MAC5711 Análise de Algoritmos

Slides de Paulo Feofiloff

[com erros do coelho]

MAC5711 Análise de Algoritmos

Slides de Paulo Feofiloff

[com erros do coelho]

"A análise de algoritmos é uma disciplina de engenharia. Um engenheiro civil, por exemplo, tem métodos e tecnologia para prever o comportamento de uma estrutura antes de construí-la.

Da mesma forma, um projetista de algoritmos deve ser capaz de prever o comportamento de um algoritmo antes de implementá-lo."

Avisos

- Página da disciplina:

  `www.ime.usp.br/~coelho/algoritmos/`
- Panda: Cadastro, fórum, entregas de trabalho
- Monitor: Paulo Silveira, 2a. das 12:30 às 14:00 (?)
- Livros:
  - CLRS = Cormen, Leiserson, Rivest, Stein, *Introduction to Algorithms*
  - AU = Aho, Ullman, *Foundations of Computer Science*
  - TAOCP = Knuth, *The Art of Computer Programming*
- Tarefas
- Alunos especiais: formulário, comissão.

Blá-Blá-Blá

Correção de algoritmos iterativos através de suas relações invariantes.

Tema recorrente e fundamental de MAC5711: a estrutura recursiva de muitos problemas.

Vamos estudar uma coleção de diferentes problemas. Eles servirão de paradigma para muitas técnicas e idéias importantes e muito gerais, aplicáveis a muitos outros problemas no futuro.

Introdução à AA

CLRS 2.1–2.2

AU 3.3, 3.6

Exercício 1.A

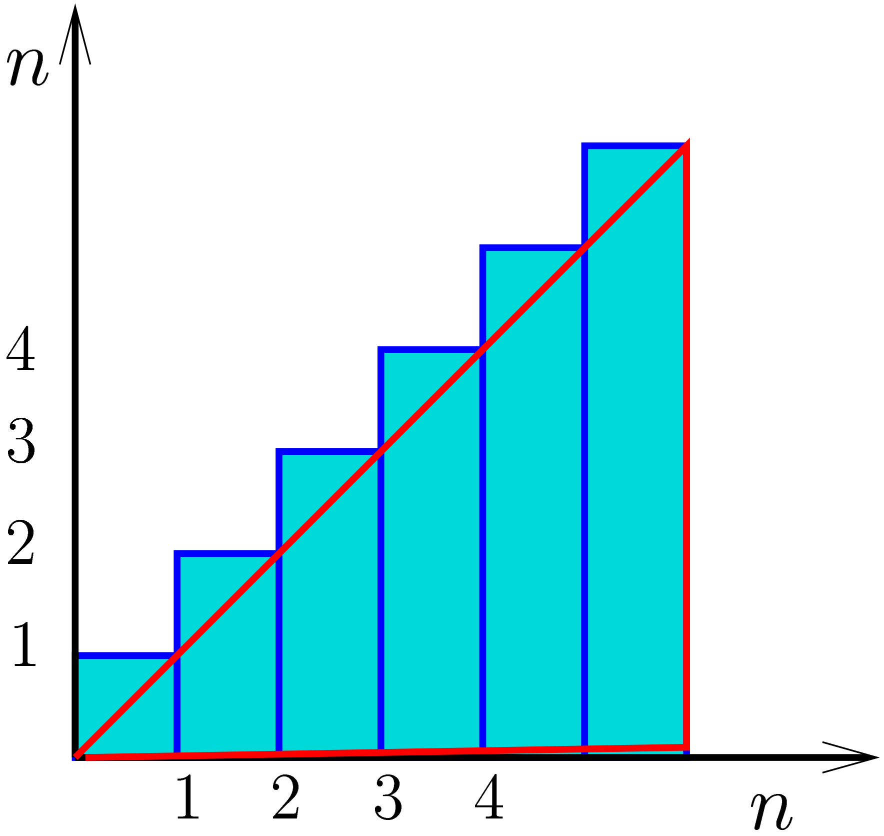
Quanto vale S no fim do algoritmo?

$S \leftarrow 0$
para $i \leftarrow 2$ **até** $n-2$ **faça**
    para $j \leftarrow i$ **até** n **faça**
        $S \leftarrow S + 1$

Escreva um algoritmo mais eficiente que tenha o mesmo efeito.

$$1+2+\cdots+(n-1)+n=?$$

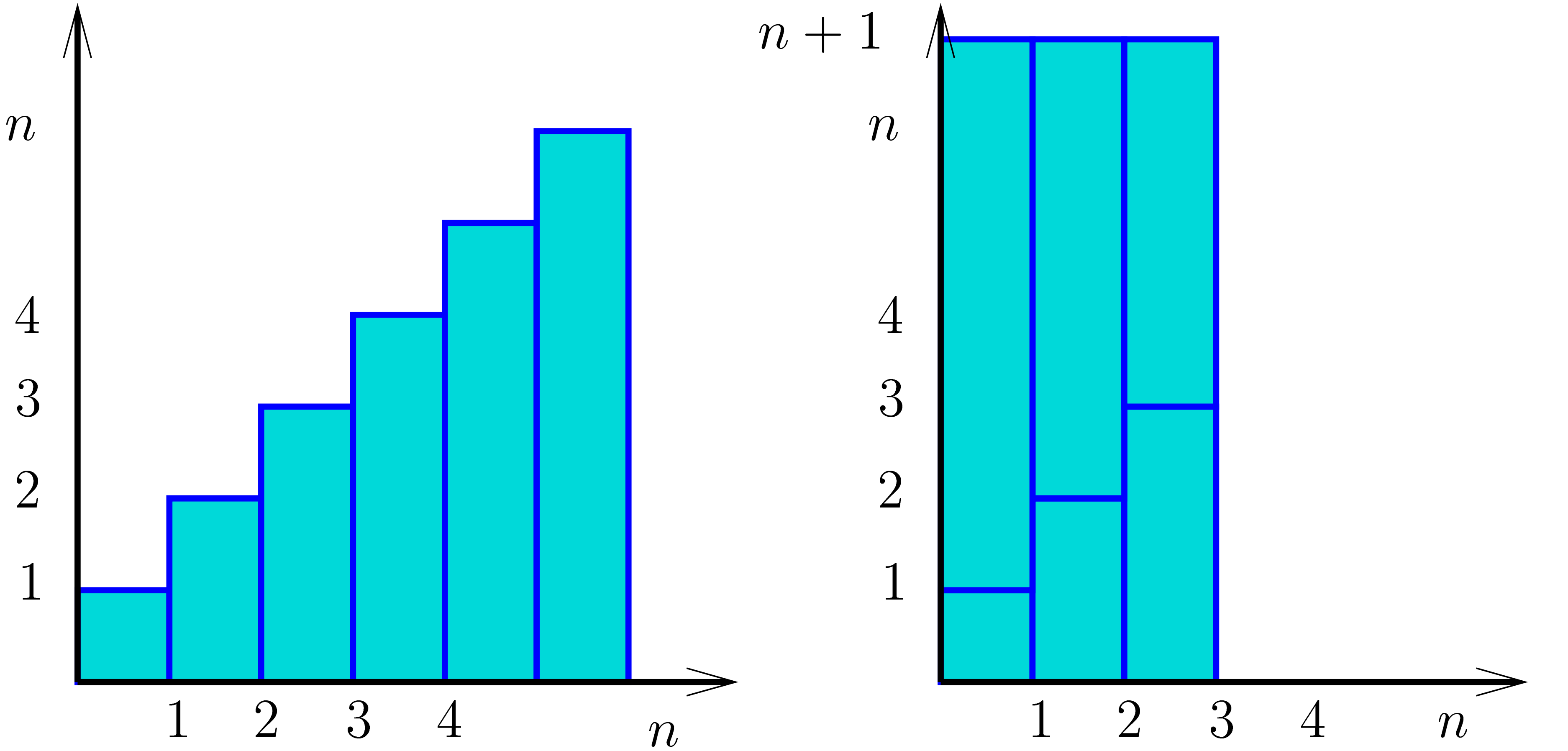
Carl Friedrich Gauss, 1787



$$\frac{n^2}{2}+\frac{n}{2}=\frac{n(n+1)}{2}$$

$$1 + 2 + \cdots + (n-1) + n = ?$$

Carl Friedrich Gauss, 1787



$$(n+1) \times \frac{n}{2} = \frac{n(n+1)}{2}$$

Solução

Se $n \geq 4$ então no fim da execução das linhas 1–4,

$$
\begin{aligned}
S &= (n-1) + (n-2) + \cdots + 4 + 3 \\
&= (n+2)(n-3)/2 \\
&= \tfrac{1}{2}n^2 - \tfrac{1}{2}n - 3\,.
\end{aligned}
$$

Ordenação

$A[1\,..\,n]$ é crescente se $A[1] \leq \cdots \leq A[n]$.

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

Entra:

1										n
33	55	33	44	33	22	11	99	22	55	77

Ordenação

$A[1\,..\,n]$ é crescente se $A[1] \leq \cdots \leq A[n]$.

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

Entra:

1										n
33	55	33	44	33	22	11	99	22	55	77

Sai:

1										n
11	22	22	33	33	33	44	55	55	77	99

Ordenação por inserção

$chave = 38$

1						j				n
20	25	35	40	44	55	38	99	10	65	50

Ordenação por inserção

$chave = 38$

1					i	j				n
20	25	35	40	44	55	38	99	10	65	50

Ordenação por inserção

$chave = 38$

1					i	j				n
20	25	35	40	44	55	38	99	10	65	50

1				i		j				n
20	25	35	40	44		55	99	10	65	50

Ordenação por inserção

$chave = 38$

1					i	j				n
20	25	35	40	44	55	38	99	10	65	50

1				i		j				n
20	25	35	40	44		55	99	10	65	50

1			i			j				n
20	25	35	40		44	55	99	10	65	50

Ordenação por inserção

$chave = 38$

1					i	j				n
20	25	35	40	44	55	38	99	10	65	50

1				i		j				n
20	25	35	40	44		55	99	10	65	50

1			i			j				n
20	25	35	40		44	55	99	10	65	50

1		i				j				n
20	25	35		40	44	55	99	10	65	50

Ordenação por inserção

$chave = 38$

1					i	j				n
20	25	35	40	44	55	38	99	10	65	50

1				i		j				n
20	25	35	40	44		55	99	10	65	50

1			i			j				n
20	25	35	40		44	55	99	10	65	50

1		i				j				n
20	25	35		40	44	55	99	10	65	50

1		i				j				n
20	25	35	38	40	44	55	99	10	65	50

Ordenação por inserção

$chave$	1							j			n
99	20	25	35	38	40	44	55	99	10	65	50

Ordenação por inserção

$chave$	1							j			n
99	20	25	35	38	40	44	55	99	10	65	50

Ordenação por inserção

$chave$	1							j			n
99	20	25	35	38	40	44	55	99	10	65	50

$chave$	1								j		n
10	20	25	35	38	40	44	55	99	10	65	50

Ordenação por inserção

chave	1							j			n
99	20	25	35	38	40	44	55	99	10	65	50

chave	1								j		n
10	10	20	25	35	38	40	44	55	99	65	50

Ordenação por inserção

$chave$	1							j			n
99	20	25	35	38	40	44	55	99	10	65	50

$chave$	1								j		n
10	10	20	25	35	38	40	44	55	99	65	50

$chave$	1									j	n
65	10	20	25	35	38	40	44	55	99	65	50

Ordenação por inserção

chave	1							j			n
99	20	25	35	38	40	44	55	99	10	65	50

chave	1								j		n
10	10	20	25	35	38	40	44	55	99	65	50

chave	1									j	n
65	10	20	25	35	38	40	44	55	65	99	50

Ordenação por inserção

chave	1							j			n
99	20	25	35	38	40	44	55	99	10	65	50

chave	1								j		n
10	10	20	25	35	38	40	44	55	99	65	50

chave	1									j	n
65	10	20	25	35	38	40	44	55	65	99	50

chave	1										j
50	10	20	25	35	38	40	44	55	65	99	50

Ordenação por inserção

$chave$	1							j			n
99	20	25	35	38	40	44	55	99	10	65	50

$chave$	1								j		n
10	10	20	25	35	38	40	44	55	99	65	50

$chave$	1									j	n
65	10	20	25	35	38	40	44	55	65	99	50

$chave$	1										j
50	10	20	25	35	38	40	44	50	55	65	99

Ordenação por inserção

Algoritmo rearranja $A[1\,..\,n]$ em ordem crescente

ORDENA-POR-INSERÇÃO (A, n)
1 **para** $j \leftarrow 2$ **até** n **faça**
2 $\quad chave \leftarrow A[j]$

3 $\quad i \leftarrow j - 1$
4 $\quad$ **enquanto** $i \geq 1$ **e** $A[i] > chave$ **faça**
5 $\quad\quad A[i+1] \leftarrow A[i]$ $\quad \triangleright$ desloca
6 $\quad\quad i \leftarrow i - 1$

7 $\quad A[i+1] \leftarrow chave$ $\quad \triangleright$ insere

Ordenação por inserção

Algoritmo rearranja $A[1\,..\,n]$ em ordem crescente

ORDENA-POR-INSERÇÃO (A, n)

$j \leftarrow 2$
enquanto $j \leq n$ **faça**
    $chave \leftarrow A[j]$

    $i \leftarrow j - 1$
    enquanto $i \geq 1$ **e** $A[i] > chave$ **faça**
        $A[i+1] \leftarrow A[i]$ $\triangleright$ desloca
        $i \leftarrow i - 1$

    $A[i+1] \leftarrow chave$ $\triangleright$ insere
    $j \leftarrow j + 1$

O algoritmo faz o que promete?

Correção do algoritmo!

O algoritmo faz o que promete?

Correção do algoritmo!

Relação invariante chave:

(i0) na linha 1 vale que: $A[1\,..\,j-1]$ é crescente.

1						j				n
20	25	35	40	44	55	38	99	10	65	50

O algoritmo faz o que promete?

Correção do algoritmo!

Relação invariante chave:

(i0) na linha 1 vale que: $A[1\,..\,j-1]$ é crescente.

1						j				n
20	25	35	40	44	55	38	99	10	65	50

Supondo que a invariante vale. Correção do algoritmo é evidente.

No início da última iteração das linhas 1–7 tem-se que $j = n+1$. Da invariante concluí-se que $A[1\,..\,n]$ é crescente.

Mais invariantes

Na linha 4 vale que:

(i1) $A[1\,..\,i]$ e $A[i+2\,..\,j]$ são crescentes

(i2) $A[1\,..\,i] \leq A[i+2\,..\,j]$

(i3) $A[i+2\,..\,j] > chave$

$chave$	1		i				j				n
38	20	25	35		40	44	55	99	10	65	50

Mais invariantes

Na linha 4 vale que:

(i1) $A[1\,..\,i]$ e $A[i+2\,..\,j]$ são crescentes

(i2) $A[1\,..\,i] \leq A[i+2\,..\,j]$

(i3) $A[i+2\,..\,j] > chave$

chave	1		i				j				n
38	20	25	35		40	44	55	99	10	65	50

invariantes (i1),(i2) e (i3)
$+$ condição de parada do **enquanto** da linha 4
$+$ atribuição da linha 7 $\Rightarrow$ validade (i0)

Demonstre!

Correção de algoritmos iterativos

Estrutura "típica" de demonstrações da correção de algoritmos iterativos através de suas relações invariantes consiste em:

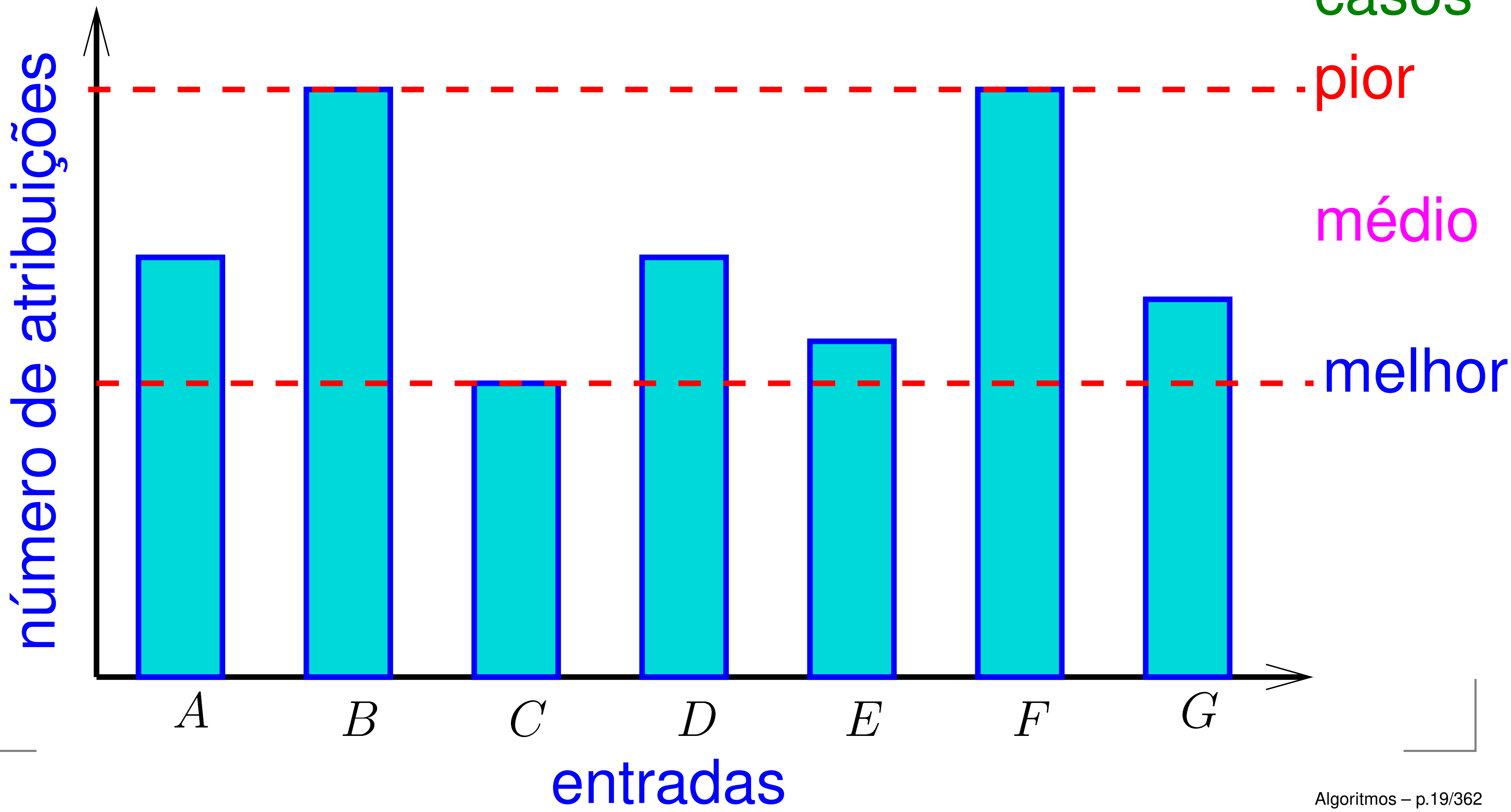
1. verificar que a relação vale no início da primeira iteração;
2. demonstrar que

   se a relação vale no início da iteração, então ela vale no final da iteração (com os papéis de alguns atores possivelmente trocados);
3. concluir que, se relação vale no início da última iteração, então a a relação junto com a condição de parada implicam na correção do algoritmo.

Quantas atribuições ($\leftarrow$) algoritmo faz?

Quantas atribuições (←) algoritmo faz?

Número mínimo, médio ou máximo?
Melhor caso, caso médio, pior caso?



Quantas atribuições ($\leftarrow$) algoritmo faz?

LINHAS 3–6 $(A, j, chave)$

3 $\quad i \leftarrow j - 1 \qquad \rhd\ 2 \leq j \leq n$

4 $\quad$ **enquanto** $i \geq 1$ **e** $A[i] > chave$ **faça**

5 $\qquad A[i+1] \leftarrow A[i]$

6 $\qquad i \leftarrow i - 1$

linha	atribuições (número máximo)
3	?
4	?
5	?
6	?
total	?

Quantas atribuições ($\leftarrow$) algoritmo faz?

LINHAS 3–6 $(A, j, chave)$

3 $\quad i \leftarrow j - 1 \qquad \triangleright\ 2 \leq j \leq n$

4 $\quad$ **enquanto** $i \geq 1$ **e** $A[i] > chave$ **faça**

5 $\qquad A[i+1] \leftarrow A[i]$

6 $\qquad i \leftarrow i - 1$

linha	atribuições (número máximo)
3	$= 1$
4	$= 0$
5	?
6	?

total ?

Quantas atribuições ($\leftarrow$) algoritmo faz?

LINHAS 3–6 $(A, j, chave)$

3 $\quad i \leftarrow j - 1 \qquad \triangleright\ 2 \leq j \leq n$

4 $\quad$ **enquanto** $i \geq 1$ **e** $A[i] > chave$ **faça**

5 $\qquad A[i+1] \leftarrow A[i]$

6 $\qquad i \leftarrow i - 1$

linha	atribuições (número máximo)
3	$= 1$
4	$= 0$
5	$\leq j - 1$
6	?

total ?

Quantas atribuições ($\leftarrow$) algoritmo faz?

LINHAS 3–6 $(A, j, chave)$

3 $\quad i \leftarrow j - 1 \qquad \triangleright\ 2 \leq j \leq n$

4 $\quad$ **enquanto** $i \geq 1$ **e** $A[i] > chave$ **faça**

5 $\qquad A[i+1] \leftarrow A[i]$

6 $\qquad i \leftarrow i - 1$

linha	atribuições (número máximo)
3	$= 1$
4	$= 0$
5	$\leq j - 1$
6	$\leq j - 1$

total $\leq 2j - 1 \leq 2n - 1$

Quantas atribuições ($\leftarrow$) algoritmo faz?

ORDENA-POR-INSERÇÃO (A, n)

1 **para** $j \leftarrow 2$ **até** n **faça** $\triangleright$ $j \leftarrow j + 1$ escondido

2 $\quad$ $chave \leftarrow A[j]$

3 $\quad$ LINHAS 3–6 $(A, j, chave)$

7 $\quad$ $A[i+1] \leftarrow chave$

linha	atribuições (número máximo)
1	?
2	?
3–6	?
7	?

total ?

Quantas atribuições ($\leftarrow$) algoritmo faz?

ORDENA-POR-INSERÇÃO (A, n)

1 **para** $j \leftarrow 2$ **até** n **faça** $\rhd$ $j \leftarrow j + 1$ escondido

2 $chave \leftarrow A[j]$

3 LINHAS 3–6 $(A, j, chave)$

7 $A[i+1] \leftarrow chave$

linha	atribuições (número máximo)
1	$= n - 1 + 1$
2	$= n - 1$
3–6	$\leq (n-1)(2n-1)$
7	$= n - 1$

total $\leq 2n^2 - 1$

Análise mais fina

linha	atribuições (número máximo)
1	$= n - 1 + 1$
2	$= n - 1$
3	$= n - 1$
4	$= 0$
5	$\leq 1 + 2 + \cdots + (n{-}1) = n(n-1)/2$
6	$\leq 1 + 2 + \cdots + (n{-}1) = n(n-1)/2$
7	$= n - 1$

total $\leq n^2 + 3n - 3$

$n^2 + 3n - 3$ **versus** n^2

n	$n^2 + 3n - 3$	n^2
1	1	1
2	7	4

$n^2 + 3n - 3$ **versus** n^2

n	$n^2 + 3n - 3$	n^2
1	1	1
2	7	4
3	15	9
10	127	100

$n^2 + 3n - 3$ **versus** n^2

n	$n^2 + 3n - 3$	n^2
1	1	1
2	7	4
3	15	9
10	127	100
100	10297	10000
1000	1002997	1000000

$n^2 + 3n - 3$ **versus** n^2

n	$n^2 + 3n - 3$	n^2
1	1	1
2	7	4
3	15	9
10	127	100
100	10297	10000
1000	1002997	1000000
10000	100029997	100000000
100000	10000299997	10000000000

n^2 domina os outros termos

Exercício 1.B

Se a execução de cada linha de código consome 1 unidade de tempo, qual o consumo total?

ORDENA-POR-INSERÇÃO (A, n)

para $j \leftarrow 2$ **até** n **faça**
    $chave \leftarrow A[j]$

    $i \leftarrow j - 1$
    enquanto $i \geq 1$ **e** $A[i] > chave$ **faça**
        $A[i+1] \leftarrow A[i]$ $\triangleright$ desloca
        $i \leftarrow i - 1$

    $A[i+1] \leftarrow chave$ $\triangleright$ insere

Solução

linha	todas as execuções da linha
1	$= \quad n$
2	$= \quad n-1$
3	$= \quad n-1$
4	$\leq \quad 2+3+\cdots+n \;=\; (n-1)(n+2)/2$
5	$\leq \quad 1+2+\cdots+(n-1) \;=\; n(n-1)/2$
6	$\leq \quad 1+2+\cdots+(n-1) \;=\; n(n-1)/2$
7	$= \quad n-1$
total	$\leq \quad (3/2)n^2+(7/2)n-4$

Exercício 1.C

Se a execução da linha i consome t_i unidades de tempo, para $i = 1, \ldots, 7$, qual o consumo total?

ORDENA-POR-INSERÇÃO (A, n)

1 **para** $j \leftarrow 2$ **até** n **faça**
2     $chave \leftarrow A[j]$

3     $i \leftarrow j - 1$
4   **enquanto** $i \geq 1$ **e** $A[i] > chave$ **faça**
5         $A[i+1] \leftarrow A[i]$ $\triangleright$ desloca
6         $i \leftarrow i - 1$

7     $A[i+1] \leftarrow chave$ $\triangleright$ insere

Solução para $t_i = 1$

linha	todas as execuções da linha			
1	$=$	n		
2	$=$	$n-1$		
3	$=$	$n-1$		
4	$\leq$	$2+3+\cdots+n$	$=$	$(n-1)(n+2)/2$
5	$\leq$	$1+2+\cdots+(n-1)$	$=$	$n(n-1)/2$
6	$\leq$	$1+2+\cdots+(n-1)$	$=$	$n(n-1)/2$
7	$=$	$n-1$		
total	$\leq$	$(3/2)n^2 + (7/2)n - 4$		

Solução

linha	todas as execuções da linha		
1	$=$	n	$\times t_1$
2	$=$	$n-1$	$\times t_2$
3	$=$	$n-1$	$\times t_3$
4	$\leq$	$2+3+\cdots+n \;=\; (n-1)(n+2)/2$	$\times t_4$
5	$\leq$	$1+2+\cdots+(n-1) \;=\; n(n-1)/2$	$\times t_5$
6	$\leq$	$1+2+\cdots+(n-1) \;=\; n(n-1)/2$	$\times t_6$
7	$=$	$n-1$	$\times t_7$

total $\leq$?

Solução

linha	todas as execuções da linha		
1	$=$	n	$\times t_1$
2	$=$	$n-1$	$\times t_2$
3	$=$	$n-1$	$\times t_3$
4	$\leq$	$2+3+\cdots+n \;=\; (n-1)(n+2)/2$	$\times t_4$
5	$\leq$	$1+2+\cdots+(n{-}1) \;=\; n(n-1)/2$	$\times t_5$
6	$\leq$	$1+2+\cdots+(n{-}1) \;=\; n(n-1)/2$	$\times t_6$
7	$=$	$n-1$	$\times t_7$

$$
\begin{array}{lll}
\text{total} & \leq & ((t_4+t_5+t_6)/2)\times n^2 \\
 & + & (t_1+t_2+t_3+t_4/2-t_5/2-t_6/2+t_7)\times n \\
 & - & (t_2+t_3+t_4+t_7)
\end{array}
$$

Solução

linha	todas as execuções da linha		
1	$=$	n	$\times t_1$
2	$=$	$n-1$	$\times t_2$
3	$=$	$n-1$	$\times t_3$
4	$\leq$	$2+3+\cdots+n \;=\; (n-1)(n+2)/2$	$\times t_4$
5	$\leq$	$1+2+\cdots+(n-1) \;=\; n(n-1)/2$	$\times t_5$
6	$\leq$	$1+2+\cdots+(n-1) \;=\; n(n-1)/2$	$\times t_6$
7	$=$	$n-1$	$\times t_7$

total $\leq$ $c_2 \times n^2 + c_1 \times n + c_0$

c_2, c_1, c_0 são constantes que dependem da máquina.

n^2 é para sempre! Esta nas entranhas do algoritmo!

Exercícios

Exercício 1.D (bom!)

Quanto vale S no fim do seguinte algoritmo?

$S \leftarrow 0$
$i \leftarrow n$
enquanto $i > 0$ **faça**
    para $j \leftarrow 1$ **até** i **faça**
        $S \leftarrow S + 1$
    $i \leftarrow \lfloor i/2 \rfloor$

Exercício 1.E

Prove a invariante do algoritmo ORDENA-POR-INSERÇÃO.

Exercício 1.F

Quantas comparações faz o algoritmo ORDENA-POR-INSERÇÃO, no pior caso, ao receber um vetor $A[1 \mathinner{..} n]$?

Exercício 1.G

Quantas atribuições faz o algoritmo abaixo?

$s \leftarrow 0$
para $i \leftarrow 1$ **até** n **faça**
    $s \leftarrow s + i$
devolva s

Escreva um algoritmo melhorado que produza o mesmo efeito com menos atribuições.

Exercícios

Exercício 1.H (AU 3.7.1)

O algoritmo abaixo opera sobre um vetor $A[1\,..\,n]$. Quantas atribuições ele faz no pior caso? Quantas comparações?

$s \leftarrow 0$
para $i \leftarrow 1$ **até** n **faça**
    $s \leftarrow s + A[i]$
$m \leftarrow s/n$
$k \leftarrow 1$
para $i \leftarrow 2$ **até** n **faça**
    se $(A[i] - m)^2 < (A[k] - m)^2$
        então $k \leftarrow i$
devolva k

Exercício 1.I (AU 3.7.2)

O fragmento abaixo opera sobre uma matriz $A[1\,..\,n, 1\,..\,n]$. Quantas atribuições faz?

para $i \leftarrow 1$ **até** $n - 1$ **faça**
    para $j \leftarrow i + 1$ **até** n **faça**
        para $k \leftarrow i$ **até**n **faça**
            $A[j,k] \leftarrow A[j,k] - A[i,k] \cdot A[j,i]/A[i,i]$

Exercícios

Exercício 1.J (AU 3.7.3*)

Quantas atribuições faz o algoritmo?

```
SUMPOWERSOFTWO (n)
    s ← 0
    para i ← 1 até n faça
        j ← i
        enquanto 2⌊j/2⌋ = j faça
        j ← ⌊j/2⌋
        s ← s + 1
    devolva s
```

Chão, teto, log etc

CLRS 3.2, A.1

AU 2.9

Definições

$\lfloor x \rfloor$:= inteiro i tal que $i \leq x < i + 1$

$\lceil x \rceil$:= inteiro j tal que $j - 1 < x \leq j$

Diz-se que

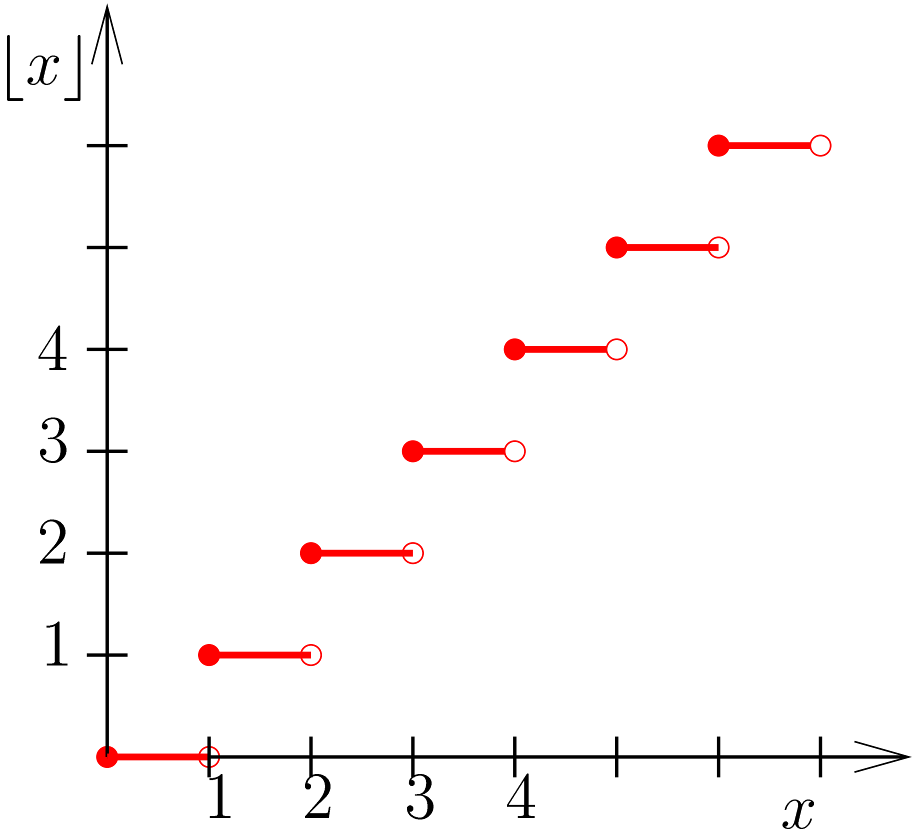
$\lfloor x \rfloor$ é o chão de x

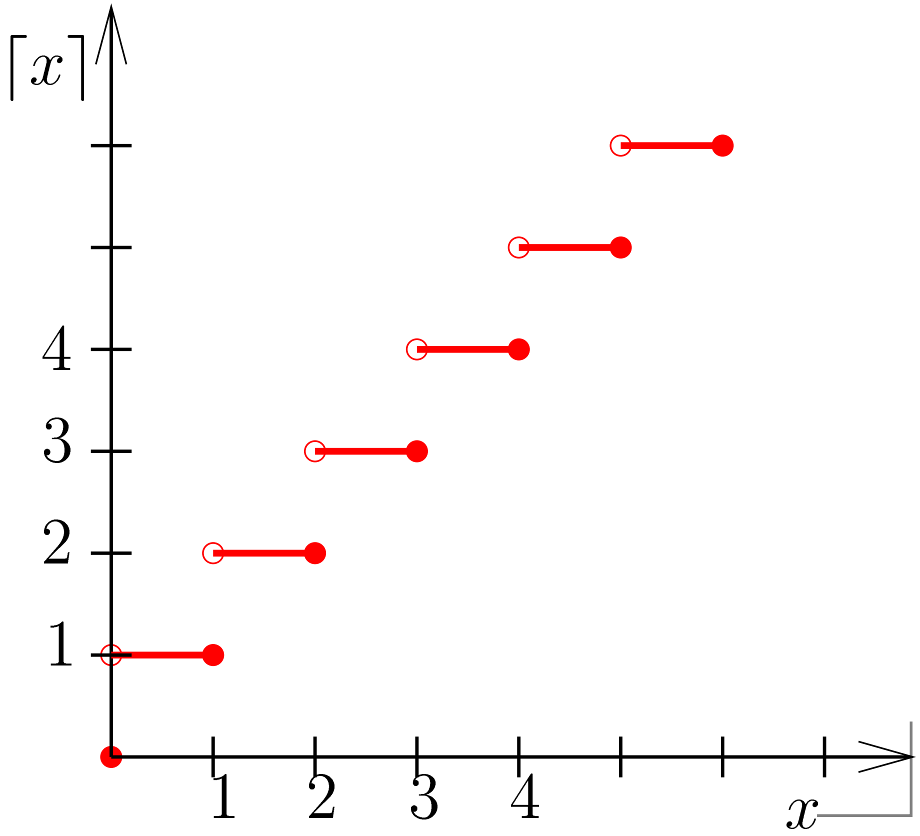
$\lceil x \rceil$ é o teto de x

Exercício A1.A

Desenhe os gráficos das funções $\lfloor x \rfloor$ e $\lceil x \rceil$ para x não-negativo.

Solução





Exercícios

Exercício A1.B
Mostre que para qualquer inteiro $n \geq 1$.

$$\frac{n-1}{2} \leq \left\lfloor \frac{n}{2} \right\rfloor \leq \frac{n}{2} \qquad \text{e} \qquad \frac{n}{2} \leq \left\lceil \frac{n}{2} \right\rceil \leq \frac{n+1}{2}$$

para qualquer inteiro $n \geq 1$.

Exercício A1.C
É verdade que $\lfloor x \rfloor + \lfloor y \rfloor = \lfloor x + y \rfloor$ para quaisquer x e y? É verdade que $\lfloor x \rfloor + \lfloor y \rfloor = \lfloor x + y \rfloor$ para quaisquer x e y?

Exercício A1.D
Desenhe os gráficos das funções $\lg x$ e 2^x para x inteiro não-negativo.

Exercícios

Exercício A1.E
Explique o significado da expressão $\log_{3/2} n$.

Exercício A1.F
Calcule $5^{\log_5 n}$, $\log_3 3^n$ e $\lg 2^n$.

Exercício A1.G
Qual a relação entre $\log_8 n$ e $\log_2 n$?

Exercício A1.H
Se $i := \lfloor \lg n \rfloor$, qual a relação entre n e 2^i ?
Se $j := \lceil \lg n \rceil$, qual a relação entre n e 2^j ?

Exercício A1.I
É verdade que $\lfloor \lg n \rfloor + 1 = \lceil \lg(n+1) \rceil$ para todo inteiro $n \geq 1$?

Exercício A1.J
Escreva um algoritmo que calcule $\lfloor \lg n \rfloor$.

Exercício A1.L
Mostre que para qualquer número real x tem-se $x - 1 < \lfloor x \rfloor \leq x \leq \lceil x \rceil < x + 1$.

Exercício A1.M
Mostre que $\lfloor n/2 \rfloor + \lceil n/2 \rceil = n$ para todo inteiro positivo n.

Mais exercícios

Exercício A1.N

Mostre que $\lfloor x \rfloor + \lfloor y \rfloor \leq \lfloor x + y \rfloor$, com igualdade se e somente se $x + y - 1 < \lfloor x \rfloor + \lfloor y \rfloor$. Encontre uma fórmula análoga para $\lceil \cdot \rceil$.

Exercício A1.O

Se c é inteiro e x é racional, é verdade que $\lceil cx \rceil = c\lceil x \rceil$?

Exercício A1.P

Use a notação $\lfloor\ \rfloor$ para representar o resto da divisão de n por 7.

Exercício A1.Q

É verdade que $\lceil 2\lceil 2n/3 \rceil/3 \rceil = \lceil 4n/9 \rceil$? É verdade que $\lfloor 2\lfloor 2n/3 \rfloor/3 \rfloor = \lfloor 4n/9 \rfloor$?

Exercício A1.R

É verdade que $\lfloor \lfloor n/2 \rfloor/2 \rfloor = \lfloor n/4 \rfloor$?

Exercício A1.S

Se n, a, b são inteiros positivos, é verdade que $\lfloor \lfloor n/a \rfloor/b \rfloor = \lfloor n/ab \rfloor$?

Exercício A1.T

É verdade que $\lfloor \lg n \rfloor \geq \lg(n-1)$ para todo inteiro $n \geq 2$? É verdade que $\lceil \lg n \rceil \leq \lg(n+1)$ para todo inteiro $n \geq 1$?

Mais exercícios ainda

Exercício A1.U

Prove que para qualquer número racional $x > 1$ tem-se

$$\lfloor \lg x \rfloor \leq \lg \lfloor x \rfloor \leq \lg x \leq \lg \lceil x \rceil \leq \lceil \lg x \rceil$$

Exercício A1.V

Quanto vale $1/2 + 1/4 + 1/8 + \cdots + 1/2^n + \cdots$?

Melhores momentos

AULA 1

Definições

$\lfloor x \rfloor$:= inteiro i tal que $i \leq x < i+1$

$\lceil x \rceil$:= inteiro j tal que $j-1 < x \leq j$

Exercício A1.B

Mostre que

$$\frac{n-1}{2} \leq \left\lfloor \frac{n}{2} \right\rfloor \leq \frac{n}{2} \qquad \text{e} \qquad \frac{n}{2} \leq \left\lceil \frac{n}{2} \right\rceil \leq \frac{n+1}{2}$$

para qualquer inteiro $n \geq 1$.

Ordenação por inserção

Algoritmo rearranja $A[1\,..\,n]$ em ordem crescente

ORDENA-POR-INSERÇÃO (A, n)
para $j \leftarrow 2$ **até** n **faça**
    $chave \leftarrow A[j]$

    $i \leftarrow j - 1$
    enquanto $i \geq 1$ **e** $A[i] > chave$ **faça**
        $A[i+1] \leftarrow A[i]$ $\triangleright$ desloca
        $i \leftarrow i - 1$

    $A[i+1] \leftarrow chave$ $\triangleright$ insere

O algoritmo faz o que promete?

