UFF - Análise e Projeto de Algoritmos - Lista de Exercício - 1/2014

1. Uma pessoa sobe uma escada composta de n degraus, com passos que podem alcançar entre 1 e $k \leq n$ deguaus. Escrever equações de recorrência que permitem determinar o número de modos distintos de a pessoa subir a escada.

2. Resolva a recorrência abaixo:
   $T(1) = 1$,
   $T(n) = 3T(n-1) + 2$, para todo $n > 1$.

3. Considere o seguinte algoritmo:
   Entrada: vetor $L[1 \ldots n]$ contendo n elementos
   para $i = 1 \ldots n-1$ faça
   $j = i + 1$
   enquanto $j \leq n$ faça
   se $L[i] > L[j]$ então troque os elementos $L[i]$ e $L[j]$
   j = j+1

   (a) Qual é a complexidade de pior caso do algoritmo acima? (Beseado em número de trocas entre elementos) Descreva uma entrada com $n = 5$ elementos que leva o algoritmo ao pior caso, mostrando as trocas de elementos efetuadas.

   (b) Idem, mas agora para melhor caso.

4. Considere o seguinte algoritmo de ordenação:
   função ordena(L)
   se $|L| \leq 1$ então retornar L
   senão
   x = primeiro elemento de L
   L_1 = vetor formado pelos elementos de L menores do que x
   L_2 = vetor formado pelos elementos de L iguais a x
   L_3 = vetor formado pelos elementos de L maiores do que x
   retornar a concatenação dos vetores: ordena(L_1), L_2, ordena(L_3)

   (a) Mostre a árvore de recursão deste algoritmo para a entrada $L = [5, 2, 1, 7, 9, 3, 1, 6]$. Cada nó desta árvore é uma chamada recursiva. Ao definir os vetores L_1, L_2, L_3, respeitar a mesma ordem relativa que os elementos têm em L.

   (b) Analise a complexidade de pior caso deste algoritmo para um vetor L com n elementos. Mostre uma entrada com n = 8 que leva o algoritmo ao pior caso.

5. Considere o seguinte algoritmo:
   função mistério(y, z)
   entrada: números naturais y e z
   x = 0
   enquanto $z > 0$ faça
       se z mod 2 = 1 então (z mod 2 é o resto da divisão de z por 2)
           x = x + y
       y = 2y
       z = $\lfloor z/2 \rfloor$
   retorne x

6. (**Subsequencia Monotônica Mais Longa**) Seja $S = s_1, s_2, \ldots, s_n$ uma sequencia de números naturais. Uma subsequencia de S é uma sequencia da forma $S' = s_{i_1}, s_{i_2}, \ldots, s_{i_k}$, onde $1 \leq i_1 < i_2 < \ldots < i_k \leq n$. O valor k é o comprimento de S'. A subsequencia S' é dita monotônica se $s_{i1} \leq s_{i_2} \leq \ldots \leq s_{i_k}$. Desenvolva um algoritmo de divisão-e-conquista para encontrar uma subsequencia monotônica de S de comprimento máximo. O algoritmo deve rodar em tempo $O(n^2)$.

7. (**Cobertura Mínima por Intervalos**) Sejam $[l_i, r_i]$, $1 \leq i \leq n$, intervalos fechados sobre a reta real. Desenvolva um algoritmo guloso que determine a menor quantidade de intervalos para cobrir completamente o intervalo $[0, M]$, para um dado $M \geq 0$, ou conclua que não é possível cobri-lo.

8. (**Otimização de Transporte**) Na ilha de Tamoa, um guia de turismo necessita transportar K turistas de uma cidade origem s para uma cidade destino t. As cidades de Tamoa são representadas por vértices de um grafo conexo G. Cada aresta (u, w) de G representam uma estrada ligando u e w, na qual opera um serviço de ônibus que transporta no máximo $w(u, v)$ passageiros, tanto num sentido como no outro. Elabore um algoritmo que determine o menor número de viagens de s a t que o guia necessita fazer para transportar todos os turistas.

9. Falso ou verdadeiro? (justifique cuidadosamente)

   (a) Seja S uma lista linear sequencial com $n > 0$ elementos, n divisível por 5. Seja M a lista das medianas de $S[1 \ldots 5], S[6 \ldots 10], \ldots, S[n-4 \ldots n]$. Então, a mediana de M é a mediana de S.

   (b) Se as complexidades de melhor e pior caso de um algoritmo são ambas $\Theta(f(n))$, então a complexidade de caso médio também é $\Theta(f(n))$.

