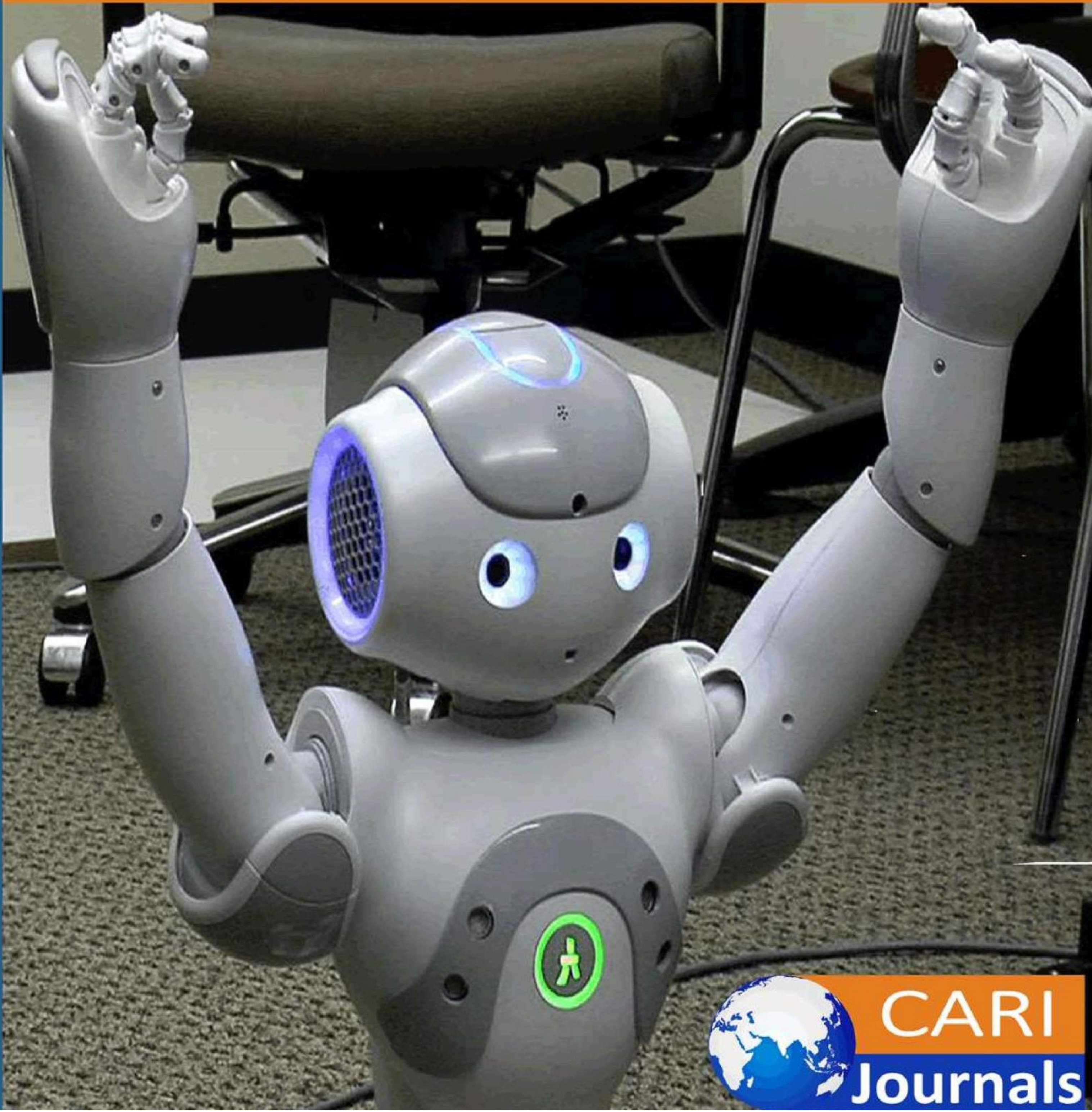
International Journal of

Computing and Engineering

(IJCE)

A Systematic Literature Review on Graphics Processing Unit Accelerated Realm of High-Performance Computing



CARI Journals

International Journal of Computing and Engineering
ISSN 2958-7425 (online)
Vol. 5, Issue No. 3, pp. 10 - 21, 2024 www.carijournals.org

A Systematic Literature Review on Graphics Processing Unit Accelerated Realm of High-Performance Computing

Rajat Suvra Das

[1*] Senior Director, Business Development

L&T Technology Services

https://orcid.org/0009-0007-0446-0599

Accepted: 18th Mar 2024 Received in Revised Form: 4th Apr 2024 Published: 18th Apr 2024

