Artificial Intelligence and Computing Innovations

Intelligence Science Publishing

https://www.ispublishing.com

ISSN: 3070-4944

Vo.4, No.1, 2024

Constraint-Aware Resource Matching for Virtual Machine Placement in Cloud Environments

Junnan Kou[1]*, Eric Pei[2]

[1]University of Washington, Seattle, USA

[2]University of Missouri, Columbia, USA

**Corresponding author: koujunnan1996@gmail.com*

Abstract: With the continuous expansion of cloud platform scale and the increasing complexity of business workloads, resource matching and placement in virtual machine arrival flow scenarios have become a critical factor affecting the operational efficiency and service quality of cloud infrastructure. Addressing the shortcomings of existing methods, such as insufficient understanding of dynamic requests, inadequate characterization of node states, and limited ability to integrate constraints, this paper proposes a unified modeling method for placement recommendation centered on virtual machine resource allocation tasks. This method focuses on the adaptation relationship between virtual machine requests and candidate nodes, incorporating request priority information, node dynamic state information, and deployment constraint information into the resource placement decision process, constructing a recommendation framework consisting of request encoding, node encoding, feature interaction, constraint modeling, and ranking optimization. During data processing, continuous arrival flow samples are constructed based on publicly available cloud platform virtual machine trajectory data, and a supervision signal is generated by combining resource demand, node remaining capacity, and deployment feasibility relationships, enabling the model to learn effective matching patterns between requests and nodes. The results show that the proposed method can better improve the ranking quality of candidate nodes and the ability to identify target nodes, enhancing the model's ability to express dynamic resource relationships in complex cloud environments while ensuring placement feasibility. Overall, this method provides an effective solution for intelligent resource scheduling in virtual machine arrival flow scenarios, taking into account task importance, node carrying capacity, and actual constraints.

Keywords: Cloud resource allocation; virtual machine placement; node sorting; constraint modeling

1. Introduction

With the continuous expansion of cloud computing infrastructure, resource scheduling and service provisioning for large-scale data centers have become a crucial research area in intelligent computing. Against the backdrop of multi-tenant sharing, increasing business concurrency, and deepening service heterogeneity, virtual machines, as the core abstract units of cloud platforms carrying computing tasks, typically exhibit significant dynamism, burstiness, and uncertainty in their arrival process. Different VM requests differ significantly in terms of computing power, storage, bandwidth, latency sensitivity, and service level, making traditional resource allocation methods relying on static rules or local heuristics increasingly inadequate for real-time decision-making in complex operating environments [1, 2]. How to efficiently, accurately, and globally coordinate resource matching and placement for arriving VMs under constantly changing resource supply and demand has become a key issue affecting cloud platform performance, cost, and service quality [3].

From a system operation mechanism perspective, VM placement is not a simple node allocation process, but a complex optimization task involving request feature identification, node load awareness, resource fragmentation control, conflict avoidance, and long-term benefit balancing [4]. On the one hand, the differences in priority, resource consumption intensity, lifecycle, and business importance of virtual machine requests dictate that the platform cannot rely solely on immediate idle resources for coarse-grained judgments during scheduling; it must comprehensively consider the service value and potential impact of the request itself [5]. On the other hand, the physical node status exhibits significant time-varying characteristics; its resource availability, load level, historical placement results, and future carrying capacity all influence subsequent scheduling feasibility. Ignoring the deep coupling between request-side and node-side information can easily lead to problems such as obstructed high-priority task responses, unbalanced resource utilization, and local node overload, thereby weakening the overall operational efficiency and service stability of the cloud environment.

In this context, introducing the resource matching and placement problem into a recommendation learning perspective has significant theoretical and methodological innovation value. Recommendation modeling can mine the adaptation relationship between requests and resources from the candidate node set, and by jointly characterizing the virtual machine demand features and node status representations, it provides stronger expressive power for decision generation in complex scheduling scenarios. Especially in scenarios where virtual machine arrival flows evolve continuously, priority-based request understanding mechanisms help strengthen the identification of critical business needs, while node state-based resource cognition mechanisms improve the system's response accuracy to dynamic environments. Constraint-aware recommendation learning, on the other hand, can improve the executability and robustness of decision-making results under multiple constraints such as capacity, performance isolation, deployment rules, and service quality [6]. This research approach breaks through the limitations of traditional scheduling methods that separate matching and constraint processing, providing a more systematic and intelligent modeling framework for cloud computing resource allocation problems.

