Recorrências e Soluções de Recorrências I

Pontos da aula

- Recorrências.
- Método da Expansão.
- Método da Árvore de Recursão.
- Método Mestre.

Recorrências e Soluções de Recorrências

- Recorrências.

Recorrências

- Uma função que chama a si mesma é dita recursiva.
- Exemplificando o uso de recursividade, tomemos o problema do cálculo do fatorial de um número.
- O fatorial de um número n, para todo n⩾0, é defiido por:

$$n! = \begin{cases} 1, \text{ se } n = 0 \text{ ou } n = 1 \\ n*(n\text{-}1)!, \text{ se } n > 1. \end{cases}$$

Recorrências

- Exemplo de cálculo recursivo de 4!:
  - 4! = 4 * 3!
  - 3! = 3 * 2!
  - 2! = 2 * 1!
  - 1! = 1
  - *2! = 2 * 1! = 2 * 1 = 2*
  - *3! = 3 * 2! = 3 * 2 = 6*
  - *4! = 4 * 3! = 4 * 6 = 24*

Recorrências

- Implementação recursiva do fatorial:

```
int fatorial(int n){
    int resp;
    if(n == 0 || n == 1)
        resp = 1;
    else
        resp = n * fatorial(n – 1);
    return resp;
}
```

Recorrências

- A análise do tempo de execução de um ***algoritmo recursivo*** é um pouco diferente dos algoritmos que não possuem recursividade.

Recorrências

- Como visto, a definição do fatorial de um número pode ser representada mediante o ***cálculo do fatorial*** de <u>um número menor</u>.
- Tal definição é um exemplo de ***recorrência***.

Recorrências

- Assim:

  “Uma relação de recorrência, ou simplesmente recorrência, é uma equação ou desigualdade que descreve uma função em termos de seu valor em entradas menores.”

Recorrências

- Como a <u>solução do problema de tamanho n</u> depende da solução de problemas de <u>tamanhos menores</u>, o ***tempo de execução*** também é representado mediante o tempo de execução do problema para tamanhos menores.

Recorrências

- No exemplo do fatorial, o tempo de execução é representado pela seguinte expressão de recorrência:

$$T(n) = T(n - 1) + 1$$

Recorrências

- Assim como o problema do fatorial, <u>existem outros inúmeros problemas</u> que podem ser resolvidos mediante uma ***solução recursiva***.

Recorrências

- Quando se ***compara a eficiência de algoritmos***, normalmente existe o interesse em comparar as funções que descrevem as suas complexidades, analisando-as de maneira assintótica e utilizando, para isso, as ***notações O, Ω e Θ***.

Recorrências

- Se uma função T(n) se encontra na forma de uma relação de recorrência, ***não*** é possível compará-la com outras funções conhecidas.

Recorrências e Soluções de Recorrências

- Método da Expansão.

Método da Expansão

- **Exemplo 1:** Considere a seguinte relação de recorrência:

$$T(1) = 1$$
$$T(n) = T\left(\frac{n}{2}\right) + 1$$

Método da Expansão

Passo 1: Expansão da Recorrência

Expandimos a recorrência repetidamente:

$$T(n) = T\left(\frac{n}{2}\right) + 1$$

Substituímos $T\left(\frac{n}{2}\right)$ pela sua própria expansão:

$$T(n) = \left(T\left(\frac{n}{4}\right) + 1\right) + 1 = T\left(\frac{n}{4}\right) + 2$$

Método da Expansão

Expandindo novamente:

$$T(n) = \left(T\left(\frac{n}{8}\right) + 1\right) + 2 = T\left(\frac{n}{8}\right) + 3$$

Generalizando para o passo k:

$$T(n) = T\left(\frac{n}{2^k}\right) + k$$

Método da Expansão

Passo 2: Definir o Caso Base

O caso base ocorre quando $\frac{n}{2^k} = 1$, ou seja, $n = 2^k$, o que implica que:

$$k = \log_2 n$$

No caso base, temos $T(1) = \Theta(1)$.

Método da Expansão

Passo 3: Substituir o Caso Base

Substituindo o valor $k = \log_2 n$ na equação expandida:

$$T(n) = T(1) + \log_2 n$$

Sabemos que $T(1) = 1$, então:

$$T(n) = 1 + \log_2 n$$

Método da Expansão

Passo 4: Expressão Final em Notação Assintótica

Finalmente, podemos expressar a solução da recorrência como:

$$T(n) = O(\log n)$$

Portanto, o tempo de execução do algoritmo cresce de forma logarítmica em relação ao tamanho da entrada n.

