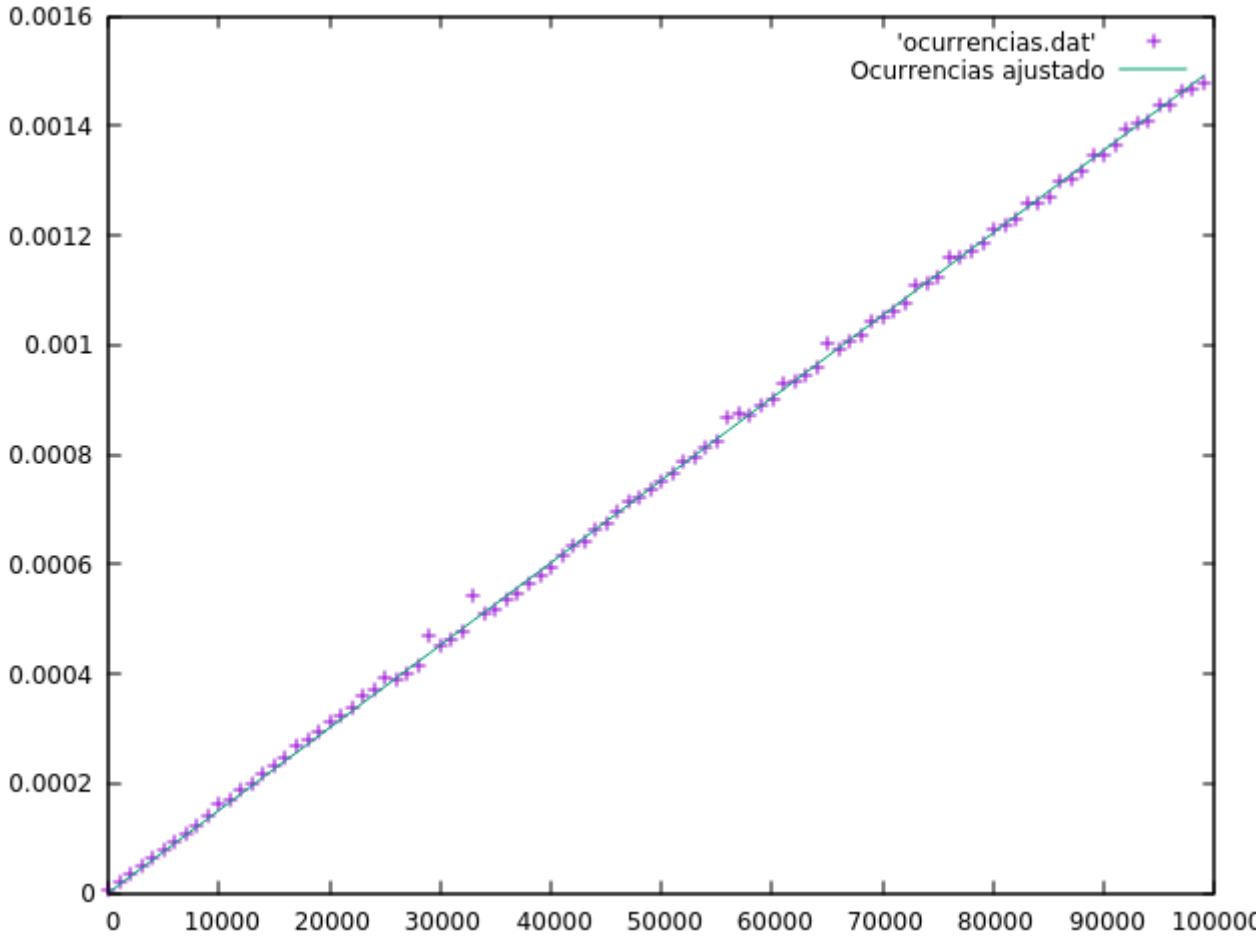
Adrián Jesús Peña Rodríguez 2º B Grupo B

Ocurrencias.dat

Esta es la gráfica de ocurrencias.dat ajustada con la recta de regresión f(x) = a * x, ya que es de orden O(n):

