Transactions on Computational and Scientific Methods | Vo. 5, No. 03, 2025
ISSN: 2998-8780
https://pspress.org/index.php/tcsm
Pinnacle Science Press

Learning-Based Intelligent Agents for Backend Resource Scheduling and Operational Decision Making

Lu Ren[1], Yixue Liu[2], Yuheng Zhao[3]
[1]University of California, Los Angeles, Los Angeles, USA
[2]Carnegie Mellon University, Pittsburgh, USA
[3]University of Vermont, Burlington, USA
*Corresponding author: Lu Ren; luren0304@gmail.com

Abstract: This study addresses large-scale backend service environments characterized by intense resource competition, frequent workload fluctuations, and long-term decision impact. It proposes a unified modeling and decision approach based on intelligent decision agents. Backend service systems are abstracted as continuously evolving decision environments. From the perspective of system operation mechanisms, the relationships among state representation, decision actions, and long-term feedback are jointly modeled. Resource scheduling and service management are therefore optimized within a unified framework. The proposed method does not rely on manually defined rules or static policy configurations. It learns decision patterns suitable for complex operational conditions through joint modeling of system states and decision feedback. Under unified data and evaluation settings, the method is systematically compared with several representative decision models. The results show more consistent advantages in resource utilization efficiency, service stability, and overall operational quality. The approach adapts more effectively to uncertainty caused by workload variation and service coupling in large-scale backend services. These findings indicate that intelligent decision mechanisms oriented toward long-term operational objectives can enhance overall coordination and efficiency of backend service systems. They provide a practical technical pathway for managing complex backend services.

Keywords: Intelligent decision-making; backend service management; resource scheduling; system autonomy

1. Introduction

With the widespread adoption of cloud computing, microservices, and service mesh technologies, modern backend service systems continue to evolve toward ultra-large scale, high concurrency, and strong dynamics. Online applications increasingly demand low latency, high availability, and stable service quality [1]. Backend systems must therefore make high-quality decisions under complex resource constraints, frequent workload fluctuations, and multi-service coexistence. Traditional rule-based or static policy-driven management relies heavily on manual experience and offline configuration. Such approaches struggle to remain effective in large-scale and rapidly changing environments. Under these conditions, intelligent decision methods that can autonomously perceive system states, understand operational patterns, and continuously improve decision behavior have become a key research direction [2].

Decision-making in large-scale backend services involves significant complexity and challenges. System states are high-dimensional, multi-source, and strongly time-dependent. Resource utilization, service invocation relationships, and performance indicators are coupled in nonlinear ways. Decision actions often have long-term

effects. Suboptimal short-term choices may trigger cascading performance degradation or stability risks later. The coexistence of multiple tenants and service dependency networks further complicates the problem. Optimization from a single local perspective can easily disrupt global system balance. These characteristics limit the effectiveness of traditional optimization methods and isolated intelligent models [3].

Agent-driven decision modeling offers a new perspective to address these issues. Backend service systems can be abstracted as interactive environments. Scheduling, resource allocation, and policy selection can be formulated as sequential decision processes. Agents can continuously perceive system feedback during operation [4]. They can adjust decision strategies based on accumulated interaction experience. This modeling paradigm focuses on learning the intrinsic relationships among states, actions, and feedback. It helps overcome the limitations of fixed rules and static models in complex scenarios. In large-scale systems, agent-based methods show strong potential for adaptability and scalability.

From an engineering and application perspective, intelligent decision algorithms for large-scale backend services can improve resource utilization and service performance stability. They can also reduce the cost of manual operation and policy tuning. As system scale grows, experience-driven configuration cannot cover all runtime scenarios. Agents with autonomous learning capability can continuously refine decision behavior in complex environments. This reduces reliance on manual intervention. It supports long-term online operation, rapid response to workload surges, and continuity of critical services. It also lays the foundation for more intelligent and automated backend service management systems.

