Algoritmo Burbuja

Explicación del Algoritmo

Es un algoritmo de ordenación de orden $O(n^2)$.

Este algoritmo utiliza la recursividad para ir dividiendo en dos el vector dado, e ir ordenando cada parte.

No es uno de los algoritmos de ordenación más rápidos, pero tampoco es de los más lentos.

Tiempos

Se midió el tiempo con la librería ctime antes y después de llamar al algoritmo (no fue necesario utilizar un bucle para el algoritmo por números demasiado pequeños).

Se creó un script para llamar al ejecutable en un bucle donde la entrada inicial era 1000 y se van aumentando las entradas de 1000 en 1000 y que redirige la salida del número de entradas más los tiempos para esa entrada a un fichero .dat

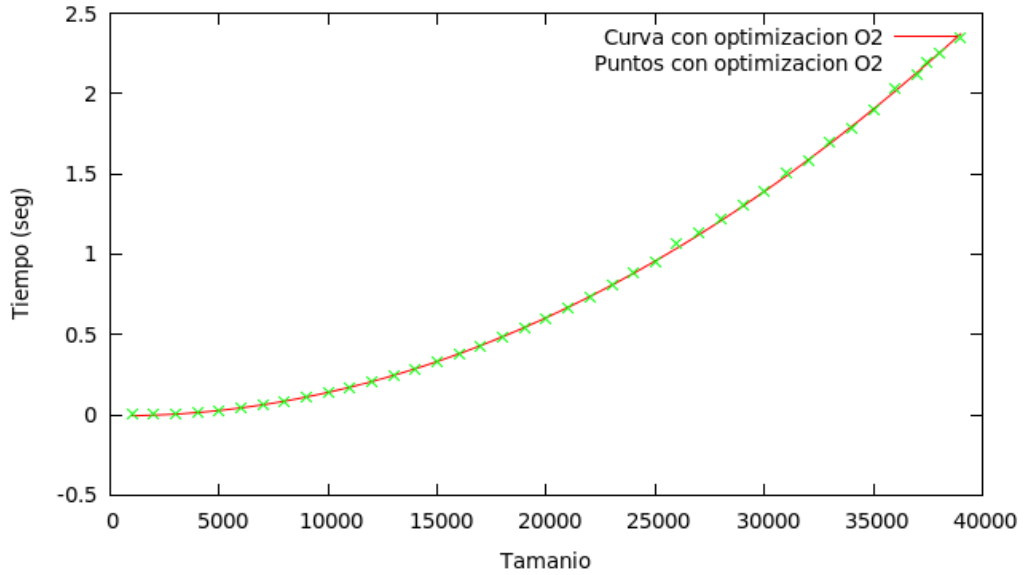
Se utilizó en gnuplot como función para calcular las constantes mediante regresión:

f(x) = a1*x*x + a2*x + a3

El resultado fue el siguiente:

```
a1              = 1.60174e-09      +/- 1.21e-11     (0.7555%)
a2              = -1.51444e-06     +/- 4.991e-07    (32.96%)
a3              = -0.00557868      +/- 0.004329     (77.6%)
```

Representación Gráfica



Algoritmo	Mergesort

Explicación del Algoritmo

Es un algoritmo de ordenación de orden O(n*log(n)).

Es un algoritmo de ordenamiento externo estable que está basado en la técnica “Divide y vencerás”.

La eficiencia de este algoritmo es mucho mayor que la de cualquiera de orden $O(n^2)$

