GUÍA DOCENTE

Computación Paralela y Arquitecturas Específicas y de Altas Prestaciones

octubre de 2010

I.- DATOS INICIALES DE IDENTIFICACIÓN

Nombre de la asignatura:	Computación Científica Avanzada y Arquitecturas Específicas y de Altas Prestaciones
Módulo al que pertenece	Computación Científica Avanzada y Arquitecturas Específicas y de Altas Prestaciones para Percepción y Simulación
Carácter:	Obligatoria
Titulación:	Máster en Computación Avanzada y Sistemas Inteligentes (http://casi.uv.es)
Ciclo:	Postgrado
Créditos asignatura/módulo	5/10 ECTS
Departamento:	Informática (http://www.uv.es/dptinf)
Profesores responsables:	Fernando Pardo, Wladimiro Díaz, Jose Antonio Boluda

II.- INTRODUCCIÓN A LA ASIGNATURA

En esta asignatura se abordan los mecanismos necesarios para alcanzar el máximo rendimiento de las aplicaciones, desde la utilización de los sistemas de computación más avanzados, hasta el uso de las técnicas paralelas en programación de aplicaciones. Es muy importante conocer las diferentes técnicas que existen para aumentar el rendimiento desde un punto de vista hardware, de esta manera, el ingeniero está capacitado para seleccionar y utilizar el sistema que mejor se adapta a la aplicación que pretende implementar. Por otro lado, dado un sistema, su conocimiento le permite al ingeniero sacarle el máximo rendimiento.

En esta asignatura se muestran las principales arquitecturas de computador que permiten obtener el máximo rendimiento dependiendo del tipo de aplicación. Se explican igualmente algunas arquitecturas y estrategias de hardware especialmente interesantes para el tratamiento y captura de información visual. Se aprende a programar algunas arquitecturas menos convencionales como matriciales y vectoriales. También se estudia la forma de sacar el máximo rendimiento de un sistema mediante su monitorización y ajuste. Por último, se aprenden diferentes técnicas paralelas para obtener un mayor rendimiento en sistemas multiprocesadores y multicomputadores.

III.- VOLUMEN DE TRABAJO

Se utiliza la equivalencia establecida por la Universitat de Valencia respecto a créditos ECTS, donde el máximo es de 26.6 horas/crédito x 5 créditos = 133 horas de trabajo total del alumno. En el caso de esta asignatura se fijan 24 horas/crédito lo cual da un total de 120 horas a distribuir en 13 semanas. La distribución por tipo de trabajo es la siguiente:

Asistencia a clases teóricas: Se estiman 17 horas de clase magistral, en las que se intenta exponer los aspectos más difíciles y atender las preguntas de los alumnos sobre el tema. Al tratarse de un público heterogéneo, en cuanto a su formación en arquitectura de computadores, se intenta obviar los detalles de bajo nivel, entrando en las restricciones que desde el punto de vista de la arquitectura debe conocer el programador.

Asistencia a clases prácticas: Aproximadamente 10 horas centradas en ejercicios prácticos con el equipamiento software apropiados, realizados en el laboratorio y guiados por el profesor.

Preparación de trabajos: Una parte importante de la evaluación de la asignatura recae sobre los trabajos que se proponen para su ejecución individualmente o con grupos de dos o tres personas en caso de trabajos de más complejidad. Habrán 3 tipos de trabajos: 1) programación de procesadores matriciales, 2) técnicas de optimización y 3) programación paralela. El volumen de trabajo correspondiente debería fijarse en unas 20 horas por trabajo y persona, lo cual da 60 horas de trabajo.

Estudio de las clases de teoría: La experiencia en cursos similares sugiere que el tiempo de estudio de una clase por parte del alumno, si se le dan los materiales para ello, no debería ser mucho mayor que el de la propia clase, teniendo en cuenta que no se trata de entender con detalle todos los aspectos, sino de saber qué es lo importante, y preparar las preguntas que se quieran hacer. Así pues, se estima en 20 horas.

Asistencia a tutorías: Esencialmente, se plantean como tutorías personales, y por ello se estima que en 10 horas cada alumno puede resolver con el profesor todas las dudas del estudio personal y prácticas que no haya podido plantear en las clases de teoría o de prácticas.

Evaluación y presentación de trabajos: La última semana se reserva para la presentación/defensa de los trabajos propuestos y/o examen en el caso de que se haya renunciado a realizar los trabajos o a criterio del profesorado. El trabajo será de 3 horas.

En síntesis:

ACTIVIDAD	Horas/curso
ASISTENCIA A CLASES TEÓRICAS	17
ASISTENCIA A CLASES PRÁCTICAS	10
PREPARACIÓN DE TRABAJOS	60
ESTUDIO CLASES	20
ASISTENCIA A TUTORÍAS	10
EVALUACIÓN	3
TOTAL VOLUMEN DE TRABAJO	120

IV.- OBJETIVOS GENERALES

El objetivo de la asignatura es que el estudiante aprenda a sacar el máximo rendimietno de un sistema y una aplicación. Para ello se familiariza al estudiante con las técnicas de programación paralela y optimización de código en máquinas paralelas. También resulta necesario que el estudiante aprenda los fundamentos de arquitectura de computadores a nivel de procesador, jerarquía de memoria y su interconexión.