Correção de algoritmos iterativos e invariantes

Relação invariante chave:

(i0) na linha 1 vale que: $A[1\,..\,j-1]$ é crescente.

1						j				n
20	25	35	40	44	55	38	99	10	65	50

Supondo que a invariante vale. Correção do algoritmo é evidente!

No início da última iteração das linhas 1–7 tem-se que $j = n+1$. Da invariante concluí-se que $A[1\,..\,n]$ é crescente.

Consumo de tempo

Se a execução de cada linha de código consome 1 unidade de tempo o consumo total é:

linha		todas as execuções da linha
1	$=$	n
2	$=$	$n-1$
3	$=$	$n-1$
4	$\leq$	$2+3+\cdots+n \;=\; (n-1)(n+2)/2$
5	$\leq$	$1+2+\cdots+(n-1) \;=\; n(n-1)/2$
6	$\leq$	$1+2+\cdots+(n-1) \;=\; n(n-1)/2$
7	$=$	$n-1$
total	$\leq$	$(3/2)n^2+(7/2)n-4$

$(3/2)n^2 + (7/2)n - 4$ **versus** $(3/2)n^2$

n	$(3/2)n^2 + (7/2)n - 4$	$(3/2)n^2$
64	6364	6144
128	25020	24576
256	99196	98304
512	395004	393216
1024	1576444	1572864
2048	6298620	6291456
4096	25180156	25165824
8192	100691964	100663296
16384	402710524	402653184
32768	1610727420	1610612736

$(3/2)n^2$ domina os outros termos

Consumo de tempo

Se a execução da linha i consome t_i unidades de tempo, para $i = 1, \ldots, 7$, o consumo totalde tempo é:

linha	todas as execuções da linha		
1	$=$	n	$\times t_1$
2	$=$	$n-1$	$\times t_2$
3	$=$	$n-1$	$\times t_3$
4	$\leq$	$2+3+\cdots+n \;=\; (n-1)(n+2)/2$	$\times t_4$
5	$\leq$	$1+2+\cdots+(n-1) \;=\; n(n-1)/2$	$\times t_5$
6	$\leq$	$1+2+\cdots+(n-1) \;=\; n(n-1)/2$	$\times t_6$
7	$=$	$n-1$	$\times t_7$

$$
\begin{array}{lll}
\text{total} & \leq & ((t_4+t_5+t_6)/2) \times n^2 \\
 & + & (t_1+t_2+t_3+t_4/2-t_5/2-t_6/2+t_7) \times n \\
 & - & (t_2+t_3+t_4+t_7)
\end{array}
$$

Consumo de tempo

linha	todas as execuções da linha	
1	$= \quad n$	$\times t_1$
2	$= \quad n-1$	$\times t_2$
3	$= \quad n-1$	$\times t_3$
4	$\leq \quad 2+3+\cdots+n \ = \ (n-1)(n+2)/2$	$\times t_4$
5	$\leq \quad 1+2+\cdots+(n-1) \ = \ n(n-1)/2$	$\times t_5$
6	$\leq \quad 1+2+\cdots+(n-1) \ = \ n(n-1)/2$	$\times t_6$
7	$= \quad n-1$	$\times t_7$

total $\leq \quad c_2 \times n^2 + c_1 \times n + c_0$

c_2, c_1, c_0 são constantes que dependem da máquina.

n^2 é para sempre! Esta nas entranhas do algoritmo!

Notação assintótica

CLRS 3.1

AU 3.4, p.96–100 (muito bom!)

Funções de n

$5n^2 - 9n$ $\qquad 4n + 8$ $\qquad \lfloor n/3 \rfloor + 4$ $\qquad 2\sqrt{n} + 7$

2^{n-3} $\qquad \lg n \quad (= \log_2 n)$

Qual é maior: $n^2 - 9$ ou $4n + 8$?

Funções de n

$5n^2 - 9n$ $\quad$ $4n + 8$ $\quad$ $\lfloor n/3 \rfloor + 4$ $\quad$ $2\sqrt{n} + 7$

2^{n-3} $\quad$ $\lg n$ $\quad$ $(= \log_2 n)$

Qual é maior: $n^2 - 9$ ou $4n + 8$?

Depende do valor de n!

Funções de n

$5n^2 - 9n \qquad 4n + 8 \qquad \lfloor n/3 \rfloor + 4 \qquad 2\sqrt{n} + 7$

$2^{n-3} \qquad \lg n \quad (= \log_2 n)$

Qual é maior: $n^2 - 9$ ou $4n + 8$?

Depende do valor de n!

Qual cresce mais?

Funções de n

$5n^2 - 9n$ $\quad 4n + 8$ $\quad \lfloor n/3 \rfloor + 4$ $\quad 2\sqrt{n} + 7$

2^{n-3} $\quad \lg n \quad (= \log_2 n)$

Qual é maior: $n^2 - 9$ ou $4n + 8$?

Depende do valor de n!

Qual cresce mais?

Comparação assintótica, ou seja, para n ENORME.

Funções de n

$5n^2 - 9n$ $\qquad 4n + 8$ $\qquad \lfloor n/3 \rfloor + 4$ $\qquad 2\sqrt{n} + 7$

2^{n-3} $\qquad \lg n \quad (= \log_2 n)$

Qual é maior: $n^2 - 9$ ou $4n + 8$?

Depende do valor de n!

Qual cresce mais?

Comparação assintótica, ou seja, para n ENORME.

$\lg n$ $\qquad 2\sqrt{n} + 7$ $\qquad \lfloor n/3 \rfloor + 4$ $\qquad 4n + 8$ $\qquad 5n^2 - 9n$

2^{n-3}

Notação O

Intuitivamente...

$O(f(n))$ $\approx$ funções q não crescem mais rápido que $f(n)$

$\approx$ funções menores ou iguais a um múltiplo de $f(n)$

n^2 $\quad (3/2)n^2$ $\quad 9999n^2$ $\quad n^2/1000$ $\quad$ etc.

crescem todas com a mesma velocidade

Notação O

Intuitivamente...

$\mathrm{O}(f(n)) \approx$ funções q não crescem mais rápido que $f(n)$

$\approx$ funções menores ou iguais a um múltiplo de $f(n)$

$n^2 \qquad (3/2)n^2 \qquad 9999n^2 \qquad n^2/1000 \qquad$ etc.

crescem todas com a mesma velocidade

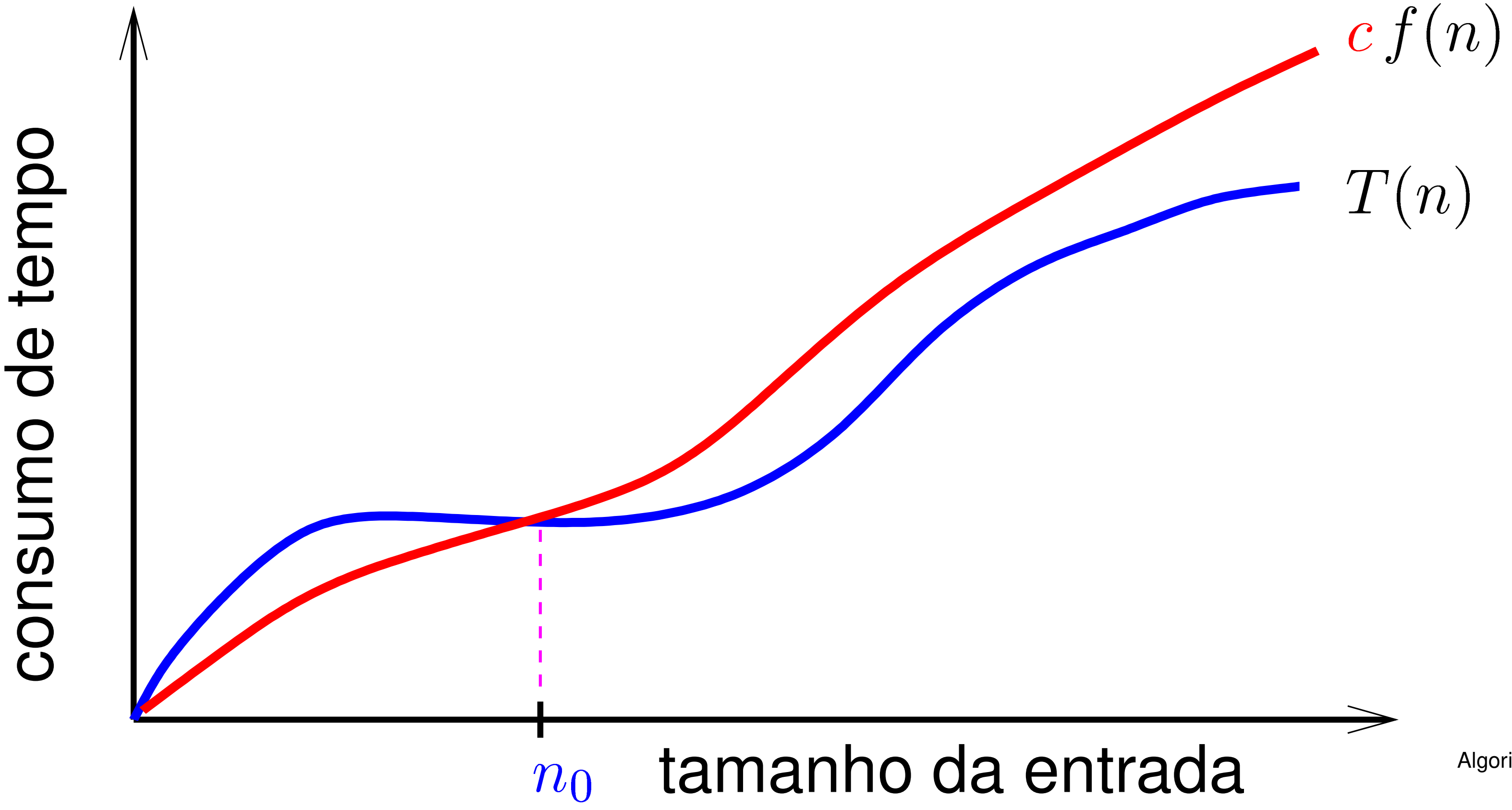
- $n^2 + 99n$ é $\mathrm{O}(n^2)$
- $33n^2$ é $\mathrm{O}(n^2)$
- $9n + 2$ é $\mathrm{O}(n^2)$
- $0{,}00001n^3 - 200n^2$ **não é** $\mathrm{O}(n^2)$

Definição

Sejam $T(n)$ e $f(n)$ funções dos inteiros no reais.
Dizemos que $T(n)$ é $\mathrm{O}(f(n))$ se existem constantes positivas c e n_0 tais que

$$T(n) \leq c\,f(n)$$

para todo $n \geq n_0$.

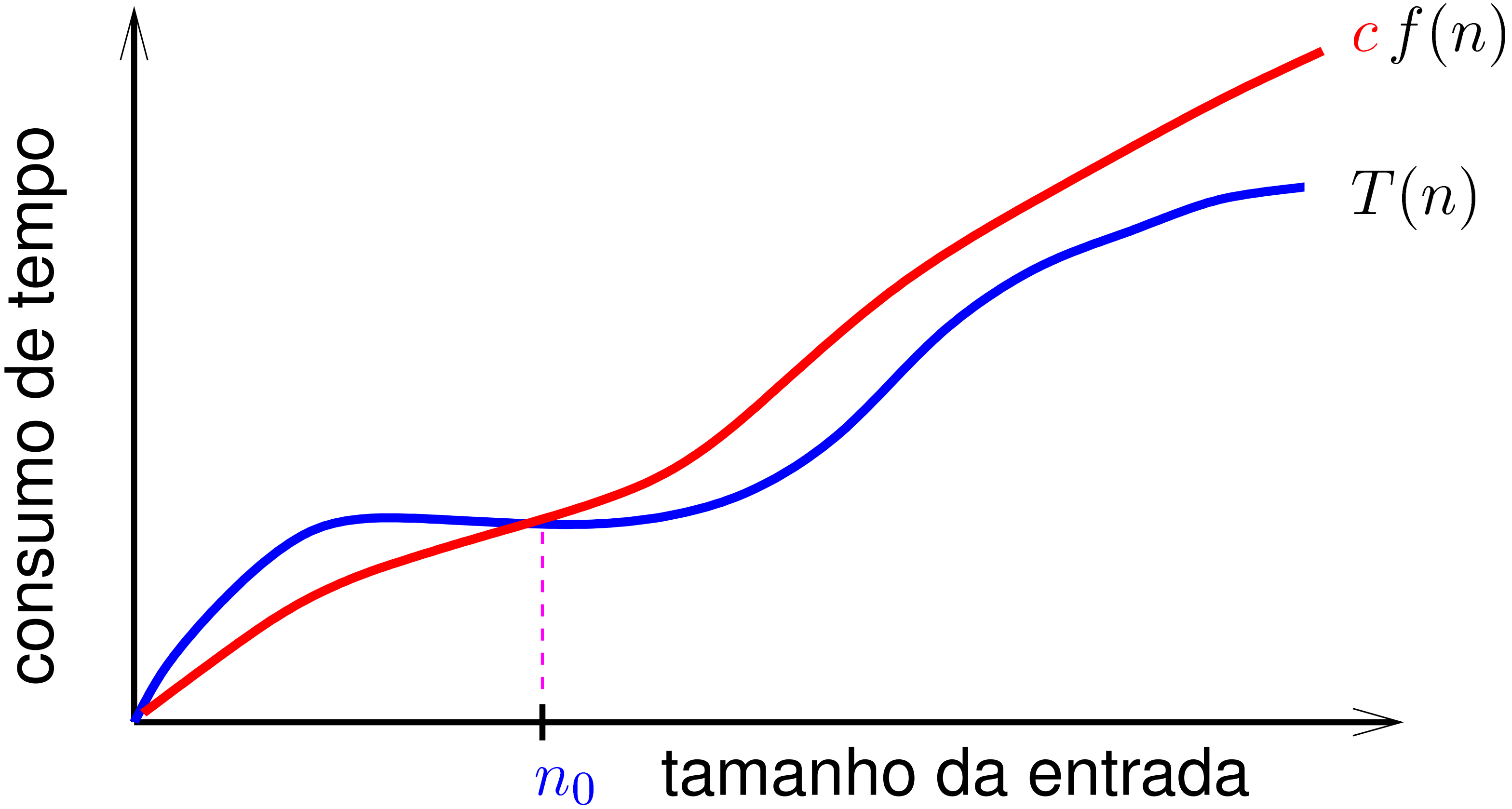


Mais informal

$T(n)$ é $\mathrm{O}(f(n))$ se existe $c > 0$ tal que

$$T(n) \leq c\,f(n)$$

para todo n suficientemente GRANDE.



Exemplos

$T(n)$ é $\mathrm{O}(f(n))$ lê-se "$T(n)$ é O de $f(n)$" ou
"$T(n)$ é da ordem de $f(n)$"

Exemplos

$T(n)$ é $\mathrm{O}(f(n))$ lê-se "$T(n)$ é O de $f(n)$" ou
"$T(n)$ é da ordem de $f(n)$"

Exemplo 1

Se $T(n) \leq 500 f(n)$ para todo $n \geq 10$, então $T(n)$ é $\mathrm{O}(f(n))$.

Exemplos

$T(n)$ é $\mathrm{O}(f(n))$ lê-se "$T(n)$ é O de $f(n)$" ou
"$T(n)$ é da ordem de $f(n)$"

Exemplo 1

Se $T(n) \leq 500 f(n)$ para todo $n \geq 10$, então $T(n)$ é $\mathrm{O}(f(n))$.

Prova: Aplique a definição com $c = 500$ e $n_0 = 10$.

Exemplos

$T(n)$ é $\mathrm{O}(f(n))$ lê-se "$T(n)$ é O de $f(n)$" ou
"$T(n)$ é da ordem de $f(n)$"

Exemplo 1

Se $T(n) \leq 500 f(n)$ para todo $n \geq 10$, então $T(n)$ é $\mathrm{O}(f(n))$.

Prova: Aplique a definição com $c = 500$ e $n_0 = 10$.

Exemplo 2

$10n^2$ é $\mathrm{O}(n^3)$.

Exemplos

$T(n)$ é $\mathrm{O}(f(n))$ lê-se "$T(n)$ é O de $f(n)$" ou
"$T(n)$ é da ordem de $f(n)$"

Exemplo 1

Se $T(n) \leq 500 f(n)$ para todo $n \geq 10$, então $T(n)$ é $\mathrm{O}(f(n))$.

Prova: Aplique a definição com $c = 500$ e $n_0 = 10$.

Exemplo 2

$10n^2$ é $\mathrm{O}(n^3)$.

Prova: Para $n \geq 0$, temos que $0 \leq 10n^2 \leq 10\, n^3$.

Outra prova: Para $n \geq 10$, temos $0 \leq 10n^2 \leq n \times n^2 = 1\, n^3$.

Mais exemplos

Exemplo 3

$\lg n$ é $\mathrm{O}(n)$.

Mais exemplos

Exemplo 3

$\lg n$ é $\mathrm{O}(n)$.

Prova: Para $n \geq 1$, tem-se que $\lg n \leq 1\, n$.

Mais exemplos

Exemplo 3

$\lg n$ é $\mathrm{O}(n)$.

Prova: Para $n \geq 1$, tem-se que $\lg n \leq 1\, n$.

Exemplo 4

$20n^3 + 10n \log n + 5$ é $\mathrm{O}(n^3)$.

Mais exemplos

Exemplo 3

$\lg n$ é $\mathrm{O}(n)$.

Prova: Para $n \geq 1$, tem-se que $\lg n \leq 1\,n$.

Exemplo 4

$20n^3 + 10n \log n + 5$ é $\mathrm{O}(n^3)$.

Prova: Para $n \geq 1$, tem-se que

$$20n^3 + 10n \lg n + 5 \leq 20n^3 + 10n^3 + 5n^3 = 35\,n^3.$$

Outra prova: Para $n \geq 10$, tem-se que

$$20n^3 + 10n \lg n + 5 \leq 20n^3 + n\,n \lg n + n \leq 20n^3 + n^3 + n^3 = 22\,n^3.$$

Mais exemplos ainda

Exemplo 5

$3\lg n + \lg\lg n$ é $\mathrm{O}(\lg n)$.

Mais exemplos ainda

Exemplo 5

$3 \lg n + \lg \lg n$ é $\mathrm{O}(\lg n)$.

Prova: Para $n \geq 2$, tem-se que

$$3 \lg n + \lg \lg n \leq 3 \lg n + \lg n = 4 \lg n.$$

[Note que $\lg \lg n$ não é definida para $n = 1$.]

Mais exemplos ainda

Exemplo 5

$3 \lg n + \lg \lg n$ é $\mathrm{O}(\lg n)$.

Prova: Para $n \geq 2$, tem-se que

$$3 \lg n + \lg \lg n \leq 3 \lg n + \lg n = 4 \lg n.$$

[Note que $\lg \lg n$ não é definida para $n = 1$.]

Exemplo 6

10^{100} é $\mathrm{O}(1)$.

Mais exemplos ainda

Exemplo 5

$3\lg n + \lg\lg n$ é $\mathrm{O}(\lg n)$.

Prova: Para $n \geq 2$, tem-se que

$$3\lg n + \lg\lg n \leq 3\lg n + \lg n = 4\lg n.$$

[Note que $\lg\lg n$ não é definida para $n = 1$.]

Exemplo 6

10^{100} é $\mathrm{O}(1)$.

Prova: Para $n \geq 1$, tem-se que

$$10^{100} = 10^{100}\, n^0 = 10^{100} \times 1.$$

[Note que n não precisa aparecer, já que estamos lidando com funções constantes.]

Uso da notação O

$\mathrm{O}(f(n)) = \{T(n) :$ existem c e n_0 tq $T(n) \leq cf(n), n \geq n_0\}$

"$T(n)$ é $\mathrm{O}(f(n))$" deve ser entendido como "$T(n) \in \mathrm{O}(f(n))$".

"$T(n) = \mathrm{O}(f(n))$" deve ser entendido como "$T(n) \in \mathrm{O}(f(n))$".

"$T(n) \leq \mathrm{O}(f(n))$" é feio.

"$T(n) \geq \mathrm{O}(f(n))$" não faz sentido!

"$T(n)$ é $g(n) + \mathrm{O}(f(n))$" significa que existe constantes positivas c e n_0 tais que

$$T(n) \leq g(n) + c\, f(n)$$

para todo $n \geq n_0$.

Nomes de classes O

classe	nome
$\mathrm{O}(1)$	constante
$\mathrm{O}(\lg n)$	logarítmica
$\mathrm{O}(n)$	linear
$\mathrm{O}(n \lg n)$	$n \log n$
$\mathrm{O}(n^2)$	quadrática
$\mathrm{O}(n^3)$	cúbica
$\mathrm{O}(n^k)$ com $k >= 1$	polinomial
$\mathrm{O}(2^n)$	exponencial[1mm]
$\mathrm{O}(a^n)$ com $a > 1$	exponencial

Exercícios

Exercício 2.A

Prove que $n^2 + 10n + 20 \ = \ \mathrm{O}(n^2)$

Exercício 2.B

Prove que $300 \ = \ \mathrm{O}(1)$

Exercício 2.C

Prove que $\lceil n/3 \rceil \ = \ \mathrm{O}(n)$
É verdade que $n = \mathrm{O}(\lfloor n/3 \rfloor)$?

Exercício 2.D

Prove que $\lg n \ = \ \mathrm{O}(\log_{10} n)$

Exercício 2.E

Prove que $n = \mathrm{O}(2^n)$

Exercício 2.F

Prove que $\lg n = \mathrm{O}(n)$

Exercício 2.G

Prove que $n/1000$ não é $\mathrm{O}(1)$

Exercício 2.H

Prove que $\frac{1}{2}n^2$ não é $\mathrm{O}(n)$

Mais exercícios

Exercício 2.I

Suponha T definida para $n = 0, 1, \ldots$
Se $T(n) = \mathrm{O}(1)$, mostre que existe c' tal que $T(n) \leq c'$ para todo $n \geq 0$.
Se $T(n) = \mathrm{O}(n)$, mostre que existe c' tal que $T(n) \leq c'n$ para todo $n \geq 1$.

Exercício 2.J

Prove que $n^2 + 999n + 9999 = \mathrm{O}(n^2)$.

Exercício 2.K

Prove que $\frac{1}{2}n(n+1) \; = \; \mathrm{O}(n^2)$.

Exercício 2.L

É verdade que $\frac{1}{100}n^2 - 999n - 9999 = \mathrm{O}(n)$? Justifique.

Exercício 2.M

Suponha que $f(n) = n^2$ quando n é par e $f(n) = n^3$ quando n é ímpar.
É verdade que $f(n) = \mathrm{O}(n^2)$?
É verdade que $f(n) = \mathrm{O}(n^3)$?
É verdade que $n^2 = \mathrm{O}(f(n))$?
É verdade que $n^3 = \mathrm{O}(f(n))$?

Mais exercícios ainda

Exercício 2.N
É verdade que $n^2 = \mathrm{O}(2^n)$?

Exercício 2.O
É verdade que $\lg n = \mathrm{O}(\sqrt{n})$?

Exercício 2.P
Suponha $f(n) = 64n \lg n$ e $g(n) = 8n^2$, com n inteiro positivo.
Para que valores de n temos $f(n) \leq g(n)$?

Exercício 2.Q (bom!)
Suponha T e f definidas para $n = 1, 2, \ldots$ Mostre que se $T(n) = \mathrm{O}(f(n))$ e $f(n) > 0$ para $n \geq 1$ então existe c' tal que $T(n) \leq c' f(n)$ para todo $n \geq 1$.

Exercício 2.R (bom!)
Faz sentido dizer "$T(n) = \mathrm{O}(n^2)$ para $n \geq 3$"?

Mais exercícios ainda ainda

Exercício 2.S

É verdade que $2^n = \mathrm{O}(n)$?
É verdade que $n = \mathrm{O}(\lg n)$?
Justifique.

Exercício 2.T

É verdade que $n + \sqrt{n}$ é $\mathrm{O}(n)$?
É verdade que n é $\mathrm{O}(\sqrt{n})$?
É verdade que $n^{2/3}$ é $\mathrm{O}(\sqrt{n})$?
É verdade que $\sqrt{n} + 1000$ é $\mathrm{O}(n)$?

Exercício 2.U

É verdade que $\lg n = \mathrm{O}(n^{1/2})$?
É verdade que $\sqrt{n} = \mathrm{O}(\lg n)$?
É verdade que $\lg n = \mathrm{O}(n^{1/3})$?
Justifique. (Sugestão: prove, por indução, que $\lg x \leq x$ para todo número real $x \geq 1$.)

Exercício 2.V

É verdade que $\lceil \lg n \rceil = \mathrm{O}(\lg n)$?

Análise com notação O

CLRS 2.1–2.2

AU 3.3, 3.6 (muito bom)

Ordenação por inserção

Algoritmo rearranja $A[p \mathinner{..} r]$ em ordem crescente

ORDENA-POR-INSERÇÃO (A, p, r)

para $j \leftarrow p + 1$ **até** r **faça**
    $chave \leftarrow A[j]$

    $i \leftarrow j - 1$
    enquanto $i \geq p$ **e** $A[i] > chave$ **faça**
        $A[i+1] \leftarrow A[i]$ $\triangleright$ desloca
        $i \leftarrow i - 1$

    $A[i+1] \leftarrow chave$ $\triangleright$ insere

Quanto tempo o algoritmo consome?

Ordenação por inserção

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

ORDENA-POR-INSERÇÃO (A, p, r)
para $j \leftarrow p + 1$ **até** r **faça**
    $chave \leftarrow A[j]$
    $i \leftarrow j - 1$
    enquanto $i \geq p$ **e** $A[i] > chave$ **faça**
        $A[i+1] \leftarrow A[i]$ $\triangleright$ desloca
        $i \leftarrow i - 1$
    $A[i+1] \leftarrow chave$ $\triangleright$ insere

Quanto tempo o algoritmo consome?

"Tamanho" do problema: $n := r - p + 1$

Consumo de tempo

linha	consumo de todas as execuções da linha
1	?
2	?
3	?
4	?
5	?
6	?
7	?
total	?

Consumo de tempo

linha	consumo de todas as execuções da linha
1	$\mathrm{O}(n)$
2	$\mathrm{O}(n)$
3	$\mathrm{O}(n)$
4	$n\mathrm{O}(n) = \mathrm{O}(n^2)$
5	$n\mathrm{O}(n) = \mathrm{O}(n^2)$
6	$n\mathrm{O}(n) = \mathrm{O}(n^2)$
7	$\mathrm{O}(n)$
total	$\mathrm{O}(3n^2 + 4n) = \mathrm{O}(n^2)$

Justificativa

Bloco de linhas 4–6 é executado $\leq n$ vezes;
cada execução consome $\mathrm{O}(n)$;
todas juntas consomem $n\mathrm{O}(n)$.

Justificativa

Bloco de linhas 4–6 é executado $\leq n$ vezes;
cada execução consome $\mathrm{O}(n)$;
todas juntas consomem $n\mathrm{O}(n)$.

Êpa!

Quem garante que $n\mathrm{O}(n) = \mathrm{O}(n^2)$?
Quem garante que $\mathrm{O}(n^2) + \mathrm{O}(n^2) + \mathrm{O}(n^2) = \mathrm{O}(3n^2)$?
Quem garante que $\mathrm{O}(3n^2 + 4n) = \mathrm{O}(n^2)$?
Veja exercícios de Mais notação O.

Justificativa

Bloco de linhas 4–6 é executado $\leq n$ vezes;
cada execução consome $\mathrm{O}(n)$;
todas juntas consomem $n\mathrm{O}(n)$.

Êpa!

Quem garante que $n\mathrm{O}(n) = \mathrm{O}(n^2)$?
Quem garante que $\mathrm{O}(n^2) + \mathrm{O}(n^2) + \mathrm{O}(n^2) = \mathrm{O}(3n^2)$?
Quem garante que $\mathrm{O}(3n^2 + 4n) = \mathrm{O}(n^2)$?
Veja exercícios de Mais notação O.

Conclusão:

$$\boxed{\text{O algoritmo consome } \mathrm{O}(n^2) \text{ unidades de tempo.}}$$

Notação O cai como uma luva!

Exercício

Exercício 3.A

Problema: rearranjar um vetor $A[p \,..\, r]$ em ordem crescente. Escreva um algoritmo “de seleção” para o problema. Analise a correção do algoritmo (ou seja, encontre e prove as invariantes apropriadas). Analise o consumo de tempo do algoritmo; use notação O.

Mais notação assintótica

CLRS 4.1

AU 4.5, p.101–108

Exercícios

Exercício 4.A
Interprete e prove a afirmação $\mathrm{O}(n^2) + \mathrm{O}(n^2) + \mathrm{O}(n^2) = \mathrm{O}(3n^2)$.

Exercício 4.B
Interprete e prove a afirmação $n\mathrm{O}(n) = \mathrm{O}(n^2)$.

Exercício 4.C
Interprete e prove a afirmação $\mathrm{O}(3n^2 + 4n) = \mathrm{O}(n^2)$.

Exercício 4.D (propriedade transitiva)
Suponha $T(n) = \mathrm{O}(f(n))$ e $f(n) = \mathrm{O}(g(n))$.
Mostre que $T(n) = \mathrm{O}(g(n))$.
Dê um exemplo interessante.

Exercício 4.E (regra da soma, caso especial)
Suponha que $T(n) = \mathrm{O}(f(n))$ e mostre que $T(n) + f(n) = \mathrm{O}(f(n))$.
Dê um exemplo interessante.

Exercício 4.E’ (regra da soma, geral)
Suponha $T_1(n) = \mathrm{O}(f_1(n))$ e $T_2(n) = \mathrm{O}(f_2(n))$. Se $f_1(n) = \mathrm{O}(f_2(n))$, mostre que $T_1(n) + T_2(n) = \mathrm{O}(f_2(n))$.

Mais exercícios

Exercício 4.F

O que significa “$T(n) = n^2 + \mathrm{O}(n)$”?
Mostre que se $T(n) = n^2 + \mathrm{O}(n)$ então $T(n) = \mathrm{O}(n^2)$.

Exercício 4.G

O que significa “$T(n) = n\mathrm{O}(\lg n)$”? Mostre que $T(n) = n\mathrm{O}(\lg n)$ se e só se $T(n) = \mathrm{O}(n \lg n)$.

Exercício 4.H

Interprete e prove a afirmação $7 \cdot \mathrm{O}(n) = \mathrm{O}(n)$.

Exercício 4.I

Interprete e prove a afirmação $\mathrm{O}(n) + \mathrm{O}(n) = \mathrm{O}(n)$.

Exercício 4.J

Prove que $\mathrm{O}(n) = \mathrm{O}(n^2)$. É verdade que $\mathrm{O}(n^2) = \mathrm{O}(n)$?

Exercício 4.K

Interprete e prove a afirmação $(n + 2) \cdot \mathrm{O}(1) = \mathrm{O}(n)$.

Mais exercícios ainda

Exercício 4.L
Interprete e prove a afirmação $\underbrace{\mathrm{O}(1) + \cdots + \mathrm{O}(1)}_{n+2} = \mathrm{O}(n)$.

Exercício 4.M
Prove que $\mathrm{O}(1) + \mathrm{O}(1) + \mathrm{O}(1) = \mathrm{O}(1)$.
É verdade que $\mathrm{O}(1) = \mathrm{O}(1) + \mathrm{O}(1) + \mathrm{O}(1)$?

Exercício 4.N
Interprete e prove a afirmação $\mathrm{O}(f) + \mathrm{O}(g) = \mathrm{O}(f + g)$.

Melhores momentos

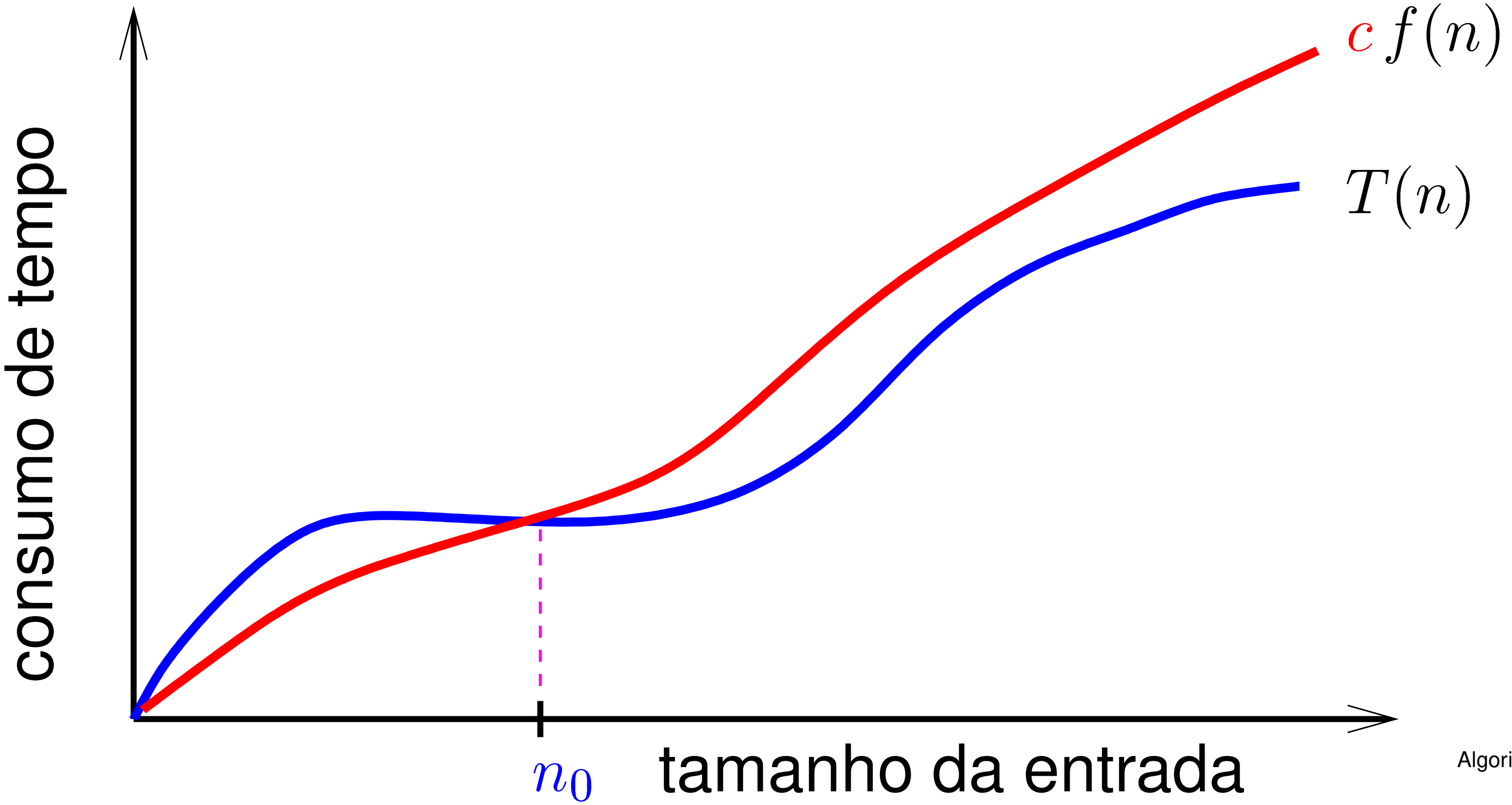
AULA 2

Definição

Sejam $T(n)$ e $f(n)$ funções dos inteiros no reais.
Dizemos que $T(n)$ é $\mathrm{O}(f(n))$ se existem constantes positivas c e n_0 tais que

$$T(n) \le c\, f(n)$$

para todo $n \ge n_0$.



Notação O

Intuitivamente...

$$\begin{aligned} \mathrm{O}(f(n)) &\approx \text{funções q não crescem mais rápido que } f(n) \\ &\approx \text{funções menores ou iguais a um múltiplo de } f(n) \end{aligned}$$

n^2 $\quad (3/2)n^2$ $\quad 9999n^2$ $\quad n^2/1000$ $\quad$ etc.

crescem todas com a mesma velocidade, são todas $\mathrm{O}(n^2)$.

- $n^2 + 99n$ é $\mathrm{O}(n^2)$
- $33n^2$ é $\mathrm{O}(n^2)$
- $9n + 2$ é $\mathrm{O}(n^2)$
- $0{,}00001n^3 - 200n^2$ **não é** $\mathrm{O}(n^2)$

Uso da notação O

$\mathrm{O}(f(n)) = \{T(n)$: existem c e n_0 tq $T(n) \leq cf(n), n \geq n_0\}$

"$T(n)$ é $\mathrm{O}(f(n))$" deve ser entendido como "$T(n) \in \mathrm{O}(f(n))$".

"$T(n) = \mathrm{O}(f(n))$" deve ser entendido como "$T(n) \in \mathrm{O}(f(n))$".

"$T(n) \leq \mathrm{O}(f(n))$" é feio.

"$T(n) \geq \mathrm{O}(f(n))$" não faz sentido!

"$T(n)$ é $g(n) + \mathrm{O}(f(n))$" significa que existem constantes positivas c e n_0 tais que

$$T(n) \leq g(n) + c\,f(n)$$

para todo $n \geq n_0$.

Ordenação por inserção

Algoritmo rearranja $A[p \mathinner{..} r]$ em ordem crescente

ORDENA-POR-INSERÇÃO (A, p, r)

para $j \leftarrow p + 1$ **até** r **faça**
    $chave \leftarrow A[j]$
    $i \leftarrow j - 1$
    enquanto $i \geq p$ **e** $A[i] > chave$ **faça**
        $A[i + 1] \leftarrow A[i]$ $\triangleright$ desloca
        $i \leftarrow i - 1$
    $A[i + 1] \leftarrow chave$ $\triangleright$ insere

Quanto tempo o algoritmo consome?
"Tamanho" do problema: $n := r - p + 1$

Consumo de tempo

linha	consumo de todas as execuções da linha
1	$\mathrm{O}(n)$
2	$\mathrm{O}(n)$
3	$\mathrm{O}(n)$
4	$n\mathrm{O}(n) = \mathrm{O}(n^2)$
5	$n\mathrm{O}(n) = \mathrm{O}(n^2)$
6	$n\mathrm{O}(n) = \mathrm{O}(n^2)$
7	$\mathrm{O}(n)$
total	$\mathrm{O}(3n^2 + 4n) = \mathrm{O}(n^2)$

Justificativa

Bloco de linhas 4–6 é executado $\leq n$ vezes;
cada execução consome $\mathrm{O}(n)$;
todas juntas consomem $n\mathrm{O}(n)$.

Êpa!

Quem garante que $n\mathrm{O}(n) = \mathrm{O}(n^2)$?
Quem garante que $\mathrm{O}(n^2) + \mathrm{O}(n^2) + \mathrm{O}(n^2) = \mathrm{O}(3n^2)$?
Quem garante que $\mathrm{O}(3n^2 + 4n) = \mathrm{O}(n^2)$?
Veja exercícios de Mais notação O.

Conclusão:

O algoritmo consome $\mathrm{O}(n^2)$ unidades de tempo.

Notação O cai como uma luva!

$n\mathrm{O}(n) = \mathrm{O}(n^2)$

"$n\mathrm{O}(n) = \mathrm{O}(n^2)$" significa "$nO(n) \subset \mathrm{O}(n^2)$".
Ou seja, se $T(n)$ é $\mathrm{O}(n)$, então $nT(n)$ é $\mathrm{O}(n^2)$.

De fato, se $T(n)$ é $\mathrm{O}(n)$ então existem constantes, digamos 10 e 10^{100}, tais que

$$T(n) \;\leq\; 10\,n$$

para todo $n \geq 10^{100}$. Desta forma,

$$n\,T(n) \;\leq\; n\,10\,n \leq 10\,n^2$$

para todo $n \geq 10^{100}$. Logo $n\,T(n)$ é $\mathrm{O}(n^2)$.

$n\mathrm{O}(n) = \mathrm{O}(n^2)$

"$n\mathrm{O}(n) = \mathrm{O}(n^2)$" significa "$nO(n) \subset \mathrm{O}(n^2)$".
Ou seja, se $T(n)$ é $\mathrm{O}(n)$, então $nT(n)$ é $\mathrm{O}(n^2)$.

De fato, se $T(n)$ é $\mathrm{O}(n)$ então existem constantes positivas c e n_0, tais que

$$T(n) \;\leq\; c\,n$$

para todo $n \geq n_0$. Desta forma,

$$n\,T(n) \;\leq\; n\,c\,n \leq c\,n^2$$

para todo $n \geq n_0$. Logo $n\,T(n)$ é $\mathrm{O}(n^2)$.

AULA 3

Mais análise com notação O

CLRS 2.1–2.2

AU 3.3, 3.6 (muito bom)

Análise da intercalação

Problema: Dados $A[p\,..\,q]$ e $A[q{+}1\,..\,r]$ crescentes, rearranjar $A[p\,..\,r]$ de modo que ele fique em ordem crescente.

