Complexidade de algoritmos

Outline

Outline

1. Complexidade assintótica
   - Notações: $\mathbf{O}$, Ω e Θ

Outline

1. Complexidade assintótica
   - Notações: $\mathbf{O}$, Ω e Θ
   - Análise caso-a-caso

Outline

1. Complexidade assintótica
   - Notações: $\mathbf{O}$, Ω e Θ
   - Análise caso-a-caso
   - Padrões de crescimento de funções

Busca sequencial

```c
int sequencial (int v[], int chave, int tamanho) {
    for (int i = 0; i < tamanho; i++) {
        if (v[i] == chave) return i;
    }
    return -1;
}
```

Complexidade assintótica

- Princípios básicos

Complexidade assintótica

- Princípios básicos
  1. Modelo abstrato – implementação, linguagem, hardware

Complexidade assintótica

- Princípios básicos
  1. Modelo abstrato – implementação, linguagem, hardware
  2. Equivalência de instruções – atribuição, comparação, aritmética

Complexidade assintótica

- Princípios básicos
  1. Modelo abstrato – implementação, linguagem, hardware
  2. Equivalência de instruções – atribuição, comparação, aritmética
  3. Análise em função da entrada – tamanho, características

Busca sequencial

```c
int sequencial (int v[], int chave, int tamanho) {
    for (int i = 0; i < tamanho; i++) {
        if (v[i] == chave) return i;
    }
    return -1;
}
```

Busca sequencial

Algoritmo 1 sequencial (vetor v, chave c, tamanho n)

Busca sequencial

Algoritmo 2 sequencial (vetor v, chave c, tamanho n)

```
for (i ← 0; i < n; i++) do

end for
```

Busca sequencial

Algoritmo 3 sequencial (vetor v, chave c, tamanho n)

```
for (i ← 0; i < n; i++) do
    if v[i] = c then
        return i
    end if
end for
```

Busca sequencial

Algoritmo 4 sequencial (vetor v, chave c, tamanho n)

for (i $\leftarrow$ 0; i $<$ n; i$++$) **do**
  if v[i] $=$ c **then**
    return i
  end if
end for
return -1

1. Modelo abstrato

Busca sequencial

Algoritmo 1 sequencial (vetor v, chave c, tamanho n)

```
for (i ← 0; i < n; i++) do
    if v[i] = c then
        return i
    end if
end for
return -1
```

2. Equivalência de instruções

Busca sequencial

Algoritmo 1 sequencial (vetor v, chave c, tamanho n)

```
for (i ← 0; i < n; i++) do
    if v[i] = c then
        return i
    end if
end for
return -1
```

3 Análise caso-a-caso

Busca sequencial

Algoritmo 1 sequencial (vetor v, chave c, tamanho n)

for (i $\leftarrow$ 0; i $<$ n; i++) **do**
  if v[i] $=$ c **then**
    return i
  end if
end for
return -1

Análise caso-a-caso

- Melhor caso:

Busca sequencial

Algoritmo 1 sequencial (vetor v, chave c, tamanho n)

```
for (i ← 0; i < n; i++) do
    if v[i] = c then
        return i
    end if
end for
return -1
```

Análise caso-a-caso

- Melhor caso:

Análise caso-a-caso

- Melhor caso:
  - Laço: 3t

Análise caso-a-caso

- Melhor caso:
  - Laço: 3t
  - Fora do laço: t

Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 4t$
  - Laço: 3t
  - Fora do laço: t

Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 4t$
  - Laço: 3t
  - Fora do laço: t
- Pior caso:

Busca sequencial

Algoritmo 1 sequencial (vetor v, chave c, tamanho n)

```
for (i ← 0; i < n; i++) do
    if v[i] = c then
        return i
    end if
end for
return -1
```

Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 4t$
  - Laço: 3t
  - Fora do laço: t
- Pior caso:

Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 4t$
  - Laço: 3t
  - Fora do laço: t
- Pior caso:
  - Laço: $3n \cdot t$

Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 4t$
  - Laço: 3t
  - Fora do laço: t
- Pior caso:
  - Laço: $3n \cdot t$
  - Fora do laço: 3t

