UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO
DEPARTAMENTO DE CIÊNCIAS DE COMPUTAÇÃO http://www.icmc.usp.br

SCC-205–Teoria da Computação e Linguagens Formais
Profa. Graça Nunes
2º. Semestre de 2011

Gabarito Prova 2
20/10/2011

Nome:____N.USP_________

(A) Considere as gramáticas a seguir:

1) $S \rightarrow a \mid b \mid aS \mid bS$
(i) (0.25) qual é o tipo da gramática?
Gramática Regular

(ii) (0.25) qual é a linguagem gerada? (se for regular, dê a expressão regular)
$(a+b)^+$

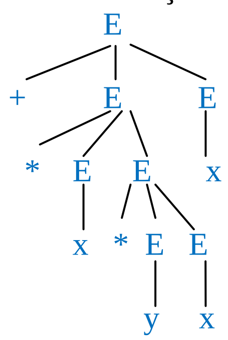
2) $S \rightarrow 0SA \mid 10SA \mid \lambda$
$A \rightarrow 1A \mid 1$
(i) (0.25) Essa gramática gera a linguagem $\mathbf{(0+10)^n\ 1^m + \lambda;\ n \geq 1;\ m \geq n}$
Verdade (X) **Falso ( )**
(ii) (0.25) A linguagem gerada é Regular, embora a gramática seja Livre de Contexto.
Verdade ( ) **Falso (X)**
(iii) (0.25) Uma Gramática Regular que gera a mesma linguagem pode ser:

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

Verdade ( ) **Falso (X)**

3) $E \rightarrow +EE \mid *EE \mid -EE \mid x \mid y$

(i) (0.25) Qual é o tipo da gramática?
Gramática Livre de Contexto
(ii) (0.5) Qual é a linguagem gerada? Ela é regular? Dê um exemplo de sentença de comprimento igual a 7.
Linguagem das expressões prefix. Não é regular. Ex.: +x*y*xyx
(iii) (0.5) Desenhe a árvore de derivação sintática para a sentença de seu exemplo.



(iv) (0.5) Esta gramática é ambígua? Justifique.

Não é ambígua. Mesmo quando há mais de um operador, a escolha sobre a regra a usar é sempre única (devido ao fato de o operador ser gerado sempre mais à esquerda nas regras de produção).

4) (1.0) S → aS | aSbS | λ

Mostre que a gramática acima é ambígua

Considere a cadeia aaabb. Ela possui 2 árvores de derivação (ou 2 dme):

1) S → aSbS → aaSbSbS → aaaSbSbS → aaabb
2) S → aS → aaSbS → aaaSbSbS → aaabb

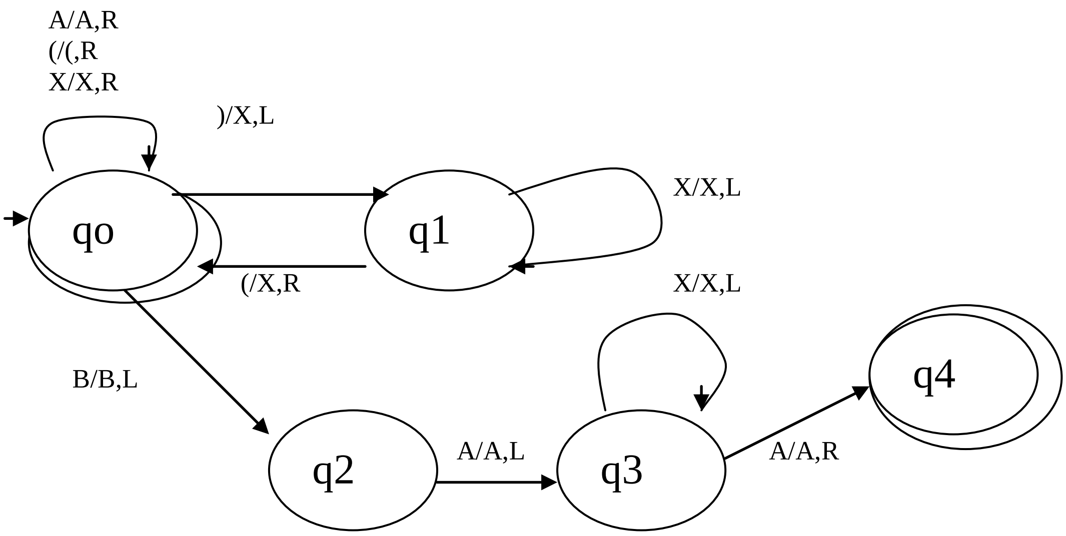
Logo, a gramática é ambígua.

