Actas de las XXII Jenui. Almería, 6-8 de julio 2016
ISBN: 978-84-16642-30-4
Páginas: 19-26

Un curso práctico de Programación Paralela basado en problemas de Concurso Español de Programación Paralela

Domingo Giménez
Departamento de Informática y Sistemas
Universidad de Murcia
30071 Murcia
domingo@um.es

Resumen

En este trabajo se presenta una experiencia de un curso de Introducción a la Programación Paralela. El curso está dedicado a herramientas y entornos de programación paralela, y principalmente al análisis, desarrollo y optimización de algoritmos paralelos. Tiene una orientación práctica, para lo que se utilizan problemas del Concurso Español de Programación Paralela. Los diferentes temas del curso se presentan en clases con la estructura tradicional, y para cada tema se organiza una sesión práctica, en la que se trabaja con problemas del concurso con los que se practica con los entornos de paralelismo y los paradigmas algorítmicos tratados en la clase anterior. En las sesiones prácticas los alumnos trabajan con problemas y con el entorno computacional del concurso, lo que facilita el seguimiento y la validación en tiempo real de su trabajo, así como la atención personalizada y el tomar acciones correctoras de la organización de la docencia y las prácticas. De esta forma se realiza una evaluación continua que ha permitido reducir la tasa de abandono e incrementar la tasa de éxito en la asignatura.

Abstract

This paper presents an experience of an introductory course on Parallel Programming. The course is dedicated to parallel programming tools and environments, and in particular to the analysis, development and optimization of parallel algorithms. It has a practical orientation and is guided with the use of problems from the Spanish Parallel Programming Contest. The different units are presented in the traditional lecture format, and a practical session accompanies each unit, with problems to work with in the tools or algorithmic paradigms presented in the previous lecture. The students work in the practical sessions on problems and using the system of the contest, which facilitates online and real time validation of their implementations. The practical approach of the course and the continuous evaluation used led to an important increase in the marks.

Palabras clave

Enseñanza de la programación paralela, docencia centrada en las prácticas, aprendizaje basado en problemas, concurso de programación paralela.

1. Introducción

En la actualidad los sistemas computacionales son paralelos, con varios núcleos en sistemas pequeños (móviles, tarjetas dedicadas, etc.), portátiles y ordenadores de sobremesa, y redes y grandes sistemas formados combinando varios sistemas básicos. Por este motivo, se están desarrollando en los últimos años proyectos de introducción del paralelismo a diferentes niveles [8]. El paralelismo estará presente en alguna medida en la vida laboral de todos los trabajadores en informática, por lo que es obligatorio iniciar a nuestros alumnos en este campo, y se están llevando a cabo proyectos en este sentido [6]. Pero hay algunas dificultades importantes para realizar esa inclusión. La mayoría de los docentes no son especialistas en paralelismo, o tienen únicamente un conocimiento vago de este campo, y en algunos casos ni siquiera son conscientes de su relevancia. Esto entorpece la introducción de nuevas asignaturas relacionadas con el paralelismo o de conceptos de paralelismo en asignaturas ya establecidas. El retraso de la inclusión de conceptos de paralelismo debe evitarse, de manera que podamos preparar a nuestros estudiantes actuales para el mundo paralelo en el que trabajarán al finalizar sus estudios. Además, el paralelismo supone una mayor complejidad en diferentes aspectos de la computación, incluyendo arquitecturas, sistemas y programación. Así, iniciar a todos los estudiantes en conceptos de paralelismo de forma que tengan un conocimiento suficiente del tema como para trabajar con él no es una tarea sencilla. Por tanto, la intro-

ducción de conceptos de paralelismo debe planificarse minuciosamente, con determinación de los conceptos a incluir junto con prácticas adecuadas para facilitar el aprendizaje activo que permita adquirir habilidades para desarrollar distintos tipos de tareas dentro de este campo.

