

PROGRAMA DE CURSO

Código	Nombre			
MA612	Aleatoriedad y Computación			
Nombre en Inglés				
Randomness and Computation				
SCT	Unidades Docentes	Horas de Cátedra	Horas Docencia Auxiliar	Horas de Trabajo Personal
9	15	3	1,5	5,5
Requisitos			Carácter del Curso	
MA5201/CC3102/Autor			Electivo de Doctorado	
Resultados de Aprendizaje				
<p>La aleatoriedad tratada como un recurso computacional juega un rol fundamental en el diseño de algoritmos, en la criptografía moderna, en resultados de inaproximabilidad, y en la teoría de la computación en general. El objetivo de este curso es abordar estos temas y sus interrelaciones. Un aspecto central y subyacente a todos los temas que se abordan es entender Como (y si es que) la aleatoriedad vista como recurso computacional posibilita soluciones algorítmicas eficientes a problemas cuyas soluciones deterministas acarrearán costos computacionales prohibitivos.</p>				

Metodología Docente	Evaluación General
Clases presenciales dictadas por el profesor.	A través de tareas a ser resueltas de forma individual por cada estudiante.

Unidades Temáticas

Número	Nombre de la Unidad	Duración en Semanas
1	Computación aleatorizada	2
Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
<p>Definición de algoritmo y clases probabilistas (e.g. RL, SL, RP, ZPP, BPP). Relaciones entre clases probabilistas y otras clases de complejidad (e.g. $ZPP = RP \cap coRP$, $BPP \subseteq PH$, $BPP \subseteq P/poli$). Reducciones aleatorizadas. BPP y la jerarquía polinomial.</p>	<p>Ser capaz de modelar la aleatoriedad como un recurso computacional, asociarle clases de problemas a los distintos modelos, y descubrir las interrelaciones entre los mismos y con otros modelos de cálculo.</p>	<p>Cap. 7 de [2], Cap. 10 de [8], Cap. 1, 2 y 4 de [1], y [3].</p>

Número	Nombre de la Unidad	Duración en Semanas	
2	Sistemas de demostración interactivos	1.5	
Contenidos		Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Sistemas interactivos de un demostrador y juegos de Arturo-Merlin. Clases de complejidad asociadas (AM, IP, IPpublic y Ipprivate). Definición de sistemas con múltiples demostradores. Clase MIP.		Ser capaz de modelar la aleatoriedad como un recurso computacional usado en un proceso de demostración interactivo, asociarle clases de problemas a los distintos modelos, y descubrir las interrelaciones entre los mismos y con otros modelos de cálculo.	Cap. 8 de [2], y Cap. 10 de [8].

Número	Nombre de la Unidad	Duración en Semanas	
3	Criptografía	1.5	
Contenidos		Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Funciones unidireccionales Generadores pseudoaleatorios criptográficamente seguros. Demostraciones de nula divulgación. Aplicaciones		Entender el papel clave que juega la impredecibilidad de la aleatoriedad en el diseño de protocolos criptográficos. Ser capaz de formalizar los requerimientos que deben cumplir las fuentes de aleatoriedad y relacionar (vía reducciones) distintos supuestos criptográficos.	Cap. 9 de [2], y Cap. 9 de [4].

Número	Nombre de la Unidad	Duración en Semanas	
4	Amplificación de Errores y Códigos Autocorrectores	3	
Contenidos		Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Complejidad de circuitos. Amplificación de la complejidad Lema del XOR de Yao. Amplificación de errores y códigos autocorrectores. Construcción de códigos autocorrectores. Decodificación local. Decodificación de lista.		Ser capaz de deducir resultados de complejidad de circuitos para casi cualquier valor a partir de resultados de complejidad de circuito para una cantidad no despreciable de valores.	Cap. 19 de [2], Cap. 1, 3, 4, y 6 de [7], y [3]

Número	Nombre de la Unidad	Duración en Semanas
5	Derandomización, construcción de grafos expansivos y de extractores	3.5
Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Generadores pseudo-aleatorios y desaleatorización. Complejidad y desaleatorización, construcción de Nisan-Wigderson. Grafos expansivos y sus aplicaciones. Construcción explícita de grafos expansivos. SL c LogSPACE. Fuentes aleatorias débiles y extractores.	Ser capaz de deducir resultados de desaleatorización de modelos probabilistas a partir de resultados de complejidad de circuitos. Entender la interrelación entre grafos aleatorios y grafos expansivos. Poder usar grafos expansivos en el diseño de algoritmos y desaleatorización de modelos probabilistas.	Cap. 20 y 21 de [2], Cap. de [1], Cap. 9 de [1], y [5]

Número	Nombre de la Unidad	Duración en Semanas
6	Demostraciones Probabilistamente Verificables (PCPs) e Inaproximabilidad	3.5
Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Probabilistically Checkable Proofs (PCPs). PCPs e inaproximabilidad. $NP \subseteq PCP(\text{poli}(n); 1)$. $NP = PCP(\log(n); 1)$	Ser capaz de utilizar técnicas de verificación probabilista en el diseño de algoritmos y caracterización de clases de complejidad. Manejar las técnicas básicas para, vía las caracterizaciones mencionadas, deducir resultados de inaproximabilidad.	Cap. 11 y 22 de [2], y [6].

Bibliografía
<p>[1] N. Alon y J. Spencer. The probabilistic method. Interscience Series in Discrete Mathematics and Optimization, John Wiley. Segunda edición. 2000.</p> <p>[2] S. Arora, B. Barak. Computational complexity theory: A modern approach. Cambridge University Press. 2009.</p> <p>[3] R. Boppana, M. Sipser: Complexity of finite functions. In Handbook of Theoretical Computer Science, ed. J. van Leeuwen, 758-804. MIT Press. 1990.</p> <p>[4] O. Goldreich. Computational complexity: A conceptual perspective. Cambridge University Press. 2008.</p> <p>[5] S. Hoory, N. Linial, y A. Wigderson. Expander graphs and their applications. Bulletin of the American Mathematical Society, 43:439-561, 2006.</p> <p>[6] D. Ron, Property testing: A learning theory perspective. Foundations and Trends in Machine Learning. Volume 1, issue 3. Now Publishers Inc. 2008.</p> <p>[7] J. H. van Lint. Introduction to coding theory. Graduate Text in Mathematics, Springer. 1998.</p> <p>[8] Sipser. Introduction to the theory of computation. PWS Publishing. Segunda edición, 2005.</p>

Vigencia desde:	Primavera 2010
Elaborado por:	Marcos Kiwi