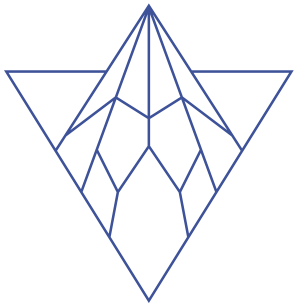
TECHNICAL BRIEF - COMPARING ALTERNATIVE SOLUTIONS

Benchmarking GPU Performance with GVM Server



ARC COMPUTE

Abstract

GVM Server is a software solution that addresses administrative code and infrastructure inefficiencies enabling increased GPU performance, reduced compute times, and 'true' utilization. These benefits are gained by granularly managing GPU compute resources. Through low-level utilization management, GVM Server users can achieve 100% GPU utilization and performance increases of 39% to 206% while utilizing half of the previously required infrastructure. GVM Server can manage all compute resources in a GPU, mitigating operational inefficiencies and creating performance opportunities compared to 'legacy' GPU management solutions. Whether an organization has compute-bound or memory-bound jobs/tasks/workloads, GVM Server can benefit by capturing the inefficiencies at low levels within hardware that the industry is currently unaware of, leading to substantial benefits.

Table of Contents

About Arc Compute

Arc Compute is a research and development company building optimization software for accelerated hardware, revolutionizing the HPC space. We aim to fully optimize the performance and utilization of the most prominent accelerated hardware technologies. Our impact on the world will involve reducing carbon footprints for organizations and individuals looking to use accelerated hardware-powered solutions in their day-to-day operations by maximizing the capabilities of the underlying hardware infrastructure. Our research and development have increased performance by enabling the complete utilization of computing resources. We expect to expand on this research as we continue mapping the capabilities of accelerated hardware at low-level utilization and optimization points. Arc Compute specializes in developing software solutions that maximize the throughput of accelerated hardware through mapping low-level utilization/optimization points, scheduling, and maximizing task compute and input/output thresholds.

What we do

- Enable 100% utilization of accelerated hardware
- Boost training/inference performance on compute infrastructure
- Reduce hardware requirements
- Optimize job scheduling

ARC COMPUTE

GVM Server Benchmarking

Arc Compute, working with an extensive North American AI/ML/NLP social network company, performed a proof of concept to investigate the feasibility of combining memory-limited and compute-limited workloads and their ranges of optimal performance with the primary goal of increasing utilization at the SM core level of a GPU. Memory-limited and compute-limited workloads were decided upon to simulate training and inference jobs. The consolidated workloads split an NVIDIA A100 40GB GPU to achieve 100% RAM and maximum SM core utilization while running operations concurrently. These results were then compared to memory-limited and compute-limited workloads running separately across the entire NVIDIA A100 40GB. The benefits of GVM Server with task-matching memory-limited and compute-limited workloads are outlined in this report.

Benchmarking Parameters

Idle Big/Job Big (Jb)

- Full pass through
- Create and fill the entire GPU with large SGEMM problems
- Run through the circular buffer of SGEMM problems on the GPU NVIDIA A100 40GB

Idle Small/Job Small (Js)

- Full pass through
- Create and fill the entire GPU with small SGEMM problems
- Run through the circular buffer of SGEMM problems on the GPU NVIDIA A100 40GB

Compute Big/Job Big (Jb) and Compute Small/Job Small (Js) [Shared]

- Split GPU with GVM Server
- Create and fill half the GPU with large SGEMM problems
- Create and fill the other half with small SGEMM problems
- Run both at the same time iterating through the circular buffer of SGEMM problems on both sides of the GPU – NVIDIA A100 40GB; 20GB given to each side

Notes

- Job Big(Jb) doubles at each case
- Job Small(Js) halves to Case 4 then remains consistent through Case 9
- New data loaded every case
- NVIDIA BLAS problems

