Proyecto Álgebra 1

El proyecto de cambio del dictado de la materia de Álgebra 1 se origina desde la Dirección del Departamento de Computación en el 2012. El mismo año se presentó un pre-proyecto al CODEP y este confirmó que la modificación de Álgebra para incluir aspectos de programación es de interés para el DC. Se encomendó la confección de un proyecto más detallado que incluyera aspectos logísticos y un programa más detallado con vistas a dictar la materia modificada en el 2013.

A continuación se presenta el proyecto en mayor detalle. El proyecto fue elaborado en forma conjunta por una comisión bi-departamental conformada por Pablo De Napoli, Matías Graña, Teresa Krick y Ariel Pacetti (del Departamento de Matemática) y por Flavia Bonomo, Santiago Figueira y Sebastián Uchitel (del Departamento de Computación). Cabe destacar que la idea de incluir aspectos de computación en materias de matemática ya estaba dando vueltas también en el Departamento de Matemática, con lo cual los objetivos globales de este proyecto fueron rápidamente coincidentes.

Motivación

La relación entre la matemática y la computación ha cambiado esencialmente en los últimos 30 años: la computadora la cambió. La matemática lentamente se está adaptando a esta revolución. Más y más la comunidad matemática está reconociendo que además de ser un instrumento numérico y experimental de gran valor, la computación es un objeto de estudio en sí misma, y una herramienta teórica importante. Es una fuente de nuevos problemas excitantes para la matemática.

El incremento del poder computacional conduce a un progreso explosivo en el diseño y análisis de los algoritmos que resuelven los problemas clásicos de la matemática, la ciencia y la ingeniería. Más aún han aparecido nuevas áreas de aplicación como por ejemplo la criptografía, la simulación, el data mining, la teoría del aprendizaje, o la navegación de la red a nivel mundial. Como es natural han aparecido nuevas disciplinas matemáticas y otras han adquirido un renovado vigor.

Sin embargo esta gran revolución no se ve reflejada en la gran mayoría de los casos en la forma en que enseñamos los cursos de matemática en esta facultad. Nuestra currícula es conocida por su excelencia, amplitud y profundidad, pero tenemos que reconocer que esencialmente estamos enseñando lo mismo que hace 30 años, con contenidos detenidos en esa época, sin presentar ni el movimiento constante ni los grandes problemas de interés mundial actuales. Esto en muchos casos tiene como consecuencia una impresión errónea y una desmoralización de los estudiantes que terminan percibiendo también a la matemática como una ciencia muerta, como el resto del mundo exterior.

Esto se agudiza aún más en el caso de la materia Algebra 1, que se dicta tanto para los alumnos de las carreras de computación como las de matemática. A la mayoría de los alumnos se les hace muy difícil apreciar la belleza, interés e importancia de los temas de la materia en su forma actual, ya que en forma general son dictados en forma puramente teórica, con escasa mención a

su resolución concreta (por ejemplo por medio de computadoras) y a las grandes temáticas actuales. Como ejemplo se puede mencionar la relación entre el Pequeño Teorema de Fermat (en su versión de Euler), la dificultad de la factorización, y los algoritmos seguros de encriptación de los cuales dependen toda la seguridad de Internet, bancaria, etc. Esto llama aún más la atención por cuanto muchos temas de la materia se intitulan "Algoritmo" (algoritmo de división, algoritmo de Euclides, algoritmo de factorización, algoritmo de primalidad,…). Agudiza aún más la situación el hecho que una gran proporción de nuestros estudiantes son alumnos de las carreras de computación que no tienen institucionalmente acceso a una computadora ni en el CBC ni en las dos primeras materias que cursan en la facultad, y pueden desmoralizarse por sentirse tan lejos aún de lo que es en principio su vocación. Además, es impensable que hoy en día futuros matemáticos puedan obtener su diploma sin haber tenido prácticamente ningún acceso a las computadoras, ignorando casi por completo las grandes temáticas planteadas por el avance de la computación.

