Algoritmos de Busca

Profa. Sheila Morais de Almeida

DAINF-UTFPR-PG

junho - 2018

Este material é preparado usando como referências os textos dos seguintes livros.

Análise de Algoritmos

Vamos analisar o algoritmo de busca em um vetor não-ordenado.

Quantas operações de comparação são executadas?

Algoritmo Busca Linear
Entrada: vetor $v[1..n]$ de inteiros;
n número de elementos no vetor; k elemento procurado

```
i ← 1
enquanto i ≤ n && v[i] ≠ k faça:
    i ← i + 1
if i ≤ n então
    retorne i
else
    retorne 0
```

Pergunta: Quantas instruções básicas do modelo computacional RAM (operações aritméticas básicas, atribuições e comparações) são executadas pelo Algoritmo Linear, considerando uma entrada de tamanho n de pior caso?

Pior caso: o elemento não está no vetor.

Considere o Algoritmo Linear:

Algoritmo Busca Linear
Entrada: vetor $v[1..n]$ de inteiros;
n número de elementos no vetor; k elemento procurado

$i \leftarrow 1$ $\quad 1$
enquanto $i \leq n \ \&\& \ v[i] \neq k$ **faça:** $\quad 2n+1$
$i \leftarrow i + 1$ $\quad 2n$
if $i \leq n$ **então** $\quad 1$
retorne i
else
retorne 0

Considere o Algoritmo Linear:

Algoritmo Busca Linear
Entrada: vetor $v[1..n]$ de inteiros;
n número de elementos no vetor; k elemento procurado

$i \leftarrow 1$ $\quad 1$
enquanto $i \leq n$ && $v[i] \neq k$ **faça:** $\quad 2n + 1$
$\quad i \leftarrow i + 1$ $\quad 2n$
if $i \leq n$ **então** $\quad 1$
$\quad$ **retorne** i
else
$\quad$ **retorne** 0

Total: $4n + 3$

Análise de complexidade de tempo no pior caso

Então, no pior caso do Algoritmo Busca Linear, $f(n) = 4n + 3$.

$$4n + 3 \leq 4n + n = 5n, \forall n \geq 3$$

Então, $4n + 3 \in O(n)$.

$$4n + 3 \geq 4n, \forall n \geq 0$$

Então, $4n + 3 \in \Omega(n)$.

Portanto, o Algoritmo Busca Linear executa em tempo $\Theta(n)$ no pior caso.

Análise de complexidade de tempo no melhor caso

Considere o Algoritmo Linear:

Algoritmo Busca Linear
Entrada: vetor $v[1..n]$ de inteiros;
n número de elementos no vetor; k elemento procurado

$i \leftarrow 1$
enquanto $i \leq n \; \&\& \; v[i] \neq k$ **faça:**
$i \leftarrow i + 1$
if $i \leq n$ **então**
retorne i
else
retorne 0

Análise de complexidade de tempo no melhor caso

Pergunta: Quantas instruções básicas do modelo computacional RAM (operações aritméticas básicas, atribuições e comparações) são executadas pelo Algoritmo Linear, considerando uma entrada de tamanho n de melhor caso?

Melhor caso: o elemento está na primeira posição do vetor.

Considere o Algoritmo Linear:

Algoritmo Busca Linear
Entrada: vetor $v[1..n]$ de inteiros;
n número de elementos no vetor; k elemento procurado

$i \leftarrow 1$ 1
enquanto $i \leq n$ && $v[i] \neq k$ **faça:** 2
$i \leftarrow i + 1$ 0
if $i \leq n$ **então** 1
retorne i
else
retorne 0

Análise de complexidade de tempo no melhor caso

Considere o Algoritmo Linear:

Algoritmo Busca Linear
Entrada: vetor $v[1..n]$ de inteiros;
n número de elementos no vetor; k elemento procurado

$i \leftarrow 1$ 1
enquanto $i \leq n$ && $v[i] \neq k$ **faça:** 2
$i \leftarrow i + 1$ 0
if $i \leq n$ **então** 1
retorne i
else
retorne 0

Total: 4

Análise de complexidade de tempo no melhor caso

Então, no melhor caso do Algoritmo Busca Linear, $f(n) = 4$.

$$4 \leq 4 \times 1, \forall n \geq 0$$

Então, $4 \in O(1)$.

$$4 \geq 4 \times 1, \forall n \geq 0$$

Então, $4 \in \Omega(1)$.