Tiempos

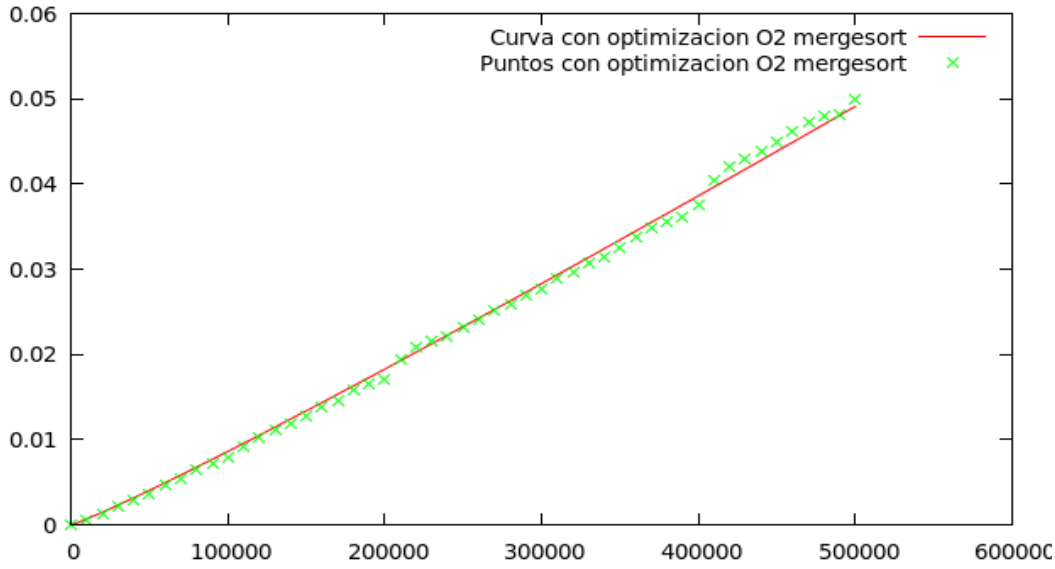
Se midió el tiempo con la librería ctime antes y después de llamar al algoritmo. Se utilizó un bucle para llamar al algoritmo por intentar suavizar los puntos que salían dispares (y después dividir la salida, claro), además de salir números pequeños debido a la gran eficiencia que tiene.

Se creó un script para llamar al ejecutable en un bucle donde la entrada inicial era 100 y se van aumentando las entradas de 10000 en 10000 hasta obtener 100 mediciones (y aún así hace las mediciones bastante rápido) y que redirige la salida del número de entradas más los tiempos para esa entrada a un fichero .dat

Se utilizó en gnuplot como función para calcular las constantes mediante regresión:

f(x) = a1*x*log(x)

Representación Gráfica



Algoritmo — Floyd

Explicación del Algoritmo

Es un algoritmo de análisis sobre grafos de orden $O(n^3)$ para encontrar el camino en grafos dirigidos ponderados. Es un ejemplo de programación dinámica.

Es un algoritmo muy poco eficiente, pero no existen otros algoritmos mejores para el uso que se le da a este algoritmo.

Tiempos

Se midió el tiempo con la librería ctime antes y después de llamar al algoritmo.

Se creó un script para llamar al ejecutable en un bucle donde la entrada inicial era 150 y se van aumentando las entradas de 150 en 150 hasta 3750 (para tener 25 mediciones, era imposible obtener más debido a que el algoritmo es demasiado costoso)

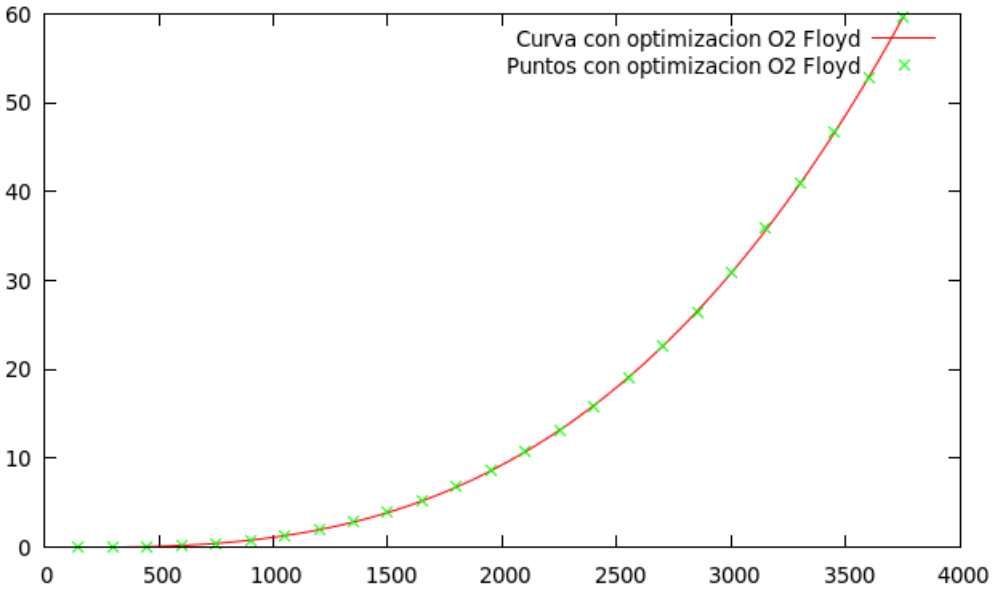
Se utilizó en gnuplot como función para calcular las constantes mediante regresión:

f(x) = a1*x*x*x + a2*x*x + a3*x + a4*x

El resultado es el siguiente:

```
a1              = 1.04639e-09      +/- 1.202e-11    (1.149%)
a2              = 4.36133e-07      +/- 7.123e-08    (16.33%)
a3              = -0.00046293      +/- 0.0001208    (26.09%)
a4              = 0.094715         +/- 0.05548      (58.58%)
```

Representación Gráfica



Comparación Diferentes PCs

Propiedades

-Mismo algoritmo.

