EJERCICIO 8 : Ordenación por mezcla

Autor : Jesús Ruiz Castellano, 76439001-L

1.- Código fuente : mergesort.cpp

```
138 const int UMBRAL_MS = 1000;
139
140 void mergesort(int T[], int num_elem)
141 {
142   mergesort_lims(T, 0, num_elem);
143 }
144
145 static void mergesort_lims(int T[], int inicial, int final)
146 {
147   if (final - inicial < UMBRAL_MS)
148     {
149       insercion_lims(T, inicial, final);
150     } else {
151       int k = (final - inicial)/2;
152
153       int * U = new int [k - inicial + 1];
154       assert(U);
155       int l, l2;
156       for (l = 0, l2 = inicial; l < k; l++, l2++)
157         U[l] = T[l2];
158       U[l] = INT_MAX;
159
160       int * V = new int [final - k + 1];
161       assert(V);
162       for (l = 0, l2 = k; l < final - k; l++, l2++)
163         V[l] = T[l2];
164       V[l] = INT_MAX;
165
166       mergesort_lims(U, 0, k);
167       mergesort_lims(V, 0, final - k);
168       fusion(T, inicial, final, U, V);
169       delete [] U;
170       delete [] V;
171     };
```

EJERCICIO 8 : Ordenación por mezcla

2.- Hardware usado:

2.1- CPU

vendor_id : GenuineIntel
model name : Intel(R) Core(TM) i3 CPU M 330 @ 2.13GHz
cpu MHz : 933.000

2.2- Velocidad de Reloj

Versión : hwclock de util-linux 2.20.1
jue 13 oct 2016 17:40:47 CEST -0.562493 segundos

2.3- Memoria RAM

MemTotal : 3907668 kB
SwapTotal : 4049916 kB

3.- Sistema Operativo

Ubuntu 14.04.3 LTS
Arquitectura : x86_64 (64 bits)

4.- Compilador usado y opciones de compilación

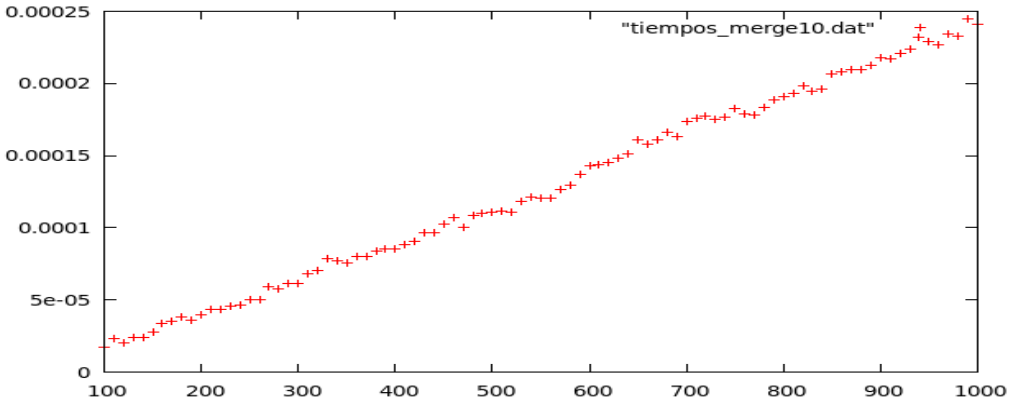
gcc - GNU project C and C++ compiler
Opción de compilación : g++ -o <nombre_ejecutable> <ejecutable.cpp>
g++ -o mergesort10 mergesort.cpp
g++ -o mergesort100 mergesort.cpp
g++ -o mergesort500 mergesort.cpp
g++ -o mergesort1000 mergesort.cpp