Furthermore, research on resource matching and placement recommendations for virtual machine arrival flows not only helps improve cloud platform resource utilization and reduce energy consumption and operational costs, but also has practical significance for enhancing service continuity, ensuring the performance of high-priority businesses, and promoting the construction of intelligent resource management systems [7]. With the development of cloud-edge collaboration, elastic computing, intelligent operation and maintenance, and large-scale online services, the types of tasks in the future cloud environment will become more diverse, resource competition will become more complex, and the requirements for real-time performance, robustness, and interpretability in scheduling decisions will continue to increase. Therefore, conducting systematic research on request priority modeling, node state representation, and constraint-aware recommendation learning can not only provide theoretical support

for refined resource management in cloud computing environments but also lay a solid foundation for building intelligent placement mechanisms and adaptive scheduling methods for complex dynamic scenarios, thus possessing outstanding academic value and broad application prospects.

2. Datasets and Dataset Preprocessing

2.1 Dataset

This paper selects Microsoft Azure VM Trace as the research dataset. This dataset, publicly released by Microsoft in the Azure Public Dataset repository, is real-world cloud platform virtual machine trace data open to academic research. The records originate from representative virtual machine workloads in a specific region of an Azure production environment. The public description indicates that the repository provides various types of Azure runtime traces to the research community, with Microsoft Azure VM trace explicitly described as a publicly released large-scale Azure virtual machine workload trace, which can be used to analyze virtual machine lifecycle, arrival behavior, and resource management issues. Therefore, this dataset provides strong support in terms of data source credibility, open accessibility, and the realism of the cloud computing scenario.

This dataset shows high consistency with research on resource matching and placement recommendations for virtual machine arrival flows in cloud computing environments. Its core data revolves around the continuous arrival, operation, and resource consumption of virtual machine requests in the cloud platform, reflecting the dynamic changes of virtual machines over different time periods. It also provides fundamental support for request-side modeling, node-side state awareness, and subsequent placement decision analysis. Compared to cluster trajectory data, which is mainly based on batch processing tasks or container jobs, Azure VM Trace is directly geared towards virtual machine workloads. Therefore, it is more suitable for characterizing the evolution of resource requirements, node carrying relationships, and placement adaptation issues in virtual machine arrival flow scenarios, and can better serve the research topic and modeling goals of this paper.

2.2 Label Construction

In the data preprocessing stage, virtual machine request records are first sorted according to timestamps, and continuous arrival stream samples are constructed using fixed time windows. Then, request-side features and node-side features are generated by combining request priority, normalized resource requirements corresponding to virtual machine type, and the remaining capacity status of candidate nodes. Since Microsoft Azure VM Trace, as public cloud trajectory data, does not directly provide supervised labels for resource matching and placement recommendation tasks, this paper adopts a weakly supervised label construction strategy oriented towards placement feasibility: For each arriving virtual machine request, the current set of candidate nodes is traversed. If a node meets the normalized resource requirements of the request in key resource dimensions such as CPU and memory, and placement does not violate capacity constraints and basic deployment constraints, then the paired sample of the request and that node is marked as a positive sample, with a label of 1. If a node cannot meet the requirements in any key resource dimension, or placement would lead to resource overruns, feasibility failure, or significant load conflicts, then the paired sample is marked as a negative sample, with a label of 0. Furthermore, to enhance the label's ability to distinguish placement quality, nodes with higher resource matching, more reasonable remaining capacity, and better ability to meet the needs of high-priority requests can be prioritized among all positive samples as better target nodes, thus forming a binary classification supervision signal that can be used for constraint-aware recommendation learning. This construction method is entirely based on the requested information and resource constraint relationship in open-source trajectories and is consistent with the positioning of this type of trajectory in public repositories for studying packing algorithms. The completed sample dataset is shown in Table 1.

