Arquitectura de Computadores

Curso Académico: (2019 / 2020) **Fecha de revisión: 29/04/2019 09:03:43**

Departamento asignado a la asignatura: Departamento de Informática

Coordinador/a: GARCIA SANCHEZ, JOSE DANIEL

Tipo: Obligatoria Créditos ECTS : 6.0

Curso : 3 Cuatrimestre : 1

Rama de Conocimiento: Ingeniería y Arquitectura

REQUISITOS (ASIGNATURAS O MATERIAS CUYO CONOCIMIENTO SE PRESUPONE)

+ Programación
+ Estructura de Computadores
+ Sistemas Operativos

OBJETIVOS

El objetivo de esta asignatura es que el estudiante conozca los conceptos básicos sobre la arquitectura de un computador y el impacto que estos tienen sobre el rendimiento de las aplicaciones y los sistemas informáticos.

Para alcanzar este objetivo, el alumno profundizará en aspectos de las siguientes competencias:

* CECRI9: Capacidad de conocer, comprender y evaluar la arquitectura de los computadores, así como los componentes básicos que los conforman.

* CECRI14: Conocimiento y aplicación de los principios fundamentales y técnicas básicas de la programación paralela y concurrente.

* CEIC3: Capacidad de analizar y evaluar arquitecturas de computadores, incluyendo plataformas paralelas, así como desarrollar y optimizar software para las mismas.

DESCRIPCIÓN DE CONTENIDOS: PROGRAMA

1. Fundamentos del diseño de computadores.
1.1. Introducción y fundamentos del diseño de computadores.
1.1.1. Introducción.
1.1.2. Perspectiva histórica.
1.1.3. Clasificación de computadores.
1.1.4. Paralelismo.
1.1.5. Arquitectura del Computador.

2. Evaluación del rendimiento de sistemas informáticos.
2.1. Clasificación de arquitecturas y evaluación.
2.1.1. Tendencias tecnológicas.
2.1.2. Tendencias de potencia y energía.
2.1.3. Tendencias en coste.
2.1.4. Evaluación del rendimiento y métricas de rendimiento.
2.1.5. Benchmarks.
2.1.6. Ley de Amdahl.
2.1.7. Rendimiento del procesador.
2.2. Fiabilidad de Sistemas Informáticos.
2.2.1. Introducción a la fiabilidad.
2.2.2. Diferencia entre fiabilidad y disponibilidad.

3. Paralelismo a nivel de instrucción.
3.1. Introducción al paralelismo a nivel de instrucción.
3.1.1. Introducción a las arquitecturas segmentadas.
3.1.2. Riesgos: estructurales, de datos y de control.
3.1.3. Predicción de salto.
3.1.4. Operaciones multi-ciclo.
3.2. Explotación del paralelismo a nivel de instrucción
3.2.1. Técnicas de compilación e ILP.
3.2.2. Técnicas avanzadas de predicción de salto.
3.2.3. Introducción a la planificación dinámica.
3.2.4. Especulación.
3.2.5. Técnicas de emisión múltiple.
3.2.6. Límites de ILP.
3.2.7. Paralelismo a nivel de hilo.

4. Jerarquía de memoria.
4.1. Optimizaciones de memoria caché.
4.1.1. Principios de memoria caché.
4.1.2. Políticas y estrategias de memoria caché.
4.1.3. Optimizaciones básicas de memoria caché.
4.2. Optimizaciones avanzadas de memoria caché.
4.2.1. Cachés pequeñas y simples.
4.2.2. Predicción de vía.
4.2.3. Acceso segmentado.
4.2.4. Cachés no bloqueantes.
4.2.5. Cachés multi-banco.
4.2.6. Palabra crítica primero y reinicio temprano.
4.2.7. Mezclas en búfer de escritura.
4.2.8. Optimizaciones del compilador.
4.2.9. Lectura adelantada hardware.
4.3. Memoria virtual y máquinas virtuales.
4.3.1. Memoria virtual.
4.3.2. Políticas en memoria virtual.
4.3.3. Tablas de páginas.
4.3.4. Máquinas virtuales.
4.3.5. MMV: Monitores de máquinas virtuales.
4.3.6. Soporte hardware para virtualización.
4.3.7. Tecnologías de virtualización.

