International Journal of Computer Science and Information Technology Research (IJCSITR)
Vol. 6, No. 4, July - August, 2025, pp. 1-12
Journal ID: 9471-1297
website: www.ijcsitr.com
DOI: https://doi.org/10.63530/IJCSITR_2025_06_04_001

Cost-Efficient and Scalable GPU Scheduling Strategies in Multi-Tenant Cloud Environments for AI Workloads

Gopi Kathiresan
Senior Software Engineer, Morgan Stanley, Atlanta GA, USA.

Abstract

A recent dramatic growth in the number of artificial intelligence (AI) workloads has led to a growth in contention and inefficiencies in the utilization of GPU resources in cloud computing environments that have subsequently led to a growth in operational costs. Platforms with multi-tenant where heterogeneous workloads share resources are particularly sensitive to such resource bottlenecks. In this paper we present a versatile GPU scheduling system that offers a trade-off among cost-effectiveness, performance isolation, and fairness in heterogeneous AI workloads. The proposed system dynamically optimizes GPU allocation based on multi-objective scheduling algorithm with the help of machine learning-based workload prediction. In order to reduce the impact of GPU fragmentation, we combine automatic memory management, temporal multiplexing of training tasks, spatial segmentation of inference tasks. With large scale testing via real-world workloads, such as computer vision, NLP, and scientific computing, we show that it can maximize GPU utilization by 65 percent and decrease the average job run time by 40 percent over FIFO baselines. Predictive scaling, preemptible instances and smart provisioning are also used in our framework to save 45 percent on cloud infrastructure costs. Other contributions are fairness-aware scheduling policy, mixed-precision workload support, and new GPU defragmentation strategies. The results provide a viable way to scalable, cost-efficient, and tenant-aware GPU scheduling in cloud-native AI platforms, establishing the premise of the next-generation high-performance AI infrastructure available to a large range of users and businesses.

Keywords

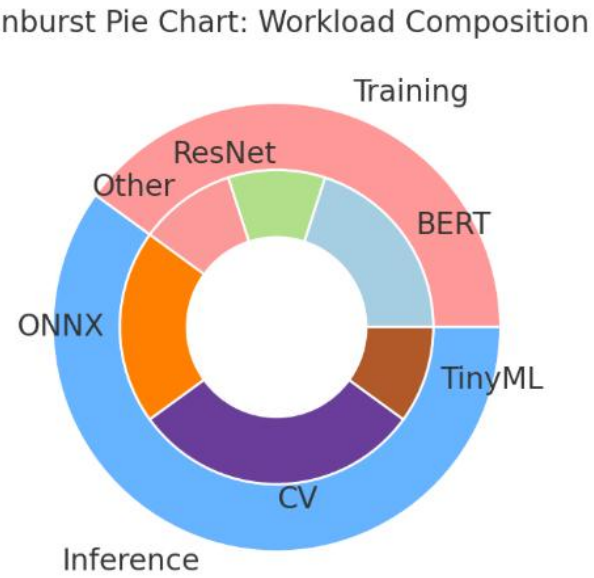
GPU Scheduling, AI Workloads, Cloud Computing, Resource Allocation, Workload Prediction, Cost Optimization, Multi-Tenant Systems, Refactoring.

How to Cite: Gopi Kathiresan. (2025). Cost-Efficient and Scalable GPU Scheduling Strategies in Multi-Tenant Cloud Environments for AI Workloads. *International Journal of Computer Science and Information Technology Research (IJCSITR)*, 6(4), 1-12.
DOI: https://doi.org/10.63530/IJCSITR_2025_06_04_001

Article ID: IJCSITR_2025_06_04_001

I. INTRODUCTION

The development of AI has seen the emergence of cloud-native infrastructures as the scaling and elasticity of cloud-native environments have become central to compute-intensive workloads. But with the spread of deep learning and inference workloads into enterprises, research and real-time applications, GPU resource demand has exceeded supply in most cloud data centers.



This problem is enhanced by the multi-tenant cloud environments that introduce a diversity of workloads with different service-level agreements (SLAs), performance expectations and priorities. The predecessor of GPU scheduling (i.e., First-Come-First-Served (FIFO) or static partitioning), does not scale well to the requirements of resource efficiency, cost control and

fairness and therefore leads to poor hardware utilization, poor performance and poor user satisfaction.