For these reasons, systematic research on agent-based algorithms and decision mechanisms for large-scale backend services has strong theoretical and practical significance. This line of research deepens the understanding of operational dynamics and decision processes in complex backend systems. It advances intelligent decision theory in engineering contexts. At the same time, it provides methodological support for next-generation intelligent operations, elastic resource management, and service scheduling frameworks. Such progress promotes the evolution of backend service systems toward higher levels of adaptivity and autonomy. It aligns with current trends in cloud native computing and intelligent operations. It also has long-term implications for the stable and efficient management of future large-scale information systems.

2. BackGround

In large-scale backend service systems, the architecture is typically composed of a large number of heterogeneous service components [5]. These components operate collaboratively through complex invocation relationships and shared resources. The overall system behavior is not determined by a single module. It emerges from the combined effects of service topology, workload patterns, and scheduling strategies. As business scale expands and service granularity becomes finer, the system state space grows exponentially. The runtime process exhibits strong dynamics and uncertainty. As a result, backend service management has gradually shifted from static configuration and manual intervention to a complex decision process that requires continuous perception, analysis, and adjustment [6].

Most existing backend service decision mechanisms rely on predefined rules, heuristic algorithms, or models based on historical statistics. Their core idea is to identify relatively stable optimization strategies under limited scenarios. In real operational environments, however, service workloads, resource contention, and dependency structures change continuously over time. Fixed strategies often fail to adapt to new system states promptly. They may even introduce global risks while pursuing local optimization. Decision objectives are usually subject to multiple constraints and long-term effects. Typical examples include the tradeoff among performance, cost, and stability. These factors reveal clear limitations of traditional methods in terms of modeling capacity and adaptability [7].

Against this background, abstracting backend service operation and management as a continuous interactive decision process provides a theoretical foundation for more general and flexible decision models. By introducing agents with perception, decision-making, and feedback capabilities, the relationship between

system state evolution and decision behavior can be captured within a unified framework. Decision-making no longer depends on fixed rules. It can be continuously adjusted as the environment changes. Research on intelligent decision making under this perspective establishes essential conceptual foundations and problem settings for subsequent algorithm design and mechanism exploration in large-scale backend services.

3. Model Design

At the methodological level, the runtime environment of large-scale backend services is abstracted as a continuously evolving decision-making system, and formal state representations are used to characterize the overall operational characteristics of the system at different times. Let the state of the system at time t be represented as:

$$s_t = [u_t^{(1)}, \ldots, u_t^{(N)}]$$

Where $u_t^{(i)}$ represents the comprehensive operational characteristic vector of the i-th service or resource node at time t, encompassing information such as load, queue length, and resource usage. The evolution of the system's state over time can be represented as:

$$s_{t+1} = f(s_t, a_t)$$

Here, a_t represents the decision action taken by the agent in the state s_t, and function $f(\cdot)$describes the state transition mechanism of the system under the influence of the decision. This modeling approach allows the dynamic evolution of complex backend services to be characterized in a unified state space. This paper also presents the overall algorithm architecture, as shown in Figure 1.

Figure 1. Overall algorithm architecture

Building upon this, a policy function is introduced to describe the behavioral pattern of an agent in generating decisions given a state. Let the policy function be:

$$\pi(a_t|s_t) = softmax(g(s_t))$$

Where $g(\cdot)$ is a learnable mapping function used to map the system state to the action preference space. To characterize long-term performance, a cumulative reward function is introduced:

$$J = \sum_{t=0}^{T} \gamma^t r_t$$

Where r_t represents the system's immediate feedback to the current decision at time t, and $\gamma \in (0,1)$ is a discount factor used to balance short-term response and long-term stability. This objective function provides a clear optimization direction for the agent's decision-making, enabling it to form stable and consistent decision-making behavior in long-term operation.

To enhance the ability to model dependencies in complex systems, the state representation is further mapped to a latent representation space. Let the latent state representation be:

$$z_t = \phi(s_t)$$

Where $\phi(\cdot)$ is the feature extraction function, used to compress the state of a high-dimensional system while retaining key structural information. Based on the latent state, a state value function can be defined:

$$V(z_t) = E[J|z_t]$$

This function is used to evaluate the overall value of the current system state over long-term operation, thus providing a global perspective for decision-making. By explicitly modeling the relationship between potential states and long-term value, the instability problem in decision-making caused by high-dimensional state spaces can be effectively mitigated.

