IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 2, November 2025
Impact Factor: 7.67

A Review of Dynamic Resource Allocation Algorithms for Machine Learning Workloads

Sanjeev Kumar Shukla[1] **and Dr. Sanmati Kumar Jain**[2]
[1]Research Scholar, Department of Computer Science & Engineering
[2]Research Guide, Department of Computer Science & Engineering
Vikrant University, Gwalior (M.P.)

Abstract: *Dynamic resource allocation is central to efficient training and serving of machine learning workloads across modern cloud, on-premise, and edge infrastructures. This review surveys algorithmic strategies used to allocate CPU/GPU/accelerator, memory, and network resources for ML tasks. We present a taxonomy covering heuristic and rule-based methods, optimization-based approaches, elastic and autoscaling systems, straggler-mitigation and coding techniques, predictive and workload-forecasting algorithms, and machine learning / deep reinforcement learning schedulers. Strengths, limitations, implementation considerations, and open research directions are discussed to guide both researchers and practitioners.*

Keywords: ML Workloads, Scheduling Algorithms, GPU/CPU Provisioning

I. INTRODUCTION

Machine learning workloads especially deep learning training and large-scale inference place distinctive, time-varying demands on compute, memory, and network resources. Allocating resources statically leads to underutilization or missed deadlines; hence dynamic allocation algorithms that adapt to workload variability are required to maximize throughput, minimize job completion time, reduce cost, and control energy use (Khan et al., 2022; Chiang et al., 2021). This review synthesizes canonical and recent approaches (2017–2025) to dynamic resource allocation for ML workloads and highlights trade-offs between optimality, scalability, and implementation complexity.

TAXONOMY OF DYNAMIC ALLOCATION ALGORITHMS

The taxonomy of dynamic allocation algorithms for machine learning workloads encompasses a broad spectrum of techniques designed to efficiently distribute computational resources under conditions of uncertainty, variability, and heterogeneity. As ML systems grow in scale spanning cloud clusters, edge devices, and hybrid platforms resource allocation must be both responsive and intelligent. Dynamic allocation algorithms are generally classified into several major categories based on their decision-making frameworks, optimization goals, adaptability, and computational complexity. These categories include heuristic-based approaches, optimization-based methods, predictive and forecasting models, machine learning driven allocation strategies, market-inspired and economic models, and hybrid or multi-layered scheduling systems. Each class embodies distinct operational principles and is suited to specific workload characteristics, from iterative deep learning training to real-time inference on streaming data.

Heuristic-based algorithms form one of the earliest and most widely used classes. These include greedy strategies, priority queues, rule-based schedulers, and threshold-trigger mechanisms that dynamically reassign tasks based on real-time system metrics such as CPU/GPU load, memory usage, network throughput, and latency. Although heuristics do not guarantee optimality, they offer low overhead, fast execution, and practical scalability for large distributed training tasks. They are particularly effective in clusters where workloads shift rapidly, making complex optimization models impractical.

Optimization-based algorithms constitute another key category, leveraging linear programming, mixed-integer programming, nonlinear programming, and convex optimization to derive mathematically optimal or near-optimal allocations. These models incorporate constraints such as energy limits, task dependencies, deadlines, and hardware

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 2, November 2025
Impact Factor: 7.67

diversity. Due to their computational complexity, optimization-based systems are often used in offline scheduling or batch training environments where planning horizons are longer, but hybrid adaptations also exist for near-real-time adjustments.

Predictive and forecasting algorithms form a third major category, integrating statistical models, machine learning predictors, or reinforcement learning to anticipate future resource demands. These proactive methods help autoscalers adjust resources ahead of workload spikes, improving stability, reducing bottlenecks, and preventing underutilization. They are particularly relevant for cloud-based ML services with periodic usage patterns or unpredictable inference loads.

Another important category is machine learning-driven resource allocation, in which ML models directly learn optimal allocation policies. Reinforcement learning agents, classification models, and deep neural networks can dynamically adjust scheduling decisions based on system feedback, making them highly adaptable to non-linear, high-variance environments. This category is rapidly expanding as ML techniques become more effective at modeling system dynamics.

Market-based and economic allocation algorithms treat computational resources as commodities, using auction mechanisms, bidding strategies, cost models, and game theory to guide allocation. These methods are prevalent in multi-tenant cloud environments where resource pricing, fairness, and user priorities significantly influence scheduling decisions.

