Resource Allocation and Parallelism in Artificial Intelligence: A Survey

JASON GARDNER *Student Member, IEEE*, n01480000@unf.edu[1]

[1]School of Computing, University of North Florida, Jacksonville, FL 32259 USA

***Abstract*—Artificial Intelligence (AI) makes use of an incredible amount of computing resources and this amount has been increasing as interest grows and the technology is adopted [1]. Because of the costs incurred during training and inference, researchers continue to seek methods to optimize the use and allocation of finite resources, optimizing their use training AI models. [2]**

Our work surveyed the most promising start-of-the art research for optimizing resource utilization and accelerating AI model development. We performed a systematic review of existing literature and a comparative analysis of different methods of reducing resource utilization by AI. Key areas explored include data parallelism, model parallelism, hybrid parallelism, cloud and distributed computing, and resource allocation methodologies.

Our findings highlight the current state of research in AI resource usage and emerging trends in the field. This survey covers work in AI resource parallelism over the past decade with a particular focus between 2022 and October 2024.

***Index Terms*—Resource sharing, AI training, deep learning, multi-tenancy, workload scheduling, hardware partitioning, model parallelism, cloud computing, distributed computing.**

I. Introduction [1]

THE rapid increase in AI popularity as a search term since 2022 is a reflection of a commensurate increase in research surrounding AI [3]. A Google Scholar search for "AI" returned approximately 775,000 article results between 2022 and October 2024 [4]. The increasing popularity of AI has created an unprecedented increase in demand for computing resources. NVIDIA, a primary supplier of hardware utilized in AI training and inference has seen their stock price increase by more than 2,700% in the past five years, with a majority of that increase occurring in the past two and a half years [5]. The training of large models, driven by transformer architectures in natural language processing (NLP), convolutional neural networks (CNNs) in computer vision, and generative advesarial networks (GANs) require significant resources in the form of processing power, memory, and storage. As organizations scale up their AI capabilities they face substantial resource constraints, making efficient resource management essential for both cost-effectiveness and model performance.

Scaling AI training workloads across hardware resources has become a critical area of research, with advancements in how resources can efficiently be partitioned, scheduled, and shared across multiple users and workloads. Effective resource-sharing strategies need to emerge as demand for hardware resources and the costs of AI compute resources continue to increase.

[1]For COP6616 Parallel Computing with Scott Piersall, Fall 2024

Key Terms

- **AI training:** The computational tasks involved in training AI models.
- **Resource Sharing:** The allocation and utilization of computational resources (e.g., Central Processing Units (CPUs), Graphics Processing Units (GPUs), Tensor Processing Units (TPUs), Neural Processing Units (NPUs), storage, and memory) among multiple AI training jobs.
- **Data parallelism:** Replicating the model across multiple devices and partitioning the data among them.
- **Model parallelism:** Dividing the model into smaller submodels and assigning them to different devices.
- **Pipeline parallelism:** Breaking the training process into stages and executing them in a pipeline fashion.
- **Hybrid parallelism:** Combining multiple parallelism techniques.
- **Resource allocation algorithm:** A strategy for assigning resources to different AI training jobs or tasks.

II. Background

A. Historical Overview

Resource management has evolved alongside the development of parallel computing and distributed systems. Early work on resource sharing focused on general-purpose computing applications, while more recent research has specifically addressed the unique requirements of AI training workloads as AI models began to grow significantly in the late 2010s. Developments in GPU, TPU, and NPU hardware along with the growth of cloud computing have led to the creation of new methods of resource management aimed specifically at AI workloads.

Key Components:

- *Computational resources:* CPUs, GPUs, TPUs, NPUs, and other hardware components used for AI training, to include memory and storage.
- *AI models:* Deep neural networks, reinforcement learning models, and other AI algorithms.

- *Training datasets:* Large-scale datasets used to train AI models.
- *Resource allocation strategies:* Techniques for assigning computational resources to AI training jobs.
- *Frameworks:* Software tools that support distributed AI training (Tensorflow, Pytorch, Horovod, CUDA)
- *Cloud Computing:* Providers that offer infrastructure and services designed to handle large-scale AI training tasks (AWS, Google Cloud, Microsoft Azure, Huggingface)

B. Foundational Work

- 2004: Google's MapReduce framework laid the groundwork for large-scale data processing. [6]
- 2006: Apache Hadoop created an open-source implementation of MapReduce and a distributed filesystem (HDFS) [7]
- 2012: AlexNet demonstrated the effectiveness of deep convolutional neural networks trained on GPUs for image classification. [8]
- 2014: Parameter servers separate model parameters from computational nodes. [9]
- 2015: "Deep Learning with Elastic Averaging SGD" provides an optimization algorithm that allows greater exploration during training by averaging parameters across workers. [10]
- 2016: Federated learning enables training machine learning models across multiple decentralized devices while keeping data localized. [11]
- 2017: Uber Engineering's Hovorod provides an open-source framework for distributed deep learning using Message Passing Interface (MPI) [12]

III. Methodology

Our search methodology utilized searches of the Springer [13], IEEE [14], and ACM [15] databases. Google Scholar [16] and the University of North Florida (UNF) library [17] were also utilized as "meta" searches that queried multiple databases. The UNF library contained 302 databases at the time of writing. These databases were searched for the terms "Artificial Intelligence," **or** "AI" **and** "resource sharing" **or** "resource allocation." Because these searches returned results that included applications of artificial intelligence, the searches were further narrowed by searching for additional key terms, "data parallelism" **or** "model parallelism" **or** "pipeline parallelism." The searches were initially reduced further by excluding the application terms "wireless," "4G", "5G," and "6G" before it was decided to manually review the results to exclude applications. The papers were further restricted to papers published between 2022 and 2024 to narrow the focus current research on the topic. Forty papers were ultimately selected for further review for potential inclusion in our work and a further 8 were removed for insufficient contribution, sourcing, or for being literature reviews themselves. We ultimately chose 32 papers for review in this work.

IV. Literature Review

The chosen research explored various aspects of optimizing deep learning model training and inference, focusing on resource allocation and parallelism strategies across diverse computing environments. Several papers examine model, data, and pipeline parallelism for improved efficiency, particularly in large language model training and on resource-constrained platforms like embedded systems. Other papers address challenges in distributed and federated learning, including resource management, communication overhead, and the use of AI for resource allocation in High-Performance Computing (HPC) clusters. Finally, some sources discuss techniques for optimizing deep learning compilers for heterogeneous hardware and GPU virtualization for improved resource utilization.

A. Resource Sharing and Parallelism Techniques in AI Training

1) Resource Sharing: This category focuses on the efficient allocation and usage of computational resources across multiple AI training jobs.