Ultimately, the strategy optimization process can be formulated as a problem of maximizing expected returns, and its optimization objective can be written as:

$$\max_{\pi} E_{\pi}[J]$$

During the optimization process, the agent continuously adjusts the policy function parameters to ensure that the decisions generated under different system states better align with the overall operational goals. This framework does not rely on fixed rules or human experience; instead, it achieves adaptive intelligent decision-making for large-scale backend services by modeling the relationship between system state evolution and decision feedback, providing a unified methodological foundation for long-term stable management in complex operating environments.

4. Implementation

4.1 Dataset

This study adopts the Google Cluster Data, which is the public trace dataset of the Borg cluster, as the experimental data source. The dataset targets large-scale data center clusters and provides long-term traces of jobs and tasks. It records event sequences such as task submission, scheduling, startup, and termination. It also includes information on resource requests and actual resource usage. The data are released in structured log and table formats. This allows the reconstruction of cluster operations along a unified timeline. The dataset is reproducible and publicly available as an open source resource. It is suitable as a fundamental dataset for research on intelligent decision-making in large-scale backend services.

From the perspective of intelligent decision modeling for backend services, this dataset directly supports the abstraction of state, action, and feedback. The state can be described by features such as node resource utilization, queue length, task priority, waiting time, and historical workload fluctuations. The action space can correspond to decisions including task placement, resource quota adjustment, migration or preemption policy selection, and triggering of rate limiting or elastic scaling. Feedback can be constructed from task completion latency, SLO violation risk, resource fragmentation, and fairness-related metrics. These elements together define optimization objectives oriented toward long-term system benefits. Due to its real-world system scale and high level of task concurrency, the dataset effectively reflects the challenges of multi-objective constraints and dynamic coupling in large-scale backend service decision-making.

In data processing, raw trace data can be aggregated into sequential samples using a fixed time granularity. Event streams can be used to reconstruct cluster snapshots at each time interval. This produces high-dimensional operational features for agent inputs. At the same time, scheduling and resource allocation behaviors can be mapped to discrete or parameterized action spaces. This ensures that the decision process forms a learnable and evaluable closed loop. To remain consistent with the research objective and paper theme, data usage emphasizes key factors such as large-scale operation, highly fluctuating workloads, multi-tenant competition, resource constraints, and service stability. This provides a unified and reproducible data foundation for agent-based intelligent decision algorithm research in large-scale backend services.

4.2 Experimental Setup

The hardware environment is fixed to a single machine with a single GPU. The CPU is an Intel Xeon Silver 4314 with 16 cores and 32 threads. The base frequency is 2.40 GHz. The system memory is 128 GB. The GPU is an NVIDIA RTX 4090 with 24 GB of memory. Both the system disk and the data disk are NVMe SSDs. The data disk capacity is 2 TB. The software environment includes Ubuntu 22.04 LTS, Python 3.10.13, PyTorch 2.1.2, CUDA 12.1, cuDNN 8.9, and NCCL 2.18. The dependency library versions are fixed to numpy 1.26.4, scikit learn 1.3.2, and pandas 2.1.4. To ensure reproducibility, the random seed is set to 2025. The cudnn deterministic option is set to True. The cudnn benchmark option is set to False. Training and evaluation are conducted in the same environment.

