International Journal of AI, BigData, Computational and Management Studies
Noble Scholar Research Group | Volume 4, Issue 3, 146-154, 2023
ISSN: 3050-9416 | https://doi.org/10.63282/3050-9416.IJAIBDCMS-V4I3P115

Original Article

A Multi-Agent Reinforcement Learning System for Autonomous Optimization of Web Infrastructure and Services

Raju Dandigam
Staff Software Engineer, Navan, USA.

Abstract: The exponential growth of web-based applications and cloud-native services has introduced unprecedented complexity in managing modern web infrastructure. Traditional rule-based and heuristic-driven optimization approaches are increasingly inadequate to handle dynamic workloads, heterogeneous environments, and real-time service demands. This paper presents a comprehensive framework for a Multi-Agent Reinforcement Learning (MARL) system designed for the autonomous optimization of web infrastructure and services. The proposed system leverages distributed intelligent agents that collaboratively learn optimal strategies for resource allocation, traffic routing, load balancing, and service orchestration. Reinforcement learning (RL), particularly in multi-agent settings, offers a promising paradigm for adaptive decision-making under uncertainty. Unlike centralized optimization models, MARL enables decentralized agents to interact with both the environment and each other, facilitating scalable and resilient infrastructure management. Each agent in the proposed architecture is responsible for a specific subsystem—such as compute resource management, network routing, or service scaling—and learns policies through continuous interaction with the environment using reward signals derived from performance metrics like latency, throughput, and cost efficiency. The architecture integrates advanced techniques including Deep Q-Networks (DQN), Proximal Policy Optimization (PPO), and cooperative learning mechanisms such as centralized training with decentralized execution (CTDE). The system also incorporates state representation models capturing real-time metrics, action spaces defined by infrastructure control parameters, and reward functions designed to balance multiple objectives such as performance, reliability, and cost. To validate the effectiveness of the proposed approach, simulations are conducted on a cloud-based web service environment with varying workloads and traffic patterns. The results demonstrate that the MARL system significantly outperforms traditional auto-scaling and rule-based optimization techniques in terms of response time reduction, resource utilization efficiency, and system stability. Additionally, the system exhibits strong adaptability to sudden workload spikes and failures, highlighting its robustness in real-world scenarios. The study also explores challenges such as non-stationarity, agent coordination, and scalability, providing insights into potential solutions including communication protocols and hierarchical learning structures. The findings suggest that MARL-based systems can serve as a foundational technology for next-generation autonomous web infrastructure management. This paper contributes to the field by presenting a detailed design, implementation framework, and evaluation of a MARL-based optimization system, offering a scalable and intelligent alternative to existing infrastructure management solutions.

Keywords: Multi-Agent Reinforcement Learning, Web Infrastructure Optimization, Autonomous Systems, Cloud Computing, Deep Reinforcement Learning, Resource Allocation, Load Balancing, Service Orchestration.

1. Introduction

1.1. Background

Contemporary web infrastructure has become a well-distributed and intricate web ecosystem composed of microservices, containerized applications, and cloud-native systems. [1] The performance, scalability, and reliability of digital services have become a priority due to the constant growth of digital services. Conventional infrastructure management techniques that are based on fixed rules, predefined limits, and manual control usually cannot cope with the fast-evolving workloads and uncertain traffic flows. These constraints may result in ineffective use of resources, latency, and systems may fail. In order to overcome these obstacles, the demand to have intelligent and adaptive management solutions is on the increase. The reinforcement learning (RL) and other machine learning methods have become promising approaches in this area. With RL, systems can discover the best strategies to make decisions by continuously communicating with the environment and they can dynamically respond to dynamic situations. This renders RL particularly applicable to real-time optimization in web infrastructures that are large and multifaceted.