Portanto, o Algoritmo Busca Linear executa em tempo $\Theta(1)$ no melhor caso.

Considere o Algoritmo Linear:

Algoritmo Busca Linear
Entrada: vetor $v[1..n]$ de inteiros;
n número de elementos no vetor; k elemento procurado

```
i ← 1
enquanto  i ≤ n && v[i] ≠ k  faça:
    i ← i + 1
if i ≤ n  então
    retorne i
else
    retorne 0
```

Análise de complexidade de tempo no caso médio

Pergunta: Quantas instruções básicas do modelo computacional RAM (operações aritméticas básicas, atribuições e comparações) são executadas pelo Algoritmo Linear, considerando uma entrada de tamanho n de caso médio?

O que é o caso médio neste contexto?

Análise de complexidade de tempo no caso médio

Vamos supor que trata-se de uma aplicação em que qualquer elemento tem a mesma chance de estar em qualquer posição do vetor, ou seja, é uma distribuição uniforme.

Suponha que o elemento procurado está no vetor.

Então, qual a chance de o elemento estar em cada posição do vetor?

Análise de complexidade de tempo no caso médio

Vamos supor que trata-se de uma aplicação em que qualquer elemento tem a mesma chance de estar em qualquer posição do vetor, ou seja, é uma distribuição uniforme.

Suponha que o elemento procurado está no vetor.

Então, qual a chance de o elemento estar em cada posição do vetor? $\frac{1}{n}$

Análise de complexidade de tempo no pior caso

Considere o Algoritmo Linear e suponha que o elemento k está na posição t.

Algoritmo Busca Linear
Entrada: vetor $v[1..n]$ de inteiros;
n número de elementos no vetor; k elemento procurado

$i \leftarrow 1$ $\quad 1$
enquanto $i \leq n$ && $v[i] \neq k$ **faça:** $\quad 2t$
$i \leftarrow i + 1$ $\quad 2t - 2$
if $i \leq n$ **então** $\quad 1$
retorne i
else
retorne 0

Total: $4t$ para encontrar o elemento na posição t.

Análise de complexidade de tempo no caso médio

Em média, a chance de um número estar em cada uma das posições se distribui igualmente.
(Isso pode variar, de acordo com a aplicação.)

A chance de encontrar o elemento k em cada posição do vetor é $\frac{1}{n}$.

O número de intruções executadas para solução é em média $(\sum_{t=1}^{n} 4t) \times \frac{1}{n} = 4\frac{(n+1)n}{2} \times \frac{1}{n} = 2n + 2$.

Análise de complexidade de tempo no caso médio

Então, no caso médio do Algoritmo Busca Linear, o número de instruções executadas é $2n + 2$.

$$2n + 2 \leq 2n + n = 3n, \forall n \geq 2$$

Então, $2n + 2 \in O(n)$.

$$2n + 2 \geq 2n, \forall n \geq 0$$

Então, $2n + 2 \in \Omega(n)$.

Portanto, o Algoritmo Busca Linear executa em tempo $\Theta(n)$ no caso médio, considerando que o elemento procurado está no vetor.

Análise de complexidade de tempo no caso médio

Portanto, o Algoritmo Busca Linear executa em tempo $\Theta(n)$ no caso médio, considerando que o elemento procurado está no vetor.

Observação: Nos demais casos, o elemento não está no vetor (é o pior caso).
Pela análise de pior caso, sabemos que quando o elemento não está no vetor, a execução é em tempo $\Theta(n)$.

Portanto, o Algoritmo Busca Linear executa em tempo $\Theta(n)$ no caso médio, considerando que o elemento procurado pode não estar no vetor.

Análise de complexidade: busca binária

Se o vetor estiver ordenado, podemos melhorar o tempo da busca?

Análise de complexidade: busca binária

Busca Binária

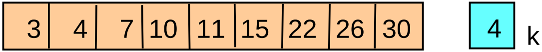
Problema

Dado um vetor ordenado $v[1..n]$ e um elemento k, responder em qual posição do vetor encontra-se o elemento k ou retornar 0.

