Algoritmos y Complejidad

Curso 2019

Pablo R. Fillottrani

Depto. Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur

Primer Cuatrimestre 2019

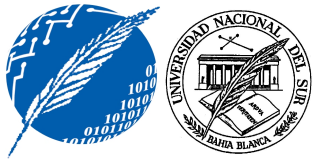
Resumen

Objetivos

Cronograma

Cursado

Recursos



- introducir y aplicar técnicas de análisis de algoritmos
- ilustrar su uso a través de ejemplos prácticos en diversas áreas de las Ciencias de la Computación, mostrando sus aplicaciones y consecuencias
- conocer los elementos básicos y problemas de la Complejidad Computacional

No se pretende memorizar soluciones, sino aprender a aplicar tecnología.

- introducir y aplicar técnicas de análisis de algoritmos
  - analizar la performance asintótica (en tiempo y memoria) de algoritmos, reconocer la importancia de la performance
  - escribir pruebas de correctitud de algoritmos
  - demostrar familiaridad con algoritmos y estructuras de datos conocidas
  - aplicar paradigmas de diseños de algoritmos y métodos de análisis
  - producir algoritmos eficientes en problemas de ingeniería comunes

- ilustrar su uso a través de ejemplos prácticos en diversas áreas de las Ciencias de la Computación, mostrando sus aplicaciones y consecuencias
  - explicar los principales algoritmos de grafos y emplearlos en problemas de ingeniería
  - comparar entre distintas estructuras de datos y elegir la más adecuada para una situación
  - conocer aplicaciones actuales de los algoritmos y predecir sus repercusiones
  - usar estas técnicas en problemas de métodos numericos, criptografía, computación gráfica, geometría computacional, procesamiento de texto, etc

- conocer los elementos básicos y problemas de la Complejidad Computacional
  - modelar el concepto de problema intratable
  - relacionarlo con el problema ya visto de la computabilidad
  - entender el problema de **P** vs **N**P
  - conocer otras clases de complejidad

Resultados esperados

Se espera que el alumno al completar el curso sea capaz de:

- competencias tecnológicas:
  - CT1 identificar, formular y resolver problemas algorítmicos avanzados utilizando estrategias de desarrollo y algoritmos básicos conocidos
  - CT2 conocer estructuras de datos avanzadas y aplicarlas en forma efectiva en el desarrollo de algoritmos
  - CT3 analizar la performance en tiempo y espacio de algoritmos
  - CT4 argumentar sobre la correctitud de algoritmos
  - CT5 conocer alcances y consecuencias de una solución algorítmica a un problema de ingeniería
  - CT6 aplicar el concepto de intratabilidad en particular, y clase de complejidad en general, a la solución de problemas de ingeniería

Resultados esperados

- competencias actitudinales:
  1. desempeñarse en equipo
  2. comunicarse con efectividad
  3. actuar con ética y responsabilidad profesional
  4. aprender en forma continua y autónoma

1. Introducción (1 clase)
   - 1.1 algoritmos y algoritmia
   - 1.2 problemas e instancias
   - 1.3 tipos de análisis de eficiencia
   - 1.4 algunos ejemplos
2. Técnicas y Herramientas (2 clases)
   - 2.1 técnicas de demostración
   - 2.2 herramientas matemáticas básicas
   - 2.3 notación asintótica
   - 2.4 análisis de algoritmos por estructuras de control
   - 2.5 estructuras de datos, algoritmo *Heapsort*
   - 2.6 resolución de recurrencias

3. Algoritmos “Greedy” (1 clase)
   - 3.1 generalidades
   - 3.2 problema de la mochila
   - 3.3 scheduling de procesos
4. Algoritmos “Dividir y Conquistar” (3 clases)
   - 4.1 generalidades
   - 4.2 ordenamiento: mergesort y quicksort
   - 4.3 elemento mediano
   - 4.4 multiplicación de matrices: Strassen
   - 4.5 par de puntos más cercanos
   - 4.6 criptografía – exponenciación modular
   - 4.7 transformada FFT

