# CIENCIA DE LA COMPUTACIÓN

## COMPUTACIÓN PARALELA Y DISTRIBUIDA

**4 CRÉDITOS**


UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

CE2A

# ÍNDICE

# UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA
# SILABO 2021-1

## 1. ASIGNATURA

CS3P01 - Computación Paralela y Distribuida

## 2. DATOS GENERALES

2.1. **Ciclo**: 7°
2.2. **Créditos:** cuatro (4) créditos
2.3. **Horas de teoría:** dos (2) semanales
2.4. **Horas de práctica:** cuatro (4) semanales
2.5. **Duración del período:** dieciséis (16) semanas
2.6. **Condición:**
- Obligatorio para Ciencia de la Computación
2.7. **Modalidad**: Virtual
2.8. **Requisitos**:
CS2102. Análisis y Diseño de Algoritmos. (5° Sem)
CS2301. Redes y Comunicaciones. (6° Sem)

## 2. PROFESORES

### 2.1. Profesor coordinador del curso

José Antonio Fiestas Iquira ( jfiestas@utec.edu.pe )
Horario de atención: Lunes, 16:00 -17:00

### 3.2. Profesor(es) instructor(es) del curso

José Antonio Fiestas Iquira (jfiestas@utec.edu.pe)
Horario de atención: Lunes, 16:00 -17:00

## 3. INTRODUCCIÓN AL CURSO

La última década ha traído un crecimiento explosivo en computación con multiprocesadores, incluyendo los procesadores de varios núcleos y centros de datos distribuidos. Como resultado, la computación paralela y distribuida se ha convertido de ser un tema ampliamente electivo para ser uno de los principales componentes en la malla de estudios en ciencia de la computación de pregrado. Tanto la computación paralela como la distribuida implica la ejecución simultánea de múltiples procesos, cuyas operaciones tienen el potencial para intercalar de manera compleja. La computación paralela y distribuida construye sobre cimientos en muchas áreas, incluyendo la comprensión de los conceptos fundamentales de los sistemas, tales como: concurrencia y ejecución en

paralelo, consistencia en el estado/manipulación de la memoria, y latencia. La comunicación y la coordinación entre los procesos tiene sus cimientos en el paso de mensajes y modelos de memoria compartida de la computación y conceptos algorítmicos como atomicidad, el consenso y espera condicional. El logro de aceleración en la práctica requiere una comprensión de algoritmos paralelos, estrategias para la descomposición problema, arquitectura de sistemas, estrategias de implementación y análisis de rendimiento. Los sistemas distribuidos destacan los problemas de la seguridad y tolerancia a fallos, hacen hincapié en el mantenimiento del estado replicado e introducen problemas adicionales en el campo de las redes de computadoras.

Por ello, se espera que el estudiante logre dominar las herramientas para lograr la escalabilidad de software, a través del paralelismo, manteniendo la precisión y minimizando el tiempo de ejecución, y de esta manera potenciar el performance logrado por un software secuencial.

## 4. OBJETIVOS

**Sesión 1**:

- Entender la transición de un computador secuencial a un sistema en paralelo (distribuido)
- Describir la arquitectura SMP y observar sus principales características
- Explicar las características de cada clasificación en la taxonomía de Flynn
- Explicar las diferencias entre memoria distribuida y memoria compartida
- Distinguir los tipos de tareas que son adecuadas para máquinas SIMD
- Describir los desafíos para mantener la coherencia de la caché

**Sesión 2**:

- Calcular las implicaciones de la ley de Amdahl para un algoritmo paralelo particular
- Definir los conceptos de camino crítico, trabajo y span
- Calcular el trabajo y el span y determinar el camino crítico con respecto a un diagrama de ejecución paralela
- Definir speed-up y explicar la noción de escalabilidad de un algoritmo en este sentido
- Identificar tareas independientes en un programa que debe ser paralelizado

**Sesión 3**:

- Explicar por qué la sincronización es necesaria en un programa paralelo específico
- Identificar oportunidades para particionar un programa serial en módulos paralelos independientes
- Paralelizar un algoritmo mediante la aplicación de descomposición basada en tareas
- Paralelizar un algoritmo mediante la aplicación de descomposición de datos en paralelo

**Sesión 4**:

- Describir los desafíos clave del desempeño en diferentes memorias y topologías de sistemas distribuidos

- Describir las ventajas y limitaciones de GPUs vs CPUs
- Brindar ejemplos de problemas donde el uso de pipelining sería un medio eficaz para la paralelización
- Usar exclusión mutua para evitar una condición de carrera
- Brindar un ejemplo de una ordenación de accesos entre actividades concurrentes (por ejemplo, un programa con condición de carrera) que no son secuencialmente consistentes
- Brindar un ejemplo de un escenario en el que el bloqueo de mensajes enviados pueden dar deadlock
- Explicar cuándo y por qué mensajes de multidifusión (multicast) o basado en eventos puede ser preferible a otras alternativas
- Escribir un programa que termine correctamente cuando todo el conjunto de procesos concurrentes hayan sido completados
- Brindar un ejemplo de un escenario en el que un intento optimista de actualización puede nunca completarse
- Usar semáforos o variables de condición para bloquear hebras hasta una necesaria pre-condición de mantenga

