Notação de Operações

Durante a análise de complexidade de um algoritmo buscamos avaliar o seu desempenho de processamento e tempo estimado de execução, no entanto como cada dispositivo, linguagem escolhida, modelo de execução (interpretado, compilador) entre outros fatores pode impactar nessa avaliação, devemos adotar um método que possa descrever o comportamento do algoritmo a medida que quantidade de entradas 'n' se torna maior. Para realizar essa projeção de comportamento, utilizamos das Notações Assintóticas O, Ω e Θ, a seguir definimos o que é uma notação assintótica e cada uma das mencionadas anteriormente.

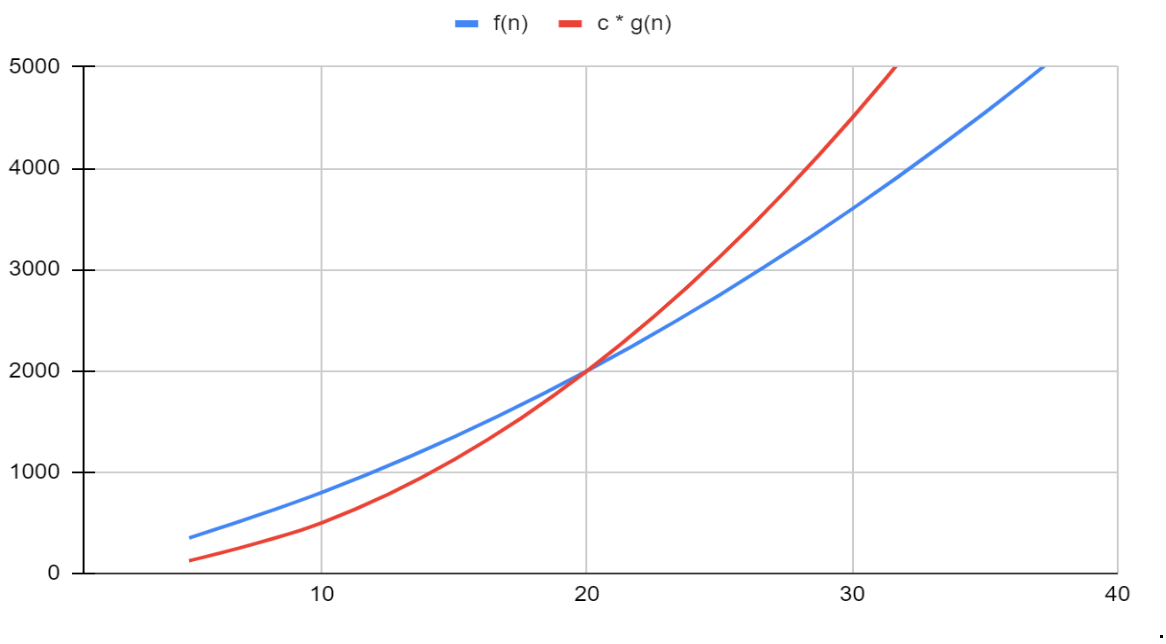
Uma notação assintótica é uma representação matemática que nos permite entender o padrão de crescimento de uma função, focando no quão rápido ela cresce dado entradas maiores, evidenciando o termo de maior relevância para o seu crescimento, por exemplo, a medida que n se torna maior na função $T(n) = 3n^3+10n$, o termo n^3 passa a valer mais que os demais a ponto de torná-los irrelevante para o cálculo. Podemos dizer, portanto, que T(n) possui ordem de crescimento igual a ‘n^3’, ou $O(n^3)$.

A notação Big-O - O(g(n)) - define o limite assintótico superior para uma função, representando o pior caso de desempenho do algoritmo analisado, e pode ser definida como:

Seja as funções $f : N \rightarrow N$ e $g : N \rightarrow N$, dizemos que $f\ (n)\ \in\ O(g(n))$ se existem duas constantes m e c positivas tais que:

$$f(n)\ \leq \mathrm{c}\ \times\ g(n)\ \ \forall\, n\ \geq\ m$$

No gráfico abaixo podemos observar essa propriedade para uma função com ordem $O(n^2)$, onde a partir do valor 20 de entrada ‘n’, a f(n) sempre se apresentará abaixo de c * g(n).