- *AlpaServe: Statistical Multiplexing with Model Parallelism for Deep Learning Serving*, *DeInfer: A GPU resource allocation algorithm with spatial sharing for near-deterministic inferring tasks*, and *Enhanced Scheduling of AI Applications in Multi-Tenant Cloud Using Genetic Optimizations* emphasized the challenges and strategies for managing resources in HPC environments, particularly for AI workloads. These papers highlighted the importance of optimizing resource utilization for complex AI training jobs [18][19][20].
- *Greening AI: A Framework for Energy-Aware Resource Allocation of ML Training Jobs with Performance Guarantees* discussed the use of techniques like spatial sharing and genetic algorithms to improve GPU utilization, addressing issues like resource fragmentation and ensuring efficient allocation. It introduced the concept of "Greening AI," focusing on energy-aware resource allocation to reduce the environmental impact of AI training [21].
- *A Load Balance Scheduling Approach for Generative AI on Cloud-Native Environments with Heterogeneous Resources* focused on a load-balancing scheduling approach for Generative AI in cloud-native environments with diverse resources. It highlighted the importance of efficient resource allocation to manage the growing demands of Generative AI, especially when dealing with varying computational and memory needs across different tasks [22].
- *DeInfer: A GPU resource allocation algorithm with spatial sharing for near-deterministic inferring tasks* introduced DeInfer, a GPU resource allocation algorithm that features spatial sharing for near-deterministic inferencing tasks. DeInfer sought to address the challenge of efficiently managing and sharing GPU resources among multiple deterministic inference tasks with varying requirements [19].
- *Dynamic Resource Allocation and Energy Optimization in Cloud Data Centers Using Deep Reinforcement Learning* proposed a Deep Reinforcement Learning (DRL) framework for dynamic resource allocation and energy optimization in cloud data centers. This framework aims

to optimize the use of resources, such as VMs and physical machines, to reduce energy consumption while maintaining high performance and resource utilization [23].
- *Enhanced Scheduling of AI Applications in Multi-Tenant Cloud Using Genetic Optimizations* examined the challenge of under-utilization of GPUs in multi-tenant environments running Machine Learning workloads. It underscores the need for effective resource management strategies to maximize GPU utilization and overall system efficiency [20].
- *Load Characterization of AI Applications using DQoES Scheduler for Serving Multiple Requests* characterized the load of AI applications using the DQoES scheduler. It focuses on serving multiple AI requests effectively, considering the impact of sequential and concurrent requests on resource consumption (specifically VRAM usage and running time) when utilizing AI models with different complexities [24].

2) Data Parallelism: This category focuses on replicating the model across multiple devices and distributing data among them.,

- *GNNPipe: Scaling Deep GNN Training with Pipelined Model Parallelism* explained how tensor parallelism, a form of data parallelism, can be used to distribute large models across multiple GPUs for training [25].
- *Optimizing DNN training with pipeline model parallelism for enhanced performance in embedded systems* clarified that in non-GNN (Graph Neural Network) models, data parallelism allows GPUs to process different layers on different training samples concurrently, improving GPU utilization. It proposes a framework combining data parallelism with pipeline model parallelism for efficient training in embedded systems, particularly for convolutional layers in CNNs [26].

3) Model Parallelism: This technique involves splitting the model into smaller sub-models and assigning them to different devices.

- *AlpaServe: Statistical Multiplexing with Model Parallelism for Deep Learning Serving* explained that model parallelism is crucial when training Large Language Models (LLMs) with limited GPU memory, enabling the model to be distributed across multiple devices [18].
- *GNNPipe: Scaling Deep GNN Training with Pipelined Model Parallelism* introduced the concept of using model parallelism not just for scaling single large models but also for statistical multiplexing in multi-model serving, highlighting a novel use case [25].
- *Group-Based Interleaved Pipeline Parallelism for Large-Scale DNN Training* described a hybrid approach combining layer-level model parallelism with graph parallelism for training GNNs, addressing challenges related to large graph sizes and limited GPU memory [27].
- *Optimizing DNN training with pipeline model parallelism for enhanced performance in embedded systems* contrasted model parallelism with pipeline parallelism, noting that conventional model parallelism can suffer from low resource utilization or high communication overhead. It uses model parallelism for fully connected layers in their proposed framework for Deep Neural Network (DNN) training on embedded systems [26].
- *GEMS: GPU-enabled Memory-Aware Model-Parallelism System for Distributed DNN Training* introduced GEMS, a GPU-enabled memory-aware model-parallelism system designed for distributed DNN training. GEMS aims to efficiently train large DNN models, particularly for high-resolution image analysis, by dividing the model into smaller parts and distributing them across multiple GPUs while managing memory constraints [28].
- *MixPipe: Efficient Bidirectional Pipeline Parallelism for Training Large-Scale Models* introduced MixPipe, a system for efficient bidirectional pipeline parallelism for training large-scale deep learning models. MixPipe aims to optimize memory usage and communication patterns when training models utilizing pipeline parallelism [29].
- *Optimizing DNN training with pipeline model parallelism for enhanced performance in embedded systems* explored optimizing DNN training using pipeline model parallelism to enhance performance in embedded systems. It emphasized the benefits of dividing the model and training process into stages to improve training speed and efficiency in resource-constrained environments [26].
- *PipeEdge: Pipeline Parallelism for Large-Scale Model Inference on Heterogeneous Edge Devices* introduced PipeEdge, a framework that leverages pipeline parallelism for large-scale model inference on heterogeneous edge devices. PipeEdge partitions and distributes the model across multiple devices to improve inference speed, especially on devices with limited resources [30].
- *Pipeline Parallelism With Elastic Averaging* explored pipeline parallelism with elastic averaging as a technique to distribute the training of deep learning models across multiple devices. It examined the performance and computational efficiency of this method compared to traditional training approaches [31].

4) Pipeline Parallelism: This technique divides the training process into stages, executing them in a pipeline across multiple devices.

- *GraphPipe: Improving Performance and Scalability of DNN Training with Graph Pipeline Parallelism* focused on the challenges and strategies for efficient pipeline parallelism in DNN training, highlighting issues like pipeline bubbles and memory footprint, and proposing solutions like sequence-level scheduling and wave-like pipeline strategies [32].
- *Optimizing DNN training with pipeline model parallelism for enhanced performance in embedded systems* introduced "graph pipeline parallelism," which considers the DNN topology for partitioning, unlike traditional pipeline parallelism that treats stages sequentially. The authors apply pipeline parallelism in their framework, particularly for the fully connected layers, to reduce the overall training time in embedded systems [26].
- *Mixtran: An Efficient and Fair Scheduler for Mixed Deep*

Learning Workloads in Heterogeneous GPU Environments proposed Mixtran as an efficient and fair scheduler for mixed deep learning workloads in heterogeneous GPU environments. Mixtran aims to optimize the scheduling of different types of deep learning jobs, considering both efficiency and fairness in resource allocation [33].

5) Hybrid Parallelism: This category involves combining multiple parallelism techniques.

- *GNNPipe: Scaling Deep GNN Training with Pipelined Model Parallelism* and *Hanayo: HarnessingWave-like Pipeline Parallelism for Enhanced Large Model Training Efficiency* explained that the choice between different parallelism techniques depends on the specific scenario and factors like tensor parallelism degree, number of GPUs, and network bandwidth [25][34].
- *GraphPipe: Improving Performance and Scalability of DNN Training with Graph Pipeline Parallelism* described combining layer-level model parallelism with graph parallelism for GNN training and integrating tensor and pipeline parallelism for large model training, respectively. It highlighted hybrid parallelism as a means to address the limitations of individual parallelism approaches and optimize for factors like memory efficiency, communication overhead, and resource utilization [32].

