Universidade Federal de Ouro Preto
Departamento de Computação

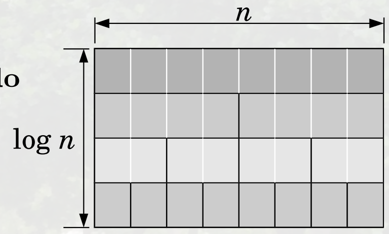
Projeto e Análise de Algoritmos II - Recursões

decom
departamento de computação

Prof. Haroldo Gambini Santos

Algoritmos Recursivos – Ex. 1

```
procedure B(n)
   for i := 1 to n do
      x := x + i
   call B(n/2)
   call B(n/2)
end
```



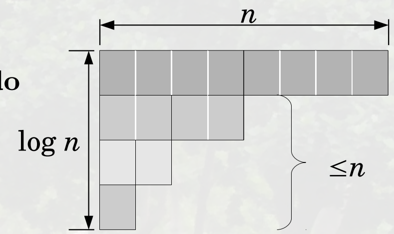
$O(n \log n)$

Tempo de Resolução $T(n)$:
$T(n)=2T(n/2)+n$
(relação de recorrência)

2

Algoritmos Recursivos – Ex. 2

```
procedure B(n)
   for i := 1 to n do
      x := x + i
   call B(n/2)
end
```



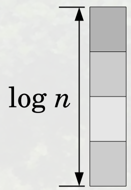
$O(n)$

Tempo de Resolução $T(n)$:
$T(n)=T(n/2)+n$

3

Algoritmos Recursivos – Ex. 3

```
procedure B(n)
   x := x + 1
   call B(n/2)
end
```



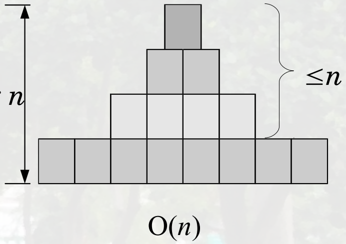
$O(\log n)$

Tempo de Resolução $T(n)$:
$T(n)=T(n/2)+1$

4

Algoritmos Recursivos – Ex. 4

```
procedure B(n)
   x := x + 1
   call B(n/2)
   call B(n/2)
end
```



Tempo de Resolução $T(n)$:

$T(n)=2T(n/2)+1$

Algoritmos Recursivos – Torres de Hanoi

Algoritmos Recursivos – Torres de Hanoi

```
procedure MoveTower(disk, source, dest, temp)
if disk = 0 then
    move disk from source to dest
else
begin
    MoveTower(disk - 1, source, temp, dest)
    move disk from source to dest
    MoveTower(disk - 1, temp, dest, source)
end
```

$T(n)=2T(n\text{-}1)+1$

Algoritmos Recursivos – Torres de Hanoi

$T(n)=2T(n\text{-}1)+1$

$T(1)=1$
$T(2)=3$
$T(3)=7$
$T(4)=15$
$T(5)=31$
$T(6)=63$

$T(n)=2^n\text{-}1$?

Algoritmos Recursivos – Torres de Hanoi

- $T(n)=2T(n\text{-}1)+1 \quad = \quad T(n)=2^n\text{-}1$?
- Checando:
  - $T(0) = 2^0\text{-}1$
  - $T(n) = 2T(n\text{-}1)+1$
  - $T(n) = 2(2^{n\text{-}1}\text{-}1)+1$
  - $T(n) = 2^n\text{-}1$ ✓

9

Algoritmos Recursivos – Torres de Hanoi

"Deve-se tomar cuidado com algoritmos recursivos, sua simplicidade pode mascarar sua ineficiência."

