Carrera / Plan

Licenciatura en Informática Plan 2021/Plan 2015

TEORÍA DE LA COMPUTACIÓN Y VERIFICACIÓN DE PROGRAMAS

Año 2026

Año: 4º
Régimen de Cursada: Semestral
Carácter: Obligatoria
Correlativas: Matemática III - Conceptos y Paradigmas de Lenguajes de Programación
Profesor: Ricardo Rosenfeld
Hs semanales teoría: 2 hs
Hs semanales práctica: 4 hs

FUNDAMENTACIÓN

Teoría de la Computación y Verificación de Programas es una materia introductoria de fundamentos de la teoría de la computación (parte 1, computabilidad y complejidad computacional) y de la verificación de programas (parte 2, axiomas y reglas para probar programas, metateoría de la verificación de programas, especificación de programas y semántica de lenguajes de programación).

Se tratan dos importantes pilares de las ciencias de la computación, necesarios en la formación de un profesional de la informática, que ya ha recibido conocimientos de matemática, algorítmica, estructuras de datos y programación. Adicionalmente, se estimula a los alumnos a profundizar en distintos temas hoy día abiertos a la investigación.

OBJETIVOS GENERALES

Parte 1. Modelización de una computadora por medio de una máquina de Turing (MT). Estudio de la capacidad de una MT para resolver problemas computacionales (computabilidad y complejidad computacional).

Parte 2. Introducción a la verificación de programas. Métodos axiomáticos para probar programas procedurales secuenciales determinísticos. Propiedades de los métodos.

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

- CGT1- Identificar, formular y resolver problemas de Informática.
- CGT4- Conocer e interpretar los conceptos, teorías y métodos matemáticos relativos a la informática, para su aplicación en problemas concretos de la disciplina.

CONTENIDOS MINIMOS

- Máquinas de Turing. Modelos equivalentes. Computabilidad, decidibilidad y complejidad computacional temporal y espacial.
- Técnicas de inducción, diagonalización y reducción de problemas.
- Lenguajes formales y autómatas. Jerarquía de Chomsky. Reconocimiento de lenguajes.
- Especificación de programas. Aplicación de la lógica de primer orden. Semántica operacional de los lenguajes de programación.

- Métodos de verificación de programas procedurales secuenciales determinísticos.
- Composicionalidad, sensatez y completitud de los métodos de verificación.

PROGRAMA ANALÍTICO

Parte 1.1. Computabilidad.

Máquinas de Turing (MT). Distintos modelos de MT. Equivalencia de modelos de MT. Computabilidad y decidibilidad. Lenguajes no recursivamente numerables, recursivamente numerables y recursivos. Propiedades de dichos lenguajes. Mapa de la computabilidad.

MT universal. El problema de la detención (Halting Problem) y el problema (de reconocimiento) universal. Diagonalización. Reducción de problemas.

Misceláneas de computabilidad. MT restringidas. Gramáticas. Jerarquía de Chomsky. Generación vs reconocimiento de lenguajes. Máquinas RAM.

Parte 1.2. Complejidad computacional.

Generalidades de la complejidad computacional temporal y espacial de problemas. Representación de problemas. Jerarquía temporal. Tiempo polinomial.

Clases de problemas P y NP. Reducción polinomial de problemas. NP-completitud. El problema de la satisfactibilidad de las fórmulas booleanas (SAT). Teorema de Cook.

Clases de problemas NPI, CO-NP y EXP. Otras clases temporales. Complejidad espacial.

Misceláneas de complejidad computacional. MT con oráculo. Complejidad temporal de los problemas de búsqueda, optimización y conteo. Circuitos booleanos. Jerarquía polinomial. Generalidades de la complejidad computacional de los algoritmos probabilísticos y cuánticos.

Parte 2. Verificación de programas.

Elementos introductorios de la verificación de programas. Estado, programa, especificación, sintaxis y semántica de un lenguaje de programación, propiedades de programas, método axiomático, sensatez y completitud de un método axiomático, inducción matemática y estructural, conjunto bien fundado.