Abstract

GPUs (Graphics Processing Units) are widely used due to their impressive computational power and parallel computing ability.It have shown significant potential in improving the performance of HPC applications. This is due to their highly parallel architecture, which allows for the execution of multiple tasks simultaneously. However, GPU computing is synonymous with CUDA in providing applications for GPU devices. This offers enhanced development tools and comprehensive documentation to increase performance, while AMD's ROCm platform features an application programming interface compatible with CUDA. Hence, the main objective of the systematic literature review is to thoroughly analyze and compute the performance characteristics of two prominent GPU computing frameworks, namely NVIDIA's CUDA and AMD's ROCm (Radeon Open Compute). By meticulously examining the strengths, weaknesses, and overall performance capabilities of CUDA and ROCm, a deeper understanding of these concepts is gained and will benefit researchers. The purpose of the research on GPU accelerated HPC is to provide a comprehensive and unbiased overview of the current state of research and development in this area. It can help researchers, practitioners, and policymakers understand the role of GPUs in HPC and facilitate evidence-based decision making. In addition, different real-time applications of CUDA and ROCm platforms are also discussed to explore potential performance benefits and trade-offs in leveraging these techniques. The insights provided by the study will empower the way to make well-informed decisions when choosing between CUDA and ROCm approaches that apply to real-world software.

Keywords: *Compute Unified Device Architecture, Graphics Processing Unit, High-Performance Computing, Performance Analysis, Radeon Open Compute.*

1. INTRODUCTION

With advancements, GPU (Graphics Processing Unit) is becoming crucial to availing enhanced processing power for HPC (High-Performance Computing). In that way, the rapid development of GPU technologies has enhanced the high memory bandwidth in data parallelism and strong floating-point capability. With this aspect, this technology is widely used in HPC applications and performs computing of huge processing simulations with a reasonable time and resources. Despite this, various sectors require increased computational power to perform execution for balancing power and energy consumption (Lai, Yu, Tian, & Li, 2020). This task is considered challenging, as performance, efficiency, and power are the conflict criteria. As per reports released on June 2023, the 61st edition of the top 10 list shows that 80% of advanced supercomputers involve heterogeneous software computing platforms ("JUNE 2023," 2023).

CUDA (Compute Unified Device Architecture) is employed with GPU as a programming model and parallel computing platform. It has been developed by NVIDIA, in which CUDA tends to achieve high performance and makes the programming model manageable to developers of HPC. CUDA is designed to deliver a scalable programming module using intellections for the hierarchy of barrier synchronization, thread groups, and shared memories. In addition, to avail clear mappings with underlying hardware, CUDA applies kernels of one, two, or three-dimensional blocks of threads. Besides, the memory is managed using device memory allocation, deallocation, and data transfer abilities (HOMERDING & TRAMM). Table 1 discusses the evolution of different NVIDIA GPU architecture along with their specifications. NVIDIA's H100 combined technology speeds up LLM (Large Language Model) to offer industry-leading conversational AI. It also maintains low latency in power constrained data center circumstances (NVIDIA, 2024).

Table 1 Development of NVIDIA GPU Designs (Hecht, Brendel, Wessels, & Bernhard, 2021) (Vanderbauwhede & Takemi) (Hashmi, Ayele, Naik, & Keskar, 2022)

Models	**GeForce GTX 690**	**GeForce GTX 590**	**GeForce GT 320**
Micro Architecture	Kepler	Fermi	Tesla
Memory Clock (GHz)	6.0	1.7	0.8
Memory Bandwidth (GB/Sec)	384	327.7	25.3
Number of Processors	3072	1024	72
Processor Clock (GHz)	1.0	1.2	1.3
Global Memory Size (GB)	4	3	1

The ROCm (Radeon Open Compute Platform) is considered as a software platform, which was developed by AMD (Advanced Micro Devices) in 2015 for hyperscale GPU computing and HPC. It is an open-source platform that enables developers to utilize AMD GPUs for their computing needs. ROCm is similar to NVIDIA's CUDA platform, widely used with NVIDIA GPUs. The ROCm software platform utilizes AMD's HIP interface. This interface allows developers to execute programs on AMD GPUs. Overall, the programming languages and code structures are similar, while most of the codes are converted directly by replacing "cuda" with "hip". Similar to AMD GPU, developers can also utilize the ROCM-HIP programming interface to involve programs running on a GPU accelerator (Cao et al., 2024). The Figure 1 shows the latest performance of AMD.

