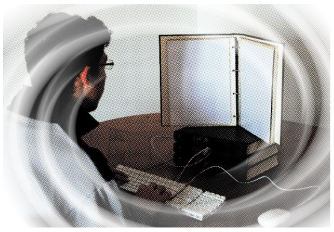

ReVisión

Artículo invitado

AENUI
Asociación de Enseñantes Universitarios de la Informática

Sobre la situación del paralelismo y la programación paralela en los Grados de Ingeniería Informática

Francisco Almeida,
Dept. de Estadística, Inv. Operativa y Computación
Universidad de La Laguna
falmeida@ull.es

Domingo Giménez
Dept. de Informática y Sistemas
Universidad de Murcia
domingo@um.es

José Miguel Mantas
Dept. de Lenguajes y Sist. Informáticos
Universidad de Granada
jmmantas@ugr.es

Antonio M. Vidal
Dept. de Sist. Informáticos y Computación
Univ. Politécnica de Valencia
avidal@dsic.upv.es

Resumen

La difusión creciente de los sistemas paralelos hace que la programación paralela haya adquirido gran relevancia, y dentro del proceso de reforma de los planes de estudio debería dársele la importancia que realmente tiene. Los ingenieros en informática trabajan con estos sistemas y tienen que desarrollar software para ellos, por lo que no puede aducirse que es un tipo de programación complejo para no incluirlo en los estudios o relegarlo de los mismos. Además, para obtener las máximas prestaciones de estos sistemas es necesario estudiar análisis y diseño de algoritmos paralelos.

Este trabajo propone una estructura general de la enseñanza de la programación paralela y de cómo incluir los contenidos de programación y algoritmos paralelos, y analiza la situación en los nuevos Grados de Ingeniería Informática, comparando la situación actual con la propuesta realizada.

Palabras clave: Programación paralela, Enseñanza de la programación paralela, Paralelismo en el Grado de Ingeniería Informática.

Recibido: 30 de marzo de 2010; **Aceptado:** 5 de mayo de 2010.

1. Introducción

Tradicionalmente, la programación paralela se ha venido utilizando para la resolución de problemas con alto coste computacional [7, 16] que estaban en la frontera de los problemas científicos abordables por sistemas computacionales. Esta programación era realizada por personal especializado, bien informáticos o científicos, que aprendían programación paralela para resolver los problemas en los que trabajaban. Para la resolución eficiente de estos problemas es necesaria la utilización de entornos paralelos y también la optimización o el rediseño de los algoritmos que se implementan para resolver los problemas. En cuanto a entornos de programación, se han generado herramientas que facilitan la programación en los distintos sistemas como Posix threads (Pthreads) [20], OpenMP (http://openmp.org/wp/), MPI [27], UPC [8] o CUDA [21].

En lo que se refiere a la importancia del estudio y diseño de algoritmos paralelos, basta con comprobar que la mayoría de los libros de introducción a la programación paralela dedican una parte importante al estudio de técnicas de programación y diseño de algoritmos [2, 11, 23, 28]. Es también común observar en los últimos tiempos que en libros de algorítmica secuencial se introducen algunos temas relacionados con la algorítmica y la programación paralela [6, 12, 13].

Más recientemente, con la interconexión de varios ordenadores para formar redes de procesadores, se ha conseguido que la programación paralela se popularice entre los grupos científicos que tienen en sus propios laboratorios un sistema computacional con el que resolver sus problemas de dimensión reducida y utilizan los supercomputadores para abordar problemas de mayor dimensión[1]. La aparición de estándares de programación paralela facilitó el desarrollo de programas portables que se podían utilizar tanto en las redes de ordenadores como en los supercomputadores.

En los últimos años la aparición de los procesadores multinúcleo ha puesto al alcance del público en general los sistemas paralelos [14]. Hoy en día es normal disponer de un portátil bi-

[1] Véase el Top 500 en http://www.top500.org/

procesador o de un ordenador de sobremesa con cuatro u ocho núcleos y la tendencia es que el número de núcleos aumente.

En la actualidad nos encontramos, por tanto, con la posibilidad de utilizar el paralelismo a varios niveles: supercomputadores, redes de ordenadores, sistemas distribuidos, procesadores multinúcleo, procesadores gráficos, vídeo consolas, *cloud computing*..., y debido a la multiplicidad y diversidad de los distintos sistemas, aparecen nuevas técnicas de gestión, virtualización y monitorización.

Esta popularización de los sistemas paralelos debería ir acompañada de una difusión similar de las técnicas de programación paralela, de forma que los desarrolladores naturales de aplicaciones informáticas para las plataformas de cómputo actuales y futuras (los ingenieros informáticos) estuvieran en condiciones de hacer un buen uso de los recursos computacionales a su alcance. Para esto es necesario introducir en los estudios de informática los conocimientos básicos de programación paralela, pero también del análisis y diseño de algoritmos paralelos.

Mientras que el estudio de técnicas algorítmicas secuenciales está más o menos unificado (en todos los planes de estudio se incluye, en un curso u otro y con mayor o menor profundidad, el análisis de algoritmos secuenciales y su diseño, ya sea con un enfoque basado en problemas o en paradigmas) no ocurre lo mismo con las técnicas algorítmicas paralelas. Teniendo en cuenta la importancia y la amplia difusión actual de los sistemas paralelos, que previsiblemente se irá ampliando más en el futuro, deberían incluirse como formación básica conceptos de algorítmica paralela en los planes de estudio de informática, y debería hacerse un esfuerzo por unificar posturas en cuanto a la docencia en este tema.

Este documento se estructura de la siguiente forma. En la Sección 2 se empieza analizando la situación tradicional de la enseñanza de la programación paralela y los algoritmos paralelos en los planes de estudio en España, y se compara la situación con la de otras universidades extranjeras, usando como referencia el trabajo de Meder *et al.* [18]. A continuación, en la Sección 3 se analizan distintas posibilidades de organización de la enseñanza de la programación paralela, revisando sus aspectos positivos y negativos. En la Sección 4, se estudia la planificación de las asignaturas relacionadas con la programación paralela en los recientes Grados de Ingeniería Informática, y se compara con la organización basada en los esquemas propuestos en este trabajo. Para este análisis se utilizan los planes de estudio del grado aprobados en distintas universidades, y la planificación que en algunas de ellas se está haciendo para la implantación de especialidades. En la Sección 5 se resumen las conclusiones y propuestas que se desprenden del análisis realizado.

