Arquitecturas Avanzadas 2015/2016

Código: 102778
Créditos ECTS: 6

Titulación	Tipo	Curso	Semestre
2502441 Ingeniería Informática	OB	3	1
2502441 Ingeniería Informática	OT	4	1

Contacto

Nombre: Juan Carlos Moure Lopez

Correo electrónico: JuanCarlos.Moure@uab.cat

Uso de idiomas

Lengua vehicular mayoritaria: español (spa)

Equipo docente

Miquel Àngel Senar Rosell

Santiago Marco Sola

Prerequisitos

Seria recomendable haber cursado y aprobado las asignaturas de Fundamentos de Informática, Fundamentos de Computadores, Estructura de Computadores, Metodología de la Programación y Arquitectura de Computadores

Objetivos y contextualización

El objectivo fundamental de la asignatura es que los alumnos adquieran las capacidades de análisis, utilización y evaluación de arquitecturas avanzadas de computadores, para el desarrollo de aplicaciones y servicios informáticos.

Los conceptos básicos que se describirán en las sesiones teóricas son: el procesador multi-núcleo, el sistema multi-procesador de memoria compartida, la evaluación de rendimiento y prestaciones, y los modelos y lenguajes de programación de estos sistemas paralelos.

Los objectivos más específicos son los siguientes:

1. Entender las claves para obtener buen rendimiento de computadores multi-núcleo y multi-procesador, y de aceleradores de cómputo (GPUs)

2. Identificar las oportunidades de paralelización en un algoritmo o aplicación, a diferentes niveles (instrucciones, iteraciones de un bucle, llamadas a funciones...), y expresarlo formalmente

3. Diseñar las estructuras de datos de una aplicación y los algoritmos que acceden a estos datos para favorecer el buen rendimiento de la jerarquía de memoria

4. Analizar, diseñar e implantar algoritmos paralelos con patrones abstractos de cómputo (reducción, transformación...) bajo paradigmas de programación basados en variables compartidas y sincronización; y con lenguajes paralelos actuales, como OpenMP, Cilk y CUDA/Thrust.

5. Verificar la funcionalidad y evaluar el rendimiento de aplicaciones paralelas, identificando los cuellos de botella respecto al rendimiento

6. A partir del análisis de rendimiento, seleccionar el computador adecuado a una aplicación y/o realizar optimizaciones del código que mejoren aún más su rendimiento

Competencias

Ingeniería Informática

- Adquirir hábitos de pensamiento
- Adquirir hábitos de trabajo personal
- Capacidad de analizar y evaluar arquitecturas de computadores, incluyendo plataformas paralelas y distribuidas, así como desarrollar y optimizar software de para las mismas
- Capacidad para analizar, evaluar, seleccionar y configurar plataformas hardware para el desarrollo y ejecución de aplicaciones y servicios informáticos
- Capacidad para concebir y desarrollar sistemas o arquitecturas informáticas centralizadas o distribuidas integrando hardware, software y redes
- Trabajo en equipo

Resultados de aprendizaje

1. Analizar las características de las aplicaciones que requieran altas prestaciones
2. Analizar los modelos, paradigmas y lenguajes de programación paralela disponibles para determinar el que mejor se adecua a las necesidades de la aplicación
3. Configurar plataformas hardware y entornos de programación para el desarrollo y ejecución de aplicaciones de altas prestaciones
4. Desarrollar aplicaciones paralelas basadas en los paradigmas existentes
5. Desarrollar la capacidad de análisis, síntesis y prospectiva
6. Determinar las plataformas más adecuadas para cada tipo de aplicación
7. Evaluar la funcionalidad y el rendimiento de las aplicaciones paralelas/distribuidos desarrolladas
8. Evaluar las prestaciones de las arquitecturas de altas prestaciones, así como la funcionalidad de las aplicaciones
9. Evaluar y predecir el rendimiento de distintas plataformas de cómputo de altas prestaciones para la ejecución de aplicaciones
10. Proponer soluciones informáticas basadas en sistemas de cómputo de altas prestaciones que integren tanto la arquitectura de los componentes hardware del sistema, así como lo interconexión de los mismos, y el diseño del software necesario
11. Trabajar cooperativamente
12. Trabajar de forma autónoma

Contenido

1. Paralelismo de Instrucciones

- Ejecución *pipeline* y múltiple
- Reordenación dinámica de la ejecución de las instrucciones
- Dependencias de datos entre instrucciones en un bucle

