PRÁCTICA 2:

ALGORITMOS DIVIDE Y VENCERÁS

Algorítmica
2016-2017

Componentes del Grupo:
Daniel Bolaños Martínez
José María Borrás Serrano
Santiago De Diego De Diego
Fernando De la Hoz Moreno

Índice:

Introducción:

Hemos diseñado un algoritmo basado en “divide y vencerás” el cual tiene como objetivo encontrar el valor máximo de una serie unimodal. El orden de eficiencia de este algoritmo es O(log(n)) y lo hemos comparado con el algoritmo trivial para este problema que es de orden O(n).

Para la comparación hemos obtenido unas tablas en las que se muestran el tiempo de ejecución según distintos número de elementos en los vectores, hemos representado los datos en una gráfica y hemos ajustado estos datos a la función obtenida por la eficiencia teórica por el ajuste de mínimos cuadrados.

Ejercicio 1:
Algoritmo Divide y Vencerás para la resolución del problema:

Función Algoritmo unimodal Divide y Vencerás:

```
int unimodal(vector<int> v)
{
        bool fin=false;
        int maximo=v.size()-1;
        int indice=maximo/2;
        int minimo;

        while(!fin)
        {
        if(v.at(indice-1)<v.at(indice))
           if(v.at(indice+1)<v.at(indice))
                           fin=true;
                   else
                   {
                                   minimo=indice;
                                   indice=indice+((maximo-indice)/2);
                           }
                else
                {
                           maximo=indice;
                           indice=minimo+((indice-minimo)/2);
                }
        }
        return indice;
}
```

El algoritmo consiste en tomar el elemento que se encuentra en mitad del vector y comprobar si es un máximo viendo si es mayor que el elemento de la izquierda y menor que el de la derecha. Si es así se ha terminado el algoritmo pues ya hemos encontrado el máximo. Si no es así vemos si el elemento esta en la zona creciente o decreciente del vector. En el caso de que este en la zona creciente el máximo se situara en la mitad de la

derecha del vector y si se encuentra en la decreciente en la mitad izquierda. En este punto se vuelve a aplicar el algoritmo sobre la mitad del vector donde se encuentre el máximo y se repite el proceso hasta que se encuentre el máximo.

Como en cada iteración lo que se hace es dividir el vector por la mitad y buscar el máximo en una mitad el número máximo de iteraciones hasta encontrar el máximo es de log(n) siendo n el tamaño del vector. Como todo las comprobaciones realizadas en cada iteración son O(1) el algoritmo es O(log(n)).

Función Algoritmo unimodal Secuencial:

```
int unimodal_secuencial(vector<int> v)
{
        bool fin=false;
        int indice=1;

        while(!fin)
        {
        if(v.at(indice+1)<v.at(indice))
                fin=true;
        else
                indice++;
        }

        return indice;
}
```

En este caso lo único que se hace es recorrer el vector hasta ver que empieza a ser decreciente. En el peor caso puede empezar a ser decreciente en el penúltimo elemento por lo que habría que recorrer todo el vector, de manera que la eficiencia de este algoritmo es O(n).

Ejercicio 2:

Datos Algoritmos Secuencial y Logarítmico:

Tamaño Vectores	Tiempo DyV	Tiempo Secuencial
1000000	0.00057548	0.00108062
2000000	0.00130396	0.00227099
3000000	0.00200936	0.00329093