El trabajo aquí presentado proviene de la colaboración continua que hemos tenido los autores durante los últimos años [2]. Trabajamos en universidades distintas y en investigación sobre programación paralela pero en campos variados: esquemas algorítmicos paralelos, entornos de programación paralela, computación heterogénea, álgebra lineal numérica paralela, aplicaciones de la programación paralela. Lo que aquí presentamos no es un enfoque personal, sino lo obtenido a partir de nuestra visión múltiple.

2. Organización tradicional de la enseñanza de la programación paralela y los algoritmos paralelos

La enseñanza de algoritmos paralelos requiere de conocimientos previos de sistemas paralelos y de programación paralela, incluyendo entornos y herramientas de programación. Estas materias se encuentran, en mayor o menor medida, en todos los libros de introducción a la programación paralela, desde los más clásicos a los más recientes [2, 3, 4, 9, 10, 11, 15, 17, 19, 22, 23, 24, 25, 26, 28].

Empecemos por analizar la organización tradicional de la enseñanza de las siguientes materias. Una división típica es la que se muestra a continuación.

- Sistemas computacionales paralelos, arquitecturas paralelas, paralelismo de bajo nivel.
- Nociones básicas de paralelismo, programación paralela, programación concurrente y programación en sistemas distribuidos.
- Análisis y diseño de algoritmos paralelos, esquemas algorítmicos paralelos. Metodología de la programación paralela.
- Lenguajes, entornos y herramientas de programación paralela. En particular las que se puedan considerar como estándares: Pthreads u OpenMP para procesamiento multihebra en entornos de memoria compartida, MPI para entornos de paso de mensajes, UPC para memoria virtual compartida o CUDA para procesadores gráficos.

No todos estos temas se tratan ni se deben tratar con la misma amplitud. Por ejemplo, las nociones básicas de programación paralela y las herramientas más básicas de programación se pueden estudiar conjuntamente, y el análisis y diseño de algoritmos deberían formar una unidad. Como en este trabajo nos centramos principalmente en la programación y algoritmos paralelos, hemos dividido estos temas en más apartados.

Además, la programación paralela se utiliza para resolver problemas científicos y de ingeniería de gran tramaño y complejidad. Sería, pues, conveniente que los alumnos trabajaran en la resolución de aplicaciones reales, aunque entendemos que un estudio sistemático de las técnicas utilizadas en la simulación y resolución de problemas científicos sólo es posible tras una formación amplia en una serie de conceptos (modelos científicos, métodos numéricos, optimización). Esto hace necesaria una formación más especializada, y por tanto un grado distinto, una nueva intensificación o especialidad, o un máster de computación científica.

Meder *et al.* [18] organizan los distintos temas relacionados con el paralelismo en: Algoritmos (Alg), Arquitectura (Arq), Programación (Pro), Computación Distribuida (Dis), Programación Multinúcleo (Mul), Computación Científica (Sci) y Teoría de la Computación Paralela (Teo). La división por apartados no coincide con la que utilizamos aquí, pero hay una correspondencia aproximada que se muestra el Cuadro 1.

En la Figura 1 se muestra un esquema (aproximado) de la distribución tradicional de los distintos temas en los planes de estudio de informática. A continuación comentamos la situación en cada uno de estos temas.

Arquitecturas paralelas. Desde los primeros cursos de las ingenierías informáticas se estudian temas de estructura, tecnología y arquitectura de ordenadores (llamaremos a estas asignaturas *ARQ1*). En estos cursos iniciales se incluyen temas de paralelismo intraprocesador y de bajo nivel (segmentación, encauzamiento, replicación de unidades, módulos de memoria, especulación). En el segundo ciclo hay una asignatura troncal de arquitectura de ordenadores (la llamamos *ARQ2*), con contenidos de arquitecturas paralelas. Adicionalmente, dependiendo de la universidad, puede haber alguna asignatura obligatoria más o una serie de asignaturas optativas que estudian temas de arquitecturas y sistemas paralelos (las llamamos *arq3*, en minúscula para indicar que corresponden a contenidos no obligatorios en todas las universidades): procesadores superescalares y vectoriales, multiprocesadores, VLSI, redes de interconexión...

Nociones básicas de programación paralela. En los planes de estudio suele haber una o dos asignaturas obligatorias con contenidos básicos de paralelismo y programación paralela en segundo o tercer curso (*CON1*), en ocasiones asociadas a materias de sistemas operativos. Estas asignaturas tienen normalmente los contenidos tradicionales de la programación concurrente (condición de carrera, abrazo mortal, semáforos, monitores) y algunos conceptos de herramientas de programación paralela, como pueden ser la programación con hilos, programación en espacio de direcciones compartido, llamadas remotas a procesos, o paso de mensajes, y también pueden contemplar contenidos más cercanos a la programación distribuida (objetos distribuidos, procesos y tecnología Java, programación web).

Suele haber otras asignaturas optativas (*con2*) relacionadas con temas de programación paralela, como pueden ser sistemas de tiempo real o programación distribuida.

Herramientas de programación paralela. Como hemos comentado, en CON1 se suele incluir el estudio de alguna herramienta para programación paralela, mayoritariamente trabajo con hilos, pero también algunas veces las herramientas que ofrecen lenguajes tradicionales para gestionar procesos. Llamamos a estos contenidos incluidos en asignaturas de primer ciclo *HER1*.

También en las asignaturas de segundo ciclo se suele incluir el estudio de herramientas, quizás en las prácticas de la asignatura. Este estudio no está tan generalizado como en las asignaturas básicas de programación paralela, y algunas veces se incluye en asignaturas optativas. Por esto llamamos a estos contenidos de segundo ciclo *her2*.