Método da Expansão

- **Exemplo 2:** Considere a seguinte relação de recorrência:

$$T(n) = 2T\left(\frac{n}{2}\right) + c$$

Método da Expansão

Passo 1: Expansão da Recorrência

Expandimos a recorrência repetidamente:

$$T(n) = 2T\left(\frac{n}{2}\right) + c$$

Expandindo $T\left(\frac{n}{2}\right)$:

$$T(n) = 2\left(2T\left(\frac{n}{4}\right) + c\right) + c = 2^2T\left(\frac{n}{4}\right) + 3c$$

Método da Expansão

Expandindo novamente:

$$T(n) = 2^3 T\left(\frac{n}{8}\right) + 7c$$

Generalizando para o passo k:

$$T(n) = 2^k T\left(\frac{n}{2^k}\right) + (2^k - 1)c$$

Método da Expansão

Passo 2: Definir o Caso Base

O caso base ocorre quando $\frac{n}{2^k} = 1$, ou seja, $k = \log_2 n$. No caso base, $T(1) = d$, onde d é uma constante.

Método da Expansão

Passo 3: Substituir o Caso Base

Substituindo $k = \log_2 n$ na equação expandida:

$$T(n) = 2^{\log_2 n} T(1) + (2^{\log_2 n} - 1)c$$

Sabemos que $2^{\log_2 n} = n$, logo:

$$T(n) = nT(1) + (n - 1)c$$

Substituindo $T(1) = d$:

$$T(n) = n \cdot d + (n - 1)c$$

Método da Expansão

Passo 4: Expressão Final em Notação Assintótica

Finalmente, podemos expressar a solução da recorrência como:

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

Portanto, a complexidade do algoritmo é linear.

Recorrências e Soluções de Recorrências

- Método da Árvore de Recursão.

Método da Árvore de Recursão

- Este método consiste em desenhar uma árvore cujos nós representam os tamanhos dos problemas correspondentes.
- Cada nível i contém todos os subproblemas de profundidade i.

Método da Árvore de Recursão

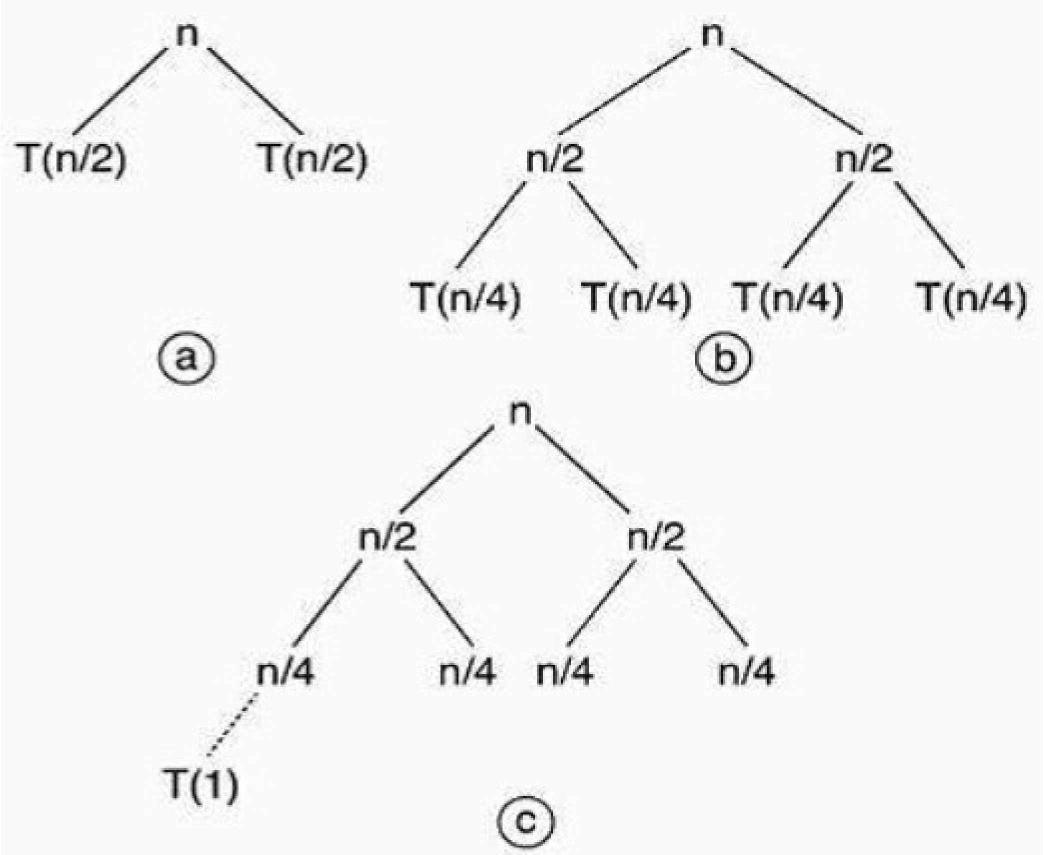
- Dois aspectos importantes devem ser considerados: a ***altura da árvore*** e o ***número de passos*** executados em cada nível.
- A solução da recorrência, que é o ***tempo de execução*** do algoritmo, é a soma de todos os passos de todos os níveis.

