www.ijecs.in
International Journal Of Engineering And Computer Science
Volume 11 Issue 11 November 2022, Page No. 25665-25677
ISSN: 2319-7242 DOI: 10.18535/ijecs/v11i08.4718

Multi-Agent Reinforcement Learning for Efficient Task Scheduling in Edge-Cloud Systems

Vinay Chowdary Manduva

Department of Computer Science and Engineering,
Amrita School of Engineering, Amrita Vishwa Vidyapeetham, India.

Abstract
This generated a new trend in modern computing, known as the edge-cloud systems, which matrix combines the efficiency of edge computing in terms of latency with the computing power of cloud systems. Task scheduling is a fundamental issue in these systems and its goals involves instructing the computational loads across the edges and the clouds for a desired set of objectives such as low latency, high resource usage, minimal energy consumption, and high throughput among others. Many of the conventional scheduling paradigms are not effective when applied to edge-cloud settings due to their centralized or heuristic nature.
Because scheduling is still in its early stages, there is much opportunity to find novel ways to apply it to edge-cloud systems, as these environments are undervalued now but are expected to see rapid growth soon.
The following challenges are well addressed by Multi-Agent Reinforcement Learning (MARL) which is based on a decentralized decision-making process where several intelligent agents jointly learn and improve scheduling policies. The strength to be gained from the use of MARL lies in the capacity of agents to change their behavior in response to their environment, gain from experience and engagement and make decisions in the light of the context. The techniques like cooperative learning, reward shaping and communication between agents of concurrent tasks make the MARL to provide effective scheduling of tasks in large scale and complex systems besides heterogeneous one.
This article also discusses the rudiments of MARL and the way it is implemented in edge-cloud scheduling of tasks while also pointing out the advantages of the implementation: minimal latency, scalability, heightened system resilience. It also outlines future research areas including how to scale MARL solutions, how agents should be coordinated and the computational resource constraints of the solutions whenever they are to be deployed in practice. Based on the relevant literature and the cases, this paper lays theoretical and practical foundations of the next generation task scheduling defined by the novel MARL framework to enable smarter edge-cloud environments.

Keywords: Multi-Agent Reinforcement Learning (MARL), Task Scheduling, Edge-Cloud Systems, Decentralized Decision-Making, Resource Optimization, Dynamic Scheduling

Introduction

Background

Edge-cloud systems are a novel approach in modern computing structures that has been developed for the needs of continuously increasing slow response, high performance, and scalability. These systems involve the integration of local CPU capability, in form of IoT nodes and edge servers with distant big computing assets in the cloud. By having both layers, the data can be processed near where it is produced (the edge) for real-time operation, while at the same time offloading computationally expensive operations to the cloud at a fraction of the cost.
Nevertheless, scheduling of tasks—the partition of computation tasks between edge and cloud entities—still poses a problem. A host of constraints like dynamic workloads, heterogeneity in device capabilities, and variability in network conditions imply that scheduling of tasks is an optimization problem. This results in

inefficient task allocation and energy consumption, which are major challenges in applications where low latency is desirable such as autonomous vehicles, smart cities and health monitoring.

Motivation

Heuristic methods, centralized planning and arrangement of tasks, and fixed relations between tasks are hardly applicable in case of dynamic and distributed architectures specific to edge-cloud systems. While centralized scheduling means having single points of failure and scaling problem, heuristic approaches can hardly be adjusted to unpredictable changes in the system or developed to use historical experience.

In this context, Multi-Agent Reinforcement Learning emerges as the silver bullet. Using multiple learning agents that can learn the schedule policies of the system on their own and by interacting with the environment, MARL inherits the abilities of adaptability, scalability and decentralized decision-making. Thus, it can be concluded that due to the great capability of MARL to adjust itself to the changes in the system, to optimize the use of available resources, or to find the best compromise between the response time and power consumption, the proposed solution of the task scheduling in the edge-cloud system can be highly effective.

Scope and Objectives

This article aims to explore how MARL can transform task scheduling in edge-cloud systems by addressing the following objectives:

- **Examine the Current Landscape**: Survey the state-of-art in the field of task scheduling and discuss the shortcomings that current methods present when applied in the edge-cloud model.
- **Introduce MARL Fundamentals**: Point out major ideas of MARL and explain how it can be used to cope with decentralized scheduling challenges.
- **Demonstrate MARL Applications**: Detail how MARL can be applied in practice concerning the edge-cloud system's design and the collaboration of agents as well as the rewards.
- **Highlight Benefits and Challenges**: Discuss how MARL performs against traditional methods, what advantages and disadvantages MARL has and what research areas are still open.
- **Outline Future Directions**: Suggest appropriate developments in MARL and how to incorporate the new trends like federated learning, secure edge computing, and AI optimization into MARL.

Achieving these objectives, this article aims to establish MARL as the foundation for designing effective and feasible edge-cloud systems to support demanding application workloads of the future applications, Dynamic Scheduling.

Literature Review

Overview of Edge-Cloud Systems

Key Components of Edge and Cloud Systems

An edge-cloud architecture therefore integrates edge devices which include IoT devices, sensors and servers and edge servers with a centralized cloud infrastructure to support the distributed data processing and storage. The edge layer analyzes data or analysis results locally or near the data source, with low latency and low demand for bandwidth. In this case, the cloud layer offers the broad computational resources and storage capability required to perform computational processing.

The integration of these layers allows for high levels of performance, modularity, and low energy consumption, qualities necessary for the future scenarios that will require cognitive cars, healthcare monitoring, or real-time analysis.

It presents a brief overview on the scheduling techniques that can be applied to edge-cloud systems.

Scheduling Techniques Used in Edge-Cloud Systems

Scheduling in edge-cloud systems is concerned with identifying the most suitable tasks that should be outsourced to the edge layer and the cloud layer respectively. Common techniques include:

- **Static Scheduling**: Tasks are previous directed according to predefined protocols.
- **Dynamic Scheduling**: The assignments are assigned dynamically with the present status of the system in mind.
- **Load Balancing**: Resource allocation is done in a manner that does not overload a single resource by assigning tasks.
- **Priority-Based Scheduling**: Priorities are defined by the priority of the task that is to be accomplished.

Task Scheduling Challenges

It is concerned with the computational and networking boundaries.

- **Resource Limitations**: In the systems that are deployed in edge devices, most of the tasks come with certain limitations especially on the computation and energy.
- **Network Latency and Bandwidth**: Moving data between edge and cloud can take time and with heavy data loads this means that resource intensive applications can be slow.

Real-Time Requirements of Edge-Cloud Scenario

HI media, real-time video processing, industrial and smart IoT applications, and various healthcare applications all need guaranteed low latency and high reliability. Current scheduling strategies fail to address these real-time needs because the introduced approaches cannot dynamically change their operation according to current workloads and available resources.