En algunos casos se incluyen, normalmente en segundo ciclo, asignaturas optativas de herramientas de programación paralela, o de programación en algún tipo de sistema particular (supercomputadores, paso de mensajes, memoria compartida, procesadores gráficos). Nos referiremos a estas asignaturas mediante *her3*.

Análisis y diseño de algoritmos paralelos. No es habitual encontrar contenidos obligatorios de análisis de algoritmos paralelos, aunque sí de algoritmos secuenciales. En *CON1* se puede tener en cuenta de manera intuitiva temas de rendimiento de los programas, pero no se suele realizar un estudio sistemático de este tema.

Es usual que en segundo ciclo se incluya alguna asignatura de programación paralela (*ana1*) donde se estudia el análisis de algoritmos paralelos, o que se tengan en cuenta aspectos de rendimiento de programas en asignaturas como her2 y her3 (nos referimos a estos contenidos estudiados en asignaturas no propiamente de programación paralela mediante *ana2*).

El diseño de algoritmos paralelos se puede estudiar junto con el análisis de los algoritmos (ana1). Llamamos a estos contenidos *dis1*. Pero también se puede estudiar metodología de programación paralela sin entrar directamente en el estudio sistemático de esquemas algorítmicos. En las asignaturas no dedicadas directamente a la programación paralela (her2 y her3) se puede abordar el diseño de programas paralelos desde este punto de vista, quizás analizando metodologías para sistemas particulares que se estudian en esas asignaturas. Llamamos a estos contenidos, más de metodología de la programación, *dis2*.

Como vemos, la situación en arquitecturas paralelas y en programación y algoritmos paralelos es totalmente distinta. En arquitectura, se incluyen en los distintos cursos y asignaturas referencias a paralelismo, que será intraprocesador y de bajo nivel en los primeros cursos (ARQ1), y de temas de estructura de sistemas paralelos en cursos superiores (ARQ2 y arq3). Sin embargo, la situación en programación no es la misma, encontrándonos con que normalmente se estudian las nociones básicas de programación, las estructuras de datos y los esquemas algorítmicos sin incluir en la mayoría de los casos referencias al paralelismo.

Además, el estudio del paralelismo se suele realizar dentro de asignaturas de arquitecturas de ordenadores o de sistemas

	Arq	Bás	Her	Alg	Apl	
Alg				+		Alg
Arc	+					Arq
Pro			+	-		Her
Dis	-	-	-			Bás
Mul	-		-			Arq + Her
Sci				-	+	Apl
The		+				Bás
	Arc	The	Pro	Alg	Sci	

Cuadro 1: Correspondencia entre la clasificación de los temas relacionados con el paralelismo utilizada en este artículo (por columnas) con las de Meder [18] (por filas). El símbolo + indica coincidencia mayor, y el − coincidencia menor. La última fila y columna indican los temas de la otra clasificación con los que mayoritariamente coincide cada uno de los temas de las dos clasificaciones.

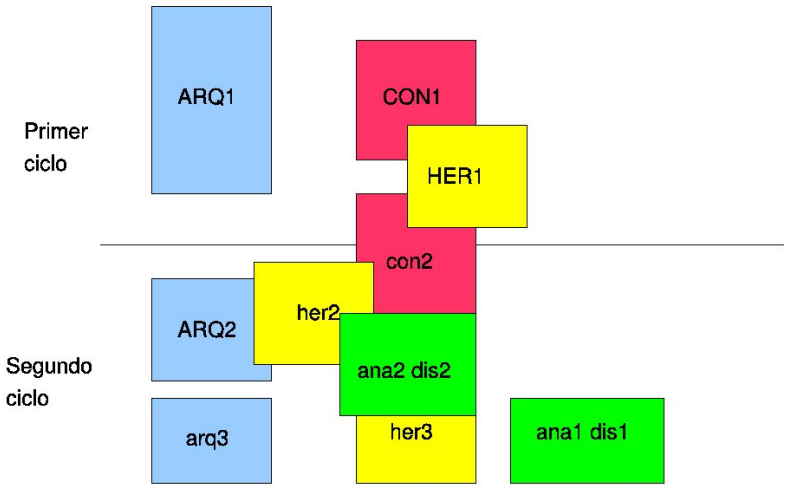

Figura 1: Distribución tradicional aproximada en los planes de estudio de informática de contenidos relacionados con la programación paralela

operativos, con lo que la visión de alto nivel propia de la algorítmica puede perderse en gran medida. También suele haber alguna asignatura específica de fundamentos de programación paralela o programación concurrente (CON1), pero aislada de las restantes asignaturas de programación y centrada en conceptos básicos, por lo que muchas veces tiene una orientación más de sistemas operativos que algorítmica.

La prolongación de la situación actual (en la que se estudian sólo nociones básicas de paralelismo y se hace de forma aislada o como parte de alguna asignatura de arquitecturas paralelas o sistemas operativos) creemos que daría lugar a formar titulados con conocimientos alejados de los necesarios para trabajar de forma adecuada en el desarrollo de aplicaciones para los sistemas computacionales actuales, por lo que se plantea la necesidad de unificar posturas en cuanto a los conocimientos de programación y algoritmos paralelos que deberían incluirse en el Grado de Ingeniería Informática. Por todo ello, en la siguiente sección se analizan distintas posibilidades.

3. Posibilidades de organización de la programación paralela y los algoritmos paralelos en el Grado de Ingeniería Informática

El análisis de los planes de estudio de la Ingeniería Informática nos lleva a formularnos la siguiente pregunta: ¿Está justificada en la situación actual la diferencia en la forma de enfocar el paralelismo en asignaturas de arquitectura y de programación? Desde nuestro punto de vista, teniendo en cuenta la amplia difusión actual de los sistemas paralelos, el enfoque de la docencia de la programación paralela se debería aproximar al enfoque tradicional en las asignaturas de arquitectura, incluyendo referencias al paralelismo en diferentes niveles de los estudios.