The hyperparameter configuration is as follows. The time granularity is 60 seconds. Continuous runtime logs are aggregated into sequential samples by time slices. The historical window length is 120 time slices. The prediction or decision update interval is one time slice. The state feature dimension is a 256-dimensional aggregated vector. The policy network consists of three fully connected layers. The hidden layer dimensions are 512, 256, and 128. The activation function is ReLU. The output is a discrete action space of size 16. The value network uses the same architecture as the policy network. The optimizer is AdamW. The learning rate is 0.0003. The weight decay is 0.0001. The gradient clipping threshold is 1.0. The training batch size is 256. The maximum number of training epochs is 80. The early stopping patience is 10. Reinforcement learning parameters include a discount factor of 0.99. The replay buffer capacity is 1000000. The minimum replay warm-up steps are 50000. The target network soft update coefficient is 0.005. The exploration rate starts at 1.0 and decays to 0.05. Linear annealing is applied over 300000 steps. Data splitting follows a chronological order. The training set accounts for 70 percent. The validation set accounts for 10 percent. The test set accounts for 20 percent. All normalization statistics are computed only on the training set and applied to the validation and test sets.

5. Experimental Results and Analysis

This paper first presents the experimental results compared with other models, as shown in Table 1.

Table 1. Comparative experimental results

Method	Resource Utilization Rate	SLO Violation Ratio	Long-Term Cumulative Reward	Average Service Latency
Aios [8]	0.61	0.143	182.4	214 ms
Cheatagent [9]	0.64	0.131	191.7	208 ms
Agentlite [10]	0.67	0.118	203.5	199 ms
Agentsquare [11]	0.69	0.110	215.2	193 ms
AgentDiagnose [12]	0.72	0.098	228.6	187 ms
Ours	0.78	0.072	264.1	171 ms

From the overall trend, the baseline methods show a clear tradeoff between resource efficiency and service stability. The proposed intelligent decision approach achieves more consistent improvements across both objectives. This indicates that the learned decision strategy can better activate cluster resources while maintaining more controllable service risk under high concurrency and dynamic fluctuations. Such a joint improvement better matches the practical requirements of large-scale backend services. It enhances resource utilization and operational efficiency without sacrificing reliability. It also supports more robust autonomous decision-making for long-term online systems.

On the resource utilization side, the proposed method demonstrates stronger global coordination capability. It reduces waste caused by resource fragmentation and local congestion. Resource allocation becomes more aligned with real-time workload demands. This suggests that the model captures system states more effectively.

It can form finer-grained scheduling preferences and more reasonable quota adjustments under multi-service competition and resource constraints. As a result, overall resource utilization is improved. This property has direct value for elastic scaling and capacity planning in cloud native environments. It helps reduce redundant resource overhead and improve the effective conversion of system throughput.

In terms of stability and service quality, the proposed method shows stronger risk suppression under service level constraints. It reduces the occurrence of violations and improves to end latency experience. Compared with strategies that focus only on short-term response, this outcome indicates that the decision process accounts for long-term effects and cross-service coupling. By jointly modeling load migration, queue backlog, and service dependency propagation, it mitigates decision oscillation and limits the amplification of local extremes on overall service behavior. Therefore, the method is more suitable for backend service systems that require continuous online operation. It can handle bursty traffic and dynamic resource changes more stably.

From the perspective of long-term optimization, the proposed method achieves a higher cumulative return. This implies a more consistent direction of policy updates in sequential decision making. It establishes a more balanced tradeoff among performance, stability, and resource efficiency. This result aligns well with the research objective of intelligent decision agents in this study. It reflects the ability to learn system evolution patterns and continuously improve decision behavior in complex backend environments. Overall, the improvement toward long-term benefits provides support for scalable automated operations and adaptive scheduling frameworks. It facilitates the transition of large-scale backend services from experience-driven management to learning driven autonomy.

The discount factor is used to characterize the trade-off preferences of intelligent decision-making over time, influencing the strategy's emphasis on immediate feedback versus long-term benefits. For autonomous decision-making in the context of large-scale backend services, changes in the discount factor further alter the stability of strategy updates and their response to operational fluctuations. Therefore, it is necessary to examine the impact of different discount factor values on key behavioral performance under uniform settings to understand the strategy's trade-offs between long-term perspectives and short-term constraints. The experimental results are shown in Figure 2.