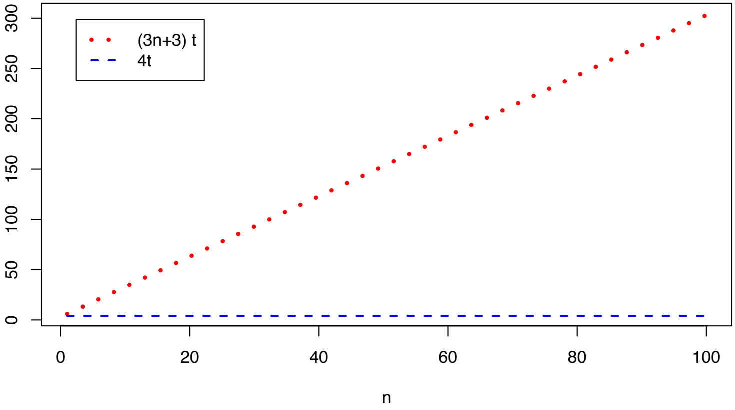
Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 4t$
  - Laço: 3t
  - Fora do laço: t
- Pior caso: $t_{\text{melhor}}(n) = 3n \cdot t + 3t$
  - Laço: $3n \cdot t$
  - Fora do laço: 3t

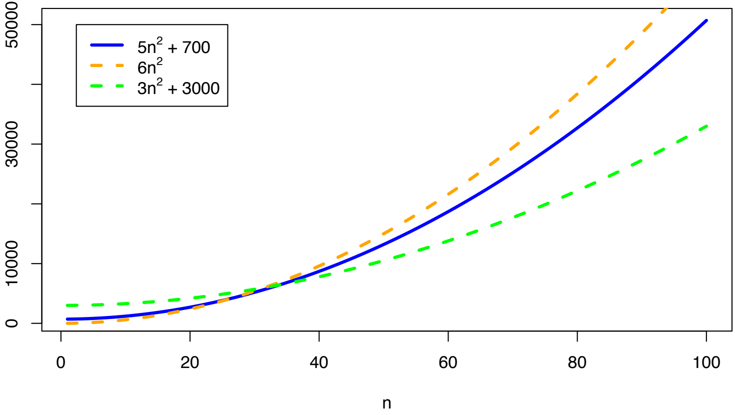
Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 4t$
  - Laço: 3t
  - Fora do laço: t
- Pior caso: $t_{\text{melhor}}(n) = 3n \cdot t + 3t = (3n+3) \cdot t$
  - Laço: 3n · t
  - Fora do laço: 3t

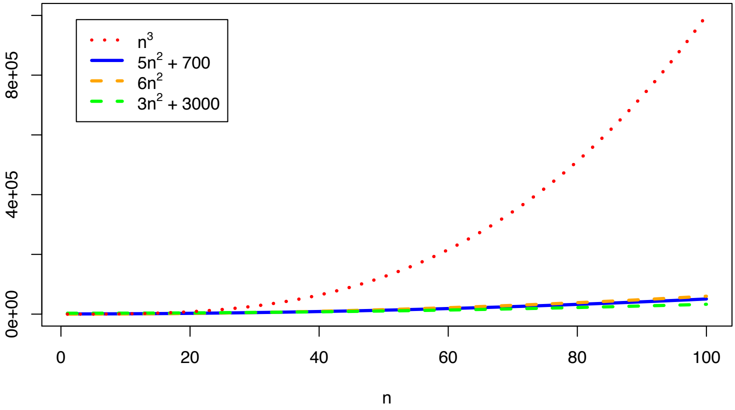
Complexidade assintótica



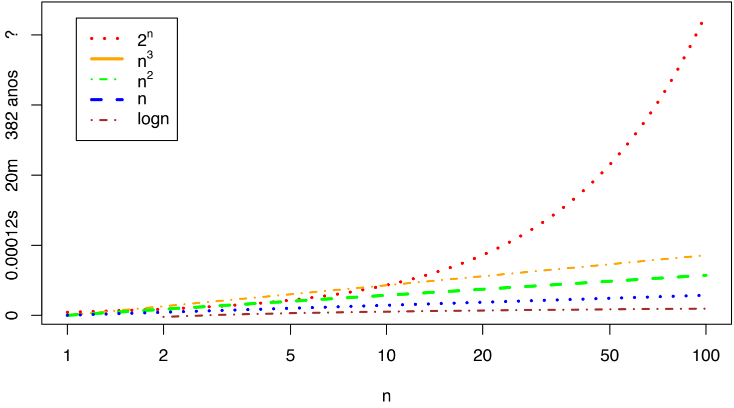
Complexidade assintótica



Complexidade assintótica



Complexidade assintótica



Notação **O** (limite superior)

