|  |  |
| --- | --- |
| **BIG O Notation**  Juan Mauricio García Meza | uNIDAD 1  Tarea 3: Big O Notation  Resumen De Vídeos  16/03/19 |

The Big O Notation describes the performance, complexity and efficiency of an algorithm. This notation describes the worst case-scenario, and tells how the algorithm will behave time in function of the data, that is been executing in the memory or disk.

Using Big O Asymptotic Analysis helps computer science to know which algorithm will provide the best efficiency.

The language and metric based on expressions:

* O(1)
* O(log N)
* O(N log N)
* O(N)
* O(N^2)
* O(2^N)

Description of Each Expression

O(1): this expression indicates a constant time, which means the algorithm will have the same performance no matter the input is on work.

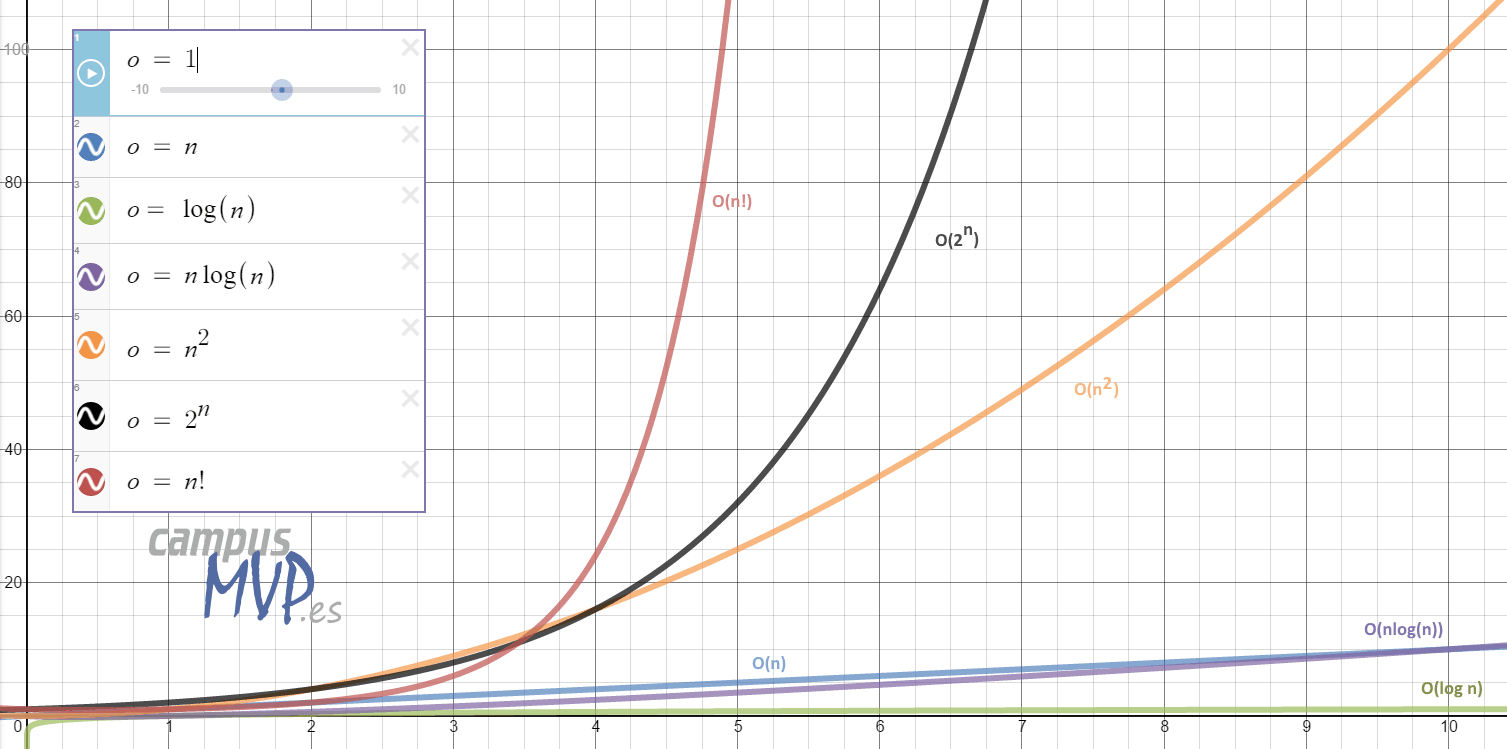
O(N): This expression describes linear growing, and the complexity is based on the proportions of the input.

O(N^2): This expression indicates the complexity based in the square of the input.

O(log N): this expression can be described as an algorithm that cut into pieces the problem, to find the answer; an example can be Binary Search.

O(N log N) Similar function to log N, but this type of expression cut into a lot pieces the problem for each one. For example Quicksort Search.

O(2^N): is based in recursive functions, is well known that in normal conditions this kind of expression is poorly efficient.



Graphic demonstration of each expression of the notation. Dt/Dx

Conclusion

In conclusion, the Big O Notation can help developers to know when to use an specific algorithm; to solve a problem and creating a quick solution to every computer difficulty, knowing the worst scenario, and providing the best path to take.