

Práctica 2: Divide y vencerás

Francisco Carrillo Pérez, Borja Cañavate Bordons,
Miguel Porcel Jiménez, Jose Manuel Rejón Santiago, Jose Arcos Aneas

5 de abril de 2016

Índice

1. Introducción	3
2. Fuerza bruta	3
3. Algoritmo DyV	4
4. Comparación	5
5. Porcentaje de error y constantes ocultas	6

Índice de figuras

2.1.	4
3.1. Figura perteneciente a la eficiencia híbrida obtenida	5
4.1. Figura perteneciente a la comparación entre ambas eficiencias	5

Índice de tablas

1. Introducción

El objetivo de ésta práctica es analizar implementar y comparar una solución divide y vencerás. Para ello, hemos recogido los diferentes tiempos de los diferentes algoritmos que hemos diseñado y los hemos comparado. En nuestro caso concreto, hemos utilizado la biblioteca de C++ más moderna y precisa destinado a obtener tiempos de reloj: la biblioteca **chrono**

La máquina que hemos utilizado tiene las siguientes características:

- Procesador: Intel Core i5-3337U (2.7GHz x 2)
- Memoria RAM: 4GB
- Disco Duro: 500GB 5400 rpm
- SO: Manjaro Linux 15.2 Capella 64 bits

2. Fuerza bruta

En esta sección hemos diseñado un algoritmo de fuerza bruta cuyo funcionamiento es el siguiente:

Require: Vectores ordenados, numero de estos y tamaño

V1 = V[0]

Vfinal = V[1]

Vfinal = OrdenarVectores(V1,Vfinal)

for i=2 hasta i=k-1 **do**

 V1=V[i];

 Vfinal= OrdenarVectores(V1,Vfinal);

 i++;

end for

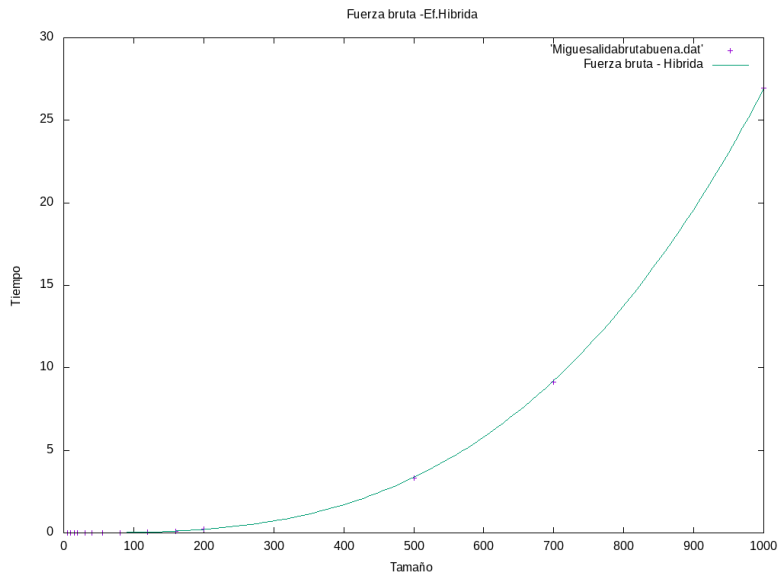


Figura 2.1:

3. Algoritmo DyV

La estructura y el funcionamiento de nuestro código es el siguiente:

Recursivo(matriz)

Require: Matriz de vectores

if Si el número de vectores menor o igual a 1 **then**

return La matriz con una fila

else Si el número de vectores es mayor que 1

 middle=n° filas/2

 Up = matriz[:middle][num_colum]

 Down = matriz[middle:][num_colum]

 Up = Recursivo(Up)

 Down = Recursivo(Down)

 Result = Merge(Up, Down)

return Result

end if

La eficiencia obtenida ha resultado ser de **n2** ,mejorando en un orden al de fuerza bruta.

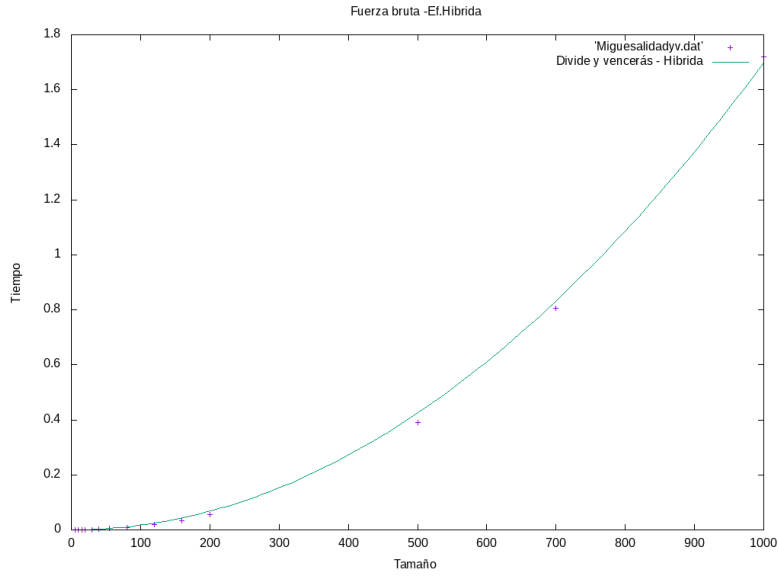


Figura 3.1: Figura perteneciente a la eficiencia híbrida obtenida

4. Comparación

En esta última sección, hemos comparado los dos algoritmos que hemos desarrollado, veremos si efectivamente o no nuestra implementación utilizando un enfoque divide y vencerás obtenemos mejoras con respecto a uno de fuerza bruta.

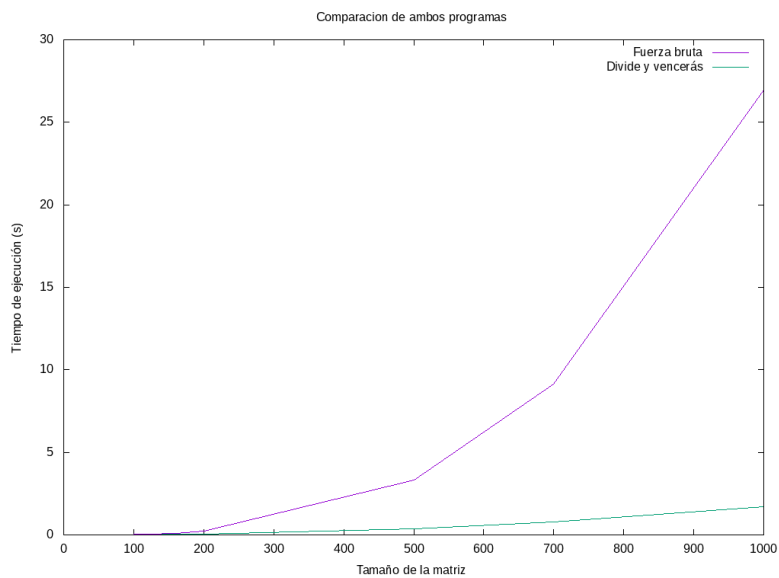


Figura 4.1: Figura perteneciente a la comparación entre ambas eficiencias

5. Porcentaje de error y constantes ocultas

Por último, mostramos el porcentaje de error así como las constantes ocultas.

Algoritmo	Constante Oculta	Error
DyV	$a_0 = 1.69804e-06$	+/- 1.226e-08 (0.7218 %)
Fuerza bruta	$a_0 = 2.69052e-08$	+/- 2.352e-11 (0.0874 %)