-Misma optimización de código (en este caso, hemos utilizado -O2).

-Diferentes PCs:

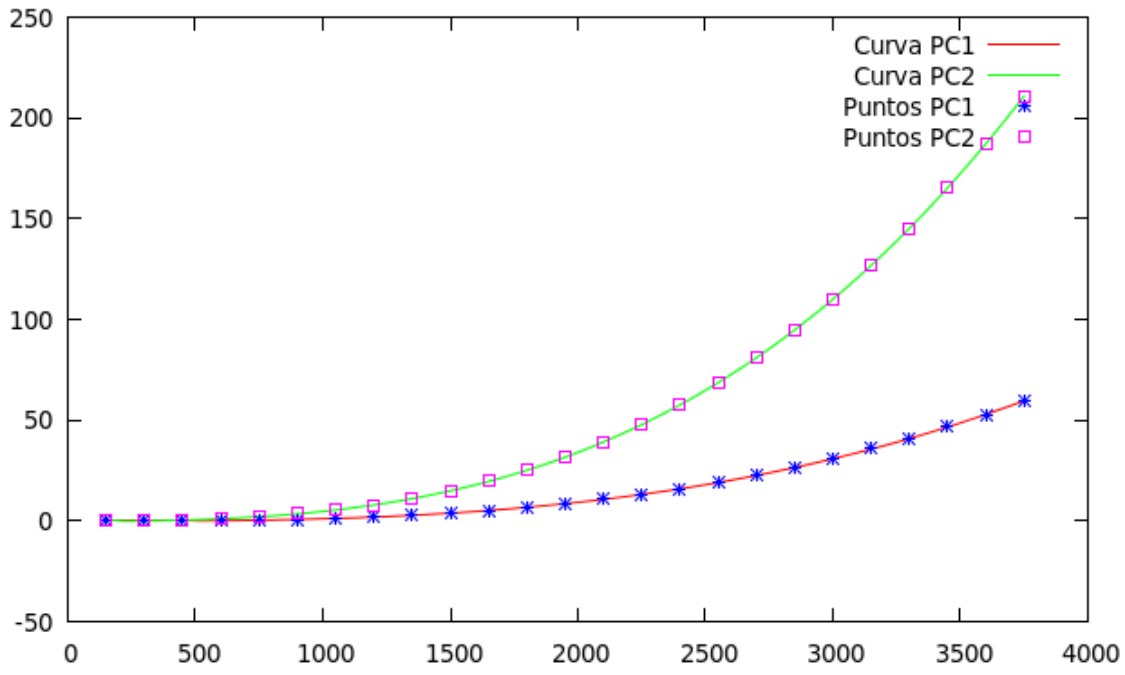
-PC1:

-PC2:

Representación Gráfica

Para esta demostración utilizamos el algoritmo de Floyd.

El resultado es el siguiente:



Comparación Diferente optimización

Propiedades

-Mismo algoritmo.

-Distinta optimización de código:

-Sin optimización.

