MAC 5711 - Análise de Algoritmos

Departamento de Ciência da Computação
Segundo semestre de 2005

Lista 3

1. Suponha que os elementos do vetor $A[1\,..\,n]$ são distintos dois a dois. Uma *inversão* é um par (i, j) de índices tal que $i < j$ mas $A[i] > A[j]$. Escreva um algoritmo que calcule o número de inversões em $A[1\,..\,n]$ em tempo $O(n \lg n)$. (*Sugestão:* Inspire-se no algoritmo Mergesort.)

2. Qual é o consumo de espaço do QUICKSORT no pior caso?

   Quando um algoritmo recursivo tem como último comando executado, em algum de seus casos, uma chamada recursiva, tal chamada é denominada *recursão de cauda* (*tail recursion*). Um exemplo de recursão de cauda acontece no QUICKSORT.

   Toda recursão de cauda pode ser substituída por uma repetição. No caso do QUICKSORT, obtemos o seguinte:

   ```
   QUICKSORT (A, p, r)
   1   enquanto p < r faça
   2       q ← PARTICIONE (A, p, r)
   3       QUICKSORT (A, p, q − 1)
   4       p ← q + 1
   ```

   Mostre como essa idéia pode ser usada (de uma maneira mais esperta) para melhorar significativamente o consumo de espaço no pior caso do QUICKSORT.

3. 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.

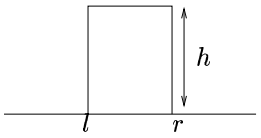

Figura 1: Silhueta (l, h, r) de um prédio.

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 $j = 1, \ldots, n-1$.

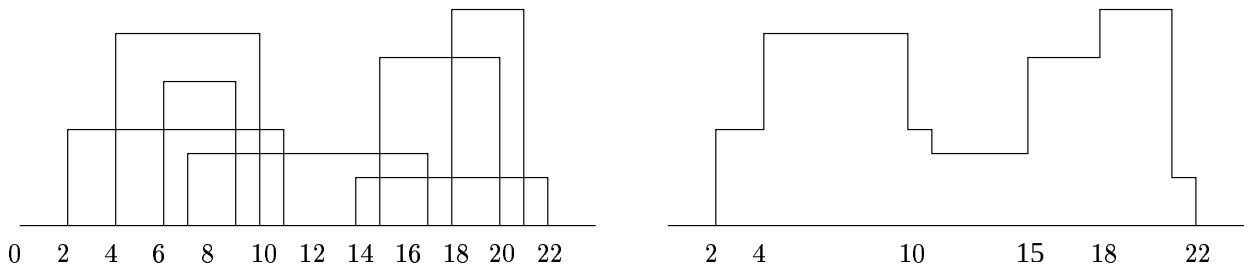

Figura 2: Coleção de silhuetas $\mathcal{S} = \{(15, 7, 20), (4, 8, 10), (18, 9, 21), (2, 4, 11), (7, 3, 17), (6, 6, 9), (14, 2, 22)\}$ e seu skyline $(0, 0, 2, 4, 4, 8, 10, 4, 11, 3, 15, 7, 18, 9, 21, 2, 22)$.

(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.

4. Considere o seguinte algoritmo recursivo para calcular o máximo de um vetor $v[p\,..\,r]$.

Algoritmo Máximo (v, p, r)

```
1. se p = r
2.     então devolva v[p]
3.     senão q ← ⌊(p+r)/2⌋
4.           m1 ←Máximo (v, p, q)
5.           m2 ←Máximo (v, q + 1, r)
6.           se m1 > m2
7.                então devolva m1
8.                senão devolva m2
```

Seja $C(n)$ o número de comparações executadas na linha 7 do algoritmo por uma chamada de Máximo(v, p, r), onde $n = r - p + 1$. Deduza do algoritmo uma recorrência que defina $C(n)$ e resolva-a, exibindo uma fórmula fechada para $C(n)$. (Ou seja, dê a solução exata da recorrência.)

5. Considere o seguinte algoritmo que determina o segundo maior elemento de um vetor $v[1\,..\,n]$ com $n \geq 2$ números positivos distintos.

Algoritmo Máximo (v, n)

```
1. maior ← 0          segundo_maior ← 0
2. para i ← 1 até n faça
3.     se v[i] > maior
4.           então segundo_maior ← maior
5.                 maior ← v[i]
6.           senão se v[i] > segundo_maior
7.                       então segundo_maior ← v[i]
8. devolva segundo_maior
```

Suponha que v é uma permutação de 1 a n escolhida ao acaso dentre todas as permutações de 1 a n, de acordo com a distribuição uniforme de probabilidade. Seja X o número de vezes que a variável *segundo_maior* é alterada (ou seja, o número de execuções das linhas 4 e 7 do algoritmo) numa chamada de Máximo(v, n). Note que X é uma variável aleatória. Calcule o valor esperado de X.

6. Considere o seguinte algoritmo que calcula o maior e o menor elemento de um vetor $v[1\,..\,n]$ com elementos distintos.

Algoritmo MaiorMenor (v, n)

```
1. maior ← v[1]          menor ← v[1]
2. para i ← 2 até n faça
3.     se v[i] > maior
4.           então maior ← v[i]
5.           senão se v[i] < menor
6.                       então menor ← v[i]
7. devolva maior, menor
```

Suponha que a entrada do algoritmo é uma permutação de 1 a n escolhida uniformemente dentre todas as permutações de 1 a n.

Qual é o número esperado de comparações executadas na linha 5 do algoritmo? Qual é o número esperado de atribuições efetuadas na linha 6 do algoritmo? Justifique a sua resposta.