Para que valores de q o problema faz sentido?

Entra:

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

Análise da intercalação

Problema: Dados $A[p\,..\,q]$ e $A[q{+}1\,..\,r]$ crescentes, rearranjar $A[p\,..\,r]$ de modo que ele fique em ordem crescente.

Para que valores de q o problema faz sentido?

Entra:

$$
\begin{array}{c|c|c|c|c|c|c|c|c|c|}
 & p & & & & q & & & & r \\
\hline
A & 22 & 33 & 55 & 77 & 99 & 11 & 44 & 66 & 88 \\
\hline
\end{array}
$$

Sai:

$$
\begin{array}{c|c|c|c|c|c|c|c|c|c|}
 & p & & & & q & & & & r \\
\hline
A & 11 & 22 & 33 & 44 & 55 & 66 & 77 & 88 & 99 \\
\hline
\end{array}
$$

Intercalação

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

B									

Intercalação

	k								
A									

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

Intercalação

		k							
A	11								

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

Intercalação

			k						
A	11	22							

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

Intercalação

				k					
A	11	22	33						

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

Intercalação

					k				
A	11	22	33	44					

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

Intercalação

						k			
A	11	22	33	44	55				

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

Intercalação

							k		
A	11	22	33	44	55	66			

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

Intercalação

								k	
A	11	22	33	44	55	66	77		

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

Intercalação

									k
A	11	22	33	44	55	66	77	88	

					$i = j$				
B	22	33	55	77	99	88	66	44	11

Intercalação

A	11	22	33	44	55	66	77	88	99

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

Intercalação

INTERCALA (A, p, q, r)

$\triangleright$ $B[p\,.\,.\,r]$ é um vetor auxiliar

para $i \leftarrow p$ **até** q **faça**

$\quad B[i] \leftarrow A[i]$

para $j \leftarrow q+1$ **até** r **faça**

$\quad B[r+q+1-j] \leftarrow A[j]$

$i \leftarrow p$

$j \leftarrow r$

para $k \leftarrow p$ **até** r **faça**

$\quad$ **se** $B[i] \leq B[j]$

$\quad\quad$ **então** $A[k] \leftarrow B[i]$

$\quad\quad\quad\quad i \leftarrow i+1$

$\quad\quad$ **senão** $A[k] \leftarrow B[j]$

$\quad\quad\quad\quad j \leftarrow j-1$

Consumo de tempo

Se a execução de cada linha de código consome 1 unidade de tempo o consumo total é:

linha	todas as execuções da linha
1	?
2	?
3	?
4	?
5	?
6	?
7	?
8	?
9–12	?
total	?

Consumo de tempo

Se a execução de cada linha de código consome 1 unidade de tempo o consumo total é ($n := r - p + 1$):

linha	todas as execuções da linha
1	$= \quad q - p + 2 \; = \; n - r + q + 1$
2	$= \quad q - p + 1 \; = \; n - r + q$
3	$= \quad r - (q+1) + 2 \; = \; n - q + p$
4	$= \quad r - (q+1) + 1 \; = \; n - q + p - 1$
5	$= \quad 1$
6	$= \quad 1$
7	$= \quad r - p + 2 \; = \; n + 1$
8	$= \quad r - p + 1 \; = \; n$
9–12	$= \quad 2\,(r - p + 1) \; = \; 2n$
total	$= \quad 8n - 2(r - p + 1) + 5 = \; 6n + 5$

Consumo de tempo em O

Quanto tempo consome em função de $n := r - p + 1$?

linha	consumo de todas as execuções da linha
1–4	?
5–6	?
7	?
8	?
9–12	?
total	?

Consumo de tempo

Quanto tempo consome em função de $n := r - p + 1$?

linha	consumo de todas as execuções da linha
1–4	$\mathrm{O}(n)$
5–6	$\mathrm{O}(1)$
7	$n\mathrm{O}(1) = \mathrm{O}(n)$
8	$n\mathrm{O}(1) = \mathrm{O}(n)$
9–12	$n\mathrm{O}(1) = \mathrm{O}(n)$
total	$\mathrm{O}(4n + 1)$ = $\mathrm{O}(n)$

Consumo de tempo

Quanto tempo consome em função de $n := r - p + 1$?

linha	consumo de todas as execuções da linha
1–4	$\mathrm{O}(n)$
5–6	$\mathrm{O}(1)$
7	$n\mathrm{O}(1) = \mathrm{O}(n)$
8	$n\mathrm{O}(1) = \mathrm{O}(n)$
9–12	$n\mathrm{O}(1) = \mathrm{O}(n)$
total	$\mathrm{O}(4n + 1) = \mathrm{O}(n)$

Conclusão:

$$\boxed{\text{O algoritmo consome } \mathrm{O}(n) \text{ unidades de tempo.}}$$

Exercício

Exercício 5.A

Analise a correção e o consumo de tempo do seguinte algoritmo:

```
EXERC (A, p, r)
    q ← ⌊(p + r)/2⌋
    ORDENA-POR-INSERÇÃO (A, p, q)
    ORDENA-POR-INSERÇÃO (A, q + 1, r)
    INTERCALA (A, p, q, r)
```

Irmãos de O

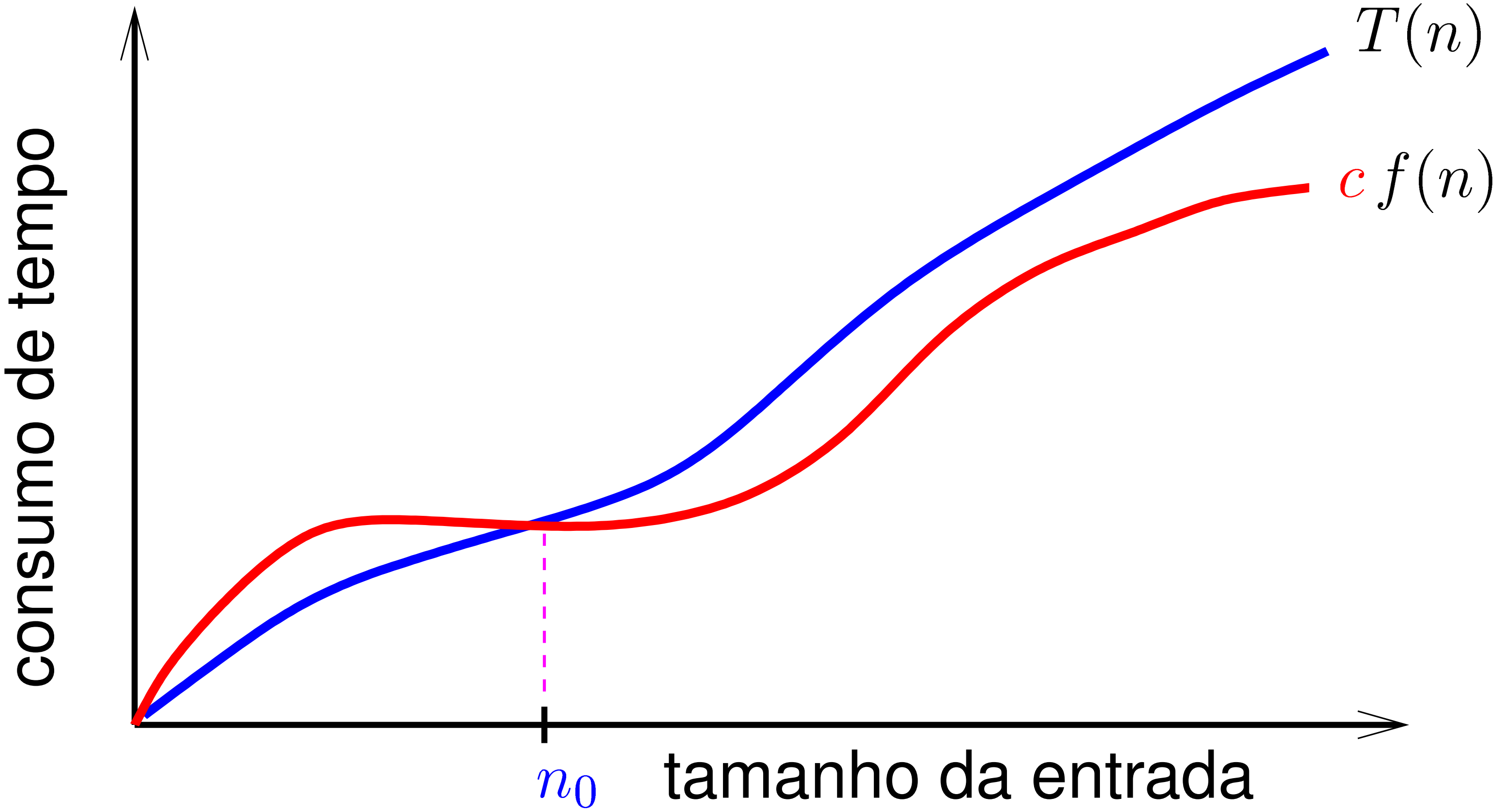
CLRS 3.1

Definição

Dizemos que $T(n)$ é $\Omega(f(n))$ se existem constantes positivas c e n_0 tais que

$$c\, f(n) \;\leq\; T(n)$$

para todo $n \geq n_0$.

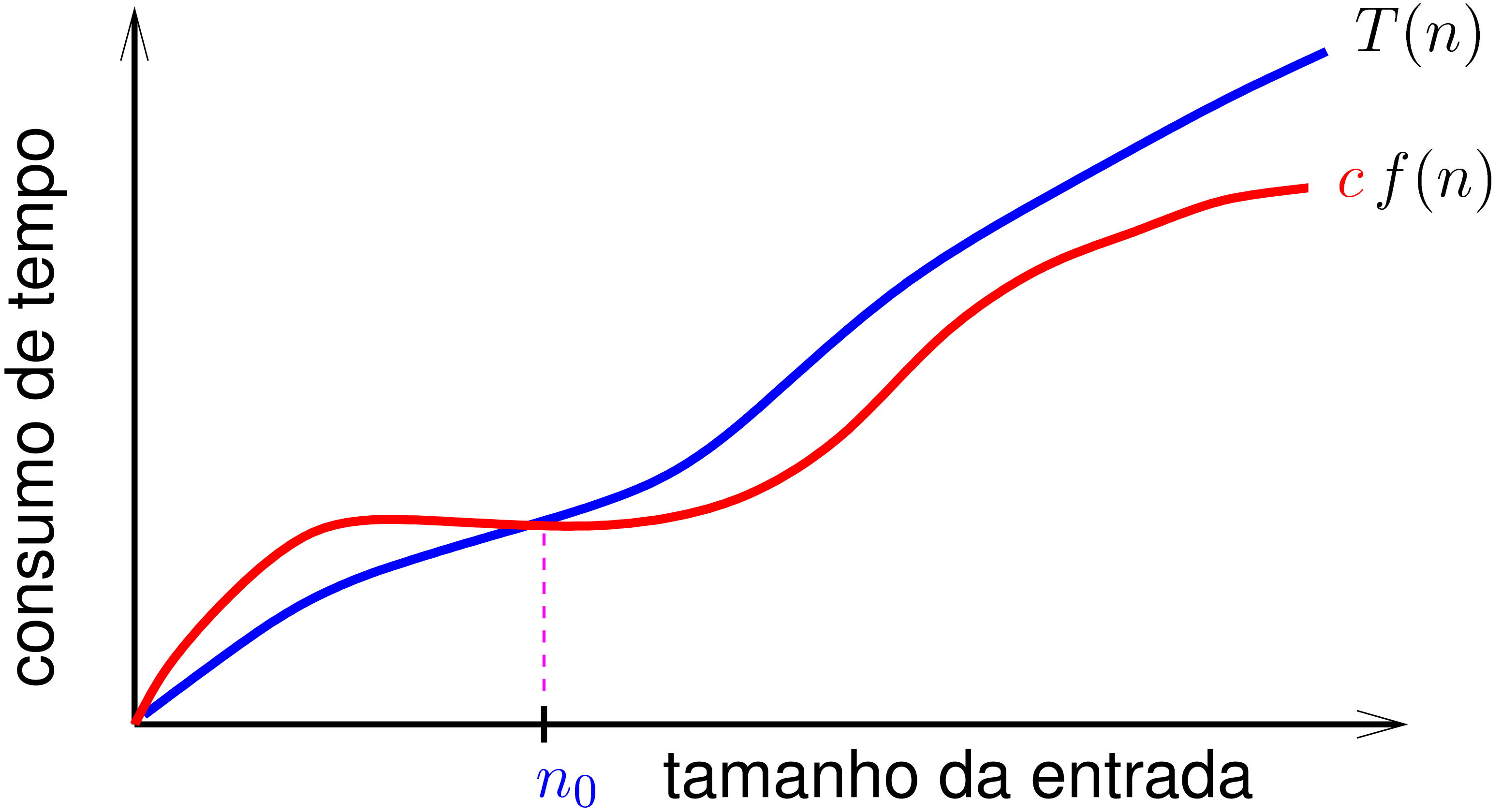


Mais informal

$T(n) = \Omega(f(n))$ se existe $c > 0$ tal que

$$c\, f(n) \leq T(n)$$

para todo n suficientemente GRANDE.



Exemplos

Exemplo 1

Se $T(n) \geq 0.001n^2$ para todo $n \geq 8$, então $T(n)$ é $\Omega(n^2)$.

Exemplos

Exemplo 1
Se $T(n) \geq 0.001n^2$ para todo $n \geq 8$, então $T(n)$ é $\Omega(n^2)$.

Prova: Aplique a definição com $c = 0.001$ e $n_0 = 8$.

Exemplo 2

O consumo de tempo do INTERCALA é $\mathrm{O}(n)$ e também $\Omega(n)$.

Exemplo 2

O consumo de tempo do INTERCALA é $\mathrm{O}(n)$ e também $\Omega(n)$.

linha		todas as execuções da linha
1	$=$	$q - p + 2 \;=\; n - r + q + 1$
2	$=$	$q - p + 1 \;=\; n - r + q$
3	$=$	$r - (q+1) + 2 \;=\; n - q + p$
4	$=$	$r - (q+1) + 1 \;=\; n - q + p - 1$
5	$=$	1
6	$=$	1
7	$=$	$r - p + 2 \;=\; n + 1$
8	$=$	$r - p + 1 \;=\; n$
9–12	$=$	$2\,(r - p + 1) \;=\; 2n$
total	$=$	$8n - 2(r - p + 1) + 5 = \; 6n + 5$

Exemplo 3

Se $T(n)$ é $\Omega(f(n))$, então $f(n)$ é $\mathrm{O}(T(n))$.

Exemplo 3

Se $T(n)$ é $\Omega(f(n))$, então $f(n)$ é $\mathrm{O}(T(n))$.

Prova: Se $T(n)$ é $\Omega(f(n))$, então existem constantes positivas c e n_0 tais que

$$c\, f(n) \;\leq\; T(n)$$

para todo $n \geq n_0$. Logo,

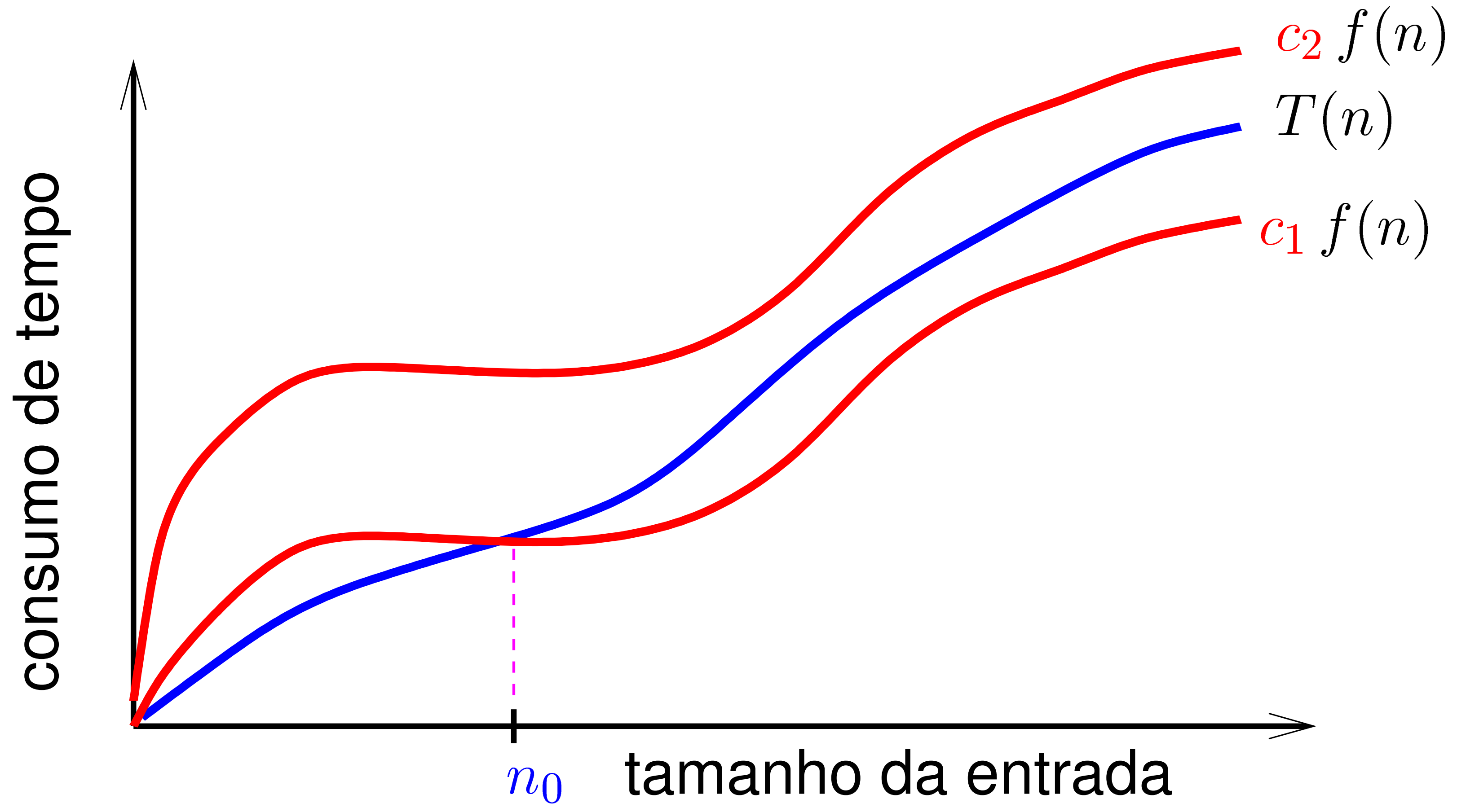
$$f(n) \;\leq\; 1/c\, T(n)$$

para todo $n \geq n_0$. Portanto, $f(n)$ é $\mathrm{O}(T(n))$.

Definição

Sejam $T(n)$ e $f(n)$ funções dos inteiros no reais.
Dizemos que $T(n)$ é $\Theta(f(n))$ se

$$T(n) \text{ é } \mathrm{O}(f(n)) \text{ e } T(n) \text{ é } \Omega(f(n)).$$

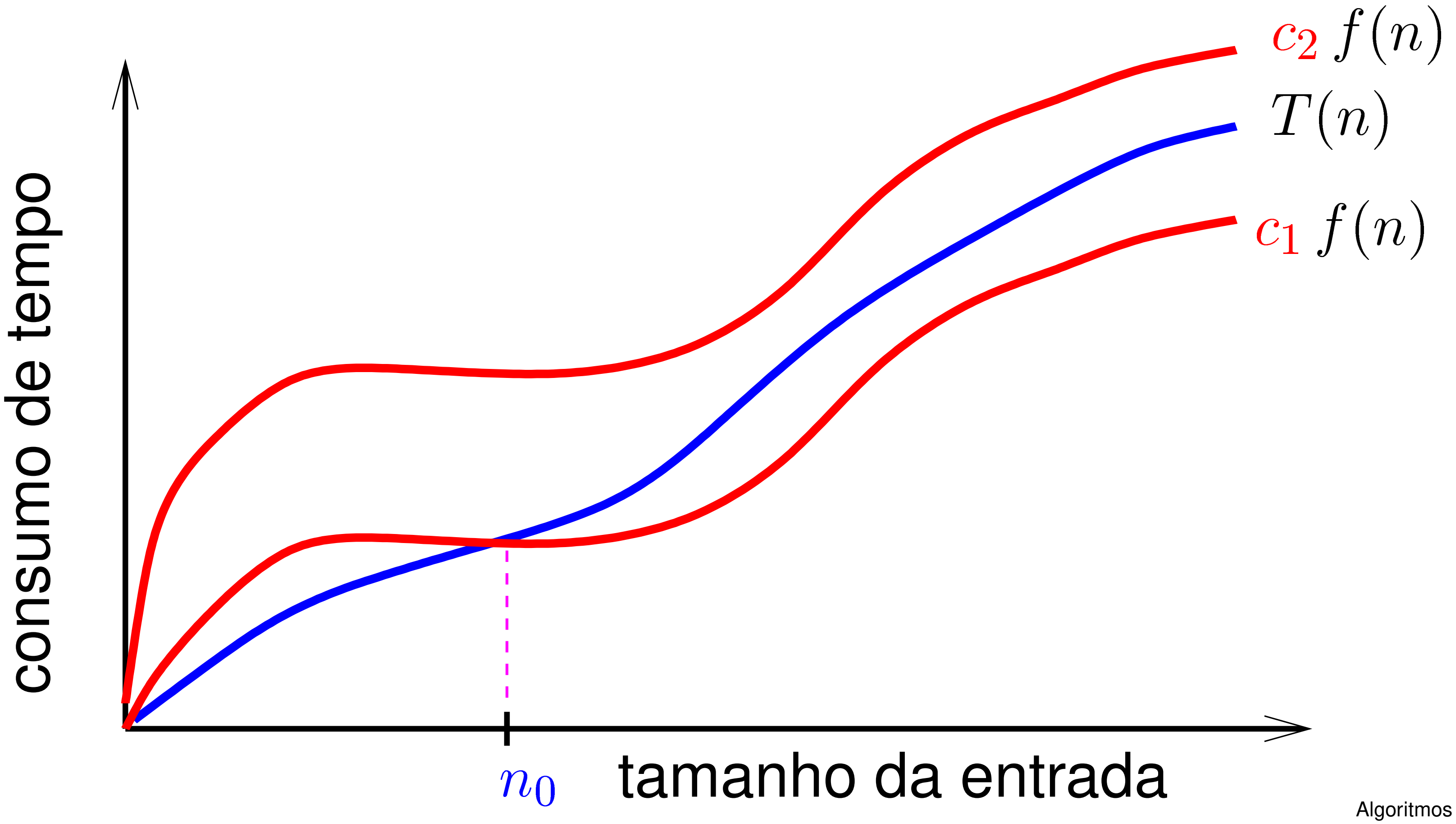


Definição

Dizemos que $T(n)$ é $\Theta(f(n))$ se se existem constantes positivas c_1, c_2 e n_0 tais que

$$c_1\, f(n) \;\leq\; T(n) \;\leq\; c_2\, f(n)$$

para todo $n \geq n_0$.



Tamanho máximo de problemas

Suponha que cada operação consome 1 microsegundo ($1\mu s$).

consumo de tempo(μs)	Tamanho máximo de problemas (n) 1 segundo	1 minuto	1 hora
$400n$	2500	150000	9000000
$20\,n\,\lceil \lg n \rceil$	4096	166666	7826087
$2n^2$	707	5477	42426
n^4	31	88	244
2^n	19	25	31

Michael T. Goodrich e Roberto Tamassia, *Projeto de Algoritmos*, Bookman.

Nomes de classes Θ

classe	nome
$\Theta(1)$	constante
$\Theta(\log n)$	logarítmica
$\Theta(n)$	linear
$\Theta(n \log n)$	$n \log n$
$\Theta(n^2)$	quadrática
$\Theta(n^3)$	cúbica
$\Theta(n^k)$ com $k >= 1$	polinomial
$\Theta(2^n)$	exponencial
$\Theta(a^n)$ com $a > 1$	exponencial

Palavras de Cautela

Suponha que A e B são algoritmos para um mesmo problema. Suponha que o consumo de tempo de A é "essencialmente" $100\,n$ e que o consumo de tempo de B é "essencialmente" $n \log_{10} n$.

Palavras de Cautela

Suponha que A e B são algoritmos para um mesmo problema. Suponha que o consumo de tempo de A é "essencialmente" $100\,n$ e que o consumo de tempo de B é "essencialmente" $n \log_{10} n$.

$100\,n$ é $\Theta(n)$ e $n \log_{10} n$ é $\Theta(n \lg n)$.
Logo, A é assintoticamente mais eficiente que B.

Palavras de Cautela

Suponha que A e B são algoritmos para um mesmo problema. Suponha que o consumo de tempo de A é "essencialmente" $100\,n$ e que o consumo de tempo de B é "essencialmente" $n \log_{10} n$.

$100\,n$ é $\Theta(n)$ e $n \log_{10} n$ é $\Theta(n \lg n)$.
Logo, A é assintoticamente mais eficiente que B.

A é mais eficiente que B para $n \geq 10^{100}$.
10^{100} = um *googol*
$\approx$ número de átomos no universo observável
$=$ número ENORME

Palavras de Cautela

Conclusão:

> Lembre das constantes e termos de baixa ordem que estão "escondidos" na notação assintótica.

Em geral um algoritmo que consome tempo $\Theta(n \lg n)$, e com fatores constantes razoáveis, é bem eficiente.

Um algoritmo que consome tempo $\Theta(n^2)$ pode, algumas vezes ser satisfatório.

Um algoritmo que consome tempo $\Theta(2^n)$ é dificilmente aceitável.

Do ponto de vista de AA, eficiente $=$ polinomial.

Exercícios

Exercício 6.A
Já sabemos que ORDENA-POR-INSERÇÃO é $\mathrm{O}(n^2)$.
Mostre que o algoritmo é $\Omega(n)$.

Exercício 6.B
Mostre que ORDENA-POR-INSERÇÃO é $\Omega(n^2)$ no pior caso.

Exercício 6.C
Mostre que ORDENA-POR-INSERÇÃO é $\mathrm{O}(n)$ no melhor caso.

Mais exercícios

Exercício 6.D

Prove que $n^2 + 10n + 20 = \Omega(n^2)$. Prove que $n^2 - 10n - 20 = \Theta(n^2)$.

Exercício 6.E

Prove que $n = \Omega(\lg n)$.

Exercício 6.F

Prove que $\lg n = \Theta(\log_{10} n)$.

Exercício 6.G

É verdade que $2^n = \Omega(3^n)$?

Exercício 6.H

É verdade que $2n^3 + 5\sqrt{n} = \Theta(n^3)$?

Mais exercícios ainda

Exercício 6.I

Suponha que os algoritmos $\mathcal{A}$ e $\mathcal{B}$ só dependem de um parâmetro n. Suponha ainda que $\mathcal{A}$ consome $S(n)$ unidades de tempo enquanto $\mathcal{B}$ consome $T(n)$ unidades de tempo. Quero provar que algoritmo $\mathcal{A}$ é pelo menos tão eficiente quanto o algoritmo $\mathcal{B}$ (no sentido assintótico). Devo mostrar que existe $f(n)$ tal que

$S(n) = \mathrm{O}(f(n))$ e $T(n) = \mathrm{O}(f(n))$?

$S(n) = \mathrm{O}(f(n))$ e $T(n) = \Omega(f(n))$?

$S(n) = \Omega(f(n))$ e $T(n) = \mathrm{O}(f(n))$?

$S(n) = \Omega(f(n))$ e $T(n)\Omega(f(n))$?

Que devo fazer para mostrar que $\mathcal{A}$ é mais eficiente que $\mathcal{B}$?

Exercício 6.J

Mostre que o consumo de tempo do algoritmo INTERCALA é $\Theta(n)$, sendo n o número de elementos do vetor que o algoritmo recebe.

Recursão

CLRS 2.3
AU 2.6, 2.7, 2.9

"To understand recursion, we must first understand recursion.”

Recursão

Recursão: resolve um problema a partir das soluções de seus subproblemas

Recursão

Recursão: resolve um problema a partir das soluções de seus subproblemas

Problema: Rearranjar $A[p \,.\,.\, r]$ de modo que ele fique em ordem crescente.

Entra:

$$
\begin{array}{c|c|c|c|c|c|c|c|c|c|}
 & p & & & & & & & & r \\
\cline{2-10}
A & 55 & 33 & 66 & 44 & 99 & 11 & 77 & 22 & 88 \\
\cline{2-10}
\end{array}
$$

Recursão

Recursão: resolve um problema a partir das soluções de seus subproblemas

Problema: Rearranjar $A[p \, . \, . \, r]$ de modo que ele fique em ordem crescente.

Entra:

$$A \quad \begin{array}{|c|c|c|c|c|c|c|c|c|} \multicolumn{1}{c}{p} & \multicolumn{7}{c}{} & \multicolumn{1}{c}{r} \\ \hline 55 & 33 & 66 & 44 & 99 & 11 & 77 & 22 & 88 \\ \hline \end{array}$$

Sai:

$$A \quad \begin{array}{|c|c|c|c|c|c|c|c|c|} \multicolumn{1}{c}{p} & \multicolumn{7}{c}{} & \multicolumn{1}{c}{r} \\ \hline 11 & 22 & 33 & 44 & 55 & 66 & 77 & 88 & 99 \\ \hline \end{array}$$

Divisão-e-conquista

Algoritmos por divisão-e-conquista têm três passos em cada nível da recursão:

Dividir: o problema é dividido em subproblemas de tamanho menor;

Conquistar: os subproblemas são resolvidos recursivamente e subproblemas "pequenos" são resolvidos diretamente;

Combinar: as solução dos subproblemas são combinadas para obter uma solução do problema original.

Exemplo: ordenação por intercalação (Merge-sort).

Merge-Sort

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

Merge-Sort

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

	p		q		r				
A	55	33	66	44	99				

Merge-Sort

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

	p		q		r				
A	55	33	66	44	99				

	p	q	r						
A	55	33	66						

Merge-Sort

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

	p		q		r				
A	55	33	66	44	99				

	p	q	r						
A	55	33	66						

	p	r							
A	55	33							

Merge-Sort

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

	p		q		r				
A	33	55	66	44	99				

	p	q	r						
A	33	55	66						

	p	r							
A	33	55							

Merge-Sort

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

	p		q		r				
A	33	55	66	44	99				

	p	q	r						
A	33	55	66						

			$p = r$						
A			66						

Merge-Sort

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

	p		q		r				
A	33	55	66	44	99				

	p	q	r						
A	33	55	66						

Merge-Sort

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

	p		q		r				
A	33	55	66	44	99				

Merge-Sort

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

	p		q		r				
A	33	55	66	44	99				

				p	r				
A				44	99				

Merge-Sort

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

	p		q		r				
A	33	55	66	44	99				

				p	r				
A				44	99				

Merge-Sort

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

	p		q		r				
A	33	55	66	44	99				

Merge-Sort

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

	p		q		r				
A	33	44	55	66	99				

Merge-Sort

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

Merge-Sort

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

						p			r
A						11	77	22	88

Merge-Sort

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

						p			r
A						11	77	22	88

						p	r		
A						11	77		

Merge-Sort

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

						p			r
A						11	77	22	88

						p	r		
A						11	77		

Merge-Sort

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

						p			r
A						11	77	22	88

Merge-Sort

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

						p			r
A						11	77	22	88

								p	r
A								22	88

Merge-Sort

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

						p			r
A						11	77	22	88

								p	r
A								22	88

Merge-Sort

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

						p			r
A						11	77	22	88

Merge-Sort

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

						p			r
A						11	22	77	88

Merge-Sort

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

Merge-Sort

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

Merge-Sort

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

Merge-Sort

Rearranja $A[p \mathrel{.\,.} r]$, com $p \leq r$, em ordem crescente.

```
MERGE-SORT (A, p, r)
    se p < r
        então q ← ⌊(p + r)/2⌋
              MERGE-SORT (A, p, q)
              MERGE-SORT (A, q + 1, r)
              INTERCALA (A, p, q, r)
```

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

Merge-Sort

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

```
MERGE-SORT (A, p, r)
  se p < r
     então q ← ⌊(p + r)/2⌋
           MERGE-SORT (A, p, q)
           MERGE-SORT (A, q + 1, r)
           INTERCALA (A, p, q, r)
```

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

Merge-Sort

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

MERGE-SORT (A, p, r)
se $p < r$
  então $q \leftarrow \lfloor (p+r)/2 \rfloor$
    MERGE-SORT (A, p, q)
    MERGE-SORT $(A, q+1, r)$
    INTERCALA (A, p, q, r)

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

Merge-Sort

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

MERGE-SORT (A, p, r)
1 **se** $p < r$
2 **então** $q \leftarrow \lfloor (p+r)/2 \rfloor$
3 MERGE-SORT (A, p, q)
4 MERGE-SORT $(A, q+1, r)$
5 **INTERCALA** (A, p, q, r)

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

Merge-Sort

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

MERGE-SORT (A, p, r)
1 **se** $p < r$
2 **então** $q \leftarrow \lfloor (p + r)/2 \rfloor$
3 MERGE-SORT (A, p, q)
4 MERGE-SORT $(A, q + 1, r)$
5 INTERCALA (A, p, q, r)

O algoritmo está correto?

Merge-Sort

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

MERGE-SORT (A, p, r)
1 **se** $p < r$
2 **então** $q \leftarrow \lfloor (p+r)/2 \rfloor$
3 MERGE-SORT (A, p, q)
4 MERGE-SORT $(A, q+1, r)$
5 INTERCALA (A, p, q, r)

O algoritmo está correto?
A correção do algoritmo, que se apóia na correção do INTERCALA, pode ser demonstrada por indução em $n := r - p + 1$.

Merge-Sort

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

MERGE-SORT (A, p, r)
1 **se** $p < r$
2 **então** $q \leftarrow \lfloor (p + r)/2 \rfloor$
3 MERGE-SORT (A, p, q)
4 MERGE-SORT $(A, q + 1, r)$
5 INTERCALA (A, p, q, r)

Consumo de tempo?

Merge-Sort

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

MERGE-SORT (A, p, r)
se $p < r$
  então $q \leftarrow \lfloor (p + r)/2 \rfloor$
    MERGE-SORT (A, p, q)
    MERGE-SORT $(A, q + 1, r)$
    INTERCALA (A, p, q, r)

Consumo de tempo?

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

Merge-Sort

MERGE-SORT (A, p, r)
1 **se** $p < r$
2 **então** $q \leftarrow \lfloor (p + r)/2 \rfloor$
3 MERGE-SORT (A, p, q)
4 MERGE-SORT $(A, q + 1, r)$
5 INTERCALA (A, p, q, r)

linha	consumo na linha
1	?
2	?
3	?
4	?
5	?

$T(n) = ?$

Melhores momentos

AULA 3

Análise da intercalação

Problema: Dados $A[p\,..\,q]$ e $A[q{+}1\,..\,r]$ crescentes, rearranjar $A[p\,..\,r]$ de modo que ele fique em ordem crescente.

Entra:

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

Sai:

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

Intercalação

INTERCALA (A, p, q, r)

$\triangleright$ $B[p\,..\,r]$ é um vetor auxiliar
para $i \leftarrow p$ **até** q **faça**
    $B[i] \leftarrow A[i]$
para $j \leftarrow q+1$ **até** r **faça**
    $B[r+q+1-j] \leftarrow A[j]$
$i \leftarrow p$
$j \leftarrow r$
para $k \leftarrow p$ **até** r **faça**
    se $B[i] \leq B[j]$
        então $A[k] \leftarrow B[i]$
                $i \leftarrow i+1$
        senão $A[k] \leftarrow B[j]$
                $j \leftarrow j-1$

Consumo de tempo

Se a execução de cada linha de código consome 1 unidade de tempo o consumo total é ($n := r - p + 1$):

linha	todas as execuções da linha
1	$= \quad q - p + 2 \; = \; n - r + q + 1$
2	$= \quad q - p + 1 \; = \; n - r + q$
3	$= \quad r - (q + 1) + 2 \; = \; n - q + p$
4	$= \quad r - (q + 1) + 1 \; = \; n - q + p - 1$
5	$= \quad 1$
6	$= \quad 1$
7	$= \quad r - p + 2 \; = \; n + 1$
8	$= \quad r - p + 1 \; = \; n$
9–12	$= \quad 2\,(r - p + 1) \; = \; 2n$
total	$= \quad 8n - 2(r - p + 1) + 5 = \; 6n + 5$

Consumo de tempo

Quanto tempo consome em função de $n := r - p + 1$?

linha	consumo de todas as execuções da linha
1–4	$\mathrm{O}(n)$
5–6	$\mathrm{O}(1)$
7	$n\mathrm{O}(1) = \mathrm{O}(n)$
8	$n\mathrm{O}(1) = \mathrm{O}(n)$
9–12	$n\mathrm{O}(1) = \mathrm{O}(n)$
total	$\mathrm{O}(4n + 1)$ = $\mathrm{O}(n)$

Conclusão:

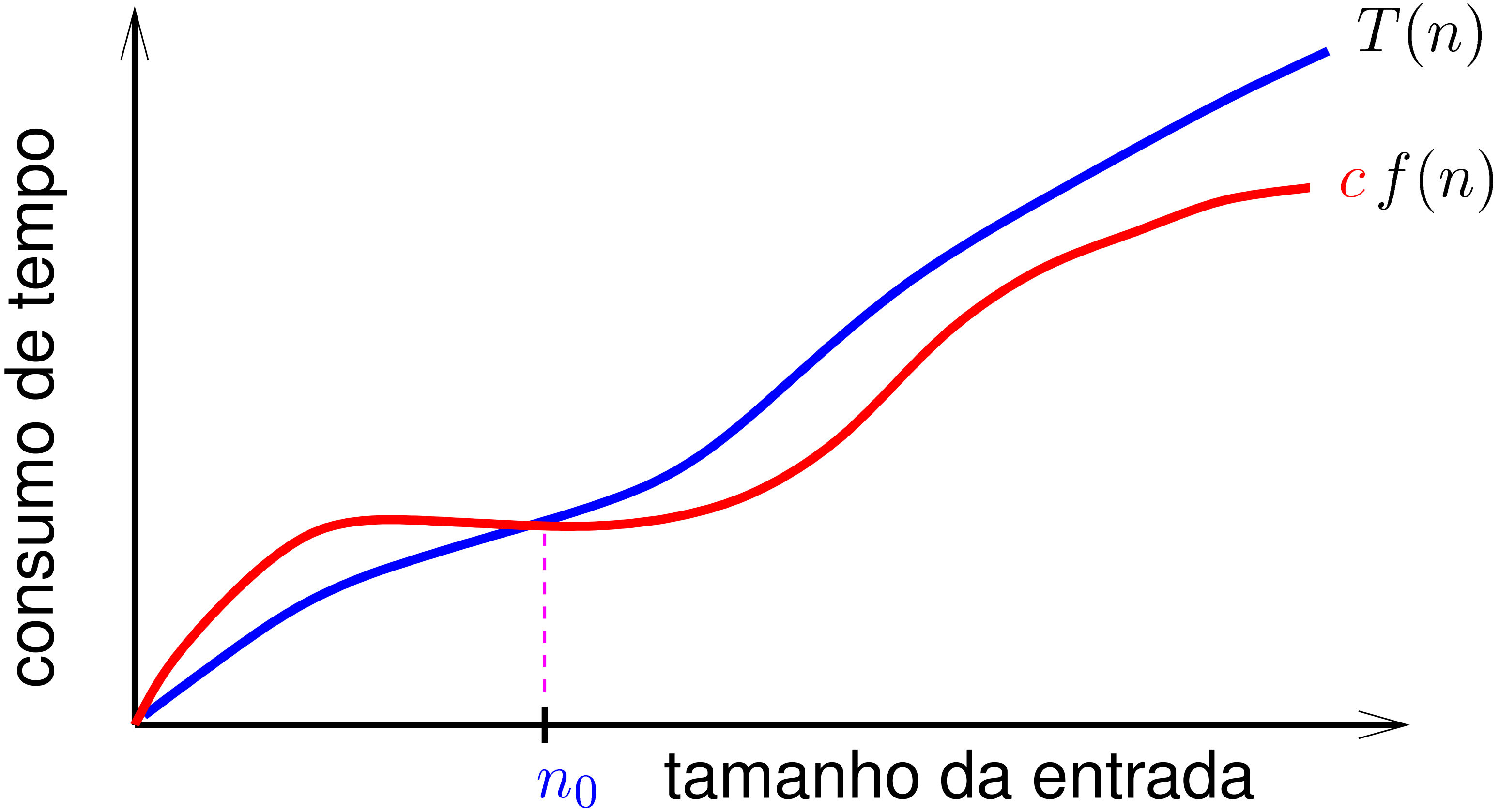
O algoritmo consome $\mathrm{O}(n)$ unidades de tempo.