5. Multiprocesadores.
5.1. Arquitecturas de memoria compartida simétrica.
5.1.1. Introducción a las arquitecturas multiprocesador.
5.1.2. Arquitecturas de memoria compartida centralizada.
5.1.3. Alternativas de coherencia de caché.
5.1.4. Protocolos de espionaje (snooping).
5.1.5. Rendimiento en SMPs.
5.2. Memoria compartida distribuida.
5.2.1. Introducción a la memoria compartida distribuida.
5.2.2. Bases de los protocolos basados en directorio.
5.2.3. Implementación de protocolos basados en directorio.
5.3. Sincronización en memoria compartida.
5.3.1. Introducción a la sincronización.
5.3.2. Primitivas hardware.
5.3.3. Cerrojos.
5.3.4. Barreras.
5.4. Modelos de consistencia de memoria.
5.4.1. Modelos de memoria.
5.4.2. Consistencia secuencial.
5.4.3. Otros modelos de consistencia.
5.4.4. Caso práctico: procesadores Intel.

6. Modelos de programación paralela y concurrente.
6.1. Introducción a la programación paralela: OpenMP.

6.1.1. Hilos en OpenMP.
6.1.2. Sincronización.
6.1.3. Bucles paralelos.
6.1.4. Sincronización con hilo maestro.
6.1.5. Compartición de datos.
6.1.6. Secciones y planificación.
6.2. Modelo de programación concurrente: Hilos en ISO C++11/14.
6.2.1. Introducción al modelo de concurrencia de C++.
6.2.2. Hilos.
6.2.3. Acceso a datos compartidos.
6.2.4. Esperas.
6.2.5. Tareas asíncronas.
6.2.6. Objetos mutex y variables condición.
6.3. Modelos de consistencia relajada en C++11.
6.3.1. Modelo de memoria de C++.
6.3.2. Tipos atómicos.
6.3.3. Relaciones de ordenación.
6.3.4. Modelos de consistencia.
6.3.5. Barreras.
6.4. Técnicas de programación libre de cerrojos.
6.4.1. Estructuras de datos libres de cerrojo.
6.4.2. Estructuras de datos estáticas libres de cerrojos.
6.4.3. Estructuras de datos dinámicas libres de cerrojos.

ACTIVIDADES FORMATIVAS, METODOLOGÍA A UTILIZAR Y RÉGIMEN DE TUTORÍAS

1. Clases magistrales, donde se presentarán los conocimientos que los alumnos deben adquirir. Para facilitar su desarrollo los alumnos dispondrán de referencias a los textos básicos que les permitirán completar y profundizar en aquellos temas en los cuales estén más interesados.
* Se realizará énfasis en el impacto del rendimiento de los sistemas informáticos en distintos dominios de aplicación ilustrando su impacto sobre las organizaciones.
* Los ejemplos incluidos pondrán énfasis en el reconocimiento de la rápida evolución de las arquitecturas de computadores y la necesidad del aprendizaje a lo largo de la vida para mantenerse como un profesional activo.

2. Prácticas en laboratorios informáticos, en donde se analizarán y desarrollarán diversos trabajos que hacen uso de los distintos conceptos impartidos en la materia. Estas prácticas se desarrollarán en grupo, lo que servirá para fomentar la capacidad de trabajo en equipo de los alumnos.
* Dependiendo de cada caso los alumnos deberán redactar una memoria explicativa de cada práctica o contestar un cuestionario de evaluación.
* Las prácticas podrán incluir, entre otros temas, la evaluación del rendimiento de computadores, la evaluación del impacto del paralelismo a nivel de instrucción, la evaluación de los parámetros del sistema de memoria caché, el estudio del impacto de los modelos de consistencia de memoria (incluyendo modelos relajados) o el estudio del impacto de los modelos de programación paralela.

3. Resolución de ejercicios por parte del profesor de forma que se afiancen los conceptos impartidos en las clases magistrales de teoría.

