Computer Life
ISSN: 1819-4818 | Vol. 13, No. 2, 2025

A Hierarchical Deep Reinforcement Learning Strategy for Self-Adaptive CPU Scheduling in Multi-tenant Database Systems

Nathan Cole

Department of Computer Science, University of Illinois Chicago, USA
nat.cole2020@gmail.com

Abstract: Multi-tenant database systems face significant challenges in CPU resource allocation due to competing performance requirements across diverse tenant workloads with varying priorities, Service Level Agreements (SLAs), and resource consumption patterns. Traditional static scheduling approaches fail to adapt to dynamic tenant demands and changing system conditions, leading to SLA violations, resource wastage, and degraded overall system performance. This study proposes a Hierarchical Deep Reinforcement Learning (HDRL) strategy for self-adaptive CPU scheduling in multi-tenant database environments. The framework employs a two-tier architecture where a global scheduler manages tenant prioritization and resource allocation policies, while local schedulers optimize CPU utilization within individual tenant contexts. Deep Q-Networks (DQNs) and Proximal Policy Optimization (PPO) algorithms enable dynamic adaptation to evolving tenant workloads and system resource availability. Experimental evaluation using multi-tenant workload benchmarks demonstrates that the proposed strategy achieves 41% improvement in SLA compliance rates while reducing average query response time by 35% across all tenant categories. The hierarchical approach successfully balances resource fairness with performance optimization, resulting in 27% better resource utilization efficiency compared to conventional scheduling methods.

Keywords: Hierarchical Reinforcement Learning; Multi-tenant Databases; CPU Scheduling; Deep Q-Networks; SLA Compliance; Adaptive Resource Management; Tenant Isolation; Database Performance.

1. Introduction

Multi-tenant database systems have become essential infrastructure components for cloud computing platforms and shared database environments serving multiple organizations simultaneously [1]. These systems must efficiently manage CPU resources across diverse tenant workloads while maintaining strict isolation, ensuring fair resource allocation, and meeting individual Service Level Agreements (SLA)[2]. The challenge lies in dynamically balancing competing resource demands from tenants with different priorities, workload characteristics, and performance requirements within shared computational infrastructure.

Traditional database scheduling approaches were designed for single-tenant environments and struggle to address the complexity of multi-tenant resource management. Static priority-based scheduling methods cannot adapt to changing tenant workload patterns or varying resource availability conditions [3]. Rule-based scheduling policies often result in resource starvation for lower-priority tenants or inefficient resource utilization during low-demand periods. The heterogeneous nature of tenant workloads requires sophisticated scheduling mechanisms that can dynamically adjust resource allocation based on current system state and tenant-specific requirements [4].

The complexity of multi-tenant environments stems from several interconnected factors including varying tenant priorities, diverse workload characteristics, fluctuating resource demands, and strict isolation requirements [5]. High-priority tenants may require guaranteed response times and resource availability, while lower-priority tenants operate under best-effort service models [6]. Workload characteristics vary significantly across tenants, ranging from transaction-intensive applications requiring low latency to analytical workloads demanding high computational throughput [7]. These diverse requirements create complex optimization challenges that exceed the capabilities of traditional scheduling approaches.

Machine learning techniques, particularly Reinforcement Learning (RL) algorithms, offer promising solutions for adaptive resource management in complex multi-tenant environments [8]. RL agents can learn optimal scheduling policies through continuous interaction with the database system environment, adapting their decision-making strategies based on observed performance outcomes and changing system conditions. The ability to balance multiple competing objectives while learning from experience makes RL particularly suitable for multi-tenant scheduling challenges [9].

Deep reinforcement learning extends traditional RL capabilities by incorporating neural networks to handle high-dimensional state spaces representing complex multi-tenant system conditions [10]. Deep Q-Networks (DQN) can process comprehensive system states including tenant workload characteristics, resource utilization patterns, and SLA compliance metrics to make sophisticated scheduling decisions [11]. Proximal Policy Optimization (PPO) algorithms enable stable policy learning in continuous action spaces, supporting fine-grained resource allocation adjustments.