Definição

Dizemos que $T(n)$ é $\Omega(f(n))$ se existem constantes positivas c e n_0 tais que

$$c\,f(n) \;\leq\; T(n)$$

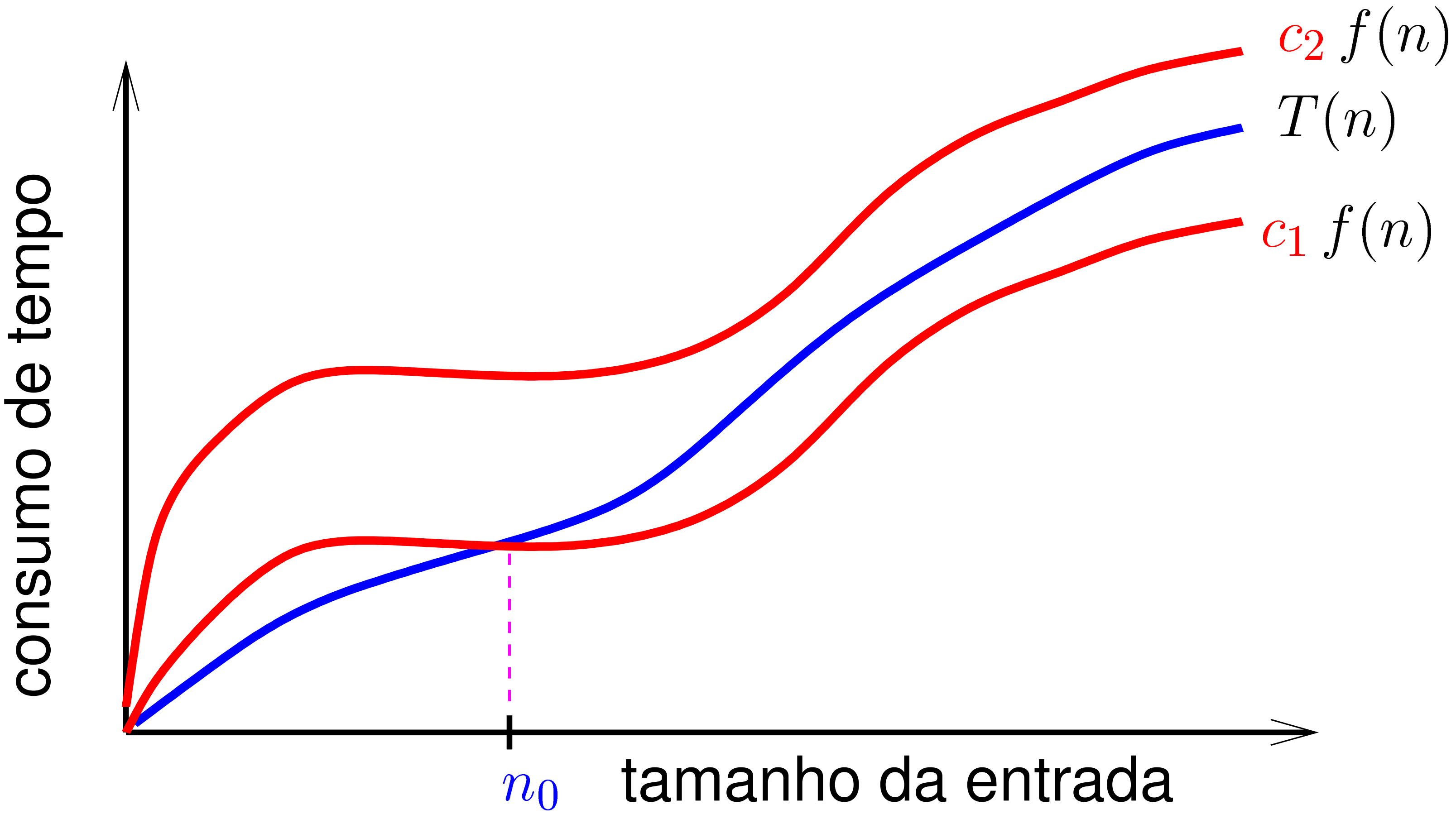
para todo $n \geq n_0$.



Definição

Sejam $T(n)$ e $f(n)$ funções dos inteiros no reais.
Dizemos que $T(n)$ é $\Theta(f(n))$ se

$$T(n) \text{ é } \mathrm{O}(f(n)) \text{ e } T(n) \text{ é } \Omega(f(n)).$$

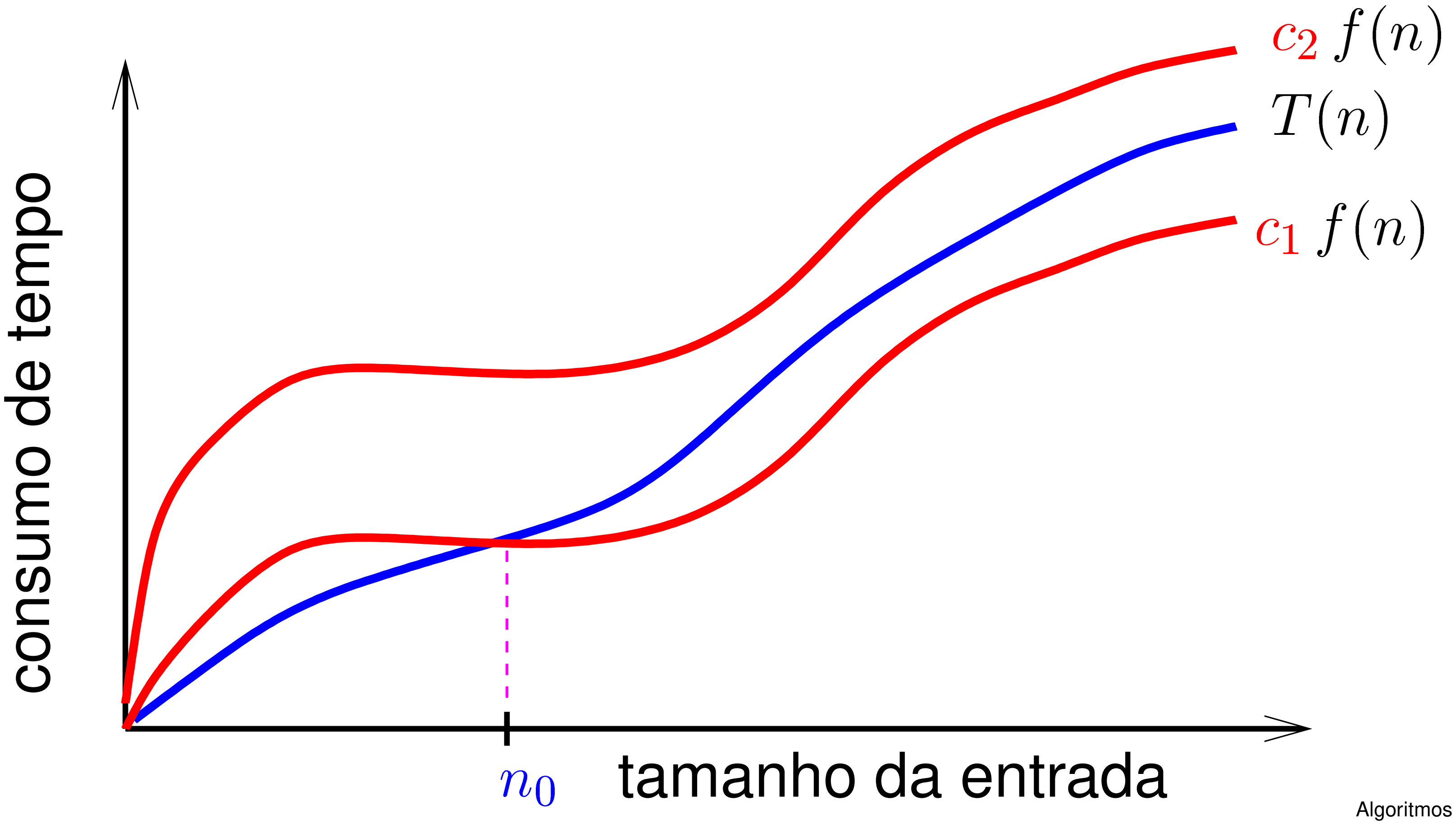


Definição

Dizemos que $T(n)$ é $\Theta(f(n))$ se se existem constantes positivas c_1, c_2 e n_0 tais que

$$c_1\, f(n) \ \le\ T(n) \ \le\ c_2\, f(n)$$

para todo $n \ge n_0$.



Exercícios

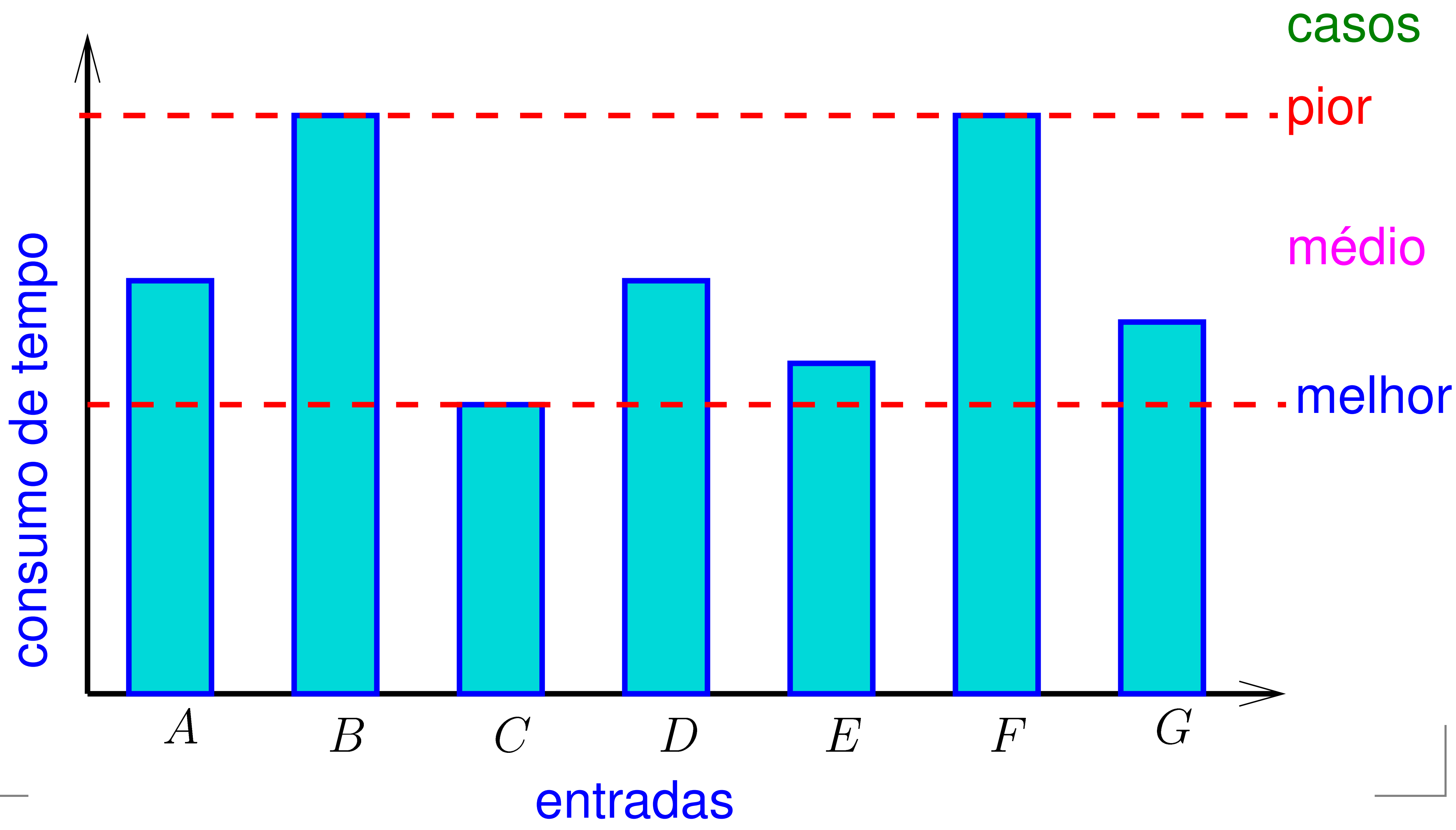
Exercício 6.A
Já sabemos que ORDENA-POR-INSERÇÃO é $\mathrm{O}(n^2)$.
Mostre que o algoritmo é $\Omega(n)$.

Exercício 6.B
Mostre que ORDENA-POR-INSERÇÃO é $\Omega(n^2)$ no pior caso.

Exercício 6.C
Mostre que ORDENA-POR-INSERÇÃO é $\mathrm{O}(n)$ no melhor caso.

Consumo de tempo

Mínimo, médio ou máximo?
Melhor caso, caso médio, pior caso?



AULA 4

Recursão (continuação)

CLRS 2.3
AU 2.6, 2.7, 2.9

"To understand recursion, we must first understand recursion.”

Merge-Sort

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

MERGE-SORT (A, p, r)
se $p < r$
  então $q \leftarrow \lfloor (p+r)/2 \rfloor$
    MERGE-SORT (A, p, q)
    MERGE-SORT $(A, q+1, r)$
    INTERCALA (A, p, q, r)

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

Merge-Sort

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

```
MERGE-SORT (A, p, r)
    se p < r
        então q ← ⌊(p + r)/2⌋
            MERGE-SORT (A, p, q)
            MERGE-SORT (A, q + 1, r)
            INTERCALA (A, p, q, r)
```

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

Merge-Sort

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

MERGE-SORT (A, p, r)
  se $p < r$
    então $q \leftarrow \lfloor (p + r)/2 \rfloor$
      MERGE-SORT (A, p, q)
      MERGE-SORT $(A, q + 1, r)$

      INTERCALA (A, p, q, r)

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

Merge-Sort

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

MERGE-SORT (A, p, r)
  se $p < r$
    então $q \leftarrow \lfloor (p + r)/2 \rfloor$
      MERGE-SORT (A, p, q)
      MERGE-SORT $(A, q + 1, r)$
      INTERCALA (A, p, q, r)

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

Merge-Sort

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

MERGE-SORT (A, p, r)
1 **se** $p < r$
2 **então** $q \leftarrow \lfloor (p + r)/2 \rfloor$
3 MERGE-SORT (A, p, q)
4 MERGE-SORT $(A, q + 1, r)$
5 INTERCALA (A, p, q, r)

O algoritmo está correto?

Merge-Sort

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

MERGE-SORT (A, p, r)
se $p < r$
  então $q \leftarrow \lfloor (p+r)/2 \rfloor$
    MERGE-SORT (A, p, q)
    MERGE-SORT $(A, q+1, r)$
    INTERCALA (A, p, q, r)

O algoritmo está correto?
A correção do algoritmo, que se apóia na correção do INTERCALA, pode ser demonstrada por indução em $n := r - p + 1$.

Merge-Sort

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

MERGE-SORT (A, p, r)
    se $p < r$
        então $q \leftarrow \lfloor (p+r)/2 \rfloor$
            MERGE-SORT (A, p, q)
            MERGE-SORT $(A, q+1, r)$
            INTERCALA (A, p, q, r)

Consumo de tempo?

Merge-Sort

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

MERGE-SORT (A, p, r)
1 **se** $p < r$
2 **então** $q \leftarrow \lfloor (p + r)/2 \rfloor$
3 MERGE-SORT (A, p, q)
4 MERGE-SORT $(A, q + 1, r)$
5 INTERCALA (A, p, q, r)

Consumo de tempo?

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

Merge-Sort

```
MERGE-SORT (A, p, r)
1   se p < r
2       então q ← ⌊(p + r)/2⌋
3           MERGE-SORT (A, p, q)
4           MERGE-SORT (A, q + 1, r)
5           INTERCALA (A, p, q, r)
```

linha	consumo na linha
1	?
2	?
3	?
4	?
5	?

$T(n) = ?$

Merge-Sort

```
MERGE-SORT (A, p, r)
1   se p < r
2       então q ← ⌊(p + r)/2⌋
3           MERGE-SORT (A, p, q)
4           MERGE-SORT (A, q + 1, r)
5           INTERCALA (A, p, q, r)
```

linha	consumo na linha
1	$\Theta(1)$
2	$\Theta(1)$
3	$T(\lceil n/2 \rceil)$
4	$T(\lfloor n/2 \rfloor)$
5	$\Theta(n)$

$$T(n) \quad = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n+2)$$

Merge-Sort

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

$$T(1) = \Theta(1)$$
$$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n) \quad \text{para } n = 2, 3, 4, \ldots$$

Merge-Sort

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

$$T(1) = \Theta(1)$$
$$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n) \quad \text{para } n = 2, 3, 4, \ldots$$

Solução: $T(n)$ é $\Theta(???)$.

Demonstração: ...

Merge-Sort

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

$$T(1) = \Theta(1)$$
$$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n) \quad \text{para } n = 2, 3, 4, \ldots$$

Solução: $T(n)$ é $\Theta(???)$.

Demonstração: ...

Veremos, mas antes estudaremos Recorrências.

Exercícios

Exercício 7.A

Problema: verificar se v é elemento de $A[p \mathinner{.\,.} r]$.
(Para quais valores de p e r faz sentido?)
Escreva um algoritmo recursivo que resolve o problema. O algoritmo deve devolver i tal que $A[i] = v$.

Exercício 7.B

Problema: verificar se v é elemento de vetor crescente $A[p \mathinner{.\,.} r]$. Escreva algoritmo recursivo de busca "linear" e outro de busca "binária".

Exercício 7.C

Escreva versão recursiva da ordenação por inserção. O algoritmo deve rearranjar em ordem crescente qualquer vetor dado $A[p \mathinner{.\,.} r]$.

Mais exercícios

Exercício 7.D (Versão sofisticada de busca)
Problema: verificar se v é elemento de vetor crescente $A[p\,..\,r]$. Escreva um algoritmo que devolva j tal que

$$A[j] \leq v < A[j+1]\,.$$

Quais os possíveis valores de j? Escreva duas versões: uma "linear" e uma "binária". Prove que os seus algoritmos estão corretos.

Exercício 7.E
Escreva uma versão recursiva do algoritmo de ordenação por seleção.

Exercício 7.F
Escreva uma versão iterativa do MERGE-SORT.

Recorrências

CLRS 4.1–4.2

AU 3.9, 3.11

Recorrências

Recorrência =

= “fórmula” que define uma função
em termos d’ela mesma

= algoritmo recursivo que calcula uma função

Exemplo 1

$$
\begin{aligned}
T(1) &= 1 \\
T(n) &= T(n-1) + 3n + 2 \quad \text{para } n = 2, 3, 4, \ldots
\end{aligned}
$$

Define função T sobre inteiros positivos:

n	1	2	3	4	5	6
$T(n)$	1	9	20	34	51	71

$T(n)$
1 **se** $n = 1$
2 **então devolva** 1
3 **senão devolva** $T(n-1) + 3n + 2$

Resolver uma recorrência

Resolver uma recorrência =

= obter uma “fórmula fechada” para $T(n)$

Método da substituição:

“chute” fórmula e verifique por indução

Exemplo 1 (continuação)

Eu acho que $T(n) = \frac{3}{2}n^2 + \frac{7}{2}n - 4$.

Exemplo 1 (continuação)

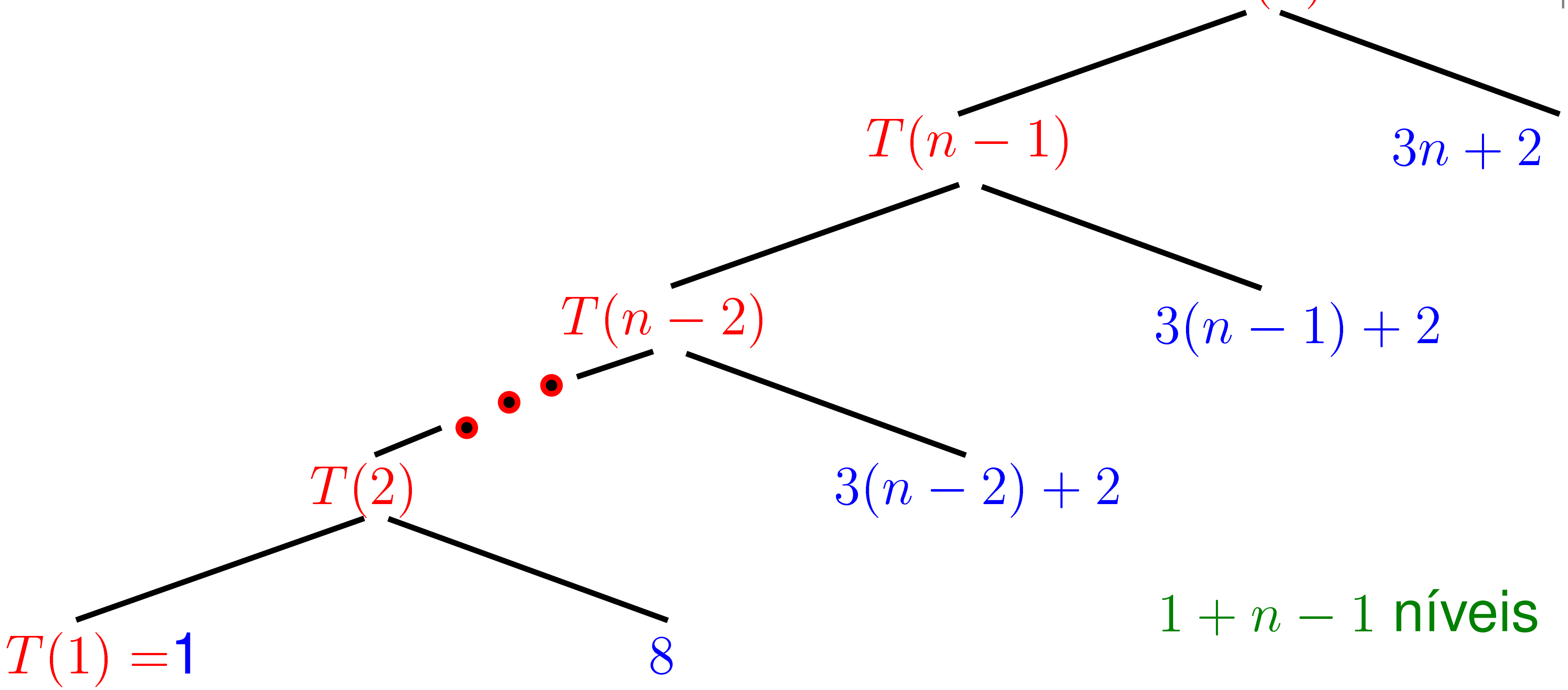
Eu acho que $T(n) = \frac{3}{2}n^2 + \frac{7}{2}n - 4$.
Verificação:
Se $n = 1$ então $T(n) = 1 = \frac{3}{2} + \frac{7}{2} - 4$.
Tome $n \geq 2$ e suponha que a fórmula está certa para $n - 1$:

$$\begin{aligned}
T(n) &= T(n-1) + 3n + 2 \\
&\overset{\text{hi}}{=} \tfrac{3}{2}(n-1)^2 + \tfrac{7}{2}(n-1) - 4 + 3n + 2 \\
&= \tfrac{3}{2}n^2 - 3n + \tfrac{3}{2} + \tfrac{7}{2}n - \tfrac{7}{2} - 4 + 3n + 2 \\
&= \tfrac{3}{2}n^2 + \tfrac{7}{2}n - 4\,.
\end{aligned}$$

Bingo!

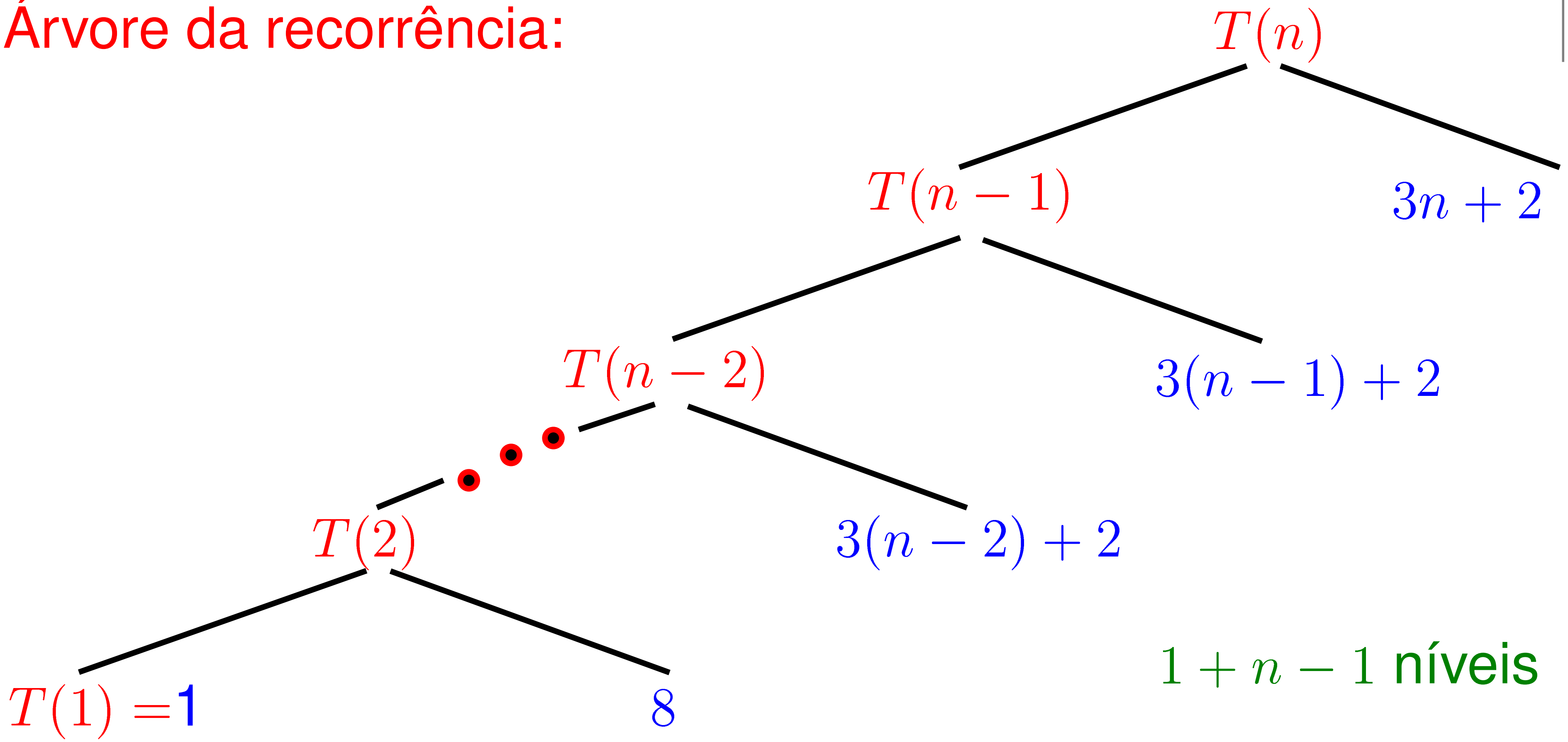
Como adivinhei fórmula fechada?

Árvore da recorrência:



Como adivinhei fórmula fechada?

Árvore da recorrência:



$$T(n) = (3n+2) + (3n-1) + \cdots + 8 + 1$$

$$= \frac{3}{2}n^2 + \frac{7}{2}n - 4$$

Exemplo 2

$$
\begin{aligned}
T(1) &= 1 \\
T(n) &= 2T(n/2) + 7n + 2 \quad \text{para } n = 2, 3, 4, 5, \ldots
\end{aligned}
$$

Exemplo 2

$$\begin{aligned} T(1) &= 1 \\ T(n) &= 2T(n/2) + 7n + 2 \quad \text{para } n = 2, 3, 4, 5, \ldots \end{aligned}$$

Não é uma recorrência! Não faz sentido!
$T(3)$ depende de $T(3/2)$...

Exemplo 3

$$
\begin{aligned}
G(1) &= 1 \\
G(n) &= 2G(n/2) + 7n + 2 \quad \text{para } n = 2, 4, 8, 16 \ldots, 2^i, \ldots
\end{aligned}
$$

n	1	2	4	8	16
$G(n)$	1	18	66	190	494

Exemplo 3

$$
\begin{aligned}
G(1) &= 1 \\
G(n) &= 2G(n/2) + 7n + 2 \quad \text{para } n = 2, 4, 8, 16 \ldots, 2^i, \ldots
\end{aligned}
$$

n	1	2	4	8	16
$G(n)$	1	18	66	190	494

Fórmula fechada: $G(n) =$???

Hmmmmmm

Acho que $G(n)$ é da forma $n \lg n$ . . .

n	$G(n)$	$6n \lg n$	$7n \lg n$	$8n \lg n$	n^2
1	1	0	0	0	1
2	18	12	14	16	4
4	66	48	56	64	16
8	190	144	168	192	64
16	494	384	448	512	256
32	1214	960	1120	1280	1024
64	2878	2304	2688	3072	4096
128	6654	5376	6272	7168	16384
256	15102	12288	14336	16384	65536

Chute

Acho que a fórmula fechada é

$$G(n) = 7n \lg n + 3n - 2$$

para $n = 1, 2, 4, 8, 16, 32 \ldots$

Lá vamos nós outra vez. . .

Chute

$G(n) = 7n \lg n + 3n - 2$ para $n = 1, 2, 4, 8, 16, 32 \ldots$

Chute

$G(n) = 7n\lg n + 3n - 2$ para $n = 1, 2, 4, 8, 16, 32\ldots$

Prova: Se $n = 1$ então $G(n) = 1 = 7 \cdot 1 \lg 1 + 3 \cdot 1 - 2$.
Se $n \geq 2$ então

$$
\begin{aligned}
G(n) &= 2G(\tfrac{n}{2}) + 7n + 2 \\
&\overset{\text{hi}}{=} 2\left(7\tfrac{n}{2}\lg\tfrac{n}{2} + 3\,\tfrac{n}{2} - 2\right) + 7n + 2 \\
&= 7n(\lg n - 1) + 3n - 4 + 7n + 2 \\
&= 7n\lg n - 7n + 3n - 2 + 7n \\
&= 7n\lg n + 3n - 2
\end{aligned}
$$

Iiiiiééééssss!

Como adivinhei fórmula fechada?

Árvore da recorrência: $G(n)$

Como adivinhei fórmula fechada?

Árvore da recorrência:

$$7n + 2$$

$$G(\tfrac{n}{2}) \qquad\qquad G(\tfrac{n}{2})$$

Como adivinhei fórmula fechada?

Árvore da recorrência:

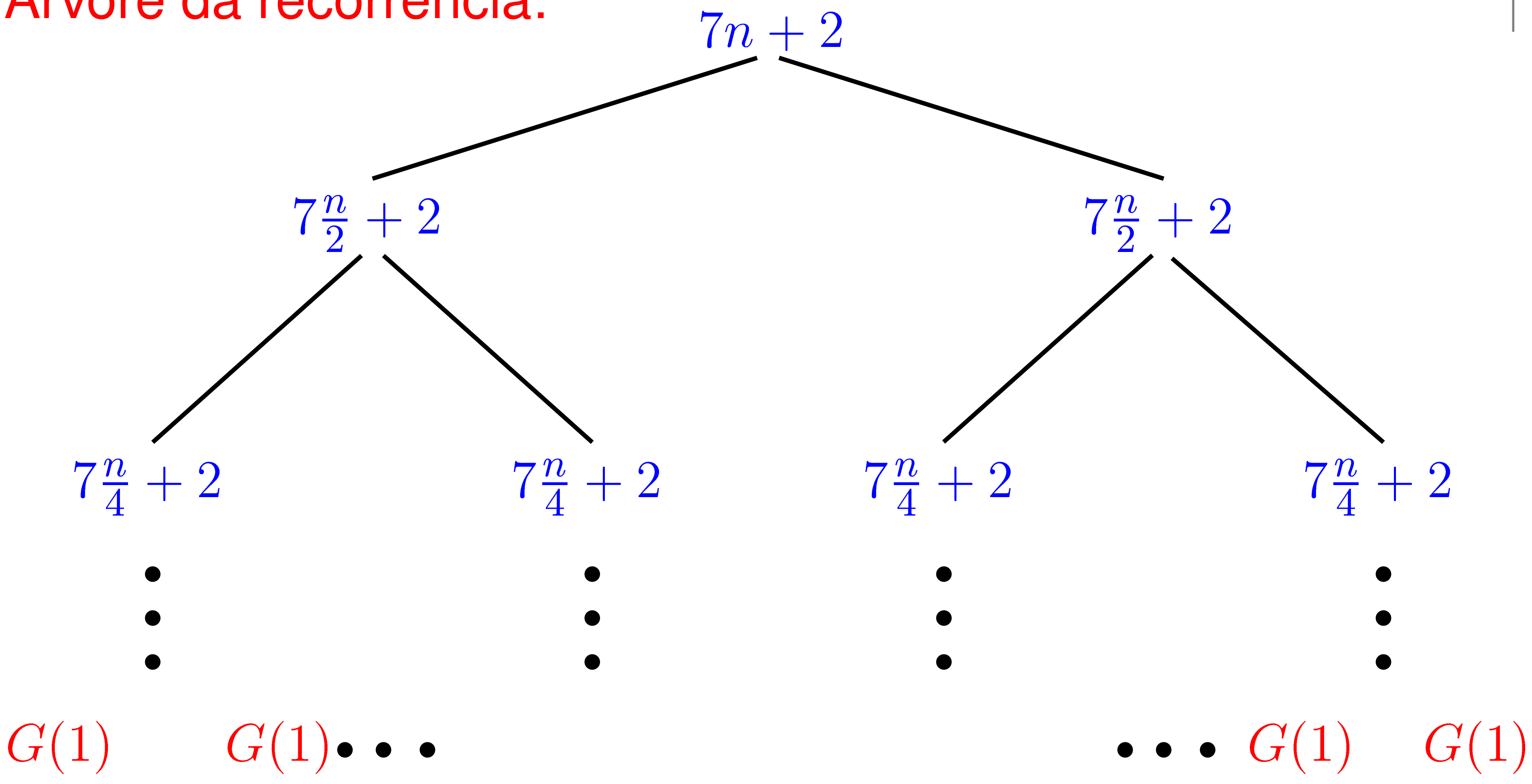
$$7n+2$$

$$7\frac{n}{2}+2 \qquad\qquad 7\frac{n}{2}+2$$

$$G(\tfrac{n}{4}) \qquad G(\tfrac{n}{4}) \qquad G(\tfrac{n}{4}) \qquad G(\tfrac{n}{4})$$

Como adivinhei fórmula fechada?

Árvore da recorrência:



total de $1 + \lg n$ níveis

Contas

nível	0	1	2	$\ldots$	$k-1$	k
soma	$7n+2$	$7n+4$	$7n+8$	$\ldots$	$7n+2^k$	$2^k G(1)$

$n = 2^k \qquad k = \lg n$

Contas

nível	0	1	2	$\ldots$	$k-1$	k
soma	$7n+2$	$7n+4$	$7n+8$	$\ldots$	$7n+2^k$	$2^k G(1)$

$n = 2^k \qquad k = \lg n$

$$
\begin{aligned}
G(n) &= 7n + 2^1 + 7n + 2^2 + \cdots + 7n + 2^k + 2^k G(1) \\
&= 7n\,k + (2 + 4 + \cdots + 2^k) + 2^k \\
&= 7n\,k + 2 \cdot 2^k - 2 + n \\
&= 7n \lg n + 2n - 2 + n \quad (k = \lg n) \\
&= 7n \lg n + 3n - 2 && \text{(-1)}
\end{aligned}
$$

liiiééééssss

Série geométrica

Para $x \neq 1$, quanto vale $1 + x^1 + x^2 + \cdots + x^{k-1} + x^k$?

(CLRS (A.5), p.1060)

Solução: Seja $S_k := 1 + x^1 + x^2 + \cdots + x^{k-1} + x^k$.
Temos que $xS_k = x^1 + x^2 + x^3 + \cdots + x^k + x^{k+1}$.
Logo,

$$xS_k - S_k = x^{k+1} - 1$$

e

$$S_k = \frac{x^{k+1} - 1}{x - 1}.$$

Conclusão:

$$\boxed{1 + x^1 + x^2 + \cdots + x^{k-1} + x^k = \frac{x^{k+1}-1}{x-1}.}$$

Exemplo 3 (continuação)

É mais fácil mostrar que $G(n) = \mathrm{O}(n \lg n)$.
Vou provar que $G(n) \leq 9n \lg n$ para $n = 2, 4, 8, 16, \ldots, 2^i, \ldots$

Exemplo 3 (continuação)

É mais fácil mostrar que $G(n) = \mathrm{O}(n \lg n)$.
Vou provar que $G(n) \leq 9n \lg n$ para $n = 2, 4, 8, 16, \ldots, 2^i, \ldots$

Prova: Se $n = 2$, $G(n) = 18 = 9 \cdot 2 \cdot \lg 2$.
Se $n \geq 4$,

$$\begin{aligned} G(n) &= 2G(n/2) + 7n + 2 \\ &\overset{\text{hi}}{\leq} 2 \cdot 9(n/2) \lg(n/2) + 7n + 2 \\ &= 9n\,(\lg n - 1) + 7n + 2 \\ &= 9n \lg n - 2n + 2 \\ &< 9n \lg n \quad \text{(pois } n > 1\text{)} \end{aligned}$$

Da linha 1 para a linha 2, a hipótese de indução vale pois $2 \leq n/2 < n$.

Exercícios

Exercício 8.A

Seja T a função definida pela recorrência

$$\begin{array}{rcll} T(1) & = & 1 & \\ T(n) & = & T(n-1) + 2n - 2 & \text{para } n = 2, 3, 4, 5, \ldots \end{array}$$

Verifique que a recorrência é honesta, ou seja, de fato define uma função. A partir da árvore da recorrência, adivinhe uma boa delimitação assintótica para $T(n)$; dê a resposta em notação O. Prove a delimitação pelo método da substituição.

Exercício 8.B

Resolva a recorrência

$$\begin{array}{rcll} T(1) & = & 1 & \\ T(n) & = & T(n-2) + 2n + 1 & \text{para } n = 2, 3, 4, 5, \ldots \end{array}$$

Desenhe a árvore da recorrência. Dê a resposta em notação O.

