Algorítmica 2019/20

Ejercicio 3.4:

Eliminar elementos repetidos divide y vencerás

Índice:

1-.Eliminar elementos repetidos con fuerza bruta

1.1-.Análisis teórico.
1.2-.Análisis empírico.
1.3-.Análisis híbrido.

2-.Eliminar elementos repetidos con divide y vencerás

2.1-.Algoritmo y caso de ejecución
2.2-.Análisis teórico.
2.3-.Análisis empírico.
2.4-.Análisis híbrido.

3-.Comparación y umbral
4-.Valoración sobre el trabajo de cada miembro

Grupo: ***JAMA***

Alberto Robles Hernández
Ahmed Brek Prieto
Jorge Medina Romero
Mohammed Lahssaini Nouijah

1-.Eliminar elementos repetidos con fuerza bruta

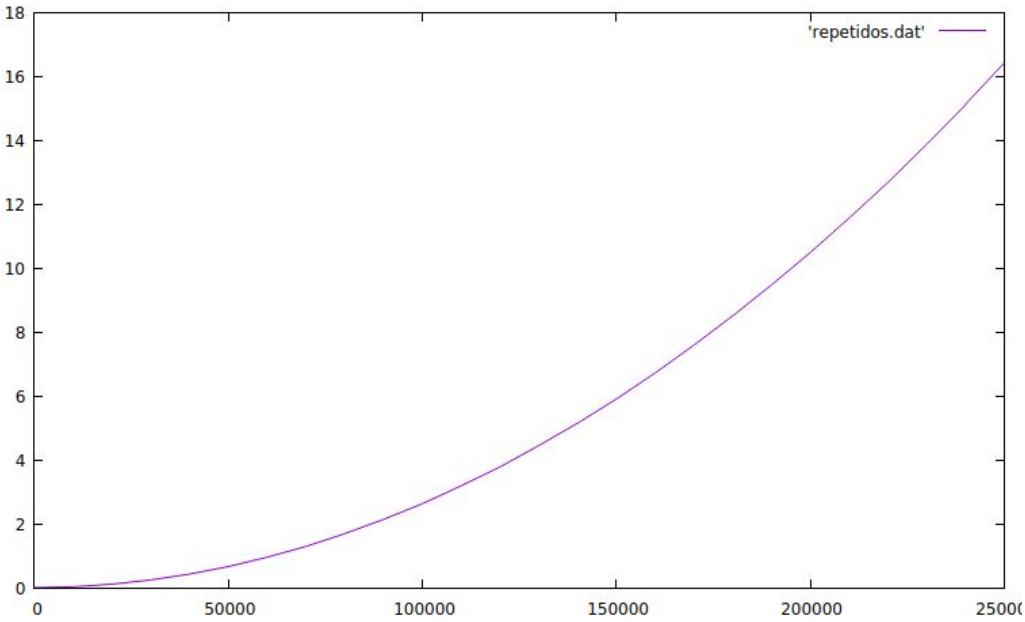
1.1-.Análisis teórico.

A pesar de que si observamos el código vemos tres bucles anidados y puede parecer que la eficiencia es n^3, no es así, ya que los bucles no ejecutan n iteraciones cada uno (sea n el tamaño del problema). Esto se debe a que el tamaño del vector se va reduciendo según se eliminan elementos repetidos. Así, aunque se vean 3 bucles anidados, por cada iteración que se ejecute el más interior (bucle k), se reduce en una unidad el tamaño del vector, por lo que al regresar a bucles exteriores, el número de repeticiones será menor. Es fácil ver cómo esto compensa las iteraciones que se hacen en un bucle con las que no se harán en otro, siendo así la eficiencia de $O(n^2)$.

