Estruturas de Dados II

Análise empírica e assintótica

Profª. Juliana de Santi
Prof. Rodrigo Minetto
Universidade Tecnológica Federal do Paraná
Material compilado de: Cormen, IC-UNICAMP e IME-USP

Sumário

Análise de algoritmos

Analisar um algoritmo significa **prever** os **recursos** que o algoritmo necessitará.

Recurso: memória, banda passante, hardware, **tempo de computação/complexidade de tempo**.

Ao analisar algoritmos podemos considerar ou descartar algoritmos candidatos para um problema.

Análise empírica e análise assintótica

Sumário

Análise empírica de algoritmos

Executa-se o algoritmo sobre amostras para obter tempo, número de operações, ..., e verificar a eficiência desse algoritmo.

É dependente do ambiente de teste utilizado (maquina, compilador, sistema, carga, ...).

Sumário

Análise assintótica de algoritmos

Para **aferir** o tempo de execução de forma independente do computador utilizado, da linguagem, compiladores, condições locais de processamento,..., utiliza-se a **análise assintótica**/matemática.

Através da análise assintótica pode-se determinar uma **expressão matemática que traduza o comportamento de tempo** do algoritmo. Com isso, tem-se a indicação da eficiência antes de qualquer investimento em desenvolvimento.

Análise de algoritmos

Qual o custo do trecho de código abaixo?

```
...
for(i = 0; i < n; i++) {
   a[i] = b[i] + c[i];
}
for(i = 0; i < n; i++) {
   d[i] = a[i]*(e[i] + 1.0);
}
...
```

Análise de algoritmos

Qual o custo do trecho de código abaixo?

```
...
for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++) {
        a[i][j] = 0.0;
    }
}
for (i = 0; i < n; i++) {
    a[i][i] = 1.0;
}
...
```