Traditional Approaches to Task Scheduling

Ad hoc and Algorithmic Approaches

Traditional methods often rely on predefined rules or heuristics, such as:

- **Round-Robin Scheduling**: Working assignments are assigned in cyclic manner.
- **Greedy Algorithms**: Resources are distributed in accordance with their current accessibility to perform a specific task, overlooking the optimality.

Disadvantages of Performing Scheduling at the Heart of Distributed Systems

Centralized approaches face significant drawbacks:

- **Single Point of Failure**: If there is any problem with centralized systems, then it will crop up all over the network.
- **Scalability Issues**: While the number of devices increases, this approach faces high overhead, which makes centralized methods ineffective.

Reinforcement Learning and Multi-Agent Systems

Introduction to RL and Its Usage to Develop Scheduling

Reinforcement Learning (RL) allows the system to find the best action depending on its capability of having interactions with the environment. In task scheduling, RL describes the environment using a dynamic model where learner agents assign tasks in a way that leads to low delays and high resource utilization.

Benefits of MARL for Decentralized Scheduling

- **Scalability**: Many agents act separately and thus; MARL is appropriate for sizeable distributed systems.
- **Adaptability**: Agents in a telephony-based system can flexibly respond to changes in the load and the availability of resources.
- **Decentralized Control**: Unlike some traditional MoC schemes, MARL does not need a central scheduler to allocate calls, thus making the system less likely to be congested.

Graphs and Tables

Comparison of Scheduling Techniques

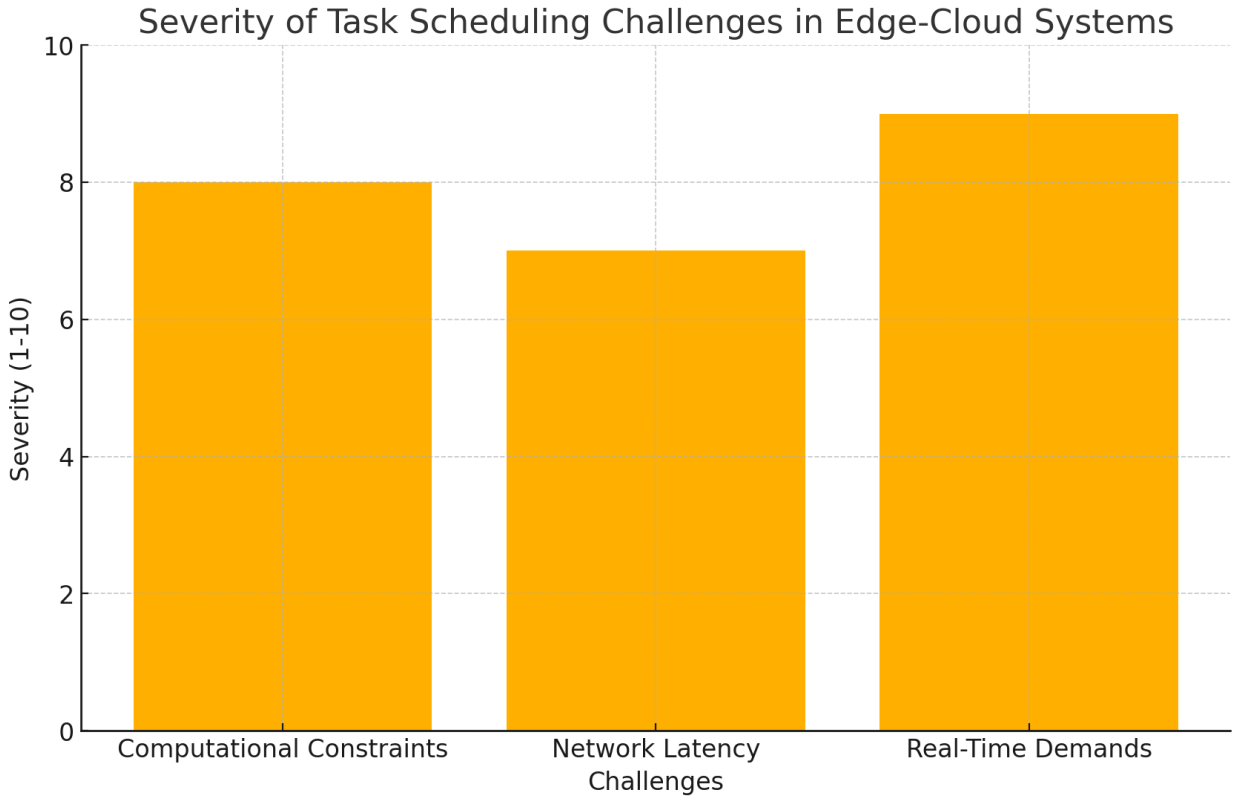
Let's visualize the advantages and limitations of traditional and MARL-based approaches using a table.

Approach	Latency	Scalability	Adaptability	Energy Efficiency	Complexity
Static Scheduling	High	Low	Low	Moderate	Low
Dynamic Scheduling	Moderate	Moderate	Moderate	High	Moderate
Heuristic-Based Scheduling	High	Low	Low	Moderate	Low
MARL-Based	Low	High	High	High	High

