GPUBenchmark: un banco de pruebas para GPUs

Sergio Barrachina Mir[1], María Isabel Castillo Catalán[1], Adrián Castelló Gimeno[1], Rafael Mayo Gual[1], Javier Ortells Lorenzo[2] y Enrique S. Quintana Ortí[1]

Resumen— **GPUBenchmark es un banco de pruebas para GPUs que proporciona información detallada sobre las prestaciones del equipo en que se ejecuta. La información mostrada se centra en aquellas características del equipo que son relevantes para el desarrollador de aplicaciones de propósito general con GPUs. De esta forma, el desarrollador puede disponer fácilmente de una completa información de referencia que le permita valorar mejor el rendimiento obtenido por sus propias aplicaciones, ya que podrá comparar sus resultados con los proporcionados por GPUBenchmark.**

Palabras clave— **banco de pruebas, benchmark, test, rendimiento, GPU, CUDA**

I. Introducción

LOS procesadores gráficos (*Graphic processing units*, GPUs) son cada vez más ampliamente utilizadas en computación de propósito general debido a su gran capacidad de cálculo paralelo, su relativo bajo coste y consumo, y su facilidad de programación.

Aunque la programación de GPUs se ha visto facilitada enormemente gracias a CUDA y a OpenCL, el desarrollador de aplicaciones para GPUs debe ser consciente de la arquitectura sobre la que está trabajando para poder obtener el máximo rendimiento en sus aplicaciones. Así pues, es habitual tener que evaluar el rendimiento de distintos algoritmos alternativos para ver cuál de ellos aprovecha mejor los recursos proporcionados por la GPU.

Para que dicha evaluación pueda ser contrastada, sería conveniente saber cuál es el rendimiento esperado para dicha GPU en aquellos aspectos que son relevantes cuando se utilizan GPUs para computación de propósito general. Así pues, sería interesante disponer de un banco de pruebas que evalúe el rendimiento de la GPU en dichos aspectos. De esta forma, el desarrollador podría comprobar si el rendimiento obtenido por los algoritmos que está evaluando está efectivamente en consonancia con los resultados proporcionados por dicho banco de pruebas.

Tras buscar bancos de pruebas para GPUs que pudieran cumplir con las anteriores características, encontramos los tres siguientes: GPUBench, Rodinia y SHOC; que describimos brevemente a continuación.

GPUBench [2] es un banco de pruebas orientado a la evaluación del rendimiento de GPUs para aplicaciones numéricas y científicas. Requiere que la GPU esté instalada en un computador con Windows y proporciona una salida detallada en formato html en la que se presenta información sobre la tarjeta gráfica, las extensiones OpenGL soportadas por dicha tarjeta y los resultados obtenidos (en forma de gráficas) para los distintos tests, así como los parámetros utilizados para cada una de las ejecuciones. La información que proporciona GPUBench podría ser utilizada por un desarrollador de aplicaciones para contrastar el rendimiento obtenido por su aplicación e incluso para ver qué estrategias de programación podrían ser más beneficiosas en el computador evaluado. Lamentablemente, puesto que su desarrollo fue previo a la aparición de CUDA, las medidas de rendimiento que proporciona están relacionadas con las primitivas gráficas que era necesario utilizar en aquel momento para la programación de GPUs para propósito general (GPGPU).

El segundo caso evaluado, Rodinia [3], es un banco de pruebas para computación sobre sistemas heterogéneos. Proporciona un conjunto de aplicaciones que han sido seleccionadas para abarcar el máximo número de dominios posibles. Para cada una de dichas aplicaciones se ha implementado una versión para GPU, otra versión para múltiples CPUs y una versión secuencial. Las versiones para GPU se han desarrollado utilizando CUDA, y las versiones para múltiples CPUs se han desarrollado utilizando OpenMP. El objetivo de Rodinia es caracterizar el rendimiento de un sistema heterogéneo en función de la aceleración obtenida por cada una de dichas aplicaciones en su versión para GPUs frente a su versión paralela con varias CPUs y a su versión secuencial. Además, proporciona información sobre los distintos recursos hardware requeridos por cada una de dichas aplicaciones. No obstante, a pesar de que este banco de pruebas proporciona información detallada sobre cómo se comportan un amplio abanico de aplicaciones en un determinado sistema heterogéneo, no tiene como objetivo cuantificar el coste de las operaciones básicas relacionadas con la programación de GPUs, por lo que, a nuestro juicio, no se podría utilizar directamente para valorar si los resultados obtenidos al desarrollar una nueva aplicación son consistentes con lo esperado en el equipo en el que se se está realizando la evaluación.