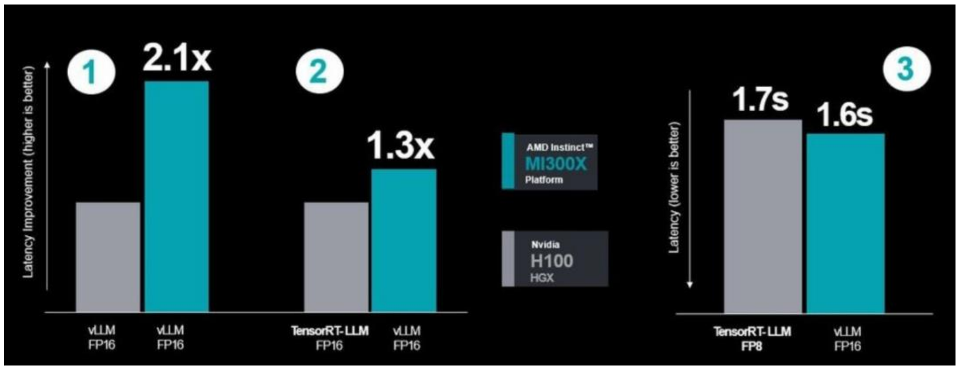


Figure 1 Inference Performance on LIama 270B (Andreadis, 2024)

Recently, AMD launched its new flagship AI GPU and accelerator namely MI300X and declared that MI300X has performed better when compared with NVIDIA’s H100 GPU.

1.1 Aim and Objective

- To investigate GPU in HPC by comparing the performance of NVIDIA's CUDA and AMD's ROCm.
- To focus on various aspects such as performance, scalability, compatibility, and applicability in the realm of HPC in CUDA and ROCm.
- To explore different application scenarios of CUDA and ROCm to achieve energy efficient trade-off.
- To analyze the performance gap and future works on enhancing the performance of both CUDA and ROCm.

1.2 Paper Organization

Initially, the performance analysis of the CUDA and ROCm platform is reviewed in the introduction section, along with the review methodology in Section 2; the conventional approaches employed in the performance analysis of CUDA and ROCm are conferred in Section 3. Different application scenarios of CUDA and ROCm are deliberated in section 4. In Section 5, research gaps and future developments are also discussed, and the review is concluded in Section 6.

2. REVIEW METHODOLOGY

The systematic literature review is a fundamental scientific analysis that gathers a wide range of research to interpret research outcomes comprehensively. Initially, the study begins with a search plan across the entire Google Scholar database using specific keywords such as "high performance computing," "Graphics Processing Unit," "NVIDIA's CUDA," "AMD's ROCm," "parallel computing," and "heterogeneous computing." The articles considered are those published from 2019 to 2024, incorporating the latest advancements in the field. The study focuses on gathering articles that reflect the current insights and showcase improved performance systems. The study follows the PRISMA guidelines to ensure transparency in the analysis process. To identify relevant research papers, a thorough search is conducted using a database called Google Scholar. Google Scholar is an open-access website that provides access to various reliable sources related to the research concept, including article citations. Further, it also serves as a valuable search engine for exploring a vast amount of intellectual materials.

Step 1 Search Strategy

The research study begins by thoroughly searching various databases and carefully selecting the most relevant sources and appropriate search terms. One of the key databases utilized in this process is Google Scholar, which provides a comprehensive collection of published research findings. Additionally, the search methodology employed in the study focuses on important aspects, such as the enhanced performance analysis of NVIDIA's CUDA and AMD's ROCm.

Step 2 Inclusion and Exclusion Conditions

The aspects on which the studies are included and excluded for investigating the performance analysis of the CUDA and ROCm platforms are discussed in this section. Table 2 represents the significant factors in choosing the appropriate studies.

Table 2 Inclusion and Exclusion Criteria

Significant Factors	Inclusion	Exclusion
Years	Papers available from 2019 to 2024	Papers available before 2018
Review Method	Relevant studies that analyze the concepts of HPC GPU based on CUDA and ROCm	Studies relevant to the method, in which the article quality is clogged, other than CUDA and ROCm.
Modality	Articles with CUDA and ROCm based approaches.	Articles without CUDA and ROCm based techniques.

Based on the conditions of inclusion and exclusion, the present study chooses the articles and follows as per criteria as shown below,

Inclusion Criteria

The studies are encompassed, if the concepts fulfills the standards as follows:

- The studies must be more relevant to the present study objectives and applications.
- Studies preferred must be of recent research papers (2019 to 2024).

Exclusion Criteria

The exclusion of studies follows a set of criteria as outlined below,

- Studies that contain duplicated content from previous research works are not considered.
- Studies without clear research objectives and research questions are excluded.
- Papers presented in languages other than English are not eligible for acceptance.

