Integración de Conocimientos de Programación y Computación Numérica en la Universidad Politécnica de Valencia.

José Luis Pérez Gómez , Paolo Rosso
Dept. de Sistemas informáticos y Computación
Universidad Politécnica de Valencia
46020 Valencia
e-mail: jlperez,prosso@dsic.upv.es

Resumen

En este artículo se presenta una experiencia desarrollada en la Escuela Universitaria de Informática (EUI) en la Universidad de Politécnica de Valencia (UPV), dentro del marco de un Proyecto de Innovación Docente (PID). El objetivo de dicho proyecto es integrar los conocimientos de las materias de programación y computación numérica, las cuales son impartidas generalmente en los primeros cursos de las titulaciones en Informática. De esta manera se pretende facilitar el aprendizaje y la motivación de los estudiantes de ambas materias mediante la impartición conjunta de clases por dos profesores, uno de cada materia. Se ha utilizado la videograbación como técnica de evaluación de la experiencia.

1. Introducción

La materia de programación es de vital importancia en la formación del futuro informático ya que los conocimientos en ella adquiridos son utilizados en un gran número de materias impartidas con posterioridad, como es el caso de la computación numérica.

Con el fin de mejorar la docencia de ambas asignaturas, se ha desarrollado una experiencia que pretende mejorar el modelo enseñanza-aprendizaje utilizado en ambas materias mediante la integración de conocimientos de las mismas.

se persiguen los siguientes objetivos.

- *Aumentar la motivación de los alumnos.* Tendrán la posibilidad de experimentar los conceptos teóricos estudiados, mediante ejemplos numéricos extraídos de aplicaciones reales.
- *Facilitar la transferencia del aprendizaje mediante la redundancia***.** En computación numérica se aplicarán a problemas concretos las estrategias de diseño y evaluación aprendidas en programación. Dada la proximidad temporal de ambas materias se espera reforzar los conocimientos adquiridos por los alumnos de programación.

2. Descripción de las asignaturas

Las asignaturas Introducción a la Programación (IPR), Algoritmos y Estructuras de Datos I (AD1) y Algoritmos y Estructuras de Datos II (AD2) conforman el núcleo básico en materia de programación en las tres titulaciones de Informática de la Universidad Politécnica de Valencia: ITIG (Ingeniero Técnico en Informática de Gestión), ITIS (Ingeniero Técnico en Informática de Sistemas) e II (Ingeniero en Informática). Dichas asignaturas proporcionan los fundamentos en materia de diseño y análisis de algoritmos y estructuras de datos.

La asignatura Computación Numérica (CNU) constituye una aproximación a los métodos numéricos en las titulaciones de ITIG e ITIS. Esta asignatura tiene su continuación natural en la asignatura, Algoritmos Numéricos (ALN), perteneciente al 4º curso de la Facultad de Informática (FI) dentro de la titulación de Ingeniero en Informática (II).

La integración de conocimientos se realizará entre las asignaturas AD2 y CNU debido a su proximidad temporal en el plan de estudios vigente. Seguidamente se describen las líneas generales de ambas asignaturas.

2.1 Algoritmos y Estructuras de Datos II

El objetivo de la asignatura del curso 1º, AD2, junto con las asignaturas IPR y AD1, es capacitar al alumno para la especificación, descripción, validación, análisis e implementación de soluciones algorítmicas eficientes a problemas específicos. Concretamente, la asignatura AD2 se centra en la eficiencia de los algoritmos desde el punto de vista de la complejidad temporal y espacial de los mismos [1],[4].

2.2 Computación Numérica

El objetivo de la asignatura del curso 2º, CNU, es que el alumno posea conocimientos de algoritmos eficientes para la resolución de problemas de tipo numérico [3] (resolución de sistemas lineales, integración y derivación numéricas, cálculo de raíces, etc.). Puesto que los problemas resueltos por algoritmos mostrados en esta asignatura son muy costosos (a nivel temporal y a nivel espacial) será de gran importancia analizar cual es la complejidad de los mismos.

2.3 Futuras modificaciones del plan de estudios

El próximo año académico entrará en vigor un nuevo plan de estudios en la EUI y la FI, donde las asignaturas IPR, AD1 y AD2 se integrarán en una única asignatura, de carácter anual, denominada Programación (PRG), y que se impartirá en el 1er curso.

