PRÁCTICA 1

FAA – 1º Ingeniería Informática

Eficiencia de algoritmos

Alba Márquez Rodríguez

3. Introducción. Algoritmo Búsqueda secuencial.

En esta práctica implementaremos un algoritmo que tiene como objetivo el análisis de la eficiencia de un algoritmo de búsqueda secuencial. Con el estudio teórico y empírico correspondiente a los 3 posibles casos (mejor, medio y peor).

La búsqueda secuencial consiste en la búsqueda de un valor en una tabla de valores. Para la elaboración del algoritmo se tienen en cuenta tablas de diferentes tamaños, desde 100 a 1000 con intervalos de 100. Así se puede observar la evolución de los tiempos.

El estudio teórico consiste en obtener la expresión de la función complejidad temporal (T(n)) del algoritmo. Esto lo debemos realizar nosotros con el cálculo de las operaciones elementales estudiados en clase, teniendo en cuenta los sumatorios correspondientes por bucles. Se debe tener en cuenta que hay 3 posibles casos: peor (no se encuentra, se realiza n veces el bucle), medio (se encuentra, pondremos que se realiza n/2 veces el bucle) y mejor (el bucle no se realiza ni una sola vez, por lo que sólo se hace 1 vez la condición del bucle).

El estudio empírico consiste en el estudio experimental del comportamiento del algoritmo. Midiendo el tiempo en los tres posibles casos: mejor, medio y peor. Este variará según la capacidad de cada ordenador y el uso que se esté haciendo en el momento de la memoria.

Lo esperado es que el tiempo del caso peor sea tanto de forma empírica como teórica el mayor de todos y la gráfica quede por encima de las otras dos. El caso mejor será constante ya que el valor siempre estará en primera posición y este tiempo será el menor. La gráfica quedará por abajo. El caso medio quedará entre las dos.

4. Cálculo del tiempo teórico:

Aplicando las reglas estudiadas en clase calcularemos el número de operaciones elementales del algoritmo. Lo pondremos como una función de T(n). Siendo n las veces que se ejecutará el bucle.

4.1. Pseudócódigo (código) y análisis de coste

Este es el pseudocódigo que se nos da en el enunciado del ejercicio, completamos con las operaciones elementales (asignaciones, accesos…):

lineas		int BusquedaSecuencial(int T[],int n,int valor)	nº OE	
		{		
1)		int i=0;	1	asignación
2)		while (T[i] != valor && i<n) {	4	Condición del Bucle (2comp.,1 ac. vector, 1 lóg.)
3)		i=i+1;	2	incremento y asignación
4)		}		
5)		if (T[i]==valor)	2	1 condición y 1 acc. al vector
6)		return i;	1	si la condición se cumple
7)		else return -1;	1	cuando condición es falsa.
		}		

Así obtenemos que T(n)= $T_{asignación}$+T_{Bucle}+T_{Si}, por separado nos queda que $T_{asignación}$=1, T_{SI}=3 y T_{bucle} = 5+7(Sum). Es decir:

$T_{Asig(1)} = 1$

$T_{Si(5)} = T_{condSi} + T_{cuerpoSi} = 2+máx/mín/medio(T_{return(6ó7)}) = 2+1=3$

$T_{Bucle(2)} = T_{condB} + T_{saltoB} + \sum(i=1;?)\ T_{cicloBucle} = 4+1+\sum(i=1;?)\ T_{cicloBucle}$

$T_{cicloBucle} = T_{condB} + T_{cuerpoB(= 0\ sólo\ instrucción\ de\ incremento\ del\ bucle)} + T_{incrementoB} + T_{saltoCicloB} = 4+2+1 = 7$

$T_{Bucle(2)} = 4+1+\sum(i=1;?)\ 7 = 5+7\sum(i=1;?)$

$\mathbf{T_{BSecuencial}(n) = T_{Asig(1)} + T_{Bucle(2)} + T_{Si(5)} = 1+5+7\sum(i=1;?) +3 = 9+7\sum(i=1;?)}$

$\mathbf{T_{BSecuencial}(n) = 9+7\sum(i=1;?)}$

La interrogación depende del caso en el que nos encontremos, así que será:

Caso mejor

El bucle se realiza 0 veces por lo que T(n)=9

Caso medio

Se ejecutará n veces el bucleo suponiendo que todas las posibilidades son igual de probables. Así el bucle se ejecutará n/2 veces. T(n)=(7/2)n+9.