Step 2.4 Quality Evaluation

The appropriateness of the concept area is focussed on accessing the inclusion and exclusion conditions, and about 16 references are used by considering the significant constraints, as discussed in Table 2. The evaluation is conducted by considering three categories of quality assessment inquiries, as presented below.

Q1: Does the paper cover related research study and explore research topics widely?

Q2: Does the paper discusses clear inference with acceptable outcome and inferences?

Q3: Does the paper provides with limitations and future developments?

The selected articles should have an answer “yes” to quality evaluation questions, and based on this aspect, almost 15 papers tend to satisfy the criteria.

Step 2.5 Data Analysis

The chosen papers were examined and evaluated according to their focus, summary of research inquiries and corresponding findings, and the variety of subject matter. The articles accumulated are classified into various performance evaluations and embedded applications involved in the analysis of CUDA and ROCm.

3. PERFORMANCE ANALYSIS OF CUDA AND ROCm

To harness the computational power of diverse processor designs, it is essential to employ advanced parallel programming models. In this regard, a compiler transformation has been conducted to convert OpenMP code into CUDA graphs, enabling efficient utilization of structured and unstructured parallelism (Yu, Royuela, & Quiñones). It has incorporated the advantages of programmability of a high-level programming platform. Moreover, the implementation has been performed on two different applications, Single Precision AX Plus Y, Saxpy, and Cholesky decomposition. By evaluating the programmability and performance of the system, the data indicates that utilizing CUDA graphs instead of OpenMP offloading for individual CUDA kernels has significantly improved execution time. As a result, the OpenMP offloading version has been significantly slower when compared to the automatically generated CUDA graph version.

Additionally, the time spent on synchronization is minimized when using CUDA graphs, whereas it increases with OpenMP. GPUs with a higher number of blocks also tend to execute CUDA programs more quickly compared to GPUs with fewer blocks (Bai, Liu, Wu, & Feng, 2021). Figure 2 illustrates the hierarchy of CUDA kernels and memory.

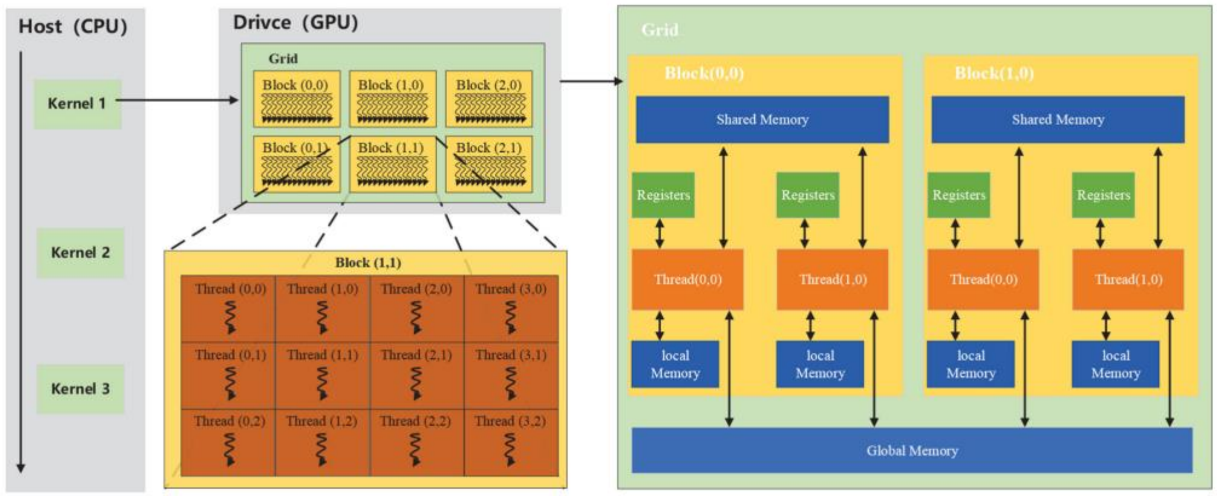


Figure 2 Kernel and Memory Hierarchy of CUDA (Bai et al., 2021)

Figure 1 shows that threads within a block are divided into smaller groups called wraps, which are used for computation on processors. Specifically, CUDA acknowledges that the CUDA kernel has been executed on the GPU (drive) while the remaining part of the C program has been computed on the CPU (host). Additionally, CUDA accesses data from various memory hierarchies, with each thread having its own local memory and private register. Within the same thread block, all threads share a common memory known as shared memory, which is visible to all threads. Consequently, global memory can be utilized by all threads. The use of RTL (Register-Transfer Level) simulation is considered a critical phase in designing and verifying highly complex accelerators, processors, and SOCs. The RTL simulation is done by performing partitioning of heuristics or event-driven techniques (Lin, Ren, Zhang, Khailany, & Huang, 2022). GPU-accelerated RTL simulation flow and batch stimulus have been developed to improve throughput performance. Initially, the RTL flow compiles RTL into CUDA kernels, in which each stimulates a partition of RTL. In addition, pipeline scheduling and CUDA graphs have also been used to improve the runtime execution.