Dicho nuevo plan de estudios también modificará la ubicación de la asignatura CNU, que pasará a impartirse en el segundo cuatrimestre del curso 1º. Estamos convencidos de que, en dicho nuevo plan de estudios, será de gran importancia realizar una integración de conocimientos entre AD2 (parte de la futura PRG) y CNU. Esta integración está motivada por la proximidad de objetivos y por la proximidad temporal de las citadas materias.

Dicha integración se espera que tenga lugar el próximo año académico junto con la aplicación del nuevo plan de estudios. En el año académico actual se ha puesto en marcha una experiencia piloto que afecta, únicamente, a los alumnos pertenecientes a dos grupos especiales de AD2 [2], y a los alumnos de estos grupos que también estén matriculados en CNU.

3. Justificación del proyecto de innovación docente

Después de varios años trabajando en estas asignaturas, se detectan ciertas insuficiencias en el proceso de enseñanza/aprendizaje que se manifiestan principalmente en las actividades de resolución de problemas, así como en los trabajos de laboratorio. Concretamente, las carencias en el proceso de enseñanza/aprendizaje se han manifestado en los siguientes aspectos:

- Los alumnos de las asignaturas de programación, y más concretamente en la asignatura AD2, presentan una elevada tasas de suspensos y no presentados [2].
- Los alumnos de la asignatura AD2 encuentran pocas aplicaciones reales donde se pongan en práctica los conocimientos teóricos.
- Los alumnos de la asignatura CNU presentan una falta de motivación en la asignatura que se ha manifestado en un alto índice de no presentados en los últimos años.
- Los alumnos de la asignatura CNU presentan carencias en el manejo de habilidades adquiridas en asignaturas de programación. Concretamente en el análisis y diseño eficiente de algoritmos iterativos.

Este proyecto se presenta para incidir en la superación de las insuficiencias detectadas a través de la integración de conocimientos de las asignaturas de AD2 y CNU. Pensamos que dicha integración puede favorecer la transferencia del aprendizaje de los alumnos de ambas asignaturas.

4. Metodología de impartición conjunta

El objetivo principal del proyecto, como previamente se ha comentado en la introducción, es la integración de los conocimientos de las asignaturas de AD2 y CNU. Sin embargo, la aplicación del proyecto no pretende modificar sustancialmente los contenidos ni la metodología

de impartición habitual de dichas asignaturas. Se pretende que el alumno aprenda una serie de conocimientos y procedimientos desde la visión ofrecida por la asignatura AD2 y la ofrecida por la asignatura CNU. Desde este punto de vista, la estrategia metodológica que se utilizará será la impartición conjunta de clases entre dos profesores.

Con estas clases se pretende incidir en proceso de enseñanza/aprendizaje dando un enfoque compartido a las dos asignaturas, proporcionando cada uno de los profesores su visión propia del concepto o problema a resolver. Se han impartido un total de tres clases en común: dos en la asignatura AD2 y una en la asignatura CNU. Se pretende incidir en la superación de las insuficiencias detectadas a través de la potenciación de la actividad de resolución de problemas.

Durante cada clase en común se ha fomentado la participación de los alumnos, proponiendo ejercicios con el fin de evaluar el grado de asimilación de los conceptos explicados. Además, al final de cada clase se han propuesto problemas a realizar en grupos de dos alumnos, favoreciendo así el aprendizaje cooperativo y la evaluación de los conocimientos adquiridos. Con la redundancia introducida mediante los ejercicios propuestos se espera que el alumno sea capaz de detectar los esquemas generales estudiados en teoría, favoreciendo así la comprensión de las materias.

Todos los contenidos de las clases han sido expuestos utilizando Power-point como herramienta gráfica.

5. Descripción de dos clases conjuntas

En este apartado se pretende realizar la descripción de dos de las clases conjuntas impartidas: una desde la perspectiva de la asignatura AD2 y otra desde la perspectiva de la asignatura CNU.

5.1 Clase conjunta de AD2

Esta clase ha sido impartida cuando gran parte de los contenidos de la asignatura AD2 (impartida en el primer cuatrimestre del curso) ya habían sido expuestos.

