A Survey of the Heterogeneous Computing Platform and Related Technologies

Fei LIU, Yi LIANG and Ling-ze WANG
Beijing University Of Technology, Beijing, China
*Corresponding author

Keywords: Heterogeneous computing, Key factors, Technology, Framework.

Abstract. A heterogeneous computing platform is an computing system that composed of different types of computing units. By fully using the computing ability of different types of computing units, the heterogeneous computing platform can achieve better performance and power efficiency than traditional homogeneous computing platform. In this paper, we firstly summarize and analyze the key factors that affect the performance of a heterogeneous computing platform. Next, we conduct a specific survey about these key factors from both software and hardware aspects and introduce some research results and key technologies at present. And then, we introduce the heterogeneous computing framework and make a comparison between OpenCL and HSA which are now both promising. At last, we analyze the future directions of the heterogeneous computing platform.

Introduction

A heterogeneous computing platform is a computing system that composed of different types of computing units. By fully using the computing ability of different types of computing units, the heterogeneous computing platform can achieve better performance and power efficiency than traditional homogeneous computing platform. Hence, the heterogeneous computing platform has become the hot topic in high performance computing area. The computing units in a heterogeneous computing platform are always heterogeneous, including CPU, GPU, DSP, FPGA, etc. Each computing unit utilizes itself as much as possible to improve the parallel efficiency of heterogeneous computing system.

Different types of computing units have different architectures and programming paradigms. The challenge of heterogeneous computing area is to concurrently utilize every computing unit efficiently. Since traditional technologies of homogeneous computing do not suitable for heterogeneous computing, in this paper, we conduct a survey of the development situation and related technologies of the heterogeneous computing platform. This research is aim to acknowledge researchers and scholars in this field the related technologies at present to give them some inspirations.

In a heterogeneous computing platform, heterogeneous hardware resources and the corresponding software support are necessary. From hardware level, heterogeneous processors are the most important in a heterogeneous computing platform. From software level, how the OS support the heterogeneous architecture of processor may significantly affect the performance of a heterogeneous system. Besides, technologies such as the design of algorithm and optimization of compiler also determine the performance of a system. Based on this, we conduct a specific survey about key factors that affect the performance of a heterogeneous computing platform from both software and hardware aspects and introduce some research results and key technologies at present. Also, we introduce the heterogeneous computing framework and make a comparison between OpenCL and HSA which are now both promising. At last, we analyze the future directions of the heterogeneous computing platform.

Key Factors That Influence the Performance of the Heterogeneous Computing Platform

A heterogeneous computing platform needs support from both hardware aspect and software aspect.
To improve the performance of heterogeneous computing platform, finding the key factors that influence it is necessary.

Factors From Hardware Level

**Technological Level.** With the continuous development of the transistor technology, the volume of the chip constantly decrease while the number of transistors increase. In 2007, Intel introduced the Penryn processor which adopt 45 nm technology, whose number of transistors can reach 820 million. In 2015, Intel introduced Broadwell processor which adopt 14 nm technology, whose number of transistor is more than 1.3 billion. With the development of technology level, the transistor density is increasing and some problem such as consumption occurs while the performance of system improves.

**Number of Cores.** The number of cores can largely determine the performance of a system. With proper software support, the system with more core number can achieve better computing efficiency. Computer system experience the process from single core to multiple cores. Nowadays, in some giant server the core number of a system can achieve more than hundreds, such as the CUDA number of GeForce GTX is 240. In future, the number of system will stay increasing to thousands. While the number of system also affects other aspects of a system, and the performance will not stay increasing, instead, decrease when the core number exceeds some certain value. Hence, the choose of the number value should combine factors from many aspects.

**Cache Technology.** The key elements that cache influence performance mainly include cache size and cache replacement strategy. Cache size is very important to cache performance. If the cache size is big, the cache hit rate is high and the time of memory access is short, and that is what we want. When cache size exceeds some certain value, although cache hit rate is very high, the large amount of data to be searched will reduce the search speed. Cache replacement also influences cache efficiency, a appropriate cache replacement strategy can improve cache efficiency largely. Some common cache replacement strategies are based on access interval, access times, size of memory use ,etc.

