Towards Improved Power Management in Cloud GPUs

Pratyush Patel[1], Zibo Gong[2], Syeda Rizvi[2], Esha Choukse[3], Pulkit Misra[3], Tom Anderson[1], Akshitha Sriraman[2]
[1]University of Washington, [2]Carnegie Mellon University, [3]Microsoft

Abstract—As modern server GPUs are increasingly power intensive, better power management mechanisms can significantly reduce the power consumption, capital costs, and carbon emissions in large cloud datacenters. This paper uses diverse datacenter workloads to study the power management capabilities of modern GPUs. We find that current GPU management mechanisms have limited compatibility and monitoring support under cloud virtualization. They have sub-optimal, imprecise, and non-intuitive implementations of Dynamic Voltage and Frequency Scaling (DVFS) and power capping. Consequently, efficient GPU power management is not widely deployed in clouds today. To address these limitations, we make actionable recommendations for GPU vendors and researchers.

Index Terms—Power management, Graphics processors, Super (very large) computers, Servers, Design for power delivery limits.

1 Introduction

The end of Dennard scaling and rise of accelerators make power a bottleneck resource in today's cloud datacenters. Power drives resource provisioning decisions. Therefore, cloud providers continually seek new ways to reduce consumption and 'oversubscribe' power, i.e., install more servers than allowed by the power budget based on server peak power consumption. For example, providers use DVFS to reduce power draw when servers are lightly utilized [1], they monitor and cap power to enable oversubscription and reduce provisioning costs [2], and they adapt power usage to varying green energy supply to lower emissions [3]. Such power management techniques reduce overall datacenter costs and improve their carbon efficiency.

Prior work on cloud efficiency has addressed CPUs and how to improve their power management mechanisms and interfaces. For example, CPUs expose a wide range of power and performance counters that guide their run-time power management decisions; they can scale frequencies per core, enabling fine-grained power reduction; they support configurable DVFS policies spanning performance- and energy-optimized settings [1]; and they guarantee limited-duration power spikes for better power provisioning and capping [4].

In contrast, GPUs have to date received less attention. Managing their power more efficiently is especially critical since they drive many vital modern workloads, such as deep learning (DL), high-performance computing (HPC), and graphics processing. Further, emerging GPUs are rated at up to 700W TDP, more than twice as much as a large server CPU. Given that servers typically host 4 to 16 GPUs, the power provisioned for a single GPU server may well exceed 4000W. Finally, the in-band power management mechanisms GPUs do offer, such as DVFS and power capping, are neither well documented nor characterized across a broad range of cloud applications.

To narrow these gaps, we comprehensively characterize power management features on modern server GPUs and identify their limitations from a cloud provider's perspective. We find that GPUs expose limited run-time information to orchestration platforms to enable dynamic power management decisions. Their DVFS policy is power hungry, lacks configurability, and performs poorly under power caps or with sharing. Further, they lack adequate power capping support since caps must be specified in band and can be exceeded due to power spikes and sub-optimal implementations. These pitfalls prevent providers from effectively deploying GPU power management at scale, at the cost of potentially millions of dollars and considerable stranded power [2], [5]. We recommend concrete actions for GPU vendor and researcher consideration to more efficiently power GPUs in the cloud.

2 Landscape of Cloud GPUs

We start by surveying cloud GPU deployments and their power management interfaces and mechanisms.

GPU Types. Different GPUs target different workloads. Cloud GPU workloads can be broadly categorized into graphics, HPC, and DL. Graphics workloads run well on GPUs that specialize in texture cores and Graphics-DDR memories, such as the NVIDIA A10 and RTX A5000. Compute workloads like HPC and DL run well on GPUs specialized with tensor cores and High-Bandwidth Memories (HBM), such as the NVIDIA A100 and T4. Both GPU types are commonly deployed in today's datacenters.

Virtualization. GPUs can be virtualized using fixed or hardware-mediated passthrough.[1] Clouds typically use fixed passthrough, such as Hyper-V's Discrete Device Assignment (DDA), to achieve near-native performance by dedicating full GPUs to individual virtual machines (VMs). In this scenario, the hypervisor completely relinquishes GPU control to the VM upon assignment, thereby preventing transparent, in-band power management by the cloud provider. In contrast, hardware-mediated passthrough via Single Root IO Virtualization (SR-IOV) shares GPUs across multiple VMs and the hypervisor, an approach that requires hardware support and is available only on newer GPUs. Perhaps owing to its newness, vendors do not currently offer fine-grained monitoring and management support for GPUs virtualized in this manner.