Con la difusión de los sistemas paralelos (que son en la actualidad los componentes básicos de todos los sistemas computacionales) la programación paralela no debe seguir considerándose elitista (como ha sido considerada con bastante frecuencia), y además el énfasis en el estudio del paralelismo no debe seguir centrado en las arquitecturas paralelas, sino que debe ampliarse al desarrollo de software. El conocimiento de las arquitecturas paralelas es conveniente para poder desarrollar aplicaciones paralelas eficientes, pero no es imprescindible su conocimiento para el estudio de herramientas, metodologías y algoritmos paralelos. Este estudio puede realizarse a un nivel más abstracto (usando modelos de arquitecturas paralelas) y al mismo tiempo que la arquitectura de ordenadores. Existen modelos simplificados de arquitecturas paralelas que permiten desarrollar, usando herramientas y técnicas de análisis/diseño estandarizadas, programas paralelos capaces de aprovechar de forma muy satisfactoria el potencial de la mayoría de arquitecturas paralelas más difundidas. De este modo, con el estudio diferenciado de arquitecturas, algoritmos y lenguajes, se lograría tener una visión general del paralelismo. Por todo ello, el aprendizaje de las competencias relacionadas con paralelismo y programación paralela debería ser obligatorio para un ingeniero informático, dadas las características de los actuales sistemas de cómputo.

Por otro lado, en el estudio de los conceptos básicos de programación paralela no se suele tener en cuenta la reducción del tiempo de ejecución, sino los aspectos semánticos relacionados con el paralelismo y los sistemas distribuidos. Sin embargo, con la programación paralela real lo que se pretende normalmente es reducir el tiempo de resolución de problemas de alto coste computacional, o resolver problemas en tiempo real, para lo que es necesario tener en cuenta el estudio del tiempo de ejecución. Creemos que deberían estudiarse los temas básicos de programación paralela con referencias a la eficiencia de los programas y algoritmos paralelos, tal como se hace con la programación secuencial, donde se suelen estudiar nociones intuitivas de análisis de rendimiento en los primeros cursos, para pasar a un estudio más sistemático del análisis y diseño de algoritmos en cursos sucesivos. La importancia de la programación y los algoritmos paralelos en el contexto general del paralelismo se observa en el mencionado libro de Meder [18], donde el porcentaje de cursos dedicados a programación y algoritmos paralelos es del 53 % (Algoritmos 21 %, Programación 20 % y Computación Científica 12 %), mientras que el de conceptos básicos de concurrencia y programación distribuida es del 31 % (Computación Distribuida 12 % y Teoría de la Computación Paralela 19 %).

Se discuten a continuación algunas posibilidades de organización del estudio de la programación paralela en los estudios de informática.

3.1. No incluir programación paralela

Se podría decidir no incluir programación paralela en el grado, si se considera que es un tema avanzado y se debe dejar para el postgrado.

Esta postura se justifica muchas veces indicando que la reducción en cursos y contenidos en el grado obliga a recortar conocimientos, por lo que sería imposible añadir conocimientos de paralelismo que no estaban incluidos en las anteriores titulaciones. Así, en el grado se estudiarían conceptos básicos (programación y algoritmos secuenciales), y los conceptos avanzados (programación paralela) se dejarían para cursos avanzados. Como ya hemos comentado, esta postura no nos parece adecuada pues los sistemas paralelos son los sistemas estándar actuales, por lo que la programación paralela no puede en ningún caso seguir considerándose como elitista, y por tanto los titulados de informática deberían tener conocimientos no sólo sobre las arquitecturas paralelas, sino también sobre su programación.

Es verdad que normalmente en algunas asignaturas de arquitectura se incluyen conceptos básicos de programación paralela, pero entendemos que el estudio de la programación debe realizarse con un enfoque abstracto, no centrado en la arquitectura sobre la que van a ejecutarse los programas. Esto,

que es válido y admitido para la programación secuencial, no siempre es admitido para la paralela, y dificulta su desarrollo.

Además, en la revisión del 13 de marzo de 2009 de las recomendaciones para los títulos de Ingeniería [5], se incluye una competencia sobre conceptos básicos de programación paralela entre las dieciocho competencias enumeradas para las materias comunes de la rama de informática («Conocimiento y aplicación de los principios fundamentales y técnicas básicas de la programación paralela, concurrente, distribuida y de tiempo real»). Por tanto, en ningún caso es posible dejar la programación paralela fuera de los estudios del grado, y creemos que tampoco sería conveniente relegarla dejándola como parte de asignaturas de arquitectura, sistemas operativos o programación secuencial.

Si consultamos el curriculum de ACM de 2008 [1], encontramos los algoritmos paralelos dentro del *Body of Knowledge*, aunque sin considerarlo como *core*. Sin embargo, se comenta que «*With the emergence of multicore processors, issues related to parallel algorithms have become more relevant. While the parallelism topics remain listed here as elective, the committee believes that the role of parallelism throughout the curriculum needs to be considered*». Además, hay que tener en cuenta que este curriculum de 2008 es una revisión mínima del de 2004, que no se realizan en él cambios importantes (que deberían hacerse por ejemplo en los contenidos de algoritmos paralelos, que están claramente obsoletos), y que desde 2008, en que el comité realiza la recomendación, hasta la actualidad han transcurrido dos años, en los que los sistemas de computación estándar han pasado a ser paralelos.

Hay que tener en cuenta también que si los estudiantes del grado van a ser los futuros profesionales de la Informática, se estarían lanzando al mercado profesionales sin formación en competencias básicas relacionadas con la programación de los computadores del mercado actual.

3.2. Asignaturas de programación paralela

Se puede decidir incluir programación paralela en el grado, pero entendiendo que es un tipo de programación diferenciado, por lo que debe haber al menos una asignatura de fundamentos básicos de paralelismo y concurrencia.

Esta es la situación actual en muchos grados e ingenierías, donde se incluye una asignatura de fundamentos básicos de la programación concurrente o de sistemas distribuidos, y complementariamente se pueden incluir en asignaturas de arquitectura conceptos de programación paralela. La ventaja de este enfoque es que el profesorado de estas asignaturas de fundamentos se especializaría en programación paralela, frente al hecho que impartir los conceptos propios de programación paralela dentro de asignaturas de arquitectura o programación secuencial podría ocasionar que el profesorado no estuviera especializado en programación paralela y diera a su enseñanza un enfoque de más bajo nivel o lo incluyera como un apéndice del contenido general del curso.