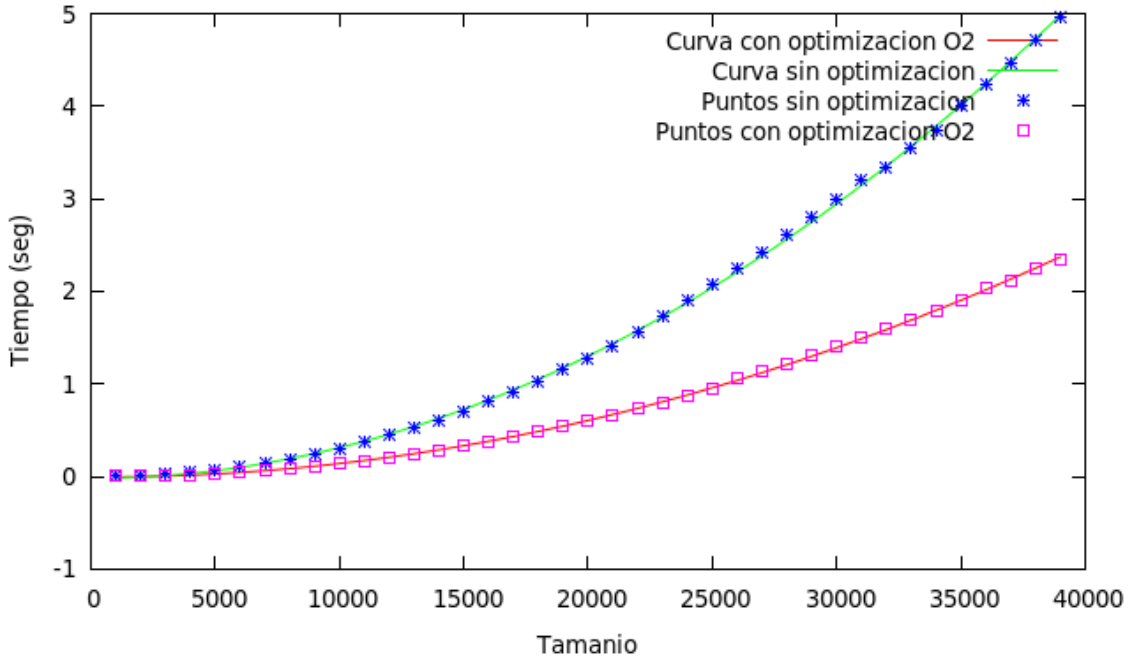
-Optimización O2.

-Mismo PC.

Representación Gráfica

Para esta demostración utilizamos el algoritmo de burbuja.

El resultado es el siguiente:



Comparación Algoritmos de ordenación

Propiedades

-Distintos algoritmos de ordenación:

-Inserción.

-Selección.

-Burbuja.

-Mergesort.

-Quicksort.

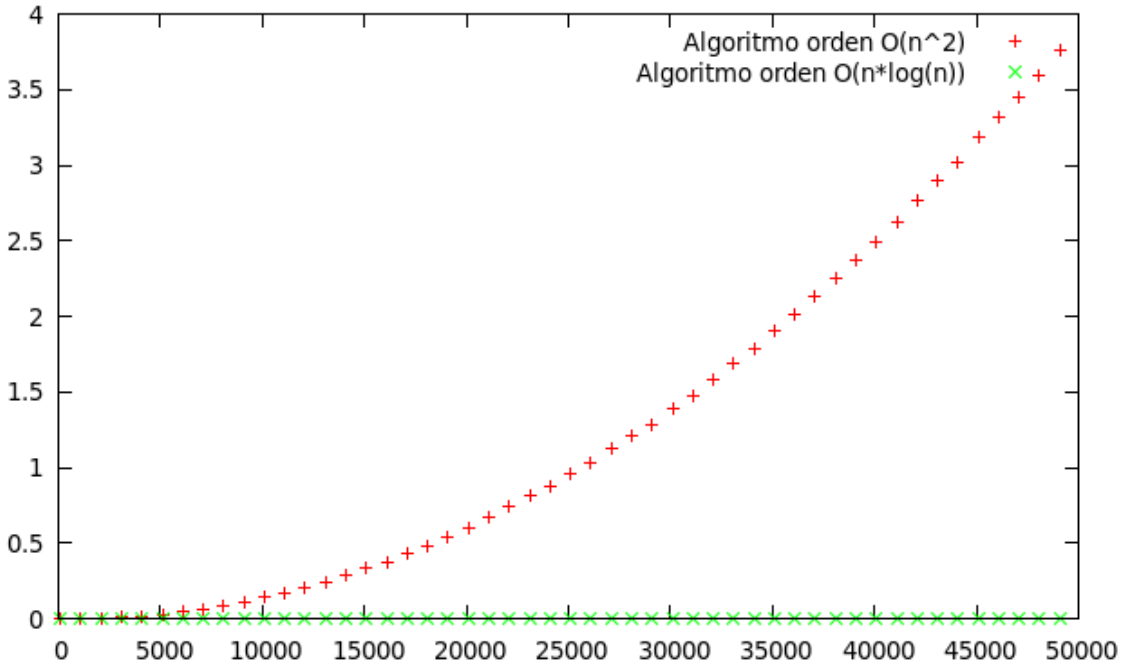
-Heapsort.

-Mismo PC.

-Misma optimización (en este caso, O2).

Representación Gráfica

Problema: los algoritmos O(n*log(n)) son demasiado eficientes en comparación con los O(n^2), por lo que no se ve nada claro un gráfico con estos dos algoritmos de diferente eficiencia:



Comparación Algoritmos de ordenación

Solución

Crear dos gráficas, una comparando los de O(n^2) y otro con los O(n*log(n))

Representación Gráfica