En este artículo se presenta una experiencia de un curso de "Metodología de la Programación Paralela" [3] centrado en entornos y herramientas de programación paralela (OpenMP, MPI y CUDA) y especialmente en el diseño, análisis y desarrollo de algoritmos paralelos. El curso se organiza en sesiones teóricas en el aula y sesiones prácticas, con una sesión práctica por cada sesión de teoría sobre herramientas y esquemas algorítmicos. Cada sesión práctica se lleva a cabo la semana siguiente a la correspondiente sesión teórica, y en ella se usan los recursos del Concurso Español de Programación Paralela (`http://luna.inf.um.es`), en el que se usa una adaptación a un *cluster* de procesadores del sistema Mooshak [4]. Con este entorno los alumnos pueden trabajar *online* y realizar una validación y evaluación parcial de su trabajo en tiempo real. Además, el profesor puede monitorizar el trabajo de los alumnos, facilitándose así la revisión, evaluación y corrección de sus trabajos, con lo que puede detectar errores y malentendidos rápidamente, redirigiendo la docencia y realizando una evaluación continua.

La estructura del artículo es la siguiente: en la Sección 2 se explica el contexto del curso y se comentan experiencias previas; en la Sección 3 se detalla el temario y la organización del curso; y en la Sección 4 se evalúa la experiencia; las conclusiones se resumen en la Sección 5.

2. Contexto y experiencias previas

La asignatura de "Metodología de la Programación Paralela" se imparte en el primer cuatrimestre en el cuarto curso del Grado de Ingeniería Informática en la Universidad de Murcia. Este cuarto curso se dedica a asignaturas de intensificación y al Proyecto Fin de Grado, y los estudiantes optan por una de las cinco intensificaciones (Computación, Ingeniería de Computadores, Sistemas de Información, Tecnologías de la Información e Ingeniería del Software). La asignatura es obligatoria para los alumnos en la intensificación de Ingeniería del Software y optativa para los de Ingeniería de Computadores.

Con los nuevos grados el número de años se ha reducido de cinco a cuatro, y en la organización anterior la asignatura equivalente se encontraba en el quinto curso y era optativa para todos los estudiantes. El cambio ha supuesto un incremento en el número de alumnos en la asignatura, y además los alumnos están ahora en general menos maduros y menos interesados por ella. Esto ha producido una reducción en el porcentaje de alumnos que aprueban la asignatura, por lo que hay que planificarla cuidadosamente para atraer su interés y evitar el abandono. La evolución del número de matriculados en la asignatura en los últimos trece años se muestra en el Cuadro 1, junto con el número y el porcentaje de alumnos que la superan. Con la nueva organización del Grado hay un incremento en el número de alumnos y una reducción en el porcentaje de alumnos que aprueban. Los valores del curso actual son estimados, pues las sesiones de prácticas presenciales han finalizado pero los alumnos tienen que realizar una práctica final con una valoración de 2.5 en la nota final. El trabajo en esta práctica es individual y autónomo. La media del porcentaje de alumnos que aprueban bajó con la nueva organización de 65 % a 45 % aproximadamente, y este curso, con la nueva planificación, se ha conseguido elevar este porcentaje al nivel anterior.

En los primeros cursos del grado hay una asignatura de "Programación Concurrente y Distribuida" y otra de "Arquitectura y Organización de Computadores" donde se introducen las nociones básicas de arquitecturas paralelas. Además, hay otra asignatura de "Programación de Arquitecturas Multinúcleo" en el segundo cuatrimestre de cuarto que es optativa en la intensificación de Ingeniería de Computadores. Nuestra asignatura se centra en programación y algoritmos paralelos con los paradigmas de memoria compartida y paso de mensajes, y se introducen algunos conceptos de GPUs y computación heterogénea.

3. Organización de la asignatura

La asignatura se ha basado siempre mayoritariamente en trabajos prácticos con trabajo autónomo de los estudiantes. Se organizaban algunas prácticas presenciales en las que los alumnos podían colaborar entre ellos y consultar al profesor. La mitad de la puntuación total se obtenía con prácticas básicas que se realizaban en esas sesiones y se completaban después y con algunos trabajos adicionales, en algunos casos trabajando en grupos de dos alumnos. La otra mitad de la puntuación correspondía a una práctica en forma de proyecto con un trabajo distinto para cada estudiante, que trabajaba de forma autónoma con el asesoramiento del profesor en tutorías y en las sesiones de prácticas. De esta forma los alumno adquirían la capacidad de desarrollar programas paralelos en diferentes entornos y el porcentaje de aprobados estaba alrededor del 65 %, lo que podemos considerar satisfactorio teniendo en cuenta que muchos de los alumnos de los últimos cursos trabajan a tiempo parcial o completo y por tanto son alumnos a tiempo parcial.