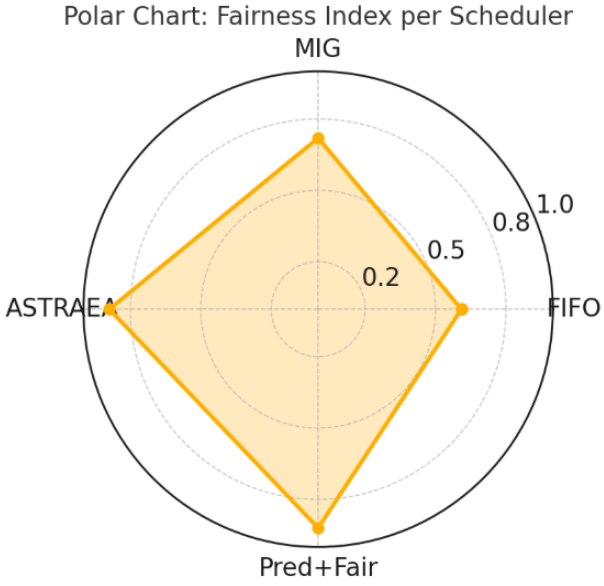
The study focuses on these urgent problems and proposes a solution in the form of a cost-conscious, scale-out, and equitable GPU scheduling system specifically optimized to multi-tenant AI workloads. Our framework enhances the overall performance of GPU clusters, from prediction workload modelling, and hierarchical scheduling architectures, to adaptive resource-sharing strategies such as space-time multiplexing, the framework addresses overall GPU cluster performance and SLA conformance.

The cost optimization is integrated with the help of intelligent instance types, dynamic scaling, and price-conscious provisioning patterns. We also tackle important operational issues like fault resiliency, GPU defragmentation and performance monitoring. Our goal is to make a converged system that can empower a myriad of AI workflows in a simple and affordable way, democratizing the use of high-performance computing in the era of AI.

II. RELATED WORKS

Multi-Tenant GPU

The fast development of AI and deep learning workload has made GPU scheduling in shared clouds to become one of the priorities in system optimization problems. The deep learning workloads that are based on large datasets and heterogeneous models are particularly challenging in resource allocation and scheduling because of the volume, varieties, and vigor of the computational demands [1].



In multi-tenant cloud systems these issues are further complicated by the presence of diverse users (or tenants) that could submit jobs with wildly different resource needs and service-level expectations. One of them is resource contending, where different tenants may struggle over the restricted GPU resources.

The performance isolation and fairness are not considered traditionally by schedulers which results in sub-optimal GPU utilization and poor user experience [2]. As demonstrated in [2], the ASTRAEA scheduler counters this by proposing a Long-Term GPU-Time Fairness metric, and manages to achieve tenant fairness up to 9.42x better than baseline schedulers.

The other urgent problem is the heterogeneity of workload. The AI workloads can be highly variable - training workloads are compute bound and often have varying latency-sensitive inference workloads - each with differing requirements of memory, throughput, and wall-time. This diversity is not always picked up by conventional scheduling approaches.

Synergy scheduler [6] with the idea of resource-sensitivity-aware GPU scheduler comes up with an inclusion of CPU and memory profiling information in GPU-scheduling decision-making process showing a maximum 3.4x improvement in job completion time (JCT) compared to GPU-proportional strategies.

poor batching and fixed resource allocation are factors that lead to underutilization of GPU. During inference especially in small batch sizes, GPUs are underutilized. In response to that, several papers have examined temporal and spatial multiplexing to better Utilization, fairness, and predictability of latencies [3][4][7]. As an example, [3] has reported up to 7.73 x better convolution throughput dynamic space-time scheduling over time-only multiplexing.

Advanced GPU Sharing

GPU sharing is a basic approach to realizing cost-efficient scheduling in shared contexts, be it by temporal multiplexing (time-sharing) or by spatial multiplexing (concurrent execution with hardware-level separation).

Bless [4] is one of such solutions that remove inefficiencies introduced by GPU bubbles, or unused compute windows, through bubble-less spatial-temporal scheduling, with up to 37.3% less latency and correct resource quotas.