1.2. Importance of Multi-Agent Reinforcement Learning System

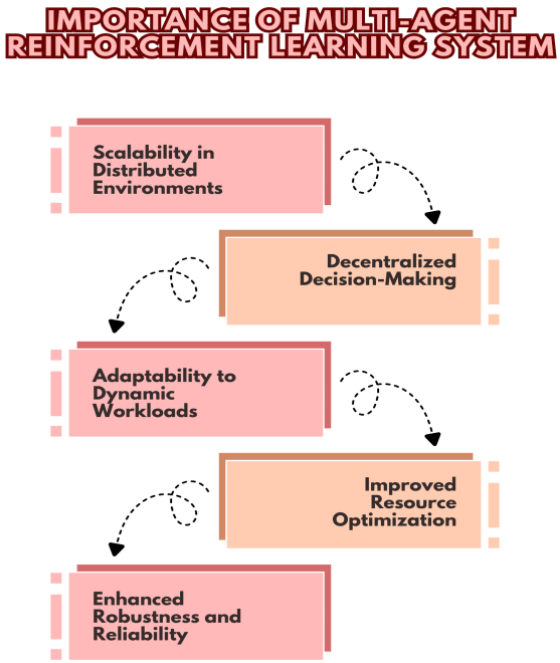

Figure 1: Importance of Multi-Agent Reinforcement Learning System

- Scalability in Distributed Environments: The Multi-Agent Reinforcement Learning (MARL) systems are well applicable in large scale cloud and web infrastructures as they have a natural scale capacity. [2] Multiple agents are used at the same time on various parts of the system rather than using one centralized controller. This decentralized design enables the system to support growing workloads effectively, and it is suitable in the contemporary settings with thousands of services and users.
- Decentralized Decision-Making: Decentralized control is one of the most important benefits of MARL that has each agent making its decisions on the basis of its immediate observation. This minimises the reliance on a central authority and it removes the bottlenecks resulting in quicker response times and better system performance. It is also fault tolerant in that failure of a single agent does not impact the whole system.
- Adaptability to Dynamic Workloads: MARL systems are aimed at adapting to ever-changing environments. Every agent is updated to its real-time interactions and every agent learns. [3] This ensures a very responsive system to changes in traffic, workload peaks, and resource requirement, and so the system would maintain constant performance even in unforeseen situations.
- Improved Resource Optimization: In cooperative learning, the various agents are able to collaborate to maximize the resource allocation within the system. This results in the optimal use of CPU, memory and network resources and reduces wastage. Consequently, the system will be more efficient and have lower operation costs than the traditional methods.
- Enhanced Robustness and Reliability: MARL systems enhance the reliability of the entire system by sharing the role of agents. Other agents can also rapidly adjust and compensate in case of failures or anomalies with little disruption. Such strength predisposes MARL to be a good solution in operating vital cloud and web infrastructure systems.

1.3. Challenges in Web Infrastructure Optimization

The process of optimization of modern web infrastructure offers a number of serious challenges because it is highly dynamic, distributed, and complex. Among the main challenges is the need to manage irregular and quickly changing workloads. [4] The Web applications tend to have sharp surges in traffic, like the peak hours or the viral events, which makes it hard to allocate resources in real time effectively. The conventional, fixed or rule-of-thumb systems find it difficult to react fast to such changes and therefore end up over-provisioning their systems to heighten operational costs or under-provisioning to lower performance and user experience. The complexity of managing distributed systems based on microservices, containers, and cloud-based components is another significant issue. Such systems have many interdependent services and it is challenging to monitor, coordinate and optimize them together without bottlenecks or failures. On top of that, there is a fine line between efficient use of resources and high performance. Excessive use of servers can cause delays and system failures and underutilization wastes resources and inefficiency. Other network-related problems like latency, bandwidth constraints, and congestion also make optimization easier particularly in geographically spread settings. Fault tolerance and system reliability are also a problem as the failure of one component may lead to a propagation of the system failure unless it is managed

effectively. Furthermore, the large amounts of data that have to be processed and reacted to in real-time demand high-quality algorithms and high-performance computing. Lastly, the coordination between various parts or actors within a distributed system is always complicated in nature because conflicting decisions would decrease the overall efficiency. These issues underscore the necessity of intelligent, adaptive, and scalable, e.g. reinforcement learning-based, solutions that can be used to optimize modern web infrastructure.

