

Aula 06: Análise matemática de algoritmos recursivos

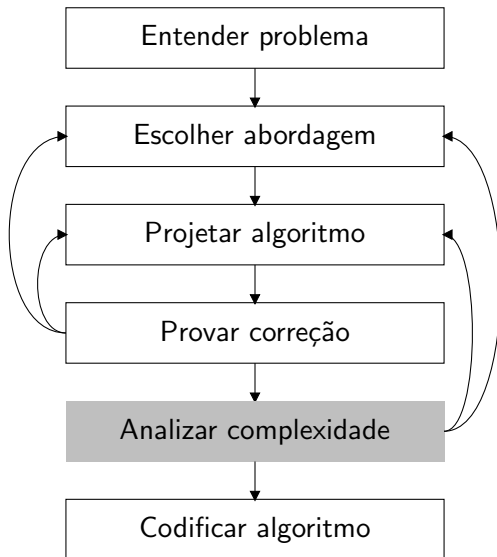
David Déharbe

Programa de Pós-graduação em Sistemas e Computação

Universidade Federal do Rio Grande do Norte

Centro de Ciências Exatas e da Terra

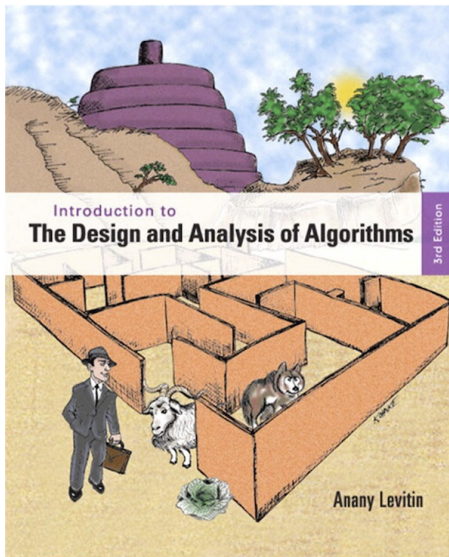
Departamento de Informática e Matemática Aplicada



Estrutura da apresentação

- 11/02 arcabouço teórico;
- 23/02 melhor caso, pior caso, caso médio;
- 25/02 notações assintóticas; O , Ω , Θ ;
- 02/03 análise de algoritmos não recursivos;
- 04/03 análise de algoritmos recursivos.

Bibliografia usada



(seções 2.4, 2.5)

Exemplo introdutório

Exemplo (Fatorial)

$F(n)$

```
1  if  $n = 0$  then return 1
2  else return  $n \times F(n - 1)$ 
```

- ▶ Tamanho da entrada: n .
- ▶ Operação básica: **multiplicação**/teste $n = 0$ /chamada recursiva
- ▶ Seja $M(n)$ o número de multiplicações:
 - ▶ $M(n) = 1 + M(n - 1)$
 - ▶ $M(0) = 0$
- ▶ M é definida por recorrência. Como encontrar uma expressão fechada, não recursiva, de $M(n)$?

Introdução

Um exemplo introdutório

Estratégia de análise

Exemplo 2

Exemplo 3

Interlúdio calculatório

Exemplo 4: A sequência de Fibonacci

Resolução de recorrência por substituição

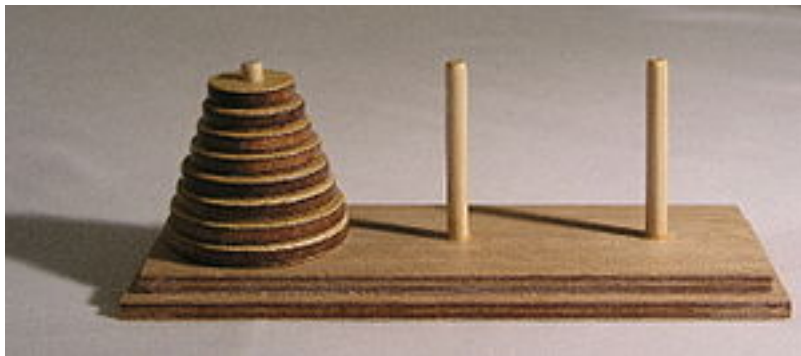
$$\begin{aligned}M(n) &= 1 + M(n-1) \\&= 1 + (1 + M(n-2)) = 2 + M(n-2) \\&= 2 + (1 + M(n-3)) = 3 + M(n-3)\end{aligned}$$

- ▶ Conjectura: $M(n) = i + M(n-i)$
- ▶ Como é conhecido $M(0) = 0 = M(n-n)$, então $M(n) = n + M(0) = n$.

