AULA 12

Quicksort aleatorizado

CLRS 7.4

Relembrar ...

Antes de mais nada, vamos relembrar rapidamente os melhores momentos da aula 9:

- PARTICIONE;
- consumo de tempo do PARTICIONE;
- QUICKSORT; e
- consumo de tempo do QUICKSORT.

Partição

Problema: Rearranjar um dado vetor $A[p\,.\,.\,r]$ e devolver um índice q, $p \leq q \leq r$, tais que

$$A[p\,.\,.\,q-1] \leq A[q] < A[q+1\,.\,.\,r]$$

Entra:

	p									r
A	99	33	55	77	11	22	88	66	33	44

Sai:

	p				q					r
A	33	11	22	33	44	55	99	66	77	88

Particione

	p									r
A	99	33	55	77	11	22	88	66	33	44

Particione

	i	j									x
A		99	33	55	77	11	22	88	66	33	44

Particione

	i	j								x
A	99	33	55	77	11	22	88	66	33	44

Particione

$$
\begin{array}{cc|c|c|c|c|c|c|c|c|c|c|}
 & \color{red}{i} & & \color{blue}{j} & & & & & & & & \color{magenta}{x} \\
\cline{3-12}
A & & \color{blue}{99} & 33 & 55 & 77 & 11 & 22 & 88 & 66 & 33 & \color{magenta}{44} \\
\cline{3-12}
\end{array}
$$

$$
\begin{array}{cc|c|c|c|c|c|c|c|c|c|c|}
 & & \color{red}{i} & & \color{blue}{j} & & & & & & & \color{magenta}{x} \\
\cline{3-12}
A & & \color{red}{33} & \color{blue}{99} & 55 & 77 & 11 & 22 & 88 & 66 & 33 & \color{magenta}{44} \\
\cline{3-12}
\end{array}
$$

Particione

i		j								x
A	99	33	55	77	11	22	88	66	33	44

	i		j							x
A	33	99	55	77	11	22	88	66	33	44

	i			j						x
A	33	99	55	77	11	22	88	66	33	44

Particione

i		j								x
A	99	33	55	77	11	22	88	66	33	44

	i		j							x
A	33	99	55	77	11	22	88	66	33	44

	i				j					x
A	33	99	55	77	11	22	88	66	33	44

Particione

i		j								x
A	99	33	55	77	11	22	88	66	33	44

	i		j							x
A	33	99	55	77	11	22	88	66	33	44

		i				j				x
A	33	11	55	77	99	22	88	66	33	44

Particione

i		j								x
A	99	33	55	77	11	22	88	66	33	44

	i		j							x
A	33	99	55	77	11	22	88	66	33	44

		i				j				x
A	33	11	55	77	99	22	88	66	33	44

			i				j			x
A	33	11	22	77	99	55	88	66	33	44

Particione

i	j									x
A	99	33	55	77	11	22	88	66	33	44

	i	j								x
A	33	99	55	77	11	22	88	66	33	44

		i				j				x
A	33	11	55	77	99	22	88	66	33	44

			i				j			x
A	33	11	22	77	99	55	88	66	33	44

			i					j		x
A	33	11	22	77	99	55	88	66	33	44

Particione

i		j								x
A	99	33	55	77	11	22	88	66	33	44

	i		j							x
A	33	99	55	77	11	22	88	66	33	44

		i				j				x
A	33	11	55	77	99	22	88	66	33	44

			i				j			x
A	33	11	22	77	99	55	88	66	33	44

			i						j	x
A	33	11	22	77	99	55	88	66	33	44

Particione

i		j								x
A	99	33	55	77	11	22	88	66	33	44

	i		j							x
A	33	99	55	77	11	22	88	66	33	44

		i				j				x
A	33	11	55	77	99	22	88	66	33	44

			i				j			x
A	33	11	22	77	99	55	88	66	33	44

				i						j
A	33	11	22	33	99	55	88	66	77	44

Particione

i		j								x
A	99	33	55	77	11	22	88	66	33	44
	i		j							x
A	33	99	55	77	11	22	88	66	33	44
		i				j				x
A	33	11	55	77	99	22	88	66	33	44
			i				j			x
A	33	11	22	77	99	55	88	66	33	44
				i						j
A	33	11	22	33	99	55	88	66	77	44
	p				q					r
A	33	11	22	33	44	55	88	66	77	99

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

O algoritmo PARTICIONE consome tempo $\Theta(n)$.

