J. Internet Comput. Serv.
ISSN 1598-0170 (Print) / ISSN 2287-1136 (Online)
http://www.jics.or.kr
Copyright ⓒ 2023 KSII

Enhancing Service Availability in Multi-Access Edge Computing with Deep Q-Learning☆

루숭구 조쉬 음와싱가[1] 샤이드 무하마드 라자[1] 리덕 타이[1] 김 문 성[3*] 추 현 승[1,2*]
Lusungu Josh Mwasinga Syed Muhammad Raza Duc-Tai Le Moonseong Kim Hyunseung Choo

ABSTRACT

The Multi-access Edge Computing (MEC) paradigm equips network edge telecommunication infrastructure with cloud computing resources. It seeks to transform the edge into an IT services platform for hosting resource-intensive and delay-stringent services for mobile users, thereby significantly enhancing perceived service quality of experience. However, erratic user mobility impedes seamless service continuity as well as satisfying delay-stringent service requirements, especially as users roam farther away from the serving MEC resource, which deteriorates quality of experience. This work proposes a deep reinforcement learning based service mobility management approach for ensuring seamless migration of service instances along user mobility. The proposed approach focuses on the problem of selecting the optimal MEC resource to host services for high mobility users, thereby reducing service migration rejection rate and enhancing service availability. Efficacy of the proposed approach is confirmed through simulation experiments, where results show that on average, the proposed scheme reduces service delay by 8%, task computing time by 36%, and migration rejection rate by more than 90%, when comparing to a baseline scheme.

☞ keyword : Edge Computing, Service Mobility, Service Availability, Beyond 5G, Deep Reinforcement Learning

1. Introduction

Beyond 5G (B5G) applications like autonomous driving require stringent ultra-Reliable Low Latency Communications (uRLLC) to meet Quality of Experience (QoE) expectations [1]. Multi-access Edge Computing (MEC) is amongst key enablers of B5G mobile networks and applications [2]. MEC provides data and computational resources at the network edge in proximity to users. This enables uRLLC application users like autonomous vehicles to offload computationally intensive tasks for expedited computing, thereby significantly reducing access latency. The substantial reduction in service access latency could enhance response times for the uRLLC applications which require swift response.

The uRLLC services offloaded to Service Containers (SCs) on MEC hosts may face severe performance degradation due to erratic vehicle mobility and MEC node resource constraints [3]. As a solution, service migration is proposed for seamless mobility of SCs to a suitable MEC host following vehicle mobility [4]. Although it enhances perceived QoE, the migration process imposes significant costs on both network and computing resources. Deciding when and where to migrate an SC to reduce migration rejection probability and service downtime are the main challenges in MEC service mobility management [3]. Existing strategies typically suffer from the highly dynamic nature of the MEC environment, and thus yield below par results in performance measures like MEC service availability.

Studies by [4],[5] proposed service mobility strategies for enhancing QoE with minimum costs based on Markov

1 Department of Computer Science and Engineering, Sungkyunkwan University, Suwon, 440-330, Korea
2 Department of Electrical and Electronics Engineering, Sungkyunkwan University, Suwon, 440-330, Korea.
3 Department of IT Convergence Software, Seoul Theological University, Bucheon, 422-742, Korea.
* Corresponding author (choo@skku.edu and moonseong@stu.ac.kr)
[Received 20 January 2023, Reviewed 31 January 2023, Accepted 19 February 2023]
☆ This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ICT Creative Consilience Program(2023-2020-0-01821), Artificial Intelligence Graduate School Program(Sungkyunkwan University) (2019-0-00421), and Artificial Intelligence Innovation Hub (2021-0-02068) supervised by the IITP(Institute for Information & communications Technology Planning & Evaluation).
☆ 본 논문은 2022년도 한국인터넷정보학회 추계학술대회 우수 논문 추천에 따라 확장 및 수정된 논문임.

Decision Process (MDP) techniques. The service mobility strategies discussed in [6],[7] exploit Deep Reinforcement Learning (DRL), where a Deep Q-Network (DQN) based agent learns the optimal migration policy by perpetually interacting with the MEC environment. Both studies capture a tradeoff between perceived QoE and service migration cost in terms of time. The studies focus on deciding when to migrate the SC for ensuring service continuity and enhancing perceived QoE. Both fall short of comprehensively addressing the where to migrate problem, which deals with selection of optimal MEC hosts for hosting the migrated SC. This leaves a gap with severe consequences that include increased service migration rejection rate. Rejected migration due to insufficient resources increases the undesirable service downtime as the mobility management strategy attempts to identify an alternative ES for hosting the rejected SC.