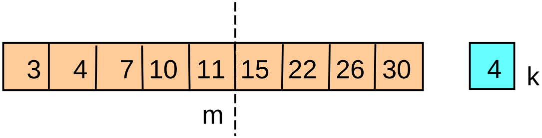
Ideia:

1. Dividir o vetor ao meio;
2. Seja m a posição do meio do vetor.
3. Se $k \leq v[m]$, procurar o elemento k em $v[1..m]$.
4. Senão, procurar o elemento k em $v[m+1..n]$.

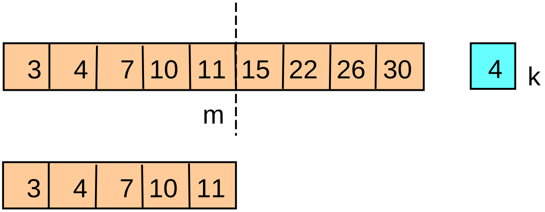
Análise de complexidade: busca binária



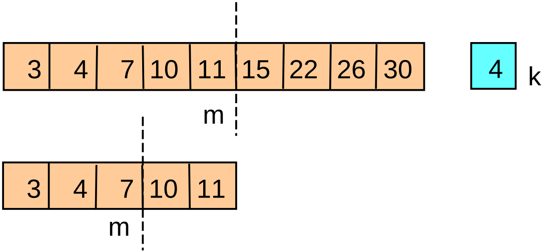
Análise de complexidade: busca binária



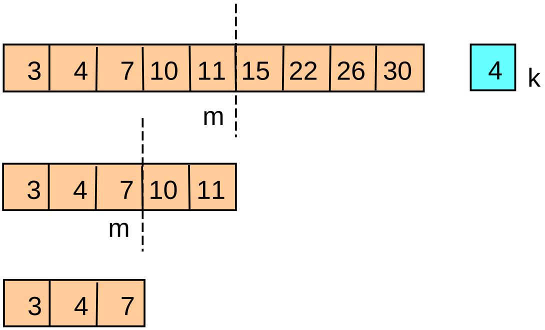
Análise de complexidade: busca binária



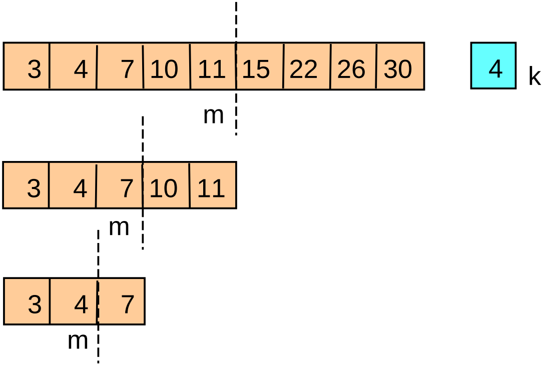
Análise de complexidade: busca binária



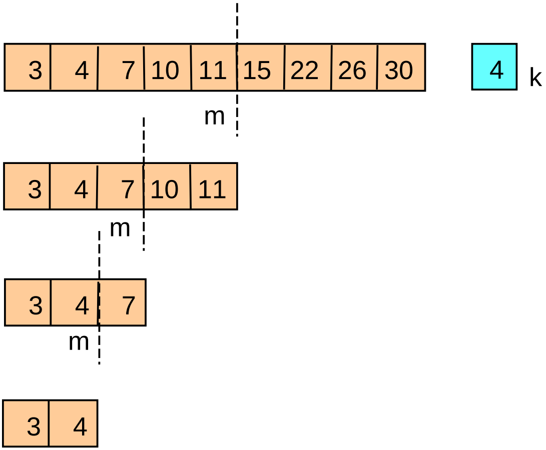