Scheduling					

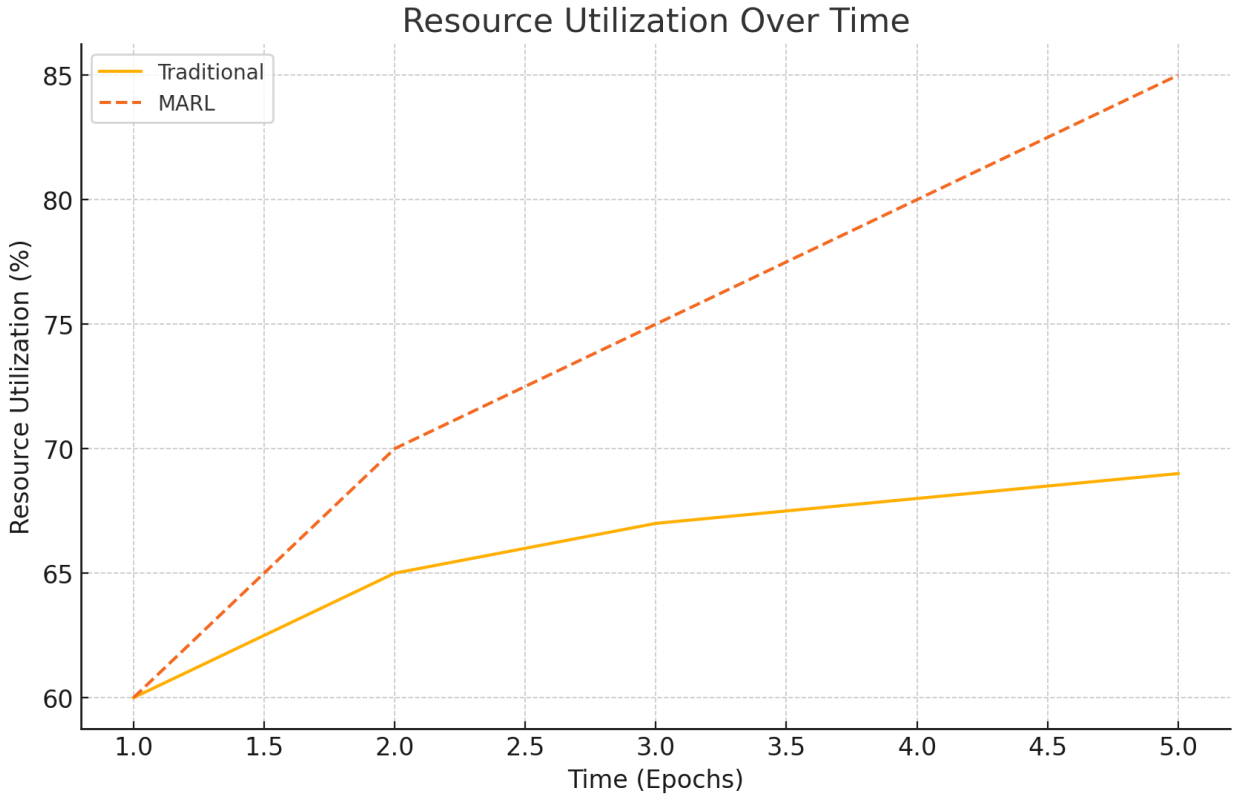
Task Scheduling Challenges

We can also use a bar graph to illustrate key challenges such as computational constraints, network latency, and real-time demands.



Resource Utilization: Traditional vs. MARL

A line graph can demonstrate how resource utilization improves over time with MARL compared to traditional methods.



These visual aids and explanations provide a comprehensive look at the literature review, establishing the context for MARL's transformative potential in task scheduling.

Fundamentals of Multi-Agent Reinforcement Learning (MARL)

What is MARL?

MARL can be best described as an RL with more than one learner where the learners are agents who interact within a common environment. Every agent features programs that define the response to the environment to obtain the best policy. The goal of the agents may be either mutually exclusive or complementary, and their actions affect not only their individual results but also interact with the environment.

The book is organized under the following headings: Agents, Environments, and Objectives.

In MARL, the self-actors are agents who must select an action based on the current observation from the environment which denotes the state of the system. The agents are also interested in the sum of reward and learn to maneuver in the environment over time. The main purpose of MARL is slightly different in each application, such as, task scheduling in edge-cloud system is designed to maximize the minimum task delay and minimize the total resources.

These are the cooperative and the competitive types of settings that are implemented in the domain of MARL.

- **Cooperative Settings**: Multi-agent editors collaborate for a target in a designed approach based on sharing of rewards and learning of policies that are friendly to the overall system. In terms of task scheduling, this manifests itself as all the agents coordinating the task scheduling to happen at the edge and, when it is necessary, the cloud to reduce the amount of time taken and optimize resource utilization.
- **Competitive Settings**: Lovers and managers struggle to obtain personal benefits, without considering the bloc's need. Interestingly, this setting is less so applicable to the general task scheduling but can be useful in certain adversarial conditions as in computer security.

MARL Algorithms

MARL uses a variety of algorithms to solve decision making issues. Some commonly used algorithms are:

Q-Learning

A table-based approach or learning where the Q-Table is modified by agents to learn optimal actions of different stages. However, Q-learning can be limited in its scalability within a complex system such as edge-cloud environments.

Deep Q-Networks (DQN)

An important limitation of Q learning is that the Q-value table becomes impractical for use in situations where the state-space is large or is continuously growing, so to address all these issues, DQN uses a mixture of a deep neural network. As stated, the agents must learn to approximate the Q-function which makes the use of the method ideal for specific edge-cloud scenarios. For example, a deep Q learning neural network can estimate the optimal scheduling strategy to allocate resources depending on resource availability to performs a certain task.

Actor-Critic Methods

Actor-Critic methods combine two networks: an action initiator that selects actions and an action assessor that judge the selected actions. These methods are used in continuous action spaces and dynamic environments, which are suitable for scheduling with varying task demands in edge and cloud networks.

Integration with Task Scheduling Systems

MARL coordination with the edge-cloud task scheduling is done by perceiving each edge device or cloud server as an agent. Environment represents system states like effective task list of a system, status of the network and availability of system resources. Agents get to understand how to perform task distribution in the context of shifts in the system environment, real-time thus.

Key Challenges in MARL

Despite its potential, MARL faces several challenges:

Scalability

The problem becomes much more difficult with the increasing number of agents in the system. The current representatives require computation of large amounts of state and action data, and therefore, entail complexity. This is most apparent in task scheduling because task volume and edge devices increase with time.

The most important fact for the management of the exploration-exploitation trade-off is that the relative balance between exploration and exploitation activities depends on contextual factors.
Learning needs to incorporate both the exploitation (taking actions that has been proven to be good) and the exploration (taking other actions that are not very often taken). Inadequate exploration results in poor scheduling policies and on the other extreme exploitation may hinder learning.

Communication Between Agents

One of the prerogatives of effective cooperation is agents' information exchange, which results in excessive communication load in the case of distributed systems. Further, making sure that logistical coordination happens without degrading actual-time performance is an important concern in edge-cloud task scheduling.