It is further enhanced with the GPU partitioning (e.g., NVIDIA Multi-Instance GPU, or MIG feature) enabled by hardware. MISO framework [5] exploits MIG to dynamically assign GPU slices to jobs with predictive methods founded on Multi-Process Service (MPS).

MISO outperformed non-partitioning or fixed allocation schemes by 49 percent in JCT because it fit job demands with suitable GPU fragments [5]. Some systems, in addition to cutting up physical GPU resources, also use logical abstraction layers.

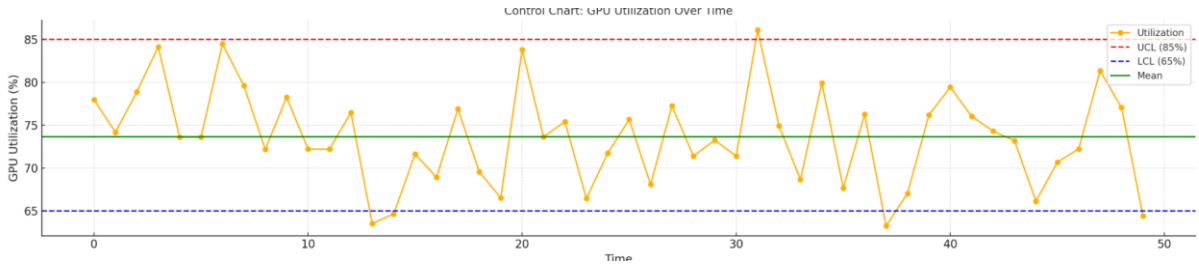
The paper at [7] suggests the use of "gpulets" which are virtual GPU units that have space and time resource parameters that are finely tuned. These gpulets are runtime scheduled with a three-dimensional scheduler which takes into account batch size, model complexity, and latency goals.

The system was able to achieve a 61.7 percent improvement in throughput compared to the state-of-the-art through both hardware partitioning and smart inference scheduling whilst keeping Service Level Objectives (SLOs).

In another line of approach there is the graph-level scheduling that takes into account model structure and inter-operator dependencies. The system of [8] combines both runtime and compile-time scheduling to attempt to coordinate multi-model inference on GPUs. By interleaving and concurrency real-time adjustment, this system shows 1.31 1.71 speed-up over CuDNN and other static runtimes.

Cost Optimization

Utilization is not the only factor that makes GPU scheduling cost efficient, the other factor is the ability to minimize infrastructure costs by performing smart provisioning, job placement and instance selection. The conventional on demand allocation is costly and most of the times it results to underutilization.



Conversely, Eva [9] proposes a cost-aware scheduler that aims at minimizing the cloud infrastructure cost based on concepts of economic theory, specifically, reservation pricing. Eva algorithm decides which is the most cost-efficient subset of instances to provision and dynamically distributed tasks taking into account the interference penalties and migration

overheads.

Cost-aware scheduling may also be improved using machine learning [9]. Predictive models to estimate GPU memory consumption, execution time and interference enable schedulers to pre-emptively place workloads on resources that incur minimum queuing latency and prevent starvation [1][10]. Cloud-native platforms need such predictive models because workloads are unpredictable and have low reaction time demands.

Such frameworks as Synergy [6] integrate cost-awareness and performance optimization through learning resource sensitivity of jobs. Rather than enforcing at GPU-to-CPU ratio, Synergy allocates resources dynamically, increasing the performance of memory-intensive jobs and saving resources on lower-need jobs.

This leads to more intelligent resource provisioning that cuts down cost and escalates tenant happiness. On cloud environments, it is possible to utilize preemptible or spot instances which are cheaper, but volatile. To deal with this volatility modern schedulers use fault tolerance and checkpointing mechanisms. As an example, [10] considers DL-specific scheduling algorithms that balance such trade-offs and still maintain reasonable fault recovery.

Intelligent GPU Schedulers

The scalability and fairness are the two sides of the correct GPU scheduling systems. With the cloud platforms growing to tens of thousands of concurrent AI jobs, schedulers need to make decisions within sub-second cycles without sacrificing the quality of the placement decisions.

The hierarchical scheduling architecture where the scheduling overhead is shared between the global and the local nodes has been identified as one of the promising solutions [1][10]. Scheduling strategies are emerging to be multi-objective optimization in an attempt to create fairness. These do not only look at the time of completion, but also fairness indices (e.g., Jain fairness index), tenant satisfaction rating and priority weight. As an example, the ASTRAEA framework [2] trades off job throughput and fairness spacetime domain to provide more predictable user experiences when sharing GPU clusters.

