Optimization of Reinforcement Learning-Based Backoff Indicator for 5G NR Random Access Procedure

Jiha Kim
dept. Information and Communication Engineering
Myongji university
Yongin-si, Korea
yaki5896@mju.ac.kr

Cheolwoo You
dept. Information and Communication Engineering
Myongji university
Yongin-si, Korea
cwyou@mju.ac.kr

Hyunhee Park*
dept. Information and Communication Engineering
Myongji university
Yongin-si, Korea
hhpark@mju.ac.kr

***Abstract*—This paper proposes a reinforcement learning approach to improve the performance of the random access procedure in the massive machine type communication (mMTC) environment, as defined by 3GPP, to accommodate an increasing number of user equipment (UEs). The existing random access process results in channel congestion when UEs that have failed to transmit due to a fixed backoff indicator choose a subsequent random access opportunity (RAO) slot. To address this, we introduce an agent that automatically adjusts the backoff indicator based on the state of the network environment. As a result of simulations using Q-learning and DDPG, DDPG achieves a higher success rate and reward, but the backoff indicator continually increases. In contrast, Q-learning exhibits relatively lower performance, yet the stability of the backoff indicator is maintained. From these findings, it is evident that the integration of reinforcement learning to optimize the random access procedure enhances performance. However, it necessitates careful algorithm selection and precise tuning of the reward mechanism.**

***Index Terms*—5G, mMTC, Random access, Reinforcement learning, Q-learning, DDPG**

I. Introduction

The random access procedure is a method utilized when a UE attempts to connect to a network in a wireless communication system. It predominantly takes place in scenarios such as establishing new connections, synchronizing transmission and reception intervals, and uplink data transmission. Due to the recent surge in the number of UEs owned by individuals, 3GPP has introduced performance indicators in the mMTC [1] environment as one of the KPIs for 5GNR. Furthermore, with advancements in hardware technology, research is also being actively conducted to enhance the communication environment by integrating artificial intelligence into communication [2]-[4]. The random access procedure is among the frequent events in a wireless communication environment and is crucial for the UE to either successfully connect to the network or be allocated resources. A variety of optimization techniques are applicable here, and the efficiency of these procedures is paramount, especially in situations where network congestion is prevalent.

However, using the existing random access procedure [5], it is challenging to accommodate all UEs. It becomes problematic to manage a significant number of UEs that suddenly attempt to access the network with limited resources and types of preambles. This paper proposes an approach to adjust the value of the backoff indicator, which governs the wait time for retransmission during the random access procedure. The backoff indicator specifies a range for determining the random wait time during retransmission and is a predefined value. Through reinforcement learning algorithms, we aim to enhance the success rate and minimize access wait time by utilizing the optimal backoff indicator in an environment with a large influx of UEs.

II. Background

A. Preamble transmission (MSG 1)

The UE transmits a random access preamble signal through a random access channel (RACH). The number of preambles that can be selected is predefined and transmitted, and these are randomly selected through the Zadoff-Chu sequence.

B. Random access response (MSG 2, RAR)

The base station (BS) receives a preamble signal and transmits a random access response (RAR) to the corresponding UE. The message includes information such as that necessary for time synchronization, initial uplink scheduling information, and temporary cell ID.

In this process, if the BS does not receive the preamble signal, or if different UEs select and transmit the same preamble, a collision occurs. The UE starts a type of timer, called the RAR window size, from the time it transmits the preamble to the time it receives the RAR. If the RAR is not received during the RAR window size, the UE proceeds with the preamble retransmission, which is immediately transmitted after waiting for a random time to avoid likely preamble collisions. Here, the random time is selected within the range of the backoff indicator.

C. RRC Connection request (MSG 3)

The UE transmits the connection request message, MSG 3, based on the scheduling information included in the RAR message. This message includes information such as the ID of the UE and the service type. In this process, the UE activates a predefined connection resolution timer and waits until it receives MSG 4.

D. RRC Connection response (MSG 4)

After receiving MSG 3, the BS transmits a connection setup message, MSG 4, containing the necessary information for connection establishment to the UE. If the UE does not receive MSG 4 within the connection resolution timer duration, it selects a random time within the backoff indicator range, similar to the previous retransmission procedure, and performs the preamble transmission process for MSG 1.