Quicksort

Rearranja $A[p \mathinner{.\,.} r]$ em ordem crescente.

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

	p									r
A	99	33	55	77	11	22	88	66	33	44

Quicksort

Rearranja $A[p\,.\,.\,r]$ em ordem crescente.

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

	p				q					r
A	33	11	22	33	44	55	88	66	77	99

No começo da linha 3,

$$A[p\,.\,.\,q-1] \leq A[q] \leq A[q+1\,.\,.\,r]$$

Quicksort

Rearranja $A[p\,..\,r]$ em ordem crescente.

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

	p				q					r
A	11	22	33	33	44	55	88	66	77	99

Quicksort

Rearranja $A[p \mathinner{.\,.} r]$ em ordem crescente.

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

	p				q					r
A	11	22	33	33	44	55	66	77	88	99

O consumo de tempo é proporcional ao número de execuções da linha 4 do PARTICIONE.

Resumo

O consumo de tempo do QUICKSORT é $O(n^2)$.

O consumo de tempo do QUICKSORT no melhor caso é $\Theta(n \log n)$.

Caso médio

O consumo de tempo do QUICKSORT no caso médio é $???$.

Partição $\frac{1}{3}$ para $\frac{2}{3}$:

$$R(n) = R(\lfloor \tfrac{n-1}{3} \rfloor) + R(\lceil \tfrac{2n-2}{3} \rceil) + \Theta(n)$$