Table 1. Example of a completed sample dataset

Time Window	Request ID	Priority	Lifecycle	CPU Demand	Memory Demand	Candidate Node ID	Residual CPU	Residual Memory	Capacity Satisfied	Deployment Constraint Satisfied	Match Label
W1	VM_001	3	12	0.25	0.30	Node_07	0.42	0.55	Yes	Yes	1
W1	VM_001	3	12	0.25	0.30	Node_12	0.18	0.27	No	Yes	0
W1	VM_002	1	6	0.10	0.12	Node_03	0.35	0.40	Yes	Yes	1
W1	VM_002	1	6	0.10	0.12	Node_09	0.08	0.11	No	Yes	0
W2	VM_003	2	18	0.32	0.28	Node_05	0.38	0.31	Yes	Yes	1
W2	VM_003	2	18	0.32	0.28	Node_14	0.36	0.20	No	Yes	0
W2	VM_004	4	24	0.40	0.45	Node_02	0.51	0.60	Yes	Yes	1

3. Methodology

In a dynamic cloud environment, virtual machine placement should not be treated as a static bin allocation problem because each arriving request is coupled with priority preference, heterogeneous resource demand, and future scheduling influence. To capture this evolving interaction, the method represents every arriving request together with the contemporaneous candidate node set, so that matching is learned as a context-dependent recommendation task rather than a one-shot feasibility check. The overall model architecture is shown in Figure 1.

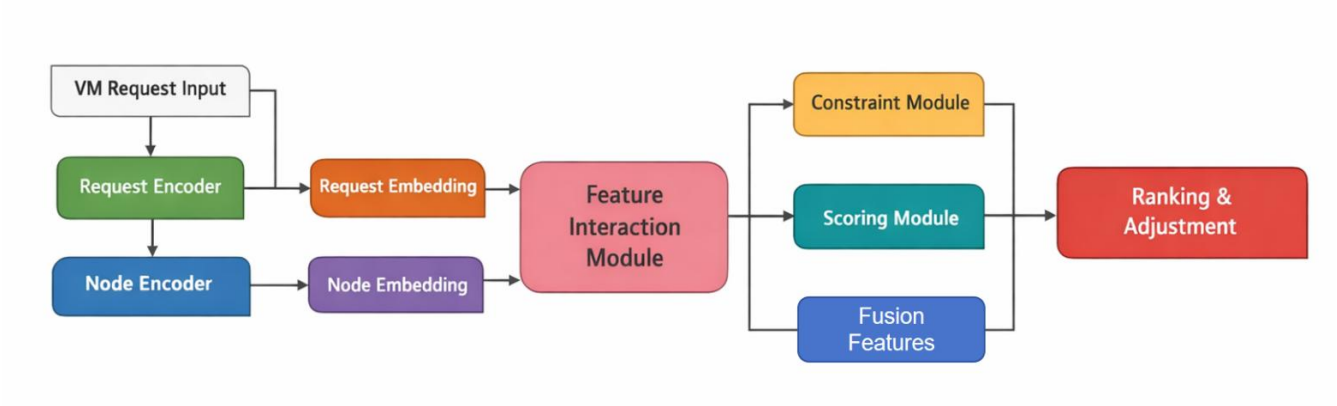

Figure 1. Overall model architecture

Given the request arriving at time step t, its raw profile is denoted by $\boldsymbol{q}_t = [\boldsymbol{d}_t, p_t, \tau_t]$, where $\boldsymbol{d}_t$ describes normalized multi-resource demand, p_t denotes request priority, and τ_t reflects temporal descriptors such as arrival interval or lifespan proxy embedded in the trace. A nonlinear request encoder is then introduced to project heterogeneous attributes into a compact semantic space, which improves the comparability between requests with different scales and service roles:

$$\boldsymbol{r}_t = \phi_r(\boldsymbol{q}_t) \tag{1}$$

Rather than relying only on remaining capacity, the state of node j at time t is characterized by both instantaneous availability and structural load condition, because a node that is currently feasible may still be unsuitable under future pressure propagation. For this reason, the node representation $\boldsymbol{n}_j^t$ integrates residual resource vector $\boldsymbol{a}_j^t$, utilization vector $\boldsymbol{u}_j^t$, and historical load descriptor $\boldsymbol{g}_j^t$, after which a node encoder maps them into a latent state embedding that is more suitable for fine-grained ranking:

$$\boldsymbol{h}_j^t = \phi_n(\boldsymbol{n}_j^t) = \phi_n([\boldsymbol{a}_j^t, \boldsymbol{u}_j^t, \boldsymbol{g}_j^t]) \tag{2}$$

Such a dual stream representation is meaningful because it allows the model to distinguish whether a request should be placed on a lightly loaded node for stability or on a partially utilized node for consolidation efficiency, which is essential for resource matching under continuous arrival flow.