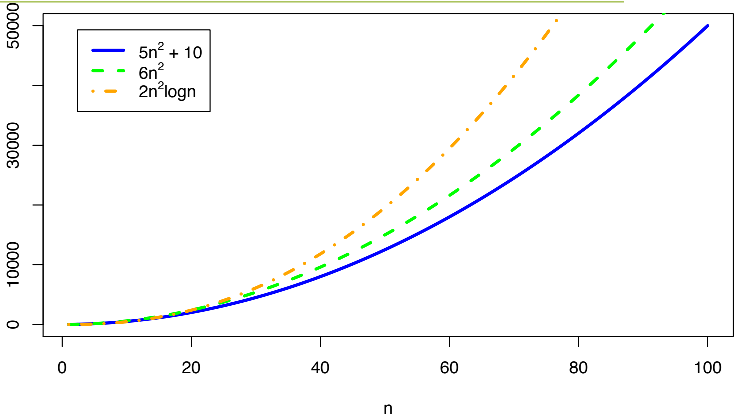


Figure: $f \in O(g) \Leftrightarrow f(n) \leq c \cdot g(n), \forall n > n_0$

Notação Ω (limite inferior)

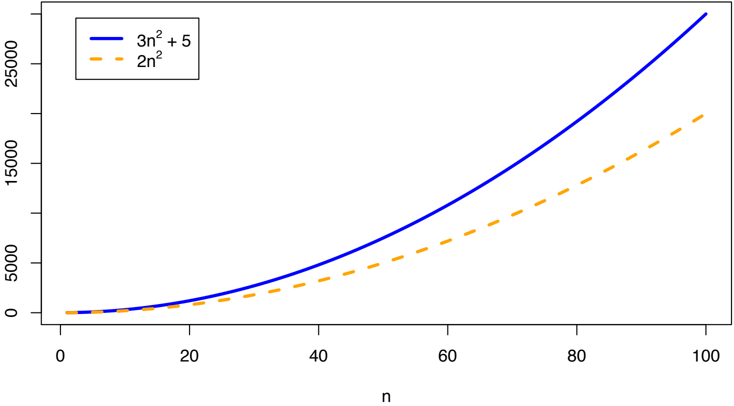


Figure: $f \in \Omega(g) \Leftrightarrow f(n) \geq c \cdot g(n),\ \forall n > n_0$

Notação Θ (crescimento equivalente)

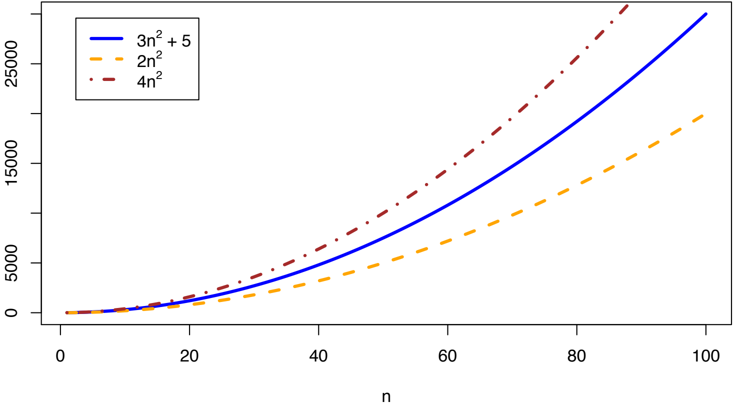


Figure: $f \in \Theta(g) \Leftrightarrow f \in O(g)$ e $f \in \Omega(g)$

Busca sequencial

Algoritmo 1 sequencial (vetor v, chave c, tamanho n)

```
for (i ← 0; i < n; i++) do
    if v[i] = c then
        return i
    end if
end for
return -1
```

Análise caso-a-caso

1. Melhor caso: $t_{\text{melhor}}(n) = 4$

Análise caso-a-caso

1. Melhor caso: $t_{\text{melhor}}(n) = 4$
   - $t_{\text{melhor}} \in O(n)$?

Análise caso-a-caso

1. Melhor caso: $t_{\text{melhor}}(n) = 4$
   - $t_{\text{melhor}} \in O(n)$?

     $c = 1, \quad t_{\text{melhor}} \leq 1 \cdot n, \quad \forall n > 3 \quad \Rightarrow t_{\text{melhor}} \in O(n)$ ✓

Análise caso-a-caso

1. Melhor caso: $t_{\text{melhor}}(n) = 4$
   - $t_{\text{melhor}} \in O(n)$?

     $c = 1, \quad t_{\text{melhor}} \leq 1 \cdot n, \quad \forall n > 3 \quad \Rightarrow t_{\text{melhor}} \in O(n)$ ✓
   - $t_{\text{melhor}} \in O(1)$?