El tercer banco de pruebas que hemos analizado, SHOC [4], también está dirigido a computación sobre sistemas heterogéneos. Proporciona un conjunto de aplicaciones diferenciadas en dos niveles. Las del primer nivel se centran en las características hardwa-

[1]Dpto. de Ingeniería y Ciencia de los Computadores, Universidad Jaume I, emails: {barrachi,castillo,adcastel,mayo,quintana}@uji.es
[2]Dpto. de Lenguajes y Sistemas Informáticos, Universidad Jaume I, email: jortells@uji.es

re y las del segundo nivel corresponden a algoritmos básicos de programación paralela. Los programas para GPUs se han implementado principalmente con OpenCL, aunque también se proporcionan versiones en CUDA para comparar el rendimiento en función de qué entorno de programación se utilice. En cuanto a la paralelización a nivel de multinúcleo, ésta se ha realizado también con OpenCL. Al igual que Rodinia, se trata de un banco de pruebas muy completo que utiliza las nuevas interfaces de programación de aplicaciones sobre GPU. Sin embargo, SHOC no tiene como objetivo presentar resultados detallados para cada caso en función de los distintos tamaños de problema posibles. Su objetivo es, al igual que en el caso de Rodinia, poder comparar el rendimiento de diversos equipos.

A la vista de los bancos de pruebas para GPUs analizados, consideramos interesante iniciar el desarrollo de un nuevo banco de pruebas que, al igual que GPUBench, generará información relevante para el desarrollador de aplicaciones de computación de propósito general sobre GPUs, pero utilizando en este caso los entornos actuales de programación para GPUs.

El banco de pruebas propuesto en este artículo, GPUBenchmark, pretende proporcionar información detallada sobre las prestaciones de un equipo con GPUs que vaya a utilizarse para la programación de aplicaciones paralelas de propósito general. Al igual que los bancos de pruebas previamente comentados, consiste en un conjunto de aplicaciones. No obstante, dichas aplicaciones, en lugar de caracterizar problemas complejos, se centran en aquellos aspectos que determinarán las prestaciones de las aplicaciones que sean objeto de evaluación, como son:

- La velocidad de transferencia entre el disco duro y la memoria principal.
- La velocidad de transferencia entre la GPU y la memoria principal.
- La velocidad de transferencia entre dos GPUs.
- El rendimiento de la multiplicación matriz-matriz en GPU y en CPU.
- El rendimiento de la multiplicación matriz-vector en GPU y en CPU.

Así pues, el objetivo de GPUBenchmark es el de proporcionar al desarrollador de aplicaciones sobre GPUs información detallada que le permita comprobar que los resultados obtenidos al medir el rendimiento de su aplicación son consistentes con el rendimiento esperado del computador sobre el que esté realizando la evaluación. Para ello, generará un informe, adaptado a las características hardware y software del equipo en el que se ejecute, en el que se describirá numérica y gráficamente las características indicadas anteriormente para diversos tamaños de problema.

El resto del artículo está organizado como sigue. El siguiente apartado describe las características de GPUBenchmark. El apartado 3 describe cómo descargar, configurar, compilar y ejecutar GPUBenchmark. El apartado 4 presenta los resultados obtenidos al ejecutar GPUBenchmark sobre varios computadores. Por último, el apartado 5 presenta las conclusiones y el trabajo futuro.

II. Características de GPUBenchmark

GPUBenchmark es un banco de pruebas para GPUs, multiplataforma (en parte, por el momento), modular, extensible, fácil de utilizar y que genera un informe detallado en formato pdf con los resultados obtenidos en el equipo evaluado.

