TALLER DE PROGRAMACION SOBRE GPU	**Carrera/ Plan:** Ingeniería en Computación Plan 2024
Año 2026	**Año:** 5to **Régimen de Cursada:** Semestral **Carácter:** Optativa **Correlativas**: Concurrencia y paralelismo **Profesor/es:** Dr. Adrian Pousa **Hs. semanales teoría**: 3hs. **Hs. semanales práctica**: 3hs.

FUNDAMENTACIÓN

Los procesadores gráficos (GPUs) han surgido como una alternativa dentro de los procesadores con múltiples núcleos, por sus características de rendimiento y consumo energético.

El uso de GPUs, tanto en computación de alto desempeño como en aplicaciones de propósito general comienza a ser una alternativa de bajo costo para el desarrollo de aplicaciones de muy alto rendimiento que tradicionalmente han sido exclusivas de los clusters de multicores y supercomputadoras. En este contexto, la metodología e implementación de aplicaciones es un tema de gran interés actual.

Son objetivos de este curso: profundizar el conocimiento de las arquitecturas tipo GPU y su programación, comparar su rendimiento con arquitecturas convencionales, analizar los modelos de resolución de problemas específicos e introducir conceptos de consumo y green computing a partir de la utilización de GPUs.

OBJETIVOS GENERALES

Son objetivos de este curso:

- Profundizar el conocimiento de las arquitecturas tipo GPU y su programación.
- Comparar su rendimiento con arquitecturas convencionales.
- Analizar los modelos de resolución de problemas específicos.
- Introducir conceptos de consumo y green computing a partir de la utilización de GPUs.

COMPETENCIAS

- LI-CE1- Planificar, dirigir, realizar y/o evaluar proyectos de especificación, diseño, implementación, verificación, validación, puesta a punto, mantenimiento y actualización para arquitecturas de sistemas de procesamiento de datos, con capacidad de incorporar aspectos emergentes del cambio tecnológico.
- LI-CE6- Controlar las normas de calidad en el software o software integrado a otros componentes. Capacidad de evaluación de perfomance de sistemas de software y sistemas que integren hardware y software.
- LS-CE5- Establecer métricas y normas de calidad y seguridad de software, contralando las mismas a fin de tener un producto industrial que respete las normas nacionales e internacionales. Control de la especificación formal del producto, del proceso de diseño, desarrollo, implementación y mantenimiento. Establecimiento de métricas de validación y certificación de calidad. Capacidad de evaluación de perfomance de sistemas de software y sistemas que integren hardware y software.
- LS-CE9- Analizar y evaluar proyectos de especificación, diseño, implementación, puesta a punto, mantenimiento y actualización de sistemas de procesamiento de datos, con capacidad de incorporación de tecnologías emergentes del cambio tecnológico.

CONTENIDOS MINIMOS (de acuerdo al Plan de Estudios)

- GPU: Introducción a GPGPU
- Arquitecturas GPU - Modelo GPU-CPU.
- Modelo de Programación GPU - Resolución de aplicaciones - Métricas.
- Modelo y jerarquía de Memoria de GPU.
- Optimizaciones sobre GPU.
- MultiGPUs. Integración con otras arquitecturas. Arquitecturas Híbridas.

PROGRAMA ANALÍTICO

Unidad 1: GPU: Introducción a HPC y GPGPU

Introducción al cómputo de altas prestaciones (HPC). Concepto de sistema paralelo. Hardware y Software paralelo. Clasificación de Hardware paralelo: Taxonomía de Flynn y Clasificación según el modelo de memoria. Jerarquías de memoria. Software paralelo: paralelismo funcional o de control y paralelismo de datos, características de las aplicaciones (CPU, Memoria, Entrada/Salida, Fases), herramientas de desarrollo. Introducción a las arquitecturas GPU y su uso en HPC. Concepto GPGPU: Computación de Propósito General en GPU.