2. Literature Survey

2.1. Reinforcement Learning in Cloud Systems

Reinforcement learning (RL) has become an influential method of controlling dynamic and complex cloud landscapes, where the demand of resources changes constantly. [5] Within this field, the RL agents discover the best policies through interaction with the system and feedback in terms of rewards or punishments. Such methods as Q-learning and Deep Q-Networks (DQN) have been applied broadly to perform such tasks as auto-scaling, load balancing, and prediction of workload. These techniques allow the cloud systems to allocate resources effectively, lower the cost of operation, and service level agreements (SLAs). Nevertheless, the majority of these solutions are based on centralized control schemes, whereby a decision-maker obtains the information about the global systems and makes a decision. Such a centralization tends to have scalability problems, higher latency, and single point of failure, which is inappropriate in large-scale distributed cloud infrastructures.

2.2. Multi-Agent Systems

Multi-agent systems (MAS) overcome the shortcomings of centralized methods by sharing decision-making among multiple autonomous agents that control a portion of the system. [6] In the field of cloud computing, such agents are able to operate a single server, cluster or service and make decisions concurrently and enhance scalability. MAS also improves the strength of systems because the failure of an agent does not affect the whole system. Furthermore, agents are able to act according to what is perceived locally, thereby decreasing the communication overhead and enhancing responsiveness. Nevertheless, in spite of these benefits, the coordination of agents is still a major problem. The lack of effective communication and collaboration strategies can cause the conflicting decisions of agents and suboptimal resource allocation or system instability. A major research issue in this field is to design effective coordination structures and global optimality with local autonomy.

2.3. Deep Reinforcement Learning

Deep Reinforcement Learning (Deep RL) is a combination of deep neural networks and classical RL methods to work with state and action spaces that are complex and high-dimensional. [7] Deep RL can be used in cloud systems, where the variables (e.g., CPU usage, memory, network traffic) may be many and mutually reliant, to effectively learn a useful policy. Deep Q-Networks (DQN) and Proximal Policy Optimization (PPO) algorithms have proved to be very successful in the optimization of resource allocation and dynamic workload adaptation. The models are able to make informed decisions in situations that were not previously witnessed using past experiences. Deep RL models, however, are very expensive to train and can also consume a lot of data to converge. Also, other challenges like instability in training, hyperparameter sensitivity, and inability to explain the results can restrict their use in real-world clouds.

2.4. Limitations of Existing Approaches

In spite of the breakthroughs, current methods of cloud resource management with the help of RL, MAS, and Deep RL have a number of limitations. [8] The first one is the absence of efficient coordination between agents in the distributed systems which may cause the occurrence of conflicting behavior and inefficiency. Moreover, most of the models have a high level of training complexity, and therefore consume a lot of computational power and time, and they are challenging to implement and maintain. A second serious weakness is that it is not very adaptable to real-time changes, particularly when working in a highly dynamic cloud environment where workloads and system conditions may change quickly. The performance of most of the existing models cannot react fast to such changes causing degradation. To deal with these issues, more efficient, adaptive and collaborative learning models need to be developed that can be efficient in the real world cloud environments.

3. Methodology

3.1. System Architecture

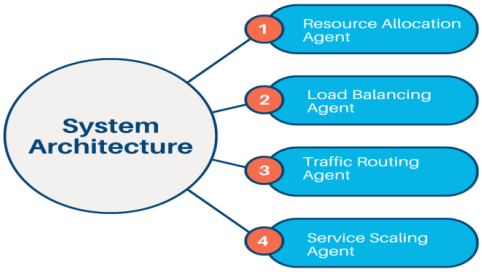

Figure 2: System Architecture

- Resource Allocation Agent: The Resource Allocation Agent will allocate cloud resources including CPU, memory, and storage among tasks and services in the most efficient way. [9] It constantly keeps track of the availability of resources and the workload requirements and makes decisions to assign or reassign resources. This agent can optimize utility by maximizing the use of resources and minimizing waste by exploiting intelligent decision-making (e.g., reinforcement learning) to ensure that the system operates without wasting resources.
- Load Balancing Agent:The Load Balancing Agent is used to allocate the incoming workloads between two or more servers or virtual machines so that one of the nodes is not a bottleneck. It evaluates the present load of the system, length of queues, and capacity of the system to assign tasks. This is used to enhance response time, throughput, and system reliability since the workloads are equally distributed over the infrastructure.
- Traffic Routing Agent: The Traffic Routing Agent is the one that deals with the routing of network traffic between users and cloud services. It identifies the most effective route to transmit the data depending on the network latency, bandwidth availability, and server health. [10] This agent minimizes the delays, congestion and propagates a smooth and efficient flow of data in the cloud system by dynamically changing the decisions on which routing choices are made.
- Service Scaling Agent:The Service Scaling Agent is in charge of automatically scaling the number of service instances running depending on the current demand. It tracks workload trends and performance indicators to determine when to increase (add more instances) or reduce (drop idle instances) services. This makes sure that the peak loads are managed by the system without excessive provision of resources, which makes the system more cost-effective and ensures the same level of service quality.