Veámoslo de otra forma, teniendo en cuenta que el peor de los casos es aquel en que todos los elementos están repetidos. Así, tendremos que recorrer el vector una vez (eficiencia O(n)) y, por cada iteración, comprobar con sus elementos anteriores si está repetido (sea i la posición del elemento, iterará i veces). Por otro lado, el bucle en que desplaza hacia la izquierda el resto de elementos de la derecha a partir de esa posición (para “pisar” el elemento repetido y eliminarlo) hace n - i iteraciones. Se observa cómo las iteraciones que ejecuta un bucle y las que ejecuta otro suman n, por lo que la eficiencia del bucle interior es n. Así, concluimos que la eficiencia del algoritmo es $n * n = n^2$.

1.2-.Análisis empírico.

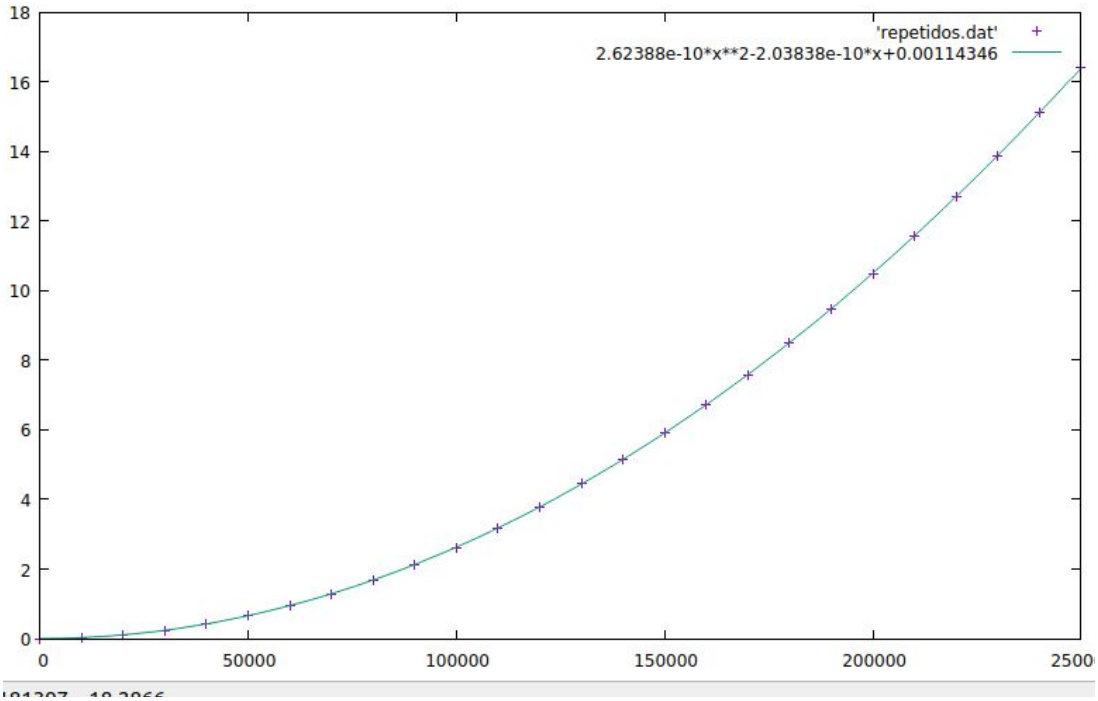
Número de elementos	Tiempo(s)
0	1e-06
10000	0.02653
20000	0.105201
30000	0.236641
40000	0.421097
50000	0.656609
60000	0.947444
70000	1.28446
80000	1.68375
90000	2.13041
100000	2.6219
110000	3.18241
120000	3.77683
130000	4.44275
140000	5.14045
150000	5.89882
160000	6.71615
170000	7.59038
180000	8.50747
190000	9.47345
200000	10.4928
210000	11.5683
220000	12.6861



230000	13.88
240000	15.1148
250000	16.4139

1.3-.Análisis híbrido.