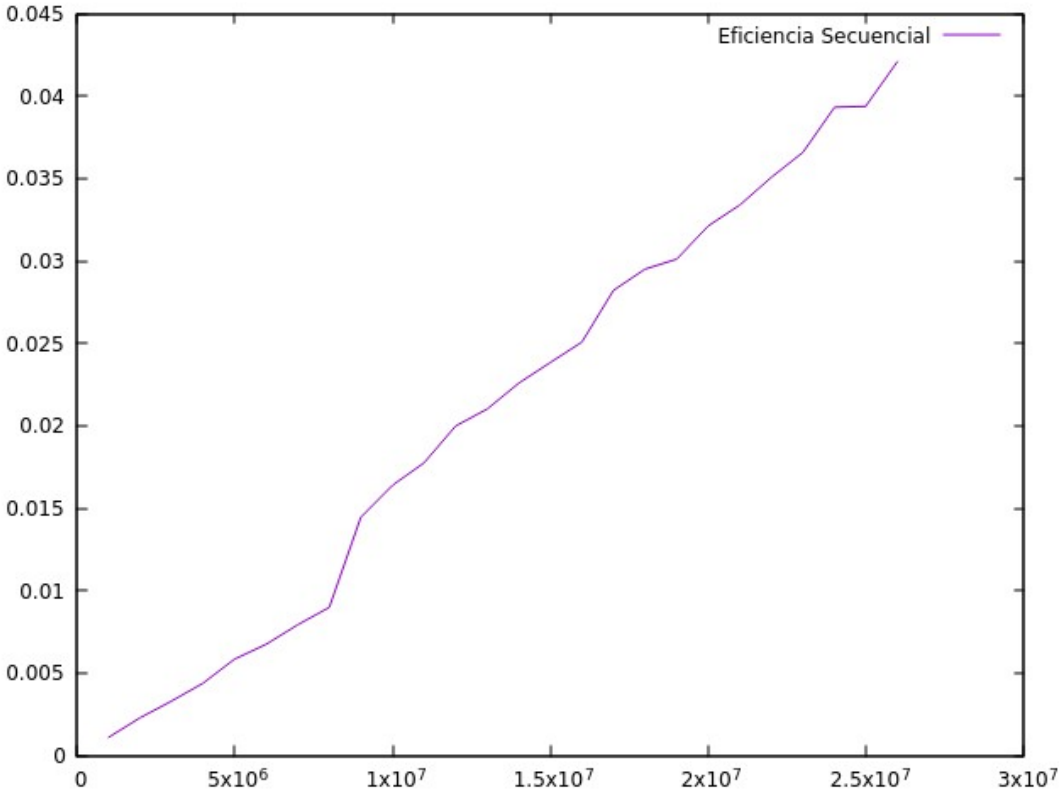
4000000	0.00265309	0.00437678
5000000	0.00345555	0.00583811
6000000	0.00409567	0.00673738
7000000	0.00464896	0.00792554
8000000	0.0052876	0.00897848
9000000	0.0103381	0.0144504
10000000	0.0116623	0.016394
11000000	0.0128614	0.0177504
12000000	0.0147915	0.0199869
13000000	0.014961	0.0210103
14000000	0.0162433	0.0225911
15000000	0.0173869	0.0238454
16000000	0.0184078	0.0250778
17000000	0.0200822	0.0282202
18000000	0.0216763	0.0295068
19000000	0.021972	0.0301068
20000000	0.0233563	0.0321234
21000000	0.0248113	0.0333933
22000000	0.0255466	0.0350716
23000000	0.0264495	0.0365949
24000000	0.0284914	0.039327
25000000	0.0284393	0.0393759
26000000	0.0297109	0.0420735

Como podemos observar los tiempos de divides y vencerás son mejores que los del secuencial.

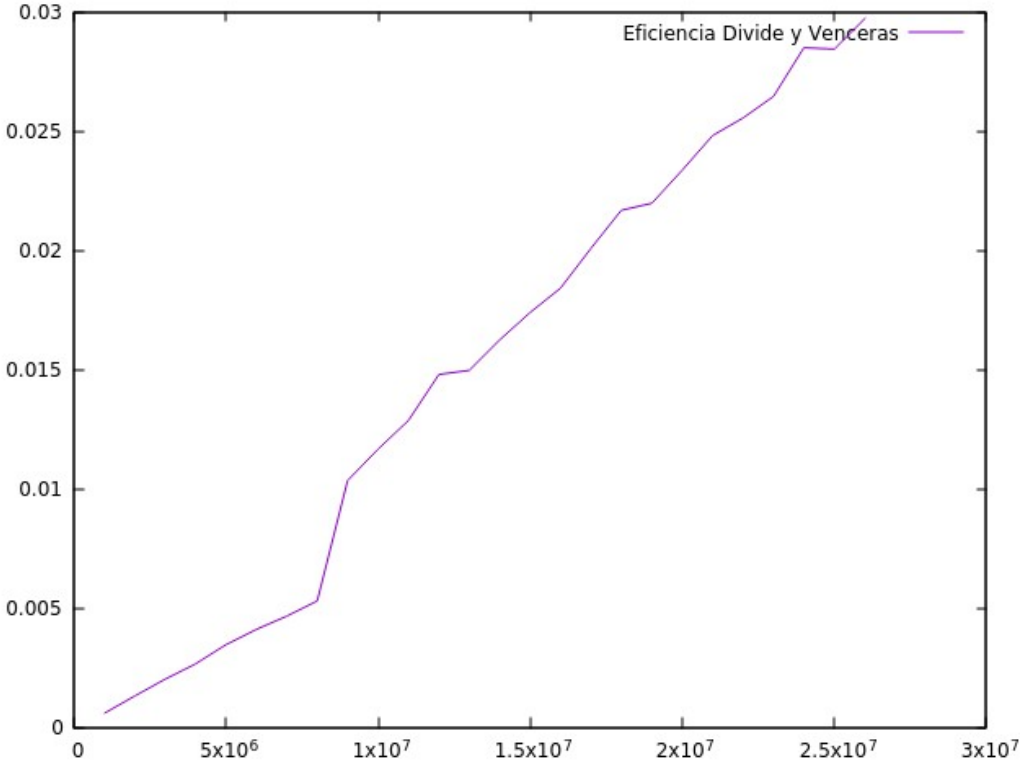
A continuación podemos observar las gráficas que nos muestran los tiempos de ejecución en función del numero de elementos del vector.