6) Resource Allocation Algorithms:

- *Advanced Resource Allocation in the Context of Heterogeneous Workflows Management* proposed a load balance scheduling approach for Generative AI on cloud environments [35].
- *Criticality-Based Data Segmentation and Resource Allocation in Machine Inference Pipelines* focued on advanced resource allocation techniques in heterogeneous workflow management for HPC systems [36].
- *DeInfer: A GPU resource allocation algorithm with spatial sharing for near-deterministic inferring tasks* discussed resource allocation in machine learning, particularly for edge computing platforms, considering demands for various resources like storage, communication, and computation and introduced DeInfer, a GPU resource allocation algorithm with spatial sharing for deterministic inferring tasks [19].
- *Enhanced Scheduling of AI Applications in Multi-Tenant Cloud Using Genetic Optimizations* reviewed various resource allocation algorithms that consider factors like model co-location, interference, and hardware parameters [20].
- *Greening AI: A Framework for Energy-Aware Resource Allocation of ML Training Jobs with Performance Guarantees* utilized genetic optimization techniques to enhance GPU utilization in multi-tenant cloud environments for AI workloads, demonstrating the effectiveness of their approach compared to traditional methods like Round-Robin scheduling [21].
- *Multi-Agent Deep Reinforcement Learning-Based Resource Allocation in HPC/AI Converged Cluster* described energy-aware multi-cluster scheduling policies and deep reinforcement learning models for resource planning in HPC, optimizing for factors like performance, energy consumption, and response time. It introduced multi-agent deep reinforcement learning for resource allocation in HPC/AI converged clusters, focusing on improving system utilization and reducing power consumption [37].
- *Optimizing DNN training with pipeline model parallelism for enhanced performance in embedded systems* summarized various resource allocation strategies, including traditional scheduling policies, deep reinforcement learning, and container-based scheduling, highlighting the evolution of approaches in HPC/AI converged systems. It proposed a DNN model parallelism framework for embedded systems that includes a resource allocation algorithm for determining optimal model partitioning and resource provisions [26].
- *Criticality-Based Data Segmentation and Resource Allocation in Machine Inference Pipelines* presented DeepSense, a framework designed for data segmentation and resource allocation in machine inference pipelines. The framework is tailored for mobile applications that process sensor data. It employs deep learning techniques (convolutional and recurrent layers) for data analysis and incorporates task-specific customization [36].
- *Developing Real-Time GPU-Sharing Platforms for Artificial-Intelligence Applications* focused on developing real-time GPU-sharing platforms for artificial intelligence applications. It highlighted the importance of accurately modeling GPU behavior to enable efficient sharing of GPU resources among multiple real-time applications [38].
- *Dynamic Resource Allocation for AI/ML Applications in Edge Computing: Framework Architecture and Optimization Methods* presented an architectural framework and optimization methods for dynamic resource allocation in AI/ML applications deployed in edge computing environments. The framework aims to address the challenges of limited computational resources in edge computing by effectively managing resource allocation for AI/ML workloads [39].
- *Efficient Task Scheduling and Resource Allocation for AI Training Services in Native AI Wireless Networks* proposed a framework for efficient task scheduling and resource allocation for AI training services in native AI wireless networks. The framework aims to optimize the allocation of network and computational resources (including access point selection and bandwidth allocation) to enhance the training process of AI models [40].
- *Multi-Agent Deep Reinforcement Learning-Based Resource Allocation in HPC/AI Converged Cluster* presented a Multi-Agent Deep Reinforcement Learning (MADRL) system for resource allocation in high-performance computing (HPC) clusters that handle both traditional HPC workloads and AI workloads. The system aims to optimize job scheduling and resource utilization to meet the diverse demands of different job types [37]
- *NEST-C: A deep learning compiler framework for heterogeneous computing systems with artificial intelligence*

accelerators presented NEST-C, a deep learning compiler framework designed for heterogeneous computing systems that include artificial intelligence accelerators. NEST-C focuses on optimizing the partitioning and scheduling of deep learning workloads across different types of processing units to enhance overall execution speed [41].

- *Resource Orchestration and Scheduling Algorithms for Enhancing Distributed Reinforcement Learning* introduced resource orchestration and scheduling algorithms to enhance distributed reinforcement learning. It focuses on improving the efficiency of resource allocation and synchronization in distributed reinforcement learning, where tasks are distributed across multiple learners [42].
- *Towards Efficient Resource Allocation for Federated Learning in Virtualized Managed Environments* explored efficient resource allocation strategies for Federated Learning deployments in virtualized managed environments. The focus was on optimizing the allocation of resources, such as CPU cores, memory, and network bandwidth, to improve the performance of federated learning tasks [43].

The sources generally agree on the importance of efficient resource allocation strategies for AI/ML applications, particularly in cloud, edge, and embedded environments. They all recognize that effective resource management is crucial for maximizing performance, minimizing costs, and ensuring fairness among users. The sources propose various techniques, algorithms, and frameworks to address these challenges. However, there are also some disagreements and areas where further research is needed.

- The Need for Accurate Modeling and Characterization of Workloads: The sources agree on the significance of accurately modeling the behavior of AI/ML workloads and the underlying hardware platforms to develop effective resource allocation strategies. For instance, *Developing Real-Time GPU-Sharing Platforms for Artificial-Intelligence Applications* emphasizes understanding GPU behavior for real-time systems, advocates for evaluating management techniques using complex applications, and highlights the importance of considering the unique characteristics of workloads like VRAM usage and running time. However, there's a lack of consensus on standardized methodologies and metrics for workload characterization, especially for emerging AI techniques like Generative AI [22]. This discrepancy suggests further research is necessary to establish robust evaluation methods for these new workloads.
- Dynamic Resource Allocation and Scheduling: The sources show a clear preference for dynamic resource allocation techniques over static approaches. Dynamic strategies allow for adaptive adjustment of resources based on real-time workload demands, leading to better utilization and efficiency. The sources propose various approaches, including Deep Reinforcement Learning (DRL) [23], which learns optimal allocation policies, and heuristic-based methods [22][35], which use predefined rules for decision-making. While dynamic allocation is widely supported, the optimal choice between specific methods, such as DRL or heuristics, remains an open question, potentially influenced by factors like system complexity and real-time constraints.
- Prioritization of Specific Objectives: While efficiency and fairness are generally acknowledged, the sources prioritize different objectives based on the specific context. For example, *A Load Balance Scheduling Approach for Generative AI on Cloud-Native Environments with Heterogeneous Resources* emphasizes minimizing task completion times for Generative AI, *DeInfer: A GPU resource allocation algorithm with spatial sharing for near-deterministic inferring tasks* focuses on achieving low violation rates in inference tasks, and *Mixtran: An Efficient and Fair Scheduler for Mixed Deep Learning Workloads in Heterogeneous GPU Environments* aims to maximize system resource utilization while maintaining fairness among users. This difference in priorities highlights the need for flexible resource allocation frameworks that can be tailored to the specific requirements of different applications and deployment scenarios.
- Focus on Specific Computing Environments: Some sources concentrate on particular computing environments, such as cloud data centers [23], edge devices [30], or embedded systems [26]. This specialization leads to tailored solutions and optimizations for each environment. However, more research is needed on unifying resource allocation strategies across diverse computing paradigms, like the cloud-edge continuum [21]. The integration of heterogeneous computing resources poses additional challenges for resource management and requires further investigation.
- Limited Discussion on Security and Privacy: While several sources emphasize security and privacy in the context of specific applications, such as federated learning [11] or data management [44], there is a general lack of in-depth discussion on security and privacy implications of resource allocation strategies themselves. For example, the sources do not explicitly address potential vulnerabilities arising from dynamic resource adjustments or the need for secure and privacy-preserving resource allocation mechanisms in multi-tenant environments. This oversight suggests a crucial area for future research, especially as AI/ML applications become more prevalent in sensitive domains like healthcare.