Es para intentar iniciar una corrección de esa situación que se propone un cambio sustancial en la presentación de los temas principales de la materia Algebra 1, tanto para alumnos de las carreras de computación como las de matemática de nuestra facultad. No se trata aquí de modificar en forma relevante los contenidos sino más bien de rever su enfoque y presentación para adecuarlos a los tiempos modernos. La modificación propuesta se orienta a sumar conceptos computacionales que permitan:

a) introducir aspectos efectivos o constructivos en la disciplina a impartir que aumente la capacidad de comprensión de los conceptos teóricos
b) fomentar el uso de la computadora como una herramienta para resolver problemas (numéricos) concretos mediante el aprendizaje de los rudimentos de un lenguaje de programación.

Creemos que estos dos objetivos son igual de valiosos para estudiantes de computación como de matemática.

Para que este proyecto no quede en un estado puramente teórico, se propone la incorporación de un taller de computación para todos los estudiantes, donde serán implementados los algoritmos centrales de la materia. En este taller se introducirán elementos de programación funcional (lenguaje Haskell) y de programación imperativa (lenguaje Python), ya que estos dos aspectos están íntimamente relacionados con los temas centrales de la materia. Por experiencia de los docentes de computación, la programación funcional aparece como natural relacionada con temas como inducción y recursión. El aspecto definicional/computacional de la recursión constituye un elemento pedagógico novedoso que permite el estudio teórico en torno a su correctitud (que resulta más complicado en un lenguaje imperativo), y el estudio práctico de implementación concreta en un lenguaje de programación real. Los estudiantes trabajarán directamente en las computadoras para familiarizarse con el lenguaje de programación y usarlo como herramienta para resolver problemas. También se introducirán nociones de programación imperativa que resultan naturales a la hora de exponer los algoritmos (pseudocódigo) y serán de utilidad para los alumnos que continúen programando, tanto computadores como matemáticos.

Este proyecto, de realizarse, exige una reorganización completa de la materia, donde los temas centrales se presentarán haciendo un hincapié constante en su contraparte efectiva computacional, con la incorporación y comparación de algoritmos. Como es obvio, este objetivo requiere tiempo, y por lo tanto se propone lo siguiente: Un par de temas que en la actualidad se dictan en capítulos separados serán incorporados en los capítulos donde aparecen en forma natural, concentrándonos más particularmente en lo necesario para el desarrollo armonioso y

completo de la materia. De estos, un par de subtemas se desplazará a materias posteriores (a las que pertenecen naturalmente), dado que nuestra experiencia nos ha mostrado que su comprensión exige una madurez que los alumnos no suelen tener a ese nivel de la carrera pero que les resulta muy natural un poco más adelante. Todo esto se detalla en la sección siguiente.

Organización

Los contenidos se reorganizan en 4 grandes capítulos:

1: Conjuntos, relaciones y funciones,
2: Números naturales e Inducción,
3: Números enteros,
4: Polinomios con coeficientes en un cuerpo.

La mayoría de los temas de Combinatoria que figuran en el programa actual se presentan a medida que van apareciendo en forma natural (cantidad de subconjuntos en Conjuntos, factorial y número combinatorio en Inducción, probabilidad elemental cuando se vean algoritmos probabilísticos de primalidad). Se propone que el tema de bolillas con repetición (Bosones) y más ejemplos de probabilidad discreta pasen a una práctica inicial en las respectivas materias de Probabilidades, donde los alumnos entienden rápidamente lo que en Algebra 1 tarda más de una clase.

