Lista 1

14 de agosto de 2013

1 Exercícios Básicos

1.1 Na bibliografia

Dasgupta: Capítulo 0, exercícios 1 e 2. Exercícios 2.16 e 2.17
Tardos & Kleinberg: Todos exercícios do cap 2 do livro texto, exceto 7 e 8 letra b.
Cormen: 10.3-2,10.3-5,10.3-6,10.3-7,10.3-9(Primeira Edição) ou 9.3-2,9.3-5,9.3-6,9.3-7,9.3-9(Segunda Edição)

1.2 Outros

1. Analise a complexidade do algoritmo abaixo determinando uma função $f(n)$ tal que $T(n)$, a complexidade de pior caso do algoritmo, é $\Theta(f(n))$.

    Leia(n);

    $x \leftarrow 0$

    Para $i \leftarrow 1$ até n faça

        Para $j \leftarrow i + 1$ até n faça

            Para $k \leftarrow 1$ até $j - i$ faça

                $x \leftarrow x + 1$

2. Dizemos que um vetor `P[1..m]` ocorre em um vetor `T[1..n]` se `P[1..m] = T[s+1,.., s+m]` para algum s. O valor de um tal s é um deslocamento válido. Projete um algoritmo para encontrar todos os deslocamentos válidos em um vetor e analise sua complexidade em função de m e n.

3. Seja $A[1..n]$ um vetor que pode conter números positivos e negativos.

    Projete um algoritmo com complexidade $O(n^3)$ para determinar os índices i e j, com $i \leq j$, tal que $A[i] + \ldots + A[j]$ é máximo. Tente reduzir a complexidade para $O(n^2)$ e depois para $O(n)$.

4. Resolvas as equações abaixo encontrando uma função $f(n)$ tal que $T(n) = \theta(f(n))$

    - $T(n) = 2T(n/2) + n^2$ para $n > 1$; $T(1) = 1$

- $T(n) = 2T(n/2) + n$, para $n > 1$; $T(1) = 1$
- $T(n) = T(n/2) + 1$, para $n > 1$; $T(1) = 1$
- $T(n) = T(n/2) + n$, para $n > 1$; $T(1) = 1$

5. Seja $A = \{a_1 < \ldots < a_n\}$ uma lista ordenada de números reais. A proximidade entre a_i e a_j é definida como $|a_i - a_j|$. Dados os inteiros j e k, encontre os k elementos de A mais próximos de a_j em $O(k)$.

6. Seja $S = \{a_1, \ldots, a_n\}$ um conjunto de n números naturais distintos e um inteiro x. Considere o problema $\mathcal{P}$ de determinar se existem três números naturais distintos em S cuja soma é x

   a) Seja $T(n)$ a complexidade de pior caso do algoritmo abaixo para resolver $\mathcal{P}$. Encontre $f(n)$ tal que $T(n) = \Theta(f(n))$.

   Para i=1,...,n-2
       Para j=i+1,...,n-1
           Para k=j+1,...,n
               Se a_i+a_j+a_k =x
                   Return SIM
   Return NÃO

   b) Projete um algoritmo com complexidade $O(n^2 \log n)$ para resolver o problema $\mathcal{P}$. Não é necessário apresentar o pseudo-código mas sim explicar com clareza os passos que o algoritmo deve realizar e explicar a complexidade.

   c) Projete um algoritmo com complexidade $O(n^2)$ para resolver o problema $\mathcal{P}$. Não é necessário apresentar o pseudo-código mas sim explicar com clareza os passos que o algoritmo deve realizar e explicar a complexidade.

7. Considere os pseudo-códigos abaixo.

   a) Determine para o pseudo código 1 uma função $f(n)$ tal que $T(n) = \theta(f(n))$.

   Pseudo1

```
t ← 0
Cont ← 1
Para i=1 até n
    Cont ← cont+1
Fim Para
Enquanto cont ≥ 1
    Cont ← cont/2
    Para j = 1 a n
        t++
    Fim Para
Fim Enquanto
```

   b) Determine para o pseudo código 2 uma função $g(n)$ tal que $T(n) = \theta(g(n))$.

   Pseudo2

```
i ← 0
Enquanto i^2 ≤ n
    i ++
    t ← 0
    Enquanto t ≤ i
        t++
    Fim Enquanto
Fim Enquanto
```

8. Seja A um vetor contendo n números reais.

   a) Explique como seria um algoritmo para devolver o número em A que aparece mais vezes. Em caso de empate, o algoritmo pode devolver qualquer um dos números empatados. Análise a complexidade do algoritmo proposto. Quanto mais eficiente melhor.

   b) Assuma agora que todo número em A pertence ao conjunto $\{n^2, n^2+1, ..., n^2+n\}$. Responda o item anterior tendo em vista esta hipótese.

9. Seja S um conjunto de n numeros reais distintos. Explique como seria um algoritmo eficiente para encontrar os $\sqrt{n}$ menores números do conjunto S e analise sua complexidade. Quanto mais eficiente o algoritmo melhor.

2 Extras (Mais difíceis)

1. Seja uma matriz quadrada A com n^2 números inteiros que satisfaz as seguintes propriedades:

   (a) $A[i, j] \leq A[i+1, j]$ para $1 \leq i \leq n-1$ e $1 \leq j \leq n$

   (b) $A[i, j] \leq A[i, j+1]$, para $1 \leq i \leq n$ e $1 \leq j \leq n-1$

   Dado um elemento x, descreva um procedimento eficiente para determinar se x pertence a A ou não. Analise a complexidade do algoritmo proposto. Mostre que este problema tem complexidade $\theta(n)$

2. Mostre como ordenar n inteiros no intervalo $[1, n^2]$ em tempo linear $O(n)$.

3. Mostre que para fazer o merge de duas listas com n elementos é necesário realizar pelo menos $2n-1$ comparações no pior caso.

4. Mostre que para fazer o merge de duas listas, uma com n elementos e outra com m elementos, é necesário realizar pelo menos comparações $\log \binom{n+m}{n}$ no pior caso.

5. Perdido em uma terra muito distante, você se encontra em frente a um muro de comprimento infinito para os dois lados (esquerda e direita). Em meio a uma escuridão total, você carrega um lampião que lhe possibilita ver apenas a porção do muro que se encontra exatamente à sua frente (o campo de visão que o lampião lhe proporciona equivale exatamente ao tamanho de um passo seu). Existe uma porta no muro que você deseja atravessar. Supondo que a mesma esteja a n passos de sua posição inicial (não se sabe se à direita ou à esquerda), elabore um algoritmo para caminhar ao longo do muro que encontre a porta em $O(n)$ passos. Considere que n é um valor desconhecido (informação pertencente à instância). Considere que a ação composta por dar um passo e verificar a posição do muro correspondente custa $O(1)$