Análise de Algoritmos

Estes slides são adaptações de slides

do Prof. Paulo Feofiloff e do Prof. José Coelho de Pina.

i-ésimo menor elemento

CLRS 9

i-ésimo menor

Problema: Encontrar o i-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 $\lfloor \frac{n+1}{2} \rfloor$-ésimo menor ou o $\lceil \frac{n+1}{2} \rceil$-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

i-ésimo menor

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

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

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

i-ésimo menor

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

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

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)
- $\mathrm{menor} \leftarrow A[1]$
- **para** $k \leftarrow 2$ **até** n **faça**
    - **se** $A[k] < \mathrm{menor}$
        - **então** $\mathrm{menor} \leftarrow A[k]$
- **devolva** menor

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)

$\text{menor} \leftarrow \min\{A[1], A[2]\}$ $\quad$ $\text{segmenor} \leftarrow \max\{A[1], A[2]\}$
para $k \leftarrow 3$ **até** n **faça**
  se $A[k] < \text{menor}$
    então $\text{segmenor} \leftarrow \text{menor}$
      $\text{menor} \leftarrow A[k]$
  senão se $A[k] < \text{segmenor}$
    então $\text{segmenor} \leftarrow A[k]$
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 i-ésimo menor?

Algoritmo linear?

Será que conseguimos fazer um algoritmo linear

para a mediana?

para o i-ésimo menor?

Sim!

Usaremos o PARTICIONE do QUICKSORT!

Particione

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

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

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

Particione

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

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

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

Particione

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

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

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

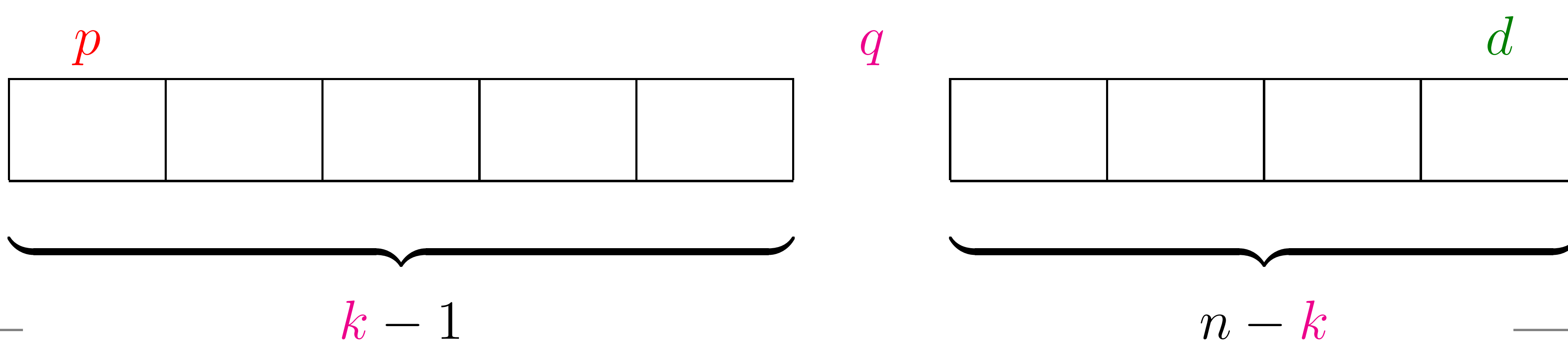
Algoritmo SELECT

Recebe $A[p \mathinner{.\,.} d]$ e i tal que $1 \leq i \leq d-p+1$
e devolve valor do i-ésimo menor elemento de $A[p \mathinner{.\,.} d]$.

SELECT(A, p, d, i)
    se $p = d$
        então devolva $A[p]$
    $q \leftarrow$ PARTICIONE (p, d)
    $k \leftarrow q - p + 1$
    se $k = i$
        então devolva $A[q]$
    se $k > i$
        então devolva SELECT $(A, p, q-1, i)$
        senão devolva SELECT $(A, q+1, d, i-k)$

Algoritmo SELECT

```
SELECT(A, p, d, i)
se p = d
    então devolva A[p]
q ← PARTICIONE (A, p, d)
k ← q − p + 1
se k = i
    então devolva A[q]
se k > i
    então devolva SELECT (A, p, q − 1, i)
    senão devolva SELECT (A, q + 1, d, i − k)
```