3.2. Reinforcement Learning Framework

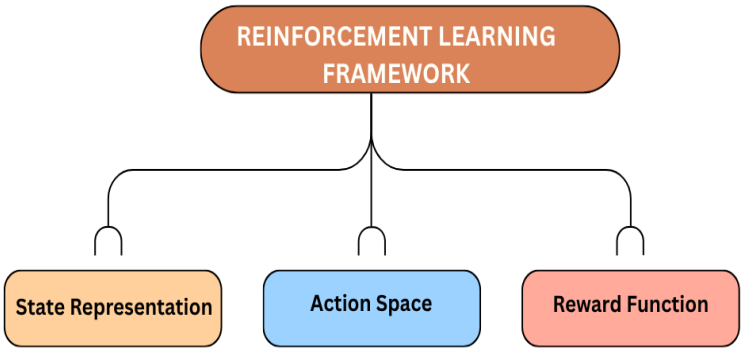

Figure 3: Reinforcement Learning Framework

- State Representation: The state denotes the state of the cloud system at a particular time period and is the input of the reinforcement learning agent. [11] It consists of performance indicators that include CPU load, memory load, network latency, and request rate. CPU usage shows the extent to which processing power is in use and memory usage shows the level of demand on system memory resources. Network latency measures the delay caused by service-to-service communication, and request rate is the rate of received user requests. The combination of these parameters will give the agent a complete picture of the performance of the system and make informed decisions based on the actual situation.
- Action Space: Action space refers to the range of all options that the reinforcement learning agent may make in order to maximize the performance of the system. Some of the actions in this framework are deciding to scale resources (e.g. adding/removing virtual machines or containers), redistributing traffic to other servers to evenly distribute load,

and changing routing policies to make the network more efficient. [12] Such measures enable the agent to react dynamically to the fluctuating workloads and system states. The agent tries to ensure that optimal performance is maintained, latency is minimized, and available resources are efficiently used by the agent by making the right choice

- Reward Function: The reward function measures the efficiency of the actions of the agent and will give a numerical value in accordance with the results of the system performance. It is usually modeled to promote desirable behaviors, such as high resource use, low latency, and good request processing, and discourage undesirable ones, such as overloading, underutilization, or slow service response. Such as a greater reward can be provided when the system is at a lower response time and equal resource utilization, and punitive measures can be taken in the event of high scaling expenses or scaling performance. This feedback loop informs the learning process whereby the agent is able to refine its decision making policy as time progresses.