To model compatibility, the interaction between the request embedding $\boldsymbol{r}_t$ and node embedding $\boldsymbol{h}_j^t$ is not reduced to a simple inner product, since resource placement quality depends on both agreement and mismatch across multiple dimensions. A richer cross-feature tensor is therefore constructed by concatenating direct features, element-wise alignment, and difference patterns, allowing the scoring network to perceive complementarity and conflict simultaneously:

$$\boldsymbol{z}_{tj} = \left[\boldsymbol{r}_t \| \boldsymbol{h}_j^t \| \left(\boldsymbol{r}_t \odot \boldsymbol{h}_j^t\right) \| \left(\boldsymbol{r}_t - \boldsymbol{h}_j^t\right)\right] \tag{3}$$

Subsequently, a learnable matching function produces the raw relevance score s_{tj}, where $\boldsymbol{W}_z$, $\boldsymbol{w}_s$, and $\boldsymbol{b}_z$ are trainable parameters and $\sigma(\cdot)$ denotes a nonlinear activation:

$$s_{tj} = \boldsymbol{w}_s^{\top} \sigma\left(\boldsymbol{W}_z \boldsymbol{z}_{tj} + \boldsymbol{b}_z\right) \tag{4}$$

Unlike conventional heuristic placement rules, this formulation evaluates how well the service demand pattern implied by the incoming virtual machine agrees with the current operating condition of each candidate node, thereby transforming placement from threshold screening into adaptive relevance estimation. Another advantage of this design is that high-priority requests can be represented in the same semantic space as ordinary requests while still preserving their stronger sensitivity to node quality through the encoded feature interaction.

Since an apparently high score may still correspond to an infeasible or risky placement, the method further injects explicit deployment constraints into the recommendation layer so that ranking remains executable in practice. Capacity violation is first measured by comparing request demand against node residual resources over all resource dimensions, and the resulting penalty becomes zero only when the candidate node can safely absorb the request without exceeding its available budget:

$$c_{tj}^{cap} = \sum_{m=1}^{M} max\left(0, d_{tm} - a_{jm}^t\right) \tag{5}$$

Besides pure capacity, cloud placement is often restricted by affinity, isolation, policy, or service locality requirements, so an auxiliary constraint indicator $\gamma_{tj} \in \{0,1\}$ is introduced to distinguish admissible and inadmissible request node pairs. The final penalty combines hard rule violation and soft resource mismatch, where η controls the cost of invalid policy assignment:

$$c_{tj} = c_{tj}^{cap} + \eta\left(1 - \gamma_{tj}\right) \tag{6}$$

By integrating this term into ranking, the method avoids the common problem that a recommender may prefer a node with strong statistical affinity but poor operational feasibility. Equally important, the explicit penalty path makes the decision process more interpretable because a low-ranked candidate can be traced either to a semantic mismatch or to a concrete constraint conflict, which is valuable for cloud-side scheduling analysis.

Beyond feasibility, the placement policy should also reflect differentiated service importance, because resource competition under bursty arrival streams requires high-priority requests to receive more robust placement opportunities. The constraint-aware recommendation score is therefore adjusted by combining raw relevance, constraint penalty, and priority-sensitive utility, where λ balances the strength of constraint suppression, and β governs the contribution of utilization-aware priority bias:

$$\hat{s}_{tj} = s_{tj} - \lambda c_{tj} + \beta p_t\left(1 - \bar{u}_j^t\right) \tag{7}$$

Here, $\bar{u}_j^t$ denotes the aggregated utilization level of node j at time t, so the last term naturally encourages important requests to be matched with nodes that preserve sufficient safety margin. A normalized recommendation probability is then produced over the candidate set C_t, enabling the model to learn relative placement preference instead of isolated binary judgment:

$$P(j|t) = \frac{exp(\hat{s}_{tj})}{\sum_{k \in \mathcal{C}_t} exp\,(\hat{s}_{tk})} \tag{8}$$