Análise de complexidade: busca binária



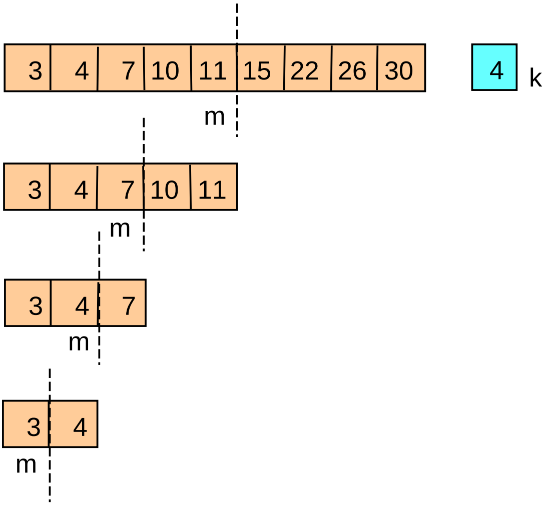
Análise de complexidade: busca binária



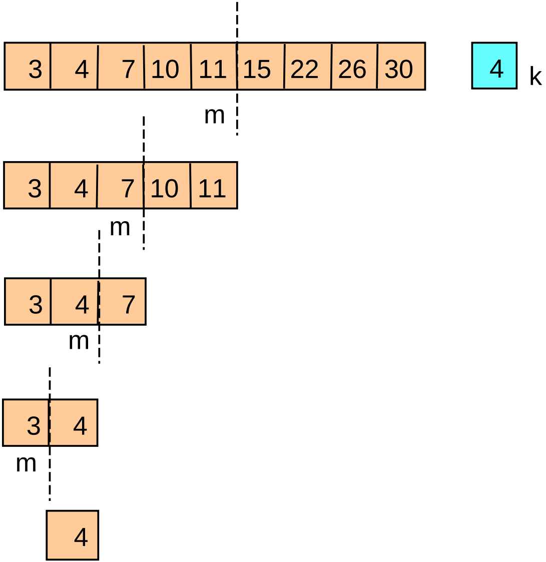
Análise de complexidade: busca binária



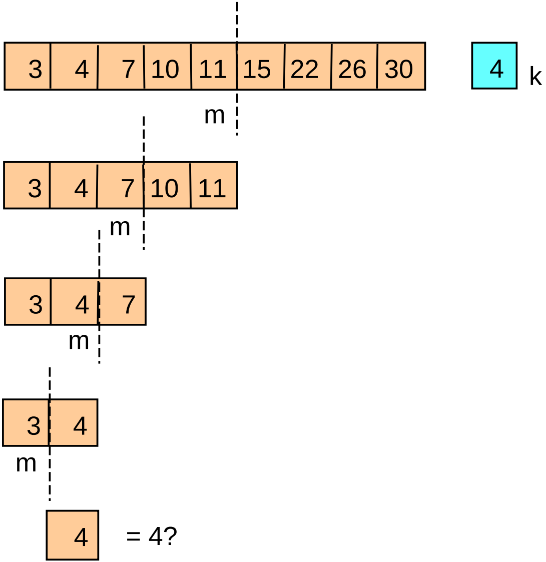
Análise de complexidade: busca binária

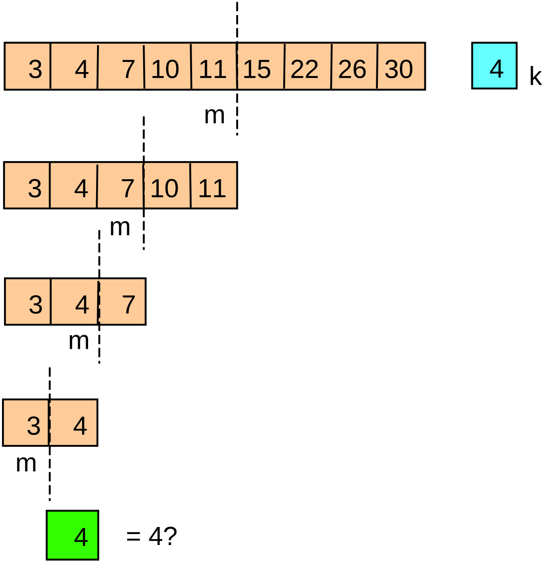


Análise de complexidade: busca binária

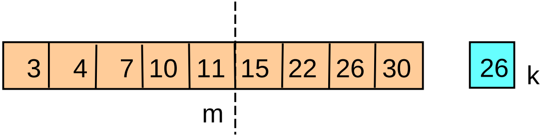


Análise de complexidade: busca binária

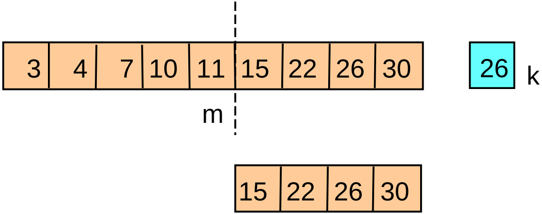



3
4
7
10
11
15
22
26
30
4
k
m
3
4
7
10
11
m
3
4
7
m
3
4
m
4
= 4?