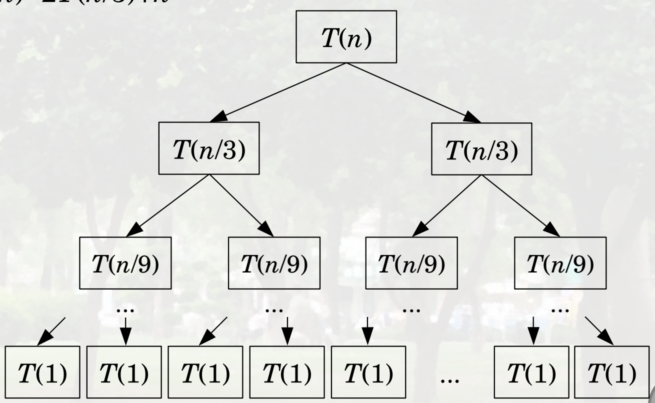
Anany Levitin

Introduction to the Design and Analysis of Algorithms. 2[nd] Edition. 2006.

10

Árvore de Recorrência – Exemplo

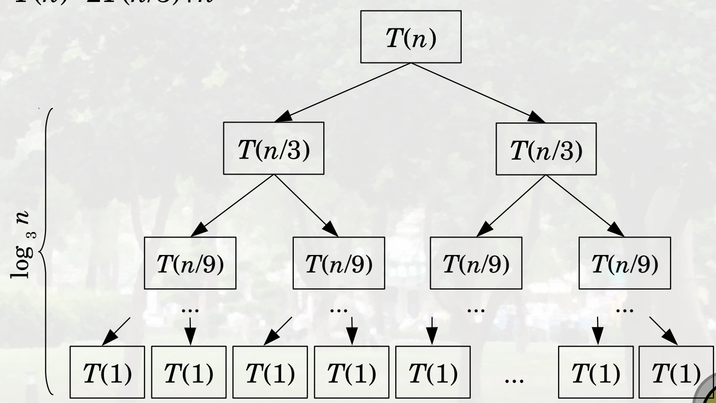
$T(n)=2T(n/3)+n$



11

Árvore de Recorrência – Exemplo

$T(n)=2T(n/3)+n$



12

Árvore de Recorrência – Exemplo

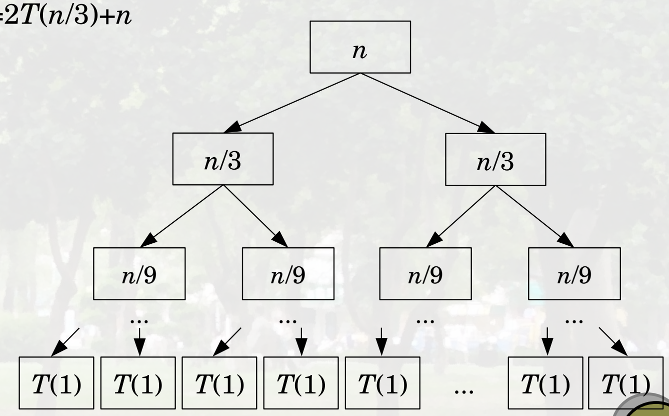
$T(n)=2T(n/3)+n$

n

$\frac{2}{3}n$

$\frac{4}{9}n$

$2^{\log_3 n} = n^{\log_3 2}$



13

Árvore de Recorrência – Exemplo

$T(n)=2T(n/3)+n$

$$
\begin{aligned}
T(n) &= n && \} \textit{ trabalho nó raiz}\\
&+\frac{2}{3}n+\frac{2^2}{3}n+\frac{2^3}{3}n+\ldots+\frac{2^{\log_3 n-1}}{3}n && \} \textit{ nós intermediários}\\
&+n^{\log_3 2} && \} \textit{ nós folha}
\end{aligned}
$$