Job Big Results

Cases	Job Big				Size in mega bytes	L2 Cache Utilization	L2 Cache Total
Case 0	1024	1024	1024	864	10.750	26.875%	21.500
Case 1	2048	1024	1024	864	18.125	45.313%	25.188
Case 2	4096	1024	1024	864	32.875	82.188%	38.094
Case 3	8192	1024	1024	864	62.375	155.938%	66.672
Case 4	16364	1024	1024	864	121.231	303.077%	125.067
Case 5	32,728	1024	1024	864	239.087	597.717%	242.923
Case 6	65,456	1024	1024	864	474.799	1186.997%	478.635
Case 7	130,912	1024	1024	864	946.223	2365.557%	950.059
Case 8	261,824	1024	1024	864	1889.070	4722.676%	1892.906
Case 9	523,648	1024	1024	864	3774.766	9436.914%	3778.602

Idle Big/Jb pass (Memory Limited)			Compute Big/Jb time when shared (Memory Limited) [GVM Server]			Big Compute Performance/Jb time when shared % (+/-)	Net Performance % (+/-)
Ave	Min	Max	Ave	Min	Max		
0.153	0.146	0.170	0.114	0.113	0.127	169.244%	160%
0.266	0.219	0.288	0.212	0.209	0.224	151.321%	161%
0.473	0.414	0.540	0.408	0.406	0.419	132.026%	135%
0.854	0.805	1.053	0.807	0.801	1.258	111.625%	147%
1.668	1.617	2.068	2.046	1.616	2.497	63.060%	135%
3.204	3.146	4.041	4.403	3.146	4.450	45.524%	127%
6.307	6.196	8.023	8.841	6.245	8.882	42.658%	125%
12.493	12.321	16.005	17.613	14.543	18.068	41.861%	124%
24.924	24.714	31.920	35.478	32.107	35.559	40.505%	124%
49.643	49.605	58.247	71.085	67.331	71.259	39.673%	123%

Observations

When comparing Case 0 through Case 3, “Compute Big/Job Big” using GVM Server performed better than “Idle Big/Job Big” with half the resources, 20GB of the NVIDIA A100 40 GB, while “Compute Small/Job Small” was running operations sharing the GPU. This proves GVM Server, with task/job matching, has benefits where you can complete tasks/jobs faster with half the resources for memory-limited workloads when matched with compute-limited workloads.

Job Small Results

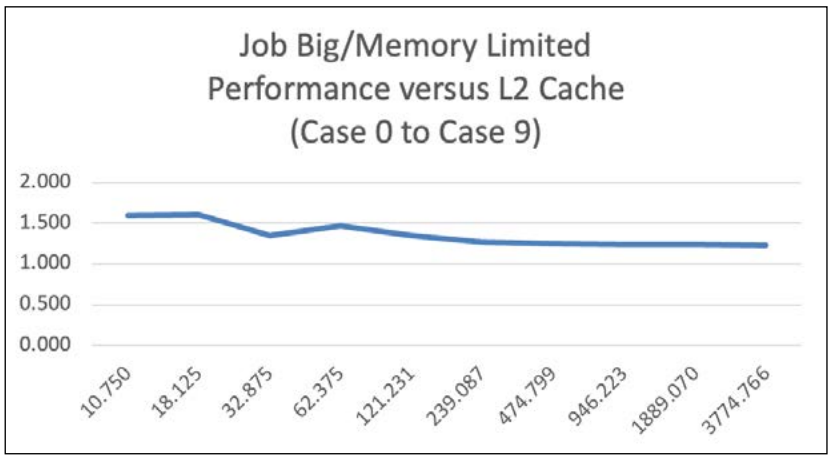
Cases	Job Small				Size in mega bytes	L2 Cache Utilization	L2 Cache Total
Case 0	1024	1024	1024	864	10.750	27%	21.500
Case 1	512	1024	1024	864	7.063	18%	25.188
Case 2	256	1024	1024	864	5.219	13%	38.094
Case 3	128	1024	1024	864	4.297	11%	66.672
Case 4	64	1024	1024	864	3.836	10%	125.067
Case 5	64	1024	1024	864	3.836	10%	242.923
Case 6	64	1024	1024	864	3.836	10%	478.635
Case 7	64	1024	1024	864	3.836	10%	950.059
Case 8	64	1024	1024	864	3.836	10%	1892.906
Case 9	64	1024	1024	864	3.836	10%	3778.602