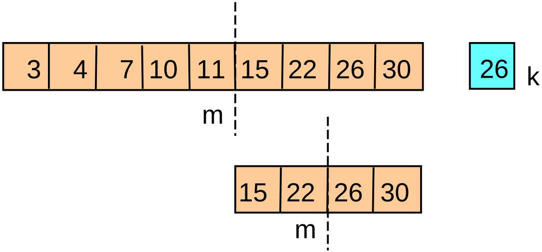
Análise de complexidade: busca binária



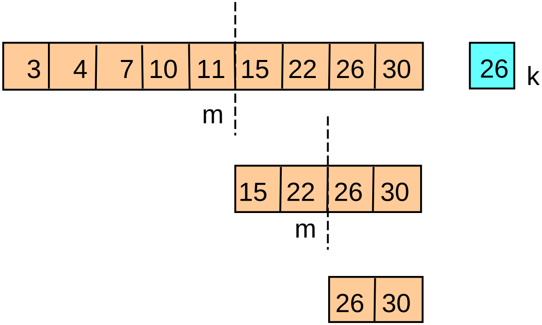
Análise de complexidade: busca binária



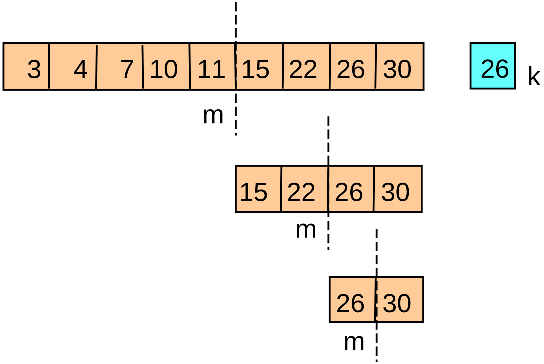
Análise de complexidade: busca binária



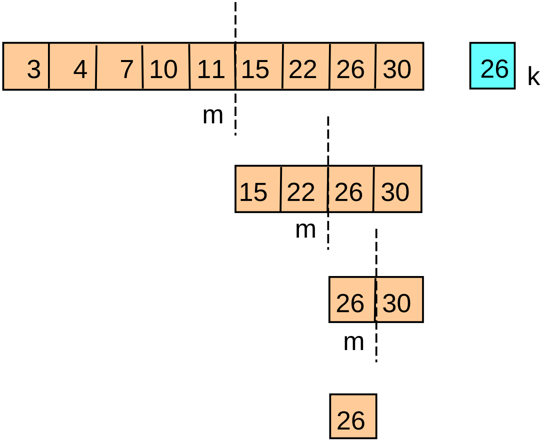
Análise de complexidade: busca binária



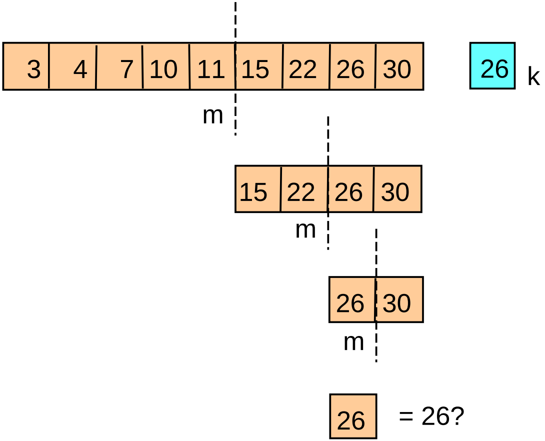
Análise de complexidade: busca binária



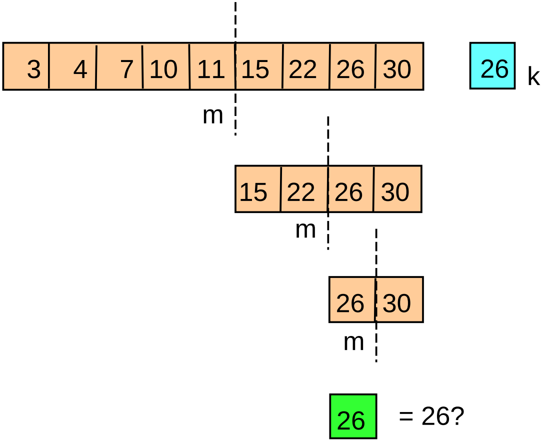
Análise de complexidade: busca binária



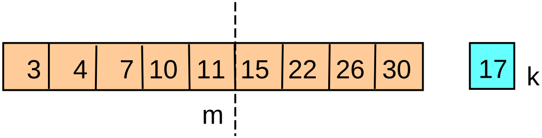
Análise de complexidade: busca binária



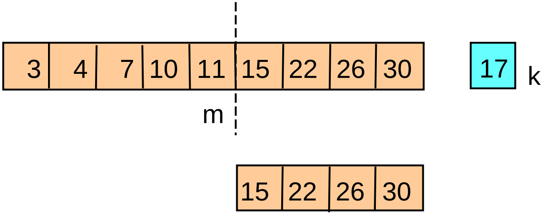