14

Árvore de Recorrência – Exemplo

Séries geométricas:

para $|x|<1$, limite ∞

$$\sum_{k=0}^{\infty} x^k = \frac{1}{1-x}$$

para $x\neq 1$, limite n

$$\sum_{k=0}^{n} x^k = \frac{x^{n+1}-1}{x-1}$$

15

Árvore de Recorrência – Exemplo

$T(n)=2T(n/3)+n$

$$
\begin{aligned}
T(n) &= n\\
&+\frac{2}{3}n+\frac{2^2}{3}n+\frac{2^3}{3}n+\ldots+\frac{2^{\log_3 n-1}}{3}n \quad (\leqslant 3n)\\
&+n^{0,63}
\end{aligned}
$$

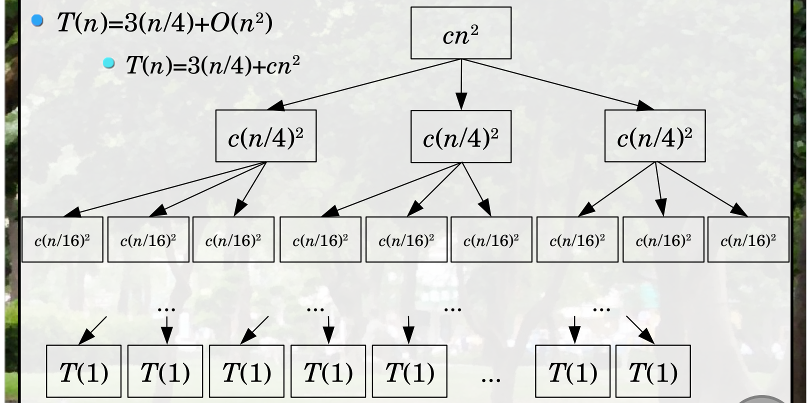
$$T(n) \leqslant 3n+n^{0,63}$$

$$T(n) = O(n)$$

16

Árvore de Recorrência – Exemplo 2

- $T(n)$=3$(n/4)$+$O(n^2)$
  - $T(n)$=3$(n/4)$+cn^2



Árvore de Recorrência – Exemplo 2

- $T(n)$=3$(n/4)$+$O(n^2)$
  - Nível 0 até o penúltimo:

$$\sum_{i=0}^{\log_4 n-1}\left(\frac{3}{16}\right)^i cn^2 \leqslant \left(\frac{1}{1-\frac{3}{16}}\right)cn^2=\frac{16}{13}cn^2$$

  - Último nível:

$$3^{\log_4 n}=n^{\log_4 3}=n^{0,79}$$

$T(n)$=$O(n^2)$

Exercícios

1) Use a árvore de recursão para determinar um bom limite superior assintótico na recorrência $T(n)$=3$T(n/2)$+n.
2) Demonstre que a solução para a recorrência $T(n)$=$T(n/3)$+$T(2n/3)$+cn, onde c é uma constante, é $\Omega(n \log n)$, utilizando árvore de recursão.
3) Trace a árvore de recursão para $T(n)$= 4T$(n/2)$ + cn, onde c é uma constante, e forneça um limite assintótico restrito sobre sua solução. Verifique o limite pelo método de substituição.
4) Use uma árvore de recursão com o objetivo de fornecer uma solução assintoticamente restrita para a recorrência $T(n)$=$T(n-a)$ + $T(a)$ + cn, onde $a\geq 1$ e c>0 são constantes.

Exercícios

5) Use uma árvore de recursão para fornecer uma solução assintoticamente restrita para a recorrência $T(n)$=$T(\alpha n)$ + $T((1-\alpha)n)$ + cn, onde α é uma constante no intervalo O<α<1 e c>0 também é uma constante.

O Teorema Mestre

$$T(n) = aT(n/b) + n^d$$

- a: número de subproblemas (>1)
- n/b: tamanho de cada subproblema (b>1)
- n^d: custo de dividir o problema e combinar resultados (d≥0)

$$T(n) = \begin{cases} O(n^d) & \text{se } d > \log_b a \\ O(n^d \log n) & \text{se } d = \log_b a \\ O(n^{\log_b a}) & \text{se } d < \log_b a \end{cases}$$

21