Estratégia de análise

1. Identificar um parâmetro representando o tamanho da entrada
2. Identificar a operação básica do algoritmo
3. Verificar se o número de vezes que a operação básica é executada pode variar com entradas do mesmo tamanho. Se for o caso, o pior caso, a complexidade média e o melhor caso devem ser averiguados individualmente.
4. Estabelecer uma relação de recorrência que corresponde ao número de vezes que uma operação é executada. Estabelecer o valor inicial também.
5. Resolver a recorrência. Pelo menos, enquadrar o crescimento assintótico da relação.

Torres de Hanoi



A Torre de Hanói é um "quebra-cabeça" que consiste em uma base contendo três pinos, em um dos quais são dispostos alguns discos uns sobre os outros, em ordem crescente de diâmetro, de cima para baixo. O problema consiste em passar todos os discos de um pino para outro qualquer, usando um dos pinos como auxiliar, de maneira que um disco maior nunca fique em cima de outro menor em nenhuma situação. (Wikipedia)

Resolução (recursiva)

```
HANOI( $N$ ,  $src$ ,  $dest$ ,  $aux$ )  
1  if  $n = 1$   
2      return MOVE( $src$ ,  $dest$ )  
3  else  
4      HANOI( $N - 1$ ,  $src$ ,  $aux$ ,  $dest$ )  
5      MOVE( $src$ ,  $dest$ )  
6      HANOI( $N - 1$ ,  $aux$ ,  $dest$ ,  $src$ )
```

Aplicação da estratégia

HANOI(N , src , $dest$, aux)

```
1  if  $n = 1$ 
2      return MOVE( $src$ ,  $dest$ )
3  else
4      HANOI( $N - 1$ ,  $src$ ,  $aux$ ,  $dest$ )
5      MOVE( $src$ ,  $dest$ )
6      HANOI( $N - 1$ ,  $aux$ ,  $dest$ ,  $src$ )
```

- ▶ Tamanho da entrada: n
- ▶ Operação básica: mover um disco
- ▶ $M(n) = M(n-1) + 1 + M(n-1) = 2 \times M(n-1) + 1$ se $n > 1$
- ▶ $M(1) = 1$.
- ▶ Resolver M .

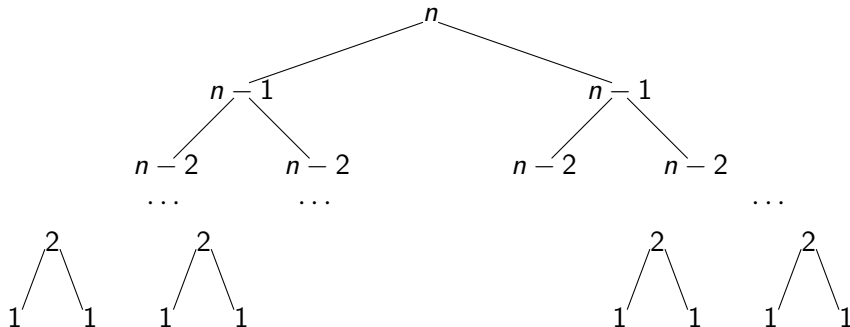
Aplicação da estratégia

$$\begin{aligned}M(n) &= 2 \times M(n-1) + 1 \\&= 2 \times (2 \times M(n-2) + 1) + 1 = 4 \times M(n-2) + 3 \\&= 4 \times (2 \times M(n-3) + 1) + 3 = 8 \times M(n-3) + 7 \\&\dots \\&= 2^i \times M(n-i) + (2^i - 1) \\&\dots \text{ (note que } 1 = n - (n-1)) \\&= 2^{n-1} \times M(n - (n-1)) + (2^{n-1} - 1) \\&= 2^{n-1} \times M(1) + 2^{n-1} - 1 \\&= 2^{n-1} + 2^{n-1} - 1 \\&= 2 \times 2^{n-1} - 1 \\M(n) &= 2^n - 1\end{aligned}$$

