# CPSC 526 Literature Review: : GPU Cluster Management

Evan Shi
*Yale University*

## 1 Introduction

The rapid growth of artificial intelligence (AI) and machine learning (ML) applications has necessitated the development of highly efficient and scalable computing infrastructures. Among these, GPU clusters have emerged as a cornerstone for training complex models such as deep learning (DL) frameworks and large language models (LLMs). The efficient management of these clusters is critical to addressing challenges such as scalability, fault tolerance, and resource utilization. This literature review explores the state-of-the-art in GPU cluster management, with a focus on the transformative MegaScale system designed for scaling LLM training to unprecedented levels. Complementary systems like Tiresias, DxPU, EDL, and Aryl are also examined to provide a comprehensive perspective on advancements in the field.

## 2 Challenges in GPU Cluster Management

Managing GPU clusters is a complex task, involving heterogeneous workloads, dynamic resource demands, and intricate communication requirements. Traditional cluster management frameworks, including Apache Mesos [4] and YARN [7], were originally designed to handle general-purpose distributed computing tasks. These systems employ static resource allocation strategies and are optimized for batch processing and fault-tolerant applications such as Hadoop MapReduce [1]. However, GPU-centric workloads require dynamic memory allocation, low-latency inter-GPU communication, and efficient scheduling to accommodate the high computational demands of AI and ML tasks. The lack of support for fine-grained resource management and real-time decision-making in these frameworks limits their effectiveness in GPU-intensive environments, leading to underutilization and increased job queuing times [8] [2]. This section outlines key challenges and their implications for cluster performance.

### 2.1 Scalability and Resource Fragmentation

The exponential increase in the size and complexity of DL and LLM models places significant demands on GPU clusters, which now often span tens of thousands of GPUs. Fixed resource allocation strategies lead to inefficiencies such as resource fragmentation, where underutilized CPUs and GPUs in host servers diminish overall performance. Advanced strategies like disaggregation and elastic scaling have emerged as promising solutions to mitigate these issues. For instance, DxPU implements GPU disaggregation, pooling GPUs for flexible allocation, thereby reducing fragmentation and supporting dynamic scaling across data centers [3].

### 2.2 Fault Tolerance and Stability

Training large models over extended periods introduces significant risks of failures and stragglers, which can disrupt system stability and prolong training times. Robust fault tolerance mechanisms are indispensable. Systems like MegaScale employ advanced checkpointing techniques and anomaly detection to localize faults and recover quickly, ensuring minimal disruption during large-scale operations [5].

## 3 MegaScale: Scaling Large Language Models

MegaScale represents a paradigm shift in GPU cluster management, particularly for training LLMs at scales exceeding 10,000 GPUs. Its full-stack algorithm-system co-design optimizes efficiency and stability, addressing critical bottlenecks in training operations.

### 3.1 Efficiency at Scale

MegaScale achieves a 55.2% Model FLOPs Utilization (MFU) during the training of a 175-billion-parameter model on 12,288 GPUs, marking a 1.34 exttimes improvement over Megatron-LM. This is accomplished through advanced mixed parallelism strategies, which include data parallelism, tensor

parallelism, pipeline parallelism, and sequence parallelism, each tailored to address specific computational bottlenecks:

- Data Parallelism: Model replicas are distributed across GPUs, and each processes a different subset of the training data. Gradients are aggregated using optimized communication strategies such as reduce-scatter and all-gather operations, ensuring minimal synchronization overhead.
- Tensor Parallelism: Large matrix operations, such as those in transformer blocks, are split across GPUs. This approach leverages fine-grained computation sharing, reducing the per-GPU memory footprint while ensuring computational efficiency. Inter-GPU communication is minimized by confining tensor operations to closely connected devices.
- Pipeline Parallelism: Model layers are divided across GPUs, allowing different stages of computation to run concurrently. MegaScale employs an interleaved 1F1B (one forward, one backward) scheduling mechanism to balance workload across GPUs, reducing pipeline stalls and maximizing utilization.
- Sequence Parallelism: Operators with high memory demands, such as LayerNorm and Dropout, are partitioned along the sequence dimension. This reduces activation memory usage and enables efficient scaling to longer input sequences.