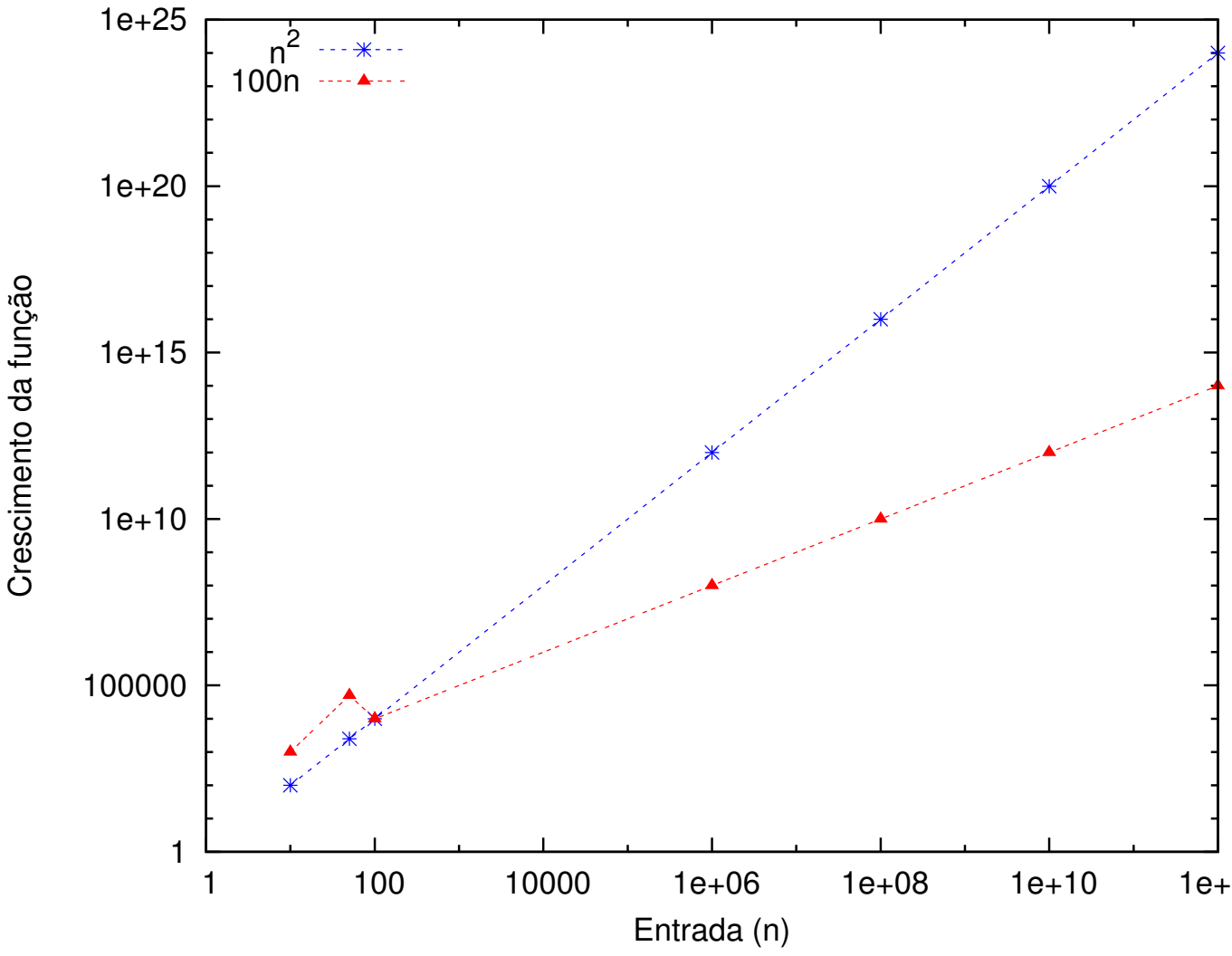
Análise assintótica de algoritmos

A análise assintótica está relacionada à velocidade de crescimento (ordem) de uma função.

Preocupação em **como o tempo** de execução **cresce com** um tamanho de **entrada muito grande**, ou seja, a entrada é a variável em relação a qual a expressão matemática avaliará o tempo de execução do algoritmo.