The conventional hardware profile fails to be relevant in constructing roofline models for AMD GPUs, primarily because of the divergence in terminologies and network architecture between NVIDIA and AMD GPUs. However, by incorporating metrics from AMD's ROCProfiler, a comprehensive instruction roofline model specifically tailored for AMD GPUs has been devised (Leinhauser et al., 2022). The application's performance is measured in-memory transactions and instructions, and the BabelStream approach has been implemented over AMD hardware. For case study scientific applications, instruction roofline models have been developed, and an application called PIC (Particle-in-Cell) simulation application has also been utilized for laser-plasma and plasma physics on AMD Instinct MI100, AMD Radeon Instinct MI60, and NVIDIA V100 GPUs. From execution, it has been analyzed that multiple kernels in PIConGPU achieve similar results as AMD MI100 in terms of execution time. The ROCm platform is considered to be largely analogous to the CUDA platform and openly composed of several modules (Otterness & Anderson). It tends to support different programming languages depending on the real-time applications. Figure 3 illustrates the ROCm stack used to program AMD GPU.

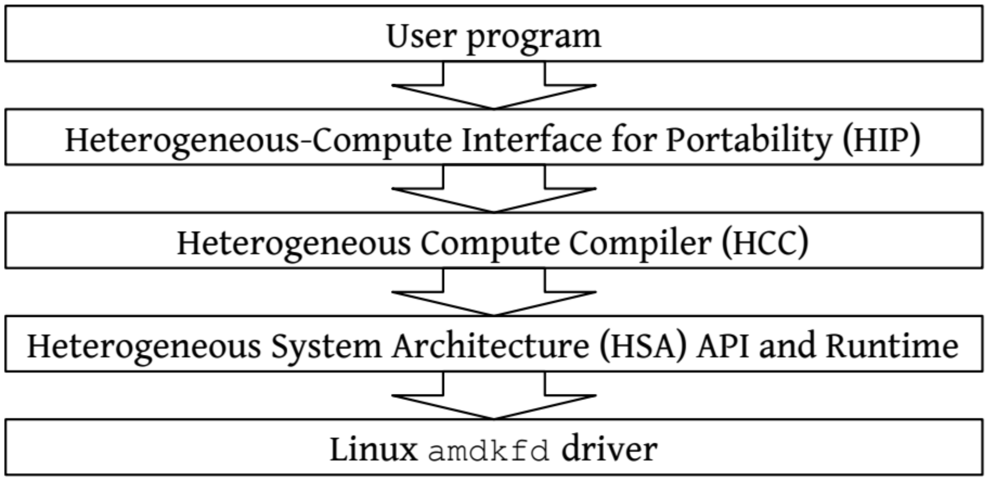

User program
Heterogeneous-Compute Interface for Portability (HIP)
Heterogeneous Compute Compiler (HCC)
Heterogeneous System Architecture (HSA) API and Runtime
Linux amdkfd driver

Figure 3 AMD's ROCm Software Stack (Otterness & Anderson)

As represented in Figure 2, the bottom of the stack comprises an amdkfd driver, which is used to mainline the Linus kernel. HSA (Heterogeneous System Architecture) is a lightweight user-level API (Application Programming Interface) wrapping low-level functionality. It includes functions of kernel queues, synchronization routines, memory management, and others. Above the HSA of the ROCm stack is HCC (Heterogeneous Compute Compiler), which is an LLVM-based compiler. It uses the HAS API and thus can compile code to target AMD GPUs. Finally, to simplify porting CUDA applications to AMD GPUs, the HIP (Heterogeneous Compute Interface for Portability) has been designed. This HIP language can target both NVIDIA and AMD GPU. When AMD is targeted, the features of HIP are implied on top of the HCC compiler. However, when NVIDIA is targeted, it utilizes the CUDA compiler. The study has conducted a case study using the ROCm software stack on AMD GPUs and used several CU-management interfaces to ROCm by altering the layers of the stack.