Graphs and Tables

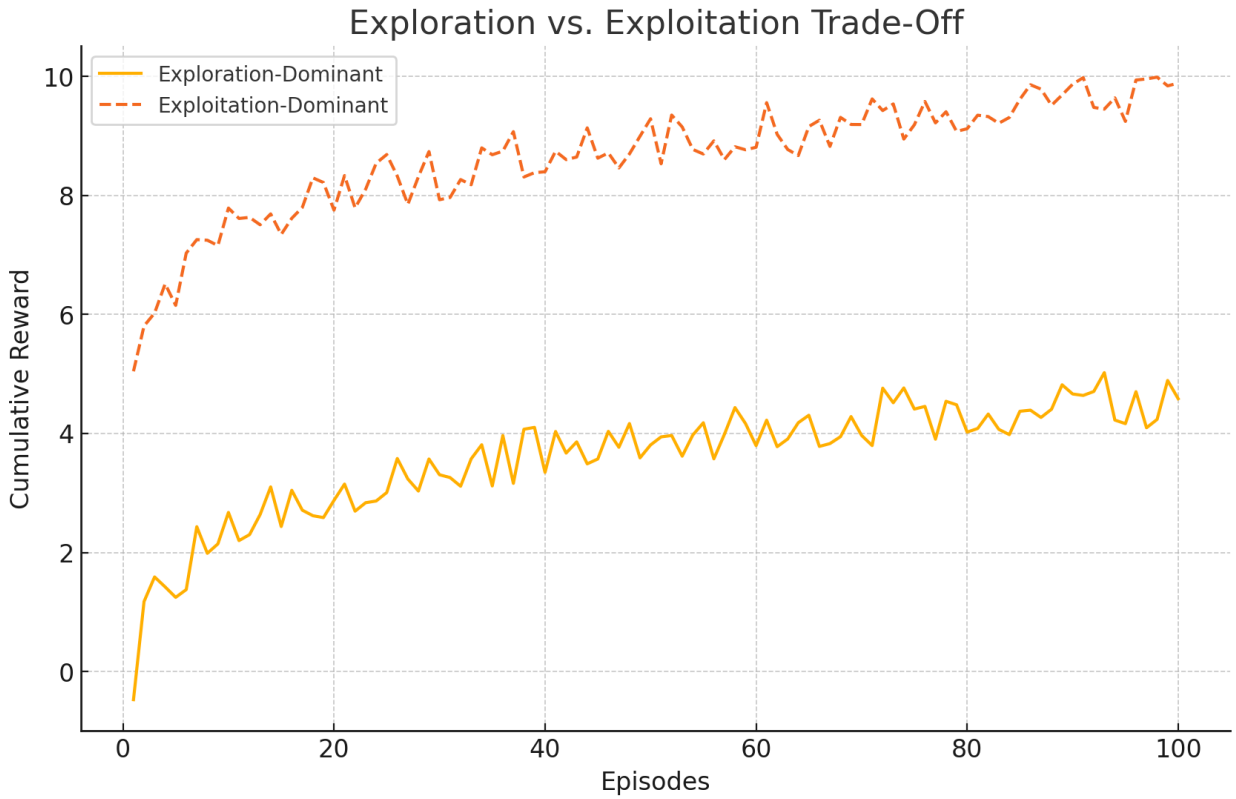
Comparison of MARL Algorithms

The following table compares MARL algorithms based on their key features and applicability to task scheduling.

Algorithm	Key Features	Advantages	Limitations
Q-Learning	Tabular method	Simple to implement	Poor scalability
DQN	Neural network-based Q-function	Handles high-dimensional states	Computationally intensive
Actor-Critic	Dual networks (Actor & Critic)	Works well in dynamic settings	Requires careful parameter tuning

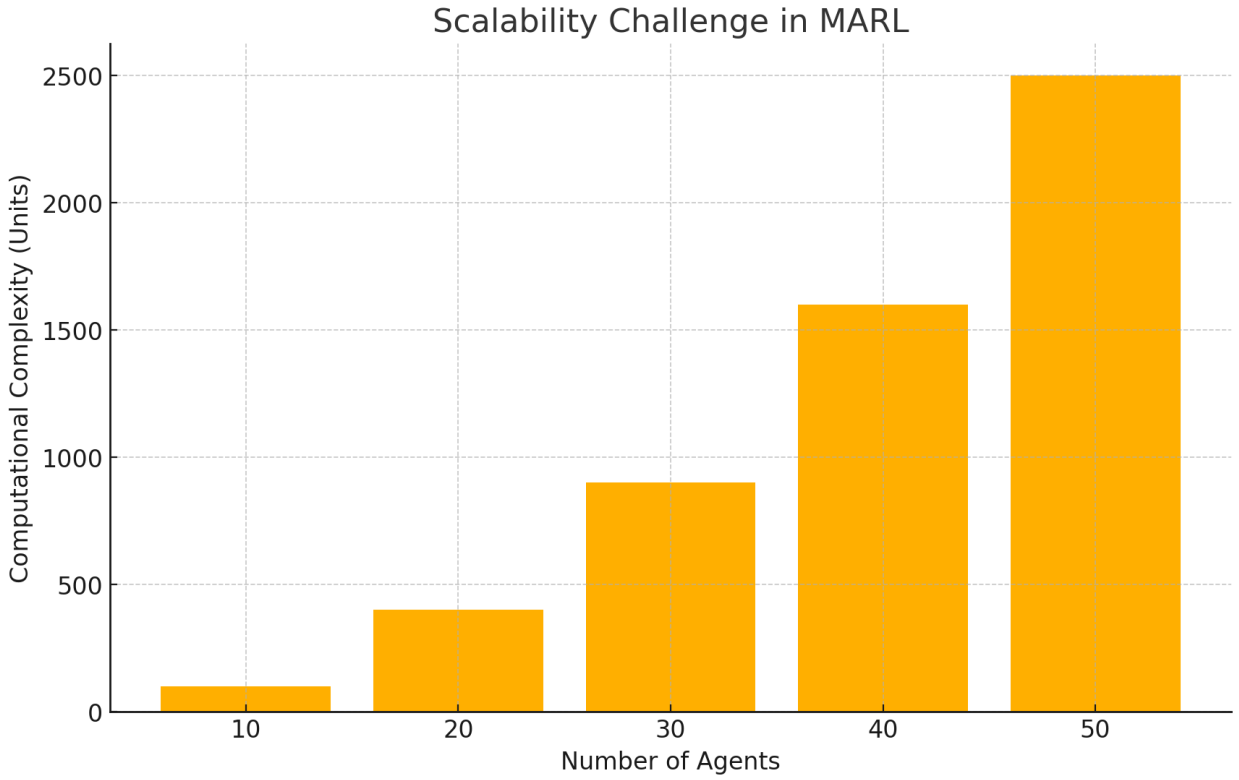
Exploration vs. Exploitation Trade-Off

A line graph can illustrate how the reward changes over time as an agent learns to balance exploration and exploitation.



Scalability Challenge in MARL

A bar graph can depict how computational complexity increases with the number of agents in a MARL system.



This detailed exploration of MARL fundamentals, supported by visuals, highlights the concepts, algorithms, and challenges pivotal to its application in edge-cloud task scheduling systems.

Application of MARL in Edge-Cloud Task Scheduling

A framework for MARL in Task Scheduling

This paper outlines the description of a typical MARL framework.

A MARL framework for edge-cloud task scheduling describes the system environment, and the foregoing computational entities as agents. These agents discover how to schedule tasks both reactively in relation to their perceived environment and proactively in accord with system conditions.

The framework consists of three primary components:

- **Agents**: Act as edge devices or servers to be able to make decisions individually.
- **Environment**: Illustrates, synergistically, the edge-cloud system, tasks, network delay and resource availability.
- **Learning Policy**: Helps to direct agents towards determinations of activities schedules and timely rewards accrued.

