FUNDAMENTOS DE ANÁLISIS DE ALGORITMOS

PRÁCTICA 1

Adrián Moreno Monterde

CONTENIDO

1. INTRODUCCIÓN AL ALGORITMO DE BÚSQUEDA SECUENCIAL

En la primera práctica realizada de Fundamentos de Análisis de algoritmos vamos a trabajar utilizando la herramienta "Visual Studio", con la finalidad de estudiar y posteriormente analizar teórica y empíricamente el código de búsqueda secuencial de un vector dentro de un conjunto de vectores definidos. Asimismo, este algoritmo compara los tiempos para el caso peor, caso medio y caso mejor generando a su vez gráficas para verlo de una manera más visual.

La finalidad de esta practica es comprender lo estudiado en las sesiones de teoría y aplicar los conocimientos aprendidos a un programa concreto a la vez que mejoramos nuestras habilidades de programación aprendidas en asignaturas estudiadas con anterioridad.

A continuación, se mostrará los cálculos obtenidos del tiempo teórico y experimental junto con su tabla o gráficas y sus respectivas conclusiones, para posteriormente realizar una comparación de los resultados obtenidos.

2. CÁLCULO DEL TIEMPO TEÓRICO

A partir de la expresión del algoritmo, se aplicarán las reglas conocidas para contar el número de operaciones que realiza un algoritmo. Este valor será expresado como una función de T(n) que dará el número de operaciones requeridas para un caso concreto del problema caracterizado por tener un tamaño n. El análisis lo realizaremos para los casos mejor, peor y medio.

ALGORITMO DE BÚSQUEDA SECUENCIAL. Para determinar el tiempo de ejecución, calculamos primero el número de operaciones elementales (OE) que se realizan (en pseudocódigo o en código C++):

El tiempo de ejecución del algoritmo será:

TBSecuencial(n)= TAsig (1) + TBucle (2) + TSi(5)

TAsig (1) = 1

TSi (5) = TcondSi+ TcuerpoSi = 2+máx/mín./medio (Treturn(6ó7))=2+1=3

TBucle (2) = TcondB + ∑(i=1;?) TcicloBucle = 4+∑(i=1;?) TcicloBucle

TcicloBucle = TcondB +TcuerpoB(= 0 sólo instrucción de incremento del bucle)+TincrementoB = 4+2 = 6

TBucle (2) = 4+∑(i=1;?) 6=4+6∑(i=1;?) TBSecuencial (n)= TAsig (1) + TBucle (2) + TSi (5) = 1+4+6∑ (i=1;?) +3= 8+6∑(i=1;?)

$$T_{BSecuencial}(n) = 8+6\sum(i=1;?)$$

2.1. PSEUDOCÓDIGO (CÓDIGO) Y ANÁLISIS DE COSTE

CASO MEJOR

En el caso mejor para el algoritmo, se efectuará la línea (1) y la línea (2) que supone 4 OE Tras ellas la función acaba ejecutando las líneas (5) a (7). En consecuencia, el ciclo del bucle no se ejecuta nunca: ∑(i=1;0) = **0** y por tanto

$$T_{BSecuencial}(n) = 8+6\sum(i=1;0) = \mathbf{8}$$

CASO PEOR

En el caso peor sucede cuando el valor no se encuentra en el vector. Se efectúa la línea (1), el bucle se repite n veces hasta que se cumple la segunda condición, después se efectúa la condición de la línea (5) y la función acaba al ejecutarse la línea (7). Cada iteración del bucle está compuesta por las líneas 2 (4OE) y 3(2OE), junto con una ejecución adicional de la línea (2) que es la que ocasiona la salida del bucle. En consecuencia, el ciclo del bucle se ejecuta n veces: ∑(i=1; n) = **n** y por tanto,

$$T_{BSecuencial}(n) = 8+6\sum(i=1;n) = \mathbf{6n+8}$$

CASO MEDIO

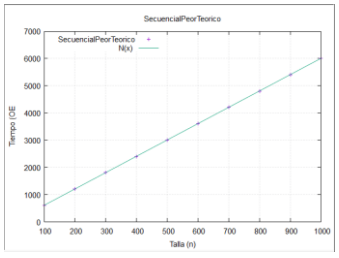
En el caso medio el bucle se ejecutará un número de veces entre 1 y n, y suponemos que cada una de ellas tiene la misma probabilidad de suceder. En consecuencia, el ciclo del bucle se ejecuta n/2 veces: ∑(i=1;n/2) = **n/2** y por tanto,