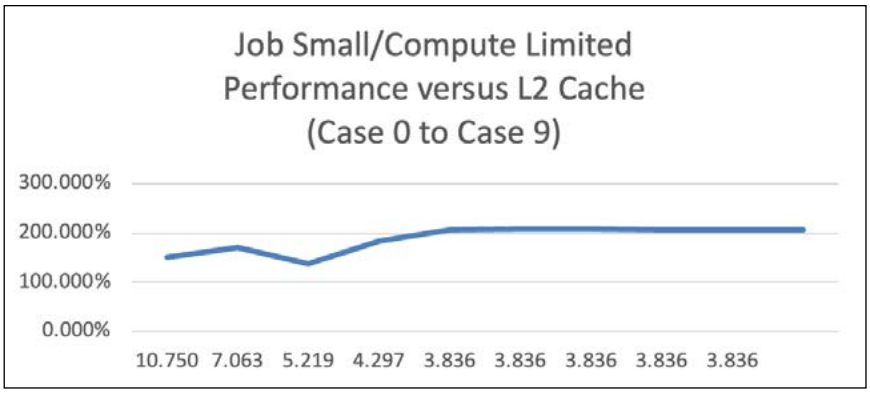
Idle Small/Js pass (Compute Limited)			Compute Small/Js time when shared (Compute Limited) [GVM Server]			Small Compute Performance/Js time when shared % (+/-)	Net Performance % (+/-)
Ave	Min	Max	Ave	Min	Max		
0.142	0.120	0.160	0.114	0.112	0.124	150.352%	160%
0.089	0.079	0.112	0.066	0.063	0.079	171.145%	161%
0.057	0.045	0.085	0.048	0.037	0.054	137.917%	135%
0.047	0.037	0.066	0.034	0.029	0.038	182.985%	147%
0.035	0.027	0.065	0.023	0.021	0.027	207.080%	135%
0.035	0.027	0.063	0.023	0.021	0.026	207.556%	127%
0.035	0.027	0.056	0.023	0.021	0.026	207.489%	125%
0.035	0.026	0.057	0.023	0.021	0.026	206.608%	124%
0.035	0.026	0.057	0.023	0.021	0.026	206.608%	124%
0.035	0.026	0.057	0.023	0.021	0.026	206.608%	123%

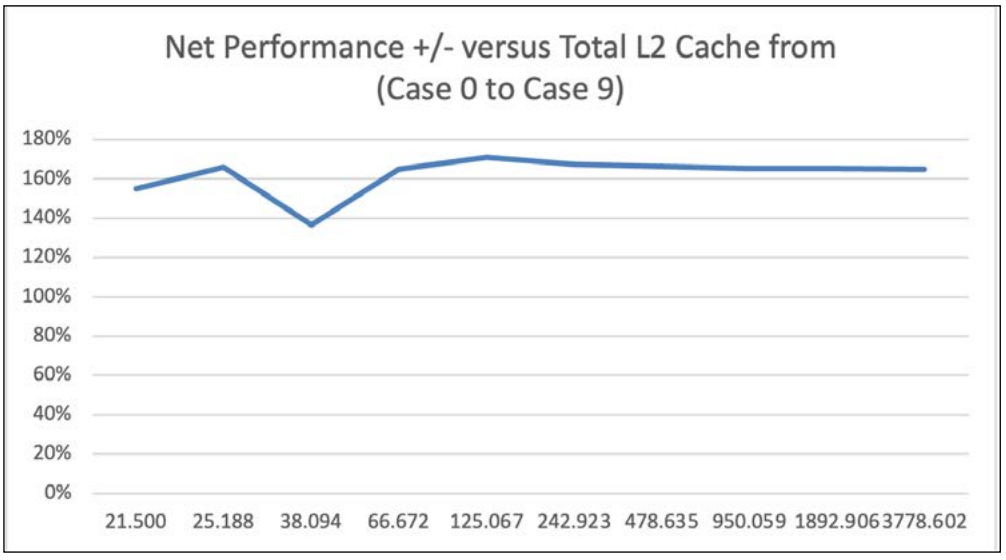
Observations

When comparing Cases 0 through Case 9, “Compute Small/Job Small” using GVM Server performed better than “Idle Small/ Job Small” with half the resources, 20GB of the NVIDIA A100 40 GB, while “Compute Big/Job Big” was running operations sharing the GPU. This proves GVM Server’s task/job matching has benefits where you can complete tasks/jobs faster with half the resources for compute-limited workloads when matched with memory-limited workloads.