Consumo de tempo

$T(n)$ = consumo de tempo máximo quando $n = d - p + 1$

linha	consumo de todas as execuções da linha
1-2	$= \Theta(1)$
3	$= \Theta(n)$
4-7	$= \Theta(1)$
8	$= T(k-1)$
9	$= T(n-k)$

$$T(n) = \Theta(n) + \max\{T(k-1), T(n-k)\}$$

Consumo de tempo

$T(n)$ = consumo de tempo máximo quando $n = d - p + 1$

linha	consumo de todas as execuções da linha
1-2	$= \Theta(1)$
3	$= \Theta(n)$
4-7	$= \Theta(1)$
8	$= T(k-1)$
9	$= T(n-k)$
$T(n)$	$= \Theta(n) + \max\{T(k-1), T(n-k)\}$

Pior caso: $T(n) \; = \; \Theta(n) + T(n-1)$

Consumo de tempo

$T(n)$ pertence a mesma classe Θ que:

$$T'(1) = 1$$

$$T'(n) = T'(n-1) + n \quad \text{para } n = 2, 3, \ldots$$

Solução assintótica: $T'(n)$ é $\Theta(n^2)$.

Solução exata:

$$T'(n) = \frac{n^2}{2} + \frac{n}{2}.$$

Algumas conclusões

No melhor caso o consumo de tempo
do algoritmo SELECT é $\Theta(n)$.

No pior caso o consumo de tempo
do algoritmo SELECT é $\Theta(n^2)$.

Algumas conclusões

No melhor caso o consumo de tempo
do algoritmo SELECT é $\Theta(n)$.

No pior caso o consumo de tempo
do algoritmo SELECT é $\Theta(n^2)$.

Nas próximas aulas, voltaremos a este algoritmo.

Seleção em tempo linear

Mais uma aplicação de divisão e conquista.

CLRS 9.3

BFPRT = Blum, Floyd, Pratt, Rivest e Tarjan

Seleção em tempo linear

Mais uma aplicação de divisão e conquista.

CLRS 9.3

BFPRT = Blum, Floyd, Pratt, Rivest e Tarjan

Se o pivô do PARTICIONE for a mediana do vetor, qual seria o consumo de tempo do SELECT?

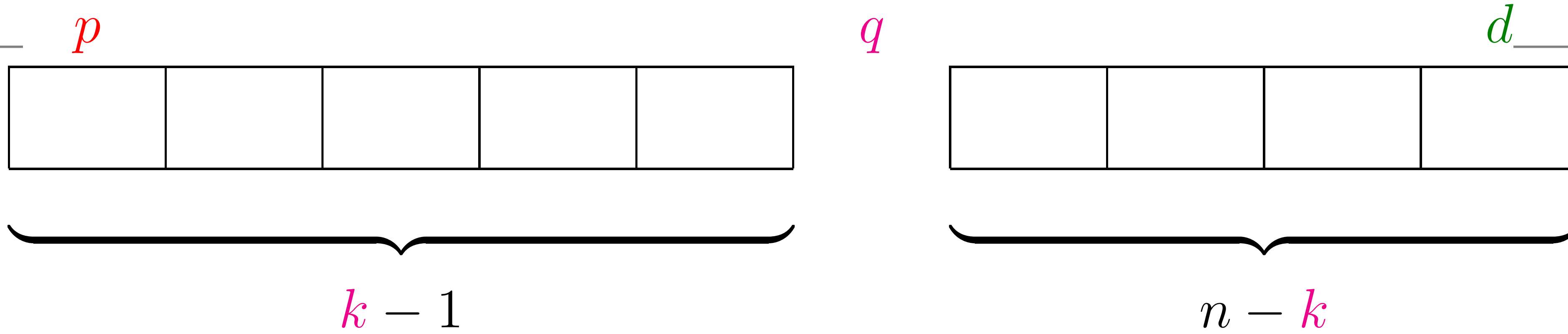
Select-BFPRT

Recebe $A[p\,..\,d]$ e i tal que $1 \leq i \leq d{-}p{+}1$ e devolve um índice q tal que $A[q]$ é o i-ésimo menor elemento de $A[p\,..\,d]$