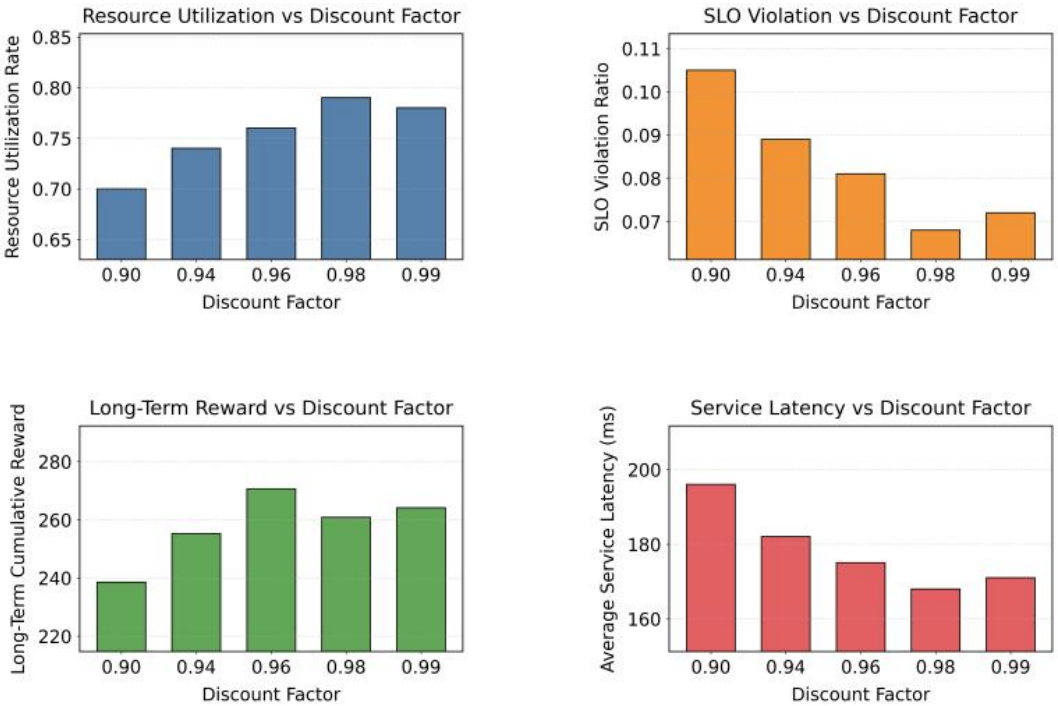

Figure 2. The effect of the discount factor on experimental results

The overall trend induced by changes in the discount factor reveals a clear balance between short-term response and long-term planning in decision policies. Lower discount settings emphasize immediate feedback. The resulting strategies focus on quickly alleviating current load and performance pressure. As the discount weight increases, decision behavior gradually reflects greater attention to long-term system operation. This observation aligns well with the requirement of large-scale backend services to maintain stability and forward-looking scheduling during continuous operation. It indicates that the discount factor plays a critical role in shaping the decision style of the agent.

At the level of resource management, adjusting the discount factor directly influences how much the policy values continuity in resource allocation. When decisions adopt a more long-term perspective, resource utilization becomes more stable. This reflects a more comprehensive characterization of inter-service competition and workload evolution trends. It suggests that the proposed intelligent decision method can reduce frequent local adjustments and resource oscillations through appropriate temporal tradeoffs. As a result, more coordinated global resource scheduling behavior emerges in complex backend environments. This contributes to improved overall system efficiency.

From the perspective of service quality constraints, variations in the discount factor are also reflected in the policy attitude toward risk control. Settings that emphasize long-term return help suppress unnecessary aggressive decisions. The policy becomes more cautious when facing workload fluctuations. System performance under service level objectives is therefore improved. This gain in stability does not stem from conservative behavior at a single moment. It arises from modeling the cumulative effects over the operational process. The decision process becomes more consistent with the reliability requirements of continuously online services.

By jointly examining long-term benefits and performance changes, it can be observed that the discount factor within an appropriate range guides the agent toward more consistent decision patterns. It enables a reasonable balance among resource efficiency, service stability, and response performance. This result further validates the effectiveness of the intelligent decision modeling approach adopted in this work for capturing long-term evolution in large-scale backend services. It also shows that a proper temporal weighting design can enhance the adaptability and practical value of intelligent decision agents in complex operational environments.

