

TP0 Organización de las Computadoras (66.20)

Grupo Nro. X - 1er. Cuatrimestre de 2012
66.20 Organización de Computadoras
Facultad de Ingeniería, Universidad de Buenos Aires



Rodriguez Genaro, Leandro

Padrón Nro. 92.098

leandrorodriguezg@yahoo.com.ar

Reale, Tomás

Padrón Nro. 92.255

tomasreale@gmail.com

Piechotka, Federico

Padrón Nro. 92.216

f_piecho@hotmail.com

Resumen

La idea principal de este trabajo practico es aprender a utilizar ciertas herramientas fundamentales para el análisis de software. Con tal fin, se implementa un programa simple cuya función es ordenar cadenas de caracteres. Este programa cuenta con 2 algoritmos de ordenamiento (Merge Sort o Seleccion Sort) y cuenta con la posibilidad de escoger 1 de estos enviándole un parámetro (-m o -s) a la aplicación. Las herramientas de análisis de software a utilizar son: Gxemul (para simular una maquina MIPS usando un SO NetBSD, y chequear portabilidad). gprof (herramienta de profiling para ver tiempos de nuestro programa). También se utilizara “time” de la consola de Linux.

1. Introducción

Con ayuda de los links provistos por la cátedra (wikipedia) se programo en el lenguaje C la aplicación antes mencionada, responsable de ordenar cadenas de caracteres, ya sea leyendo los datos de 1 o varios archivos o por consola(lo que seria entrada standar). Todo esto bajo el SO Ubuntu 10.04. Además de los algoritmos de ordenamiento, el programa cuenta con funciones para leer y manejar archivos, y un parser para interpretar los argumentos que se le pasan al ejecutarlo por consola. El parser de argumentos se implemento usando la útil librería getopt, y para manipular archivos, se siguió lo que dicta la ortodoxia de los manuales que se encuentran dando vuelta por la red. Se configuro el gxemul para correr NetBSD sobre una maquina mips, y se lo comunico con nuestro usuario de ubuntu a través de ssh. Con el programa ya programado en C, se envió el código fuente a la maquina virtual del gxemul, se compilo por consola (cuidándose de generar el código .s), y se ejecuto allí para poder verificar que efectivamente el programa fuese portable.

2. *Previo a realizacion de medidas*

2.1. Merge Sort

El algoritmo de ordenamiento por mezcla (merge sort en inglés) es un algoritmo de ordenamiento externo estable basado en la técnica divide y vencerás. Es de complejidad $O(n * \log(n))$, ya sea para el mejor caso, el peor caso, o el caso promedio.

2.2. Selection Sort

El ordenamiento por selección (Selection Sort en inglés) es un algoritmo de ordenamiento que requiere $O(n^2)$ operaciones para ordenar una lista de n elementos. Su complejidad no varia para el mejor, el peor, o el caso promedio.

3. Desarrollo

3.1. Mediciones de tiempo

Siguiendo las instrucciones del tp, se crearon varios archivos de caracteres aleatorios, con un tamaño dado por $Size = 100 * 2^N$, $N = 0, 1, 2, 3, \dots, 16$ Antes

de empezar a medir, se hicieron 3 copias para cada tamaño. Una se dejo tal cual fue creada. Otra se ordeno de menor a mayor. Otra se ordeno de mayor a menor. Con esas copias, se procedio a construir tablas del tiempo medido en funcion del tamaño, para cada uno de los algoritmos de ordenamiento. A continuacion mostramos los graficos obtenidos. Esto se ejecutando el programa ya compilado, usando la funcion time de la consola de Ubuntu.

3.2. Analisis de las mediciones de tiempo

Es facil ver que se verifica lo que se esperaba : que los 2 metodos de ordenamiento mantienen su orden de complejidad independientemente de como se encuentra el vector.

3.3. Resultados

Una subseccion de la parte principal. Ponele que aca van los resultados de las pruebas

3.3.1. Alguna otra cosa

Una sub subseccion. Mas info.

Referencias

- [1] Leslie Lamport, *LaTeX: A Document Preparation System*. Addison Wesley, Massachusetts, 2nd Edition, 1994.

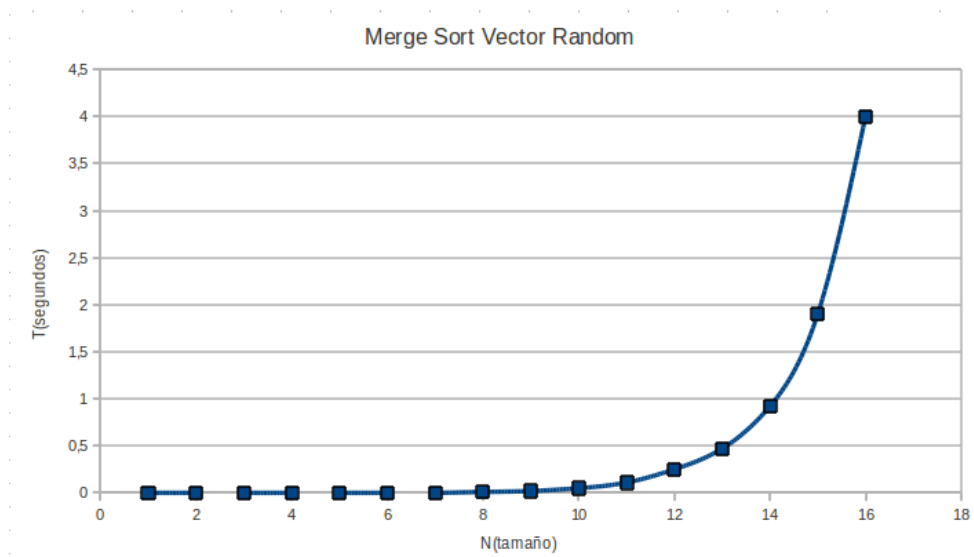


Figura 1: Selection sort: Tiempo insumido Vs. $100 * 2^N$ en vector random