Universidad de Chile
Facultad de Ciencias Físicas y Matemáticas
Departamento de Ciencias de la Computación

CC 4102 - Tarea 1: Mergesort

Profesor: Pablo Barceló **Auxiliar:** Felipe Contreras Salinas

3 de abril de 2016

1. Introducción

El objetivo de esta tarea es implementar y comparar algunas variantes de *mergesort* para ordenar datos en memoria secundaria. Para compararlas, se considerarán instancias donde alguna fracción de los datos ya está ordenada.

2. Algoritmos

Cada algoritmo debe leer un archivo de enteros separados por salto de línea y debe escribir otro archivo con los datos ordenados. Como se vio en clases, un *run* es una secuencia de datos consecutivos que están ordenados.

A continuación, se describen las variantes de *mergesort* a considerar.

- **Two-way mergesort:** Separa los datos del archivo de entrada en dos archivos f_1 y f_2 de manera balanceada. En cada iteración, mezcla los runs de tamaño k de f_1 y f_2 en runs de tamaño $2k$ y los escribe en los archivos g_1 y g_2, alternadamente. En un principio, $k = 1$ y el algoritmo termina cuando tiene un solo run de tamaño n.
- **Two-phase multiway mergesort:** Se divide el archivo en trozos de tamaño k, los que se ordenan en memoria principal y luego se escriben en $m = n/k$ archivos. Ahora, se mezclan los m archivos tomando el menor de los primeros elementos de cada archivo en cada paso.
- **Mergesort adaptativo:** Se divide el archivo original según sus runs. Es decir, si la entrada tiene m runs, entonces se escriben m archivos con un run cada uno. Ahora, se mezclan los dos archivos con menos elementos en cada paso. El algoritmo termina cuando queda un solo archivo.

3. Experimentos

Se quiere comparar el desempeño en tiempo de los algoritmos. Para ello, se deben generar instancias aleatorias de tamaño $n = 2^{30}$ que contengan un run de al menos $20\,\%, 50\,\%, 80\,\%$ de la cantidad total.

Usted debe explicar en su informe la metodología usada para sus experimentos, incluyendo el cómo genero las instancias.

4. Entrega

- Se puede trabajar en grupos de a lo más dos personas.
- La implementación debe ser en Java o C++ y debe incluir las instrucciones de como compilarse y ejecutarse.
- El informe debe entregarse en formato PDF. Se recomienda usar LaTeX.

5. Referencias

- External-Memory Sorting `http://people.cs.aau.dk/~simas/aalg04/esort.pdf`