Teoria da Computação

Tópicos II

Thiago Silva Vilela

Linguagens Formais

- Uma linguagem formal:
  - tem uma sintaxe bem definida, de tal forma que, dada uma sentença, é sempre possível saber se ela pertence ou não à linguagem;
  - tem uma semântica precisa, de tal forma que não contém sentenças sem significado ou ambíguas.
- Exemplos de linguagens formais: Java, C, HTML, etc.

Linguagens Formais

- Toda linguagem tem um alfabeto associado.
- **Alfabeto**: conjunto finito não vazio de elementos. São geralmene chamados de *símbolos*.
- Uma **palavra** sobre um alfabeto Σ é uma sequência finita de símbolos de Σ.
- O **tamanho** de uma palavra x, $|x|$, é o número de símbolos que a compõem.
  - geralmente existe uma palavra vazia, formada de zero símbolos, designada por λ.

Linguagens Formais

- Exemplo:
  - $\Sigma = \{0, 1\}$;
  - símbolos: 0 e 1;
  - palavras sobre Σ: λ, 0, 1, 00, 01, 10, 000, etc;
  - esse alfabeto pode representar todos os números inteiros (usando codificação na base 2);
  - λ é a palavra vazia,

Linguagens Formais

- A notação a^n é utilizada para designar a palavra constituída de n a's em sequência.

- Por exemplo, no alfabeto $\Sigma = \{0, 1\}$:
  - $1^0 = \lambda$
  - $0^4 = 0000$
  - $1^3 0 1^2 = 111011$

- Uma **linguagem** sobre um alfabeto Σ é um conjunto de palavras sobre Σ.
  - Σ^*: conjunto de todas as palavras sobre Σ;
  - uma linguagem sobre Σ é qualquer subconjunto de Σ^*.

Linguagens Formais

- Exemplo:
  - Alfabeto: $\Sigma=\{0, 1\}$
  - Palavras sobre Σ: $\Sigma^* = \{\lambda, 0, 1, 00, 01, 10, 11, ...\}$
  - Linguagens sobre Σ:
    - $\{\lambda\}$.
    - $\{w \in \Sigma^* \mid 1 \leq |w| \leq 5\}$.
    - $\{0^n \mid n$ é um número primo$\}$.
    - Σ^*.

Linguagens Formais

- Frequentemente linguagens de interesse são infinitas.
- É comum utilizar a operação fecho de Kleene para especificá-las.
- Fecho de Kleene (*) pode ser definido como:
  - $\lambda \in L^*$;
  - se $x \in L^*$ e $y \in L$, então $xy \in L^*$.

Linguagens Formais

- Exemplos: algumas linguagens e seus fechos de Kleene
  - $\varnothing^* = \{\lambda\}$;
  - $\{\lambda\}^* = \{\lambda\}$;
  - $\{0\}^* = \{0^n \mid n \in N\}$;
  - $\{\lambda, 00, 11\}^* = \{\lambda, 00, 11, 0011, 1100, 1111, 000011, \ldots\}$

Linguagens Formais

- Exemplos de linguagens:
  - O conjunto das palavras que começam com 0:
    - $\{0\}\{0, 1\}^*$;
  - O conjunto das palavras que contêm 00 ou 11:
    - $\{0, 1\}^*\{00, 11\}\{0, 1\}^*$;
  - O conjunto das palavras que terminam com 0 seguido de um número ímpar de 1's consecutivos:
    - $\{0, 1\}^*\{01\}\{11\}^*$;
  - O conjunto das palavras com um prefixo de um ou mais 0's seguido (imediatamente) de um sufixo de 1's de mesmo tamanho:
    - $\{0^n1^n \mid n \geq 1\}$.

Gramáticas

- Formalismo projetado para a definição de linguagens.
- Gramáticas possuem uma ou mais regras.
- **Regra**: par ordenado (*u, v*), geralmente escrito da forma $u \rightarrow v$, onde *u* e *v* são palavras.
- As palavras podem conter dois tipos de símbolos:
  - Variáveis ou não terminais: auxiliares para geração das palavras da linguagem. Geralmente representadas por letras maiúsculas.
  - Terminais: alfabeto da linguagem definida. Geralmente representados por letras minúsculas.

Gramáticas

- A seguinte regra:

$$aAB \rightarrow baA$$

especifica que, dada uma palavra que tenha a subpalavra aAB, tal subpalavra pode ser substituída por baA.

