Universidade Federal do Ceará

Campus de Quixadá

QXD0041 - Projeto e Análise de Algoritmos

Lista de exercícios 1

1. Uma pessoa sobe uma escada composta de n degraus, com passos que podem alcançar entre 1 e $k \leq n$ degraus. Escrever equações de recorrência que permitem determinar o número de modos distintos da pessoa subir a escada. Dica: comece com $k = 2$.

2. Prove as seguintes afirmações sobre notação assintótica:

   (a) $n^3/100 - 25n^2 - 100n + 7$ é $\theta(n^3)$

   (b) $77n^3 - 13n^2 + 29n - 5$ é $\theta(n^3)$

   (c) $34nlog_7n + 13n$ é $\Omega(n)$ e $O(n^2)$

3. É 2^{n+1} é $O(2^n)$? É 2^{2n} é $O(2^n)$?

4. Considere a seguinte recorrência:

   $$T(n) = \begin{cases} T(n/2) + logn & n > 1 \\ 1 & n = 1 \end{cases}$$

   Mostre que $T(n) = O(log^2 n)$

5. No seguinte problema, assuma que n é uma potência de 3:

   $$T(n) = \begin{cases} 3T(n/3) + \theta(n^2) & n > 1 \\ \theta(1) & n \leq 1 \end{cases}$$

   (a) Use o método da árvore da recursão para encontrar um limite superior e um inferior para $T(n)$

   (b) Use a indução para verificar os limites encontrados.

6. Resolva as seguintes equações de recorrência segundo o método da árvore de recursão:

   (a) $T(n) = 2T(n-2) + 1, T(1) = 1$

   (b) $T(n) = T(0.9n) + 7, T(1) = 1$

   (c) $T(n) = 3T(n/2) + n^2, T(1) = 1$

   (d) $T(n) = 4T(n/2) + n^2, T(1) = 1$

   (e) $T(n) = 5T(n/2) + n^2, T(1) = 1$

   (f) $T(n) = T(\sqrt{n}) + log_2 n, T(1) = 1$

(g) $T(n) = 2T(\sqrt{n}) + log_2 n, T(1) = 1$

(h) $T(n) = 2T(n/5) + 3T(n/6) + n, T(1) = 1$

7. Use o teorema Mestre para encontrar a complexidade das seguintes funções:

   (a) $T(n) = 2T(n/4) + 7n - 15, T(1) = 1$

   (b) $T(n) = 9T(n/3) + 3n^2 + 5n + 16, T(1) = 1$

   (c) $T(n) = 8T(n/2) + 15, T(1) = 1$

8. Dado a recorrência $T(n) = 4T(n/2) + n^k$, qual é o maior valor de k tal que $T(n)$ seja $O(n^3)$?

9. O algoritmo de ordenação por inserção pode ser expresso como um algoritmo recursivo da seguinte maneira. Para ordenar $A[1 \ldots n]$, nós recursivamente ordenamos $A[1 \ldots n-1]$ e então inserimos A[n] no vetor ordenado $A[1 \ldots n-1]$. Escreva a recorrência para o tempo de execução da versão recursiva da ordenação por inserção.

10. Suponha que você está tentando escolher entre os três algoritmos abaixo. Qual o tempo de cada um em notação assintótica e qual você escolheria?

    - Algoritmo A resolve o problema dividindo a entrada em cinco subproblemas com a metade do tamanho, resolve cada subproblema recursivamente e depois combina-os em tempo linear.
    - Algoritmo B resolve o problema dividindo a entrada em dois subproblemas de tamaho $n-1$ (onde n é o tamanho da entrada), resolve cada subproblema recursivamente e depois combina-os em tempo constante.
    - Algoritmo C resolve o problema dividindo a entrada em nove subproblemas com um terço do tamanho, resolve cada subproblema recursivamente e depois combina-os em tempo quadrático.

11. Dona Inês preparou uma pilha de n tapiocas $p_1, p_2, \ldots, p_n$ e deseja ordenar esta pilha em ordem crescente de tamanho, com a menor tapioca no topo. Uma única operação é possível: inserir a espátula entre duas tapiocas na pilha e virar. Por exemplo, considere a seguinte pilha $[2, 5, 3, 1, 7, 6, 4]$, se ela insere a espátula logo abaixo da tapioca de número 1 então a pilha $[1, 3, 5, 2, 7, 6, 4]$ é obtida depois de virar a espátula. Defina um algoritmo que soluciona o problema de Dona Inês utilizando apenas a operação de virar para ordenar as tapiocas. É claro que é permitido a Dona Inês examinar a pilha de tapiocas antes de escolher a posição de enfiar a espátula. Considere que o número na posição p_i representa o tamanho da tapioca. Escreva um algoritmo que resolva o problema da Dona Inês. Qual é a ordem de complexidade do seu algoritmo?