Solução: $R(n)$ é $\Theta(n \lg n)$. veja exercício a seguir

Exercício

Considere a recorrência

$$T(1) = 1$$

$$T(n) = T(\lceil n/3 \rceil) + T(\lfloor 2n/3 \rfloor) + 5n$$

para $n = 2, 3, 4, \ldots$

Solução assintótica: $T(n)$ é $\mathrm{O}(???)$, $T(n)$ é $\Theta(???)$

Exercício

Considere a recorrência

$$T(1) = 1$$

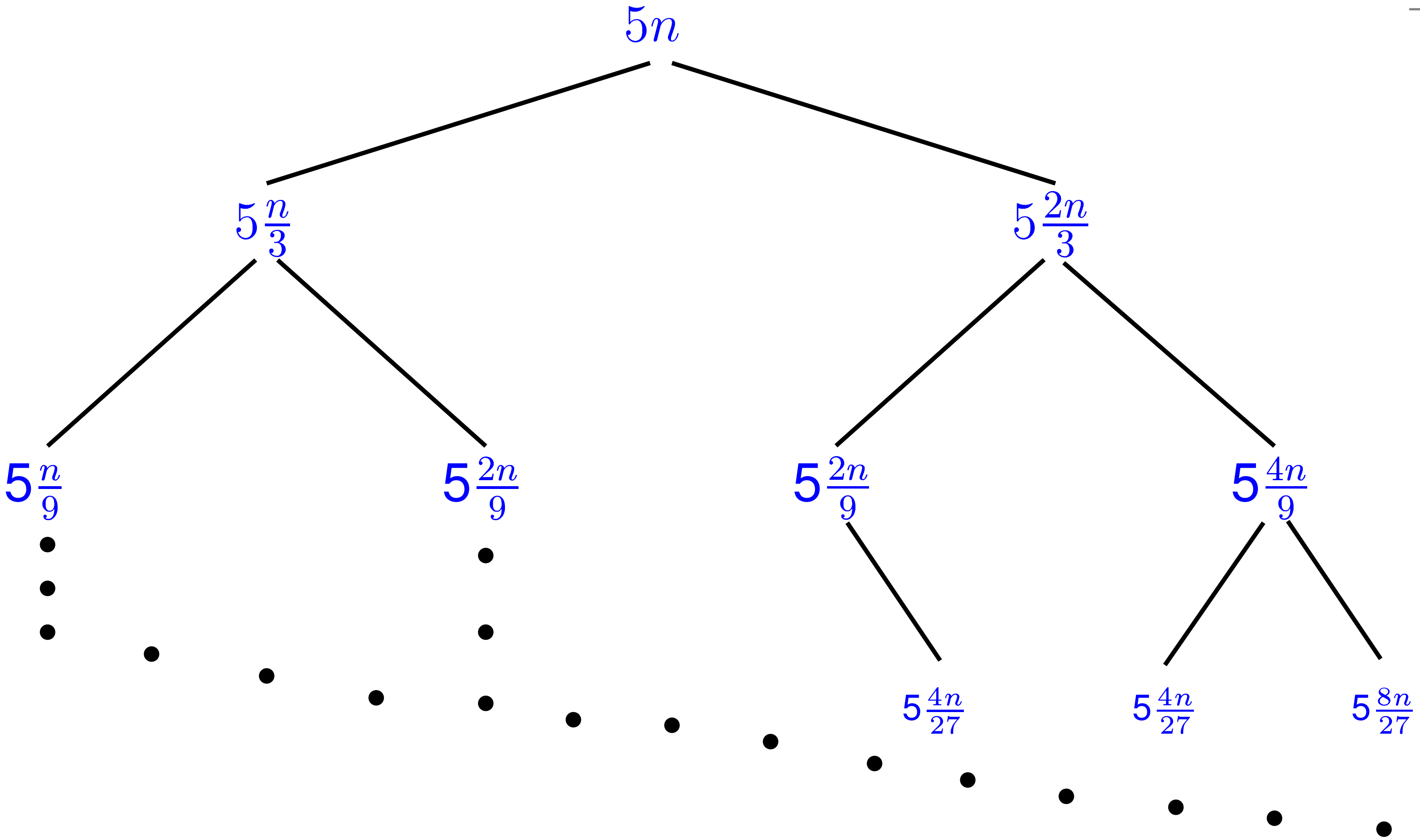
$$T(n) = T(\lceil n/3 \rceil) + T(\lfloor 2n/3 \rfloor) + 5n$$

para $n = 2, 3, 4, \ldots$

Solução assintótica: $T(n)$ é $\mathrm{O}(???)$, $T(n)$ é $\Theta(???)$

Vamos olhar a árvore da recorrência.

Arvoré da recorrência



total de níveis $\leq \log_{3/2} n$

Arvoré da recorrência

soma em cada horizontal $\leq 5n$

número de “níveis” $\leq \log_{3/2} n$

$T(n)$ = a soma de tudo

$$T(n) \;\leq\; 5n \log_{3/2} n \;+\; \underbrace{1 + \cdots + 1}_{\log_{3/2} n}$$

$T(n)$ é $\mathrm{O}(n \lg n)$.

De volta a recorrência

$$T(1) = 1$$

$$T(n) = T(\lceil n/3 \rceil) + T(\lfloor 2n/3 \rfloor) + 5n$$

para $n = 2, 3, 4, \ldots$

n	$T(n)$
1	1
2	$1 + 1 + 5 \cdot 2 = 12$
3	$1 + 12 + 5 \cdot 3 = 28$
4	$12 + 12 + 5 \cdot 4 = 44$

Vamos mostrar que $T(n) \le 100\, n \lg n$ para $n = 2, 3, 4, 5, 6, \ldots$
Para $n = 2$ temos $T(2) = 12 < 100 \cdot 2 \cdot \lg 2$.
Para $n = 3$ temos $T(3) = 28 < 100 \cdot 3 \cdot \lg 3$.

