VMware vSphere 7 with NVIDIA AI Enterprise Time-sliced vGPU vs MIG vGPU: Choosing the Right vGPU Profile for Your Workload

Performance Study - June 17, 2022

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com

Table of Contents

Executive Summary

NVIDIA AI Enterprise offers two methods to share GPUs among multiple VMs in VMware vSphere®: you can use NVIDIA vGPU with the time-sliced scheduler or with multi-instance GPU (MIG) technology. In this paper, we describe each profile and show that each one performs better under certain workload conditions. We list key takeaways at the end of this paper.

Introduction

NVIDIA vGPU allows vSphere to share NVIDIA GPUs among multiple VMs by using either the time-sliced vGPU profile or the MIG-with-vGPU profile (we will call this MIG vGPU throughout this paper). These two vGPU modes provide a flexible choice on how GPUs are shared to best leverage the GPU resource. With two options available, you might wonder if you should choose vGPU or MIG vGPU. This paper explores multiple use cases of the two profiles with different workloads, from ML training to ML inference and other workloads. We also show you how to choose the right profile for your workloads to maximize the benefits of vGPU and MIG vGPU.

NVIDIA GPU Virtualization in VMware vSphere

NVIDIA vGPU technology allows many GPU-enabled VMs to share a single physical GPU or several GPUs to be aggregated and allocated to a single VM, thereby exposing the GPU to VMs as one or multiple vGPU instances. With NVIDIA vGPU, a single GPU can be partitioned into multiple virtual GPU (vGPU) devices, as shown in figure 1. Each vGPU is allocated a portion of GPU memory specified by the NVIDIA vGPU profile.

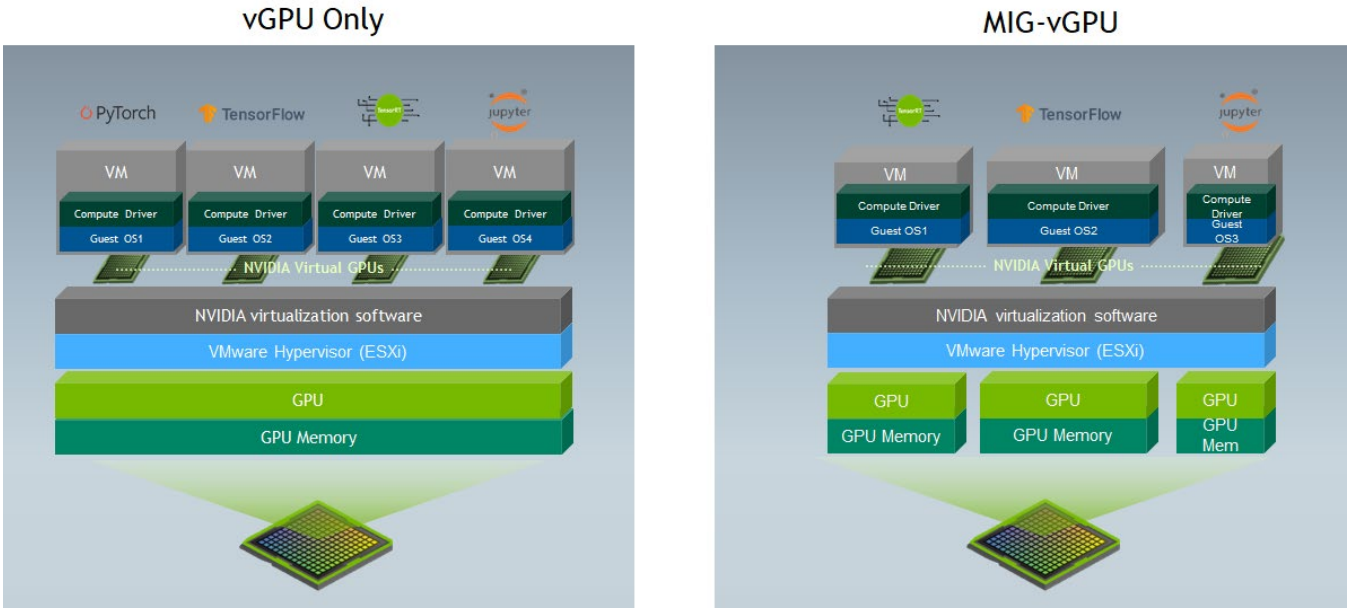

Figure 1. Multiple VMs sharing GPUs using vGPU or MIG-vGPU

There are two ways of sharing a vGPU:

- Using only NVIDIA vGPU software—CUDA cores of the GPU are shared among VMs using time slicing.
- Using NVIDIA multi-instance GPU (MIG) technology[1] with vGPU software—Each GPU can be partitioned into as many as seven GPU instances, fully isolated at the hardware level with their own high-bandwidth memory, cache, and compute cores, and then statically partitioned among VMs as separate vGPUs.

GPU virtualization is managed by the drivers installed inside the VM and the hypervisor. It exposes vGPUs to VMs and shares a physical GPU across multiple VMs. NVIDIA vGPU software is available in different editions designed to address specific use cases. You should use NVIDIA AI Enterprise software for virtualizing compute with VMware vSphere.

A blog we previously published about this topic shows the performance analysis of vGPU and MIG vGPU in which vGPU delivers the best performance, measured using wall-clock time, to complete the task for workloads with data transfers and/or CPU computations interspersed with CUDA computations. MIG vGPU best performs for workloads that execute heavy, large CUDA kernels with little or no interruption for data transfers or CPU computations. MIG vGPU shows the best performance for two or fewer VMs running concurrently for workloads with aggregated data transfers and aggregated CUDA computations. In contrast, the vGPU mode delivers the best performance with three or more VMs running concurrently.

