (ou só a e b, ou só b e c)
- Desta forma, note que novamente se $i = 0$ temos que $uv^0xy^0z \notin L$, pois certamente terá um número diferente de a, b e c
- Note que estaremos removendo novamente ao menos um símbolo da palavra, pois ou v ou y certamente não é vazio

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{w | w \in \{a, b, c\}^* \text{ e } w = a^n b^n c^n \text{ e } n \geq 1\}$

Vamos mostrar que L não é uma LLC.

Prova: por contradição

- Portanto, podemos concluir que não há outras formas de dividir a palavra w respeitando as restrições do lema do bombeamento

$L = \{w | w \in \{a, b, c\}^* \text{ e } w = a^n b^n c^n \text{ e } n \geq 1\}$

Vamos mostrar que L não é uma LLC.

Prova: por contradição

- Portanto, podemos concluir que não há outras formas de dividir a palavra w respeitando as restrições do lema do bombeamento
- Todas as formas de dividir a palavra w levaram a construção de uma palavra, ao escolher i, que não pertence a L

$L = \{w | w \in \{a, b, c\}^* \text{ e } w = a^n b^n c^n \text{ e } n \geq 1\}$

Vamos mostrar que L não é uma LLC.

Prova: por contradição

- Portanto, podemos concluir que não há outras formas de dividir a palavra w respeitando as restrições do lema do bombeamento
- Todas as formas de dividir a palavra w levaram a construção de uma palavra, ao escolher i, que não pertence a L
- Assim, L não é uma LLC

Paradigma do adversário.

Vamos inicialmente reescrever o lema do bombeamento em expressões lógicas para o caso de L ser uma linguagem livre de contexto.

1. $\exists n$, onde n é uma constante inteira qualquer
2. $\forall w \in L$, onde $|w| \geq n$
3. $\exists u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
4. $\forall i \geq 0$, $uv^ixy^iz \in L$

Paradigma do adversário.

Vamos agora reescrever o paradigma do adversário considerando L uma linguagem que não seja uma linguagem livre de contexto.

1. $\forall n$, onde n é uma constante inteira qualquer
2. $\exists w \in L$, onde $|w| \geq n$
3. $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
4. $\exists i \geq 0$, $uv^i xy^i z \notin L$

Ling. Não Livres de Contexto - Lema do Bombeamento

Paradigma do adversário.

Vamos agora reescrever o paradigma do adversário considerando L uma linguagem que não seja uma linguagem livre de contexto.

1. $\forall n$, onde n é uma constante inteira qualquer
2. $\exists w \in L$, onde $|w| \geq n$
3. $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
4. $\exists i \geq 0$, $uv^ixy^iz \notin L$

Durante o processo de prova utilizando o paradigma do adversário, faremos uma espécie de "jogo", onde o adversário escolhe sempre $\forall$ e o provador (você) escolhe $\exists$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

Vamos mostrar que L não é livre de contexto utilizando o paradigma do adversário

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0, 1\}^*\}$

Vamos mostrar que L não é livre de contexto utilizando o paradigma do adversário

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

Vamos mostrar que L não é livre de contexto utilizando o paradigma do adversário

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
    - $n = k$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

Vamos mostrar que L não é livre de contexto utilizando o paradigma do adversário

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = k$
2. Provador: $\exists w \in L$, onde $|w| \geq k$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

Vamos mostrar que L não é livre de contexto utilizando o paradigma do adversário

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = k$
2. Provador: $\exists w \in L$, onde $|w| \geq k$
   - $w = 0^k10^k1$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

Vamos mostrar que L não é livre de contexto utilizando o paradigma do adversário

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = k$
2. Provador: $\exists w \in L$, onde $|w| \geq k$
   - $w = 0^k 1 0^k 1$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

Vamos mostrar que L não é livre de contexto utilizando o paradigma do adversário

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = k$
2. Provador: $\exists w \in L$, onde $|w| \geq k$
   - $w = 0^k 1 0^k 1$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
   - $u = 0^{k-1}$, $v = 0$, $x = 1$, $y = 0$, $z = 0^{k-1}1$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

Vamos mostrar que L não é livre de contexto utilizando o paradigma do adversário

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
    - $n = k$
2. Provador: $\exists w \in L$, onde $|w| \geq k$
    - $w = 0^k 1 0^k 1$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
    - $u = 0^{k-1}$, $v = 0$, $x = 1$, $y = 0$, $z = 0^{k-1}1$
4. Provador: $\exists i \geq 0$, $uv^ixy^iz \notin L$
    - Note que qualquer i que o provador tentar escolher, tem-se que v e y serão repetidos zero ou mais vezes, mas a palavra resultante sempre obedecerá o formato ww.