This study proposes DQN based Service Mobility Management (DQN-SEMM), a rejection-aware service service migration strategy based on Deep Q-Network (DQN). It tackles the MEC service migration decision problem using migration policy and MEC host selection policy. Besides deciding when to relocate the SC, the proposed DQN-SEMM also learns optimal selection of MEC host for service placement. Among others, the SC placement component uses resource utilization of neighboring hosts to select the optimal one. Most importantly, it imposes a penalty for each rejected service migration. The resulting server selection affects QoE of vehicle and is thus associated with a reward calculation for migration decision policy, which is a function of QoE and the migration cost. To this end, the main contributions of this work are summarized as follows:

- We propose a DQN-based Service Mobility Management (DQN-SEMM) system to seamlessly relocate service instances with marginal rejection rate.
- Development of a comprehensive simulation environment for evaluating performance of the proposed DQN-SEMM and baseline scheme.

Organization of this papers is as follows: section 2 presents a review of recent studies on service migration in the MEC paradigm. The proposed system is outlined under section 3 whereas performance evaluation results are presented in Section 4. Section 5 concludes this study.

2. Related Works

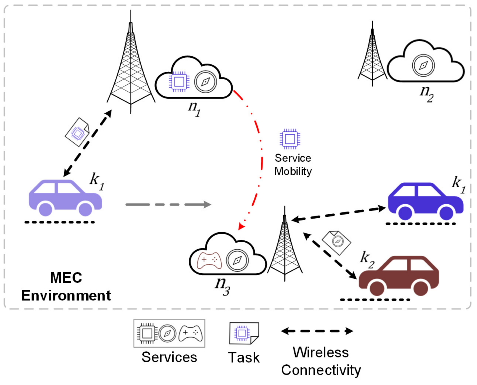
Various studies [8],[5] have devised service mobility strategies using MDP-based techniques with distance as the main criterion. A priority-induced strategy by [9] relocates services when higher-priority requests arrive at the MEC host. It employs an MDP-based method to select optimal servers to minimize migration costs. The strategy in [10] formulated the migration problem using MDP framework, aiming to minimize task completion time. However, it does not address the optimal server selection problem. These MDP based strategies require complete knowledge of environment transition probabilities, which can hardly be obtained with accuracy. In addition, they disregard important attributes such as resource availability during decision-making. Meanwhile, other studies [11],[12] have proposed prediction-based techniques for addressing the service mobility problem. However, such strategies require prior knowledge of various parameters, and their performance is highly dependent on prediction accuracy of these parameters.

A decentralized DQN-based scheme in [13] learns an optimal migration policy to achieve a trade-off between service delay and energy consumption in multi-user MEC environments. A distributed task migration method uses multi-agent DQN to enhance QoS while minimizing energy consumed by MEC infrastructure when migrating [14]. Moreover, the proposed scheme always selects the MEC server within the current location of vehicle as the best destination for service migration. Nonetheless, these strategies have limitations in terms of prioritizing migration cost and energy consumption reduction while overlooking key factors like service downtime and bandwidth consumption. This results in a trade-off that tilts towards cost reduction by reducing the number of service migrations at the expense of service QoE.

A closely related study [6] proposed DQN-based Location-aware (DQN-LOC) mechanism for mostly deciding when to relocate the service. DQN-LOC learns the optimal migration decision policy from past experiences gathered while interacting with its environment. The agent observes MEC system current environment state, defined as the distance between MEC host and vehicle serving base station (BS) locations. Thereafter, it decides the best action between migration and no migration. If it has decided to migrate, the MEC system relocates task data to the MEC server in the current location of vehicle, assuming the server has sufficient computing resources. This assumption is a major limitation since it reduces the network delay but causes significant computing delay when the associated MEC host is overloaded. It also imposes extended service disruption due to longer migration time, resulting in failed migration. In any case, taking resource availability into account is vital during service migration and MEC selection decisions.

So far, existing strategies seemingly provide notable improvements toward efficient seamless service mobility management, thereby facilitating the MEC paradigm to meet service quality demands for different vehicles. Nevertheless, a DQN-SEMM strategy is necessary to alleviate limitations observed in existing studies, especially in terms of optimal MEC host selection during service migration. In a case where the MEC host attached to the local BS is regarded as optimal to host the migrated service, disregarding factors such as long-term workload variations that could have an adverse impact on QoE for vehicle. Thus, our proposed DQN-based seamless service migration scheme seeks to balance the trade-off between migration costs and vehicle perceived QoE by deciding when to migrate, and most importantly, selecting an optimal server to host the service and ensure enhanced QoE.