Finally, supervision is imposed through a placement label $y_{tj} \in \{0,1\}$ derived from the constructed feasible pairing relation, and the optimization objective minimizes cross entropy over all request candidate pairs while regularizing parameters Θ to improve generalization:

$$\mathcal{L} = -\sum_t \sum_{j \in \mathcal{C}_t} \left[y_{tj} log P(j|t) + (1 - y_{tj}) log\left(1 - P(j|t)\right) \right] + \mu \parallel \Theta \parallel_2^2 \tag{9}$$

Through this objective, the method simultaneously learns semantic matching, respects operational constraints, and preserves service differentiation, which gives the overall framework clear significance for intelligent resource placement in continuous virtual machine arrival scenarios.

4. Experimental Results and Analysis

To systematically examine the relative performance of the proposed method in virtual machine resource matching and placement recommendation tasks in cloud computing environments, representative studies related to virtual machine allocation, placement optimization, constraint-aware scheduling, and learning-driven resource management were selected as comparative objects. These studies cover different technical paths, including rule-driven allocation, reinforcement learning placement, multi-objective optimization, lifecycle-aware allocation, security-aware placement, and planned virtual machine request placement. They provide a relatively comprehensive horizontal reference in terms of request modeling capabilities, node state utilization capabilities, constraint handling capabilities, and ranking decision-making capabilities. The experimental results are shown in Table 2.

Table 2. Experimental results compared with other models

Method	HR@10	NDCG@10	MRR	AUC
Hadary et al. [8]	0.79	0.72	0.68	0.83
Barbalho et al. [9]	0.82	0.75	0.71	0.86
Caviglione et al. [10]	0.80	0.73	0.69	0.84
Saxena et al. [11]	0.77	0.70	0.66	0.82
Long et al. [12]	0.83	0.76	0.72	0.87
Rjeib et al. [13]	0.81	0.74	0.70	0.85
Koubaa et al. [14]	0.84	0.77	0.73	0.88
Ours	0.89	0.83	0.79	0.92

The results show that the proposed algorithm exhibits superior overall performance in resource matching and placement recommendation tasks, maintaining strong advantages in candidate node ranking quality, target node identification accuracy, and overall discriminative ability. This indicates that the constructed method can more effectively characterize the adaptation relationship between virtual machine request features and node states, while demonstrating good modeling capabilities in constraint integration, priority information utilization, and placement decision optimization. Furthermore, this method not only improves the effectiveness of recommendation results but also enhances the model's ability to express dynamic resource relationships in complex cloud environments, thus providing more reliable technical support for intelligent resource allocation based on virtual machine arrival flows.

To further verify the role of each key component in the overall resource matching and placement recommendation performance, it is necessary to conduct targeted analysis around request-side modeling, node-side state modeling, and constraint awareness mechanisms. As shown in Table 3, this paper sets up ablation schemes according to the core modules in the method design, and examines the impact of different components on the ranking quality and discriminative ability of candidate nodes under a unified evaluation

standard, thereby more clearly illustrating the collaborative relationship between the modules in the overall framework.

Table 3. Ablation study results of the proposed recommendation framework

Ablation settings	HR@10	NDCG@10	MRR	AUC
w/o Request Priority Modeling	0.84	0.77	0.73	0.88
w/o Node State Representation	0.85	0.78	0.74	0.89
w/o Constraint-aware Recommendation	0.86	0.79	0.75	0.90
Ours	0.89	0.83	0.79	0.92

The ablation experiments further demonstrate that the proposed recommendation framework achieves satisfactory performance in virtual machine resource matching and placement tasks, and the overall method design is highly effective and complete. These results show that request priority modeling, node state representation, and constraint-aware recommendation learning together constitute the key components supporting the model's performance improvement. This enables the model to more accurately understand the importance of virtual machine requests, more fully characterize the dynamic carrying capacity of candidate nodes, and generate higher-quality recommendation results while meeting actual deployment constraints. Therefore, the proposed method not only demonstrates good recommendation accuracy but also reflects resource scheduling adaptability and decision-making rationality in complex cloud computing environments.

