TÉCNICO LISBOA

Análise e Síntese de Algoritmos

Teorema Mestre. Divide & Conquer.

CLRS Cap. 4

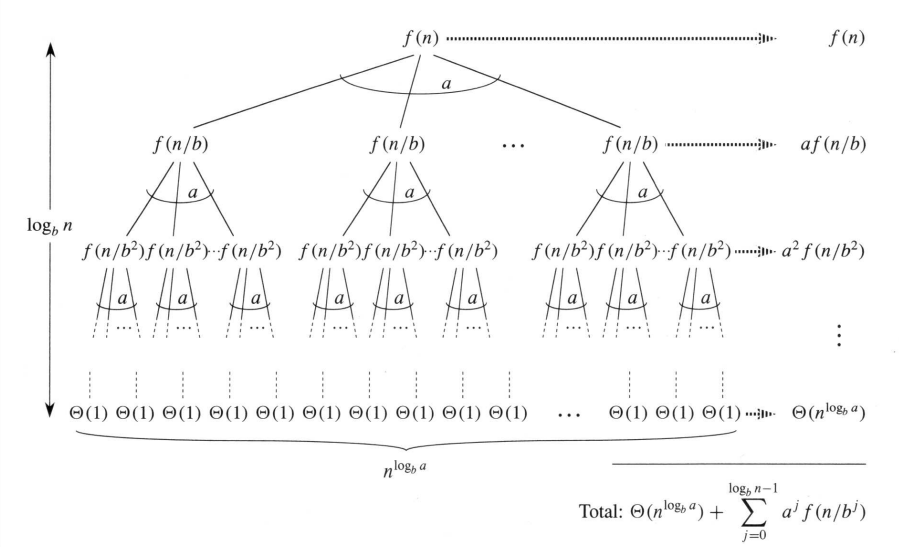
Instituto Superior Técnico

2022/2023

Programa

- Revisão [CLRS, Cap.1-13]
  - Fundamentos; notação; exemplos
- Técnicas de Síntese de Algoritmos [CLRS, Cap.15-16]
  - Programação dinâmica
  - Algoritmos greedy
- Algoritmos em Grafos [CLRS, Cap. 21-26]
  - Algoritmos elementares
  - Árvores abrangentes
  - Caminhos mais curtos
  - Fluxos máximos
- Programação Linear [CLRS, Cap.29]
  - Algoritmos e modelação de problemas com restrições lineares
- Tópicos Adicionais [CLRS, Cap.32-35]
  - Emparelhamento de Cadeias de Caracteres
  - Complexidade Computacional

Recorrências



Teorema Mestre

Permite resolver recorrências da forma:

$$T(n) = aT(n/b) + f(n) \quad , \quad a \geq 1 \, , \; b > 1$$

Problema é dividido em a subproblemas, cada um com dimensão n/b.

Problema simplificado:

$$T(n) = aT(n/b) + O(n^d) \quad , \quad a \geq 1 \, , \; b > 1 \, , \; d \geq 0$$

$T(n)$ é limitado assimptoticamente da seguinte forma:

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

Teorema Mestre

Permite resolver recorrências da forma (simplificado):

$$T(n) = aT(n/b) + O(n^d) \quad , \quad a \geq 1 \ , \ b > 1 \ , \ d \geq 0$$

$T(n)$ é limitado assintoticamente da seguinte forma:

$$T(n) = \begin{cases} \Theta(n^{log_b\, a}) & if\ d < log_b\, a \\ \quad \text{Custo das folhas domina custo total} & \\ \Theta(n^d\, log\ n) & if\ d = log_b\, a \\ \quad \text{Custo uniformemente distribuído} & \\ \Theta(n^d) & if\ d > log_b\, a \\ \quad \text{Custo da raíz domina custo total} & \end{cases}$$

Motivação

Algoritmos - divide & conquer

- Partir o problema em sub-problemas (instâncias do mesmo problema)
- Resolver (recursivamente) o sub-problema
- Combinar resultados

Exemplos: mergesort, quicksort, procura binária, min/max, multiplicação matrizes, ...