III. Reinforcement learning

Reinforcement learning is a subset of machine learning and is a technique that learns optimal policies through the process of receiving rewards when an agent takes an action in an environment. The agent observes the current state in the environment and selects an action based on that state. After applying the selected action, the environment provides feedback in the form of a change in state and a reward as a result of the action. The agent continues to refine its action policy based on this feedback. Figure 1 illustrates the learning process of reinforcement learning.

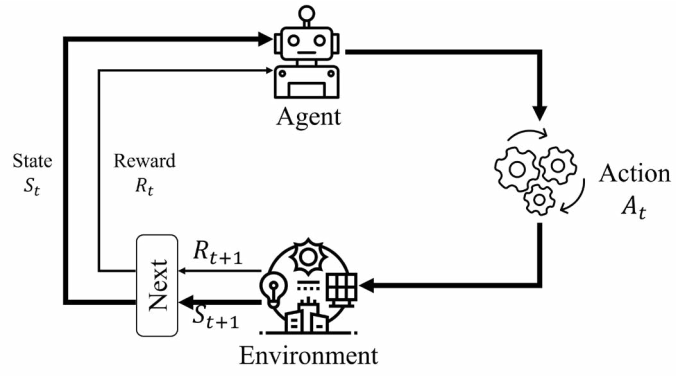


Fig. 1. Architecture of Reinforcement learning

1) Q-learning: Q-learning [6] is one of the most well-known off-policy learning algorithms in reinforcement learning. In this approach, the agent estimates the optimal action-value function, often referred to as the Q-function, using experiences obtained from interactions with the environment. The learning process of Q-learning is grounded in the Bellman equation, as depicted in equation (1) below.

$$Q(s,a) \leftarrow Q(s,a) + \alpha[R(s,a) + \gamma \max_{a'}(s',a') - Q(s,a)] \quad (1)$$

Use equation (1) to estimate the action value (Q-value) of the current state. Here, α and γ are used to transform future rewards into current values, representing the learning rate and discount factor, respectively. s, a, s', and a' represent the current state, current action, next state, and next action, respectively, while $R(s,a)$ denotes the reward.

At the onset of learning, the Q-value can be initialized to any value. As the agent selects and performs an action in the environment, the state, reward, and subsequent state are determined. This information is then utilized to update the Q-value. By repeatedly performing these steps, the Q-function is optimized. Among the advantages of Q-learning are its straightforward processes and guaranteed convergence. Given an appropriate learning rate and ample interactions with diverse environments, Q-learning converges to the optimal Q-function. Figure 2 shows the architecture related to the learning of Q-learning.

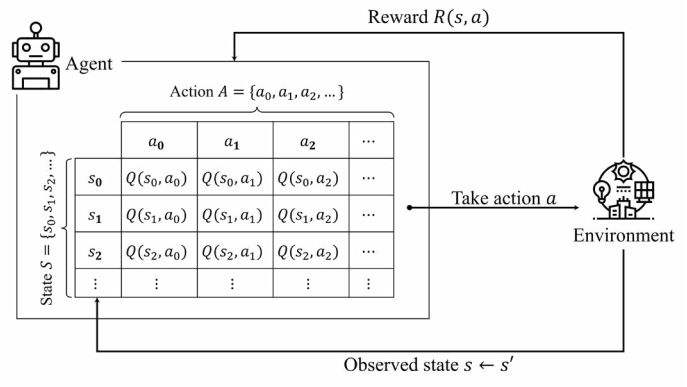


Fig. 2. Architecture of Q-learning

2) Deep Deterministic Policy Gradient (DDPG): It is challenging to employ Q-learning in a continuous action space. Exploring a continuous space with a greedy policy requires optimization for all possible actions at each step, making it very time-consuming when applied to non-parametric function approximators. DDPG [7] is a variant of the model-free actor-critic algorithm designed for continuous action spaces. Figure 3 shows the learning process of critic and actor in DDPG. It integrates the principles of deep learning with reinforcement learning to function effectively in intricate environments.