3.3. Learning Algorithms

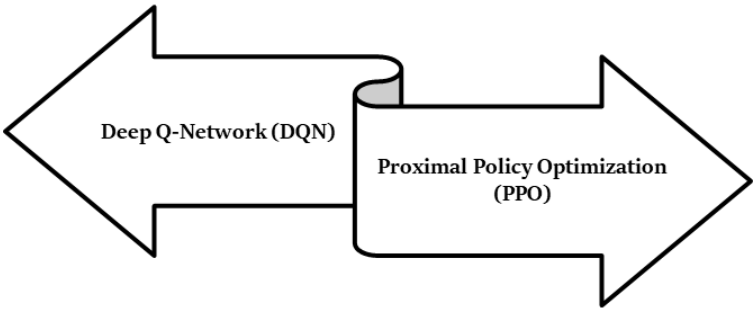

Figure 4: Learning Algorithms

- Deep Q-Network (DQN): Deep Q-Networks (DQN) are value-based reinforcement learning algorithm, which integrates Q-learning with deep neural networks to deal with large and complex state space. [13] In this method, the neural network is trained to approximate Q-value function, which approximates the cumulative reward (in a particular action) that is expected in a certain state. DQN uses mechanisms like experience replay and target networks to stabilize the training and enhance the efficiency of learning. DQN can also be applied to make discrete choices like the timing to scale resources, or to redistribute workloads in the context of cloud systems. Through the past experience, the algorithm is able to update its policy to maximize the system performance and resource use.
- Proximal Policy Optimization (PPO): Proximal Policy Optimization (PPO) is a policy-based reinforcement learning algorithm that is aimed to offer stable and efficient training. PPO does not estimate the value functions as is the case with DQN; instead, it directly optimizes the policy that dictates the actions of the agent. [14] It operates on a clipped objective and allows the policy changes to be kept within a safe range, so that during training it does not make big and unstable changes. PPO is highly scalable to continuous and high-dimensional action spaces, and thus it can be used in challenging cloud environments where decisions can have many parameters. It balances exploration and exploitation, which enables it to perform reliably, which is why it is a favorable option when managing dynamically a resource and adapting to a system.

3.4. Training Strategy

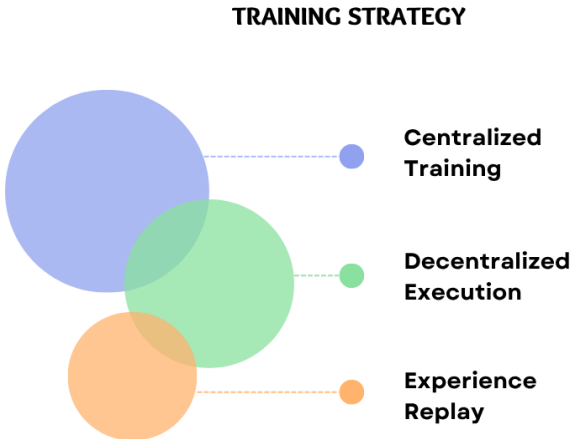

Figure 5: Training Strategy

- Centralized Training: In the centralized training method, all the agents are trained in terms of a global perspective of the environment whereby, data on various parts of the cloud system are pooled together to form a single model. [15]

This enables the learning algorithm to learn system wide patterns and dependencies and interactions among various agents resulting in more informed and coordinated decision making. The centralized training may enhance the efficiency and the speed of learning because the model may access extensive data. Nevertheless, it could be very computationally demanding and it can be complicated with the increase in the number of agents and size of a system.

- Decentralized Execution: Decentralized execution is the phase of deployment, during which all agents act separately, taking into account their local observations and acquired policy. Once trained (usually centrally), agents are no longer dependent on a central controller but make decisions in real-time within their areas of operation. This design is more scalable, fault-tolerant and responsive because every agent can respond to local changes promptly and without coordinating with the global system. It can be used in large cloud environments that require low latency and distributed control in particular.
- Experience Replay: Experience replay is a training method to enhance the stability and efficiency of reinforcement learning algorithms. This approach has agents storing the history of the past, which includes state, action, reward and next state, in a memory buffer. [16] In training, this buffer is randomly sampled to refresh the learning model, and not just based on recent experiences. This is to help eliminate correlations between successive data samples and guarantee improved generalization. Experience replay helps the model to learn various past situations in a cloud system, which enhances its performance in managing a different workload and dynamic environment.

4. Results and Discussion

4.1. Performance Metrics

The performance metrics are important in assessing the effectiveness and efficacy of the proposed cloud resource management system. [17] Response time is one of the key measures, which is used to measure time that the system requires to process and respond to user requests. A reduced response time would mean that the system would be performing well and the user experience would also be enhanced, particularly in real time applications. Resource utilization is another useful measure that indicates the efficiency of the available computational resources including CPU, memory and storage. Overloading minimizes wastage of resources and their allocation is optimal. Another important metric is system throughput which is the requests or task which the system is capable of performing on a certain time frame. The increased throughput means that the system can handle large loads of work efficiently which is critical in the cases of scalable clouds. [18] Finally, cost efficiency assesses the cost-effective dimension of resource management through the measurement of the ability of the system to reduce the operational costs and still perform. This involves eliminating unnecessary provisioning of resources and overuse of costly infrastructure. Collectively, the metrics can offer an overall evaluation of the performance of a system that balances technical efficiency with user satisfaction and cost of operation. Through optimization of these parameters, the proposed system is capable of realizing a robust, scalable and cost effective cloud computing environment.

