Trabajo Practico N°1

Estructura de Datos y Algoritmos I

Blas Barbagelata - Tomas Castro Rojas

Introducción

El objetivo de este trabajo es la implementación de distintos algoritmos de ordenación, en particular, Selection Sort, Insertion Sort y Merge Sort, y comparar su rendimiento con juegos de datos de distinto tamaño. Los algoritmos serán aplicados en Listas Enlazadas Generales.

Implementación de Listas

Para este trabajo elegimos una implementación de Listas doblemente enlazadas con una estructura con dos campos, uno que apunta al principio de la lista y el otro al final de la misma.

```
typedef struct _GNodo {
        void *dato;
        struct _GNodo *sig;
        struct _GNodo *ant;
}GNodo;
typedef struct {
        GNodo *primero;
        GNodo *ultimo;
}Glist;
```

Nuestra elección se basó en que al ser doblemente enlazada simplifica ciertos procesos en los algoritmos de ordenación, Insertion Sort y al dividir la lista en Merge Sort. Luego, una estructura con punteros al inicio y final ya que nos resultó más intuitivo y entendible a la hora de trabajar.

Funcionamiento del programa

El programa trabaja del siguiente modo:

Se compila con las reglas que se encuentran en el makefile.

Para ejecutar el programar se deben pasar 4 argumentos por consola: el nombre del archivo con los nombres, el nombre del archivo con las localidades, la cantidad de personas a generar y el nombre del archivo de personas que se genera.

Lo primero que hace el programa es validar la cantidad de argumentos pasados, una vez verificado, configura la codificación del programa a la codificación del sistema operativo donde se esta ejecutando.

Luego, crea el archivo de personas con las funciones que se encuentran en generar.c. Después, leyendo el archivo persona, crea una lista de persona con las funciones que encuentran en persona.c y glist.c.

Finalmente, invoca cada algoritmo de ordenación dos veces donde ordenan la lista comparando un dato distinto de la persona (nombre, edad o localidad) y escribe la lista ordenada en un nuevo archivo.

Comparación algoritmos de ordenación

Ordenamiento por edad en segundos

Cantidad de personas	20000	50000	80000	100000
Insertion Sort	0,968	6,663	19,163	33,180
Selection Sort	1,061	6,960	20,492	35,119
Merge Sort	0,011	0,038	0,044	0,056

Ordenamiento por nombre en segundos

Cantidad de personas	20000	50000	80000	100000
Insertion Sort	1,573	10,898	36,723	69,049
Selection Sort	2,126	14,192	42,792	75,034
Merge Sort	0,015	0,062	0,072	0,095

Como se puede observar, a medida que la cantidad de personas aumentan, Insertion Sort y Selection Sort aumentan considerablemente su tiempo de ejecucion. Mientras que Merge Sort, crece muy minimamente.