4. APPLICATION-SPECIFIC COMPARISON OF CUDA AND ROCm

Real-time applications of video and image processing approaches are rapidly emerging and opened up several ways to improve the ability of surveillance. Detection of moving humans has been implemented on GPU based on CUDA programming language (Bahri, Chouchene, Sayadi, & Atri, 2020). Initially, the moving object was retrieved by eliminating the background using GMM (Gaussian Mixture Model) on GPU. For performing classification, two complementary features, contour-based and region-based descriptions, have been extracted. To reduce the processing time to extract the Fourier descriptor model, CUDA CUFFT has been involved. The evaluation results have shown the improved performance of GPU compared to CPU in terms of the execution time of moving person detection on different video bases. A method of CUDA Fortran-based GPU accelerated Laplace equation model has been implemented to reduce the computational time (Kim, Yoon, & Kim, 2021). The work flow of CUDA data processing is shown in Figure 4.

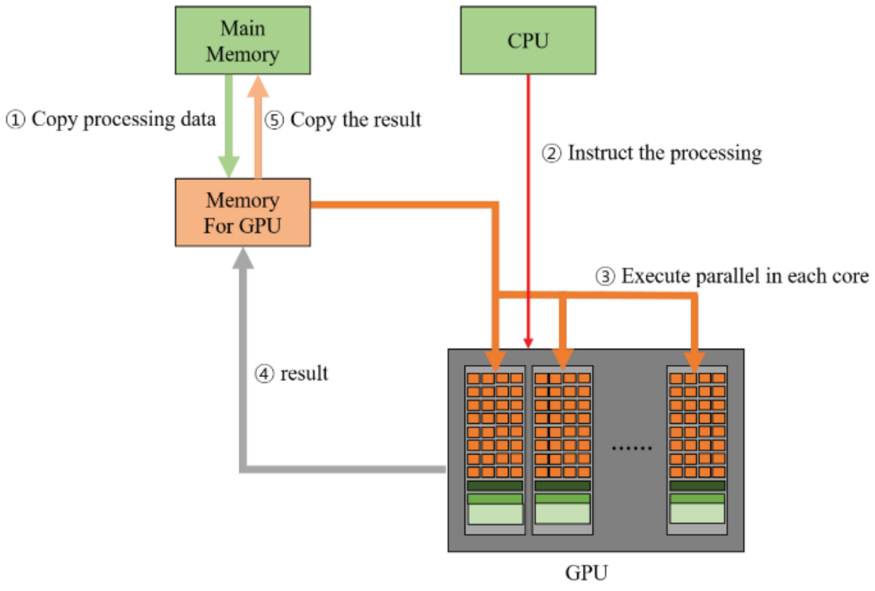


Figure 4 Flow Diagram of CUDA Data Processing (Kim et al., 2021)

In CUDA programming, the CPU and GPU possess separate memory units. The process of handling data in the GPU involves transferring it from the CPU memory to the GPU memory, where the CPU issues commands for data processing to the GPU. Subsequently, the GPU carries out parallel data processing and subsequently transmits the resulting data from the GPU memory back to the CPU memory. An analysis of the outcomes derived from an accurate GPU-accelerated Laplace equation model has been conducted, utilizing both an analytical solution and a numerical simulation. It has been observed that the GPU-accelerated Laplace equation model has performed better when compared with CPU-based numerical models. In order to harness the computational prowess of GPUs, a novel methodology utilizing CUDA was implemented to create a parallel rendition of the Fisher classification scheme (Al Sideiri, Alzeidi, Al Hammoshi, Chauhan, & AlFarsi, 2020). The study's primary objective is to expedite the encoding process of fractal images by curtailing the search operation involved in determining the optimal match for each fractal. To assess the performance of the algorithm, metrics such as peak signal-to-noise ratio, compression ratio, and encoding time have been used, and experiments have been conducted. It has shown that a speed of 6.4x has been achieved using NVIDIA GeForce GT 660M GPU. Most widely, the GPU using CUDA is employed to rectify Shallow wave equations and their related problems. To improve faster computation for resolving Shallow wave equations, a multi-GPU version of time explicit finite volume solver has been developed (Delmas & Soulaïmani, 2022).