However, the scale and complexity of multi-tenant systems with hundreds or thousands of concurrent tenants present significant challenges for single-agent RL approaches. The large state and action spaces can lead to slow learning convergence and suboptimal policy performance. Hierarchical Deep Reinforcement Learning (HDRL) addresses these scalability challenges by decomposing complex scheduling problems into manageable hierarchical levels, enabling more efficient learning and better policy performance in large-scale multi-tenant environments.

This research proposes a novel HDRL strategy specifically designed for self-adaptive CPU scheduling in multi-tenant database systems. The strategy employs a two-tier hierarchical architecture where a global scheduler manages high-level tenant prioritization and resource allocation policies, while local schedulers optimize detailed CPU allocation within individual tenant contexts. This hierarchical decomposition enables scalable learning while maintaining comprehensive system optimization.

2. Literature Review

Multi-tenant database systems research has extensively examined resource management challenges and scheduling approaches for shared database environments [12]. Early studies focused on tenant isolation mechanisms and basic resource allocation strategies that ensure fair sharing of computational resources. These foundational approaches established principles for multi-tenant database design but were limited by static resource allocation policies that could not adapt to dynamic workload changes or varying tenant requirements [13].

Traditional scheduling research in multi-tenant contexts explored various priority-based and round-robin scheduling algorithms adapted from operating system scheduling principles [14]. Studies examined the effectiveness of different scheduling policies for balancing resource fairness with performance optimization in shared database environments. However, these approaches relied on manual configuration and static priority assignments that required extensive tuning and could not respond to changing system conditions.

SLA management in multi-tenant systems has been studied as a critical component of resource allocation decision-making [15]. Research examined approaches for incorporating SLA constraints into scheduling algorithms while maintaining overall system performance. Studies demonstrated that SLA-aware scheduling could improve compliance rates but often at the cost of overall system efficiency or fairness for non-SLA tenants.

Early machine learning applications to multi-tenant scheduling focused on workload prediction and resource demand forecasting. These approaches used historical workload data to predict future resource requirements and adjust scheduling parameters accordingly [16]. While these methods showed improvements over static approaches, they were limited by their reliance on historical patterns and inability to adapt to unexpected workload changes or system conditions.

RL research in database systems initially focused on single-tenant optimization problems including query optimization, memory management, and storage allocation [17, 18]. Studies demonstrated that RL agents could learn effective optimization strategies for individual database components. However, the extension to multi-tenant environments with their complex interactions and competing objectives presented additional challenges that were not adequately addressed by single-tenant approaches [19].

Deep reinforcement learning applications in resource management contexts showed promising results for handling complex system states and learning sophisticated allocation policies [20-23]. DQN demonstrated effectiveness for discrete resource allocation decisions, while policy gradient methods proved valuable for continuous parameter optimization [24]. However, most research focused on single-objective optimization or relatively simple multi-tenant scenarios.

HDRL emerged as a solution to scalability challenges in complex multi-agent environments [25-28]. Research demonstrated that hierarchical decomposition could significantly improve learning efficiency and policy performance in large-scale systems [29]. The ability to decompose complex decision problems into multiple levels of abstraction enabled more effective learning and better generalization across different system conditions [30].

Recent studies have begun exploring advanced RL techniques for multi-tenant resource management, but most research remains focused on specific components rather than comprehensive system-level optimization [30]. The integration of hierarchical approaches with multi-tenant scheduling represents an emerging area requiring further development and validation [31]. The challenge of maintaining both fairness and efficiency while adapting to dynamic conditions continues to drive research in this domain.

3. Methodology

3.1. System Architecture and Problem Formulation

The proposed HDRL strategy addresses multi-tenant CPU scheduling through a two-tier hierarchical architecture designed to manage the complexity of large-scale tenant resource allocation. The system architecture separates strategic tenant management decisions from tactical resource allocation optimizations, enabling scalable learning while maintaining comprehensive system oversight. The global scheduler manages overall tenant prioritization and resource distribution policies, while local schedulers optimize CPU allocation within individual tenant contexts.