- a e b são terminais.
- A e B são variáveis, ou não terminais.

- A partir da palavra aABBaAB, podemos derivar:
  - baABaAB, substituindo a primeira ocorrência de aAB.
  - aABBbaA, substituindo a segunda ocorrência de aAB.

Gramáticas

- Uma gramática consta de um conjunto de regras e da indicação de uma **variável de partida**.
  - Qualquer derivação deve começar com a palavra constituída apenas da variável de partida.
- Palavras de variáveis e/ou terminais geradas a partir da variável de partida são chamadas de **formas sentenciais**.
- Uma forma sentencial sem variáveis é denominada **sentença**.
- A linguagem definida (ou gerada) pela gramática é o conjunto de sentenças geradas pela gramática.
  - Chamamos de L(G) a linguagem gerada pela gramática G.

Gramáticas

- Exemplo: Gramática G com variável de partida P.

  $P \rightarrow aAbc$
  $A \rightarrow aAbC$
  $A \rightarrow \lambda$

  $Cb \rightarrow bC$

  $Cc \rightarrow cc$

- Sentenças de G?
  - abc, aabbcc, aaabbbccc, etc.

Gramáticas

- Uma mesma linguagem pode ser gerada por diferentes gramáticas.
- Duas gramáticas G e G' são **equivalentes** quando L(G) = L(G').

- Exemplo:

G:	G'
$S \rightarrow AB$	$S \rightarrow aS \mid ab$
$AB \rightarrow AAB$	
$A \rightarrow a$	
$B \rightarrow b$	

- G e G' são equivalentes. Geram sentenças da forma a^nb.

Autômatos Finitos

- Autômatos finitos (AFs) são um tipo de máquina de estados finitos do tipo reconhecedor de linguagens.
  - Ou seja, eles podem reconhecer as palavras de uma dada linguagem.

- Linguagens reconhecidas por máquinas de estados são conhecidas como **linguagens regulares**.

Autômatos Finitos

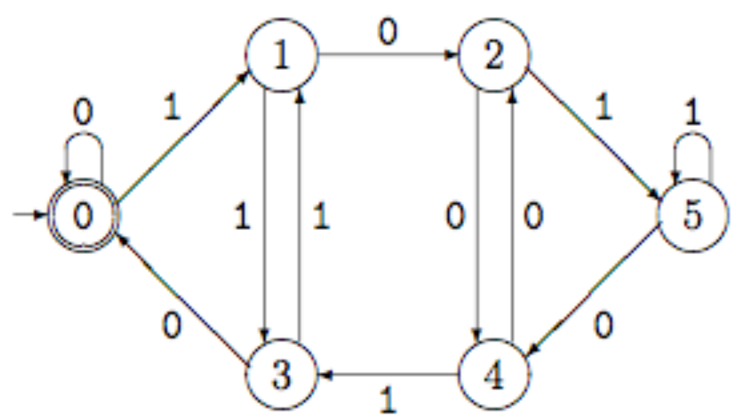
- Exemplo: usar uma máquina de estados para projetar uma máquina que, dada uma sequência de 0’s e 1’s, determine se o número representado por ela na base 2 seja divisível por 6 (ou seja, reconheça números divisíveis por 6).
  – devemos construir um AF que reconheça uma palavra na forma de um número binário, como 110;
  – a palavra só deve ser reconhecida quando for divisível por 6;
  – a máquina processa os dígitos da palavra um a um, da esquerda para a direita.

Autômatos Finitos

- Seja $n(x)$ o número representado pela palavra x:
  - $x0$ representa o número $2n(x)$;
  - $x1$ representa o número $2n(x)+1$;
- Sendo r o resto da divisão por 6 do número $n(x)$:
  - o resto da divisão por 6 do número $2n(x)$ é o resto da divisão por 6 de $2r$;
  - o resto da divisão por 6 do número $2n(x) + 1$ é o resto da divisão por 6 de $2r + 1$;

Autômatos Finitos

- Podemos fazer uma máquina de estados com 6 estados.
  - Cada um representa um resto de divisão (de 0 a 5).
  - O estado 0 é tanto o estado inicial quanto o final.



Autômatos Finitos Determinísticos

- É uma quíntupla (E, Σ, δ, i, F):
  - E é um conjunto finito de estados;
  - Σ é um alfabeto;
  - δ é uma função de transição, o conjunto de transições;
    - cada par (estado, símbolo) é mapeado para **somente um** estado.
  - i, um estado de E, é o estado inicial;
  - F, subconjunto de E, é o conjunto de estados finais.