Verificación de programas procedurales secuenciales determinísticos. Métodos axiomáticos para probar correctitud parcial y terminación.

Composicionalidad, sensatez y completitud de los métodos axiomáticos.

Misceláneas de verificación de programas. Verificación con arreglos y procedimientos. Desarrollo sistemático de programas basado en los métodos axiomáticos. Introducción a la verificación de programas concurrentes.

BIBLIOGRAFÍA

Básica

- Teoría de la Computación y Verificación de Programas. Rosenfeld & Irazábal. McGraw Hill y EDULP. 2010.
- Computabilidad, Complejidad Computacional y Verificación de Programas. Rosenfeld & Irazábal. EDULP. 2013.
- Lógica para Informática. Pons, Rosenfeld & Smith. EDULP. 2017.
- Verificación de Programas. Programas Secuenciales y Concurrentes. Rosenfeld. EDULP. 2024.
- Computabilidad y Complejidad Computacional. Rosenfeld. EDULP. 2026.

Complementaria

- Introduction to Automata Theory, Language & Computation. J. Hopcroft & J. Ullman. Prentice-Hall. 1979.
- Introduction to the Theory of Computation. M. Sipser. PWS Publishing. 1997.
- Elements of the Theory of Computation. H. Lewis & C. Papadimitriou. Prentice-Hall 1998.
- Introduction to the Theory of Complexity. D. Bovet & P. Crescenzi. Prentice-Hall. 1994.
- Computational Complexity. C. Papadimitriou. Addison-Wesley. 1995.
- Computational Complexity: A Modern Approach. S. Arora & B. Barak. Princeton Univ. 2007.
- Computational Complexity: A Conceptual Perspective. O. Goldreich. Cambridge University Press. 2008.
- The Nature of Computation. C. Moore & S. Mertens. Oxford University Press. 2011.
- Program Verification. N. Francez. Addison-Wesley. 1992.
- Verification of Sequential and Concurrent Programs. K. Apt & E. Olderog. Springer. 1997.
- Logic in Computer Science. M. Huth & M. Ryan. Cambridge University Press. 2004.
- Introduction to Mathematical Logic. E. Mendelson. CRC Press. 2010.

METODOLOGÍA DE ENSEÑANZA

La asignatura consiste en el dictado de clases de teoría y en clases de ejercitación (clases prácticas), ámbas estrechamente vinculadas y articuladas.

En las clases teóricas se brindan explicaciones conceptuales, con participación e intercambio con los alumnos, que sirven para que los mismos logren resolver satisfactoriamente los trabajos prácticos propuestos.

En las clases prácticas se trabaja a partir del enunciado de ejercicios que se resuelven en las mismas clases, con plena participación de los alumnos.

Para asegurar el aprendizaje de los contenidos dictados, se entrega cada dos semanas un trabajo práctico, a resolver opcionalmente por los alumnos (las entregas correctas implican un bonus en la calificación).

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, se pretenden alcanzar las competencias indicadas previamente:

CGT1- Identificar, formular y resolver problemas de Informática:

En la cátedra se pone énfasis en el proceso de identificación de problemas del mundo real, su especificación como problemas de la informática y el desarrollo de soluciones verificables para los mismos.

CGT4- Conocer e interpretar los conceptos, teorías y métodos matemáticos relativos a la informática, para su aplicación en problemas concretos de la disciplina:

Se intenta orientar al alumno, en el contexto de la Informática, en el uso de los conceptos y métodos matemáticos que se enseñan en la asignatura. Esta contextualización es informativa, 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 material bibliográfico para profundizar la relación entre los temas matemáticos y las soluciones informáticas.

EVALUACIÓN

La aprobación de la cursada consiste en una examinación al final de la materia. La calificación considera además si el alumno cumplió con la entrega correcta de los trabajos prácticos quincenales. Se recomienda enfáticamente la realización de dichos trabajos, que aseguran un mayor aprendizaje de los contenidos dictados.