2. Paralelismo de Datos

- Análisis del paralelismo de datos de un algoritmo

- Paralelismo SIMD (vectorización) y MIMD (multi-*thread*)
- Arquitecturas Vectoriales e instrucciones SIMD
- GPUs y aceleradores de cómputo (Xeon Phi)

3. Arquitectura de computadores paralelos

- Estructura y arquitectura de los procesadores multi-hilo y multi-núcleo, y de los nodos multi-procesador
- Jerarquía de memoria (NUMA) y soporte para la coherencia de datos en *cache*
- Soporte H/W para la sincronización y para la comunicación entre *threads*

4. Algoritmos Paralelos

- Modelo de paralelismo con variables compartidas y sincronización. Patrones de cómputo paralelo
- Paralelismo de datos (SIMD): operaciones de reducción y de transformación sobre estructures multi-dimensionales
- Paralelismo detareas (MIMD): *master/worker, divide & conquer, pipeline*
- Programación no determinista y sincronización sin *locks*
- Lenguajes OpenMP, Cilk y CUDA (Thrust)

5. Evaluación del Rendimiento de Aplicaciones Paralelas

- Análisis de complejidad, paralelismo y localidad
- Medida del tiempo de ejecución, CPI, ancho de banda e intensidad aritmética
- Trabajo total y camino crítico de la ejecución paralela
- *Speedup* y eficiencia de la ejecución paralela
- Sobrecarga (*overhead*) de la sincronización y de las comunicaciones

Metodología

Clases de Teoría: se expondrán los conocimientos propios de la asignatura. Se describirán los conceptos básicos y se ilustrarán con ejemplos prácticos. Se resaltarán los problemas de aprendizaje más importantes y se mostrará como completar y profundizar en los contenidos. Se discutirán casos prácticos y el profesor detectará los problemas de comprensión y razonamiento más comunes y los resolverá para todos los estudiantes.

Clases de Problemas: se harán actividades cooperativas de resolución de problemas. A partir del trabajo previo individual de los alumnos, éstos harán una puesta en común en grupo y resolverán las dudas que hayan podido surgir. El profesor detectará los problemas de comprensión y razonamiento más comunes y los resolverá por grupos o para todos los estudiantes. Estos seminarios permitirán aportar los conocimientos que le falten al estudiante o indicar dónde se pueden adquirir. Servirán de puente entre las clases de teoría y el trabajo práctico, promoviendo la capacidad de análisis y síntesis del alumno, el razonamiento crítico y la capacidad de resolución de problemas. Las clases de problemas son evaluables, y en ciertas sesiones preestablecidas los alumnos resolverán problemas entregados al comienzo de la sesión y que se deberán entregar al finalizar la sesión como evidencias evaluables.

Clases de Laboratorio: servirán de apoyo a la teoría. Los alumnos dispondrán de la información de prácticas con tiempo suficiente antes de cada sesión, y deberán preparar la parte previa indicada en el informe para que el profesor, al comienzo de la sesión, la pueda revisar. Durante la sesión los alumnos deberán informar al profesor sobre sus avances y los problemas que se puedan encontrar, y al finalizar la sesión entregarán un documento con los resultados de la práctica y un resumen de los problemas encontrados. Durante el curso, los alumnos deberán presentar y defender de forma oral un trabajo práctico, indicando los objetivos, el desarrollo, los resultados obtenidos con gráficas comprensibles, las incidencias más notables y, finalmente, las conclusiones más relevantes.

Actividades

Título	Horas	ECTS	Resultados de aprendizaje
Tipo: Dirigidas			
Prácticas de Laboratorio	12	0,48	9, 7, 4, 11
Problemas	12	0,48	1, 9, 7, 8, 4, 6, 11, 12
Teoría	21	0,84	2, 8, 3, 5, 10
Tutorías Aula	2	0,08	2, 8, 3, 5, 10
Tipo: Supervisadas			
Preparación de la presentación de prácticas	5	0,2	5, 11, 12
Preparación del trabajo a realizar en el laboratorio	15	0,6	9, 7, 4, 11
Tipo: Autónomas			
Estudio autónomo	24	0,96	2, 8, 3, 5, 10, 11, 12
Preparación de Prácticas	30	1,2	1, 9, 4, 11, 12
Resolución de Problemas	20	0,8	1, 7, 8, 5, 6, 11, 12

Evaluación

Las fechas de evaluación continua y entrega de trabajos se publicarán en el Campus Virtual (CV) y pueden estar sujetas a cambios de programación por motivos de adaptación a posibles incidencias. Siempre se informará en el CV sobre estos cambios ya que se entiende que es la plataforma habitual de intercambio de información entre profesor y estudiantes.