El primer paso que se ha tomado para conseguir que en un futuro GPUBenchmark sea completamente multiplataforma, ha sido la utilización del sistema de construcción `CMake` [1] para la configuración y compilación de GPUBenchmark. El procedimiento de configuración y compilación se ha desarrollado de tal forma que solo se compilan aquellos tests para los que las bibliotecas necesarias están disponibles. Además, durante el procedimiento de configuración, GPUBenchmark avisa al usuario de qué bibliotecas no están disponibles y qué tests no se compilarán por dicho motivo (ver la figura 1). Conviene destacar que no es necesario instalar las bibliotecas indicadas para que el resto de GPUBenchmark se configure, compile y ejecute correctamente.

```
$ cmake ./
-- The C compiler identification is GNU
-- The CXX compiler identification is GNU
-- Check for working C compiler: /usr/bin/gcc
-- Check for working C compiler: /usr/bin/gcc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Found CUDA: /usr/local/cuda/4.0/cuda (found version "4.0")
CMake Warning at CMakeLists.txt:45 (message):
  Can not find the 'cblas' library.  Please, install it or if it is already
  installed, set its path in the environment variable 'LD_LIBRARY_PATH'.  Some
  tests will not compile if this library is not found.


CMake Warning at gb_matrixmatrix/CMakeLists.txt:12 (message):
  'gb_matrixmatrix' test is not being compiled because no CBLAS library could
  be found (see CBLAS warning above).


CMake Warning at gb_matrixvector/CMakeLists.txt:12 (message):
  'gb_matrixvector' test is not being compiled because no CBLAS library could
  be found (see CBLAS warning above).
```

Fig. 1. Salida del procedimiento de configuración de GPUBenchmark informando de que la biblioteca CBLAS no está instalada y de los tests que no se compilarán por dicho motivo

GPUBenchmark se ha desarrollado de forma modular. Así, cada módulo es responsable de generar la información correspondiente a cada uno de los tests disponibles. Además, cada módulo consiste simplemente en un directorio que, una vez compilado, contendrá tres ficheros.

El primero de dichos ficheros es el ejecutable que lleva a cabo el test en sí. El ejecutable realiza varias veces la operación cuyo rendimiento se desea evaluar y devuelve el resultado obtenido. La operación se realizará únicamente con los parámetros que se le hayan indicado al ejecutarlo. Es decir, no devuelve resultados para la misma operación con tamaños de datos distintos.

El segundo de los ficheros es un guión que: (i) ejecuta el test con distintos tamaños de problema y, en su

caso, con distintos parámetros; (II) procesa los resultados; y (III) genera las gráficas y tablas que se mostrarán en el apartado correspondiente a dicho test en el informe final. Los parámetros que se utilizarán en la ejecución por defecto están indicados en las primeras líneas del guión. Naturalmente, en el caso de que el usuario decida modificar dichos parámetros, la información generada por el guión sobre la ejecución del test se modificará consecuentemente.

El tercero de los ficheros que constituye un módulo es una plantilla LaTeX que describe el test y que se completará, tras la ejecución del test, con la información sobre los parámetros utilizados en su ejecución y los resultados obtenidos, tanto en forma de tablas como de gráficas.

Gracias a esta arquitectura modular, GPUBenchmark puede extenderse fácilmente para soportar nuevos tests. Para ello, basta con crear un nuevo directorio con los tres ficheros anteriormente indicados y añadir en el fichero de configuración global de `CMake` el nuevo directorio. Cuando se ejecute GPUBenchmark, se utilizará la información disponible en dicho fichero de configuración para decidir qué tests se deben ejecutar, en qué orden deberán ejecutarse y qué información deberá aparecer en el informe que se genere. A grandes rasgos, la ejecución de cada test consiste en llamar a su guión y en copiar su plantilla LaTeX y los resultados generados a un directorio de salida. Todo se ha estructurado de tal forma que no sea necesario modificar el guión principal cuando se añadan nuevos módulos.

En cuanto a la facilidad de uso de GPUBenchmark, éste se ha diseñado para que la ejecución completa y la generación del informe correspondiente requieran únicamente la ejecución de dos guiones: uno para la ejecución de todos los tests, «`gb_run.sh`», y otro para la generación del informe, «`gb_make_pdf.sh`». No se utiliza un único guión ya que, para la generación del informe, son necesarias aplicaciones que podrían no estar instaladas en el equipo que se desea evaluar. Si éste fuera el caso, bastaría con copiar el directorio de salida a otra máquina y ejecutar allí el guión «`gb_make_pdf.sh`» (que se habrá copiado automáticamente a dicho directorio).