(B)

5) (2.0) A gramática a seguir define a linguagem das cadeias de parênteses balanceados.

```
S -> SS | AB | AC | λ
A -> (
B -> )
C -> SB
```

O diagrama abaixo corresponde a uma MT que verifica se uma cadeia de parênteses, limitada por dois ‘A’ é balanceada ou não, segundo o algoritmo: Procura por um **)** e substitui por X, e, em seguida, volta à esquerda procurando o **(** mais próximo para substituir por X também.



Como você classifica a linguagem dos parênteses balanceados: Recursiva, Recursivamente Enumerável ou Não-Computável? Se Recursiva, ela estaria em **P** ou em **NP-P**? Por quê?

Resp.: Por existir uma MT para ela, então ela é Recursivamente Enumerável. No entanto, como a MT é sempre é capaz de parar para qualquer entrada – num estado final, se for balanceada; num estado não-final, se não for – então, ela é Recursiva. O algoritmo subjacente a esta MT prevê um certo número de vezes em que passa por toda a cadeia de entrada, ou seja, no máximo um certo fator multiplicado pelo comprimento da cadeia. Portanto, seu pior tempo é polinomial e, logo, está em P.

6. A redução de um problema P_1 em um problema P_2 consiste num algoritmo de transformação de toda entrada (instância) *w* de P_1 para uma entrada *x* de P_2 de tal forma que a resposta de P_2 a *x* sempre coincida com a resposta de P_1 a *w*. Assim, se for possível decidir P_2, então é possível decidir P_1.
(a) (1.0) O que significa uma "redução de tempo polinomial"?

Resp.: Uma redução em tempo polinomial de P_1 a P_2 é aquela cujo algoritmo de transformação leva um tempo $O(n^k)$ onde n é o comprimento de w, entrada de P_1.

(b) (1.0) Se houver uma redução em **tempo exponencial** de P_1 a P_2, por que não podemos concluir que "se P_2 está em **P (Classe de Problemas Polinomiais)**, então P_1 também está"?

Resp.: Caso a redução ocorra em tempo exponencial, se o comprimento de w é n, então a transformação gerará x de comprimento $O(2^n)$, e, ainda que P_2 seja resolvido em tempo polinomial sobre sua entrada x, como esta tem comprimento exponencial em relação a w, o tempo gasto para resolver P_1 será também exponencial (um polinômio em 2^n). Logo, P_2 estaria em P, mas P_1 não estaria em P. Assim, apenas a redução polinomial garante que o tempo gasto para decidir P_2 é o mesmo necessário para decidir P_1. Em outras palavras, se P_2 for resolvido em tempo polinomial, ou seja, se pertencer a P, então P_1 também será.

7. (a) (1.0) As Linguagens Recursivamente Enumeráveis (LRE) coincidem com aquelas reconhecidas por Máquinas de Turing. Elas podem ainda ser classificadas (divididas) em Recursivas ou Decidíveis e Não-Recursivas ou Indecidíveis. Defina cada uma das classificações.

Resp.: As LRE podem ser divididas em:

- *(a) Linguagens Recursivas ou Decidíveis: aquelas cujas MT sempre param para toda entrada;*
- *(b) Linguagens Não Recursivas ou Indecidíveis: aquelas cujas MT podem não parar para entradas que não pertencem à Linguagem.*

8) (1.0) Verifique se é Falsa (F) ou Verdadeira (V) cada uma das asserções abaixo. No caso de Falsa, faça a correção necessária.

a) (V) Para provar que uma função é computável, basta exibir uma Máquina de Turing que a compute.

b) (F) Se um problema admitir um algoritmo de ordem exponencial que o resolva, então ele pertence à Classe de Problemas NP.

Não basta que tenha um algoritmo exponencial; é necessário que não haja um polinomial que o resolva. Além disso, a tarefa de verificação tem que ser polinomial.

c) (V) A linguagem $L=\{a^n b^n \mid n \geq 0\}$ é Livre de Contexto. Pode-se então dizer que também são livres de contexto as linguagens bloco-estruturadas do tipo $BEGIN^n$ S END^n, e as linguagens com parênteses balanceados na forma $(^n$ S $)^n$.

d) (F) Linguagens Recursivas são aquelas geradas por gramáticas que têm produções recursivas.

LR são as LRE decidíveis.