[1] MIG technology is available on the NVIDIA A100 and A30 Tensor Core GPUs

vGPU vs. MIG vGPU: Choosing the Right Profile for Your Workload

We present here our test results and analysis that highlight the capabilities of vGPU and MIG vGPU and their differences in supporting and scaling ML training, ML inference, and NFV workloads, as well as their best practices. Please review the test results for these use cases to select the right profile for your applications that need GPUs in vSphere.

Testbed Setup

Our testbed setup included the following hardware and software:

- Dell EMC PowerEdge DSS 8440
  Intel Xeon 5218R @ 2.30 GHz
  40 CPU cores with 1.4 TB of memory
- NVIDIA A100 Ampere architecture–based[2] GPUs
- NVIDIA AI Enterprise software
- ESXi 7.0 U3
- VMs running Ubuntu 20.04, 32GB RAM, and 8 vCPU cores

We ran the following workloads on the VMs to simulate different use cases:

- Machine learning:
  - Mask R-CNN training and inference
- Network function:
  - Our own CUDA-based IPSec

We used the NVIDIA vGPU best-effort scheduler for the vGPU profile. The best effort scheduler shares GPU resources using a round-robin scheduling algorithm that can optimize resource utilization. The best effort scheduling policy best utilizes the GPU during idle and not fully used times, allowing for optimized density and a good quality of service (QoS).

[2] For more information about the NVIDIA A100 Tensor Core GPU architecture, see https://images.nvidia.com/aem-dam/en-zz/Solutions/data-center/nvidia-ampere-architecture-whitepaper.pdf

vGPU vs. MIG vGPU When Scaling the Number of VMs per GPU with Machine Learning Workloads

In this experiment, we ran the same Mask R-CNN workload (training and inference) scaling from 1–7 VMs concurrently, sharing a single A100 GPU using either the vGPU or MIG vGPU profile. We used different profile settings for each test case, as shown in Table 1, to allow the maximum utilization of GPU time and memory in each scenario. Hence, we set `batch size = 2` for Mask R-CNN so that the workload fits this GPU memory allocation when using the lowest profile A100-5c or A100-1-5c.

Number of VMs	vGPU	MIG vGPU
1 VM	A100-40c	A100-7-40c
2 VMs	A100-20c	A100-3-20c
3 VMs	A100-10c	A100-2-10c
4 VMs	A100-10c	3 VMs with A100-2-10c 1 VM with A100-1-5c
7 VMs	A100-5c	A100-1-5c

Table 1. vGPU profile settings for sizing the number of VMs

ML Training Results

For Mask R-CNN training, MIG vGPU shows better performance for both training time and throughput, as shown in figures 2 and 3, especially when the number of VMs sharing the same GPU increase. Due to the use of `batch size = 2` for Mask R-CNN, the training task in each VM uses less compute resources (fewer CUDA cores are utilized) and less GPU memory (less than 5GB compared to the 40GB each GPU has). So, this workload can be considered a lightweight ML training workload.

MIG partitions CUDA cores among all VMs to allow each workload to use its CUDA core partition fully. For MIG vGPU, all VMs use the GPU without waiting for the others. When workloads use less compute resources, as in this experiment, MIG vGPU performs better for multiple VMs sharing a single GPU. For vGPU, the GPU is shared among VMs using time slicing, and only one VM uses all the CUDA cores of the GPU at a time while the other VMs are waiting for their turn. The performance difference is higher when there are more VMs per GPU, and the workload in each VM utilizes fewer CUDA cores. While vGPU does not perform as well as MIG vGPU in this experiment, when there is a requirement for a high consolidation of VMs per GPU, vGPU shows a benefit because it allows up to 10 VMs per GPU compared to MIG vGPU, which currently supports up to 7 VMs per GPU.