Las prácticas presenciales iniciales se organizaban

Curso	03	04	05	06	07	08	09	10	11	12	13	14	15
alumnos	10	8	13	18	10	4	13	12	19	29	26	41	37
aprobados	6	5	9	7	7	4	8	7	10	12	9	19	24
% aprobados	60.0	62.5	69.2	38.9	70.0	100	61.5	58.3	52.6	41.4	34.6	46.3	65.9

Cuadro 1: Número de estudiantes de la asignatura y número y porcentaje de alumnos que superan la asignatura en los últimos trece cursos

utilizando los recursos del Concurso Español de Programación Paralela, cuya primera edición fue en 2011 [1], y en la nueva organización de la asignatura se utilizan estos recursos más exhaustivamente para facilitar la evaluación continua. Normalmente se utilizaba una aproximación de aprendizaje basado en problemas [2], que parecía una buena opción para una asignatura optativa, pero al pasar la asignatura a obligatoria para los alumnos de la intensificación de Ingeniería del Software el porcentaje de aprobados bajó considerablemente (Cuadro 1). Por tanto, para evitar que los alumnos dejen de seguir la asignatura, se ha reducido la valoración de la última práctica en la nota final y se han aumentado las prácticas guiadas, y la práctica final se realiza también con problemas del Concurso de forma que se facilita su monitorización, validación y evaluación.

3.1. Temario y organización

La asignatura se organiza en trece sesiones de teoría y diez de prácticas presenciales, y su contenido se muestra en el Cuadro 2. En la primera sesión se revisan conceptos de paralelismo y computación distribuida y se justifica la necesidad del paralelismo. Siguen cinco sesiones sobre entornos y herramientas de paralelismo, y cada una de estas sesiones tiene la correspondiente sesión práctica la semana siguiente. Los alumnos trabajan en parejas en estas sesiones prácticas y cada una de ellas se evalúa con un máximo de 0.5. A continuación hay cinco sesiones sobre paradigmas algorítmicos paralelos, con las correspondientes sesiones prácticas donde el trabajo es individual y con una puntuación máxima de 1 cada una de ellas. Se acaba con una sesión de análisis de algoritmos paralelos y otra de metodología de programación paralela, donde se completan los conceptos de análisis y metodología que se han utilizado en las sesiones anteriores. Cada estudiante trabaja finalmente en una práctica individual, diferente para cada alumno, con una puntuación máxima de 2.5, y en la que se realizan implementaciones OpenMP, MPI, MPI+OpenMP y opcionalmente CUDA de un algoritmo secuencial, con un análisis teórico y experimental de las implementaciones realizadas.

3.2. Prácticas

Como se ha mencionado, la asignatura se guía por las prácticas, que se organizan en tres bloques: entornos, algoritmos y proyecto. A continuación se explica el contenido de cada bloque.

3.2.1. Herramientas y entornos de paralelismo

Se dedican cinco sesiones prácticas a trabajar con diferentes entornos de paralelismo y a que los alumnos se familiaricen con las herramientas para implementar algoritmos paralelos y que usarán en el proyecto final. Se da una descripción breve del trabajo en cada sesión:

- Entornos de paralelismo. Se proporcionan ejemplos básicos de paralelización de un algoritmo de ordenación en C++, Java y Pthreads. Los ejemplos usan únicamente dos hilos o procesos, y los alumnos deben modificar los ejemplos para que funcionen con más hilos y procesos, y tienen que comparar las prestaciones obtenidas y la programabilidad de los tres entornos.
  Se practica con el sistema del Concurso Español de Programación Paralela (CPP) experimentando con un problema de un proyecto de introducción del paralelismo en formación profesional [7]. El mismo problema se utiliza para practicar con el *cluster* heterogéneo que se usa para los experimentos en la práctica final. El *cluster* consta de seis nodos, con un total de 64 núcleos y con tarjetas GPU y Xeon Phi cada nodo (10 GPUs y 2 Xeon Phi) [5].
  Cada grupo (de dos alumnos) envía al profesor al final de la sesión la respuesta a una serie de cuestiones planteadas en la práctica. En esta primera práctica el profesor contesta a los alumnos el día siguiente, para aclarar cuestiones que no hayan quedado claras sobre el uso de los sistemas, y los alumnos disponen de cuatro días para mejorar las respuestas de su envío inicial. Para el resto de prácticas se utiliza el mismo sistema de dos envíos, uno al final de la sesión y otro cuatro días después, pero no hay realimentación inmediata por parte del profesor. Las prácticas se evalúan con una mayor puntuación en la primera entrega, y la puntuación de la segunda entrega se determina en el boletín de cada práctica.