Si este enfoque de asignaturas diferenciadas podía ser adecuado hace unos años, cuando el paralelismo no estaba tan ampliamente difundido, no está tan claro que sea una buena opción para el futuro, pues seguiría diferenciando la programación paralela de la secuencial (que efectivamente son diferentes) situándolas a un nivel distinto de importancia (no estando claro que lo estén).

3.3. Integración en otras asignaturas

Se puede considerar incluir programación paralela en el grado, pues todos los computadores actuales son paralelos, tener en cuenta los diferentes aspectos del paralelismo (arquitecturas, herramientas, programación, algoritmos...) y decidir qué aspectos de paralelismo deben incluirse en otras asignaturas.

Quizás esta sería la opción más adecuada, dada la omnipresencia (actual y prevista para el futuro) de los sistemas paralelos, pero por los motivos mencionados en el apartado anterior (falta de especialización del profesorado y seguridad de impartición de los temas de programación paralela que podrían verse relegados al final de las asignaturas), puede que cambiar a un enfoque de este tipo en el momento actual no sea lo más adecuado. Está claro que esta postura se puede compaginar con la anterior: puede ser conveniente incluir alguna asignatura específica de fundamentos de programación paralela e incluir nociones de programación paralela en las asignaturas de arquitectura, pero no se dejaría el peso de la programación en estas asignaturas, que tratan el paralelismo a más bajo nivel.

La organización en este caso se presenta en la Figura 2, donde se muestra una división por asignaturas comunes que contendrían los conocimientos básicos de paralelismo, y por contenidos que se podrían incluir en las distintas especialidades. Comentamos los aspectos generales de la organización de la parte común. Se incluirían conocimientos obligatorios de paralelismo a todos los niveles: arquitecturas (ARQ), programación y herramientas paralelas (CON + HER) y análisis y diseño de algoritmos paralelos (ANA + DIS). En algunas de las especialidades deberían incluirse contenidos de paralelismo que complementaran los conocimientos obligatorios y tuvieran relación con las competencias de la especialidad (administración de sistemas paralelos, virtualización, evaluación de software y hardware paralelo). En los postgrados estarían la mayoría de las asignaturas optativas, aunque podrían incluirse algunas en las especialidades correspondientes de los grados.

El estudio de las nociones básicas de programación paralela y de herramientas de programación paralela se contempla como un todo (CON + HER) y el análisis y diseño de algoritmos paralelos se abordaría de forma conjunta (ANA + DIS). Esto no necesariamente se hace de una manera diferenciada, sino que puede ir dentro de CON + HER y de asignaturas propias de programación secuencial PRO. Discutimos brevemente los contenidos de cada uno de los puntos.

Arquitecturas paralelas. Entendemos que la estructura actual de la enseñanza de estructura de ordenadores y arquitecturas paralelas es la adecuada, por lo que se in-

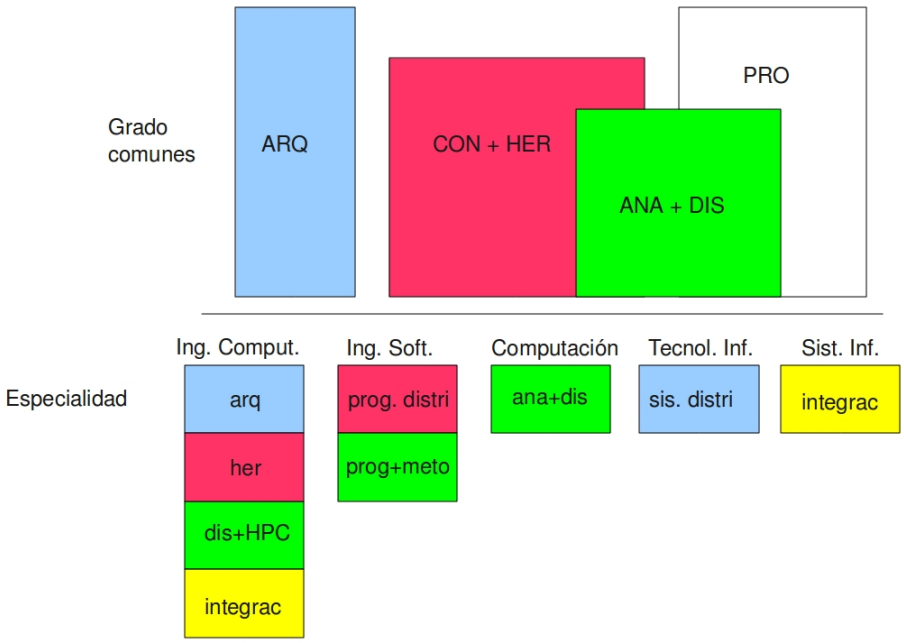

Figura 2: Propuesta de organización de las enseñanzas de paralelismo en el Grado de Ingeniería Informática

cluye un módulo ARQ que abarca todos los cursos del grado. Estas asignaturas contendrían temas de programación paralela sólo para ejemplificar el uso de las arquitecturas paralelas que se estudien, pero el estudio de la programación y los algoritmos paralelos se dejaría a asignaturas propias de programación.

Programación concurrente, herramientas de programación. Algunas nociones de programación paralela se pueden incluir en otras asignaturas. Por ejemplo, el estudio de *threads* y algunas nociones básicas de su programación puede incluirse en asignaturas de sistemas operativos o arquitectura. Y al utilizar lenguajes que incluyen opciones de paralelización se puede indicar la existencia de estas opciones y mostrar su utilización (por ejemplo en C la gestión de procesos, o la opción de compilación `-openmp`, y en Java la gestión de la concurrencia). Pero, en cualquier caso, este tema debería incluirse en una o dos asignaturas especializadas que aseguraran los conocimientos comunes y trasversales para todas las especialidades. Estas asignaturas estarían dedicadas a fundamentos de la programación paralela, programación concurrente, distribuida, de tiempo real..., y no se incluirían en el primer curso.

