Memoria Práctica 1

Algoritmo de Búsqueda Secuencial

Fundamentos de Análisis de Algoritmos

Álvaro Esteban Muñoz y Francisco José Gil Durán

Grado en Ingeniería Informática (1º Curso)



# Índice

[*0.* *Introducción. Algoritmo de Búsqueda secuencial.* 3](#_Toc5205287)

[*1.* *Cálculo del tiempo teórico.* 3](#_Toc5205288)

[*1.1.* *Pseudocódigo y análisis del coste.* 3](#_Toc5205289)

[*2.1.1* *Caso Mejor.* 4](#_Toc5205290)

[*2.1.2* *Caso Peor.* 4](#_Toc5205291)

[*2.1.3 Caso Medio.* 4](#_Toc5205292)

[*2.2* *Tablas y Gráficas de coste.* 4](#_Toc5205293)

[*2.3* *. Conclusiones.* 5](#_Toc5205294)

[*3.* *Cálculo del tiempo experimental.* 5](#_Toc5205295)

[*3.1.* *Tablas y Gráficas de coste.* 5](#_Toc5205296)

[*3.2.* *Conclusiones.* 6](#_Toc5205297)

[*4.* *Comparación de los resultados teórico y experimental.* 6](#_Toc5205298)

1. Introducción. Algoritmo de Búsqueda secuencial.

El objetivo de la práctica es realizar un análisis del algoritmo de búsqueda secuencial el cual consiste en la búsqueda de un dato en un vector de longitud n de datos del mismo tipo. En dicho algoritmo se pasan como parámetros el vector, la longitud de este y el dato a buscar.

1. Cálculo del tiempo teórico.
   1. Pseudocódigo y análisis del coste.

Vamos a proceder a realizar un análisis de la eficiencia del algoritmo, para ello debemos realizar tanto un análisis teórico como experimental.

Analizaremos el coste del tiempo de nuestro algoritmo.

|  |  |  |  |
| --- | --- | --- | --- |
| **Líneas** | Int BúsquedaSecuencial(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(2 comp, 1 acceso y 1 lóg.) |
| 3 | i=i+1; | 2 | Incremento y asignación |
| 4 | } |  |  |
| 5 | If (T[i]==valor) | 2 | 1 condición y 1 acceso al vector |
| 6 | return i; | 1 | Si la condición se cumple |
| 7 | else return -1; | 1 | Si la condición es falsa |
|  | } |  |  |

Cuando sumamos el tiempo que tardan sus diferentes partes, obtenemos lo que tardaría el algoritmo en ejecutarse

TAlgoritmo = TAsig(1) + TBucle(2) + TSi(5)

TAsig(1) = 1

TBucle(2) = TCond + TSalto + Σ(i=1;?) TCicloBucle = 4 + 1 + Σ(i=1;?)

TCicloBucle = TCondB + TCuerpoB(= 0 solo instrucción de incremento del bucle) + TIncrementoB + TSaltoCicloB = 4 + 2 + 1 = 7

TBucle(2) = 4 + 1 + Σ(i=1;?)7 = 5 + 7 Σ(i=1;?)

TSi(5) = TCondSi + TCuerpoSi = 2 + máx/mín/medio(Treturn(6o7)) = 2 + 1 = 3

**TBsecuencial(n)** = TAsig(1) + TBucle(2) + TSi(5) = 1 + 5 + 7 Σ(i=1;?) + 3 = **9 + 7 Σ(i=1;?)**

* + 1. Caso Mejor.

El mejor caso es siempre cuando el valor que buscamos se encuentra en la primera posición independientemente del tamaño del vector.

**TBS(n)** = 9 + 7\*Σ(i=0;0) = **9**

* + 1. Caso Peor.

Este caso se desarrolla cuando el dato a buscar no se encuentra en el vector, es decir, se recorre el vector entero.

**TBS(n)** = 9 + 7\* Σ(i=0;n) = **7n + 9**

2.1.3 Caso Medio.

En el caso medio el bucle se ejecutará entre 1 y n veces, como sabemos que todos los casos tienen la misma posibilidad, decimos que el ciclo del bucle se ejecuta n/2

**TBS(n)** = 9 + 7\* Σ (i=0;n/2) = **(7/2)n + 9**

* 1. Tablas y Gráficas de coste.

