PROGRAMA DE CURSO

Código	Nombre			
CC7515	Computación en GPU			
Nombre en Inglés				
Gpu Computing				
SCT	**Unidades Docentes**	**Horas de Cátedra**	**Horas Docencia Auxiliar**	**Horas de Trabajo Personal**
6	10	3	0	7
Requisitos			**Carácter del Curso**	
CC3301,CC3501/AUTOR			Electivo	

Resultados de Aprendizaje

Las tarjetas de procesamiento gráfico, conocidas como GPU, fueron desarrolladas originalmente para acelerar el proceso de rendering gráfico, motivadas principalmente por los videojuegos. Sin embargo, hoy en día, GPUs son intensamente usadas por aplicaciones de propósito general que requieren alto poder de cálculo. Las GPUS se han transformando en una alternativa eficiente, y más económica que los clusters de CPUs, para los algoritmos en que el procesamiento de grandes bloques de información se puede hacer en paralelo.

El propósito de este curso es que las y los alumnos aprendan a: detectar qué problemas son paralelizables en la GPU, cómo diseñar y programar una solución paralela, conocer y aplicar técnicas de optimización, y evaluar el desempeño de sus soluciones paralelas. El ámbito de problemas a resolver son tanto problemas que surgen de la computación gráfica, computación en general y de aplicaciones científicas e ingenieriles. En resumen, al final de este curso, las y los alumnos serán capaces de:

- Entender las arquitecturas de las GPUs
- Detectar si un problema puede ser paralelizable en la GPU
- Diseñar soluciones paralelas
- Programar soluciones paralelas (para aplicaciones gráficas y de propósito general)
- Implementar soluciones para resolver problemas que clásicamente han sido resueltos en supercomputadores
- Evaluar el desempeño de soluciones paralelas
- Enfrentar y resolver problemas científicamente interesantes
- Conocer aplicaciones paralelizables en GPU en distintos ámbitos
- Conocer los desafíos existentes en computación en GPU.

Metodología Docente	Evaluación General
El curso consiste en clases de cátedra tradicionales y en clases usando la metodología de aprendizaje basado en problemas. Las y los alumnos deberán desarrollar 3 tareas, una en cada uno de los siguientes modelos de programación: Cuda, OpenCl y shaders (GLSL). Cada alumno debe desarrollar un proyecto computacional identificando un problema desafiante, que tenga una solución eficiente al programarlo en la GPU. El problema a resolver en el proyecto puede ser propuesto por el estudiante. El curso también requiere leer artículos científicos/capítulos de libro en inglés.	El curso posee tres controles de lectura (cuyo promedio es el control C1), dos evaluaciones grupales (2 o tres personas) y una presentación oral de algún tema de interés relacionado al curso, cuyo promedio es el control C2, 3 tareas de programación (NT) y proyecto computacional (NPC). La nota final (NF) se calcula como sigue: NC = (C1+C2)/2. El examen (NE) consistirá en una presentación oral del proyecto abordado en donde al alumno presentará el problema, la solución y una discusión crítica de lo realizado. NP= 60%NC + 40%NE NF = 40% NP + 20% NPC + 40%NT NP, NPC y MT deben >= 4.0 independientemente.

Unidades Temáticas

Número	Nombre de la Unidad	Duración en Semanas
1	Introducción	2

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
• Motivación • Evolución e historia • Conceptos Básicos • Medidas de Desempeño • Aplicaciones: videojuegos, ciencia e ingeniería. • Repaso de aspectos avanzados de programación en c/c++	• Conocer la motivación y contexto de desarrollo de la computación en gpu. • Aprender medidas de desempeño. • Conocer las aplicaciones en donde usa computación en gpu. • Programar en c++	[1,3,5]

Número	Nombre de la Unidad	Duración en Semanas
2	Modelos de computación y programación paralela	1

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliograf
• Modelos de computación paralela: PRAM, PMH, BSP, etc. • Modelos de programación paralela: memoria compartida traspaso de mensajes, implícito. • Ejemplos	• Aprender los modelos de computación y programación paralela. • Conocer el alcance de estos modelos.	[3,4]

Número	Nombre de la Unidad	Duración en Semanas
3	Arquitecturas	1

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
• Detalles técnicos de CPUs y GPUS Modernas. • Diferencias fundamentales entre GPUs yCPUs.	• Conocer las características de las GPU y sus diferencias con las CPU.	[3,6,7]