$L = \{ww | w \in \{0, 1\}^*\}$

Vamos mostrar que L não é livre de contexto utilizando o paradigma do adversário

...

Por fim, o adversário venceu o jogo, o que significa que as escolhas feitas pelo provador foram ruins ou a linguagem é realmente uma linguagem livre de contexto

$L = \{ww | w \in \{0,1\}^*\}$

Vamos tentar novamente.

$L = \{ww | w \in \{0,1\}^*\}$

Vamos tentar novamente.

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer

$L = \{ww | w \in \{0,1\}^*\}$

Vamos tentar novamente.

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
    - $n = k$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

Vamos tentar novamente.

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = k$
2. Provador: $\exists w \in L$, onde $|w| \geq k$

$L = \{ww | w \in \{0,1\}^*\}$

Vamos tentar novamente.

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = k$
2. Provador: $\exists w \in L$, onde $|w| \geq k$
   - $w = 0^k 1^k 0^k 1^k$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0, 1\}^*\}$

Vamos tentar novamente.

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = k$
2. Provador: $\exists w \in L$, onde $|w| \geq k$
   - $w = 0^k 1^k 0^k 1^k$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$

$L = \{ww | w \in \{0,1\}^*\}$

Vamos tentar novamente.

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = k$
2. Provador: $\exists w \in L$, onde $|w| \geq k$
   - $w = 0^k 1^k 0^k 1^k$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
   - Há dois casos que conseguem englobar todas as formas que o adversário pode fazer a quebra de w

$L = \{ww | w \in \{0,1\}^*\}$

Vamos tentar novamente.

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = k$
2. Provador: $\exists w \in L$, onde $|w| \geq k$
   - $w = 0^k 1^k 0^k 1^k$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
   - Há dois casos que conseguem englobar todas as formas que o adversário pode fazer a quebra de w
   - Caso 1: vxy pode possuir apenas um único símbolo (ou tudo 0 ou tudo 1)

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

Vamos tentar novamente.

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = k$
2. Provador: $\exists w \in L$, onde $|w| \geq k$
   - $w = 0^k 1^k 0^k 1^k$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
   - Há dois casos que conseguem englobar todas as formas que o adversário pode fazer a quebra de w
   - Caso 1: vxy pode possuir apenas um único símbolo (ou tudo 0 ou tudo 1)
   - Caso 2: vxy pode possuir dois símbolos diferentes (0 ou 1)

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

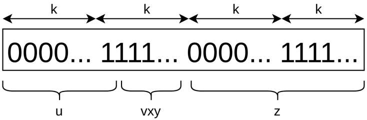
Vamos tentar novamente.

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = k$
2. Provador: $\exists w \in L$, onde $|w| \geq k$
   - $w = 0^k 1^k 0^k 1^k$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
   - Há dois casos que conseguem englobar todas as formas que o adversário pode fazer a quebra de w
   - Caso 1: vxy pode possuir apenas um único símbolo (ou tudo 0 ou tudo 1)
   - Caso 2: vxy pode possuir dois símbolos diferentes (0 ou 1)
   - Vamos analisar cada um dos dois casos

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

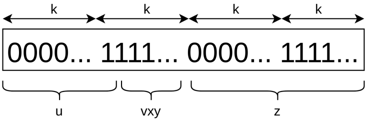
1. Adversário: $n = k$
2. Provador: $w = 0^k 1^k 0^k 1^k$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
   - Caso 1: vxy pode possuir apenas um único símbolo (ou tudo 0 ou tudo 1)



Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

1. Adversário: $n = k$
2. Provador: $w = 0^k 1^k 0^k 1^k$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
   - Caso 1: vxy pode possuir apenas um único símbolo (ou tudo 0 ou tudo 1)

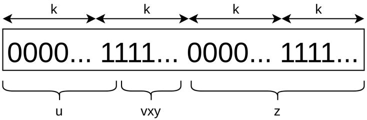


4. Provador: $\exists i \geq 0$, $uv^ixy^iz \notin L$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

1. Adversário: $n = k$
2. Provador: $w = 0^k 1^k 0^k 1^k$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
   - Caso 1: vxy pode possuir apenas um único símbolo (ou tudo 0 ou tudo 1)

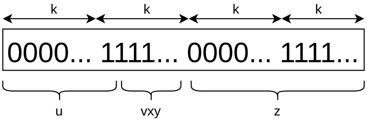


4. Provador: $\exists i \geq 0$, $uv^i xy^i z \notin L$
   - Note que se $i = 0$ temos que ao menos um símbolo 0 ou um símbolo 1 será removido da palavra.