Power Provisioning. Historically, a safe rule of thumb for server power provisioning has been to allocate the rated Thermal Design Power (TDP) in expectation that the actual power usage will be lower [6]. However, GPUs may incur power spikes of up to 3$\times$ their TDP when powering on many on-chip components simultaneously [7]. In fact, recent Intel ATX 3.0 standards have legalized such limited-duration power excursions [8]. Poor adherence to power limits causes GPU vendors to recommend Power Supply Units (PSUs) over 2$\times$ the device TDP, causing significant power stranding, i.e., overprovisioned power that cannot be repurposed.

1. GPUs can also be virtualized using API remoting, which we do not discuss here because it is typically not deployed today.

Out-of-Band Management. Cloud providers require Out-Of-Band (OOB) power management for large-scale orchestration. It is also used as a reliable fallback if in-band power capping fails [2]. Unfortunately, no standardized OOB management protocol exists for GPUs today. Some vendors have proprietary OOB interfaces; e.g., NVIDIA's SMBPBI allows power caps to be set [9]. However, these protocols lack fine-grained telemetry and power control commands.

In-Band Management. GPU vendors provide various software tools for GPU monitoring and configuration on bare-metal and container clusters [10], [11]. Cloud providers cannot deploy these tools if the GPU is assigned to the VM under fixed-passthrough virtualization. However, we study them for insights into existing GPU power management features since it can help inform future mechanisms and interfaces.

NVIDIA provides two such tools: SMI [12] and DCGM [11]. SMI can configure GPU operating modes such as power caps, clock frequencies, and multi-tenancy, and can monitor basic run-time statistics, such as GPU utilization, memory usage, and power draw. DCGM, a management framework for GPU clusters, supports health checks and power management, and it can monitor performance counters like Streaming Multiprocessor (SM) occupancy, DRAM activity, and PCIe TX/RX usage.

Frequency Scaling. GPUs expose two clock domains—SM and memory—whose operating frequencies impact power draw. These frequencies can be varied via hardware DVFS policies or software frequency pinning. On NVIDIA GPUs, hardware DVFS is enabled by default and it can rapidly adapt voltage and frequencies based on utilization, thereby improving energy proportionality [13].[2] Frequency pinning, though slower, can be used in an application-aware manner to achieve desired performance-energy trade-offs [14].[3] The set of configurable frequencies depends on the GPU type, as shown in Table 1. SM clock frequencies on most NVIDIA GPUs can be set in 15MHz increments. Memory clocks are configurable on GDDR memory GPUs like the A5000 but not on HBM2 GPUs like the A100.

Power Capping. Power capping limits GPU power consumption to a software-specified value by throttling frequencies appropriately. This mechanism is similar to RAPL for CPUs, which cloud providers use for server power capping [2]. By default, GPU power caps are set to the device TDP but they can be configured to a lower range as shown in Table 1.

Sharing. Since many applications do not fully utilize GPUs, vendors have introduced temporal and spatial sharing to pack multiple workloads on the same device. Temporal sharing timeslices applications, whereas spatial sharing partitions the GPU to run applications in parallel. Cloud providers use NVIDIA's virtual GPUs (vGPUs) for timeslicing VMs and Multi-Instance GPU (MIG) partitions for spatial sharing [15]. Unfortunately, NVIDIA does not currently offer the ability to individually monitor or configure the power usage of VMs sharing a GPU.

3 GPU Power Management Analysis

We next characterize in-band power management features on modern GPUs and identify takeaways for future improvements.

3.1 Experimental Setup

We run 38 single-device workloads spanning three categories (Table 2) on three NVIDIA GPUs (Table 1). Workload/GPU combinations are run with a range of SM frequencies, memory frequencies, power caps, and MIG statuses. We use default benchmark configurations. DL training workloads are run for one epoch. DL inference workloads are run with different batch sizes using NVIDIA Triton `perf_analyzer` [16]. Each run is monitored using SMI and DCGM. We discuss a subset of results.

