# MAC0338 - ANÁLISE DE ALGORITMOS

## LISTA 3

### CHECKLIST

entregue fiz incompleto não entendi

questão 1

questão 2

questão 3

questão 4

questão 5

questão 6

questão 7

questão 8

questão 9

questão 10

questão 11

Nome: Sabrina Araújo da Silva    nº USP: 12566182

# MAC0338 – ANÁLISE DE ALGORITMOS

## LISTA 03

exercícios 1, 6 e 8

1. 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
2. segundo_maior ← 0
3. para i ← 1 até n faça
4.     se v[i] > maior
5.         então segundo_maior ← maior
6.               maior ← v[i]
7.         senão se v[i] > segundo_maior
8.                   então segundo_maior ← v[i]
9. 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 5 e 8 do algoritmo) numa chamada de Máximo$(v, n)$. Note que $X$ é uma variável aleatória. Calcule o valor esperado de $X$.

cada permutação de v tem probabilidade $\frac{1}{n!}$

$X$ = número total de execuções da linha 5 e 8 $= Y + W$

$Y$ = número total de execuções da linha 5

$W$ = número total de execuções da linha 8

$Y_i = \begin{cases} 1, \text{ se "segundo\_maior} \leftarrow \text{maior" é executado} \\ 0, \text{ caso contrário} \end{cases}$

$Y = Y_1 + \ldots + Y_n$

$E[Y_i]$ = probabilidade de que $v[i]$ seja máximo em $v[1 \ldots i]$

$= \frac{1}{i}$

$E[Y] = E[Y_1 + \ldots + Y_n] = E[Y_1] + \ldots + E[Y_n]$

$= \frac{1}{2} + \ldots + \frac{1}{n} < \ln n$

$W_i = \begin{cases} 1, \text{ se "segundo\_maior} \leftarrow v[i]\text{" é executado} \\ 0, \text{ caso contrário} \end{cases}$

$W = W_1 + \dots + W_n$

$E[W_i]$ = probabilidade de que $v[i]$ seja o segundo máximo em $v[1 \dots i]$

$$= \underbrace{\left(1 - \frac{1}{i}\right)}_{\text{prob de não executar a linha 5}} \cdot \frac{1}{i-1} = \frac{1}{i}$$

$$E[W] = E[W_1 + \dots + W_n] = E[W_1] + \dots + E[W_n]$$

$$= \frac{1}{2} + \dots + \frac{1}{n} < \ln n$$

$$E[X] = E[Y] + E[W] < \ln n + \ln n$$

$$< 2 \ln n$$

2. 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* $\leftarrow v[1]$
2. *menor* $\leftarrow v[1]$
3. **para** $i \leftarrow 2$ **até** $n$ **faça**
4.     **se** $v[i] >$ *maior*
5.         **então** *maior* $\leftarrow v[i]$
6.         **senão se** $v[i] <$ *menor*
7.             **então** *menor* $\leftarrow v[i]$
8. **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 6 do algoritmo? Qual é o número esperado de atribuições efetuadas na linha 7 do algoritmo?

Número esperado de comparações executadas na linha 6 do algoritmo:

$X$ = número total de execuções da linha 6

$$X_i = \begin{cases} 1 & \text{se } v[i] \leq \text{maior} \\ 0 & \text{caso contrário} \end{cases}$$

$X = X_1 + \cdots + X_n$

$E[X_i]$ = probabilidade de que $v[i] \leq$ maior

$= \left(1 - \frac{1}{i}\right)$

$E[X] = E[X_1 + \dots + X_n]$

$$= \sum_{i=2}^{n} 1 - \frac{1}{i} = \sum_{i=2}^{n} 1 - \sum_{i=2}^{n} \frac{1}{i} < n - \ln n = O(n)$$

Número esperado de atribuições efetuadas na linha 7

$Y$ = número total de execuções da linha 7

$$Y_i = \begin{cases} 1 & \text{se } v[i] < \text{menor} \\ 0 & \text{caso contrário} \end{cases}$$

$Y = Y_1 + \cdots + Y_n$

$E[Y_i]$ = probabilidade de $v[i] \leq$ maior e $v[i] <$ menor

$$= \left(1 - \frac{1}{i}\right) \cdot \frac{1}{i} = \frac{i-1}{i^2}$$

$E[Y] = E[Y_1 + \cdots + Y_n]$

$$= \sum_{i=2}^{n} \frac{i-1}{i^2} = \frac{1}{4} + \frac{2}{9} + \frac{3}{16} + \cdots + \frac{i-1}{i^2} < \ln n = O(\ln n)$$

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

- A complexidade de espaço é de ordem (log n)
- Uma vez que todas as operações são feitas in-place, ou seja, diretamente no vetor e que a cada particionamento temos um vetor menor a ser ordenado.