También con la idea de facilitar el uso de la aplicación, se ha hecho especial hincapié en que la interfaz de usuario y la salida generada por cada uno de los tests sean lo más uniformes posibles. Así, si varios tests utilizan los mismos parámetros, éstos se indicarán de la misma forma en todos ellos.

Finalmente, GPUBenchmark genera un informe en formato pdf que presenta las características hardware del computador evaluado y los resultados obtenidos tras la ejecución de cada uno de los tests. Este informe recoge, para cada test, sus características, los parámetros utilizados y los resultados obtenidos en forma de tablas y gráficas.

III. Utilizando GPUBenchmark

Para utilizar GPUBenchmark no es necesario poseer privilegios de administración en la máquina que se desee evaluar. Esto es así justamente para permitir que el desarrollador de aplicaciones para GPUs pueda evaluar la máquina que esté utilizando, aunque no tenga acceso más que como usuario normal.

Señalado lo anterior, el primer paso para utilizar GPUBenchmark consiste, como es de esperar, en descargarlo. GPUBenchmark es software libre y puede descargarse de:

`http://lorca.act.uji.es/projects/gpubenchmark/`

Una vez descargado y descomprimido, se debe ejecutar en el directorio «`GPUBenchmark`»:

```
$ cmake ./
```

En el caso de que alguno de los tests no pudiera compilarse, debido a que faltara alguna biblioteca, aparecerá un aviso en pantalla informando de dicha situación (ver de nuevo la figura 1).

Una vez ejecutada correctamente la orden «`cmake ./`», y para que se compilen los tests cuyos prerrequisitos sí que se cumplan, basta con ejecutar la orden:

```
$ make
```

Una vez configurado y compilado GPUBenchmark, éste puede utilizarse de dos formas. La primera de ellas consiste en ejecutar GPUBenchmark por completo. Para hacerlo se debe ejecutar la orden:

```
$ ./gb_run.sh
```

La anterior orden se encargará de: (I) recolectar información sobre el equipo, (II) lanzar todos los tests compilados que puedan ejecutarse sobre la máquina evaluada, con una serie de parámetros predefinidos, y (III) preparar un directorio con los resultados obtenidos, así como con los ficheros necesarios para generar un informe en pdf con dichos resultados. La figura 2 muestra la salida de «`./gb_run.sh`».

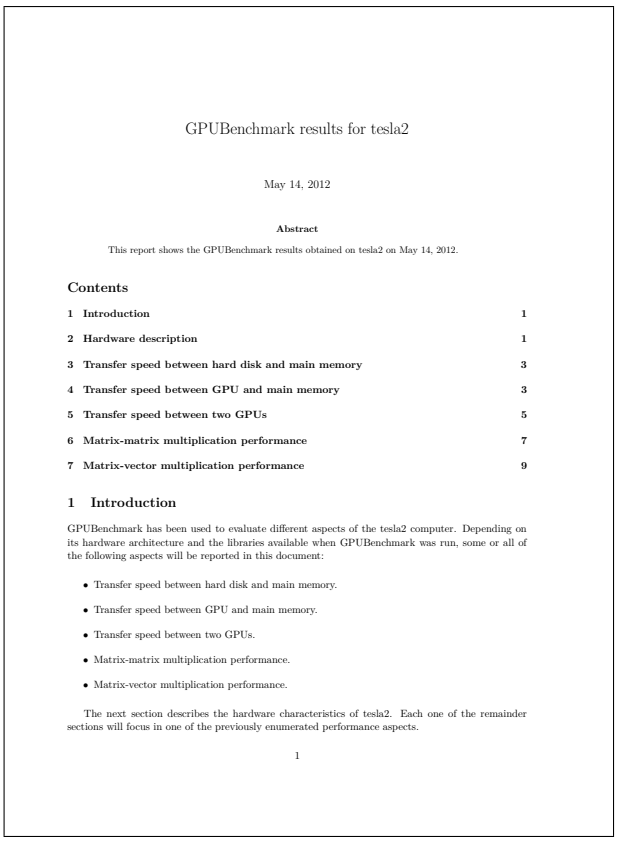
```
$ ./gb_run.sh
Writing down cpu and gpu info...
Copying LaTeX templates to gpubenchmark_tesla2_results/templates/...
Copying 'gb_make_pdf' to gpubenchmark_tesla2_results/...
Running transfer speed test between main memory and hard disk: [********]
Running transfer speed tests between main memory and GPU...
Running transfer speed test between main memory and GPU (without padding): [********]
Running transfer speed test between main memory and GPU (with padding): [********]
Running transfer speed test between two gpus: [********]
Running matrix per matrix tests...
Running matrix per matrix test (1/4): [******]
Running matrix per matrix test (2/4): [******]
Running matrix per matrix test (3/4): [******]
Running matrix per matrix test (4/4): [******]
Running matrix per vector tests...
Running matrix per vector test (1/4): [********]
Running matrix per vector test (2/4): [********]
Running matrix per vector test (3/4): [********]
Running matrix per vector test (4/4): [********]

The GPUBenchmark tests have ended. You can now either run the script:
'./gpubenchmark_tesla2_results/gb_make_pdf.sh'; or copy the contents of
the './gpubenchmark_tesla2_results/' directory to another computer and
run the 'gb_make_pdf.sh' script there.
```