In conclusion, the sources present a diverse range of approaches to resource allocation for AI/ML applications, reflecting the evolving nature of this field. While they agree on the fundamental importance of efficient and fair resource management, they also highlight the need for further research to address challenges related to workload characterization, algorithm selection, security, and privacy. Developing comprehensive and adaptable resource allocation frameworks that cater to the unique needs of various AI/ML applications and computing environments remains a key area for future exploration.

B. Literature

1) A Load Balance Scheduling Approach for Generative AI on Cloud-Native Environments with Heterogeneous Resources [22]: Proposed a new framework for deploying generative AI applications in cloud environments with varying computational and memory needs. The study focuses on addressing the challenges of running computationally intensive generative AI tasks, such as stable diffusion, on cloud platforms with heterogeneous resources. A two-step method is proposed:

1) Profiling to analyze resource allocation relationships.
2) Developing a load balance scheduling approach using a knapsack problem approach to optimize GPU load balancing.

2) Advanced Resource Allocation in the Context of Heterogeneous Workflows Management [35]: This paper highlighted the growing complexity of High-Performance Computing (HPC) workflows, which now encompass a diverse range of tasks, including numerical simulations, AI model training and inference, and large-scale data analytics. These heterogeneous workflows often involve intricate dependencies, automated job submissions, and substantial I/O operations, posing challenges for efficient resource allocation. While existing batch schedulers are adept at maximizing resource utilization, they often fall short in optimizing the execution of these complex workflows due to their focus on global resource allocation rather than individual job or workflow completion times. To address these limitations, the paper proposed WARP (Workflow-aware Advanced Resource Planner), a tool designed to integrate with workflow management tools and batch schedulers. WARP enables the reservation of resources in advance based on factors like job duration, dependencies, and machine load. Its primary goal is to minimize the overall workflow execution time without disrupting the priority policies established for cluster users by system administrators.

3) AlpaServe: Statistical Multiplexing with Model Parallelism for Deep Learning Serving [18]: Introduced AlpaServe, a system for predicting servings of multiple large deep learning models using statistical multiplexing and integrating model parallelism. The paper challenged the conventional view that model parallelism is only suitable for very large models and demonstrated its effectiveness in improving resource utilization and reducing tail latency in multi-model serving scenarios.

4) BPIPE*: Memory-Balanced Pipeline Parallelism for Training Large Language Models [45]:* Pipeline parallelism has become an essential technique for training LLMs on GPU clusters. However, it frequently leads to memory imbalance, where some GPUs experience significant memory pressure while others have underutilized capacity. This imbalance negatively impacts training performance, even when the total GPU memory is sufficient for more efficient configurations. BPIPE, a novel approach designed to address this problem, aims to achieve memory balance in pipeline parallelism. It employs an activation balancing method to transfer intermediate activations between GPUs during training, thus ensuring all GPUs utilize comparable amounts of memory. By balancing memory utilization, BPIPE enhances the efficiency of LLM training by reducing redundant computations or enabling larger micro-batch sizes. Evaluation results, using GPT-3 models trained on 48 A100 GPUs, demonstrate that BPIPE can achieve 1.25x to 2.17x speedups compared to Megatron-LM, a leading LLM training framework.

5) Criticality-Based Data Segmentation and Resource Allocation in Machine Inference Pipelines [36]: Focused on resource allocation in machine learning inference pipelines. It introduced a data segmentation approach based on criticality levels to optimize resource allocation for inference tasks. It highlighted the importance of self-supervised learning in addressing the challenge of limited training labels in Internet of Things (IoT) applications and presented customized self-supervised learning methods for this domain.

6) DeInfer: A GPU resource allocation algorithm with spatial sharing for near-deterministic inferring tasks [19]: Introduceed DeInfer, a GPU resource allocation algorithm designed for near-deterministic deep learning inferring tasks (DLI). The paper highlighted the challenges of achieving predictable latency in DLI and proposes DeInfer to address these issues. The key innovation lies in exploiting spatial sharing within the GPU to ensure that multiple DLI tasks can meet their latency Service Level Agreements (SLAs).

7) Developing Real-Time GPU-Sharing Platforms for Artificial-Intelligence Applications [38]: Discussed the development of real-time GPU-sharing platforms for AI applications. It analyzeed the challenges of managing GPU resources for predictable timing in real-time systems and provides insights into various real-time GPU management tactics. The source highlighted the importance of accurate GPU behavior modeling for predictable performance and explored various techniques for evaluating the performance of real-time GPU sharing systems.

8) Dynamic Resource Allocation and Energy Optimization in Cloud Data Centers Using Deep Reinforcement Learning [23]: Proposed a deep reinforcement learning (DRL) based framework for dynamic resource allocation and energy optimization in cloud data centers. The framework uses a deep neural network to learn optimal resource allocation policies that can adapt to changing workloads and minimize energy consumption while maintaining high performance.

9) Dynamic Resource Allocation for AI-ML Applications in Edge Computing: Framework Architecture and Optimization Methods [39]: Introduced an architectural framework and optimization strategies for dynamic resource allocation for AI/ML applications in edge computing. It highlighted the challenges of resource allocation in edge computing, such as limited resources and diverse application requirements, and discussed various optimization techniques, including heuristic algorithms, machine learning driven approaches, and game theory.

10) Dynamic Scaling Strategies for AI Workloads in Cloud Environments [44]: Explored and evaluated various dynamic scaling strategies for AI workloads in cloud environments. It examined the strengths, limitations, and potential areas for improvement of different scaling approaches and identified automatic adjustment of resources in response to workload variations as a key solution for addressing the challenges posed by AI workloads.

11) Efficient Task Scheduling and Resource Allocation for AI Training Services in Native AI Wireless Networks [40]: Proposed a task scheduling and resource allocation scheme for AI training services in native AI wireless networks. The paper presented an improved Non-dominated Sorting Genetic Algorithm II with a task scheduling strategy to enhance resource utilization and reduce communication overhead in these networks. The proposed algorithm aims to ensure the efficient execution of training tasks with varying sizes, computational demands, and quality of service requirements while optimizing resource allocation and network bandwidth utilization.