3. Proposed System



(Figure 1) Illustration of MEC environment

3.1 System Model

Figure 1 depicts a model of the MEC environment. It comprises a set $N=\{1,2,...,|N|\}$ of MEC hosts. The N hosts are collocated with cellular network base stations (BSs), creating a network of distributed MEC servers. The servers are equipped with cloud computing resources to host SCs for a set $K=\{1,2,...,|K|\}$ of vehicles. Every k instantiates a resource-intensive and delay-stringent SC, denoted as O_k, deployed on the closest MEC host. The role of SC O_k is to coordinate the computation of tasks offloaded by vehicle k to its associated n for expedited computation. Each task $H_k(t)$ is identified by the tuple that defines its properties. In $H_k(t)$, the property $b_k(t)$ represents size of input data for the task; $c_k(t)$ is the amount of computing power required to process the task; and $d_k(t)$ refers to maximum task execution deadline.

Each vehicles offloads tasks to the serving MEC host through the associated BS over the RAN [3]. The uplink transmission rate achieved by vehicle $k \in K$ when transmitting tasks to the serving BS is given by:

$$R_k(t) = W_{RAN} \bullet \left\lfloor \frac{G}{U} \right\rfloor \bullet \log_2(1+\mathrm{SNR}_k) \quad (1)$$

where W_{RAN} is resource block bandwidth, G represents total number of resource blocks; U denotes the number of vehicles connected to the serving BS; and SNR_k corresponds to the received signal-to-noise ratio, given by:

$$\mathrm{SNR}_k = \frac{P \bullet h \bullet l_k^{\alpha(\mu-1)}}{N_0} \quad (2)$$

with P denoting constant transmission power of vehicle, h defines channel power gain, and N_0 is noise power. Moreover, l_k is the distance from location of vehicle k to its serving BS, while α denotes the path loss exponent, and μ is the fractional channel inversion control component. Using (1), the average wireless transmission delay experienced by k in time slot t while offloading task to its associated MEC host is:

$$\tau_k^{comm}(t) = \frac{b_k(t)}{R_k(t)} \quad (3)$$

where $b_k(t)$ denotes size of task being offloaded by k during time slot t. Downlink transmission delay is not entertained in this paper as the size of processed task from the MEC server is usually negligible [15].

In case SC O_k is deployed on a distant MEC server, offloaded tasks traverse the network backhaul for processing at the MEC host. The tasks experience transmission, propagation, and queueing delays. Transmission delay is a function of bandwidth and size of task input data, $b_k(t)$, being transmitted to the MEC host for processing. In addition, the round-trip propagation, processing, and queuing delays are determined by hop count between the associated BS and the serving edge server of k. Therefore, the backhaul delay for computation offloading is given by:

$$d_k^{back}(t) = \frac{b_k(t)}{W_{MAN}} + 2\lambda\,\omega_k(t) \tag{4}$$

where W_{MAN} is wired link bandwidth in the network backhaul, λ is a positive coefficient while $\omega_k(t)$ quantifies the number of hops between source and destination

During timeslot t, vehicles move to different locations according to random walk mobility model. When k connects to another BS, the proposed DQN-SEMM decides whether to relocate O_k in order to enhance service availability. Thus, it selects the most optimal n in terms of both resource availability and geographical location to host the migrated SC. The selection phase is critical because sometimes the selected MEC node may reject the migrated SC due to resource availability constraints as it is noted that hosts in the MEC paradigm are equipped with limited resources.

Meanwhile, optical fiber communication links in the network backhaul interconnect the MEC hosts to facilitate rapid mobility of the SCs. The target SC is transferred to the selected n using backhaul links. It incurs considerable migration costs which mainly depend on the size of migrated O_k, wired network bandwidth, propagation delay, and the hop count between MEC hosts. Similar to [3], the cost of moving service instance objects between source and destination MEC hosts is given by:

$$\eta_k(t) = \frac{O_k^{size}}{W_{MAN}} + 2\lambda\omega_k(t) \tag{5}$$

where O_k^{size} is the size of service container for k.

Multiple SCs share computing resources on the host to rapidly process the offloaded tasks. The computing capacity of MEC host n is denoted as F_n, quantified in Million Instructions Per Second (MIPS). The capacity is evenly shared by the hosted SCs. Thus, computing delay experienced by an offloaded task of k on MEC node is given by:

$$\tau_k^{comp}(t) = \frac{b_k(t)}{f_k(t)} \tag{6}$$

where $f_k(t)$ represents amount of computing power in MIPS allocated to the task of k as coordinated by its SC O_k. Combining (3), (4), and (6), the service delay for a task offloaded by k is given by:

$$T_k(t) = \tau_k^{comm}(t) + \tau_k^{back}(t) + \tau_k^{comp}(t) \tag{7}$$

The proposed DQN-SEMM employs Deep reinforcement Learning (DRL) to efficiently make service migration decision and selecting the optimal for maintaining service and enhancing availability for high-mobility users.