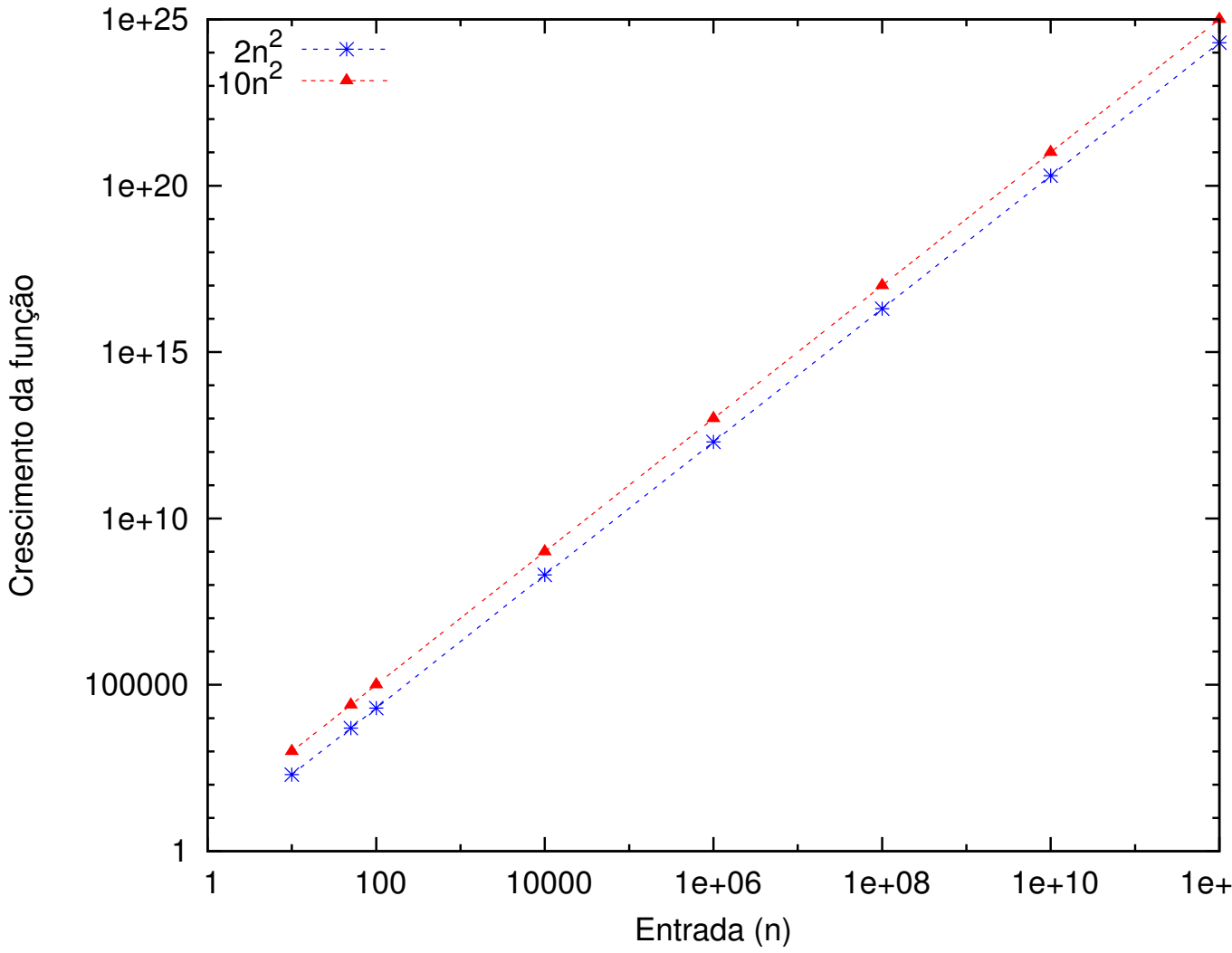
Análise assintótica de algoritmos

Despreza valores pequenos do argumento, ou termos de **menor ordem**, (n^2 cresce mais rápido do que $100n$, embora n^2 seja menor que $100n$ quando n é pequeno).



Análise assintótica de algoritmos

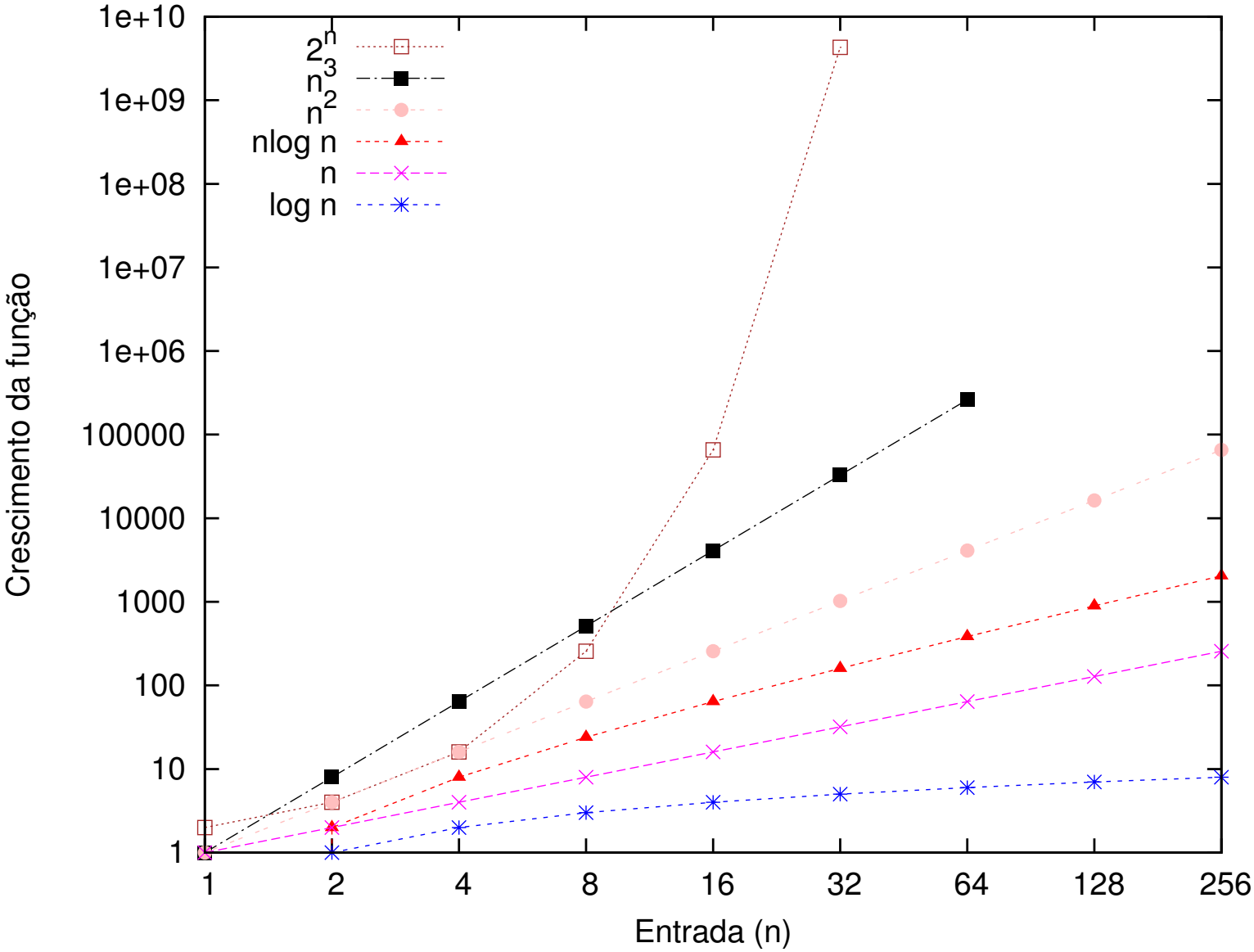
Despreza fatores multiplicativos ($\mathbf{2}n^2$ cresce tão rápido quanto $\mathbf{10}n^2$).