12) Enhanced Scheduling of AI Applications in Multi-Tenant Cloud Using Genetic Optimizations [20]: Exploreed bin-packing models tailored for GPU scheduling to address the challenges of low GPU utilization in multi-tenant cloud environments running AI applications. The paper investigated two bin-packing approaches:

- Consolidating multiple GPUs into a single machine.
- Treating each GPU as a separate resource dimension.

13) Evolution of GPU Virtualization to Resource Pooling [46]: The evolution of GPU virtualization to resource pooling is driven by the need to optimize performance and utilization of GPU resources, particularly for computationally demanding AI applications. Different stages of GPU virtualization, from initial device emulation to GPU pass-through, API redirection, and full virtualization, offer varying levels of resource sharing and isolation, with trade-offs between flexibility and performance. GPU resource pooling represents the culmination of this evolution, aggregating multiple GPUs into a shared resource pool, enabling dynamic allocation and release based on workload demands. This approach offers several benefits, including improved GPU utilization, enhanced flexibility for deploying diverse applications, and reduced infrastructure costs. Current resource pooling implementations leverage intelligent scheduling algorithms and dynamic migration technologies to allocate and schedule GPU resources efficiently, further boosting the performance of GPU-intensive applications.

14) GEMS: GPU-enabled Memory-Aware Model-Parallelism System for Distributed DNN Training [28]: Introduced GEMS, a GPU-enabled memory-aware model-parallelism system for distributed DNN training. The paper focused on addressing the challenges of high-resolution image processing in DNN training, particularly in the context of digital pathology. GEMS incorporates memory-aware techniques to optimize memory utilization and enhance the scalability of model parallelism.

15) GNNPipe: Scaling Deep GNN Training with Pipelined Model Parallelism [25]: Communication overhead is a major bottleneck in distributed graph neural network (GNN) training, especially for large graphs and deep GNN models. GNNPipe offers a novel approach to tackle this challenge by being the first to employ layer-level model parallelism for GNN training. It partitions the GNN layers across multiple GPUs, with each GPU handling computations for a subset of consecutive layers on the entire graph. This strategy significantly reduces communication volume compared to traditional graph parallelism, which partitions the graph itself. GNNPipe overcomes the unique challenges of pipelined layer-level model parallelism on the entire graph by employing techniques like graph chunking to break it into dependent chunks, utilizing historical vertex embeddings, and incorporating specific training techniques to guarantee convergence. It also provides a hybrid approach that combines GNNPipe with graph parallelism to handle extremely large graphs, thereby achieving better resource utilization and ensuring convergence. The system supports all three parallelism settings and delivers substantial performance gains over graph parallelism, achieving up to 2.45x speedup and reducing communication volume by up to 22.89x while maintaining comparable model accuracy and convergence.

16) GRAPHPIPE: Improving Performance and Scalability of DNN Training with Graph Pipeline Parallelism [32]: Explored graph pipeline parallelism for DNN training, aiming to enhance the performance and scalability of training large-scale models. The authors discuss the limitations of existing pipeline parallelism approaches and propose GraphPipe, a novel technique that exploits graph-based model partitioning and optimized communication strategies for efficient distributed training.

17) Greening AI: A Framework for Energy-Aware Resource Allocation of ML Training Jobs with Performance Guarantees [21]: Proposed a framework for energy-aware resource allocation of ML training jobs with performance guarantees. The paper focused on reducing the energy consumption of ML training while ensuring timely completion of training tasks. The framework considers various factors, such as energy efficiency of different hardware resources, workload characteristics, and performance requirements, to make informed resource allocation decisions.

18) Group-based Interleaved Pipeline Parallelism for Large-scale DNN Training [27]: Investigated group-based interleaved pipeline parallelism for large-scale DNN training. It addressed the limitations of conventional pipeline parallelism approaches and proposes a new technique that leverages interleaved scheduling of micro-batches to enhance training efficiency. This approach aims to reduce the pipeline bubble problem and improve GPU utilization.

19) Hanayo: Harnessing Wave-like Pipeline Parallelism for Enhanced Large Model Training Efficiency [34]: Presented Hanayo, a wave-like pipeline parallelism scheme for efficient large model training. The paper introduced a novel approach that partitions the model into multiple waves, allowing for concurrent execution of micro-batches within each wave. This strategy aims to achieve low bubble ratios and high performance by reducing the impact of pipeline bubbles.

20) Load Characterization of AI Applications using DQoES Scheduler for Serving Multiple Requests [24]: Analyzed AI application load characteristics using the DQoES scheduler for serving multiple requests. It focused on understanding the behavior of AI models, specifically super-resolution generative adversarial networks (SRGANs), when handling sequential and concurrent requests. The paper investigated the impact of different request types on resource utilization, particularly VRAM usage, and proposed scheduling strategies to optimize performance in multi-request scenarios.

21) MixPipe: Efficient Bidirectional Pipeline Parallelism for Training Large-Scale Models [29]: Proposed MixPipe, a bidirectional pipeline parallelism technique for training large-scale models. It aims to enhance training efficiency by utilizing both forward and backward passes in the pipeline. The paper introduced a flexible micro-batch scheduling scheme to balance computation and communication overhead and achieve high GPU utilization.

22) Mixtran: An Efficient and Fair Scheduler for Mixed Deep Learning Workloads in Heterogeneous GPU Environments [33]: Training deep learning (DL) models often requires a cluster of GPUs to handle the increasing complexity and scale of data. Efficiency and fairness are two major concerns in managing mixed DL workloads in heterogeneous GPU clusters. Mixtran, an efficient and fair scheduler, was designed to tackle these challenges. This scheduler abstracts heterogeneous GPU resources as virtual tickets and distributes them fairly to users, ensuring that each user receives a fair share of resources regardless of the underlying hardware capabilities. Mixtran incorporates a global optimization model that considers various factors, including quantified resource requests, node constraints, and fairness constraints, to optimize the utilization of GPUs efficiently. Moreover, it features a resource trading mechanism based on the second-price auction model, allowing users to trade resources for mutual benefit, further enhancing both resource utilization and user satisfaction.

23) Multi-Agent Deep Reinforcement Learning-Based Resource Allocation in HPC/AI Converged Cluster [37]: Presented a multi-agent deep reinforcement learning (mDRL) based resource allocation scheme for deep learning (DL) jobs in high-performance computing (HPC) and AI converged clusters. The paper focused on improving system efficiency by dynamically allocating resources to DL jobs based on their characteristics and requirements using a multi-agent approach.