The learning rate determines the update pace of the policy and value function during training, directly affecting optimization stability and the speed of convergence to an effective decision pattern. For intelligent decision-making tasks targeting large-scale backend services, differences in the learning rate can alter the model's adaptation rhythm to dynamic environmental perturbations, thus affecting the final policy characteristics. The experimental results are shown in Figure 3.

The overall trend induced by learning rate variation shows a clear tradeoff between the magnitude of policy updates and decision stability. A smaller learning rate leads to more conservative updates. Decision behavior remains relatively smooth. The response to environmental changes becomes slower. A larger learning rate accelerates policy updates. It also amplifies the influence of environmental noise on the policy. This pattern highlights the critical role of the learning rate in regulating the adaptation pace of intelligent decision processes. It is particularly consistent with the requirement of large-scale backend services to balance stable evolution and continuous optimization.

In terms of resource management behavior, different learning rate settings significantly alter how the policy represents resource allocation relationships. A moderate update step helps the model form more coordinated scheduling preferences under multi-service competition and resource constraints. Resource utilization becomes more balanced. In contrast, overly aggressive or overly conservative updates weaken the ability to capture the global workload structure. This reduces the consistency and effectiveness of resource allocation during long-term operation.

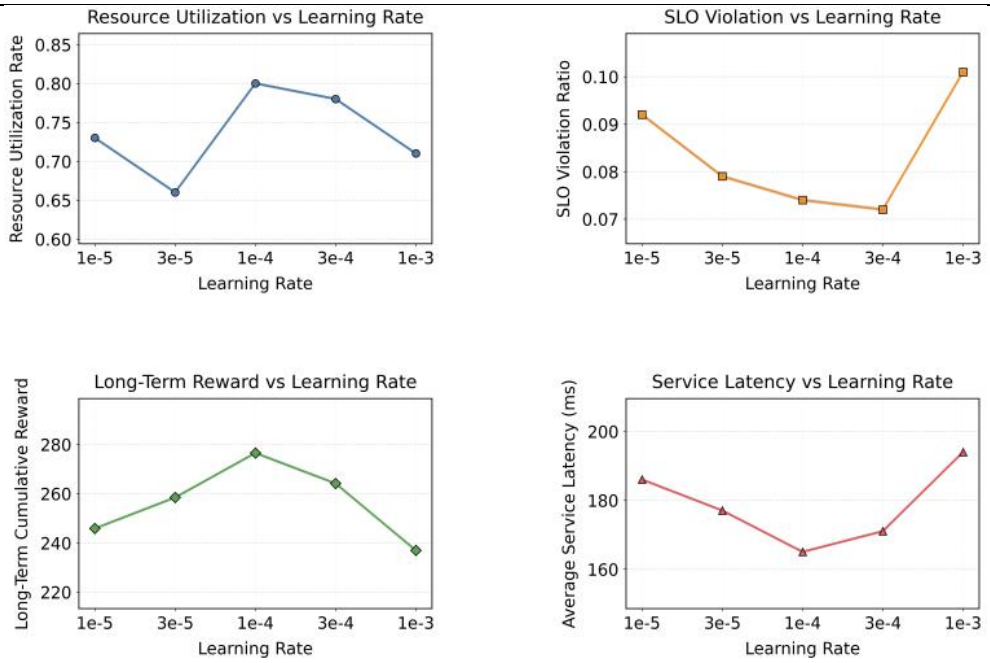

Figure 3. The impact of the learning rate on experimental results

From the perspective of service stability and quality constraints, the learning rate also affects policy sensitivity to risk. When updates are too rapid, decisions tend to change sharply under local state fluctuations. This increases operational uncertainty. With an appropriate update pace, the policy can better balance immediate feedback and historical experience. Decision behavior becomes smoother when facing workload variation. Such stability improvement is especially important for continuously online backend service systems. It helps reduce oscillations and uncontrollable behavior during operation.