In addition, the integration of MPI with CUDA-Fortran enables the utilization of multiple GPUs, while the METIS library facilitates domain decomposition operations. A CUDA-Aware version of OpenMPI has been implemented to enhance the effectiveness and speed of communication between MPI processes. By implementing this approach, the study successfully solved Shallow wave equations using a 12 million-cell mesh of an actual river. This accelerated speed greatly contributed to generating extensive datasets containing high-precision solutions. These datasets play a crucial role in conducting uncertainty propagation analyses. Multi-agent robotics and autonomy are comprised of an ensembling of interacting agents. So, GPU- GPU-accelerated NVIDIA CUDA has been utilized to model the behavior of multi-agent and soft-body robots at massive scales (Austin, Corrales-Fatou, Wyetzner, & Lipson, 2019). The main motive of the method is to develop an asynchronous computation model and CUDA kernel design to perform kinematics simulations in parallel. This has been done for optimizing and simulating actuating soft robots in real-time, and asynchronous simulation ensures the GPU operates separately. The results of the study have projected that it has been capable of simulating 300 million primitive updates per second.

Real-time data analysis and information processing are significant in providing efficient communication link bandwidth. A radiant-tolerant CubeSat-compatible onboard information processing architecture has been developed and evaluated (Bruhn, Tsog, Kunkel, Flordal, & Troxel, 2020). The HPC software stack of AMD's ROCm has been patched to enable embedded devices. Further, it supports ML (Machine Learning) software, namely TensorFlow, and computation of CUDA code in hardened silicon or radiation tolerant has been performed. From in-depth analysis, it has been acquired that heterogeneous architecture tested in the expanded AMD SOC with Intel Movidius Myriad X neural accelerator has shown a significant increase in AI processing speed with low bit resolution. A method of Arax known as a runtime system that decouples applications from heterogeneous accelerators within a server has been designed and developed (Pavlidakis, Mavridis, Chazapis, Vasiliadis, & Bilas, 2023). Initially, the available resources are mapped with Arax application tasks. This approach can manage task dependencies, required task states, and memory allocations. The study has applied ROCm v4.1.0 along with different accelerators and two server types for evaluation. From computation, the overheads of Arax are less pronounced in terms of AMD's ROCm.

5. RESEARCH GAPS AND FUTURE DEVELOPMENTS

To claim research gaps, there are a few extents that could assist from further exploration. One among them is the optimization of GPU computing algorithms to achieve even greater efficiency and performance (Otterness & Anderson). Besides, CUDA and ROCm possess a set of drawbacks, regardless of the availability of an alternative. Additionally, research could focus on improving the interoperability between ROCm and CUDA, allowing developers to seamlessly utilize both platforms in their applications. To mitigate the drawbacks, future works can be considered to further develop improved GPU-accelerated software.

- Investigating ways to enhance interoperability between ROCm and CUDA is a valuable research area that enables developers to seamlessly utilize both platforms in different applications. This allows for more compatibility and flexibility across various hardware architectures.
- Performance Optimization of CUDA and ROCm can benefit from further research on optimizing GPU computing algorithms to improve performance. This includes exploring new techniques for parallelization, memory management, and workload distribution to maximize the utilization of GPU resources.
- Further research on improving the energy efficiency of GPU computing is another important area that involves exploring techniques for reducing power consumption, optimizing memory access patterns, and developing energy-aware scheduling algorithms.
- The focus on enhancing the programming models and tools provided by ROCm and CUDA is significant, including providing better support for high-level languages and frameworks, enhanced debugging and profiling capabilities, and simplifying the development process.
- Developments could leverage the combined power of GPUs and other accelerators, namely specialized AI (Artificial Intelligence) chips or FPGAs (Field Programmable Gate Array). Effective analysis of this particular area could explore techniques for workload partitioning, efficient task offloading, and data sharing between different accelerators.

Overall, the limitations and future developments of CUDA and ROCm encompass performance optimization, machine learning, interoperability, programming models/tools, and energy efficiency. Continued research in these areas will contribute to the advancement of GPU computing and its applications in various sectors.

6. CONCLUSION

In this comprehensive review, the performance analysis of CUDA and ROCm is extensively explored and compared regarding their performance capabilities for GPU computing. Numerous studies have evaluated the performance of these platforms across various applications and hardware configurations. Based on these considerations, it was analyzed that most reviewed studies indicate that CUDA generally outperforms ROCm regarding compatibility, performance, and reliability with a wide range of GPUs. The contribution of CUDA's performance, optimization efforts, and extensive developer support ensures its dominance in various fields and applications. However, it is significant to note that ROCm has also shown optimal results in specific aspects, particularly when used with AMD GPUs. In addition, some studies have reported comparable performance between CUDA and ROCm, especially in certain hardware configurations and workloads. Finally, the review highlights the limitations and future research aiding in CUDA and ROCm developments. In summary, CUDA currently holds a wider industry adoption and performance advantage, and ROCm exhibits potential in specific contexts. Therefore, continuous research and development in both platforms will improve performance analysis for GPU accelerated computing. It has identified key areas that require further attention and provided recommendations for future studies. As GPUs continue to evolve and become more powerful, it is crucial for researchers to stay updated with the latest advancements and utilize them to improve the performance of HPC systems.