Análise assintótica de algoritmos

Quando observamos tamanhos de entradas grandes o suficiente para tornar **relevante apenas a ordem de crescimento do tempo** de execução, estamos estudando a **eficiência assintótica** dos algoritmos — Algoritmos. 2 edição. Cormen et al.

Análise assintótica de algoritmos



Algoritmos e tecnologia

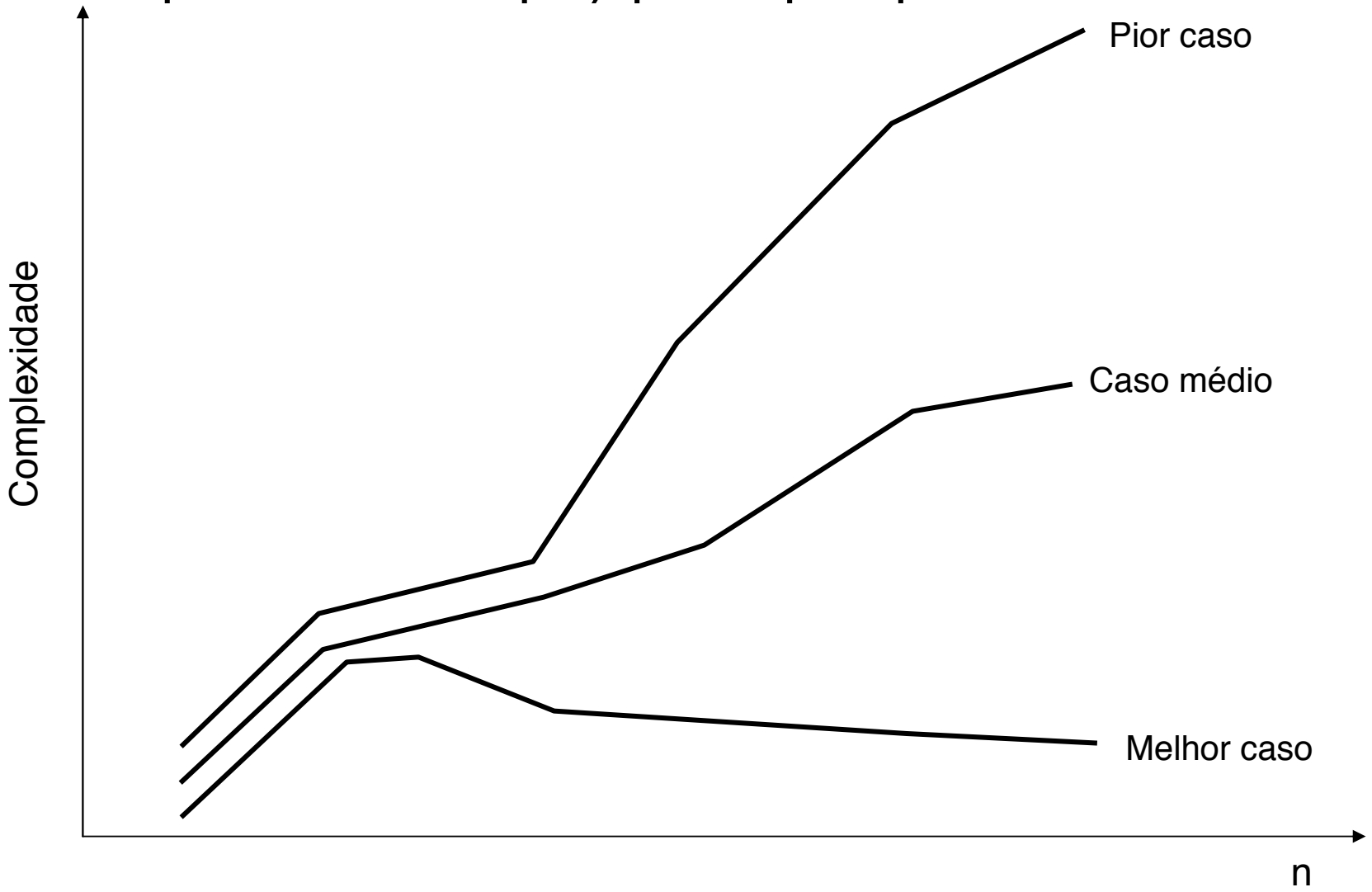
Considere 5 algoritmos com as complexidades de tempo. Suponha que uma operação leve 1 ms.