3.2 DQN-SEMM Method

The proposed DQN-SEMM method aims to efficiently manage service mobility for enhancing service availability which affects QoE. DQN-SEMM employs a DQN-based RL (RL) agent to competently learn the optimal policy deciding when to migrate and selecting the optimal MEC host. Formulation of the service mobility problem follows the RL framework. In the context of DQN-SEMM, the key elements of RL framework, namely, state, action, and reward function are defined as follows.

State: The agent in DQN-SEMM interacts with the MEC environment through states, observed at discrete time steps, . During time slot t, the agent observes a state, $s(t) = \{\omega_k(t), \phi_N(t), \chi_N(t)\}$, comprising distance,

$\omega_k(t)$, from the BS of k to that of its serving MEC host; amount of computing resource in vector, $\phi_N(t)$, currently available on each MEC node, and vector containing number of vehicles in the current cell, $\chi_N(t)$. Due to the dynamic nature of MEC environments, the proposed DQN-SEMM exploits Queuing theory to estimate average available resource on candidate MEC hosts.

Action: In DQN-SEMM, $A=\{a_1, a_2, ..., a_N\}$ denotes the action space which comprises N candidate destination MEC nodes for SC at time t. The agent selects and performs action $a(t) \in A(t)$ on the environment, resulting in migration of SC O_k to another MEC node. This changes environment to its subsequent state, $s(t')$.

Reward: The agent receives an immediate reward signal from the environment as evaluation of executed action, $a(t)$, and new state, $s(t')$. The immediate reward received in time slot is expressed as:

$$r(t) = T(t) - \eta(t) + \rho^{O_k}(t) \tag{8}$$

where $T(t)$ quantifies benefits experienced by k due to migration of SC O_k to selected MEC node. The term $\eta(t)$ represents costs in terms of migration time and bandwidth consumption as defined by (5). The agent receives a penalty, $\rho^{O_k}(t)$, if the selected destination MEC node rejects deployment of SC O_k due to resource constraints. Therefore, the objective for the migration decision problem becomes maximization of the expected discounted accumulative rewards over all time steps, defined as

$$R(t) = \max \mathrm{E} \sum_{t=0}^{T-1} \gamma^t r(t) \tag{9}$$

where the discount factor $\gamma = [0,1]$ determines the decreasing importance of immediate rewards over future ones. Thus, the agent aims at learning the optimal policy which maximizes the accumulative reward by selecting actions that return the highest reward at every time step t.

This study employs the classic Q-learning [16] algorithm to learn the optimal policy for maximizing (8). The algorithm defines a Q-value function, $Q_\pi(s(t), a(t))$, for evaluating the quality of action $a(t)$ under policy π when performed in state $s(t)$. At time t, the Q-learning agent approximates and stores a Q-value in the Q-table. The obtained value corresponds to the long-term expected discounted reward under a specific π. The Q-value function can be defined as:

$$Q_\pi(s(t), a(t)) = r(t) + \gamma \max Q_\pi(s(t'), a(t')) \tag{10}$$

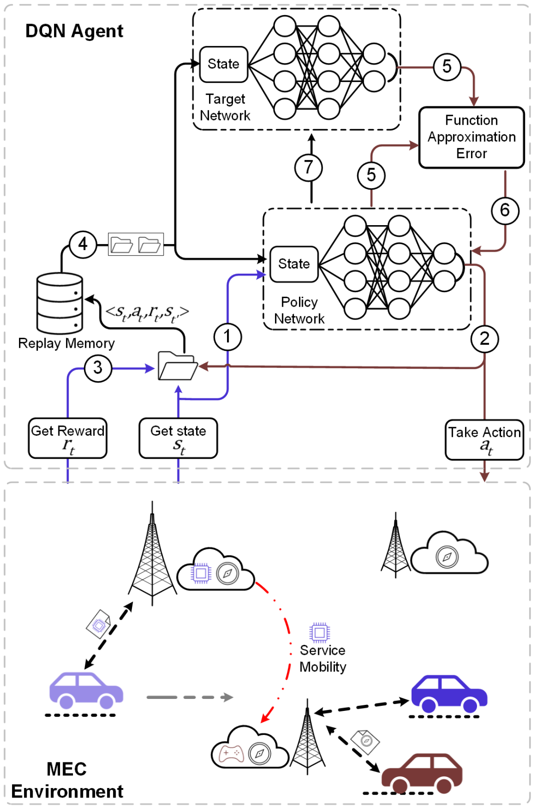
where the second term is the Q-value function in the next state $s(t')$. Using the Bellman equation, an agent iteratively updates Q-values until converging to the optimal Q-values, denoted $Q_\pi^*(s(t), a(t))$. The agent focuses on learning the optimal policy that achieves highest accumulative discounted reward.

