Resource Allotment Utilizing Multi-Armed Bandit Fostered Reinforcement Learning in Mobile Edge Computing Ecosystems

Priyabrata Nayak[1][a], Dipti Dash[1][b], Sapthak Mohajon Turjya[1][c] and Anjan Bandyopadhyay[2][d]
[1] *Kainga Institute of Industrial Technology University, Bhubaneswar, 751024, India*
[2] *Department of Computing Science Engineering, India*

Keywords: Mobile Edge Computing, Multi-Armed Bandit, Reinforcement Learning, Resource Allotment, Upper-Confidence Bound, Social Welfare.

Abstract: Mobile Edge Computing (MEC) leverages the nearness of computational elements to end-users in wireless networks, pledging low latency and elevated throughput for emerging mobile utilities. Nevertheless, efficient resource allotment remains a notable challenge in MEC ecosystems due to the varying and heterogeneous character of mobile networks. Classic static resource allotment strategies often fail to adjust to varying network conditions, showing suboptimal allotments. In this paper, we propose a novel strategy for resource allotment in MEC ecosystems utilizing a Multi-Armed Bandit (MAB) based Reinforcement Learning (RL) approach. By viewing the resource allotment problem as an MAB problem, our strategy enables the dynamic assignment of resources founded on real-time feedback, thereby enhancing resource usage and user satisfaction. We offer a comprehensive evaluation of our technique through simulations in diverse MEC scenarios, which includes a comprehensive comparison with the round robin task scheduling algorithm to represent the efficacy of our proposed methodology, exhibiting its efficacy in acclimating to changing network conditions and surpassing traditional static allocation procedures. Thereby, our results showcase the prospect of MAB-based RL strategies in improving resource administration in MEC ecosystems, curving the path for better adaptive and productive mobile edge computing applications.

1 INTRODUCTION

Mobile Edge Computing (MEC) has appeared as a revolutionary paradigm in wireless communication systems, striving to provide low-latency, high-throughput services to mobile users by integrating computational elements at the network edge (Wang et al., 2023b). With the expansion of latency-prone utilities like augmented reality, autonomous automobiles, and Internet of Things (IoT) gadgets, the need for efficient resource allotment in MEC ecosystems has evolved paramount. Nevertheless, conventional static allotment strategies often fail to dynamically adjust to the evolving network requirements and user needs, leading to suboptimal resource usage and tarnished quality of service (QoS).

Considering an augmented reality (AR) (Ateya et al., 2023) utility where users interact with virtual entities superimposed onto the real-world ecosystem in real time. In such a scenario, the latency/delay between user communication and the processing of virtual entities performs a crucial role in delivering a flawless and immersive venture. Static allotment of computational elements may lead to erratic performance, as the need for resources varies dynamically with user mobility and utility needs (Gong et al., 2024). Similarly, in the context of automobiles, which laboriously depend on real-time data processing for direction and decision-making, productive resource allotment evolves critically(Ray and Banerjee, 2024). Considering a scenario, during the time of day when the traffic is at its peak, the processing requirement for the data generated by various sensors functioning with complicated algorithms changes with the traffic variance of different regions. Resource allotment with static allotment procedures will result in misutilization of resources in areas where crowd accumulations are minimal, and uplifting latency issues in crowded regions, which may result in security breaches and

[a] https://orcid.org/0009-0002-9069-3420
[b] https://orcid.org/0000-0002-4625-2727
[c] https://orcid.org/0009-0004-5190-0305
[d] https://orcid.org/0000-0001-7670-2269

Nayak, P., Dash, D., Turjya, S. M. and Bandyopadhyay, A.
Resource Allotment Utilizing Multi-Armed Bandit Fostered Reinforcement Learning in Mobile Edge Computing Ecosystems.
DOI: 10.5220/0013335900004646

In *Proceedings of the 1st International Conference on Cognitive & Cloud Computing (IC3Com 2024)*, pages 106-114
ISBN: 978-989-758-739-9