Análise de complexidade: busca binária



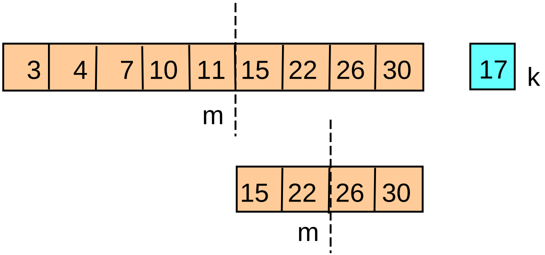
Análise de complexidade: busca binária



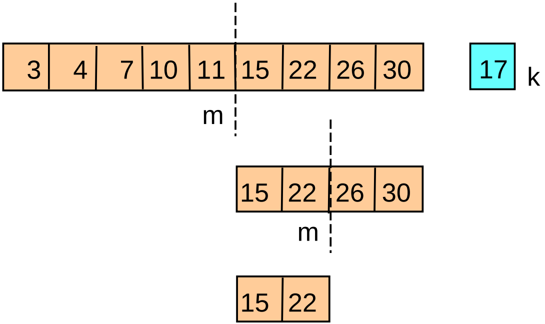
Análise de complexidade: busca binária



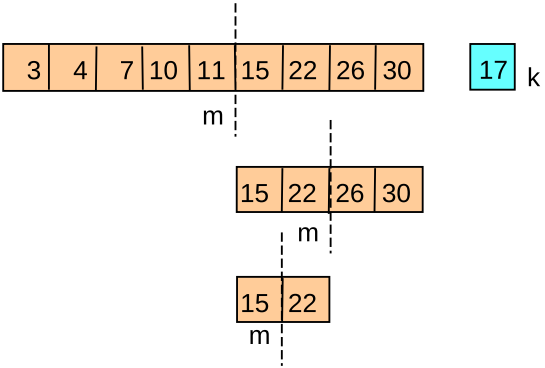
Análise de complexidade: busca binária



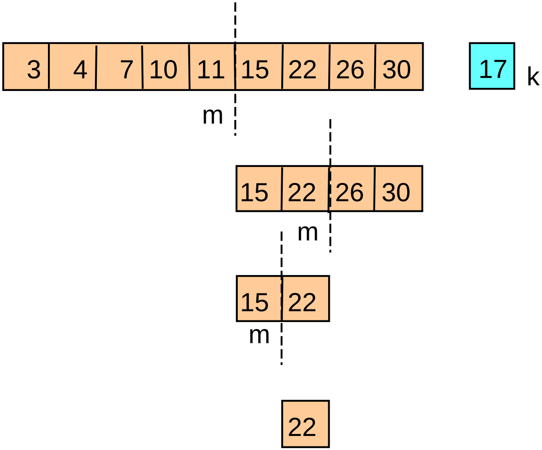
Análise de complexidade: busca binária



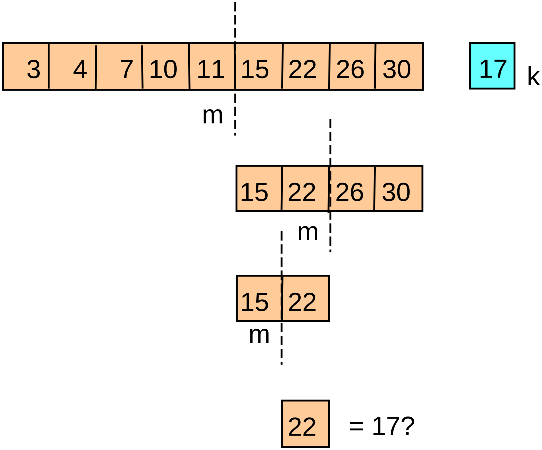
Análise de complexidade: busca binária

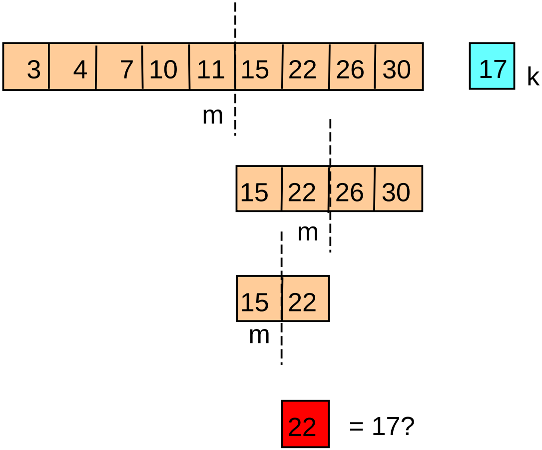


Análise de complexidade: busca binária



Análise de complexidade: busca binária




3
4
7
10
11
15