**Communication between Cores.** The speed that different cores communicate also determine the performance of a system, and the speed of the data transmission is determined by bandwidth. Traditional Multi-core devices always communicate through PCI-e BUS which provides low speed and the communication efficiency is not high enough. In 2014, NIVIDA push out a GPU of Pascal architecture which adopts NVLink BUS instead of PCI-e BUS, whose bandwidth is several times higher than PCI-e 3.0BUS. The efficiency of communication between cores is improved.

**Energy Consumption.** When comparing two systems that spend the same time to deal with the same task, the one that consume less energy is better than another. When holding the purpose of improve system efficiency, reducing energy consumption is also important. So the tradeoff between efficiency and energy consumption is necessary.

Factors from Software Level

**The Partition of Task.** The partition of task can be based on the characteristic of PU and the nature of task. When the partition is based on the characteristic, improving the utilization of each PU is the final purpose. If some PU is idle during execution period, the idle resource cannot be fully utilized and this will cause some waste. When the partition is based on the nature of task, the nature of task should be taken into consideration to reduce the overhead of communication between cores, e.g. if the execution process are consisted of several stages, then we can divide the task by stage so that the large amount of communication overhead between different cores can be avoided.

**Dynamic or Static Scheduling.** The operating system for task scheduling can be divided into static scheduling and dynamic schedule. If static scheduling is adopted, the scheduling of all subtasks are determined before task execution and the scheduling solution is fixed and will not change with the current state of system. However, for dynamic scheduling, the scheduling solution is determined at runtime according to the current system status and the characteristics of the task itself. Compared to static scheduling, dynamic scheduling can always offer better performance.
Survey of the Heterogeneous Computing Platform From Hardware Level

Existing Heterogeneous Architecture

**CEBA.** CBEA architecture[1] is consisted of one CPU and 8 SIMD cores, in which every core have strong execution efficiency. These cores are connected with bus with high bandwidth so that they can work together as well as separately. The parallel memory between cores makes the high bandwidth can be fully used. Although CBEA architecture has quite high theoretical bandwidth, there are still some limitations, such as the parallel problem, programmability problem and performance problem. There are still a lot of room to improve for CBEA.

**CPU+GPU.** CPU+GPU architecture[2] is the most common heterogeneous architecture. It can be divided into fused CPU+GPU and discrete CPU+GPU. In discrete system, CPU has dependent memory space to GPU, and there are a large amount of overhead such as communication time and memory bandwidth to communicate between CPU and GPU which make it not efficient enough. In fused system, CPU can share memory space with GPU and the communication overhead is avoided. Even though there are some extra overhead of maintaining memory consistency for fused system, the efficiency is better than discrete system overall.

**FPGA +GPU.** FPGA +GPU architecture[3] is the collaboration of FPGA and GPU. FPGA chip is suitable for dealing with task of pipeline and GPU is suitable for the task of high parallelism. Therefore FPGA+GPU can provide good performance. However there are not a technology to communicate between FPGA and GPU directly at yet, and hence an alternative is to communicate through PCI-e bus. According to some research about this, this way to communicate can provide high speed data transmission with very low latency which can close to the theoretical peak.

**MIC.** MIC architecture[4] is designed for high performance computing. In this architecture, a chip with many cores are connected to CPU host through PCI-e bus as a co-processor, and this co-processor has dependent space to host. A co-processor includes several process cores, high speed cache, memory controller etc. To make MIC achieve the best performance, some limitations is to be taken into consideration, e.g. highly scalability is necessary to enable task execute on multiple cores system and the communication latency should be concealeed.

Related Technologies from Hardware Level

**Core Number Based Technologies.** A research[5] discuss the influence of core number on system performance. Their experiment shows with the increase of core number, performance of system increase at the very start and present a downward trend after a certain point. They point out that the choose of core number should take various factors into consideration instead of the viewpoint that the more cores the better performance.

**Energy Consumption Based Technologies.** A research[6] present a hybrid architecture to reduce power consumption. They point out that energy consumption is the bottleneck of scalability in system. They conduct experiments on four different architectures which are single CPU, CPU+FPGA, CPU+GPU and CPU+GPU+FPGA respectively and the result shows the last one can achieve the lowest energy consumption.