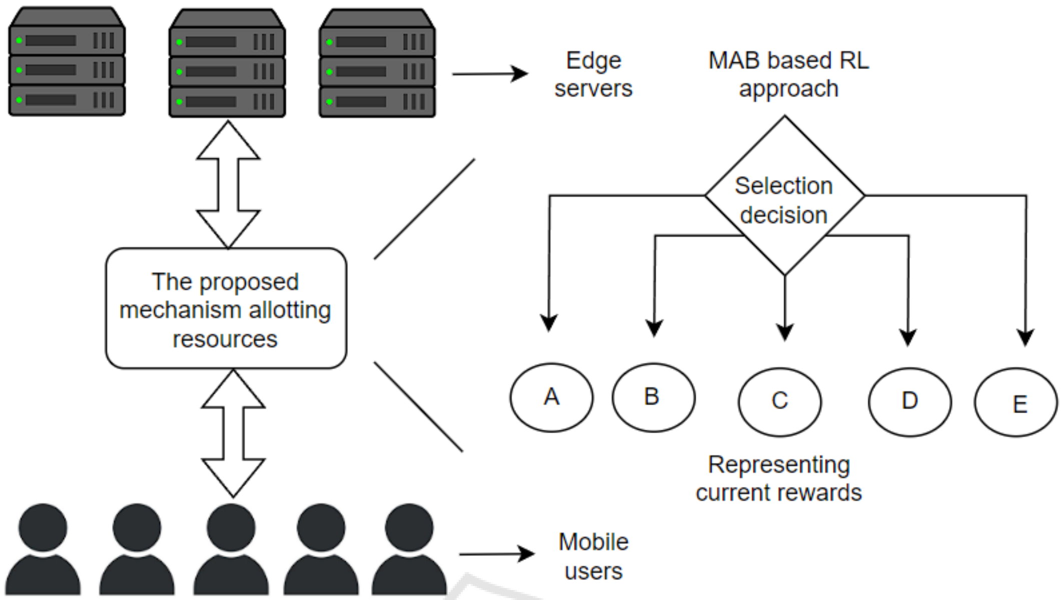

Figure 1: The MAB fostered RL approach in mobile edge computing ecosystems

lessening of reliability with autonomous driving vehicles (Gong et al., 2024).

Following up on the IoT domain, where the employed devices produce data in variable quantities requires special attention for processing and extracting useful information. Productive resource allotment is paramount for efficient processing and optimal utilization of computational resources (Wang et al., 2023a). Citing an example, in smart cities where integrated sensors capture various data like air pollution levels, traffic movement/hotspots, and energy usage, therefore the employed resource allotment procedures must be accustomed to the rapid variance of data produced along with their dynamic processing demands. To handle these dynamic issues, there has been a growing interest in the enactment of adjustable resource allotment procedures that would function in dynamic MEC ecosystems. The proposed mechanism must be able to consider the user requirement and real-time data feedback to allot computational resources. Thereby, this research paper focuses on introducing and enacting a Multi Arm Bandit (MAB) (Simchi-Levi and Wang, 2023) fostered Reinforcement Learning (RL) techniques for allotting resources in MEC ecosystems (Preil and Krapp, 2023) (fig.1). Here the resource allotment problem is formulated as an MAB problem, and the proposed technique enables diligent decision enactment for allotting resources, thereby improving resource utilization, lessening latency, and enhancing the satisfaction of the users.

2 RELATED WORKS

Vast research has been carried out in the domain of Mobile Edge Computing ecosystems focusing on resource allotment, which has examined various techniques to enhance system productivity/performance while functioning on dynamic edge ecosystems (Chi et al., 2023). Conventional allotment strategies like Robin-Robin (Zhou et al., 2023), which is simpler, but yet suffer when exposed to varying network and resource needs, thereby resulting in a substantial system performance drop. To tackle these inefficiencies, resource allotment procedures that perform dynamically have grown increasingly popular. RL strategies like the MAB algorithms enable dynamic allotment of resources founded upon the input, which is given in real-time, thereby improving resource usage and enhancing system performance (Galli et al., 2023).

Researchers have worked on strategies like multi-objective enhancement to handle various aspects like throughput, latency/delay, and enable efficient consumption of energy (Dehghani and Movahhedinia, 2023). These strategies employ generic procedures, evolutionary procedures, and Pareto enhancement techniques to address the various sophisticated trade-offs contained in MEC resource allotment situations (Khan et al., 2024).