Performance Graphs







SM Core Utilization

Case #	Idle Big/Jb pass (Memory Limited)	Idle Small/Js pass (Compute Limited)	Compute Big/Jb time when shared (Memory Limited)	Compute Small/Js time when shared (Compute Limited)
Case 0	SM Utils	SM Utils	SM Utils	SM Utils
	82%	82%	54%	44%
	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth
	8%	8%	2%	1%
Case 1	SM Utils	SM Utils	SM Utils	SM Utils
	89%	73%	58%	40%
	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth
	8%	7%	2%	2%
Case 2	SM Utils	SM Utils	SM Utils	SM Utils
	93%	70%	71%	27%
	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth
	13%	7%	6%	2%
Case 3	SM Utils	SM Utils	SM Utils	SM Utils
	96%	60%	82%	16%
	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth
	46%	6%	29%	6%
Case 4	SM Utils	SM Utils	SM Utils	SM Utils
	97%	50%	90%	8%
	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth
	85%	6%	60%	5%
Case 5	SM Utils	SM Utils	SM Utils	SM Utils
	98%	48%	76%	22%
	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth	HBM2 to L2 Bandwidth
	15%	6%	12%	3%

Observations

GVM Server's task matching captured SM core compute opportunities, maintaining SM core utilization at 98% while computing Job Big and Job Small simultaneously and mitigating I/O to compute inefficiencies from HBM2 to L2, seen in a lower total.

Benchmarking Conclusions

GVM Server, with task matching, provides a means to consolidate tasks/jobs generally executed in separate computing environments, along with a pathway to address macro and micro SM core optimization. With GVM Server, organizations should now categorize tasks/jobs/workloads relative to each other with business objectives factored as compute-limited or memory-limited for consolidation in HPC environments. Sharing the same accelerated hardware while jobs execute concurrently captures the most value from investments in HPC.

Evaluating Alternatives

Limiting the GPU core count with Arc Compute's GVM Server software increases AI job performance by over 200% for specific NVIDIA BLAS problems. The following section aims to provide an in-depth analysis of the different techniques on the market for GPU job sharing and the pros/cons associated with each method to show why Arc Compute's GVM Server software is the most beneficial way for organizations to accelerate their AI advancements.

GVM Server increases performance by optimizing SM cores and maximizing throughput while identifying and reallocating idle and under-utilized resources at runtime. This allows GPU compute-intensive jobs, such as AI training and inference, to be run simultaneously on the same GPU while simultaneously increasing workload performance and expediting workload completion compared to legacy methods, such as bare metal and legacy fractionalizing GPU approaches.

Before we begin, we must define the features that will benefit HPC managers moving forward. As such, we compiled the following features that each GPU job sharing/ scheduling technique should be mindful of. These features are part of a Pugh (Design Selection) Matrix that will help develop a recommendation for HPC managers to select different mechanisms to share GPUs/compute resources for maximal efficiency.

The weighting for each feature is as follows:

- Automatic Selection of Optimal GPUs - 4 / 4
- Potential For Growth - 3 / 4
- Unique Communication Channels - 2 / 4
- In-depth Utilization Metrics - 1 / 4

Automatic Selection of Optimal GPUs

To capitalize on this phenomenon, an organization requires individuals skilled in analyzing low-level GPU calls. These calls are then mapped across programs to match them with the optimal kernels, resulting in optimized runtime between programs A and B. This methodology is increasingly impractical as the number of programs increases, with the scheduling of an arbitrary combination of different programs across data center nodes. These jobs may also have other vRAM requirements for optimal performance, leaving the GPUs in a scenario of uneconomical provisioning. The solution must include mechanisms to select optimal configurations of the GPU to execute portions of the tasks at hand, simplifying the jobs of the individuals managing the infrastructure.

