CC4102 - Diseño y Análisis de Algoritmos
Tarea 1: Ordenamiento en Memoria Secundaria

Prof. Gonzalo Navarro; Aux. Mauricio Quezada

Fecha de Entrega: 14 de Diciembre de 2011

Introducción

En esta tarea deberá implementar dos algoritmos de ordenamiento en memoria secundaria: Mergesort multiario y Samplesort. Para ambos deberá determinar experimental y analíticamente los mejores parámetros para su funcionamiento y comparar sus resultados. En particular, deberá:

1. Plantear una hipótesis con respecto a los resultados esperados en esta actividad, *antes* de realizar la experimentación;
2. Determinar los parámetros óptimos de ambos algoritmos, analítica y experimentalmente en una instancia en particular, y luego comparar y presentar los resultados obtenidos; y
3. Probar sus implementaciones en distintas instancias para observar la tasa de crecimiento en el tiempo; presentar los resultados, análisis comprobando (*o no*) su hipótesis, y conclusiones.

Deberá entregar un informe completo que incluya hipótesis, diseño experimental, presentación de los resultados, análisis, conclusiones y anexos.

Algoritmos

1. **Mergesort multiario:** la aridad debe ser determinada analítica y experimentalmente de acuerdo a lo visto en clase de cátedra. Para ello, obtenga los datos del disco duro y memoria principal que utilizará y con ellos (M, B, t_{seek}, $t_{latency}$, y $t_{transfer}$) estimar la aridad óptima, analítica y experimentalmente para un cierto conjunto de datos.
2. **Samplesort:** similar a Quicksort, donde en vez de particionar una vez en cada pasada, lo hace k veces, donde k lo debe determinar analíticamente de acuerdo a la estrategia vista en clase auxiliar[1], y luego recursivamente, hasta que cada segmento quepa en memoria principal, y ordenar usando algún algoritmo tradicional; comprobar experimentalmente su rendimiento.

[1] O bien en "Algorithms and Data Structures" de Mehlhorn y Sanders donde aparece la explicación detallada.

Instancias y Experimentos

Las instancias corresponden a archivos con números enteros de 32 bits escogidos al azar uniformemente entre 0 y $2^{31} - 1$.

Determinación de parámetros

Para determinar los parámetros adecuados de sus algoritmos, considere $M = 26214400$, es decir, `100MB` de números enteros; $N = 256M$, es decir, 6710886400 elementos (`25GB`).

Repita sus experimentos bajo esta instancia de forma de obtener los parámetros óptimos para su implementación, de acuerdo al intervalo de confianza deseado. Si la varianza de su experimentación es muy alta y exige muchas repeticiones, explíquelo y justifíquelo en el informe, y podrá disminuir N hasta la mitad, o bien permitir un mayor error relativo.

Experimentación con otras instancias

Una vez obtenidos los parámetros adecuados, repita su experimentación para distintos tamaños de N, con $N \in \{2M, 2^2M, \ldots, 2^{10}M\}$, con todos los demás valores constantes, esto para comprobar empíricamente la tasa de crecimiento en el tiempo de sus algoritmos y la hipótesis que debe incluir en su informe.

Note que para la última instancia, $1024M$, el tamaño del archivo (y el tiempo en generarlos) puede ser considerable.

Medidas de Rendimiento

1. Accesos a memoria secundaria (I/O).
2. Accesos aleatorios a memoria secundaria.
3. Tiempo transcurrido (`user`, `system`, `real`).

Asegúrese de documentar todos los datos relevantes de su implementación, como el sistema operativo, RAM, características del disco duro, lenguaje utilizado, etc.

Entrega de Informe y otras consideraciones

Se recomienda utilizar el lenguaje `C` para su implementación; o bien utilizar `C++`, `Java` o `Python` explicando los problemas que pueda tener a lo largo del desarrollo, especialmente en la generación de instancias.

Entregue el informe impreso en Secretaría Docente hasta el mediodía del día siguiente a la fecha de entrega. Por medio de U-Cursos suba tanto el informe como el código de su implementación, junto con un archivo README indicando instrucciones de su ejecución, de forma de poder reproducir eventualmente los mismos experimentos.