Q-learning does not always find the optimal policy when solving complex real-world RL problems comprising large state action spaces. Furthermore, it fails to utilize learned knowledge when make decisions in newly encountered states. To overcome the limitations, this study exploits Deep Q-Learning [17] which uses deep neural networks to approximate the Q-value function for finding the optimal policy. Known as a Deep Q-Network, the neural network in Deep Q-Learning is represented as $Q_\pi(s(t), a(t); \theta)$, where θ is model parameters/weights for approximating the optimal value function. Thus, DQN agent focuses on finding θ that minimizes the Mean-Squared Error (MSE) loss function, given as:

$$L_i(\theta_i) = \mathrm{E}\left[(z - Q_\pi(s(t), a(t); \theta_i)^2\right] \tag{11}$$

in which $z = r(t) + \gamma \max_{a(t)} Q_\pi(s(t'), a(t'); \theta_i^-)$ represents the target Q-value function; θ_i are the parameters of the Q-network during iteration i while θ_i^- are the weights for computing the target Q-values, which are only updated at fixed intervals. The loss function, $L_i(\theta_i)$, is differentiated with respect to the weights θ_i, obtaining a gradient vector. Consequently, the loss function is optimized using Stochastic Gradient Descent, resulting in updated network weights during iteration i.

Figure 2 illustrates service mobility management DQN-SEMM procedure, with respect to vehicle, k_1. Initially,



(Figure 2) Service mobility management using DQN-SEMM

the vehicle is within the service area of n_1, thus it is determined as initial host of service instance for k_1. Mobility of vehicle k to another location triggers the proposed DQN-SEMM to obtain the current state, $s(t)$ from MEC environment. DQN-SEMM passes the observed state $s(t)$ to the neural network to approximate Q-values for all actions ①. It selects an action with the highest Q-value to perform on environment ②, corresponding to the decision that O_k should either be migrated to an optimal or remain on current host. In return, the agent receives a reward from the environment following the executed action and resulting state transition ③.

Initially, the agent does not have adequate knowledge about behavior of its environment and actions/decisions. As such, the learning process starts with using the ϵ-greedy policy to select actions. The ϵ-greedy policy selects an action with probability ϵ and performs on the environment. This phase enables the agent to explore available environment states and actions for improved decision-making. At time t, the agent stores explored states, actions, next states and rewards in a Replay Memory dataset, $D = \{e_1, e_2, ..., e_t\}$ as experience sample tuple, $e_t = (s(t), a(t), r(t), s(t'1))$. An exploration decay rate parameter controls the transition into exploitation phase where the agent uses saved experience samples for training the DQN.

The exploitation phase involves the agent utilizing stored experiences samples to train the DQN model for improving its decision-making effectiveness. Once D contains sufficiently enough e_t samples for training, the agent uniformly draws sample mini-batches, denoted $(s_k(t), a_n(t), r_k(t), s_k(t')) \sim U(D)$ at random from the dataset ④. The states $s_k(t)$ and $s_k(t')$ are input to the Policy and target networks, respectively for forward propagation to approximate corresponding Q-values. Using Q-value outputs from the policy and target networks ⑤, DQN computes MSE using (10) to evaluate learning progress and optimal Q-value function approximation capability. After computing MSE, the agent updates weight parameters of the policy network using stochastic gradient descent ⑥. After C-training episodes, weight parameters θ of the Policy network are cloned to update the weights in the Target network, which are held fixed for C-steps ⑦. All the while, the agent replaces old experience samples in the Replay Memory with newly obtained samples until the model converges to optimality.