Autômatos Finitos Determinísticos

- No exemplo mostrado anteriormente, temos:
  - $(E, \Sigma, \delta, i, F) = (\{0,1,2,3,4,5\}, \{0,1\}, \delta, 0, \{0\})$;
  - δ é dado por:

δ	**0**	**1**
0	0	1
1	2	3
2	4	5
3	0	1
4	2	3
5	4	5

Autômatos Finitos Determinísticos

- Para cada palavra existe um, e apenas um, estado que pode ser alcançado a partir do estado inicial.

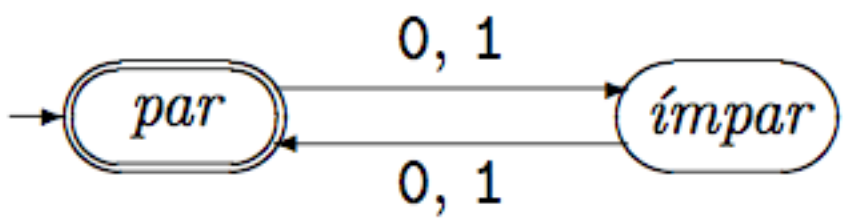
- Uma palavra é reconhecida (ou aceita) por um AFD se, após todo o processamento da palavra, um estado final do AFD foi alcançado.
  - Por isso AFDs são reconhecedores de linguagens.

Autômatos Finitos Determinísticos

- Exemplo: Fazer o diagrama de estados de um AFD M, tal que:
  - $L(M) = \{w \in \{0, 1\}^* \mid w$ tem um número par de símbolos$\}$.

Autômatos Finitos Determinísticos

- Exemplo: Fazer o diagrama de estados de um AFD M, tal que:
  – L(M) = {w ∈ {0, 1}* | w tem um número par de símbolos}.

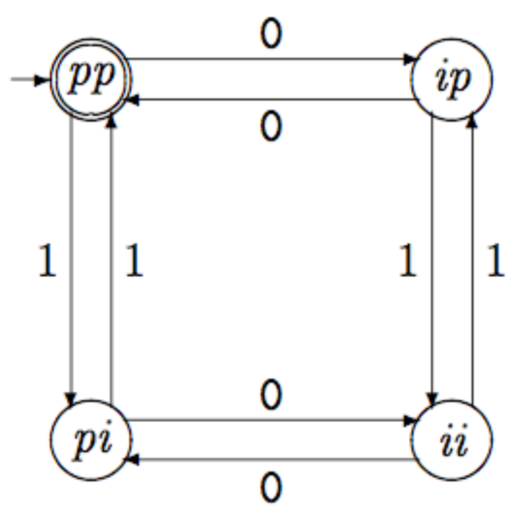


Autômatos Finitos Determinísticos

- Exemplo: Fazer o diagrama de estados de um AFD N, tal que:
  - L(N) = {w $\in$ {0, 1}* | w tem um número par de 0's e um número par de 1's}.

Autômatos Finitos Determinísticos

- Exemplo: Fazer o diagrama de estados de um AFD N, tal que:
  – L(N) = {w ∈ {0, 1}* | w tem um número par de 0's e um número par de 1's}.



Propriedades de Autômatos Finitos Determinísticos

- Se existem AFD's para duas linguagens, então existem também AFD's para a união e a interseção delas.

- Existe AFD para o complemento da linguagem aceita por qualquer AFD.

- Para toda linguagem finita existe um AFD que a reconhece.
  – Além disso, sempre é possível construir um AFD sem ciclos para essas linguagens.