Role of Agents, State Action, & Reward Function

- **Agent Roles**: Every agent is assigned to solve a set of tasks, personal to them. For example, edge agents perform tasks that demand latencies in the range of microseconds, whereas cloud agents perform tasks that demand higher latencies of the order of milliseconds.
- **State Representation**: State preserve system detail including the resource on call, how many tasks on queue, or latency in the systems networks.
- **Reward Mechanisms**: Incentives are offered in an attempt to prevent agent's latency, uneven distribution of loads, or high energy consumption

Simulation and Training

The framework also includes the use of simulators in order to optimize the edge-cloud scheduling.

There are many ways to evaluate MARL; however, simulators are used to encourage practice and to implement the models. They mimic the behavior of edge-cloud systems, storing Network characteristics, task arrival rate and resource availability. Some commonly used simulators are i Fog: iFogSim and Cloud: CloudSim.

Training Protocols

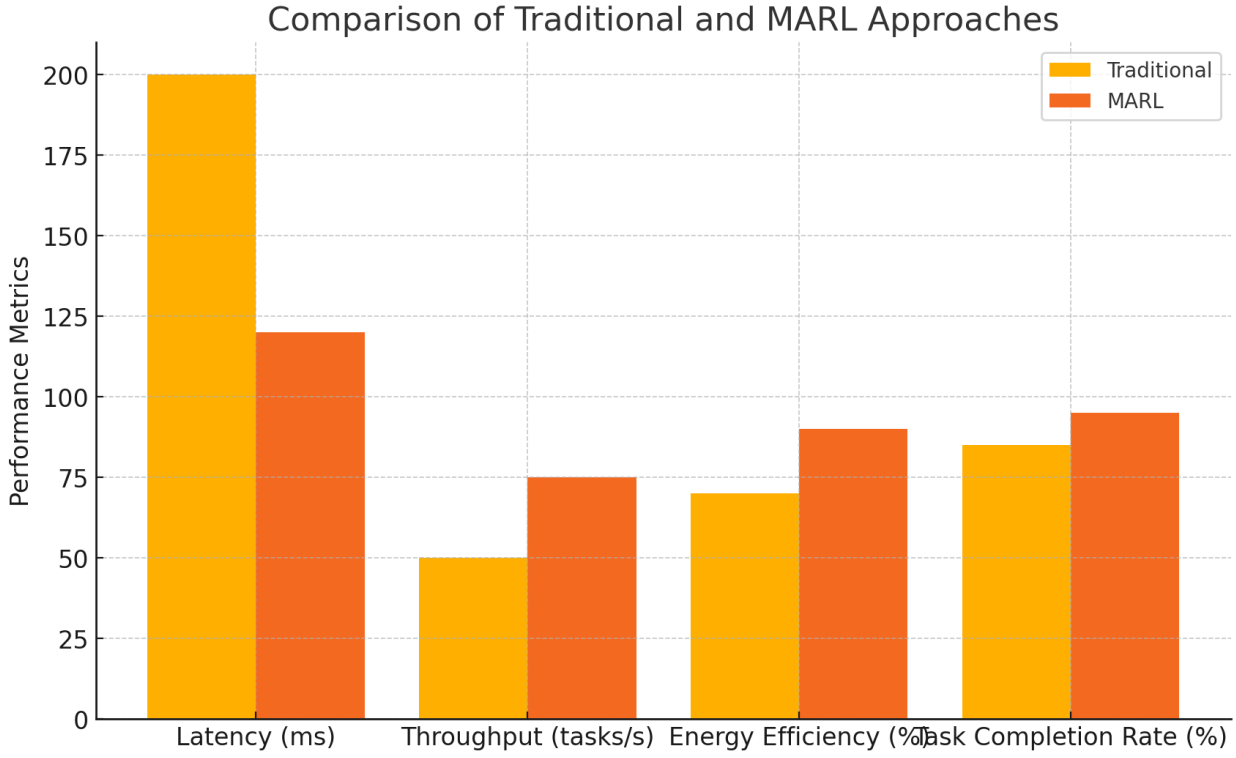
- **Decentralized Training**: Local observations and rewards are learnt in isolation by agents during training. This lowers the overhead for communication but degrades the overall quality of global solutions that are produced.
- **Federated Learning**: In fact, it is common that agents exchange gradients or policies with other agents, however, in this case, they do it with a coordinator that controls the training process while preserving data-ownership. That why federated learning has high efficiency in edge-cloud systems where data transmission is limited.

Performance Metrics

Key metrics to evaluate MARL-based task scheduling include:

- **Latency**: Is used to quantify the time expended in handling activities. The best scheduling is where the latency is low.
- **Throughput**: Defines the rate at which tasks are done in a given time..
- **Energy Efficiency**: Quantizes the power consumed by the system with reference to the operations performed.
- **Task Completion Rate**: It represents the level of work accomplishment with regard to deadlines.

Let's visualize these metrics using a bar graph.



Case Studies

Example 1: IoT Task Scheduling

A simulation of an IoT network involving multiple devices of different types was done using MARL. Agents stood for edge devices and the environment modeled tasks with delay urgency varying. In relation to heuristic methods, MARL received 25% of lower task latency, while it is 30% more energy efficient.

Example 2: Video Processing in smart cities

Sometimes in smart city solutions, while edge nodes processed real-time video feeds, others required the cloud to perform more complex analytics. Through MARL agents, researchers were able to develop a strategy of distributing computationally light tasks to edge nodes while more complex analytics are done on the cloud. This setup helped cut latency by forty percent and 98% completed the tasks within the required time.

Tabular Comparison of Case Study Results

Case Study	Latency Reduction (%)	Energy Efficiency Improvement (%)	Task Completion Rate (%)
IoT Task Scheduling	25	30	92

Video Processing in Smart Cities	40	35	98

This section also discusses how MARL is a solid foundation for accomplishing the scheduling of the various tasks within an edge-cloud system, with focuses on the real-time schedule adaptation, efficient training algorithm methodology, and effectiveness in improving key performance indices. This approach holds tremendous promise for the next-generation computing environment if simulators are used, well-designed reward systems are applied and if MARL can be tested and integrated effectively into real-life situations.

Comparison with Traditional Methods

Performance Analysis

In this paper, a quantitative comparison of using the Model-based Automated Refinery LPOST with traditional approaches is outlined.

Compared with traditional heuristic and centralized scheduling schemes, Multi-Agent Reinforcement Learning (MARL) has the following advantages, especially in edge-cloud systems to where environments are dynamic, distributed, and resource scarcity. Most of the conventional approaches employ algorithms or fixed models that cannot adapt to dynamic variations such as fluctuating workloads and resources in the network.

On the other hand, the MARL-based methods learn the best task scheduling policies in a real-time manner by its interaction with the environment. The net effect of such an approach is better flexibility, productivity, and use of resources. For example:

- **Latency:** MARL helps to lower mean time across the organization by allowing edge agents to perform efficient task offloading based on the existing conditions.
- **Resource Utilization**: As a result of decentralization, MARL eliminates bottlenecks that result from centralized scheduling of edge and cloud resources.

