Formulario de aprobación de curso de posgrado/educación permanente

Asignatura: Programación masivamente paralela en procesadores gráficos (GPUs)

Modalidad:	**Posgrado**	X
	Educación permanente	

Profesor de la asignatura: Dr. Ing. Martín Pedemonte, Prof. Adjunto, Instituto de Computación

Profesor Responsable Local: -

Otros docentes de la Facultad: Dr. Ing. Ernesto Dufrechou, Prof. Adjunto, Instituto de Computación
Dr. Ing. Pablo Ezzatti, Prof. Adjunto, Instituto de Computación

Docentes fuera de Facultad: No corresponde

Programa(s) de posgrado: Maestría y Doctorado en Informática y Maestría en Ciencia de Datos y Aprendizaje Automático.

Instituto o unidad: Instituto de Computación

Departamento o área: Laboratorio de Computación Heterogénea (HCL)

Horas Presenciales: 38

Nº de Créditos: 10
[
Público objetivo: Estudiantes de posgrado y profesionales interesados en el uso de procesadores gráficos para la resolución de problemas de propósito general, computación de alta performance y computación heterogénea.

Cupos: No corresponde

Objetivos:

El objetivo es introducir al estudiante en el uso de los procesadores gráficos y otros dispositivos de hardware secundario para la resolución de problemas de propósito general.

Al finalizar el curso se espera que el estudiante:
- Pueda resolver problemas generales de complejidad media en unidades de procesamiento gráfico (GPU, por sus siglas en inglés).
- Sea capaz de implementar rutinas en el lenguaje CUDA C que hagan un buen aprovechamiento de los recursos de cómputo de las GPUs.
- Logre estudiar el tiempo de ejecución de rutinas que ejecutan en GPU, analizando sus cuellos de botella, y sea capaz de plantear optimizaciones con el fin de solucionarlos.
- Tenga un conocimiento general del ecosistema CUDA incluyendo librerías, profilers, debuggers, etc.
- Tenga un conocimiento general de otros dispositivos de hardware secundario que se utilizan para realizar cómputo de propósito general.
- Tenga un conocimiento general de conceptos de Computación Heterogénea.

Conocimientos previos exigidos: Conocimientos de programación en C y arquitectura de sistemas.
Conocimientos previos recomendados: Es recomendable el manejo de conceptos de sistemas operativos, computación gráfica y computación de alto desempeño (HPC).

Metodología de enseñanza:

Descripción de la metodología:

Dictado de clases teóricas. Resolución de ejercicios prácticos por parte de los estudiantes, incluyendo la elaboración de informes. Realización de un proyecto final de laboratorio.

Detalle de horas:

- Horas de clase (teórico): 24
- Horas de clase (práctico): 5
- Horas de clase (laboratorio): 4
- Horas de consulta: 5
- Horas de evaluación: 0
    - Subtotal de horas presenciales: 38
- Horas de estudio: 24
- Horas de resolución de ejercicios/prácticos: 55
- Horas proyecto final/monografía: 35
    - Total de horas de dedicación del estudiante: 152

Forma de evaluación:
El curso cuenta con las siguientes instancias de evaluación:

• Realización de ejercicios de práctico durante el curso en forma individual.
• Trabajo de laboratorio final en forma individual.

Para aprobar la unidad curricular será necesario aprobar cada una de las instancias de evaluación. En caso contrario el curso se pierde. La nota del curso valorará ambas instancias de evaluación con la siguiente ponderación: Entregas de práctico (50%) y Laboratorio final (50%).

Temario:

1. Introducción: Motivación. Repaso histórico del surgimiento de las GPUs, su uso para cómputo general y eficiencia.
2. Introducción al paralelismo:
    a. Computación paralela: Paralelismo en máquinas secuenciales. Paralelismo en máquinas distribuidas. Paralelismo en máquinas distribuidas. Modelos y Estrategias para Programación Paralela. Paralelismo de Memoria Compartida. Paralelismo de Memoria Distribuida.
    b. Programación paralela: Estrategias de scheduling y balance de carga. Distribución de datos/cálculos. Reducción. Condiciones de carrera. Operaciones atómicas.
3. Introducción a CUDA: Arquitectura CUDA. Modelo de ejecución.
4. Programación en CUDA:
    a. Introducción: Device. Jerarquía de hilos (grids, bloques, warps). Uso de índices. Tipos de funciones. Keywords básicas y API.

b. Tipos de memoria: Acceso coalesced. Acceso a memoria (conflictos de bancos, tiling). Errores en tiempo de ejecución. Código PTX.

c. Transferencia de memoria: Transferencias sincrónicas y asincrónicas. Uso de memoria compartida.

5. Patrones de cómputo: Histograma. Reduce. Stencil. Scan.
6. Alternativas de otros fabricantes: Arquitecturas. Modelos de plataforma. Modelos de ejecución. Modelos de memoria. Modelos de programación.
7. Ecosistema CUDA: Herramientas (Debugging, Profiling), Librerías específicas (cusparse, curand, etc.) y Librerías de programación de alto nivel (thrust, cusp, modern gpu).
8. Nuevas Tendencias: Arquitecturas recientes. Paralelismo dinámico. Computación Heterogénea.
9. Aplicaciones de GPGPU: Álgebra lineal numérica. Computación gráfica. Deep Learning.

Bibliografía:
(título del libro-nombre del autor-editorial-ISBN-fecha de edición)

1. Programming Massively Parallel Processors: A Hands-on Approach Fourth Edition. Hwu, Wen-mei W., Kirk, David B., El Haj, Izzat. Morgan Kaufmann. 2023.
2. Computer Architecture A Quantitative Approach. 6th Edition. Hennessy, John L. and Patterson, David A. Elsevier. 2017.
3. An Introduction to Parallel Programming. 2nd Edition. Pacheco, Peter and Malensek, Matthew. Elsevier. 2021.
4. CUDA C++ Programming Guide 12.3, NVIDIA. 2024. Disponible en línea: https://docs.nvidia.com/cuda/cuda-c-programming-guide/
5. CUDA C++ Best Practices Guide 12.3. NVIDIA. 2024. Disponible en línea: https://docs.nvidia.com/cuda/cuda-c-best-practices-guide
6. CUDA by example: an introduction to general-purpose GPU programming. Sanders, J., & Kandrot, E. Addison-Wesley Professional. 2010.