SELECT-BFPRT(A, p, d, i)

    se $p = d$
        então devolva p $\triangleright$ p e não $A[p]$
    $q \leftarrow$ PARTICIONE-BFPRT (A, p, d)
    $k \leftarrow q - p + 1$
    se $k = i$
        então devolva q $\triangleright$ q e não $A[q]$
    se $k > i$
        então devolva SELECT-BFPRT $(A, p, q-1, i)$
        senão devolva SELECT-BFPRT $(A, q+1, d, i-k)$

Particione-BFPRT



Rearranja $A[p \,..\, d]$ e devolve um índice q, $p \leq q \leq d$, tal que $A[p \,..\, q{-}1] \leq A[q] < A[q{+}1 \,..\, d]$ **e**

$$\max\{k-1, n-k\} \leq \frac{7n}{10} + 3,$$

onde $n = d - p + 1$ e $k = q - p + 1$.

Suponha que

$P(n) :=$ consumo de tempo máximo do algoritmo PARTICIONE-BFPRT quando $n = d - p + 1$

Consumo de tempo

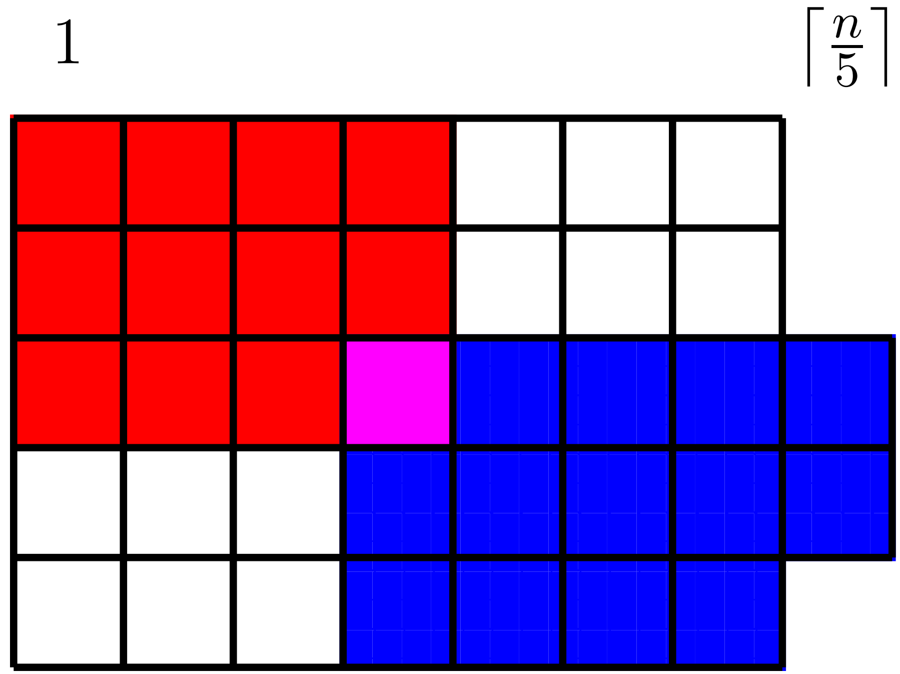
$T(n) :=$ consumo de tempo máximo do algoritmo
SELECT-BFPRT quando $n = d - p + 1$

linha	consumo de todas as execuções da linha
1-2	$= \Theta(1)$
3	$= P(n)$
4-7	$= \Theta(1)$
8	$= T(k-1)$
9	$= T(n-k)$

$$T(n) = \Theta(1) + P(n) + \max\{T(k-1), T(n-k)\}$$

$$\leq \Theta(1) + P(n) + T(\lceil \tfrac{7n}{10} \rceil + 3)$$

Particione-BFPRT



$$
\begin{aligned}
\max\{k-1, n-k\} &\leq n - \left(3\left\lceil \frac{1}{2}\left\lceil \frac{n}{5} \right\rceil \right\rceil - 3\right) \\
&\leq n - \left(\frac{3n}{10} - 3\right) = \frac{7n}{10} + 3
\end{aligned}
$$

Particione-BFPRT

$n := d - p + 1$

PARTICIONE-BFPRT (A, p, d)

para $j \leftarrow p, p{+}5, p{+}5 \cdot 2, \ldots$ **até** $p{+}5(\lceil n/5 \rceil{-}1)$ **faça**