2. We refer to NVIDIA's hardware DVFS policy as DVFS in this paper.
3. Most NVIDIA server GPUs do not support explicit voltage control.

TABLE 1: Power cap and clock frequency ranges on our GPUs.

GPU	Power Caps	SM Freqs	Mem Freqs
A100	150–300 W	0.2–1.4 GHz	1.5 GHz
RTX A5000	100–230 W	0.2–2.1 GHz	0.4–8 GHz
RTX 6000	100–260 W	0.3–2.1 GHz	0.4–7 GHz

TABLE 2: Cloud GPU workloads.

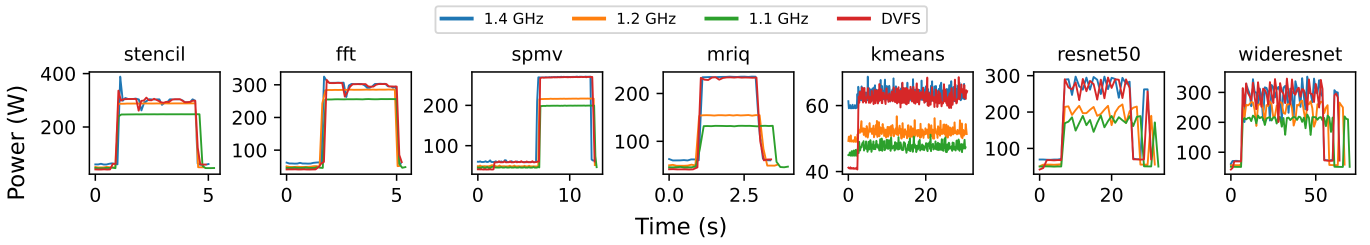
	Graphics	Deep Learning	HPC
Suite	Superposition [17]	Multiple [16], [18]	SPECAccel [19]
#Apps	8	12	18
GPU API	OpenGL	CUDA	OpenCL

3.2 Frequency Scaling

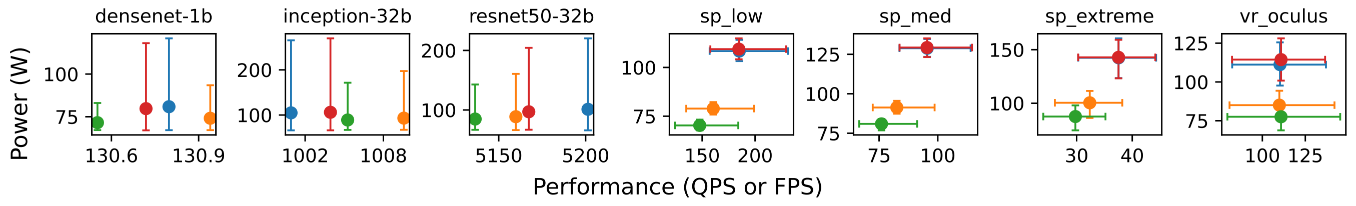
Power vs. Performance Trade-offs. Fig. 1 shows the impact of frequency scaling on power draw and performance on the A100 GPU. We make four observations. First, DVFS closely matches the highest frequency configuration. Second, based on the SM frequency sensitivity, we find there is a good mix of compute-bound and memory-bound workloads. About half of the HPC and DL workloads are memory-bound, while most graphics workloads are compute-bound. Third, for the memory-bound workloads, the performance difference between DVFS and 1.2GHz fixed frequency configuration is relatively small—1.6% lower on average. Finally, compute-bound workloads are very sensitive to SM frequencies: performance at 1.2GHz is 13% lower on average than DVFS.

