Instituto Federal de Educação, Ciência e Tecnologia do Estado do Ceará

IFCE

Estrutura de Dados

Aula 3

Prof.: Thiago Queiroz de Oliveira

Tempo de Execução de um Algoritmo

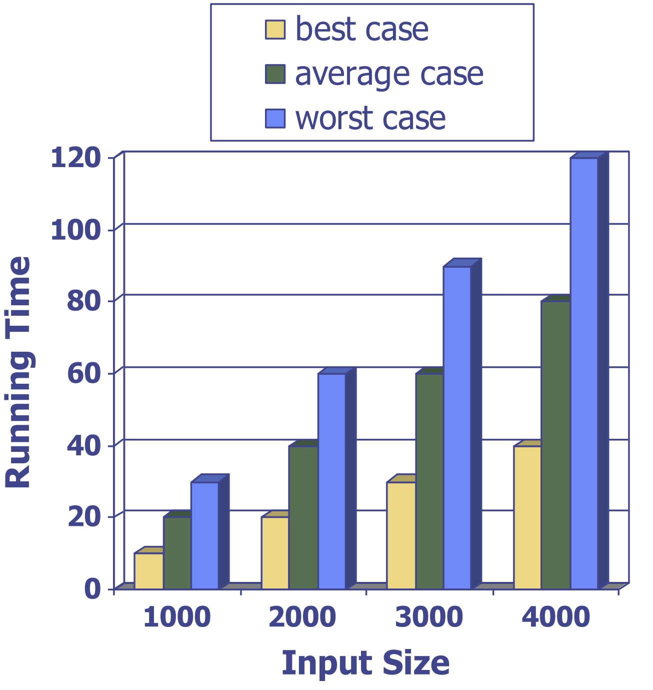
- Muitos algoritmos transformam uma entrada em uma saída.
- Geralmente o tempo de um algoritmo aumenta de acordo com o tamanho da entrada.
- Nosso estudo foca no pior caso.

- Fácil de determinar

- Melhorando o algoritmo

  para o pior caso,

  os demais são afetados.



Estudo Experimental

- Escrever o programa implementando o algoritmo.
- Executar o programa com diferentes entradas de dados.
- Usar o método como System.currentTimeMillis() para obter o tempo de execução do programa.

Limitação do Experimento

- É necessário implementar o algoritmo, que poderá demandar muito tempo.
- Resultados não indicam o tempo de execução de entradas não testadas.
- Para comparar dois programas, o mesmo ambiente de hardware e software devem ser usados.

Complexidade

Porquê o estudo da Complexidade?

- Escolher entre vários algoritmos o mais eficiente;
- Desenvolver algoritmos mais eficientes;
- Complexidade Computacional - torna possível determinar se a definição de determinado algoritmo é viável.

Algoritmos e Funções

Todo algoritmo define uma função matemática f(n)= # de instruções

Ex.:

```
void Algoritmo1(int n){
int contador = 0;
for(int i = 0; I < n; I++){
  contador++;
}
}
```

➔ n passos para terminar

Algoritmos e Funções

Todo algoritmo define uma função matemática f(n)= # de instruções

Ex.:

```
void Algoritmo2(int n){
int contador = 0;
for(int i = 0; i < n; i++){
 for(int j = 0; j < n; j++){
        contador++;
 }
}
}
```

➔ n*n passos para terminar

Algoritmos e Funções

	Algoritmo1	Algoritmo2	Algoritmo3
n = 1	1	1	1
n = 10	10	10^2	10^3
n = 1000	10^3	10^6	10^9
n = 10^6	10^6	10^{12}	10^{18}
	$f(n) = n$	$f(n) = n^2$	$f(n) = n^3$

Algoritmos e Funções

Qual algoritmo você escolheria?

