TEORÍA DE LA COMPUTACIÓN Y VERIFICACIÓN DE PROGRAMAS AVANZADA

Carrera/ Plan

Licenciatura en Informática Plan 2015/Plan 2012/Plan 2003-07

Año 2021

Año: 4º
Régimen de Cursada: Semestral (2do)
Carácter: Optativa
Correlativas:
Teoría de la Computación y Verificación de Programas
Profesor/es: Ricardo Rosenfeld
Hs. semanales: 6 hs.

FUNDAMENTACIÓN

La materia Teoría de la Computación y Verificación de Programas Avanzada profundiza y extiende los contenidos de la materia correlativa Teoría de la Computación y Verificación de Programas. Fundamentalmente trata la complejidad computacional espacial y la verificación de programas no determinísticos y concurrentes. Por lo tanto, su fundamentación como materia es la misma que la de la materia precedente:

(a) Introduce fundamentos de la teoría de la computación (computabilidad y complejidad computacional) y de la teoría de correctitud de programas (semántica de lenguajes de programación, verificación formal de programas, desarrollo sistemático de programas).

(b) Trata dos importantes pilares de las ciencias de la computación, necesarios en la formación de un profesional de la informática, habiendo éste ya recibido y madurado entre otros, conocimientos de algorítmica y estructuras de datos, matemáticas discretas y lógica matemática.

(c) Como distintos contenidos de la complejidad computacional y de la verificación de programas hoy día están abiertos a distintos caminos de investigación, con esta materia se pretende estimular dicho estudio brindando herramientas básicas y esenciales.

OBJETIVOS GENERALES

Parte 1. Profundización en el estudio de la complejidad computacional espacio-temporal tratada en la materia básica, con foco en la complejidad computacional espacial.

Parte 2. Profundización en el estudio de la teoría de correctitud de programas, con foco en la programación no determinística y concurrente.

RESULTADOS DE APRENDIZAJE

1.1. Describir y explicar los conceptos, teorías y métodos matemáticos relativos a la informática, equipamiento informático, comunicaciones informáticas y aplicaciones informáticas de acuerdo con el plan de estudios (Adecuado).

2.3. Seleccionar y utilizar los correspondientes métodos analíticos, de simulación y de modelización (Básico).

COMPETENCIAS

- LI-CE4- Planificar, dirigir, realizar y/o evaluar proyectos de relevamiento de problemas del mundo real, especificación formal de los mismos, diseño, implementación, prueba, verificación, validación, mantenimiento y control de calidad de sistemas de software/sistemas de información que se ejecuten sobre equipos de procesamiento de datos, con capacidad de incorporación de tecnologías emergentes del cambio tecnológico. Capacidad de análisis, diseño y evaluación de interfases humano computador y computador-computador.

CONTENIDOS MINIMOS

- Jerarquía computacional espacial. Espacio logarítmico y polinomial, determinístico y no determinístico.
- Relación de la jerarquía espacial con la jerarquía temporal. Problemas completos de la jerarquía espacio-temporal.
- Verificación de programas no determinísticos. Fairness.
- Verificación de programas concurrentes. Modelos de memoria compartida y de pasaje de mensajes. Propiedades de tipo safety y liveness.
- Introducción a la lógica temporal. Aplicación en la verificación de programas.
- Introducción a la semántica denotacional.

PROGRAMA ANALÍTICO

Parte 1. Complejidad computacional.

Jerarquía espacial. Espacio logarítmico determinístico y no determinístico. Espacio polinomial determinístico y no determinístico. Teorema de Savitch. Reducciones log-space de problemas. Teorema de Immerman.

Problemas completos de las distintas clases de la jerarquía espacial. Jerarquía espacio-temporal.

Misceláneas: Jerarquía polinomial. Sistema de pruebas interactivas. Criptografía. Máquinas cuánticas.

Parte 2. Verificación de programas.

Verificación de programas no determinísticos. Métodos D y D* de verificación de programas no determinísticos. Sensatez y completitud. El concepto de fairness.

Verificación de programas concurrentes con memoria compartida. Métodos de verificación de programas concurrentes con memoria compartida, sin y con primitivas de sincronización (O, O*, R y R*). Sensatez y completitud de los métodos. Propiedades safety y liveness (ausencia de deadlock, exclusión mutua, ausencia de starvation, etc).