Motivação

Exemplo 1: Mergesort

MergeSort(A, p, r)

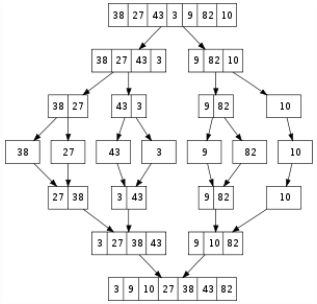
if $p < r$ **then**
  $q \leftarrow \lfloor (p + r)/2 \rfloor$
  MergeSort(A, p, q)
  MergeSort$(A, q + 1, r)$
  Merge(A, p, q, r)
end if

Qual o tempo execução no pior caso?

- Admitindo que tempo de execução de Merge cresce com n

Exemplo 1: Mergesort

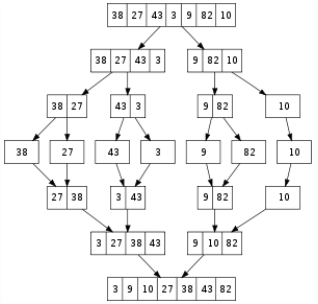
MergeSort(A, p, r)
if $p < r$ **then**
  $q \leftarrow \lfloor (p + r)/2 \rfloor$
  MergeSort(A, p, q)
  MergeSort$(A, q + 1, r)$
  Merge(A, p, q, r)
end if



- Partir: as duas chamadas a MergeSort dividem cada problema em dois sub-problemas com metade dos elementos. Repetido recursivamente até cada sub-problema ter apenas 1 elemento.

Exemplo 1: Mergesort

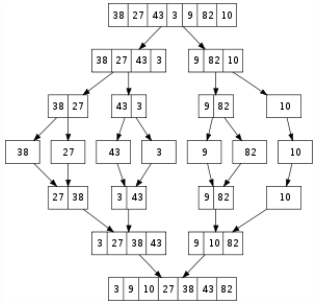
MergeSort(A, p, r)
if $p < r$ **then**
  $q \leftarrow \lfloor (p + r)/2 \rfloor$
  MergeSort(A, p, q)
  MergeSort$(A, q + 1, r)$
  Merge(A, p, q, r)
end if



- Partir: as duas chamadas a MergeSort dividem cada problema em dois sub-problemas com metade dos elementos. Repetido recursivamente até cada sub-problema ter apenas 1 elemento.
- Resolver: a função Merge ordena os elementos em $A[p, r]$, considerando que os elementos em $A[p, q]$ e $A[q + 1, r]$ já estão ordenados entre si.

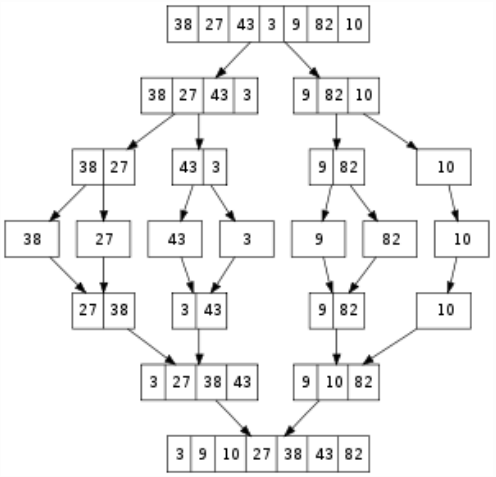
Exemplo 1: Mergesort

MergeSort(A, p, r)
if $p < r$ **then**
  $q \leftarrow \lfloor (p + r)/2 \rfloor$
  MergeSort(A, p, q)
  MergeSort$(A, q + 1, r)$
  Merge(A, p, q, r)
end if