Ejercicio 3:

Eficiencia empírica (Gráficos):



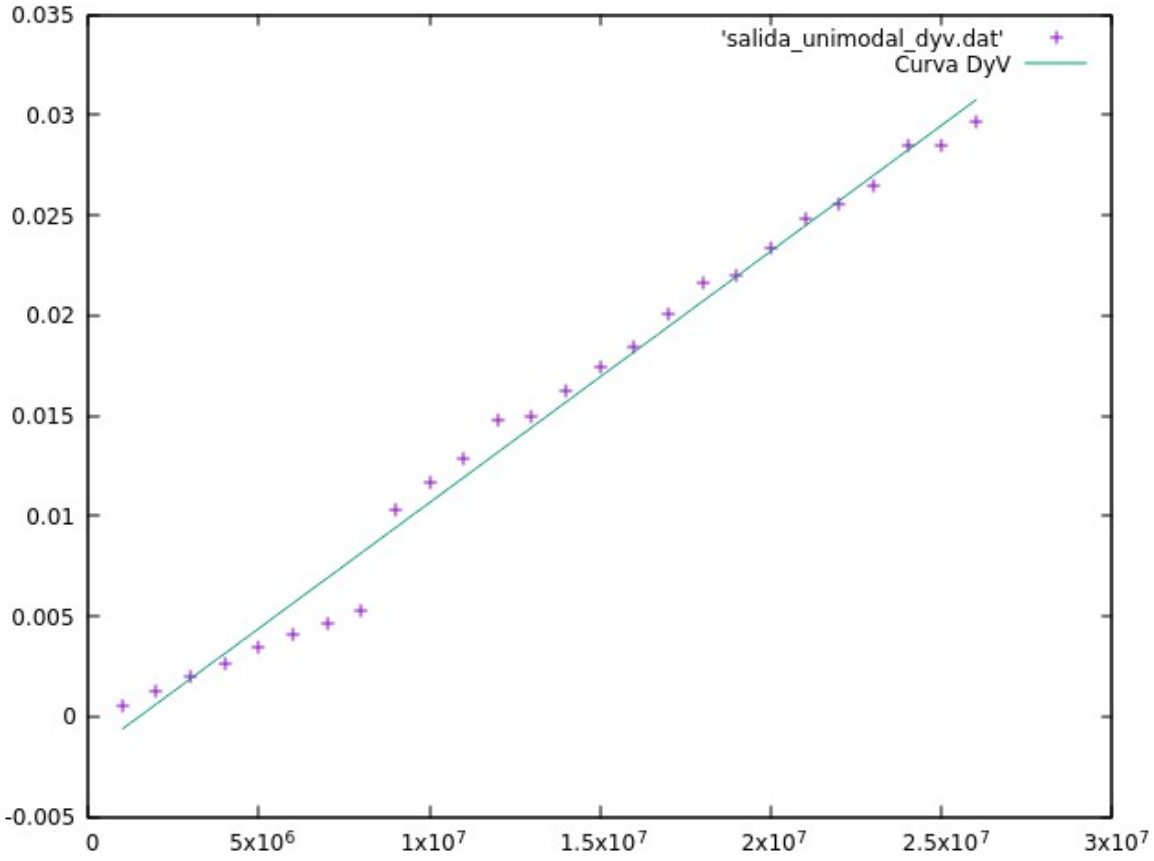
Representación de los datos obtenidos por el algoritmo secuencial para el problema planteado.