Suponha agora que $n > 3$. Então

Continuação da prova

$$
\begin{aligned}
T(n) &= T(\lceil \frac{n}{3} \rceil) + T(\lfloor \frac{2n}{3} \rfloor) + 5n \\
&\overset{\text{hi}}{\leq} 100\lceil \frac{n}{3} \rceil \lg\lceil \frac{n}{3} \rceil + 100\lfloor \frac{2n}{3} \rfloor \lg\lfloor \frac{2n}{3} \rfloor + 5n \\
&\leq 100\frac{n+2}{3} \lceil \lg \frac{n}{3} \rceil + 100\frac{2n}{3} \lg \frac{2n}{3} + 5n \\
&< 100\frac{n+2}{3} (\lg \frac{n}{3} + 1) + 100\frac{2n}{3} \lg \frac{2n}{3} + 5n \\
&= 100\frac{n+2}{3} \lg \frac{2n}{3} + 100\frac{2n}{3} \lg \frac{2n}{3} + 5n \\
&= 100\frac{n}{3} \lg \frac{2n}{3} + 100\frac{2}{3} \lg \frac{2n}{3} + 100\frac{2n}{3} \lg \frac{2n}{3} + 5n
\end{aligned}
$$

Continuação da continuação da prova

$$
\begin{aligned}
&< \quad 100n \lg \frac{2n}{3} + 67 \lg \frac{2n}{3} + 5n \\
&= \quad 100n \lg n + 100n \lg \frac{2}{3} + 67 \lg n + 67 \lg \frac{2}{3} + 5n \\
&< \quad 100n \lg n + 100n(-0.58) + 67 \lg n + 67(-0.58) + 5n \\
&< \quad 100n \lg n - 58n + 67 \lg n - 38 + 5n \\
&= \quad 100n \lg n - 53n + 67 \lg n - 38 \\
&< \quad 100n \lg n
\end{aligned}
$$

Iiiééééésssss!

De volta ao caso médio

O consumo de tempo do QUICKSORT no caso médio é ???.

Partição $\frac{1}{10}$ para $\frac{9}{10}$:

$$R(n) = R(\lfloor \tfrac{n-1}{10} \rfloor) + R(\lceil \tfrac{9n-9}{10} \rceil) + \Theta(n)$$

De volta ao caso médio

O consumo de tempo do QUICKSORT no caso médio é $???$.

Partição $\frac{1}{10}$ para $\frac{9}{10}$:

$$R(n) = R(\lfloor \tfrac{n-1}{10} \rfloor) + R(\lceil \tfrac{9n-9}{10} \rceil) + \Theta(n)$$

Solução: $R(n)$ é $\Theta(n \lg n)$

De volta ao caso médio

O consumo de tempo do QUICKSORT no caso médio é $???$.

Partição $\frac{1}{10}$ para $\frac{9}{10}$:

$$R(n) = R(\lfloor \tfrac{n-1}{10} \rfloor) + R(\lceil \tfrac{9n-9}{10} \rceil) + \Theta(n)$$

Solução: $R(n)$ é $\Theta(n \lg n)$

Isso sugere que consumo médio é $\Theta(n \lg n)$.

Confirmação?

Exemplos

Número médio de execuções da linha 4 do PARTICIONE.

Suponha que $A[p \mathinner{.\,.} r]$ é permutação de $1 \mathinner{.\,.} n$.

$A[p \mathinner{.\,.} r]$	execs
1,2	1
2,1	1
média	1

$A[p \mathinner{.\,.} r]$	execs
1,2,3	2+1
2,1,3	2+1
1,3,2	2+0
3,1,2	2+0
2,3,1	2+1
3,2,1	2+1
média	16/6

Mais um exemplo

$A[p\,..\,r]$	execs	$A[p\,..\,r]$	execs
1,2,3,4	3+3	1,3,4,2	3+1
2,1,3,4	3+3	3,1,4,2	3+1
1,3,2,4	3+2	1,4,3,2	3+1
3,1,2,4	3+2	4,1,3,2	3+1
2,3,1,4	3+3	3,4,1,2	3+1
3,2,1,4	3+3	4,3,1,2	3+1
1,2,4,3	3+1	2,3,4,1	3+3
2,1,4,3	3+1	3,2,4,1	3+3
1,4,2,3	3+1	2,4,3,1	3+2
4,1,2,3	3+1	4,2,3,1	3+2
2,4,1,3	3+1	3,4,2,1	3+3
4,2,1,3	3+1	4,3,2,1	3+3
		média	116/24

