Facultad de Ciencias Exactas | UNLP

Facultad de INFORMÁTICA | UNIVERSIDAD NACIONAL DE LA PLATA

Especialización en Bioinformática

Computación paralela. Aplicaciones en bioinformática

Año 2022

Duración total: 70hs.

Responsable: Facultad de Informática

Docente Responsable: Dr. Marcelo Naiouf

Docentes: Dr. Adrían Pousa
Dra. Victoria Sanz
Esp. Chichizola Franco

OBJETIVO GENERAL

El objetivo general del curso es conocer técnicas básicas de la computación paralela para el desarrollo de soluciones en bioinformática.

Objetivos Específicos:

Capacitar a los alumnos para:

- Estudiar los elementos básicos de las arquitecturas de procesamiento paralelo y paradigmas de programación paralela.
- Dominar las técnicas básicas para el diseño y análisis de algoritmos paralelos.
- Estudiar las métricas de performance asociadas al paralelismo
- Estudiar un Lenguaje de Programación con facilidades para la programación paralela.
- Plantear casos concretos en bioinformática resolubles sobre arquitecturas multiprocesador

COMPETENCIAS A DESARROLLAR EN RELACION CON EL OBJETIVO DE LA CARRERA

C. 1 - Poder analizar problemas de BioInformática con conocimiento de los fundamentos biológicos e informáticos, para luego resolverlos seleccionando los métodos y herramientas más adecuadas/eficientes para cada caso.

C. 2 - Utilizar distintas técnicas de procesamiento de datos biológicos para su representación/visualización y análisis eficiente, mediante algoritmos de software ejecutados sobre plataformas adecuadas para el tipo y volumen de datos en cuestión.

C. 3 - Tener capacidad para evaluar la eficiencia (en tiempo, recursos utilizados y consumo energético) así como la calidad de las soluciones obtenidas, considerando el ámbito crítico de la aplicación de las mismas.

PROGRAMA

Unidad 1: Conceptos básicos

Paralelismo. Objetivos del procesamiento paralelo.
Proceso y Procesador. Interacción, comunicación y sincronización de procesos.
Concurrencia y Paralelismo. Modelos de Concurrencia.
Impacto del procesamiento paralelo sobre los sistemas operativos y lenguajes de programación.
Concepto de Sistema Paralelo. Concepto de asignación de tareas y balance de carga.
Balance de carga estático y dinámico.

Unidad 2: Arquitecturas orientadas a Procesamiento Paralelo

Paralelismo implícito: tendencias en las arquitecturas de microprocesadores.
Optimización de performance en los sistemas de memoria. Manejo de memoria cache.
Estructura de control y modelos de comunicaciones en plataformas de procesamiento paralelo.
Clasificación por mecanismo de control (SISD, SIMD, MISD, MIMD), por la organización del espacio de direcciones, por la granularidad de los procesadores y por la red de Interconexión.
Análisis del impacto del tiempo de comunicación en el *speed up* alcanzable.
Clusters de PCs. Multiclusters.
Evolución de los procesadores. Memory Wall. Power Wall. ILP Wall. Multicores.
Nuevas arquitecturas multiprocesador (GPUs, Xeon Phi, FPGAs).

Unidad 3: Principios de diseño y evaluación de algoritmos paralelos

Problemas paralelizables y no paralelizables.
Paralelismo perfecto. Paralelismo de datos. Paralelismo funcional. Paralelismo mixto.
Metodología de diseño de algoritmos paralelos.
Técnicas de descomposición. Características de los procesos. Interacción.
Técnicas de mapeo de procesos/procesadores. Balance de carga.
Métodos para minimizar el *overhead* de la interacción entre procesos.
Modelos de algoritmos paralelos. *Master/Slave*. *Divide/Conquer*. *Pipelining*. SPMD. Combinación de modelos.
Métricas de rendimiento tradicionales.
Fuentes de *overhead* en procesamiento paralelo.
Speed up. Eficiencia. Rango de valores. Grado de paralelismo alcanzable. Efecto de la heterogeneidad.
Ley de Amdhal. Ley de Gustafson. Escalabilidad de sistemas paralelos.

Unidad 4: Programación de algoritmos paralelos sobre plataformas multiprocesador.

Concepto de *threads* y multi-*threading*. Estándar Pthreads. Modelo de ejecución. Primitivas de sincronización. Control de atributos en threads.
Estándar OpenMP. Modelo de ejecución. Directivas. Funciones. Técnica de optimización.
Principios de la comunicación/sincronización por pasaje de mensajes. Estándar MPI.
Comunicaciones punto a punto y colectivas.
Combinación de memoria compartida y pasaje de mensajes. Modelo híbrido.
Modelo de programación en GPU. Lenguaje CUDA. Estándar OpenCL. Técnicas de optimización.