24) Multiple-Deep Neural Network Accelerators for Next-Generation Artificial Intelligence Systems [47]: This paper investigated the scheduling challenges faced when managing a diverse set of machine learning workloads in large-scale, multi-tenant cloud environments that utilize heterogeneous GPUs. As the integration of AI expands across various industries, from smart logistics and FinTech to entertainment and cloud computing, the demand for efficient scheduling becomes increasingly critical. The paper pointed out that conventional scheduling methods struggle to achieve satisfactory results in these complex environments, often leading to low GPU utilization and resource fragmentation. To address this issue, the paper proposed a novel scheduling approach based on genetic optimization techniques. This approach, implemented within a process-oriented discrete-event simulation framework, aims to effectively orchestrate the execution of diverse machine learning tasks and improve GPU utilization. Evaluation results, using workload traces from Alibaba's MLaaS cluster, demonstrate that the proposed scheduling approach can enhance GPU utilization by 12.8% compared to the commonly used Round-Robin scheduling algorithm without compromising performance, showcasing its effectiveness in optimizing cloud-based GPU scheduling for heterogeneous workloads.

25) NEST-C: A deep learning compiler framework for heterogeneous computing systems with artificial intelligence accelerators [41]: This work suggested adopting smarter strategies for managing GPU resources to address the increasing need for better planning methods for AI applications on the cloud. Existing methods for managing GPUs often result in underutilization. The goal is to develop more efficient methods to allocate and schedule these resources to specific tasks. One approach is to use virtualization technology to pool GPU resources. By implementing resource pooling, AI and machine learning tasks can dynamically request and release GPU resources as needed. This improves the overall performance of the workload by reducing the time it takes to complete. It also allows for better sharing of GPU resources between different applications and users. The paper proposes using a knapsack algorithm with a focus on load balancing. This approach has been shown to improve the overall performance of GPU-intensive tasks by 64.1% compared to using the knapsack algorithm without load balancing. This technique reduces the overall standard deviation in load by 40.8% compared to other methods, ensuring tasks are assigned to GPUs to make better use of resources. Another approach is to transfer activations between GPUs to solve the memory imbalance problem. For example, by intelligently allocating resources across a pipeline of stages, memory usage can be optimized. These findings highlight the significance of improving how tasks are mapped to GPUs to make better use of resources.

26) Optimizing DNN training with pipeline model parallelism for enhanced performance in embedded systems [26]: Proposed a pipeline-based model parallelism framework for optimizing DNN training in embedded systems. The framework uses an approach to find the optimal number of splits for a DNN model, ensuring efficient distribution across multiple processors. It then applies pipeline parallelism to these partitioned sub-modules to accelerate execution.

27) PipeEdge - Pipeline Parallelism for Large-Scale Model Inference on Heterogeneous Edge Devices [30]: Introduced PipeEdge, a distributed pipeline parallelism framework for large-scale model inference on heterogeneous edge devices. It aims to accelerate inference by partitioning large models and executing them across multiple devices with varying capabilities. The framework addresses the challenges of limited resources and diverse hardware configurations in edge environments and proposes a dynamic partitioning strategy to optimize performance.

28) Pipeline Parallelism With Elastic Averaging [31]: Introduced a novel pipeline parallelism technique called Elastic Averaging (EA-Pipe) for training large-scale models. It focused on addressing the weight inconsistency and delayed gradient problems in traditional pipeline parallelism approaches. EA-Pipe proposes a novel weight update scheme that mitigates these issues by elastically averaging model parameters during training.

29) Resource Orchestration and Scheduling Algorithms for Enhancing Distributed Reinforcement Learning [42]: Proposed resource orchestration and scheduling algorithms for distributed reinforcement learning (DRL). The paper introduced two algorithms:

- LeaderFirst: Prioritizes resource allocation for specific learners to enhance their learning capabilities.
- Batch Tasks Centralized: Aims to optimize resource utilization by centralizing the management of batch tasks.

30) Seq1F1B: Efficient Sequence-Level Pipeline Parallelism for Large Language Model Training [48]: Presented Seq1F1B, an efficient sequence-level pipeline parallelism approach for training large language models. It addresses the challenges of training these models on multi-node systems and proposes a novel pipeline partitioning strategy to improve performance.

31) Towards accelerating model parallelism in distributed deep learning systems [49]: Focused on accelerating model parallelism in distributed deep learning systems. It proposed a micro-batch size search algorithm to optimize pipeline parallelism in multi-GPU environments. The paper aimed to find an optimal balance between throughput and overhead in model parallel training by determining the most efficient micro-batch size.

32) Towards Efficient Resource Allocation for Federated Learning in Virtualized Managed Environments [43]: Discussed the challenges and opportunities associated with resource allocation for Federated Learning (FL) in virtualized managed environments. The paper highlighted the importance of efficient resource management to ensure the performance and scalability of FL deployments. It presented an overview of the testing infrastructure, tools, and experiment methodology used to evaluate the performance of different resource allocation strategies for FL workflows.

V. Contributions

Our work contributes to the field by aggregating state-of-the-art work in resource allocation and parallelism in AI and presenting it in a single location. This allows future research to utilize our work and work like ours to get an idea of what the current state of the field is, to use existing state-of-the-art methods, or to pursue their own research. Additionally, the research are sorted, so specific sub-topics can be evaluated in a specific area of research. Existing surveys focused on areas like data parallelism and excluded all other parallelism, so our paper should provide additional benefit to researchers that is not present in other work.

VI. Conclusions

Our literature review left us with a much broader understanding of resource management and parallelism in AI. An understanding of key terminology was crucial to effectively search the existing work on the topic. Because this is such an active area of research, it's possible that with the scope chosen for this work that it might not provide a comprehensive picture of the state-of-the-art research into AI resource management and parallelism. While it's possible to narrow the focus to a very specific methodology, we felt it was useful to address all methods, to provide a more comprehensive picture of the state of the field alongside surveying the research and be able to show how the methods interact and present trade-offs.

References

[1] McKinsey. (2024) *The state of AI in early 2024 — McKinsey* [Online]. Available: https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai#/. [Accessed: 6 December 2024].

[2] B. Cottier, "Trends in the Dollar Training Cost of Machine Learning Systems — epochai.org," https://epochai.org/blog/trends-in-the-dollar-training-cost-of-machine-learning-systems, 2024, [Accessed 01-10-2024].

[3] Google, "AI - Explore - Google Trends," https://trends.google.com/trends/explore?date=all&q=AI, 2024, [Accessed 01-10-2024].

[4] ——, ""AI" - Google Scholar," https://scholar.google.com/scholar?as_ylo=2022&q=%22AI%22, 2024, [Accessed 10-01-2024].

[5] ——, ""NVDA NVIDIA Corp Google Finance," https://www.google.com/finance/quote/NVDA:NASDAQ?sa=X&window=5Y, 2024, [Accessed 10-01-2024].