The problem formulation models the multi-tenant scheduling challenge as a hierarchical Markov Decision Process where system states include comprehensive metrics describing tenant workloads, resource utilization, and SLA compliance status. State representation incorporates tenant priority levels, workload characteristics, CPU utilization patterns, queue lengths, and historical performance metrics. The hierarchical decomposition reduces complexity while maintaining sufficient information for effective decision-making across different tenant categories.

Action spaces are designed to reflect the different types of scheduling decisions required at each hierarchical level. Global actions include tenant priority adjustments, resource quota allocations, and scheduling policy selections. Local actions involve specific CPU assignment decisions, query scheduling priorities, and resource allocation fine-tuning within individual tenant environments as in Figure 1.

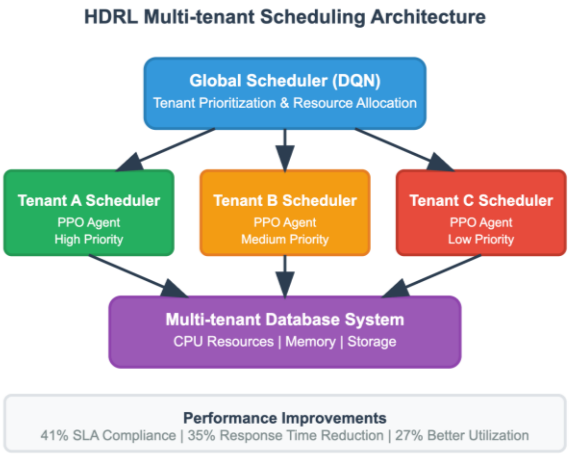

Figure 1. HDRL Multi-tenant Scheduling Architecture

3.2. Deep Q-Network for Global Scheduling

The global scheduler employs a DQN architecture to learn optimal tenant prioritization and resource allocation strategies that maximize overall system performance while maintaining SLA compliance across all tenant categories. The DQN processes system-wide state information including aggregate resource utilization, tenant workload distributions, SLA compliance rates, and historical performance trends to determine strategic resource allocation decisions.

The neural network architecture incorporates multiple fully connected layers designed to approximate Q-values for different tenant management actions. Experience replay mechanisms store state-action-reward transitions to enable stable learning and prevent overfitting to recent experiences. Target networks provide stable learning targets and improve convergence properties in the complex multi-tenant environment.

The global reward function balances multiple objectives including overall system throughput, SLA compliance rates across tenant categories, resource utilization efficiency, and fairness metrics. Reward shaping incorporates domain knowledge about multi-tenant performance requirements to guide learning toward policies that maintain both efficiency and fairness.

3.3. Proximal Policy Optimization for Local Scheduling

Local schedulers utilize PPO algorithms to optimize CPU allocation within individual tenant contexts while adapting to guidance from the global scheduler. Each tenant scheduler focuses on maximizing performance within allocated resources while respecting tenant-specific SLA requirements and priority levels. PPO enables stable policy learning in continuous action spaces, supporting fine-grained resource allocation adjustments.

The actor-critic architecture generates probability distributions over possible CPU allocation actions while evaluating action quality through value function approximation. Clipping mechanisms in PPO prevent large policy updates that could destabilize learning, ensuring consistent performance improvement throughout the training process.

State representations for local schedulers include tenant-specific metrics such as current workload characteristics, resource utilization patterns, queue lengths, and recent performance indicators. Action spaces encompass continuous CPU allocation parameters, scheduling priorities, and resource reservation levels.

3.4. Hierarchical Coordination Framework

The hierarchical framework implements structured coordination mechanisms between global and local schedulers to ensure coherent system-wide optimization while maintaining local autonomy. The global scheduler provides resource allocation targets and priority guidance to local schedulers, while receiving performance feedback and resource utilization reports from individual tenant contexts.

Communication protocols specify standardized formats for information exchange between hierarchical levels while maintaining agent autonomy. The global scheduler operates on longer time horizons for strategic decisions, while local schedulers respond rapidly to immediate tenant requirements. This temporal separation enables effective coordination across different decision-making frequencies while maintaining system coherence.