Verificación de programas distribuidos (concurrentes sin memoria compartida). Métodos de verificación de programas distribuidos (AFR y AFR*). Sensatez y completitud de los métodos. Propiedades safety y liveness (ausencia de deadlock, exclusión mutua, ausencia de starvation, etc).

Profundización en la verificación de programas secuenciales determinísticos con procedimientos. Recursión y parámetros.

Lógica temporal. Métodos de verificación de programas reactivos basados en la lógica temporal.

Introducción a la semántica denotacional de programas.

BIBLIOGRAFÏA

Básica

- Computabilidad, Complejidad Computacional y Verificación de Programas. Rosenfeld & Irazábal. EDULP. 2013.
- Teoría de la Computación y Verificación de Programas. Rosenfeld & Irazábal. McGraw Hill y EDULP. 2010.
- Lógica para Informática. Pons, Rosenfeld & Smith. EDULP. 2017.
- Apuntes publicados en la plataforma virtual de gestión de cursos, que varían año a año.

Alguna Bibliografía Complementaria

- Introduction to Automata Theory, Language & Computation. J. Hopcroft & J. Ullman. Prentice-Hall. 1979.
- Computational Complexity. Christos Papadimitriou. Addison-Wesley. 1995.
- Introduction to the Theory of Complexity. Bovet & Crescenzi. Prentice-Hall. 1994.
- Computational Complexity: A Conceptual Perspective. O. Goldreich. Cambridge University Press. 2008.
- Computational Complexity: A Modern Approach. S. Arora & B. Barak. Princeton Univ. 2007.
- The Nature of Computation. C. Moore & S. Mertens. Oxford University Press. 2011.
- Structural Complexity I. J. Balcázar, J. Díaz & J. Gabarró. Springer. 1988.
- Structural Complexity II. J. Balcázar, J. Díaz & J. Gabarró. Springer. 1990.
- Quantum Computing since Democritus. S. Aaronson. Cambridge University Press. 2013.
- Program Verification. N. Francez. Addison-Wesley. 1992.
- Fairness. N. Francez. Springer-Verlag. 1986.
- Verification of Sequential and Concurrent Programs. Apt & Olderog. Springer. 1997.
- Parallel Program Design: A Foundation. K. Chandy & J. Misra. Addison-Wesley. 1988.
- Software Requirements & Specifications. M. Jackson. Addison-Wesley. 1995.
- Logic in Computer Science. M. Huth & M. Ryan. Cambridge University Press. 2004.
- Logic for Mathematicians. A. Hamilton. Cambridge University Press. 1980.
- Introduction to Mathematical Logic. E. Mendelson. CRC Press. 2010.
- The Temporal Logic of Reactive and Concurrent Systems - Specification. Z. Manna & A. Pnueli. Springer-Verlag. 1991.
- Temporal Verification of Reactive Systems - Safety. Z. Manna & A. Pnueli. Springer-Verlag. 1995.
- The Anchored Version of the Temporal Framework. Z. Manna & A. Pnueli. Weizmann Institute Of Science. 1988.
- Mathematical Theory of Program Correctness. J. de Bakker. Englewood Cliffs NJ, Prentice-Hall. 1980.
- The Semantics of Programming Languages. M. Hennessy. John Wiley & Sons.1980.
- The Formal Semantics of programming Languages: An Introduction. G. Winskel. The MIT Press. 1993.
- Computability, Complexity, and Languages. M. Davis. Elsevier. 1983.

METODOLOGÍA DE ENSEÑANZA

Publicación de clases de teoría y de ejercitación (clases prácticas), ámbas estrechamente vinculadas y articuladas. Posibilidad de dictada a distancia.

En las clases teóricas se brindan explicaciones conceptuales, imprescindibles para el abordaje de los trabajos prácticos.

Para asegurar el aprendizaje de los contenidos, se entrega cada dos semanas un trabajo práctico, a resolver obligatoriamente por los alumnos.

Se utiliza la plataformal virtual de gestión de cursos para la publicación de las clases, trabajos prácticos y artículos de interés. También para las consultas de los alumnos, promoviendo un foro de discusión permanente.