Análisis y diseño de algoritmos paralelos. No creemos necesario en el grado la inclusión de una asignatura obligatoria específica para estos temas diferenciada de las asignaturas de fundamentos de programación paralela, concurrente y distribuida. No obstante, si se considerara conveniente incluir más asignaturas relacionadas con la programación paralela, alguna de ellas podría estar dedicada a supercomputación y a la reducción del tiempo de solución de problemas de alto coste, y sería esta asignatura la que incluiría estos contenidos.

Lo que sí creemos necesario es el estudio de algoritmos paralelos con un nivel de abstracción similar al que se utiliza en la actualidad para abordar el estudio de algoritmos secuenciales. Se deberían incluir temas de análisis y diseño de algoritmos paralelos en las asignaturas propias de programación, tanto las de programación secuencial (PRO) como las de programación concurrente y distribuida (CON + HER). Esta visión se ve reforzada por la inclusión de capítulos de paralelismo en libros clásicos y recientes de algoritmos [6, 12, 13]. Cuando se describe el modelo de máquina secuencial se puede incluir una descripción de uno o varios modelos de máquina paralela como extensión natural del modelo secuencial. Al estudiar el análisis de algoritmos secuenciales se puede ver de forma intuitiva la reducción en el tiempo que se puede conseguir con una versión paralela del algoritmo, nociones básicas de análisis de algoritmos paralelos y las dificultades de obtener las máximas prestaciones. Se podría incluir algún esquema paralelo adicional a los secuenciales. Las herramientas de programación se incluirían en CON + HER pero, como hemos mencionado, algunas que sean parte de un lenguaje

secuencial se pueden estudiar junto con éste.

3.4. Paralelismo en las especialidades del Grado de Ingeniería Informática

En cualquiera de los tres supuestos anteriores (no incluimos como obligatorios conocimientos de programación paralela sea de forma diferenciada o no diferenciada) hay que considerar la organización de la enseñanza del paralelismo y la programación paralela en las diferentes especialidades: Ingeniería de Computadores, Computación, Ingeniería del Software, Sistemas de Información y Tecnologías de la Información. Cómo se organicen dependerá mucho del entorno de cada universidad (profesorado, recursos, especialidades impartidas), pero analizamos una propuesta para el caso ideal en que se implanten las cinco especialidades, se disponga de profesorado suficiente y preparado y los contenidos a impartir prevalezcan sobre el reparto de créditos.

Nuestra propuesta se resume en el apartado de especialidades de la Figura 2 y analizamos la situación por especialidades. Consideramos las cinco propuestas para el Grado de Ingeniería Informática y tenemos en cuenta las actuales competencias relacionadas con paralelismo que han sido enumeradas para cada especialidad, aunque a juicio de los autores este catálogo de competencias podría ser ampliado.

Sistemas de información. De las cinco especialidades, esta sería quizás la que menos contenidos de paralelismo necesitaría, aunque en el caso de no haberse incluido nociones de programación paralela en la parte común deberían incluirse algunas nociones básicas aquí. Esta especialidad está orientada a la integración de soluciones TIC en las organizaciones. Para ello consideramos que es necesario tener conocimiento de las tecnologías existentes, y entre ellas las de programación paralela. No obstante, no es necesario que este conocimiento sea profundo, pues los especialistas en sistemas de integración no dominarán todas las tecnologías que es posible integrar en una organización, pero sí deberían tener un conocimiento integrado (bloque *integrac* en la Figura 2) de manera que pudieran evaluar todos los aspectos a tener en cuenta a la hora de evaluar necesidades de un centro de computación: refrigeración, suministro eléctrico, seguridad, tipos de sistemas y redes, aplicaciones software paralelas, entornos de programación paralelos apropiados para distintos tipos de aplicaciones...

Tecnologías de la información. Estos especialistas tendrán capacidad para *seleccionar, diseñar, desplegar, integrar, evaluar, construir, gestionar, explotar y mantener las tecnologías de hardware, software y redes*. Por lo tanto, consideramos necesario que tengan los conocimientos básicos de paralelismo que indicamos en la parte común de la Figura 2 y que profundicen en la especialidad algo más dentro de la vertiente de sistemas distribuidos (*Sis. distri*), especialmente en su configuración pero también en menor medida en la programación en estos sistemas.

Computación. Dentro de las competencias hay una que corresponde al análisis y diseño de algoritmos (*evaluar la complejidad computacional de un problema, conocer estrategias algorítmicas que puedan conducir a su resolución*) y entendemos que estos especialistas deben ser capaces de desarrollar esta capacidad para los sistemas computacionales estándar en la actualidad (paralelos), por lo que se propone la intensificación en este sentido (ana+dis). Se puede aducir que con los conceptos básicos de paralelismo que se incluyan en la parte común se ha adquirido esta capacidad, pero esto no creemos que sea cierto pues supondría que todos los especialistas en cada una de las intensificaciones la adquirirían. Dependiendo de cuánto se haya profundizado en el estudio de algoritmos (tanto secuenciales como paralelos) en la parte común, el bloque ana+dis correspondería únicamente a contenidos de programación paralela o incluiría también conceptos algorítmicos más avanzados que los estudiados en los primeros cursos.

Ingeniería del software. Para poder *desarrollar o implementar soluciones software sobre modelos y técnicas actuales* se hace imprescindible tener conocimientos de programación en una gama amplia de sistemas actuales, por lo que será necesario adquirir conocimientos de tecnologías de programación en entornos distribuidos y web (Prog. distri) y también los básicos de herramientas y metodología de desarrollo de software paralelo (prog+meto), de manera que se tenga capacidad para abordar soluciones en una gama amplia de sistemas (multicore, clusters, procesadores gráficos, supercomputadores). Los bloques prog+meto y ana+dis coincidirían en parte, pero en el primero se haría más énfasis en la metodología y en el segundo en el estudio algorítmico. En cualquier caso, si no se hubiera profundizado lo suficiente en el análisis de algoritmos paralelos dentro de los contenidos comunes, sería necesario completar ese estudio aquí, ya que es la base para poder cuantificar la calidad de un diseño.