Next, since peak power consumption drives power provisioning decisions, we investigate in Fig. 2(a) the peak power reduction opportunity under minimal performance loss. Specifically, for HPC and DL workloads, we limit performance loss to within 3% of DVFS. For graphics workloads, we use a minimum frame rate of 60 FPS as a service-level objective (SLO). We find that the workloads draw an average of 27.5W ($\sim$20%) lower peak power (up to 114W lower for inference workloads) with minimal performance loss using fixed frequencies. This result is an underestimate because we evaluate frequencies only in steps of 100+MHz—finer-grained steps would yield greater reductions. RTX GPUs show similar results; they provide an additional opportunity to reduce the memory frequency for compute-bound workloads with minimal performance impact.
T1: *GPU DVFS draws high peak power with no significant performance benefits for many workloads..*

Energy Efficiency. Batch workloads like HPC and DL training can benefit from reducing energy usage to lower costs and emissions [20]. To estimate the energy reduction opportunity, we compare the energy usage of fixed-frequency batch workload configurations to that of DVFS. Fig. 2(b) plots the maximum energy reduction across all evaluated frequencies relative to the energy used by DVFS. We find that the energy usage is typically minimized at middling clock frequencies rather than at the extremes [14]. This behaviour occurs because low frequencies significantly slow down workloads, whereas high frequencies draw considerable power. Since DVFS closely tracks the highest frequency configuration, it performs sub-optimally, resulting in a 1.3× higher energy usage (geomean) than that of the best frequency configuration across workloads.



(a) Power draw over time for five HPC and two DL training workloads. The X-axis implicitly indicates workload latency.



(b) Power draw vs. performance for three DL inference and four Graphics workloads. Circles show averages, and whiskers show 5th–95th percentiles.

Fig. 1: Power consumption and performance on A100 at different SM frequency configurations.

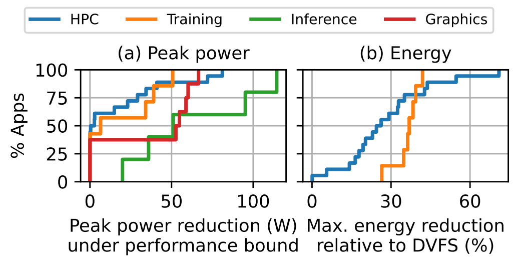


Fig. 2: (a) Peak power reduction under a performance bound. (b) Maximum energy reduction for batch workloads relative to DVFS.

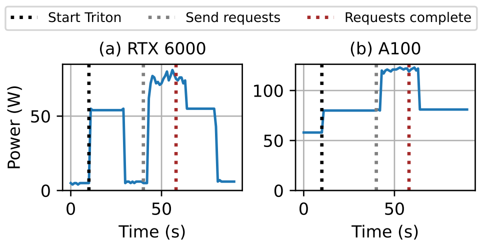


Fig. 3: GPU power draw timeseries for the Triton Inference Server on (a) RTX 6000 and (b) A100.

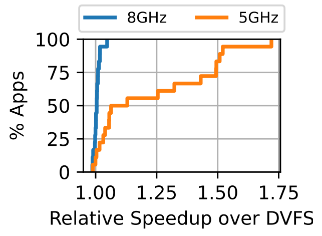


Fig. 4: Fixed memory frequency speedup over DVFS for HPC apps on A5000 with 100W cap.

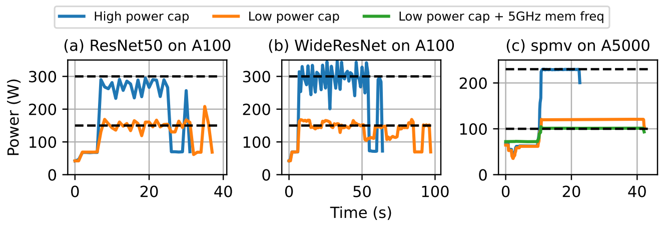


Fig. 5: Power cap violations under the threshold power limits.

T2: *GPU DVFS is energy inefficient for batch workloads.*

Idle Power Draw. If a GPU idles with compute state still loaded, DVFS runs it at the highest frequency for some duration (possibly forever), causing an unnecessarily high power draw. Fig. 3 shows an example by spinning up NVIDIA Triton instances on the RTX 6000 and A100 GPUs and making inference requests before letting the servers idle. For RTX 6000, DVFS runs at maximum frequency for about 20 seconds after GPU use and then reverts to minimum (∼50W higher). On A100, DVFS continuously runs at maximum frequency after Triton is loaded (∼20W higher than minimum). We observe similar behaviour when a CUDA context is opened, regardless of whether the GPU is actually used. Retaining high GPU frequency, potentially in anticipation of future work, is likely unnecessary since hardware DVFS can scale frequencies within nanoseconds [13].