- ▶ A resolução recursiva do problema das Torres de Hanoi cabe em 5 linhas.
- ▶ Tem complexidade exponencial
- ▶ No caso, não tem solução mais eficiente.
- ▶ Mas cuidado: concisão não é sinônimo de eficiência!

Árvore das chamadas recursivas

Permite visualizar todas as chamadas realizadas por um algoritmo recursivo: pode ser útil para analisar a complexidade.



Número de algoritmos binários de um inteiro

BINARY(n)

1 **if** $n \leq 1$

2 **return** 1

3 **else**

4 **return** BINARY($\lfloor n/2 \rfloor$) + 1

Aplicação da estratégia

BINARY(n)

```
1  if  $n \leq 1$ 
2      return 1
3  else
4      return BINARY( $\lfloor n/2 \rfloor$ ) + 1
```

- ▶ Tamanho da entrada: n
- ▶ Operação básica: adição
- ▶ $A(n) = 1 + A(\lfloor n/2 \rfloor)$
- ▶ $A(1) = A(0) = 0$.
- ▶ Resolver A .

- ▶ $A(n) = 1 + A(\lfloor n/2 \rfloor)$
- ▶ $A(1) = A(0) = 0$.
- ▶ Resolver A ?

Vamos considerar o caso de $n = 2^k$ (n é potência de 2).

$$\begin{aligned} A(n) &= 1 + A(\lfloor 2^k/2 \rfloor) = 1 + A(2^{k-1}) \\ &= 1 + (1 + A(\lfloor 2^{k-1}/2 \rfloor)) = 2 + A(2^{k-2}) \\ &\dots \\ &= i + A(2^{k-i}) \\ &\dots \\ &= k + A(2^{k-k}) = k + A(2^0) = k + A(1) \\ &= k \\ &= \log_2 n \in \Theta(\log n) \end{aligned}$$

- ▶ Teorema Master
- ▶ Relações de recorrência de segunda ordem

O teorema Master

Hipóteses

⇒ http://en.wikipedia.org/wiki/Master_theorem

Considere a sequência $T(n)$ definida pela seguinte relação de recorrência:

$$T(n) = aT\left(\frac{n}{b}\right) + f(n), \text{ onde } a \geq 1, b \geq 1$$

Aplicação a algoritmos recursivos:

- ▶ n tamanho do problema
- ▶ a quantidade de chamadas recursivas
- ▶ n/b tamanho dos sub-problemas
- ▶ $f(n)$ custo sem contar as chamadas recursivas
 - ▶ divisão em sub-problemas,
 - ▶ combinação dos resultados parciais.

O teorema Master

$$T(n) = aT\left(\frac{n}{b}\right) + f(n), \text{ onde } a \geq 1, b \geq 1$$

1. Se $f(n) \in O(n^c)$ onde $c < \log_b a$, então $T(n) \in \Theta(n^{\log_b a})$
2. Se existe $k \geq 0$ tal que $f(n) \in \Theta(n^c \log^k n)$ onde $c = \log_b a$ então $T(n) \in \Theta(n^c \log^{k+1} n)$
3. Se
 - 3.1 $f(n) \in \Omega(n^c)$, onde $c > \log_b a$ e
 - 3.2 (condição de regularidade) $af\left(\frac{n}{b}\right) \leq kf(n)$ para $k < 1$ e n suficientemente grande,então $T(n) \in \Theta(f(n))$.

