Merge Sort Secuencial VS Merge Sort Paralelo

{Angelica Aguirre Castro, Kenny Bedoya Cifuentes}
Universidad Tecnologica de Pereira

ABSTRACT

In this report the results of analysis of two sorting algorithms that use the model merge sort on an array of dynamic size, which is essential for comparing execution times by varying the amount of items containing the same show, thus obtaining a performance analysis which follows. One of the algorithms was implemented to be executed sequentially, while the other was implemented for parallel execution, which was executed on a GPU Nvidia Gforce GTx 980 (card made available for the course of High Performance Computing, HPC , by the hotbed of Sirius Research of the Faculty of Engineering of the Technological University of Pereira).

ALGORITMO

Para nuestro análisis fueron usados dos algoritmos que implementan el ordenamiento por mezcla (mergesort)l primero un algoritmo secuencial implementado en el lenguaje de programación C, en donde se evidencia el uso de ciclos que realizan los respectivos recorridos sobre los arreglos.

El segundo es un algoritmo paralelo implementado en el lenguaje de programación Cuda C, en donde se evidencia que la cantidad de hilos y bloques (necesarios para que la ejecución sea en paralelo) se calcula según la cantidad de elementos que contenga el arreglo a ordenar.

```
__global__ void gpu_mergesort(long* source, long*
dest, long size, long width, long slices, dim3* threads,
dim3* blocks) {
   unsigned int idx = getIdx(threads, blocks);
   long start = width*idx*slices,
        middle,
        end;
   for (long slice = 0; slice < slices; slice++) {
      if (start >= size)
         break;
      middle = min(start + (width >> 1), size);
      end = min(start + width, size);
      gpu_bottomUpMerge(source, dest, start, middle,
end);
      start += width;   } }
```

INTRODUCTION

En este informe se presenta el resultado de análisis de dos algoritmos de ordenamiento que usan el modelo de Merge sort sobre un arreglo de tamaño dinámico, el cual es esencial para la comparación de tiempos de ejecución al variar la cantidad de elementos que contiene el mismo, obteniendo así un análisis de desempeño el cual se expone a continuación. Uno de los algoritmos fue implementado para ser ejecutado de manera secuencial, mientras que el otro fue implementado para una ejecución en paralelo, el cual fue ejecutado sobre una GPU Nvidia Gforce GTx 980 (tarjeta puesta a disposición para el curso de High Performance Computing, HPC, por parte del semillero de Investigación Sirius de la Facultad de Ingenierías de la Universidad Tecnológica de Pereira).

TRABAJOS FUTUROS

Debido al crecimiento de la información que almacenan las diversas bases de datos con las que los usuarios interactúan en su diario vivir, vemos que el uso de algoritmos de ordenamiento y búsqueda implementados para una ejecución en paralelo, pueden ser usados para el análisis de dicha información, con la cual se puede obtener una nueva información para uso estratégico (Big data).

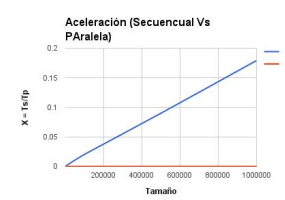
REFERENCIAS

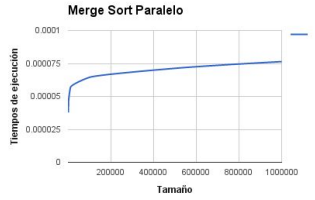
[1] http://www.nvidia.com/object/cuda_home_new.html
[2] Thomas H. Cormen, Charles E. Leirserson, Ronald L. Rivest, Clifford Stein, Introduction to Algorithms

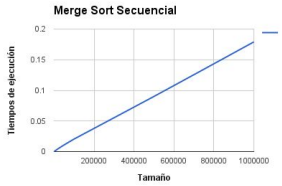
CONTACTO:

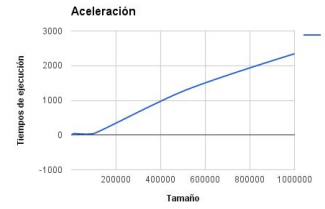
anaguirre@utp.edu.co
kebedoya@utp.edu.co

RESULTADOS









Tamaño	Ts	Tp	X = Ts / Tp
10	0.000004	0.00003	0.1333333333
100	0.0000256	0.0000378	0.6772486772
1000	0.0002184	0.000046	4.747826087
10000	0.0025498	0.0000568	44.89084507
100000	0.0205928	0.0000645	44.89084507
500000	0.090101	0.0000713	1263.68864
1000000	0.1790582	0.0000764	2343.693717

CONCLUSIONES

Se evidencia que el debido uso de una GPU para la ejecución de algoritmos que trabajan con mucha información, en nuestro caso un algoritmo de ordenamiento, aumenta la velocidad de ejecución con respecto a la ejecución del mismo en una CPU.
Cabe mencionar que es necesario conocer las especificaciones de la GPU a utilizar debido a que no se deben sobrepasar los límites de memoria e hilos de la misma, para así identificar los límites tanto de información a cargar como de las técnicas de las que se deben hacer uso en caso tal de necesitar una mayor capacidad en memoria.