T3: *Power draw is avoidably high on idle-but-allocated GPUs.*

Non-Intuitive Frequency Throttling. GPU clock frequencies are typically throttled when power or temperature limits are exceeded. Such throttling can prevent maximal utilization of the GPU. Unfortunately, we observe unexpected throttling patterns that could cause issues in production deployments. Specifically, A5000 throttles memory clock frequency by 5% from the maximum when running workloads, even if the frequency is explicitly configured otherwise and no power or thermal limits are reached. NVIDIA communicated that they implement such memory clock throttling for stable performance on memory-heavy workloads. However, we found that it triggers on all workloads that we tested.

T4: *Undocumented patterns in GPU frequency throttling may cause unexpected performance and power draw behaviours.*

3.3 Power Capping

Performance Under a Power Cap. Fig. 4 shows that for HPC workloads under a 100W power cap, DVFS performs 23% worse on average (up to 1.72×) than a fixed 5GHz memory frequency configuration on A5000. In contrast, DVFS performance closely matches that of the 8GHz (highest) memory frequency configuration. This occurs because a limited power budget must be distributed between SMs and memory. When throttling clocks, DVFS never scales down memory frequency below 5% from the maximum even though the SM frequency may throttled down to the minimum. Therefore, when running compute-bound workloads, power that could have been supplied to the

SMs to enable higher performance is unnecessarily consumed to maintain a high memory clock frequency.
T5: *GPUs poorly allocate power between SMs and memory.*

Power Limit Violations. Fig. 5 shows that workloads may exceed power limits under three scenarios: (a) a short power spike, typically at the start of an intensive phase of the workload—e.g., `ResNet50` and `stencil` exceed power limits by up to 38%; (b) a zig-zag pattern of power spikes for power-intensive, time-varying workloads—e.g., `WideResNet` causes repeated power spikes up to 15% over the power limit; and (c) a long, continuous power cap violation for memory-intensive workloads under a low cap (because DVFS does not throttle memory frequencies)—e.g., `spmv` under a 100W power cap on A5000 actually draws 120W. We confirm that `spmv` meets the power cap if the memory frequency is explicitly lowered to 5GHz. Power cap violations and a lack of fallback OOB capping mechanisms limit oversubscription since additional power headroom must be provisioned.
T6: *GPU power capping exceeds set limits, limiting oversubscription.*

3.4 GPU Sharing

Shared Clock Domains. All MIG partitions share the same GPU clocks, often causing high power usage if there is a mismatch in desired frequencies. To show this behaviour, we run `spmv` and `vgg16`-training separately on A100 with 2 MIG partitions, at each workload's lowest feasible SM frequency under a 3% bound on performance loss from DVFS. We also run them together under the same configuration. We then estimate the peak dynamic power (by subtracting idle power from measured power) in the three scenarios: (1) `spmv`-only draws 66W at 0.8GHz, (2) `vgg16`-only draws 100.9W at 1.4GHz, and (3) together they draw 192.9W at 1.4GHz, which is 26W higher than the sum when running them separately. When run together, the `spmv` partition is forced to operate at 1.4GHz to meet `vgg16`'s performance constraints.
T7: *Underutilized GPU partitions unnecessarily consume high power due to a shared clock domain.*

4 Call to Action

Based on these takeaways, we conclude by laying out recommendations for improving GPU power management in the cloud.

Improving DVFS. DVFS is performance optimized under TDP; however, it is wasteful from an energy perspective (***T1, T2***). GPUs should offer *software-configurable DVFS policies* that span a spectrum from performance to energy optimized, similar to Linux CPU power governors. Cloud providers or users can then configure the GPU based on their desired objectives [1]. Further, DVFS does not correctly throttle memory frequencies, which can lead to sub-optimal and occasionally incorrect behaviour (***T4, T5***). Since capping is critical to achieving oversubscription, *future DVFS policies should control memory frequencies* so that workloads can still achieve good performance under a cap.

Reducing Idle Power. GPUs are often allocated but idle; for example, the median GPU job on HPC clusters is idle for 16% of its lifetime [5]. Such GPUs may consume higher power than necessary (***T3***), reducing overall energy efficiency and oversubscription. Given fast hardware DVFS support [13], *idle-but-allocated GPUs should be run at the lowest frequency*, similar to CPUs idling in low power C-states with applications loaded in memory. Furthermore, the *controls for sending the GPU into these lower power states should be exposed to the software*.