4. Resolución de ejercicios por parte del alumno, que servirán para autoevaluar sus conocimientos y adquirir las capacidades necesarias.

5. Exámenes.

SISTEMA DE EVALUACIÓN

Peso porcentual del Examen/Prueba Final:	50
Peso porcentual del resto de la evaluación:	50

Un 50% de la calificación final se obtendrá mediante un examen final de evaluación de los conocimientos adquiridos.

El 50% restante será el resultado de un proceso de evaluación continua.

El proceso de evaluación continua incluye:

* Evaluaciones de conocimientos adquiridos a lo largo del curso con un peso del 15%.

Peso porcentual del Examen/Prueba Final:	50
Peso porcentual del resto de la evaluación:	50

* Prácticas a lo largo del cuatrimestre con un peso del 35%.

EXAMEN FINAL DE LA CONVOCATORIA ORDINARIA

En la convocatoria ordinaria los alumnos realizarán un examen final que incluirá todo el contenido de la asignatura tanto teórico como práctico. Para aquellos alumnos que hayan seguido el proceso de evaluación continua el valor de este examen supondrá el 50% de la nota final. El otro 50% corresponderá a la nota obtenida en la evaluación continua.

Se considera que un alumno ha seguido el proceso de evaluación continua si:

* Ha obtenido una calificación mínima en cada práctica de al menos 2 puntos sobre 10.
* Ha obtenido una calificación media ponderada de todas las prácticas de 4.5 puntos sobre 10.

Para la superación de la asignatura, la presentación al examen final es obligatoria para todos los estudiantes. Será necesario acreditar un desempeño mínimo obteniendo una calificación de al menos 4 puntos sobre 10 en el examen. En otro caso, se considerará que el estudiante no supera la asignatura y no se procederá a calcular su calificación.

La nota final se incrementará en 1 punto a aquellos alumnos que realicen todas las pruebas de evaluación continua, obtengan más un 7 en la evaluación continua y al menos 4 puntos en el examen final.

Para aquellos alumnos que no hayan completado el proceso de evaluación continua, el examen final tendrá un valor del 60% de la calificación total de la asignatura. Por tanto, para poder aprobar, el alumno deberá obtener una calificación superior a 8,33 sobre 10 en este examen.

Los alumnos conocerán antes del inicio del periodo de exámenes de la convocatoria ordinaria, la calificación obtenida en el proceso de evaluación continua.

CONVOCATORIA EXTRAORDINARIA

Los alumnos que no superen la asignatura en la convocatoria ordinaria realizarán un examen final en esta convocatoria. La calificación en la convocatoria extraordinaria se realizará de la siguiente manera:

a) Si el estudiante completó, el proceso de evaluación continua (ver apartado anterior), el examen final de esta convocatoria tendrá un peso del 50% y la calificación final tendrá en cuenta el otro 50% obtenido en el proceso de evaluación continua, siempre que se haya obtenido en el examen una calificación superior a 4 puntos.

b) Si el estudiante no completó el proceso de evaluación continua, el examen de esta convocatoria tendrá un valor del 100% de la calificación final de la asignatura.

c) Aunque el estudiante haya seguido el proceso de evaluación continua, el examen de esta convocatoria tendrá un valor del 100% si este criterio le resulta más favorable.

BIBLIOGRAFÍA BÁSICA

- Hennessy, JL y Patterson, DA. Computer Architecture: A Quantitative Approach. 6th Edition., Morgan Kaufmann,, 2017

BIBLIOGRAFÍA COMPLEMENTARIA

- Barlas, G. Multicore and GPU Programming, Morgan Kaufmann, 2014

- Mattson, TG, Sanders, BA y Massingill, BL. Patterns for Parallel Programming., Addison-Wesley., 2004

- Pacheco, P. An introduction to parallel programming, Morgan Kaufmann, 2011

- Stallings, W. Computer Organization and Architecture. 9th Edition., Addison-Wesley., 2012

- Williams, A. C++ Concurrency in Action. Practical Multithreading. 2nd Edition, Manning., 2018