Podemos obtener una tabla de los distintos resultados para cada caso a partir de las funciones obtenidas para cada caso:

|  |  |  |  |
| --- | --- | --- | --- |
| Talla | Caso Mejor | Caso Medio | Caso Peor |
| 1000 | 9 | 3509 | 7009 |
| 2000 | 9 | 7009 | 14009 |
| 3000 | 9 | 10509 | 21009 |
| 4000 | 9 | 14009 | 28009 |
| 5000 | 9 | 17509 | 35009 |
| 6000 | 9 | 21009 | 42009 |
| 7000 | 9 | 24509 | 49009 |
| 8000 | 9 | 28009 | 56009 |
| 9000 | 9 | 31509 | 63009 |
| 10000 | 9 | 35009 | 70009 |

Si observamos la gráfica que dibujan estos tres casos nos daremos cuenta que a excepción del mejor caso, los demás aumenta conforme aumenta n.

* 1. . Conclusiones.

Al ver la gráfica nos damos cuentas que las funciones en los tres casos son rectas del tipo **ax + b** donde x es la pendiente o en nuestro caso, las veces que se ejecuta el bucle, además esto cobra más sentido si observamos que en nuestra función **a** toma los valores 0, 7 y 7/2, mientras que **b** siempre es 9.

A diferencia del caso teórico en el caso experimental la gráfica no tendrá rectas tan bien definidas, serán resultados más variados y atípicos.

1. Cálculo del tiempo experimental.

Las mediciones en este caso se toman del tiempo de ejecución y variarán dependiendo del tamaño del vector, por eso, no nos saldrán unos resultados tan constantes e ideales.

* 1. Tablas y Gráficas de coste.

Una vez realizadas las mediciones necesarias para cada caso con diferentes valores de la talla (usamos los mismos que en el cálculo experimental para poder comparar mejor)

Para diferenciar los valores de cada caso sometemos al algoritmo a diferentes situaciones, en el mejor caso le ponemos que busque siempre el primer elemento del vector mientras que en el peor caso le dimos siempre un elemento que no se encontraba en el vector, de esta forma lo recorrería entero.

Para finalizar en el caso medio dimos diferentes valores aleatorios que podía estar en el vector o no, repetimos este proceso varias veces para cada talla y realizamos una media del tiempo para más o menos hacernos una idea de cuanto tardaría para cada talla aproximadamente. Estos son los resultados obtenidos en milisegundos:

|  |  |  |  |
| --- | --- | --- | --- |
| Talla | Caso Mejor | Caso Medio | Caso Peor |
| 1000 | 0 |  |  |
| 2000 | 0 |  |  |
| 3000 | 0 |  |  |
| 4000 | 0 |  |  |
| 5000 | 0 |  |  |
| 6000 | 0 |  |  |
| 7000 | 0 |  |  |
| 8000 | 0 |  |  |
| 9000 | 0 |  |  |
| 10000 | 0 |  |  |

A continuación se muestra una gráfica de los resultados empíricos para poder observar el comportamiento del algoritmo a medida que se varía la talla

**-Caso Mejor:**

Observamos que es una recta debido a que el tiempo que tarda en encontrar el primer elemento del vector es siempre el mismo y no solo eso, sino que además el tiempo son prácticamente 0 ms.

**-Caso Peor:**

En este caso observamos también algo que se asemeja a una recta bastante bien definida puesto que el tiempo que tarda en recorrer el vector entero es constante pero aumenta a medida que aumenta la talla, es decir, el tiempo que tarda el ordenador en recorrer el vector de talla **n** es directamente proporcional a **n**.

**-Caso Medio:**

Puesto que los resultados son aleatorios, los resultados no son uniformes.

* 1. Conclusiones.

Los casos mejor y peor producen resultados uniformes debido a que siempre ejecutan el bucle el mismo número de veces, sin embargo en el caso medio esto no es así, notamos que los resultados son prácticamente aleatorios llegando a verse en la gráfica que no necesariamente una talla más grande hace que tarde más en encontrarse el elemento

1. Comparación de los resultados teórico y experimental.

Como podemos observar en las gráficas, el estudio teórico nos aporta unos resultados ideales, los cuales al ser repetidos en una situación experimental se enfrentan a multitud de situaciones diferentes que modifican los datos teóricos tales como el uso de la CPU durante la ejecución del programa o el simple hecho de la aleatoriedad generada por el algoritmo. Por ello los casos mejor y peor se asemejan tanto a los cálculos teóricos, esto es debido a que las funciones en estos casos son constantes sin embargo en el caso medio, es una media de una serie de experimentos aleatorios.

1. Diseño de la aplicación.
2. Conclusiones y valoraciones personales de la práctica.