A research[7] present a technology of Dynamic Voltage And Frequency Scaling to save energy consumption. It is consisted of two parts, one is the static analysis stage to analyze the application model and performance model, another is the problem solved stage include task partition and scheduling. This technology can reduce energy efficiency.

A research[8] point out that the system energy consumption includes one fixed part and another variable part. They adopt Dynamic Voltage And Frequency Scaling to reduce the variable part to achieve energy saving.

A research[9] present a technology of dynamic cache redistribution. They find out a cache scheme that consume the lowest energy and configure cache, then the spare cache is shut down to save energy.

**Communication Between Cores Based Technologies.** A research[10] introduce a wireless connection technology called WiNOC which can solve the high latency and low bandwidth of
traditional wire connection technology. Without complex interconnect, WiNOC has simple layout and is convenient in applying.

A research[11] present an optimal NOC design. The design includes topological synthesis stage and analytic statistic stage. Their experiment shows their design can achieve more 16% performance over standard mesh structure.

A research[12] present a new framework to manage communication. Their new framework retain shared memory and add a set of ISA of hardware implementation to accelerate the communication. It is suitable for applications that need lots of communication.

**Cache Based Technologies.** A research[13] present a cache predict technology to avoid moving in useless data. They make predictions using history records. Although there is the case that mistakenly predict, the negative impact of cache efficiency is negligible.

A research[14] point out that some frequently used data are not taken into hotspots cache, and hence they present a new cache framework that introduce hotspots cache and filter cache between L1 cache and memory at the same time. Their experiment shows the cache efficiency is improved.

---

**Survey of The Heterogeneous Computing Platform From Software Level**

**Existing Heterogeneous OS**

**Barrellfish.** Giving the limitations of the shared memory model to system’s scalability, Microsoft research Cambridge and ETH Zurich develop a new OS model called multikernel. This model abandoned the mechanism of shared memory and adopt messages to replace it and this makes the overhead of extending to more cores less compared to shared memory. Barrellfish is implemented according to multikernel model.

**FOS.** FOS is developed for system with lots of cores by Massachusetts Institute of Technology. FOS is designed based on distributed thought and aiming at extending to hundreds of cores. There are still limitations exists, e.g. the handle of service in FOS should be low latency and the consistency of namespace should be ensured.

**Corey.** Corey is developed by Massachusetts Institute of Technology, and it gets inspiration from Exokernel. Too much shared data may influence the performance of system and the scalability may be influenced by some useless share. Corey put forward three methods to improve the scalability, they are address range, kernel cores and shares respectively. Corey achieves the spatial multiplexing among many cores and makes performance improved.

**Related Technologies from Software Level**

**Task Partition Based on the Characteristic of PU.** A research[15] present a dynamic load balancing library that suitable for kinds of problems in heterogeneous computing area. They also present a dynamic task partition scheme and this makes parallel tasks execute in system balancing. Their result has been used in many heterogeneous systems including single GPU and multiple GPU system.

A research[16] present a method to achieve load balancing. They divide task according to traditional scheme firstly and then achieve load balancing by work-steal algorithm. To avoid the task migration between cores, large size task are always handled by GPU. Their experiment shows by avoiding GPU to handle small size task can achieve better performance than other ways.

A research[17] present a SKMD system to improve the utilization of available resources. They can divide task and dispatch them to specified PU automatically and the returned results from different PU can be integrated by using this system. Their experiment shows SKDM achieves better performance than common systems.

**Task Partition Based on the Nature of Task.** A research[18] take the nature of tasks into consideration to divide tasks. They find that parts of NRR algorithm are regular and the spare are irregular, and hence they dispatch the regular ones to GPU and the irregular ones to CPU, which fully utilize the high parallelism of GPU.
A research[19] presents an optimal method to dynamically divide tasks. They implement an
compiler to transfer codes in homogeneous platforms to codes that available in heterogeneous
platforms. They also present a performance predict model that using ML technology to divide tasks
dynamically.