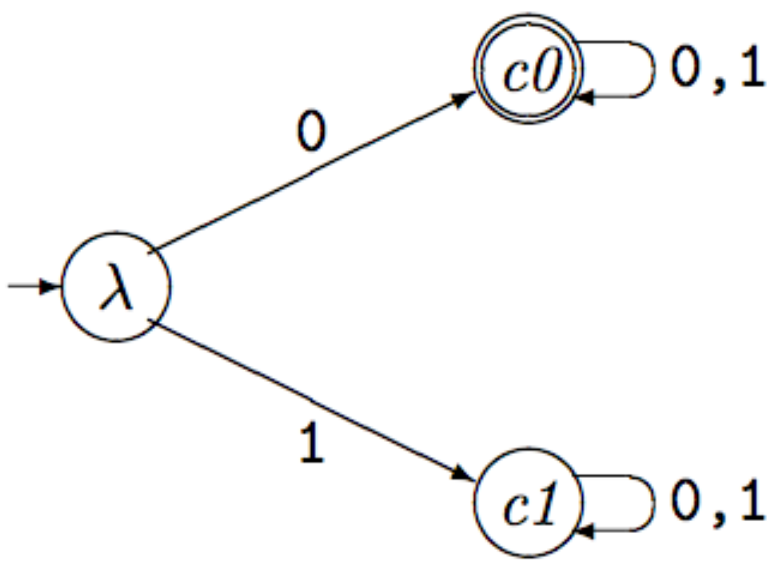
- Uma linguagem L é infinita se e somente se:
  – não existe um AFD que a reconhece; ou
  – o diagrama de estados de qualquer AFD que reconhece L tem ciclos.

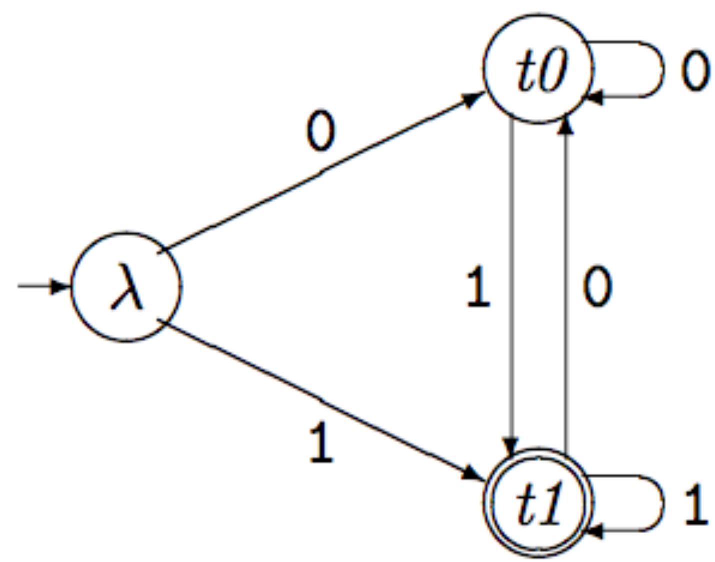
Propriedades de Autômatos Finitos Determinísticos

- Exemplo:
  - Faça um AFD que reconheça a linguagem $A_0 = \{0y \mid y \in \{0,1\}^*\}$.
  - Faça um AFD que reconheça a linguagem $A_1 = \{x1 \mid x \in \{0, 1\}^*\}$.
  - Faça um AFD que reconheça $A_0 \cup A_1$;

Propriedades de Autômatos Finitos Determinísticos

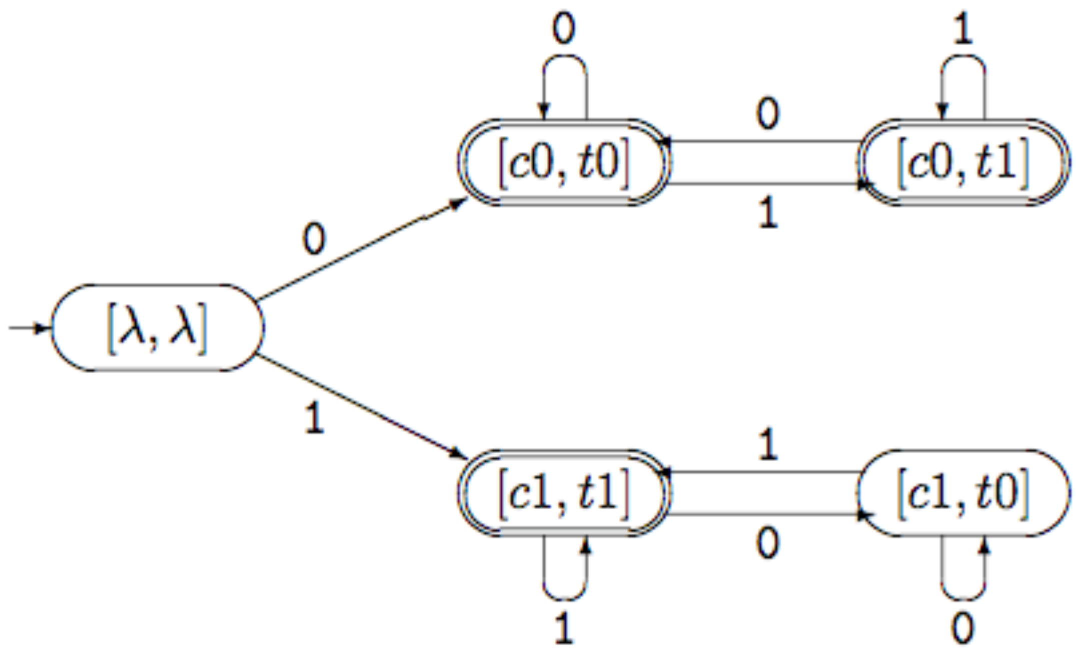
- A_0 e A_1:





