

PROGRAMA DE ASIGNATURA
“INFO 335 - Computación de alto rendimiento”

1. INFORMACIÓN GENERAL

1.1 Nombre de la asignatura	: Computación de alto rendimiento
1.2 Código	: <i>por definir</i>
1.3 Créditos	: 4
1.4 Período académico	: I semestre
1.5 Tipo de asignatura	: obligatoria
1.6 Horas Teóricas	: 2
1.7 Horas Prácticas	: 2
1.8 Horas no presenciales	: 6
1.9 Cupo	: 20
1.10: Prerrequisitos	: -
1.11 Prof. Responsable	: Dr. Navarro
1.12 Prof. (es) Colaborador(es)	: Dr. Vega y Dr. Ferrada

2. DESCRIPCIÓN DE LA ASIGNATURA

Este curso se estudia el conocimiento teórico y el manejo práctico de la computación de alto rendimiento para procesar información utilizando arquitecturas paralelas de computadores. La parte teórica se enfoca en los modelos de computación paralela y la forma de analizar la complejidad de ciertos algoritmos paralelos que son los bloques fundamentales para la construcción de soluciones más elaboradas. La parte práctica se enfoca en conocer y usar herramientas de programación paralela para implementar las técnicas en CPUs multicore, Clusters y GPUs.

3. OBJETIVOS

3.1 Objetivo general

- Sintetizar bases teóricas y metodológicas de la computación de alto rendimiento para resolver problemas de que se benefician del cómputo paralelo.

3.2 Objetivos específicos

- Analizar y diseñar algoritmos utilizando el marco teórico de computación paralela.
- Aprender y utilizar las principales herramientas de programación en paralelo.
- Resolver problemas de concurrencia en memoria compartida y distribuida.
- Extraer las principales medidas de rendimiento de una implementación en paralelo.

4. CONTENIDOS

Unidad 1: ¿ Qué es la computación de alto rendimiento ?

Evolución histórica de su necesidad y de la arquitectura Von Neumann.
Concurrencia y Paralelismo, Taxonomía de Flynn.
Arquitecturas de CPU, GPU, FPGA, TPU, Cluster, Grid y Cloud.
Impacto de la computación de alto rendimiento en otras disciplinas.
Desafíos del área en el presente.

Unidad 2: Medidas de Rendimiento, modelos de cómputo y de programación

Trabajo (*work*) y alcance (*span*) en el tiempo, speedup, eficiencia.
Escalabilidad fuerte y débil (*Strong Scaling, Weak Scaling*).
Modelos de cómputo y de programación en memoria compartida y distribuida.
Herramientas de programación para CPU y GPU.

Unidad 3: Estrategias para paralelizar problemas

Técnicas para paralelizar un problema en CPUs y GPUs.
Algoritmos paralelos.
Algorithmic Skeletons.
Modelo Map-Reduce (Hadoop + Spark).

Aplicación: *Relative Lempel-Ziv*.
Algoritmos Monte Carlo paralelos.
Errores más frecuentes de implementación.
Distribución balanceada de trabajo.

Unidad 4: Computación Distribuida a Gran Escala

Escalabilidad vertical y horizontal.
Herramientas de programación en memoria distribuida con múltiples CPUs.
Problemas de rendimiento al trabajar a gran escala y cómo abordarlos.
Modelo de programación en GPUs y Multi-GPUs.

5. METODOLOGÍA DEL TRABAJO

El curso se compone de:

1. Clases expositivas-activas por parte de los profesores.
2. Presentaciones (por parte de los estudiantes) de síntesis de tópicos de investigación basado en uno o más artículos. Dichas actividades serán evaluadas constituyendo un 40% de la nota final.
3. El estudiante, en paralelo, deberá explorar el estado del arte y seleccionar un problema de investigación dentro del estado del arte de la computación paralela y distribuida (idealmente vinculado a su futuro proyecto de tesis). Como resultado, deberá elaborar un reporte técnico donde analiza el problema seleccionado, propone una solución y la implementa. Dicha actividad constituirá el 60% de la nota final.

6. EVALUACIÓN

La nota final se determinará de la siguiente manera:

- | | | |
|----|----------------------------|--------|
| a) | Lecturas y presentaciones. | : 40 % |
| b) | Reporte Técnico y defensa. | : 60 % |

7. BIBLIOGRAFIA

1. Jaja J., *An Introduction to Parallel Algorithms*, 1992.
2. Kumar et al., *Introduction to Parallel Computing*, 1994.
3. Cormen et al., *Introduction to Algorithms*, 2010.
4. Stallings W., *Computer Organization and Architecture*, 2015.
5. Nvidia, *CUDA C Programming Guide*, 2018.
6. OpenCL Specification, Khronos Group, 2018.
7. C.A. Navarro et al., *A Survey on Parallel Computing and its Applications in Data-Parallel Problems Using GPU Architectures*, 2014.