Reliably Power Capping. GPU power spikes can significantly exceed power caps, which limits oversubscription (***T6***). To resolve this, GPU vendors should *restrict power spikes to fixed, small time constants* similar to CPUs. GPUs also currently support a narrow range of power caps (Table 1). Emulating lower caps requires software to explicitly set frequencies, which is slow and error prone. Instead, GPUs should *offer a wider range of power caps* to simplify capping and improve oversubscription.

Partitioning Clock Domains. MIG partitions may wastefully draw power due to a shared clock domain (***T7***). It would be worthwhile to *explore whether the operational power savings from separating clock domains (similar to CPU cores) is more beneficial than its design complexity and costs.*

Developing Virtualized Management. Current out-of-band GPU monitoring and management mechanisms are vendor-specific and incompatible with virtualized deployments, as noted in Sec. 2. It is critical to *develop and standardize cloud GPU power management interfaces* to realize benefits at scale. Providers would particularly benefit from *detailed run-time metrics and configurability per GPU (or per MIG partition) at low overhead*, both OOB and in band.

Acknowledgments

We thank Srikant Bharadwaj, Sandy Kaplan, Pratikkumar Patel, Muralikrishna Rao, Gaurav Uppal, and Huseyin Yildiz. This work is supported by NSF CNS-2104548, VMware, and Cisco.

References

[1] D. Lo and C. Kozyrakis, “Dynamic management of TurboMode in modern multi-core chips,” in HPCA, 2014.
[2] A. G. Kumbhare et al., “Prediction-based power oversubscription in cloud platforms,” in USENIX ATC, 2021.
[3] N. Bashir et al., “Enabling sustainable clouds: The case for virtualizing the energy system,” in SoCC, 2021.
[4] “12th Generation Intel Core Processors, Datasheet,” Intel, 2022.
[5] B. Li et al., “AI-enabling workloads on large-scale GPU-accelerated system: Characterization, opportunities, and implications,” in HPCA, 2022.
[6] X. Fan et al., “Power provisioning for a warehouse-sized computer,” ACM SIGARCH Computer Architecture News, 2007.
[7] A. Mpitziopoulos. Intel's ATX v3.0 PSU Standard Has More Power for GPUs. [Online]. Available: https://www.tomshardware.com/news/intel-atx-v3-psu-standard
[8] Intel Introduces New ATX PSU Specifications. [Online]. Available: https://www.intel.com/content/www/us/en/newsroom/news/intel-introduces-new-atx-psu-specifications.html
[9] NVIDIA A30 GPU Accelerator Product Brief. [Online]. Available: https://www.nvidia.com/content/dam/en-zz/Solutions/data-center/products/a30-gpu/pdf/a30-product-brief.pdf
[10] AMD ROCm. [Online]. Available: https://www.amd.com/en/graphics/servers-solutions-rocm
[11] NVIDIA Data Center GPU Manager. [Online]. Available: https://developer.nvidia.com/dcgm
[12] NVIDIA System Management Interface. [Online]. Available: https://developer.nvidia.com/nvidia-system-management-interface
[13] S. Bharadwaj et al., “Predict; do not react for enabling efficient fine grain DVFS in GPUs,” arXiv, 2022.
[14] Z. Tang et al., “The impact of GPU DVFS on the energy and performance of deep learning: An empirical study,” in e-Energy, 2019.
[15] NVIDIA A100 80GB PCIe GPU Product Brief. [Online]. Available: https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/a100/pdf/PB-10577-001_v02.pdf
[16] NVIDIA Triton Inference Server. [Online]. Available: https://github.com/triton-inference-server/server
[17] Superposition 2017. Unigine. [Online]. Available: https://benchmark.unigine.com/superposition
[18] GitHub. PyTorch CIFAR100. [Online]. Available: https://github.com/weiaicunzai/pytorch-cifar100
[19] SPEC ACCEL benchmark suite. [Online]. Available: https://www.spec.org/accel/
[20] J. You et al., “Zeus: Understanding and optimizing GPU energy consumption of DNN training,” in NSDI, 2023.