[6] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," in OSDI'04: Sixth Symposium on Operating System Design and Implementation, San Francisco, CA, 2004. doi: 10.1145/1327452.1327492 pp. 137–150. [Online]. Available: https://doi.org/10.1145/1327452.1327492

[7] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The hadoop distributed file system," in 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), 2010. doi: 10.1109/MSST.2010.5496972 pp. 1–10. [Online]. Available: https://doi.org/10.1109/MSST.2010.5496972

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in Neural Information Processing Systems 25 (NIPS), vol. 60, no. 6. New York, NY, USA: Association for Computing Machinery, May 2017. doi: 10.1145/3065386. ISSN 0001-0782 p. 84–90. [Online]. Available: https://doi.org/10.1145/3065386

[9] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su, "Scaling distributed machine learning with the parameter server," in Proceedings of the 11th USENIX Conference on Operating Systems Design and Implementation, ser. OSDI'14. USA: USENIX Association, 2014. doi: 10.1145/2640087.2644155. ISBN 9781931971164 p. 583–598. [Online]. Available: https://doi.org/10.1145/2640087.2644155

[10] S. Zhang, A. Choromanska, and Y. LeCun, "Deep learning with elastic averaging sgd," 2015. [Online]. Available: https://doi.org/10.48550/arXiv.1412.6651

[11] H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Federated learning of deep networks using model averaging," Computing Research Repository (CoRR), vol. abs/1602.05629, 2016. doi: 10.48550/arXiv.1602.05629. [Online]. Available: https://doi.org/10.48550/arXiv.1602.05629

[12] A. Sergeev and M. D. Balso, "Horovod: fast and easy distributed deep learning in tensorflow," 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1802.05799

[13] Springer. (2024) *Home — Springer Link* [Online]. Available: https://link.springer.com/. [Accessed: 6 December 2024].

[14] IEEE. (2024) *IEEE Xplore* [Online]. Available: https://ieeexplore.ieee.org/Xplore/home.jsp. [Accessed: 6 December 2024].

[15] ACM. (2024) *ACM Digital Library* [Online]. Available: https://dl.acm.org/. [Accessed: 6 December 2024].

[16] Google. (2024) *Google Scholar* [Online]. Available: https://scholar.google.com/. [Accessed: 6 December 2024].

[17] University of North Florida. (2024) *Database List* [Online]. Available: https://libguides.unf.edu/az/databases. [Accessed: 6 December 2024].

[18] Z. Li, L. Zheng, Y. Zhong, V. Liu, Y. Sheng, X. Jin, Y. Huang, Z. Chen, H. Zhang, J. E. Gonzalez, and I. Stoica, "AlpaServe: Statistical multiplexing with model parallelism for deep learning serving," in 17th USENIX Symposium on Operating Systems Design and Implementation (OSDI 23). Boston, MA: USENIX Association, Jul. 2023. doi: arXiv:2302.11665. ISBN 978-1-939133-34-2 pp. 663–679. [Online]. Available: https://www.usenix.org/conference/osdi23/presentation/li-zhouhan

[19] Y. Chen, W. Li, H. Zhou, X. Yang, and Y. Yin, "Deinfer: A gpu resource allocation algorithm with spatial sharing for near-deterministic inferring tasks," in Proceedings of the 53rd International Conference on Parallel Processing, ser. ICPP '24. New York, NY, USA: Association for Computing Machinery, 2024. doi: 10.1145/3673038.3673091. ISBN 9798400717932 p. 701–711. [Online]. Available: https://doi.org/10.1145/3673038.3673091

[20] S. Kwon and H. Bahn, "Enhanced scheduling of ai applications in multi-tenant cloud using genetic optimizations," Applied Sciences, vol. 14, no. 11, 2024. doi: 10.3390/app14114697. [Online]. Available: https://doi.org/10.3390/app14114697

[21] R. Sala, F. Filippini, D. Ardagna, D. Lezzi, F. Lordan, and P. Thiem, "Greening ai: A framework for energy-aware resource allocation of ml training jobs with performance guarantees," in Advanced Information Networking and Applications, L. Barolli, Ed. Cham: Springer Nature Switzerland, 2024. doi: 10.1007/978-3-031-57931-8_11. ISBN 978-3-031-57931-8 pp. 110–121. [Online]. Available: https://doi.org/10.1007/978-3-031-57931-8_11

[22] C.-K. Chun and K.-C. Lai, "A load balance scheduling approach for generative ai on cloud-native environments with heterogeneous resources," in 2024 10th International Conference on Applied System Innovation (ICASI), 2024. doi: 10.1109/ICASI60819.2024.10547947 pp. 223–225. [Online]. Available: https://doi.org/10.1109/ICASI60819.2024.10547947

[23] H. Li, G. Wang, L. Li, and J. Wang, "Dynamic resource allocation and energy optimization in cloud data centers using deep reinforcement learning," Journal of Artificial Intelligence General science (JAIGS), vol. 1, no. 1, p. 230–258, Jan. 2024. doi: 10.60087/jaigs.v1i1.243. [Online]. Available: https://doi.org/10.60087/jaigs.v1i1.243

[24] T. O. Dwi Putra, R. M. Ijtihadie, and T. Ahmad, "Load characterization of ai applications using dqoes scheduler for serving multiple requests," in 2024 12th International Symposium on Digital Forensics and Security (ISDFS), April 2024. doi: 10.1109/ISDFS60797.2024.10527227. ISSN 2768-1831 pp. 01–06. [Online]. Available: https://doi.org/10.1109/ISDFS60797.2024.10527227

[25] J. Chen, Z. Chen, and X. Qian, "Gnnpipe: Scaling deep gnn training with pipelined model parallelism," 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2308.10087

[26] M. A. Maruf, A. Azim, N. Auluck, and M. Sahi, "Optimizing dnn training with pipeline model parallelism for enhanced performance in embedded systems," Journal of Parallel and Distributed Computing, vol. 190, p. 104890, 2024. doi: 10.1016/j.jpdc.2024.104890. [Online]. Available: https://doi.org/10.1016/j.jpdc.2024.104890

[27] P. Yang, X. Zhang, W. Zhang, M. Yang, and H. Wei, "Group-based interleaved pipeline parallelism for large-scale DNN training," in International Conference on Learning Representations (ICLR), 2022. [Online]. Available: https://openreview.net/forum?id=cw-EmNq5zfD

[28] A. Jain, A. A. Awan, A. M. Aljuhani, J. M. Hashmi, Q. G. Anthony, H. Subramoni, D. K. Panda, R. Machiraju, and A. Parwani, "GEMS: Gpu-enabled Memory-aware Model-parallelism System for distributed dnn training," in SC20: International Conference for High Performance Computing, Networking, Storage and Analysis, Nov 2020. doi: 10.1109/SC41405.2020.00049 pp. 1–15. [Online]. Available: https://doi.org/10.1109/SC41405.2020.00049

[29] W. Zhang, B. Zhou, X. Tang, Z. Wang, and S. Hu, "Mixpipe: Efficient bidirectional pipeline parallelism for training large-scale models," in 2023 60th ACM/IEEE Design Automation Conference (DAC), July 2023. doi: 10.1109/DAC56929.2023.10247730 pp. 1–6. [Online]. Available: https://doi.org/10.1109/DAC56929.2023.10247730

[30] Y. Hu, C. Imes, X. Zhao, S. Kundu, P. A. Beerel, S. P. Crago, and J. P. Walters, "Pipeedge: Pipeline parallelism for large-scale model inference on heterogeneous edge devices," in 2022 25th Euromicro Conference on Digital System Design (DSD), Aug 2022. doi: 10.1109/DSD57027.2022.00048. ISSN 2771-2508 pp. 298–307. [Online]. Available: https://doi.org/10.1109/DSD57027.2022.00048