Reasoning behind highest possible value of 4/4 is the following:

- Improper allocation/GPU selection can lead to massive degradation in performance
- Solution should include mechanisms to prevent the over-provisioning of jobs on nodes
- GPUs must be paired with optimally matched tasks
- Mitigates bureaucratic job administrative risk and repercussions (delayed start times)
- Automatic GPU selection can help determine maximum thermal/wattage levels generated by nodes in the system, allowing us to accurately schedule the jobs of various systems while hitting a more comprehensive range of organizational targets such as sustainability projects/green initiatives

Business Impact:

- Simplification of HPC resource management
- Maximization of outcomes in alignment with business objectives
- Mitigation of technical expertise gaps in resource management
- Ability for operational management adjustments based on business initiatives

Potential for Growth

As the AI arms race accelerates, we see the primary competitive advantage being the speed of model iteration and improvement. The tools managing AI/HPC infrastructure must grow consistently. Over time, different optimizations may be discovered and implemented for more significant performance increases. As a result, the tools recommended in the AI/HPC infrastructure field must easily integrate with future GPU performance enhancements.

We gave this a 3/4 for the following reasons:

- Researchers should not have to create device-specific code
- The solution's performance increases should continuously provide improved optimization mechanisms for future-proofed development
- New hardware needs to be supported
- Integration of new future initiatives put into place by HPC management teams should be possible with the solution

Business Impact:

- Removal of technical adoption debt as new technologies emerge
- Mapping the "right path" of available technologies based on proprietary code bases
- Simplification of current and future procurement decisions based on code and business objectives

Unique Communication Channels

A system may have various concurrent running tasks, and these must not be able to interact with one another. We need a unique communication channel where hardware memory is separated between our various running AI/HPC tasks. This channel will limit zero-day exploits where one task can undesirably influence the outcome of other tasks on the system. We must also ensure that a task cannot affect another task's internal memory.

We weighted this section at 2/4 for the following reasons:

- Unique communication lines allow us to maximize throughput to the compute device
- Preventing undesirable task exploits is accomplished through proper separation of communication channels

Business Impact:

- Maximization of HPC resource capabilities
- Secure operating environments
- Mitigate vulnerabilities and contamination risk

In-Depth Utilization Metrics

To help HPC managers achieve maximum performance of data center investments, capitalization of this phenomenon should include detailed GPU utilization measurement mechanisms. To the benefit of researchers and the HPC managers, software should survey system components and each unique task's utilization of GPU elements. A suitable metrics system provides researchers with metrics they can continue to optimize, along with supplying HPC managers with increased visibility into future investments in hardware. These utilization metrics are also fundamental to the automatic selection feature previously mentioned.

We weighted this section at 1/4 for the following reasons:

- Can use current tools by NVIDIA, such as NVML, to provide quite in-depth utilization reporting
- Easy-to-use tools for future GPU/Compute device task usage analysis

Business Impact:

- Ability to accurately forecast growth based on resource needs at a granular level and support objectively with empirical information and data.
- Business intelligence

Solutions

To simplify this analysis, we have rejected any possible solution that can be layered on top of another solution, such as NVIDIA vGPU + MIG. For reference, NVIDIA vGPU + MIG has the same limitations that MIG by itself has. We are focusing on NVIDIA's GPU footprint in data center infrastructure because AMD only makes up approximately 9% of the AI/HPC space. The following solutions are rated on a scale of 0-10 to provide values for the Pugh Matrix.
NOTE: Several Chinese market alternatives are available; however, we cannot analyze them.

- **Bare Metal - De Facto standard**
- **MIG - Multi-Instance GPU**
- **MPS - Multi-Process Server**
- **vGPU - Virtual GPU**
- **GVM - Arc Compute's Proprietary System**

Bare Metal

To cover all bases, we consider a solution in which we run two arbitrary programs on the same bare metal instance. Therefore, we create two Unix users on the same system, each issuing a task to run individually.