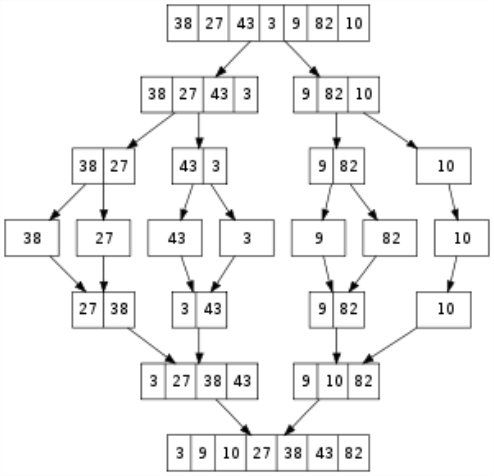


- Partir: as duas chamadas a MergeSort dividem cada problema em dois sub-problemas com metade dos elementos. Repetido recursivamente até cada sub-problema ter apenas 1 elemento.
- Resolver: a função Merge ordena os elementos em $A[p, r]$, considerando que os elementos em $A[p, q]$ e $A[q + 1, r]$ já estão ordenados entre si.
- Combinar: a combinação é efetuada de forma implícita, dado que a informação está a ser diretamente manipulada sobre A

Exemplo 1: Mergesort

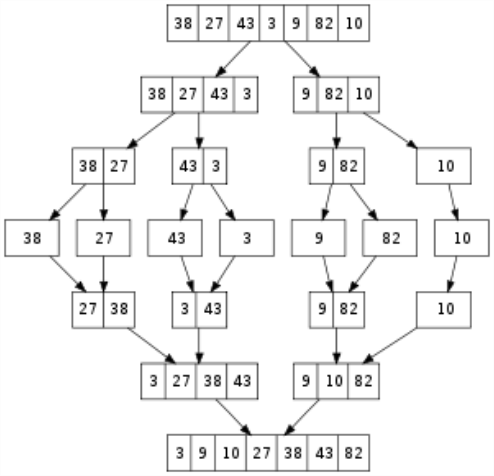


Exemplo 1: Mergesort



$$T(n) = 2T(n/2) + \Theta(n)$$

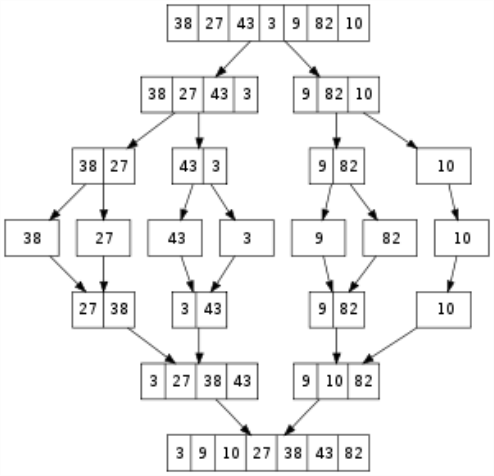
Exemplo 1: Mergesort



$$T(n) = 2T(n/2) + \Theta(n)$$

- $a = 2, b = 2, d = 1$

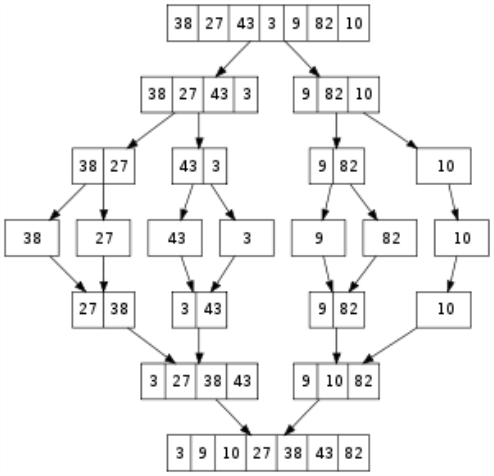
Exemplo 1: Mergesort



$$T(n) = 2T(n/2) + \Theta(n)$$

- $a = 2, b = 2, d = 1$
- $d = 1$ é $=$ a $log_2 2$

Exemplo 1: Mergesort



$$T(n) = 2T(n/2) + \Theta(n)$$

- $a = 2, b = 2, d = 1$
- $d = 1$ é $=$ a $log_2 2$
- $T(n) = \Theta(n^d \log n) = \Theta(n \log n)$ (caso 2 do Teorema Mestre)

Exemplo 2: Quicksort

Partition(A, p, r)

$x \leftarrow A[r]$
$i \leftarrow p - 1$
for $j \leftarrow p$ **to** $r - 1$ **do**
  if $A[j] \leq x$ **then**
    $i \leftarrow i + 1$
    $A[i] \leftrightarrow A[j]$
  end if
end for
$A[i + 1] \leftrightarrow A[j + 1]$
return $i + 1$