Los *objetivos* que se esperaban obtener en esta clase conjunta eran los siguientes:

- Dar a conocer al alumno la conexión existente entre la asignatura de AD2 y CNU.
- Se espera que el alumno de AD2 posea una visión práctica de la asignatura.
- El alumno debe ser consciente de la importancia de diseñar y analizar algoritmos eficientemente.
- Proporcionar a los alumnos de AD2 ejemplos de problemas procedentes de CNU (problemas numéricos) como ejemplos de uso de las metodologías de diseño estudiadas en AD2.
- Reforzar los conocimientos del alumno en el diseño de algoritmos eficientes utilizando diferentes estrategias (programación iterativa y programación recursiva).
- Reforzar los conocimientos del alumno en el análisis del coste computacional.
- Dar a conocer a los alumnos de AD2 otros paradigmas de programación distintos a los estudiados.

Seguidamente se muestran los *contenidos* de la clase junto con una descripción de los mismos:

I. *Introducción. Conexión entre AD2 y CNU.*

Se ha justificado la importancia de conectar los contenidos de diferentes asignaturas con el fin de que los alumnos posean una visión más amplia de los conocimientos adquiridos, incidiendo especialmente en la conexión de AD2 con la asignatura del siguiente cuatrimestre CNU.

II. *Complejidad temporal de algoritmos.*

Se pretendía reforzar los conocimientos ya adquiridos por el alumno sobre este tema durante el transcurso de la asignatura AD2. Se ha incidido en dos tipos de análisis de la complejidad temporal:

- análisis a priori y
- análisis a posteriori (temporización).

Dentro del análisis de la complejidad temporal a priori se ha incidido en las diferentes estrategias que pueden ser utilizadas según el punto de vista de cada asignatura: en AD2 se estudia el análisis

asintótico, donde únicamente es evaluado el orden de magnitud del número de ejecuciones de la "instrucción crítica". Sin embargo, en CNU el estudio se centra en todas las operaciones donde son involucrados operandos de tipo real, también denominadas "FLOPS" (FLOating Point operationS).

Además, se propuso a los alumnos que razonasen qué tipo de análisis temporal (a priori o a posteriori) es el más adecuado ante dos posibles situaciones reales (a. y b.) que podrían darse en el contexto de una empresa:

a. La empresa dispone de una máquina y tengo que averiguar si un algoritmo se ejecutará en menos de un tiempo dado.
b. Tengo que averiguar que máquina le interesa comprar a la empresa para que un algoritmo se ejecute en menos de un tiempo dado.

III. *Ejemplo de computación numérica.*

Se pretendía mostrar un ejemplo el que se pusiesen en práctica los conocimientos anteriormente comentados utilizando un problema numérico sobradamente conocido por los alumnos y que por tanto no requiriese un extenso desarrollo teórico. El problema escogido fue el *cálculo del valor numérico de un polinomio* p_n en un punto x_0, cuyo desarrollo teórico se muestran en la figura 1.

$$p_n(x_0) = a_n x_0^n + a_{n-1} x_0^{n-1} + \mathbf{K} + a_1 x_0 + a_0$$

Figura 1. Cálculo del valor numérico de un polinomio p_n en un punto x_0

La resolución del problema se dividió en tres fases:

- *Elección de la estructura de datos.*
- *Diseño del algoritmo.* En primer lugar se mostraron dos algoritmos iterativos para resolver el problema: uno sin tener en cuenta criterios de eficiencia y otro en el cual se optimizasen las operaciones realizadas. Finalmente se mostró un diseño recursivo para la resolución del problema .
- *Análisis del coste temporal.* Para cada uno de los algoritmos diseñados se mostró el análisis de la complejidad temporal utilizando la metodología estudiada en AD2 (análisis asintótico de la instrucción crítica) y la futura metodología a utilizar en CNU (número de FLOPS). Además, se utilizó una aplicación en MATLAB, diseñada a tal efecto, que permitía mostrar gráficamente el coste, tanto a priori como a posteriori, de cada uno de los algoritmos estudiados.