EJERCICIO 8 : Ordenación por mezcla

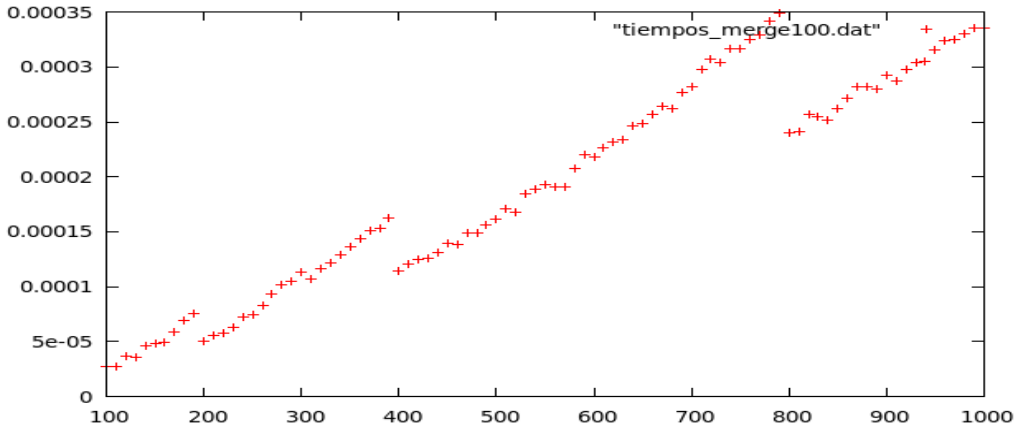
5.- Parámetros usados para el cálculo de la eficiencia empírica y gráfica

He ejecutado el programa con los siguientes valores de la variable UMBRAL_MS : 10, 100, 300, 500 y 1000.

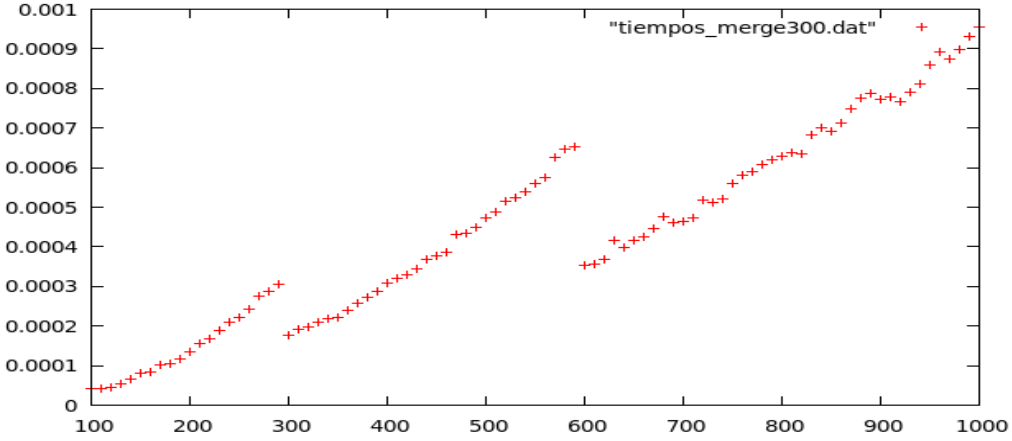
5.1- Gráfica de tiempos para la ejecución del programa con UMBRAL_MS = 10