Tema	Semana Teo.	Prac.
Introducción a la computación paralela, necesidad del paralelismo	1	
Sistemas paralelos y paradigmas de programación paralela	2	3
Programación de memoria compartida, OpenMP	3	4
Programación por paso de mensajes, MPI	4	5
Computación híbrida y heterogénea	5	6
Programación de *manycores*, CUDA	6	7
Paralelismo y particionado de datos	7	8
Algoritmos relajados y paralelismo síncrono	8	9
Algoritmos sobre árboles y grafos, *Pipeline*	9	10
Paralelismo en Divide y Vencerás y Programación Dinámica	10	11
Paralelismo en árboles de búsqueda, Maestro-Esclavo, Bolsa de Tareas, Granja de Procesos, Trabajadores Replicados	11	12
Análisis de algoritmos paralelos	12	
Metodología de la programación paralela	13	

Cuadro 2: Planificación temporal de las sesiones teóricas y prácticas

- OpenMP. Se trabaja con el mismo ejemplo de la práctica anterior en una prueba del CPP. Se analiza la influencia del número de hilos en el tiempo de ejecución. También se trabaja con una multiplicación de matrices clásica, en la que hay que analizar el tiempo obtenido con paralelización de cada uno de los tres bucles y con el uso de la cláusula `collapse`. Los alumnos tienen que desarrollar versiones que usen secciones y tareas.
- MPI. Se utilizan versiones MPI de los mismos algoritmos anteriores. Del algoritmo de ordenación los alumnos hacen versiones sustituyendo comunicaciones punto a punto por comunicaciones colectivas, y en la multiplicación sustituyendo comunicaciones síncronas por asíncronas.
- Computación híbrida y heterogénea. A partir de las versiones anteriores de multiplicación de matrices con OpenMP y MPI los alumnos realizan una versión híbrida MPI+OpenMP, y analizan las prestaciones al variar el tamaño de la matriz y el número de procesos MPI e hilos OpenMP. La versión híbrida se utiliza en el *cluster* heterogéneo para determinar la distribución óptima de procesos. Adicionalmente, los alumnos deben desarrollar una versión heterogénea, con un proceso por nodo pero distinto volumen de datos y número de hilos en cada nodo.
- CUDA. La programación de *manycores* no es un tema fundamental en la asignatura ya que hay una asignatura optativa sobre este tema en el segundo cuatrimestre, por lo que en esta práctica no se piden implementaciones originales. Se proporciona una versión CUDA de la multiplicación de matrices y los alumnos analizan los tiempos de ejecución obtenidos en el CPP y en las distintas GPUs del *cluster* al variar el número de hilos por bloque, y proponen alguna estrategia de búsqueda del valor óptimo.

3.2.2. Paradigmas algorítmicos paralelos

Al igual que en el bloque anterior, los alumnos envían respuestas a las cuestiones planteadas al final de la sesión y pueden hacer un segundo envío completando el primero hasta cuatro días después. Las cuestiones planteadas y su valoración en cada envío varía con la práctica, y la evaluación continua permite al profesor adaptar el tipo de cuestiones y su valoración según el progreso de los alumnos. Las sesiones de teoría duran dos horas, de las que se dedican aproximadamente los veinte primeros minutos a comentar los problemas detectados en la evaluación de la práctica anterior, y al final de la sesión se dedican unos diez minutos a clarificar algunos puntos del enunciado de la siguiente práctica.