IV. *Ejercicios propuestos*

Se enunciaron dos ejercicios propuestos, los cuales debían ser resueltos fuera del horario de la clase por grupos de dos alumnos. La nota obtenida en la resolución de dichos ejercicios podía repercutir en un punto adicional a la nota de teoría de AD2 y medio punto adicional a la nota de CNU. En cada ejercicio propuesto se requería diseñar un algoritmo, bien iterativo o bien recursivo de un problema a resolver. Además se debía realizar el estudio de la complejidad temporal, y opcionalmente realizar la implementación del mismo. Los problemas a resolver en cada ejercicio eran los siguientes:

1. Cálculo de la integral de una función con el método de los rectángulos [3](ver figura 2).

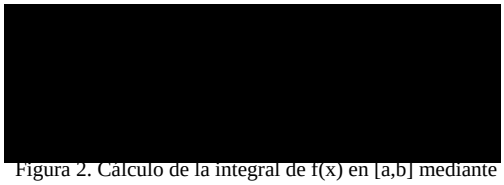

Figura 2. Cálculo de la integral de f(x) en [a,b] mediante el método de los rectángulos, donde N=4, h=(b-a)/4 y $\int_a^b f(x)dx = \sum_{i=1}^{N} f(a + h*(i-1))*h \cdot$

2. Cálculo de la raíz de una función con el método de bisección [3].

V. *El paradigma de programación paralela.*

Para concluir la clase, se comento a los alumnos la existencia de otros paradigmas de programación como el paradigma de programación paralela, el cual puede ser visto como una extensión del paradigma de programación iterativo y que permite reducir el coste de los algoritmos implementados.

5.2 Clase conjunta de CNU

Esta clase ha sido impartida durante la segunda sesión de teoría (CNU se imparte durante el segundo cuatrimestre del curso).

Los *objetivos* que se esperaban obtener en esta clase conjunta eran los siguientes:

- Dar a conocer al alumno la conexión existente entre la asignatura de AD2 y CNU.
- Reforzar conocimientos adquiridos en la asignatura de AD2.
- El alumno debe ser consciente de la importancia de diseñar y analizar algoritmos eficientemente utilizando estrategias de AD2.
- Proporcionar ejemplos de problemas numéricos como ejemplos de uso de las metodologías de diseño estudiadas en AD2.
- Reforzar los conocimientos del alumno en el análisis del coste computacional de algoritmos (temporal y espacial).
- Introducir nuevos conceptos en el análisis del coste computacional propios de CNU.

Seguidamente se muestran los *contenidos* de la clase junto con una descripción de los mismos:

I. *Introducción. Conexión entre AD2 y CNU.*

II. *Complejidad computacional de algoritmos.*

Se pretendía reforzar los conocimientos ya adquiridos por el alumno sobre este tema durante el transcurso de la asignatura AD2. Se ha incidido en los dos tipos de análisis de la complejidad computacional :

- complejidad temporal y
- complejidad espacial.

II. *Complejidad temporal.*

Dentro del análisis de la complejidad temporal a priori se ha incidido en las diferentes estrategias que pueden ser utilizadas según el punto de vista de cada asignatura: en AD2 se estudia el análisis asintótico, en CNU el estudio se centra en estimar el número de FLOPS (ver apartado 5.1).

III. *Ejemplo de estudio de complejidad temporal.*

Se pretendía mostrar un ejemplo que el que se pusiesen en práctica los conocimientos anteriormente comentados utilizando un problema numérico con un planteamiento teórico muy sencillo. El problema escogido fue el *producto de una matriz triangular superior* (T) *por vector* (x). La resolución del problema se dividió en tres fases:

- *Tiempo reservado a los alumnos*. Se utilizaron 15 minutos de la clase para que los alumnos, individualmente o en grupo, diseñasen un algoritmo que resolviese el problema y que analizasen el coste temporal del mismo.

- *Diseños de algoritmos*. Después se mostraron dos algoritmos iterativos utilizando la notación MATLAB para resolver el problema: en el primero de ellos se opera sobre todos los elementos de la matriz (función "mtv") y en el segundo únicamente se opera sobre los elementos de la matriz que son distintos de cero (función "mtv2", ver figura 3).

```
Function y=mtv2(T,x,n)
     for i=1:n;
        y(i)=0;
        for j=i:n;
           y(i)=y(i)+T(i,j)*x(j);
        end
     end
%Fin función mv2
```

Figura 3. Función MATLAB, "mtv2", encargada de realizar el producto de una matriz triangular superior T, de dimensión n, por un vector x

- *Análisis del coste temporal.* Para cada una de las dos funciones diseñadas ("mtv" y "mtv2"), se mostró el análisis de la complejidad temporal utilizando la metodología estudiada en AD2 (análisis asintótico de la instrucción crítica) y la metodología utilizada en CNU (número de FLOPS). Además, se utilizó la aplicación MATLAB, implementada en la clase anteriormente descrita de AD2, con el fin de comparar los costes temporales de ambas funciones.