En este proyecto, el tema Números Complejos aparece en forma natural como ejemplo principal de cuerpo donde se estudian los polinomios, al principio del capítulo Polinomios. Luego, al introducir el Teorema Fundamental del Algebra, se presenta como ejemplo esencial la resolución concreta de los polinomios del tipo $x^n - z$, con z complejo (habiendo desarrollado previamente la forma polar de los números complejos y las fórmulas de De Moivre). Se propone que el tratamiento específico del grupo G_n como tal y de las raíces primitivas de la unidad se haga en la materia Algebra II para los alumnos de la orientación pura de matemática, como ejemplo esencial de grupo abeliano. Nuevamente a ese nivel los estudiantes ya no tienen dificultad para entender lo que en Algebra 1 según nuestra experiencia no terminan de comprender.

Como mencionado arriba, se incorpora un taller de computación que avanzará a la par de la materia, tratando en forma algorítmica los temas presentados en las teóricas y ejercitados en las prácticas. El taller tendrá una sesión semanal, con explicaciones concentradas en un lapso preestablecido de 2 horas. Además ofrecerá, para aquellos alumnos que sientan la necesidad, 1 hora más de consulta semanal distribuida antes y después de las explicaciones. Las prácticas de la materia concentrarán las explicaciones en un lapso preestablecido de 3 horas semanales (1,5 hora cada vez) y ofrecerán además 3 horas semanales de consultas antes y después de las explicaciones en el pizarrón.

De esta manera la carga horaria total de la materia entre teóricas, explicaciones en el pizarrón de las prácticas y explicaciones del taller suman 9 horas semanales preestablecidas (4 horas de teórica, 3 horas de práctica y 2 horas de taller), 1 hora menos que el tiempo total que ocupa la materia hoy en día (10 horas semanales). Pero al ser conscientes de la necesidad de los estudiantes de ese nivel de poder hacer consultas en forma regular, nos parece importante seguir ofreciendo las horas de consulta que existen en la actualidad para que puedan ser aprovechadas por aquellos que las necesitan: como hemos mencionado arriba, serán 3 horas más de consultas semanales de las prácticas y 1 hora más de consultas de taller. Así la disponibilidad total de

docentes de la materia pasa a ser de 13 horas semanales.

El taller de computación se desarrollará en los laboratorios de computación que tienen una capacidad actual de 50 personas cada uno. Los docentes auxiliares del taller no serán los mismos que los docentes de las prácticas. Cada docente cubrirá dos turnos de taller (6 horas semanales en total). A nuestro criterio lo ideal sería que una mitad de los auxiliares dependan del DC y la otra mitad del DM, para que ésta sea realmente una experiencia interdisciplinaria entre dos departamentos, y así poder enriquecerse mutuamente. No tendrían que ser docentes que solo dicten los talleres año tras año, sino que por ejemplo en el DC los docentes provendrán de una área de docencia llamada *Algoritmos* mientras que en el DM confiamos que a medida que este proyecto vaya avanzando más y más docentes auxiliares estarán interesados en ir a los talleres (que figurará además como una de las opciones obligatorias de la encuesta de distribución docente).

Lanzamiento del proyecto

Se propone iniciar este proyecto en el 2do Cuatrimestre del 2013: al ser un segundo cuatrimestre, la materia tiene menos inscriptos que en el primero, y nos parece mejor para experimentar y ajustar los cambios. Durante el primer cuatrimestre se confeccionarán las nuevas prácticas de la materia y se escribirán apuntes teóricos someros para garantizar la continuidad del proyecto. Asimismo los miembros de la comisión (del DM) nos hemos comprometido, de aceptarse este proyecto, a estar a cargo de las teóricas de la materia durante el 2do C. 2013 y el 1r. C. 2014, para ayudar al éxito del plan. El DC se comprometió también a destinar un profesor al taller de computación, durante el 2do C. 2013, quién estará a cargo de dirigir a los auxiliares. Todos los profesores estarán en contacto constante.

Tomamos como modelo Algebra 1, 2do C. 2012, para la preparación del lanzamiento: un total de 200 alumnos inscriptos distribuidos en:
Práctica de la mañana: 100 alumnos = 2 turnos de taller, con 3 docentes al frente c/u
Práctica de la tarde: 30 alumnos = 1 turno de taller, con 2 docentes al frente
Práctica de la noche: 70 alumnos = 2 turnos de talleres con 2 docentes al frente c/u

Esto implica un total de 6 auxiliares para los talleres (cada auxiliar cubre 2 turnos), idealmente 3 provenientes del DM y 3 del DC.