La evaluación del alumno se basará en diferentes elementos recogidos de las diferentes actividades de la asignatura :

Problemas resueltos en grupo (Prb1; 1 punto): se hacen durante una sesión y la entrega se hace al finalizar la clase (2 entregas).

Problemas resueltos en grupo (Prb2; 1 punto): se hacen durante una sesión y la entrega se hace al finalizar la clase (2 entregas).

Prácticas de Laboratorio (Lab; 1 punto): con discusión y ayuda del profesor. Es necesario entregar documentos de resultados, respuestas y conclusiones en cada una de las entregas.

Trabajo Práctico en grupo (TrbPract ; 4-6 puntos): análisis, diseño y realización de la optimización de una aplicación seleccionada por los alumnos. Hay que hacer presentaciones orales y entregas durante el curso para hacer el seguimiento del trabajo, y una defensa oral y un documento final donde se exponga todo el trabajo realizado. Se evaluará el grado de dificultad de los retos propuestos, el logro de los objetivos, la exposición y defensa oral, y los documentos escritos.

Prueba Individual (I; 3 puntos) : una prueba de evaluación de conceptos y problemas, sin apuntes.

Nota FINAL = Prb1 + Prb2 + Lab + TrbLab + I

Es necesaria una nota igual o superior a 5 puntos en Nota FINAL, y superior a 1,2 puntos en la Prueba Individual (I) para aprobar la asignatura (límite estricto y innegociable)

Se considerará una evaluación de **No Evaluable** en los casos que el alumnono haya sido evaluado por actividades que supongan más del 20% de la nota total, o en casos excepcionales.

En caso de no superar la evaluación regular existirá una evaluación de recuperación para la prueba individual (I) y para la presentación final del Trabajo Práctico (TrbLAB). Esta recuperación se hará en la fecha marcada por la Escuela.

Si el alumno no alcanza la nota mínima en alguna de las pruebas de evaluación, la calificación numérica en el expediente será el mínimo entre 4,5 puntos y la nota FINAL que sale del cálculo anterior.

++++++++++++++++++++++++++++++

NOTA: Sin perjuicio de otras medidas disciplinarias que se estimen oportunas, de acuerdo con la normativa académica vigente, se calificarán con un cero las irregularidades cometidas por un estudiante que puedan conducir a una variación de la calificación de un acto de evaluación. Por lo tanto, plagiar, copiar, o dejar copiar una práctica o cualquier otra actividad de evaluación implicará suspender con un cero. Si esta actividad tiene una nota mínima asociada entonces la asignatura quedará suspendida. No serán recuperables las actividades de evaluación calificadasde esta forma y por este procedimiento, y por tanto la asignatura quedará suspendida directamente sin oportunidad de recuperarla en el mismo curso académico.

Actividades de evaluación

Título	Peso	Horas	ECTS	Resultados de aprendizaje
Laboratorio	10%	0	0	2, 1, 7, 5, 6, 11
Práctica: presentaciones y discusiones por grupos	40%	3	0,12	2, 1, 9, 7, 8, 3, 4, 5, 6, 10, 11, 12
Pruebas Individuales de Problemas y Teoría	30%	2	0,08	2, 1, 9, 7, 8, 5, 6, 12
Resolución de Problemas en Grupo (I)	10%	2	0,08	2, 1, 9, 7, 8, 4, 5, 6, 10, 11
Resolución de Problemas en Grupo (II)	10%	2	0,08	2, 1, 9, 7, 4, 5, 6, 11

Bibliografía

Computer Architecture: A Quantitative Approach. 5th Edition
John Hennessy, David Patterson, Morgan Kaufmann (Elsevier) 2012 (Cap. 4 y 5)

Structured Parallel Programming: Patterns for efficient computation
M. McCool, J. Reinders, A. Robison, Elsevier, 2012

Parallel Programming for Multicore and Cluster Systems
T. Rauber, G. Rünger, Springer (Elsevier), 2010

Programming Massively Parallel Processors: A Hands-on Approach
D. Kirk, & W.M. Hwu, Morgan Kaufmann (Elsevier), 2010