Prácticas de Algorítmica.
3º de Grado en Ingeniería Informática.
Curso 2014-2015.

Práctica 1.

Objetivos.

Con esta práctica se pretende que el alumno se familiarice con el cálculo de tiempos de ejecución de un determinado algoritmo en función del tamaño del ejemplar y hacer una estimación empírica de esos tiempos en función de dicho tamaño. Para ello, el alumno deberá implementar un programa en C++ donde se calculen los tiempos de ejecución de dos algoritmos de ordenación y posteriormente se estime, utilizando un enfoque híbrido, la complejidad computacional de esos métodos.

Enunciado.

Implementad en C++ un programa que permita obtener los tiempos de ejecución de dos métodos de ordenación y a partir de esos tiempos estimar su complejidad computacional usando un enfoque híbrido. Para ello se implementará un método no sofisticado y otro sofisticado. Los elementos del conjunto a ordenar se almacenarán en un vector de la STL.

De entre los métodos no sofisticados se seleccionará uno de los siguientes, en función del último dígito del dni (suponemos que el dígito es d8):

1. *Método de inserción. (si d8 mod 5 == 0)*
2. *Método de inserción binaria. (si d8 mod 5 == 1)*
3. *Método burbuja. (si d8 mod 5 == 2)*
4. *Método de la sacudida. (si d8 mod 5 == 3)*
5. *Método de selección. (si d8 mod 5 == 4)*

De entre los métodos sofisticados se seleccionará uno de los siguientes, en función del penúltimo dígito del dni (suponemos que el dígito es d7):

6. *Método Shell. (si d7 mod 5 == 0)*
7. *Método de ordenación por montículo. (si d7 mod 5 == 1)*
8. *Método quicksort recursivo. (si d7 mod 5 == 2)*
9. *Método quicksort no recursivo. (si d7 mod 5 == 3)*
10. *Método por contabilización de frecuencias. (si d7 mod 5 == 4)*

Para realizar las pruebas y calcular los tiempos, se procederá como sigue:

1. *El usuario introducirá, en línea de comandos, el valor mínimo del número de elementos del vector, el valor máximo, el incremento del valor del número de elementos y el número de veces que se repetirá la ordenación para cada valor del número de elementos. Por ejemplo, si el mínimo es 1000, el máximo es 5000, el incremento es 100 y el número de repeticiones es 50, se probará primero con 1000 elementos, repitiendo el experimento 50 veces, después con 1100 y se repite 50 veces y así hasta llegar a 5000.*
2. *El vector será de elementos de tipo entero y se rellenará aleatoriamente con valores entre 0 y 9999. Para ello usad la función* **`void rellenarVector(vector<int> &v, const int &n);`**
3. *Si el número de repeticiones es por ejemplo 50, para cada valor de* **n** *entre el mínimo y el máximo se harán 50 pruebas de tal forma que si n=1200 se harán 50 pruebas para 1200 elementos en los dos métodos. Posteriormente se calculará la media de tiempos de esas 50 pruebas y ese será el valor correspondiente de*

tiempo empleado para ese ***n=1200****. Esto se hará para todos los posibles valores de* ***n****. En la documentación se adjunta un ejemplo para ver como se calculan los tiempos.*