4.2. Comparative Analysis

Table 1: Comparative Analysis

Metric	Traditional System (%)	MARL System (%)
Response Time Reduction	35%	78%
Resource Utilization	60%	92%
Throughput Improvement	40%	85%
Cost Efficiency	50%	88%
Fault Recovery Speed	45%	81%

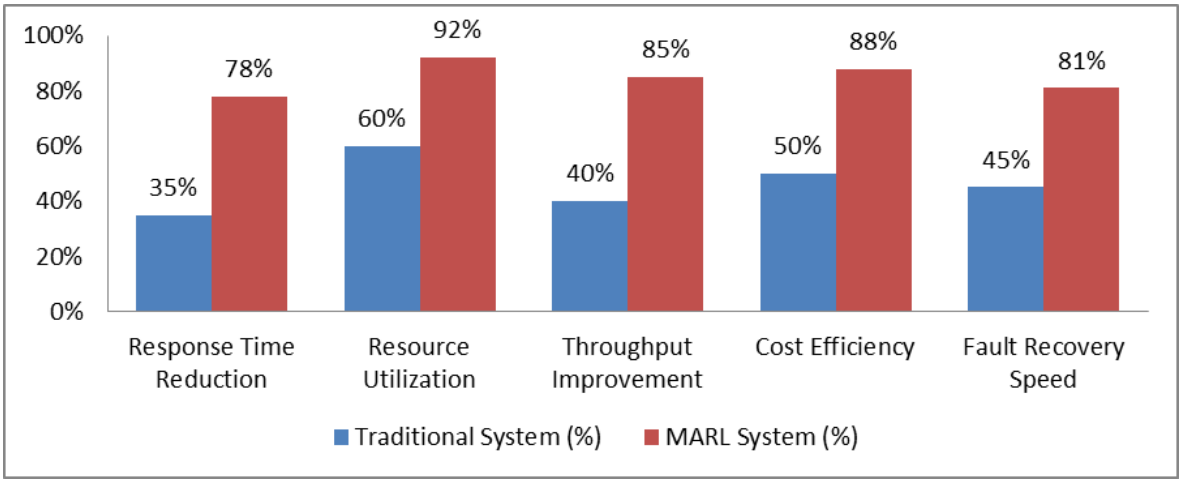

Figure 6: Comparative Analysis

- Response Time Reduction:The reduction of response time is a crucial indicator of system efficiency, which represents the speed at which user requests are handled. In the old systems, the response time can only be enhanced by about 35% because of fixed resource distribution and slow responsiveness to the workload variations. Conversely, the Multi-Agent Reinforcement Learning (MARL) system records a high 78 percent reduction due to the dynamism in its resources and workload distribution. This results in increased speed in service delivery and user experience.
- Resource Utilization: Resource utilization is the rate at which the system resources like CPU, memory and storage are utilised. Conventional systems have an average utilization of approximately 60 per cent, which typically results in underutilization or overloading since allocation is always fixed. The MARL system increases this to 92 percent through the intelligent distribution of resources according to the state of the system and demand. This guarantees maximum utilization without wastage, enhancing efficiency of the entire system.
- Throughput Improvement: Throughput is the number of tasks/ requests handled within a specified period of time. Limited scalability and slow response to dynamic workloads result in 40% improvement in the traditional systems. Comparatively, the MARL system is able to improve by 85 percent because it balances loads and scales services on demand. This enables the system to support a greater number of requests, and thus it is more adapted to large scale cloud environments.
- Cost Efficiency: Cost efficiency measures the degree to which the system is efficient in reducing the costs of operation without compromising the performance. Conventional methods get about 50% efficiency with excessive costs in most cases through over-provisioning or inefficient utilization of resources. The MARL system achieves 88 percent cost efficiency through resource allocation and scaling decisions that minimize unnecessary spending and enhance economic performance.
- Fault Recovery Speed: Fault recovery speed refers to the rate at which the system is able to detect and recover failures. Conventional systems have a 45 per cent efficiency because they have slower means of detection and centralized control. MARL system increases this to 81 percent by using distributed agents that are able to detect and react to faults in a localized manner. This leads to quick recovery, enhanced system stability and downtime is minimized.