Utilization of autonomous procedures and coinciding it with edge brilliance was stood up as a noteworthy solution for resource allotment in edge ecosystems. Approaches like edge-catching and federated learning provide the facility of dispersed decision enactment for variable resource allotment at the network edge. By utilizing edge intelligence and automation, MEC systems can dynamically adjust to varying conditions, improving resource allotment in real time and enriching overall system productivity and performance (Kar et al., 2023).

Likewise, real-world integrations and case analyses provide valuable understandings of the practical feasibility and efficacy of resource allotment algorithms in myriad MEC ecosystems, comprising smart cities, industrial IoT, and vehicular networks. These analyses close the gap between theoretical improvements and real-world applications, validating the effectiveness of resource allotment procedures and advising future research pathways in Mobile Edge Computing.

3 THE PROPOSED SYSTEM MODEL

Considering a Mobile Edge Computing (MEC) ecosystem comprising of a group of N edge servers represented by $SR = \{sr_1, sr_2, \ldots, sr_N\}$, serving a set of M mobile users represented by $UR = \{ur_1, ur_2, \ldots, ur_M\}$. The objective is to dynamically allot computational elements to mobile users to reduce latency and increase system throughput while guaranteeing efficient resource usage.

We model the resource allotment scenario as a Multi-Armed Bandit (MAB) problem, where every edge server sr_j serves as an arm and the available computational elements at each server represent the arms' reward allotments. Let $RW_j(t)$ represent the reward received from allotting resources at server sr_j at time t, where $RW_j(t)$ is a random variable with an unknown allotment.

At every time step t, the MEC regulator chooses an edge server founded on a policy that balances inquiry (trying out various servers to learn their reward allotments) and exploitation (allotting resources to servers with potentially higher rewards founded on current knowledge). Allowing $AC(t)$ indicate the step taken by the regulator at time t, where $AC(t) \in \{1, 2, ..., N\}$ illustrates the chosen edge server index.

The goal is to maximize the cumulative reward over T time steps:

$$\max_{\pi} \sum_{t=1}^{T} RW_{AC(t)}(t) \tag{1}$$

in the above equation π denotes the utilized policy for selecting actions.

In order to perform this, we utilize the Upper Confidence Bound (UCB) algorithm coming under RL strategies, which functions by balancing inquiry and exploitation. The algorithm utilizes the upper confidence limit expected rewards to select the actions, enabling the identification of novel/new arms while utilizing the older high-reward arms.

4 THE PROPOSED MECHANISM

Now, we provide the algorithmic design for the proposed Multi-Arm Bandit (MAB) fostered Reinforcement Learning (RL) procedure for resource allotment in MEC ecosystems dynamically. The proposed procedure strives to intelligently select the intended edge servers, and allot them to the mobile edge users efficiently, which in turn will improve the latency, resource usage, and throughput, thereby enhancing the user's overall system performance.

4.1 Action Choosing

- The Upper Confidence Bound (UCB) procedure is used by the MEC regulator to choose the intended edge servers at each time step t. The below-given equation is utilized to calculate the action $AC(t)$:

$$AC(t) = \arg\max_{i} \left(\hat{RW}_j(t) + \sqrt{\frac{2\log(t)}{N_j(t)}} \right) \tag{2}$$

where,

- Up and until time t, the estimated average reward for server sr_j is represented by $\hat{RW}_j(t)$.
- Up and until time t, the number of times edge server sr_j was chosen is depicted by $N_j(t)$.
- The present time step is represented by t

4.2 The Reward Calculation

- At time t, utilizing the calculated average reward $\hat{RW}_j(t)$ for server sr_j the below given equation estimates/adjusts the sample mean.

$$\hat{RW}_j(t) = \frac{\sum_{\tau=1}^{t} RW_j(\tau) \cdot \mathbb{I}(AC(\tau) = j)}{N_j(t)} \tag{3}$$

where,

- $RW_j(\tau)$ is the perceived reward received from server sr_j at time τ.
- $\mathbb{I}(AC(\tau) = j)$ is an indicator function that becomes 1 if server sr_j was chosen at time τ, and 0 otherwise.
- $N_j(t)$ depicts the number of times server sr_j has been chosen up to time t.

4.3 The Proposed Algorithm

Algorithm 1 MAB-RL for Dynamic Resource Allotment in MEC.