By jointly examining long-term benefit and performance response, it can be observed that the learning rate within a suitable range guides the agent toward more consistent and sustainable decision patterns. A reasonable learning rate allows sufficient learning capacity without disrupting established, effective decision structures. The model, therefore, maintains good adaptability in complex dynamic environments. This further indicates that in intelligent decision research for large-scale backend services, the learning rate is not only a training hyperparameter. It is also a key factor influencing the long-term evolution quality of decision policies.

The decision update cycle determines the time granularity at which the agent adjusts its scheduling strategy, thus affecting its response rhythm to fluctuations in backend service load and resource contention. The experimental results are shown in Figure 4.

The trend observed from varying the decision update interval indicates that the adjustment frequency of the agent plays a critical role in resource utilization efficiency. A shorter update interval allows the policy to perceive workload changes more quickly and respond promptly. However, frequent adjustments may introduce additional scheduling overhead. When the update interval is too long, the policy response to environmental changes becomes noticeably delayed. It becomes difficult to capture the evolution of resource competition. This behavior highlights the importance of decision time granularity in intelligent scheduling for large-scale backend services.

Within a moderate range of update intervals, resource utilization performance is relatively better. This suggests that the policy can remain sensitive to system dynamics without disrupting existing resource allocation structures through overly frequent corrections. It indicates that the proposed intelligent decision method can more effectively integrate historical state information with current operational features at an appropriate time

scale. As a result, more stable resource scheduling behavior is achieved. Such stability is particularly important in backend environments with concurrent services and continuously fluctuating workloads.

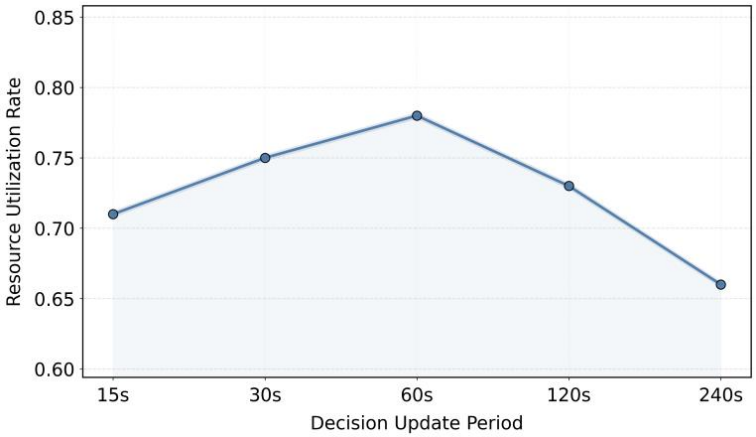

Figure 4. Sensitivity experiment of the decision update cycle to resource utilization

As the update interval is further extended, resource utilization efficiency shows a clear decline. This reflects the negative impact of delayed policy updates on system performance. Under these conditions, the agent cannot adjust resource allocation in time to adapt to load migration and changing service demands. Idle resources and local congestion, therefore, coexist. This result demonstrates that relying on long, unchanged decision policies alone cannot satisfy the elasticity and adaptability requirements of large-scale backend services.

Overall, the sensitivity analysis of the decision update interval confirms the central role of time granularity design in intelligent decision frameworks. A reasonable update interval establishes an effective balance between response speed and system stability. It enables the agent to maintain high resource utilization in complex operational environments. This conclusion is highly consistent with the research objective of this work on large-scale backend services. It further shows that fine-grained modeling of decision pacing can significantly improve the practicality and reliability of intelligent decision methods in real system scenarios.

6. Conclusion

This work focuses on decision processes in large-scale backend services that are highly dynamic, strongly coupled, and characterized by long-term effects. It systematically investigates modeling ideas and methodological frameworks based on intelligent decision agents. By abstracting backend service operation as a continuously evolving decision environment, the proposed paradigm jointly represents system states, decision actions, and long-term feedback. It overcomes the limitations of traditional static rules and local optimization methods in complex scenarios. It offers a new perspective for autonomous decision-making oriented toward global system objectives. Comparative results demonstrate more consistent advantages in resource efficiency, service stability, and overall operational quality. These findings validate the potential of intelligent decision approaches in complex backend systems.