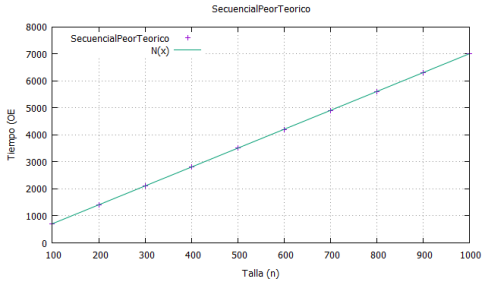
Caso Peor

El valor no se encuentra en el vector así el bucle se repite n veces. T(n)=7n+9

4.2. Tablas y Gráficas de coste

Caso Peor

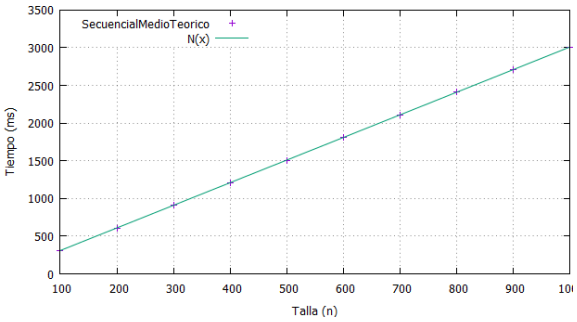
```
Busqueda SecuencialSecuencialPeor TeoricoTiempos de ejecucion

        Talla           Tiempo (oe)

        100               7.1e+002
        200               1.4e+003
        300               2.1e+003
        400               2.8e+003
        500               3.5e+003
        600               4.2e+003
        700               4.9e+003
        800               5.6e+003
        900               6.3e+003
        1000                7e+003

Datos guardados en el fichero SecuencialPeorTeorico.dat

Generar grafica de resultados? (s/n):
```



En este caso la tabla se recorre al completo sin éxito ya que el bucle se ejecuta n vece. Como se estudia en el caso en el que la talla mida de 100 a 1000 (en intervalos de 100) obtenemos una función de una recta con una pendiente. A mayor tamaño, mayor tiempo.

Caso Medio

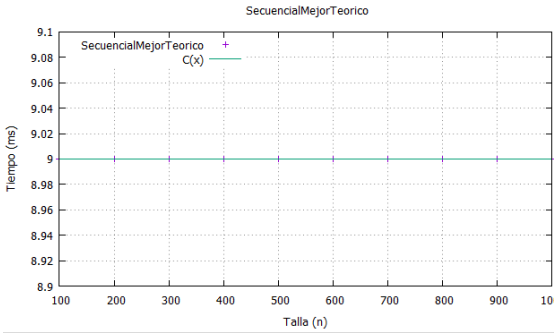
```
Busqueda SecuencialSecuencialMedio TeoricoTiempos de ejecucion

        Talla           Tiempo (oe)

        100               3.1e+002
        200               6.1e+002
        300               9.1e+002
        400               1.2e+003
        500               1.5e+003
        600               1.8e+003
        700               2.1e+003
        800               2.4e+003
        900               2.7e+003
        1000                3e+003

Datos guardados en el fichero SecuencialMedioTeorico.dat

Generar grafica de resultados? (s/n):
```



En este caso la tabla se recurre hasta que se encuentra el vector. Como es una media pusimos que se ejecutaría n/2 veces por lo que es la misma gráfica que caso peor pero con una escala a ½ (la mitad).

Caso Mejor

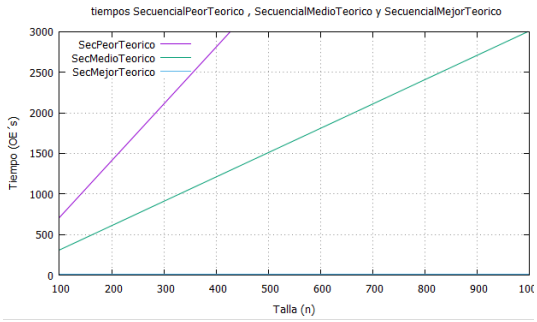
```
Busqueda SecuencialSecuencialMejor TeoricoTiempos de ejecucion

        Talla           Tiempo (oe)

        100                     9
        200                     9
        300                     9
        400                     9
        500                     9
        600                     9
        700                     9
        800                     9
        900                     9
        1000                    9

Datos guardados en el fichero SecuencialMejorTeorico.dat

Generar grafica de resultados? (s/n):
```



En este caso el tiempo será constante ya que el vector a buscar siempre estará en la primera posición de la tabla. Independientemente del tamaño de la tabla. Nos sale una recta horizontal.