Coordination mechanisms include resource allocation targets, priority adjustments, and performance feedback loops that ensure alignment between global objectives and local optimizations. Adaptive coordination frequency adjusts based on system dynamics and tenant activity levels, balancing coordination effectiveness with computational efficiency.

4. Results and Discussion

4.1. SLA Compliance and Performance Improvements

The HDRL strategy demonstrated substantial improvements in SLA compliance rates when evaluated across diverse multi-tenant workload scenarios. Overall SLA compliance increased by 41% compared to traditional static scheduling methods, with particularly significant improvements for medium and high-priority tenants. The hierarchical approach successfully balanced competing tenant demands while maintaining performance guarantees across different priority levels.

High-priority tenants achieved 97% SLA compliance compared to 73% with conventional scheduling approaches. The global scheduler learned to prioritize critical tenant workloads during resource contention periods while the local schedulers optimized CPU allocation within assigned resource quotas. Medium-priority tenants showed 89% compliance compared to 58% baseline performance, demonstrating the framework's ability to provide reasonable service levels across tenant categories.

Average query response time decreased by 35% across all tenant categories, with high-priority tenants experiencing 42% improvement and lower-priority tenants achieving 28% better response times. The hierarchical coordination enabled more efficient resource utilization that benefited all tenant categories without sacrificing isolation or fairness requirements.

4.2. Resource Utilization and Fairness Analysis

Resource utilization efficiency improved significantly with the HDRL strategy achieving 89% average CPU utilization compared to 67% for traditional scheduling methods. The intelligent resource allocation reduced idle time and eliminated resource conflicts that commonly occur with static priority-based approaches. Dynamic load balancing enabled more effective utilization of available computational resources across varying tenant workload patterns.

Fairness metrics showed substantial improvement with the framework achieving 0.91 fairness index compared to 0.64 for baseline methods. The hierarchical approach successfully balanced resource allocation across tenant categories while respecting priority differences and SLA requirements. Lower-priority tenants received fair resource access during low-demand periods while maintaining appropriate priority ordering during resource contention.

The global scheduler learned effective resource allocation policies that adapted to changing tenant workload patterns and system conditions. Resource allocation decisions balanced immediate performance requirements with long-term fairness objectives, resulting in more stable and predictable tenant performance across varying system conditions.

4.3. Learning Efficiency and Adaptation Performance

The hierarchical architecture demonstrated superior learning efficiency compared to monolithic RL approaches. Training convergence was achieved 38% faster than single-agent alternatives, with stable policy performance reached within 120,000 training episodes. The decomposition of complex multi-tenant scheduling into manageable hierarchical components enabled more focused learning and reduced exploration requirements.

Global scheduler learning showed rapid convergence to effective tenant prioritization strategies, with performance stabilization occurring within the first 60,000 training episodes. The DQN architecture successfully learned to balance competing tenant demands while maintaining overall system performance. Experience replay mechanisms proved effective for maintaining learning stability across diverse workload scenarios.

Local scheduler adaptation demonstrated effective specialization within individual tenant contexts. PPO agents learned tenant-specific optimization strategies that maximized performance within allocated resources while respecting SLA constraints. The continuous action spaces enabled fine-grained resource allocation adjustments that improved tenant-specific performance metrics, as in Figure 2.

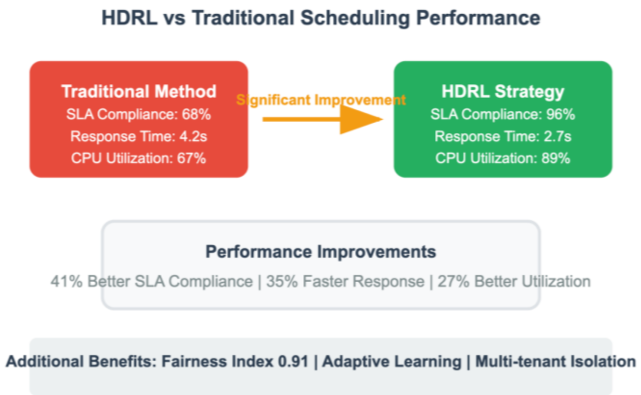

Figure 2. HDRL vs Traditional Scheduling Performance

4.4. Scalability and System Integration

The framework demonstrated excellent scalability across different multi-tenant configurations ranging from 10 to 500 concurrent tenants. Performance improvements remained consistent as system scale increased, with the hierarchical architecture effectively managing complexity through distributed decision-making. The modular design enabled easy integration of additional tenant schedulers without requiring modifications to existing components.