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

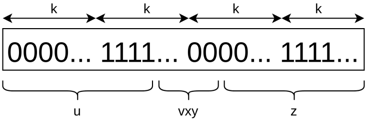
1. Adversário: $n = k$
2. Provador: $w = 0^k 1^k 0^k 1^k$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
   - Caso 1: vxy pode possuir apenas um único símbolo (ou tudo 0 ou tudo 1)



4. Provador: $\exists i \geq 0$, $uv^ixy^iz \notin L$
   - Note que se $i = 0$ temos que ao menos um símbolo 0 ou um símbolo 1 será removido da palavra.
   - Assim, a primeira parte da palavra será certamente diferente da segunda parte da palavra.

$L = \{ww | w \in \{0,1\}^*\}$

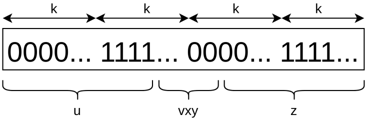
1. Adversário: $n = k$
2. Provador: $w = 0^k 1^k 0^k 1^k$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
   - Caso 2: vxy pode possuir dois símbolos diferentes (0 ou 1)



Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

1. Adversário: $n = k$
2. Provador: $w = 0^k 1^k 0^k 1^k$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
   - Caso 2: vxy pode possuir dois símbolos diferentes (0 ou 1)

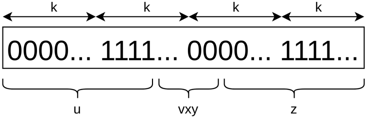


4. Provador: $\exists i \geq 0$, $uv^i xy^i z \notin L$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

1. Adversário: $n = k$
2. Provador: $w = 0^k 1^k 0^k 1^k$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
   - Caso 2: vxy pode possuir dois símbolos diferentes (0 ou 1)

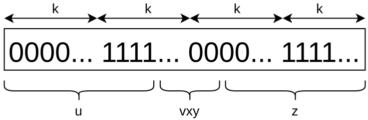


4. Provador: $\exists i \geq 0$, $uv^i xy^i z \notin L$
   - Note que novamente se $i = 0$ temos que ao menos um símbolo 0 ou um símbolo 1 será removido da palavra.

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{ww | w \in \{0,1\}^*\}$

1. Adversário: $n = k$
2. Provador: $w = 0^k 1^k 0^k 1^k$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
   - Caso 2: vxy pode possuir dois símbolos diferentes (0 ou 1)



4. Provador: $\exists i \geq 0$, $uv^ixy^iz \notin L$
   - Note que novamente se $i = 0$ temos que ao menos um símbolo 0 ou um símbolo 1 será removido da palavra.
   - Assim, a primeira parte da palavra será certamente diferente da segunda parte da palavra.

$L = \{ww | w \in \{0,1\}^*\}$

Por fim, o provador venceu o jogo, pois ao escolher $i = 0$ e sabendo que v ou y contém ao menos um 0 ou 1 estaremos removendo um símbolo 0 ou 1 da palavra w. Veja que ao fazer isso, a primeira parte da palavra sempre será diferente da segunda parte e portanto a palavra resultante não pertencerá à L. Assim a linguagem não é uma linguagem livre de contexto.

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{w | w \in \{0\}^* \text{ e } w = 0^n \text{ e } n \text{ é primo } \}$

Vamos mostrar que L não é livre de contexto

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{w | w \in \{0\}^* \text{ e } w = 0^n \text{ e } n \text{ é primo } \}$

Vamos mostrar que L não é livre de contexto

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer

$L = \{w | w \in \{0\}^*$ e $w = 0^n$ e n é primo $\}$

Vamos mostrar que L não é livre de contexto

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = t$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{w | w \in \{0\}^* \text{ e } w = 0^n \text{ e } n \text{ é primo } \}$

Vamos mostrar que L não é livre de contexto

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = t$
2. Provador: $\exists w \in L$, onde $|w| \geq t$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{w | w \in \{0\}^* \text{ e } w = 0^n \text{ e } n \text{ é primo } \}$

Vamos mostrar que L não é livre de contexto

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
    - $n = t$
2. Provador: $\exists w \in L$, onde $|w| \geq t$
    - $w = 0^p$, com p sendo primo e $|0^p| \geq t$