f(x) = 2.62388e-10*x²-2.03838e-10*x+0.00114346
R²=0.99999896

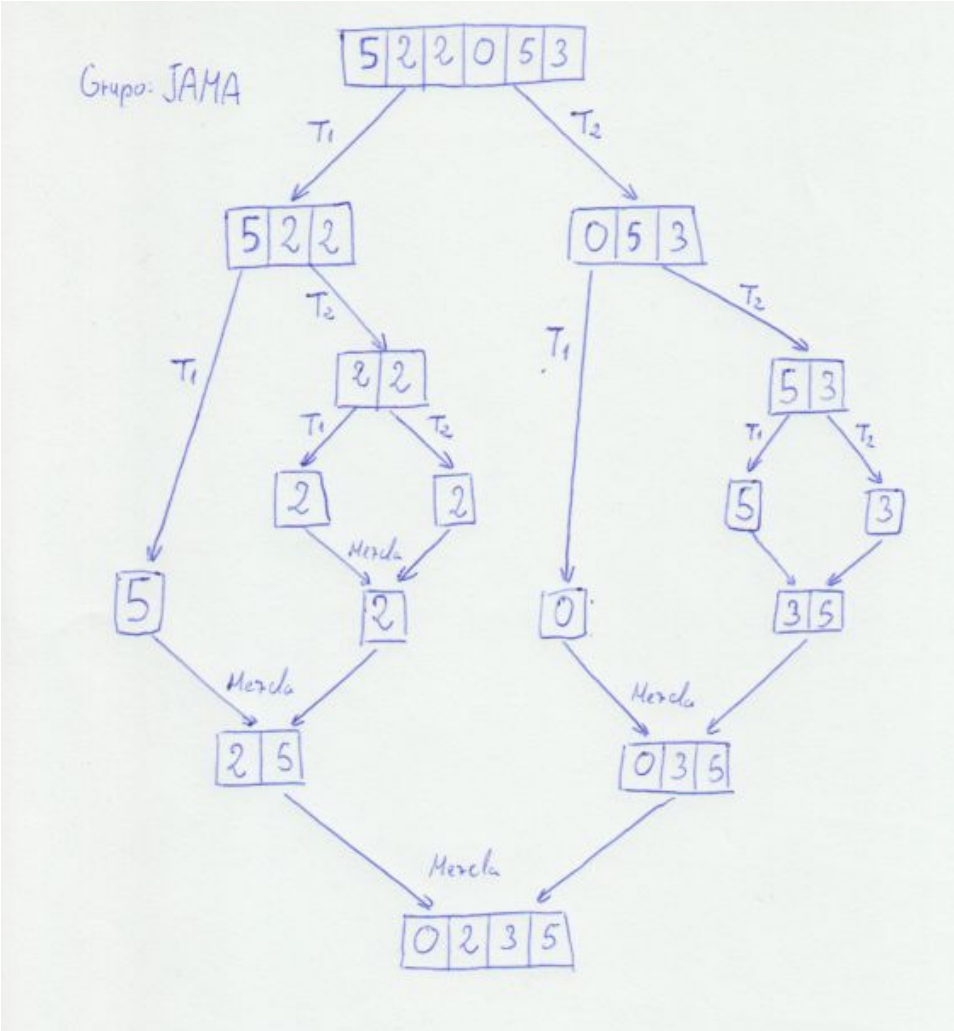


2-.Eliminar elementos repetidos con divide y vencerás

2.1-.Algoritmo y caso de ejecución

Al ejecutar el algoritmo con un vector de tamaño 6, por ejemplo, y una vez relleno el vector quedaría: [5,2,2,0,5,3].
Como podemos observar el algoritmo divide el vector en:

