ESTRUCTURA DE DATOS

Práctica 1. Eficiencia de algoritmos

Doble Grado de Informática y Matemáticas

Víctor Castro Serrano
Maximino Suárez van Gelderen

28 de septiembre de 2017

Condiciones de ejecución.

Dado que en los siguientes ejercicios hablaremos de la eficiencia de distintos programas, conviene detallar las condiciones en las que se han llevado a cabo las pruebas.

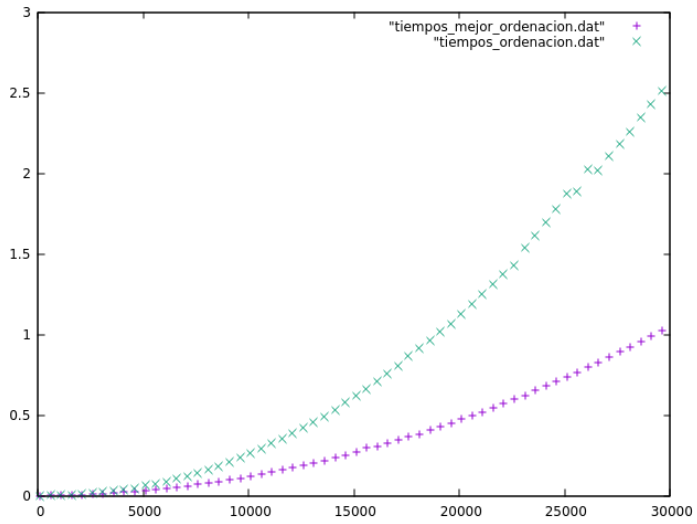
Hardware: Asus GL552VW, Intel Core i5-6300HQ CPU @ 2.30GHz 4 cores, Intel HD Graphics 530 (Skylake GT2), 12GB RAM.
Sistema Operativo: Ubuntu 16.04.3 LTS 64-bit.
Compilador: g++ -o
Opciones de compilación: -Wall -g

Ejercicio 4.

a) Para este apartado, construimos un vector ordenado para representar el caso mejor. Usaremos, por ejemplo, un bucle for:

```
// Generación del vector ordenado
for (int i=0; i<tam; i++)
  v[i] = i;
```

La gráfica que representa la eficiencia empírica es la siguiente:

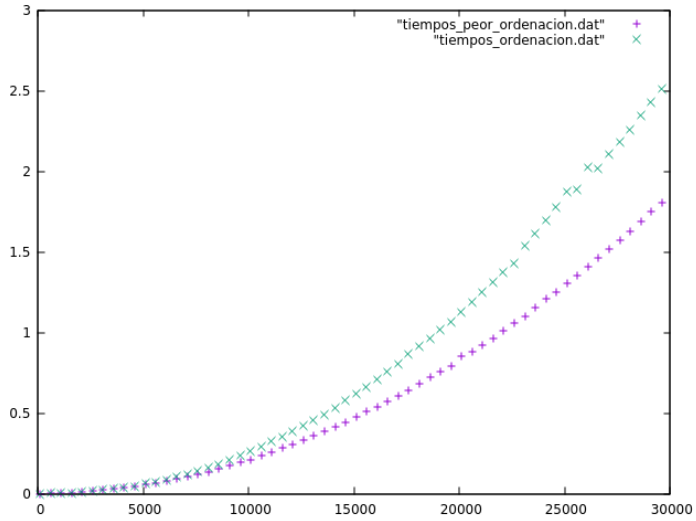


Como se puede observar, el tiempo de ejecución es mucho menor para el mejor caso que para el vector con los elementos generados aleatoriamente.

b) Para este apartado, construimos un vector ordenado en orden inverso, para representar el caso peor.

```
// Generar vector ordenado inversamente
for (int i=0; i<tam; i++)
  v[i] = tam - i ;
```

La gráfica que representa la eficiencia empírica es la siguiente:



Curiosamente nuestro 'peor caso' es mejor que el aleatorio, y esto probablemente se deba a que la opción de rellenar el vector de numeros aleatorios sea más pesada que rellenar el vector mediante nuestro for utilizado.