R.: basta olhar na função definida por cada algoritmo

A função de um algoritmo ➔ melhor algoritmo para um dado problema

Fatores: tempo de processamento, memória, acessos a disco, largura de banda, entre outros

Algoritmos e Funções

A prática de se reduzir um algoritmo a uma função matemática denomina-se

“Análise de complexidade de algoritmos”

Funções mais Comuns

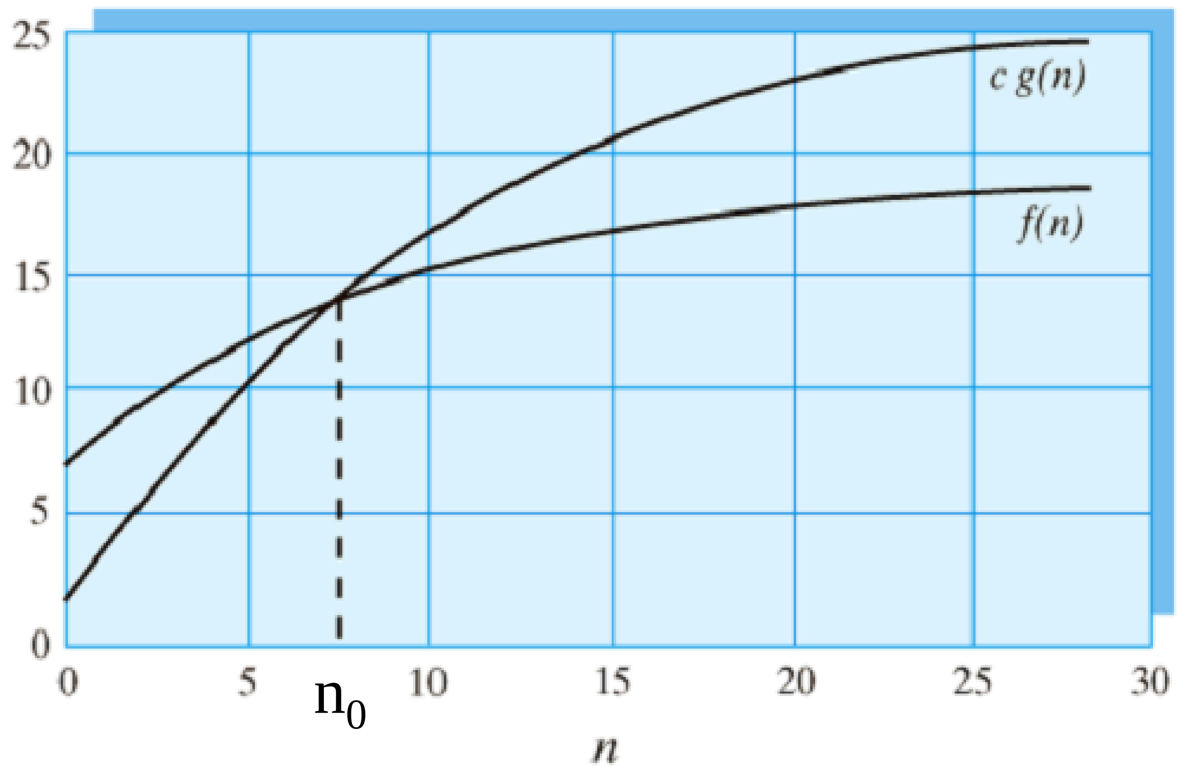
constante	logaritmo	linear	nlogn	quadrática	cúbica	exponencial
1	logn	n	nlogn	n^2	n^3	a^n

n	$\log n$	n	$n \log n$	n^2	n^3	2^n
8	3	8	24	64	512	256
16	4	16	64	256	4.096	65.536
32	5	32	160	1.024	32.768	4.294.967.296
64	6	64	384	4.096	262.144	$1{,}84 \times 10^{19}$
128	7	128	896	16.384	2.097.152	$3{,}40 \times 10^{38}$
256	8	256	2.048	65.536	16.777.216	$1{,}15 \times 10^{77}$
512	9	512	4.608	262.144	134.217.728	$1{,}34 \times 10^{154}$

Notação Assíntotica

1) Limite assintótico superior: O(g(n))