Initialize estimated rewards $\hat{RW}_j(0)$ and selection counts $N_j(0)$ for all servers sr_j.
for time step $t = 1, 2, \ldots, T$ **do**
  Select action $AC(t)$ using UCB algorithm:
  $AC(t) = \arg\max_j \left(\hat{RW}_j(t-1) + \sqrt{\frac{2\log(t)}{N_j(t-1)}} \right)$
  Execute action $AC(t)$ and observe reward $RW_{AC(t)}(t)$.
  Update estimated mean reward $\hat{RW}_{AC(t)}(t)$:
  $\hat{RW}_{AC(t)}(t) = \frac{\sum_{\tau=1}^{t} RW_{AC(t)}(\tau) \cdot \mathbb{I}(AC(\tau)=AC(t))}{N_{AC(t)}(t)}$
  Increment selection count $N_{AC(t)}(t)$: $N_{AC(t)}(t) = N_{AC(t)}(t-1) + 1$
end for

The algorithm initializes the calculated rewards and choosing counts for all servers. At each time step, it chooses an action (edge server) founded on the UCB algorithm. It accomplishes the selected action, observes the reward, and updates the calculated mean reward for the selected server. Finally, it increments the choice count for the chosen server. The process continues until the specified time horizon T is attained. This algorithm enables dynamic resource allotment in MEC ecosystems by adaptively choosing edge servers founded on real-time feedback, thus improving system performance.

4.4 Algorithm Properties

Proposition 1: The UCB algorithm guarantees that the calculated mean reward converges to the true mean reward with high probability.

Proof: Let $\hat{RW}_j(t)$ denote the calculated mean reward for server sr_j at time t, and $RW_j(t)$ represent the true mean reward. By Hoeffding's inequality, we have:

$$\text{Prb}\left(\hat{RW}_j(t) - RW_i(t) > \varepsilon\right) \leq e^{-2\varepsilon^2 N_j(t)} \tag{4}$$

where, $\varepsilon > 0$ is a constant depicting the deviation from the true mean reward, and $N_j(t)$ is the number of times server sr_j has been chosen up to time t.

To confirm that $\hat{RW}_j(t)$ is near to $RW_j(t)$ with high probability, we set $\varepsilon = \sqrt{\frac{\log(t)}{N(t)}}$. Then, we have:

$$\text{Prb}\left(\hat{RW}_j(t) - RW_j(t) > \sqrt{\frac{\log(t)}{N_j(t)}}\right) \leq \frac{1}{t^2} \tag{5}$$

By the union bound, the probability that any server sr_j varies from its true mean reward declines exponentially with time. Therefore, the UCB algorithm guarantees convergence to the true mean reward for all servers equipped with high probability.

Proposition 2: Regret Bound for UCB Algorithm The regret of the UCB algorithm is determined by $O(G\log(T))$, where G is the servers present and T is the time horizon.

Proof: Let RW^* represent the maximum mean reward among all servers. The regret at time T is described as:

$$\text{Regret}(T) = T \cdot RW^* - \sum_{j=1}^{G} \mathbb{E}\left[\sum_{t=1}^{T} RW_j(t)\right] \tag{6}$$

By the UCB algorithm's investigation-exploitation trade-off, the envisioned cumulative reward of the algorithm satisfies:

$$\mathbb{E}\left[\sum_{t=1}^{T} RW_{AC(t)}(t)\right] \geq \sum_{j=1}^{G} \mathbb{E}\left[\sum_{t=1}^{T} RW_j(t)\right] - \sum_{j=1}^{G} \sqrt{\frac{2\log(T)}{N_j(T)}} \tag{7}$$

Using Proposition 1, we have:

$$\sum_{j=1}^{G} \sqrt{\frac{2\log(T)}{N_j(T)}} = O(\sqrt{G\log(T)}) \tag{8}$$

Thus, the regret of the UCB algorithm is determined by $O(G\log(T))$, demonstrating sublinear regret growth with relation to the time horizon T.

These theorems deliver theoretical assurances for the convergence and productivity of the UCB algorithm in dynamic resource allotment procedures.

4.5 Time Complexity Analysis

The time complexity study of the suggested Multi-Armed Bandit (MAB) based Reinforcement Learning (RL) mechanism for dynamic resource allotment in Mobile Edge Computing (MEC) ecosystems concerns evaluating the computational cost of key processes within the algorithm.