Análise caso-a-caso

1. Melhor caso: $t_{\text{melhor}}(n) = 4$
   - $t_{\text{melhor}} \in O(n)$?

     $c = 1, \quad t_{\text{melhor}} \leq 1 \cdot n, \quad \forall n > 3 \quad \Rightarrow t_{\text{melhor}} \in O(n)$ ✓
   - $t_{\text{melhor}} \in O(1)$?

     $c = 4, \quad t_{\text{melhor}} \leq 4 \cdot 1, \quad \forall n > 0 \quad \Rightarrow t_{\text{melhor}} \in O(1)$ ✓

Análise caso-a-caso

1. Melhor caso: $t_{\text{melhor}}(n) = 4$
   - $t_{\text{melhor}} \in O(n)$?

     $c = 1, \quad t_{\text{melhor}} \leq 1 \cdot n, \quad \forall n > 3 \quad \Rightarrow t_{\text{melhor}} \in O(n)$ ✓
   - $t_{\text{melhor}} \in O(1)$?

     $c = 4, \quad t_{\text{melhor}} \leq 4 \cdot 1, \quad \forall n > 0 \quad \Rightarrow t_{\text{melhor}} \in O(1)$ ✓
   - $t_{\text{melhor}} \in \Omega(1)$?

Análise caso-a-caso

1. Melhor caso: $t_{\text{melhor}}(n) = 4$
   - $t_{\text{melhor}} \in O(n)$?

     $c = 1, \quad t_{\text{melhor}} \leq 1 \cdot n, \quad \forall n > 3 \quad \Rightarrow t_{\text{melhor}} \in O(n)$ ✓
   - $t_{\text{melhor}} \in O(1)$?

     $c = 4, \quad t_{\text{melhor}} \leq 4 \cdot 1, \quad \forall n > 0 \quad \Rightarrow t_{\text{melhor}} \in O(1)$ ✓
   - $t_{\text{melhor}} \in \Omega(1)$?

     $c = 1, \quad t_{\text{melhor}} \geq 1 \cdot 1, \quad \forall n > 0 \quad \Rightarrow t_{\text{melhor}} \in \Omega(1)$ ✓

Análise caso-a-caso

1. Melhor caso: $t_{\text{melhor}}(n) = 4$
   - $t_{\text{melhor}} \in O(n)$?

     $c = 1, \quad t_{\text{melhor}} \leq 1 \cdot n, \quad \forall n > 3 \quad \Rightarrow t_{\text{melhor}} \in O(n)$ ✓
   - $t_{\text{melhor}} \in O(1)$?

     $c = 4, \quad t_{\text{melhor}} \leq 4 \cdot 1, \quad \forall n > 0 \quad \Rightarrow t_{\text{melhor}} \in O(1)$ ✓
   - $t_{\text{melhor}} \in \Omega(1)$?

     $c = 1, \quad t_{\text{melhor}} \geq 1 \cdot 1, \quad \forall n > 0 \quad \Rightarrow t_{\text{melhor}} \in \Omega(1)$ ✓

$\Rightarrow t_{\text{melhor}} \in \Theta(1) \rightarrow$ Complexidade constante

Análise caso-a-caso

1. Pior caso: $t_{\text{pior}}(n) = 3n + 3$

Análise caso-a-caso

1. Pior caso: $t_{\text{pior}}(n) = 3n + 3$
   - $t_{\text{pior}} \in O(n)$?

Análise caso-a-caso

1. Pior caso: $t_{\text{pior}}(n) = 3n + 3$
   - $t_{\text{pior}} \in O(n)$?

     $c = 4, \quad t_{\text{pior}} \leq 4 \cdot n, \quad \forall n > 2 \quad \Rightarrow t_{\text{pior}} \in O(n)$ ✓

Análise caso-a-caso

1. Pior caso: $t_{\text{pior}}(n) = 3n + 3$
   - $t_{\text{pior}} \in O(n)$?

     $c = 4, \quad t_{\text{pior}} \leq 4 \cdot n, \quad \forall n > 2 \quad \Rightarrow t_{\text{pior}} \in O(n)$ ✓
   - $t_{\text{pior}} \in \Omega(n)$?

