Quicksort e Select Aleatorizados

CLRS Secs 7.3, 7.4 e 9.2

Relembremos o Particione

Rearranja $A[p\,..\,r]$ de modo que $p \leq q \leq r$ e
$A[p\,..\,q{-}1] \leq A[q] < A[q{+}1\,..\,r]$

PARTICIONE (A, p, r)
1 $x \leftarrow A[r]$ $\triangleright$ x é o "pivô"
2 $i \leftarrow p{-}1$
3 **para** $j \leftarrow p$ **até** $r-1$ **faça**
4   **se** $A[j] \leq x$
5   **então** $i \leftarrow i+1$
6                 $A[i] \leftrightarrow A[j]$
7 $A[i{+}1] \leftrightarrow A[r]$
8 **devolva** $i+1$

Invariantes: no começo de cada iteração de 3–6,

(i0) $A[p\,..\,i] \leq x$ (i1) $A[i{+}1\,..\,j{-}1] > x$ (i2) $A[r] = x$

Relembremos o Particione

Rearranja $A[p\,..\,r]$ de modo que $p \le q \le r$ e
$A[p\,..\,q{-}1] \le A[q] < A[q{+}1\,..\,r]$

```
PARTICIONE (A, p, r)
x ← A[r]          ▷ x é o "pivô"
i ← p−1
para j ← p até r − 1 faça
    se A[j] ≤ x
        então i ← i + 1
              A[i] ↔ A[j]
A[i+1] ↔ A[r]
devolva i + 1
```

Consumo de tempo: $\Theta(n)$ onde $n := r - p$.

Quicksort aleatorizado

PARTICIONE-ALEA(A, p, r)
1 $i \leftarrow$ RANDOM (p, r)
2 $A[i] \leftrightarrow A[r]$
3 **devolva** PARTICIONE (A, p, r)

Quicksort aleatorizado

```
PARTICIONE-ALEA(A, p, r)
    i ← RANDOM(p, r)
    A[i] ↔ A[r]
    devolva PARTICIONE(A, p, r)
```

```
QUICKSORTALEAT (A, p, r)
    se p < r
        então q ← PARTICIONE-ALEA(A, p, r)
              QUICKSORTALEAT(A, p, q − 1)
              QUICKSORTALEAT(A, q + 1, r)
```

Quicksort aleatorizado

PARTICIONE-ALEA(A, p, r)
1 $i \leftarrow$ RANDOM(p, r)
2 $A[i] \leftrightarrow A[r]$
3 **devolva** PARTICIONE(A, p, r)

QUICKSORTALEAT (A, p, r)
1 **se** $p < r$
2 **então** $q \leftarrow$ PARTICIONE-ALEA(A, p, r)
3 QUICKSORTALEAT$(A, p, q-1)$
4 QUICKSORTALEAT$(A, q+1, r)$

Para um vetor em que todos os elementos são iguais, qual é o consumo de tempo do QUICKSORTALEAT?

Quicksort aleatorizado

```
PARTICIONE-ALEA(A, p, r)
    i ← RANDOM(p, r)
    A[i] ↔ A[r]
    devolva PARTICIONE(A, p, r)
```

```
QUICKSORTALEAT (A, p, r)
    se p < r
        então q ← PARTICIONE-ALEA(A, p, r)
              QUICKSORTALEAT(A, p, q − 1)
              QUICKSORTALEAT(A, q + 1, r)
```

Para um vetor em que todos os elementos são iguais, qual é o consumo de tempo do QUICKSORTALEAT?

Para um vetor A sem repetições, qual é o consumo esperado de tempo?

Consumo esperado de tempo

Basta contar o número esperado de comparações na linha 4 do PARTICIONE.