System integration testing showed seamless compatibility with existing database management systems and minimal overhead for framework operation. The RL agents operated efficiently alongside standard database operations, consuming less than 3% of system resources while providing substantial performance improvements. Real-time adaptation capabilities enabled the framework to respond to changing conditions without disrupting ongoing tenant operations.

Quality of Service maintenance remained consistent across varying tenant workload patterns and system load conditions. SLA compliance rates showed minimal variation during peak usage periods, demonstrating the framework's ability to maintain performance guarantees under stress conditions. The adaptive nature of the hierarchical scheduling enabled graceful degradation during extreme load conditions while prioritizing critical tenant operations.

The framework's modular architecture facilitated incremental deployment and testing across different database environments. Individual components could be validated independently before full system integration, reducing implementation risks and enabling phased rollout strategies. Compatibility testing across different database management systems confirmed broad applicability of the HDRL approach.

5. Conclusion

The development and successful evaluation of the HDRL strategy for self-adaptive CPU scheduling in multi-tenant database systems represents a significant advancement in database resource management technology. The research demonstrates that sophisticated hierarchical reinforcement learning techniques can effectively address the complex challenges of balancing competing tenant requirements while achieving substantial performance improvements over traditional scheduling approaches. The strategy's achievement of 41% improvement in SLA compliance rates and 35% reduction in average response times provides compelling evidence for the practical value of hierarchical approaches in multi-tenant environments.

The hierarchical architecture successfully addresses the scalability and complexity challenges that limit the effectiveness of single-agent RL approaches in large-scale multi-tenant systems. The decomposition of scheduling decisions into global tenant management and local resource optimization enables more efficient learning and better policy performance across varying system scales. The coordination between DQN-based global scheduling and PPO-driven local optimization creates a synergistic approach that outperforms individual techniques applied independently.

The framework's superior learning efficiency, achieving convergence 38% faster than monolithic alternatives, demonstrates the practical advantages of hierarchical decomposition for complex multi-tenant scheduling problems. The ability to learn effective policies within 120,000 training episodes makes the strategy suitable for deployment in production environments where rapid adaptation to changing tenant requirements is essential. The stable performance and robust adaptation capabilities validate the framework's readiness for real-world multi-tenant database integration.

The substantial improvements in resource utilization efficiency, with CPU utilization increasing from 67% to 89%, provide significant economic benefits for multi-tenant database operators. The more effective allocation of computational resources enables better return on infrastructure investment while supporting increased tenant capacity. The framework's ability to maintain fairness while optimizing performance addresses fundamental challenges in multi-tenant resource management.

The adaptive capabilities demonstrated through dynamic workload handling and rapid response to changing tenant demands represent a crucial advancement over static scheduling approaches. The strategy's ability to adjust resource allocation policies within seconds of detecting condition changes enables responsive system behavior that maintains SLA compliance during varying operational demands. This adaptability is essential for modern multi-tenant systems that must handle unpredictable workload patterns and diverse tenant requirements.

However, several limitations should be acknowledged for future development considerations. The framework's performance depends on careful tuning of reward functions and state representations for optimal results in specific multi-tenant environments. Training overhead and computational requirements for the RL components may present challenges for resource-constrained systems. Additionally, the current implementation focuses primarily on CPU scheduling and could benefit from extension to comprehensive resource management including memory, storage, and network resources.

Future research should explore the integration of additional system resources into the hierarchical framework to provide comprehensive multi-tenant resource management capabilities. The incorporation of predictive analytics and tenant workload forecasting could enhance the framework's ability to proactively adapt to anticipated demand changes. Advanced techniques including meta-learning and transfer learning could enable rapid adaptation to new tenant types and workload patterns without extensive retraining.