The Actor represents a policy function that determines the optimal action in the current state. The Critic evaluates the value (Q-value) of the action chosen by the actor. In this context, both the actor and critic are treated as deep learning models. The updates for the critic resemble the approach in Q-learning, as depicted in equation (2), but consider the continuous action spaces for the action.

$$Q^{\text{target}}(s,a) = R(s,a) + \gamma Q(s', \mu(s')) \quad (2)$$

where $R(s,a)$ represents the reward obtained from the current state and action, γ denotes the discount factor, and s' signifies the subsequent state. μ represents the current policy as determined by the actor network. Thus, $\mu(s')$ indicates the action recommended by the actor in the next state s'. The TD-error, used by the Critic to update both the critic and actor networks, is computed as follows:

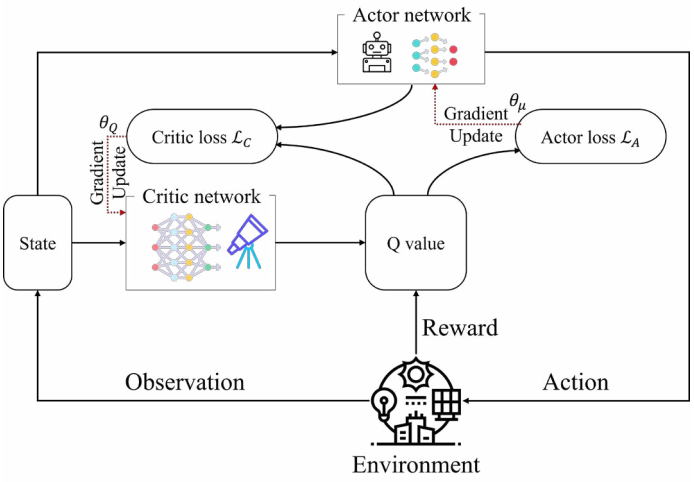


Fig. 3. Architecture of DDPG

$$TD-error = Q^{\text{target}}(s,a) - Q(s,a) \tag{3}$$

The update of the Actor uses the policy gradient method as shown in equation (3). Update the actor network in the direction of maximizing the objective function.

$$J = \mathbb{E}[Q(s,\mu(s))] \tag{4}$$

Here, J represents the expected reward (objective function) for the action recommended by the actor network in the current state, denoted by $\mu(s)$. We update the actor network to maximize this objective function. During the learning process, the actor selects and executes actions based on the current policy. The Critic calculates the Q-value for this action. The policy of actor is then updated based on the TD-error from the Critic. Furthermore, techniques like experience replay and the target network are incorporated to enhance the stability and convergence of the learning process.

IV. Proposed Method

In this paper, we conduct learning in an environment using Traffic Model 2 proposed in the 3GPP standard document. Traffic Model 2 represents the distribution of UEs according to a beta distribution. This model allows us to simulate a rapid increase in traffic volume. UEs attempting the RA procedure select a random backoff count using a predefined backoff indicator for retransmission if MSG 2 is not received. However, since all accessing UEs use the same backoff indicator, there is a likelihood of collisions occurring again in subsequent RAO slots. To address this issue, the Agent is trained to automatically adjust the value of the backoff indicator based on the state of the network environment. The actions chosen by the agent are defined as either increasing or decreasing the backoff indicator. In one simulation, the reward is defined using metrics such as the success rate, congestion, and average access delay. The reward is defined as shown in the expression (5) below:

$$\begin{aligned} Reward =& (\text{success rate} * (1 - \text{congestion})) \\ & - (\text{average access delay} * \text{congestion}) \end{aligned} \tag{5}$$

Equation (5) for the reward promotes learning by increasing the success rate and decreasing the average access delay. Since the average access delay is measured in milliseconds (ms), it is normalized to a value between 0 and 1. High congestion reduces the success rate, which in turn decreases the weight proportion used for rewards. In other words, when congestion and average access delay are high, a negative reward is received and utilized for learning. In such scenarios, when the agent receives a negative reward, it interprets that the channel is congested and increases the value of the backoff indicator. Conversely, when a positive reward is received, the agent acts to decrease the backoff indicator and reduce the average access delay.