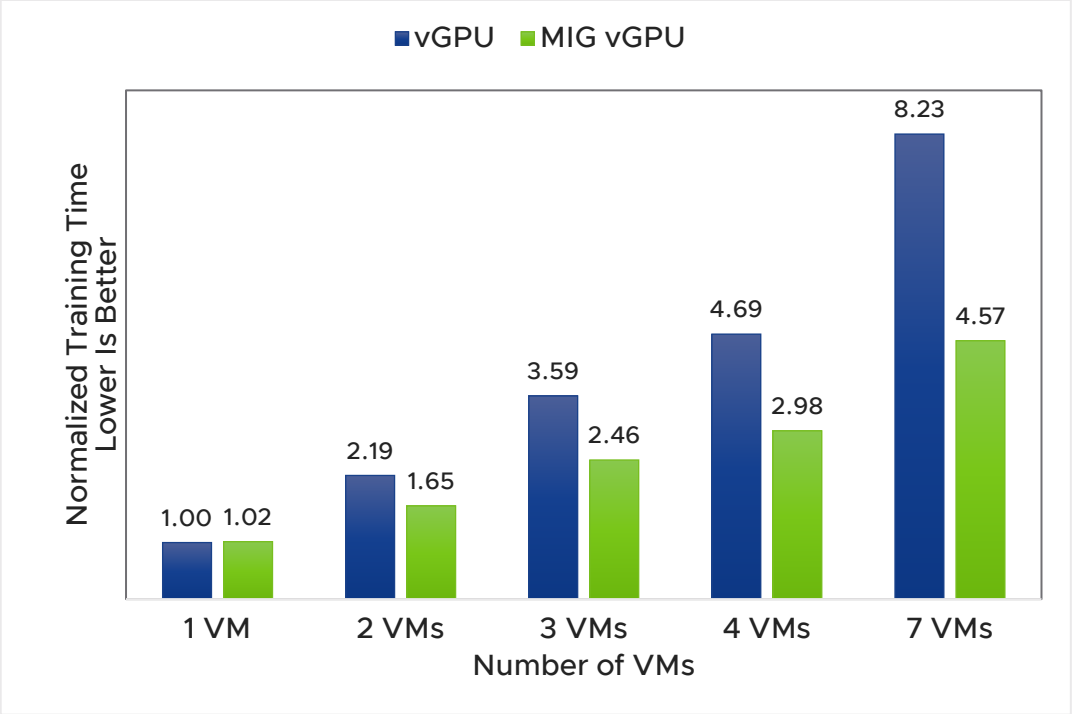

Figure 2. ML training time of vGPU vs. MIG with vGPU

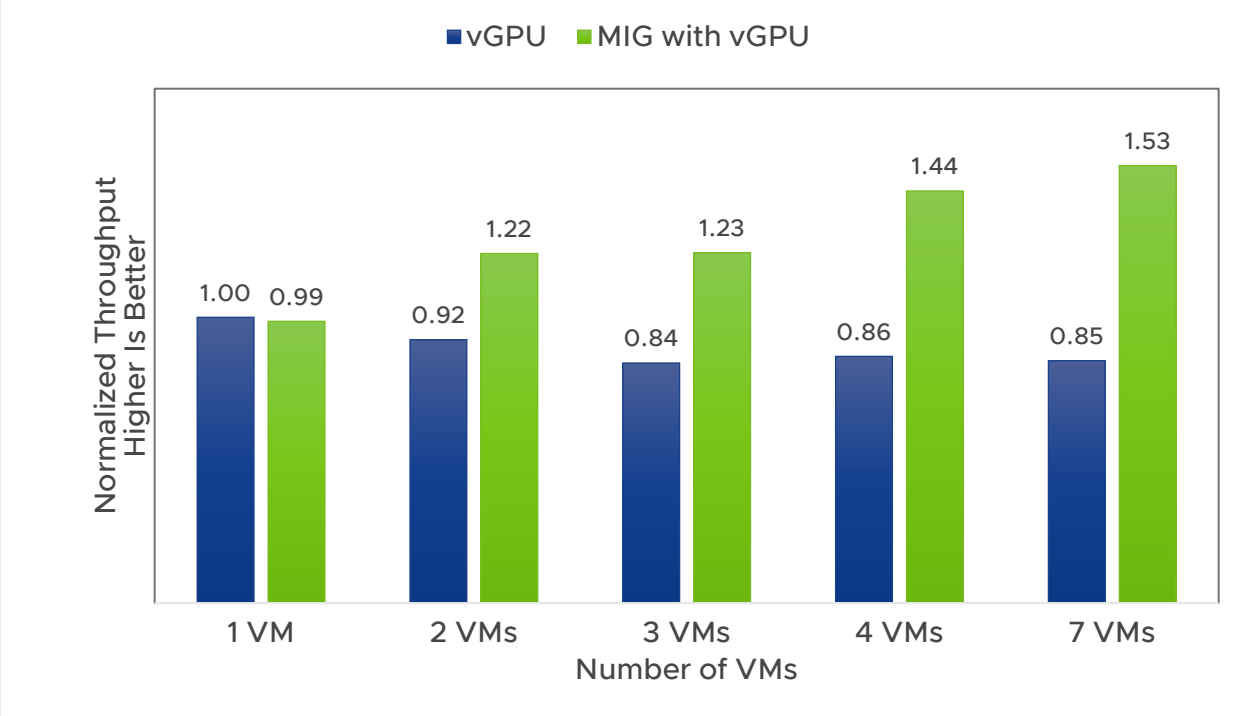

Figure 3. ML training throughput of vGPU vs. MIG with vGPU

While vGPU does not perform as well as MIG vGPU in this experiment, when there is a requirement for a high consolidation of VMs per GPU, vGPU shows a benefit because it allows up to 10 VMs per GPU compared to MIG vGPU, which currently supports up to 7 VMs per GPU. vGPU gives higher consolidation (10 VMs with vGPU vs. 7 VMs with MIG vGPU) because the lowest profile supported by vGPU is A100-4c (4GB per vGPU), while the lowest profile supported by MIG vGPU is A100-5c (5GB per MIG vGPU). In the use cases where GPU shared among multiple engineers/data scientists are highly underutilized by the users, the training time of vGPU can potentially be lower than MIG vGPU. For example, given 7 VMs sharing GPU using either vGPU (A100-5c profile) or MIG vGP (A100-1-5c profile) and only one VM runs Mask R-CNN (batch size = 2) at a time, the normalized training time of vGPU can be close to 1 because the VM can use all CUDA cores of the GPU for its entire training process. Meanwhile, the normalized training time of MIG vGPU, in this case, is still 4.57 because the VM with A100-1-5c profile is allocated fewer CUDA cores of GPU for its computation.

ML Inference Results

We observe a similar performance impact when scaling the number of VMs per GPU for Mask R-CNN inference. MIG vGPU provides much better performance than vGPU (see figures 4 and 5). The explanation for this performance difference is like that of the previous training case.