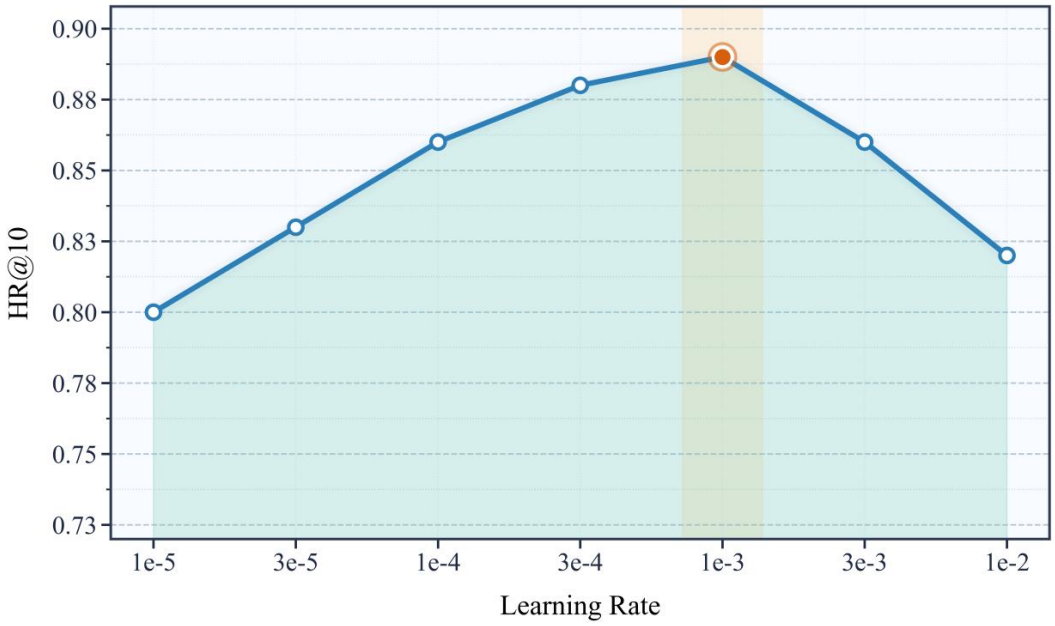

Figure 2. The impact of the learning rate on the experimental results of HR@10

Figure 2 demonstrate that the proposed algorithm maintains good recommendation performance and stable ranking ability under the current learning rate setting, indicating that the constructed resource matching and placement recommendation framework has strong parameter adaptability. This result further reflects that request priority modeling, node state representation, and constraint-aware recommendation learning form a relatively coordinated synergy, enabling the model to effectively capture the matching relationship between requests and nodes in virtual machine arrival flow scenarios and generate high-quality placement decisions. Overall, the proposed method performs reliably in the learning process, demonstrating good optimization feasibility and modeling effectiveness for complex cloud computing environments. The effect of the optimizer on the HR@10 experimental results is further shown in Figure 3.