A research[20] present a technology to divide and dispatch tasks. If a task cannot be dispatched to
a core to execute, it will be divided into several subtasks and be dispatched to several different cores
concurrently. The tasks cannot be migrated to other core until completion of the execution and the
overhead of migration is saved.

**Static/Dynamic Scheduling Based Technologies.** A research[21] develops a new algorithm
about dense linear algebra problem. They replace the traditional scheduling method with optimal
asynchronous dynamic scheduling and this enables codes execute on several cores concurrently.
Besides, the algorithm fully utilize the idle cores and improve the efficiency effectively.

A research[22] present a dynamic scheduling scheme to achieve load balancing. They evaluate
performance of every PU dynamically at adaptive stage and take it as a basis of task partition. They
found a proper block size can maximize the utilization.

A research[23] present a dynamic scheduling scheme to improve the performance of
heterogeneous computing platforms. They evaluate the time to complete the current task and predict
the time to handle the new task according to history record. And then they can choose a best scheme
to schedule.

**Heterogeneous Computing Framework (Comparison between OpenCL and HAS)**

As a industrial standard, OpenCL is not only a kind of program language, but also a computing
framework includes API, runtime and many libraries. OpenCL framework has four model, they are
platform model, execution model, memory model and programming model respectively. OpenCL is
now widely accepted and used in heterogeneous computing area, and there are many mature support
tools exists. For the good performance OpenCL provide, it is very likely to not be replaced by other
framework in the near future. Besides, the addition of new feature of OpenCL 2.0 makes it also
available in mobile platform.

HSA is a framework that support for various different heterogeneous platforms. With the release
of HSA 1.0 specification, many famous vendors denote to support for the HSA architecture
specification and this means codes will achieve good portability between platforms which support
HSA specification. Although the HSA related technologies are not mature enough, but the promise
to solve the portability issue caused by the increasing of cores number makes HSA may become the
mainstream framework in the future.

**The Future Directions**

The heterogeneous computing is universal to large scale complex problems in many areas. Since
most of these problems have kinds of different calculation requirements, solving thus problems by
heterogeneous computing model is an effective pattern.

Compared with the homogeneous computing model, the complexity of the design of
heterogeneous computing model itself makes lots of efforts needed to be placed in. And as an new
technology, the related support tools of heterogeneous computing remain to be strengthened. All of
these are the challenges faced by heterogeneous computing model. With the rapidly increasing of
the number of cores and the diversity of processor units, how to improve the efficiency of system at
the same time as reducing energy consumption and keeping portability of codes becomes an
important issue to be solved.

**Acknowledgments**

We are very grateful to the anonymous reviewers. This work is supported by the NSFC project
(Grants No.61202075), the Chinese 863 project (Grants No. 2015AA016002), the National Science
and Technology Major Project (Grants No.2012ZX01039-004), and the BNSF project (Grant No.4133081).

References

[1] Poremba M, Yuan Xie, Wolf M, Accelerating adaptive background subtraction with GPU and CBEA architecture, In IEEE Workshop on Signal Processing Systems (SIPS), pp.305-310, 2010

[2] Tayler H, Timothy G, Lisa, Mike, TorM, Characterizing and evaluating a key-value store application on heterogeneous CPU-GPU systems, In ISPASS, pp.88–98, 2012

[3] Thoma Y, Dassatti A, Molla D, FPGA2: An open source framework for FPGA-GPU PCIe communication, In International Conference on Reconfigurable Computing and FPGAs, pp.1-6, 2013

[4] Peng Hui Yao, Yong Hong Hu, Improved Parallel Randomized Quasi Monte Carlo Algorithm of Asian Option Pricing on MIC Architecture, In DCABES, pp.157-161, 2014

[5] Hong Jun Choi, Dong Oh Son, Cheol Hong Kim, Jong Myron Kim, Impact of Clock Frequency and Number of Cores on GPU Performance, International Conference on IT Convergence and Security (ICITCS), pp.1-4, 2014

[6] Tao Li, JohnL K, OS-aware tuning: Improving instruction cache energy efficiency on system workloads, In Performance, Computing, and Communications Conference, pp.10-33, 2006