Ingeniería de computadores. Esta especialidad está centrada en la arquitectura de los computadores, incluyendo *plataformas paralelas y distribuidas*, pero también se contempla *desarrollar y optimizar software* para ellas. Por tanto habrá asignaturas de arquitectura de ordenadores, sistemas distribuidos y, en particular, de arquitecturas paralelas (arq). Se puede incluir el estudio de arquitecturas específicas (multicore, procesadores gráficos) y las herramientas de programación correspondientes (her). Se considera también que se debería incluir conocimientos de técnicas generales de programación y algoritmos paralelos similares a los de ana+dis y prog+meto, pero en este caso se podría dar un enfoque más de computación de altas prestaciones (dis+HPC), teniendo en cuenta que en esta especialidad el estudio de la programación paralela se considera como un medio, no como un fin. Además, consideramos necesario

un bloque de integración de conocimientos (integrac), similar al propuesto para la especialidad de Sistemas de Información, pero en este caso no sólo enfocado a la evaluación de los sistemas, sino también al desarrollo integrado de todos los conocimientos adquiridos en los restantes bloques que se proponen.

Como mencionamos con anterioridad, con la planificación actual de las especialidades no creemos que se pueda formar a los ingenieros informáticos en la resolución de problemas científicos de gran complejidad, de manera que se facilite la interacción de los ingenieros informáticos con especialistas de diversos campos científicos y de ingeniería, pues resulta evidente que ninguna de las cinco especialidades puede satisfacer esta necesidad.

Por un lado, sólo Ingeniería de Computadores incluye de manera explícita la programación paralela en su lista de competencias, pero esta intensificación está orientada al diseño y uso de sistemas paralelos, y no a la computación, por lo que no incluye otros conocimientos necesarios para abordar la resolución de problemas científicos (simulación, modelado, optimización, métodos numéricos, algorítmica)

Por otro lado, quizás la especialidad más cercana a la computación científica debería ser la de Computación, pero de su lista de competencias no está muy clara la formación que se pretende dar en ella. Se combinan competencias de sistemas inteligentes, conocimiento, interacción persona-computador y aprendizaje con lenguajes de programación y otras de computación y algoritmos, con lo que parece que esta especialidad debería combinar la formación en fundamentos de la computación y en inteligencia artificial. Para abordar la solución de problemas científicos de gran complejidad faltan algunos de los conocimientos mencionados en el párrafo anterior, y en particular los de computación de altas prestaciones.

Con esta situación, parece difícil que la formación de informáticos especializados en Computación Científica se pueda realizar en el marco actual de los grados de Ingeniería Informática, por lo que puede ser conveniente planificar grados o másteres con esta orientación, definir una nueva especialidad o redefinir alguna de las actuales. Quizás el lugar más adecuado para planificar la formación en computación científica sea en los másteres. Esta posición se refrenda con el análisis del curriculum de ACM [1] y las directrices del máster de informática [5]. Así, en el curriculum de ACM se indica que la computación científica es una disciplina importante y que ha crecido en importancia en los últimos tiempos, pero no se considera dentro del núcleo de la informática. Y en las directrices del máster, de entre las doce competencias establecidas para los másteres de informática dentro del módulo de tecnologías informáticas, una corresponde a la computación de altas prestaciones y métodos numéricos [5] (*Capacidad para comprender y poder aplicar conocimientos avanzados de computación de altas prestaciones y métodos numéricos o computacionales a problemas de ingeniería*).

4. Conclusiones

En este trabajo se analiza la situación de la enseñanza de la programación y los algoritmos paralelos en los planes de estudio de Informática. Debido a la difusión creciente de los sistemas paralelos y al proceso de reforma de los planes de estudio, creemos que es el momento adecuado para replantearse la organización de la enseñanza de estos temas.

Los ingenieros en informática son los profesionales que trabajan con estos sistemas y que tienen que desarrollar software para sistemas paralelos, por lo que en nuestra opinión no puede aducirse que es un tipo de programación complejo para no incluirlo o relegarlo de los estudios. Para poder aprovechar adecuadamente el potencial de estos sistemas, es necesario estudiar técnicas de análisis y diseño de algoritmos paralelos, y deberían estudiarse con un nivel de abstracción similar al que se utiliza en la programación secuencial.

Se pretende motivar la discusión sobre cómo incluir contenidos de programación y algoritmos paralelos en el grado de informática. Se da una visión de cómo incluir estos contenidos en la parte común del grado y en las distintas especialidades. En nuestra opinión, una buena opción sería incluir en la parte común conocimientos básicos de programación paralela como parte de las asignaturas de programación, aunque con alguna asignatura específica de conceptos básicos de concurrencia y programación paralela, que se complementarían en las especialidades con conocimientos de paralelismo adecuados para las competencias que deben adquirirse en cada una de ellas.

Sería interesante realizar un estudio de la situación de la programación paralela en el grado y las distintas especialidades una vez planificados los grados en las universidades españolas, y comparar la situación general con las propuestas que se hacen en este trabajo.

Agradecimientos

Este trabajo ha sido financiado en parte por la Fundación Séneca, proyecto 08763/PI/08, y por el Ministerio de Educación, proyecto TIN2008-06570-C04.

Referencias

[1] ACM: Curricula recommendations, `http://www.acm.org/education/curricula-recommendations`.

[2] Almeida, Francisco; Giménez, Domingo; Mantas, José Miguel y Vidal, Antonio M.: *Introducción a la programación paralela*. Paraninfo Cengage Learning, 2008.

[3] Bertsekas, Dimitri P. y Tsitsiklis, John N.: *Parallel and Distributed Computation*. Prentice Hall, 1989.

[4] Bisseling, Rob H.: *Parallel Scientific Computation*. Oxford University Press, 2004.

[5] BOE: Recomendaciones para la propuesta por las universidades de memorias de solicitud de títulos oficiales en los ámbitos de la Ingeniería Informática, Número 187, 4 agosto, 2009.

[6] Brasard, Gilles y Bratley, Paul: *Fundamentos de algoritmia*. Prentice-Hall, 1997.

[7] Dongarra, Jack; Foster, Ian; Fox, Geoffrey; Gropp, William; Kennedy, Ken; Torczon, Linda y White, Andy (Eds.): *Sourcebook of Parallel Computing*. Morgan Kaufmann Publishers, 2003.