6. Escreva uma função que recebe um vetor com $n$ letras A's e B's e, por meio de trocas, move todos os A's para o início do vetor. Sua função deve consumir tempo $O(n)$.

seja v[1 ... n] o vetor com n letras A's e B's

```
TROCA(v, n)
    j ← n
    PARA i ← 1 ENQUANTO i < j FAÇA
        SE v[i] == "B" E v[j] == "A"
            v[i] = "A"
            v[j] = "B"
            j = j - 1
        SENÃO SE v[i] = "A" E v[j] == "B"
            j = j - 1
        SENÃO SE v[i] = "B" E v[j] == "B"
            j = j - 1
            i = i - 1
        i = i + 1
    DEVOLVA v
```

7. Escreva uma função que rearranje um vetor $v[p \,..\, r]$ de inteiros de modo que tenhamos $v[p \,..\, j-1] \leq 0$ e $v[j \,...\, r] > 0$ para algum $j$ em $p \,..\, r+1$. Faz sentido exigir que $j$ esteja em $p \,..\, r$? Procure fazer uma função rápida que não use vetor auxiliar. Repita o exercício depois de trocar $v[j \,..\, r] > 0$ por $v[j \,..\, r] \geq 0$. Faz sentido exigir que $v[j]$ seja 0?

2) não

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

seja X [ ix ... fx ] e Y [ iy ... fy ]

```
MEDIANA (X, Y, ix, fx, iy, fy)
    SE ix É IGUAL A iy
        DEVOLVA O MÍNIMO ENTRE ix E iy
    // MEDIANAS DE X E Y
    mx = (ix + fx) / 2
    my = (iy + fy) / 2
    x = X[i]
    y = Y[j]
    SE x == y
        DEVOLVA x
    SE x < y
        DEVOLVA MEDIANA (X, Y, mx, fx, iy, my)
    SE y < x
        DEVOLVA MEDIANA (X, Y, ix, mx, my, fy)
```

9. Para esta questão, vamos dizer que a mediana de um vetor $A[p\,..\,r]$ com número inteiros é o valor que ficaria na posição $A[\lfloor (p+r)/2 \rfloor]$ depois que o vetor $A[p\,..\,r]$ fosse ordenado.

   Dado um algoritmo linear "caixa-preta" que devolve a mediana de um vetor, descreva um algoritmo simples, linear, que, dado um vetor $A[p\,..\,r]$ de inteiros distintos e um inteiro $k$, devolve o $k$-ésimo mínimo do vetor. (O $k$-ésimo mínimo de um vetor de inteiros distintos é o elemento que estaria na $k$-ésima posição do vetor se ele fosse ordenado.)

KESIMO (A, K, r)

ideia do select aleatorizado
porém na chamada do particione
o pivô não é aleatório, mas sim
a mediana que o algoritmo
caixa-preta encontra.

PARTICIONE (A, p, r)

$x \leftarrow$ CAIXA PRETA (A)

$i \leftarrow p-1$

...

10. **(CLRS 8.3-2)** Quais dos seguintes algoritmos de ordenação são estáveis: insertionsort, mergesort, heapsort, e quicksort. Descreva uma maneira simples de deixar qualquer algoritmo de ordenação estável. Quanto tempo e/ou espaço adicional a sua estratégia usa?

estáveis: insertion sort, mergesort

instável: heapsort, quicksort

- qualquer algoritmo pode ser estável, mas há um preço
  - exemplo: armazenar os índices originais em um vetor e ordernar comparando com esses índices

  gasta tempo e espaço adicional de ordem (n)

11. Qual a diferença de consumo de tempo entre uma busca binária em um vetor com $n$ elementos e uma busca binária em um vetor com $n^2$ elementos?

- O consumo de tempo entre uma busca binária em um vetor com n elementos e uma busca binária em um vetor com $n^2$ elementos é proporcional a lg n, pois lg transforma multiplicações em somas.
- Por exemplo, se a busca em um vetor de tamanho n exige C comparações, então a busca em um vetor de tamanho 2n exigirá apenas C + 1 comparações, em um vetor de tamanho 100n exigirá menos que C + 7.