[7] Guibin Wang, Wei Song, Communication-aware Task Partition and Voltage Scaling for Energy Minimization on Heterogeneous Parallel Systems, In 12th International Conference on Parallel and Distributed Computing, Applications and Technologies, pp.327-333, 2011

[8] GuibinWang, Xiaoguang Ren, Power-efficient Work Distribution Method for CPU-GPU Heterogeneous System, In International Symposium on Parallel and Distributed Processing with Applications, pp.122-129, 2010

[9] Sparsh Mittal, Yanan Cao, Zhao Zhang, MASTER: A Multicore Cache Energy-Saving Technique Using Dynamic Cache Reconfiguration, pp.1653-1665, 2013

[10] Sujay Deb, Amlan Ganguly, Partha Pratim Pande, Benjamin Belzer, Deukhyoun Heo, Wireless NoC as Interconnection Backbone for Multicore Chips: Promises and Challenges, IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 2(2), pp.228-239, 2012

[11] Bharath Phanibhushana, Sandip Kundu, Network-on-Chip Design for Heterogeneous Multiprocessor System-on-Chip, In IEEE Computer Society Annual Symposium, pp.486-491, 2014

[12] Farrukh Hijaz, Brian Kahne, Peter Wilson, Omer Khan, Efficient Parallel Packet Processing using a Shared Memory Many-core Processor with Hardware Support to Accelerate Communication, In IEEE International Conference on Networking, Architecture and Storage, pp.122-129, 2015

[13] Pujara P, Aggarwal A, Increasing the cache efficiency by eliminating noise, In The Twelfth International Symposium on High-Performance Computer Architecture, pp.145-154, 2006.

[14] Kashif Ali, Mokhtar Aboelaze, Suprakash Datta, Modified Hotspot Cache Architecture: A Low Energy Fast Cache for Embedded Processors, In International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation, pp.35-42, 2006

[15] Alejandro Acosta, Robert Corujo, Vicente Blanco, Francisco Almeida, Dynamic load balancing on heterogeneous multicore/multi-GPU systems, In International Conference, pp.467–476, 2010

[16] Everton Hermann, Bruno Raffin, Francois Faure, Thierry Gautier, Multi-GPU and multi-CPU parallelization for interactive physics simulations, In Euro-Par conference, pp.235–246, 2010
[17] Janghaeng Lee, Mehrzad Samadi, Yongjun Park, Scott Mahlke, SKMD: Single Kernel on Multiple Devices for Transparent CPU-GPU Collaboration, ACM Transactions on Computer Systems (TOCS), 33(9), pp.9-35, 2015

[18] Yixun Liu, Andriy Fedorov, Ron Kikinis, Nikos Chrisochoides, Real-time non-rigid registration of medical images on a cooperative parallel architecture, In IEEE International Conference on Bioinformatics and Biomedicine, pp.401–404, 2009

[19] Klaus Kofler, Ivan Grasso, Biagio Cosenza, Thomas Fahringer, An Automatic Input-Sensitive Approach for Heterogeneous Task Partitioning, In 27th international ACM conference on supercomputing, pp.149-160, 2013

[20] Saranya N, Hansdah R C, Dynamic Partitioning Based Scheduling of Real-Time Tasks in Multicore Processors, In IEEE 18th International Symposium on Real-Time Distributed Computing(ISORC), pp.190-197, 2015

[21] Horton M, Tomov S, Dongarra J, A Class of Hybrid LAPACK Algorithms for Multicore and GPU Architectures, In High-Performance Computing (SAAHPC) Symposium, pp.150-158, 2011

[22] Mehmet E, Laxmi N, Rajiv Gupta, A dynamic self-scheduling scheme for heterogeneous multiprocessor architectures, ACM Transactions on Architecture and Code Optimization Special Issue, ACM Transactions on Architecture and Code Optimization (TACO) - Special Issue on High-Performance Embedded Architectures, 9(4), pp.57-76, 2013

[23] Hong Jun Choi, Dong Oh Son, Seung Gu Kang, Jong Myon Kim, Hsien-Hsin Lee, Cheol Hong Kim, An efficient scheduling scheme using estimated execution time for heterogeneous computing systems, The Journal of Supercomputing, 65(2), pp.1–17, 2013