Fig. 2. Salida de «`gb_run.sh`»

Al ejecutar «`./gb_run.sh`» se guardarán los resultados de los distintos tests en el directorio «`./gpubenchmark_HOSTNAME_results/`». Dentro de dicho directorio también se creará el guión «`gb_make_pdf.sh`», que puede utilizarse para generar un informe en formato pdf con los resultados y gráficas de los distintos tests. A modo de ejemplo, la figura 3 muestra la primera página

de un informe generado por GPUBenchmark. Los informes completos generados por GPUBenchmark para distintas máquinas pueden consultarse en la siguiente dirección:
`http://lorca.act.uji.es/projects/gpubenchmark/`



GPUBenchmark results for tesla2

May 14, 2012

Abstract

This report shows the GPUBenchmark results obtained on tesla2 on May 14, 2012.

Contents

1 Introduction

GPUBenchmark has been used to evaluate different aspects of the tesla2 computer. Depending on its hardware architecture and the libraries available when GPUBenchmark was run, some or all of the following aspects will be reported in this document:

- Transfer speed between hard disk and main memory.
- Transfer speed between GPU and main memory.
- Transfer speed between two GPUs.
- Matrix-matrix multiplication performance.
- Matrix-vector multiplication performance.

The next section describes the hardware characteristics of tesla2. Each one of the remainder sections will focus in one of the previously enumerated performance aspects.

1

Fig. 3. Primera página del informe generado por GPUBenchmark

Puesto que en el computador evaluado puede que no estén instaladas las aplicaciones necesarias para generar el informe (`pdflatex` y `gnuplot`), es posible copiar dicho directorio a otro computador, y ejecutar allí el guión «`gb_make_pdf.sh`», generándose entonces el informe con los resultados obtenidos en la máquina evaluada.

La segunda forma en la que puede utilizarse GPUBenchmark es ejecutando uno o varios de sus tests de forma manual, indicando en este caso en la línea de comandos los parámetros deseados.

Cada uno de los tests individuales se encuentran en un directorio que comienza por «`gb_`». Así por ejemplo, el test para medir la velocidad de transferencia entre disco duro y memoria, «`gb_hdtoram`», se encuentra en el directorio «`./gb_hdtoram/`».

Para obtener ayuda sobre la función de cada test y las opciones que dicho test soporta, se puede utilizar la opción «`--help`». La figura 4 muestra la salida de ayuda del test «`gb_hdtoram`».

Cuando se ejecute manualmente cualquiera de los tests con las opciones deseadas, éste mostrará una información similar a la que puede verse en la figura 5.

```
$ ./gb_hdtoram/gb_hdtoram --help
Usage: ./gb_hdtoram [OPTION]..
Measures the transfer speed between the hard disk and main memory.

Required options:
      --rows=NUM       number of rows of the matrix to be copied
      --cols=NUM       number of columns of the matrix to be copied

Other options:
      --double         runs the test using double precision elements
      --rep=NUM        number of repetitions of the test (default: 10)
  -v, --verbose        be more verbose
  -o, --output=FNAME   sends the output to the specified file
      --help           displays this help
```