[31] B. Jang, I.-C. Yoo, and D. Yook, "Pipeline parallelism with elastic averaging," IEEE Access, vol. 12, pp. 5477–5489, 2024. doi: 10.1109/ACCESS.2024.3350193. [Online]. Available: https://doi.org/10.1109/ACCESS.2024.3350193

[32] B. Jeon, M. Wu, S. Cao, S. Kim, S. Park, N. Aggarwal, C. Unger, D. Arfeen, P. Liao, X. Miao, M. Alizadeh, G. R. Ganger, T. Chen, and Z. Jia, "Graphpipe: Improving performance and scalability of dnn training with graph pipeline parallelism," 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2406.17145

[33] X. Zhang, "Mixtran: An efficient and fair scheduler for mixed deep learning workloads in heterogeneous gpu environments," Cluster Computing, vol. 27, no. 3, pp. 2775–2784, Jun 2024. doi: 10.1007/s10586-023-04104-9. [Online]. Available: https://doi.org/10.1007/s10586-023-04104-9

[34] Z. Liu, S. Cheng, H. Zhou, and Y. You, "Hanayo: Harnessing wave-like pipeline parallelism for enhanced large model training efficiency," in Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, ser. SC '23. New York, NY, USA: Association for Computing Machinery, 2023. doi: 10.1145/3581784.3607073. ISBN 9798400701092. [Online]. Available: https://doi.org/10.1145/3581784.3607073

[35] F. Lubrano, C. Vercellino, G. Vitali, P. Viviani, A. Scionti, and O. Terzo, "Advanced resource allocation in the context of heterogeneous workflows management," in Proceedings of the 2nd Workshop on Workflows in Distributed Environments, ser. WiDE '24. New York, NY, USA: Association for Computing Machinery, 2024. doi: 10.1145/3642978.3652835. ISBN 9798400705465 p. 14–20. [Online]. Available: https://doi.org/10.1145/3642978.3652835

[36] S. Liu, L. Sha, and T. Abdelzaher, Criticality-Based Data Segmentation and Resource Allocation in Machine Inference Pipelines. Cham: Springer International Publishing, 2023, pp. 335–352. ISBN 978-3-031-40787-1. [Online]. Available: https://doi.org/10.1007/978-3-031-40787-1_11

[37] S. P. J. K. Jargalsaikhan Narantuya, Jun-Sik Shin, "Multi-agent deep reinforcement learning-based resource allocation in hpc/ai converged cluster," Computers, Materials & Continua, vol. 72, no. 3, pp. 4375–4395, 2022. doi: 10.32604/cmc.2022.023318. [Online]. Available: https://doi.org/10.32604/cmc.2022.023318

[38] N. M. Otterness, "Developing real-time gpu-sharing platforms for artificial-intelligence applications," Ph.D. dissertation, "University of North Carolina at Chapel Hill". ISBN 9798841734833 2022, copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2023-03-08. [Online]. Available: https://doi.org/10.17615/y9wq-8q72

[39] M. M. Islam, "Dynamic resource allocation for ai/ml applications in edge computing: Framework architecture and optimization methods," Journal of Artificial Intelligence General science (JAIGS), vol. 3, no. 1, p. 220–234, Apr. 2024. doi: 10.60087/jaigs.v3i1.116. [Online]. Available: https://doi.org/10.60087/jaigs.v3i1.116

[40] T. Chen, Q. Tang, and G. Liu, "Efficient task scheduling and resource allocation for ai training services in native ai wireless networks," in 2023 IEEE International Conference on Communications Workshops (ICC Workshops). IEEE, 2023. doi: 10.1109/ICCWorkshops57953.2023.10283537 pp. 637–642. [Online]. Available: https://doi.org/10.1109/ICCWorkshops57953.2023.10283537

[41] J. Park, M. Yu, J. Kwon, J. Park, J. Lee, and Y. Kwon, "NEST-C: A deep learning compiler framework for heterogeneous computing systems with artificial intelligence accelerators," ETRI Journal, vol. 46, no. 5, pp. 851–864, 2024. doi: 10.4218/etrij.2024-0139. [Online]. Available: https://doi.org/10.4218/etrij.2024-0139

[42] M. Li, F. Wu, Q. Kou, L. Qian, X. Chen, and X. Lan, "Resource orchestration and scheduling algorithms for enhancing distributed reinforcement learning," in 2023 China Automation Congress (CAC), Nov 2023. doi: 10.1109/CAC59555.2023.10451295. ISSN 2688-0938 pp. 8450–8455. [Online]. Available: https://doi.org/10.1109/CAC59555.2023.10451295

[43] F. Nikolaidis, M. Symeonides, and D. Trihinas, "Towards efficient resource allocation for federated learning in virtualized managed environments," Future Internet, vol. 15, no. 8, 2023. doi: 10.3390/fi15080261. [Online]. Available: https://doi.org/10.3390/fi15080261

[44] H. Yue and L. Chen, "Dynamic Scaling Strategies for AI Workloads in Cloud Environments," Asian American Research Letters Journal, vol. 1, no. 2, Apr 2024. [Online]. Available: https://aarlj.com/index.php/AARLJ/article/view/25

[45] T. Kim, H. Kim, G.-I. Yu, and B.-G. Chun, "BPipe: Memory-balanced pipeline parallelism for training large language models," in Proceedings of the 40th International Conference on Machine Learning, ser. Proceedings of Machine Learning Research, A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, Eds., vol. 202. Proceedings of Machine Learning Research (PMLR), 23–29 Jul 2023, pp. 16 639–16 653. [Online]. Available: https://proceedings.mlr.press/v202/kim23l.html

[46] G. Liang, S. N. Daud, and N. A. Ismail, "Evolution of gpu virtualization to resource pooling," in Proceedings of SPIE, 08 2023. doi: 10.1117/12.2685490 p. 35. [Online]. Available: https://doi.org/10.1117/12.2685490

[47] S. I. Venieris, C.-S. Bouganis, and N. D. Lane, "Multiple-deep neural network accelerators for next-generation artificial intelligence systems," Computer, vol. 56, no. 3, pp. 70–79, March 2023. doi: 10.1109/MC.2022.3176845. [Online]. Available: https://doi.org/10.1109/MC.2022.3176845

[48] A. Sun, W. Zhao, X. Han, C. Yang, X. Zhang, Z. Liu, C. Shi, and M. Sun, "Seq1F1B: Efficient sequence-level pipeline parallelism for large language model training," 2024. [Online]. Available: https://arxiv.org/abs/2406.03488

[49] H. Choi, B. H. Lee, S. Y. Chun, and J. Lee, "Towards accelerating model parallelism in distributed deep learning systems," PLOS ONE, vol. 18, no. 11, pp. 1–15, 11 2023. doi: 10.1371/journal.pone.0293338. [Online]. Available: https://doi.org/10.1371/journal.pone.0293338