Lista de exercícios 1
Cana - 2023.1

Questão 1. Prove ou refute as seguintes afirmações sobre notação assintótica:

(a) $2n + 3n + 4 = O(n^2)$

(b) $n^3/100 - 25n^2 + 100n - 7 = \Theta(n^3)$

(c) $\log_2 n = O(n)$

(d) $n^3 = O(n^2)$

(e) $2n^2 = o(n^3)$

(f) $\frac{n^2}{2} = \omega(n^2)$

Questão 2. Considere o algoritmo abaixo que recebe um vetor ordenado $A[1 \dots n]$ de números inteiros positivos e um outro número inteiro positivo x.

Algoritmo 1: Algoritmo $B(A[i \dots f], x)$

se $f < i$ **então**
  Retorna -1;
$j \leftarrow \lfloor (i + f)/2 \rfloor$
se $A[j] = x$ **então**
  Retorna j;
se $A[j] < x$ **então**
  Retorna $B(A[j + 1 \dots f], x)$;
se $A[j] > x$ **então**
  Retorna $B(A[i \dots j - 1], x)$;

(a) Simule a execução do Algoritmo B no vetor $< 3, 5, 9, 14, 17, 23, 29 >$ com os números 23 e 6, indicando as comparações de elementos que são realizadas durante a execução.

(b) Descreva sucintamente a funcionalidade do Algoritmo B.

(c) Prove a corretude do Algoritmo B.

Questão 3. Escreva um algoritmo para realizar a busca linear recursiva em um vetor e mostre a sua corretude.

Questão 4. Considere o seguinte algoritmo para o cálculo da subsequência contígua de soma máxima em um vetor $A[1..n]$ de números inteiros (a subsequência vazia tem soma 0).

Algoritmo 2: Algoritmo $SomaMax(A[1 \dots n])$

$m \leftarrow 0$;
$mt \leftarrow 0$;
para todo $i \leftarrow 1 \dots n$ **faça**
    $mt \leftarrow \max(A[i], mt + A[i])$;
    $m \leftarrow \max(m, mt)$;

Prove a corretude de $SomaMax(A[1 \dots n])$ utilizando a seguinte invariante: m é a subsequência de soma máxima de $A[1..i-1]$, e mt é a subsequência de soma máxima terminada em $i-1$ no subvetor $A[1..i-1]$.

Questão 5. Elabore um algoritmo em $\mathcal{O}(n)$ 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 $S[p \dots r]$ e retornar dois índices q_1 e q_2 satisfazendo as seguintes propriedades:

(a) se $p \leq k \leq q_1$, então $S[k] < piv$;

(b) se $q_1 < k \leq q_2$, então $S[k] = piv$;

(c) se $q_2 < k \leq r$, então $S[k] > piv$.

Questão 6. Considere uma sequência ordenada armazenada no vetor $A[1..n]$, e suponha que queremos encontrar a posição do elemento de valor x. O processo de busca binária resolve este problema de maneira eficiente. A sua análise mostra que o vetor original pode ser dividido em duas metades no máximo $\log_2 n$ vezes. Para cada divisão, é necessário realizar uma comparação para decidir em que metade o elemento x. Portanto, o processo requer $\log_2 n$ comparações.

Considere agora o processo de busca quaternária, que a cada passo divide a lista em quatro partes. Faça um algoritmo que implemente o procedimento de busca quaternária. Apresente uma estimativa do número de comparações requeridas para realizar este processo (no pior caso).

Questão 7. Projete um algoritmo que recebe um vetor $A[1..n]$ de números em ordem não decrescente e um número x, e retorna a localização da primeira ocorrência de x em $A[1..n]$, ou o local em que x poderia ser inserido sem violar a ordenação se x não ocorrer no vetor. Calcule a complexidade do seu algoritmo.

Questão 8. Elabore um algoritmo em $\Theta(n \log n)$ para resolver o seguinte problema: dado um vetor com n números inteiros positivos e um outro número inteiro positivo x, determinar se existem ou não dois elementos cuja soma é igual a x.

Questão 9. Elabore um algoritmo em $\Theta(n \log n)$ 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. Prove esta complexidade.

Questão 10. Altere o algoritmo HEAP-SORT para trabalhar com heaps mínimos ao invés de heaps máximos. Argumente porque é melhor trabalhar com heaps máximos neste caso.