Horarios

Aquí el problema más delicado es encontrar una solución para los dos turnos de taller correspondientes al turno noche, ya que por un lado todas las noches están en la actualidad ocupadas por las prácticas y consultas de las materias Algebra 1 y Análisis 1, y por otro lado los horarios nocturnos son los más demandados para los laboratorios de computación. Sin embargo, dada la necesidad de garantizar un turno noche completo para los alumnos que trabajan y el carácter prioritario de este proyecto, el DC está dispuesto a dedicar 2 laboratorios para que los dos talleres del turno noche se realicen simultáneamente con el objetivo de no perturbar las prácticas de la materia Análisis 1. La solución que proponemos para lograr el funcionamiento del turno noche es entonces la siguiente:

Práctica turno noche: Ma-Vi 17-20 hs, con explicaciones incluidas en el lapso [18-19:30]
Teórica turno noche: Ma-Vi 20-22 hs.

Taller turno noche: Mi 17:30–20:30 hs con explicaciones incluidas en el lapso [18-20]

Además hay que garantizar que en el turno noche de práctica de Análisis 1 de Mi 20-22 hs no se den explicaciones en el pizarrón (así fue ideado ese turno) porque habría un solapamiento de ½ hora con el horario de consulta del taller, cosa que no parece muy grave ya que los alumnos podrían decidir a qué consulta ir por esa ½ hora en función de sus necesidades.
De la misma forma la práctica de Ma-Vi 17:30-20 hs empezaría ½ hora antes del turno nocturno oficial, pero sería un mal menor que terminarlas a las 22:30 a nuestro criterio ya que serían solo consultas, y los alumnos que no pueden llegar a esa hora podrán de todos modos consultar de 19:30 a 20 hs.

Para los horarios de mañana y tarde no habría problema. Por ejemplo:

Turno 1:
Práctica: Ma-Vi 9 a 12 hs.
Teórica: Ma-Vi 12 a 14 hs.
Taller: Ma 15 a 18 hs o Mi 10 a 13 hs

Turno 2:
Teórica: Ma-Vi 12 a 14 hs. (misma teórica que la del Turno 1)
Práctica: Ma-Vi 15 a 18 hs
Taller: Vi 9 a 12 hs o Mi 15 a 18 hs

Turno 3:
Práctica: Ma-Vi 17 a 20 hs.
Teórica: Ma-Vi 20 a 22 hs.
Taller: Mi 17:30 a 20:30 hs

Aprobación

El taller tendrá su instancia de aprobación también, donde los alumnos deberán mostrar, mediante ejercicios sencillos de programación de algoritmos relacionados con la materia, que entendieron. Llegamos a la conclusión que es necesario exigir algún tipo de aprobación si se le quiere dar al taller la importancia que va a tener dentro de la materia.

Luego los trabajos prácticos de la materia se aprobarían con los dos parciales y el taller. Para no penalizar alumnos que no aprueben el taller en primera instancia, no se requerirá su aprobación para inscribirse en los trabajos prácticos de las materias siguientes, pero sí para dar el final de Algebra 1 (esto le da la posibilidad al alumno de aprobar los talleres en el cuatrimestre siguiente por ejemplo).

Programa tentativo

Este programa cubre 27 clases teóricas y prácticas, y 14 talleres, para un cuatrimestre de 16 semanas. Se descontaron de las 32 clases teóricas y prácticas que podría haber en principio 3 clases correspondientes a la clase inmediatamente previa al 1er parcial y la última semana de clase, y 2 clases más por feriados y eventualidades. Si falta tiempo habría que reorganizarlo, si sobra tiempo se pueden agregar contenidos. Se trató de que el programa fuera razonable, pudiendo eventualmente ajustarlo durante la experiencia piloto del 2do C. 2013.