CORMEN, LEISERSON, RIVEST, STEIN

O teorema Master

Propriedades: caso 1

$$T(n) = aT\left(\frac{n}{b}\right) + f(n), \text{ onde } a \geq 1, b \geq 1$$

- ▶ Se $f(n) \in O(n^c)$ onde $c < \log_b a$, então $T(n) \in \Theta(n^{\log_b a})$
- ▶ Exemplo:

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

- ▶ Temos que: $a = 8, b = 2, f(n) = n^2$,
- ▶ e $f(n) \in O(n^2)$,
- ▶ logo $c = 2 < 3 = \log_2 8 = \log_b a$.
- ▶ Pelo teorema $T(n) \in \Theta(n^{\log_b a}) = \Theta(n^3)$

O teorema Master

Caso 2

$$T(n) = aT\left(\frac{n}{b}\right) + f(n), \text{ onde } a \geq 1, b \geq 1$$

- ▶ Se existe $k \geq 0$ tal que $f(n) \in \Theta(n^c \log^k n)$ onde $c = \log_b a$ então $T(n) \in \Theta(n^c \log^{k+1} n)$
- ▶ Exemplo:

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

- ▶ Temos $a = 2, b = 2$, logo $c = \log_2 2 = 1$,
- ▶ e $f(n) = n \in \Theta(n) = \Theta(n^1 \cdot 1) = \Theta(n^c \log^k n)$, com $k = 0$,
- ▶ logo $T(n) \in \Theta(n \log n)$.

O teorema Master

Caso 3

$$T(n) = aT\left(\frac{n}{b}\right) + f(n), \text{ onde } a \geq 1, b \geq 1$$

► Se

1. $f(n) \in \Omega(n^c)$, onde $c > \log_b a$ e
2. (condição de regularidade) $af\left(\frac{n}{b}\right) \leq kf(n)$ para $k < 1$ e n suficientemente grande,

então $T(n) \in \Theta(f(n))$.

► Exemplo

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

- Temos $a = 2, b = 2, f(n) = n^2$,
- logo $f(n) \in \Omega(n^c)$, onde $c = 2 > \log_b a$,
- a condição de regularidade $2 \cdot \left(\frac{n}{2}\right)^2 = 2\left(\frac{n^2}{4}\right) \leq kn^2$ é satisfeita a partir de $k = \frac{1}{2}$,
- então $T(n) \in \Theta(f(n)) = \Theta(n^2)$

O teorema Master

Tamanho da representação binária

$$A(n) = A(\lfloor n/2 \rfloor) + 1$$

Hipóteses do teorema master:

$$T(n) = aT\left(\frac{n}{b}\right) + f(n), \text{ onde } a \geq 1, b \geq 1$$

- ▶ Temos $a = 1$, $b = 2$, $\log_2 1 = 0$, e $n^c = n^0 = 1$
- ▶ temos também $f(n) = 1 \in \Theta(n^c \log^k n)$, com $k = 0$,
- ▶ logo $A(n) \in \Theta(n^c \log^{k+1} n)$, ou seja $A(n) \in \Theta(\log n)$.

Resolução de relações de recorrência

Recorrências lineares de segunda ordem com coeficientes constantes:

$$a \times x(n) + b \times x(n-1) + c \times x(n-2) = f(n).$$

Caso $f(n) = 0$, então a recorrência é *homogênea*.

A equação $ax^2 + bx + c = 0$ é a *equação característica* da recorrência.

Resolução de relações de recorrência

Teorema

Seja r_1, r_2 as raízes da equação característica de uma recorrência linear homogênea de segunda ordem com coeficientes constantes.

caso 1 *Se r_1 and r_2 são números reais e distintos, a solução geral é*
$$x(n) = \alpha r_1^n + \beta r_2^n.$$

caso 2 *Se $r_1 = r_2 = r$, uma solução geral é*
$$x(n) = \alpha r^n + \beta n r^n.$$

caso 3 *Se $r_{1,2} = u \pm iv$ são complexos distintos, a solução geral é*
$$x(n) = \gamma^n (\alpha \cos n\theta + \beta \sin n\theta), \text{ onde } \theta = \arctan v/u,$$

$$\gamma = \sqrt{u^2 + v^2}.$$

Em todos os casos α e β são constantes reais.