[5,2,2] y [0,5,3], que a la vez subdivide recursivamente hasta que el vector queda subdivido en vectores de tamaño 1.
A continuación, comienza a mezclar los vectores de dos en dos, (T1 y T2) ordenando y eliminando las componentes que estén repetidas en ambos vectores.
A continuación, se muestra una imagen que complementa la explicación del caso de ejecución de forma gráfica.



```
alberto@alberto-System-Product-Name:~/Escritorio
Vector inicial: 5 2 2 0 5 3
T1: 5 2 2  /// T2: 0 5 3
T1: 5  /// T2: 2 2
T1: 2  /// T2: 2
Mezcla: 2
Mezcla: 2 5
T1: 0  /// T2: 5 3
T1: 5  /// T2: 3
Mezcla: 3 5
Mezcla: 0 3 5
Mezcla: 0 2 3 5
Vector final: 0 2 3 5
6  8.1e-05
alberto@alberto-System-Product-Name:~/Escritorio
```

2.2-.Análisis teórico.

Suponiendo que n es potencia de 2

$$T(n) \begin{cases} c_1 & \text{si } n = 1 \\ 2T(n/2) + c_2 n & \text{si } n > 1 \end{cases}$$

Aplicamos expansión:

$T(n) = 2T(n/2) + c_2 n$; $T(n/2) = 2T(n/4) + c_2 n/2$

$T(n) = 4T(n/4) + 2\, c_2 n$; $T(n) = 8T(n/8) + 3\, c_2 n/2$

En general,

$$T(n) = 2^i T(n/2^i) + i c_2 n$$

Tomando $n = 2^k$, la expansión termina cuando llegamos a T(1) en el lado de la derecha, lo que ocurre cuando i=k

$$T(n) = 2^k T(1) + k c_2 n$$

Como $2^k = n$, entonces $k = \log(n)$;

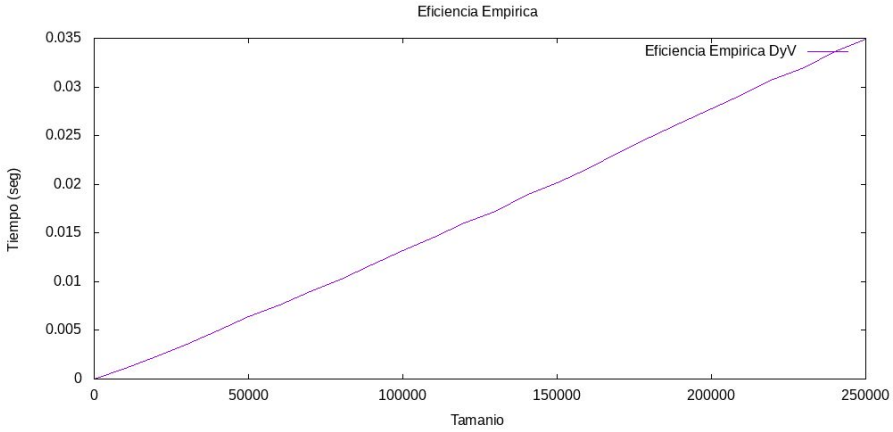
Como además

$$T(1) = c_1 n + c_2 n \log(n)$$

Por tanto el tiempo para el algoritmo de ordenación por mezcla es O(nlog(n)).