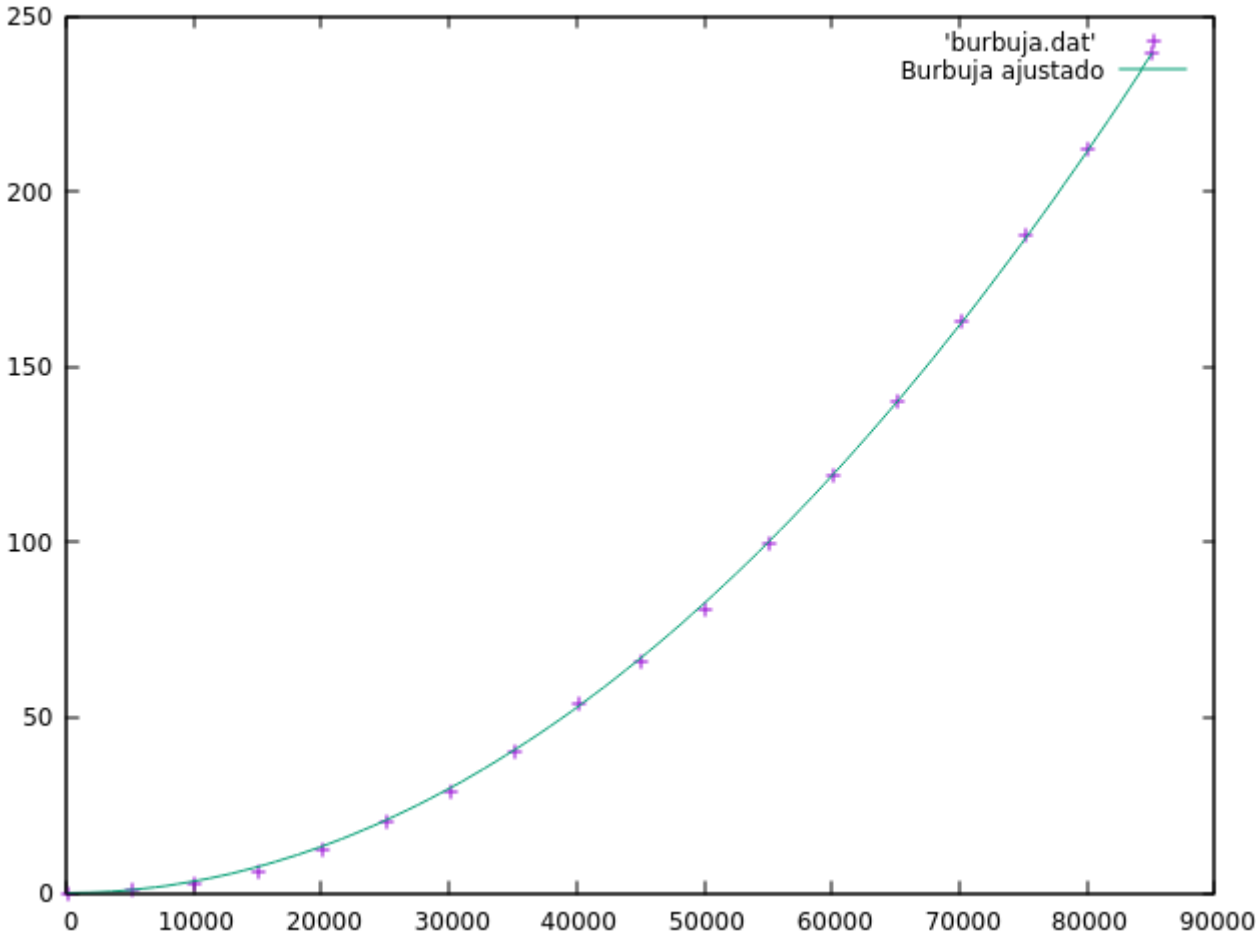


Podemos observar que nuestro calculo teórico es muy acertado y se ajusta muy bien a nuestro modelo. Al ser de orden n vemos como su gráfica es prácticamente una linea recta al igual que la recta con la que la hemos ajustado.