V. Simulation Environment and Results

A. Environment and learning parameter

TABLE I
Environment of Random Access [8]

Parameter	Value
Total number of UEs	15,000
α, β	3, 4
Simulation time	10s
PRACH configure index	6
Total number of preambles	54
Number of UL grants per RAR	3
Number of CCEs per PDCCH	4
RA-Response window size	5ms
RRC Connection resolution timer	48ms

Table I defines the environmental parameters used for the simulation. The parameters are referenced from the 3GPP standard document [8] and employ traffic model 2 to represent a rapid UE access environment. The simulation includes a total of 15,000 UEs, with the number of accessing UEs per RAO set to follow the beta distribution of traffic model 2. The parameters α and β for the Beta distribution are specified as 3 and 4, respectively. The number of uplink grants per RAR is set to three, and the number of control channel elements (CCE) that can be allocated per physical downlink control channel (PDCCH) is set at four. The RA-Response window size and the RRC Connection resolution timer are set to 5 ms and 48 ms, respectively.

TABLE II
Q-learning Reinforcement Learning Parameters

Parameter	Value
Learning Rate	0.5
Discount Factor	0.9
Exploration Rate	0.3
Max Exploration Rate	1.00
Min Exploration Rate	0.01
Exploration Decay Rate	0.01

Tables II and III define parameters used in Q-learning and DDPG learning. The learning rate for Q-learning is set to 0.5,

determining how much the Q-value is updated in each episode. The discount factor is set to 0.9, meaning the agent considers future rewards as 90% of the current reward. The exploration rate is set to 0.3, allowing the agent to sometimes explore new actions rather than always choosing the optimal ones. The maximum exploration rate and minimum exploration rate, set to 1.00 and 0.01 respectively, define the range of the exploration activity of agent. The exploration decay rate influences how quickly the exploration rate decreases as the agent becomes more familiar with the environment.

For DDPG, the replay buffer size is set to 1000, indicating the size of the memory buffer that the agent uses to recall past actions. The discount factor works the same role as in Q-learning. To update the target network, Deep Deterministic Policy Gradient performs soft updates on both the critic and the actor. In this paper, we denote this value as τ and set it to 0.001.

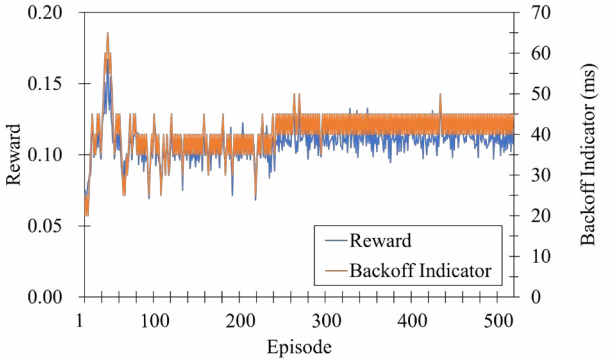
TABLE III
DDPG REINFORCEMENT LEARNING PARAMETERS

Parameter	Value
Reply buffer size	1000
Batch size	32
Discount Factor	0.99
τ (Ratio of soft updates on the target network)	0.001
Learning Rate of Actor	0.005
Learning Rate of Critic	0.005