La aprobación de la materia consiste en otra examinación al final de la materia, salvo que el alumno haya obtenido muy buena calificación en la examinación asociada a la cursada, en cuyo caso queda eximido de la segunda prueba.

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, previos al primer examen, es un buen esquema en este sentido.

En cuanto a la evaluación de las competencias indicadas previamente:

CGT1- Identificar, formular y resolver problemas de Informática:

La evaluación de esta competencia forma parte de las evaluaciones de los trabajos prácticos y de los exámenes finales de la asignatura. La evaluación se refleja en la corrección de los entregables del alumno.

CGT4- Conocer e interpretar los conceptos, teorías y métodos matemáticos relativos a la informática, para su aplicación en problemas concretos de la disciplina:

La evaluación de esta competencia también forma parte de las evaluaciones de los trabajos prácticos y de los exámenes finales, donde se incorporan preguntas específicas del tipo: "¿dónde cree Ud. que es aplicable este conocimiento/método matemático?". La evaluación se refleja en la corrección de los entregables del alumno.

CRONOGRAMA DE CLASES Y EVALUACIONES

Clase	Fecha	Contenidos/Actividades
1	**10/03**	**PARTE 1.1.** Máquinas de Turing (MT). Modelos de MT.
2	**17/03**	Lenguajes recursivos, recursivamente numerables y no recursivamente numerables. Propiedades. Mapa de la computabilidad.
3	**26/03**	Lenguajes y problemas de decisión. MT universal. El problema de la detención y del reconocimiento universal. Diagonalización.
4	**31/03**	Reducciones de problemas. Misceláneas de computabilidad: MT como generadoras de lenguajes, Gramáticas, Jerarquía de lenguajes de Chomsky, Máquinas RAM.
5	**07/04**	**PARTE 1.2.** Generalidades de la complejidad computacional. Jerarquía temporal. Representación de problemas. Las clases de problemas P y NP.
6	**14/04**	Reducciones polinomiales de problemas. Problemas NP-completos. El problema de la satisfactibilidad de las fórmulas booleanas (SAT). Teorema de Cook. Propiedades de los problemas NP-completos.
7	**21/04**	Clases de problemas NPI, CO-NP y EXP. Introducción a la complejidad computacional espacial. Jerarquía espacio-temporal.
8	**28/04**	Misceláneas de complejidad computacional: MT con oráculo, Problemas de búsqueda, optimización y conteo.
9	**05/05**	Misceláneas de complejidad computacional: jerarquía polinomial, circuitos booleanos, algoritmos probabilísticos, pruebas interactivas, algoritmos cuánticos, etc.
10	**12/05**	**PARTE 2.** Definiciones iniciales de la verificación de programas. Estado, programa, especificación, semántica operacional de los lenguajes de programación, correctitud parcial y total de programas, métodos de verificación de programas, sensatez y

		completitud de los métodos de verificación de programas. Lenguaje de programación procedural secuencial determinístico. Sintaxis y semántica operacional del lenguaje.
11	**19/05**	Método de verificación para la correctitud parcial de programas.
12	**26/05**	Método de verificación para la terminación de programas.
13	**02/06**	Composicionalidad. Pruebas de sensatez y completitud de los métodos. Misceláneas de verificación de programas: Verificación con arreglos y procedimientos, Desarrollo sistemático de programas a partir de los sistemas axiomáticos, Introducción a la verificación de programas concurrentes.

Evaluaciones promocionales	**Fecha**
1ra examinación	25/06
2da examinación	02/07
3ra examinación	16/07

Contactos de la cátedra (mail, sitio WEB, plataforma virtual de gestión de cursos):
Prof. Ricardo Rosenfeld (rosenfeldricardo@gmail.com)
JTP Leandro Mendoza (leandro.mdza@gmail.com)
Ayudante Pedro Dal Bianco (dalbianco.pedro@gmail.com)
Adscripto Matías Manzín (matias.manzin@gmail.com)
Adscripto Jeremías Salsamendi (jere4352681@gmail.com)
Plataforma Ideas: Teoría de la Computación y Verificación de Programas

Profesor Ricardo Fabián Rosenfeld