Propriedades de Autômatos Finitos Determinísticos

- $A_0 \cup A_{1}$:



Autômatos Finitos não Determiníticos

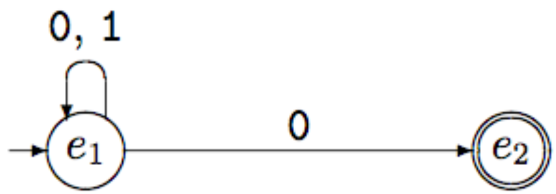
- Em AFDs, de cada par (estado, símbolo) só há transição para um estado.

- AFNs removem essa restrição dos AFDs.

- Em um AFN podem existir várias computações possíveis para a mesma palavra.

Autômatos Finitos não Determiníticos

- No seguinte AFN:



- O não determinismo pode ser visto pela indecisão no estado e_1.
  - existem duas transições possíveis sobre o símbolo 0.
  - para a palavra 10, por exemplo, existem diferentes computações possíveis.

Autômatos Finitos não Determiníticos

- Em que situações é considerado que um AFN reconhece uma palavra?
  - uma palavra é reconhecida se, e somente se, existe uma computação que a consome e termina em estado final.

- O AFN anterior, então, reconhece o conjunto das palavras de $\{0, 1\}^*$ que terminam em 0.

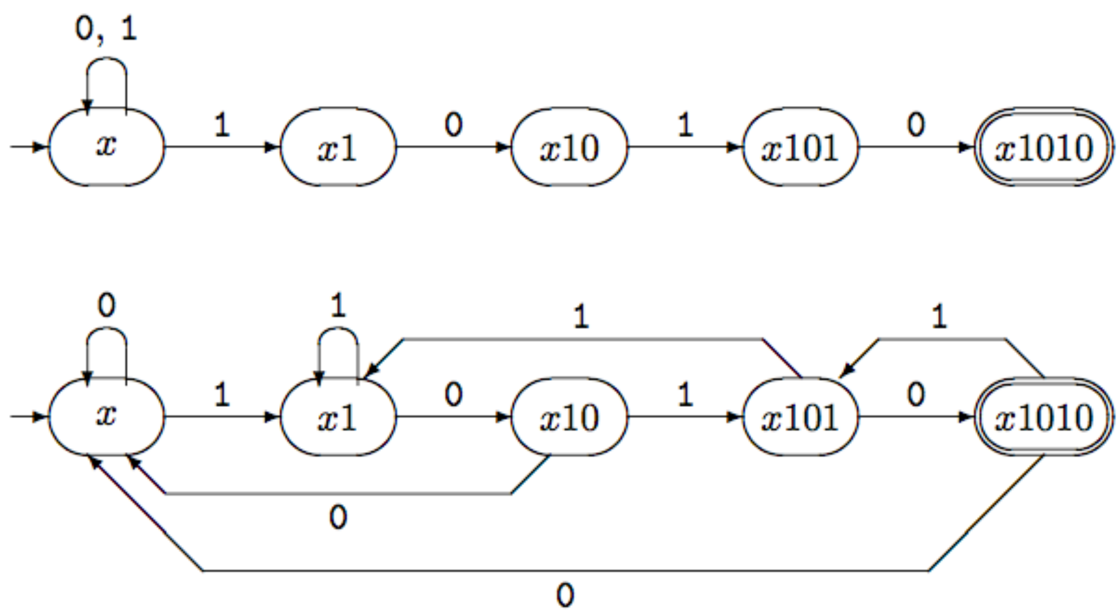
Autômatos Finitos não Determiníticos

- Para todo AFN existe um AFD equivalente.