Fig. 4. Ayuda del test `gb_hdtoram`

```
$ ./gb_hdtoram/gb_hdtoram --rows=100 --cols=100
#   M       N     Size (MiB)   HD->MM (MiB/s)  MM->HD (MiB/s)
#-------+-------+------------+---------------+---------------+
     100     100         0.04          291.20           98.32
```

Fig. 5. Salida del test `gb_hdtoram` para matrices de 100 filas y 100 columnas

IV. Resultados

GPUBenchmark se ha probado en seis computadores del grupo de investigación «High Performance Computer & Architectures» (HPC&A) de la Universidad Jaume I: `aic`, `lorca`, `tesla`, `tesla2`, `testbed` y `zape2`. El informe generado por GPUBenchmark para cada uno de dichos computadores puede consultarse en la siguiente dirección web:
`http://lorca.act.uji.es/projects/gpubenchmark/`

Aunque uno de los objetivos de GPUBenchmark es que sea multiplataforma, la versión actual se ha centrado en el soporte de GNU/Linux. En relación a este aspecto, se ha comprobado que GPUBenchmark puede configurarse, compilarse y ejecutarse sin problemas en las siguientes distribuciones de GNU/Linux, que son las instaladas en los computadores anteriormente mencionados: CentOS release 5.5, CentOS release 5.7, Gentoo Base System release 2.0.3, openSUSE 10.2 y Scientific Linux release 6.2 (Carbon). También se ha comprobado la compatibilidad con las versiones de 32 y 64 bits, puesto que uno de dichos equipos tiene instalada una versión de GNU/Linux de 32 bits y los 4 restantes, versiones de 64 bits.

Además, también se ha comprobado que GPUBenchmark se ha adaptado a las características de dichos equipos. Así, puesto que uno de los computadores, `lorca`, no disponía de GPUs con soporte para CUDA, solo se compiló y ejecutó el test encargado de medir la velocidad de transferencia entre disco duro y memoria principal (que es el único de los tests actualmente proporcionados que no requiere de CUDA). Además, durante el proceso de configuración se avisó al usuario de que el soporte para CUDA no estaba disponible y qué tests no se iban a instalar por dicho motivo. Otro de los computadores en los que se realizaron las pruebas, `aic`, no tenía instalada la biblioteca CBLAS; en consecuencia, quedaron deshabilitados automáticamente los tests de evaluación del rendimiento de las multiplicaciones matriz-matriz y matriz-vector en esta plataforma. Por último, salvo dos de los computadores evaluados, `aic` y `tesla2`, el resto de equipos con GPUs tan solo disponían de una GPU, por lo que el test de trans-

ferencia entre GPUs se deshabilitó automáticamente en dichos computadores. Se ha comprobado que en el informe generado para cada uno de dichos computadores tan solo se han mostrado aquellos apartados relacionados con los aspectos que sí habían podido ser evaluados.

Los informes generados comienzan con una descripción de las características hardware tanto de la CPU como de la GPU instaladas en el equipo evaluado. A continuación, se describen con detalle los resultados obtenidos para cada uno de los tests realizados, en apartados independientes.

Para cada test se describen los parámetros con los que se ha ejecutado el test y se presentan los resultados obtenidos en forma de tablas y gráficas.

A modo de ejemplo, se muestran a continuación una de las tablas y algunas de las gráficas generadas al ejecutar GPUBenchmark en uno de los computadores evaluados: `Tesla2`. Dicho computador tiene 8 procesadores *Intel Xeon* a 2,83 GHz, 15 GiB de memoria RAM, 4 GPUs *Tesla T20 Processor* a 1,15 GHz con 2,62 GiB cada una y 6 discos duros *Seagate Cheetah* SAS con una velocidad de transferencia de 300 Mbps y una capacidad de 146,8 GB cada uno, configurados en RAID-5, lo que proporciona un total de 800 GB. La tabla I y las figuras 6-12 muestran una de las tablas presentadas en dicho informe y varias de las gráficas generadas.

