Functional Performance Models of Scientific Applications for Heterogeneous Multicore and Multi-GPU Systems

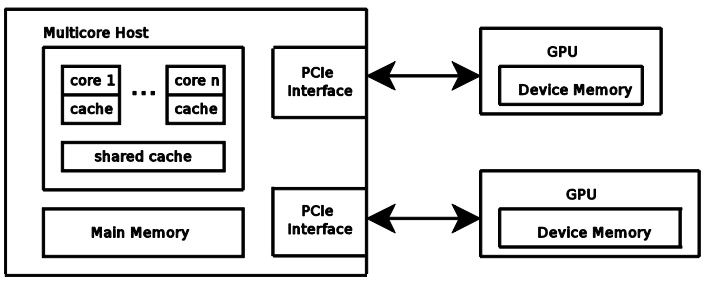
Ziming Zhong, Alexey Lastovetsky, Vladimir Rychkov
Heterogeneous Computing Laboratory
UCD School of Computer Science and Informatics

Introduction

Transition to hybrid CPU/GPU platforms in high performance computing is challenging in the aspect of efficient utilisation of the heterogeneous hardware and existing optimised software. We model the performance of parallel scientific applications in order to execute them efficiently on hybrid platforms. We apply FPM-based data partitioning [1] to balance the load between cores and GPUs. In our experiments, we couple the existing software optimised for multicores and GPUs and achieve high performance of the whole hybrid system.

Typical Hybrid Architecture

- The host and devices are connected via the PCI Express
- The host and devices have disjoint memory locations
- Explicit memory transfers are required for communication



Performance Measurement

The hybrid node executes a heterogeneous parallel application that invokes the libraries optimised for multicore and GPU respectively. We measure the speed of different parts of the system.

Multicore:

- Bind processes to cores
- Synchronise processes to minimise idle computational cycles
- Repeat experiments multiple times to ensure reliability of results

GPU:

- A CPU core is dedicated to a GPU
- Measure the combined performance of dedicated core and GPU
- Date transfer time is included

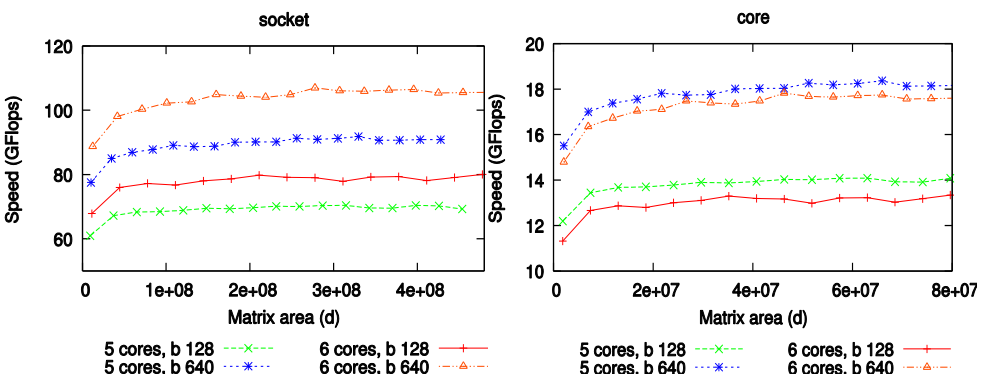
Testbed

Our experimental testbed is a hybrid multicore and Multi-GPU node, which consists of 8 six-core sockets and 2 different GPUs.

FPMs of Multicore

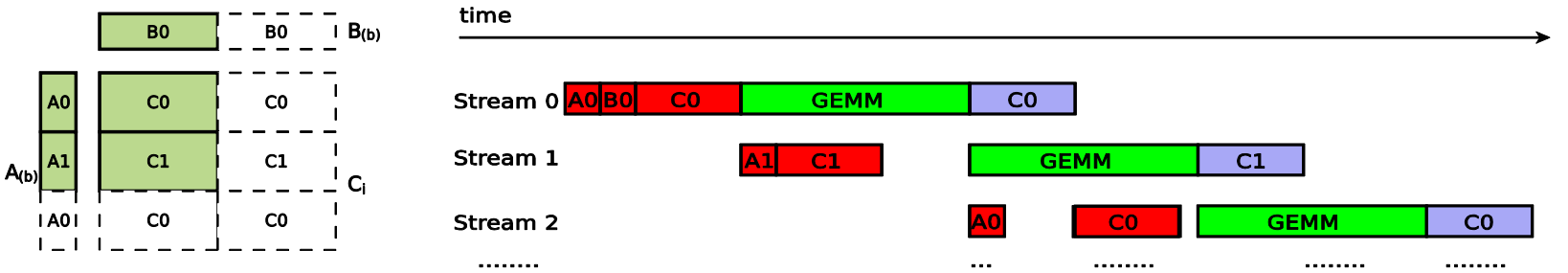
We build the functional performance models (FPM) of the parallel matrix-matrix multiplication [2] for multiple cores.