UCB algorithm based action choosing: The time complexity of choosing an action utilizing the UCB algorithm relies on computing the upper confidence bounds for each arm (edge server). For G servers and

T time steps, the time complexity for action choosing is $O(G)$.

Reward Calculation: Revising the calculated mean reward for each server concerns calculating the sample average of observed rewards. For every time step t, the time complexity of revising the calculated mean reward for G servers is $O(G)$.

Overall Time Complexity: The overall time complexity of the suggested strategy can be approximated as $O(G)$, where G is the number of servers (arms) and T is the time horizon. This complexity results from making action decisions and evaluating rewards at each time step.

Regret Bound Analysis: The regret bound involves calculating the cumulative regret across T time steps. Regret increases sublinearly with T (limited by $O(G\log(T))$), resulting in a modest processing cost relative to the major operations of action selection and reward computation.

The suggested approach for dynamic resource allocation in MEC ecosystems is highly impacted by the number of edge servers (G) and the time period (T). The algorithm's complexity scales linearly with both G and T, making it computationally feasible for practical usages in real-time MEC procedures. Furthermore, the regret-bound investigation demonstrates the algorithm's efficacy in having sublinear regret growth with respect to time, further backing its scalability and usefulness.

4.6 Scenario Analysis

Considering a scenario where there are 3 edge servers (arms) represented by sr_1, sr_2, and sr_3, providing service to a group of mobile users. The objective is to dynamically allot computational elements to reduce latency and increase throughput.

Step 1. Initialization:

- At first, the algorithm initializes the estimated mean rewards $\left(R\hat{W}_j(t)\right.$) and choice counts ($N_j(t)$) for each server sr_j.
- Let's assume the initial estimated mean rewards are:
  - $R\hat{W}_1(0) = 0.5$
  - $R\hat{W}_2(0) = 0.3$
  - $R\hat{W}_3(0) = 0.2$
- And the initial choice counts are all set to 0 : $N_1(0) = N_2(0) = N_3(0) = 0$.

Step 2. Action Choosing (Time Step 1-5):

- At each time step, the algorithm chooses an action (server) utilizing the UCB algorithm.
- For instance, at time step 1, the UCB algorithm chooses server sr_1 as it has the maximum upper confidence bound.
- Likewise, actions are chosen for time steps 2 to 5 founded on the UCB algorithm.

Step 3. Reward Observation and Computation:

- After choosing an action, the algorithm performs it, notices the reward, and revises the calculated mean reward for the chosen server.
- For example, at time step 1, if server sr_1 obtains a reward of 0.6, the calculated mean reward $R\hat{W}_1(t)$ is updated consequently.

Step 4. Steps Repetition 2-3 (Time Step 6-10):

- The procedure of action selection, reward observance, and computation persists for subsequent time steps.

Step 5. Performance Evaluation:

- After a certain number of time steps, the algorithm's productivity is assessed in concerning cumulative reward, regret, and convergence to the true mean reward.

By iteratively revising the estimated mean rewards and choosing actions founded on the UCB algorithm, the algorithm adjusts to the various ecosystem dynamics and enhances resource allotment to improve system productivity. In real-world cases, the algorithm would run continually, dynamically modifying resource allotment founded on real-time feedback and user needs.

5 SOCIAL WELFARE OF THE PROPOSED MECHANISM

In the context of resource allotment in Mobile Edge Computing (MEC) ecosystems, social welfare can be described as a metric that captures the overall benefit emanated from the allotment of computational utilities to mobile users. Social welfare regards not only the individual utilities of users but also the system-wide purposes such as lowering latency, increasing throughput, and improving resource usage.

Social welfare (sW) can be represented as the summation of individual utilities (UT_j) of all mobile users (UT):

$$SW = \sum_{j=1}^{M} UT_j \tag{9}$$

where:

- M is the total number of mobile users.

- UT_j denotes the utility acquired by user j.

The individual utility (UT_j) can be determined based on diverse factors such as latency, throughput, quality of service (QoS), energy usage, or any other suitable performance metric. For example, in the context of MEC, the utility of a user may rely on the latency experienced in completing a task, the quantity of computational resources allotted, and the trustworthiness of the service delivered.