Mais exercícios

Exercício 8.C

Resolva a recorrência

$$\begin{aligned} T(1) &= 1 \\ T(2) &= 2 \\ T(n) &= T(n-2) + 2n + 1 \quad \text{para } n = 3, 4, 5, 6, \ldots \end{aligned}$$

Exercício 8.D

Resolva a recorrência

$$\begin{aligned} T(1) &= 1 \\ T(n) &= T(n/2) + 1 \quad \text{para } n = 2, 3, 4, 5, \ldots \end{aligned}$$

Exercício 8.E

Resolva a recorrência

$$\begin{aligned} T(1) &= 1 \\ T(n) &= T(\lfloor n/2 \rfloor) + 1 \quad \text{para } n = 2, 3, 4, 5, \ldots \end{aligned}$$

Mais exercícios ainda

Exercício 8.F
Resolva a recorrência

$$\begin{aligned} T(1) &= 1 \\ T(n) &= T(\lfloor n/2 \rfloor) + n \quad \text{para } n = 2, 3, 4, 5, \ldots \end{aligned}$$

Exercício 8.G
Resolva a recorrência

$$\begin{aligned} T(1) &= 1 \\ T(n) &= 2T(\lfloor n/2 \rfloor) + n \quad \text{para } n = 2, 3, 4, 5, \ldots \end{aligned}$$

Exercício 8.H
Resolva a recorrência

$$\begin{aligned} T(1) &= 1 \\ T(n) &= 2T(\lceil n/2 \rceil) + n \quad \text{para } n = 2, 3, 4, 5, \ldots \end{aligned}$$

Melhores momentos

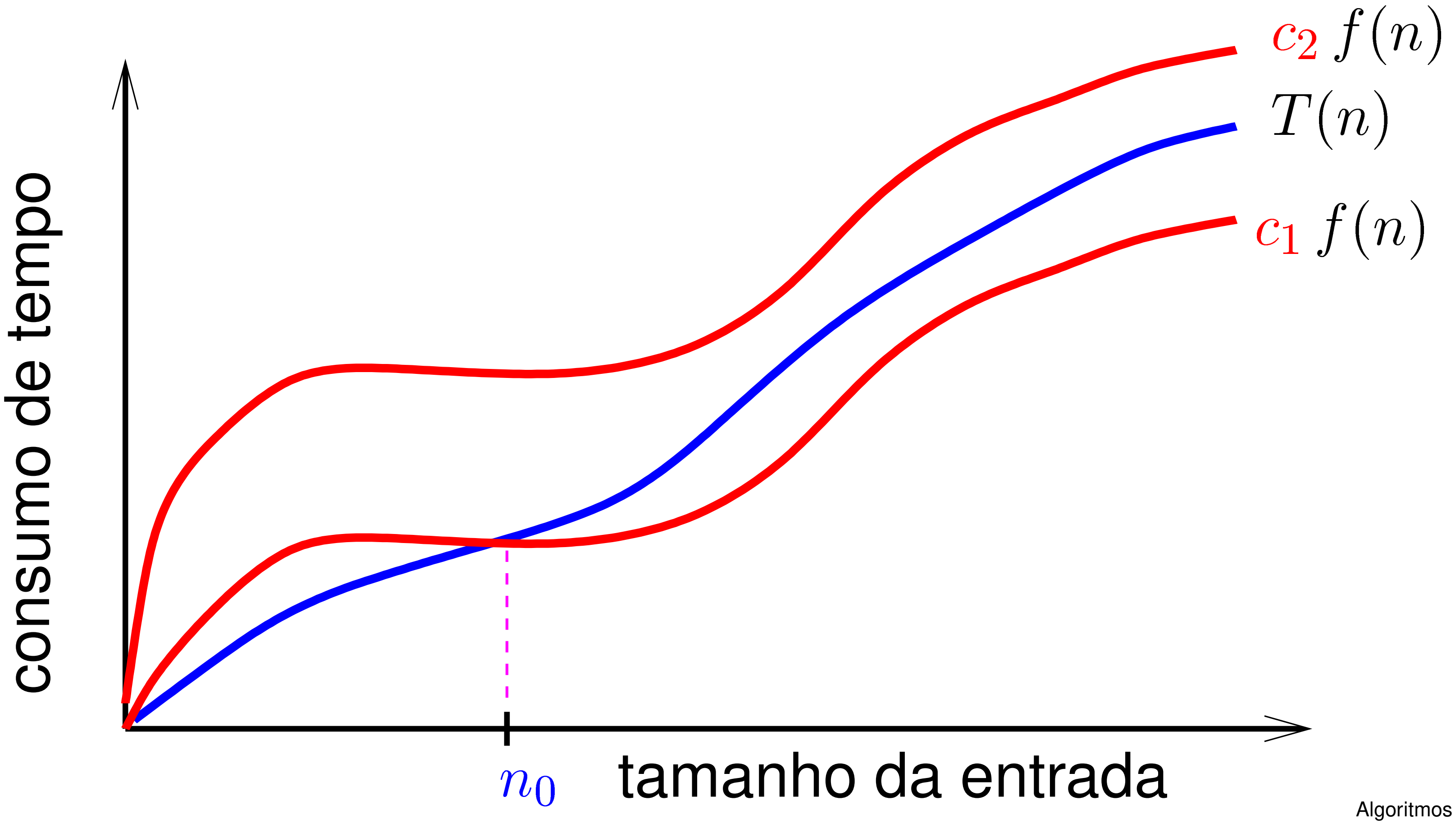
AULA 4

Definição

Dizemos que $T(n)$ é $\Theta(f(n))$ se se existem constantes positivas c_1, c_2 e n_0 tais que

$$c_1\, f(n) \;\leq\; T(n) \;\leq\; c_2\, f(n)$$

para todo $n \geq n_0$.



Análise da intercalação

Problema: Dados $A[p \mathinner{.\,.} q]$ e $A[q{+}1 \mathinner{.\,.} r]$ crescentes, rearranjar $A[p \mathinner{.\,.} r]$ de modo que ele fique em ordem crescente.

Entra:

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

Sai:

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

Intercalação

INTERCALA (A, p, q, r)

$\triangleright$ $B[p\,..\,r]$ é um vetor auxiliar
para $i \leftarrow p$ **até** q **faça**
    $B[i] \leftarrow A[i]$
para $j \leftarrow q+1$ **até** r **faça**
    $B[r+q+1-j] \leftarrow A[j]$
$i \leftarrow p$
$j \leftarrow r$
para $k \leftarrow p$ **até** r **faça**
    se $B[i] \leq B[j]$
        então $A[k] \leftarrow B[i]$
                $i \leftarrow i+1$
        senão $A[k] \leftarrow B[j]$
                $j \leftarrow j-1$

Consumo de tempo

Se a execução de cada linha de código consome 1 unidade de tempo o consumo total é ($n := r - p + 1$):

linha	todas as execuções da linha
1	$= \quad q - p + 2 \; = \; n - r + q + 1$
2	$= \quad q - p + 1 \; = \; n - r + q$
3	$= \quad r - (q + 1) + 2 \; = \; n - q + p$
4	$= \quad r - (q + 1) + 1 \; = \; n - q + p - 1$
5	$= \quad 1$
6	$= \quad 1$
7	$= \quad r - p + 2 \; = \; n + 1$
8	$= \quad r - p + 1 \; = \; n$
9–12	$= \quad 2\,(r - p + 1) \; = \; 2n$
total	$= \quad 8n - 2(r - p + 1) + 5 = \; 6n + 5$

Consumo de tempo

Quanto tempo consome em função de $n := r - p + 1$?

linha	consumo de todas as execuções da linha
1–4	$\Theta(n)$
5–6	$\Theta(1)$
7	$n\Theta(1) = \Theta(n)$
8	$n\Theta(1) = \Theta(n)$
9–12	$n\Theta(1) = \Theta(n)$

total $\Theta(4n + 1) = \Theta(n)$

Conclusão:

O algoritmo consome $\Theta(n)$ unidades de tempo.

Merge-Sort

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

MERGE-SORT (A, p, r)
  se $p < r$
    então $q \leftarrow \lfloor (p + r)/2 \rfloor$
      MERGE-SORT (A, p, q)
      MERGE-SORT $(A, q + 1, r)$
      INTERCALA (A, p, q, r)

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

Merge-Sort

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

MERGE-SORT (A, p, r)
1   **se** $p < r$
2   **então** $q \leftarrow \lfloor (p+r)/2 \rfloor$
3   **MERGE-SORT** (A, p, q)

---

4   MERGE-SORT $(A, q+1, r)$
5   INTERCALA (A, p, q, r)

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

Merge-Sort

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

MERGE-SORT (A, p, r)
1 **se** $p < r$
2 **então** $q \leftarrow \lfloor (p+r)/2 \rfloor$
3 MERGE-SORT (A, p, q)
4 **MERGE-SORT** $(A, q+1, r)$

---

5 INTERCALA (A, p, q, r)

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

Merge-Sort

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

MERGE-SORT (A, p, r)
se $p < r$
  então $q \leftarrow \lfloor (p + r)/2 \rfloor$
    MERGE-SORT (A, p, q)
    MERGE-SORT $(A, q + 1, r)$
    INTERCALA (A, p, q, r)

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

Merge-Sort

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

MERGE-SORT (A, p, r)
    se $p < r$
        então $q \leftarrow \lfloor (p + r)/2 \rfloor$
                MERGE-SORT (A, p, q)
                MERGE-SORT $(A, q + 1, r)$
                INTERCALA (A, p, q, r)

Consumo de tempo?

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

Merge-Sort

```
MERGE-SORT (A, p, r)
1   se p < r
2       então q ← ⌊(p + r)/2⌋
3             MERGE-SORT (A, p, q)
4             MERGE-SORT (A, q + 1, r)
5             INTERCALA (A, p, q, r)
```

linha	consumo na linha
1	$\Theta(1)$
2	$\Theta(1)$
3	$T(\lceil n/2 \rceil)$
4	$T(\lfloor n/2 \rfloor)$
5	$\Theta(n)$

$$T(n) \quad = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n+2)$$

Merge-Sort

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

$$T(1) = \Theta(1)$$
$$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n) \quad \text{para } n = 2, 3, 4, \ldots$$

Solução: $T(n)$ é $\Theta(???)$.

Demonstração: ...

Veremos, mas antes estudaremos Recorrências.

Recorrências

$$
\begin{aligned}
T(1) &= 1 \\
T(n) &= T(n-1) + 3n + 2 \quad \text{para } n = 2, 3, 4, \ldots
\end{aligned}
$$

Define função T sobre inteiros positivos:

n	1	2	3	4	5	6
$T(n)$	1	9	20	34	51	71

$T(n)$
1 **se** $n = 1$
2 **então devolva** 1
3 **senão devolva** $T(n-1) + 3n + 2$

Método da substituição

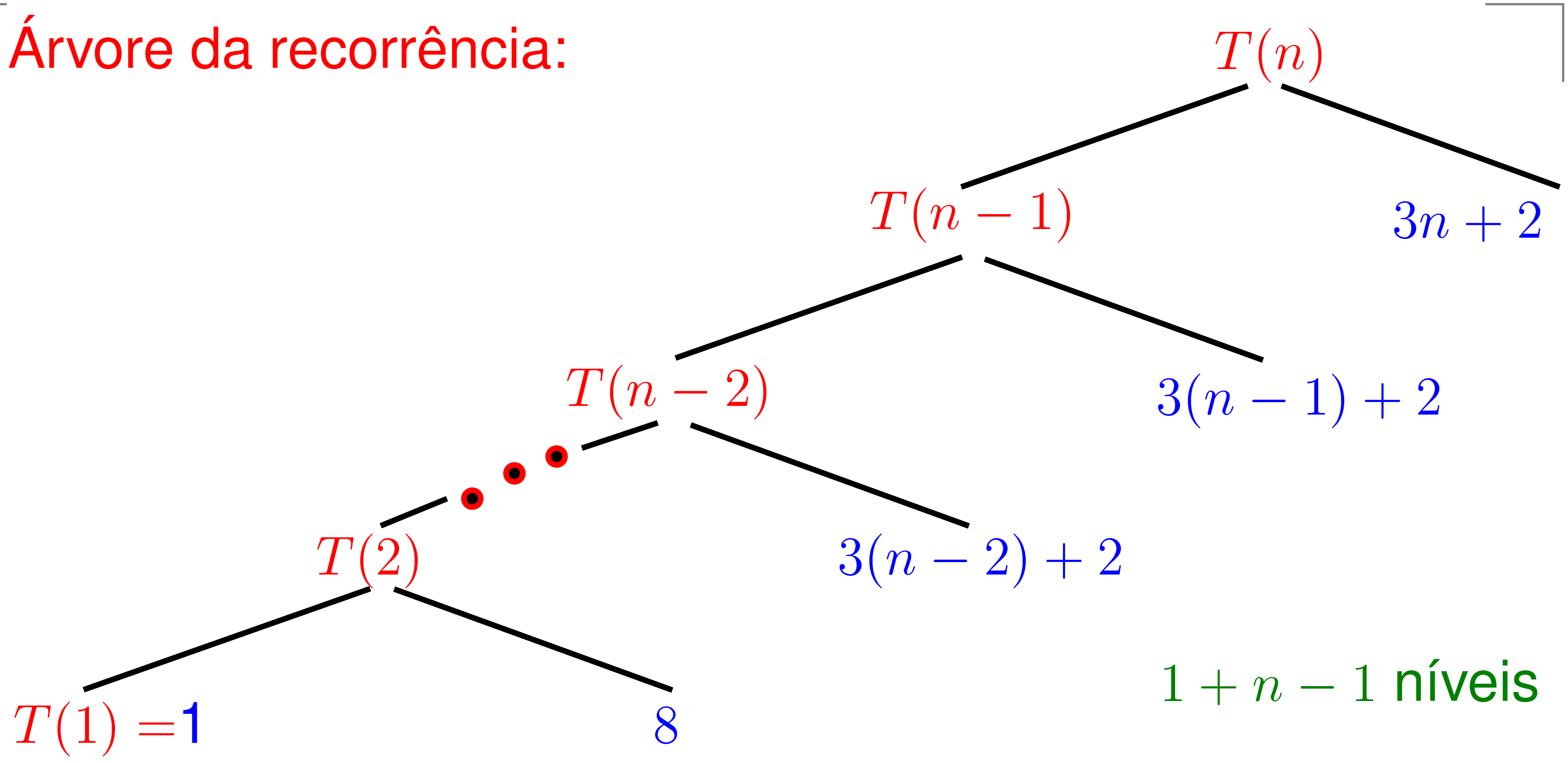
Chute:
Eu acho que $T(n) = \frac{3}{2}n^2 + \frac{7}{2}n - 4$.

Verificação por indução:
Se $n = 1$ então $T(n) = 1 = \frac{3}{2} + \frac{7}{2} - 4$.
Tome $n \geq 2$ e suponha que a fórmula está certa para $n - 1$:

$$\begin{aligned} T(n) &= T(n-1) + 3n + 2 \\ &\overset{\text{hi}}{=} \tfrac{3}{2}(n-1)^2 + \tfrac{7}{2}(n-1) - 4 + 3n + 2 \\ &= \tfrac{3}{2}n^2 - 3n + \tfrac{3}{2} + \tfrac{7}{2}n - \tfrac{7}{2} - 4 + 3n + 2 \\ &= \tfrac{3}{2}n^2 + \tfrac{7}{2}n - 4\,. \end{aligned}$$

Bingo!

Como adivinhei fórmula fechada?



$$T(n) = (3n+2) + (3n-1) + \cdots + 8 + 1$$

$$= \frac{3}{2}n^2 + \frac{7}{2}n - 4$$

Exemplo 3

$$
\begin{aligned}
G(1) &= 1 \\
G(n) &= 2G(n/2) + 7n + 2 \quad \text{para } n = 2, 4, 8, 16 \ldots, 2^i, \ldots
\end{aligned}
$$

n	1	2	4	8	16
$G(n)$	1	18	66	190	494

Fórmula fechada: $G(n) =$???

Resolvendo a recorrência

$G(n) = 7n \lg n + 3n - 2$ para $n = 1, 2, 4, 8, 16, 32 \ldots$

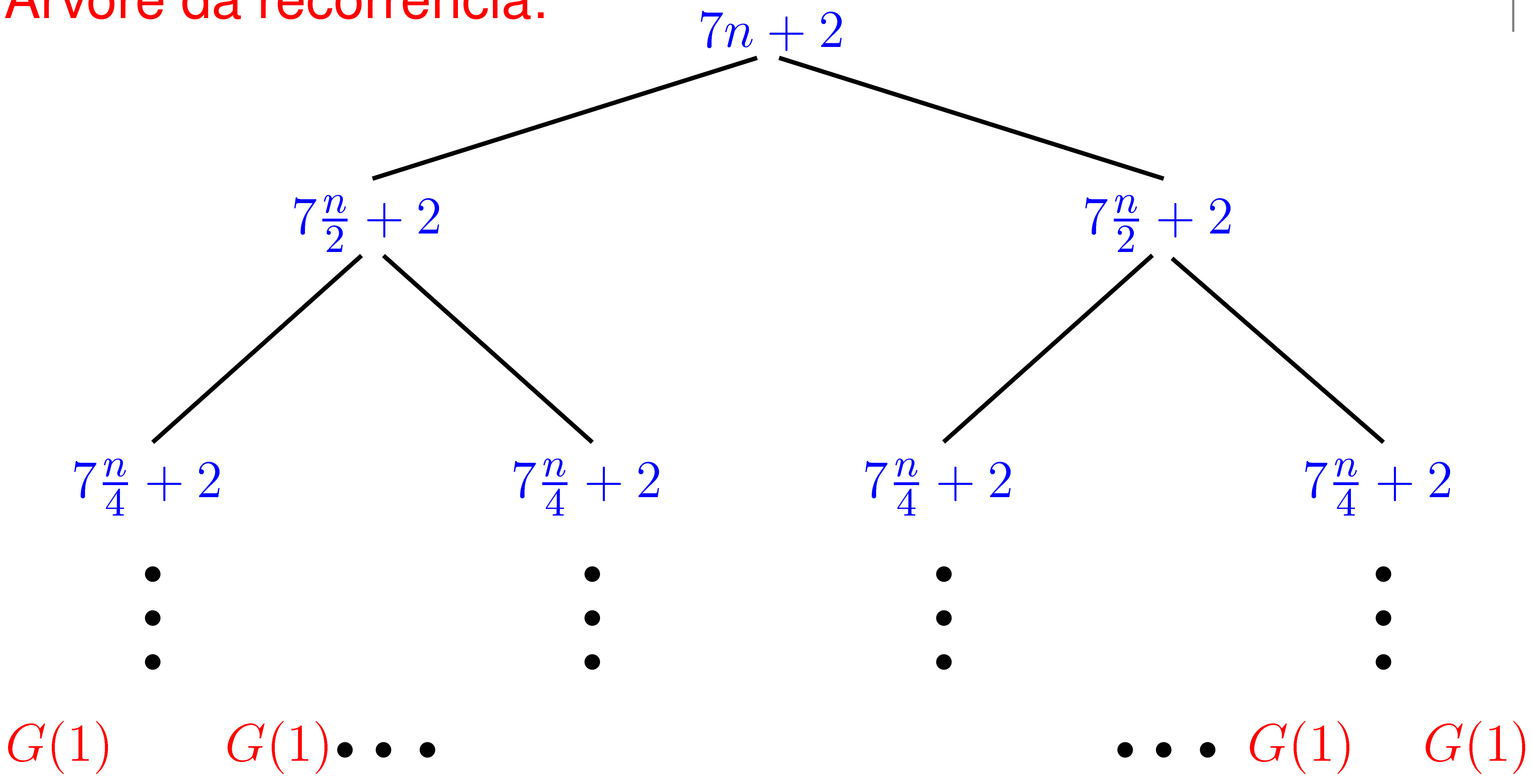
Prova: Se $n = 1$ então $G(n) = 1 = 7 \cdot 1 \lg 1 + 3 \cdot 1 - 2$.
Se $n \geq 2$ então

$$
\begin{aligned}
G(n) &= 2G(\tfrac{n}{2}) + 7n + 2 \\
&\stackrel{\text{hi}}{=} 2\left(7\tfrac{n}{2} \lg \tfrac{n}{2} + 3\,\tfrac{n}{2} - 2\right) + 7n + 2 \\
&= 7n(\lg n - 1) + 3n - 4 + 7n + 2 \\
&= 7n \lg n - 7n + 3n - 2 + 7n \\
&= 7n \lg n + 3n - 2
\end{aligned}
$$

Iiiiiéééésss!

Como adivinhei fórmula fechada?

Árvore da recorrência:



total de $1 + \lg n$ níveis

Contas

nível	0	1	2	$\ldots$	$k-1$	k
soma	$7n+2$	$7n+4$	$7n+8$	$\ldots$	$7n+2^k$	$2^k G(1)$

$n = 2^k \qquad k = \lg n$

$$\begin{aligned} G(n) &= 7n + 2^1 + 7n + 2^2 + \cdots + 7n + 2^k + 2^k G(1) \\ &= 7n\,k + (2 + 4 + \cdots + 2^k) + 2^k \\ &= 7n\,k + 2 \cdot 2^k - 2 + n \\ &= 7n \lg n + 2n - 2 + n \quad (k = \lg n) \\ &= 7n \lg n + 3n - 2 \end{aligned}$$ (-3)

liiiééééssss

Exemplo 3 (continuação)

É mais fácil mostrar que $G(n) = \mathrm{O}(n \lg n)$.
Vou provar que $G(n) \leq 9n \lg n$ para $n = 2, 4, 8, 16, \ldots, 2^i, \ldots$

Prova: Se $n = 2$, $G(n) = 18 = 9 \cdot 2 \cdot \lg 2$.
Se $n \geq 4$,

$$
\begin{aligned}
G(n) &= 2G(n/2) + 7n + 2 \\
&\overset{\text{hi}}{\leq} 2 \cdot 9(n/2) \lg(n/2) + 7n + 2 \\
&= 9n\,(\lg n - 1) + 7n + 2 \\
&= 9n \lg n - 2n + 2 \\
&< 9n \lg n \quad \text{(pois } n \geq 2\text{)}
\end{aligned}
$$

Da linha 1 para a linha 2, a hipótese de indução vale pois $2 \leq n/2 < n$.

Exemplo 3 (novamente)

É mais fácil mostrar que $G(n) = \mathrm{O}(n \lg n)$.
Vou provar que $G(n) \leq 8n \lg n$ para $n = 8, 16, \ldots, 2^i, \ldots$

Prova: Se $n = 8$, $G(n) = 190 < 192 = 8 \cdot 8 \cdot \lg 8 = 64 \cdot 3$.
Se $n \geq 16$,

$$
\begin{aligned}
G(n) &= 2G(n/2) + 7n + 2 \\
&\overset{\text{hi}}{\leq} 2 \cdot 8(n/2)\lg(n/2) + 7n + 2 \\
&= 8n\,(\lg n - 1) + 7n + 2 \\
&= 8n \lg n - n + 2 \\
&< 9n \lg n \quad \text{(pois } n \geq 16\text{)}
\end{aligned}
$$

Da linha 1 para a linha 2, a hipótese de indução vale pois $2 \leq n/2 < n$.

AULA 5

Recorrências (continuação)

CLRS 4.1–4.2

AU 3.9, 3.11

Exemplo 4

$$
\begin{aligned}
T(1) &= 1 \\
T(n) &= T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + 6n + 5 \quad \text{para } n = 2, 3, 4, 5, \ldots
\end{aligned}
$$

n	1	2	3	4	5	6	7	8	9	10
$T(n)$	1	19	43	67	97	127	157	187	223	259

Exemplo 4

$$\begin{aligned} T(1) &= 1 \\ T(n) &= T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + 6n + 5 \quad \text{para } n = 2, 3, 4, 5, \ldots \end{aligned}$$

n	1	2	3	4	5	6	7	8	9	10
$T(n)$	1	19	43	67	97	127	157	187	223	259

$T(n) = \mathrm{O}(\text{???})$

Exemplo 4

Vou mostrar que $T(n) \leq 20\,n \lg n$ para $n = 2, 3, 4, \ldots$

n	1	2	3	4	5	6	7	8	9	10
$T(n)$	1	19	43	67	97	127	157	187	223	259
$20\,n\lfloor \lg n \rfloor$	0	40	60	160	200	240	280	480	540	600

Exemplo 4

Vou mostrar que $T(n) \leq 20\, n \lg n$ para $n = 2, 3, 4, \ldots$

n	1	2	3	4	5	6	7	8	9	10
$T(n)$	1	19	43	67	97	127	157	187	223	259
$20\, n \lfloor \lg n \rfloor$	0	40	60	160	200	240	280	480	540	600

Prova:
Se $n = 2, 3$, então $T(n) \leq 20\, n \lfloor \lg n \rfloor \leq 20\, n \lg n$, como mostra a tabela.

Note que a base da indução é $n = 2, 3$. Por quê???

Exemplo 4

Prova: (continuação) Se $n \geq 4$

$$
\begin{aligned}
T(n) &= T(\lceil n/2\rceil) + T(\lfloor n/2\rfloor) + 6n + 5 \\
&\overset{\text{hi}}{\leq} 20 \left\lceil \frac{n}{2} \right\rceil \lg \left\lceil \frac{n}{2} \right\rceil + 20 \left\lfloor \frac{n}{2} \right\rfloor \lg \left\lfloor \frac{n}{2} \right\rfloor + 6n + 5 \\
&\leq 20 \left\lceil \frac{n}{2} \right\rceil \lg n + 20 \left\lfloor \frac{n}{2} \right\rfloor \lg \frac{n}{2} + 6n + 5 \\
&= 20 \left\lceil \frac{n}{2} \right\rceil \lg n + 20 \left\lfloor \frac{n}{2} \right\rfloor (\lg n - 1) + 6n + 5 \\
&= 20 \left(\left\lceil \frac{n}{2} \right\rceil + \left\lfloor \frac{n}{2} \right\rfloor \right) \lg n - 20 \left\lfloor \frac{n}{2} \right\rfloor + 6n + 5 \\
&= 20\, n \lg n - 20 \left\lfloor \frac{n}{2} \right\rfloor + 6n + 5 \\
&\leq 20\, n \lg n \quad \text{(pois } n \geq 4\text{)}
\end{aligned}
$$

Iiiiééééssss!

Como achei as contantes?

$$
\begin{aligned}
T(1) &= 1 \\
T(n) &= T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + 6n + 5 \quad \text{para } n = 2, 3, 4, \ldots
\end{aligned}
$$

Supeito que $T(n) = \mathrm{O}(n \lg n)$.

Como achei as contantes?

$$
\begin{aligned}
T(1) &= 1 \\
T(n) &= T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + 6n + 5 \quad \text{para } n = 2, 3, 4, \ldots
\end{aligned}
$$

Supeito que $T(n) = \mathrm{O}(n \lg n)$.

Vamos tentar mostrar isto!

Precisamos encontrar um número real positivo c e um número inteiro positivo n_0 tais que

$$T(n) \leq c\, n \lg n$$

para todo $n \geq n_0$.

Rascunho

Suponha que existam as tais constantes c e n_0.
Vamos descobrir a "cara" delas.

$$
\begin{aligned}
T(n) &= T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + 6n + 5 \\
&\overset{\text{hi}}{\leq} c\left\lceil \frac{n}{2} \right\rceil \lg \left\lceil \frac{n}{2} \right\rceil + c\left\lfloor \frac{n}{2} \right\rfloor \lg \left\lfloor \frac{n}{2} \right\rfloor + 6n + 5 \\
&\leq c\left\lceil \frac{n}{2} \right\rceil \lg n + c\left\lfloor \frac{n}{2} \right\rfloor \lg n + 6n + 5 \\
&= c\left(\left\lceil \frac{n}{2} \right\rceil + \left\lfloor \frac{n}{2} \right\rfloor\right) \lg n + 6n + 5 \\
&= c\, n \lg n + 6n + 5 \ldots \quad \text{Hmmm, não deu...}
\end{aligned}
$$

Nova tentativa

Suponha que existam c e n_0.

$$
\begin{aligned}
T(n) &= T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + 6n + 5 \\
&\overset{\text{hi}}{\leq} c \left\lceil \frac{n}{2} \right\rceil \lg \left\lceil \frac{n}{2} \right\rceil + c \left\lfloor \frac{n}{2} \right\rfloor \lg \left\lfloor \frac{n}{2} \right\rfloor + 6n + 5 \\
&\leq c \left\lceil \frac{n}{2} \right\rceil \lg n + c \left\lfloor \frac{n}{2} \right\rfloor \lg \frac{n}{2} + 6n + 5 \\
&= c \left\lceil \frac{n}{2} \right\rceil \lg n + c \left\lfloor \frac{n}{2} \right\rfloor (\lg n - 1) + 6n + 5 \\
&= c \left(\left\lceil \frac{n}{2} \right\rceil + \left\lfloor \frac{n}{2} \right\rfloor \right) \lg n - c \left\lfloor \frac{n}{2} \right\rfloor + 6n + 5 \\
&= c\, n \lg n - c \left\lfloor \frac{n}{2} \right\rfloor + 6n + 5 \quad \text{Agora vai!}
\end{aligned}
$$

Agora vai

Queremos saber "quando"

$$c\left\lfloor\frac{n}{2}\right\rfloor \geq 6n+5.$$

Bem, para $c = 20$ e todo $n \geq n_0 = 4$ temos que

$$\begin{aligned} c\left\lfloor\frac{n}{2}\right\rfloor &\geq c\,\frac{n-1}{2} \\ &= 20\,\frac{n-1}{2} \\ &= 10n-10 \\ &> 6n+5 \quad \text{(pois } n \geq 4\text{)} \end{aligned}$$

Eis constantes que fazem o serviço: $c = 20$ e $n_0 = 4$

Conclusão

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

MERGE-SORT (A, p, r)
1 **se** $p < r$
2 **então** $q \leftarrow \lfloor (p + r)/2 \rfloor$
3 MERGE-SORT (A, p, q)
4 MERGE-SORT $(A, q + 1, r)$
5 INTERCALA (A, p, q, r)

Conclusão:

O MERGE-SORT consome $\mathrm{O}(n \lg n)$ unidades de tempo.

Exemplo 4 (continuação)

Vou mostrar que $T(n) \geq n \lg n$ para $n = 1, 2, 3, 4, \ldots$

n	1	2	3	4	5	6	7	8	9	10
$T(n)$	1	19	43	67	97	127	157	187	223	259
$n\lceil \lg n \rceil$	0	2	6	8	15	18	21	24	36	40

Exemplo 4 (continuação)

Vou mostrar que $T(n) \geq n \lg n$ para $n = 1, 2, 3, 4, \ldots$

n	1	2	3	4	5	6	7	8	9	10
$T(n)$	1	19	43	67	97	127	157	187	223	259
$n\lceil \lg n \rceil$	0	2	6	8	15	18	21	24	36	40

Prova:
Se $n = 1$, então $T(1) = 1 > 1 \cdot \lg 1 = 0$.

Exemplo 4

Prova: (continuação) Se $n \geq 2$, então

$$
\begin{aligned}
T(n) &= T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + 6n + 5 \\
&\overset{\text{hi}}{\geq} \left\lceil \frac{n}{2} \right\rceil \lg \left\lceil \frac{n}{2} \right\rceil + \left\lfloor \frac{n}{2} \right\rfloor \lg \left\lfloor \frac{n}{2} \right\rfloor + 6n + 5 \\
&\geq \left\lceil \frac{n}{2} \right\rceil \lg \frac{n}{2} + \left\lfloor \frac{n}{2} \right\rfloor \lg \frac{n-1}{2} + 6n + 5 \\
&= \left\lceil \frac{n}{2} \right\rceil (\lg n - 1) + \left\lfloor \frac{n}{2} \right\rfloor (\lg(n-1) - 1) + 6n + 5 \\
&\geq \left\lceil \frac{n}{2} \right\rceil (\lg n - 1) + \left\lfloor \frac{n}{2} \right\rfloor (\lg n - 2) + 6n + 5 \\
&= \left(\left\lceil \frac{n}{2} \right\rceil + \left\lfloor \frac{n}{2} \right\rfloor\right) \lg n - \left\lceil \frac{n}{2} \right\rceil - 2 \left\lfloor \frac{n}{2} \right\rfloor + 6n + 5 \\
&\geq n \lg n - 3 \left\lceil \frac{n}{2} \right\rceil + 6n + 5 \geq n \lg n.
\end{aligned}
$$

Iiiiéééésssss!

Exemplo 4

$$\begin{aligned} T(1) &= 1 \\ T(n) &= T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + 6n + 5 \quad \text{para } n = 2, 3, 4, 5, \ldots \end{aligned}$$

n	1	2	3	4	5	6	7	8	9	10
$T(n)$	1	19	43	67	97	127	157	187	223	259

Conclusão:

$T(n)$ é $\Theta(n \lg n)$.

Conclusão da conclusão

O consumo de tempo do MERGE-SORT é $\Theta(n \lg n)$ no pior caso.

Exercício. Mostre que:

O consumo de tempo do MERGE-SORT é $\Theta(n \lg n)$.

Hmmmm... Qual a diferença entra as duas afirmações?

Recorrências com notação O

CLRS 4.1–4.2

AU 3.9, 3.11

Classe O da solução de uma recorrência

Não faço questão de solução exata: basta solução aproximada (em notação O, se possível Θ)

Classe O da solução de uma recorrência

Não faço questão de solução exata: basta solução aproximada (em notação O, se possível Θ)

Exemplo:

$$\begin{aligned} G(1) &= 1 \\ G(n) &= 2G(n/2) + 7n + 2 \quad \text{para } n = 2, 4, 8, 16, \ldots \end{aligned}$$

Solução exata: $G(n) = 7n \lg n + 3n - 2$

Solução aproximada: $G(n) = \mathrm{O}(n \lg n)$ $\quad (G(n) = \Theta(n \lg n))$

Classe O da solução de uma recorrência

Não faço questão de solução exata: basta solução aproximada (em notação O, se possível Θ)

Exemplo:

$$\begin{aligned} G(1) &= 1 \\ G(n) &= 2G(n/2) + 7n + 2 \quad \text{para } n = 2, 4, 8, 16, \ldots \end{aligned}$$

Solução exata: $G(n) = 7n \lg n + 3n - 2$

Solução aproximada: $G(n) = \mathrm{O}(n \lg n)$ $\quad (G(n) = \Theta(n \lg n))$

Em geral, é mais fácil obter e provar solução aproximada que solução exata

Dica prática (sem prova)

A solução da recorrência

$$\begin{aligned} T(1) &= 1 \\ T(n) &= 2T(\lfloor n/2 \rfloor) + 7n + 2 \quad \text{para } n = 2, 3, 4, 5, \ldots \end{aligned}$$

está na mesma classe Θ que a solução de

$$\begin{aligned} T'(1) &= 1 \\ T'(n) &= 2T'(n/2) + n \quad \text{para } n = 2, 2^2, 2^3, \ldots \end{aligned}$$

Dica prática (sem prova)

A solução da recorrência

$$\begin{aligned} T(1) &= 1 \\ T(n) &= 2T(\lfloor n/2 \rfloor) + 7n + 2 \quad \text{para } n = 2, 3, 4, 5, \ldots \end{aligned}$$

está na mesma classe Θ que a solução de

$$\begin{aligned} T'(1) &= 1 \\ T'(n) &= 2T'(n/2) + n \quad \text{para } n = 2, 2^2, 2^3, \ldots \end{aligned}$$

e na mesma classe Θ que a solução de

$$\begin{aligned} T''(4) &= 10 \\ T''(n) &= 2T''(n/2) + n \quad \text{para } n = 2^3, 2^4, 2^5, \ldots \end{aligned}$$

Recorrências com O do lado direito

A “recorrência”

$$T(n) = 2T(n/2) + \mathrm{O}(n)$$

representa todas as recorrências da forma $T(n) = 2T(n/2) + f(n)$ em que $f(n)$ é $\mathrm{O}(n)$.

Recorrências com O do lado direito

A "recorrência"

$$T(n) = 2T(n/2) + \mathrm{O}(n)$$

representa todas as recorrências da forma
$T(n) = 2T(n/2) + f(n)$ em que $f(n)$ é $\mathrm{O}(n)$.

Melhor: representa todas as recorrências do tipo

$$\begin{aligned} T'(n) &\leq a \quad \text{para } n = k, k+1, \ldots, 2k-1 \\ T'(n) &\leq 2T'(\lfloor n/2 \rfloor) + b\,n \quad \text{para } n \geq 2k \end{aligned}$$

quaisquer que sejam a, $b > 0$ e $k > 0$
(poderíamos tomar $n_0 = 1$; veja ex ??.I)

Recorrências com O do lado direito

A "recorrência"

$$T(n) = 2T(n/2) + \mathrm{O}(n)$$

também representa todas as recorrências do tipo

$$\begin{aligned} T''(n) &= a \quad \text{para } n = 2^{k-1} \\ T''(n) &\le 2T''(n/2) + b\,n \quad \text{para } n = 2^k, 2^{k+1}, \ldots \end{aligned}$$

quaisquer que sejam a, $b > 0$ e $k > 0$

Recorrências com O do lado direito

A "recorrência"

$$T(n) = 2T(n/2) + \mathrm{O}(n)$$

também representa todas as recorrências do tipo

$$\begin{aligned} T''(n) &= a \quad \text{para } n = 2^{k-1} \\ T''(n) &\le 2T''(n/2) + b\,n \quad \text{para } n = 2^k, 2^{k+1}, \ldots \end{aligned}$$

quaisquer que sejam a, $b > 0$ e $k > 0$

As soluções exatas vão depender de a, b, k;
mas todas estarão na mesma classe O (no caso, em $\mathrm{O}(n \lg n)$)

Exemplo com O

Recorrência com O do lado direito

Na análise do consumo de tempo do MERGE-SORT tinhamos:

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

$$T(1) = \mathrm{O}(1)$$

$$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \mathrm{O}(n) \quad \text{para } n = 2, 3, 4, \ldots$$

Solução aproximada: $T(n)$ é $\mathrm{O}(n \lg n)$.

Exemplo com Θ

Recorrências com Θ do lado direito

Na análise do consumo de tempo do MERGE-SORT poderíamos ter feito:

$T(n)$:= consumo de tempo quando $n = r - p + 1$

$$T(1) = \Theta(1)$$
$$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n) \quad \text{para } n = 2, 3, 4, \ldots$$

Solução aproximada: $T(n)$ é $\Theta(n \lg n)$.

Dica prática

Suponha dada uma "recorrência" como

$$T(n) = 2T(n/2) + \mathrm{O}(n).$$

Para ter uma idéia da classe O a que T pertence, resolva a recorrência

$$\begin{aligned} T''(2^0) &= 1 \\ T''(n) &\leq 2T''(n/2) + n \quad \text{para } n = 2^1, 2^2, 2^3, \ldots \end{aligned}$$

Mais dicas práticas

recorrência	condição	solução
$T(n) = T(n-1) + 4n^3$		$\Theta(n^{3+1})$
$T(n) = 6T(n-1) + 4n^3$		$\Theta(6^n)$
$T(n) = aT(n/5) + 4n^3$	$a < 5^3$	$\Theta(n^3)$
$T(n) = aT(n/5) + 4n^3$	$a = 5^3$	$\Theta(n^3 \log n)$
$T(n) = aT(n/5) + 4n^3$	$a > 5^3$	$\Theta(n^{\log_5 a})$

Veja AU, sec 3.11, p.151

Mais dicas práticas (continuação)

No lugar de

- "$n/5$", posso escrever "$\lfloor n/5 \rfloor$" ou "$\lceil n/5 \rceil$"
- "5", posso escrever qualquer $b > 1$
- "4", posso escrever qualquer número real
- "$4n^3$" posso escrever qualquer polinômio de grau 3
- "3", posso escrever qualquer inteiro $k \geq 0$
- "6", posso escrever qualquer número $a > 1$

Mais dicas práticas ainda

A mesma coisa, escrita de maneira um pouco diferente (Master Theorem, CLRS, sec. 4.3, p.73):

Suponha $T(n) = aT(n/5) + f(n)$ para algum $a \geq 1$. Então, em geral,

$$
\begin{array}{lll}
\text{se } f(n) = \mathrm{O}(n^{\log_5 a - \epsilon}) & \text{então} & T(n) = \Theta(n^{\log_5 a}) \\
\text{se } f(n) = \Theta(n^{\log_5 a}) & \text{então} & T(n) = \Theta(n^{\log_5 a} \lg n) \\
\text{se } f(n) = \Omega(n^{\log_5 a + \epsilon}) & \text{então} & T(n) = \Theta(f(n))
\end{array}
$$

para qualquer $\epsilon > 0$

Exercícios

Exercício 9.A
A que classe O pertencem as solução de recorrência do tipo $T(n) = T(n/3) + \mathrm{O}(1)$?

Exercício 9.B
Seja T a função definida pela recorrência

$$\begin{aligned} T(1) &= 1 \\ T(n) &= 4T(\lfloor n/2 \rfloor) + n \qquad \text{para } n = 2, 3, 4, 5, \ldots \end{aligned}$$

A que ordem Θ pertence T?

Exercício 9.C [CLRS 4.2-1]
Seja T a função definida pela recorrência

$$\begin{aligned} T(1) &= 1 \\ T(n) &= 3T(\lfloor n/2 \rfloor) + n \qquad \text{para } n = 2, 3, 4, 5, \ldots \end{aligned}$$

A partir da árvore da recorrência, adivinhe a que classe Θ pertence $T(n)$. Prove a delimitação pelo método da substituição.

Mais exercícios

Exercício 9.D
Resolva a recorrência

$$\begin{aligned} T(1) &= 1 \\ T(n) &= 2T(\lceil n/2 \rceil) + 7n + 2 \quad \text{para } n = 2, 3, 4, 5, \ldots \end{aligned}$$

Exercício 9.E
Resolva a "recorrência" $T(n) = T(n-2) + \mathrm{O}(n)$.

Exercício 9.F
Resolva a "recorrência" $T(n) = 5T(n-1) + \mathrm{O}(n)$.

Melhores momentos

AULA 5

Recorrências

$$
\begin{aligned}
T(1) &= 1 \\
T(n) &= T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + 6n + 5 \quad \text{para } n = 2, 3, 4, 5, \ldots
\end{aligned}
$$

n	1	2	3	4	5	6	7	8	9	10
$T(n)$	1	19	43	67	97	127	157	187	223	259

Vimos que:

$T(n)$ é $\Theta(n \lg n)$.

Conclusão

O consumo de tempo do MERGE-SORT é $\Theta(n \lg n)$ no pior caso.

Exercício. Mostre que:

O consumo de tempo do MERGE-SORT é $\Theta(n \lg n)$.

Hmmmm... Qual a diferença entra as duas afirmações?

Classe O da solução de uma recorrência

Não faço questão de solução exata: basta solução aproximada (em notação O; melhor ainda Θ)

Exemplo:

$$\begin{aligned} G(1) &= 1 \\ G(n) &= 2G(n/2) + 7n + 2 \quad \text{para } n = 2, 4, 8, 16, \ldots \end{aligned}$$

Solução exata: $G(n) = 7n \lg n + 3n - 2$

Solução aproximada: $G(n) = \mathrm{O}(n \lg n)$ $\quad$ ($G(n) = \Theta(n \lg n)$!)