Particularmente, para alcanzar la competencia indicada previamente:

> **LI-CE4-** Planificar, dirigir, realizar y/o evaluar proyectos de relevamiento de problemas del mundo real, especificación formal de los mismos, diseño, implementación, prueba, verificación, validación, mantenimiento y control de calidad de sistemas de software/sistemas de información que se ejecuten sobre equipos de procesamiento de datos, con capacidad de incorporación de tecnologías emergentes del cambio tecnológico. Capacidad de análisis, diseño y evaluación de interfases humano computador y computador-computador,

en la cátedra se pone énfasis en el proceso de identificación de problemas del mundo real, su especificación como problemas resolubles desde la informática, y el desarrollo de soluciones verificables para los mismos Se intenta poner al alumno en el contexto de aplicación en el campo de la Informática de los conceptos y métodos matemáticos que se enseñan en el programa de la asignatura. Esta contextualización es informativa y se discuten diferentes casos de aplicación para mostrar la utilidad de las teorías y herramientas matemáticas para resolver diferentes problemas informáticos conocidos por el alumno. Además, se pone a disposición de los alumnos material bibliográfico para profundizar la relación entre los temas matemáticos y las soluciones informáticas.

EVALUACIÓN

Se considera directamente la aprobación de la materia, que consiste en la aprobación de los trabajos prácticos quincenales.

Nota: La diversidad y complejidad de algunos temas y su encadenamiento lógico, ameritan que se haga un seguimiento bastante personalizado sobre los alumnos. El mecanismo de trabajos prácticos quincenales ha demostrado ser un buen esquema en este sentido.

CRONOGRAMA DE CLASES Y EVALUACIONES

Clase	Fecha	Contenidos/Actividades
1	**17/08**	**PARTE 1 (4 clases).** Introducción a la complejidad espacial. Jerarquía espacial.
2	**24/08**	Problemas solubles en espacio logarítimico determinístico y no determinístico. Problemas solubles en espacio polinomial determinístico y no determinístico. Teorema de Savitch. Reducciones log-space de problemas. Teorema de Immerman.
3	**31/08**	Problemas completos de las distintas clases de la jerarquía espacial. Jerarquía espacio-temporal.
4	**07/09**	Misceláneas: jerarquía polinomial, pruebas interactivas, criptografía, máquinas cuánticas.
5	**14/09**	**PARTE 2 (11 clases).** Verificación de programas no determinísticos. Lenguaje GCL. Sintaxis y semántica operacional. Métodos D y D* de verificación de correctitud parcial y total de programas GCL. Sensatez y completitud de los métodos D y D*.
6	**21/09**	El concepto de fairness. Fairness débil y fuerte. Verificación de programas no determinísticos asumiendo hipótesis de fairness.
7	**28/09**	Verificación de programas concurrentes con memoria compartida. Lenguaje SVL. Sintaxis y semántica operacional. Métodos de verificación de correctitud parcial y total de programas SVL (O, O*). Sensatez y completitud de los métodos O y O*. Propiedades safety y liveness (ausencia de deadlock, exclusión mutua, ausencia de starvation, etc).
8	**05/10**	Verificación de programas RVL. Sintaxis y semántica operacional. Métodos de verificación de correctitud parcial y total de programas RVL (R, R*). Sensatez y completitud de los métodos R y R*. Propiedades safety y liveness (ausencia de deadlock, exclusión mutua, ausencia de starvation, etc).
9	**12/10**	Verificación de programas concurrentes sin memoria compartida (distribuidos). Lenguaje CSP. Sintaxis y semántica operacional. Método de verificación de correctitud parcial de programas CSP (AFR). Sensatez y completitud del método AFR.
10	**19/10**	Propiedades safety y liveness en el marco de CSP (ausencia de deadlock, exclusión mutua, ausencia de starvation, etc). Método de verificación de correctitud total de programas CSP (AFR*). Sensatez y completitud del método AFR*.
11 y 12	**26/10 y 02/11**	Ampliación de los métodos H y H* considerando la verificación de procedimientos en el lenguaje PLW. Pasaje de parámetros y recursión.
13 y 14	**09/11 y 16/11**	Lógica temporal lineal y computacional. Modelo computacional concurrente. Verificación de programas reactivos utilizando la lógica temporal.
15	**23/11**	Clase 15. Introducción a la semántica denotacional.

Evaluaciones previstas	Fecha
examinación final (eventual) para promoción	14/12

Contactos de la cátedra (mail, sitio WEB, plataforma virtual de gestión de cursos):

Prof. Ricardo Rosenfeld (rrosenfeld@practia.global)
Plataforma Ideas: Teoría de la Computación y Verificación de Programas

Profesor Ricardo Fabián Rosenfeld