Carrera/ Plan

Licenciatura en Sistemas Planes 2015, 2012, 2003-07

TEORÍA DE LA COMPUTACIÓN Y VERIFICACIÓN DE PROGRAMAS

Año 2019

Año: 4º
Régimen de Cursada: Semestral (1ro)
Carácter: Optativa
Correlativas: Fundamentos de Teoría de la Computación
Profesor/es: Ricardo Rosenfeld
Hs. semanales: 6 hs.

FUNDAMENTACIÓN

Teoría de la Computación y Verificación de Programas es una materia introductoria de 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).

De este modo 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. Al mismo tiempo, como distintos contenidos de la complejidad computacional y de la verificación de programas hoy día están abiertos a distintos caminos de investigación, se pretende con la materia estimular este estudio brindando herramientas básicas y esenciales.

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 (computabilidad, decidibilidad y complejidad computacional temporal), y de la relación entre las MT que reconocen lenguajes, generan lenguajes y calculan funciones.

Parte 2. Estudio introductorio de elementos de la teoría de correctitud de programas (métodos de verificación de programas, propiedades de los métodos). Instanciación inicial en la programación de entrada/salida secuencial determínistica.

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.
- Técnicas de inducción, diagonalización y reducción de problemas.
- Lenguajes formales y autómatas. Jerarquía de Chomsky. Reconocimiento
- de lenguajes.
- Especificación de programas. Semántica operacional de los lenguajes
- de programación. Aplicación de la lógica de primer orden.
- Métodos de verificación de programas. Propiedades de sensatez y completitud.
- Verificación de programas de entrada/salida secuenciales determinísticos.

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.

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

Misceláneas de computabilidad. MT restringidas. Gramáticas. Jerarquía de Chomsky. Generación vs reconocimiento de lenguajes. MT con oráculo. 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.

Misceláneas de complejidad computacional. Complejidad temporal de los problemas de búsqueda, optimización y enumeración. MT probabilísticas. Circuitos booleanos. Jerarquía polinomial. Generalidades de la computación cuántica. Introducción a la complejidad computacional espacial.

Parte 2. Verificación de programas.

Elementos de correctitud de programas (imperativos). Estado, programa, especificación, semántica de lenguajes de programación, correctitud parcial y total de programas, métodos de verificación de programas, sensatez y completitud de los métodos de verificación de programas. Lógica de primer orden, inducción matemática y estructural, relaciones de orden bien fundadas.

Verificación de programas de entrada/salida secuenciales determinísticos. Sistemas deductivos H y H*.

Sensatez y completitud de H y H*.

Misceláneas de verificación de programas. Uso de arreglos y procedimientos. Desarrollo sistemático de programas basado en los sistemas H y H*. Introducción a la verificación de programas concurrentes.

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.

Complementaria

- Introduction to Automata Theory, Language & Computation. Hopcroft y Ullman. Prentice-Hall. 1979.

- Computational Complexity. Christos Papadimitriou. Addison-Wesley. 1995.
- Introduction to the Theory of Complexity. Bovet y Crescenzi. Prentice-Hall. 1994.
- Computational Complexity: A Conceptual Perspective. O. Goldreich. Cambridge University Press. 2008.
- Computational Complexity: A Modern Approach. S. Arora y B. Barak. Princeton Univ. 2007.
- Program Verification. Nissim Francez. Addison-Wesley. 1992.
- Verification of Sequential and Concurrent Programs. Apt y Olderog. Springer. 1997.
- Logic in Computer Science. M. Huth y M. Ryan. Cambridge University Press. 2004.

METODOLOGÍA DE ENSEÑANZA

La asignatura consiste en el dictado de clases de teoría y 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 servirá 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, para 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 resolubles desde 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 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

La aprobación de la cursada consiste en una examinación al final de la materia. La calificación considera si el alumno cumple 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, ha demostrado ser 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 del examen final de la asignatura, reflejándose 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 el examen final, 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	**12/03**	**PARTE 1.1 (5 clases).** Máquinas de Turing. Distintos modelos de máquinas de Turing y equivalencia entre ellos.
2	**19/03**	Lenguajes recursivos, recursivamente numerables y no recursivamente numerables. Propiedades. Mapa de la computabilidad.
3	**26/03**	Lenguajes y problemas de decisión. Máquina de Turing universal. El problema de la detención y del reconocimiento universal. Diagonalización.
4	**01/04**	Reducciones de problemas. Teorema de Rice.
5	**09/04**	Misceláneas de computabilidad. Máquinas de Turing como generadoras de lenguajes. Gramáticas. Jerarquía de lenguajes de Chomsky. Máquinas de Turing restringidas: autómatas finitos, autómatas con pila, autómatas acotados linealmente. Propiedades de las distintas clases de lenguajes de la Jerarquía de Chomsky. Máquinas de Turing con oráculo. Máquinas RAM.
6	**16/04**	**PARTE 1.2 (4 clases).** Generalidades de la complejidad computacional. Jerarquía temporal. Representación de problemas.
7	**23/04**	Tiempo polinomial. Las clases de problemas P y NP.
8	**30/04**	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.
9	**07/05**	Clases de problemas NPI, CO-NP y EXP. Otras clases temporales. Problemas de búsqueda, optimización y enumeración. Misceláneas de complejidad computacional. MT probabilísticas. Circuitos booleanos. Jerarquía polinomial. Generalidades de la computación cuántica. Introducción a la complejidad computacional espacial.
10	**14/05**	**PARTE 2 (4 clases).** Definiciones iniciales de la teoría de correctitud 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 secuencial determinístico PLW. Sintaxis y semántica operacional de PLW.
11	**21/05**	Método de verificación H para la correctitud parcial de programas PLW.
12	**28/05**	Método de verificación H* para la correctitud total de programas PLW (corrección parcial más terminación).
13	**04/06**	Prueba de sensatez y completitud de H y H*. Misceláneas de verificación de programas. Verificación con arreglos y procedimientos. Desarrollo sistemático de programas a partir de los sistemas axiomáticos H y H*. Introducción a la verificación de programas concurrentes.

Evaluaciones previstas	Fecha
1ra examinación final para promoción	11/07
2da examinación final para promoción	16/07
3ra examinación final para promoción	06/08

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

Prof. Ricardo Rosenfeld (rrosenfeld@practia.global)
JTP Leandro Mendoza (leandro.mdza@gmail.com)
Ayudante Ilán Rosenfeld (ilanrosenfeld7@gmail.com)
Plataforma Ideas: Teoría de la Computación y Verificación de Programas

Profesor Ricardo Fabián Rosenfeld