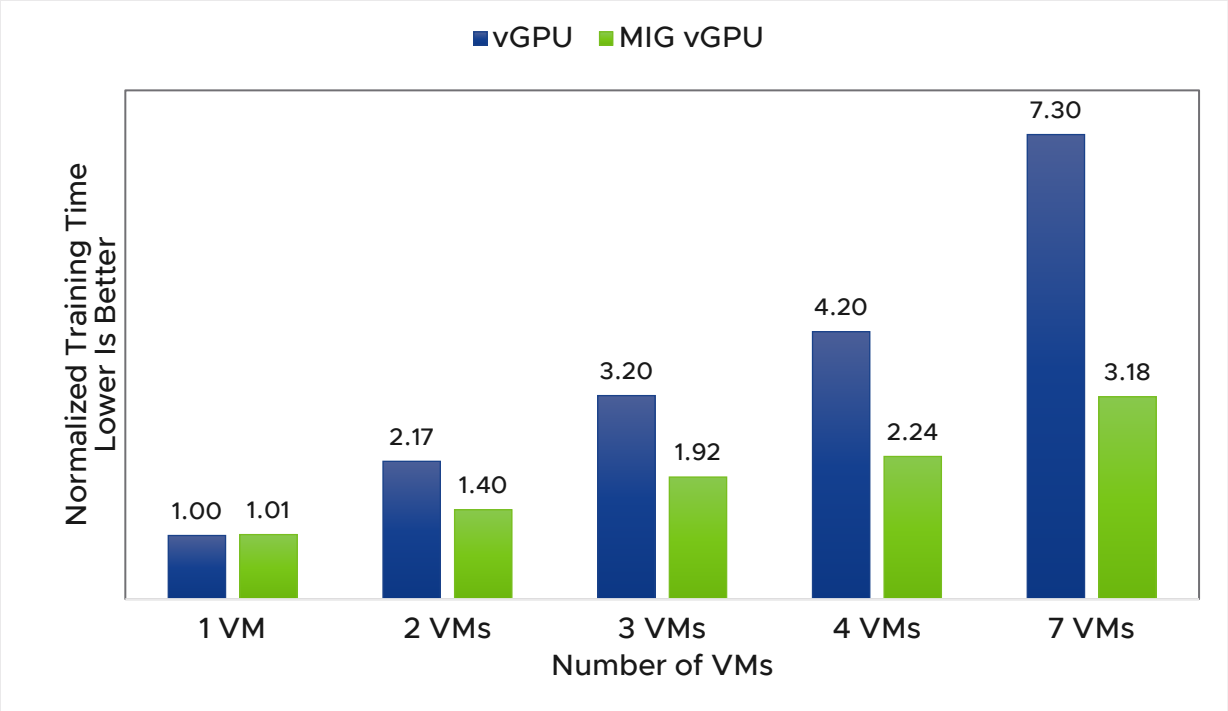

Figure 4. ML inference time of vGPU vs. MIG vGPU when scaling VMs per GPU

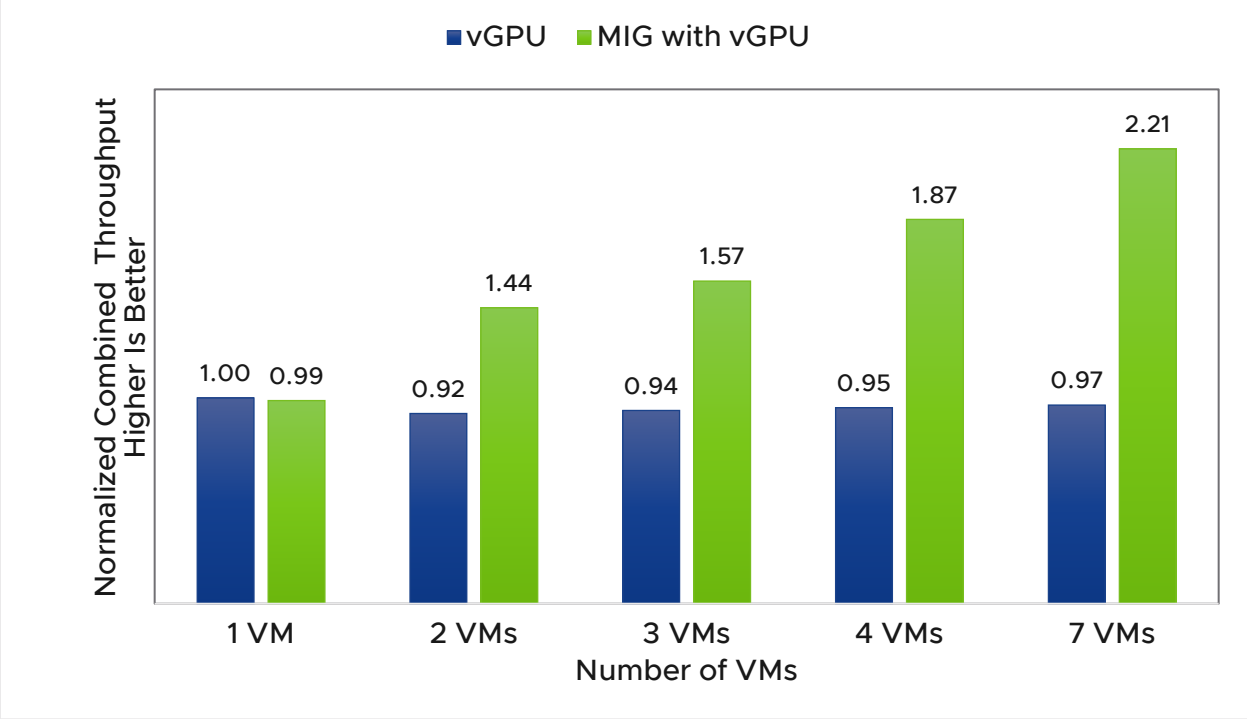

Figure 5. ML inference throughput of vGPU vs. MIG with vGPU when scaling VMs per GPU

vGPU vs. MIG vGPU When Sizing Machine Learning Workloads

In the previous experiments, we showed the performance impact of vGPU vs. MIG vGPU when multiple VMs share a single GPU. The workload used in that case is a less demanding one in which not all CUDA cores are utilized. But what is the performance difference when the workloads are more computational intensive? To answer this question, we conducted experiments that do Mask R-CNN training and inference in three different scenarios (shown in table 2) that mimic the ML load in three cases: lightweight, moderate, and heavy.