Adrián Jesús Peña Rodríguez 2º B Grupo B

Burbuja.dat:

Esta es la gráfica de burbuja.dat, la cual la hemos ajustado con una recta de regresión f(x) = a * x * x ya que es de orden O(n * n):

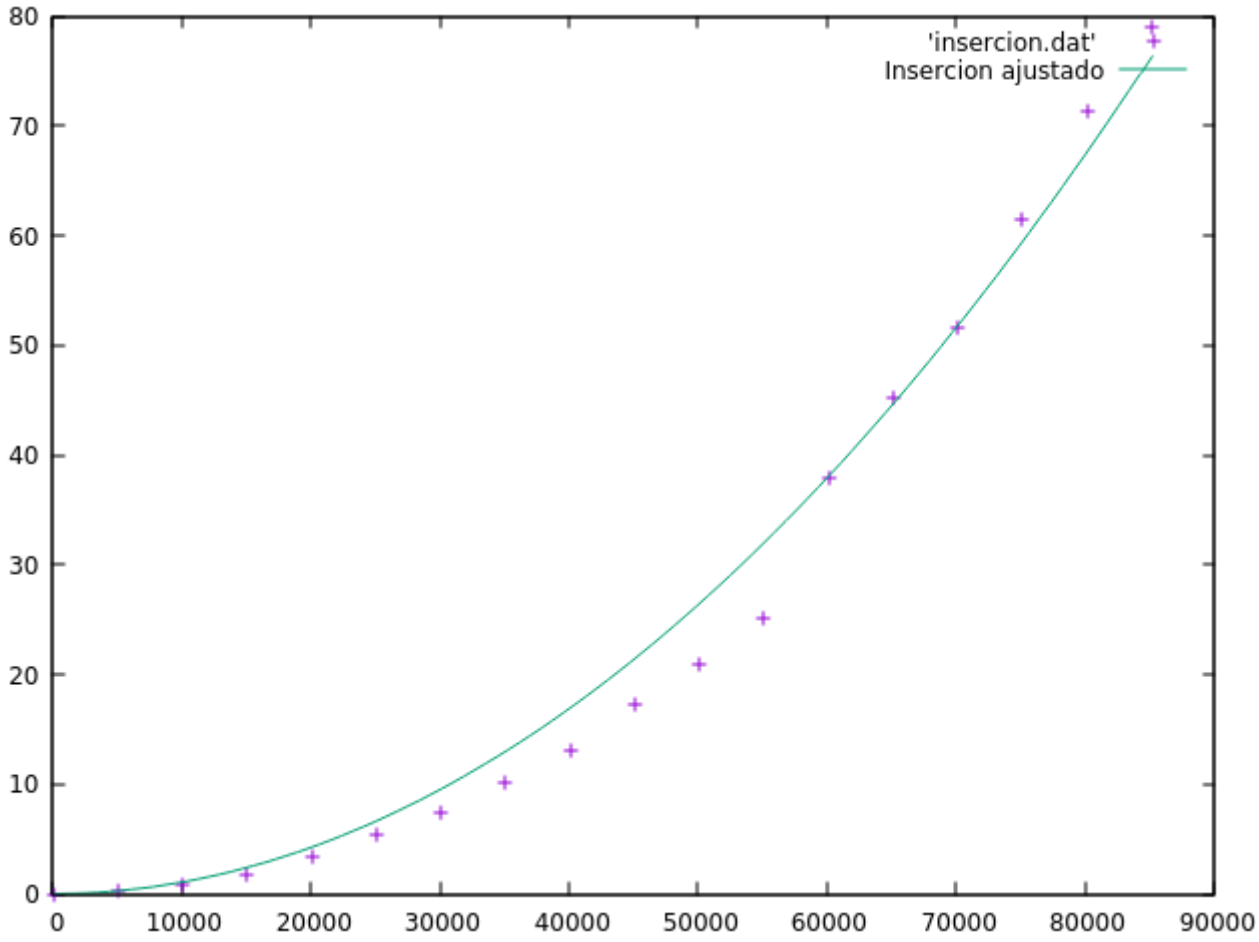


Podemos ver que se ajusta casi perfectamente a nuestro calculo teórico, esto se debe a que tiene orden O(n*n) debido a sus dos bucles internos los cual son de orden n independientemente, al multiplicarlos tenemos que se nos queda en orden n*n.