By combining these parallelism strategies into a cohesive framework, MegaScale optimally utilizes the computational and memory resources available, enabling balanced workloads and maximizing resource efficiency [5].

### 3.2 Stability and Observability

MegaScale integrates a robust observability framework designed to address the unique challenges of large-scale LLM training. It employs a multi-layered diagnostic approach that combines fine-grained system monitoring with advanced real-time analytics. Custom-built diagnostic tools provide granular visibility into hardware performance, such as GPU utilization, memory bandwidth, and network latency, while software-level metrics track computational bottlenecks and synchronization delays. In-depth observability is further enhanced by tools like distributed tracing, which captures the interdependencies and communication patterns between GPUs, and event-driven logging systems that generate actionable insights from anomalies. To address failures and stragglers, MegaScale uses an automated fault localization mechanism that employs heartbeat messages and fine-grained system telemetry to identify problematic nodes or processes in real-time.

Advanced checkpointing mechanisms are designed for high efficiency. These include segmented and parallelized model state checkpoints, which enable rapid recovery by allowing individual chunks of the model to be reloaded independently. Additionally, the system supports incremental checkpointing, where only updated states are saved, minimizing both time and storage overhead. This comprehensive design ensures that training jobs can continue with minimal disruption, maintaining high efficiency and stability even in the face of frequent failures or unpredictable stragglers during extended training runs [5].

### 3.3 Communication Optimization

Inter-GPU communication is a significant bottleneck in large-scale training. MegaScale addresses this challenge through a combination of advanced communication strategies and network optimization. Non-blocking asynchronous operations are employed to decouple communication from computation, allowing GPUs to overlap data transfer with computation cycles. This includes techniques such as overlapping reduce-scatter and all-gather operations with forward and backward passes to minimize idle GPU time.

MegaScale also implements optimized network topologies tailored to minimize inter-GPU latency. Custom tree-based communication hierarchies are designed to reduce the number of hops required for data transfer between GPUs, while advanced ECMP (Equal-Cost Multi-Path) configurations reduce hash conflicts and improve bandwidth utilization. Congestion control is further enhanced through dynamic packet routing, which actively avoids congested paths in the network fabric. Additionally, retransmission timeout parameters are fine-tuned to minimize delays caused by packet loss, ensuring smoother data flow and higher throughput. These measures collectively optimize the overall inter-GPU communication, significantly improving scalability and efficiency for large-scale LLM training [5].

## 4 Innovations in GPU Cluster Management

Several complementary systems have introduced novel approaches to GPU cluster management, addressing specific challenges and advancing the state-of-the-art.

### 4.1 Elastic Scaling

Elastic scaling is a pivotal innovation in GPU cluster management. Systems like EDL and Aryl dynamically adjust GPU allocations to optimize resource utilization. EDL employs stop-free scaling and dynamic data pipelines, significantly reducing the overhead associated with parallelism adjustments [8]. Aryl extends this concept by enabling capacity loaning, wherein idle inference GPUs are temporarily repurposed for training tasks, thereby improving overall cluster efficiency [6].

## 4.2 Disaggregated Architectures

DxPU pioneers GPU disaggregation, decoupling GPUs from host servers to create a flexible resource pool. This architecture enables dynamic GPU allocation across data centers, reduces hardware maintenance costs, and enhances resource utilization. By implementing disaggregation at the PCIe level, DxPU achieves minimal performance overhead while scaling effectively to data center levels [3].

## 4.3 Advanced Scheduling

Tiresias exemplifies cutting-edge advancements in scheduling algorithms. By employing the Discretized Two-Dimensional Gittins index, Tiresias balances resource contention and reduces job completion times. Its information-agnostic design allows efficient scheduling even in the absence of precise job duration predictions [2].