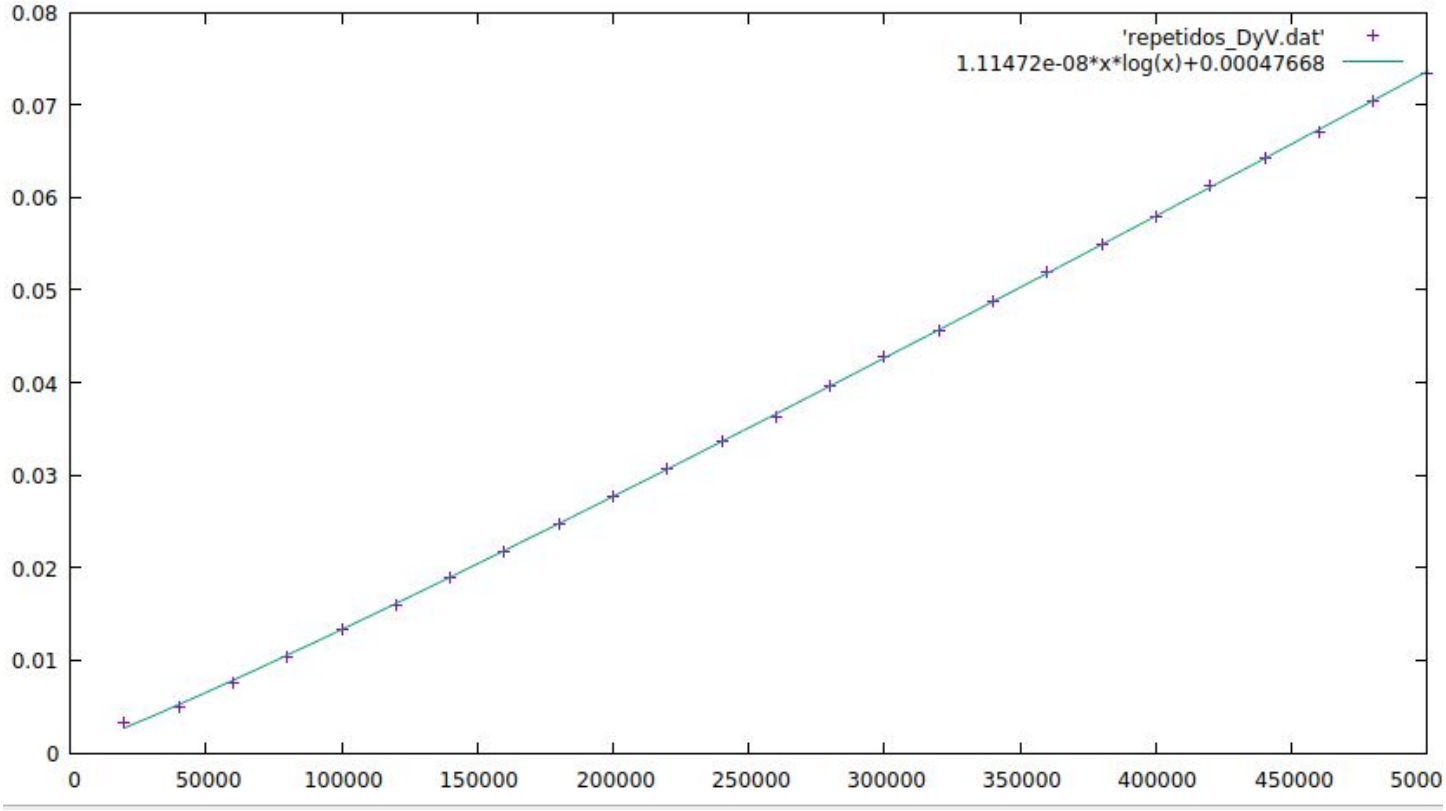
2.3-.Análisis empírico.

Numero de elementos	Tiempo(s)
0	1e-066
10000	0.001108
20000	0.002314
30000	0.00356
40000	0.004913
50000	0.006392
60000	0.007596
70000	0.008976
80000	0.010278
90000	0.011725
100000	0.013204
110000	0.014583
120000	0.015989
130000	0.017244
140000	0.018885
150000	0.020189
160000	0.021636
170000	0.023306
180000	0.024806
190000	0.026303
200000	0.027721
210000	0.029246
220000	0.030744
230000	0.031959
240000	0.033583
250000	0.034903



2.4-.Análisis híbrido.

f(x)=1.11472e-08*x*log(x)+0.00047668
R^2=0.9998249



3-.Comparación y umbral

f(x) = 1.11472e-08*x*log(x)+0.00047668
f(x) = 2.62388e-10*x²-2.03838e-10*x+0.00114346

Tras resolver el sistema de ecuaciones nos sale que si el tamaño del vector es menor de 298,7843 es más eficiente usar la fuerza bruta, pero para tamaños mayores es más eficiente usar el divide y vencerás.