Adrián Jesús Peña Rodríguez 2º B Grupo B

Insercion.dat:

Esta es la gráfica de insercion.dat, la cual la hemos ajustado con una recta de regresión f(x) = a * x * x ya que es de orden O(n * n):

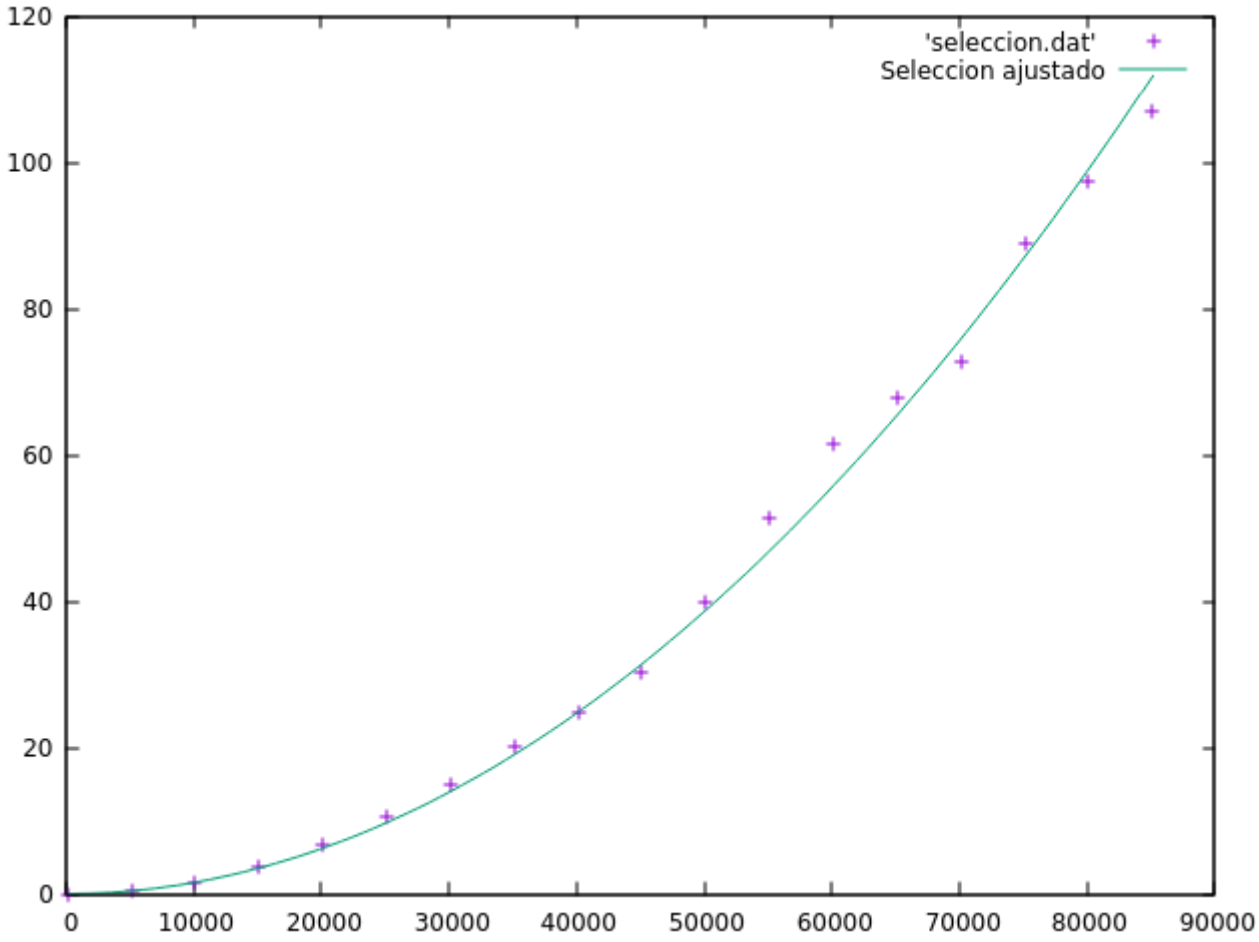


Podemos apreciar que al igual que con el algoritmo de la burbuja, tenemos que se ajusta bien aunque vemos que no es totalmente fiel al calculo teórico. Esto se debe a la inclusión del bucle interno while en lugar del for.