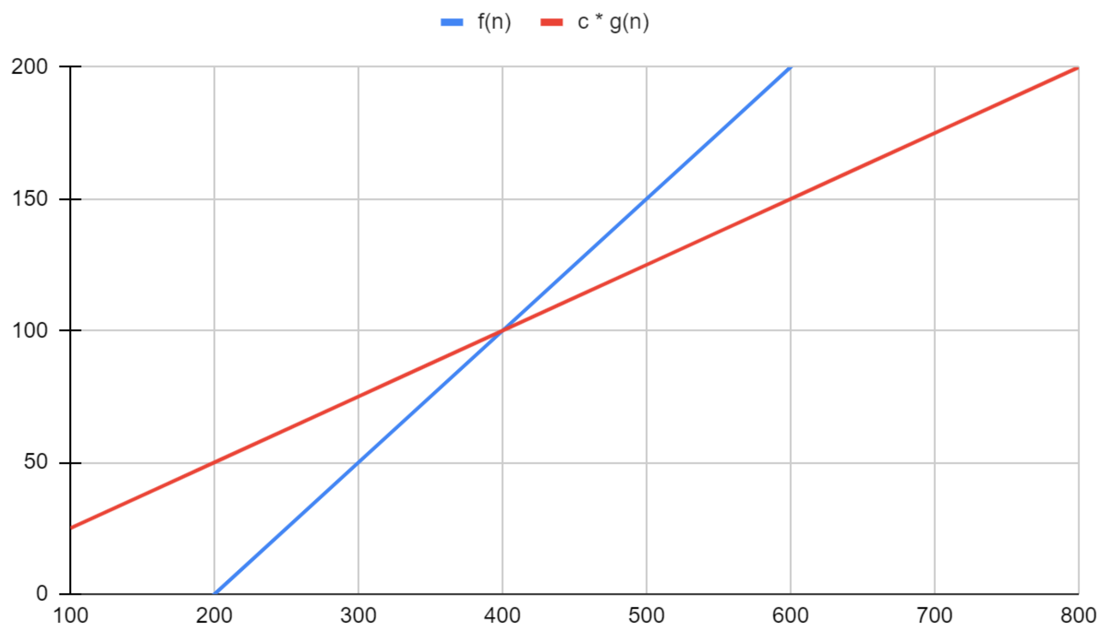
.

A notação Omega - Ω(g(n)) - define o limite assintótico inferior para uma função, representando o melhor caso de desempenho do algoritmo analisado, e pode ser definida como:

Seja as funções $f : N \rightarrow N$ e $g : N \rightarrow N$, dizemos que $f\ (n)\ \in\ \Omega(g(n))$ se existem duas constantes m e c positivas tais que:

$$f(n)\ \geq \mathrm{c}\ \times\ g(n)\ \ \forall\, n\ >\ m$$

No gráfico abaixo podemos observar essa propriedade para uma função com ordem Ω(n), onde a partir do valor 400 de entrada 'n', a f(n) sempre se apresentará acima de g.