Complexidade

- $O(n)$

Propriedades

- todos os elementos à esquerda do pivot são $\leq$ pivot
- todos os elementos à direita do pivot são $>$ pivot
- após execução de Partition() o pivot está na sua posição final

Prova por invariante de ciclo

No início de cada iteração do for:

- todos os elementos à esquerda do pivot são $\leq$ pivot
- todos os elementos à direita do pivot são $>$ pivot

Exemplo 2: Quicksort

Quicksort(A, p, r)

if $p < r$ **then**
  $q \leftarrow$ Partition(A, p, r)
  Quicksort$(A, p, q - 1)$
  Quicksort$(A, q + 1, r)$
end if

- Vector não necessariamente dividido em 2 partes iguais
- Constantes menores (*in place*)

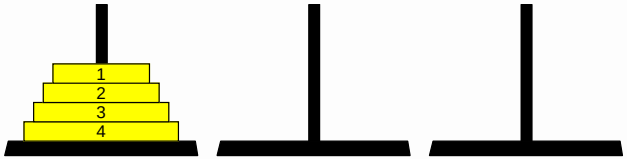
Complexidade

- Pior caso: $O(n^2)$
- Melhor caso: $O(n \log n)$

Na prática, QuickSort (aleatorizado) é mais rápido que MergeSort

Exemplo 3: Torres de Hanoi

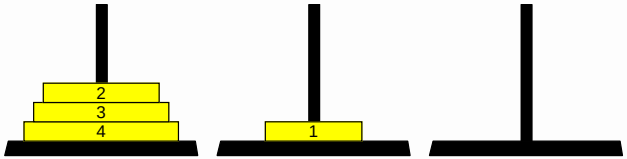
- 3 suportes
- n discos colocados num suporte, por ordem decrescente de tamanho
- Transferir os n discos para outro suporte, mantendo sempre a relação de ordem entre eles



Q: Qual o menor número de movimentos de disco individuais que é necessário e suficiente para deslocar os n discos de um suporte para outro ?

Exemplo 3: Torres de Hanoi

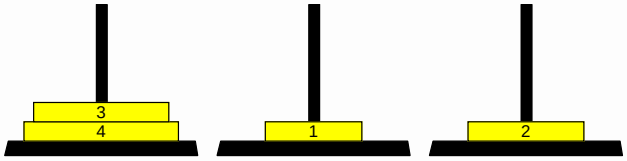
- 3 suportes
- n discos colocados num suporte, por ordem decrescente de tamanho
- Transferir os n discos para outro suporte, mantendo sempre a relação de ordem entre eles



Q: Qual o menor número de movimentos de disco individuais que é necessário e suficiente para deslocar os n discos de um suporte para outro ?

Exemplo 3: Torres de Hanoi

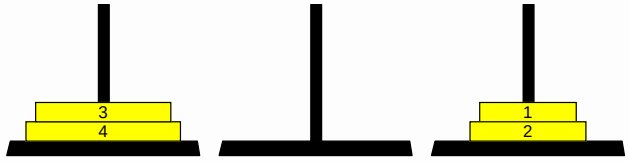
- 3 suportes
- n discos colocados num suporte, por ordem decrescente de tamanho
- Transferir os n discos para outro suporte, mantendo sempre a relação de ordem entre eles



Q: Qual o menor número de movimentos de disco individuais que é necessário e suficiente para deslocar os n discos de um suporte para outro ?

Exemplo 3: Torres de Hanoi

- 3 suportes
- n discos colocados num suporte, por ordem decrescente de tamanho
- Transferir os n discos para outro suporte, mantendo sempre a relação de ordem entre eles

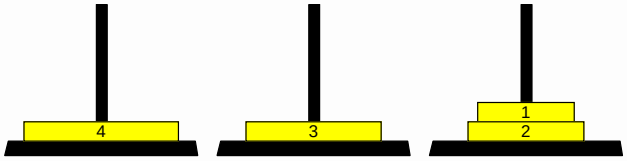


Q: Qual o menor número de movimentos de disco individuais que é necessário e suficiente para deslocar os n discos de um suporte para outro ?