Unidad 2: Arquitecturas GPU - Modelo GPU-CPU

Evolución de las GPUs. Arquitecturas Nvidia. Otras arquitecturas manycore. Modelo de interacción GPU-CPU. Introducción a la planificación de hilos en GPU Nvidia - Concepto de Grid, Bloque, Thread y Warp. Rendimiento y consumo de las arquitecturas GPU según Top500 y Green500.

Unidad 3: Modelo de Programación GPU

Modelo de programación en GPU. Relación con SIMD, modelo SIMT. Modelo de programación CUDA. Concepto de Host y Device. Identificadores. Tipos de datos. Definición de Constantes. Variables: alcance y tiempo de vida. Gestión de memoria, copia explicita CPU-GPU (transferencias H2D) y GPU-CPU (transferencias D2H). Gestión de Hilos: Grid, Bloques, Threads. Dimensiones: 1D, 2D y 3D. Kernel, llamados Síncronos y Asíncronos. Funciones. Identificadores de Threads y Bloques.Planificación de Threads. Sincronización de Threads. Diseño de programas en GPU. Estudio experimental de casos. Métrica de rendimiento: speedup. Análisis de rendimiento. Aceleración en GPU con respecto a CPU.

Ejemplos de resolución de aplicaciones.

Unidad 4: Modelo y jerarquía de Memoria de GPU

Modelo de Memoria de GPU. Jerarquía de Memoria: Registros, Memoria Compartida, Memoria de constantes, Memoria de Texturas, Memoria Global. Memorias Cache: Constantes, Texturas, Nivel 1, Nivel 2. Patrones de Acceso a Memoria Global, relación entre segmentos y cantidad de transacciones. Patrones de Acceso a Memoria Compartida, bancos de memoria, conflicto de bancos, accesos sin conflictos. Concepto de Acceso Coalescente. El problema de la latencia.

Unidad 5: Optimizaciones

Coalescencia y prefetching. Divergencia. Mezcla y granularidad de instrucciones. Ocupación y asignación de recursos.

Ejemplos de resolución de aplicaciones.

Unidad 6: CUDA Streams

Impacto de las transferencias H2D y D2H. Ocultamiento de la latencia. CUDA Streams. Concurrencia a nivel Kernel y a nivel de memoria. Copia de memoria asíncrona. Gestión de eventos.

Unidad 7: Modelos Híbridos

Máquinas multi-GPUs. Arquitectura Híbridas: Modelo Multicore-GPU, Modelo Cluster-GPU y cluster de GPUs, Modelo Multicore-Cluster-GPUs. Heterogeneidad. Distribución de carga.

Unidad 8: Interoperabilidad entre lenguajes de programación

CUDA y su dependencia de lenguajes de programación específicos (C/C++). Interoperabilidad entre lenguajes de programación. Llamadas desde otros lenguajes a CUDA C: Ejecución independiente, bibliotecas y CUDA C embebido en otros lenguajes. Caso de estudio Python. Just-in-Time compilation (JIT). Numba, CuPy y PyCUDA.

BIBLIOGRAFÏA

Grama A, Gupta A, Karypis G, Kumar V. “Introduction to parallel computing”. Second Edition. Pearson Addison Wesley, 2003.

Jason Sanders, Edward Kandrot “CUDA by Example An Introduction to General Purpose GPU Programming”. NVIDIA Corporation, Addison Wesley, 2011.

M. F. Piccoli, “Computación de Alto Desempeño utilizando GPU”. XV Escuela Internacional de Informática. Editorial Edulp, 2011.

John Cheng, Max Grossman, Ty McKercher, “Professional CUDA C Programming”, John Wiley & Sons, 2014.

David B. Kirk, Wen-mei W. Hwu, “Programming Massively Parallel Processors - A Hands-on Approach” 3ed, NVIDIA Corporation and Wen-mei W. Hwu, Elsevier, 2017.

METODOLOGÍA DE ENSEÑANZA

Modalidad presencial

La asignatura sigue una modalidad taller en la cual se alternan clases teóricas con prácticas experimentales. Durante el desarrollo de la asignatura no se tomará asistencia. La existencia de un control de asistencia es con fines estadísticos.