PARTICIONE (A, p, r)
1 $x \leftarrow A[r]$ $\triangleright$ x é o "pivô"
2 $i \leftarrow p - 1$
3 **para** $j \leftarrow p$ **até** $r - 1$ **faça**
4   **se** $A[j] \leq x$
5   **então** $i \leftarrow i + 1$
6                 $A[i] \leftrightarrow A[j]$
7 $A[i+1] \leftrightarrow A[r]$
8 **devolva** $i + 1$

Consumo de tempo esperado

A: vetor com n números, sem repetição

$$X_{ab} \quad = \quad \text{número de comparações entre o } a\text{-ésimo e o } b\text{-ésimo menor número de } A \text{ na linha 4 do } \textsc{Particione} \text{ do } \textsc{QuicksortAleat}$$

Queremos calcular

$$\begin{aligned} X &= \text{total de comparações } \text{“}A[j] \leq x\text{”} \\ &= \sum_{a=1}^{n-1} \sum_{b=a+1}^{n} X_{ab} \end{aligned}$$

Consumo de tempo esperado

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se o primeiro pivô que, para } i \text{ em } \{a, \ldots, b\}, \\ & \text{é o } i\text{-ésimo número do vetor } A \text{ ocorre com } i = a \text{ ou } i = b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

Consumo de tempo esperado

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se o primeiro pivô que, para } i \text{ em } \{a, \ldots, b\}, \\ & \text{é o } i\text{-ésimo número do vetor } A \text{ ocorre com } i = a \text{ ou } i = b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

Consumo de tempo esperado

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se o primeiro pivô que, para } i \text{ em } \{a, \ldots, b\}, \\ & \text{é o } i\text{-ésimo número do vetor } A \text{ ocorre com } i = a \text{ ou } i = b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

$$\Pr\{X_{ab}=1\} \;=\; \frac{2}{b-a+1} \;=\; \mathrm{E}[X_{ab}]$$

Consumo de tempo esperado

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se o primeiro pivô que, para } i \text{ em } \{a, \ldots, b\}, \\ & \text{é o } i\text{-ésimo número do vetor } A \text{ ocorre com } i = a \text{ ou } i = b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

$$\Pr\{X_{ab}=1\} \;=\; \frac{2}{b-a+1} \;=\; \mathrm{E}[X_{ab}]$$

Novamente $X = \displaystyle\sum_{a=1}^{n-1} \sum_{b=a+1}^{n} X_{ab}$.

Quanto vale $\mathrm{E}[X]$?

Consumo de tempo esperado

$$
\begin{aligned}
\mathrm{E}[X] &= \sum_{a=1}^{n-1} \sum_{b=a+1}^{n} \mathrm{E}[X_{ab}] \\
&= \sum_{a=1}^{n-1} \sum_{b=a+1}^{n} \Pr\{X_{ab}=1\} \\
&= \sum_{a=1}^{n-1} \sum_{b=a+1}^{n} \frac{2}{b-a+1} \\
&= \sum_{a=1}^{n-1} \sum_{k=1}^{n-a} \frac{2}{k+1} \\
&< \sum_{a=1}^{n-1} 2\left(\frac{1}{1}+\frac{1}{2}+\cdots+\frac{1}{n}\right) \\
&< 2n\left(\frac{1}{1}+\frac{1}{2}+\cdots+\frac{1}{n}\right) < 2n\,(1+\ln n)
\end{aligned}
$$

CLRS (A.7), p.1060

Conclusões

O consumo de tempo esperado do algoritmo
QUICKSORTALEAT é $O(n \log n)$.

Do exercício 7.4-4 do CLRS temos que

O consumo de tempo esperado do algoritmo
QUICKSORTALEAT é $\Theta(n \log n)$.

k-ésimo menor elemento

CLRS 9

k-ésimo menor

Problema: Encontrar o k-ésimo menor elemento de $A[1\,..\,n]$.

Suponha $A[1\,..\,n]$ sem elementos repetidos.

Exemplo: 33 é o 4o. menor elemento de:

1									10	
22	99	32	88	34	**33**	11	97	55	66	A

1			4						10	
11	22	32	**33**	34	55	66	88	97	99	ordenado

Mediana

Mediana é o $\left\lfloor \frac{n+1}{2} \right\rfloor$-ésimo menor ou o $\left\lceil \frac{n+1}{2} \right\rceil$-ésimo menor elemento.

Exemplo: a mediana é 34 ou 55:

1									10	
22	99	32	88	**34**	33	11	97	**55**	66	A

1				5	6				10	
11	22	32	33	**34**	**55**	66	88	97	99	ordenado

k-ésimo menor

Recebe $A[1\,..\,n]$ e k tal que $1 \leq k \leq n$
e devolve valor do k-ésimo menor elemento de $A[1\,..\,n]$.