Número	Nombre de la Unidad	Duración en Semanas
4	Resolviendo problemas en la GPU	3

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
• Estrategia para diseñar algoritmos paralelos: (Particionamiento, Comunicación, Aglomeración y Mapping). • El modelo de programación de paralelismo masivo. • Manejo de Threads y concurrencia • Consideraciones técnicas para una implementación en GPU. • Ejemplos	• Aprender a resolver problemas usando la gpu. • Conocer los conceptos y las consideraciones técnicas para el diseño e implementación de soluciones paralelas.	[3,6,7]

Número	Nombre de la Unidad	Duración en Semanas
5	Modelos de programación de GPU actuales para cómputo general	4

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
• Modelo de programación en Cuda • Modelo de programación en OpenCl • Ventajas/desventajas de ambos modelos. • Ejemplos de programas clásicos en ambos modelos. • Caso de estudio: Generación de triangulaciones de Delaunay	• Aprender a programar usando Cuda y OpenCl. • Analizar y evaluar implementaciones existentes. • Identificar las ventajas desventajas de usar uno u otro modelo.	[1,5,6,7,10]

Número	Nombre de la Unidad	Duración en Semanas
6	Programación de la GPU para aplicaciones gráficas	3

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
• Pipeline gráfico: - Vertex shaders - Fragment shaders - Geometry shaders - Tessellation shaders • Algoritmos y aplicaciones - OpenGl+GLSL y Vulkan - Motores gráficos y shaders (unity y unreal) Caso de estudio: visualizador Camarón.	• Aprender a enfrentar y resolver problemas desafiantes en el área de la computación gráfica y video juegos. • Aprender a programar usando shaders. • Analizar y evaluar algoritmos programados en gpu.	[1,2,5,9]

Número	Nombre de la Unidad	Duración en Semanas
7	Ultimos avances y problemas abiertos en GPU	1

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
• Problemas de mapeo • Evolución de tarjetas gráficas • Desafíos científicos, en general	• Aprender los desafíos existentes relacionados a la etapa de mapeo. • Conocer problemas científicamente interesantes que están siendo abordados. • Problemas no resueltos.	[1,3,8]

Bibliografía

[1] Hubert Nguyen. NVIDIA corporation. GPUGems 3. Addison Wesley. 2008

[2] Hearn, Baker, Carithers. Computer graphics with OpenGl. Fourth edition. 2011.

[3] Cristobal Navarrro, Nancy Hitschfeld-Kahler, Luis Mateu, A Survey on Parallel Computing and its Applications in Data-Parallel Problems Using GPU Architectures, Communications in Computational Physics, 15:285-329, 2014.

[4] Jaja, Joseph, An introduction to Parallel Algorithms, 1992. Pearson.

[5] Bjarne Stroustrup. The c++ programming language (c++11). Fourth Edition. Addison ritherWesley. 2013.

[6] Tutorial Cuda. https://developer.nvidia.com/cuda-education-training

[7] Tutorial OpenCl: http://developer.amd.com/tools-and-sdks/opencl-zone/opencl-resources/introductory-tutorial-to-opencl/

[8] Cristobal A. Navarro, Nancy Hitschfeld, GPU maps for the space of computation in triangular domain problems Proceedings of the 11th IEEE International Conference on High Performance Computing and Communications (HPCC 2014), Paris, France. August 20-22 2014, pp:375-382.

Casos de estudio:

[9] Aldo Canepa, Gonzalo Infante, Nancy Hitschfeld-Kahler, Claudio Lobos, Camaron: An open-source visualization tool for the quality in-spection of polygonal and polyhedral meshes. Proceedings of the 11th International Conference on Computer Graphics Theory and Applications (GRAPP 2016). Roma, Italia, February 2016. pp:128-135.

[10] Cristobal A. Navarro, Nancy Hitschfeld-Kahler, Eliana Scheihing: Quasi-Delaunay Triangulations Using GPU-Based Edge-Flips, Computer Vision, Imaging and Computer Graphics: Theory and Applications, pp 36-49, vol. 458 of Comm. in Computer and Information Science, Springer, 2014.

Vigencia desde:	Primavera 2017
Elaborado por:	Nancy Hitschfeld Kahler