**CSCI596**

**Dr. Aiichiro Nakano.**

**Viterbi school of Engineering**

**GPU-Cuda sharing solution**

**12/09/2024**

**Jiancong Zhang**

**Background**

Graphics processing units (GPUs) are essential but often underutilized resources in high-performance computing (HPC) environments. This paper aims to analyze various GPU sharing technologies-rCUDA, Nvidia MPS, cGPU, and vGPU-to determine the most effective solution for different computing environments, from small-scale labs to large data centers. By addressing the problem of underutilized GPU resources, this study aims to improve resource allocation, management efficiency, and overall computing performance.

**introduction**

Inefficient utilization of GPU resources continues to cause significant economic and performance inefficiencies in many computing environments. This article explores four major GPU-sharing technologies: rCUDA, Nvidia Multi-Process Service (MPS), cGPU, and vGPU. We will evaluate the suitability of these technologies for different needs and environments to guide the best infrastructure decisions.

**GPU Sharing Technology**

rCUDA (remote CUDA) rCUDA allows time-multiplexed sharing by hijacking calls to the CUDA driver API. It achieves API-level forwarding by replacing the CUDA library. Then, it achieves the purpose of GPU sharing among multiple containers by modifying API functions such as video memory allocation and task submission. The disadvantage of this solution is that statically linked programs need to be recompiled, and modifications are required to adapt to the new version when the CUDA library is upgraded. Advantages: API is open source, making it easier for non-Nvidia technicians to implement. Disadvantages: Continuous adaptation is required as the CUDA library is updated; it may not cover all scenarios and low security.

Nvidia MPS (Multi-Process Service) Nvidia MPS aggregates multiple tasks into one context and uses shared memory for better spatial reuse. NVIDIA launched this solution to share GPU by having multiple CUDA programs share the same GPU context. At the same time, on Volta GPUs, users can set the proportion of GPU computing power occupied by each CUDA program through the CUDA\_MPS\_ACTIVE\_THREAD\_PERCENTAGE variable. Advantages: Best performance; minimal impact on job completion time under multi-task sharing conditions. Disadvantages: Errors in one task may affect others; there is no memory isolation, and computing resources can only be limited.

cGPU (Container GPU) cGPU provides virtual GPU devices for containers through a new kernel driver module, hijacking calls to Nvidia drivers. This solution offers virtual GPU devices for containers through a kernel driver, achieving isolation of video memory and computing power; it also configures virtual GPU devices in containers through lightweight user-state runtime libraries. Alibaba Cloud heterogeneous computing cGPU not only isolates computing power scheduling from video memory but also does not require the replacement of CUDA static or dynamic libraries, does not require the recompilation of CUDA applications, and CUDA and cuDNN versions can be upgraded at any time without adaptation. Advantages: high security, low sharing loss, and less need for driver updates. Disadvantages: dependency on specific operating system settings; complex to update; high development requirements; due to computing power time slice settings

rCuda

rCUDA is a approach to use remote GPU units instead of on-prem GPU server. Thus, the performance would decrease is expected since GPU has the longer distance reaching back to the on-prem side. The side effect of the performance depends on the type of apps/lab would run, the network bandwidth available for the connection running the application demanding GPU services.

“The above numbers should be taken as preliminary figures as they have been gathered with a low-performance 1Gbps Ethernet network. Additionally, as depicted in the figure on the right, the adapter in the Kayla system could only deliver 435 Mbps, whereas the adapter in the Xeon system delivered 943 Mbps. This is the leading cause of the performance loss in the Kayla-based configuration above. We are confident that with better interconnects (InfiniBand, for example), the performance of rCUDA remotely accelerated ARM-based systems will be similar to that of Xeon-based systems while requiring noticeably less energy.”[1]

When we talk about rCUDA, we think about remotely utilizing GPU resources over the network, which can significantly save energy and equipment costs in an ideal situation. However, this technical implementation is challenging.

First, the performance test of 1Gbps Ethernet shows that there is indeed a non-negligible performance loss when using rCUDA. For example, the Kayla system can only reach a transfer rate of 435 Mbps, while the Xeon system can reach 943 Mbps. Such network limitations are the key factor that leads to the performance degradation of the Kayla-based system configuration.

The extent of this performance loss depends on the application's needs and the bandwidth of the network connecting the two computers. For example, high-performance InfiniBand networks can significantly reduce this performance loss, making the performance of the ARM-based remote system close to that of an Xeon-based system while consuming less energy.

Taking these factors into consideration, rCUDA is particularly suitable for the following scenarios:

Energy-sensitive applications: For environments where energy costs are high or in areas where energy supply is limited, using rCUDA can effectively reduce energy consumption.

Resource allocation optimization: When resources are unevenly distributed, or GPU resources are insufficient to meet all local requirements, rCUDA can achieve more reasonable resource allocation.

Environmental constraints: In environments with limited physical space or budget, rCUDA can avoid the need to purchase additional GPUs.