5.2- Gráfica de tiempos para la ejecución del programa con UMBRAL_MS = 100

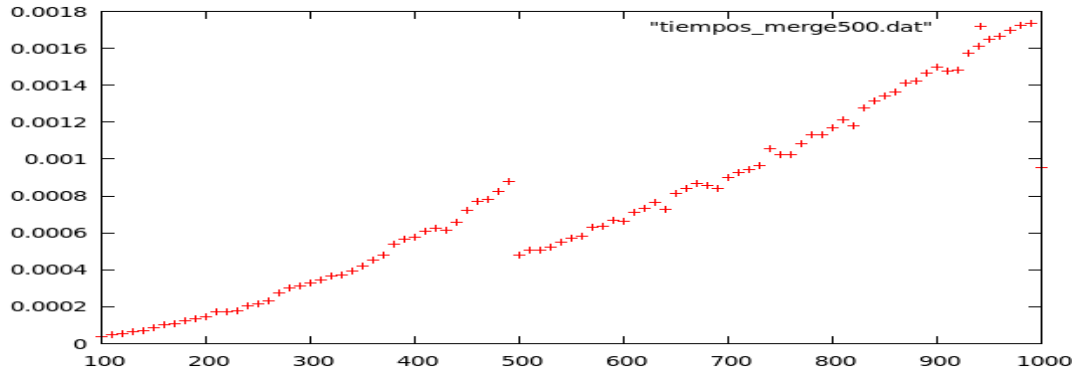


5.3- Gráfica de tiempos para la ejecución del programa con UMBRAL_MS= 300

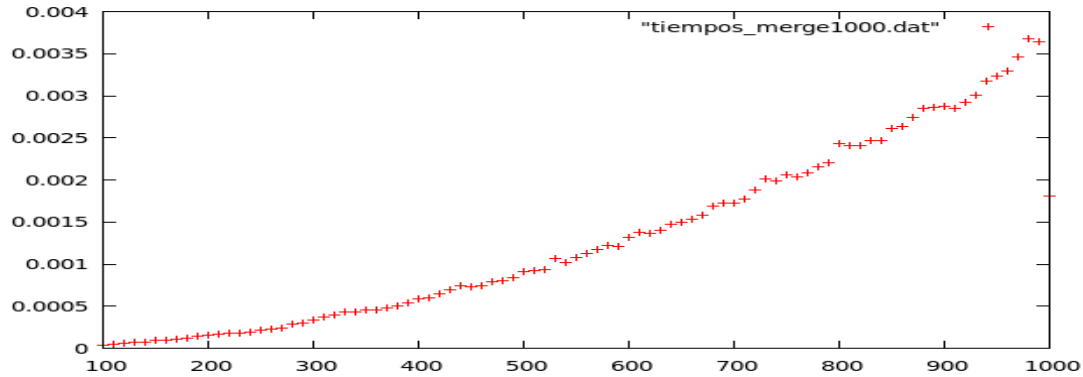


EJERCICIO 8 : Ordenación por mezcla

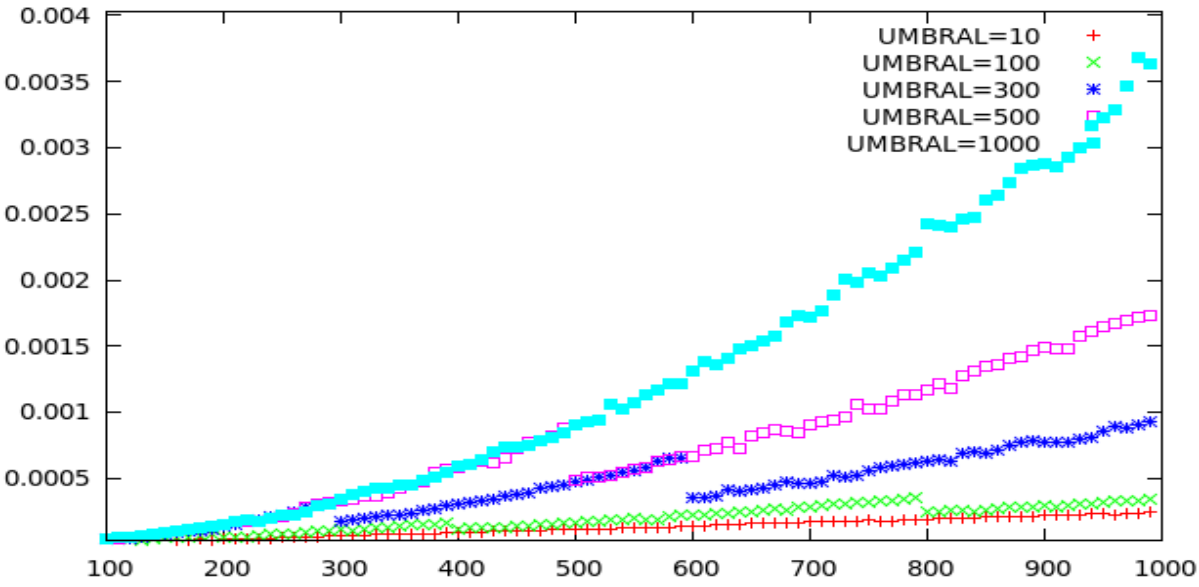
5.4- Gráfica de tiempos para la ejecución del programa con UMBRAL_MS = 500



5.5- Gráfica de tiempos para la ejecución del programa con UMBRAL_MS = 1000



5.6 - COMPARATIVA DE TODAS LAS GRAFICAS



Como podemos apreciar, a mayor valor para la variable UMBRAL_MS, más eficiente es el algoritmo.