4. Performance Evaluation

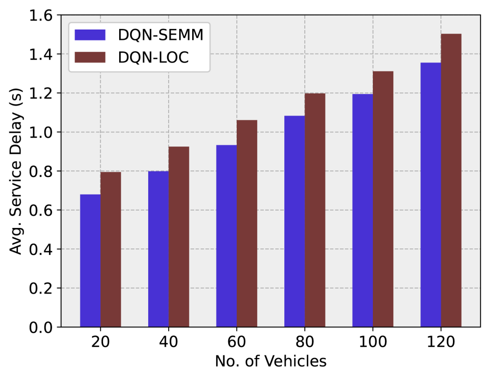
This section presents performance evaluation results of the proposed DQN-SEMM strategy. The evaluation is based on data obtained through simulation experiments. DQN-SEMM performance is compared with DQN-based Location-aware (DQN-LOC) migration strategy [6]. Unlike DQN-SEMM, decision criteria in DQN-LOC includes only distance, ignoring crucial time-variant factors like resource availability. This renders the strategy vulnerable to increased service time especially during peak hours when more users offload tasks to MEC hosts. Most importantly, it does not address optimal MEC node selection as the SC is always migrated to the serving MEC node. Performance metrics for comparison include service delay, task processing delay, and migration rejection rate.

(Table 1) Simulation Parameters

Parameter	Value
No. of servers, $\lvert N \rvert$	18
No. of AVs, $\lvert K \rvert$	20 ~ 120
Container size, O_k^{size}	0.5 GB
Task input data size, $b_k(t)$	2 ~ 5 MB
Task length, $c_k(t)$	50 ~ 100 MI
Task execution delay, $d_k(t)$	0.35 s
Rejection penalty, $\rho^{O_k}(t)$	0.5
Computing capacity, F_n	700 MIPS
No. of Res. blocks, G	50
Res. block bandwidth, W_{RAN}	180 kHz
Vehicle transmission power, P	200 mW
Noise spectral density, N_0	- 174dBm/Hz
Path loss exponent, α	3.75
Power control factor, μ	0.25
Wired link bandwidth, W_{back}	1 Gbps
Training episodes	500
Replay Memory size	10000
Exploration rate	0.05
Minibatch size	16
Discount factor, γ	0.99
Learning rate	0.001

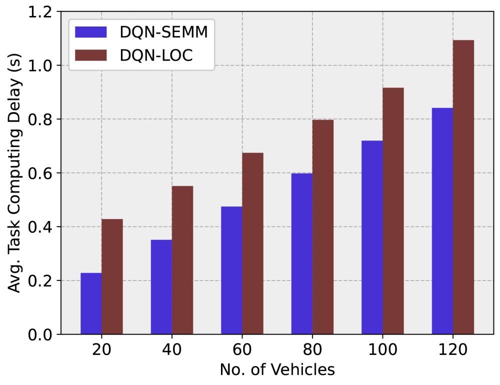
We implemented a simulation environment using Python. It consists of 36 cells, with each having a BS that serves AVs within the cell [6]. Although deploying a Mobile Edge Computing (MEC) server with each Base Station (BS) simplifies the simulation, it is not feasible in practice because it results in significant capital and operational expenses for the mobile operator. Thus, we deployed 18 Servers across the network to provide cloud computing resources to vehicles. The computation capacity of each MEC server is $F_n = 700$ MIPS while the number of vehicles is varied from 20 to 120. Each vehicle generates tasks according to Poisson process with $\lambda = [1, 2]$. Table 1 provides a summary of parameters used during simulation experiments.

Figure 3 shows performance of DQN-SEMM and DNQ-LOC on average service delay for varying number of vehicles. As observed, average service delay is directly proportional to the number of vehicles exploiting MEC resources. DQN-SEMM shows outstanding performance over DQN-LOC as it reduces the average service delay by up to 8%, on average. This is due to its ability to strategically deploy migrated SCs on MEC hosts that are both in proximity to the vehicle and equipped with adequate computing resources to expeditiously process offloaded tasks. Notably, the proposed approach achieves a 15.57% and 10.36% service delay reduction comparing to DQN-LOC when there are 20 and 120 vehicles exploiting MEC resources, respectively.

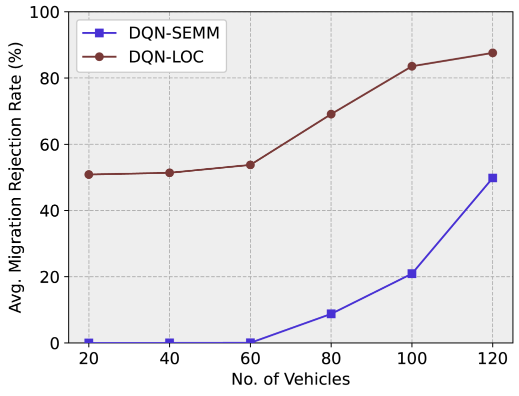


(Figure 3) Impact of varying number of vehicles on average service delay.