$$T_{BSecuencial}(n) = 8+6\sum(i=1;n/2) = 8+6(n/2) = \mathbf{3n+8}$$

2.2. TABLAS Y GRÁFICAS DE COSTE

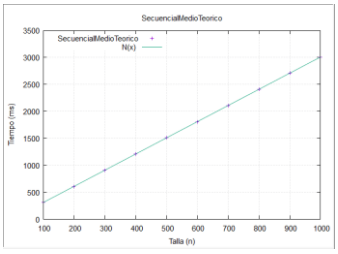
CASO PEOR

Talla	Tiempo (oe)
100	6.1e+02
200	1.2e+03
300	1.8e+03
400	2.4e+03
500	3e+03
600	3.6e+03
700	4.2e+03
800	4.8e+03
900	5.4e+03
1000	6e+03



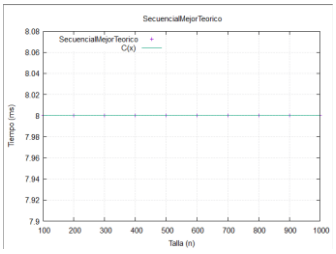
CASO MEDIO

Talla	Tiempo (oe)
100	3.1e+02
200	6.1e+02
300	9.1e+02
400	1.2e+03
500	1.5e+03
600	1.8e+03
700	2.1e+03
800	2.4e+03
900	2.7e+03
1000	3e+03



CASO MEJOR

Talla	Tiempo (oe)
100	8
200	8
300	8
400	8
500	8
600	8
700	8
800	8
900	8
1000	8

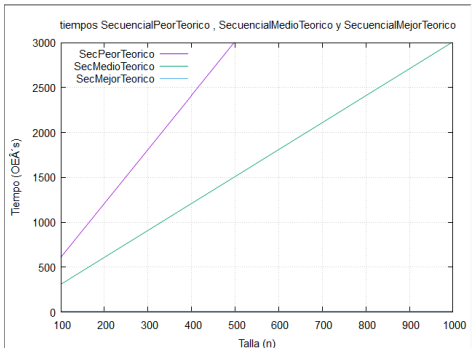


2.3. CONCLUSIONES

Su caso mejor se da cuando el elemento está en la primera posición del vector O(1).

El caso peor se produce cuando el elemento no está en el vector.

El caso medio ocurre cuando consideramos equiprobables cada una de las posiciones en las que puede encontrarse el elemento dentro del vector (incluyendo la posición especial -1, que indica que el elemento a buscar no se encuentra en el vector) pero ambos son de O



3. CÁLCULO DEL TIEMPO EXPERIMENTAL

3.1. TABLAS Y GRAFICAS DE COSTE

CASO PEOR

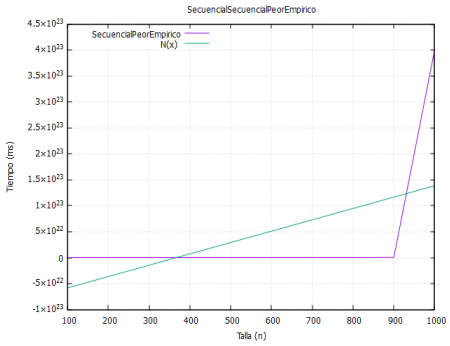
Talla	Tiempo (ms)
100	0.00014
200	0.00026
300	0.00076
400	0.0014
500	0.0016
600	0.0026
700	0.0034
800	0.0036
900	0.0041
1000	0.0055

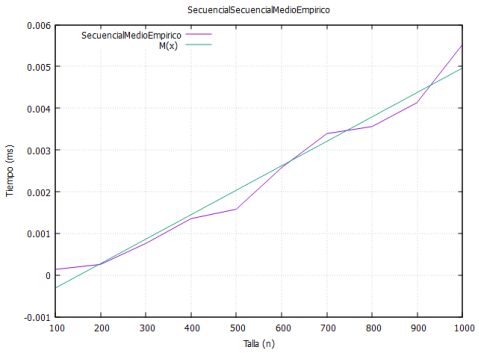
CASO MEDIO

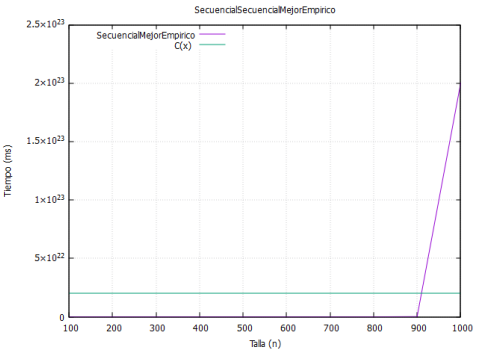
Talla	Tiempo (ms)
100	0.0004
200	0.4
300	4e+02
400	4e+05
500	4e+08
600	4e+11
700	4e+14
800	4e+17
900	4e+20
1000	4e+23