	Light Load	Moderate Load	Heavy Load
Workload	Mask R-CNN	Mask R-CNN	Mask R-CNN
Batch Size	2	4	8
Type of vGPU / MIG profile	5c or 1-5c	20c or 3-20c	40c or 7-40c
Number of VMs	4	2	1

Table 2. Test scenarios of sizing the ML workloads

ML Training Results

The training time and throughput results of our experiments for Mask R-CNN training are shown in figures 6 and 7. For MIG, when the workload running in each VM uses less GPU compute and memory resources, the training time and throughput of MIG vGPU is better than vGPU. The performance difference between vGPU and MIG vGPU is highest for the light load. MIG vGPU is also better for a moderate load, but the performance difference between vGPU and MIG vGPU is narrower. For a heavy load, vGPU performs better than MIG vGPU, even though the difference is negligible.

So, if you know the workload characteristics, you can choose the appropriate option for your application. If you do not know the workload characteristics, we recommend you test both vGPU and MIG vGPU options to select the better one.

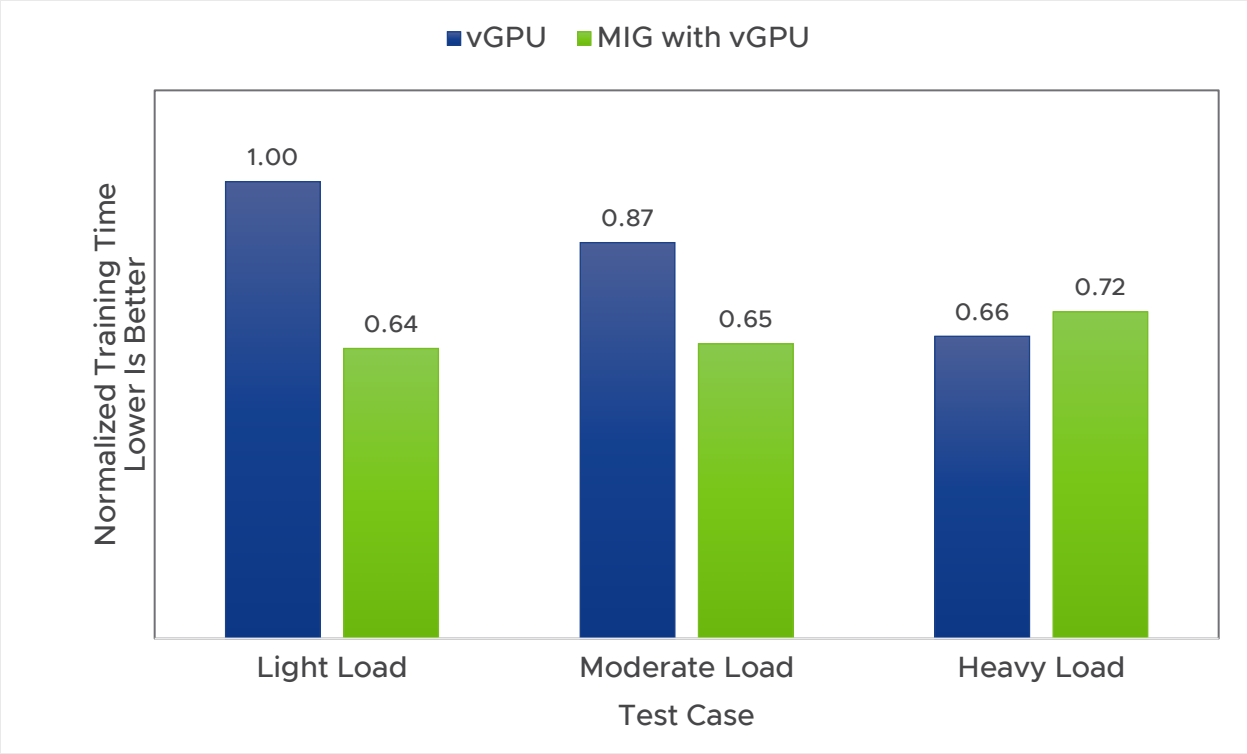

Figure 6. ML training time of vGPU vs. MIG vGPU with different GPU loads

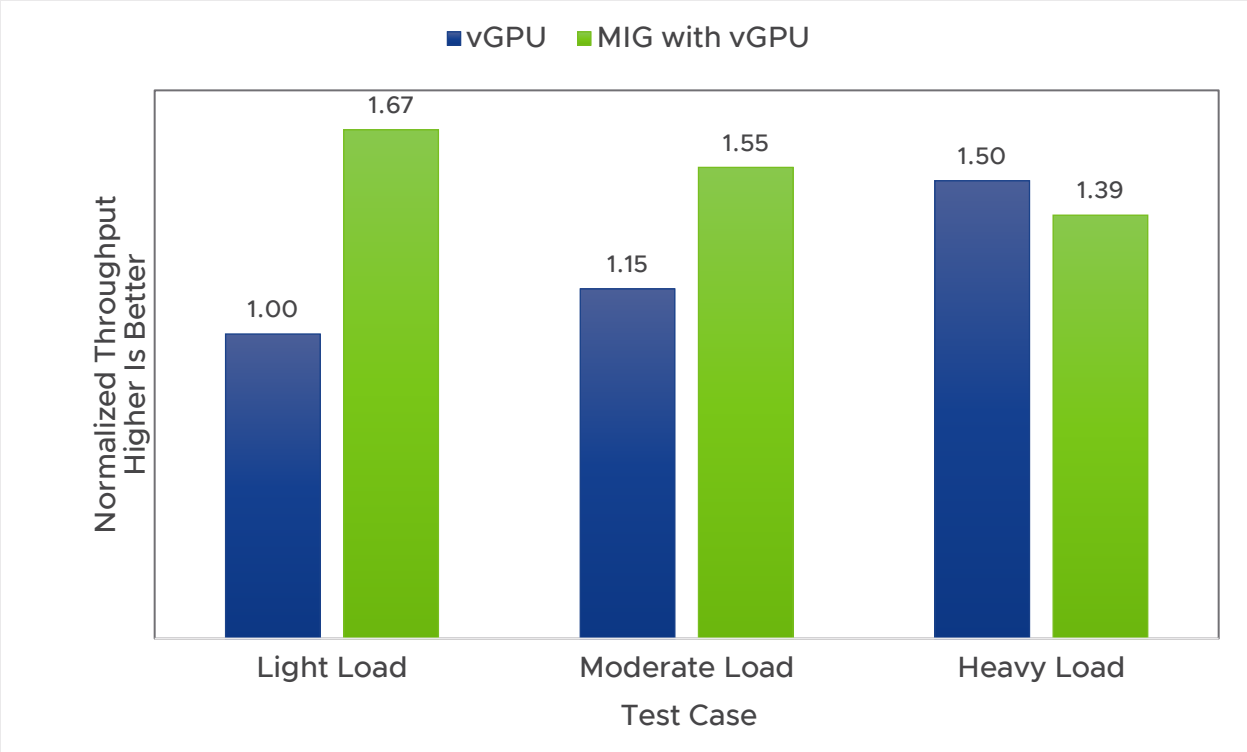

Figure 7. ML training throughput of vGPU vs. MIG vGPU with different GPU loads

ML Inference Results

Our experiments' inference time and throughput results for Mask R-CNN inference are shown in figures 8 and 9. The performance impact is like that of the training case. For MIG vGPU, the less GPU compute and memory resources that the workload running in each VM uses, the better performance is in both training time and throughput of MIG vGPU compared to vGPU.