4. *Implementad una función que, utilizando asertos, compruebe que la ordenación se ha realizado correctamente (****bool estaOrdenado(const vector <int> v);****)*
5. *Una vez obtenidos y almacenados los tiempos de ambas ordenaciones éstos se ajustarán por mínimos cuadrados usando las siguientes funciones, en base al ajuste que queramos realizar:*
   1. *Para el ajuste lineal o log-lineal usar la función* ***void calcularAjusteLineal(const vector<double> &x, const vector<double> &y, double &a0, double &a1, double &r2);*** *donde* ***x*** *será el vector que almacena los números de elementos del vector de cada prueba (****n*** *para regresión lineal,* ***nln(n)*** *para regresión log-lineal),* ***y*** *almacenará un valor obtenido a partir del tiempo, en función del tipo de regresión que se use calculados en cada prueba;* ***a0*** *y* ***a1*** *serán los parámetros de la recta de regresión que se calcularán en esta función y* ***r2*** *será el coeficiente de determinación que se calculará también en esta función. Como resultado obtendremos dos rectas de regresión (una para cada método).*
   2. *Para el ajuste cuadrático (****n²****) usar la función* ***void calcularAjustePolinomico(const vector<double> &x, const vector<double> &y, double &a0, double &a1, double &a2, double &r2);*** *donde* ***x*** *será el vector que almacena los números de elementos del vector de cada prueba (****n*** *para regresión lineal,* ***nln(n)*** *para regresión log-lineal),* ***y*** *almacenará un valor obtenido a partir del tiempo, en función del tipo de regresión que se use calculados en cada prueba;* ***a0****,* ***a1*** *y* ***a2*** *serán los parámetros del polinomio de segundo grado ajustado que se calcularán en esta función y* ***r2*** *será el coeficiente de determinación que se calculará también en esta función. Como resultado obtendremos dos parábolas (una para cada método).*
6. *Ahora, teniendo en cuenta las curvas de ajuste obtenidas en el paso 5, se calcularán los tiempos estimados a partir de dichas curvas usando las funciones (en su caso):*
   1. ***void calcularTiemposEstimadosLineales(const vector<double> &x, const double &a0, const double &a1, vector<double> &yEstimada);*** *donde* ***x*** *será el vector de los números de elementos (búsqueda secuencial) ,* ***a0*** *y* ***a1*** *los parámetros de la recta de regresión, el vector* ***yEstimada*** *almacenará los tiempos estimados con la regresión.*
   2. ***void calcularTiemposEstimadosCuadraticos(const vector<double> &x, const double &a0, const double &a1, vector<double> &yEstimada);*** *donde* ***x*** *será el vector de los números de elementos (búsqueda secuencial) ,* ***a0, a1*** *y* ***a2*** *los parámetros de la regresión cuadrática, el vector* ***yEstimada*** *almacenará los tiempos estimados con la regresión.*
7. *Una vez realizado el paso 6, tendremos que para cada uno de los métodos y cada posible valor de* ***n*** *se tiene un tiempo real (el obtenido en el paso 3) y un tiempo estimado por el modelo (obtenido en el paso 6). En el siguiente apartado se detalla la forma de representar graficamente los datos reales y los estimados por el modelo.*
8. *Una vez obtenidos los tiempos estimados, se guardarán en un fichero de texto, para poder representarlos posteriormente usando el programa* ***gnuplot*** *(se suministra un ejemplo de uso). Se guardará por columnas en el siguiente orden: números de elementos probados, los tiempos calculados en el método no sofisticado, los tiempos estimados en el método no sofisticado, los tiempos calculados en el método sofisticado, los tiempos estimados en el método sofisticado. Para ello usar la función* ***void guardarTiempos(const vector <double> n, const vector<double> &tNS, const vector<double> &tNSE, const vector <double> &tS, const vector <double> &tSE, string fichero);***
9. *Para finalizar, el programa ha de mostrar las ecuaciones de la regresión obtenidas, sus coeficientes de determinación y preguntará al usuario si quiere hacer una estimación de tiempos para un determinado valor del número de elementos, en cuyo caso mostrará el tiempo de ese estimación en días. Esta opción ha de poder repetirse hasta que el usuario introduzca un número de elementos igual a 0.*

- ***Notas estadísticas:***

  Para estimar los parámetros de un ajuste polinómico de orden m (en nuestro caso m vale 1 o 2) se puede usar el siguiente sistema de acuaciones (http://es.slideshare.net/diegoegas/regresion-polinomial-2512264):

$$
\begin{aligned}
a_0 n &+ a_1 \sum x_i &+ a_2 \sum x_i^2 &+ \dots &+ a_m \sum x_i^m &= \sum y_i \\
a_0 \sum x_i &+ a_1 \sum x_i^2 &+ a_2 \sum x_i^3 &+ \dots &+ a_m \sum x_i^{m+1} &= \sum x_i y_i \\
a_0 \sum x_i^2 &+ a_1 \sum x_i^3 &+ a_2 \sum x_i^4 &+ \dots &+ a_m \sum x_i^{m+2} &= \sum x_i^2 y_i \\
\vdots & \quad \vdots & \vdots & \quad \vdots & \vdots & \quad \vdots \\
a_0 \sum x_i^m &+ a_1 \sum x_i^{m+1} &+ a_2 \sum x_i^{m+2} &+ \dots &+ a_m \sum x_i^{2m} &= \sum x_i^m y_i
\end{aligned}
$$

-

El coeficiente de determinación será la varianza de la y estimada mediante el ajuste(tiempos estimados mediante el ajuste) dividida entre la varianza de la y observada (tiempos observados).

Fecha de comienzo: 17 de setiembre de 2015.
Fecha de Entrega: 8 de Octubre de 2015.