Método da Árvore de Recursão

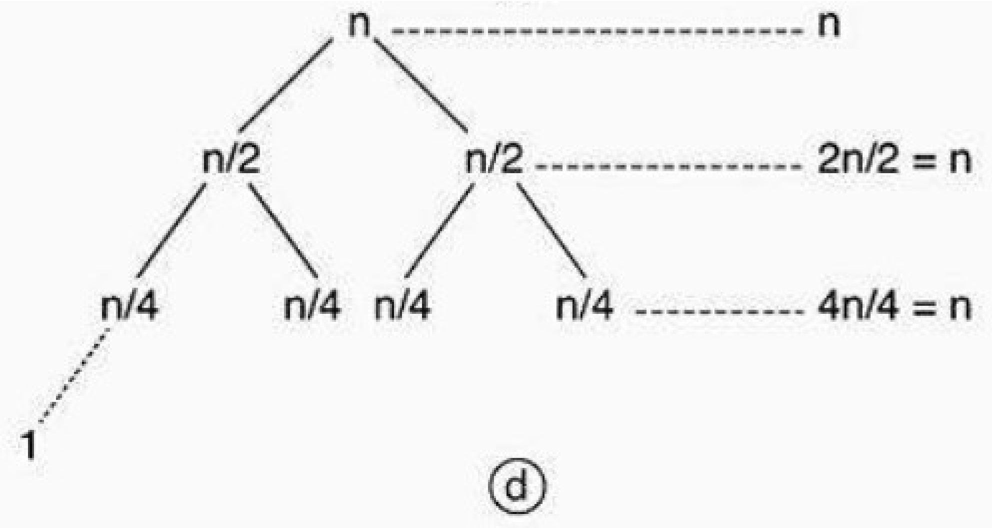
- Considere a recorrência

$$T(n) = 2T\left(\frac{n}{2}\right) + n$$

Método da Árvore de Recursão



Método da Árvore de Recursão



Método da Árvore de Recursão

$$T\left(\frac{n}{2^h}\right) = T(1)$$

$$\frac{n}{2^h} = 1$$

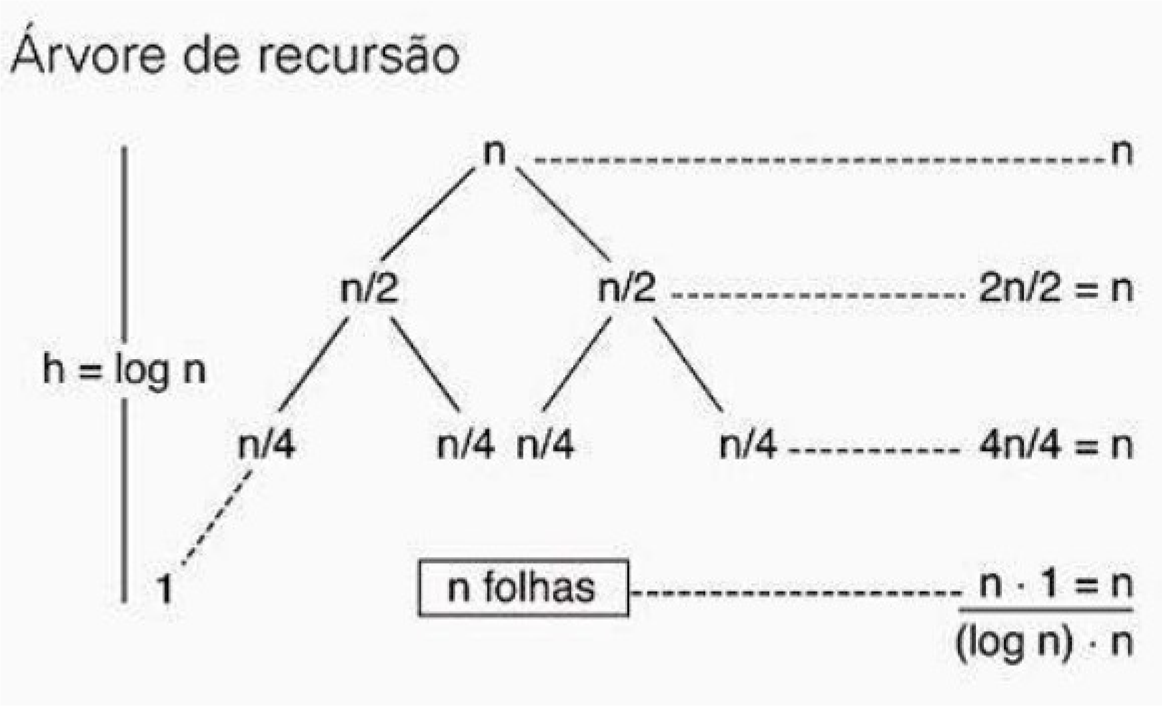
$$n = 2^h$$

$$\log_2 n = \log_2 2^h$$

$$h = \log_2 n.$$

$$T(n) = \sum_{i=0}^{h} n = n \cdot \sum_{i=0}^{h} 1 = n \cdot (\log_2 n + 1) = O(n \cdot \log_2 n).$$

Método da Árvore de Recursão



Recorrências e Soluções de Recorrências

- Método Mestre

Método Mestre

- Este método apresenta um teorema para resolver quase todas as recorrências T(n) que possuam a forma

$$a \cdot T\left(\frac{n}{b}\right) + f(n), \text{ sendo } a \geq 1 \text{ e } b > 1$$

Método Mestre

- O método possui três casos:

1. Se $f(n) = O(n^{\log_b a - \varepsilon})$, para alguma constante $\varepsilon > 0$, então $T(n) = \Theta(n^{\log_b a})$.
2. Se $f(n) = \Theta(n^{\log_b a})$, então $T(n) = \Theta(n^{\log_b a} \cdot \log n)$.
3. Se $f(n) = \Omega(n^{\log_b a + \varepsilon})$, para algum $\varepsilon > 0$ e se $a \cdot f\left(\frac{n}{b}\right) \leq c \cdot f(n)$ para alguma constante $c < 1$ e para todo n suficientemente grande, então $T(n) = \Theta(f(n))$.

Método Mestre

- ***Exemplo 1:*** T(n) = 4.T(n/2) + n
  - a = 4, b = 2 e f(n) = n.
  - $n^{\log_b a} = n^{\log_2 4} = n^2.$

Como $f(n) = n = O(n^{\log_b a-\varepsilon}) = O(n^{2-\varepsilon})$ para $\varepsilon = 1$