    ORDENE $(A, j, j{+}4)$

ORDENE $(A, p{+}5\lfloor n/5 \rfloor, n)$

para $j \leftarrow 1$ **até** $\lceil n/5 \rceil{-}1$ **faça**

    $B[j] \leftarrow A[p{+}5j{-}3]$

$B[\lceil n/5 \rceil] \leftarrow A[\lfloor (p{+}5\lfloor n/5 \rfloor{+}n)/2 \rfloor]$

$k \leftarrow$ SELECT-BFPRT$(B, 1, \lceil n/5 \rceil, \lfloor (\lceil n/5 \rceil{+}1)/2 \rfloor)$

$A[k] \leftrightarrow A[d]$

devolva PARTICIONE (A, p, d)

Particione-BFPRT

$n := d - p + 1$

PARTICIONE-BFPRT (A, p, d)

para $j \leftarrow p, p{+}5, p{+}5 \cdot 2, \ldots$ **até** $p{+}5(\lceil n/5 \rceil{-}1)$ **faça**

  ORDENE $(A, j, j{+}4)$

ORDENE $(A, p{+}5\lfloor n/5 \rfloor, n)$

para $j \leftarrow 1$ **até** $\lceil n/5 \rceil{-}1$ **faça**

  $A[p{-}1{+}j] \leftrightarrow A[p{+}5j{-}3]$

$A[p{-}1{+}\lceil n/5 \rceil] \leftrightarrow A[\lfloor (p{+}5\lfloor n/5 \rfloor{+}n)/2 \rfloor]$

$k \leftarrow$ SELECT-BFPRT$(A, p, p{+}\lceil n/5 \rceil{-}1, \lfloor (\lceil n/5 \rceil{+}1)/2 \rfloor)$

$A[k] \leftrightarrow A[d]$

devolva PARTICIONE (A, p, d)

Consumo de tempo do Particione-BFPRT

$P(n) :=$ consumo de tempo máximo do algoritmo
PARTICIONE-BFPRT quando $n = d - p + 1$

linha	consumo de todas as execuções da linha
1-3	$= \lceil n/5 \rceil\, \Theta(1) \;=\; \Theta(n)$
4-6	$= \lceil n/5 \rceil\, \Theta(1) \;=\; \Theta(n)$
7	$= T(\lceil n/5 \rceil)$
8	$= \Theta(1)$
9	$= \Theta(n)$

$P(n) \quad = \Theta(n) + T(\lceil n/5 \rceil)$

Consumo de tempo do Select-BFPRT

$T(n) :=$ consumo de tempo máximo do algoritmo SELECT-BFPRT quando $n = d - p + 1$

Temos que

$$
\begin{aligned}
T(1) &= \Theta(1) \\
T(n) &\le \Theta(1) + P(n) + T\left(\left\lceil \frac{7n}{10} \right\rceil + 3\right) \\
&\le \Theta(1) + \Theta(n) + T\left(\left\lceil \frac{n}{5} \right\rceil\right) + T\left(\left\lceil \frac{7n}{10} \right\rceil + 3\right) \\
&= \Theta(n) + T\left(\left\lceil \frac{n}{5} \right\rceil\right) + T\left(\left\lceil \frac{7n}{10} \right\rceil + 3\right)
\end{aligned}
$$

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

Consumo de tempo do Select-BFPRT

$T(n)$ pertence a mesma classe O que:

$$S(n) = 1 \quad \text{para } n < 30$$

$$S(n) \leq S\left(\left\lceil \frac{n}{5} \right\rceil\right) + S\left(\left\lceil \frac{7n}{10} \right\rceil + 3\right) + n \quad \text{para } n \geq 30$$

n	30	60	90	120	150	180	210	240	270	300
$S(n)$	32	144	280	362	514	640	802	940	1114	1261

Vamos verificar que $S(n) < 80\,n$ para $n = 1, 2, 3, 4, \ldots$

Prova: Se $n = 1, \ldots, 29$, então $S(n) = 1 < 80 < 80n$.
Se $n = 30, \ldots, 99$, então

$$S(n) < S(120) = 362 < 80 \times 30 \leq 80n.$$

Recorrência

Se $n \geq 100$, então

$$S(n) \leq S\left(\left\lceil \frac{n}{5} \right\rceil\right) + S\left(\left\lceil \frac{7n}{10} \right\rceil + 3\right) + n$$

$$\overset{\text{hi}}{<} 80 \left\lceil \frac{n}{5} \right\rceil + 80 \left(\left\lceil \frac{7n}{10} \right\rceil + 3\right) + n$$

$$\leq 80 \left(\frac{n}{5} + 1\right) + 80 \left(\frac{7n}{10} + 4\right) + n$$

$$= 80 \frac{n}{5} + 80 + 80 \frac{7n}{10} + 320 + n$$

$$= 16n + 56n + n + 400$$

$$= 73n + 400$$

$$< 80n \quad \text{(pois } n \geq 100\text{).}$$

Logo, $T(n)$ é $O(n)$.