4.3. Conclusiones

Tanto caso peor como medio tienen un crecimiento constante al tener una variable que va aumentando. Son proporcionales, siendo la de caso medio la mitad que la del caso peor (en la ecuación podemos observar que es porque tenemos n/2).



El caso mejor será siempre constante (=9) por lo que la gráfica se corresponde con una línea recta.

Además, se corresponde con lo esperando. La gráfica del caso peor es la que queda por arriba, la del caso medio entre las dos y la de caso mejor abajo.

5. Cálculo del tiempo experimental:

Para el estudio empírico tendremos que medir los recursos empleados (tiempo). Para ello elaboramos pruebas en diferentes condiciones (caso mejor, peor y caso medio que sería en el caso aleatorio). Para cada caso la medida computacional será el tiempo de ejecución.

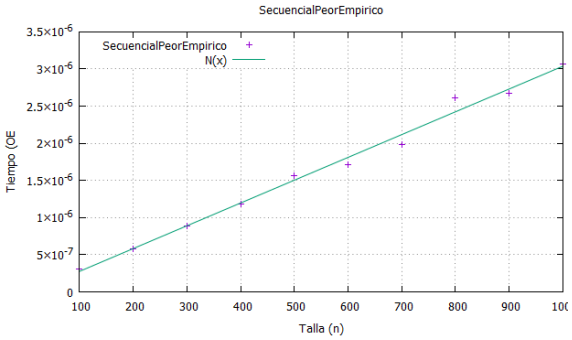
Para el caso “medio” al ser aleatorio introduciremos un bucle for que tome diferentes valores y luego haga una media. Haremos esto en caso peor y mejor también. Lo haremos con la constante NUMREPETICIONES.

Esto queda implementado gracias a la clase Mtime, con QueryPerformanceCounter obtenemos el tiempo inicial y final, hacemos la diferencia y así obtenemos el tiempo de ejecución del caso correspondiente. Haciendo uso de los métodos de la clase ConjuntoInt podemos obtener el tiempo mejor y medio. En el caso peor no se encuentra. Para el caso mejor es el primer dato de la tabla y para el caso medio usamos generaKey() que coge una al azar. Hacemos uso de un bucle for para calcular una media y que sea más exacto y real ese caso medio.

5.1. Tablas y Gráficas de coste

Caso Peor

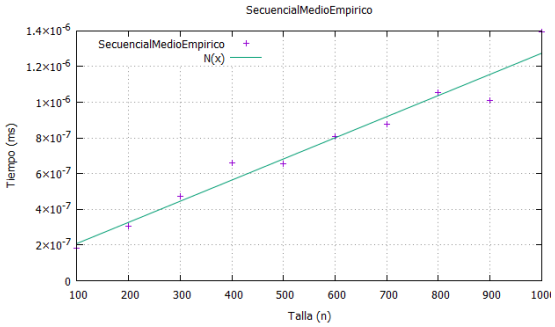
```
Busqueda SecuencialSecuencialPeor EmpiricoTiempos de ejecucion

        Talla           Tiempo (oe)

        100               3.1e-007
        200               5.8e-007
        300               8.9e-007
        400               1.2e-006
        500               1.6e-006
        600               1.7e-006
        700                 2e-006
        800               2.6e-006
        900               2.7e-006
        1000              3.1e-006

Datos guardados en el fichero SecuencialPeorEmpirico.dat

Generar grafica de resultados? (s/n): s
```



En este caso obtenemos una gráfica creciente, hay puntos que no coinciden con la recta. Esto se debe a que al tratarse de una medida empírica y al momento el ordenador puede tardar más o menos en el proceso (por programas que tengamos abiertos por ejemplo). Observaremos que esto pasa en el resto de casos.