, pode-se aplicar o caso 1 do teorema Mestre.

Conclui-se então que a solução da recorrência é

$$T(n) = \Theta(n^{\log_b a}) = \Theta(n^2).$$

Método Mestre

- ***Exemplo 2:*** T(n) = T((2n) / 3) + 1
  - a = 1, b = 3/2 e f(n) = 1.
  - $n^{\log_b a} = n^{\log_{\frac{3}{2}} 1} = n^0 = 1.$

Como $f(n) = 1 = \Theta\left(n^{\log_b a}\right) = \Theta(1)$

, pode-se aplicar o caso 2 do teorema Mestre.

Conclui-se então que a solução da recorrência é

$$T(n) = \Theta\left(n^{\log_b a} \cdot \log n\right) = \Theta(1 \cdot log\, n) = \Theta(log\, n).$$

Método Mestre

- ***Exemplo 3:*** T(n) = 9.T(n/3) + n^3
  - a = 9, b = 3 e f(n) = n^3.
  - $n^{\log_3 9} = n^2$.
- Como $f(n) = n^3 = \Omega(n^{\log_b a+\varepsilon}) = \Omega(n^{2+\varepsilon})$, para $\varepsilon = 1$

  , pode-se aplicar o caso 3 do teorema Mestre desde que se prove também a condição $a \cdot f(n/b) \leq c \cdot f(n)$

  para alguma constante c < 1 e para todo n suficientemente grande.

Método Mestre

$$a \cdot f(n/b) \leq c \cdot f(n)$$

$$9 \cdot f(n/3) \leq c \cdot f(n), \rightarrow \; f(n) = n^3$$

$$9 \cdot \frac{n^3}{27} \leq c \cdot n^3$$

$$\frac{1}{3} n^3 \leq c \cdot n^3, c = \frac{1}{3}, n \geq 1.$$

Portanto, de acordo com o caso 3, a solução da recorrência é

$$T(n) = \Theta(f(n)) = \Theta(n^3).$$

Método Mestre

- ***Exercício:*** Use o método Mestre para fornecer limites assintóticos para as recorrências a seguir:

1. $T(n) = 4T(n/2) + n$.
2. $T(n) = 4T(n/2) + n^2$.
3. $T(n) = 4T(n/2) + n^3$.

Fim da aula