From our analysis, we have the following scoring for the bare metal system:

- **Automatic Selection of Optimal GPUs - 0/10**
- **Potential For Growth - 0/10**
- **Unique Communication Channels - 3/10**
- **In-Depth Utilization Metrics - 5/10**

Automatic Selection of Optimal GPUs

This does not exist past the default GPU selection inside the NVIDIA driver. It does not select the optimally matched GPU and very quickly leads to substantial degradation. The GPU cores are not pinned by default, and it would require the developer/researcher to identify the optimal GPU cores and pin the task to those cores.

Potential For Growth

Each accelerator card has a drastically different development approach and is device-specific. It isn't easy to grow any solution based on optimizing other kernel modules in the data center.

Unique Communication Channels

There are some mechanisms of unique communication channels which are possible through the use of pinned optimal GPU cores. However, there is still only one hardware device that handles inputs, so all multiplexing occurs by GPU threads.

In-Depth Utilization Metrics

There are per-process utilization metrics that can provide information on a second basis. Using the open-source kernel module, it is also possible to build your own utilization measuring tools; however, this is quite complex.

MIG

MIG is the current approach NVIDIA encourages developers and researchers to use for fractionalizing GPUs. Unfortunately, it is constrained by how many cores can be assigned to each task. This approach only benefits if the task fits within one of the possible slices. It should also be noted that it is quite complex to set up correctly for individual use cases. From our analysis, we have the following scoring for MIG:

- **Automatic Selection of Optimal GPUs - 0/10**
- **Potential For Growth - 0/10**
- **Unique Communication Channels - 10/10**
- **In-Depth Utilization Metrics - 0/10**

Automatic Selection of Optimal GPUs

There is no current internal mechanism to find the optimal MIG device for a task, nor is it in the pipeline at NVIDIA.

Potential For Growth

This is tied to hardware advancements by NVIDIA and only works on a limited set of NVIDIA GPUs.

Unique Communication Channels

Compared with other solutions, this has the best structure for unique communication channels to a GPU.

In-Depth Utilization Metrics

In the current form of MIG, it is impossible to measure the utilization metrics efficiently.

MPS

CUDA's multi-process server system allows multiple CUDA tasks to be allocated and executed regarding the percentage of the time active on the GPU.

From our analysis, we have the following scoring for MPS:

- **Automatic Selection of Optimal GPUs - 2/10**
- **Potential For Growth - 3/10**
- **Unique Communication Channels - 0/10**
- **In-Depth Utilization Metrics - 0/10**

Automatic Selection of Optimal GPUs

There is a slightly better automatic selection than is available for bare metal. It, however, prefers to block the execution of tasks rather than find a more optimal GPU on the system to place the task on.

Potential For Growth

It is unlikely that NVIDIA will change the MPS system to provide organizations with a more significant edge.

Unique Communication Channels

This has the same limitations as bare metal but also prevents multiple users from accessing one server simultaneously. It forces Exclusive Operating Mode on the GPUs.

In-Depth Utilization Metrics

This has per-process utilization metrics, as is exposed by the bare metal approach.

vGPU

vGPU is the second-best approach we evaluated, although the initial setup and configuration are entirely convoluted. For example, the researchers would need to license the GPU inside the VM, leading to a poor customer experience.

From our analysis, we have the following scoring for vGPU:

- **Automatic Selection of Optimal GPUs - 0/10**
- **Potential For Growth - 3/10**
- **Unique Communication Channels - 5/10**
- **In-Depth Utilization Metrics - 0/10**

Automatic Selection of Optimal GPUs

This does not exist under this approach, leaving it up to the HPC manager to select the best GPU. Also, only homogeneous vGPUs are supported. For example, you cannot select half of a GPU and then provide another model with a fourth of a GPU without using MIG.

Potential For Growth

Although possible, it is unlikely that NVIDIA will change the vGPU system to provide organizations with a more significant edge.