Quicksort aleatorizado

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

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

Análise do consumo medio?
Basta contar o número esperado de comparações na linha 4 do PARTICIONE

Exemplo

1	3	6	2	5	7	4

1	3	2	4	5	7	6

1	2	3	4	5	6	7

$$\begin{array}{c|ccccccc|} & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \hline 1 & & 1 & 0 & 1 & 0 & 0 & 0 \\ 2 & 1 & & 1 & 1 & 0 & 0 & 0 \\ 3 & 0 & 1 & & 1 & 0 & 0 & 0 \\ 4 & 1 & 1 & 1 & & 1 & 1 & 1 \\ 5 & 0 & 0 & 0 & 1 & & 1 & 0 \\ 6 & 0 & 0 & 0 & 1 & 1 & & 1 \\ 7 & 0 & 0 & 0 & 1 & 0 & 1 & \\ \hline \end{array}$$

Consumo de tempo esperado

Suponha $A[p \mathinner{.\,.} r]$ permutação de $1 \mathinner{.\,.} n$.

$$X_{ab} = \text{número de comparações entre } a \text{ e } b \text{ na linha 4 de PARTICIONE}$$

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

Supondo $a < b$,

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

Qual a probabilidade de X_{ab} valer 1?

Consumo de tempo esperado

Supondo $a < b$,

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

Qual a probabilidade de X_{ab} valer 1?

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

Consumo de tempo esperado

Supondo $a < b$,

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

Qual a probabilidade de X_{ab} valer 1?

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

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

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

Consumo de tempo esperado

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

CLRS (A.7), p.1060

Conclusões

O consumo de tempo esperado do algoritmo QUICKSORT-ALE é $\mathrm{O}(n \log n)$.

Do exercício 7.4-4 do CLRS temos que

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

Limites inferiores

CLRS 8.1

Máximo: limite inferior

Problema: Encontrar o maior elemento de um vetor $A[1\,..\,n]$.

Existe um algoritmo que faz o serviço com $n-1$ comparações.

Máximo: limite inferior

Problema: Encontrar o maior elemento de um vetor $A[1 \,.\,.\, n]$.

Existe um algoritmo que faz o serviço com $n - 1$ comparações.

Existe um algoritmo que faz menos comparações?

Máximo: limite inferior

Problema: Encontrar o maior elemento de um vetor $A[1\,..\,n]$.

Existe um algoritmo que faz o serviço com $n-1$ comparações.

Existe um algoritmo que faz menos comparações?

Não, se o algoritmo é baseado em comparações:

> dados dois número $A[i]$ e $A[j]$ podemos apenas compará-los a fim de encontrar o maior elemento.

Suponha dado um algoritmo baseado em comparações que resolve o problema.

Algoritmo baseado em comparações

O algoritmo consiste, no fundo, na determinação de uma coleção $\mathcal{A}$ de pares ou arcos $\langle i, j \rangle$ de elementos distintos em $\{1, \ldots, n\}$ tais que $A[i] < A[j]$ e existe um "sorvedouro".

Eis o paradigma de todo algoritmo baseado em comparações:

MAX (A, n)
    $\mathcal{A} \leftarrow \emptyset$
    enquanto $\mathcal{A}$ "não possui sorvedouro" **faça**
        escolha índice i e j em $\{1, \ldots, n\}$
        se $A[i] < A[j]$
            então $\mathcal{A} \leftarrow \mathcal{A} \cup \langle i, j \rangle$
            senão $\mathcal{A} \leftarrow \mathcal{A} \cup \langle j, i \rangle$
    devolva $\mathcal{A}$

Conclusão

Qualquer conjunto $\mathcal{A}$ devolvido pelo método contém uma "árvore enraizada" e portanto contém pelo menos $n-1$ arcos.