**Sesión 5**:
- Representar características de una carga de trabajo que permita o evite que sea naturalmente paralelizable
- Implementar un algoritmo dividir y conquistar paralelo (y/o algoritmo de un grafo) y medir empíricamente su desempeño relativo a su análogo secuencial
- Descomponer un problema (por ejemplo, contar el número de ocurrencias de una palabra en un documento) vía operaciones map y reduce

**Sesión 6**:
- Detectar y corregir un desbalance de carga
- Describir como la distribución/disposición de datos puede afectar a los costos de comunicación de un algoritmo
- Detectar y corregir una instancia de uso compartido falso (false sharing)
- Explicar el impacto de la planificación en el desempeño paralelo
- Explicar el impacto en el desempeño de la localidad de datos
- Explicar el impacto y los puntos de equilibrio relacionados al uso de energía en el desempeño paralelo

## 6. COMPETENCIAS Y CRITERIOS DE DESEMPEÑO

Los criterios de desempeño que se van a trabajar en este curso son:

**1.3.** Aplica conocimientos de computación apropiados para la solución de problemas definidos y sus requerimientos en la disciplina del programa. (*nivel 3*)

**2.4**. Resuelve problemas de computación y otras disciplinas relevantes en el dominio. (*nivel 3*)

**7.2**. Analiza y valora el impacto local y global de la computación sobre las personas, las organizaciones y la sociedad. (*nivel 3*)

**9.1**. Reconoce la necesidad del aprendizaje autónomo. (*nivel 2*)

## 7. RESULTADOS DE APRENDIZAJE

Al final del curso de Computación paralela y distribuida se espera que el estudiante sea capaz de:

**RA1**. Construir algoritmos en paralelo utilizando técnicas de concurrencia en hilos y comunicación óptima entre procesos para lograr el alto rendimiento de un programa.

**RA2**. Diseñar soluciones integrales de problemas aplicando técnicas de paralelismo distribuído, compartido, o híbrido en casos prácticos y de carácter interdisciplinario.

**RA3**. Justificar en base a métricas de performance y escalabilidad la implementación de algoritmos paralelos y su impacto global en la solución de problemas.

**RA4**. Identificar la solución a problemas de paralelismo a través del uso de métodos de particionamiento adecuado de la data, minimización de tiempos de ejecución y niveles altos de escalabilidad.

## 8. TEMAS

1. Fundamentos de paralelismo y arquitecturas paralelas
    - 1.1 Del procesamiento secuencial al paralelo
    - 1.2 Procesadores mutlinúcleo. Taxonomia de Flynn
    - 1.3 Metas del Paralelismo: velocidad y precisión
    - 1.4 Memoria compartida vs memoria distribuida.

2: Métodos de paralelismo
    - 2.1 Métricas de velocidad, eficiencia, escalabilidad. Ley de Ahmdal
    - 2.2 DAG
    - 2.3 Operaciones básicas de paralelismo (tabulate, iterate)
    - 2.4 Secuencias, Divide y vencerás
    - 2.5 Modelos computacionales en paralelo (PRAM)
    - 2.6 Broadcast/Reducción

3. Descomposición en paralelo
    - 3.1 Paradigmas de programación en paralelo
    - 3.2 Paralelismo ideal
    - 3.3 Uso de random en paralelo
    - 3.4 Particionamiento, Divide y Vencerás

4. Comunicación y coordinación
    - 4.1 Comunicacion Global, topologias
    - 4.2 Pasos de Mensaje:
        - 4.1.1 MPI, Mensajes Punto a Punto
        - 4.1.2 MPI, Comunicación Colectiva
        - 4.1.3 Blocking vs non-blocking
    - 4.3 Memoria Compartida
        - 4.2.1 OMP, Constructores y cláusulas
        - 4.2.2 CUDA, optimizacion con GPUs
    - 4.4 Programación Híbrida

5. Análisis y programación de algoritmos paralelos
   - 5.1 Aceleración y escalabilidad
   - 5.2 Naturalmente (vergonzosamente) algoritmos paralelos.
   - 5.3 Algoritmos de ordenamiento en paralelo
   - 5.4 Algoritmos de búsqueda en paralelo
   - 5.5 Algoritmos de grafos en paralelo
   - 5.6 Ecuaciones diferenciales parciales
   - 5.7 Ejemplos de algoritmos paralelos no-escalables.