Análise caso-a-caso

1. Pior caso: $t_{\text{pior}}(n) = 3n + 3$
   - $t_{\text{pior}} \in O(n)$?

     $c = 4, \quad t_{\text{pior}} \leq 4 \cdot n, \quad \forall n > 2 \quad \Rightarrow t_{\text{pior}} \in O(n)$ ✓
   - $t_{\text{pior}} \in \Omega(n)$?

     $c = 3, \quad t_{\text{pior}} \geq 3 \cdot n, \quad \forall n > 0 \quad \Rightarrow t_{\text{pior}} \in \Omega(n)$ ✓

Análise caso-a-caso

1. Pior caso: $t_{\text{pior}}(n) = 3n + 3$

   - $t_{\text{pior}} \in O(n)$?

     $c = 4, \quad t_{\text{pior}} \leq 4 \cdot n, \quad \forall n > 2 \quad \Rightarrow t_{\text{pior}} \in O(n)$ ✓

   - $t_{\text{pior}} \in \Omega(n)$?

     $c = 3, \quad t_{\text{pior}} \geq 3 \cdot n, \quad \forall n > 0 \quad \Rightarrow t_{\text{pior}} \in \Omega(n)$ ✓

   $\Rightarrow t_{\text{pior}} \in \Theta(n) \rightarrow$ Complexidade linear

Busca binária

```c
int binaria (int v[], int chave, int tamanho) {
    int idx, inicio = 0, fim = tamanho;
    tamanho = fim - inicio;
    while (tamanho > 0) {
        idx = inicio + tamanho / 2;
        if (chave == v[idx]) { return idx; }
        else if (chave < v[idx]) { fim = idx; }
        else { inicio = idx + 1; }
        tamanho = fim - inicio;
    }
    return -1;
}
```

Algoritmo 2 binaria (vetor v, chave c, tamanho n)

Algoritmo 3 binaria (vetor v, chave c, tamanho n)

inicio $\leftarrow$ 0, fim $\leftarrow$ n, n $\leftarrow$ fim - inicio

Algoritmo 4 binaria (vetor v, chave c, tamanho n)

inicio $\leftarrow$ 0, fim $\leftarrow$ n, n $\leftarrow$ fim - inicio

while (n $>$ 0) **do**

end while

Algoritmo 5 binaria (vetor v, chave c, tamanho n)

```
inicio ← 0, fim ← n, n ← fim - inicio
while (n > 0) do
    idx ← inicio + n / 2
end while
```

Algoritmo 6 binaria (vetor v, chave c, tamanho n)

```
inicio ← 0, fim ← n, n ← fim - inicio
while (n > 0) do
    idx ← inicio + n / 2
    if (c = v[idx]) then
        return idx



end while
```

Algoritmo 7 binaria (vetor v, chave c, tamanho n)

```
inicio ← 0, fim ← n, n ← fim - inicio
while (n > 0) do
    idx ← inicio + n / 2
    if (c = v[idx]) then
        return idx
    else if (c < v[idx]) then
        fim ← idx



end while
```

Algoritmo 8 binaria (vetor v, chave c, tamanho n)

```
inicio ← 0, fim ← n, n ← fim - inicio
while (n > 0) do
    idx ← inicio + n / 2
    if (c = v[idx]) then
        return  idx
    else if (c < v[idx]) then
        fim ← idx
    else
        inicio ← idx + 1
    end if

end while
```

Algoritmo 9 binaria (vetor v, chave c, tamanho n)

```
inicio ← 0, fim ← n, n ← fim - inicio
while (n > 0) do
    idx ← inicio + n / 2
    if (c = v[idx]) then
        return idx
    else if (c < v[idx]) then
        fim ← idx
    else
        inicio ← idx + 1
    end if
    n ← fim - inicio
end while
```

Análise caso-a-caso

- Melhor caso:

Algoritmo 2 binaria (vetor v, chave c, tamanho n)

```
inicio ← 0, fim ← n, n ← fim - inicio
while (n > 0) do
    idx ← inicio + n / 2
    if (c = v[idx]) then
        return  idx
    else if (c < v[idx]) then
        fim ← idx
    else
        inicio ← idx + 1
    end if
    n ← fim - inicio
end while
```