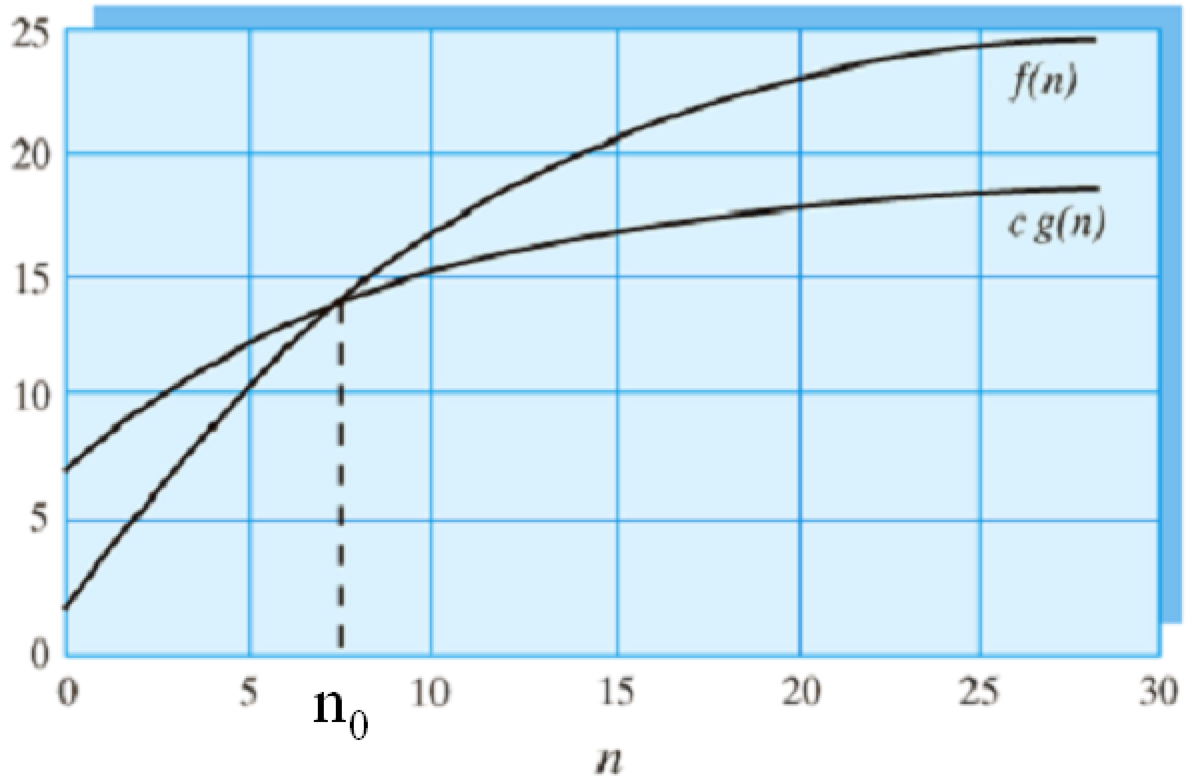
Uma função $f(n) = O(g(n))$, se $f(n) \leq c_1 * g(n)$ para uma constante $c_1 > 0$ e para $n \geq n_0$



Notação Assíntotica

2) Limite assintótico inferior: $\Omega(g(n))$

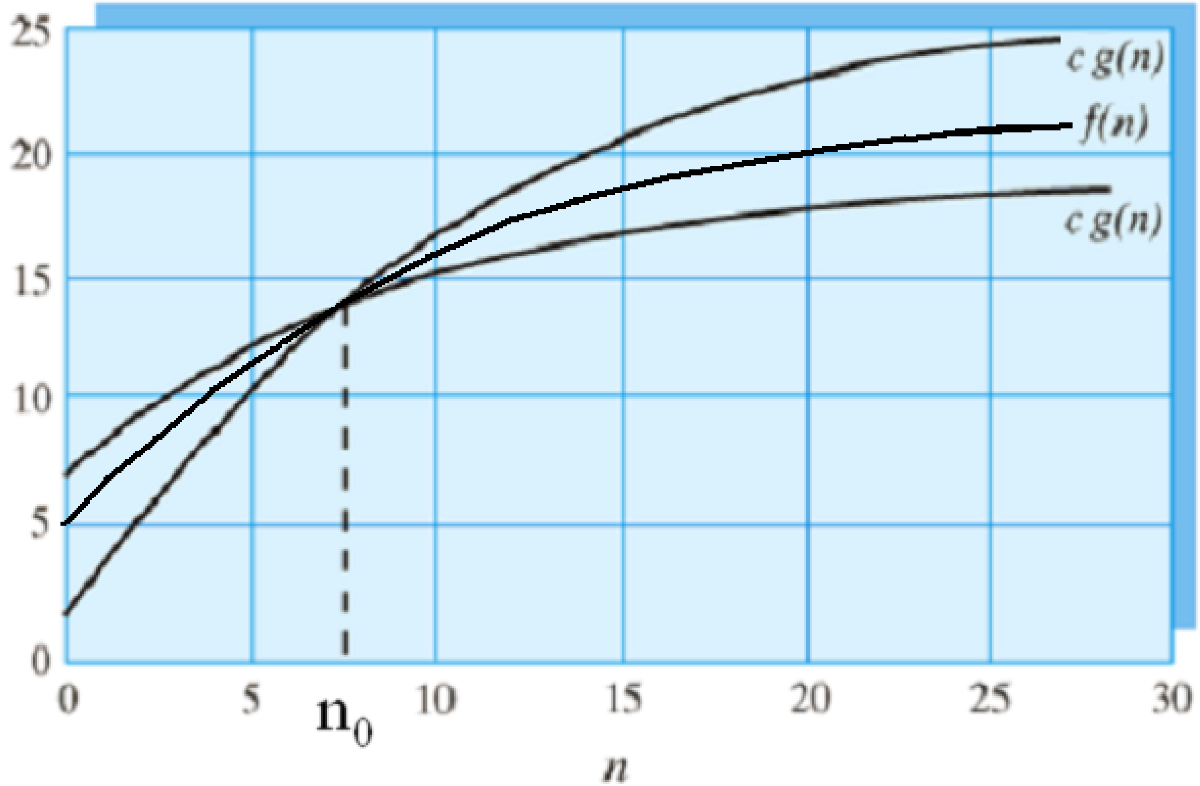
Uma função $f(n) = \Omega(g(n))$, se $f(n) \geq c_1 * g(n)$ para uma constante $c_1 > 0$ e para $n \geq n_0$