n	n	$nlogn$	n^2	n^3	2^n
16	0.016s	0.064s	0.256s	4s	1m 4s
32	0.032s	0.16s	1s	33s	46 dias
512	0.512s	9s	4m 22s	1 dia 13h	10137 séc.

Para uma máquina mais rápida onde uma operação leve 1 ps ao invés de 1 ms, precisariamos 10128 séculos ao invés de 10137 séculos :-p

Medida de complexidade e eficiência de algoritmos

Interessa-nos o pior caso, i.e., o maior número de operações (limite superior de tempo) para qualquer entrada n.



Sumário

Notação assintótica

As notações que usamos para descrever o tempo de execução assintótica de um algoritmo são definidas em termos das seguintes funções:

- O - limite assintótico superior;
- Ω - limite assintótico inferior;
- Θ - limite assintótico restrito.

Notação O (Limite assintótico superior)

Quando afirmamos que "o tempo de execução do algoritmo é $O(n^2)$", equivale a dizer que o tempo de execução do pior caso é $O(n^2)$, considerando a entrada mais desfavorável.

Quando nos referimos a complexidade, no geral, estamos nos referindo ao **pior caso**.

Exemplos de funções $O(n^2)$

- $f(n) = n^2$
- $f(n) = n^2 + n$
- $f(n) = n^2 - n$
- $f(n) = 1000n^2 + 1000n$
- $f(n) = n$
- $f(n) = n/1000$
- $f(n) = n^{1.999999}$
- $f(n) = logn$
- $f(n) = \sqrt{n}$

Notação Ω (limite assintótico inferior)

Considerando que a notação Ω descreve um limite inferior, quando a usamos para limitar o tempo de execução do **melhor caso** de um algoritmo, por implicação também limitamos o tempo de execução do algoritmo sobre entradas arbitrárias.

Exemplos de funções $\Omega(n^2)$

- $f(n) = n^2$
- $f(n) = n^2 + n$
- $f(n) = n^2 - n$
- $f(n) = 1000n^2 + 1000n$
- $f(n) = 1000n^2 - 1000n$
- $f(n) = n^3$
- $f(n) = n^{2.0000001}$
- $f(n) = n^2 log_2 log_2 log_2 n$
- $f(n) = 2^n$

Notação Θ (limite assintótico restrito)

Dadas duas funções $f(n)$ e $g(n)$, temos $f(n) = \Theta(g(n))$ se e somente se

- $f(n) = O(g(n))$
- $f(n) = \Omega(g(n))$