Análise e Síntese de Algoritmos - 2022/2023 12/15

Exemplo 3: Torres de Hanoi

- 3 suportes
- n discos colocados num suporte, por ordem decrescente de tamanho
- Transferir os n discos para outro suporte, mantendo sempre a relação de ordem entre eles

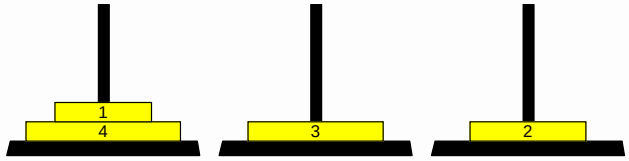


Q: Qual o menor número de movimentos de disco individuais que é necessário e suficiente para deslocar os n discos de um suporte para outro ?

Análise e Síntese de Algoritmos - 2022/2023 12/15

Exemplo 3: Torres de Hanoi

- 3 suportes
- n discos colocados num suporte, por ordem decrescente de tamanho
- Transferir os n discos para outro suporte, mantendo sempre a relação de ordem entre eles

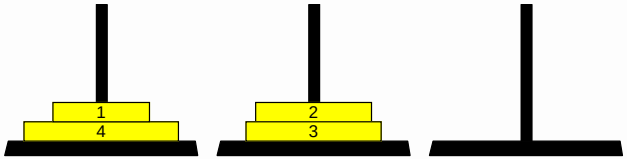


Q: Qual o menor número de movimentos de disco individuais que é necessário e suficiente para deslocar os n discos de um suporte para outro ?

Análise e Síntese de Algoritmos - 2022/2023 12/15

Exemplo 3: Torres de Hanoi

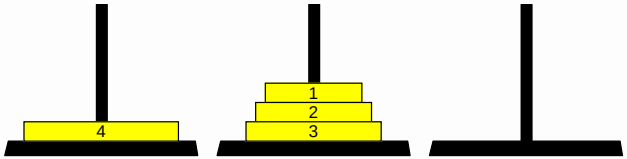
- 3 suportes
- n discos colocados num suporte, por ordem decrescente de tamanho
- Transferir os n discos para outro suporte, mantendo sempre a relação de ordem entre eles



Q: Qual o menor número de movimentos de disco individuais que é necessário e suficiente para deslocar os n discos de um suporte para outro ?

Exemplo 3: Torres de Hanoi

- 3 suportes
- n discos colocados num suporte, por ordem decrescente de tamanho
- Transferir os n discos para outro suporte, mantendo sempre a relação de ordem entre eles

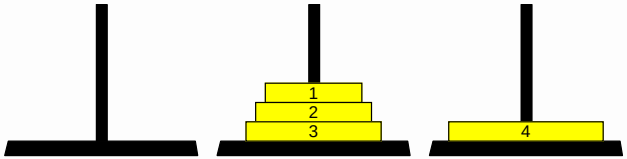


Q: Qual o menor número de movimentos de disco individuais que é necessário e suficiente para deslocar os n discos de um suporte para outro ?

Análise e Síntese de Algoritmos - 2022/2023 12/15

Exemplo 3: Torres de Hanoi

- 3 suportes
- n discos colocados num suporte, por ordem decrescente de tamanho
- Transferir os n discos para outro suporte, mantendo sempre a relação de ordem entre eles

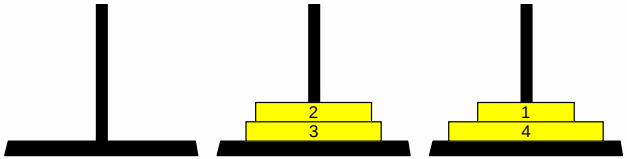


Q: Qual o menor número de movimentos de disco individuais que é necessário e suficiente para deslocar os n discos de um suporte para outro ?