Notação Assíntotica

3) Limite assintótico estrito: $\theta(g(n))$

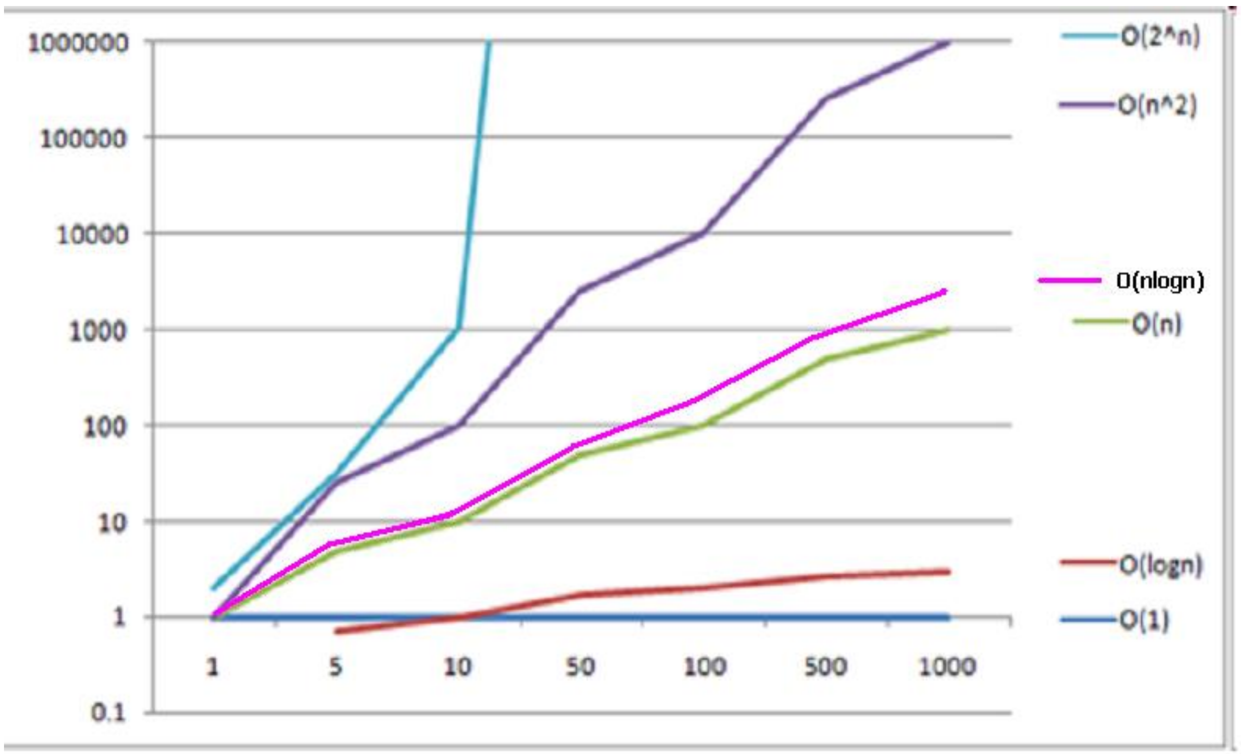
Uma função $f(n) = \theta(g(n))$, se $c_1 * g(n) \leq f(n) \leq c_2 * g(n)$ para constantes c_1 e $c_2 > 0$ e para $n \geq n_0$