En estas sesiones se utilizan algunos de los problemas propuestos en el CPP. Estos problemas se pueden asignar de distintas formas a los distintos paradigmas en el temario. El Cuadro 3 muestra una posible asignación, y los problemas usados en las sesiones prácticas se etiquetan con la semana de la sesión en que se han usado. Los paradigmas no son exclusivos, por lo que algunos problemas aparecen asociados a varios de ellos. En el CPP se proporcionan a los participantes versiones secuenciales para cada problema, y ellos deben desarrollar versiones paralelas intentando maximizar la ganancia de velocidad. En 2012 hubo dos concursos, uno de OpenMP y MPI y otro de CUDA, que comprendía tres problemas, sólo uno de los cuales era distinto de los del concurso OpenMP+MPI. En la página del concurso se pueden consultar los problemas, la solución secuencial que se proporciona y la tabla de

récords obtenidos durante los concursos y después de ellos, con los correspondientes códigos.

En cada sesión práctica los alumnos responden una serie de cuestiones que incluyen la resolución de problemas con programación paralela en el sistema del CPP. Las cuestiones y la forma en que se evalúan son distintas en sesiones distintas, con dos plazos de entrega con una puntuación menor para las soluciones entregadas en el segundo plazo. Como ejemplo, se comentan las cuestiones y la guía de la evaluación para la sesión de Algoritmos en árbol y *Pipeline*:

- Cuestión 1: Hay que resolver utilizando el esquema de *Pipeline* el problema A del concurso de 2015. Se consideran submatrices de tamaño $m \times m$ en la diagonal de una matriz de dimensión $n \times n$, con el bloque inferior/derecha de tamaño $l \times l$ de cada submatriz solapando con la siguiente submatriz en la diagonal. Se trata de realizar varias iteraciones de cuadrados de las submatrices, primero la submatriz al principio de la matriz, a continuación la siguiente submatriz en la diagonal, etc. El cuadrado de la segunda iteración de la primera submatriz se puede empezar a calcular una vez se ha calculado el cuadrado de la segunda submatriz, con lo que se puede trabajar con la primera y tercera submatriz al mismo tiempo, y se pueden hacer simultáneamente operaciones de distintas iteraciones en submatrices distintas. Se puede dar una solución simplificada válida para 4 y 6 hilos o la solución general válida para cualquier número de hilos.
- Cuestión 2: En el problema C de 2012 se trabaja con una matriz cuadrada realizando varios pasos, en cada uno de los cuales se realiza una ordenación de los datos de la matriz y a continuación se multiplica la matriz por sí misma. Se trabaja sólo en la paralelización de la ordenación. La solución secuencial que se proporciona usa *Quicksort* y los alumnos deben sustituirla por una implementación MPI del *Mergesort*, con la mezcla realizada con un esquema de árbol. Se puede dar una solución simplificada válida para 2, 4 y 8 procesos o una general sin restricciones en el número de procesos.

Las sesiones prácticas son de una hora y tres cuartos, con lo que el tiempo que tienen los alumnos para responder a las cuestiones es limitado, por lo que se les da la oportunidad de trabajar durante las prácticas en las versiones simplificadas y completar las soluciones para el segundo envío. La versión simplificada de cada cuestión se puntúa sobre 0.5 en la primera entrega y sobre 0.3 en la segunda, y las versiones generales 0.7 y 0.4, y la puntuación máxima de la práctica es 1. El entorno del CPP facilita controlar el trabajo de cada alumno, validar las soluciones, evaluar las prestaciones y comparar soluciones de distintos alumnos.

3.2.3. Proyecto final

Después de las prácticas de evaluación continua hay una práctica en forma de proyecto en la que los alumnos trabajan de forma individual y autónoma, cada uno con un problema distinto del CPP. Este trabajo se evalúa sobre 2.5. Se resumen a continuación las distintas partes del estudio y la documentación a realizar y la puntuación asignada a cada parte:

- Explicación del método secuencial proporcionado por la organización y propuestas de posibles mejoras (0.2).
- Solución OpenMP, con pseudocódigo y código comentado. Se deben explicar las soluciones OpenMP que hay para este problema en la tabla de récords, y las modificaciones realizadas para reducir el tiempo de ejecución (0.4).
- Solución MPI, con pseudocódigo y código comentado. Se deben explicar las soluciones MPI para este problema en la tabla de récords, y las modificaciones realizadas para reducir el tiempo de ejecución (0.4).
- Solución MPI+OpenMP, con pseudocódigo y código comentado (0.2).
- Análisis teórico de los algoritmos, con estimación del tiempo de ejecución, el *speed-up*, la eficiencia y la isoeficiencia, y con justificación del paradigma algorítmico utilizado (0.4).
- Estudio experimental de las implementaciones. Los alumnos deben decidir el tipo y tamaño de las entradas en función de las peculiaridades de su problema, y analizan el tiempo de ejecución, el *speed-up*, la eficiencia y la isoeficiencia (0.5).
- Se puede usar el CPP para validar, pero habrá que diseñar un método de validación más general (0.2).
- Contraste de los estudios teórico y experimental (0.2).

Se puede subir la nota de la asignatura con trabajos complementarios:

- Una implementación CUDA para el problema que se tiene asignado puede puntuarse hasta con un punto adicional, dependiendo de la originalidad y la eficiencia de la implementación, y se debe realizar un análisis teórico y experimental.
- Se puede incrementar la puntuación hasta un punto con nuevos récods de problemas del CPP, y cada nuevo récord se puede puntuar hasta con 0.4, dependiendo de la originalidad y de la mejora obtenida.

Prob	Paralelismo de datos	Particionado de datos	Algoritmos relajados	Paralelismo síncrono	Algoritmos en árbol	*Pipeline*	Divide y Vencerás	Programación Dinámica	Búsqueda en árbol	Trabajadores replicados
11A	X	X								
11B	X			9				X		
11C							X			
11D	X	X	X							
11E									X	X
12A	8	8	X					X		
12B	X	X	X				X			
12C	X	X			10		X			
12D									X	X
12E								11		
12A-CUDA	X	X		X						
13A	X	X								
13D	X	X		X						
13F									12	12
14A	X	X	X							
14D	X	X	X							
14F									X	X
15A	X			X	X	10				X
15C	X	X	X	X						
15E	X	X	X							

Cuadro 3: Asociación de problemas del CPP a los paradigmas de programación paralela estudiados en la asignatura

4. Evaluación

Con la reorganización de la asignatura se pretendía principalmente aumentar el número de alumnos que alcanzan los conocimientos suficientes para aprobar. Este objetivo se ha alcanzado, con un incremento de aproximadamente 45 % aprobados durante los últimos cursos a una previsión de un 65 % el presente curso (Cuadro 1).

Los estudiantes que no participan en las sesiones prácticas o que no obtienen la puntuación mínima establecida para estas sesiones pueden seguir un sistema de evaluación no continua. En el Cuadro 4 se muestra la evolución del número de alumnos que optaron por cada tipo de evaluación después de cada sesión. El número total de estudiantes matriculados es 37, y 6 de ellos no participaron en ninguno de los dos tipos de evaluación, lo que está en concordancia con lo que suele pasar todos los cursos. El porcentaje de alumnos que se espera que superen la asignatura con respecto al número de alumnos que la siguen sube al 77 %. También se muestra la puntuación media de las soluciones de cada sesión. Esta puntuación es mayor en las sesiones iniciales, en las que se trabaja por pares, pero en las últimas sesiones el número de envíos y su puntuación disminuye, lo que puede ser debido a la acumulación de prácticas de otras asignaturas. Para propiciar que los alumnos no abandonen la asignatura, se les da la oportunidad de completar una de las prácticas y entregarla junto con el proyecto final.

El porcentaje de alumnos que aprueba ha aumentado, pero esto puede ser a costa de una mayor carga de trabajo del profesor y los alumnos. El número de horas que el profesor estima que ha dedicado a las prácticas presenciales ha aumentado aproximadamente de 40 a 100 horas. Se ha dedicado más tiempo a la preparación de cada sesión práctica, y se ha dedicado a la evaluación de cada sesión aproximadamente 4 horas para la primera entrega y otras 4 para la segunda.