22
26
30
17
k
m
15
22
26
30
m
15
22
m
22
= 17?

Algoritmo Busca Binária

Entrada: vetor $v[1..n]$ de inteiros;

n número de elementos no vetor; k elemento procurado

$i \leftarrow 1$; $j \leftarrow n$;

enquanto $i < j$ **faça:**

$m \leftarrow \left\lfloor \frac{i+j}{2} \right\rfloor$

if $k \leq v[m]$ **então** $j \leftarrow m$

else $i \leftarrow m + 1$

if $i = j$ e $k = v[i]$ **então**

retorne i

else

retorne 0

Análise de complexidade: busca binária

No pior caso da busca, o elemento não está no vetor.

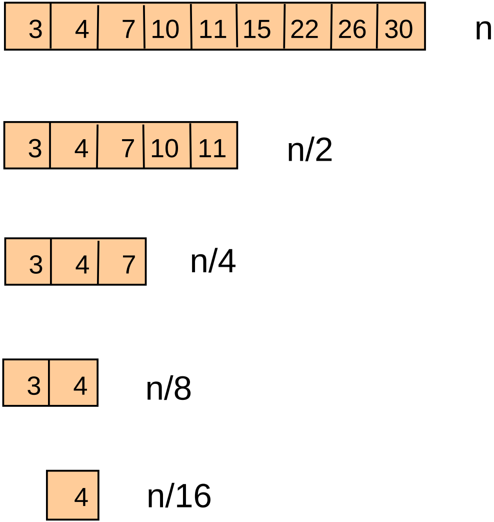
Quantas instruções de comparação serão executadas?