IV. *Ejemplo de estudio de complejidad espacial.*

Para ejemplificar la utilidad del estudio de la complejidad espacial de algoritmos se utilizó un

problema numérico típico, como es el almacenamiento en memoria de una matriz dispersa, es decir, una matriz donde el número de elementos distintos de cero es muy inferior al número total de elementos. Seguidamente se muestran las diferentes etapas en las que se estructuró el ejemplo:

- *Ejemplos de matrices dispersas.* Se presentaron situaciones reales donde aparecen las matrices dispersas.

- *Estructuras de datos para almacenar la matriz dispersa.* Dada una matriz dispersa ejemplo (ver figura 4), se analiza el coste espacial que requiere su almacenamiento en memoria utilizando diferentes estructuras de datos: arrays bidimensiones, un vector de punteros a vectores filas y un vector de punteros a listas enlazadas (ver figura 5).

$$A = \begin{pmatrix} 6 & 0 & 1 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 7 & 0 & 0 & 3 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 5 & 0 & 9 \end{pmatrix}$$

Figura 4. Ejemplo de una matriz dispersa *A* (no real)

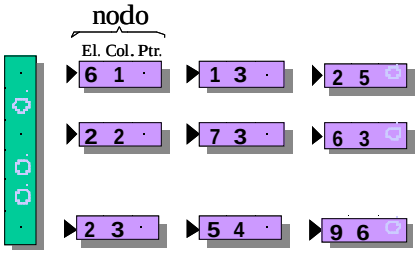

Figura 5. Representación de la matriz *A* mediante un vector de listas enlazadas. Cada nodo posee tres campos: el elemento representado (El.), el índice de columna (Col.) y el puntero al siguiente elemento (Ptr.)

6. Evaluación y videograbación

La evaluación de la experiencia no se basa únicamente en nuestra apreciación personal durante el transcurso de la clase; se ha realizado una encuesta a los alumnos de AD2, se ha realizado una vidoegrabación de una de las clases conjuntas (la clase conjunta de CNU) y en un futuro próximo se realizará una evaluación externa por parte del Instituto de Ciencias de la Educación de la UPV.

Consideramos importante destacar el alto nivel de asistencia observado en los grupos donde se ha impartido las clases conjuntas. Además, los niveles de participación de los alumnos durante la clase fueron satisfactorios, no sintiéndose coaccionados por la presencia en la clase de un profesor extraño.

La encuesta realizada sobre los alumnos de AD2 a reflejado, en general, que los alumnos se sintieron satisfechos con la experiencia y que valoraron el esfuerzo realizado por los profesores.

Durante el visionado del video se ha podido apreciar que la sincronización entre ambos profesores ha dotado a la clase de un tono distendido ya que un profesor podía interrumpir la actuación del otro con el fin de aclarar o remarcar algún concepto. Además, se ha podido observar la actuación propia y de otro profesor durante la impartición de una clase, así como el lenguaje no verbal de los alumnos.

7. Conclusión

Consideramos que la experiencia reflejada en este trabajo es innovadora en el campo de la docencia de la informática y que ha supuesto una mejora en la calidad de la misma, lo que nos motiva a ampliar el ámbito de aplicación de la misma a la totalidad de los alumnos de las materias de Programación y Computación Numérica en los próximos años.

Referencias

[1] A. Aho, J. Hopcroft, J. Ullman. *Estructuras de datos y algoritmos.* Ed. Addison Wesley Iberamericana, 1988.

[2] Paolo Rosso. *Clases dirigidas a grupos especiales de alumnos para la asignatura "Algoritmos y Estructuras de Datos II".* JENUI 2000, 2000.

[3] V. Vidal y J. Garayoa. *Introducción a la computación numérica.* Ed. Universidad Politécnica de Valencia, 2000.

[4] N. Wirth. *Algoritmos + Estructuras de datos = programas.* Ed. del Castillo. Madrid, 1980.