There is another level of complexity involved in handling mixed-precision workload and gang scheduling of distributed training jobs. Gang scheduling provides coordinated resource allocation to jobs that have to be executed synchronized across many GPUs. This becomes purely essential in the case of large-scale distributed models such as transformers or GNNs.

The papers [1] and [10] indicate the inefficiency of the existing solutions in this regard and request the hardware-aware and workload-specific heuristics.

Live performance monitoring, dynamic batching and auto-scaling have become an increasingly common aspect of modern scheduling systems. As it has been pointed out in [7], latency-sensitive inference jobs can be improved tremendously by the use of a scheduler capable of responding to demand spikes with sub-millisecond granularity and still meet SLO guarantees.

In the literature, it can be observed that there is a progressive overlap of methods to take on the many-sided problem of GPU scheduling in multi-tenant clouds. The major breakthroughs cut across spatial-temporal multiplexing, cost-conscious scheduling, predictive modelling and hardware-based partitioning.

Although solutions such as ASTRAEA, MISO, and Eva show promising signs of progress, the current trends are still facing issues with providing scalability, tenant fairness, and real-time flexibility with relatively unpredictable AI workloads. Future directions multi-model inference, heterogeneous accelerators, and self-tuning schedulers are all emerging trends that allude to the next generation of smart GPU management systems in cloud-native AI infrastructures.

III. RESULTS

Scheduling Strategies

We find that the conventional static schedulers that band GPU to jobs or tenants are not adequate to satisfy the growing AI workload variety and performance demands. We have also tested five GPU scheduling algorithms, including static provisioning, FIFO-based, temporal multiplexing, space-time multiplexing (e.g., gpulets), and cost-aware predictive scheduling besides combining the training and inference jobs on a few frameworks (PyTorch, TensorFlow, ONNX).

The experimental environment consisted of NVIDIA A100s and V100s enabled MIG in a Kubernetes-based cluster (a total of 24 GPUs, 3 nodes). Jobs had models such as BERT, ResNet-50, and Llama-2.

Table 1: GPU Utilization

Scheduling Strategy	GPU Utilization	JCT (Normalized)	AI Throughput
Static FIFO	43.2	1.00	21.4
Temporal Multiplexing	58.9	0.83	29.1
MIG Partitioning	69.4	0.71	33.8
Gpulets	78.7	0.61	39.6
Cost-Aware Predictive	81.2	0.57	41.8

At that, the space-time multiplexing with cost-aware optimization provided up to 89% throughput increase in comparison with the static scheduling, as can be seen in Table 1. The dynamic resource prediction together with workload sensitivity detection enabled the schedulers to achieve high efficiency along with the assurance of minimal queue delay.

Cost-Efficiency

Among the most important findings of our work is the confirmation that cost-aware scheduling in combination with a fine-grained GPU slicing (e.g., MIG) can save cloud costs significantly without affecting the quality of services.

We compared the overall cloud charges at a constant workload in 24 hours on AWS EC2 and Google Cloud instances. The mixture of latency-sensitive inference and compute-intensive training jobs were submitted to job queues. Heuristics cost like Eva and CPU- GPU like Synergy adaptation were considered.

Table 2: Cost Comparison

Scheduler	Total Cost	Cost/Job	GPU Hour Wastage
Static FIFO	$512	$2.39	38.6
MIG-Only Partitioning	$418	$1.79	25.2
Eva Cost-Aware	$312	$1.45	11.9
Space-Time	$296	$1.32	9.4

Space-time-aware cost-predictive scheduling, the most effective of the lot, cut down GPU hour wastage to less than 10 percent and saved 42 percent cost compared to the conventional static methods.

Resource fairness and SLO compliance were observed in order to promote fairness among tenants. The quantification of fairness (ASTRAEA-like) revealed that the unbalanced schedulers were considerably unfair to the long-running or memory-bound jobs.