6: Desempeño en paralelo
   - 6.1 Equilibrio de carga.
   - 6.2 La medición del desempeño.
   - 6.3 Programación y contención.
   - 6.4 Consumo de energía y gestión.

## 9. PLAN DE TRABAJO

### 9.1 Metodología

La metodología del curso se enfoca en clases magistrales y proyectos de investigación, en las clases teóricas (1 a la semana) y prácticas (2 a la semana). En las clases teóricas se imparten conocimientos necesarios para el desarrollo de ejercicios prácticos semanales. De esta manera se logrará una evaluación continua del avance del alumno. El conocimiento teórico adquirido se evaluará en los exámenes parcial y final, mientras que el práctico en el proyecto del curso.

### 9.2 Sesiones Teóricas:

Las sesiones de teoría se llevan a cabo en clases magistrales donde se realizarán actividades que propicien un aprendizaje activo, con descripción de casos prácticos, que permitan a los estudiantes interiorizar los conceptos.

### 9.3 Sesiones de Práctica:

Para verificar que los alumnos hayan alcanzado el logro planteado para cada una de las unidades de aprendizaje, realizarán actividades que les permita aplicar los conocimientos adquiridos durante las sesiones de teoría y se les propondrá retos que permitan evaluar el desempeño de los alumnos.

Del mismo modo, se fomenta la participación en equipo a través de proyectos de investigación que se realizan de manera grupal. Estos se presentarán en forma escrita y oral, para dar la oportunidad al alumno, de exponer sus ideas.

Como parte de la evaluación continua se asignan tareas semanales en forma grupal o individual, motivándolos con puntos adicionales en las diferentes etapas de la evaluación del curso.

## 10. SISTEMA DE EVALUACIÓN

| EVALUACIÓN | TEORÍA | PRÁCTICA Y/O LABORATORIO |
|---|---|---|

| | | |
|---|---|---|
| *La ponderación de la evaluación se hará si ambas partes están aprobadas | Evaluación Contínua (**C1**) (10%)<br>Evaluación Contínua (**C2**) (10%)<br>Exámen Parcial (**E1**) (25%)<br>Exámen Final (**E2**) (25%) | Proyecto (**P**) (30%) |
| | 70% | 30% |
| | **100%** | |

Se utilizarán las siguientes rúbricas para medir las competencias del curso, y evaluar las actividades más significativas del curso: enlace

## 11. SESIONES DE APOYO O TUTORÍAS

Este apartado permite formalizar los espacios de apoyo a los estudiantes y que éstos tengan la atención NECESARIA y el tiempo disponible para presentar sus dudas y consultas acerca del curso:

| Semana | Fecha/ Hora | Tema a tratar | Objetivos de la sesión |
|---|---|---|---|
| 2 | 19/04/2021<br>5:00 PM – 6:00 PM | Evaluaciones y rúbricas | Resolver dudas sobre exámenes y rúbricas , así como participación en clase |
| 4 | 03/05/2021<br>5:00 PM – 6:00 PM | Tópicos del curso | Resolver dudas sobre primeras 3 semanas |
| 6 | 17/05/2021<br>5:00 PM – 6:00 PM | Proyecto | Consultas sobre proyecto y rúbricas |
| 8 | 31/05/2021<br>5:00 PM – 6:00 PM | Parcial | Solucionario examen parcial |
| 10 | 14/06/2021<br>5:00 PM – 6:00 PM | Tópicos del curso | Resolver dudas sobre temas tratados en clase |
| 12 | 28/06/2021<br>5:00 PM – 6:00 PM | Proyecto | Consultas sobre presentación de proyecto |

| | | | |
|---|---|---|---|
| 14 | 12/07/2021<br>5:00 PM – 6:00 PM | Final | Consultas sobre ultimas notas y examen final |

## 12. REFERENCIAS BIBLIOGRÁFICAS


- David B. Kirk and Wen-mei W. Hwu. (2013) Programming Massively Parallel Processors: A Hands-on Approach. 2nd. Morgan Kaufmann, isbn: 978-0-12-415992-1.

- Norm Matloff. (2014) Programming on Parallel Machines. University of California, Davis. url: http://heather. cs.ucdavis.edu/~matloff/158/PLN/ParProcBook.pdf.

- Peter S. Pacheco. (2011) An Introduction to Parallel Programming. 1st. Morgan Kaufmann. isbn: 978-0-12-374260- 5.

- Michael J. Quinn. (2003) Parallel Programming in C with MPI and OpenMP. 1st. McGraw-Hill Education Group. isbn: 0071232656.

- Jason Sanders and Edward Kandrot. (2010) CUDA by Example: An Introduction to General-Purpose GPU Program- ming. 1st. Addison-Wesley Professional. isbn: 0131387685, 9780131387683.