Universidade do Estado de Santa Catarina
Ciência da Computação
Complexidade de Algoritmos
Lista de Exercícios II

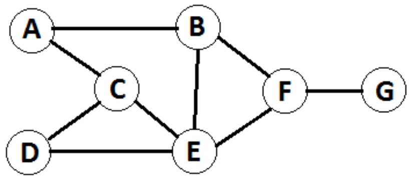
Exercício 1: Reduzir a seguinte instância de SAT em uma instância de 3-CNF-SAT:

$$\phi = x_1 \wedge (\sim x_1 \vee \sim x_2)$$

Exercício 2: Reduzir a seguinte instância de 3-CNF-SAT em uma instância de CLIQUE:

$$\phi = (x_1 \vee \sim x_2 \vee \sim x_3) \wedge (x_1 \vee x_2 \vee \sim x_3) \wedge (\sim x_1 \vee \sim x_2 \vee \sim x_3) \wedge (\sim x_1 \vee x_2 \vee \sim x_3)$$

Exercício 3: Reduzir a seguinte instância de VERTEX-COVER em uma instância de CLIQUE. Informe também uma solução para ambos os problemas:



Exercício 4: Como seria codificada a seguinte sequência de caracteres utilizando a codificação de *Huffman*? Mostre a árvore criada no processo de codificação:

`Abracadabra pe de cabra`

Exercício 5: Considere as matrizes com as seguintes dimensões: A1 = (2,5), A2 = (5,4), A3 = (4,3) e A4 = (3x7). Diga qual a melhor sequencia de multiplicações entre estas matrizes (A1*A2*A3*A4) e quantas multiplicações serão necessárias no mínimo para obter o resultado. Construa as tabelas de multiplicação entre matrizes.

Exercício 6: Escreva um algoritmo em C (i.e. determinista) que verifica em tempo polinomial se a solução abaixo (certificado) é válida para a seguinte instância do problema do SUBSET-SUM:

S[] = {2,3,7,11,13,17,19,22}; t = 42 S'[] = { 3,7,13,19 }

Exercício 7: Verifique se as afirmações abaixo são verdadeiras justificando as falsas:

a) Se um problema *NP* for resolvido em tempo polinomial então *P* = *NP*.

b) Se um problema *NP-Difícil* (ou *NP-Hard*) for resolvido em tempo polinomial então *P* = *NP*.

c) Se *P* = *NP* então todos os problemas considerados *NP-Difícil* (ou *NP-Hard*) podem ser resolvidos em tempo polinomial.

d) Podemos afirmar que os problemas *NP-Completo* possuem apenas soluções em tempo exponencial ou maior.

e) Considerando um problema *P1* que tem uma solução em tempo polinomial conhecida, e um problema *P2* que é *NP-Completo*, apresentando uma redução que pode ser executada em tempo polinomial de *P1* a *P2* (*P1* ≤p *P2*) estamos provando que *P* = *NP*.

f) Se $P \cap$ *NP-Completo* $\neq \emptyset$ então *P* = *NP*.