TABLA I

VELOCIDAD DE TRANSFERENCIA ENTRE EL DISCO DURO Y MEMORIA, Y VICEVERSA, PARA VARIOS TAMAÑOS DE MATRICES

Rows	Columns	Size (MiB)	Hard disk → Main memory (MiB/s)	Main memory → Hard disk (MiB/s)
128	128	0.06	244.14	113.22
256	256	0.25	411.18	200.16
512	512	1.00	451.67	274.12
1024	1024	4.00	410.21	271.65
2048	2048	16.00	588.04	275.00
4096	4096	64.00	529.61	235.49
8192	8192	256.00	167.95	272.18
10240	10240	400.00	170.60	268.33

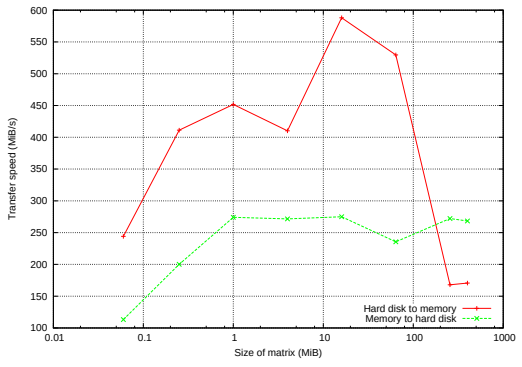


Fig. 6. Velocidad de transferencia entre memoria principal y el disco duro

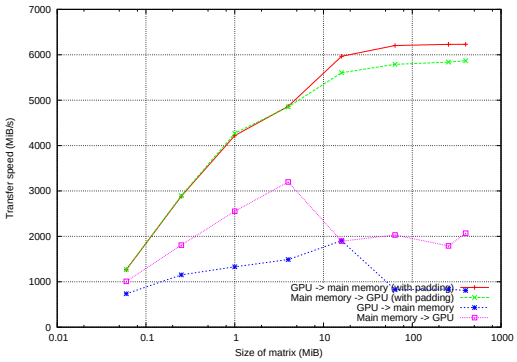


Fig. 7. Velocidad de transferencia entre GPU y memoria principal (con y sin *padding*)

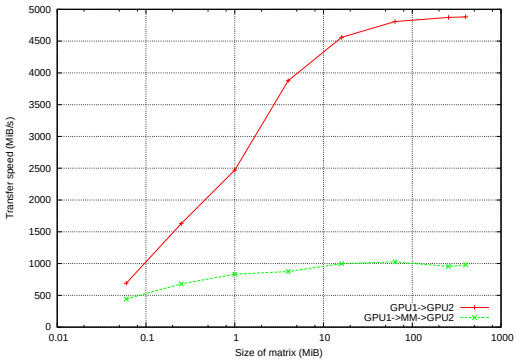


Fig. 8. Velocidad de transferencia entre dos GPUs (con GPUDirect o a través de la memoria)

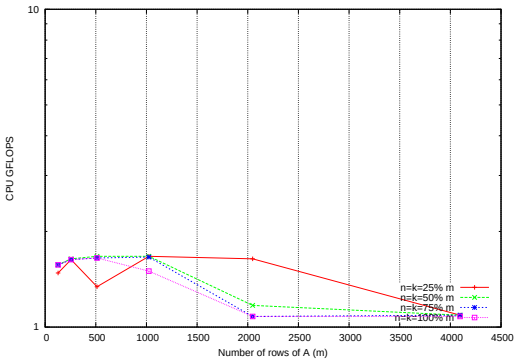


Fig. 9. GFLOPS obtenidos para distintos tamaños de matrices al realizar la multiplicación matriz-matriz en la CPU

V. CONCLUSIONES Y TRABAJO FUTURO

Se ha desarrollado un banco de pruebas, GPUBenchmark, que proporciona información detallada sobre aquellos aspectos relevantes para determinar las prestaciones de las aplicaciones de propósito general sobre GPUs. El objetivo de dicho banco de pruebas es el de permitir al desarrollador de aplicaciones sobre GPUs comprobar que el rendimiento obtenido por su aplicación es consistente con el rendimiento esperado en el equipo en el que

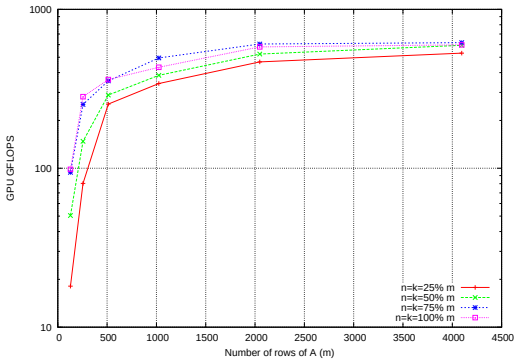


Fig. 10. GFLOPS obtenidos para distintos tamaños de matrices al realizar la multiplicación matriz-matriz en la GPU