Como adivinhei classe O?

Árvore da recorrência:

$$S(n)$$

Como adivinhei classe O?

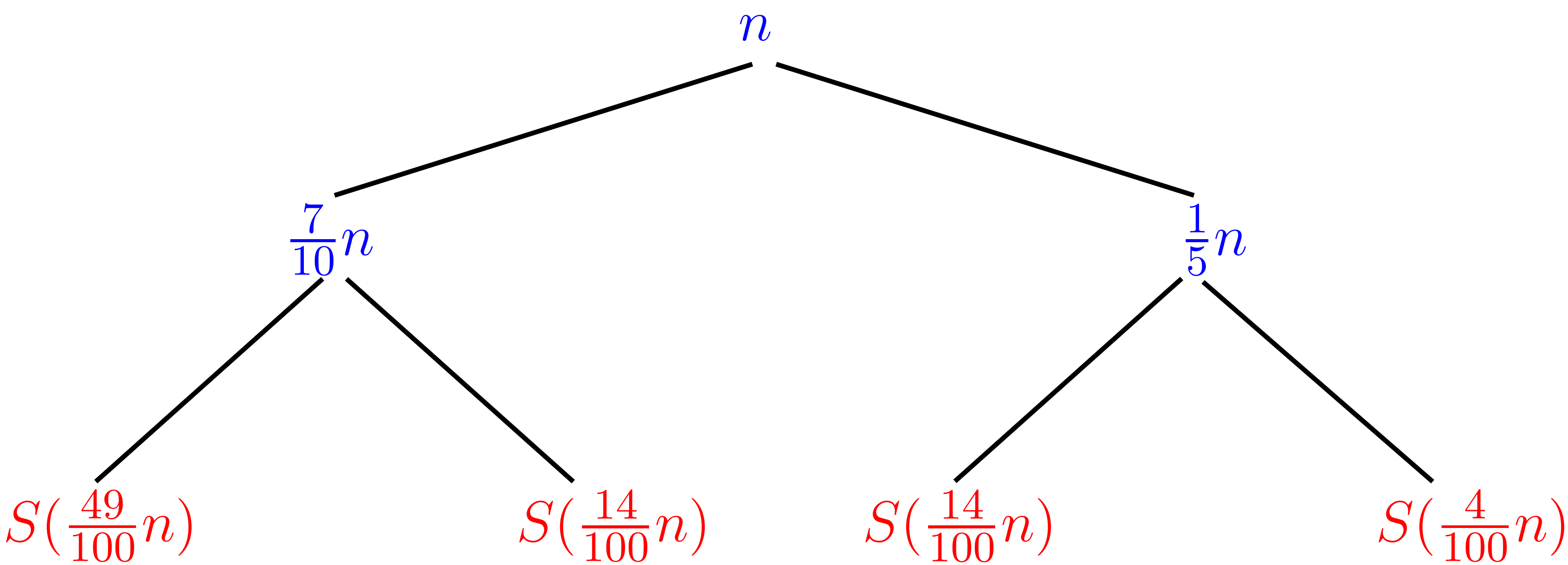
Árvore da recorrência:

$$n$$

$$S(\tfrac{7}{10}n) \qquad\qquad S(\tfrac{1}{5}n)$$

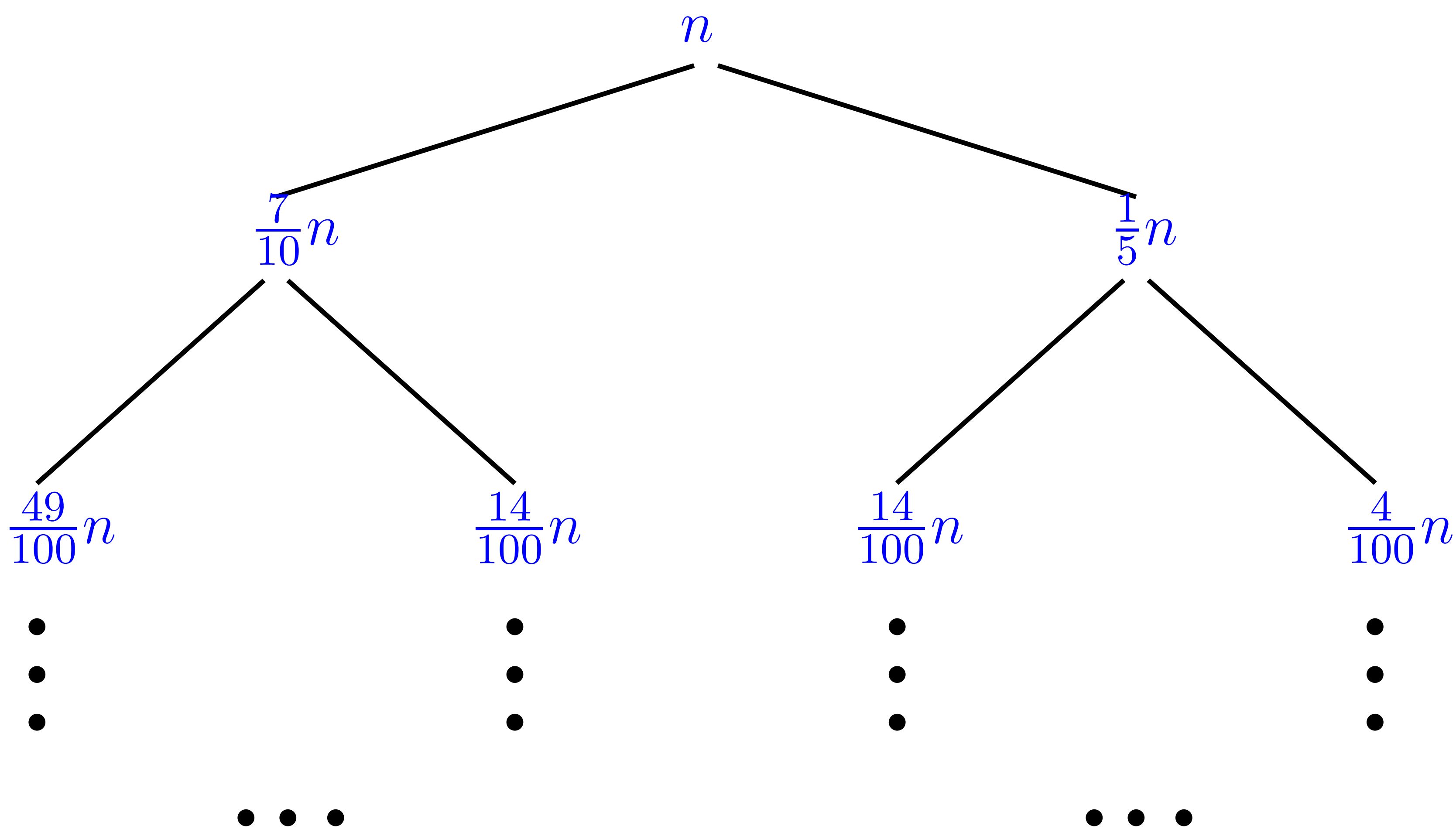
Como adivinhei classe O?

Árvore da recorrência:



Como adivinhei classe O?

Árvore da recorrência:



Contas

nível	0	1	2	...	$k-1$	k
soma	n	$\frac{9}{10}n$	$\frac{9^2}{10^2}n$	...	$\frac{9^{k-1}}{10^{k-1}}n$	$\frac{9^k}{10^k}n$

$$\frac{10^{k-1}}{9^{k-1}} < n \leq \frac{10^k}{9^k} \Rightarrow k = \lceil \log_{\frac{10}{9}} n \rceil$$

$$\begin{aligned} S(n) &= n + \frac{9}{10}n + \cdots + \frac{9^{k-1}}{10^{k-1}}n + \frac{9^k}{10^k}n \\ &= (1 + \frac{9}{10} + \cdots + \frac{9^k}{10^k})n \\ &= 10(1 - \frac{9^{k+1}}{10^{k+1}})n \\ &< 10n \end{aligned}$$

Conclusão

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