Se ha preparado una encuesta para consultar a los alumnos sobre el número de horas que estiman que han dedicado a cada parte de la asignatura y la percepción que tienen sobre la organización en este curso. La encuesta la han contestado sólo 13 alumnos, con lo que, teniendo también en cuenta la subjetividad de las respuestas, las conclusiones no pueden ser significativas. Se pregunta por el tipo de organización del curso que prefieren: el seguido este año, con evaluación continua y proyecto final (Curso); el utilizado los años anteriores, con algunas prácticas cortas y un trabajo final más largo (Más proyecto); la organización tradicional con examen y prácticas (Examen+Prácticas)). Todos los alumnos prefieren el sistema utilizado este curso, salvo uno que prefiere el sistema con un proyecto más largo, lo que se puede hacer con la opción de evaluación no continua. Para cada sistema de eva-

Sesión:	1	2	3	4	5	6	7	8	9	10
No continua	3	3	7	7	7	7	7	7	7	8
Continua	28	28	24	24	24	24	24	24	24	23
Soluciones	27	27	24	23	24	24	24	24	19	13
Puntuación media	7.2	6.8	5.1	5.6	9.9	6.1	6.2	3.4	4.9	3.4

Cuadro 4: Número de alumnos que siguen los sistemas de evaluación continua y no continua, y número de soluciones y puntuación media para cada sesión práctica

luación se les consulta sobre tres aspectos: el sistema facilita aprobar la asignatura; se necesita trabajar más para aprobar; el método propicia el aprendizaje. A cada una de las cuestiones de contesta con un valor entre 1 (totalmente en desacuerdo) y 5 (totalmente de acuerdo). La Figura 1 muestra la media de las respuestas para las tres cuestiones y los tres métodos de evaluación. Los alumnos perciben que la organización seguida este curso requiere más trabajo por su parte, pero al mismo tiempo es el sistema con el que es más fácil aprobar y el que más propicia el aprendizaje.

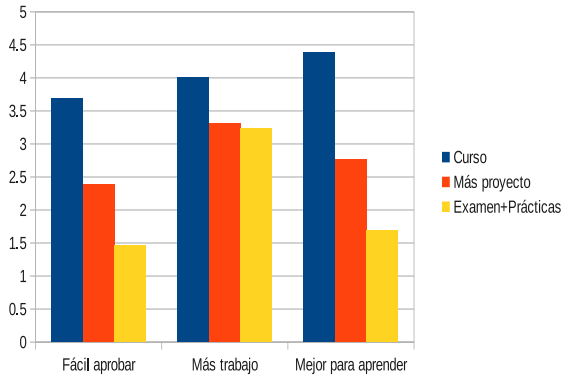

Figura 1: Media de las respuestas de los alumnos para las tres cuestiones y tres sistemas de evaluación de la asignatura

Se les ha consultado también sobre el tiempo que estiman que han dedicado a asistir a las clases de teoría, a estudiar lo tratado en ellas, al trabajo presencial en las sesiones prácticas y para completar las prácticas. El Cuadro 5 resume el tiempo que el profesor estima que habría que dedicar a cada parte (Estimado), la media de las respuestas de los alumnos (Media), y la media sin considerar dos casos extremos (Media filtrada), que corresponden a un alumno que no sigue la evaluación continua y otro que estima que ha dedicado 400 horas a completar las prácticas (la media es de 84 y el segundo mayor tiempo estimado es 120). El tiempo que cada alumno (sin considerar los casos extremos) estima que ha dedicado a cada parte se muestra en la Figura 2, junto con la media de estas estimaciones y los tiempos estimados por el profesor. Las medias dan valores cercanos a las estimaciones del profesor.

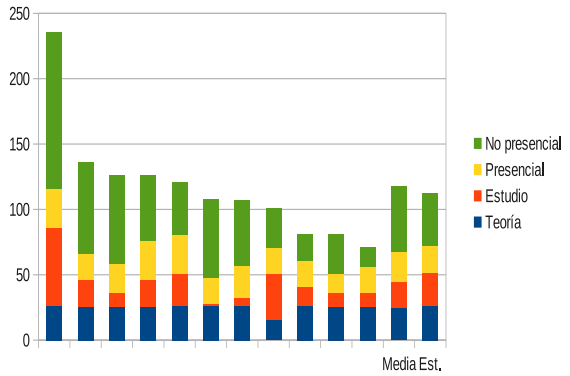

Figura 2: Tiempos estimados por los alumnos para las distintas partes, medias de las estimaciones y tiempos estimados por el profesor

