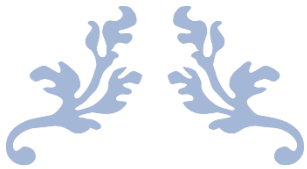


PRACTICA 1: EFICIENCIA

Ejercicio 3: Problemas de precisión

FRANCISCO RUIZ ADÁN

Hardware

CPU: Intel Core i7-7700HQ @ 2.80 GHz 8 cores, Intel HD Graphics 630

RAM: 8GB

Sistema operativo: Ubuntu 18.04 LTS 64-bit

Compilador: g++ versión 7.4.0

Opciones de compilación: -o

El algoritmo es semejante al de la búsqueda binaria con la única diferencia de que este último solo funciona en vectores **ordenados**.

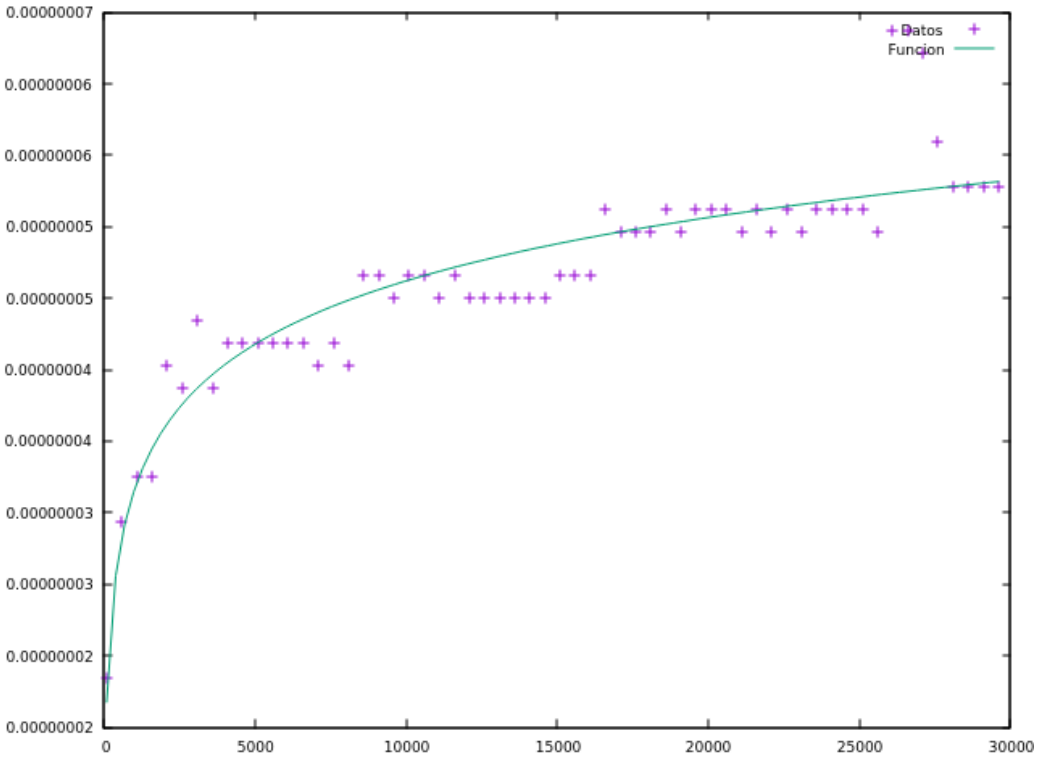
El algoritmo comienza por comparar el elemento del medio del arreglo con el valor buscado. Si el valor buscado es igual al elemento del medio, su posición en el arreglo es retornada. Si el valor buscado es menor o mayor que el elemento del medio, la búsqueda continua en la primera o segunda mitad, respectivamente, dejando la otra mitad fuera de consideración.

Para el cálculo de la eficiencia teórica de la búsqueda binaria, nos basamos en el hecho de que se divide sucesivamente en dos un vector de tamaño n. En el caso peor, el proceso continuaría hasta que no se puedan hacer más divisiones del vector. Por definición, el número máximo de veces que se puede dividir por la mitad un vector de tamaño n es log2(n) veces.

Como el resto de operaciones son elementales (O(1)), concluimos que la eficiencia del algoritmo es logarítmica, es decir, O(log(n)).

Al visualizar la eficiencia empírica, obtenemos una gráfica horizontal. Esto se debe a que el algoritmo se ejecuta tan rápido que es imposible apreciar cambios en el tiempo de ejecución. Para solucionar este problema podemos optar por ejecutar el programa más de una vez para un mismo tamaño y luego dividir el tiempo total de ejecucion por el número de ejecuciones.

Tras realizar la prueba con 10 millones hemos obtenido los siguientes resultados:



Ajustando los puntos a la función $f(x)= 6.40331 \cdot 10^{-9} \log(x) - 7.77186 \cdot 10^{-9}$ se puede observar que la eficiencia empírica describe una gráfica logarítmica.