# 5 Trends and Future Directions

The future of GPU cluster management is shaped by emerging trends that aim to address the limitations of current systems. Key areas of focus include:

- **Disaggregated Architectures:** Systems like DxPU highlight the potential of disaggregation to redefine data center designs by enabling flexible and efficient resource management.
- **Enhanced Elasticity:** Elastic scaling, as demonstrated by EDL and Aryl, will continue to evolve, reducing overhead and improving cluster responsiveness.
- **Observability and Fault Tolerance:** Comprehensive monitoring and diagnostic tools, as seen in MegaScale, will become standard, ensuring resilience against large-scale failures.

# 6 Conclusion

Advances in GPU cluster management have been driven by the escalating demands of AI workloads. Systems like MegaScale, DxPU, and Aryl exemplify innovative approaches to addressing challenges of scalability, efficiency, and fault tolerance. MegaScale, in particular, sets a new standard for LLM training, enabling operations at unprecedented scales with high efficiency and stability. These advancements not only enhance the capabilities of current GPU clusters but also lay the foundation for the next generation of AI infrastructure.

# References


[1] Dean, J., and Ghemawat, S. Mapreduce: simplified data processing on large clusters. *Commun. ACM 51*, 1 (Jan. 2008), 107–113.

[2] Gu, J., Chowdhury, M., Shin, K. G., Zhu, Y., Jeon, M., Qian, J., Liu, H., and Guo, C. Tiresias: A GPU cluster manager for distributed deep learning. In *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)* (Boston, MA, Feb. 2019), USENIX Association, pp. 485–500.

[3] He, B., Zheng, X., Chen, Y., Li, W., Zhou, Y., Long, X., Zhang, P., Lu, X., Jiang, L., Liu, Q., Cai, D., and Zhang, X. Dxpu: Large-scale disaggregated gpu pools in the datacenter. *ACM Transactions on Architecture and Code Optimization 20*, 4 (Dec. 2023), 1–23.

[4] Hindman, B., Konwinski, A., Zaharia, M., Ghodsi, A., Joseph, A. D., Katz, R., Shenker, S., and Stoica, I. Mesos: a platform for fine-grained resource sharing in the data center. In *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation* (USA, 2011), NSDI'11, USENIX Association, p. 295–308.

[5] Jiang, Z., Lin, H., Zhong, Y., Huang, Q., Chen, Y., Zhang, Z., Peng, Y., Li, X., Xie, C., Nong, S., Jia, Y., He, S., Chen, H., Bai, Z., Hou, Q., Yan, S., Zhou, D., Sheng, Y., Jiang, Z., Xu, H., Wei, H., Zhang, Z., Nie, P., Zou, L., Zhao, S., Xiang, L., Liu, Z., Li, Z., Jia, X., Ye, J., Jin, X., and Liu, X. MegaScale: Scaling large language model training to more than 10,000 GPUs. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)* (Santa Clara, CA, Apr. 2024), USENIX Association, pp. 745–760.

[6] Li, J., Xu, H., Zhu, Y., Liu, Z., Guo, C., and Wang, C. Lyra: Elastic scheduling for deep learning clusters. In *Proceedings of the Eighteenth European Conference on Computer Systems* (May 2023), EuroSys '23, ACM.

[7] Vavilapalli, V. K., Murthy, A. C., Douglas, C., Agarwal, S., Konar, M., Evans, R., Graves, T., Lowe, J., Shah, H., Seth, S., Saha, B., Curino, C., O'Malley, O., Radia, S., Reed, B., and Baldeschwieler, E. Apache hadoop yarn: yet another resource negotiator. In *Proceedings of the 4th Annual Symposium on Cloud Computing* (New York, NY, USA, 2013), SOCC '13, Association for Computing Machinery.

[8] Wu, Y., Ma, K., Yan, X., Liu, Z., Cai, Z., Huang, Y., Cheng, J., Yuan, H., and Yu, F. Elastic deep learning in multi-tenant gpu clusters. *IEEE Transactions on Parallel and Distributed Systems 33*, 1 (2022), 144–158.