Qualquer algoritmo baseado em comparações que encontra o maior elemento de um vetor $A[1\,.\,.\,n]$ faz pelo menos $n-1$ comparações.

Ordenação: limite inferior

Problema: Rearranjar um vetor $A[1\,..\,n]$ de modo que ele fique em ordem crescente.

Existem algoritmos que consomem tempo $\mathrm{O}(n \lg n)$.

Ordenação: limite inferior

Problema: Rearranjar um vetor $A[1\,..\,n]$ de modo que ele fique em ordem crescente.

Existem algoritmos que consomem tempo $\mathrm{O}(n \lg n)$.

Existe algoritmo assintoticamente melhor?

Ordenação: limite inferior

Problema: Rearranjar um vetor $A[1\,..\,n]$ de modo que ele fique em ordem crescente.

Existem algoritmos que consomem tempo $\mathrm{O}(n \lg n)$.

Existe algoritmo assintoticamente melhor?

NÃO, se o algoritmo é baseado em comparações.

Prova?

Ordenação: limite inferior

Problema: Rearranjar um vetor $A[1\,..\,n]$ de modo que ele fique em ordem crescente.

Existem algoritmos que consomem tempo $\mathrm{O}(n \lg n)$.

Existe algoritmo assintoticamente melhor?

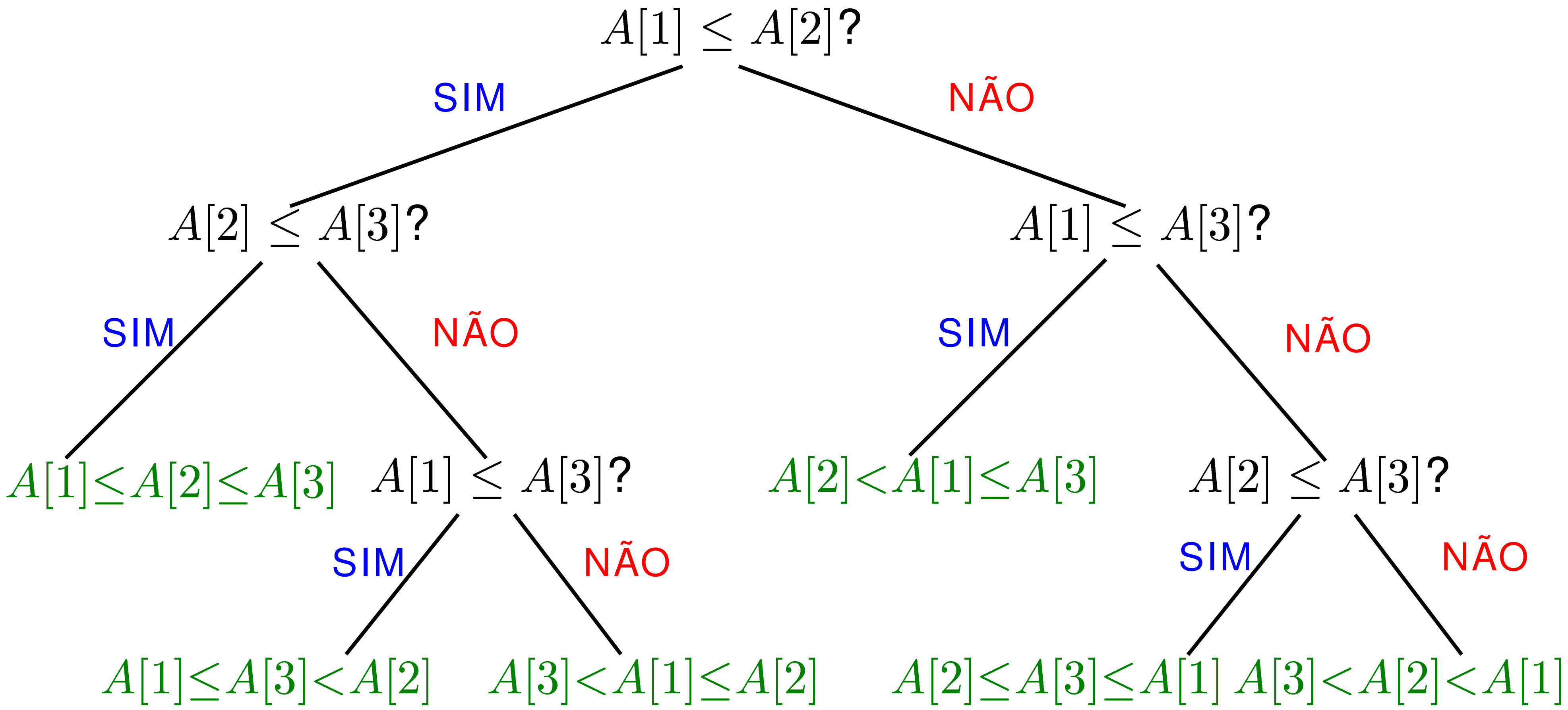
NÃO, se o algoritmo é baseado em comparações.