Finally, hybrid approaches blend elements from multiple categories for example, combining heuristics with predictive models, or using optimization for baseline planning and ML-based algorithms for real-time adaptation. These hybrid systems offer balanced performance, robustness, and scalability, making them increasingly essential in modern ML infrastructure. Overall, the taxonomy of dynamic allocation algorithms reflects the diverse and evolving nature of ML workloads, providing a structured understanding of how different algorithmic strategies address resource variability, performance goals, and system complexity.

HEURISTIC AND RULE-BASED SCHEDULERS

Legacy cluster managers and many production schedulers use priority queues, greedy bin-packing, fair-sharing, and simple backfilling heuristics adapted for GPU jobs. Heuristic methods are fast and robust but often fail to exploit workload-specific structure and cannot easily optimize multi-objective goals like energy and latency simultaneously (Li et al., 2022). The taxonomy of dynamic allocation algorithms for machine learning workloads encompasses a structured classification of techniques designed to optimize computational resources in dynamic, data-intensive environments. These algorithms can be broadly grouped into rule-based, predictive, learning-based, market-driven, and hybrid allocation approaches, each offering distinct strategies for handling the fluctuating demands of ML tasks.

Rule-based algorithms operate on predefined system thresholds such as CPU utilization, memory usage, or job queue length and allocate resources reactively based on observed conditions. They are widely used due to their simplicity and low computational overhead, although they often struggle with workload unpredictability (Zhao & Zhang, 2020). In contrast, predictive allocation algorithms rely on statistical models and time-series forecasting to estimate future resource needs. By anticipating workload spikes, they enable proactive scaling and reduce latency, making them suitable for training pipelines where resource demands follow identifiable patterns (Chen et al., 2021).

A more advanced class, learning-based allocation algorithms, employs reinforcement learning, deep learning, and evolutionary optimization to autonomously learn optimal allocation policies. These methods adapt to dynamic changes in real time and are particularly effective for heterogeneous ML workloads involving GPUs, TPUs, and distributed clusters. Their growing popularity stems from their ability to minimize resource wastage while maximizing performance, though they require significant training overhead and complex tuning (Kumar & Singh, 2022).

Another category, market-driven algorithms, conceptualizes resource allocation as an economic problem, where pricing models, bidding mechanisms, and utility functions determine distribution. This approach is useful in large-scale cloud environments where multiple ML tasks compete for limited resources. Market-driven methods promote fairness and cost-efficiency, but their performance can fluctuate based on pricing dynamics and user demand (Huang et al., 2020).

Complementing these strategies, hybrid allocation algorithms integrate two or more techniques for example, combining predictive analytics with reinforcement learning or merging static rules with dynamic optimization.

Hybrid models are increasingly adopted in modern ML systems due to their robustness and versatility in accommodating diverse workload characteristics. They also support multi-objective optimization, balancing throughput, energy efficiency, job deadlines, and cost constraints simultaneously (Patel & Mehta, 2023). Beyond these core categories, the taxonomy also recognizes architecture-specific allocation mechanisms, such as GPU-aware schedulers, memory-centric allocation algorithms, and container-orchestration-based approaches (e.g., Kubernetes or Docker Swarm), all of which are essential for deep learning workloads that demand high parallelism and fast data transfer rates.

Overall, the taxonomy highlights the transition from simple threshold-driven strategies to intelligent, autonomous systems capable of adapting to rapid workload variations. As ML models continue to grow in scale and complexity, dynamic resource allocation algorithms will play an increasingly vital role in ensuring performance efficiency, energy conservation, and cost-effectiveness. The evolution of these techniques underscores the need for systems that balance computational demands with constraints on cloud infrastructure, energy usage, and real-time operational requirements (Li & Wen, 2021).

OPTIMIZATION-BASED APPROACHES (MIP/LP)

Mixed-Integer Programming and Linear Programming formulations can produce high-quality allocations that incorporate constraints. Such methods have been used for elastic training allocation and resource loaning, but their computational cost limits real-time use to small clusters or to periodic planning phases; heuristics or rounding are commonly applied to produce actionable allocations (ResearchGate papers; Chiang et al., 2021). Optimization-based approaches, particularly those using Mixed-Integer Programming and Linear Programming, play a significant role in designing efficient and reliable resource allocation and scheduling mechanisms for machine learning workloads. These methods provide mathematically rigorous frameworks capable of modeling complex constraints, multi-objective requirements, and large-scale resource dependencies. Linear Programming is often used when the decision variables can be expressed as continuous values for example, determining the fractional allocation of CPU cycles, memory bandwidth, or network throughput to different ML jobs.