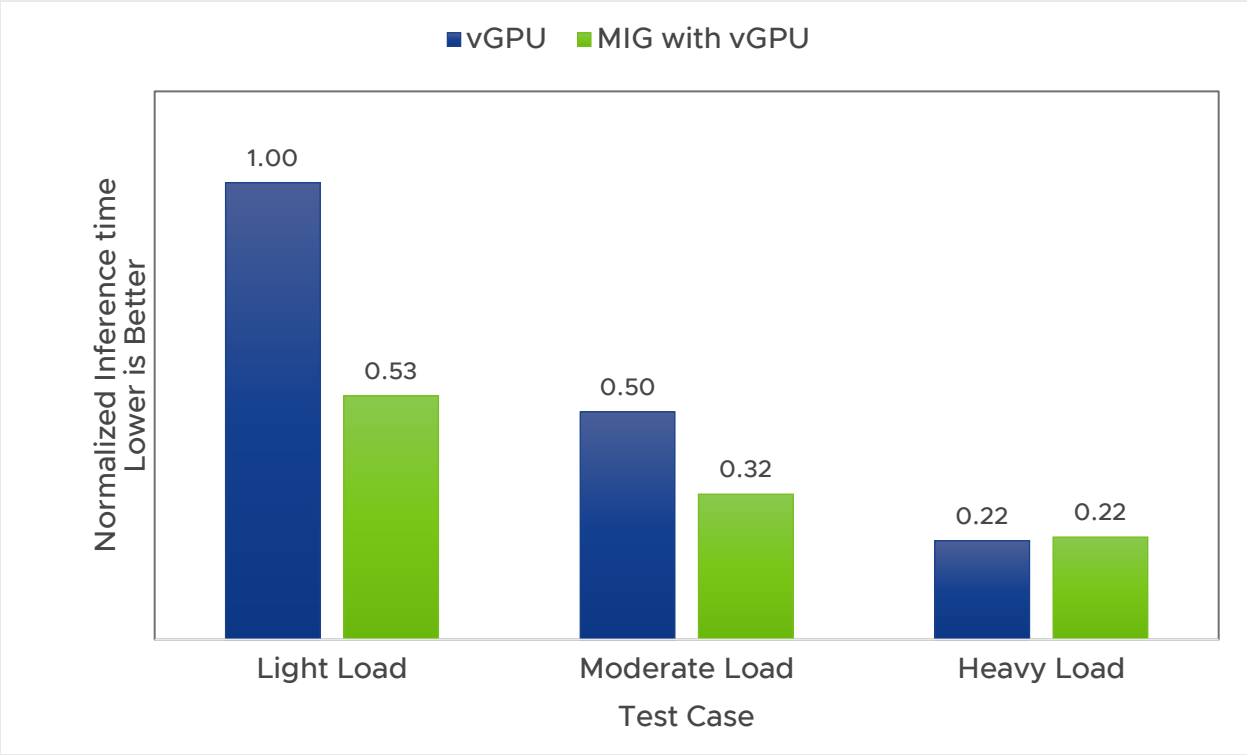

Figure 8. ML inference time of vGPU vs. MIG with vGPU with different GPU loads

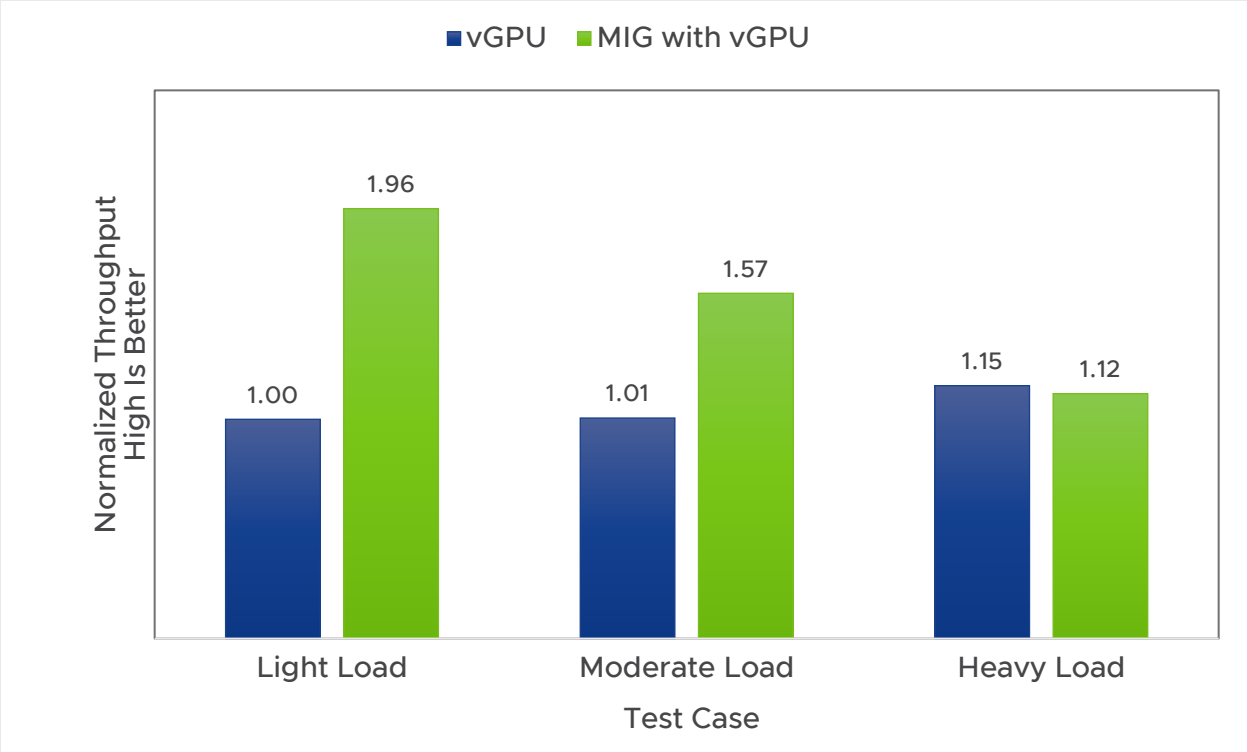

Figure 9. ML inference throughput of vGPU vs. MIG with vGPU with different GPU loads

vGPU vs. MIG vGPU for Workloads with Heavy I/O Communications Like Network Function with Encryption

Based on the previous experiments, MIG is better suited for lightweight-medium loads, which may make MIG ideal for development and testing scenarios. When running heavy workloads, vGPU or MIG performance differences are marginal. This section shows another set of experiments with other use cases for you to consider before choosing vGPU or MIG vGPU for your workload. Our experiments implemented a network function workload called Internet Protocol Security (IPSec) that is both compute-intensive and I/O-intensive. This workload uses CUDA to copy data between CPU and GPU and off-loads its computation to GPU. Our IPSec used HMAC-SHA1 and AES-128 bit in CBC mode. The OpenSSL AES 128-bit CBC encryption and decryption algorithm was rewritten in CUDA as part of our implementation[3].

Our test results in figure 10 show that vGPU performs better than MIG vGPU for this scenario. This workload is compute-heavy and utilizes a lot of GPU memory bandwidth. For MIG vGPU, this memory bandwidth is partitioned among VMs. In contrast, vGPU shares the memory bandwidth among VMs.

[3] For more information about OpenSSL TLS/SSL and crypto library, see https://github.com/openssl/openssl.

This explains the performance difference between vGPU and MIG vGPU in this use case. Hence, the workload characteristics are unknown; it is better to test both the profiles to select the better one.

