MAC 338 - Análise de Algoritmos

Departamento de Ciência da Computação

Primeiro semestre de 2008

Lista 2

1. Resolva as recorrências abaixo. Para todas elas, suponha que $T(1) = \Theta(1)$.

   (a) $T(n) = 8T(n/2) + \Theta(n^2)$

   (b) $T(n) = 2T(n/2) + \Theta(n^3)$

   (c) $T(n) = 7T(n/3) + \Theta(n^2)$

   (d) $T(n) = T(\lfloor 9n/10 \rfloor) + \Theta(n)$

2. Escreva um algoritmo eficiente que busca um valor x em uma matriz $A_{m \times n}$ cujas linhas e colunas estão ordenadas em ordem não-decrescente.

3. Escreva um algoritmo que ordena uma lista de n itens dividindo-a em três sublistas de aproximadamente $n/3$ itens, ordenando cada sublista recursivamente e intercalando as três sublistas ordenadas. Analise seu algoritmo concluindo qual é o seu consumo de tempo.

4. Considere o seguinte problema: dados n, uma seqüência de números reais $a_n, \ldots, a_0$ e um número real x, determinar o valor do polinômio $p_n(x) = a_n x^n + \cdots + a_1 x + a_0$. Pode-se resolver esse problema calculando o valor de $q_{n-1}(x) = a_n x^{n-1} + \cdots + a_2 x + a_1$ e, posteriormente, $p_n(x) = x\, q(x) + a_0$. Esse método é chamado de *regra de Horner*. Projete um algoritmo de divisão e conquista para resolver o problema e compare-o ao método derivado da regra de Horner. Quantas adições e quantas multiplicações faz o seu algoritmo?

5. Sejam $X[1\,..\,n]$ e $Y[1\,..\,n]$ dois vetores, cada um contendo n números ordenados. Escreva um algoritmo $O(\lg n)$ para encontrar a mediana de todos os $2n$ elementos nos vetores X e Y.

6. Seja $X[1\,..\,n]$ um vetor de inteiros e i e j dois índices distintos de X, ou seja, i e j são inteiros tais que $1 \leq i < j \leq n$. Dizemos que (i, j) são uma *inversão* de X se $X[i] > X[j]$. Escreva um algoritmo $O(n \lg n)$ que devolva o número de inversões em um vetor X, onde n é o número de elementos em X.

7. A remoção da superfície escondida é um problema em computação gráfica que raramente precisa de introdução: quando o João tá na frente da Maria, você pode ver o João, mas não a Maria; quando a Maria tá na frente do João, ... Você entendeu a idéia.

   A beleza desse problema é que você pode resolvê-lo mais rapidamente do que a intuição em geral sugere. Aqui está uma versão simplificada do problema onde já podemos apresentar um algoritmo mais eficiente do que a primeira solução em que se pode pensar. Imagine que são dadas n retas não verticais no plano, denotadas por $L_1, \ldots, L_n$. Digamos que L_i é dada pela equação $y = a_i x + b_i$, para $i = 1, \ldots, n$. Suponha que não há três retas entre as retas dadas que se interseptam mutuamente num mesmo ponto. Dizemos que a reta L_i é a *mais alta* numa dada coordenada $x = x_0$ se sua coordenada y em x_0 é maior que a coordenada y em x_0 de todas as outras retas dadas. Ou seja, se $a_i x_0 + b_i > a_j x_0 + b_j$ para todo $j \neq i$. Dizemos que L_i é *visível* se existe uma coordenada x na qual ela é a mais alta. Intuitivamente, isso corresponde a uma parte de L_i ser visível se você olhar para baixo a partir de $y = \infty$.

   Escreva um algoritmo $O(n \lg n)$ que recebe uma seqüência de n retas, como descrito acima, e devolve a subseqüência delas que é visível.

8. A *silhueta de um prédio* é uma tripla (l, h, r) de números positivos com $l < r$, onde h representa a altura do prédio, l representa a posição inicial da sua base e r a posição final.

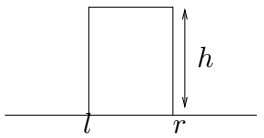

Figura 1: Silhueta (l, h, r) de um prédio.

Considere uma coleção $\mathcal{S} = \{(l_1, h_1, r_1), \ldots, (l_n, h_n, r_n)\}$ com a silhueta de n prédios. Para cada número positivo x, denote por $\mathcal{S}_x$ o conjunto $\{i : 1 \leq i \leq n \text{ e } l_i \leq x \leq r_i\}$. Denote ainda por $h(\mathcal{S}_x)$ o número $\max\{h_i : i \in \mathcal{S}_x\}$.

O *skyline* de $\mathcal{S}$ é uma seqüência $(x_0, t_1, x_1, \ldots, t_k, x_k)$ tal que

1. $x_0 = 0$ e $x_k = \max\{r_i : 1 \leq i \leq n\}$;
2. $x_{j-1} < x_j$ para $j = 1, \ldots, n$;
3. para $j = 1, \ldots, n$, $t_j = h(\mathcal{S}_x)$ para todo x tal que $x_{j-1} < x < x_j$;
4. $t_j \neq t_{j+1}$ para $j = 1, \ldots, n-1$.

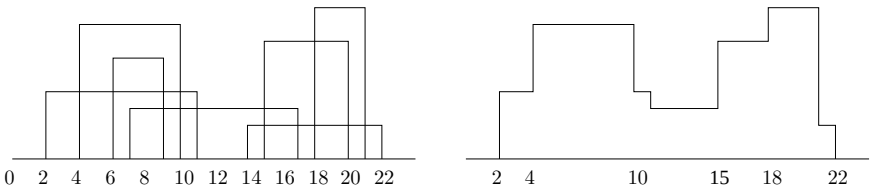

Figura 2: Coleção de silhuetas $\mathcal{S} = \{(15, 7, 20), (4, 8, 10), (18, 9, 21), (2, 4, 11), (7, 3, 17), (6, 6, 9), (14, 2, 22)\}$ e seu skyline $(0, 0, 2, 4, 4, 8, 10, 4, 11, 3, 15, 7, 18, 9, 21, 2, 22)$.

(a) Escreva um algoritmo que recebe o skyline de uma coleção $\mathcal{S}_1$ de silhuetas de prédios e o skyline de uma coleção $\mathcal{S}_2$ de silhuetas de prédios e devolve o skyline de $\mathcal{S}_1 \cup \mathcal{S}_2$. Seu algoritmo deve consumir tempo $O(n)$, onde $n = |\mathcal{S}_1 \cup \mathcal{S}_2|$. Explique por que seu algoritmo faz o que promete e por que consome o tempo pedido.

(b) Escreva um algoritmo que recebe um inteiro n e uma coleção $\mathcal{S}$ de n silhuetas de prédios e devolve o skyline de $\mathcal{S}$. Seu algoritmo deve consumir tempo $O(n \log n)$. Explique por que seu algoritmo faz o que promete e por que consome o tempo pedido.