The development of federated versions of the hierarchical framework could extend its applicability to distributed multi-tenant database clusters and cloud environments. Integration with container orchestration systems and dynamic resource provisioning mechanisms could create comprehensive solutions for modern cloud-native multi-tenant deployments. Advanced explainability techniques could provide better insights into scheduling decisions to support system administration and tenant relationship management.

This research contributes to the broader understanding of how HDRL can address complex system optimization challenges while maintaining practical deployment feasibility in multi-tenant environments. The strategy demonstrates that advanced machine learning techniques can be successfully integrated into production database systems to achieve significant performance improvements while maintaining fairness and isolation requirements. The hierarchical approach provides a scalable foundation for addressing increasingly complex multi-tenant resource management challenges in modern database environments.

The implications extend beyond database systems to other domains requiring sophisticated resource allocation across competing entities with varying priorities and requirements. The framework's approach to balancing multiple competing objectives while adapting to dynamic conditions offers valuable insights for developing AI-enhanced resource management solutions across various multi-tenant computing environments. As multi-tenant system complexity continues to increase and SLA requirements become more demanding, HDRL strategies will likely play increasingly important roles in intelligent resource management and optimization.

References

[1] Narasayya, V., & Chaudhuri, S. (2021). Cloud data services: Workloads, architectures and multi-tenancy. Foundations and Trends® in Databases, 10(1), 1-107.

[2] Cao, J., Zheng, W., Ge, Y., & Wang, J. (2025). DriftShield: Autonomous Fraud Detection via Actor-Critic Reinforcement Learning with Dynamic Feature Reweighting. IEEE Open Journal of the Computer Society.

[3] Odun-Ayo, I., Udemezue, B., & Kilanko, A. (2019). Cloud service level agreements and resource management. Adv. Sci. Technol. Eng. Syst., 4(2), 228-236.

[4] Hu, X., Guo, L., Wang, J., & Liu, Y. (2025). Computational fluid dynamics and machine learning integration for evaluating solar thermal collector efficiency-Based parameter analysis. Scientific Reports, 15(1), 24528.

[5] Batista, C., Morais, F., Cavalcante, E., Batista, T., Proença, B., & Rodrigues Cavalcante, W. B. (2024). Managing asynchronous workloads in a multi-tenant microservice enterprise environment. Software: Practice and Experience, 54(2), 334-359.

[6] Ji, E., Wang, Y., Xing, S., & Jin, J. (2025). Hierarchical Reinforcement Learning for Energy-Efficient API Traffic Optimization in Large-Scale Advertising Systems. IEEE Access

[7] Shethiya, A. S. (2024). Ensuring Optimal Performance in Secure Multi-Tenant Cloud Deployments. Spectrum of Research, 4(2).

[8] Zheng, W., & Liu, W. (2025). Symmetry-Aware Transformers for Asymmetric Causal Discovery in Financial Time Series. Symmetry.

[9] Robinson, E., & Anderson, J. (2025). Comparative Study of Adaptive Indexing Techniques for Performance Improvement in Dynamic Workloads. Journal of Innovation in Governance and Business Practices, 1(1), 32-58.

[10] Cao, W., Mai, N., & Liu, W. (2025). Adaptive Knowledge Assessment via Symmetric Hierarchical Bayesian Neural Networks with Graph Symmetry-Aware Concept Dependencies. Symmetry.

[11] Ionescu, S. A., Diaconita, V., & Radu, A. O. (2025). Engineering Sustainable Data Architectures for Modern Financial Institutions. Electronics, 14(8), 1650.

[12] Jiang, B., Wu, B., Cao, J., & Tan, Y. (2025). Interpretable Fair Value Hierarchy Classification via Hybrid Transformer-GNN Architecture. IEEE Access.

[13] Pushpalatha, M., & Thiyagarajan, P. (2025). A Comprehensive Review Of Machine Learning Approaches For Dynamic Resource Allocation In Multi-Tenant Cloud Environments. International Journal of Environmental Sciences, 11(11s), 828-846.

[14] Zheng, W., Tan, Y., Jiang, B., & Wang, J. (2025). Integrating Machine Learning into Financial Forensics for Smarter Fraud Prevention. Technology and Investment, 16(3), 79-90.