Algoritmo Busca Binária

Entrada: vetor $v[1..n]$ de inteiros;
n número de elementos no vetor; k elemento procurado

$i \leftarrow 1$; $j \leftarrow n$;
enquanto $i < j$ **faça:**
  $m \leftarrow \left\lfloor \frac{i+j}{2} \right\rfloor$
  if $k \leq v[m]$ **então** $j \leftarrow m$
  else $i \leftarrow m + 1$
if $i = j$ e $k = v[i]$ **então**
  retorne i
else
  retorne 0

Análise de complexidade: busca binária



Análise de complexidade: busca binária

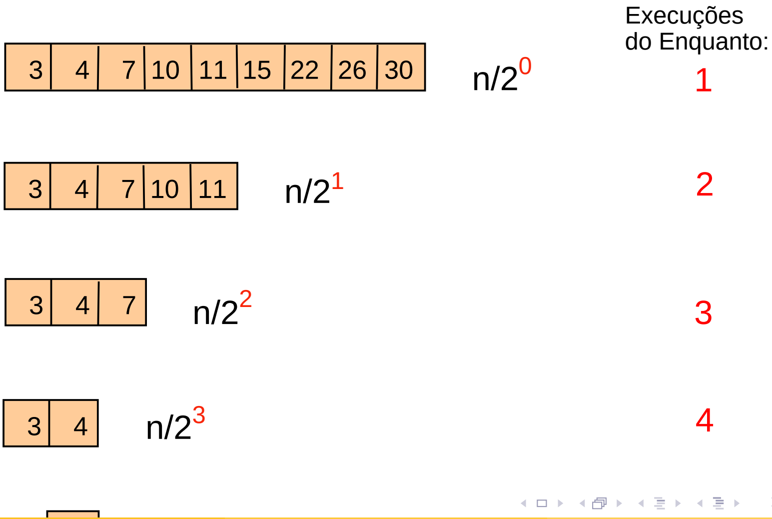
		Execuções do Enquanto:
3 | 4 | 7 | 10 | 11 | 15 | 22 | 26 | 30	$n/2^0$	1
3 | 4 | 7 | 10 | 11	$n/2^1$	2
3 | 4 | 7	$n/2^2$	3
3 | 4	$n/2^3$	4
4	$n/2^4$	5

Análise de complexidade: busca binária

Execuções do Enquanto:

3 | 4 | 7 | 10 | 11 | 15 | 22 | 26 | 30	$n/2^0$	1
3 | 4 | 7 | 10 | 11	$n/2^1$	2
3 | 4 | 7	$n/2^2$	3
3 | 4	$n/2^3$	4
4	$n/2^4$	5

O número de execuções do laço **enquanto** é 1 a mais que o expoente da respectiva potência que divide n.



Análise de complexidade: busca binária

Qual o valor do expoente quando o número de elementos do vetor é 1?

$\frac{n}{2^p} = 1.$

$n = 2^p.$

$p = \log n.$

Como o laço **enquanto** ocorre $p + 1$ vezes, o total de iterações será $(\log n) + 1.$

Algoritmo Busca Binária
Entrada: vetor $v[1..n]$ de inteiros;
n número de elementos no vetor; k elemento procurado

$i \leftarrow 1$; $j \leftarrow n$; 2
enquanto $i < j$ **faça:** $(\log n) + 1$
$m \leftarrow \left\lfloor \frac{i+j}{2} \right\rfloor$ $3 \log n$
if $k \leq v[m]$ **então** $j \leftarrow m$ $\log n$
else $i \leftarrow m + 1$ $2 \log n$
if $i = j$ e $k = v[i]$ **então** 2
retorne i
else
retorne 0

Total: $7 \log n + 5$.

Análise de complexidade: busca binária

No pior caso, a Busca Binária executa em tempo $\Theta(\log n)$:

$7 \log n + 5 \in O(\log n)$: $7 \log n + 5 \leq 7 \log n + 5 \log n = 12 \log n, \forall n \geq 2.$

$7 \log n + 5 \in \Omega(\log n)$: $7 \log n + 5 \geq 7 \log n, \forall n \geq 1.$

Análise de complexidade: busca binária

Qual o comportamento da Busca Binária no caso médio e no melhor caso?

A análise é diferente?

Análise de complexidade: busca binária

Qual o comportamento da Busca Binária no caso médio e no melhor caso?

A análise é diferente? Não! O algoritmo sempre divide o vetor até ter um único elemento.

Então a Busca Binária é sempre $\Theta(\log n)$.