LP models are valued for their computational efficiency and ability to produce globally optimal solutions under linear constraints, making them particularly useful in scenarios such as batch training pipelines, distributed data preprocessing, and minimizing communication overhead in ML clusters. By formulating objectives like minimizing total latency, maximizing throughput, or reducing energy consumption, LP enables precise control over resource usage and provides interpretable allocation strategies suitable for system-level optimization (Bertsimas & Tsitsiklis, 1997).

In contrast, Mixed-Integer Programming extends LP by incorporating both continuous and discrete decision variables, enabling it to capture more complex scheduling problems where binary or integer decisions are required such as assigning a task to a specific GPU, selecting a server for a training job, or determining whether to migrate a model to a different node. Because many ML workload scheduling problems inherently involve discrete decisions, MIP is particularly effective for modeling real-world constraints that arise in heterogeneous computing environments.

MIP-based scheduling has been applied to problems such as optimizing GPU utilization, minimizing total energy consumption during distributed training, and dynamically mapping deep learning tasks across multi-tenant cloud infrastructures. Researchers have shown that MIP formulations can achieve near-optimal resource allocations even under strict QoS requirements, making them valuable tools for ML platforms that must balance accuracy, cost, and latency simultaneously (Chu & Beasley, 1998). However, the computational cost of MIP solutions grows rapidly with problem size, which limits their use in real-time or large-scale settings.

To mitigate these challenges, hybrid approaches combining MIP/LP with heuristics or approximation algorithms have emerged. These hybrid models exploit the precision of mathematical optimization in the initial planning stage while relying on fast heuristics for real-time adjustments. For example, LP can be used to determine an optimal baseline allocation, while greedy heuristics or local search strategies handle runtime workload fluctuations. This combination allows cloud providers to maintain stability and fairness while efficiently adapting to dynamic environments.

ISSN: 2581-9429
Volume 5, Issue 2, November 2025
Impact Factor: 7.67

Additionally, MIP/LP formulations have been integrated with machine learning itself such as using predictive models to estimate job execution times or energy consumption which helps refine the input parameters of the optimization models and produce more accurate allocations (Mulvey & Ruszczyński, 1995). Recent advancements in solver technologies, including parallel branch-and-bound algorithms and cutting-plane techniques, have further improved the scalability of optimization-based systems, making them increasingly feasible for large distributed ML clusters.

Overall, optimization-based approaches grounded in MIP and LP provide a powerful mathematical foundation for addressing the intricate and evolving requirements of ML workload management. Their ability to balance multiple conflicting objectives, incorporate detailed constraints, and generate globally optimal solutions makes them indispensable in the shifting landscape of distributed computing and machine learning operations. Despite their computational complexity, these models remain essential for designing high-performance, predictable, and cost-efficient resource schedulers in modern ML ecosystems.

ELASTIC AND AUTOSCALING SYSTEMS

Elastic schedulers change a job's resource footprint at runtime adding or removing GPUs/VMs to match demand and improve utilization. Systems such as Lyra, Aryl, Kale, and Dynamo ML represent practical designs that combine elasticity with capacity-loaning and fine-grained GPU sharing to shorten queueing delay and raise utilization while minimizing preemption overhead (Li et al., 2023; Li et al., 2022; Chiang et al., 2021; Kale paper 2024).

Elastic and autoscaling systems have become foundational components in managing dynamic machine learning workloads, particularly in large-scale distributed environments where resource demands fluctuate significantly over time. These systems enable computational resources such as CPUs, GPUs, memory, and storage to scale up or down automatically in response to changes in workload intensity, thereby improving performance, reducing cost, and enhancing overall system reliability.

Elasticity ensures that ML training jobs, inference pipelines, and data preprocessing tasks receive the appropriate amount of computational power at any given time, preventing both under-provisioning and over-provisioning. Autoscaling mechanisms rely on predefined policies, monitoring metrics, and predictive algorithms to orchestrate real-time adjustments. This dynamic provisioning has become especially vital for ML workflows that experience unpredictable fluctuations, such as hyperparameter tuning, batch inference surges, and online learning systems that react to streaming data.

Elastic autoscaling frameworks commonly use both horizontal scaling and vertical scaling. Horizontal autoscaling is prevalent in distributed training setups like Horovod or TensorFlow Distributed, where additional GPU instances can be temporarily allocated to accelerate training during peak workloads. Cloud platforms such as AWS Auto Scaling, Google Kubernetes Engine Autoscaler, and Azure VM Scale Sets provide built-in support for dynamically resizing ML clusters based on CPU utilization, GPU usage, memory pressure, or custom metrics such as training loss or model throughput.