Em geral, é mais fácil obter e provar solução aproximada que solução exata

Dica prática (sem prova)

A solução da recorrência

$$\begin{aligned} T(1) &= 1 \\ T(n) &= 2T(\lfloor n/2 \rfloor) + 7n + 2 \quad \text{para } n = 2, 3, 4, 5, \ldots \end{aligned}$$

está na mesma classe Θ que a solução de

$$\begin{aligned} T'(1) &= 1 \\ T'(n) &= 2T'(n/2) + n \quad \text{para } n = 2, 2^2, 2^3, \ldots \end{aligned}$$

e na mesma classe Θ que a solução de

$$\begin{aligned} T''(4) &= 10 \\ T''(n) &= 2T''(n/2) + n \quad \text{para } n = 2^3, 2^4, 2^5, \ldots \end{aligned}$$

Recorrências com O do lado direito

A "recorrência"

$$T(n) = 2T(n/2) + \mathrm{O}(n)$$

representa todas as recorrências da forma $T(n) = 2T(n/2) + f(n)$ em que $f(n)$ é $\mathrm{O}(n)$.

Melhor: representa todas as recorrências do tipo

$$\begin{aligned} T'(n) &\leq a \quad \text{para } n = k, k+1, \ldots, 2k-1 \\ T'(n) &\leq 2T'(\lfloor n/2 \rfloor) + b\,n \quad \text{para } n \geq 2k \end{aligned}$$

quaisquer que sejam a, $b > 0$ e $k > 0$ (poderíamos tomar $n_0 = 1$; veja ex ??.I)

Exemplo com O

Recorrência com O do lado direito

Na análise do consumo de tempo do MERGE-SORT tinhamos:

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

$$T(1) = \mathrm{O}(1)$$

$$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \mathrm{O}(n) \quad \text{para } n = 2, 3, 4, \ldots$$

Solução aproximada: $T(n)$ é $\mathrm{O}(n \lg n)$.

Exemplo com Θ

Recorrências com Θ do lado direito

Na análise do consumo de tempo do MERGE-SORT tinhamos:

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

$$T(1) = \Theta(1)$$

$$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n) \quad \text{para } n = 2, 3, 4, \ldots$$

Solução aproximada: $T(n)$ é $\Theta(n \lg n)$.

Teorema Ban-Ban-Ban

Teorema Mestre (Master Theorem, CLRS, sec. 4.3, p.73):

Suponha

$$T(n) = a\,T(n/b) + f(n)$$

para algum $a \geq 1$ e $b > 1$ e onde n/b significa $\lceil n/b \rceil$ ou $\lfloor n/b \rfloor$.
Então, em geral,

se $f(n) = \mathrm{O}(n^{\log_b a - \epsilon})$ então $T(n) = \Theta(n^{\log_b a})$
se $f(n) = \Theta(n^{\log_b a})$ então $T(n) = \Theta(n^{\log_b a} \lg n)$
se $f(n) = \Omega(n^{\log_b a + \epsilon})$ então $T(n) = \Theta(f(n))$

para qualquer $\epsilon > 0$.

AULA 6

Técnicas em projeto de algoritmos

Programming Pearls: Algorithm Design Techniques,

Jon Bentley, Addison-Wesley, 1986

Segmento de soma máxima

Um segmento de um vetor $A[1\,..\,n]$ é um qualquer subvetor da forma $A[e\,..\,d]$.

Problema: Dado um vetor $A[1\,..\,n]$ de números inteiros, determinar um segmento $A[e\,..\,d]$ de soma máxima.

Entra:

$$
\begin{array}{c c|c|c|c|c|c|c|c|c|c|c|}
 & \multicolumn{1}{c}{1} & \multicolumn{1}{c}{} & \multicolumn{1}{c}{} & \multicolumn{1}{c}{} & \multicolumn{1}{c}{} & \multicolumn{1}{c}{} & \multicolumn{1}{c}{} & \multicolumn{1}{c}{} & \multicolumn{1}{c}{} & \multicolumn{1}{c}{n} \\
\cline{2-11}
A & \multicolumn{1}{|c|}{31} & -41 & 59 & 26 & -53 & 58 & 97 & -93 & -23 & 84 \\
\cline{2-11}
\end{array}
$$

Segmento de soma máxima

Um segmento de um vetor $A[1 \mathinner{.\,.} n]$ é um qualquer subvetor da forma $A[e \mathinner{.\,.} d]$.

Problema: Dado um vetor $A[1 \mathinner{.\,.} n]$ de números inteiros, determinar um segmento $A[e \mathinner{.\,.} d]$ de soma máxima.

Entra:

	1									n
A	31	−41	59	26	−53	58	97	−93	−23	84

Sai:

	1		3				7			n
A	31	−41	59	26	−53	58	97	−93	−23	84

$A[e \mathinner{.\,.} d] = A[3 \mathinner{.\,.} 7]$ é segmento de soma máxima.

$A[3 \mathinner{.\,.} 7]$ tem soma 187.

Segmento de soma máxima

Problema (versão simplificada): Determinar a maior soma máxima de um segmento de um dado um vetor $A[1\,..\,n]$.

Entra:

	1									n
A	31	-41	59	26	-53	58	97	-93	-23	84

Sai:

	1		3				7			n
A	31	-41	59	26	-53	58	97	-93	-23	84

A soma máxima é 187.

Algoritmo café-com-leite

Algoritmo determina um segmento de soma máxima de $A[1\,..\,n]$.

SEG-MAX-3 (A, n)

$somamax \leftarrow 0$
$e \leftarrow 0 \quad d \leftarrow -1 \quad \triangleright\ A[e\,..\,d]$ é vazio
para $i \leftarrow 1$ **até** n **faça**
    para $f \leftarrow i$ **até** n **faça**
        $soma \leftarrow 0$
        para $k \leftarrow i$ **até** f **faça**
            $soma \leftarrow soma + A[k]$
        se $soma > somamax$ **então**
            $somamax \leftarrow soma \quad e \leftarrow i \quad d \leftarrow f$
devolva e, d **e** $somamax$

Correção

Relação invariante chave:

(i0) na linha 3 vale que: $A[e\,..\,d]$ é um segmento de soma máxima com $e < i$.

	e		i				d			n
A	31	−41	59	26	−53	58	97	−93	−23	84

Correção

Relação invariante chave:

(i0) na linha 3 vale que: $A[e\,..\,d]$ é um segmento de soma máxima com $e < i$.

	e		i				d			n
A	31	−41	59	26	−53	58	97	−93	−23	84

Mais relações invariantes:

(i1) na linha 3 vale que:

$$somamax = A[e] + A[e+1] + A[e+2] + \cdots + A[d];$$

(i2) na linha 6 vale que:

$$soma = A[i] + A[i+1] + A[i+2] + \cdots + A[k-1].$$

Consumo de tempo

Se a execução de cada linha de código consome 1 unidade de tempo o consumo total é:

linha		todas as execuções da linha	
1-2	$=$	2	$= \Theta(1)$
3	$=$	$n+1$	$= \Theta(n)$
4	$=$	$(n+1)+n+(n-1)+\cdots+1$	$= \Theta(n^2)$
5	$=$	$n+(n-1)+\cdots+1$	$= \Theta(n^2)$
6	$=$	$(2+\cdots+(n+1))+(2+\cdots+n)+\cdots+2$	$= \Theta(n^3)$
7	$=$	$(1+\cdots+n)+(1+\cdots+(n-1))+\cdots+1$	$= \Theta(n^3)$
8	$=$	$n+(n-1)+(n-2)+\cdots+1$	$= \Theta(n^2)$
9	$\leq$	$n+(n-1)+(n-2)+\cdots+1$	$= \mathrm{O}(n^2)$
10	$=$	1	$= \Theta(1)$
total	$=$	$\Theta(2n^3+3n^2+n+2)+\mathrm{O}(n^2)$	$= \Theta(n^3)$

Algoritmo arroz-com-feijão

Algoritmo determina um segmento de soma máxima de $A[1\,.\,.\,n]$.

SEG-MAX-2 (A, n)
    $somamax \leftarrow 0$
    $e \leftarrow 0 \quad d \leftarrow -1 \quad \triangleright\ A[e\,.\,.\,d]$ é vazio
    para $i \leftarrow 1$ **até** n **faça**
        $soma \leftarrow 0$
        para $f \leftarrow i$ **até** n **faça**
            $soma \leftarrow soma + A[f]$
            se $soma > somamax$ **então**
                $somamax \leftarrow soma \quad e \leftarrow i \quad d \leftarrow f$
    devolva e, d **e** $somamax$

Correção

Relação invariante chave:

(i0) na linha 3 vale que: $A[e\,..\,d]$ é um segmento de soma máxima com $e < i$.

	e						d	i		n
A	31	-41	59	26	-53	58	97	-93	-23	84

Mais relações invariante:

(i1) na linha 3 vale que:

$$somamax = A[e] + A[e+1] + A[e+2] + \cdots + A[d];$$

(i2) na linha 5 vale que:

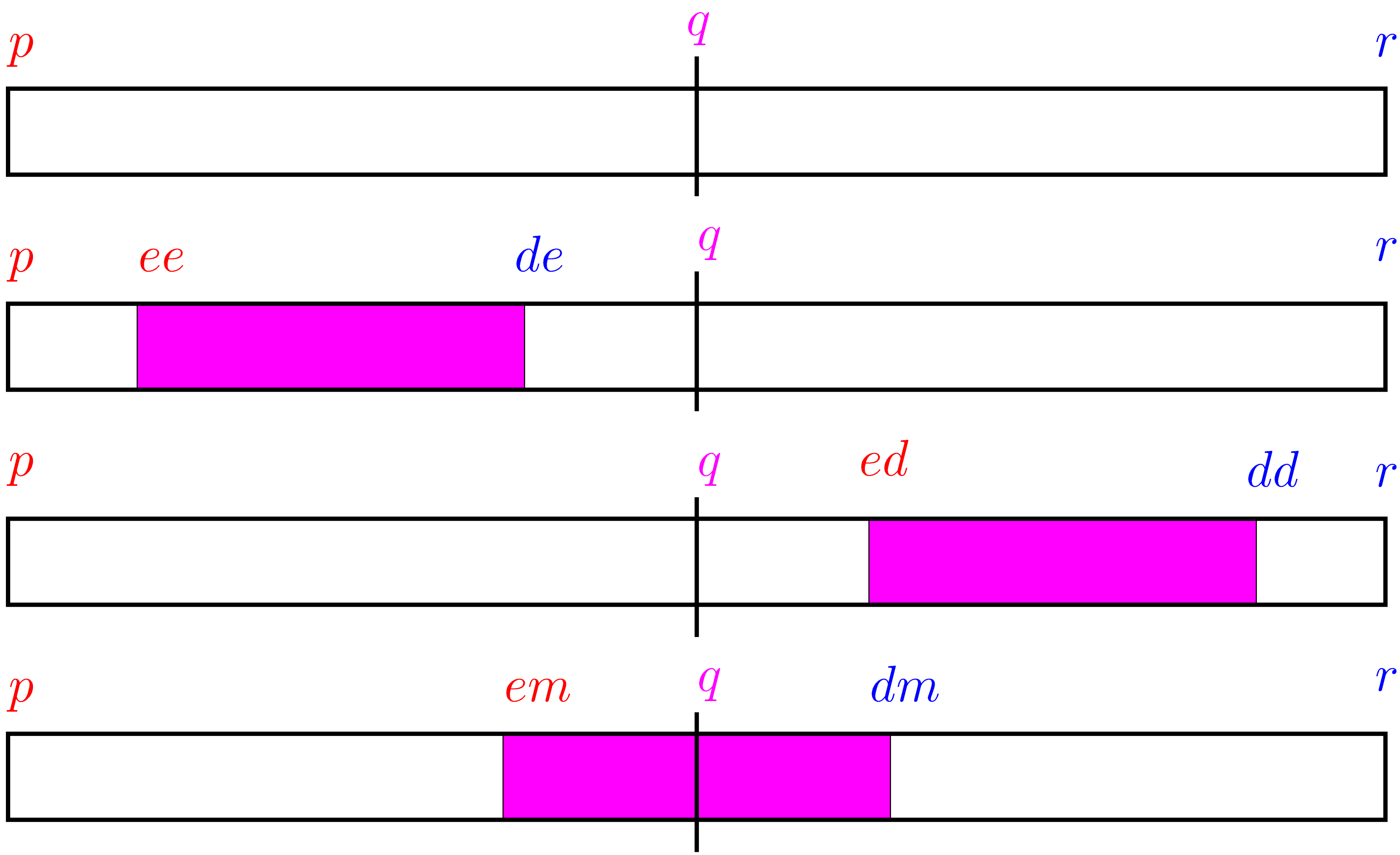
$$soma = A[i] + A[i+1] + A[i+2] + \cdots + A[f-1];$$

Consumo de tempo

Se a execução de cada linha de código consome 1 unidade de tempo o consumo total é:

linha		todas as execuções da linha	
1-2	$=$	2	$= \Theta(1)$
3	$=$	$n+1$	$= \Theta(n)$
4	$=$	n	$= \Theta(n)$
5	$=$	$(n+1)+n+\cdots+2$	$= \Theta(n^2)$
6	$=$	$n+(n-1)+\cdots+1$	$= \Theta(n^2)$
7	$=$	$n+(n-1)+\cdots+1$	$= \Theta(n^2)$
8	$\leq$	$n+(n-1)+\cdots+1$	$= \mathrm{O}(n^2)$
9	$=$	1	$= \Theta(1)$
total	$=$	$\Theta(3n^2+2n+2)+\mathrm{O}(n^2)$	$= \Theta(n^2)$

Solução de divisão-e-conquista



Algoritmo de divisão-e-conquista

Algoritmo determina soma máxima de um seg. de $A[p \mathinner{.\,.} r]$.

SEG-MAX-DC (A, p, r)

se $p = r$ **então devolva** $\max(0, A[p])$
$q \leftarrow \lfloor (p + r)/2 \rfloor$
$maxesq \leftarrow$ SEG-MAX-DC(A, p, q)
$maxdir \leftarrow$ SEG-MAX-DC$(A, q + 1, r)$
$max2esq \leftarrow soma \leftarrow A[q]$
para $i \leftarrow q - 1$ **decrescendo até** p **faça**
    $soma \leftarrow soma + A[i]$
    $max2esq \leftarrow \max(max2esq, soma)$
$max2dir \leftarrow soma \leftarrow A[q + 1]$
para $f \leftarrow q + 2$ **até** r **faça**
    $soma \leftarrow soma + A[f]$
    $max2dir \leftarrow \max(max2dir, soma)$
$maxcruz \leftarrow max2esq + max2dir$
devolva $\max(maxesq, maxcruz, maxdir)$

Correção

Verifique que:

- $maxesq$ é a soma máxima de um segmento de $A[p \mathinner{..} q]$;
- $maxdir$ é a soma máxima de um segmento de $A[q+1 \mathinner{..} r]$; e
- $maxcruz$ é a soma máximo de um segmento da forma $A[i \mathinner{..} f]$ com $i \leq q$ e $q+1 \leq f$.

Conclua que o algoritmo devolve a soma máxima de um segmento de $A[p \mathinner{..} r]$.

Consumo de tempo

Se a execução de cada linha de código consome 1 unidade de tempo o consumo total é:

linha		todas as execuções da linha	
1-2	$=$	2	$= \Theta(1)$
3	$=$	$T(\lceil \frac{n}{2} \rceil)$	$= T(\lceil \frac{n}{2} \rceil)$
4	$=$	$T(\lfloor \frac{n}{2} \rfloor)$	$= T(\lfloor \frac{n}{2} \rfloor)$
5	$=$	1	$= \Theta(1)$
6	$=$	$\lceil \frac{n}{2} \rceil + 1$	$= \Theta(n)$
7-8	$=$	$\lceil \frac{n}{2} \rceil$	$= \Theta(n)$
9	$=$	1	$= \Theta(1)$
10	$=$	$\lfloor \frac{n}{2} \rfloor + 1$	$= \Theta(n)$
11-12	$=$	$\lfloor \frac{n}{2} \rfloor$	$= \Theta(n)$
13-14	$=$	2	$= \Theta(1)$

total $= T(\lfloor \frac{n}{2} \rfloor) + T(\lfloor \frac{n}{2} \rfloor) + \Theta(4n + 4)$

Consumo de tempo

$T(n) :=$ consumo de tempo quando $n = r - p + 1$

Na análise do consumo de tempo do SEG-MAX-DC chegamos a (já manjada) recorrências com Θ do lado direito:

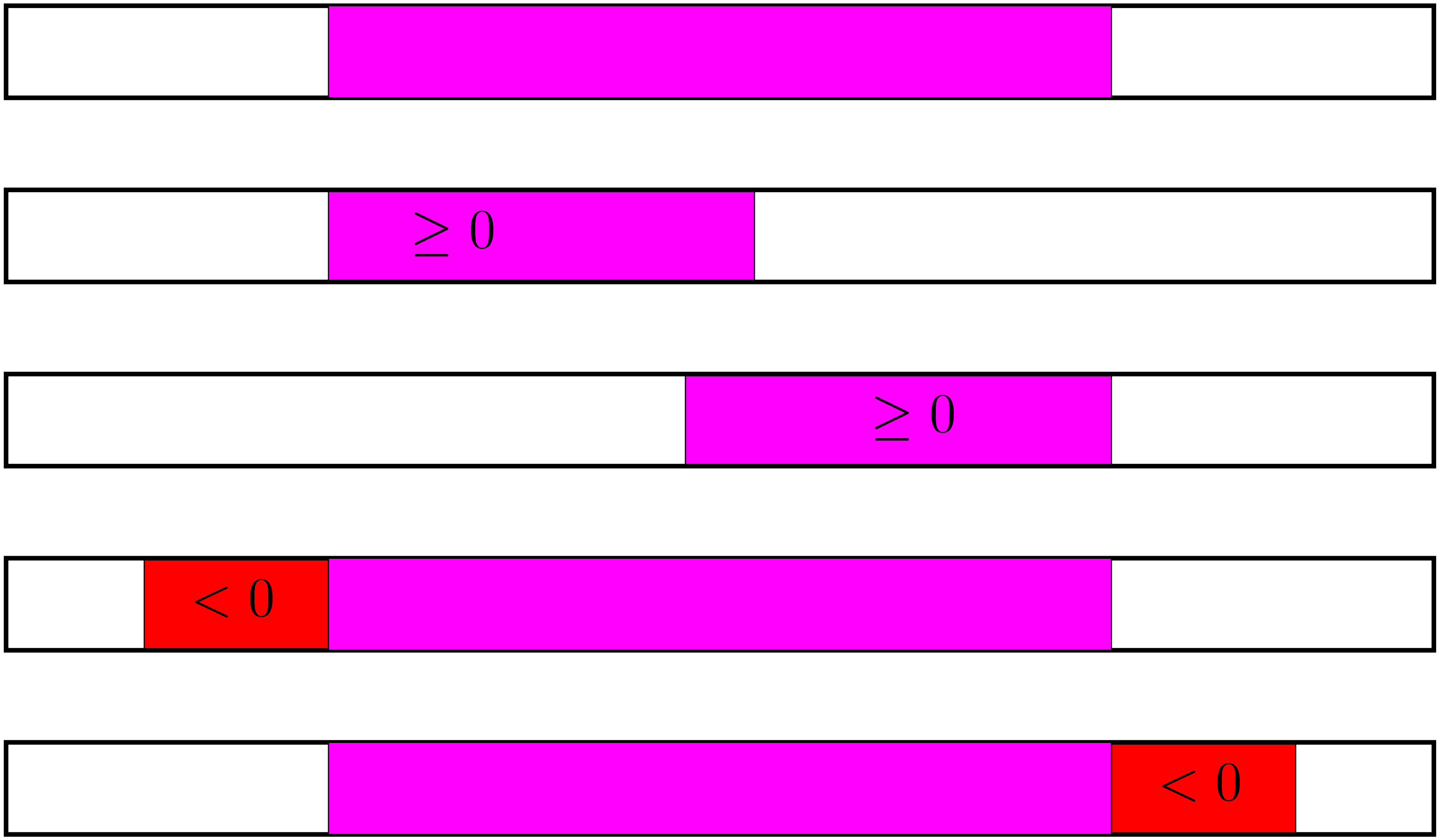
$$T(1) = \Theta(1)$$

$$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n) \quad \text{para } n = 2, 3, 4, \ldots$$

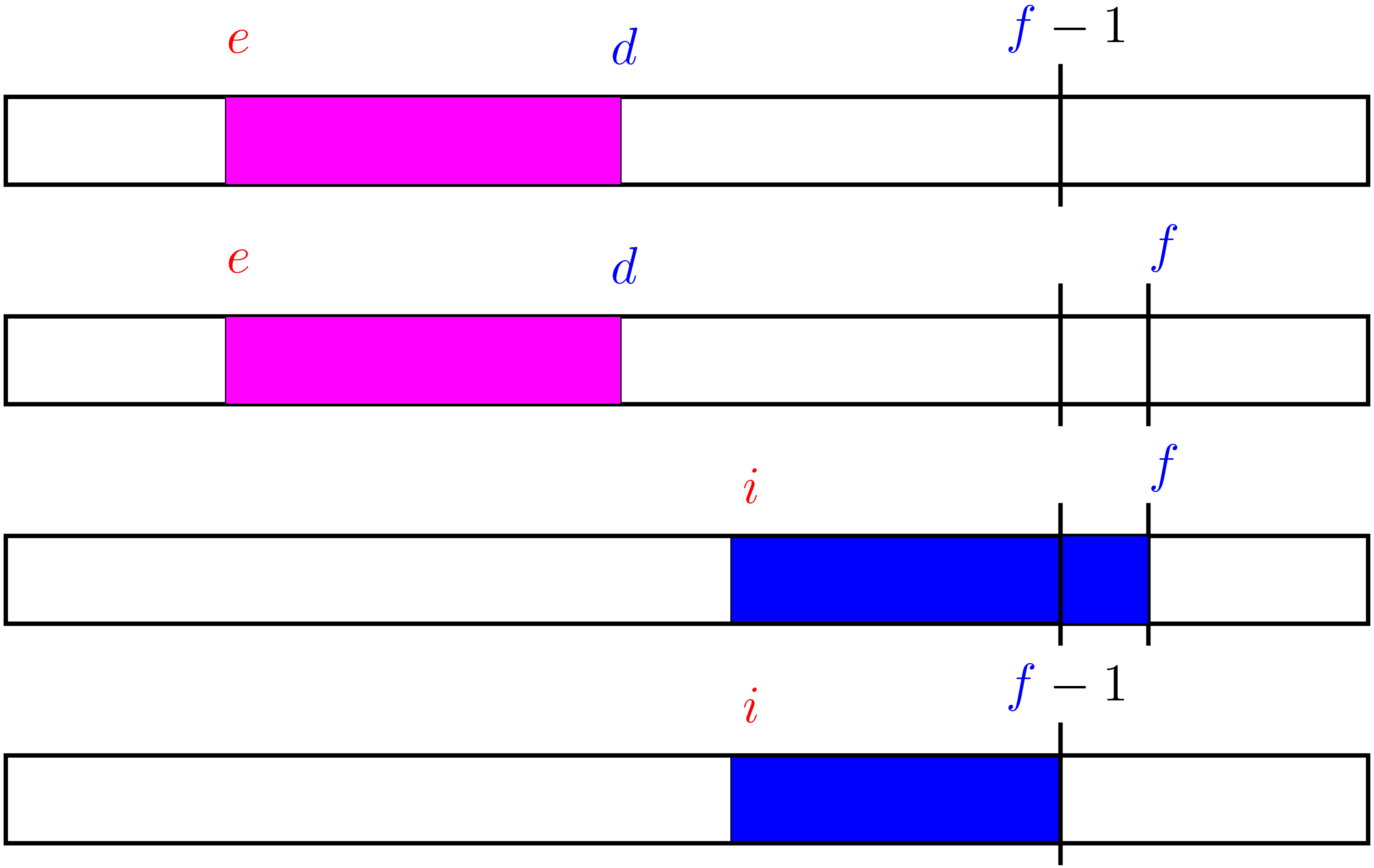
Solução aproximada/assintótica: $T(n)$ é $\Theta(n \lg n)$.

Cara da solução

Solução



Solução indutiva



Algoritmo linear

Algoritmo determina um segmento de soma máxima de $A[1\,..\,n]$ (por Jay Kadane).

SEG-MAX-1 (A, n)

$somamax \leftarrow 0$
$e \leftarrow 0 \quad d \leftarrow -1 \quad \triangleright\ A[e\,..\,d]$ é vazio
$i \leftarrow 1$
$soma \leftarrow 0$
para $f \leftarrow 1$ **até** n **faça**
    se $soma < 0$
        então $i \leftarrow f \quad soma \leftarrow A[f]$
        senão $soma \leftarrow soma + A[f]$
    se $soma > somamax$ **então**
        $somamax \leftarrow soma \quad e \leftarrow i \quad d \leftarrow f$
devolva e, d **e** $somamax$

Correção

Relação invariante chave:

(i0) na linha 5 vale que: $A[e \mathinner{..} d]$ é segmento de soma máxima com $d < f$.

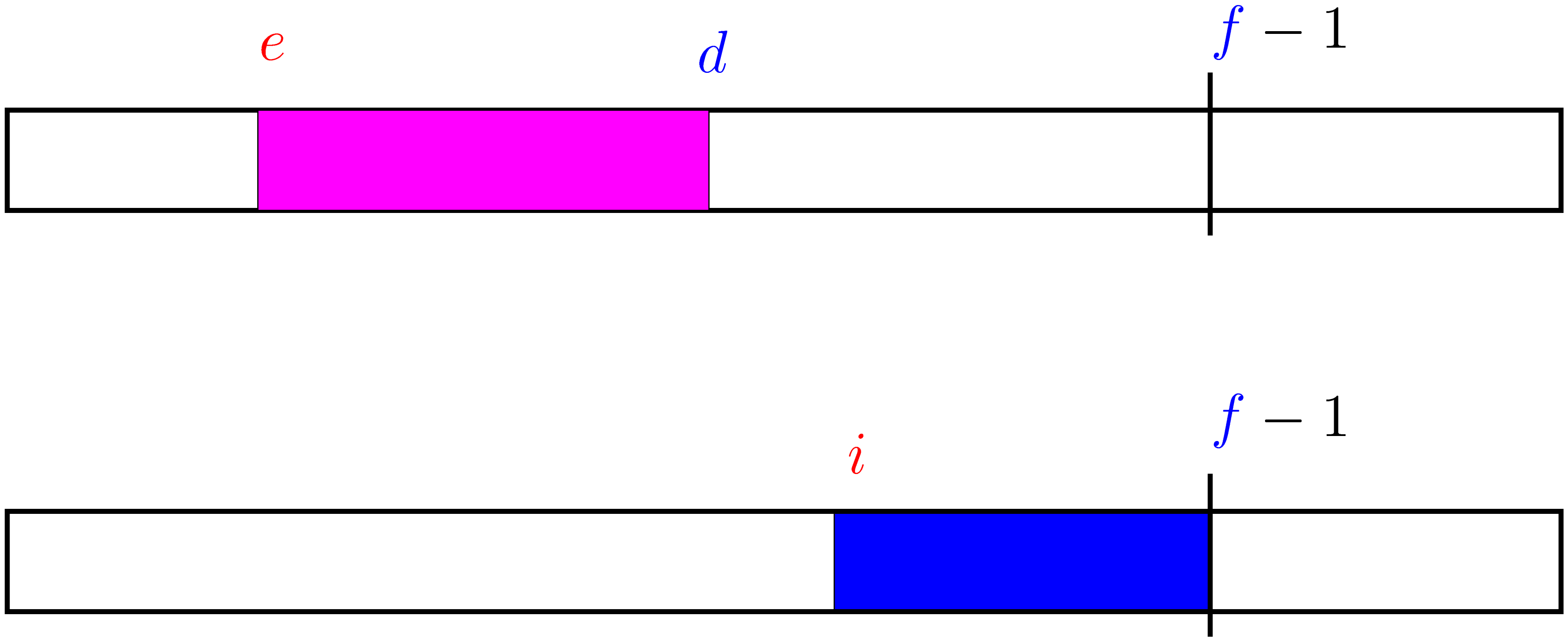
			e	d		f				n
A	31	−41	59	26	−53	58	97	−93	−23	84

Mais uma relação invariante:

(i1) na linha 5 vale que:

$$somamax = A[e] + A[e+1] + A[e+2] + \cdots + A[d].$$

Invariantes (i0) e (i3)



Correção (continuação)

Mais relações invariantes.
Na linha 5 vale que:

(i2) $A[i\,..\,f-1]$ é segmento de soma máxima com termino em $f-1$;
(i3) $soma = A[i] + A[i+1] + A[i+2] + \cdots + A[f-1]$;
(i4) $A[i] + A[i+1] + \cdots + A[k] \geq 0$
para todo $k = i, i+1, \ldots, f-2$;

Na linha 7 vale que:

(i5) $soma = A[i] + A[i+1] + \cdots + A[f-1] < 0$

As relações invariantes (i4) e (i5) implicam que na linha 7:

(i6) $A[k] + A[k+1] + \cdots + A[f-1] < 0$
para todo $k = i, i+1, \ldots, f-1$;

Consumo de tempo

Se a execução de cada linha de código consome 1 unidade de tempo o consumo total é:

linha		todas as execuções da linha	
1-2	$=$	2	$= \Theta(1)$
3-4	$=$	2	$= \Theta(1)$
5	$=$	$n+1$	$= \Theta(n)$
6	$=$	n	$= \Theta(n)$
7-8	$=$	n	$= \Theta(n)$
9	$=$	n	$= \Theta(n)$
10	$\leq$	n	$= \mathrm{O}(n)$
11	$=$	1	$= \Theta(1)$
total	$=$	$\Theta(4n+3) + \mathrm{O}(n)$	$= \Theta(n)$

Conclusões

O consumo de tempo do algoritmo SEG-MAX-3 é $\Theta(n^3)$.

O consumo de tempo do algoritmo SEG-MAX-2 é $\Theta(n^2)$.

O consumo de tempo do algoritmo SEG-MAX-DC é $\Theta(n \lg n)$.

O consumo de tempo do algoritmo SEG-MAX-1 é $\Theta(n)$.

Técnicas

- Evitar recomputações. Usar espaço para armazenar resultados a fim de evitar recomputá-los (SEG-MAX-2, SEG-MAX-1, programação dinâmica).
- Pré-processar os dados. O HEAPSORT pré-processa os dados armazenandos em uma entrutura de dados para realizar computações futuras mais eficientemente.
- Divisão-e-conquista. Os algoritmos Mergesort e SegMaxdc utilizam uma forma conhecidade dessa técnica.
- Algoritmos incrementais/varredura. Como estender a solução de um subproblema a uma solução do problema (SegMaxu).
- Delimitação inferior. Projetistas de algoritmos só dormem em paz quando sabem que seus algoritmos são o melhor possível (SegMaxu).

Análise experimental de algoritmos

"O interesse em experimentação, é devido ao reconhecimento de que os resultados teóricos, freqüentemente, não trazem informações referentes ao desempenho do algoritmo na prática."

Análise experimental de algoritmos

Segundo D.S. Johnson, pode-se dizer que existem quatro motivos básicos que levam a realizar um trabalho de implementação de um algoritmo:

- usar o código em uma aplicação particular, cujo propósito é descrever o impacto do algoritmo em um certo contexto;
- proporcionar evidências da superioridade de um algoritmo;
- melhor compreensão dos pontos fortes e fracos e do desempenho das operações algorítmicas na prática; e
- produzir conjecturas sobre o comportamento do algoritmo no caso-médio sob distribuições específicas de instâncias onde a análise probabilística direta é muito difícil.

Ambiente experimental

A plataforma utilizada nos experimentos é um PC rodando Linux Debian ?.? com um processador Pentium II de 233 MHz e 128MB de memória RAM .

Os códigos estão compilados com o gcc versão ?? e opção de compilação ??.

As instâncias são obtidas utilizando-se o gerador `gera.c` disponível em

`http://www.ime.usp.br/~coelho/algoritmos/tarefa1/.`

As implementações comparadas neste experimento são SEG-MAX-3, SEG-MAX-2 e SEG-MAX-1.

Ambiente experimental

A estimativa do tempo é calculada utilizando-se:

```
#include <time.h>
[...]
clock_t start, end;
double time;

start = clock();

[...implementação...]

end  = clock();
time = ((double)(end - start))/CLOCKS_PER_SEC;
```

O tempo de entrada/saída é computado separadamente.

Resultados experimentais

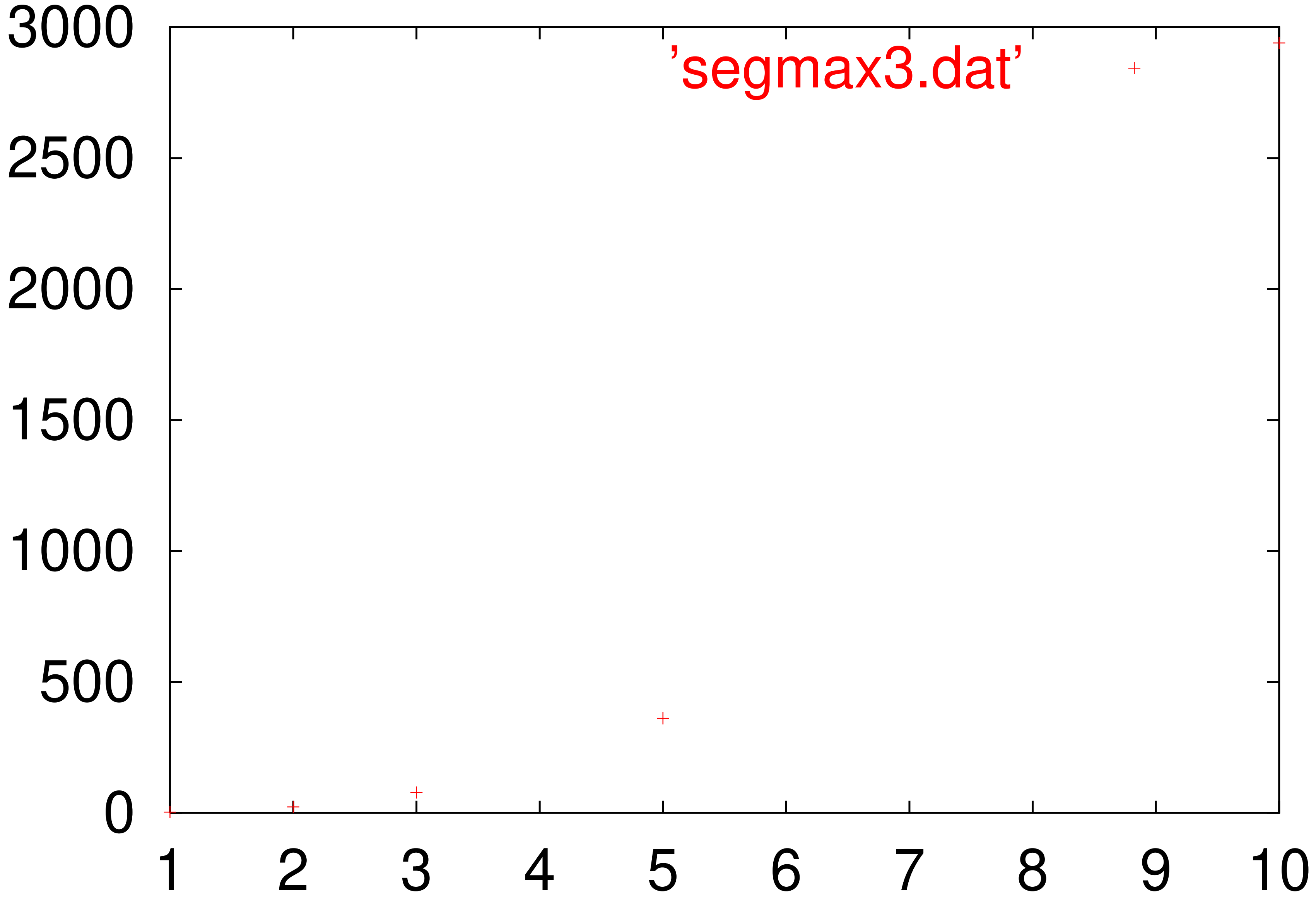
n	SEG-MAX-2		SEG-MAX-3	
	`clock`	`user time`	`clock`	`user time`
1000	0.01	0.01	2.91	2.92
2000	0.03	0.03	23.00	23.04
3000	0.08	0.08	77.58	1m17.57
5000	0.24	0.24	360.78	6m0.45s
10000	1.16	1.62	2940	49m27.32
20000	4.8	4.82	?	?
50000	42.8	42.96	?	?
100000	188.0	3m7.58s	?	?
200000	911.74	15m10.34s	?	?
400000	?	64m22.880s	?	?

Resultados experimentais

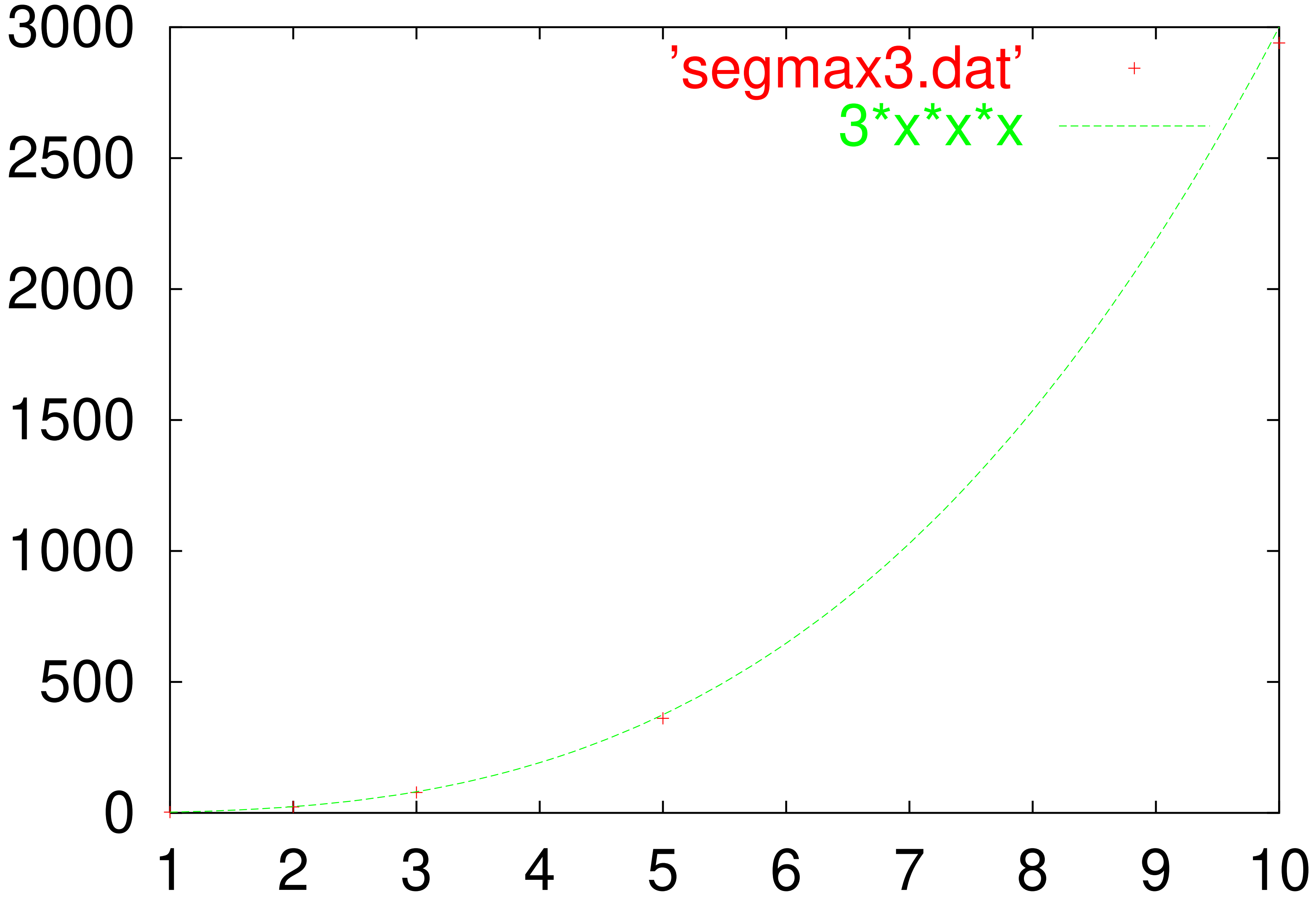
n (M)	SEG-MAX-1 A		SEG-MAX-1 B	
	`clock`	`user time`	`clock`	`user time`
1	0.06	0.11	0.08	0.07
2	0.11	0.23	0.15	0.06
3	0.16	0.28	0.30	0.12
6	0.32	0.63	0.56	0.29
8	0.43	0.80	0.71	0.43
10	0.55	1.03	1.00	0.52
15	0.82	1.55	1.37	0.64
20	1.08	2.03	2.17	0.95
25	1.37	2.52	2.44	1.38
30	1.70	3.13	3.28	1.68

Todos os tempos estão em segundos.