[8] El-Ghazawi, Tarek; Carlson, William; Sterling, Thomas y Yelick, Katherine: *UPC: Distributed Shared Memory Programming*. John Wiley & Sons, 2005.

[9] Foster Ian T.: *Designing and Building Parallel Programs*. Addison-Wesley, 1995.

[10] Freeman T. L. y Phillips C.: *Parallel Numerical Algorithms*. Prentice Hall, 1992.

[11] Grama, A.; Gupta, A.; Karypis, G. y Kumar, V.: *Introduction to Parallel Computing*. Addison-Wesley, second edition, 2003.

[12] Harel, David y Feldman, Yishai: *Algorithms. The Spirit of Computing*. Addison-Wesley, third edition, 2004.

[13] Horowitz, Ellis; Shani, Sartaj y Rajasekaran Sangthevar: *Computer Algorithms/C++*. Computer Science Press, 1997.

[14] Hughes, Cameron y Hughes, Tracey: *Professional Multicore Programming: Design and Implementation for C++ Developers*. Wrox, 2008.

[15] Karniadakis G.E. y Kirby, R.M.: *Parallel Scientific Computing in C++ and MPI*. Cambridge University Press, 2003.

[16] Koniges, Alice E.: *Industrial strength parallel computing*. Morgan Kaufman, 2000.

[17] Lester, Bruce P.: *The art of Parallel Programming*. Prentice Hall, 1993.

[18] Meder, David J.; Pankratius, Victor y Tichy, Walter F.: `http://www.multicore-systems.org/separs/downloads/gi-wg-surveyparallelismcurricula.pdf`, diciembre 2009. Consultado en febrero 2010.

[19] Modi, Jagdish J.: *Parallel Algorithms and Matrix Computation*. Oxford University Press, 1988.

[20] Nichols, Bradford; Buttlar, Dick y Farrel, Jacqueline Proulx: *Pthreads programming: A Posix Standard for Better Multiprocessing*. O'Reilly, 1996.

[21] Nickolls, John; Buck, Ian; Garland, Michael y Skadron, Kevin: Scalable parallel programming with CUDA. *Queue*, 6(2):40–53, 2008.

[22] Parhami, B.: *Introduction to Parallel Processing*. Kluwer Academic Publishers, 2002.

[23] Quinn, Michael J.: *Parallel Programming in C with MPI and OpenMP*. McGraw Hill, 2004.

[24] Roosta, Seyed H.: *Parallel Processing and Parallel Algorithms*. Springer, 2000.

[25] Scott, L.R.; Clark, T. y Bagheri, B.: *Scientific Parallel Computing*. Princeton University Press, 2005.

[26] Shonkwiler, R.W. y Lefton, L.: *An introduction to Parallel and Vector Scientific Computing*. Cambridge University Press, 2006.

[27] Snir, Marc y Gropp, William: *MPI. The Complete Reference. 2nd edition*. The MIT Press, 1998.

[28] Wilkinson, Barry y Allen, Michael: *Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers*. Prentice-Hall, second edition, 2005.

Francisco Almeida es Profesor Titular en el área de Lenguajes y Sistemas Informáticos en la Universidad de La Laguna, Tenerife. Sus líneas de investigación han estado asociadas principalmente a las áreas de la computación en paralelo: algoritmos paralelos para problemas de optimización, análisis y predicción del rendimiento de sistemas paralelos y herramientas para la programación paralela. Ha estado implicado desde el año 1990 en la docencia de asignaturas relacionadas con la programación en los títulos de Informática, y desde el año 1994 ha impartido docencia sobre programación paralela en asignaturas de segundo y tercer ciclo.

Domingo Giménez es licenciado en Matemáticas por la Universidad de Murcia (1981) y doctor en Informática por la Universidad Politécnica de Valencia (1995). Actualmente es Titular de Universidad del área de Lenguajes y Sistemas Informáticos en el Departamento de Informática y Sistemas de la Universidad de Murcia. Es director del Grupo de Computación Científica y Programación Paralela de la Universidad de Murcia. Ha sido director de departamento, secretario de la Facultad de Informática y coordinador del Máster Informática y Matemáticas Aplicadas en Ciencias e Ingeniería en la Universidad de Murcia. Imparte docen-

cia de Algoritmos y Programación Paralela y de Computación de Altas Prestaciones. Ha dirigido cuatro tesis doctorales y ha publicado en actas de congresos y revistas internacionales artículos sobre computación paralela, computación científica y docencia de la informática.

José M. Mantas Ruiz es Licenciado en Informática por la Universidad de Granada (1994) y doctor en Informática por la Universidad de Granada (2003). Actualmente es profesor del Departamento de Lenguajes y Sistemas Informáticos de la Universidad de Granada, donde imparte diversas asignaturas de grado y postgrado relacionadas con la programación paralela, programación concurrente y los sistemas operativos. Ha participado en numerosos proyectos de investigación relacionados con computación científica, procesamiento paralelo, informática gráfica e inteligencia artificial. Ha publicado numerosos artículos relacionados con la programación paralela en actas de congresos y revistas internacionales.

Antonio M. Vidal es licenciado en Ciencias Físicas por la Universidad de Valencia (1972) y doctor en Informática por la Universidad Politécnica de Valencia (1990). Actualmente es Catedrático de Universidad del área de Ciencias de la Computación e Inteligencia Artificial en el Departamento de Sistemas Informáticos y Computación de la Universidad Politécnica de Valencia. Es director del Grupo Interdisciplinar de Computación y Comunicaciones de este Departamento e investigador principal del proyecto Computación de Altas Prestaciones sobre Arquitecturas Actuales en Problemas de Procesado Múltiple de Señal (PROMETEO 2009/013) financiado en el marco del Programa PROMETEO para Grupos de Investigación de Excelencia de la Generalitat Valenciana. Ha sido coordinador del programa de doctorado Computación Paralela y Distribuida y director del Master Computación Paralela y Distribuida de la Universidad Politécnica de Valencia. Ha dirigido varias tesis doctorales y ha publicado numerosos artículos relacionados con la Computación Paralela en actas de congresos y revistas internacionales.