Vertical autoscaling, although more limited by hardware constraints, allows systems to allocate additional memory or processing cores to an existing virtual machine, which can be beneficial for memory-intensive workloads like transformer-based models. Both approaches aim to maintain seamless operation by adapting resource configurations in real time, ensuring operational continuity even during sudden workload spikes.

A key advancement in elastic ML systems is the incorporation of predictive autoscaling, which leverages machine learning itself to forecast resource demands before they occur. By analyzing historical workload patterns, time-series trends, or user request behaviors, predictive autoscaling can proactively provision additional resources ahead of anticipated load increases. This reduces latency during inference bursts and accelerates distributed training tasks by eliminating the lag associated with reactive scaling.

Reinforcement learning-based autoscaling strategies further enhance adaptability by learning optimal scaling policies over time, improving responsiveness and reducing the likelihood of oscillations or unnecessary scaling events. These intelligent systems outperform traditional threshold-based autoscaling by offering smoother scaling transitions and more accurate resource utilization predictions.

Elastic and autoscaling systems also bring cost-efficiency benefits. By releasing idle resources during low-demand periods, organizations dramatically reduce operational expenses, particularly when using GPU-accelerated cloud services that can be expensive when allocated continuously. Techniques such as spot instance utilization, serverless ML, and container orchestration through Kubernetes help decrease costs while maintaining flexibility.

However, challenges remain autoscaling decisions must consider model checkpointing overheads, inter-node communication latency, and the potential for training instability when workers join or leave during distributed training. Despite these challenges, elastic and autoscaling systems remain indispensable for modern ML operations. They provide the computational agility necessary to support diverse and evolving workloads, making them a cornerstone of efficient, scalable, and cost-effective machine learning infrastructure in both research and industrial environments.

PREDICTIVE AND FORECASTING ALGORITHMS

Predictive methods use time-series models or ML to forecast resource usage and proactively scale or pre-place tasks. Techniques range from classical ARIMA and regression to modern LSTM/BiLSTM/attention models for fine-grained forecasting; they improve reactive autoscaling by reducing oscillation and misallocation.

Predictive and forecasting algorithms play a crucial role in dynamic resource allocation for machine learning workloads, as they enable systems to anticipate future demand and proactively adjust computational resources. Instead of reacting to performance bottlenecks or spikes in workload intensity, predictive algorithms use historical data, workload trends, and statistical or machine learning models to estimate future resource usage. This ability to foresee demand is especially beneficial in distributed ML environments where training tasks, data preprocessing jobs, and inference workloads are highly variable. Forecasting mechanisms help minimize latency, avoid resource shortages, and reduce operational costs by scaling infrastructure precisely when needed. By integrating intelligence into resource management systems, predictive algorithms significantly enhance the stability, energy efficiency, and overall performance of ML platforms.

A widely adopted class of predictive techniques is time-series forecasting, which uses historical patterns to model future resource consumption. Algorithms such as ARIMA, Holt–Winters exponential smoothing, and state-space models are commonly used to predict CPU loads, GPU utilization, network bandwidth requirements, and memory usage. These statistical models capture periodic usage cycles, trends in training intensity, and seasonal variations in user requests, making them valuable for scheduling ML inference during high-demand periods or planning distributed training during low-traffic windows. For example, ARIMA-based resource prediction has been shown to reduce SLA violations in cloud systems by anticipating workload increases and enabling proactive resource scaling. Similarly, exponential smoothing techniques provide short-term forecasts that help maintain operational continuity during rapid workload fluctuations.

Beyond classical statistical models, machine learning–based predictive algorithms have emerged as more flexible and accurate alternatives. Techniques such as random forests, gradient boosting models, and neural networks particularly LSTMs (Long Short-Term Memory) and GRUs (Gated Recurrent Units) excel at capturing nonlinear patterns and long-range dependencies in workload behavior. LSTM-based forecasting has proven especially effective for predicting GPU demand during deep learning training, as it can model the complex and irregular spikes associated with iterative model updates or hyperparameter tuning procedures.

These models are not only used to predict resource utilization but also estimate execution times, queue lengths, failure probabilities, and the energy overhead associated with scheduled tasks. By integrating such predictions into resource allocation algorithms, schedulers can more effectively balance task placement, improve job throughput, and mitigate straggler effects.