Prova?

Qualquer algoritmo baseado em comparações é uma "árvore de decisão".

Exemplo

ORDENA-POR-INSERÇÃO $(A[1\,.\,.\,3])$:



Limite inferior

Considere uma árvore de decisão para $A[1\mathinner{.\,.}n]$.

Limite inferior

Considere uma árvore de decisão para $A[1\,..\,n]$.

Número de comparações, no pior caso?

Limite inferior

Considere uma árvore de decisão para $A[1 \, . \, . \, n]$.

Número de comparações, no pior caso?
Resposta: altura, h, da árvore de decisão.

Limite inferior

Considere uma árvore de decisão para $A[1\,..\,n]$.

Número de comparações, no pior caso?
Resposta: altura, h, da árvore de decisão.

Todas as $n!$ permutações de $1, \ldots, n$ devem ser folhas.

Limite inferior

Considere uma árvore de decisão para $A[1\,..\,n]$.

Número de comparações, no pior caso?
Resposta: altura, h, da árvore de decisão.

Todas as $n!$ permutações de $1, \ldots, n$ devem ser folhas.

Toda árvore binária de altura h tem no máximo 2^h folhas.
Prova: Por indução em h. A afirmação vale para $h = 0$
Suponha que a afirmação vale para toda árvore binária de altura menor que h, $h \geq 1$.
O número de folhas de uma árvore de altura h é a soma do número de folhas de suas sub-árvores; que têm altura $\leq h - 1$. Logo, o número de folhas de uma árvore de altura h é não superior a

$$2 \times 2^{h-1} = 2^h.$$

Limite inferior

Logo, devemos ter $2^h \geq n!$, donde $h \geq \lg(n!)$.

$$(n!)^2 \; = \; \prod_{i=0}^{n-1} (n-i)(i+1) \; \geq \; \prod_{i=1}^{n} n \; = \; n^n$$

Portanto,

$$h \; \geq \; \lg(n!) \; \geq \; \frac{1}{2}\, n \lg n \,.$$

Conclusão

Todo algoritmo de ordenação baseado em comparações faz

$$\Omega(n \lg n)$$

comparações no pior caso

Exercícios

Exercício 16.A
Desenhe a árvore de decisão para o SELECTION-SORT aplicado a $A[1\,..\,3]$ com todos os elementos distintos.

Exercício 16.B [CLRS 8.1-1]
Qual o menor profundidade (= menor nível) que uma folha pode ter em uma árvore de decisão que descreve um algoritmo de ordenação baseado em comparações?

Exercício 16.C [CLRS 8.1-2]
Mostre que $\lg(n!) = \Omega(n \lg n)$ sem usar a fórmula de Stirling. Sugestão: Calcule $\sum_{k=n/2}^{n} \lg k$. Use as técnicas de CLRS A.2.