SELECT-ORD (A, n, k)
1 ORDENE (A, n)
2 **devolva** $A[k]$

O consumo de tempo do SELECT-ORD é $\Theta(n \lg n)$.

k-ésimo menor

Recebe $A[1\,..\,n]$ e k tal que $1 \leq k \leq n$
e devolve valor do k-ésimo menor elemento de $A[1\,..\,n]$.

SELECT-ORD (A, n, k)
1 ORDENE (A, n)
2 **devolva** $A[k]$

O consumo de tempo do SELECT-ORD é $\Theta(n \lg n)$.

Dá para fazer melhor?

Menor

Recebe um vetor $A[1\,..\,n]$ e devolve o valor do menor elemento.

```
MENOR (A, n)
menor ← A[1]
para k ← 2 até n faça
    se A[k] < menor
        então menor ← A[k]
devolva menor
```

$\boxed{\text{O consumo de tempo do algoritmo MENOR é } \Theta(n).}$

Segundo menor

Recebe um vetor $A[1\,..\,n]$ e devolve
o valor do segundo menor elemento, supondo $n \geq 2$.

```
SEG-MENOR (A, n)
menor ← min{A[1], A[2]}     segmenor ← max{A[1], A[2]}
para k ← 3 até n faça
    se A[k] < menor
        então segmenor ← menor
              menor ← A[k]
    senão se A[k] < segmenor
        então segmenor ← A[k]
devolva segmenor
```

Segundo menor

Recebe um vetor $A[1\,..\,n]$ e devolve
o valor do segundo menor elemento, supondo $n \geq 2$.

```
SEG-MENOR (A, n)
menor ← min{A[1], A[2]}     segmenor ← max{A[1], A[2]}
para k ← 3 até n faça
    se A[k] < menor
        então segmenor ← menor
              menor ← A[k]
    senão se A[k] < segmenor
        então segmenor ← A[k]
devolva segmenor
```

O consumo de tempo do SEG-MENOR é $\Theta(n)$.

Algoritmo linear?

Será que conseguimos fazer um algoritmo linear

para a mediana?

para o k-ésimo menor?

Algoritmo linear?

Será que conseguimos fazer um algoritmo linear

para a mediana?

para o k-ésimo menor?

Sim!

Usaremos o PARTICIONE do QUICKSORT!

Select aleatorizado

PARTICIONE-ALEA(A, p, r)
$k \leftarrow$ RANDOM (p, r)
$A[k] \leftrightarrow A[r]$
devolva PARTICIONE (A, p, r)

SELECT-ALEA (A, p, r, k)
se $p = r$ **então devolva** $A[p]$
$q \leftarrow$ PARTICIONE-ALEA (A, p, r)
se $k = q - p + 1$
    então devolva $A[q]$

Select aleatorizado

```
PARTICIONE-ALEA(A, p, r)
    k ← RANDOM(p, r)
    A[k] ↔ A[r]
    devolva PARTICIONE(A, p, r)

SELECT-ALEA (A, p, r, k)
    se p = r então devolva A[p]
    q ← PARTICIONE-ALEA(A, p, r)
    se k = q − p + 1
        então devolva A[q]
    se k < q − p + 1
        então devolva SELECT-ALEA(A, p, q − 1, k)
```

Select aleatorizado

```
PARTICIONE-ALEA(A, p, r)
    k ← RANDOM(p, r)
    A[k] ↔ A[r]
    devolva PARTICIONE(A, p, r)
```

```
SELECT-ALEA (A, p, r, k)
    se p = r então devolva A[p]
    q ← PARTICIONE-ALEA(A, p, r)
    se k = q − p + 1
        então devolva A[q]
    se k < q − p + 1
        então devolva SELECT-ALEA(A, p, q − 1, k)
        senão devolva SELECT-ALEA(A, q+1, r, k−(q−p+1))
```

Select aleatorizado

PARTICIONE-ALEA(A, p, r)

$k \leftarrow$ RANDOM(p, r)
$A[k] \leftrightarrow A[r]$
devolva PARTICIONE(A, p, r)

SELECT-ALEA (A, p, r, k)

se $p = r$ **então devolva** $A[p]$
$q \leftarrow$ PARTICIONE-ALEA(A, p, r)
se $k = q - p + 1$
  então devolva $A[q]$
se $k < q - p + 1$
  então devolva SELECT-ALEA$(A, p, q - 1, k)$
  senão devolva SELECT-ALEA$(A, q{+}1, r, k{-}(q{-}p{+}1))$

Para um vetor A sem repetições,
qual é o consumo esperado de tempo?