El taller se ubica siempre en este programa entre una clase impar y la par siguiente pues se calcula que en general estará en medio de la semana con respecto a las teóricas y prácticas.

Práctica 1:

Clase 1: Conjuntos: definiciones, pertenencia, contenciones, operaciones (unión, intersección, diferencia). Leyes de De Morgan. Cardinal para conjuntos finitos.

> Taller 0: Introducción al manejo de las computadoras del laboratorio

Clase 2: Tablas de verdad y relación con lógica proposicional. Lógica y conjuntos. Igualdad de conjuntos (diagramas de Venn, tablas).

Clase 3: Producto cartesiano. Conjunto de Partes (y su cardinal para cjtos finitos). Relaciones y su representación como grafos. Relaciones de orden y equivalencia.

> Taller 1: Introducción a lenguaje funcional: idea de que todo es una función. Haskell: Instalación y entorno de programación (grabar, cargar, etc.). Definición de funciones simples sin recursión. Tipos básicos: Integer, Bool, Char. Definición por casos. If then else. Operaciones aritméticas, evaluación de fórmulas de la aritmética. Operaciones booleanas. Evaluación de fórmulas de la lógica proposicional solo con constantes True y False.

Clase 4: Particiones y clases de equivalencia. Funciones (composición, inyectiva, sobreyectiva, inversa, etc).

Clase 5: Funciones (final). Para completar, reordenando los temas de las clases anteriores (que pueden llevar más que lo previsto). Cuantificadores? (noción intuitiva)

> Taller 2: Currificación. Producto cartesiano vs. currificación. Composición de funciones. Evaluación parcial. Evaluación eager vs. lazy. (Tal vez ejemplos con listas infinitas, pero es difícil sin haber dado recursión antes).

Práctica 2:

Clase 6: Inducción, definición “intuitiva” de los números naturales, primeras demostraciones por inducción (simple). Sumatoria, productoria y su escritura como ciclos en un programa.

Clase 7: Factorial y su interpretación combinatoria (biyecciones en conjuntos finitos). Número combinatorio y su interpretación combinatoria (subconjuntos en un conjunto finito), escritura como suma de dos combinatorios, definición recursiva del combinatorio. Definición de funciones recursivas en pseudocódigo (o código en algún lenguaje concreto).

> Taller 3: Recursión. Sumatoria y productoria con y sin filtros (por ejemplo, la suma de todos los naturales <= n que cumplan con el predicado P). Definición recursiva de factorial, números combinatorios, etc.

Clase 8: Definición por los axiomas de Peano de los números naturales. Ejemplos de demostración por inducción global.

Clase 9: Ejemplos de algoritmos recursivos (por ej., algún sort, Hanoi, Fibonacci) y análisis de complejidad. Calculo de a^n por distintos algoritmos (introducción de noción informal de complejidad).

> Taller 4: Tipos algebraicos en Haskell. Pattern matching. Tipo algebraico 1: Naturales como cero y sucesor. Definición de la suma, producto, etc. Tipo algebraico 2: Listas. Recorrido de listas. Búsqueda de un elemento, palíndromo, reverso, etc. Búsqueda de patrones en listas. Lista de listas. Tipo algebraicos 3: fórmula de la lógica proposicional. Programación de satisfacibilidad y validez. ¿Introducción temprana de P y NP? (Se ve más adelante en teórica.)

Clase 10: Inducción global y principio de buena ordenación.

Práctica 3:

Clase 11: Enteros. Divisibilidad y primeras propiedades. Primos y Compuestos. Algoritmo de división.

> Taller 5: Sorting de listas. Fibonacci: algoritmo directo (ineficiente) y con programación dinámica (eficiente). Tipo algebraico 4: Árboles binarios. Recorrido sobre árboles binarios: BFS, DFS.