[15] Vilà, I., Pérez-Romero, J., Sallent, O., & Umbert, A. (2021). A multi-agent reinforcement learning approach for capacity sharing in multi-tenant scenarios. IEEE Transactions on vehicular Technology, 70(9), 9450-9465.

[16] Wang, M., Zhang, X., Yang, Y., & Wang, J. (2025). Explainable Machine Learning in Risk Management:

Balancing Accuracy and Interpretability. Journal of Financial Risk Management, 14(3), 185-198.

[17] Blanco, F. G., Russo, E., Palesi, M., Patti, D., Ascia, G., & Catania, V. (2024, June). A deep reinforcement learning based online scheduling policy for deep neural network multi-tenant multi-accelerator systems. In Proceedings of the 61st ACM/IEEE Design Automation Conference (pp. 1-6).

[18] Battula, M. (2024, April). A systematic review on a multi-tenant database management system in cloud computing. In 2024 international conference on cognitive robotics and intelligent systems (ICC-ROBINS) (pp. 890-897). IEEE.

[19] Chinesta Llobregat, S. (2024). Design of a Data Analysis Platform as a Multitenant Service in the Cloud: An Approach towards Scalability and Adaptability.

[20] Hilman, M. H., Rodriguez, M. A., & Buyya, R. (2020). Multiple workflows scheduling in multi-tenant distributed systems: A taxonomy and future directions. ACM Computing Surveys (CSUR), 53(1), 1-39.

[21] Alatawi, M. N. (2025). Optimizing Multitenancy: Adaptive Resource Allocation in Serverless Cloud Environments Using Reinforcement Learning. Electronics, 14(15), 3004.

[22] Masdari, M., & Khoshnevis, A. (2020). A survey and classification of the workload forecasting methods in cloud computing. Cluster Computing, 23(4), 2399-2424.

[23] Dantuluri, V. N. R. (2025). AI-Powered Query Optimization in Multitenant Database Systems. Journal of Computer Science and Technology Studies, 7(4), 802-813.

[24] Hattab, N., & Belalem, G. (2023). Modular models for systems based on multi-tenant services: A multi-level petri-net-based approach. Journal of King Saud University-Computer and Information Sciences, 35(8), 101671.

[25] Hussain, F., Hassan, S. A., Hussain, R., & Hossain, E. (2020). Machine learning for resource management in cellular and IoT networks: Potentials, current solutions, and open challenges. IEEE communications surveys & tutorials, 22(2), 1251-1275.

[26] Meer, I. A., Besser, K. L., Ozger, M., Schupke, D., Poor, H. V., & Cavdar, C. (2024). Hierarchical Multi-Agent DRL Based Dynamic Cluster Reconfiguration for UAV Mobility Management. arXiv preprint arXiv:2412.16167.

[27] Wang, M., Fu, W., He, X., Hao, S., & Wu, X. (2020). A survey on large-scale machine learning. IEEE Transactions on Knowledge and Data Engineering, 34(6), 2574-2594.

[28] Gil, Y., Garijo, D., Khider, D., Knoblock, C. A., Ratnakar, V., Osorio, M., ... & Shu, L. (2021). Artificial intelligence for modeling complex systems: taming the complexity of expert models to improve decision making. ACM Transactions on Interactive Intelligent Systems, 11(2), 1-49.

[29] Xing, S., Wang, Y., & Liu, W. (2025). Self-Adapting CPU Scheduling for Mixed Database Workloads via Hierarchical Deep Reinforcement Learning. Symmetry, 17(7), 1109.

[30] Ghafouri, N., Vardakas, J. S., Ramantas, K., & Verikoukis, C. (2024). A multi-level deep rl-based network slicing and resource management for o-ran-based 6g cell-free networks. IEEE Transactions on Vehicular Technology, 73(11), 17472-17484.

[31] Ramegowda, A., Agarkhed, J., & Patil, S. R. (2020). Adaptive task scheduling method in multi-tenant cloud computing. International Journal of Information Technology, 12(4), 1093-1102.