CASO MEJOR

Talla	Tiempo (ms)
100	0.0002
200	0.2
300	2e+02
400	2e+05
500	2e+08
600	2e+11
700	2e+14
800	2e+17
900	2e+20
1000	2e+23







3.2. CONCLUSIONES

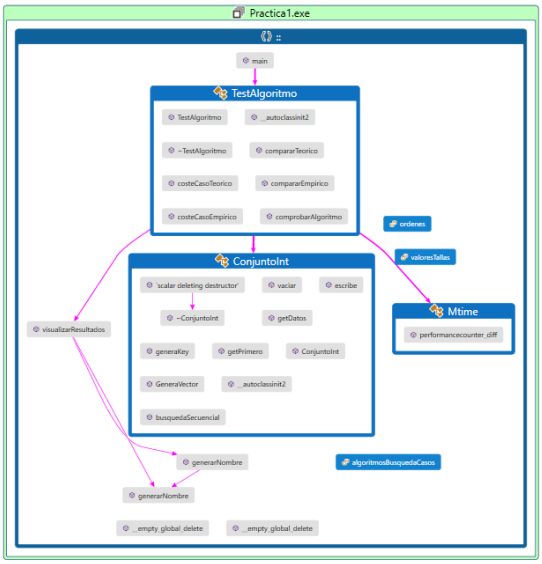
Como podemos observar, las diferentes gráficas tienen una gran variación de valores con respecto al eje y en el caso medio y caso peor. En cambio, el caso peor presenta unos resultados más proporcionales de manera ascendente. En conclusión, existen irregularidades en las gráficas de caso medio y caso peor.

4. COMPARACIÓN DE LOS RESULTADOS TEORICO Y EXPERIMENTAL

Al contrastar las distintas gráficas obtenidas en el estudio teórico y en el estudio empírico, llegamos a la conclusión de que, en el caso del estudio teórico, las gráficas son linealmente proporcionales, en los tres casos tenemos una línea recta, en cambio si nos fijamos en el estudio experimental, los resultados varían y las graficas son distintas, resaltando la del caso medio y la del caso mejor.

5. DISEÑO DE LA APLICACIÓN

El método escogido para la implementación fue el diseño modular con orientación a objetos.



Tenemos un main o programa principal llamado Principal, en el se hacen los menús y llamadas a los métodos necesarios de TestAlgoritmo, este a su vez se apoya en la clase ConjuntoInt y Mtime para medir tiempos transcurridos.

TestAlgoritmo tiene los siguientes métodos principales:

- CosteCasoTeorico:

  Recibe por parámetro un caso a estudiar de forma teórica, lo muestra por pantalla y guarda en un archivo los valores que toma la función complejidad para distintas tallas y casos.
- CompararTeórico:

  Recibe por parámetro los tres casos a comparar teóricamente. Muestra por pantalla y guarda en un archivo los valores de la función complejidad para todos los casos y diferentes tallas.
- CosteCasoEmpirico:

  Recibe por parámetro el caso empírico. Genera un conjunto de datos aleatorios, y mide el tiempo que transcurre entre el inicio y el fin de la llamada al algoritmo de búsqueda. Repite el proceso para distintas tallas, muestra los resultados por pantalla y los guarda en un fichero.

  - En el caso peor, se le introduce el dato de búsqueda -1, valor que no existe en el conjunto de datos.

  - En el caso medio, se escoge un dato aleatorio de entre los elementos contenidos en el conjunto. Las mediciones se repiten múltiples veces y se toma la media.

- En el caso mejor, se pasa el primer elemento del conjunto como clave.

- CompararEmpirico:

Recibe por parámetro los tres casos a simular empíricamente. Realiza las mismas mediciones de los tres casos, las muestra por pantalla y las guarda en un archivo.

La interfaz consta de un menú principal y dos submenús, para el estudio teórico y empírico, en ambos submenús se puede probar el algoritmo de búsqueda, generar resultados para un caso en particular o comparar los resultados de los tres casos.

6. VALORACIONES

Esta práctica nos ha servido para mejorar un poco más en nuestro nivel de programación además de seguir poniendo en práctica el diseño modular, profundizar en el uso de ficheros y tener una perspectiva mas centrada a la hora de analizar un algoritmo, el software “Gnuplot” ha sido bastante útil ya que nos ha ayudado a entender los cálculos que nos salían.