5. Conclusión y perspectivas

Se ha presentado una experiencia de un curso de programación paralela con evaluación continua basada en la realización de prácticas tras cada sesión de teoría. Se han utilizado los recursos (sistema, problemas y soluciones de la tabla de récods) del Concurso Español de Programación Paralela para las sesiones prácticas y para la práctica final. El objetivo principal de la nueva forma de organizar y evaluar la asignatura era conseguir que aumentara el número de alumnos que aprueban, lo que ha sido alcanzado, con un porcentaje de 65 % aprobados sobre el número de matriculados y de 77 % sobre el número de alumnos que siguen la asignatura. Además, las respuestas de los alumnos a una encuesta sobre la organización del curso muestran que prefieren esta organización a otras utilizadas anteriormente, al ser con la que más aprenden y con la que es más fácil aprobar, aun a costa de un incremento en el tiempo dedicado a la asignatura, aunque el tiempo

	Teoría	Estudio	Presencial	No presencial	Total
Estimado	26	26	20	40	112
Media filtrada	25	19	21	50	115
Media	24	19	21	73	137

Cuadro 5: Estimación del tiempo dedicado a las clases de teoría (Teoría), a estudiar lo visto en ellas (Estudio), al trabajo presencial (Presencial) y no presencial (No presencial) en prácticas

medio que estiman está cercano al estimado por el profesor y que corresponde a la carga de la asignatura en el plan de estudios.

Este tipo de organización y evaluación se piensa seguir utilizando en los próximos cursos. Los problemas del CPP (cuyo número se incrementa con cada nueva edición del concurso) y las soluciones en la tabla de récords se seguirán utilizando para planificar nuevas prácticas, y pueden utilizarse en otros cursos de programación paralela. Además, las diferentes formas en que se pueden asignar los problemas a distintos paradigmas algorítmicos paralelos permiten múltiples configuraciones en la planificación de las prácticas.

Agradecimientos

Este trabajo ha sido financiado por MINECO y por fondos FEDER de la Comisión Europea, a través del proyecto TIN2015-66972-C5-3-R.

El profesor agradece a los alumnos que han contestado la encuesta las críticas positivas y negativas y las sugerencias recibidas.

Referencias

[1] Francisco Almeida, Vicente Blanco Pérez, Javier Cuenca, Ricardo Fernández-Pascual, Ginés García-Mateos, Domingo Giménez, José Guillén, Juan Alejandro Palomino Benito, María Eugenia Requena y José Ranilla. An experience on the organization of the First Spanish Parallel Programming Contest. *Olympiads in Informatics*, 6:133–147, 2012.

[2] Ángel Calvo, Ana Cortés, Domingo Giménez y Carmela Pozuelo. Using metaheuristics in a parallel computing course. En Marian Bubak, G. Dick van Albada, Jack Dongarra y Peter M. A. Sloot, editores, *ICCS (2)*, volumen 5102 de *Lecture Notes in Computer Science*, páginas 659–668. Springer, 2008.

[3] Domingo. Giménez. Página web del curso Metodología de la Programación Paralela, Universidad de Murcia, `http://dis.um.es/~domingo/mpp.html`.

[4] José Paulo Leal y Fernando Silva. Mooshak: a web-based multi-site programming contest system. *Softw., Pract. Exper.*, 33(6):567–581, 2003.

[5] Página web del grupo de Computación Científica y Programación Paralela de la Universidad de Murcia. `http://luna.inf.um.es/grupo_investigacion/`.

[6] Sushil K. Prasad, Almadena Yu. Chtchelkanova, Anshul Gupta, Arnold L. Rosenberg y Alan Sussman. NSF/IEEE-TCPP curriculum initiative on parallel and distributed computing: core topics for undergraduates. En *The 45th ACM Technical Symposium on Computer Science Education, SIGCSE '14*, página 735, 2014.

[7] Universidad de Murcia-Instituto Juan de la Cierva. Proyecto de incorporación de contenidos de programación paralela en estudios de la rama de Tecnologías de la Información en Formación Profesional, `http://dis.um.es/~domingo/fp14.html`.

[8] Vladimir V. Voevodin, Victor Gergel y Nina Popova. Challenges of a systematic approach to parallel computing and supercomputing education. En *Euro-Par 2015: Parallel Processing Workshops*, páginas 90–101, 2015.