Caso Medio

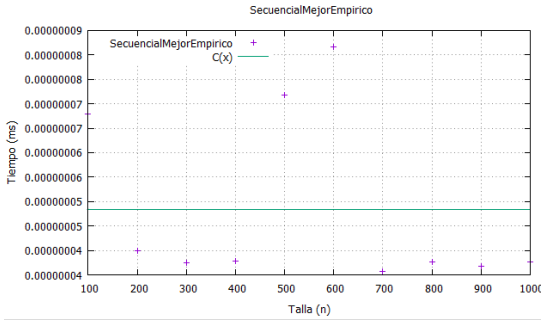
```
Busqueda SecuencialSecuencialMedio EmpiricoTiempos de ejecucion

        Talla           Tiempo (oe)

        100               1.8e-007
        200                 3e-007
        300               4.7e-007
        400               6.6e-007
        500               6.6e-007
        600               8.1e-007
        700               8.8e-007
        800               1.1e-006
        900                 1e-006
        1000              1.4e-006

Datos guardados en el fichero SecuencialMedioEmpirico.dat

Generar grafica de resultados? (s/n):
```



Como en el caso anterior obtenemos una recta creciente a medida que la talla de la tabla sea mayor, según el cálculo teórico debe ser la mitad al caso peor. Probamos con talla 100, la mitad de 3,1*10^-7 = 1,55*10^-7 que se aproxima a 1.8*10^-7. Como ya hemos dicho, al tratarse de un estudio empírico los resultados no serán exactos y se debe a que el valor no estará justo en el centro de la tabla, a pesar de haber hecho un bucle for para que el valor sea lo más aproximado posible.

Caso Mejor

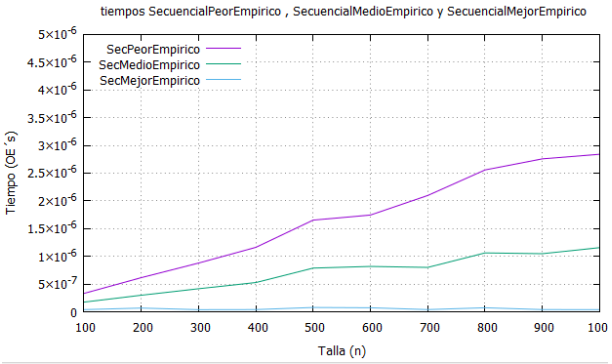
```
Busqueda SecuencialSecuencialMejor EmpiricoTiempos de ejecucion

        Talla           Tiempo (oe)

        100               6.8e-008
        200                 4e-008
        300               3.8e-008
        400               3.8e-008
        500               7.2e-008
        600               8.2e-008
        700               3.6e-008
        800               3.8e-008
        900               3.7e-008
        1000              3.8e-008

Datos guardados en el fichero SecuencialMejorEmpirico.dat

Generar grafica de resultados? (s/n):
```



En este caso se supone que el tiempo es constante, son valores muy pequeños con variaciones mínimas, los picos se deben a lo explicado en los otros dos casos. Comparada con las otras dos gráficas estos picos deben ser mínimos.

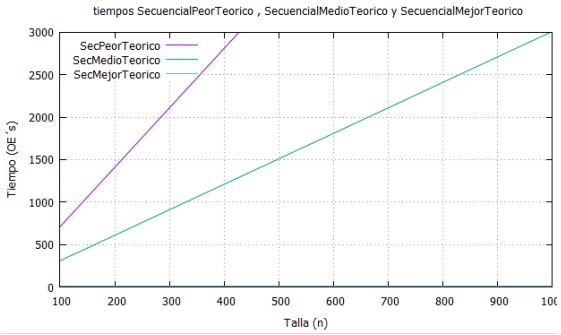
5.2. Conclusiones

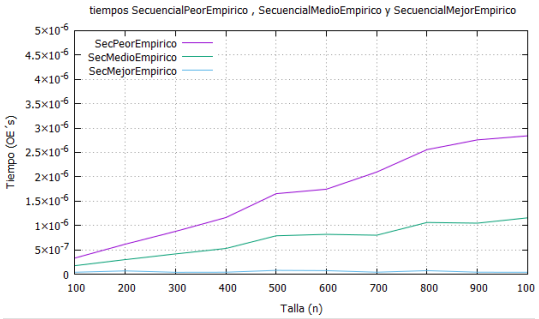
En este caso estamos usando el tiempo de ejecución del ordenador. Las gráficas deben seguir un crecimiento constante pero sin embargo nos encontramos con picos. Esto se debe a que podemos estar haciendo uso del ordenador con otros programas abiertos (por ejemplo, codeblocks, Spotify, opera…).