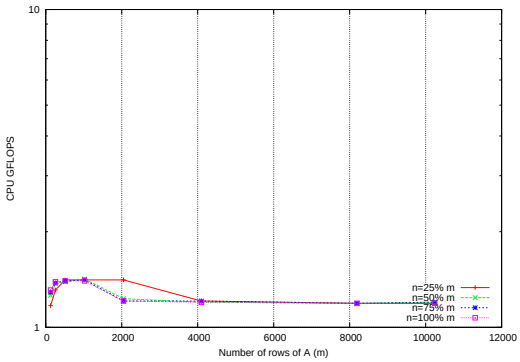


Fig. 11. GFLOPS obtenidos para distintos tamaños de matrices al realizar la multiplicación matriz-vector en la CPU

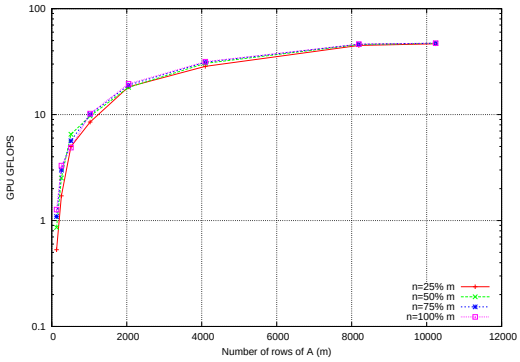


Fig. 12. GFLOPS obtenidos para distintos tamaños de matrices al realizar la multiplicación matriz-vector en la GPU

esté realizando su desarrollo. GPUBenchmark es software libre y puede descargarse desde:
`http://lorca.act.uji.es/projects/gpubenchmark/`

El banco de pruebas propuesto se puede utilizar para obtener un informe detallado en formato pdf con los resultados obtenidos en cada uno de los tests proporcionados o para obtener los resultados obtenidos por un test individual con los parámetros especificados por el usuario.

Aunque se ha desarrollado con el objetivo de que sea multiplataforma, de momento tan solo se ha probado sobre diversos equipos GNU/Linux. En los computadores en los que ha sido evaluado, GPUBenchmark se ha configurado, compilado y ejecutado correctamente, a pesar de las diferencias entre distribuciones de GNU/Linux, bibliotecas disponibles y hardware de dichos equipos. También se ha comprobado que el informe generado en cada uno de dichos equipos se ha adaptado a los tests que efectivamente podían ejecutarse en dichos computadores (en función de las bibliotecas y hardware disponibles).

En un futuro pretendemos modificar GPUBenchmark para que sea realmente multiplataforma y comprobar su funcionamiento al menos en sistemas con Microsoft Windows y Mac OSX.

Además, queremos aprovechar sus características de modularidad y extensibilidad para ampliar la batería de pruebas incluidas actualmente.

Agradecimientos

El presente trabajo ha sido financiado en parte por los proyectos: «Power-aware High Performance Computing» (Ministerio de Ciencia e Innovación, TIN2011-23283) y «rOpenCL. Sistema de acceso remoto a GPUs para cálculo de propósito general» (Bancaja, P11B2011-19).

Referencias

[1] *CMake, the cross-platform, open-source build system.* `http://www.cmake.org/`

[2] Ian Buck, Kayvon Fatahalian, Pat Hanrahan, *GPUBench: Evaluating GPU performance for numerical and scientific applications*, ACM Workshop on General-Purpose Computing on Graphics Processors, 2004.

[3] Shuai Che, Michael Boyer, Jiayuan Meng, David Tarjan, Jeremy W. Sheaffer, Sang-Ha Lee, Kevin Skadron, *Rodinia: A benchmark suite for heterogeneous computing*, IEEE International Symposium on Workload Characterization, 2009

[4] Anthony Danalis, Gabriel Marin, Collin McCurdy, Jeremy S. Meredith, Philip C. Roth, Kyle Spafford, Vinod Tipparaju, Jeffrey S. Vetter, *The Scalable Heterogeneous Computing (SHOC) benchmark suite*, Proceedings of the 3rd Workshop on General-Purpose Computation on Graphics Processing Units, 2010