

Universidade Federal de Santa Catarina / Centro Tecnológico
Departamento de Informática e de Estatística
INE5317/INE5421 - Linguagens Formais e Compiladores

Lista de Exercícios nº 1

0 – Você deve saber ... (não precisa entregar a resposta desta questão)

a) A diferença entre problemas decidíveis e problemas indecidíveis.

Se para qualquer entrada e saída pertencente ao domínio do problema o programa retorna uma resposta válida (true ou false), o problema é decidível, caso contrário ele é um problema indecidível

b) De que forma problemas computacionais em geral, podem ser vistos como problemas de Linguagem.

Todo problema computacional pode ser transformado em uma gramática, e portanto podem ser vistos como problemas de linguagem

c) A relação entre problemas decidíveis e problemas indecidíveis com a teoria das linguagens formais.

Problemas indecidíveis correspondem a gramáticas tipo 0

Problemas decidíveis correspondem a gramáticas tipo 1, 2 ou 3

d) Diferenças entre aspectos léxicos, sintáticos e semânticos de linguagens de programação.

Aspectos léxicos são aspectos relativos a quais símbolos foram utilizados na linguagem

Aspectos sintáticos são aspectos relativos a posição dos símbolos na linguagem

Aspectos semânticos são aspectos relativos ao sentido da linguagem

e) A estrutura geral de um compilador e as funções básicas de cada fase.

O compilador é dividido em uma fase de análise e uma fase de síntese

A fase de análise é composta por uma análise, léxica, sintática e semântica

A fase de síntese pode ter ou não uma fase de geração de código intermediário e depois ter fase de otimização de código, e geração de código

f) As formas de implementação de um compilador, e os critérios usados na escolha de uma delas para uma determinada implementação.

O compilador pode ser implementado em um ou mais passos dependendo da memória, processamento e tempo disponível para compilação do programa

g) A importância do uso de Código Intermediário na implementação de um compilador.

O código intermediário pode ser utilizado para vários fins, tais quais permitir a otimização de um código, ajudar na portabilidade permitindo transformar em um código que possa ser lido por outro

programa(inclusive um compilador) e outras

g) As técnicas de otimização de código mais comuns.

h) A definição formal de sentença, forma sentencial, gramática, linguagem, derivação e redução.

i) Mostrar que GSC são recursivas.

1 – Diferencie Linguagem finita, infinita e vazia e exemplifique cada caso usando GR.

2 - Construa uma gramática G |

- G não seja recursiva mas possua uma GR equivalente G1 | L(G1) seja Regular, infinita e não contenha a sentença vazia.
- G seja recursiva, L(G) não seja Livre de Contexto e L(G) seja finita e contenha a sentença vazia..

3 - Toda linguagem finita é uma L.L.C. ? é uma L. Regular ? Justifique sua resposta.

4 – a) A União e a Concatenação de duas Linguagens (L1 e L2) de um mesmo tipo, resultam sempre em Linguagens do mesmo tipo que L1 e L2? Justifique e exemplifique.

b) Toda linguagem LLC pode ser gerada por uma GLC formada apenas por produções da forma $A \rightarrow BC | a$ onde A, B e $C \in V_n$ e $a \in V_t$?

c) Dada uma GLC ou uma GR qualquer, existe algoritmo para determinar se G é vazia? Se sim, forneça esse algoritmo (em passos gerais); senão, justifique.

d) Dada uma GLC ou uma GR qualquer G, existe algoritmo para verificar se L(G) é ou não finita? Se sim, forneça esse algoritmo (em passos gerais); senão, justifique.

5 - Construa uma Gramática Regular G |

- $L(G) = \{ x | x \in (a, b, c)^+ \wedge \text{ todos os } \underline{b}'\text{s de } x \text{ sejam consecutivos} \}$
- $L(G) = \{ x | x \in a^n (b, c)^* \wedge n + \#c\text{'s é par} \wedge x \text{ não possui } \underline{b}'\text{s consecutivos} \}$
- $L(G) = \{ 0^n 1^m 2^k | n, m, k \geq 0, m \text{ seja par} \wedge n + m + k \text{ seja ímpar} \}$
- $L(G) = \{ x | x \in (a, b)^* \wedge \#a\text{'s seja par ou } \#b\text{'s seja divisível por 3} \}$

6 - Construa uma Gramática Livre de Contexto (GLC) G |

- $L(G) = \{ a^n b^m (c, d)^* | 0 \leq n \leq m \wedge \#c\text{'s} = 2 * \#d\text{'s} \}$
- $L(G) = \{ a^n b^m c^k d^p | n, m, k, p \geq 0, n \geq p \wedge k \geq m \}$
- $L(G) = \{ x | x \in (a, b)^* (c, d)^* \wedge \#a\text{'s} + \#b\text{'s} \neq \#c\text{'s} + \#d\text{'s} \}$
- $L(G) = \{ a^n b^m c^j | n, m, j \geq 0 \wedge n = m \vee m = j \}$

7 – Construa uma GLC que especifique:

a) a sintaxe da **declaração** de procedimentos (métodos) de uma linguagem de programação qualquer. OBS. Para simplificar, considere apenas parâmetros de tipos simples.

b) a sintaxe da **chamada** de procedimentos (métodos) de uma linguagem de programação qualquer.

c) a sintaxe de uma seqüência de estruturas de controle if-then-else e while-do (possivelmente aninhadas).