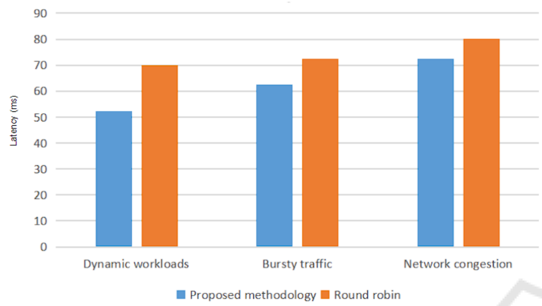

Figure 2: Comparison between the proposed and round robin strategy in handling latency (ms) between dynamic workloads, bursty traffic, and network congestion

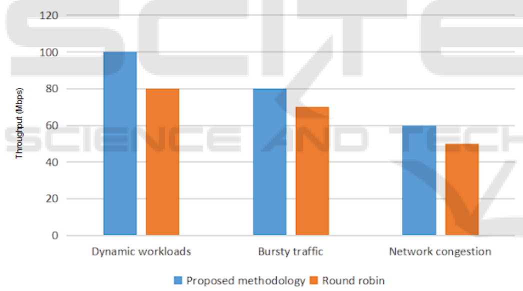

Figure 3: Impact of the proposed and round robin strategy in managing system throughput (Mbps) between dynamic workloads, bursty traffic, and network congestion

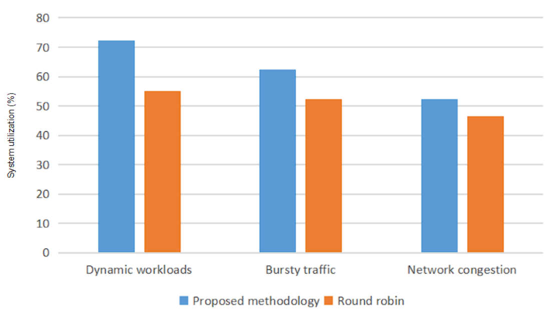

Figure 4: Impact of the proposed and round robin strategy in managing system utilization (%) between dynamic workloads, bursty traffic, and network congestion

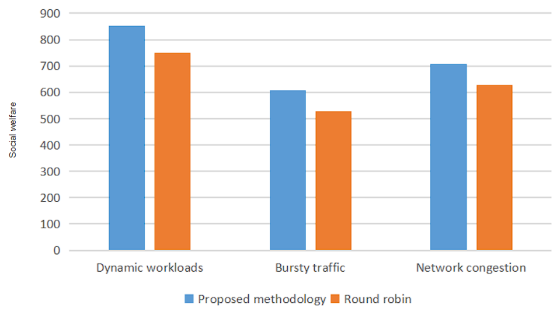

Figure 5: Impact of the proposed and round robin strategy in achieving social welfare between dynamic workloads, bursty traffic, and network congestion

6 THE SIMULATION RESULTS

6.1 Simulation setup

Our research activity traverses across three stem domains—dynamic workloads, bursty traffic, and network congestion—to evaluate the efficacy of our suggested methodology in a simulation ecosystem. For comparison purposes, we employed the round-robin task scheduling algorithm, which is a static scheduling algorithm that works by preempting the functioning of the executing processes at a specific time quantum and resuming them again from the ready queue to accomplish the tasks. Alongside this, parameters such as latency (ms), throughput (Mbps), resource utilization, and social welfare are estimated by allotting five mobile devices among three edge servers. This setup assures clarity while retaining applicability to real-world designs.

6.2 Results

In Table 1, we demonstrate the allotment of 5 mobile devices among 3 edge servers utilizing the proposed methodology to estimate latency (ms), throughput (Mbps), resource utilization (%), and social welfare in terms of dynamic workloads. Similarly, in Table 2 and Table 3, we execute the same investigation for bursty traffic and network congestion scenarios respectively utilizing the proposed methodology. The same approach is performed for the round-robin procedure, but for the sake of simplicity, it is not included in this research paper, thereby only the mean values are reported.

