Memoria de Seguimiento del Trabajo Fin de Grado

Paralelización de algoritmos irregulares en arquitecturas GPU

Luis Arija González

1. Resumen del Trabajo Realizado:

Este proyecto empezó su desarrollo hace 6 semanas, el 28 de septiembre. A lo largo del tiempo he estado manteniéndome fiel al plan de trabajo y a las opiniones de mis tutores a la hora de estructurar los objetivos por realizar.

En la primera semana de trabajo nos familiarizamos con el concepto del proyecto a realizar: La resolución de algoritmos irregulares mediante paralelización en una arquitectura GPU.

Para trabajar con GPU's haría falta un lenguaje de programación especializado en ellas, por ello escogí el lenguaje de programación de GPU's basado en C y especializado en las GPU's de Nvidia: CUDA C.

A lo largo de las siguientes cuatro semanas estuve aprendiendo a programar en CUDA C.

En la segunda semana estuve familiarizándome con el concepto de programar para la GPU. Debido a ello, los códigos realizados eran simples, tal como un "HelloWorld" o la realización de sumas vectoriales.
Desde un comienzo la información se trabaja desde la CPU con el lenguaje de programación clásico de C. Tras ello, la información pertinente se envía a la GPU mediante funciones de CUDA, se procesa mediante un Kernel y la información procesada se envía de vuelta a la CPU.

En la tercera semana pasamos a tener como objetivo la realización de sumas y multiplicaciones matriciales mediante la paralelización.
Esto implica la creación de programas que invoquen a un Kernel con tantos Threads como valores tiene la matriz, y usen la información de qué Thread es para la correcta obtención del valor deseado.
Los programas son realizados, pero hay problemas al comprobar su eficacia debido a la dificultad de pasar matrices de datos a la GPU.

En la cuarta semana decidimos reescribir el código y las funciones Kernel de tal manera que trabajen con matrices formalizadas como vectores de datos. Surge la necesidad de guardar el número de columnas y filas para el correcto funcionamiento de las funciones Kernel.
Dicho eso, los objetivos son cumplidos y los resultados demuestran ser correctos en todo momento.

En la quinta semana decidimos que el aprendizaje de CUDA está avanzando a buen paso, así que nos centramos en objetivos menos visibles: Comentar correctamente el funcionamiento de los códigos, comprobar que la GPU realmente se está usando, y averiguar los límites numéricos respecto a Bloques y Threads de la GPU con la que trabajo. Todo ello es realizado.

En la sexta semana nos familiarizamos con algoritmo que iremos a paralelizar: El problema de los N cuerpos. Nuestro progreso se ve ligeramente ralentizado por el desarrollo de la memoria intermedia.

2. *Explicación y Justificación de las modificaciones al Plan de Trabajo*

1. Revisión de la lista de objetivos del trabajo:

Debido a que no están habiendo complicaciones mayores en el desarrollo del proyecto, la lista de objetivos original se mantiene, siendo estos:

- Aprender a desarrollar algoritmos paralelos sobre GPUs.

- Realizar una o varias implementaciones de un algoritmo no regular no trivial en GPUs y analizar la eficiencia de las mismas.

- Extraer conclusiones que ayuden a buscar patrones algorítmicos útiles para resolver problemas similares.

2. Revisión de la lista de tareas:

Debido a que no están habiendo complicaciones mayores en el desarrollo del proyecto, la lista de tareas original se mantiene, siendo estas:

- Familiarización con el proyecto y planificación: 27 horas.

- Elección y aprendizaje del marco de programación de GPUs a utilizar -- OpenCL, CUDA, etc.: 40 horas.

- Elección y estudio del algoritmo a paralelizar -- en principio sería un algoritmo para resolver el problema de los "N cuerpos" usando octrees: 20 horas.

- Desarrollo de las diferentes implementaciones a comparar, incluyendo la posible adaptación de alguno de estos algoritmos para resolver problemas similares: 100 horas.

- Desarrollo de un front-end de visualización de estos algoritmos: 20 horas. - Análisis de la eficiencia de las diferentes implementaciones: 20 horas.

- Memoria y presentación: 70 horas.

3. Revisión del diagrama de Gantt:

Debido a que no están habiendo complicaciones mayores en el desarrollo del proyecto, el diagrama de Gantt se mantiene, siendo este visible en la siguiente página.

Tarea\Semana	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Familiarización con el proyecto y planificación																		
Elección y aprendizaje del marco de programación de GPUs a utilizar																		
Elección y estudio del algoritmo a paralelizar																		
Desarrollo de las diferentes implementaciones a comparar, incluyendo la posible adaptación de alguno de estos algoritmos para resolver problemas similares																		
Desarrollo de un front-end de visualización de estos algoritmos																		
Análisis de la eficiencia de las diferentes implementaciones																		
Memoria y presentación																		