Unidad 5: Algoritmos paralelos en bioinformática

Análisis de algoritmos y aplicaciones paralelas en bioinformática. Alineamiento de secuencias (proteínas y ADN). Alineamiento de *short reads*. Métodos y modelos filogenéticos.

METODOLOGIA Y MODALIDAD DE EVALUACION

La metodología se basa en clases presenciales combinadas con actividades experimentales en el Aula y/o Laboratorio para aplicar los conceptos teóricos y que así el alumno adquiera las competencias y habilidades sobre cada uno de los temas que forman parte del contenido de la asignatura.
Se requiere un 80% de asistencia a las clases presenciales (teóricas y prácticas).
El trabajo se complementa con un proyecto que debe desarrollar el alumno para cumplimentar las horas asignadas con soporte tutorizado por el profesor / docentes de la asignatura. El proyeecto se realiza sobre equipamiento físico (o virtualizado en el caso de aplicaciones informáticas).
En las actividades prácticas se utilizarán ejemplos y casos de estudio en los que se realiza el análisis y diseño de algoritmos paralelos. Asimismo se aplicarán métricas de perfomance estudiadas en la teoría, para ver la calidad de los algoritmos desarrollados.
Asimismo se estudiará uno o más lenguajes de programación paralela (o un modelo de lenguaje clásico + biblioteca de comunicaciones entre procesos paralelos) a fin de comparar las facilidades/dificultades de los mismos. En los casos concretos a resolver se buscarán ejemplos vinculados con BioInformática
La evaluación se realizará mediante un examen escrito al final de las clases del curso para evaluar el grado de conocimientos del alumno (20%), el proyecto/desarrollo experimental que deberá entregar el alumno al final de las horas programadas (70%) y la participación y aportaciones de calidad/excelencia a las soluciones propuestas (10%).

ACTIVIDADES EXPERIMENTALES y DE INVESTIGACION

Los trabajos experimentales pueden desarrollarse en cada clase o continuarse en más de una clase. Parten de una especificación/consigna del docente (explicada en la clase) y un trabajo individual o en grupos que interactúan en el que los alumnos resuelven un problema experimental concreto relacionado con la temática.

Los trabajos podrán ser individuales o grupales (máximo 3 alumnos).

Estos trabajos pretenden desarrollar y/o fortalecer las aptitudes de opinión crítica en los temas relativos del curso. Los alumnos deberán sintetizar su comprensión de los temas, al realizar correctamente la tarea experimental propuesta.

También se pretende desarrollar la capacidad de poder comunicar y transmitir los resultados, en presentaciones pautadas a lo largo del curso.

En general, finalizada una actividad, hay una sesión de discusión conjunta donde los participantes comunicarán sus opiniones e intercambiarán los distintos puntos de vista.

Como materiales de estudio, se dispone de:

- Presentaciones multimedia desarrolladas ad-hoc para introducir cada uno de los diferentes ejes temáticos.
- Ejemplos donde se aplican los conceptos teóricos
- Ejercicios prácticos que son desarrollados en clase
- Píldoras formativas con la explicación de algunos temas
- Material de lectura para estudiar y profundizar conceptos abordados en las clases
- Enlaces a artículos de actualidad de repositorios reconocidos en el área
- Libros digitales
- Acceso a equipamiento de laboratorio físico disponible en la Facultad de Informática y la Facultad de Ciencias Exactas, relacionado directamente con el contenido de la asignatura.
- Software específico para determinadas actividades de laboratorio que se detallan en este programa.

BIBLIOGRAFÍA BASICA

- Introduction to Parallel Computing (2003). Grama, Gupta, Karypis, Kumar. Addison Wesley.

- Parallel Programming (2005). Wilkinson, Allen. Prentice Hall.

- Introduction to High Performance Computing for Scientists and Engineers (2011). Georg Hager, Gerard Wellein. CRC Press (2011).

- Parallel Programming for Multicore and Cluster Systems (2010). Thomas Rauber, Gudulla Runger. Springer.

- An introduction to parallel programming (2011). Peter Pacheco. Elsevier.

- Programming Massively Parallel Processors (2017). 3era edición. Kirk, Hwu. Elsevier.

- Bioinformatics: high performance parallel computer architectures (2011). Bertil Schmidt (ed). CRC Press. 2011.