REFERENCES

1. Al Sideiri, A., Alzeidi, N., Al Hammoshi, M., Chauhan, M. S., & AlFarsi, G. (2020). CUDA implementation of fractal image compression. *Journal of Real-Time Image Processing, 17*, 1375-1387.
2. Andreadis, K. (2024). AMD MI300X vs NVIDIA H100
3. Austin, J., Corrales-Fatou, R., Wyetzner, S., & Lipson, H. (2019). Titan: A Parallel Asynchronous Library for Multi-Agent and Soft-Body Robotics using NVIDIA CUDA. *arXiv preprint arXiv:1911.10274*.
4. Bahri, H., Chouchene, M., Sayadi, F. E., & Atri, M. (2020). Real-time moving human detection using HOG and Fourier descriptor based on CUDA implementation. *Journal of Real-Time Image Processing, 17*, 1841-1856.
5. Bai, Y., Liu, Q., Wu, W., & Feng, Y. (2021). cuSCNN: A Secure and Batch-Processing Framework for Privacy-Preserving Convolutional Neural Network Prediction on GPU. *Frontiers in Computational Neuroscience, 15*, 799977.
6. Bruhn, F. C., Tsog, N., Kunkel, F., Flordal, O., & Troxel, I. (2020). Enabling radiation tolerant heterogeneous GPU-based onboard data processing in space. *CEAS Space Journal, 12*(4), 551-564.
7. Cao, K., Wu, Q., Wang, L., Guo, H., Wang, N., Cheng, H., . . . Wu, H. (2024). GPU-HADVPPM4HIP V1. 0: higher model accuracy on China's domestically GPU-like accelerator using heterogeneous compute interface for portability (HIP) technology to accelerate the piecewise parabolic method (PPM) in an air quality model (CAMx V6. 10). *Geoscientific Model Development Discussions, 2024*, 1-22.
8. Delmas, V., & Soulaïmani, A. (2022). Multi-GPU implementation of a time-explicit finite volume solver using CUDA and a CUDA-Aware version of OpenMPI with application to shallow water flows. *Computer Physics Communications, 271*, 108190.
9. Hashmi, M. F., Ayele, E., Naik, B. T., & Keskar, A. G. (2022). A parallel computing framework for real-time moving object detection on high resolution videos. *Journal of Intelligent Information Systems*, 1-22.
10. Hecht, H., Brendel, E., Wessels, M., & Bernhard, C. (2021). Estimating time-to-contact when vision is impaired. *Scientific Reports, 11*(1), 21213.

11. HOMERDING, B., & TRAMM, J. EVALUATING THE PERFORMANCE OF THE HIPSYCL TOOLCHAIN FOR HPC KERNELS ON NVIDIA V100 GPUS.

12. JUNE 2023. (2023).

13. Kim, B., Yoon, K. S., & Kim, H.-J. (2021). GPU-Accelerated Laplace Equation Model Development Based on CUDA Fortran. *Water, 13*(23), 3435.

14. Lai, J., Yu, H., Tian, Z., & Li, H. (2020). Hybrid MPI and CUDA parallelization for CFD applications on multi-GPU HPC clusters. *Scientific Programming, 2020*, 1-15.

15. Leinhauser, M., Widera, R., Bastrakov, S., Debus, A., Bussmann, M., & Chandrasekaran, S. (2022). Metrics and design of an instruction roofline model for AMD GPUs. *ACM Transactions on Parallel Computing, 9*(1), 1-14.

16. Lin, D.-L., Ren, H., Zhang, Y., Khailany, B., & Huang, T.-W. (2022). From RTL to CUDA: A GPU Acceleration Flow for RTL Simulation with Batch Stimulus.

17. NVIDIA. (2024). NVIDIA H100 Tensor Core GPU.

18. Otterness, N., & Anderson, J. H. AMD GPUs as an Alternative to NVIDIA for Supporting Real-Time Workloads.

19. Pavlidakis, M., Mavridis, S., Chazapis, A., Vasiliadis, G., & Bilas, A. (2023). Arax: A Runtime Framework for Decoupling Applications from Heterogeneous Accelerators. *arXiv preprint arXiv:2305.01291*.

20. Vanderbauwhede, W., & Takemi, T. Twinned buffering: A simple and highly effective scheme for. *Red, 500*(600), 700.

21. Yu, C., Royuela, S., & Quiñones, E. OpenMP to CUDA graphs: a compiler-based transformation to enhance the programmability of NVIDIA devices.