$L = \{w | w \in \{0\}^* \text{ e } w = 0^n \text{ e } n \text{ é primo } \}$

Vamos mostrar que L não é livre de contexto

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = t$
2. Provador: $\exists w \in L$, onde $|w| \geq t$
   - $w = 0^p$, com p sendo primo e $|0^p| \geq t$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$

$L = \{w | w \in \{0\}^* \text{ e } w = 0^n \text{ e } n \text{ é primo } \}$

Vamos mostrar que L não é livre de contexto

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
    - $n = t$
2. Provador: $\exists w \in L$, onde $|w| \geq t$
    - $w = 0^p$, com p sendo primo e $|0^p| \geq t$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
    - Note que para qualquer quebra que o adversário fizer, temos que $v = 0^k$ e $y = 0^m$, com $k + m \geq 1$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{w | w \in \{0\}^* \text{ e } w = 0^n \text{ e } n \text{ é primo } \}$

Vamos mostrar que L não é livre de contexto

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = t$
2. Provador: $\exists w \in L$, onde $|w| \geq t$
   - $w = 0^p$, com p sendo primo e $|0^p| \geq t$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
   - Note que para qualquer quebra que o adversário fizer, temos que $v = 0^k$ e $y = 0^m$, com $k + m \geq 1$
4. Provador: $\exists i \geq 0$, $uv^ixy^iz \notin L$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{w | w \in \{0\}^* \text{ e } w = 0^n \text{ e } n \text{ é primo } \}$

Vamos mostrar que L não é livre de contexto

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = t$
2. Provador: $\exists w \in L$, onde $|w| \geq t$
   - $w = 0^p$, com p sendo primo e $|0^p| \geq t$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
   - Note que para qualquer quebra que o adversário fizer, temos que $v = 0^k$ e $y = 0^m$, com $k + m \geq 1$
4. Provador: $\exists i \geq 0$, $uv^ixy^iz \notin L$
   - Note que para $i = 1 + p$ temos que $|uvxyz| = p$, $|v^p| = kp$, $|y^p| = mp$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{w | w \in \{0\}^* \text{ e } w = 0^n \text{ e } n \text{ é primo } \}$

Vamos mostrar que L não é livre de contexto

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
    - $n = t$
2. Provador: $\exists w \in L$, onde $|w| \geq t$
    - $w = 0^p$, com p sendo primo e $|0^p| \geq t$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
    - Note que para qualquer quebra que o adversário fizer, temos que $v = 0^k$ e $y = 0^m$, com $k + m \geq 1$
4. Provador: $\exists i \geq 0$, $uv^ixy^iz \notin L$
    - Note que para $i = 1 + p$ temos que $|uvxyz| = p$, $|v^p| = kp$, $|y^p| = mp$
    - Ou seja, $|uv^{1+p}xy^{1+p}z| = p + kp + mp = p(1 + k + m)$

Ling. Não Livres de Contexto - Lema do Bombeamento

$L = \{w | w \in \{0\}^* \text{ e } w = 0^n \text{ e } n \text{ é primo } \}$

Vamos mostrar que L não é livre de contexto

1. Adversário: $\forall n$, onde n é uma constante inteira qualquer
   - $n = t$
2. Provador: $\exists w \in L$, onde $|w| \geq t$
   - $w = 0^p$, com p sendo primo e $|0^p| \geq t$
3. Adversário: $\forall u, v, x, y, z$, tal que $w = uvxyz$, $|vxy| \leq n$, $|vy| \geq 1$
   - Note que para qualquer quebra que o adversário fizer, temos que $v = 0^k$ e $y = 0^m$, com $k + m \geq 1$
4. Provador: $\exists i \geq 0$, $uv^ixy^iz \notin L$
   - Note que para $i = 1 + p$ temos que $|uvxyz| = p$, $|v^p| = kp$, $|y^p| = mp$
   - Ou seja, $|uv^{1+p}xy^{1+p}z| = p + kp + mp = p(1 + k + m)$
   - Mas, $0^{p(1+k+m)} \notin L$, pois $p(1 + k + m)$ é divisível por p e $(1 + k + m)$ e portanto L não é livre de contexto

Conclusão

- Linguagens livres de contexto
- Gramáticas livres de contexto
- Forma normal de Chomsky
- Autômatos com pilha
- Linguagens não livres de contexto e lema do bombeamento

Material de apoio

- Livro Sipser, Capítulo 2
- Livro Hopcroft, Capítulos 5,6,7