12. Seja $X[1 \ldots n]$ um vetor ordenado de números distintos. Escreva um algoritmo $O(log\ n)$ que determina se existe um índice i tal que $X[i] = i$.

13. Seja $X[1 \dots n]$ um vetor ordenado de inteiros positivos tal que $X[i] \leq M$, para todo $1 \leq i \leq n$. Escreva um algoritmo $O(log\ n)$ que conte o número de ocorrências para cada elemento no vetor.

14. Seja $X[1 \dots n]$ e $Y[1 \dots n-1]$ dois vetores ordenados com uma única diferença entre eles. O primeiro vetor tem um único elemento extra. Escreva um algoritmo $O(log\ n)$ que encontre o elemento extra. Por exemplo, considerando $X = [2, 4, 6, 8, 9, 10, 12]$ e $Y = [2, 4, 6, 8, 10, 12]$, o elemento extra é 9.

15. Seja $X[1 \dots n]$ e $Y[1 \dots n]$ de tamanho n. Escreva um algoritmo $O(log\ n)$ que encontra a mediana do vetor obtido pelo merge dos dois vetores ordenados, ou seja, o vetor de tamanho $2n$. Por exemplo, se $X = \{1, 12, 15, 26, 38\}$ e $Y = \{2, 13, 17, 30, 45\}$, a mediana do vetor ordenado obtido pelo merge dos dois vetores ordenados é 16.

16. Seja $X[1 \dots n]$ um vetor que representa um progressão aritmética. Nessa progressão aritmética, temos um elemento faltando. Escreva um algoritmo $O(log\ n)$ que encontre o elemento que está faltando.

17. Um vetor bitônico é uma vetor formado por uma parte estritamente crescente seguida por um$O(log\ n)$ parte estritamente decrescente. Mais precisamente, um vetor $A[1 \dots n]$ é bitônico se somente se existe um índice i, $1 \leq i \leq n$, tal que $A[1 \dots i]$ é estritamente crescente e $A[i + 1 \dots n]$ é estritamente decrescente. Por exemplo, [1,2,3,7,6,4] é um vetor bitônico com $i = 4$. Proponha um algoritmo $\theta(logn)$ que encontre o maior elemento de um vetor bitônico(você pode assumir que todos os elementos são distintos).

18. Seja $X[1 \dots n]$ um vetor bitônico. Escreva um algoritmo de tempo $\theta(log\ n)$ que encontre um elemento x no vetor bitônico X. Por exemplo, o elemento 20 está no vetor bitônico $[-3, 9, 8, 20, 17, 5, 1]$ na posição 4.

19. Seja uma matriz $M[1 \dots n][1 \dots m]$ em que toda linha e toda coluna está ordenada em ordem crescente. Escreva um algoritmo $O(n + m)$ que encontra a posição de um elemento x na matriz M se ele está presente nela caso contrário mostra uma mensagem "não encontrado".

```
Entrada: M[3][4] = { {10,20,30,40},
                     {15,25,35,45},
                     {27,29,37,48}};
Saída: (3,2)
```

20. Seja uma matriz $M[1 \dots n][1 \dots n]$ em que toda linha e toda coluna está ordenada em ordem crescente. Escreva um algoritmo $O(n^{1.58})$ que encontra a posição de um elemento x na matriz M se ele está presente nela caso contrário mostra uma mensagem "não encontrado". Dica: Mostre que é possível descartar uma matriz $n/2 \times n/2$ em tempo constante.

21. Seja $X[1 \dots n]$ um vetor qualquer (os elementos desse vetor não são necessariamente inteiros ou caracteres; podem ser objetos quaisquer, como frutas ou arquivos executáveis). Suponha que você possui apenas um operador $=$ que permite comparar se um objeto é igual a outro. Dizemos que X tem um elemento majoritário x se mais da metade de seus elementos são iguais a x. Escreva um algoritmo de tempo $\theta(nlogn)$ que diz se X possui ou não um elemento majoritário. Caso sim, devolva o seu valor. Dica: Se x é majoritário em X, então x é majoritário na primeira ou na segunda metade de X (explique porquê).

22. Seja $X[1 \dots n]$ um vetor de inteiros. Dados $i < j$ em $\{1, \dots, n\}$, dizemos que (i, j) é uma inversão de X se $X[i] > X[j]$. Escreva um algoritmo $\theta(nlogn)$ que devolva o número de inversões em um vetor X. Dica: Tenta fazer essa contagem enquanto ordena o vetor no Merge-Sort.