Utilizing the above tables in Tables 4 and 5, we introduce the mean latency, mean throughput, mean resource usage, and mean social welfare for manag-

Table 1: Dynamic Workload management via the proposed methodology for allotting edge devices to mobile users. ES:Edge Server,LT:Latency (ms),TP:Throughput (Mbps), RU:Resource Utilization (%),SW:Social Welfare

Mobile Device	ES	LT	TP	RU	SW
Device 1	Server 1	50	100	70	850
Device 2	Server 2	60	95	65	820
Device 3	Server 3	45	105	75	870
Device 4	Server 1	55	98	80	880
Device 5	Server 2	52	102	72	840

Table 2: Bursty traffic management via the proposed methodology for allotting edge devices to mobile users. ES:Edge Server,LT:Latency (ms),TP:Throughput (Mbps), RU:Resource Utilization (%),SW:Social Welfare

Mobile Device	ES	LT	TP	RU	SW
Device 1	Server 1	60	80	60	700
Device 2	Server 2	70	75	55	680
Device 3	Server 3	55	85	65	720
Device 4	Server 1	65	78	70	730
Device 5	Server 2	62	82	62	710

ing dynamic workloads utilizing both the proposed methodology and the round-robin procedure. Similarly, in Tables 6 and 7, we supply these metrics for managing bursty traffic utilizing the suggested methodology and round-robin procedure. Eventually, in Table 8 and 9, the same investigation is performed for handling network congestion utilizing both methodologies.

Visualizing the tabular data, fig. 2 demonstrates the effectiveness of the suggested methodology in comparison to the round-robin technique in lowering latency across three domains: handling dynamic workloads, handling bursty traffic, and network congestion. Fig. 3 indicates the usefulness of the suggested procedure in improving throughput corresponding to the round-robin technique across the same domains. Similarly, Fig. 4 showcases the progress in resource utilization accomplished by the suggested methodology over round-robin across these domains. Lastly, Fig. 5 portrays the enhancement in social welfare, directing to better customer satisfaction, attained by the presented methodology over the round-robin procedure across the same domains.

7 CONCLUSION AND FUTURE WORKS

This research paper presents a newly integrated resource allotment approach for Mobile Edge Computing (MEC) utilizing Multi-Armed Bandit (MAB) and Reinforcement Learning (RL). By presenting resource allotment as an MAB problem and employing the Upper Confidence Bound (UCB) algorithm, it allows adaptive allotment of computational resources in real-time, improving performance and minimizing latency. The investigation exhibits the significance of the procedure in dynamically determining edge servers founded on real-time feedback, exceeding static allotment strategies. Future research avenues comprise upgrading dynamic MEC models, investigating multi-objective improvements, and handling security issues. Real-world integration and partnership with industry stakeholders are crucial for validation and implementation. Overall, the research donates to growing MEC systems for latency-prone utilities in the 5G era and beyond.

ACKNOWLEDGMENT

We wholeheartedly thank the Department of Computer Science Engineering of Kalinga Institute of Industrial Technology for giving us this opportunity to work in this cutting-edge field. Furthermore, we thank our esteemed professors who constantly supported us throughout this research study by reviewing our work with productive feedback and suggestions to improve further.

REFERENCES

Ateya, A. A., Muthanna, A., Koucheryavy, A., Maleh, Y., and El-Latif, A. A. A. (2023). Energy efficient offloading scheme for mec-based augmented reality system. *Cluster Computing*, 26(1):789–806.

Chi, H. R., Silva, R., Santos, D., Quevedo, J., Corujo, D., Abboud, O., Radwan, A., Hecker, A., and Aguiar, R. L. (2023). Multi-criteria dynamic service migration

Table 3: Network congestion management via the proposed methodology for allotting edge devices to mobile users. ES:Edge Server,LT:Latency (ms),TP:Throughput (Mbps), RU:Resource Utilization (%),SW:Social Welfare

Mobile Device	ES	LT	TP	RU	SW
Device 1	Server 1	70	60	50	600
Device 2	Server 2	80	55	45	580
Device 3	Server 3	65	65	55	620
Device 4	Server 1	75	58	60	630
Device 5	Server 2	72	62	52	610

Table 4: Mean values comprising all the edge servers and the mobile devices in managing dynamic workloads utilizing the proposed methodology

Metric	Mean Value
Mean Latency (ms)	52.4
Mean Throughput (Mbps)	100.0
Mean Resource Utilization (%)	72.4%
Mean Social Welfare	852