Consumo de tempo esperado

Podemos assumir, sem perda de generalidade, que o vetor A é uma permutação de 1 a n.

Para inteiros a e b tais que $1 \leq a < b \leq n$, defina

$$X_{ab} = \text{número de comparações entre } a \text{ e } b \text{ na linha 4 do } \textsc{Particione} \text{ do } \textsc{Select-Alea}.$$

Observe que X_{ab} não é a mesma de antes, pois o algoritmo para a qual é definida é outro.

De novo, queremos calcular

$$\begin{aligned} X &= \text{total de comparações "} A[j] \leq x \text{"} \\ &= \sum_{a=1}^{n-1} \sum_{b=a+1}^{n} X_{ab} \end{aligned}$$

Consumo de tempo esperado

Vamos supor que $k = n$.

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se} \\ & \end{cases}$$

Consumo de tempo esperado

Vamos supor que $k = n$.

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se primeiro pivô em } \{a, \ldots, n\} \text{ é } a \text{ ou } b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

Consumo de tempo esperado

Vamos supor que $k = n$.

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se primeiro pivô em } \{a, \ldots, n\} \text{ é } a \text{ ou } b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

$$\Pr\{X_{ab}=1\} \;=\; \frac{2}{n-a+1} \;=\; \mathrm{E}[X_{ab}]$$

Consumo de tempo esperado

Vamos supor que $k = n$.

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se primeiro pivô em } \{a,\ldots,n\} \text{ é } a \text{ ou } b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

$$\Pr\{X_{ab}{=}1\} \;=\; \frac{2}{n-a+1} \;=\; \mathrm{E}[X_{ab}]$$

Como antes, $X = \sum_{a=1}^{n-1} \sum_{b=a+1}^{n} X_{ab}$.

$\mathrm{E}[X] = ????$

Consumo de tempo esperado

$$
\begin{aligned}
\mathrm{E}[X] &= \sum_{a=1}^{n-1} \sum_{b=a+1}^{n} \mathrm{E}[X_{ab}] = \sum_{a=1}^{n-1} \sum_{b=a+1}^{n} \Pr\{X_{ab}=1\} \\
&= \sum_{a=1}^{n-1} \sum_{b=a+1}^{n} \frac{2}{n-a+1}
\end{aligned}
$$

Consumo de tempo esperado

$$
\begin{aligned}
\mathrm{E}[X] &= \sum_{a=1}^{n-1} \sum_{b=a+1}^{n} \mathrm{E}[X_{ab}] = \sum_{a=1}^{n-1} \sum_{b=a+1}^{n} \Pr\{X_{ab}=1\} \\
&= \sum_{a=1}^{n-1} \sum_{b=a+1}^{n} \frac{2}{n-a+1} \\
&= \sum_{a=1}^{n-1} \frac{2(n-a)}{n-a+1} \\
&< \sum_{a=1}^{n-1} 2 < 2n.
\end{aligned}
$$

Consumo de tempo esperado

$$\begin{aligned}
\mathrm{E}[X] &= \sum_{a=1}^{n-1}\sum_{b=a+1}^{n} \mathrm{E}[X_{ab}] = \sum_{a=1}^{n-1}\sum_{b=a+1}^{n} \Pr\{X_{ab}{=}1\} \\
&= \sum_{a=1}^{n-1}\sum_{b=a+1}^{n} \frac{2}{n-a+1} \\
&= \sum_{a=1}^{n-1} \frac{2(n-a)}{n-a+1} \\
&< \sum_{a=1}^{n-1} 2 \;<\; 2n.
\end{aligned}$$

Exercício: Refaça os cálculos para um k arbitrário.

Conclusões

O consumo de tempo esperado do algoritmo SELECT-ALEA é $\mathrm{O}(n)$.

Conclusões

O consumo de tempo esperado do
algoritmo SELECT-ALEA é $O(n)$.

Outros comentários:

- Espaço do quicksort, recursão de cauda.

Conclusões

O consumo de tempo esperado do
algoritmo SELECT-ALEA é $O(n)$.

Outros comentários:

- Espaço do quicksort, recursão de cauda.
- Exercícios das últimas listas.