Clase 12: Aplicaciones del algoritmo de división. Escrituras en distintas bases, sistemas de numeración. Máximo común divisor.

Clase 13: Algoritmo de Euclides (y su complejidad), escritura del gcd como combinación lineal. Números coprimos. Propiedades.

> Taller 6: Algoritmo de División. Programación del algoritmo de Euclides y cálculo de su complejidad. En esta parte todo se puede programar directo en funcional.

Clase 14: Teorema Fundamental de la aritmética. Cantidad de primos. Criba. Aplicaciones del TFA (cantidad de divisores, cálculo de gcd y del mcm).

Practica 4:

Clase 15: Curiosidades de los primos (clase más informativa, sin práctica). Inicio de Congruencias y propiedades.

> Taller 7: Descomposición en factores primos (calcular lista de divisores). la Criba de Eratóstenes (para contestar primalidad de varios números es más eficiente que el test intuitivo hasta las raíz cuadrada y lo de Goldbach.

Clase 16: Congruencias, propiedades y aplicaciones (criterios de divisibilidad). Restos modulo m. Grupos y Anillos (comparación de Z/mZ con Z).

Clase 17: Ecuaciones lineales diofánticas y ecuaciones de congruencia. Algoritmos.

Taller 8: Programación de criterios de divisibilidad con enteros representados como listas de dígitos decimales.

Clase 18: Sistemas de ecuaciones de congruencia. Teorema Chino del Resto.

Clase 19: Pequeño Teorema de Fermat. Algoritmos probabilisticos de primalidad I (introducción informal de algoritmos Montecarlo y Las Vegas)

Taller 9: Programación de resolución de ecuaciones Diofánticas lineales. Programación del teorema Chino del Resto: resolución de sistemas de congruencias.

Clase 20: Algoritmos probabilísticos de primalidad II (números de Carmichael). Relación informal con P vs. NP.

Clase 21: Teorema de Euler-Fermat en general o para pq. RSA.

Taller 10: Algo sobre números pseudo aleatorios. Programación de algoritmos probabilístico de primalidad u otros algoritmos probabilísticos. Logaritmo discreto.

Practica 5: Polinomios en K[X]

Clase 22: Cuerpos. Definición y ejemplos, Q, R, C, Z/pZ. Anillo de polinomios K[x]: generalidades (suma, producto, unidades), grado, divisibilidad, irreducibles y compuestos, algoritmo de división (s/d).

Clase 23: Paralelismo con Z: Mcd, algoritmo de Euclides, coprimos. Factorización única.

Taller 11: Tipo algebraico 5: complejo. Programación de operaciones básicas. Representación de polinomios. Operaciones básicas con las diferentes representaciones. Complejidad de Tiempo y espacio de cada una.

Clase 24: Aspecto funcional: Evaluación de polinomios (definición y algoritmos). Raíces. Teorema del resto. Resolución de cuadráticas en K[X]. Multiplicidad. Equivalencias. Cota para el número de raíces con multiplicidad sobre un cuerpo.

Clase 25: C[X]: Raíces/factorizacion de $X^n - z$ en C[x] con todo lo necesario(coordenadas polares, de Moivre). Teorema Fundamental del Algebra, irreducibles de C[X].

Taller 12: Ejemplos de programación imperativa. Algoritmos para encontrar raíces en polinomios. Ver como funciona algun algoritmo ingenuo de factorizacion sobre Q.

Clase 26: R[X]: Raíces no reales de polinomios reales vienen de a dos, factorización en R[X]. Q[X]: Teorema de Gauss para calcular raíces racionales. Ejemplos de factorización en K[X] para distintos K.

Clase 27: Chismes sobre criterios de irreducibilidad sobre Q y algoritmos de factorización sobre los distintos cuerpos (problemas en general, Kronecker sobre Q)

Taller 13: Más ejemplos de programación imperativa.

Para el 1er parcial entraría hasta la Práctica 3 inclusive. Para el 2do las prácticas 4 y 5.