8 – Construa uma gramática G, do maior tipo possível |

- a) $L(G) = \{ x \mid x \in (a, b)^* \wedge \#a's \neq \#b's \}$
- b) $L(G) = \{ a^n b^m c^k \mid n, m, k \geq 0 \wedge m > n \wedge n > k \}$
- c) $L(G) = \{ x \mid x \in (a, b)^* \wedge x \text{ seja palíndromo} \}$
- d) $L(G) = \{ x \mid x \in a^n (b, c)^* \wedge n \text{ é par e } \#b's + \#c's \text{ é divisível por } 3 \}$

9 - Construa uma GSC (Gramática Sensível ao Contexto) G |

- $\alpha)$ $L(G) = \{ a^n b^n c^m \mid n \geq 0, m \geq 0 \wedge n \neq m \}$
- $\beta)$ $L(G) = \{ a^n b^m c^p d^q \mid n, m, p, q \geq 0 \wedge n > p \wedge q < m \}$
- $\chi)$ $L(G) = \{ x \mid x \in (a, b, c)^* \wedge \#a's \neq \#b's \neq \#c's \}$
- $\delta)$ $L(G) = \{ x \mid x \in (a, b, c)^* d^+ \wedge \#a's > \#c's \wedge \#b's = \#d's \}$

10 - Seja G a seguinte gramática:

- $S \rightarrow a S \mid S C \mid c S A \mid b$
- $A C \rightarrow C A \quad C A \rightarrow A C$
- $b C \rightarrow b c \quad b A \rightarrow b a$
- $a A \rightarrow a a \quad a C \rightarrow a c$
- $c A \rightarrow c a \quad c C \rightarrow c c$

Pede-se:

- a) Verifique se $x = acba$ e $y = abca$ pertencem a $L(G)$, usando o algoritmo apropriado;
- b) Determine $L(G)$;
- c) Construa, se possível, uma GLC $G_1 \mid L(G_1) = L(G)$
- d) Construa $G_2 \mid G_2$ seja do mesmo tipo de $G \wedge L(G_2) = L(G) \cup \{ \epsilon \}$

11 - Determine $L(G)$ onde G é dada por:

- a) $S \rightarrow 0 S 1 \mid 1 S 0 \mid 0 1 \mid 1 0$

- b) $S \rightarrow a S \mid b A \mid b \mid a C \mid b D$
 $A \rightarrow b S \mid a A \mid a \mid b C \mid a D$
 $C \rightarrow d C \mid c D \mid c$
 $D \rightarrow d D \mid c C \mid d$

- c) $S \rightarrow a S B C \mid X \mid Y$
 $X \rightarrow a X C \mid a C$
 $Y \rightarrow Y Y \mid B$
 $C B \rightarrow B C$
 $B C \rightarrow C B$
 $B \rightarrow b$
 $C \rightarrow c$

- d) $S \rightarrow a A \mid b A \mid a \mid b$
 $A \rightarrow a B \mid b B$
 $B \rightarrow a C \mid b C$
 $C \rightarrow a D \mid b D$

(* $L(G) \cup \{ \epsilon \}$ é também uma LR? *)

$D \rightarrow aE \mid bE \mid a \mid b$
 $E \rightarrow aS \mid bS$

e) $S \rightarrow CD$
 $C \rightarrow aCA \mid bCB$
 $AD \rightarrow aD$
 $BD \rightarrow bD$
 $Aa \rightarrow aA$
 $Ab \rightarrow bA$
 $Ba \rightarrow aB$
 $Bb \rightarrow bB$
 $C \rightarrow \epsilon$
 $D \rightarrow \epsilon$

f) $S \rightarrow AD$ (* $L(G)$ é também uma LLC? *)
 $A \rightarrow aAC \mid aA \mid a$
 $D \rightarrow BDd \mid Dd \mid d$
 $CB \rightarrow BC$
 $BC \rightarrow CB$
 $B \rightarrow b$
 $C \rightarrow c$

g) $S \rightarrow aSC \mid bSD \mid aSD \mid bSC \mid \epsilon$ (* $L(G)$ é também uma LLC? *)
 $DC \rightarrow CD$
 $CD \rightarrow DC$
 $C \rightarrow c$
 $D \rightarrow d$