Análise e Síntese de Algoritmos - 2022/2023 12/15

Exemplo 3: Torres de Hanoi

- 3 suportes
- n discos colocados num suporte, por ordem decrescente de tamanho
- Transferir os n discos para outro suporte, mantendo sempre a relação de ordem entre eles

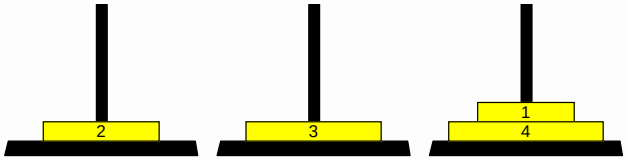


Q: Qual o menor número de movimentos de disco individuais que é necessário e suficiente para deslocar os n discos de um suporte para outro ?

Análise e Síntese de Algoritmos - 2022/2023 12/15

Exemplo 3: Torres de Hanoi

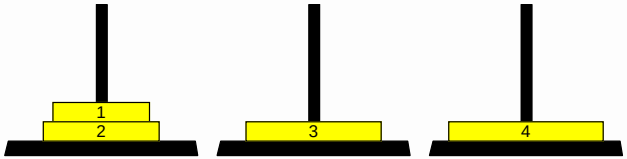
- 3 suportes
- n discos colocados num suporte, por ordem decrescente de tamanho
- Transferir os n discos para outro suporte, mantendo sempre a relação de ordem entre eles



Q: Qual o menor número de movimentos de disco individuais que é necessário e suficiente para deslocar os n discos de um suporte para outro ?

Exemplo 3: Torres de Hanoi

- 3 suportes
- n discos colocados num suporte, por ordem decrescente de tamanho
- Transferir os n discos para outro suporte, mantendo sempre a relação de ordem entre eles

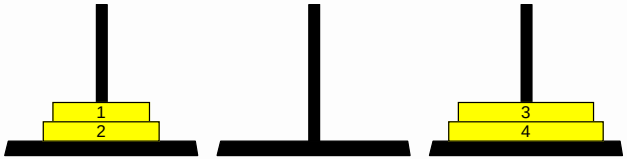


Q: Qual o menor número de movimentos de disco individuais que é necessário e suficiente para deslocar os n discos de um suporte para outro ?

Análise e Síntese de Algoritmos - 2022/2023 12/15

Exemplo 3: Torres de Hanoi

- 3 suportes
- n discos colocados num suporte, por ordem decrescente de tamanho
- Transferir os n discos para outro suporte, mantendo sempre a relação de ordem entre eles

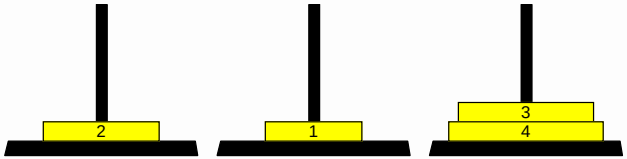


Q: Qual o menor número de movimentos de disco individuais que é necessário e suficiente para deslocar os n discos de um suporte para outro ?

Análise e Síntese de Algoritmos - 2022/2023 12/15

Exemplo 3: Torres de Hanoi

- 3 suportes
- n discos colocados num suporte, por ordem decrescente de tamanho
- Transferir os n discos para outro suporte, mantendo sempre a relação de ordem entre eles

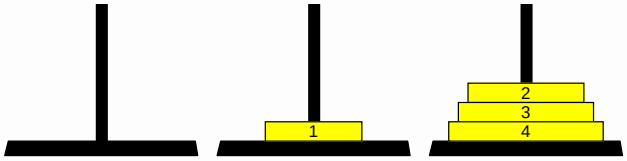


Q: Qual o menor número de movimentos de disco individuais que é necessário e suficiente para deslocar os n discos de um suporte para outro ?

Análise e Síntese de Algoritmos - 2022/2023 12/15

Exemplo 3: Torres de Hanoi

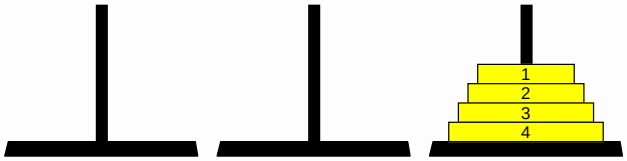
- 3 suportes
- n discos colocados num suporte, por ordem decrescente de tamanho
- Transferir os n discos para outro suporte, mantendo sempre a relação de ordem entre eles



Q: Qual o menor número de movimentos de disco individuais que é necessário e suficiente para deslocar os n discos de um suporte para outro ?