Las clases teóricas introducen los conceptos teóricos de la asignatura aplicables en las prácticas experimentales.

Las prácticas experimentales se desarrollan en la Sala de Cómputo de Postgrado (configurada como un cluster de GPUs) y equipamiento especial del III-LIDI accedido de forma remota.

Modalidad semi-presencial

Dada la no obligatoriedad de las clases, los alumnos en modalidad semi-presencial pueden seguir los temas por el entorno IDEAS y asistir a las consultas que se fijen para los alumnos presenciales.

Se hace notar que por la característica de las tareas experimentales, el alumno deberá tener acceso al menos a un modelo de arquitectura paralela que incluya GPUs para poder realizar los trabajos que se requieren durante el curso. Es recomendable que asistan a las clases prácticas experimentales para el desarrollo de los conocimientos que se adquieren en dichas clases.

Redictado

La cátedra no ve necesidad alguna de realizar un redictado.

EVALUACIÓN

Modalidad presencial

La asignatura se aprueba a partir de un trabajo experimental integrador y un coloquio. El trabajo consiste en resolver un problema específico sobre GPU, puede realizarse en grupos de a lo sumo 2 personas y tiene una única reentrega. El coloquio consiste en una defensa oral del trabajo entregado, contestar preguntas sobre temas conceptuales vistos en las clases y no tiene instancia recuperatoria.

La nota final se computa como el promedio entre la nota del trabajo y la nota del coloquio.

La nota final será válida por un semestre posterior a la finalización de la materia, pasado ese período el alumno deberá revalidar la nota del final mediante un examen escrito.

Modalidad semi-presencial y Evaluación por promoción

Deben cumplir con los mismos requisitos que los alumnos en modalidad presencial.

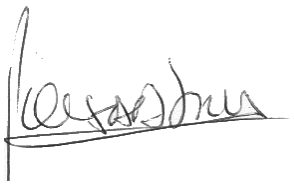
CRONOGRAMA DE CLASES Y EVALUACIONES

Clase	Fecha	Contenidos/Actividades
1	**Semana del 17/8**	Unidad 1: GPU: Introducción a HPC y GPGPU
2	**Semana del 24/8**	Unidad 2: Arquitecturas GPU - Modelo GPU-CPU
3	**Semana del 31/8**	Unidad 3: Modelo de Programación GPU
4	**Semana del 7/9**	Actividades experimentales y consultas Prácticas.
5	**Semana del 14/9**	Actividades experimentales y consultas Prácticas.
6	**Semana del 21/9**	Unidad 4: Modelo y jerarquía de Memoria de GPU
7	**Semana del 28/9**	Unidad 5: Optimizaciones
8	**Semana del 5/10**	Actividades experimentales y consultas Prácticas.
9	**Semana del 12/10**	Actividades experimentales y consultas Prácticas.
10	**Semana del 19/10**	Unidad 6, Unidad 7 y Unidad 8: CUDA Streams, modelos Híbridos e interoperabilidad entre lenguajes de programación. Presentación del trabajo a entregar.
11	**Semana del 26/10**	Consultas.
12	**Semana del 2/11**	Consultas.
13	**Semana del 9/11**	Consultas y Entrega.
14	**Semana del 16/11**	Corrección, Muestra y Coloquios a aprobados en la entrega.
15	**Semana del 23/11**	Consultas.
16	**Semana del 30/11**	Consultas. Reentrega.
17	**Semana del 7/12**	Corrección, Muestra y Coloquios a aprobados en la Re-Entrega.

Evaluaciones previstas	Fecha
Coloquios entrega	Semana 16/11
Coloquios re-entrega	Semana 7/12

Contacto de la cátedra:

Mail: apousa@lidi.info.unlp.edu.ar
Sitio WEB: -
Plataforma virtual: https://ideas.info.unlp.edu.ar, Webex
Otros: -



Firma del/los profesor/es