V.- CONTENIDOS

- Parte I: Arquitectura de Computadores
  - Introducción
  - Procesador
  - La Memoria
  - Arquitecturas paralelas
    - Multiprocesadores y Multicomputadores
    - Vectoriales y Matriciales
    - Cell, GPUS
  - Arquitecturas específicas para visión artificial: *Vision chips*
- Parte II: Computación de altas prestaciones.
  - Supercomputación y supercomputación hoy.
  - El proceso de optimización de las aplicaciones de supercomputación.
  - Análisis de rendimiento y eficiencia de un programa.
  - Análisis del rendimiento de aplicaciones secuenciales. *Timing* y *Profiling*.
  - Análisis del rendimiento de aplicaciones paralelas. *Speedup* y *Efficiency*. Escalabilidad.
  - Técnicas generales de optimización de aplicaciones secuenciales.
  - Computación paralela. Modelos de programación. El proceso de paralelización

PROGRAMACIÓN PARALELA MEDIANTE PASO DE MENSAJES.VI.- DESTREZAS A ADQUIRIR

- Capacidad para distinguir las diferentes arquitecturas y saber aplicar la más adecuada en caso.
- Capacidad para evaluar las diferentes técnicas hardware que permiten aumentar el rendimiento.
- Capacidad para distinguir los sistemas especialmente interesantes para el análisis y captura de información visual.

- Capacidad para programar arquitecturas específicas matriciales, multiprocesador y multicomputador.
- Capacidad para diseñar y llevar a cabo el proceso de optimización de aplicaciones de computación de altas prestaciones.
- Capacidad para detectar puntos calientes y analizar el rendimiento de una aplicación en un sistema.
- Capacidad para desarrollar aplicaciones eficientes en el campo de la computación científica de altas prestaciones según el tipo de sistema utilizado.

VII.- HABILIDADES SOCIALES

- Capacidad para elaborar documentación sobre un trabajo realizado

VIII.- BIBLIOGRAFÍA DE REFERENCIA

- Ortega J.; Anguita M.; Prieto A. Arquitectura de Computadores. Ed. Thomsom, 2005.
- Hennessy J. L.; Patterson D. A. Computer Architecture a Quantitative Approach. Third Edition. Morgan Kaufmann Publishers, 2003.
- Sima, D.; Fountain, T.; Kacsur, P. Advanced Computer Architecture. Addison-Wesley, 1998.
- Optimization and Tuning Guide for Fortran, C and C++. Publication No. SC09-1705-00. IBM Canada Ltd. Laboratory, 1993.
- WILKINSON, B.; ALLEN. M. Parallel Programming. Techniques and Applications Using Networked Workstations and Parallel Computers. Prentice Hall, 1999.
- KUMAR, V.; GRAMA, A.; GUPTA, A.; KARYPIS, G. Introduction to Parallel Computing. The Benjamin/Cummings Publishing Company, 1994.
- K.R. Wadleigh; I.L. Crawford. Software Optimization for High Performance Computing: Creating Faster Applications. Prentice Hall, 2000.
- T. Rauber; G. Rünger. Parallel Programming for multicore and cluster systems. Springer-Verlag, 2010.

IX.- METODOLOGÍA

El proceso enseñanza/aprendizaje de esta asignatura se basa principalmente en el seguimiento de las clases de teoría, realización de sesiones prácticas en laboratorio y en la realización de trabajos por parte del estudiante.

En las clases de teoría se describen los aspectos fundamentales que el estudiante debe conocer sobre los objetivos que se buscan. Estas clases no se limitan al profesor impartiendo

una serie de contenidos, sino que el estudiante participa mediante la realización de ejercicios que estimulan el seguimiento de los contenidos teóricos.

En las sesiones de laboratorio el estudiante desarrolla y pone en práctica las destrezas que se han explicado en las sesiones teóricas mediante la utilización de equipamiento y casos de estudio reales. Durante estas secciones los estudiantes reciben el apoyo del profesor, fomentando al mismo tiempo la interacción con el resto de sus compañeros.

X.- EVALUACIÓN DEL APRENDIZAJE

Este curso tiene una componente práctica fuerte, la parte II tiene como objetivo que los estudiantes adquieran sólida formación en el análisis y evaluación de las aplicaciones científicas de alto rendimiento tanto secuenciales como paralelas, desarrollen técnicas efectivas de optimización y mejora del rendimiento y adquieran una cierta destreza en la programación de sistemas paralelos. Para ello se precisa la asistencia a las clases teóricas y prácticas, con lo que la asistencia se considera obligatoria.

La mejor forma de evaluar que se han asumido los contenidos, además de comprobar la asistencia a un 80% de las clases, consiste en completar los trabajos de programación propuestos en la parte II. La parte I tiene también una componente práctica que se debe realizar y evaluar. Los conocimientos más teóricos de la parte I son para la fundamentación del trabajo posterior, con lo que si se hace el trabajo práctico correctamente no es necesaria una evaluación, mediante prueba escrita, de esta primera parte. Sólo en el caso de no asistencia o no poder completar los trabajos propuestos se propondrá un examen, aunque se recomienda no recurrir a esta metodología de evaluación

La presentación/discusión de los trabajos frente a los compañeros y el examen se realiza el último día de la asignatura.