La notación Big O

La notación Big O es una manera de describir la rapidez o complejidad de un algoritmo dado. Si tu proyecto actual requiere un algoritmo predefinido, es importante entender qué tan rápido o lento es comparado con otras opciones.

¿Qué es la notación Big O y cómo funciona?

En palabras simples, la notación Big O te dice el número de operaciones que hará un algoritmo. Toma su nombre de la "O grande" en frente del número estimado de operaciones.

Lo que la notación Big O no te dice es la rapidez del algoritmo en segundos. Hay demasiados factores que influyen en el tiempo que tarde en ejecutarse un algoritmo. En su lugar, usarás la notación Big O para comparar diferentes algoritmos por el número de operaciones que hacen.

Big O establece un tiempo de ejecución en el peor de los casos

Imagina que eres un maestro y tienes una estudiante de nombre Jane. Quieres encontrar sus registros, así que usas un algoritmo de búsqueda simple para recorrer la bases de datos de tu distrito escolar.

Sabes que a la búsqueda simple le toma O(n) veces ejecutarse. Esto significa que, en el peor de los casos, tendrás que buscar en cada uno de los registros (representados por n) para encontrar el de Jane.

Pero cuando ejecutas la búsqueda simple, encuentras que los registros de Jane son la primera entrada en la base de datos. No tienes que mirar cada entrada – la encontraste en tu primer intento.

¿Este algoritmo tardó O(n) tiempo? ¿O tardó O(1) porque encontraste los registros de Jane en el primer intento?

En este caso, O(1) es el mejor de los casos: tuviste suerte de que los registros de Jane estuvieran al principio. Pero la notación Big O se enfoca en el peor de los casos, el cual es O(n) para la búsqueda simple. Es una garantía de que la búsqueda simple nunca será más lenta que el tiempo O(n).

Los tiempos de ejecución de los algoritmos crecen a ritmos diferentes

Asume que toma 1 milisegundo revisar cada elemento de la base de datos del distrito escolar.

Con la búsqueda simple, si tienes que revisar 10 entradas, le tomará 10 ms en ejecutarse. Pero con el algoritmo de búsqueda binaria, solo tienes que revisar 3 elementos, lo que toma 3 ms en ejecutarse.

En la mayoría de los casos, la lista o base de datos en la que necesitas buscar tendrá cientos o miles de elementos.

Si hay mil millones de elementos, usar la búsqueda simple tardará hasta 1 mil millones de ms, u 11 días. En cambio, la búsqueda binaria tardará solamente 32 ms en el peor de los casos:

Número de elementos	Búsqueda simple	Búsqueda binaria
Tiempo de ejecución en notación Big O	O(n)	O(log n)
10	10 ms	3 ms
100	100 ms	7 ms
10.000	10 seg	14 ms
1000.000.000	11 días	32 ms

Claramente, los tiempos de ejecución de la búsqueda simple y de la búsqueda binaria no crecen ni de cerca al mismo ritmo. Mientras más aumenta la lista de entradas, a la búsqueda binaria solo le toma un poco más de tiempo en ejecutarse. El tiempo de ejecución de la búsqueda simple crece exponencialmente mientras la lista de entradas aumente.

Por esto es muy importante saber cómo el tiempo de ejecución se incrementa en relación con el tamaño de una lista. Y aquí es exactamente donde la notación Big O es muy útil.

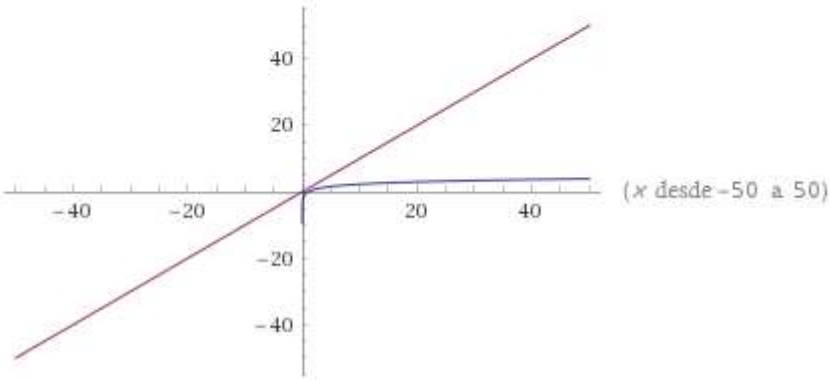
La notación Big O muestra el número de operaciones

Como se mencionó antes, la notación Big O no muestra el *tiempo* que tardará en ejecutarse un algoritmo. En su lugar, muestra el número de operaciones que procesará. Te dice qué tan rápido crece un algoritmo y te permite compararlo con otros.

Interpretación de los datos de entrada:

puntos a trazar	$\log(x)$	$x = -50$ a 50
	x	$y = -50$ a 50

Gráfico:



He aquí algunos algoritmos comunes y sus tiempos de ejecución en notación Big O:

NOTACIÓN BIG O	ALGORITMO DE EJEMPLO
O(log n)	Búsqueda binaria
O(n)	Búsqueda simple
O(n * log n)	Ordenación rápida (Quicksort)
O(n2)	Ordenación por selección
O(n!)	Vendedor viajero