From the perspective of practical value, this study provides a feasible technical pathway for automated management of cloud platforms, microservice architectures, and large-scale online systems. By introducing decision mechanisms that can continuously learn system behavior, backend service management no longer relies on experience-driven policy configuration. It can adapt and optimize during real operation. This capability is essential for supporting high-concurrency workloads, handling bursty demand, and ensuring continuity of critical services. It also helps reduce operational costs and improve the predictability and controllability of system behavior.

In a broader application context, the proposed intelligent decision framework is not limited to resource scheduling and service management. It also offers a general modeling approach for autonomous control and

cooperative optimization in complex systems. As backend services continue to scale and system structures become more complex, intelligent methods with long-term decision capability and a global perspective will be increasingly important. The results indicate that decision models grounded in system operation mechanisms can effectively mitigate conflicts between local optima and global objectives. This provides a foundation for the stable operation of complex information systems.

Looking ahead, intelligent decision methods for backend services have substantial room for further development as computing infrastructure, operational environments, and business forms continue to evolve. Incorporating richer state perception, more detailed modeling of service dependencies, and more efficient online learning mechanisms can further enhance robustness and generalization. Deeper integration of intelligent decision-making with automated operations and autonomous system governance will promote higher levels of adaptivity and self-management. Overall, this study offers a constructive exploration of next-generation intelligent backend service management. It has long-term implications for the advancement of related application domains.

References

[1] H. Qiu, W. Mao, C. Wang et al., "AWARE: Automate Workload Autoscaling With Reinforcement Learning in Production Cloud Systems," Proceedings of the 2023 USENIX Annual Technical Conference (USENIX ATC 23), pp. 387-402, 2023.

[2] S. Li, "Adaptive Scheduling for Multi-Model Collaborative Distributed Inference Under Resource Heterogeneity and Dynamic Workloads," 2024.

[3] J. Santos, E. Reppas, T. Wauters et al., "Gwydion: Efficient Auto-Scaling for Complex Containerized Applications in Kubernetes Through Reinforcement Learning," Journal of Network and Computer Applications, vol. 234, Art. no. 104067, 2025.

[4] Y. Garí, D. A. Monge, E. Pacini et al., "Reinforcement Learning-Based Application Autoscaling in the Cloud: A Survey," Engineering Applications of Artificial Intelligence, vol. 102, Art. no. 104288, 2021.

[5] A. Kallel, M. Rekik and M. Khemakhem, "DRL4HFC: Deep Reinforcement Learning for Container-Based Scheduling in Hybrid Fog/Cloud System," Proceedings of the International Conference on Agents and Artificial Intelligence (ICAART), pp. 231-242, 2024.

[6] J. Qiu, "Learning Collaborative and Robust Scheduling Policies for Microservice Backends Under Uncertainty," 2024.

[7] X. Wei, J. Zhang, H. Li et al., "Agent.XPU: Efficient Scheduling of Agentic LLM Workloads on Heterogeneous SoC," arXiv preprint arXiv:2506.24045, 2025.

[8] K. Mei, X. Zhu, W. Xu et al., "AIOS: LLM Agent Operating System," arXiv preprint arXiv:2403.16971, 2024.

[9] L. Ning, S. Wang, W. Fan et al., "CheatAgent: Attacking LLM-Empowered Recommender Systems via LLM Agent," Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 2284-2295, 2024.

[10] Z. Liu, W. Yao, J. Zhang et al., "AgentLite: A Lightweight Library for Building and Advancing Task-Oriented LLM Agent System," arXiv preprint arXiv:2402.15538, 2024.

[11] Y. Shang, Y. Li, K. Zhao et al., "AgentSquare: Automatic LLM Agent Search in Modular Design Space," arXiv preprint arXiv:2410.06153, 2024.

[12] T. Ou, W. Guo, A. Gandhi et al., "AgentDiagnose: An Open Toolkit for Diagnosing LLM Agent Trajectories," Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 207-215, 2025.