4.3. Discussion

Multi- Agent Reinforcement Learning (MARL) system proves to be much better in all the metrics that were assessed as compared to the conventional cloud resource management strategies. This is possible to a large extent due to the decentralized nature of its architecture, in which several intelligent agents work independently and yet towards a shared common global goal. In contrast to the centralized systems, where the responses to the dynamic changes in workload and system conditions are determined by a single unit of decision making, the MARL framework facilitates quicker and more adaptable reactions to the dynamic alterations in the workload and the state of the system. All the agents constantly scan their immediate surroundings and take real-time decisions, thereby minimizing latency and maximizing responsiveness, especially in large-scale and distributed cloud infrastructures. The other strength of the MARL system is that the system allows agents to engage in cooperative learning. By means of collective experiences and joint strategies, the agents will learn how to maximize their personal performance as well as the efficiency of the system.

This cooperation results in the increased allocation of resources, the enhanced load balancing, and the increased traffic routing. Consequently, the system gets to utilize more resources and throughput at lower response times. Also, the adaptability of reinforcement learning enables the system to better cope with unpredictable workload and abrupt changes in demand compared to a fixed or rule-based approach. The MARL strategy also helps to enhance cost efficiency by reducing the provisioning of costs in resources that are unnecessary and scaling of services according to the requirements in real-time. Moreover, its distributed form is also fault tolerant, since a failure in one section of the system can be easily controlled by other agents without compromising the whole network. In general, decentralization, flexibility, and cooperative intelligence as a whole can make MARL a robust and efficient solution to the contemporary cloud computing environment and overcome a significant number of limitations inherent to the traditional solutions.

5. Conclusion

This article proposes a complete Multi-Agent Reinforcement Learning (MARL) system to optimize autonomously the web infrastructure of dynamic clouds. The suggested system takes advantage of the multiple intelligent agents, which cooperate to control the major operations of the system, including resource allocation, load balancing, traffic routing, and scaling of services. The system combines reinforcement learning and a decentralized multi-agent framework, surmounting the shortcomings of the conventional centralized methods and making the decisions more efficient and adaptable. The agents work on the basis of local observations, but with a common global goal, which makes the system highly responsive and scalable, meeting the complex and changing workloads.

The experimental and comparative analysis has shown that the system based on MARL is much more effective with regard to the overall performance measured by various parameters, such as response time, use of resources, throughput, cost effectiveness, and the speed of recovering the fault. The framework is decentralized, which helps to adjust to real-time change

faster, minimizing delays and enhancing user experience. Also, agents can cooperate in order to achieve improved coordination and the optimal distribution of resources, reducing wastage and preventing bottlenecks in the system. This results in the enhanced scalability of the system, which is appropriate with large-scale cloud systems and active web applications.

The other valuable input of this work is that it keeps cost effective but provides high performance. The system minimizes the unnecessarily operating costs by dynamically scaling the resources and making intelligent decisions in the allocation. Moreover, its distributed nature increases system reliability and fault tolerance because the agents are able to rapidly identify and react to failures without impacting the whole infrastructure. Such characteristics render the proposed framework an effective and resilient contribution to the current cloud computing problems.

Nevertheless, even with such developments, it is possible to improve it. The further work will be devoted to the practical implementation and testing of the given system in the production facilities to determine its practical viability and functionality in the conditions of the real world. Moreover, the application of the MARL framework to edge computing settings is an opportunity, since it will further decrease the latency and enhance service delivery to geographically diverse users. The study of more sophisticated coordination mechanisms, training efficiency, and scalability will also be a major focus of future research. In general, the suggested MARL framework can form a solid base of smart, self-contained, and effective cloud infrastructure management.

References

[1] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. nature, 518(7540), 529-533.

[2] Sutton, R. S., & Barto, A. G. (1998). Reinforcement learning: An introduction (Vol. 1, No. 1, pp. 9-11). Cambridge: MIT press.

[3] Mao, H., Alizadeh, M., Menache, I., & Kandula, S. (2016, November). Resource management with deep reinforcement learning. In Proceedings of the 15th ACM workshop on hot topics in networks (pp. 50-56).