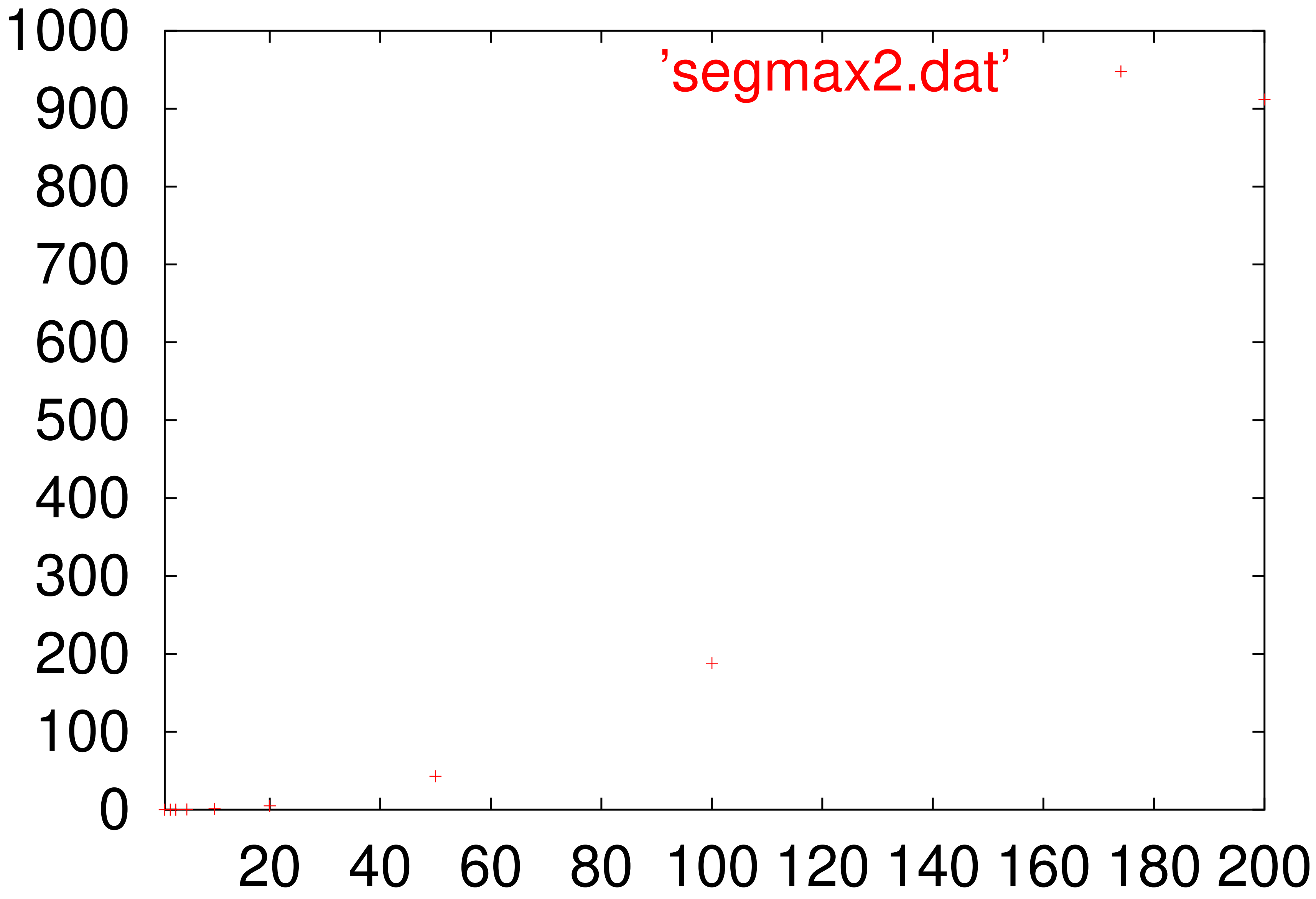
SEG-MAX-3



SEG-MAX-3

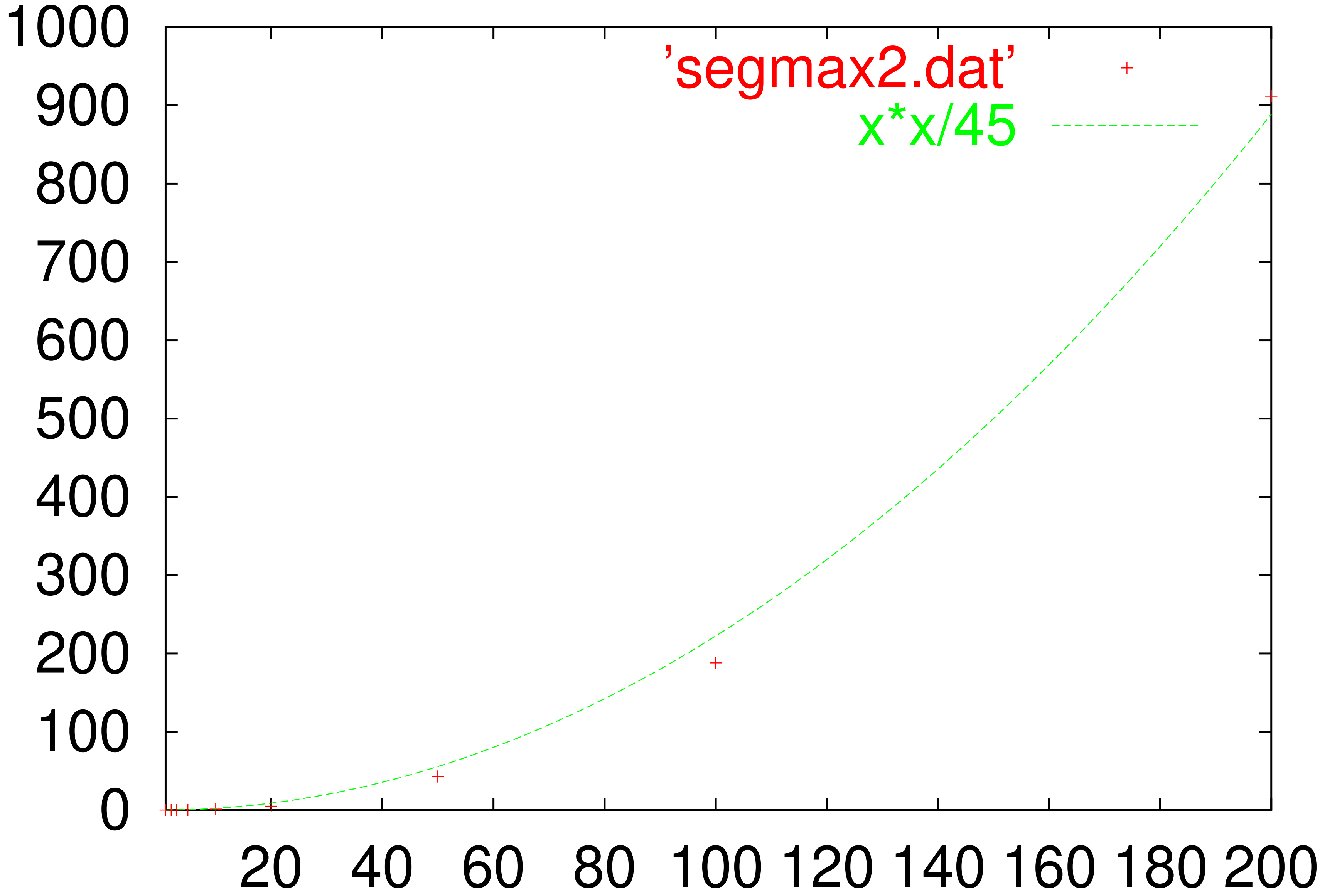


SEG-MAX-2

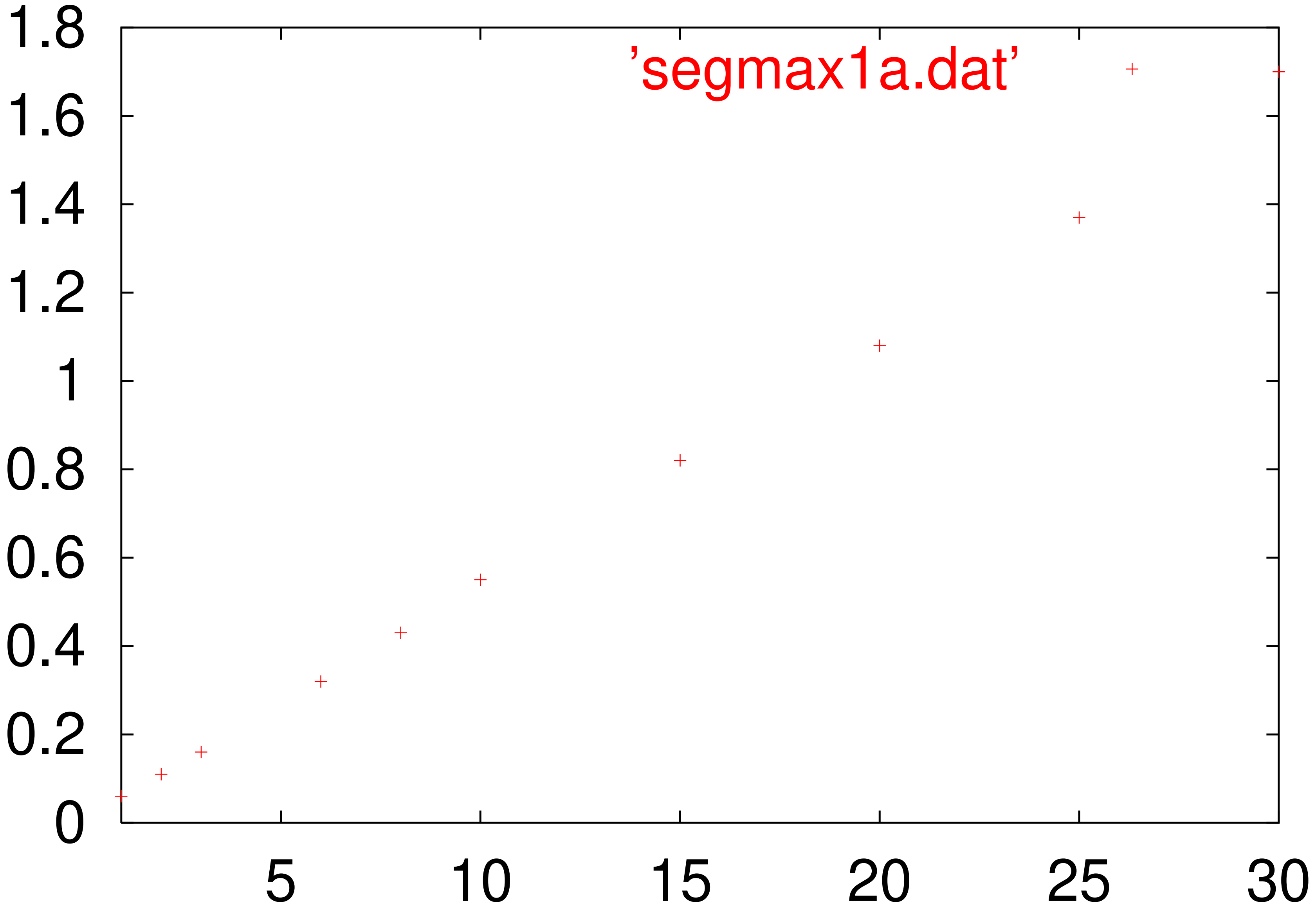


'segmax2.dat'
1000
900
800
700
600
500
400
300
200
100
0
20
40
60
80
100
120
140
160
180
200

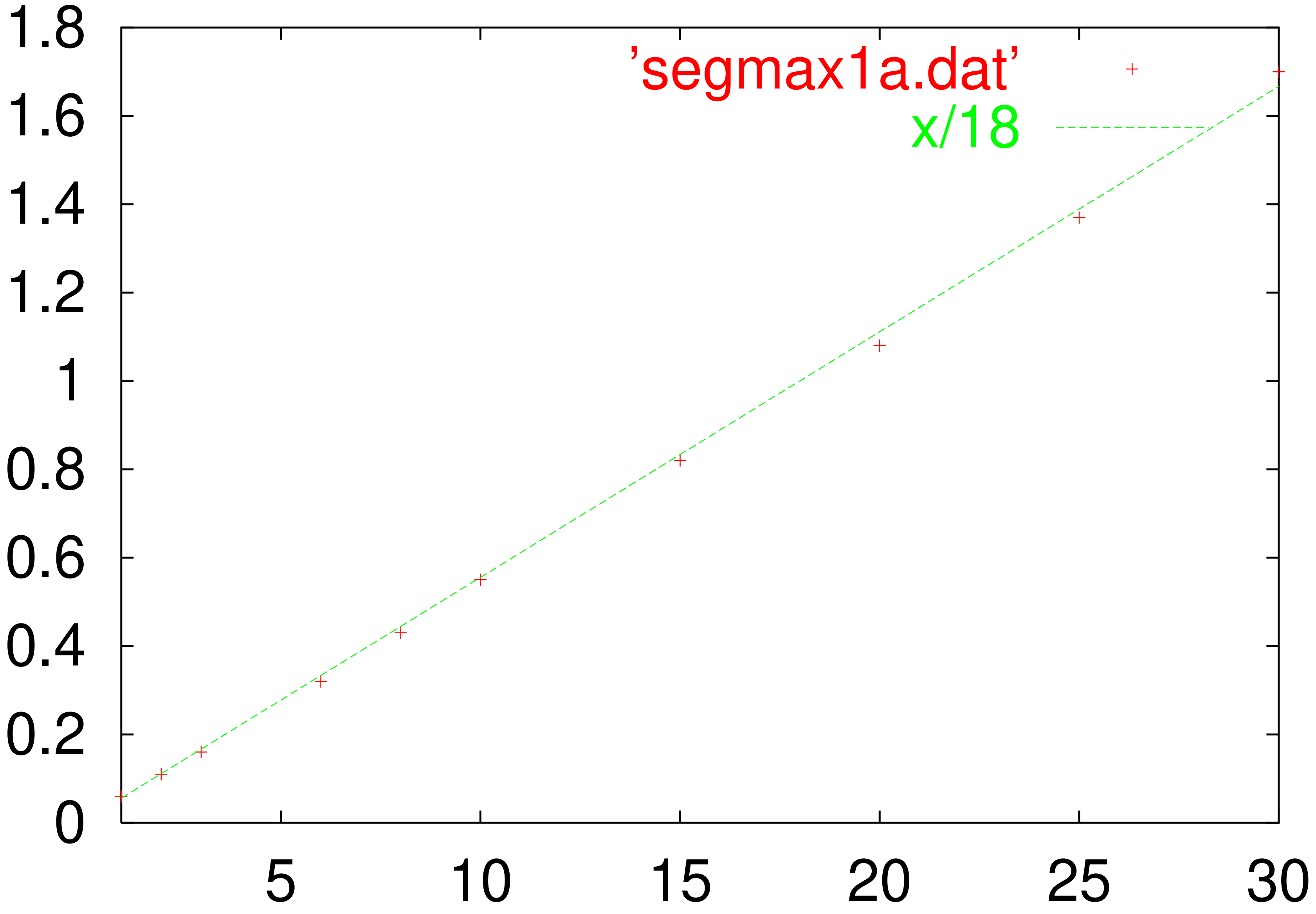
SEG-MAX-2


'segmax2.dat'
x*x/45
1000
900
800
700
600
500
400
300
200
100
0
20
40
60
80
100
120
140
160
180
200

SEG-MAX-1



SEG-MAX-1



Exercícios

Exercício 10.A
Nos algoritmos convencionamos o segmento vazio como sendo o de soma máxima se $A[1 \, .. \, n]$ tem apenas números negativos. Suponha que em vez disso tivéssemos decidido que nesse caso o maior elemento de $A[1 \, .. \, n]$ é o segmento de soma máxima. Como isto alteraria s quatro algoritmos?

Exercício 10.B
Suponha que desejamos determinar um segmento como soma mais próxima de zero, em vez do de soma máxima. Projete um algoritmo para esta tarefa. Qual é o consumo de tempo do seu algoritmo? E se estivéssemos interessados em um algoritmo para encontrar um segmento de soma mais próxima de um dado valor?

Exercício 10.C
No problema do retângulo de soma máxima é dada uma matriz $A[1 \, .. \, n, 1 \, .. \, n]$ de números inteiros e deseja-se determinar um retângulo de soma máxima. Projete um algoritmo para este problema. Qual o consumo de tempo do seu algoritmo?

Exercício 10.D
Modifique o algoritmo SEG-MAX-DC para que seu consumo de tempo seja linear.

Mais exercícios

Exercício 10.E
Demonstre que qualquer algoritmo para o problema do segmento de soma máxima deve examinar cada um dos n elementos do vetor. (Algoritmos para alguns problemas podem, corretamente, ignorar alguns dos dados; o algoritmo de Boyer e Moore para busca de padrões é um exemplo.)

Exercício 10.F
Projete um algoritmo que recebe um vetor $A[1\,..\,n]$ de números inteiros e um número inteiro m e devolve um índice i tal que $1 \leq i \leq n - m$ e a soma $A[i] + \cdots + A[i+m]$ seja próxima de zero. Qual o consumo de tempo do seu algoritmo.

Exercício 10.G
Resolva a recorrência

$$
\begin{aligned}
T(1) &= 0 \\
T(n) &= 2\,T(n/2) + n \quad \text{para } n = 2, 4, 8, \ldots
\end{aligned}
$$

Demonstre por indução que a sua solução está correta.

Melhores momentos

AULAS ANTERIORES

Classe O da solução de uma recorrência

Não faço questão de solução exata: basta solução aproximada (em notação O; melhor ainda Θ)

Exemplo:

$$\begin{array}{rcl} G(1) & = & 1 \\ G(n) & = & 2G(n/2) + 7n + 2 \quad \text{para } n = 2, 4, 8, 16, \ldots \end{array}$$

Solução exata: $G(n) = 7n \lg n + 3n - 2$

Solução aproximada: $G(n) = \mathrm{O}(n \lg n)$ $\quad (G(n) = \Theta(n \lg n)!)$

Em geral, é mais fácil obter e provar solução aproximada que solução exata

Dica prática (sem prova)

A solução da recorrência

$$\begin{aligned} T(1) &= 1 \\ T(n) &= 2T(\lfloor n/2 \rfloor) + 7n + 2 \quad \text{para } n = 2, 3, 4, 5, \ldots \end{aligned}$$

está na mesma classe Θ que a solução de

$$\begin{aligned} T'(1) &= 1 \\ T'(n) &= 2T'(n/2) + n \quad \text{para } n = 2, 2^2, 2^3, \ldots \end{aligned}$$

e na mesma classe Θ que a solução de

$$\begin{aligned} T''(4) &= 10 \\ T''(n) &= 2T''(n/2) + n \quad \text{para } n = 2^3, 2^4, 2^5, \ldots \end{aligned}$$

Recorrências com O do lado direito

A "recorrência"

$$T(n) = 2T(n/2) + \mathrm{O}(n)$$

representa todas as recorrências da forma
$T(n) = 2T(n/2) + f(n)$ em que $f(n)$ é $\mathrm{O}(n)$.

Melhor: representa todas as recorrências do tipo

$$\begin{aligned} T'(n) &\leq a \quad \text{para } n = k, k+1, \ldots, 2k-1 \\ T'(n) &\leq 2T'(\lfloor n/2 \rfloor) + b\,n \quad \text{para } n \geq 2k \end{aligned}$$

quaisquer que sejam a, $b > 0$ e $k > 0$
(poderíamos tomar $n_0 = 1$; veja ex ??.I)

Teorema Ban-Ban-Ban

Teorema Mestre (Master Theorem, CLRS, sec. 4.3, p.73):

Suponha

$$T(n) = a\,T\big(n/b\big) + f(n)$$

para algum $a \geq 1$ e $b > 1$ e onde n/b significa $\lceil n/b \rceil$ ou $\lfloor n/b \rfloor$. Então, em geral,

se $f(n) = \mathrm{O}(n^{\log_b a-\epsilon})$	então	$T(n) = \Theta(n^{\log_b a})$
se $f(n) = \Theta(n^{\log_b a})$	então	$T(n) = \Theta(n^{\log_b a} \lg n)$
se $f(n) = \Omega(n^{\log_b a+\epsilon})$	então	$T(n) = \Theta(f(n))$

para qualquer $\epsilon > 0$.

Teorema Ban-Ban-Ban

Teorema Mestre Simplificado:

Suponha

$$T(n) = a\,T(n/b) + c\,n^k$$

para algum $a \geq 1$ e $b > 1$ e onde n/b significa $\lceil n/b \rceil$ ou $\lfloor n/b \rfloor$. Então, em geral,

se $a > b^k$ então $T(n) = \Theta(n^{\log_b a})$
se $a = b^k$ então $T(n) = \Theta(n^k \lg n)$
se $a < b^k$ então $T(n) = \Theta(n^k)$

Segmento de soma máxima

Um segmento de um vetor $A[1 \mathinner{..} n]$ é um qualquer subvetor da forma $A[e \mathinner{..} d]$.

Problema: Dado um vetor $A[1 \mathinner{..} n]$ de números inteiros, determinar um segmento $A[e \mathinner{..} d]$ de soma máxima.

Entra:

$$
\begin{array}{c|c|c|c|c|c|c|c|c|c|c|}
 & 1 & & & & & & & & & n \\
\hline
A & 31 & -41 & 59 & 26 & -53 & 58 & 97 & -93 & -23 & 84 \\
\hline
\end{array}
$$

Sai:

$$
\begin{array}{c|c|c|c|c|c|c|c|c|c|c|}
 & 1 & & 3 & & & & 7 & & & n \\
\hline
A & 31 & -41 & 59 & 26 & -53 & 58 & 97 & -93 & -23 & 84 \\
\hline
\end{array}
$$

$A[e \mathinner{..} d] = A[3 \mathinner{..} 7]$ é segmento de soma máxima.
$A[3 \mathinner{..} 7]$ tem soma 187.

Conclusões

O consumo de tempo do algoritmo SEG-MAX-3 é $\Theta(n^3)$.

O consumo de tempo do algoritmo SEG-MAX-2 é $\Theta(n^2)$.

O consumo de tempo do algoritmo SEG-MAX-DC é $\Theta(n \lg n)$.

O consumo de tempo do algoritmo SEG-MAX-1 é $\Theta(n)$.

Técnicas

- Evitar recomputações. Usar espaço para armazenar resultados a fim de evitar recomputá-los (SEG-MAX-2, SEG-MAX-1, programação dinâmica).
- Pré-processar os dados. O HEAPSORT pré-processa os dados armazenandos em uma entrutura de dados para realizar computações futuras mais eficientemente.
- Divisão-e-conquista. Os algoritmos Mergesort e SegMaxdc utilizam uma forma conhecidade dessa técnica.
- Algoritmos incrementais/varredura. Como estender a solução de um subproblema a uma solução do problema (SegMaxu).
- Delimitação inferior. Projetistas de algoritmos só dormem em paz quando sabem que seus algoritmos são o melhor possível (SegMaxu).

Resultados experimentais

n	SEG-MAX-2		SEG-MAX-3	
	clock	user time	clock	user time
1000	0.01	0.01	2.91	2.92
2000	0.03	0.03	23.00	23.04
3000	0.08	0.08	77.58	1m17.57
5000	0.24	0.24	360.78	6m0.45s
10000	1.16	1.62	2940	49m27.32
20000	4.8	4.82	?	?
50000	42.8	42.96	?	?
100000	188.0	3m7.58s	?	?
200000	911.74	15m10.34s	?	?
400000	?	64m22.880s	?	?

Resultados experimentais

n (M)	SEG-MAX-1 A		SEG-MAX-1 B	
	`clock`	`user time`	`clock`	`user time`
1	0.06	0.11	0.08	0.07
2	0.11	0.23	0.15	0.06
3	0.16	0.28	0.30	0.12
6	0.32	0.63	0.56	0.29
8	0.43	0.80	0.71	0.43
10	0.55	1.03	1.00	0.52
15	0.82	1.55	1.37	0.64
20	1.08	2.03	2.17	0.95
25	1.37	2.52	2.44	1.38
30	1.70	3.13	3.28	1.68

Todos os tempos estão em segundos.

AULA 7

Mais análise de algoritmos recursivos

CLRS 2.1–2.2, 28.2

AU 3.3, 3.6 (muito bom)

Notas de AA de Jeff Edmonds:

`http://www.cs.yorku.ca/~jeff/notes/3101/`

Multiplicação de inteiros gigantescos

$n :=$ número de algarismos.

Problema: Dados dois números inteiros $X[1 \mathinner{.\,.} n]$ e $Y[1 \mathinner{.\,.} n]$ calcular o produto $X \cdot Y$.

Entra: Exemplo com $n = 12$

12 (posição à esquerda) … 1 (posição à direita)

X	9	2	3	4	5	5	4	5	6	2	9	8
Y	0	6	3	2	8	4	9	9	3	8	4	4

Multiplicação de inteiros gigantescos

n := número de algarismos.

Problema: Dados dois números inteiros $X[1\,..\,n]$ e $Y[1\,..\,n]$ calcular o produto $X \cdot Y$.

Entra: Exemplo com $n = 12$

	12											1
X	9	2	3	4	5	5	4	5	6	2	9	8
Y	0	6	3	2	8	4	9	9	3	8	4	4

Sai:

$X \cdot Y$

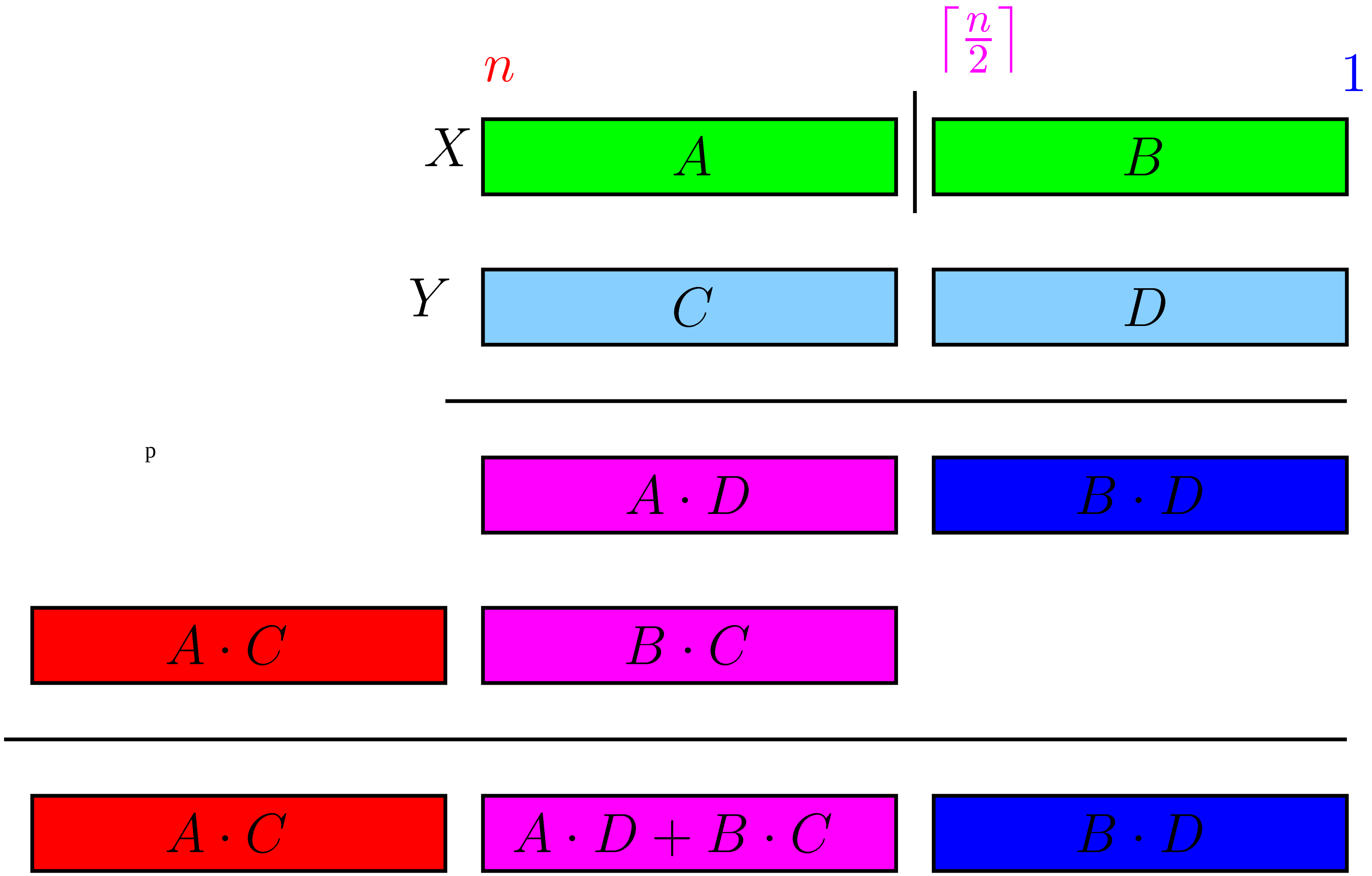
23																						1
5	8	4	4	0	8	7	2	8	6	7	0	2	7	1	4	1	0	2	9	5	1	2

Algoritmo do ensino fundamental

$$
\begin{array}{ccccccccccccccccc}
&&&&&&&&&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast\\
&&&&&&&&\times&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast\\
\cline{9-17}
&&&&&&&&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast\\
&&&&&&&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\\
&&&&&&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&&\\
&&&&&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&&&\\
&&&&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&&&&\\
&&&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&&&&&\\
&&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&&&&&&\\
&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&&&&&&&\\
\hline
\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast&\ast
\end{array}
$$

O algoritmo do ensimo fundamental é $\Theta(n^2)$.

Divisão e conquista



$$X \cdot Y = A \cdot C \times 10^n + (A \cdot D + B \cdot C) \times 10^{\lceil n/2 \rceil} + B \cdot D$$

Exemplo

$$X \quad \begin{array}{cccc} \color{red}{4} & & & \color{blue}{1} \\ \hline \multicolumn{1}{|c|}{3} & \multicolumn{1}{c|}{1} & \multicolumn{1}{c|}{4} & \multicolumn{1}{c|}{1} \\ \hline \end{array} \qquad Y \quad \begin{array}{cccc} \color{red}{4} & & & \color{blue}{1} \\ \hline \multicolumn{1}{|c|}{5} & \multicolumn{1}{c|}{9} & \multicolumn{1}{c|}{3} & \multicolumn{1}{c|}{6} \\ \hline \end{array}$$

Exemplo

$$\begin{array}{ccccc} & 4 & & & 1 \\ X & \boxed{3} & \boxed{1} & \boxed{4} & \boxed{1} \end{array} \qquad \begin{array}{ccccc} & 4 & & & 1 \\ Y & \boxed{5} & \boxed{9} & \boxed{3} & \boxed{6} \end{array}$$

$$A \ \boxed{3}\boxed{1} \qquad B \ \boxed{4}\boxed{1} \qquad C \ \boxed{5}\boxed{9} \qquad D \ \boxed{3}\boxed{6}$$

Exemplo

$$X \;\; \begin{array}{|c|c|c|c|} \multicolumn{2}{c}{4} & \multicolumn{2}{c}{1} \\ \hline 3 & 1 & 4 & 1 \\ \hline \end{array} \qquad Y \;\; \begin{array}{|c|c|c|c|} \multicolumn{2}{c}{4} & \multicolumn{2}{c}{1} \\ \hline 5 & 9 & 3 & 6 \\ \hline \end{array}$$

$$A \; \boxed{3 \mid 1} \qquad B \; \boxed{4 \mid 1} \qquad C \; \boxed{5 \mid 9} \qquad D \; \boxed{3 \mid 6}$$

$$X \cdot Y = A \cdot C \times 10^4 + (A \cdot D + B \cdot C) \times 10^2 + B \cdot D$$

$A \cdot C = 1829$ $\qquad (A \cdot D + B \cdot C) = 1116 + 2419 = 3535$

$B \cdot D = 1476$

$$\begin{array}{lcccccccc} A \cdot C & 1 & 8 & 2 & 9 & 0 & 0 & 0 & 0 \\ (A \cdot D + B \cdot C) & & & 3 & 5 & 3 & 5 & 0 & 0 \\ B \cdot D & & & & & 1 & 4 & 7 & 6 \\ \hline X \cdot Y = & 1 & 8 & 6 & 4 & 4 & 9 & 7 & 6 \end{array}$$

Algoritmo de Multi-DC

Algoritmo recebe inteiros $X[1\,..\,n]$ e $Y[1\,..\,n]$ e devolve $X \cdot Y$.

MULT (X, Y, n)

se $n = 1$ **devolva** $X \cdot Y$
$q \leftarrow \lceil n/2 \rceil$
$A \leftarrow X[q+1\,..\,n]$ $\qquad B \leftarrow X[1\,..\,q]$
$C \leftarrow Y[q+1\,..\,n]$ $\qquad D \leftarrow Y[1\,..\,q]$
$E \leftarrow$ MULT$(A, C, \lfloor n/2 \rfloor)$
$F \leftarrow$ MULT$(B, D, \lceil n/2 \rceil)$
$G \leftarrow$ MULT$(A, D, \lceil n/2 \rceil)$
$H \leftarrow$ MULT$(B, C, \lceil n/2 \rceil)$
$R \leftarrow E \times 10^n + (G + H) \times 10^{\lceil n/2 \rceil} + F$
devolva R

$T(n) =$ consumo de tempo do algoritmo para multiplicar dois inteiros com n algarismos.

Consumo de tempo

linha	todas as execuções da linha	
1	$=$	$\Theta(1)$
2	$=$	$\Theta(1)$
3	$=$	$\Theta(n)$
4	$=$	$\Theta(n)$
5	$=$	$T(\lfloor n/2 \rfloor)$
6	$=$	$T(\lceil n/2 \rceil)$
7	$=$	$T(\lceil n/2 \rceil)$
8	$=$	$T(\lceil n/2 \rceil)$
9	$=$	$\Theta(n)$
10	$=$	$\Theta(n)$
total	$=$	$T(\lfloor n/2 \rfloor) + 3\,T(\lceil n/2 \rceil) + \Theta(4n+2)$

Consumo de tempo

As dicas no nosso estudo de recorrências sugeri que a solução da recorrência

$$
\begin{aligned}
T(1) &= \Theta(1) \\
T(n) &= T(\lfloor n/2 \rfloor) + 3\,T(\lceil n/2 \rceil) + \Theta(n) \quad \text{para } n = 2, 3, 4, \ldots
\end{aligned}
$$

está na mesma classe Θ que a solução de

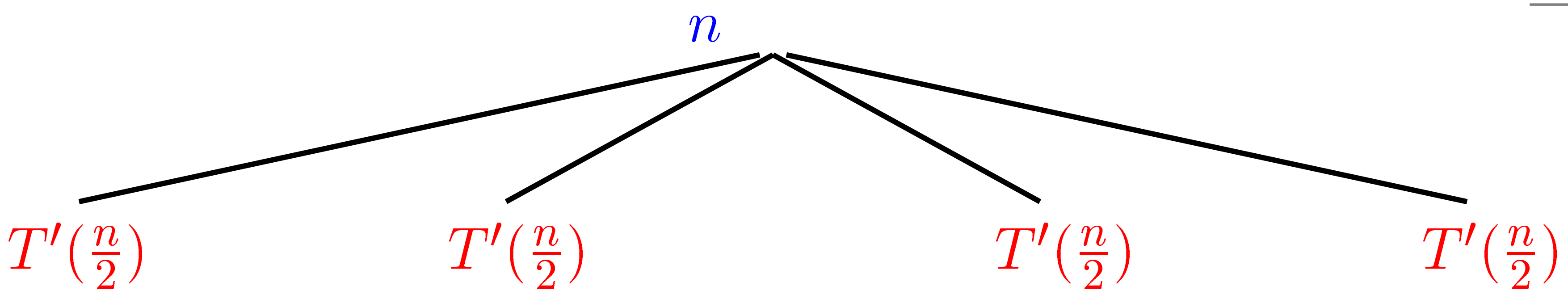
$$
\begin{aligned}
T'(1) &= 1 \\
T'(n) &= 4T'(n/2) + n \quad \text{para } n = 2, 2^2, 2^3, \ldots
\end{aligned}
$$

n	1	2	4	8	16	32	64	128	256	512
$T'(n)$	1	6	28	120	496	2016	8128	32640	130816	523776

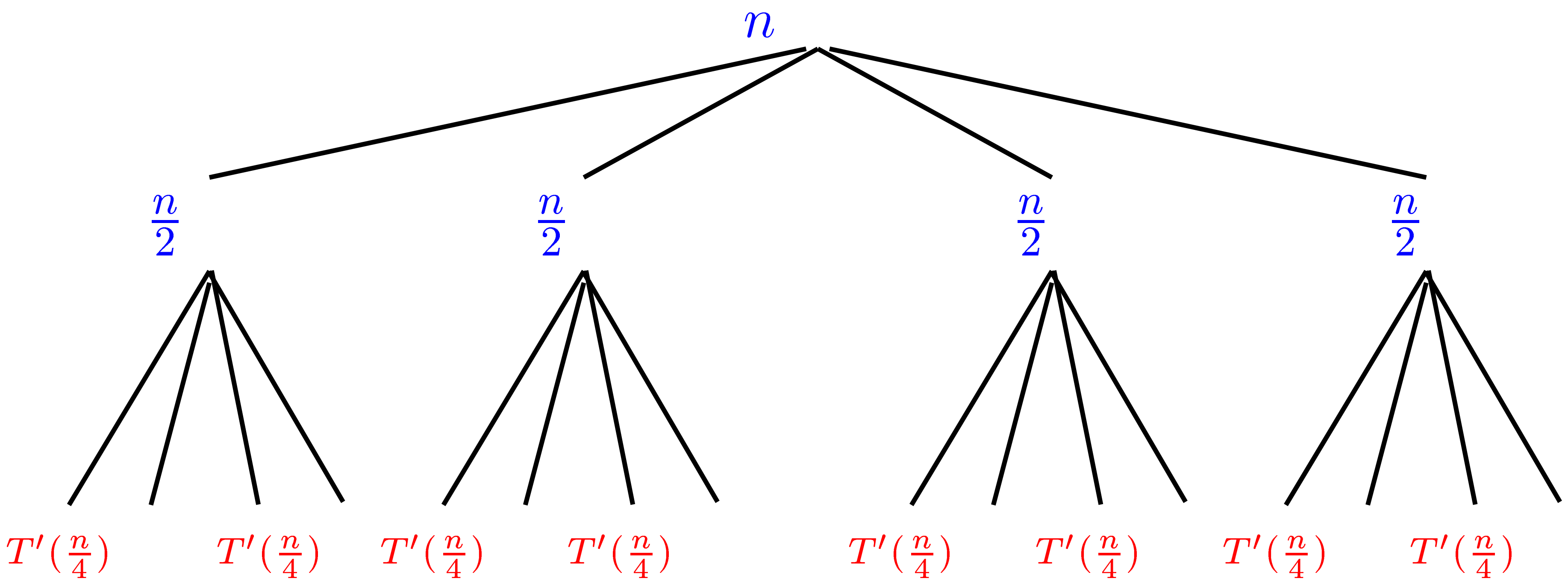
Árvore da recorrência

$$T'(n)$$

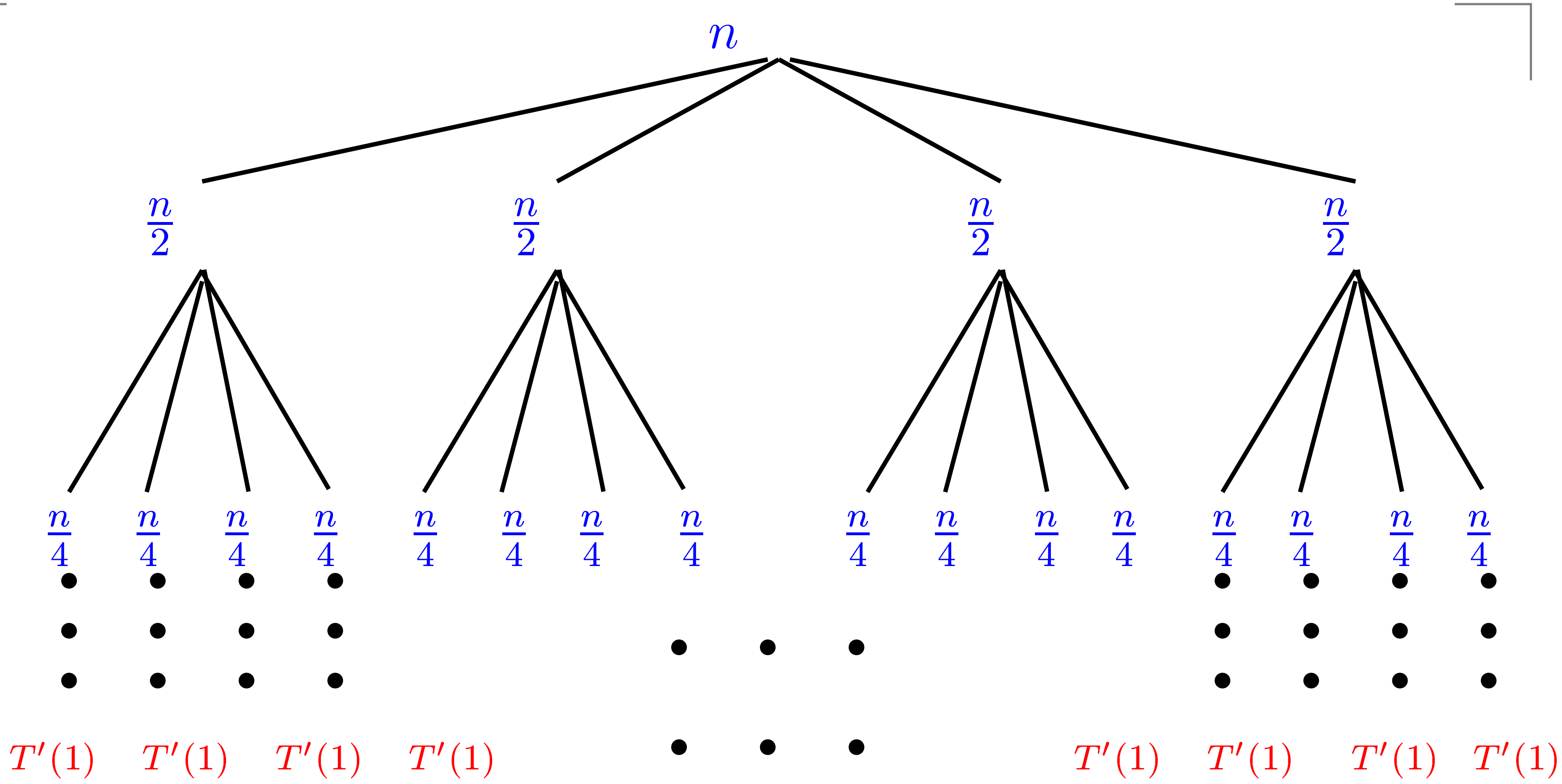
Árvore da recorrência



Árvore da recorrência



Árvore da recorrência



total de $1 + \lg n$ níveis

Contas

nível	0	1	2	$\ldots$	$k-1$	k
soma	n	$2n$	$4n$	$\ldots$	$2^{k-1}n$	$2^k n$

$n = 2^k \qquad\qquad k = \lg n$

Contas

nível	0	1	2	$\ldots$	$k-1$	k
soma	n	$2n$	$4n$	$\ldots$	$2^{k-1}n$	$2^k n$

$n = 2^k \qquad\qquad k = \lg n$

$$\begin{aligned} T'(n) &= n + 2n + \cdots + 2^{k-1}n + 2^k n \\ &= n\,(1 + 2 + 4 + \cdots + 2^k) \\ &= n\,(2^{k+1} - 1) \\ &= n\,(2n - 1) \qquad (k = \lg n) \\ &= 2n^2 - n = \Theta(n^2) \end{aligned}$$

ÊPA, não melhorou ...