Unique Communication Channels

This exists as of the Ampere Series.

In-Depth Utilization Metrics

Like bare metal, the organization would need to build the utilities themselves. If using MIG on top of this, then the same limitations of MIG exist.

GVM Server

While some bias may exist, we are dedicated to building this system to be more and more successful over time. Constant iteration and new tools/features development are essential to Arc Compute's vision.

From our analysis, we have the following scoring for GVM Server:

- **Automatic Selection of Optimal GPUs - 4/10**
- **Potential For Growth - 10/10**
- **Unique Communication Channels - 5/10**
- **In-Depth Utilization Metrics - 5/10**

Automatic Selection of Optimal GPUs

The current GPU selection approach is available through either GUI or data center specific rules regarding vRAM and core utilization. As time goes on, our system continually develops. We want to analyze the run-time of tasks even before they run. Additionally, we are creating optimization algorithms for matching underutilized components of GPUs/accelerator chips to each other. This will allow us to generate and hit different initiative marks set by HPC management teams across various organizations, as well as even more significant AI/HPC codebase optimizations.

Potential For Growth

A plugin system is under development that will allow organizations to build their own proprietary or tribal code for GPU message scheduling. This will help produce more customizability into the product for individual organization's use cases.

Unique Communication Channels

NVIDIA Ampere devices and above are supported for this.

In-Depth Utilization Metrics

These tools exist for specific utilization metrics, and we are working towards expanding this further to provide more data points for further optimizations.

Design Selection Matrix (Pugh Matrix)

We can now plug our analysis into a Pugh Matrix and select the best-suited solution for HPC managers. Based on the matrix, GVM Server is currently the best solution for fulfilling the features and capabilities essential for growth in the AI/HPC space.

Feature	Bare Metal	MIG	MPS	vGPU	GVM
Auto Selection (4)	0	0	2	0	4
Potential for Growth (3)	0	0	3	3	10
Unique Communications (2)	3	10	0	5	5
Utilization Metrics (1)	5	0	0	0	5
Total	**11**	**20**	**17**	**19**	**56**

Conclusion

With the benchmarking data in this brief, we demonstrate how HPC managers can increase the performance of their workloads by over 200%, all while utilizing half the number of GPUs, thanks to Arc Compute's software.

GVM Server offers a tremendous advantage from a cost efficiency perspective, enabling HPC managers to accelerate their AI innovation at a substantially faster rate, at a drastically lower cost, and with a significantly lower operational carbon footprint.

Organizations using GVM Server can leapfrog ahead of their competitors in AI advancements thanks to this increase in utilization and performance, setting them up for AI superiority now and into the future.

With the current Nvidia monopoly and pricing for their new generations of GPUs (H100 @ ~$25,000 each) and demand for these GPUs significantly outpacing supply, GVM Server offers significant value by negating the need for ordering the same quantity of new GPUs that organizations with industry average GPU utilization levels would need to purchase. Instead, a much lower number of GPUs can be bought while achieving the same level of value and performance.

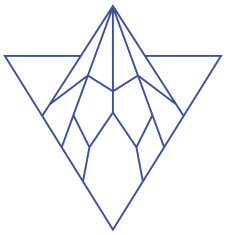
GVM Server positions organizations to accelerate AI advancements at lightning speeds, vastly outpacing competitors and cementing their positions as leaders in AI, infrastructure, and environmentally friendly operations.

Hardware Savings Example

Get More with Less

Industry Average (30% GPU Utilization)	
Required NVIDIA GPUs: 30,000	
3,750 Servers ($300k/Server)	$1,125,000,000
3-Year Software Cost	$18,750,000
Annual Power Cost ($0.07/kwh)	$18,396,000
Total 3-Year Cost	**$1,198,938,000**

With GVM Server (100% GPU Utilization)	
Required NVIDIA GPUs: 10,000	
1,250 Servers ($300k/Server)	$375,000,000
3-Year Software Cost	$100,000,000
Annual Power Cost ($0.07/kwh)	$6,132,000
Total 3-Year Cost	**$493,396,000**



ARC COMPUTE