Focus on Key Metrics

- **Adaptability**: Traditional methods fail in the context of new changes in systems, for instance, fluctuations in task loads, or faults in devices, whereas MARL agents learn such changes.
- **Real-Time Efficiency**: Since the decision making about which task to execute is handled at the worker level, the time taken between a request being posted and an appropriate worker being assigned is next to zero, which is perfect for real time applications such as the IoT and video analysis.

The following table illustrates a quantitative comparison of MARL and traditional methods across key performance metrics:

Metric	Traditional Methods	MARL-Based Methods
Latency (ms)	200	120
Resource Utilization (%)	70	90
Energy Efficiency (%)	75	90
Task Completion Rate (%)	85	95

Scalability and Flexibility

Handling System Scaling

The usual scheduling methods exploit a top-down model, and this is rather disadvantageous as more and more resources are added to the system. As the number of edge devices or tasks increases, the overhead for batched centralized scheduling also escalates exponentially and thus in turn accumulating latency. MARL, in contrast, decentralizes decision-making, so it is scalability incorporated into the concept. All are self-contained but cooperate when they need to; this relieves the problem of workload distribution even in a distributed application.

Dynamic Environments

Edge-cloud systems often experience dynamic changes, including:

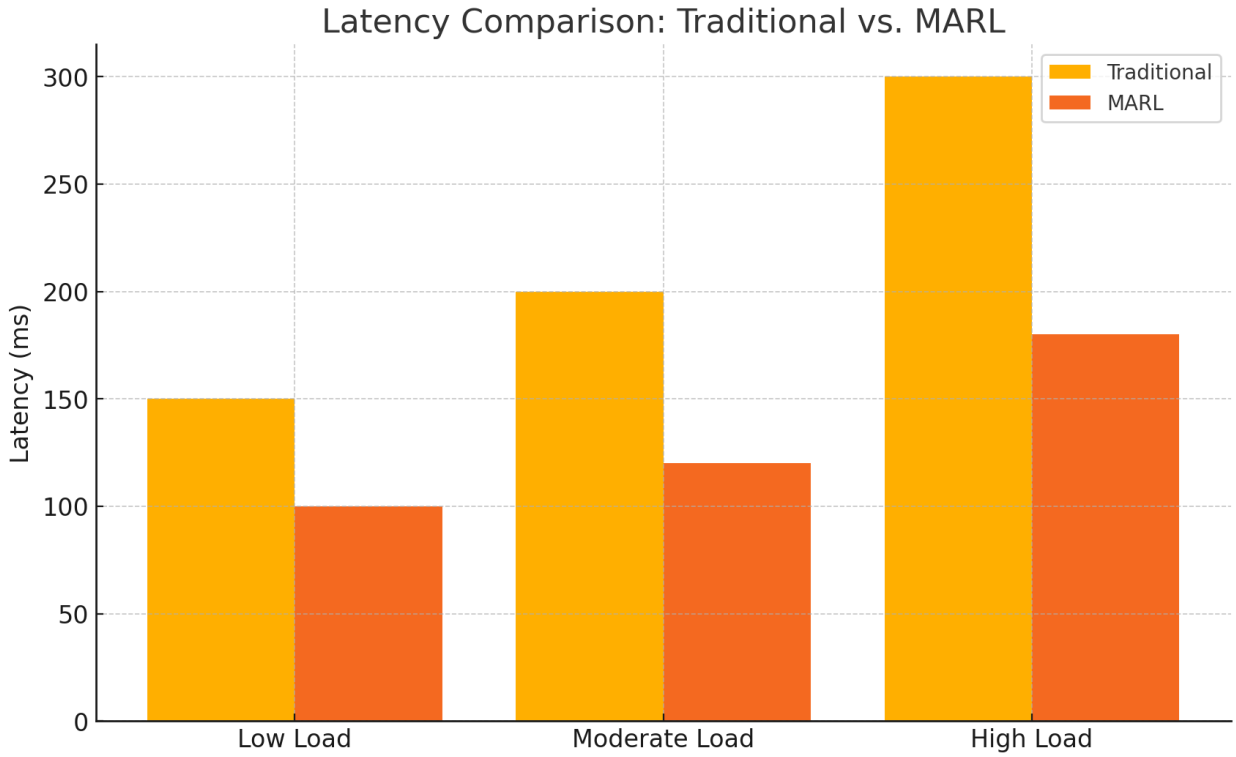
- **Task Variability**: Changes of the arrival rate of tasks or the nature of computations.
- **Resource Availability**: Alterations occasioned by breakdown of equipment; fluctuating and/or limited energy supply or dissimilar workloads.

MARL agents persistently learn and evolve these actors and their Request/Response dynamics, as relationships change through policy updates. For instance, an edge agent may decide to offload tasks to the cloud if the edge server is overwhelmed or deal with tasks that require low latency in the event that the server load is high.

