UNIVERSIDAD NACIONAL DE LA PLATA
FACULTAD DE INFORMÁTICA

TEORÍA DE LA COMPUTACIÓN Y VERIFICACIÓN DE PROGRAMAS

Año 2017

Carrera/ Plan:
Licenciatura en Informática
Planes 2003/07-2012-2015
Año: 4º

Régimen de Cursada: Semestral

Carácter: Obligatoria
Correlativas: Matemática III - Conceptos y Paradigmas de Lenguajes de Programación

Profesor: 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 determinística.

CONTENIDOS MÍNIMOS

- ✓ 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. 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 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. Introducción a la complejidad computacional espacial.

Parte 3. 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.

METODOLOGÍA DE ENSEÑANZA

La asignatura consiste en el dictado de 15 clases de teoría y 15 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.

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.

BIBLIOGRAFÍA 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 (saldrá en marzo de 2017). Pons, Rosenfeld & Smith. EDULP 2017.

- ✓ Apuntes publicados en la plataforma virtual de gestión de cursos, que varían año a año.

ALGUNA BIBLIOGRAFÍA COMPLEMENTARIA RECOMENDADA

- ✓ 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.

CRONOGRAMA DE CLASES Y EVALUACIONES

Contenidos/Actividades	Evaluaciones
Parte 1. Computabilidad. Clase 1. Máquinas de Turing. Distintos modelos de máquinas de Turing y equivalencia entre ellos. Clase 2. Lenguajes recursivos, recursivamente numerables y no recursivamente numerables. Propiedades. Mapa de la computabilidad. Clase 3. Lenguajes y problemas de decisión. Máquina de Turing universal. El problema de la detención y del reconocimiento universal. Diagonalización. Clase 4. Reducciones de problemas. Teorema de Rice. Clase 5. 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. **Parte 2. Complejidad computacional.** Clase 6. Generalidades de la complejidad computacional. Jerarquía temporal. Representación de problemas. Clase 7. Tiempo polinomial. Las clases de problemas P y NP.	Trabajos prácticos optativos cada dos clases. Examen final para la cursada, al final de la última clase. Examen final para la aprobación de la materia, luego de la prueba anterior. Los siguientes exámenes de aprobación de la materia se desarrollan mensualmente el primer martes de cada mes.

Clase 8. 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. Clase 9. Clases de problemas NPI, CO-NP y EXP. Otras clases temporales. Problemas de búsqueda, optimización y enumeración. Clase 10. Misceláneas de complejidad computacional. MT probabilísticas. Circuitos booleanos. Introducción a la complejidad computacional espacial. **Parte 3. Verificación de programas.** Clase 11. 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. Clase 12. Método de verificación H para la correctitud parcial de programas PLW. Clase 13. Método de verificación H* para la correctitud total de programas PLW (corrección parcial más terminación). Clase 14. Prueba de sensatez y completitud de H y H*. Clase 15. 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.	

Contactos con la cátedra (mail, página, plataforma virtual de gestión de cursos)

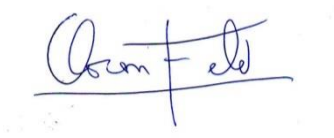
Direcciones de e-mail del plantel docente:

rrosenfeld@pragmaconsultores.com
leandro.mdza@gmail.com
ilanrosenfeld7@gmail.com

Plataforma virtual de gestión de cursos:

Teoría de la Computación y Verificación de Programas. WebUNLP.

Firma del profesor responsable:



Prof. Ricardo Rosenfeld