Teorema Ban-Ban-Ban

Teorema Mestre Simplificado:

Suponha

$$T(n) = a\,T(n/b) + c\,n^k$$

para algum $a \geq 1$ e $b > 1$ e onde n/b significa $\lceil n/b \rceil$ ou $\lfloor n/b \rfloor$. Então, em geral,

se $a > b^k$ então $T(n) = \Theta(n^{\log_b a})$

se $a = b^k$ então $T(n) = \Theta(n^k \lg n)$

se $a < b^k$ então $T(n) = \Theta(n^k)$

Conclusões

$T'(n)$ é $\Theta(n^2)$.

$T(n)$ é $\Theta(n^2)$.

O consumo de tempo do algoritmo MULT é $\Theta(n^2)$.

Tanto trabalho por nada . . .
Será?!?

Pensar pequeno

Olhar para números com 2 algarismos (n=2).

Suponha $X = a\,b$ e $Y = c\,d$.
Se cada multiplicação custa R$ 1,00 e
cada soma custa R$ 0,01, quanto custa $X \cdot Y$?

Pensar pequeno

Olhar para números com 2 algarismos (n=2).

Suponha $X = a\,b$ e $Y = c\,d$.
Se cada multiplicação custa R$ 1,00 e
cada soma custa R$ 0,01, quanto custa $X \cdot Y$?
Eis $X \cdot Y$ por R$ 4,03:

$$
\begin{array}{lccc}
X & & a & b \\
Y & & c & d \\
\hline
 & & ad & bd \\
 & ac & bc & \\
\hline
X \cdot Y & ac & ad + bc & bd
\end{array}
$$

$$X \cdot Y = ac \times 10^2 + (ad + bc) \times 10^1 + bd$$

Pensar pequeno

Olhar para números com 2 algarismos (n=2).

Suponha $X = a\,b$ e $Y = c\,d$.
Se cada multiplicação custa R$ 1,00 e
cada soma custa R$ 0,01, quanto custa $X \cdot Y$?
Eis $X \cdot Y$ por R$ 4,03:

$$\begin{array}{cccc}
X & & a & b \\
Y & & c & d \\
\hline
 & & ad & bd \\
 & ac & bc & \\
\hline
X \cdot Y & ac & ad+bc & bd
\end{array}$$

$$X \cdot Y = ac \times 10^2 + (ad + bc) \times 10^1 + bd$$

Solução mais barata?

Pensar pequeno

Olhar para números com 2 algarismos (n=2).

Suponha $X = a\,b$ e $Y = c\,d$.
Se cada multiplicação custa R$ 1,00 e
cada soma custa R$ 0,01, quanto custa $X \cdot Y$?
Eis $X \cdot Y$ por R$ 4,03:

$$\begin{array}{lccc}
X & & a & b \\
Y & & c & d \\
\hline
 & & ad & bd \\
 & ac & bc & \\
\hline
X \cdot Y & ac & ad + bc & bd
\end{array}$$

$$X \cdot Y = ac \times 10^2 + (ad + bc) \times 10^1 + bd$$

Solução mais barata?
Gauss faz por R$ 3,06!

$X \cdot Y$ por apenas R$ 3,06

$$
\begin{array}{llll}
X & & a & b \\
Y & & c & d \\
\hline
 & & ad & bd \\
 & ac & bc & \\
\hline
X \cdot Y & ac & ad + bc & bd
\end{array}
$$

$X \cdot Y$ por apenas R$ 3,06

X		a	b
Y		c	d
		ad	bd
	ac	bc	
$X \cdot Y$	ac	$ad + bc$	bd

$$(a+b)(c+d) = ac + ad + bc + bd \Rightarrow$$
$$ad + bc = (a+b)(c+d) - ac - bd$$

$g = (a+b)(c+d) \qquad e = ac \qquad f = bd \qquad h = g - e - f$

$$\boxed{X \cdot Y \ \text{(por R\$ 3,06)} = e \times 10^2 + h \times 10^1 + f}$$

Exemplo

$$X = \; 2133 \quad Y = \; 2312 \qquad\qquad X \cdot Y = \; ?$$

$$ac = \; ? \qquad bd = \; ? \qquad (a+b)(c+d) = \; ?$$

Exemplo

$$X = 2133 \quad Y = 2312 \qquad X \cdot Y = ?$$

$$ac = ? \quad bd = ? \quad (a+b)(c+d) = ?$$

$$X = 21 \quad Y = 23 \qquad X \cdot Y = ?$$

$$ac = ? \quad bd = ? \quad (a+b)(c+d) = ?$$

Exemplo

$$X = 2133 \quad Y = 2312 \qquad X \cdot Y = ?$$
$$ac = ? \qquad bd = ? \qquad (a+b)(c+d) = ?$$

$$X = 21 \quad Y = 23 \qquad X \cdot Y = ?$$
$$ac = ? \qquad bd = ? \qquad (a+b)(c+d) = ?$$

$$X = 2 \quad Y = 2 \quad X \cdot Y = 4$$

Exemplo

$$X = 2133 \quad Y = 2312 \qquad X \cdot Y = \ ?$$
$$ac = \ ? \quad bd = \ ? \qquad (a+b)(c+d) = \ ?$$

$$X = 21 \quad Y = 23 \qquad X \cdot Y = \ ?$$
$$ac = 4 \quad bd = \ ? \qquad (a+b)(c+d) = \ ?$$

Exemplo

$$X = 2133 \quad Y = 2312 \qquad X \cdot Y = ?$$
$$ac = ? \qquad bd = ? \qquad (a+b)(c+d) = ?$$

$$X = 21 \quad Y = 23 \qquad X \cdot Y = ?$$
$$ac = 4 \qquad bd = ? \qquad (a+b)(c+d) = ?$$

$$X = 1 \quad Y = 3 \quad X \cdot Y = 3$$

Exemplo

$$X = 2133 \quad Y = 2312 \qquad X \cdot Y = \ ?$$
$$ac = \ ? \quad bd = \ ? \qquad (a+b)(c+d) = \ ?$$

$$X = 21 \quad Y = 23 \qquad X \cdot Y = \ ?$$
$$ac = 4 \quad bd = 3 \qquad (a+b)(c+d) = \ ?$$

Exemplo

$$X = 2133 \quad Y = 2312 \qquad X \cdot Y = ?$$
$$ac = ? \qquad bd = ? \qquad (a+b)(c+d) = ?$$

$$X = 21 \quad Y = 23 \qquad X \cdot Y = ?$$
$$ac = 4 \qquad bd = 3 \qquad (a+b)(c+d) = ?$$

$$X = 3 \quad Y = 5 \quad X \cdot Y = 15$$

Exemplo

$$\begin{array}{llllll} X = & 2133 & Y = & 2312 & X \cdot Y = & ? \\ ac = & ? & bd = & ? & (a+b)(c+d) = & ? \end{array}$$

$$\begin{array}{llllll} X = & 21 & Y = & 23 & X \cdot Y = & 483 \\ ac = & 4 & bd = & 3 & (a+b)(c+d) = & 15 \end{array}$$

Exemplo

$$X = 2133 \quad Y = 2312 \qquad X \cdot Y = ?$$
$$ac = 483 \quad bd = ? \qquad (a+b)(c+d) = ?$$

Exemplo

$$X = \ 2133 \quad Y = \ 2312 \qquad\qquad X \cdot Y = \ ?$$
$$ac = \ 483 \quad bd = \ ? \qquad (a+b)(c+d) = \ ?$$

$$X = \ 33 \quad Y = \ 12 \qquad\qquad X \cdot Y = \ ?$$
$$ac = \ ? \quad bd = \ ? \qquad (a+b)(c+d) = \ ?$$

Exemplo

$$\begin{array}{llllll} X = & 2133 & Y = & 2312 & X \cdot Y = & ? \\ ac = & 483 & bd = & ? & (a+b)(c+d) = & ? \end{array}$$

$$\begin{array}{llllll} X = & 33 & Y = & 12 & X \cdot Y = & 396 \\ ac = & 3 & bd = & 6 & (a+b)(c+d) = & 27 \end{array}$$

Exemplo

$$\begin{array}{llllll} X = & 2133 & Y = & 2312 & X \cdot Y = & ? \\ ac = & 483 & bd = & 396 & (a+b)(c+d) = & ? \end{array}$$

Exemplo

$$X = 2133 \quad Y = 2312 \qquad X \cdot Y = ?$$
$$ac = 483 \quad bd = 396 \qquad (a+b)(c+d) = ?$$

$$X = 54 \quad Y = 35 \qquad X \cdot Y = ?$$
$$ac = ? \quad bd = ? \qquad (a+b)(c+d) = ?$$

Exemplo

$$\begin{array}{llllll} X = & 2133 & Y = & 2312 & X \cdot Y = & ? \\ ac = & 483 & bd = & 396 & (a+b)(c+d) = & ? \end{array}$$

$$\begin{array}{llllll} X = & 54 & Y = & 35 & X \cdot Y = & 1890 \\ ac = & 15 & bd = & 20 & (a+b)(c+d) = & 72 \end{array}$$

Exemplo

$$\begin{array}{llllll} X = & 2133 & Y = & 2312 & X \cdot Y = & ? \\ ac = & 483 & bd = & 396 & (a+b)(c+d) = & 1890 \end{array}$$

Exemplo

$$X = 2133 \quad Y = 2312 \qquad X \cdot Y = 4931496$$
$$ac = 483 \quad bd = 396 \qquad (a+b)(c+d) = 1890$$

Algoritmo Multi

Algoritmo recebe inteiros $X[1\,..\,n]$ e $Y[1\,..\,n]$ e devolve $X \cdot Y$ (Karatsuba e Ofman).

KARATSUBA (X, Y, n)
- **se** $n \leq 3$ **devolva** $X \cdot Y$
- $q \leftarrow \lceil n/2 \rceil$
- $A \leftarrow X[q+1\,..\,n]$ $\quad B \leftarrow X[1\,..\,q]$
- $C \leftarrow Y[q+1\,..\,n]$ $\quad D \leftarrow Y[1\,..\,q]$
- $E \leftarrow$ KARATSUBA$(A, C, \lfloor n/2 \rfloor)$
- $F \leftarrow$ KARATSUBA$(B, D, \lceil n/2 \rceil)$
- $G \leftarrow$ KARATSUBA$(A+B, C+D, \lceil n/2 \rceil + 1)$
- $H \leftarrow G - F - E$
- $R \leftarrow E \times 10^n + H \times 10^{\lceil n/2 \rceil} + F$
- **devolva** R

$T(n) =$ consumo de tempo do algoritmo para multiplicar dois inteiros com n algarismos.

Consumo de tempo

linha	todas as execuções da linha
1	$= \Theta(1)$
2	$= \Theta(1)$
3	$= \Theta(n)$
4	$= \Theta(n)$
5	$= T(\lfloor n/2 \rfloor)$
6	$= T(\lceil n/2 \rceil)$
7	$= T(\lceil n/2 \rceil + 1)$
8	$= \Theta(n)$
9	$= \Theta(n)$
10	$= \Theta(n)$
total	$= T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + T(\lceil n/2 \rceil + 1) + \Theta(5n + 2)$

Consumo de tempo

As dicas no nosso estudo de recorrências sugerem que a solução da recorrência

$$
\begin{aligned}
T(n) &= \Theta(1) \quad \text{para } n = 1, 2, 3 \\
T(n) &= T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + T(\lceil n/2 \rceil + 1) + \Theta(n) \quad n \geq 4
\end{aligned}
$$

está na mesma classe Θ que a solução de

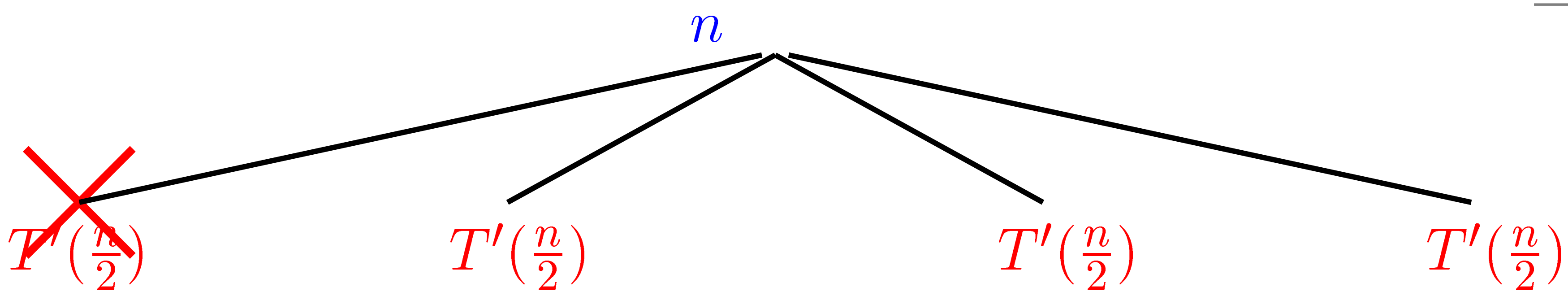
$$
\begin{aligned}
T'(1) &= 1 \\
T'(n) &= 3T'(n/2) + n \quad \text{para } n = 2, 2^2, 2^3, \ldots
\end{aligned}
$$

n	1	2	4	8	16	32	64	128	256	512
$T'(n)$	1	5	19	65	211	665	2059	6305	19171	58025

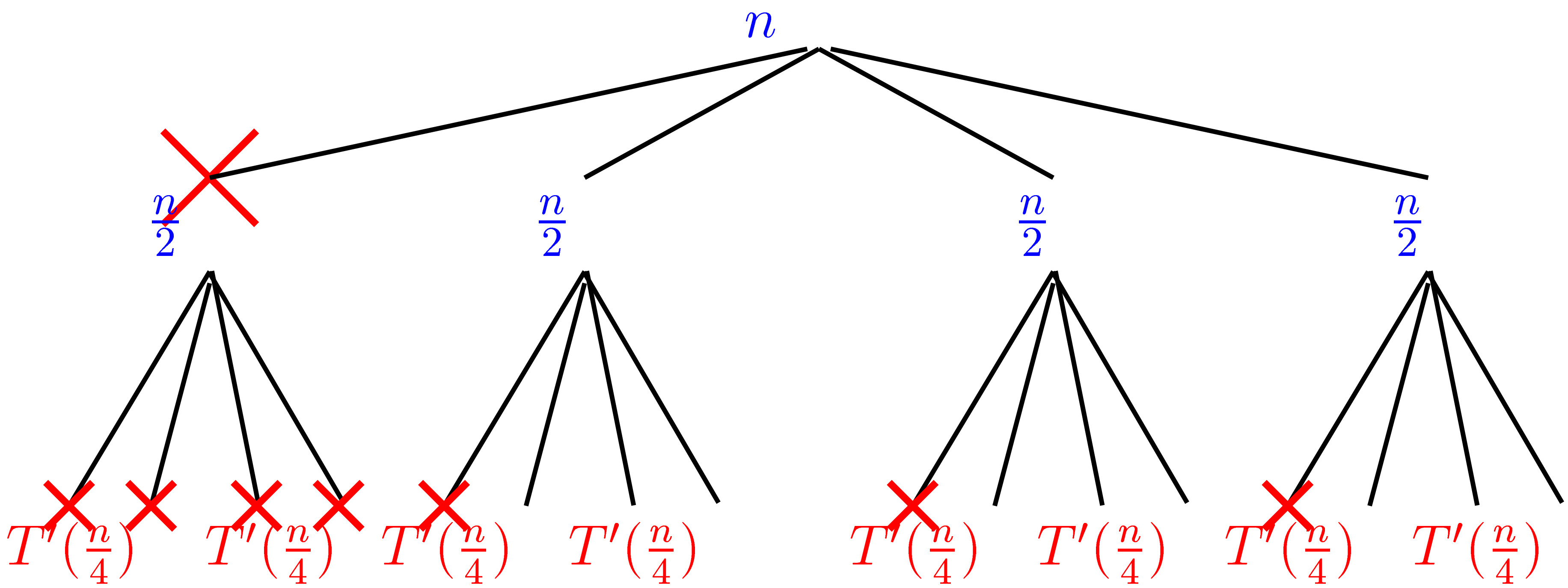
Árvore da recorrência

$T'(n)$

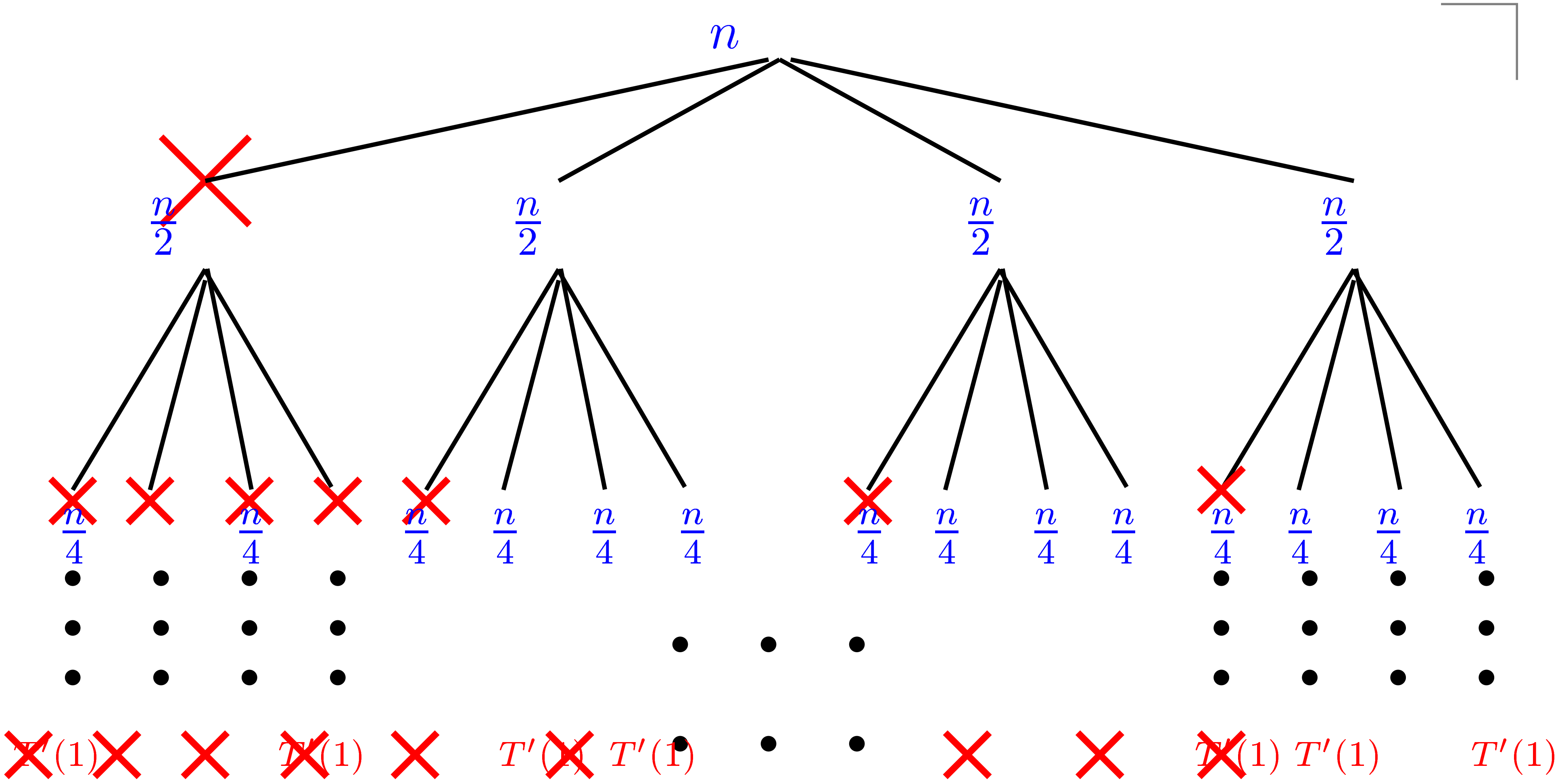
Árvore da recorrência



Árvore da recorrência



Árvore da recorrência



total de $1 + \lg n$ níveis

Contas

nível	0	1	2	...	$k-1$	k
soma	n	$(3/2)n$	$(3/2)^2 n$	...	$(3/2)^{k-1} n$	$(3/2)^k n$

$n = 2^k$ $\qquad$ $k = \lg n$

Contas

nível	0	1	2	...	$k-1$	k
soma	n	$(3/2)n$	$(3/2)^2n$	...	$(3/2)^{k-1}n$	$(3/2)^k n$

$n = 2^k \qquad k = \lg n$

$$
\begin{aligned}
T'(n) &= n + (3/2)n + \cdots + (3/2)^{k-1}n + (3/2)^k n \\
&= n\,(1 + (3/2) + (3/2)^2 + \cdots + (3/2)^k) \\
&= n\,(3(3/2)^k - 2) \\
&= n\,(3\frac{3^{\lg n}}{2^{\lg n}} - 2) \qquad (\text{pois}, k = \lg n) \\
&= 3 \cdot 3^{\lg n} - 2n = 3n^{\lg 3} - 2n = \Theta(n^{\lg 3})
\end{aligned}
$$

$1{,}584 < \lg 3 < 1.585$

LEGAL, estamos de volta ao jogo!

Recorrência

Considere a recorrência

$$\begin{aligned} R(1) &= 1 \\ R(2) &= 1 \\ R(3) &= 1 \\ R(n) &= 3R(\lceil \tfrac{n}{2} \rceil + 1) + n \quad \text{para } n = 4, 5, 6 \ldots \end{aligned}$$

n	1	2	3	4	5	6	7	8	9	10
$T(n)$	1	1	1	7	14	15	29	36	45	53
$R(n)$	1	1	1	7	26	27	85	86	90	91

Recorrência

Considere a recorrência

$$\begin{aligned} R(1) &= 1 \\ R(2) &= 1 \\ R(3) &= 1 \\ R(n) &= 3R(\lceil \tfrac{n}{2} \rceil + 1) + n \quad \text{para } n = 4, 5, 6 \ldots \end{aligned}$$

n	1	2	3	4	5	6	7	8	9	10
$T(n)$	1	1	1	7	14	15	29	36	45	53
$R(n)$	1	1	1	7	26	27	85	86	90	91

Vamos mostra que $R(n)$ é $\mathrm{O}(n^{\lg 3})$.
Isto implica que $T(n)$ é $\mathrm{O}(n^{\lg 3})$.

Solução assintótica da recorrência

Vou mostrar que $R(n) \leq 31\,(n-3)^{\lg 3} - 6n$ para $n = 4, 5, 6, \ldots$

n	1	2	3	4	5	6	7	8	9	10
$R(n)$	1	1	1	7	26	27	85	86	90	91
$31(n-3)^{\lg 3} - 6n$	$*$	$*$	$*$	7	63	119	237	324	473	5910

Solução assintótica da recorrência

Vou mostrar que $R(n) \leq 31\,(n-3)^{\lg 3} - 6n$ para $n = 4, 5, 6, \ldots$

n	1	2	3	4	5	6	7	8	9	10
$R(n)$	1	1	1	7	26	27	85	86	90	91
$31(n-3)^{\lg 3} - 6n$	$*$	$*$	$*$	7	63	119	237	324	473	5910

Prova:

Se $n = 4$, então $R(n) = 7 = 31(n-3)^{\lg 3} - 6n$.

Solução assintótica da recorrência

Prova: (continuação) Se $n \geq 5$ vale que

$$
\begin{aligned}
R(n) &= 3R(\lceil n/2 \rceil + 1) + n \\
&\overset{\text{hi}}{\leq} 3(31(\lceil n/2 \rceil + 1 - 3)^{\lg 3} - 6(\lceil n/2 \rceil + 1)) + n \\
&\leq 3(31(\frac{(n+1)}{2} - 2)^{\lg 3} - 6(\frac{n}{2} + 1)) + n \\
&= 3(31(\frac{(n-3)}{2})^{\lg 3} - 3n - 6) + n \\
&= 3(31\frac{(n-3)^{\lg 3}}{2^{\lg 3}} - 3n - 6) + n \\
&= 3 \cdot 31\frac{(n-3)^{\lg 3}}{3} - 9n - 18 + n \\
&= 31(n-3)^{\lg 3} - 6n - 2n - 18 \\
&< 31(n-3)^{\lg 3} - 6n = \Theta(n^{\lg 3})
\end{aligned}
$$

Teorema Ban-Ban-Ban

Teorema Mestre Simplificado:

Suponha

$$T(n) = a\,T(n/b) + c\,n^k$$

para algum $a \geq 1$ e $b > 1$ e onde n/b significa $\lceil n/b \rceil$ ou $\lfloor n/b \rfloor$.
Então, em geral,

se $a > b^k$ então $T(n) = \Theta(n^{\log_b a})$
se $a = b^k$ então $T(n) = \Theta(n^k \lg n)$
se $a < b^k$ então $T(n) = \Theta(n^k)$

Conclusões

$R(n)$ é $\Theta(n^{\lg 3})$.

Conclusão anterior$+$Exercício 9.B $\Rightarrow$
$T(n)$ é $\Theta(n^{\lg 3})$.

O consumo de tempo do algoritmo KARATSUBA é
$\Theta(n^{\lg 3})$ $(1{,}584 < \lg 3 < 1{,}585)$.

Mais conclusões

Consumo de tempo de algoritmos para multiplicação de inteiros:

Jardim de infância	$\Theta(n\,10^n)$
Ensino fundamental	$\Theta(n^2)$
Karatsuba e Ofman	$\Theta(n^{1.585})$
Schönhage e Strassen	$\Theta(n \lg n \lg \lg n)$

Ambiente experimental

A plataforma utilizada nos experimentos é um PC rodando Linux Debian ?.? com um processador Pentium II de 233 MHz e 128MB de memória RAM .

Os códigos estão compilados com o gcc versão 2.7.2.1 e opção de compilação -O2.

As implementações comparadas neste experimento são as do algoritmo do ensino fundamental e do algoritmo KARATSUBA.

O programa foi escrito por Carl Burch:

`http://www-2.cs.cmu.edu/~cburch/251/karat/.`

Resultados experimentais

n	Ensino Fund.	KARATSUBA
4	0.005662	0.005815
8	0.010141	0.010600
16	0.020406	0.023643
32	0.051744	0.060335
64	0.155788	0.165563
128	0.532198	0.470810
256	1.941748	1.369863
512	7.352941	4.032258

Tempos em 10^3 segundos.

Multiplicação de matrizes

Problema: Dadas duas matrizes $X[1\,..\,n, 1\,..\,n]$ e $Y[1\,..\,n, 1\,..\,n]$ calcular o produto $X \cdot Y$.

Os algoritmo tradicional de multiplicação de matrizes consome tempo $\Theta(n^3)$.

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \times \begin{pmatrix} e & f \\ g & h \end{pmatrix} = \begin{pmatrix} r & s \\ t & u \end{pmatrix}$$

$$\begin{aligned} r &= ae + bg \\ s &= af + bh \\ t &= ce + dg \\ u &= cf + dh \end{aligned} \tag{-3}$$

Solução custa R$ 8,04

Divisão e conquista

$$\begin{array}{ccccc} X & & Y & & \\ \begin{bmatrix} A & B \\ C & D \end{bmatrix} & \text{x} & \begin{bmatrix} E & F \\ G & H \end{bmatrix} & = & \begin{bmatrix} R & S \\ T & U \end{bmatrix} \end{array}$$

$$R = AE + BG$$
$$S = AF + BH$$
$$T = CE + DG$$
$$U = CF + DH$$

(-3)

Algoritmo de Multi-Mat

Algoritmo recebe inteiros $X[1\,..\,n]$ e $Y[1\,..\,n]$ e devolve $X \cdot Y$.

MULTI-M (X, Y, n)
se $n = 1$ **devolva** $X \cdot Y$
$(A, B, C, D) \leftarrow$ PARTICIONE(X, n)
$(E, F, G, H) \leftarrow$ PARTICIONE(Y, n)
$R \leftarrow$ MULTI-M$(A, E, n/2)$ $+$ MULTI-M$(B, G, n/2)$
$S \leftarrow$ MULTI-M$(A, F, n/2)$ $+$ MULTI-M$(B, H, n/2)$
$T \leftarrow$ MULTI-M$(C, E, n/2)$ $+$ MULTI-M$(D, G, n/2)$
$U \leftarrow$ MULTI-M$(C, F, n/2)$ $+$ MULTI-M$(D, H, n/2)$
$P \leftarrow$ CONSTRÓI-MAT(R, S, T, U)
devolva P

$T(n) =$ consumo de tempo do algoritmo para multiplicar duas matrizes de n linhas e n colunas.

Consumo de tempo

linha	todas as execuções da linha
1	$= \Theta(1)$
2	$= \Theta(n^2)$
3	$= \Theta(n^2)$
4	$= T(n/2) + T(n/2)$
5	$= T(n/2) + T(n/2)$
6	$= T(n/2) + T(n/2)$
7	$= T(n/2) + T(n/2)$
8	$= \Theta(n^2)$
9	$= \Theta(n^2)$

total $= 8\,T(n/2) + \Theta(4n^2 + 1)$

Consumo de tempo

As dicas no nosso estudo de recorrências sugeri que a solução da recorrência

$$
\begin{aligned}
T(1) &= \Theta(1) \\
T(n) &= 8\,T(n/2) + \Theta(n^2) \quad \text{para } n = 2, 3, 4, \ldots
\end{aligned}
$$

está na mesma classe Θ que a solução de

$$
\begin{aligned}
T'(1) &= 1 \\
T'(n) &= 8T'(n/2) + n^2 \quad \text{para } n = 2, 2^2, 2^3, \ldots
\end{aligned}
$$

n	1	2	4	8	16	32	64	128	256
$T'(n)$	1	12	112	960	7936	64512	520192	4177920	33488896

Solução assintótica da recorrência

Considere a recorrência

$$\begin{aligned} R(1) &= 1 \\ R(n) &= 8\,R(\lceil \tfrac{n}{2} \rceil) + n^2 \quad \text{para } n = 2, 3, 4, \ldots \end{aligned}$$

Verifique por indução que $R(n) \leq 20(n-1)^3 - 2n^2$ para $n = 2, 3, 4 \ldots$.

n	1	2	3	4	5	6	7	8
$R(n)$	1	12	105	112	865	876	945	960
$20\,(n-1)^3 - 2n^2$	-2	12	142	508	1230	2428	4222	6732

Conclusões

$R(n)$ é $\Theta(n^3)$.

Conclusão anterior$+$Exercício $\Rightarrow$
$T(n)$ é $\Theta(n^3)$.

O consumo de tempo do algoritmo MULTI-M é $\Theta(n^3)$.

Strassen: $X \cdot Y$ por apenas R\$ 7,18

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \times \begin{pmatrix} e & f \\ g & h \end{pmatrix} = \begin{pmatrix} r & s \\ t & u \end{pmatrix}$$

Strassen: $X \cdot Y$ por apenas R$ 7,18

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \times \begin{pmatrix} e & f \\ g & h \end{pmatrix} = \begin{pmatrix} r & s \\ t & u \end{pmatrix}$$

$$p_1 = a(f - h) = af - ah$$

$$p_2 = (a + b)h = ah + bh$$

$$p_3 = (c + d)e = ce + de$$

$$p_4 = d(g - e) = dg + de$$

$$p_5 = (a + d)(e + h) = ae + ah + de + dh$$

$$p_6 = (b - d)(g + h) = bg + bh - dg - dh$$

$$p_7 = (a - c)(e + f) = ae + af - ce - cfd$$

(-2)

Strassen: $X \cdot Y$ por apenas R\$ 7,18

$$p_1 = a(f - h) = af - ah$$

$$p_2 = (a + b)h = ah + bh$$

$$p_3 = (c + d)e = ce + de$$

$$p_4 = d(g - e) = dg + de$$

$$p_5 = (a + d)(e + h) = ae + ah + de + dh$$

$$p_6 = (b - d)(g + h) = bg + bh - dg - dh$$

$$p_7 = (a - c)(e + f) = ae + af - ce - cfd$$

$$r = p_5 + p_4 - p_2 + p_6 = ae + bg$$

$$s = p_1 + p_2 = af + bh$$

$$t = p_3 + p_4 = ce + dg$$

$$u = p_5 + p_1 - p_3 - p_7 = cf + dh$$

Algoritmo de Strassen

STRASSEN (X, Y, n)

se $n = 1$ **devolva** $X \cdot Y$
$(A, B, C, D) \leftarrow$ PARTICIONE(X, n)
$(E, F, G, H) \leftarrow$ PARTICIONE(Y, n)
$P_1 \leftarrow$ STRASSEN$(A, F - H, n/2)$
$P_2 \leftarrow$ STRASSEN$(A + B, H, n/2)$
$P_3 \leftarrow$ STRASSEN$(C + D, E, n/2)$
$P_4 \leftarrow$ STRASSEN$(D, G - E, n/2)$
$P_5 \leftarrow$ STRASSEN$(A + D, E + H, n/2)$
$P_6 \leftarrow$ STRASSEN$(B - D, G + H, n/2)$
$P_7 \leftarrow$ STRASSEN$(A - C, E + F, n/2)$
$R \leftarrow P_5 + P_4 - P_2 + P_6$
$S \leftarrow P_1 + P_2$
$T \leftarrow P_3 + P_4$
$U \leftarrow P_5 + P_1 - P_3 - P_7$
devolva $P \leftarrow$ CONSTRÓI-MAT(R, S, T, U)

Consumo de tempo

linha		todas as execuções da linha
1	$=$	$\Theta(1)$
2-3	$=$	$\Theta(n^2)$
4-10	$=$	$7, T(n/2) + \Theta(n^2)$
11-14	$=$	$\Theta(n^2)$
15	$=$	$\Theta(n^2)$
total	$=$	$7\,T(n/2) + \Theta(4n^2 + 1)$

Consumo de tempo

As dicas no nosso estudo de recorrências sugeri que a solução da recorrência

$$\begin{aligned} T(1) &= \Theta(1) \\ T(n) &= 7\,T(n/2) + \Theta(n^2) \quad \text{para } n = 2, 3, 4, \ldots \end{aligned}$$

está na mesma classe Θ que a solução de

$$\begin{aligned} T'(1) &= 1 \\ T'(n) &= 7T'(n/2) + n^2 \quad \text{para } n = 2, 2^2, 2^3, \ldots \end{aligned}$$

n	1	2	4	8	16	32	64	128	256
$T'(n)$	1	11	93	715	5261	37851	269053	1899755	13363821

Solução assintótica da recorrência

Considere a recorrência

$$\begin{aligned} R(1) &= 1 \\ R(n) &= 7\,R(\lceil \tfrac{n}{2} \rceil) + n^2 \quad \text{para } n = 2, 3, 4, \ldots \end{aligned}$$

Verifique por indução que $R(n) \le 19(n-1)^{\lg 7} - 2n^2$ para $n = 2, 3, 4 \ldots$.

$2{,}80 < \lg 7 < 2{,}81$

n	1	2	3	4	5	6	7	8
$R(n)$	1	11	86	93	627	638	700	715
$19\,(n-1)^{\lg 7} - 2n^2$	-1	11	115	327	881	1657	2790	4337

Teorema Ban-Ban-Ban

Teorema Mestre Simplificado:

Suponha

$$T(n) = a\,T(n/b) + c\,n^k$$

para algum $a \geq 1$ e $b > 1$ e onde n/b significa $\lceil n/b \rceil$ ou $\lfloor n/b \rfloor$. Então, em geral,

se $a > b^k$ então $T(n) = \Theta(n^{\log_b a})$
se $a = b^k$ então $T(n) = \Theta(n^k \lg n)$
se $a < b^k$ então $T(n) = \Theta(n^k)$

Conclusões

$R(n)$ é $\Theta(n^{\lg 7})$.

$T(n)$ é $\Theta(n^{\lg 7})$.

O consumo de tempo do algoritmo STRASSEN é $\Theta(n^{\lg 7})$ $(2{,}80 < \lg 7 < 2{,}81)$.

Mais conclusões

Consumo de tempo de algoritmos para multiplicação de matrizes:

Ensino fundamental	$\Theta(n^3)$
Strassen	$\Theta(n^{2.81})$
...	...
Coppersmith e Winograd	$\Theta(n^{2.38})$

Exercícios

Exercício 11.A

O algoritmo abaixo promete rearranjar $A[p\,..\,r]$, com $p \leq r$, em ordem crescente.

ORDENA-POR-INS (A, p, r)

```
se p < r
    então ORDENA-POR-INS (A, p, r − 1)
        chave ← A[r]
        i ← r − 1
        enquanto i ≥ p e A[i] > chave faça
            A[i + 1] ← A[i]
            i ← i − 1
        A[i + 1] ← chave
```

O algoritmo está correto? Escreva uma recorrência que expresse o consumo de tempo. Qual o consumo de tempo do algoritmo?

Exercício 11.B

Que acontece se trocarmos $\lfloor (p + r)/2 \rfloor$ por $\lceil (p + r)/2 \rceil$ na linha 2 do MERGE-SORT?

Mais exercicios

Exercício 11.C

Quantas vezes a comparação "$A[r] \neq 0$" é executada? Defina esse número por meio de um recorrência.

```
LIMPA (A, p, r)
  se p = r
      então devolva r
      senão q ← LIMPA (A, p, r − 1)
            se A[r] ≠ 0
                então q ← q + 1
                      A[q] ← A[r]
            devolva q
```

Dê uma fórmula exata para a função definida pela recorrência. Em que classe Θ está a função definida pela recorrência?