Projeto e Análise de Algoritmos
2024.2

Complexidade de Algoritmos Recursivos

Prof. Marcio Costa Santos DCC/UFMG

Resolução de Recorrências

- Podemos resolver equações na forma

$$T(n) = aT(\frac{n}{b}) + f(n)$$

- Podemos resolver equações na forma

$$T(n) = aT(n - b) + f(n)$$

- Não podemos resolver equações na forma

$$T(n) = T(n - a) + T(n - b)$$

Exemplo 4

$$F(n) = \begin{cases} F(\frac{n}{2}) + 1 & \text{, se } n > 1 \\ 1 & \text{, em caso contrário} \end{cases}$$

Complexidade: $\log_2 n$

Exemplo 4

$$
\begin{aligned}
F(n) &= F(\frac{n}{2}) + 1 \\
&= (F(\frac{n}{2^2}) + 1) + 1 \\
&= ((F(\frac{n}{2^3}) + 1) + 1) + 1 \\
&= \ldots
\end{aligned}
$$

Exemplo 4

$$
\begin{aligned}
F(n) = & \quad F(\frac{n}{2^i}) + \sum_{j=1}^{i} 1 \\
= & \quad F(1) + \sum_{j=1}^{\log_2 n - 1} 1 \\
= & \quad 1 + \log_2 n
\end{aligned}
$$

Exemplo 5

$$F(n) = \begin{cases} 2F(\frac{n}{2}) + 1 & \text{, se } n > 1 \\ 1 & \text{, em caso contrário} \end{cases}$$

Complexidade: $\Theta(n)$

Exemplo 5

$$
\begin{aligned}
F(n) &= 2F(\frac{n}{2}) + 1 \\
&= 2(2F(\frac{n}{2^2}) + 1) + 1 \\
&= 2(2(2F(\frac{n}{2^3}) + 1) + 1) + 1 \\
&= \ldots
\end{aligned}
$$

Exemplo 5

$$F(n) = 2^i F(\frac{n}{2^i}) + \sum_{j=0}^{i-1} 2^i$$

$$= 2^{\log_2 n} F(1) + \sum_{j=0}^{\log_2 n - 1} 2^i$$

$$= 2^{\log_2 n} + 2^{\log_2 n} - 1 = 2.2^{\log_2 n} - 1 = 2n - 1$$

Exemplo 6

$$F(n) = \begin{cases} 2F(\frac{n}{3}) + 1 & \text{, se } n > 1 \\ 1 & \text{, em caso contrário} \end{cases}$$

Complexidade: $\sqrt[\log_3 2]{n}$

Exemplo 6

$$
\begin{aligned}
F(n) &= 2F(\frac{n}{3}) + 1 \\
&= 2(2F(\frac{n}{3^2}) + 1) + 1 \\
&= 2(2(2F(\frac{n}{3^3}) + 1) + 1) + 1 \\
&= \ldots
\end{aligned}
$$

- 1

Teorema Mestre

Teorema Mestre

Sejam $a \leq 1$ e $b > 1$ constantes, $f(n)$ uma função e

$$T(n) = aT(\frac{n}{b}) + f(n),$$

Então, para algum $\epsilon > 0$

Se $f(n) = O(n^{\log_b a - \epsilon})$ $T(n) = \Theta(n^{\log_b a})$

Se $f(n) = \Theta(n^{\log_b a})$ $T(n) = \Theta(n^{\log_b a} \log n)$

Se $f(n) = \Omega(n^{\log_b a + \epsilon})$ e $af(\frac{n}{b}) \leq cf(n)$ então
$T(n) = \Theta(f(n))$

Teorema Mestre - Exemplos

- $F(n) = F(\frac{n}{2}) + 1$
- $a = 1$, $b = 2$ e $\log_2 1$
- $\log_2 1 = 0$ então $n^0 = 1$
- $\Theta(n^{\log_b a)} = \theta(1)$ e $f(n) = 1$
- Caso 2! Logo complexidade é $\Theta(n^{\log_b a} \log n) = \Theta(\log n)$

Teorema Mestre - Exemplos

- $F(n) = F(\frac{n}{2}) + n$
- $a = 1$, $b = 2$ e $\log_2 1$
- $\log_2 1 = 0$ então $n^0 = 1$
- $\Theta(n^{\log_b a)} = \Theta(1)$ e $f(n) = n$
- Caso 3! Logo complexidade é $\Theta(n)$

Teorema Mestre - Exemplos

- $F(n) = 2F(\frac{n}{2}) + 1$
- $a = 2$, $b = 2$ e $\log_2 2$
- $\log_2 2 = 1$ então $n^1 = n$
- $\Theta(n^{\log_b a)} = \Theta(n)$ e $f(n) = 1$
- Caso 1! Logo complexidade é $\Theta(n^{\log_b a}) = \Theta(n)$

Teorema Mestre - Exemplos

- $F(n) = 2F(\frac{n}{2}) + n \log n$
- $a = 2,\ b = 2$ e $\log_2 2$
- $\log_2 2 = 1$ então $n^1 = n$
- $\Theta(n^{\log_b a)} = \Theta(n)$ e $f(n) = n \log n$
- Caso 3?
- $n \log n$ não é POLIMONIALMENTE MAIOR QUE n!

Teorema Mestre - Exemplos

- $F(n) = 2F(\sqrt{n}) + n + 1$?
- Substituição de variável $m = \log n$, logo $2^m = n$
- $F(n) = F(2^m) = 2F(\sqrt{2^m}) + 2^m + 1$
- $F(2^m) = 2F(2^{\frac{m}{2}}) + 2^m + 1$
- Defina $G(m) = F(2^m)$
- $G(m) = 2G(\frac{m}{2}) + 2^m + 1$

Teorema Mestre - Exemplos

- $G(m) = 2G(\frac{m}{2}) + 2^m + 1$
- $a = 2$, $b = 2$ e $\log_2 2$
- $\log_2 2 = 1$ então $m^1 = m$
- $\Theta(m^{\log_b a)} = \Theta(m)$ e $f(m) = 2^m + 1$
- Caso 3, logo a complexidade é $\Theta(2^m + 1)$
- $F(n) = G(m) = \Theta(2^m + 1) = \Theta(n + 1) = \Theta(n)$