5. Programación Dinámica (4 clases)
   - 5.1 generalidades
   - 5.2 problema del cambio
   - 5.3 problema de la mochila
   - 5.4 caminos más cortos
   - 5.5 producto de cadenas de matrices (triangularización optimal de polígonos)
6. Algortimos de Grafos (5 clases)
   - 6.1 generalidades.
   - 6.2 árboles de cubrimiento minimales: algoritmos de Kruskal y Prim
   - 6.3 caminos más cortos con origen único: algoritmo de Dijkstra
   - 6.4 problema del viajante
   - 6.5 Recorridos. Propiedades.
   - 6.6 orden topológico
   - 6.7 componentes fuertemente conexos
   - 6.8 puntos de articulación y puentes
   - 6.9 flujo máximo

7. Análisis Amortizado (2 clases)
   - 7.1 uso y principios. Formas de análisis.
   - 7.2 tabla dinámica
   - 7.3 skew heaps
   - 7.4 heaps de Fibonacci
8. Algoritmos Probabilísticos (2 clases)
   - 8.1 introducción
   - 8.2 clasificación
   - 8.3 análisis probabilístico
   - 8.4 ejemplos

9. Complejidad Computacional (5 clases)
   - 9.1 objetivos, conceptos básicos.
   - 9.2 clases de complejidad.
   - 9.3 clase **P**.
   - 9.4 clase **NP**.
   - 9.5 clase **NPC**.
   - 9.6 problema $\mathbf{P} \stackrel{?}{=} \mathbf{NP}$.
   - 9.7 otras clases de complejidad

- sistema de créditos por competencias: actividades permanentes (por lo menos dos por semana) que otorgan (o quitan) créditos para cada una de las competencias tecnológicas presentadas
- cada actividad se evaluará con una de las siguientes calificaciones:
  - 1 otorga el crédito en la competencia de la actividad
  - 0 no otorga crédito
  - -1 resta un crédito en la competencia de la actividad

Cursado

- cada CT tiene un nivel de créditos para cursado y otro para promoción:

competencia	descripción	cursado	promoción	total
CT1	estrategias	5	7	10
CT2	estructuras de datos	4	6	8
CT3	performance	6	8	10
CT4	correctitud	2	3	5
CT5	consecuencias	3	5	8
CT6	complejidad	0	4	6

Cursado

- la nota de promoción será el promedio de los porcentajes del total de créditos obtenidos en cada competencia
- por ejemplo si se obtienen 8, 7, 10, 3, 6, 4 créditos en las CT1-6 respectivamente la nota será

$$(\frac{8}{10} + \frac{7}{8} + \frac{10}{10} + \frac{3}{5} + \frac{6}{8} + \frac{4}{6}) * 10/6 = 7,8 \approx 8$$

- este sistema de evaluación servirá también para las notas de los finales

- Bibliografía básica
  - *Introduction to Algorithms, 3rd. ed*, T. Cormen, C. Leiserson, R. Rivest, C. Stein. MIT Press 2009.
  - *P, NP, and NP-completeness*, O.Goldreich. Cambridge 2010.

Bibliografía adicional

- *The Design and Analysis of Algorithms*, A. Levitin. Addison Wesley 2003.
- *Fundamentals of Algorithms*, G. Brassard, P. Bratley. Prentice Hall 1996.
- *Algorithmics, the Spirit of Computing, 2nd. ed*, D. Harel. Addison Wesley 1992.
- *Computational Complexity, A Modern Approach*, S. Arora, B. Barak. Cambridge 2009.
- *Introduction to the Theory of Complexity*, P. Bovet, P. Crescenzi. Prentice Hall 1993.
- *Computational Complexity*, C. Papadimitriou. Adison Wesley 1994.
- *The Art of Computer Programming, vol. I-III*, D. Knuth. Addison Wesley.
- *Analysis of Algorithms, 2nd ed*, R. Sedgewick. Addison Wesley 2013.

- cualquier otro libro avanzado sobre algoritmos es fuente de técnicas y ejemplos
- Página web del curso

  `cs.uns.edu.ar/~prf/teaching/AyC19`

  para transparencias, prácticos, enunciado del proyecto, links a sitios de interés, noticias, etc.
- Curso moodle-UNS para foros de discusión y compartir archivos digitales