As depicted in Figure 4, DQN-SEMM reduces processing delays for tasks on MEC hosts by 26.04% compared to DQN-LOC when 120 vehicles share MEC resources. Overall, DQN-SEMM achieves an average reduction of 36% in task computing time compared to DQN-LOC. This is because DQN-SEMM consistently transfers services to MEC hosts with sufficient resources, leading to faster processing of offload tasks. In contrast, DQN-LOC assumes that the host associated with serving BS has adequate resources, causing services to migrate to the ideal MEC server in terms of location but not computing resources. Therefore,



(Figure 4) Impact of varying number of vehicles on average task processing time.



(Figure 5) Impact of varying number of vehicles on average migration rejection rate.

DQN-SEMM enables operators to reduce service delays with fewer servers, leading to lower capital and operational costs.

Service migration may fail if MEC hosts lack sufficient resources, leading to the system either searching for an alternative host or abandoning migration. The migration rejection rate impacts the failure rate of migration decisions, resulting in a longer migration wait time and reduced QoE for vehicles. Figure 5 shows that DQN-SEMM significantly outperforms DQN-LOC in terms of reducing migration rejection rates. On average, DQN-SEMM reduces rejection rates by over 90%, with a 54.96% reduction when the MEC system is congested with requests from 120 vehicles. This is because DQN-SEMM considers resource availability and imposes penalties on the agent for rejected migrations.

5. Conclusion

In this paper, dynamic service mobility is discussed, aiming to maintain service continuity and satisfy URLLC service requirements. Due to its recent advances and performance, Deep Reinforcement Learning is exploited to implement solution for autonomous efficient MEC service mobility management. Efficacy of the proposed approach is confirmed through simulation experiments and results reveal outstanding performance when benchmarked against the target scheme. As future work, we intend to expand our scheme by employing multi-agent deep reinforcement learning.

Reference

[1] F. Tang, Y. Kawamoto, N. Kato, and J. Liu, “Future Intelligent and Secure Vehicular Network Toward 6G: Machine-Learning Approaches,” *Proceedings of the IEEE*, vol. 108, no. 2, pp. 292–307, 2020. https://doi.org/10.1109/JPROC.2019.2954595.

[2] S. D. A. Shah, M. A. Gregory, S. Li, and R. D. R. Fontes, “SDN enhanced multi-access edge computing (MEC) for E2E mobility and QoS management,” *IEEE Access*, vol. 8, pp. 77459–77469, 2020. https://doi.org/10.1109/ACCESS.2020.2990292.

[3] Q. Yuan, J. Li, H. Zhou, T. Lin, G. Luo, and X. Shen, “A Joint Service Migration and Mobility Optimization Approach for Vehicular Edge Computing,” *IEEE Transactions on Vehicular Technology*, vol. 69, no. 8, pp. 9041–9052, 2020. https://doi.org/10.1109/TVT.2020.2999617.

[4] T. Taleb, A. Ksentini, and P. A. Frangoudis, “Follow-me cloud: When cloud services follow mobile users,” *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp.369–382, 2019. https://doi.org/10.1109/TCC.2016.2525987.

[5] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. Chan, and K. K. Leung, “Dynamic Service Migration in Mobile Edge Computing Based on Markov Decision Process,”

IEEE/ACM Transactions on Networking, vol. 27, no. 3, pp. 1272–1288, 2019. https://doi.org/10.1109/TNET.2019.2916577.

[6] C. Zhang and Z. Zheng, “Task migration for mobile edge computing using deep reinforcement learning,” *Future Generation Computer Systems*, vol. 96, pp. 111–118, 2019. https://doi.org/10.1016/j.future.2019.01.059.

[7] Z. Tang, X. Zhou, F. Zhang, W. Jia, and W. Zhao, “Migration Modeling and Learning Algorithms for Containers in Fog Computing,” *IEEE Transactions Services Computing*, vol. 12, no. 5, pp. 712–725, 2019. https://doi.org/10.1109/TSC.2018.2827070.

[8] L. Ye, K. Ling, Q. Han, Y. Yan, L. Zeng, L. Qi, L. Lin, and J. Zhang, “DIMDP: A Driving Intention-Based MDP Service Migration Model under MEC/MSCN Architecture,” *Mobile Information Systems*, vol. 2022, 2022. https://doi.org/10.1155/2022/4988266.

[9] A. Mukhopadhyay, G. Iosifidis, and M. Ruffini, “Migration-Aware Network Services With Edge Computing,” *IEEE Transactions on Network and Service Management*, vol. 19, no. 2, pp. 1458–1471, 2022. https://doi.org/10.1109/TNSM.2021.3139857.