Aún así los resultados son los esperados. El caso peor es el superior, el mejor el inferior y el medio el central. Además supusimos que en el caso mejor los picos serían mínimos comparados con las otras gráficas y así ha sido, eran variaciones muy pequeñas.

6. Comparación de los resultados teórico y experimental.





Lo primero en lo que nos debemos fijar es en las escalas. Podemos observar que la escala de la teórica es mucho más grande que la empírica (orden de 10 frente a 10^-6). Teniendo en cuenta que esto se debe al tiempo de ejecución de cada ordenador. Los resultados son coherentes. Siendo ambas gráficas más o menos proporcionales teniendo en cuenta las escalas. El caso mejor es una recta horizontal (en el caso empírico tiene leves modificaciones por lo que explicamos en el punto 5.2) al tratarse de valores constantes. En el caso medio y peor obtenemos rectas crecientes (a mayor tamaño más tiempo tarda). Quedando el peor por encima de todas las gráficas, el mejor el más bajo y el medio entre ambas.

7. Diseño de la aplicación.

(Mostrar un esquema gráfico global de la estructura de tipos de datos existentes. Detallar la descomposición modular del programa, qué módulos existen, cuál es la responsabilidad de cada uno y la relación de uso. Documentar cualquier otra decisión de diseño que pueda resultar de interés. Funcionamiento y explicación de los métodos implementados en la práctica.)

El programa se compone de 3 clases: ConjuntoInt, Mtime y TestAlgoritmo además de un fichero de constantes.

En el fichero constantes, como su nombre indica, de definen unas constantes para recurrir a ellas y simplificar el programa.

La clase ConjuntoInt nos permite crear vectores de tamaños variables. Se genera el vector de forma aleatoria para el tamaño dado.

La clase Mtime nos permite obtener el valor del contador y la frecuencia. Que nos permite obtener el tiempo para medir la eficiencia del algoritmo de forma empírica.

La clase TestAlgoritmo es la que contiene las funciones principales del programa que se implementan en el main. Tiene un constructor que inicia con los nombres de los 3 posibles casos (peor, medio y mejor).

El método comprobar algoritmo lo comparten el teórico y el empírico. Lo que hace es generar una tabla con x vectores. Si ejecutamos el programa nosotros podremos introducir el tamaño de la tabla. Acto seguido llena la tabla con valores aleatorios y nos pregunta por el valor a buscar. Así podemos observar el tiempo que tarda en buscar un valor en concreto (el primero, en el caso de que no esté o cualquier otro).

El main está compuesto de 1 menú principal que nos permite acceder a 2 submenús: teórico o empírico. Ambos tienen las mismas opciones pero se aplican los métodos correspondientes. En ambos la opción 2 abre otro submenú con 4 posibles casos (caso peor, medio, mejor y volver). En las opciones de obtener los casos y comparar casos nos pregunta si queremos abrir una gráfica. Al seleccionar que sí se nos abre un fichero de gnuplot con las gráficas correspondientes. Además se guardan en un archivo de pdf. Para salir del programa simplemente tendremos que seleccionar volver y una vez nos hallemos de nuevo en el menú principal: salir.

8. Conclusiones y valoraciones personales de la práctica.

Esta práctica nos ha permitido comprobar, comparar y aprender como calcular el tiempo que tarda un programa en ejecutarse en los 3 posibles casos. Lo que debíamos implementar nosotros era el main, con sus menús correspondientes y haciendo las llamadas a los métodos necesarios. Y en TestAlgoritmo implementar los métodos empíricos.

Ha sido una práctica interesante aunque al principio sin la guía del profesor no la comprendía bien. Gracias a los vídeos además de asegurarme de haber hecho las cosas por mi cuenta correctamente pude entender conceptos que quizás no había terminado de comprender.

Por otro lado al principio se hacen uso de punteros y memoria dinámica. Esto apenas lo habíamos visto en ED pero a medida que avanzamos en ambas asignaturas comprendí el uso aunque cuesta la implementación y hacer uso de ellos ya que apenas estamos viéndolos ahora y no es algo que esté muy interiorizado. Pero con prácticas como estás se nos obliga a usarlos y ver un uso más práctico y ayuda a su comprensión.