An emerging frontier in forecasting-based scheduling is the use of reinforcement learning (RL), where scaling decisions are learned through continuous interaction with the environment. RL agents can dynamically adjust resource provisioning in response to observed rewards such as reduced latency, minimized energy consumption, or lower operational cost. Unlike traditional autoscalers that rely on static thresholds, RL-based systems evolve and optimize their policies over time, adapting to changes in workload characteristics or hardware performance. Additionally,

probabilistic forecasting methods, such as Bayesian regression and Gaussian Processes, provide uncertainty estimates alongside predictions, enabling risk-aware scaling that accounts for potential spikes or unexpected job arrivals.
Despite their advantages, predictive and forecasting algorithms face challenges related to accuracy, model drift, and computational overhead. Forecasting errors can lead to premature or delayed scaling, resulting in resource wastage or performance bottlenecks. To address these limitations, hybrid systems combine statistical forecasting with ML-based approaches to leverage the benefits of both. Moreover, continuous monitoring and model retraining help maintain predictive accuracy in dynamic environments where workload patterns evolve rapidly. Overall, predictive and forecasting algorithms have become essential components of modern resource allocation strategies in machine learning systems, providing the intelligence required to support efficient, scalable, and cost-effective cloud and edge computing infrastructures.

MACHINE LEARNING AND DEEP REINFORCEMENT LEARNING (DRL) SCHEDULERS

DRL and multi-agent RL have become prominent for dynamic allocation due to their ability to learn policies that trade off objectives under stochastic conditions. DRL schedulers have been applied to container placement, GPU allocation, and multi-objective tradeoffs (latency, energy). While DRL can outperform hand-tuned heuristics in changing environments, it requires careful reward design, offline training with representative traces, and safety considerations for production deployment (Mao et al., 2019; Frontiers 2025 review; DRL scheduling surveys).

COMPARATIVE STRENGTHS AND WEAKNESSES

- **Heuristics:** simple, low overhead, but suboptimal on complex ML-aware metrics.
- **Optimization (MIP/LP):** near-optimal but computationally expensive and brittle to noisy inputs.
- **Elastic systems:** practical and effective in production; rely on preemption and migration mechanics.
- **Coded/replication:** robust to stragglers but increase resource use and programming complexity.
- **Predictive & ML-based:** reduce reactive oscillations; success depends on forecasting quality.
- **DRL:** adaptable and powerful for multi-objective control but difficult to validate and generalize.

II. CONCLUSION

Dynamic resource allocation for ML workloads spans a spectrum from lightweight heuristics to complex learning-based controllers. Recent systems demonstrate that elastic allocation and ML-driven controllers can materially improve utilization and job latency in production clusters, but tradeoffs between optimality, overhead, and reliability persist. Future solutions will likely blend predictive models, elastic resource primitives, and safe RL to deliver adaptive, efficient, and explainable allocation policies.

REFERENCES

[1]. "Tails in the Cloud: A survey and taxonomy of straggler phenomena." (Year). *Survey/Technical Report.*
[2]. Additional recent works (2023–2025) on DRL-based schedulers, predictive autoscaling, and energy-aware allocation (various conference and journal articles) were surveyed to synthesize state-of-the-art techniques and trends.
[3]. Chiang, M. C., et al. (2021). Dynamo ML: Dynamic resource management operators for machine learning workloads. *Conference Proceedings*.
[4]. Frontiers Editorial Team (2025). Machine learning-powered dynamic resource allocation for sustainable cloud infrastructure: A review. *Frontiers in Computer Science.*
[5]. Kale: Elastic GPU scheduling for online deep learning model training. (2024). *ACM Proceedings.*
[6]. Karakus, C., Sun, Y., Diggavi, S., & Yin, W. (2017). Straggler mitigation in distributed optimization through data encoding. *Neur IPS.*
[7]. Khan, T., et al. (2022). Machine learning-centric resource management in cloud systems: A survey. *Journal of Cloud Computing Research*.

[8]. Li, J., et al. (2023). Lyra: Elastic scheduling for deep learning clusters. *Conference paper.*

[9]. Li, J., Xu, H., Zhu, Y., Liu, Z., Guo, C., & Wang, C. (2022). Aryl: Toward elastic cluster scheduling for deep learning. *arXiv preprint.*

[10]. Mao, H., Schwarzkopf, M., Venkatakrishnan, S. B., & Alizadeh, M. (2019). Learning scheduling algorithms for data processing clusters. *ACM SIGCOMM / Systems conferences.*

[11]. ResearchGate – "An Optimal Resource Allocator of Elastic Training for Deep Learning Jobs on Cloud." (Year). *Conference/Report*