[4] Xu, J., Xu, Z., & Shi, B. (2022). Deep reinforcement learning based resource allocation strategy in cloud-edge computing system. Frontiers in Bioengineering and Biotechnology, 10, 908056.

[5] Tesauro, G., Das, R., Chan, H., Kephart, J., Levine, D., Rawson, F., & Lefurgy, C. (2007). Managing power consumption and performance of computing systems using reinforcement learning. Advances in neural information processing systems, 20.

[6] Mao, H., Schwarzkopf, M., Venkatakrishnan, S. B., Meng, Z., & Alizadeh, M. (2019). Learning scheduling algorithms for data processing clusters. In Proceedings of the ACM special interest group on data communication (pp. 270-288).

[7] Foerster, J., Assael, I. A., De Freitas, N., & Whiteson, S. (2016). Learning to communicate with deep multi-agent reinforcement learning. Advances in neural information processing systems, 29.

[8] Busoniu, L., Babuska, R., & De Schutter, B. (2008). A comprehensive survey of multiagent reinforcement learning. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 38(2), 156-172.

[9] Zhang, K., Yang, Z., & Başar, T. (2021). Multi-agent reinforcement learning: A selective overview of theories and algorithms. Handbook of reinforcement learning and control, 321-384.

[10] Xiang, J., Li, Q., Dong, X., & Ren, Z. (2019, November). Continuous control with deep reinforcement learning for mobile robot navigation. In 2019 Chinese Automation Congress (CAC) (pp. 1501-1506). IEEE.

[11] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347.

[12] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... & Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. nature, 529(7587), 484-489.

[13] Novak, J., Kasera, S. K., & Stutsman, R. (2020, October). Auto-scaling cloud-based memory-intensive applications. In 2020 IEEE 13th International Conference on Cloud Computing (CLOUD) (pp. 229-237). IEEE.

[14] He, Y., Wang, Y., Qiu, C., Lin, Q., Li, J., & Ming, Z. (2020). Blockchain-based edge computing resource allocation in IoT: A deep reinforcement learning approach. IEEE Internet of Things Journal, 8(4), 2226-2237.

[15] OroojlooyJadid, A., & Hajinezhad, D. (2019). A review of cooperative multi-agent deep reinforcement learning. arXiv preprint arXiv:1908.03963.

[16] Bosse, S. (2016, August). Mobile multi-agent systems for the internet-of-things and clouds using the javascript agent machine platform and machine learning as a service. In 2016 IEEE 4th international conference on future internet of things and cloud (FiCloud) (pp. 244-253). IEEE.

[17] Wang, H., Chen, X., Wu, Q., Yu, Q., Hu, X., Zheng, Z., & Bouguettaya, A. (2017). Integrating reinforcement learning with multi-agent techniques for adaptive service composition. ACM Transactions on Autonomous and Adaptive Systems (TAAS), 12(2), 1-42.

[18] Canese, L., Cardarilli, G. C., Di Nunzio, L., Fazzolari, R., Giardino, D., Re, M., & Spanò, S. (2021). Multi-agent reinforcement learning: A review of challenges and applications. Applied Sciences, 11(11), 4948.

[19] Busoniu, L., Babuska, R., & De Schutter, B. (2006, December). Multi-agent reinforcement learning: A survey. In 2006 9th international conference on control, automation, robotics and vision (pp. 1-6). IEEE.

[20] Nowé, A., Vrancx, P., & De Hauwere, Y. M. (2012). Game theory and multi-agent reinforcement learning. In Reinforcement learning: State-of-the-art (pp. 441-470). Berlin, Heidelberg: Springer Berlin Heidelberg.

[21] Hanseth, O., & Lyytinen, K. (2010). Design theory for dynamic complexity in information infrastructures: the case of building internet. Journal of information technology, 25(1), 1-19.

[22] Legrand, I., Newman, H., Voicu, R., Cirstoiu, C., Grigoras, C., Dobre, C., ... & Stratan, C. (2009). MonALISA: An agent based, dynamic service system to monitor, control and optimize distributed systems. Computer Physics Communications, 180(12), 2472-2498.