66:20/86.37 Organización de Computadoras
Primer cuatrimestre de 2017
Trabajo práctico 0: Infraestructura básica

1. Objetivos

Familiarizarse con las herramientas de software que usaremos en los siguientes trabajos, implementando un programa (y su correspondiente documentación) que resuelva el problema piloto que presentaremos más abajo.

2. Alcance

Este trabajo práctico es de elaboración grupal, evaluación individual, y de carácter obligatorio para todos alumnos del curso.

3. Requisitos

El trabajo deberá ser entregado personalmente, en la fecha estipulada, con una carátula que contenga los datos completos de todos los integrantes.

Además, es necesario que el trabajo práctico incluya (entre otras cosas, ver sección 8), la presentación de los resultados obtenidos, explicando, cuando corresponda, con fundamentos reales, las causas o razones de cada resultado obtenido.

El informe deberá respetar el modelo de referencia que se encuentra en el grupo, y se valorarán aquellos escritos usando la herramienta TEX / LATEX.

4. Recursos

Usaremos el programa GXemul [1] para simular el entorno de desarrollo que utilizaremos en este y otros trabajos prácticos, una máquina MIPS corriendo una versión reciente del sistema operativo NetBSD [2]. En éste utilizaremos los programas `gcc` [5] y `gcov` [6] para compilar y para examinar el desempeño de los programas a desarrollar.

5. Algoritmos de ordenamiento

La capacidad de ordenar conjuntos de datos según algún criterio arbitrario es vital para la informática. Existen muchos algoritmos para ordenar conjuntos de datos, y no todos son igual de eficientes. En este caso, veremos el desempeño de los algoritmos Quicksort [4] y Bubblesort [3].

6. Programas a desarrollar

El programa a escribir, en lenguaje C, recibirá un nombre de archivo que contiene texto (o el archivo mismo por `stdin`) y devolverá otro con las palabras que contiene el archivo original pero ordenadas; según las opciones de invocación se ordenarán mediante Quicksort o mediante Bubblesort. Como palabra se entiende cualquier secuencia de caracteres alfanuméricos entre dos espacios en blanco (como espacios en blanco consideramos los espacios propiamente dichos, los tabs y los saltos de línea o página). En ambos casos se deberá utilizar tanto una función que compara dos elementos del array como una que intercambia dos elementos del array: nuestra intención es analizar cuántas veces se llama a cada función en uno y otro de estos algoritmos para un conjunto determinado de archivos a ordenar, así como cuánto tarda el programa en ejecutarse para estos archivos según el algoritmo elegido.

6.1. Ejemplos

Primero, usamos la opción `-h` para ver el mensaje de ayuda:

```
$ tp0 -h
Usage:
  tp0 -h
  tp0 -V
  tp0 [options] file
Options:
  -V, --version     Print version and quit.
  -h, --help        Print this information.
  -o, --output      Path to output file.
  -i, --input       Path to input file.
  -q, --qsort       Use quicksort.
  -b, --bsort       Use bubblesort.
Examples:

  tp0 -q -i input.txt -o output.txt
```

Luego, lo usamos para ordenar un pequeño fragmento de texto:

```
$echo -n echo "El tractorcito rojo que silbo y bufo" > entrada.txt
$tp0 -b entrada.txt
bufo El que rojo silbo tractorcito y
$
```

7. Mediciones

Se deberá medir el tiempo insumido por el programa y la cantidad de llamadas a las funciones de comparación y de intercambio para uno y otro algoritmo, en el caso de los archivos `alice.txt`, `beowulf.txt`, `cyclopedia.txt` y `elquijote.txt`. La cantidad de llamadas a función puede obtenerse mediante el programa `gcov`, visto en clase. Graficar para cada algoritmo el tiempo insumido contra el tamaño de muestra. Graficar el speedup de *Quicksort* contra *Bubblesort* para los diversos tamaños de archivo. Se deberá también comprobar que el programa acepta las opciones dadas, y que reporta un error ante situaciones anómalas (como no encontrar el archivo a ordenar) y ante opciones incompatibles (como `-q` y `-b` al mismo tiempo). La ejecución del programa debe realizarse bajo el entorno MIPS.

8. Informe

El informe deberá incluir:

- Documentación relevante al diseño e implementación del programa;
- Las corridas de prueba, con los comentarios pertinentes;
- Gráficos de cantidad de llamadas a la función de comparación, a la función de intercambio, y de tiempo completo para los dos algoritmos, en función del tamaño de la entrada.
- El código fuente, en lenguaje C;
- Este enunciado.

9. Fecha de entrega

El cierre de este trabajo práctico es el jueves 30 de Marzo de 2017.

Referencias

[1] GXemul, `http://gavare.se/gxemul/`

[2] The NetBSD project, `http://www.netbsd.org/`

[3] Bubble sort, `https://en.wikipedia.org/wiki/Bubble_sort`

[4] Quicksort, `https://en.wikipedia.org/wiki/Quicksort`

[5] GCC, `https://gcc.gnu.org/onlinedocs/gcc-3.3.6/gcc/`

[6] Gcov,`https://gcc.gnu.org/onlinedocs/gcc/Gcov.html#Gcov`