However, the use of SCADA is only suitable for some occasions. In an environment with low network bandwidth, a significant drop in performance may offset the energy-saving advantages. In addition, deploying and maintaining a system involving remote GPU access also requires high technical support and maintenance costs. Therefore, before adopting rCUDA, the network environment and application scenarios must be carefully evaluated to ensure its benefits outweigh the costs.

MPS-MIG

As GPU capabilities continue to increase, a single application often cannot fully utilize a GPU's resources. NVIDIA's Multi-Process Service (MPS) allows multiple CPU processes to execute computing cores on the same GPU simultaneously, thereby better using GPU resources and improving overall data processing capabilities.

MPS technology helps the performance of applications when there is higher efficiently allocating hardware resources on multiple GPU units, and enhanced CPU parallel processing capabilities. At 2017, when NVIDIA launched the new Volta architecture, MPS can support up to 48 clients per GPU as a major update. Now the new features also widely used in V100, Quadro, and GeForce GPUs after the Volta series release.

NVIDIA introduced Multi-Instance GPU (MIG) technology, allow to share the GPU resources through strict partitioning, can made it ideal for multi-user environments. Each of the MIG instance has independent resources, and complete isolation between different users. When we review the NVIDIA's Ampere architecture, like the A100 GPU, each GPU may support up to 7 MIG instances. MIG can pair with MPS to run multiple MPS clients unit in the MIG instance, which improving resource utilization efficiency and application scalability.

MIG not only allows running a single application in multiple instances, but also through MPI, however it is not specifically designed to increase the performance in these scenarios. The main purpose is to enable multi independent applications to run on the same GPU unit when try to ensure the efficient resource allocation and strict isolation.

A graph of different types of atomic energy

Description automatically generated with medium confidence

The Testing for the 8-GPU DGX A100 server showed that adding the number of simulation tasks per GPU significantly can help the overall throughput performance. For example, in the RNAse test case,the throughput went up about 1.8x, compare to the ADH test case, the improvement was only about 1.3x.

The tests evaluated various configurations, including setups with and without MIG (Multi-Instance GPU) and MPS (Multi-Process Service). When MIG was enabled, each GPU was divided into 7 partitions, with 6 MPS clients unit running in each partition, allowing with a maximum of 42 clients per GPU. This is the baseline configuration without MIG or MPS was also tested for comparison. In addition, the "MIG+MPS" mode was also evaluated, each MIG partition handled a single simulation task, referred to as the "pure MIG" setup.

In the RNAse tests, "pure MIG" delivered performance close to that of "pure MPS." However, in the ADH tests, "pure MIG" underperformed compared to "pure MPS." When MIG was combined with MPS, RNAse achieved the best results, found out a 7% improvement over the best "pure MPS" configuration. In contrast, for ADH, the MIG+MPS configuration performed close to "pure MPS" but was slightly lower.

These results highlight that while MIG offers enhanced resource isolation, its impact on performance depends on the specific application. MPS, on the other hand, excels in managing concurrent tasks and maximizing resource utilization, making it suitable for workloads requiring high throughput.

Best Configurations: For the RNAse, the optimal setup run four MPS clients per MIG partition, with total of 28 simulations per GPU. For ADH, the best configuration was running 16 simulations per GPU without using MIG ("pure MPS").

Observations and Analysis: Enabling MIG isolates each simulation task to a specific hardware partition on the GPU, which can be advantageous for certain data access patterns. Without MIG, simulations dynamically share GPU resources, which might be more beneficial in other scenarios. The optimal configuration depends on the hardware and application requirements.

Additional Hardware Testing: To validate the results, tests were repeated on NVIDIA A40 and V100-SXM2 GPUs for the RNAse case. While increasing the number of simulations per GPU also improved throughput on these GPUs, the improvement was less pronounced than on the A100. In other hands, the throughput went up by 1.5x on the A40 and 1.4x on the V100-SXM2, in compression to 1.8x on the A100. This gap is directly tied to the performance capabilities of the testing GPU models.

Thus, when performing via MPS, with or without MIG, significantly helps wit the throughput. MIG does offer stronger task isolation in the observation, leading it particularly helful for scenarios need for multi-user GPU sharing. However, when focus more on the dynamic resource allocation ,the configurations without MIG does perform better. More ever, regarding for deploying, it's important to tailor the setup for the specific workload , the hardware and fine-tune runtime parameters via the experimentation to achieve optimal performance.

For the test using the 8-GPU DGX A100, throughput significantly improved by increasing the number of simulations per GPU using the MPS (Multi-Process Service) and MIG (Multi-Instance GPU). Suitable for high-performance computing would need multiple computationally intensive simulations simultaneously, like molecular dynamics simulations, GPU resource optimization helps to maximizes throughput and computational efficiency when we have limited GPU resources, and furthermore, in data centers or scientific research institutions, and multi-user sharing effectively allocates GPU resources among multiple users to ensure that each user can obtain predictable performance. Advantages include significantly improving the computational throughput of each GPU, especially when various simulations are running simultaneously, isolating each simulation to a specific partition of the GPU through MIG, reducing interference between different simulations, allowing different types of computational loads to run on the same GPU, and being suitable for a variety of computing scenarios. It has shown throughput improvements on various types of GPUs (such as NVIDIA A40 and V100-SXM2). However, the disadvantages of this technology include complex configuration, requiring fine-tuning and configuration to find the best settings, and may require multiple experiments; in some cases (such as more extensive ADH simulations), pure MPS may perform better than combined with MIG; the performance improvement is highly dependent on the specific test case and hardware configuration, and the effect varies significantly in different situations.