- Por que AFNs são importantes?
  - várias vezes é mais fácil construir AFNs que AFDs;
  - AFN pode ser mais claro que um AFD equivalente;

Autômatos Finitos não Determiníticos

- Exemplo: construir diagramas de estados de um AFD e um AFN para a linguagem {0, 1}*{1010}.



Linguagens Regulares

- Linguagens regulares são aquelas que podem ser reconhecidas por AFNs ou AFDs.

- São uma classe importante das linguagens formais.
  - particularmente úteis no processo de análise sintática de compiladores.

- Uma das formas mais comuns de representar(ou definir) essas linguagens é através de gramáticas regulares.

Gramáticas Regulares

- Gramáticas regulares mostram como gerar todas, e apenas, as palavras de certa linguagem regular.

- Uma grámatica regular é uma gramática em que cada regra tem uma das formas:
  - $X \rightarrow a$
  - $X \rightarrow aY$
  - $X \rightarrow \lambda$

- Ou seja, gramáticas regulares não possuem símbolos terminais à esquerda.

- Além disso, cada regra da gramática tem somente um símbolo não terminal à direita.

Gramáticas Regulares

- Exemplo: Seja L = $\{w \in \{a, b, c\}^* \mid w$ não comtém abc$\}$. Escreva uma gramática regular que gere L. Precisamos de:
  - Conjunto de variáveis não terminais;
  - Alfabeto;
  - Regras;
  - Variável de partida.

Gramáticas Regulares

- Conjunto de variáveis não terminais: {A, B, C}; Alfabeto: {a, b, c}; Variável de partida: A.
- Regras:

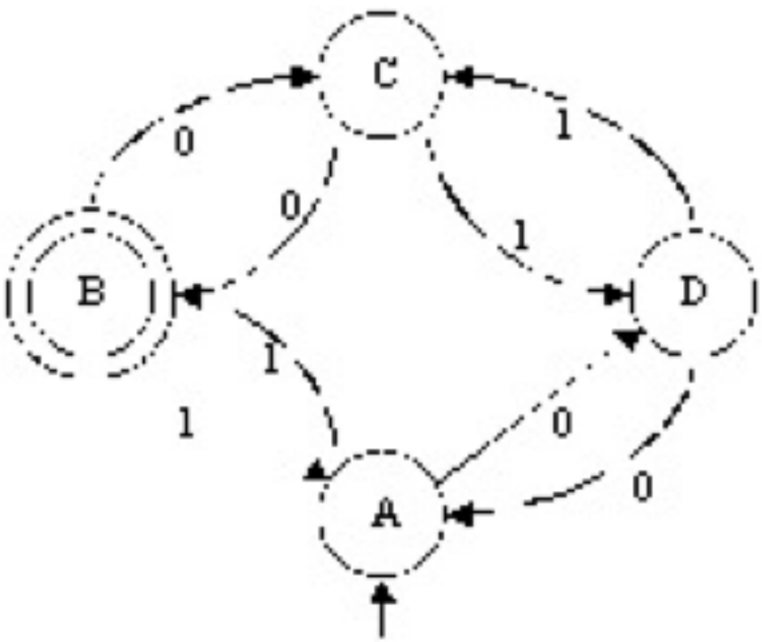
$$A \rightarrow aB|bA|cA|\lambda$$

$$B \rightarrow aB|bC|cA|\lambda$$

$$C \rightarrow aB|bA|\lambda$$

Exemplos de questões

Que cadeia é reconhecida pelo autômato representado pelo diagrama de estados?



a) 101010

b) 111011000

c) 11111000

d) 10100

e) 00110011

Exemplos de questões

Considera a gramática a seguir, em que S, A e B são símbolos não terminais, 0 e 1 são terminais e λ é a cadeia vazia.

$S \rightarrow 1S \mid 0A \mid \lambda$
$A \rightarrow 1S \mid 0B \mid \lambda$
$B \rightarrow 1S \mid \lambda$

A respeito dessa gramática analise as afirmações a seguir:

I. Nas cadeias geradas por essa gramática, o último símbolo é 1.
II. O número de zeros consecutivos nas cadeias geradas pela gramática é, no máximo, dois.
III. O número de uns em cada cadeia gerada pela gramática é maior que o número de zeros.
IV. Nas cadeias geradas por essa gramática,todos os uns estão à esquerda de todos os zeros.