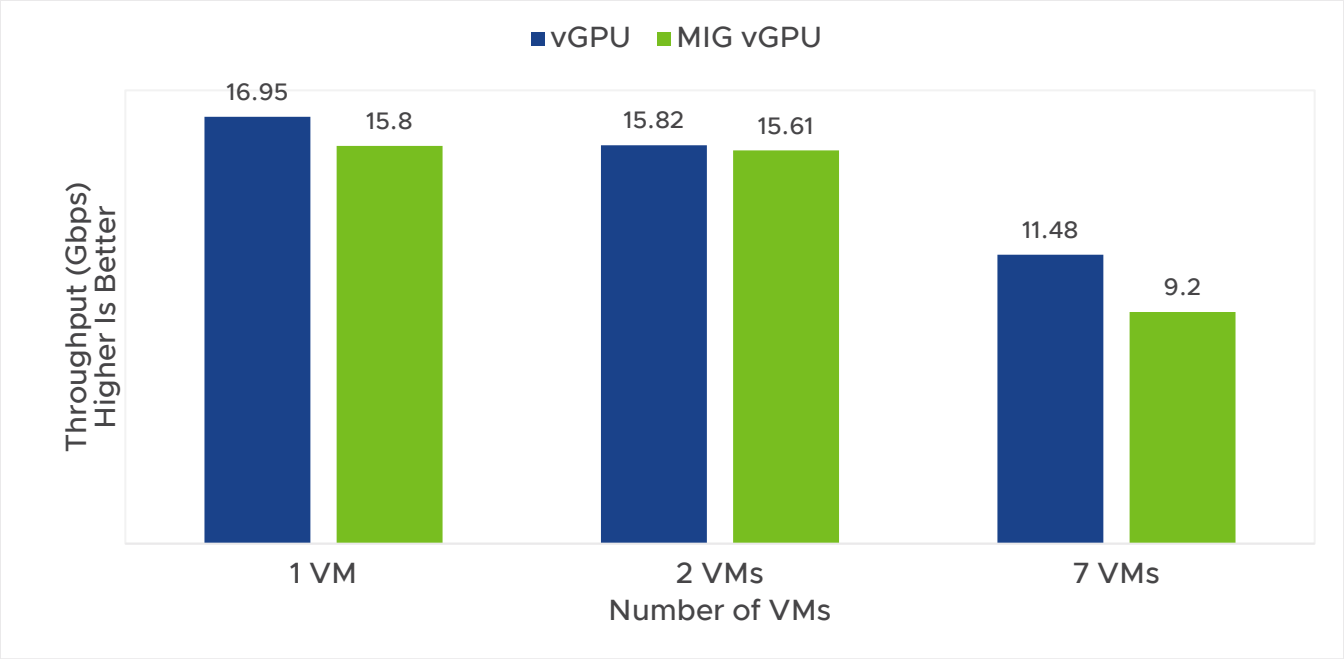

Figure 10. IPSec throughput of 7 VMs with MIG vGPU profile vs. vGPU profile on A100

Takeaways

We explored multiple workloads that compare the performance of vGPU vs. MIG vGPU. A few key takeaways from our experiments include:

- When your workloads are lightweight (small models, small batch size, and small input data), choosing MIG vGPU can give you a high consolidation of VMs per GPU with better performance. This also means saving money for your ML/AI infrastructure.
- For workloads that heavily use GPU (large models, large batch size, and large input data), which also means the number of VMs sharing one GPU is small, the difference between vGPU and MIG vGPU is relatively small, in which vGPU shows slightly better performance.
- For compute-heavy and I/O-heavy workloads like NFV, vGPU performs better than MIG vGPU in most test cases.

You can rely on these tests to choose the suitable profile for your specific workloads. If you do not know the workload characteristics, we recommend you test both vGPU and MIG vGPU to select the one that gives better performance results.

References

[1] Hari Sivaraman, Uday Kurkure, and Lan Vu, MIG or vGPU Mode for NVIDIA Ampere GPU: Which One Should I Use? (Part 1 of 3), https://blogs.vmware.com/performance/2021/09/mig-or-vgpu-part1.html

[2] NVIDIA Deep Learning Examples, https://github.com/NVIDIA/DeepLearningExamples

[3] NVIDIA A100 Tensor Core GPU Architecture, https://images.nvidia.com/aem-dam/en-zz/Solutions/data-center/nvidia-ampere-architecture-whitepaper.pdf

[4] Hari Sivaraman, Uday Kurkure, Lan Vu "Task Assignment in a Virtualized GPU-enabled Cloud," International Conference on High Performance Computing and Simulation, 2018.

About the Authors

Lan Vu has been working at VMware for eight years, focusing on GPU virtualization on vSphere and machine learning. Lan is interested in developing solutions that bring high performance and low cost to customers so they can configure their systems to best utilize cloud resources. Lan holds a PhD in Computer Science from the University of Colorado Denver and has 19 issued and pending patents.

Hari Sivaraman is a staff engineer at VMware. He is interested in making vSphere an outstanding platform on which to run ML/AI applications and in using ML/AI to solve cloud management problems.

Uday Kurkure works on accelerators for machine learning at VMware. He has a very broad skill set ranging from writing compilers, designing ASICS/FPGAs for computer graphics, and working on reconfigurable computing to virtualizing systems with VMware. His current interests are machine learning and high performance computing. He is a co-chair of the Virtualization in High Performance Computing and Simulation session (VIRT) at the IEEE High Performance Computing and Simulation Conference (HPCS). VMware awarded him the most prolific inventor award: he has 21 granted patents and 14 pending patent applications. He has published 13 research papers. His educational background includes an MS degree in Computer Science from Stanford and a B. Tech. in Electronics and Telecommunications from the Indian Institute of Technology. Before VMware, he worked for Synopsys, Adobe Systems, Transmeta, and MIPS Computers.

Acknowledgments

VMware thanks Charlie Huang, Manvendar Rawat, and Andy Currid of NVIDIA for reviewing and providing the input for this paper. The authors acknowledge Juan Garcia-Rovetta and Tony Lin of VMware for the management support.