cCuda

Alibaba Cloud's heterogeneous computing GPU team launched the cGPU solution, a revolutionary innovation. This solution provides a virtual GPU device for the container through a kernel driver, achieving isolation of video memory and computing power. Users can configure the virtual GPU device in the container through a lightweight runtime library. In addition, Alibaba Cloud's cGPU solution implements computing power scheduling and video memory isolation without replacing CUDA static or dynamic libraries or recompiling CUDA applications. It can also upgrade CUDA and cuDNN versions without adaptation.

The cGPU kernel driver is a self-developed host kernel driver. Advantages:

Supports open-source standard Kubernetes and NVIDIA Docker solutions.

It is transparent to users, and AI applications do not need to be recompiled or the CUDA library replaced.

The underlying operations on NVIDIA GPU devices are more stable and optimized.

It also supports GPU memory and computing power isolation.

This innovative technology improves GPU resource utilization and simplifies the user's operation process, significantly improving computing efficiency and flexibility.

In practical applications, performance is a matter of great concern. How does Alibaba Cloud's cGPU container solution perform in this regard? We compare a set of test data to test the performance of model reasoning and training under the commonly used TensorFlow framework. The test device is a GPU instance on Alibaba Cloud, configured with an 8-core CPU, 32G memory, and an NVIDIA T4 graphics card with 16G video memory. Since the test data is a single result, some errors may occur.

When compared the performance inside a cGPU container and a standard Docker container, allocating all the memory and computing power to the GPU instance in the cGPU container to see if there would be any performance loss in the cGPU when the GPU is not shared.

The results show that in the ResNet50 training test, under different precision and batch-size settings, the green column represents the standard container performance, and the orange column represents the cGPU container performance. It can be seen that no matter what the situation, the performance of the GPU container instance is almost unchanged, and the performance is very close to that of the standard container. This shows that cGPU is a very efficient solution in terms of performance.

A graph of green and orange bars

Description automatically generated

Alibaba Cloud's cGPU solution provides a virtual GPU device for the container through a kernel driver, realizes the isolation of video memory and computing power, and configures the virtual GPU device in the container with a lightweight runtime library. The advantages of the cGPU solution are efficient resource utilization, user transparency, version flexibility, standard compatibility, stability, and isolation performance. Specifically, it adapts to the open-source standard Kubernetes and NVIDIA Docker solutions. AI applications do not need to be recompiled or replaced with CUDA libraries. Libraries such as CUDA and cuDNN can be upgraded anytime without adaptation. At the same time, it supports the isolation of GPU video memory and computing power to reduce interference between different tasks. However, the GPU solution also has some disadvantages, including complex configuration and not an open-source option. This solution currently only exists in the public cloud environment of Alibaba Cloud. The effect varies greatly in different situations. The GPU solution is particularly suitable for high-performance computing, data centers, and scientific research institutions that need to utilize GPU resources efficiently and want to simplify configuration and management processes, as well as multi-user sharing environments to ensure that each user can obtain predictable performance.

**Conclusion**

Effective GPU sharing is critical to maximizing computing resources and reducing operating costs. This study analyzes multiple GPU-sharing technologies, such as rCUDA, Nvidia MPS, cGPU, and MIG, and identifies their strengths and weaknesses. Choosing the right technology based on specific needs such as security, flexibility, and performance can significantly enhance computing infrastructure. The study shows that technologies like MPS and cGPU, which are efficient with minimal performance loss, are particularly suitable for high-performance computing environments, data centers, and multi-user settings. However, careful testing and matching technologies based on specific use cases are still necessary to achieve the best results.

Future Work

**Future**

Future research may focus on developing hybrid models that combine the advantages of these technologies to exploit their respective strengths. Integrating AI-driven workload prediction systems can optimize dynamic resource allocation and ensure the most efficient use of GPU resources. In addition, it is also critical to explore the scalability and robustness of these solutions in larger-scale and diverse environments. Enhancing security features and improving deployment ease are other vital areas for future development to promote the broader adoption of these advanced GPU-sharing technologies.

**Reference**:

1.Universidad Polit√©cnica de Valencia. (nd). rCUDA Middleware and Applications. Retrieved from https://network.nvidia.com/pdf/whitepapers/rCUDA\_Middleware\_and\_Applications.pdf

2. NVIDIA. (2023, October 17). Maximizing GROMACS throughput with multiple simulations per GPU using MPS and MIG. Retrieved from https://developer.nvidia.com/zh-cn/blog/maximizing-gromacs-throughput-with- multiple-simulations-per-GPU-using-mps-and-mig/

3. He, M. (2020, November 3). Alibaba Cloud's Heterogeneous Computing cGPU Solution. Alibaba Cloud Developer Community. Retrieved from https://developer.aliyun.com/article/771984