Heterogeneous Memory System Framework for HPC

Kazi Asifuzzaman, Narasinga Rao Miniskar, Aaron R. Young, Frank Liu, Jeffrey S. Vetter
Oak Ridge National Laboratory, Oak Ridge, TN, USA
{asifuzzamank,miniskarnr,youngar,liufy,vetter}@ornl.gov

1 Challenge

The increasing demand for computational performance along with the slowdown of Moore's law results in the need for extremely heterogeneous systems with new accelerator designs, memory technologies, and memory hierarchies to continue improving application performance and reduce energy consumption. In the current environment, applications are written with the static mapping of their memory objects based on the kernel placement on the specific computing resource (CPU, GPU, FPGA) decided at design time. It is a challenge to provide an environment to the developer with application performance portability targeting extremely heterogeneous systems which have not only heterogeneous accelerators but also have heterogeneous memories. Recent state-of-the-art techniques have proposed either the run-time mapping of tasks to heterogeneous computing units with the static mapping of memory objects [1], [2], or have proposed the run-time selection of memory without considering task mapping constraints [3]. A Heterogeneous memory system framework is needed to address the performance and portability of applications for extreme heterogeneous systems, which studies the traditional memory mapping techniques, task mapping techniques and proposes a run-time adaptable heterogeneous memory mapping approach to work in conjunction with the heterogeneous task mapping approach. It is challenging to address this problem in multiple areas such as programming model, compiler support, and run-time framework.

There are some studies that partially address this issue. Narayan et al. [3] propose a novel page allocation approach to utilize heterogeneous memory systems at the memory object level, which conducts profiling and classification of memory objects offline. Wu et al. [4] propose a lightweight runtime solution called Unimem to minimize unnecessary data movement. Unimem works in phases of profiling, modeling, and placement of memory objects, and the concept is evaluated with non-cycle accurate simulation. Olson et al. [5] extend the application run-time layer with automated monitoring and implement an online data tiring solution that facilitates data allocation and placement across heterogeneous memory only based on the latency and if there is a benefit to migrating data. Although these studies provide great ideas and insights to approach the issue, a detailed and dedicated exploration is needed in order to develop a complete solution for accommodating future heterogeneous memory systems that can effortlessly accommodate a range of memory technologies.

2 Opportunity

Applications can have very different memory requirements. Research is required to profile applications to understand their memory requirements when determining how the application can best leverage the available memory. For example, an application can be latency-sensitive, or bandwidth-bound; while DDRx DRAM (still) provides the fastest interface but High Bandwidth Memory (HBM) provides superior bandwidth performance. There could be a need for low power (LPDDR) or Non-Volatile (STT-MRAM, ReRAM, PCM) memory for a specific application as well. Therefore, there would be a great opportunity to analyze a broad range of applications from a memory requirement perspective and this research can lead to interesting insights.

Concurrently, there would also be an opportunity to conduct elaborate analysis of candidate memory technologies, many of which are novel/emerging and lack a detailed understanding of underlying technologies and concepts. A detailed exploration of these memory technologies would certainly enrich the community with useful knowledge.

The performance and portability challenges of heterogeneous memories need to be addressed with an extension to existing programming models [2] with unique input/output (IO) objects specification of computational kernels along with the task specification. The IO object's specification should have sufficient details for dynamic mapping. Researchers have to balance the details and gains. It also provides opportunities to support the specification with compiler (LLVM) extensions. Researchers have an opportunity to investigate dynamic mapping techniques for heterogeneous memories which include heuristics (genetic, dynamic programming, etc.) and machine learning techniques, and consider trade-off axes such as performance, energy consumption, complexity, memory usage, and bandwidth requirement.

DOE applications are expected to fully exploit the advantages of the proposed heterogeneous memory system framework with both performance gains and portability to variations of the emerging extremely heterogeneous systems. A efficient heterogeneous memory system would effortlessly facilitate allocation and mapping to memory sub-modules during run-time, while maintaining coherence, and preferably without placing a large burden on the programmer.

3 Timeliness

The end of simple technology scaling for easy performance gains along with the rise in diversity of computation accelerators and memory systems are resulting in extremely heterogeneous computing systems. New emerging memory technologies like ReRAM, SST-RAM, PCM, HBM, and HMC have matured and are now being included within accelerator designs. For example, FPGA designs are now including HBM and SSD attached memories. The open-source movement further transformed the emergence of new processing cores (RISC-V), computing accelerators (artificial intelligence, neuromorphic, HPC) along with the integration of new memory technologies. In response, we have already seen the emergence of true heterogeneous commercial platforms incorporating CPUs, GPUs, accelerators, and FPGAs each potentially including diverse sets of memory. Although static mapping of tasks to accelerators and static mapping of memory to memory devices has been researched before, we are now facing a new challenge of maintaining performance portability as codes are ported between heterogeneous systems with different accelerator and memory structures. Advanced dynamic mapping of both the accelerator and memory systems will be needed to maintain portability and performance in these increasingly heterogeneous and diverse systems without creating a large burden on the programmers to write system-specific code.

References

[1] Ruyman Reyes and Victor Lomüller. "SYCL: Single-source C++ accelerator programming". In: *Parallel Computing: On the Road to Exascale*. IOS Press, 2016, pp. 673–682.

[2] Jungwon Kim, Seyong Lee, Beau Johnston, et al. "IRIS: A portable runtime system exploiting multiple heterogeneous programming systems". In: *2021 IEEE High Performance Extreme Computing Conference (HPEC)*. IEEE. 2021, pp. 1–8.

[3] Aditya Narayan, Tiansheng Zhang, Shaizeen Aga, et al. "MOCA: Memory Object Classification and Allocation in Heterogeneous Memory Systems". In: *2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. May 2018, pp. 326–335. DOI: 10.1109/IPDPS.2018.00042.

[4] Kai Wu, Yingchao Huang, and Dong Li. "Unimem: Runtime Data Managementon Non-Volatile Memory-Based Heterogeneous Main Memory". In: *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. SC '17. Denver, Colorado: Association for Computing Machinery, 2017. ISBN: 9781450351140. DOI: 10.1145/3126908.3126923. URL: https://doi.org/10.1145/3126908.3126923.

[5] M. Ben Olson, Brandon Kammerdiener, Kshitij A. Doshi, et al. *Online Application Guidance for Heterogeneous Memory Systems*. 2021. arXiv: 2110.02150 [cs.PF].