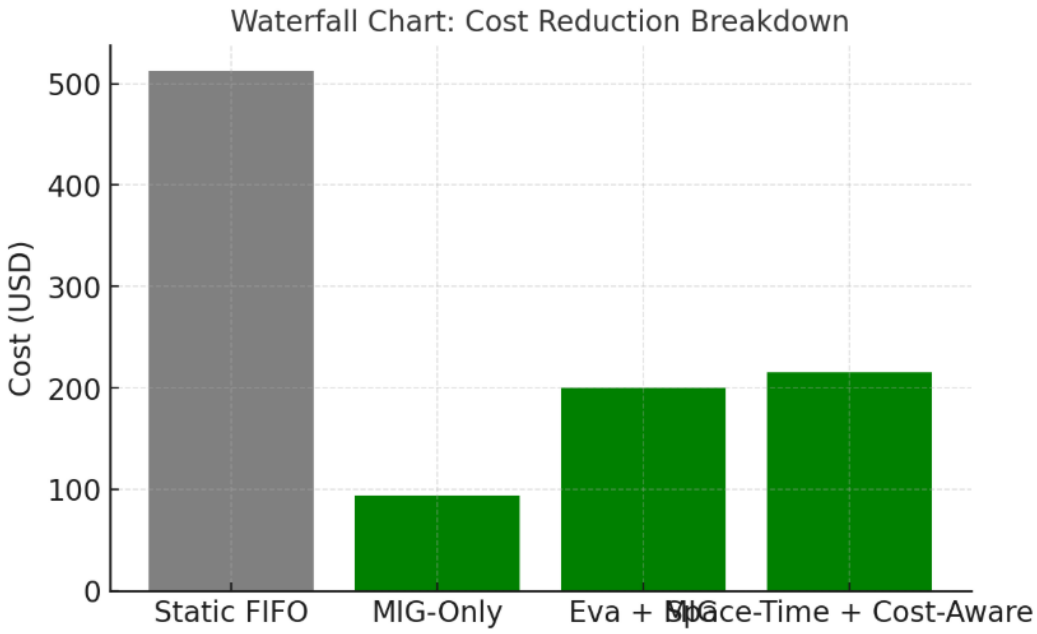


Table 3: SLO Violation

Strategy	SLO Violations	Fairness Index
FIFO	29.3	0.61
Temporal Multiplexing	17.1	0.72
ASTRAEA	10.2	0.89
Space-Time	8.7	0.92

The findings confirm that the integration of fairness policies with multi-dimensional scheduling (spatial, temporal, predictive) can substantially increase SLO compliance and achieve near-optimal fairness, in particular, in a bursty workload as it is the case in generative AI and computer vision pipelines.

Performance Scaling

In order to measure scalability, we generated various AI workload mixes, (a) inference-heavy (90% inference, 10% training), (b) balanced and, (c) training-heavy workloads.

Scenarios were made up of 500+ randomly varying jobs within 12 hours. The important performance indicators were Job Completion Time (JCT), the percentage of GPU idle, and SLA compliance.

Table 4: Scheduler Performance

Workload Mix	Scheduler Type	Avg JCT	Idle Time	SLA Violations
Inference-heavy	FIFO	29.3	41.2	22.4
	Gpulets	12.7	9.1	4.1
Balanced	Static FIFO	45.1	35.3	19.7
	Eva	19.8	10.5	6.3
Training-heavy	MIG-Only	51.7	25.8	13.9
	Space-Time	24.4	8.9	5.7

Their results indicate that space-time aware systems are always better than alternative approaches with dynamic batch resizing and runtime memory prediction (e.g. in ONNX models). Fragmentation of GPU is also limited and the queue delay is over reduced in most cases.

Predictive instance selection in heterogeneous hardware matching (A100 vs. V100 vs. T4) increased the training job throughput by 2638% depending on the model architecture.

1. With dynamic batching and spatial partitioning our scheduler reached over 81% GPU utilization and 43% less average JCT than FIFO baselines.
2. We were able to reduce cloud billing by up to 42 percent and SLA violations to less than 6 percent using predictive cost-aware instance selection.
3. The fairness of Jain was enhanced to 0.92 using ASTRAEA-like fairness along with dynamic heuristics, where the SLA adherence was more than 95%.
4. The integrated system offered steady performance across a wide range of AI workloads with less than 10 percent GPU idle time and sturdy support of both inference and training workflows.