Sequência de Fibonacci (exemplo)

- ▶ É uma sequência infinita, crescente, de números inteiros;
http://en.wikipedia.org/wiki/Fibonacci_number
- ▶ Definição da sequência, por recorrência;
- ▶ Aplicação do teorema: formulação não recorrente;
- ▶ Análise da complexidade de diferentes algoritmos.

Sequência de Fibonacci (exemplo)

Sequência definida pelo matemático Fibonacci:

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, ...

A sequência é definida por recorrência:

- ▶ caso geral: $F(n) = F(n-1) + F(n-2)$, se $n > 1$,
- ▶ condições iniciais: $F(0) = 0$ e $F(1) = 1$.

Sequência de Fibonacci

- ▶ Note que: $F(n) - F(n-1) - F(n-2) = 0$, se $n > 1$.
- ▶ É uma relação de recorrência linear homogênea de segunda ordem, com coeficientes constantes.
- ▶ Sua equação característica é $x^2 - x - 1 = 0$.
- ▶ As raízes são $r_1 = \frac{1+\sqrt{5}}{2}$ e $r_2 = \frac{1-\sqrt{5}}{2}$.
- ▶ O termo geral é $F(n) = \alpha\left(\frac{1+\sqrt{5}}{2}\right)^n + \beta\left(\frac{1-\sqrt{5}}{2}\right)^n$.
- ▶ Com as condições iniciais, temos $\alpha + \beta = 0$ e $\frac{1+\sqrt{5}}{2}\alpha + \frac{1-\sqrt{5}}{2}\beta = 1$.
- ▶ $F(n) = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^n = \frac{1}{\sqrt{5}}(\phi^n - \hat{\phi}^n) \in \Theta(\phi^n)$.

$$\phi \approx 1,61803 \text{ e } \hat{\phi} \approx -0,61803$$

Sequência de Fibonacci

Exemplo (Fibonacci)

$FIB(n)$

```
1  if  $n \leq 1$  then return  $n$   
2  else return  $FIB(n - 1) + FIB(n - 2)$ 
```

- ▶ tamanho da entrada: n
- ▶ operação básica: adição
- ▶ custo:
 - ▶ caso geral: $A(n) = A(n - 1) + A(n - 2) + 1$
 - ▶ condições gerais: $A(0) = A(1) = 0$.
- ▶ resolver $A(n)$...

Sequência de Fibonacci

- ▶ Resolver:
 - ▶ caso geral: $A(n) = A(n-1) + A(n-2) + 1$
 - ▶ condições gerais: $A(0) = A(1) = 0$.
- ▶ temos: $A(n) - A(n-1) - A(n-2) = 1$ não é homogênea.
- ▶ seja $B(n) = A(n) + 1$: $B(n) - B(n-1) - B(n-2) = 0$ é homogênea!
- ▶ mais ainda: $B(n) = F(n+1)$, logo
 $B(n) = \frac{1}{\sqrt{5}}(\phi^{n+1} - \hat{\phi}^{n+1}) \in \Theta(\phi^n)$.

Sequência de Fibonacci: versão eficiente

- ▶ Sem recursividade

FIB-NR(n)

```
1   $F[0] = 0, F[1] = 1$   
2  for  $i = 2$  to  $n$   
3       $F[i] = F[i - 1] + F[i - 2]$   
4  return  $F[n]$ 
```

- ▶ Com recursividade

FIB-ACC(n, k, x, y)

```
1  if  $n = k$  then return  $x + y$   
2  else return FIB-ACC( $n, k + 1, y, x + y$ )
```

FIB-R(n)

```
1  if  $n \leq 1$  then return  $n$   
2  else return FIB-ACC( $n, 2, 0, 1$ )
```


Estabeleça a complexidade de FIB-NR e de FIB-R .