Adrián Jesús Peña Rodríguez 2º B Grupo B

Seleccion.dat:

Esta es la gráfica de seleccion.dat, la cual al igual que las otras la hemos ajustado con una recta de regresión f(x) = a * x * x ya que es de orden O(n * n):



Se ajusta perfectamente a nuestra eficiencia teórica ya que al igual que los otros dos algoritmos de ordenación es de orden n*n.

En conclusión, tenemos que el algoritmo más eficiente de los tres es sin duda alguna el de inserción ya que este a diferencia de los otros no necesita de tantos recorridos.
El algoritmo de la burbuja como bien se aprecia en la gráfica es el peor de los tres, llegando a ser terriblemente lento en textos grandes.

Aunque lo cierto es que en los 5000 primero elementos la diferencia entre estos algoritmos es mínima pero debido a su orden cuando vamos aumentando el tamaño la velocidad varía mucho de unos a otros.

Adrián Jesús Peña Rodríguez 2º B Grupo B

Frecuencias.dat:

En cuanto a este archivo, tenemos que tiene varias versiones de algoritmo:

- El algoritmo versión 1 es de orden n claramente y además podemos garantizar que no proporciona una salida correcta.

- El algoritmo versión 2 de de orden n*n, tenemos un bucle for O(n) y dente de este llamamos al método buscar el cual es O(n) por lo que tenemos O(n*n) en total.

- El algoritmo versión 3 es de orden n*log(n)*n*n por lo que tenemos que es de orden n*n*n.

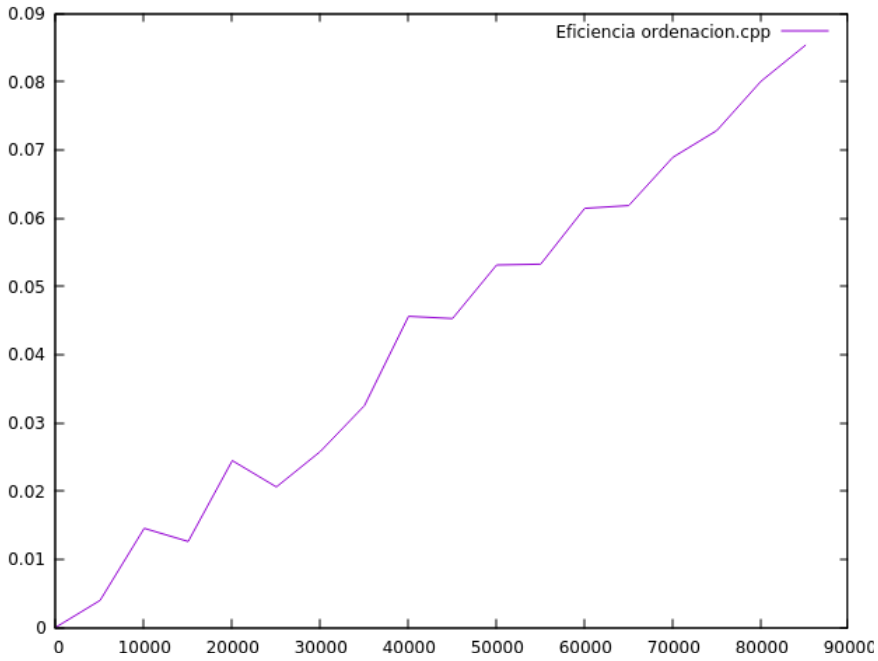
- El algoritmo versión 4 es de orden n*log(n)

Adrián Jesús Peña Rodríguez 2º B Grupo B

ordenacion.cpp

g++ -std=c++11 -o ejecutable fuente.cpp
./ordenacion > ordenacionB.dat
Mi archivo ordenacionB.dat contiene los siguientes datos y su gráfica al lado:

100 4.9546e-05
5100 0.00395796
10100 0.014529
15100 0.0125955
20100 0.0244679
25100 0.0205892
30100 0.0257928
35100 0.0325082
40100 0.045616
45100 0.0452844
50100 0.0531296
55100 0.053253
60100 0.0614405
65100 0.0618526
70100 0.0689207
75100 0.0728411
80100 0.0800552
85100 0.0852936



A continuación muestro la gráfica tiempo teórico vs tiempo empírico, así como la regresión:

*y = -0.097487 - 5.3679e-06 * x +4.2447e-08 * x * x*