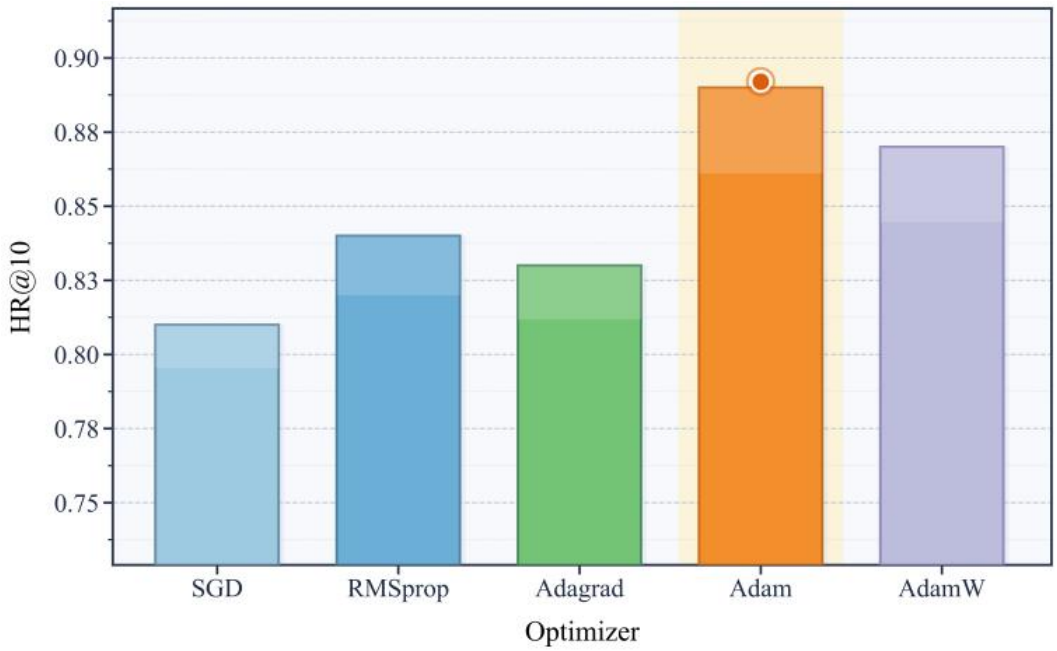

Figure 3. The effect of the optimizer on the experimental results of HR@10

This result further reflects the good synergy between request priority modeling, node state representation, and constraint-aware recommendation learning, enabling the model to more fully capture the matching relationship between virtual machine requests and candidate nodes, and thereby generate high-quality placement decisions. Overall, the proposed method exhibits good robustness and optimizability during the training phase, demonstrating resource matching and placement recommendation capabilities for complex cloud computing environments.

5. Conclusion

This paper focuses on the resource matching and placement recommendation problem in virtual machine arrival flow scenarios within a cloud computing environment. Addressing the shortcomings of traditional resource allocation methods—insufficient adaptability, coarse decision granularity, and limited intelligence—in the face of increasing dynamism, heterogeneity, and constraint complexity, this paper constructs a unified framework combining request priority modeling, node state representation, and constraint-aware recommendation learning. This research expands the virtual machine placement task from a simple resource feasibility judgment to a recommendation modeling problem oriented towards the adaptation relationship between requests and nodes. This allows the resource allocation process to not only focus on local capacity satisfaction at the current moment but also consider the impact of task importance, node dynamic states, and deployment constraints on the final decision quality. Therefore, this paper enhances the consistency between the virtual machine placement task and the actual cloud platform operation mechanism at the problem modeling level and provides a more systematic research approach for intelligent resource scheduling in complex cloud environments.

From a methodological design perspective, the overall framework constructed in this paper has strong completeness and practical relevance. Request priority modeling strengthens the model's understanding of the differences in the importance of different businesses and service requirements, enabling placement decisions to move beyond resource allocation under a uniform standard and reflect a fine-grained scheduling approach oriented towards service value. Node state representation, starting from the dynamic operating environment, jointly characterizes the carrying capacity, resource reserves, and load characteristics of candidate nodes, thereby improving the model's ability to identify node availability and long-term placement

quality. Constraint-aware recommendation learning further integrates capacity limitations, deployment rules, and placement feasibility directly into the decision-making process, making the generated recommendation results more rational in execution and adaptable to engineering. These designs collectively drive the transformation of virtual machine resource matching problems from experience-driven to data-driven, and from static rules to intelligent decision-making, providing a scalable technical foundation for subsequent research.

From an application perspective, this work has significant practical value in improving cloud platform resource utilization efficiency, enhancing service quality assurance capabilities, and promoting the construction of intelligent operation and maintenance systems. In large-scale data centers, elastic computing platforms, cloud-edge collaborative systems, and multi-tenant shared infrastructure, virtual machine requests often exhibit characteristics such as continuous arrival, significant load fluctuations, and complex resource competition. Without a placement mechanism with priority awareness and constraint understanding capabilities, problems such as limited response to high-value tasks, node load imbalance, and insufficient overall resource utilization can easily occur. The research approach proposed in this paper provides theoretical support for cloud service providers to build more refined, intelligent, and interpretable resource management mechanisms, and also offers methodological references for application scenarios such as online scheduling, service orchestration, computing power distribution, and cost control. Furthermore, this research establishes a closer connection between recommendation learning and cloud computing resource management, which will help promote the further application and development of intelligent recommendation methods in infrastructure scheduling, task placement optimization, and platform autonomous control.