- The figure on the left side shows FPMs of a socket, executing the ACML kernel on 5 and 6 cores simultaneously, with blocking factor *b* as 128 and 640 respectively. The maximum performance of the socket is observed when all cores are involved in computations.
- The figure on the right side shows the speed of one core while the core was involved in the speed measurement of the socket.
- The higher performance of the socket and each core with the larger blocking factor can be explained by better exploited optimisations implemented in the ACML kernel, which take into account memory hierarchy.



Overlapping Communication and Computation

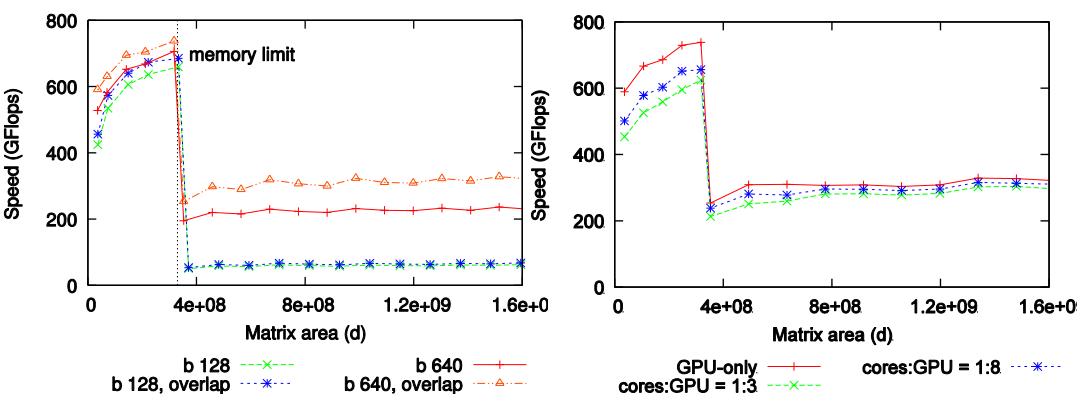
Techniques such as concurrent data transfers and overlapping communication and computation are used to improve the utilisation of limited PCIe bandwidth. The following example is a parallel matrix multiplication. The matrix partitioned to GPU does not fit in the local memory. We split the matrix into many small blocks and update the small blocks one by one. The sending and receiving of blocks and the computation on GPU are overlapped.



FPMs of GPU

We build the functional performance models (FPM) of the parallel matrix-matrix multiplication [2] for GPUs.

- The figure on the left side shows the FPMs of a GPU, with blocking factor *b* as 128 and 640 and with or without overlapping respectively.
- Before reaching to GPU memory limit, performance can grow up to 700 Gflops, then performance drop down after problem size cannot fit in the local memory.
- Large blocking factor and overlapping improve performance
- The figure on the right side show the FPMs of a GPU when CPU cores are running simultaneously. Performance decreases around 10% resulting from resource contention



FPM-based Data Partitioning

We experimented FPM-based data partitioning with heterogeneous matrix multiplication [2] on a hybrid node, which consists of 48 cores and two different GPUs. As we can see in the table, column 3 and 4 use FPM-based data partition and best performance is achieved when both CPU cores and GPUs are involved in computing.

Matrix Size	CPUs	GPUs	CPUs+GPUs
12800 x 12800	14.6s	10.5s	5.8s
19200 x 19200	43.4s	32.5s	16.2s
25600 x 25600	99.8s	147.9s	38.2s
32000 x 32000	189.2s	265.3s	114.1s

[1] Lastovetsky, A., Reddy, R.: Data partitioning with a functional performance model of heterogeneous processors. International Journal of High Performance Computing Applications 21, 76–90 (2007)

[2] Clarke, D., Lastovetsky, A., Rychkov, V.: Column-based matrix partitioning for parallel matrix multiplication on heterogeneous processors based on functional performance models. In: HeteroPar'11 (2011)

Acknowledgments. This publication has emanated from research conducted with the financial support of Science Foundation Ireland under Grant Number 08/IN.1/I2054.