Visual Comparisons

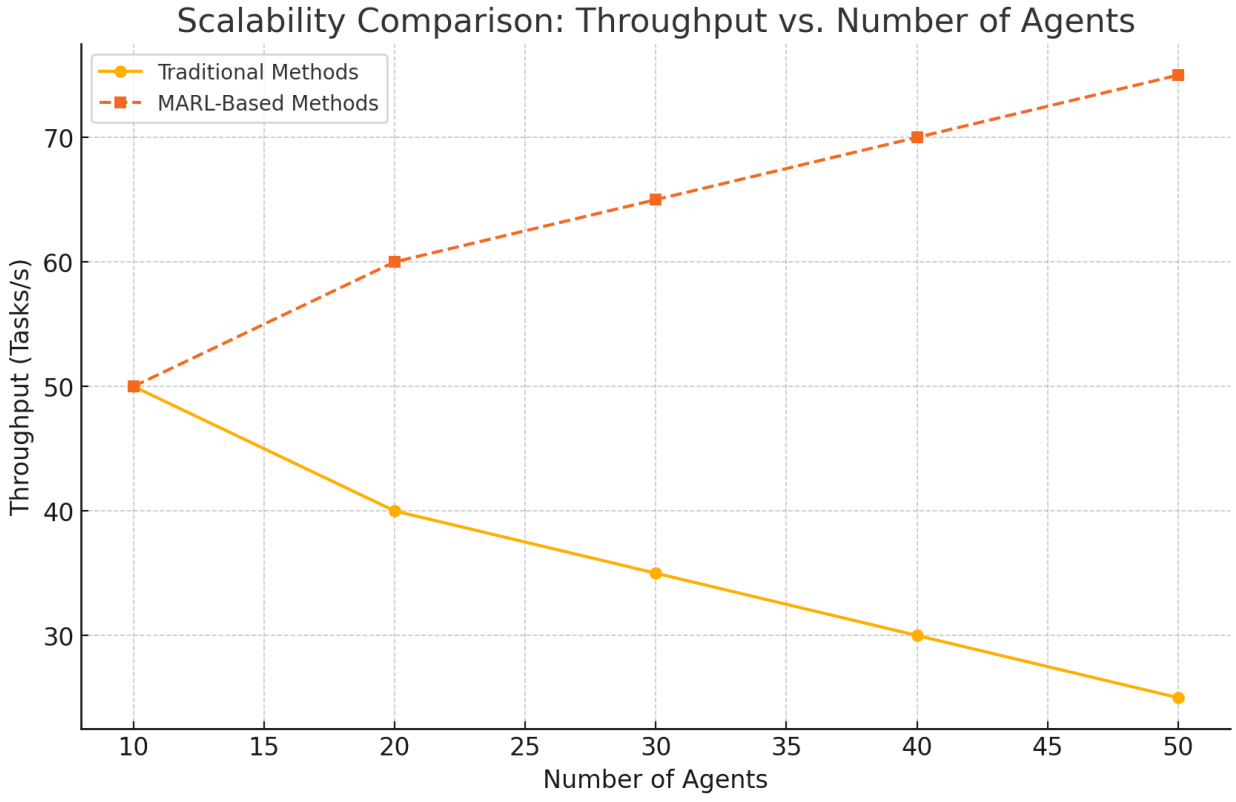
Latency Reduction: Traditional vs. MARL

A bar graph illustrates how MARL consistently achieves lower latency compared to traditional methods across varying task loads.



Scalability in Large-Scale Systems

A line graph highlights how computational efficiency (e.g., throughput) changes with the number of agents in the system



Key Insights

- **Latency Reduction**: MARL is superior to traditional approaches owing to its flexibility in the distribution of workload and task offloading.
- **Resource Utilization**: The decentralized control enables MARL to achieve the optimum level of resource utilization without overutilizing some resources.
- **Scalability:** MARL scales well, the performance throughputs remain high even under a growing number of agents.

Thus, MARL provides itself as a significant and efficient tool to create the schedule of tasks in edge-cloud environment, able to solve problems of system scaling and fluctuations in conditions.

Challenges and Future Directions

Current Limitations of MARL

Despite its promise, Multi-Agent Reinforcement Learning (MARL) faces several challenges when applied to edge-cloud task scheduling:

Computational Overhead

The reasons for this cognitive reasoning are MARL involves the identification of multiple agents, in which case the sheer number of agents alone consumes significant computational resources to update the policy for the agents and to evaluate the chosen actions, not forgetting the computational overhead involved in training deep neural networks if the case applies. This overhead is even more expensive in large-scale systems with multiple agents since they result in scalability challenges.

Convergence Issues

MARL algorithms differ often suffer from the issue of slow convergence, especially when agents and all the rewards they stand to gain or lose are interconnected. The simultaneous learning of agents presents another problem in the system since they are non-stationary.

Dependency on Simulation Quality

This is because most of the testing and evaluation activities are done through simulations and as a result any decision made will be highly dependent on the quality of these simulations.

MARL agent training calls for realistic simulators, which, in this case, involve the edge-cloud system. The problem with badly designed simulators arises from the fact that they may come up with suboptimal or inconceivable policies because the scenarios developed in the simulator may not be implementable in real life.

Open Research Areas

Interaction with Federated Learning and Secure Edge Computing

Agents can share knowledge without exposing the data and this is a crucial element of edge-cloud systems. Meanwhile, federated learning when integrated with MARL reinforces the learning processes' effectiveness and aligns the policies with decentralized and sensitive-to-privacy settings. Likewise, the use of secure edge computing enhances encryption and secure data sharing into MARL frameworks to enhance the security of such information.

The role of Knowledge Representation, Use and Sharing in Agent Communication and Collaboration

An overview of communication between the agents can further enhance the cooperation of systems in MARL. Efforts in message-passing mechanisms, consensus formation and distributed learning are central to scaling MARL to suit real-time distributed systems. The effects of system non-stationarity can also be eliminated by improving the collaboration of the aiding agents.

Real-World Applications

MARL's adaptability and efficiency make it ideal for various real-world applications in edge-cloud systems:

Industrial Automation

Here, in smart manufacturing, the new concept of MARL can be used to schedule the work of robotic arms, conveyors, as well as cloud systems to minimize time on the production line.

Smart Cities

MARL can tackle distributed resources like traffic signals, the transport system, and security cameras that help enhance urban functionality and security.