Future research can be further expanded at several levels. First, with the increasing richness of task types, resource types, and deployment forms in cloud environments, subsequent work can further consider unified modeling for multi-resource coupling constraints, heterogeneous acceleration devices, and cross-node collaborative scenarios to enhance the adaptability of the method to complex real-world environments. Second, virtual machine placement decisions in practical platforms are often closely related to objectives such as energy consumption control, service reliability, latency assurance, and migration overhead. Therefore, it is necessary to further explore multi-objective joint optimization mechanisms based on existing research to improve the practical value of the model in comprehensive scheduling scenarios. Furthermore, with the continuous development of cloud-edge-device collaborative architectures and large-scale online service systems, intelligent placement mechanisms oriented towards continuous decision-making, long-term benefit modeling, and real-time feedback learning will become an important direction. Further research on these issues will not only help improve the theoretical framework of resource matching and placement recommendation methods but will also have a profound impact on the efficient operation, intelligent management, and industrial application of cloud computing platforms.

References

[1] Ghasemi and A. Keshavarzi, "Energy-efficient virtual machine placement in heterogeneous cloud data centers: a clustering-enhanced multi-objective, multi-reward reinforcement learning approach," Cluster Computing, vol. 27, no. 10, pp. 14149-14166, 2024.

[2] Alourani, A. Khalid, M. Tahir, et al., "Energy efficient virtual machines placement in cloud datacenters using genetic algorithm and adaptive thresholds," PLOS ONE, vol. 19, no. 1, p. e0296399, 2024.

[3] N. Alharbe, M. A. Rakrouki, and A. Aljohani, "An improved ant colony algorithm for solving a virtual machine placement problem in a cloud computing environment," IEEE Access, vol. 10, pp. 44869-44880, 2022.

[4] K. Balaji, P. Sai Kiran, and M. Sunil Kumar, "Power aware virtual machine placement in IaaS cloud using discrete firefly algorithm," Applied Nanoscience, vol. 13, no. 3, pp. 2003-2011, 2023.

[5] K. Singh, S. R. Swain, D. Saxena, et al., "A bio-inspired virtual machine placement toward sustainable cloud resource management," IEEE Systems Journal, vol. 17, no. 3, pp. 3894-3905, 2023.

[6] J. Peake, M. Amos, N. Costen, et al., "PACO-VMP: parallel ant colony optimization for virtual machine placement," Future Generation Computer Systems, vol. 129, pp. 174-186, 2022.

[7] E. Hormozi, S. Hu, Z. Ding, et al., "Energy-efficient virtual machine placement in data centres via an accelerated Genetic Algorithm with improved fitness computation," Energy, vol. 252, p. 123884, 2022.

[8] O. Hadary, L. Marshall, I. Menache, et al., "Protean:VM allocation service at scale," in Proceedings of the 14th USENIX Symposium on Operating Systems Design and Implementation, 2020, pp. 845-861.

[9] H. Barbalho, G. Kovaleski, B. Li, et al., "Virtual machine allocation with lifetime predictions," Proceedings of Machine Learning and Systems, vol. 5, pp. 232-253, 2023.

[10] L. Caviglione, M. Gaggero, M. Paolucci, et al., "Deep reinforcement learning for multi-objective placement of virtual machines in cloud datacenters," Soft Computing, vol. 25, no. 19, pp. 12569-12588, 2021.

[11] D. Saxena, I. Gupta, J. Kumar, et al., "A secure and multiobjective virtual machine placement framework for cloud data center," IEEE Systems Journal, vol. 16, no. 2, pp. 3163-3174, 2021.

[12] S. Long, Z. Li, Y. Xing, et al., "A reinforcement learning-based virtual machine placement strategy in cloud data centers," in Proceedings of the 2020 IEEE 22nd International Conference on High Performance Computing and Communications, IEEE 18th International Conference on Smart City, and IEEE 6th International Conference on Data Science and Systems, 2020, pp. 223-230.

[13] H. D. Rjeib and G. Kecskemeti, "VMP-ER: an efficient virtual machine placement algorithm for energy and resources optimization in cloud data center," Algorithms, vol. 17, no. 7, p. 295, 2024.

[14] M. Koubàa, A. S. Karar, and F. Bahloul, "Optimizing scheduled virtual machine requests placement in cloud environments: a Tabu search approach," Computers, vol. 13, no. 12, p. 321, 2024.