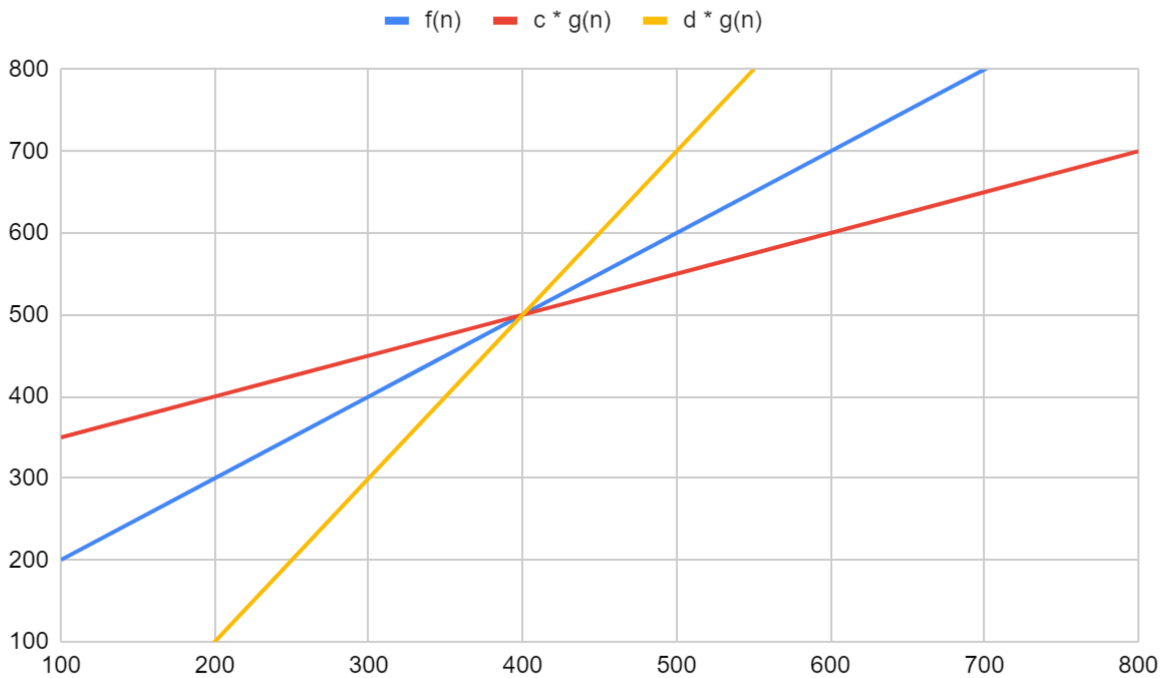


Por fim, a notação Theta - Θ(g(n)) - define o limite assintótico firme para uma função, representando o caso médio de desempenho do algoritmo analisado, e pode ser definida como:

Seja as funções $f : N \rightarrow N$ e $g : N \rightarrow N$, dizemos que $f\ (n)\ \in\ O(g(n))$ se existem três constantes m, c e d positivas tais que:

$$\mathrm{c}\ \times\ g(n) \leq f(n)\ \leq d\ \times\ g(n)\ \ \forall\ n\ \geq\ m$$

No gráfico abaixo podemos observar essa propriedade para uma função com ordem Ω(n), onde a partir do valor 400 de entrada 'n', a f(n) se mantém entre as funções c * g(n) e d * g(n).



Qual é a notação O, Ω e Θ para todos os exercícios feitos nesta Unidade 1b?

1) O(1), Ω(1), Θ(1)
2) O(1), Ω(1), Θ(1)
3) O(1), Ω(1), Θ(1)
4) O(1), Ω(1), Θ(1)
5) O(n), Ω(n), Θ(n)
6) O(1), Ω(1), Θ(1)
7) O(1), Ω(1), Θ(1)
8) O(1), Ω(1), Θ(1)
9) O(1), Ω(1), Θ(1)
10) O(1), Ω(1), Θ(1)
11) O(1), Ω(1), Θ(1)
12) O(1), Ω(1), Θ(1)
13) O(n^2), Ω(n^2), Θ(n^2)
14) O(1), Ω(1), Θ(1)
15) O(n^2), Ω(n^2), Θ(n^2)
16) O(n^2), Ω(n^2), Θ(n^2)
17) O(lg(n)), Ω(lg(n)), Θ(lg(n))
18) O(lg(n)), Ω(lg(n)), Θ(lg(n))
19) O(n^2), Ω(n^2), Θ(n^2)
20) O(lg(n)), Ω(lg(n)), Θ(lg(n))
21) O(lg(n)), Ω(lg(n)), Θ(lg(n))
22) O(lg(n)), Ω(lg(n)), Θ(lg(n))
23) O(lg(n)), Ω(lg(n)), Θ(lg(n))
24.a) O(n^2), Ω(n^2), Θ(n^2)
24.b) O(n^3), Ω(n^3), Θ(n^3)
24.c) O(lg(n)), Ω(lg(n)), Θ(lg(n))
24.d) O(n^3), Ω(n^3), Θ(n^3)
24.e) O(n^4), Ω(n^4), Θ(n^4)
24.f) O(lg(n)), Ω(lg(n)), Θ(lg(n))