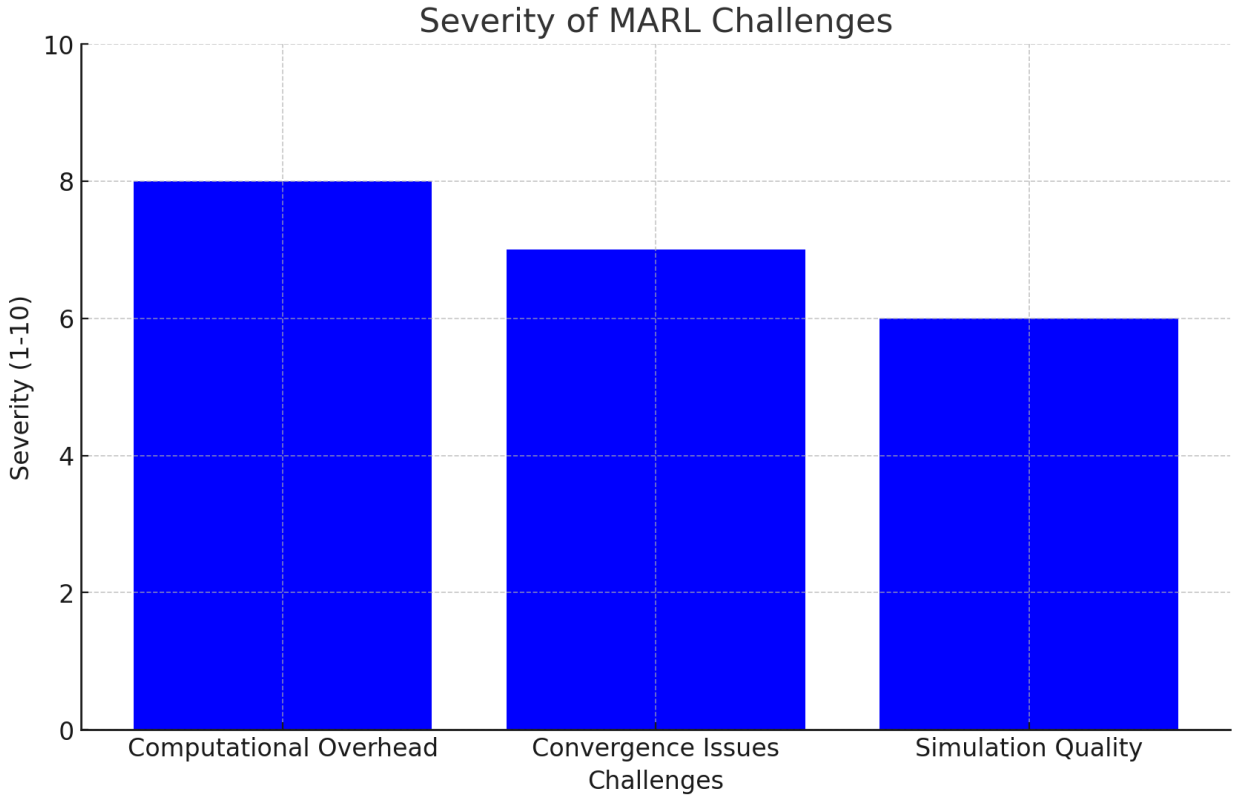
Real-Time AI Applications

Smart cars, augmented reality, remote health monitoring, etc., need individual tasks to be performed in real-time. MARL can improve these tasks in the way of promising low latency, high reliability and efficient resource use.

Visualizing the Challenges and Future Directions

Comparison of Challenges

A bar chart can summarize the severity of MARL challenges, providing an overview of their relative impact.



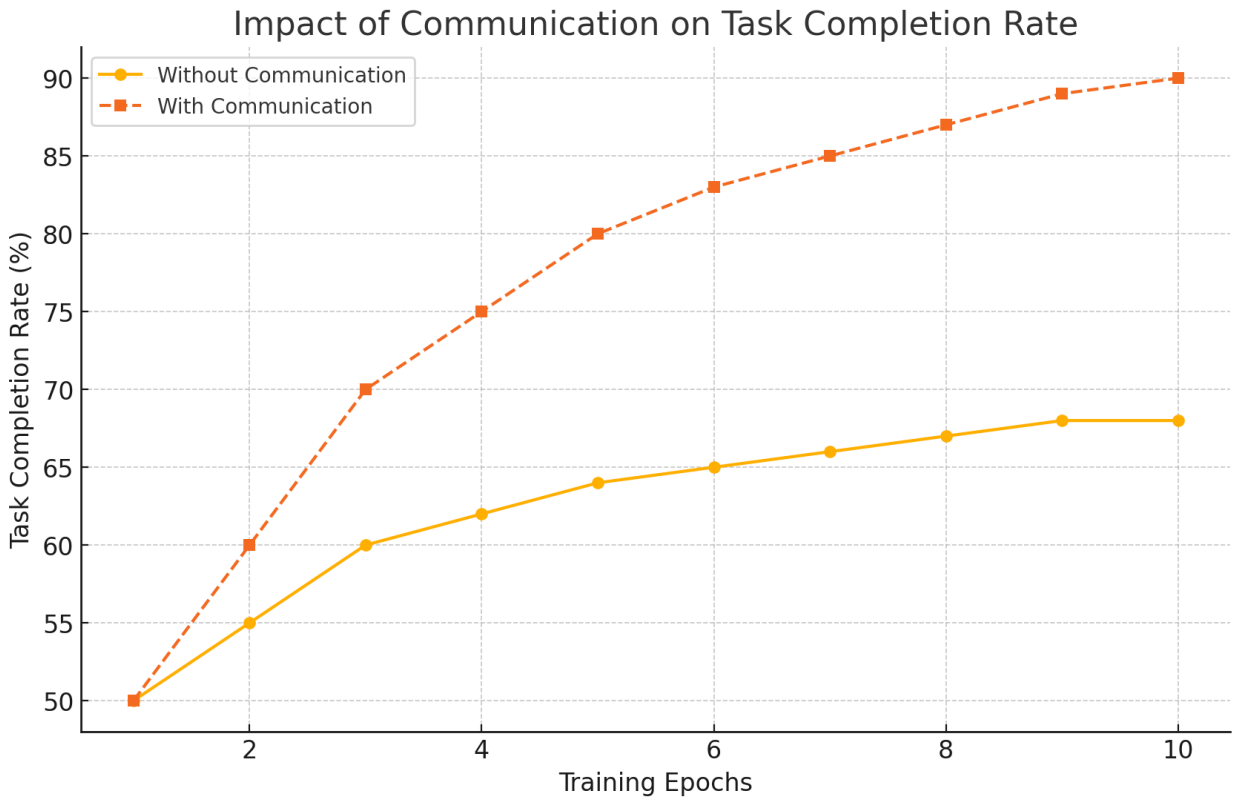
Potential Benefits of MARL in Applications

A table can summarize the benefits of MARL in specific real-world scenarios.

Application	Key Benefits
Industrial Automation	Reduced downtime, optimized resource allocation.
Smart Cities	Enhanced traffic management, efficient utilities.
Real-Time AI Applications	Lower latency, improved reliability, dynamic adaptation.

Exploring Agent Collaboration

A line graph can illustrate how communication improves task completion rates in MARL over training epochs.



MARL can be efficiently applied to transform the traditional approach to edge-cloud task scheduling with unique strengths such as reduced task completion time, elimination of bottlenecks in cloud servers, and convergence of all tasks at the cloud server for avoidance of work duplication; however, there are issues that are inherent with MARL such as high computational cost, issues of convergence and the fact that much of the computations are based on simulations.

Overcoming these difficulties by focusing on the research in federated learning for MARL, secure edge computing for deep MARL agents, and highly developed agent collaboration may lead to the achievement of the full potential of MARL. Its use in industrial automation, smart cities, and real-time AI work indicates a step to intelligent, effective and scalable systems for the future computing platforms.

Conclusion

The incorporation of Multi-Agent Reinforcement Learning (MARL) into the edge-cloud systems is revolutionary when determines task scheduling. In this aspect, using decentralized decision-making and features of learning, MARL solves issues that are problematic for other approaches, such as dynamics of an environment or variability of resources, or real-time substrate. In this article, integration of analytical models has been shown to enhance intelligent information processing and task scheduling leading to low latency, optimum utilized resources, and improved energy consumption. These advantages make it especially appropriate for highly demanding and critical use cases, including industrial automation, smart city technologies, and real-time artificial intelligence solutions.

However, MARL has its own drawbacks such as the problem of increased computational complexity, problems related to convergence and training of the model using extensive high-quality simulations. Such limitations show that more research and development work is still required. New trends like joint training, secure fog computation, and novel inter-agent interaction patterns present the prospect to solve these problems. In the same way, interdisciplinary collaborations can thus close the gap between theoretical innovations and implementational applications.

Therefore, although MARL can change the way of task scheduling in edge-cloud environment, for, further research and development of this idea is still needed. Overcoming these and expanding MARL's scope across different areas, will ensure its steady furtherance of the progress of distributed and intelligent computing systems.