Notação Assíntotica

Principais limites assintóticos:

$O(1) < O(\log n) < O(n) < O(n\log n) < O(n^2) < O(2^n)$



Exemplo

Suponha o algoritmo para calcular o somatório de números:

```
AlgForcaBruta(int n){
soma = 0                      → 1
for(i = 1 to n)               → n
 soma = soma + i              → n
}
```

- ➔ Custo total: $1 + n + n = 2n + 1$
- ➔ $f(n) = 2n + 1$

Exemplo

Suponha o algoritmo para calcular o somatório de números:

```
AlgInstataneo(int n){
 soma = n*(n+1)/2          → 4
}
```

➔ Custo total: 4

➔ $f(n) = 4$

Exemplo

Suponha o algoritmo para calcular o somatório de números:

```
AlgBobo(int n){
 soma = 0                     → 1
 for(i = 0 to n){             → n
   for(j = 1 to i){           → n((n+1)/2)
      soma = soma + 1         → n((n+1)/2)
   }
 }
}
```

➔ Custo total: $1 + n + n((n+1)/2) + n((n+1)/2)$

➔ $f(n) = n^2 + n + 1$

Exemplo

Resultados até então:

AlgForcaBruta: $f(n) = 2n + 1$
AlgInstantaneo: $f(n) = 4$
AlgBobo: $f(n) = n^2 + n + 1$

Resultados – Notação Assíntotica

O AlgForcaBruta tem complexidade
$f(n) = \theta(g(n)) = \theta(n)$, isto é, $g(n) = n$

Mas por quê, se $f(n) = 2n + 1$?
R.: segundo a definição de limite assintótico estrito $c1 * g(n) \leq f(n) \leq c2 * g(n)$ temos:

$$1 * n \leq f(n) = 2n + 1 \leq 3 * n$$

Resultados – Notação Assíntotica

O AlgInstantaneo tem complexidade
$f(n) = \theta(g(n)) = \theta(1)$, isto é, $g(n) = 1$

Mas por quê, se $f(n) = 4$?
R.: segundo a definição de limite assintótico estrito $c1 * g(n) \leq f(n) \leq c2 * g(n)$ temos:

$$3*1 \leq f(n) = 4 \leq 5*1$$

Resultados – Notação Assíntotica

O AlgBobo tem complexidade

$f(n) = \theta(g(n)) = \theta(n^2)$, isto é, $g(n) = n^2$

Mas por quê, se $f(n) = n^2 + n + 1$?

R.: segundo a definição de limite assintótico estrito $c1*g(n) \leq f(n) \leq c2*g(n)$ temos:

$$1* n^2 \leq f(n) = n^2 + n + 1 \leq 2* n^2$$

Resultados – Notação Assíntotica

O que temos então:

AlgForcaBruta: $f(n) = 2n + 1 = \theta(n)$

AlgInstantaneo: $f(n) = 4 = \theta(1)$

AlgBobo: $f(n) = n^2 + n + 1 = \theta(n^2)$

Exercícios

Utilizando as definições para as notações assintóticas, prove se são verdadeiras ou falsas as seguintes afirmativas:

1) $3n^3 + 2n^2 + n + 1 = O(n^3)$

2) $7n^2 = O(n)$

3) $2^{n+2} = O(2^n)$

4) $2^{2n} = O(2^n)$

5) $5n^2 + 7n = \Theta(n^2)$

6) $6n^3 + 5n^2 \neq \Theta(n^2)$

7) $9n^3 + 3n = \Omega(n)$

Exercícios

Utilizando as definições para as notações assintóticas, prove se são verdadeiras ou falsas as seguintes afirmativas:

1) $3n^3 + 2n^2 + n + 1 = O(n^3)$

2) $7n^2 = O(n)$

3) $2^{n+2} = O(2^n)$

4) $2^{2n} = O(2^n)$

5) $5n^2 + 7n = \Theta(n^2)$

6) $6n^3 + 5n^2 \neq \Theta(n^2)$

7) $9n^3 + 3n = \Omega(n)$