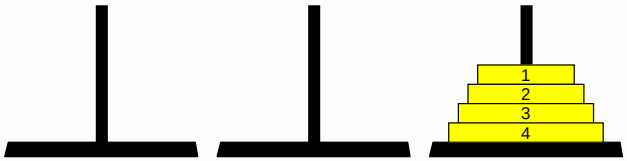
Exemplo 3: Torres de Hanoi

- 3 suportes
- n discos colocados num suporte, por ordem decrescente de tamanho
- Transferir os n discos para outro suporte, mantendo sempre a relação de ordem entre eles



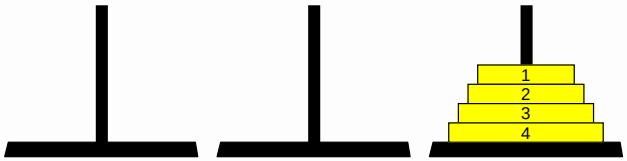
Q: Qual o menor número de movimentos de disco individuais que é necessário e suficiente para deslocar os n discos de um suporte para outro ? Para $n = 4$, foram necessários 15 movimentos

Exemplo 3: Torres de Hanoi



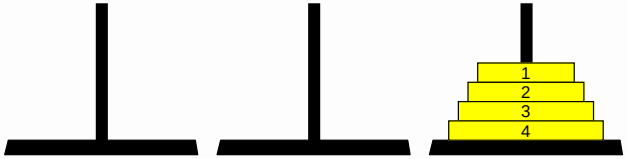
- $T(n)$ - menor número de movimentos de disco individuais, necessário e suficiente para transferir n discos, de um suporte para outro
- $T(0) = 0,\ T(1) = 1,\ T(4) = 15$
- $T(n)$?

Exemplo 3: Torres de Hanoi



- $T(n)$ - menor número de movimentos de disco individuais, necessário e suficiente para transferir n discos, de um suporte para outro
- $T(0) = 0,\ T(1) = 1,\ T(4) = 15$
- $T(n)$?
  - Transferir $n-1$ discos para outro suporte ($T(n-1)$ movimentos)
  - Deslocar maior disco para suporte final
  - Transferir $n-1$ discos para o suporte final ($T(n-1)$ movimentos)

Exemplo 3: Torres de Hanoi



- $T(n)$ - menor número de movimentos de disco individuais, necessário e suficiente para transferir n discos, de um suporte para outro
- $T(0) = 0,\ T(1) = 1,\ T(4) = 15$
- $T(n)$?
  - Transferir $n-1$ discos para outro suporte ($T(n-1)$ movimentos)
  - Deslocar maior disco para suporte final
  - Transferir $n-1$ discos para o suporte final ($T(n-1)$ movimentos)
- $T(n) = 2T(n-1) + 1$, para $n > 0$

Exemplo 3: Torres de Hanoi

Recorrência

- $T(0) = 0$
- $T(n) = 2T(n-1) + 1$, para $n > 0$

Exemplo 3: Torres de Hanoi

Recorrência

- $T(0) = 0$
- $T(n) = 2T(n-1) + 1$, para $n > 0$

Solução por substituição

- Tentativa: $T(n) = 2^n - 1$ correcta.
- Substituir: $T(n-1) = 2^{n-1} - 1$ em
  $T(n) = 2T(n-1) + 1$
  $T(n) = 2\,(2^{n-1} - 1) + 1 = 2^n - 1$!

Exemplo 3: Torres de Hanoi

Recorrência

- $T(0) = 0$
- $T(n) = 2T(n-1) + 1$, para $n > 0$

Solução por substituição

- Tentativa: $T(n) = 2^n - 1$ correcta.
- Substituir: $T(n-1) = 2^{n-1} - 1$ em
  $T(n) = 2T(n-1) + 1$
  $T(n) = 2\,(2^{n-1} - 1) + 1 = 2^n - 1$!

Complexidade

- $O(2^n)$

Notação Assimptótica