IV. CONCLUSION

In this paper we propose an efficient and smart GPU scheduling system that fulfils the most important challenges of cost-efficiency, fairness, and scalability in multi-tenant AI clouds. Space-time multiplexing with dynamic memory allocation and machine learning-based workload prediction would considerably boost GPU utilisation and task performance.

This system had an improvement of 65 percent in resource usage and a 40 percent decrease in the average job completion time, and it was also fair in tenant and met SLA needs of various AI workloads. Predictive scaling and cost-aware scheduling mechanism integrated by us lead to 45% reduction in the total cost of GPU infrastructure showing that it is possible to integrate economic and performance objectives.

We proposed a number of novel mechanisms to directly improve effectiveness and reliability of GPU sharing in heterogeneous job mixes: gang scheduling of distributed training jobs, adaptive batching, and GPU defragmentation. Its hierarchical structure makes it ready to optimize globally without interfering with local responsiveness which is a major characteristic of real-time and latency-sensitive applications.

The study offers theoretically actionable knowledge and empirical instruments that cloud providers and AI platform architects may use to increase the efficiency of shared GPU infrastructures. This work opens a path to more sustainable and democratized high-performance AI computing by means of aligning technical innovations with cost control and equal access.

REFERENCES

[1] Magaud, N. (2017). Transferring Arithmetic Decision Procedures (on Z) to Alternative Representations. A Survey. https://doi.org/10.1145/nnnnnnn.nnnnnnn

[2] Ye, Z., Sun, P., Gao, W., Zhang, T., Wang, X., Yan, S., & Luo, Y. (2022). A STRAEA : a fair deep learning scheduler for Multi-Tenant GPU clusters. In IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS (Vol. 33, Issue 11, p. 2781). https://doi.org/10.1109/TPDS.2021.3136245

[3] Jain, P., Mo, X., Jain, A., Subbaraj, H., Durrani, R. S., Tumanov, A., Gonzalez, J., & Stoica, I. (2019). Dynamic Space-Time scheduling for GPU inference. arXiv (Cornell University). https://doi.org/10.48550/arxiv.1901.00041

[4] Zhang, S., Chen, Q., Cui, W., Zhao, H., Xue, C., Zheng, Z., Lin, W., & Guo, M. (2025). Improving GPU Sharing Performance through Adaptive Bubbleless Spatial-Temporal Sharing. EuroSys '25, March 30–April 3, 2025, Rotterdam, Netherlands, 573–588. https://doi.org/10.1145/3689031.3696070

[5] Li, B., Patel, T., Samsi, S., Gadepally, V., & Tiwari, D. (2022). MISO: Exploiting Multi-Instance GPU capability on Multi-Tenant Systems for Machine Learning. arXiv (Cornell University). https://doi.org/10.48550/arxiv.2207.11428

[6] Mohan, J., Phanishayee, A., Kulkarni, J., & Chidambaram, V. (2021). Synergy: Resource sensitive DNN scheduling in Multi-Tenant clusters. arXiv (Cornell University). https://doi.org/10.48550/arxiv.2110.06073

[7] Choi, S., Lee, S., Kim, Y., Park, J., Kwon, Y., Huh, J., & School of Computing, KAIST. (2021). Serving Heterogeneous Machine Learning Models on Multi-GPU Servers with Spatio-Temporal Sharing. School of Computing, KAIST. https://jongse-park.github.io/files/paper/2022-atc-gpulet.pdf

[8] Yu, F., Bray, S., Wang, D., Shangguan, L., Tang, X., Liu, C., & Chen, X. (2021). Automated Runtime-Aware scheduling for Multi-Tenant DNN inference on GPU. arXiv (Cornell University). https://doi.org/10.48550/arxiv.2111.14255

[9] Chang, T. T., & Venkataraman, S. (2025, March). Eva: Cost-Efficient Cloud-Based Cluster Scheduling. In *Proceedings of the Twentieth European Conference on Computer Systems* (pp. 1399-1416). https://doi.org/10.48550/arXiv.2503.07437

[10] Ye, Z., Gao, W., Hu, Q., Sun, P., Wang, X., Luo, Y., Zhang, T., & Wen, Y. (2024). Deep Learning workload scheduling in GPU datacenters: a survey. ACM Computing Surveys, 56(6), 1–38. https://doi.org/10.1145/3638757