Análise caso-a-caso

- Melhor caso:

Análise caso-a-caso

- Melhor caso:
  - Laço: 4t

Análise caso-a-caso

- Melhor caso:
  - Laço: 4t
  - Fora do laço: 3t

Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 7t$
  - Laço: 4t
  - Fora do laço: 3t

Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 7t$
  - Laço: 4t
  - Fora do laço: 3t
- Pior caso:

Algoritmo 2 binaria (vetor v, chave c, tamanho n)

```
inicio ← 0, fim ← n, n ← fim - inicio
while (n > 0) do
    idx ← inicio + n / 2
    if (c = v[idx]) then
        return idx
    else if (c < v[idx]) then
        fim ← idx
    else
        inicio ← idx + 1
    end if
    n ← fim - inicio
end while
```

Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 7t$
  - Laço: 4t
  - Fora do laço: 3t
- Pior caso:

Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 7t$
  - Laço: 4t
  - Fora do laço: 3t
- Pior caso:
  - Fora do laço: 4t

Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 7t$
  - Laço: 4t
  - Fora do laço: 3t
- Pior caso:
  - Fora do laço: 4t
  - Laço: 6t · ?

Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 7t$
  - Laço: 4t
  - Fora do laço: 3t
- Pior caso:
  - Fora do laço: 4t
  - Laço: 6t · ? $\longrightarrow$ $6t \cdot (\lceil log_2(n+1) \rceil + 1)$

Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 7t$
  - Laço: 4t
  - Fora do laço: 3t
- Pior caso:
  - Fora do laço: 4t
  - Laço: 6t · ? $\longrightarrow$ $6t \cdot (\lceil log_2(n+1) \rceil + 1)$

$$\longrightarrow \quad (6\lceil log_2(n+1) \rceil + 6)t$$

Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 7t$
  - Laço: 4t
  - Fora do laço: 3t
- Pior caso: $t_{\text{pior}}(n) = (6\lceil log_2(n+1)\rceil + 10)t$
  - Fora do laço: 4t
  - Laço: 6t · ? $\longrightarrow$ $6t \cdot (\lceil log_2(n+1)\rceil + 1)$

    $\longrightarrow$ $(6\lceil log_2(n+1)\rceil + 6)t$

Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 7t$
  - $t_{\text{melhor}}(n) \in \Theta(1)$

- Pior caso: $t_{\text{pior}}(n) = (6\lceil log_2(n+1)\rceil + 10)t$

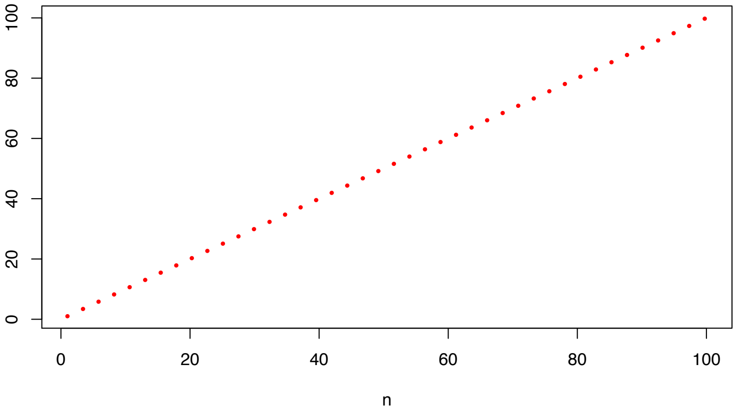
Análise caso-a-caso

- Melhor caso: $t_{\text{melhor}}(n) = 7t$
  - $t_{\text{melhor}}(n) \in \Theta(1)$
- Pior caso: $t_{\text{pior}}(n) = (6\lceil log_2(n+1)\rceil + 10)t$
  - $t_{\text{pior}}(n) \in \Theta(log\ n)$

Análise comparativa

	Sequencial	**Binária**
Melhor	$\Theta(1)$	$\Theta(1)$
Pior	$\Theta(n)$	$\Theta(log\ n)$

Complexidade assintótica



Padrões de crescimento

Função	Ordem
1	Constante
$log\ n$	Logarítmica
n	Linear
$n\ log\ n$	Logarítmica linear
n^2	Quadrática
n^3	Cúbica
2^n	Exponencial
$n!$	Fatorial

Padrões de crescimento