(c) Dados n blocos de base idêntica B, mas com alturas distintas $h_1, \ldots, h_h$, onde cada h_i é um número real no intervalo $[0, 1]$, deseja-se colocá-los em caixas com base B e altura unitária, denotadas por C_1, C_2, etc. O seguinte algoritmo guloso resolve o problema de encontrar o menor número de caixas necessárias para armazenar todos os blocos: para cada $i = 1, \ldots, n$ coloque o bloco de altura h_i na caixa de menor índice que o possa comportar. (Portanto, o maior índice utilizado pelo algoritmo será o número de caixas usadas.)

10. Suponha uma lista de n números, representando os resultados de computação ("votos") de n processadores. Deseja-se decidir se há um voto majoritário e qual é este voto. (Dizemos que há um voto majoritário quando mais da metade dos processadores chegou ao mesmo resultado em sua computação.) Escreva um algoritmo eficiente para determinar se há um voto majoritário, baseado somente em comparações entre pares de votos, isto é, não é permitido fazer contagem. Calcule a complexidade do seu algoritmo. Você também pode interpretar este problema da seguinte forma: existem n esferas, cada qual com uma cor, e deseja-se saber se há uma cor majoritária entre elas.

11. Elabore um algoritmo de decomposição de um vetor S em três subvetores. Esse algoritmo recebe como entrada, além do vetor S, um valor piv pertencente a S, e os índices p e r, $1 \leq p \leq r$. O algoritmo deve rearrumar os elementos em $S[p \ldots r]$ e retornar dois índices i e j satisfazendo as seguintes propriedades:

    (a) se $p \leq k \leq i$, então $S[k] < piv$;
    (b) se $i \leq k \leq j$, então $S[k] = piv$;
    (c) se $j \leq k \leq r$, então $S[k] > piv$.

    A complexidade do seu algoritmo deve ser $O(n)$.

12. (**Os k Menores**) Suponhamos que em vez de selecionar o k-ésimo menor elemento de um conjunto com n elementos, estamos interessados em determinar os k menores elementos (sem nos preocuparmos com a ordem relativa entre eles). É possível realizar esta tarefa em tempo $O(n)$?

13. Elabore um algoritmo que, dado um vetor S com $n > 0$ elementos, retorna um vetor V, de tamanho n, com a seguinte propriedade: $V[i]$ é o número de ocorrências de $S[i]$ em S. A complexidade do seu algoritmo deve ser $O(n \log n)$.

14. Alguns projetores foram alocados para um congresso, mas na "hora H"apenas um estava funcionando. Entre as n palestras previstas para o primeiro dia, os organizadores decidiram selecionar o maior número possível delas para serem apresentadas neste projetor, deixando as restantes para dias posteriores. Esta seleção deveria respeitar rigorosamente os horários marcados na programação. Na programação do evento, cada palestra p_i tem um horário de início s_i e um horário de término t_i $(1 \leq i \leq n)$. Um dos organizadores propôs o seguinte algoritmo guloso, onde A denota o conjunto de palestras selecionadas: (suponha que $t_1 \leq t_2 \leq \ldots \leq t_n$):

    $A \leftarrow \{p_1\}$
    $j = 1$
    para $i = 2 \ldots n$ faça:
        se $s_i \geq t_j$ então
            $A \leftarrow A \cup \{p_i\}$
            $j = i$

    Este algoritmo funciona? Justifique sua resposta.

15. (**Sequenciamento de tarefas**) Seja $J = \{J_1, \ldots, J_n\}$ um conjunto de tarefas que devem ser executadas sequencialmente em um mesmo processador. Cada tarefa J_i consome uma unidade de tempo do processador e produz um lucro $r_i > 0$ caso seja concluída até o tempo limite t_i, $i = 1, \ldots, n$. Se J_i for concluída após t_i , nenhum lucro é obtido desta tarefa. O lucro total de uma sequência de execução das tarefas é a soma dos lucros obtidos pelas tarefas que foram concluídas até o tempo limite. Descreva um algoritmo que determina uma sequência de execução das tarefas em que o lucro total é maximizado. Prove que o algoritmo está correto. Qual a sua complexidade?

16. O Problema do Caixeiro Viajante consiste em um conjunto de n cidades e um custo não negativo C_{ij} associado a cada par de cidades i, j. O objetivo é determinar um caminho que, partindo da cidade 1, passe por todas as cidades (exatamente uma vez em cada cidade) e retorne à cidade 1, de modo que a soma dos custos no caminho seja mínima. Mediante um contra-exemplo, mostrar que o algoritmo guloso não resolve o Problema do Caixeiro Viajante.