[10] X. Li, S. Chen, Y. Zhou, J. Chen, and G. Feng, “Intelligent Service Migration Based on Hidden State Inference for Mobile Edge Computing,” *IEEE Transactions Cognitive Communications Networking,* vol. 8, no. 1, pp. 380–393, 2022. https://doi.org/10.1109/TCCN.2021.3103511.

[11] S. Wang, R. Urgaonkar, T. He, K. Chan, M. Zafer, and K. K. Leung, “Dynamic Service Placement for Mobile Micro-Clouds with Predicted Future Costs,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 4, pp. 1002–1016, 2017. https://doi.org/10.1109/TPDS.2016.2604814.

[12] H. Ma, Z. Zhou, and X. Chen, “Leveraging the Power of Prediction: Predictive Service Placement for Latency-Sensitive Mobile Edge Computing,” *IEEE Transactions on Wireless Communications*, vol. 19, no. 10, pp. 6454–6468, 2020. https://doi.org/10.1109/TWC.2020.3003459.

[13] W. Chen, Y. Chen, J. Wu, and Z. Tang, “A multi-user service migration scheme based on deep reinforcement learning and SDN in mobile edge computing,” *Physical Communications*, vol. 47, p. 101397, 2021. https://doi.org/10.1016/j.phycom.2021.101397.

[14] Y. Cui, D. Zhang, J. Zhang, T. Zhang, L. Cao, and L. Chen, “Distributed task migration optimization in MEC by deep reinforcement learning strategy,” *Conference Local Computer Networks,* 2021. https://doi.org/10.1109/LCN52139.2021.9524987

[15] T. X. Tran and D. Pompili, “Joint task offloading and resource allocation for multi-server mobile-edge computing networks,” *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 856–868, 2019. https://doi.org/10.1109/TVT.2018.2881191

[16] R. S. Sutton and A. G. Barto, *Reinforcement Leaning*. MIT Press, 2018.

[17] V. Mnih *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, 2015. https://doi.org/10.1038/nature14236

◐ 저 자 소 개 ◑

루숭구 조쉬 음와싱가 (Lusungu Josh Mwasinga)
2018년 Daeyang University, College of ICT (Bsc. ICT)
2019년～현재 성균관대학교 대학원 소프트웨어학과 석박사통합과정
관심분야 : AI-driven resource orchestration solutions for edge clouds, proactive service mobility, semi-supervised and distributed learning techniques for NextGen intelligent networks.
E-mail : lusujosh@skku.edu

샤이드 무하마드 라자(Syed Raza)
2006년 파키스탄 PIEAS 대학교 컴퓨터공학부(공학사)
2010년 스웨덴 룬드 대학교 무선통신공학부(공학석사)
2018년 성균관대학교 일반대학원 전자전기컴퓨터공학과(공학박사)
2020년～현재 성균관대학 전자전기컴퓨터공학과 연구교수
관심분야 : 소프트웨어 정의 네트워킹, 네트워크 기능 가상화 등
E-mail : s.moh.raza@skku.edu

리덕 타이 (Duc-Tai Le)
2010년 베트남 국립 대학교 대학원 컴퓨터공학과(공학석사)
2016년 성균관대학교 대학원 컴퓨터공학과(공학박사)
2016년～2019년 성균관대학교 박사후 연구원
2019년～현재 성균관대학교 전자전기컴퓨터공학과 연구 교수
관심분야 : 무선 에드혹 및 센서 네트워크, 소프트웨어 정의 네트워킹, 엣지/클라우드 컴퓨팅, IoT 등
E-mail : ldtai@skku.edu

김 문 성 (Moonseong Kim)
2002년 성균관대학교 일반대학원 수학과(이학석사)
2007년 성균관대학교 일반대학원 전기전자및컴퓨터공학부(공학박사)
2007년～2009년 미국 미시간주립대학교 컴퓨터과학공학과 연구원
2009년～2018년 특허청 사무관
2018년～현재 서울신학대학교 IT융합소프트웨어학과 조교수
관심분야 : 유무선 네트워크, 모바일 컴퓨팅, 머신러닝, 지식재산권 등
E-mail : moonseong@stu.ac.kr

추 현 승 (Hyunseung Choo)
1990년 댈러스 텍사스 대학 컴퓨터공학과(공학석사)
1996년 알링턴 텍사스 대학 컴퓨터공학과(공학박사)
1997년～1998년 특허청 사무관
1998년～현재 성균관대학교 소프트웨어대학 교수
관심분야 : 모바일 센서 네트워크, 소프트웨어 정의 네트워킹, 지능형 모바일 컴퓨팅, 멀티 액세스 엣지 컴퓨팅, 머신러닝 및 인공지능 등
E-mail : choo@skku.edu