Table 5: Mean values comprising all the edge servers and the mobile devices in managing dynamic workloads utilizing the round robin methodology

Metric	Mean Value
Mean Latency (ms)	70.0
Mean Throughput (Mbps)	80.0
Mean Resource Utilization	55.0%
Mean Social Welfare	750

Table 6: Mean values comprising all the edge servers and the mobile devices in managing bursty traffic utilizing the proposed methodology

Metric	Mean Value
Mean Latency (ms)	62.4
Mean Throughput (Mbps)	80.0
Mean Resource Utilization (%)	62.4%
Mean Social Welfare	708

Table 7: Mean values comprising all the edge servers and the mobile devices in managing bursty traffic utilizing the round-robin methodology

Metric	Mean Value
Mean Latency (ms)	72.4
Mean Throughput (Mbps)	70.0
Mean Resource Utilization (%)	52.4%
Mean Social Welfare	628

Table 8: Mean values comprising all the edge servers and the mobile devices in managing network congestion utilizing the proposed methodology

Metric	Mean Value
Mean Latency (ms)	72.4
Mean Throughput (Mbps)	60.0
Mean Resource Utilization (%)	52.4%
Mean Social Welfare	608

Table 9: Mean values comprising all the edge servers and the mobile devices in managing bursty traffic utilizing the round-robin methodology

Metric	Mean Value
Mean Latency (ms)	80.3
Mean Throughput (Mbps)	50.0
Mean Resource Utilization (%)	46.4%
Mean Social Welfare	528

for ultra-large-scale edge computing networks. *IEEE Transactions on Industrial Informatics*.

Dehghani, F. and Movahhedinia, N. (2023). On the energy-delay trade-off in ccn caching strategy: a multi-objective optimization problem. *Wireless Networks*, pages 1–15.

Galli, A., Moscato, V., Romano, S. P., and Sperlí, G. (2023). Playing with a multi armed bandit to optimize resource allocation in satellite-enabled 5g networks. *IEEE Transactions on Network and Service Management*.

Gong, Y., Huang, J., Liu, B., Xu, J., Wu, B., and Zhang, Y. (2024). Dynamic resource allocation for virtual machine migration optimization using machine learning. *arXiv preprint arXiv:2403.13619*.

Kar, B., Yahya, W., Lin, Y.-D., and Ali, A. (2023). Offloading using traditional optimization and machine learning in federated cloud–edge–fog systems: A survey. *IEEE Communications Surveys & Tutorials*, 25(2):1199–1226.

Khan, S., Jiangbin, Z., Irfan, M., Ullah, F., and Khan, S. (2024). An expert system for hybrid edge to cloud computational offloading in heterogeneous mec-mcc environments. *Journal of Network and Computer Applications*, page 103867.

Preil, D. and Krapp, M. (2023). Genetic multi-armed bandits: a reinforcement learning approach for discrete optimization via simulation. *arXiv preprint arXiv:2302.07695*.

Ray, K. and Banerjee, A. (2024). Autonomous automotives on the edge. In *2024 37th International Conference on VLSI Design and 2024 23rd International Conference on Embedded Systems (VLSID)*, pages 264–269. IEEE.

Simchi-Levi, D. and Wang, C. (2023). Multi-armed bandit experimental design: Online decision-making and adaptive inference. In *International Conference on Artificial Intelligence and Statistics*, pages 3086–3097. PMLR.

Wang, D., Bai, Y., Huang, G., Song, B., and Yu, F. R. (2023a). Cache-aided mec for iot: Resource allocation using deep graph reinforcement learning. *IEEE Internet of Things Journal*.

Wang, X., Li, J., Ning, Z., Song, Q., Guo, L., Guo, S., and Obaidat, M. S. (2023b). Wireless powered mobile edge computing networks: A survey. *ACM Computing Surveys*, 55(13s):1–37.

Zhou, J., Lilhore, U. K., Hai, T., Simaiya, S., Jawawi, D. N. A., Alsekait, D., Ahuja, S., Biamba, C., and Hamdi, M. (2023). Comparative analysis of metaheuristic load balancing algorithms for efficient load balancing in cloud computing. *Journal of cloud computing*, 12(1):85.