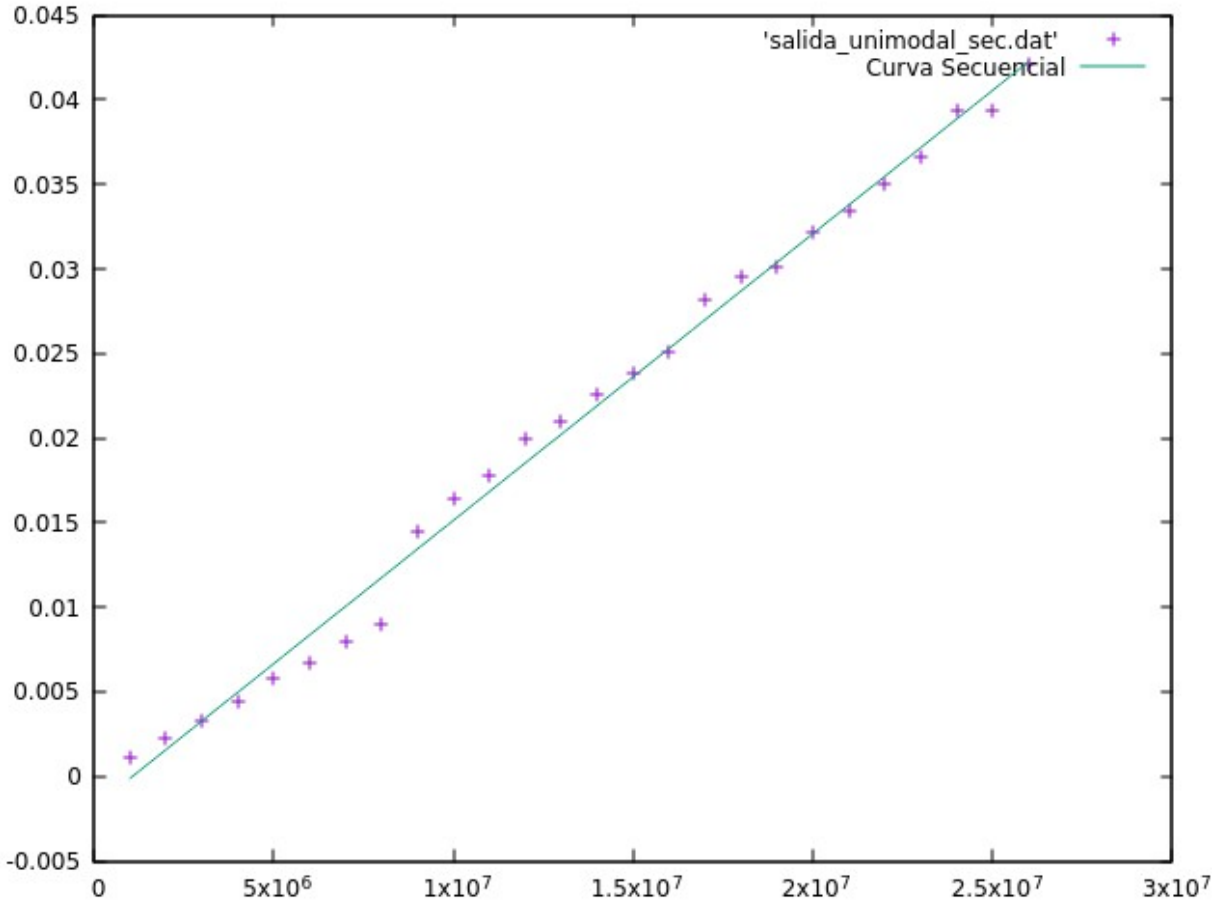
Representación de los datos obtenidos por el algoritmo divide y vencerás para el problema planteado.

Ajustes híbridos:

A continuación, los ajustes de los datos a las expresiones obtenidas de la eficiencia teórica.



f(x)=a0*log(x)+a1*x+a2
a0=1.69539e-09
a1=1.25523e-09
a2=-0.00189877



f(x)=a0*x+a1
a0=1.69539e-09
a1=-0.00183396