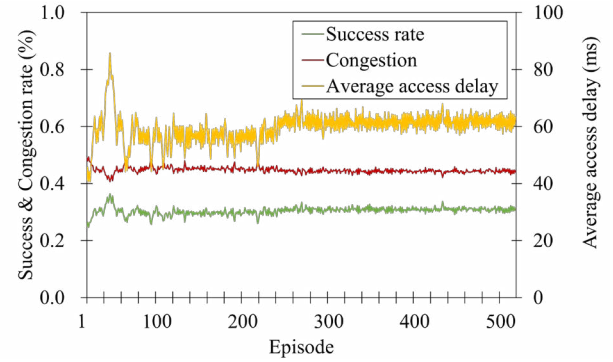
B. Simulation results

Figure 4 presents the simulation results of the Q-learning algorithm. Subfigure (a) in Figure 4 illustrates the evolution of the backoff indicator selected by the Q-learning algorithm based on rewards as the learning progresses. Subfigure (b) in Figure 4 compares the success rate, congestion rate, and average access delay over the course of the learning process of the Q-learning algorithm. The rewards fluctuate between 0.066 and 0.132 throughout the episodes. The backoff indicator demonstrates a steady trend, climaxing at 45 ms by the end of the simulation. The success rate enhances performance by as much as 32%. Regarding congestion, it decreases from approximately 48.8% to 43%, highlighting the growing capability of the agent to manage network traffic efficiently. In terms of the average access delay, significant fluctuations are observed. The 30th episode recorded the peak delay at 76.54 ms, while in subsequent episodes, the delay generally hovered around an average of 60 ms.

Figure 5 presents the learning results of the DDPG algorithm. Similar to Figure 4, subfigure (a) in Figure 5 illustrates the changes in the success rate and the backoff indicator, while subfigure (b) depicts the variations in success rate, congestion, and average access delay. The reward experiences a notable peak of 0.740 in the 120th episode, starting from 0.062. At the same time, the success rate demonstrates a performance close to 100%, highlighting the efficiency of the DDPG algorithm in optimizing system performance. In terms of congestion, there is a significant decrease, declining to around 22% by the

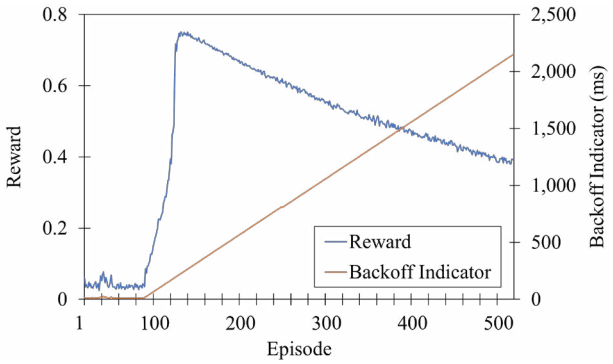


(a) Reward with backoff indicator

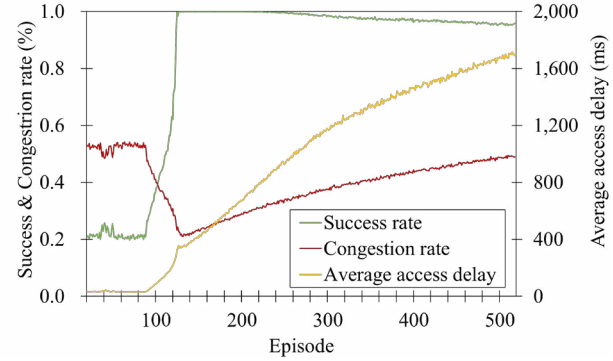


(b) Success rate with Congestion

Fig. 4. Result of applying Q-learning algorithm



(a) Reward with backoff indicator



(b) Success rate with Congestion

Fig. 5. Result of applying DDPG algorithm

110th episode. Although backoff indicators play a crucial role in access delay, they consistently rise as learning continues. Starting at an initial 15 ms, it is observed that by the 280th episode, it escalates to 1050 ms. As a result, the average access delay, which began at 35.05 ms, also surges to approximately 1178.4 ms by the 280th episode. This sharp increase adversely impacts the delay, especially concerning the performance of the User Equipment.

Overall, Q-learning appears to yield a relatively unstable return value as the episodes progress. Notably, backoff indicators tend to rise over time, which could pose challenges for network access delay. Moreover, the average access delay also witnesses a substantial increase during certain episodes. The performance of DDPG seems to surpass that of Q-learning. Although the early episodes present comparable outcomes, the reward value displays noticeable improvements as more episodes unfold. However, the value of the backoff indicator has grown more markedly over time. This observation suggests that reducing access delay in the DDPG algorithm may not be a primary concern when determining rewards.

VI. Conclusion

In this paper, we propose an approach to adjust the backoff indicator to address the issues caused by rapid UE access in the mMTC environment. In the existing random access procedure, all UEs cannot evade the congested channel environment when using a fixed backoff indicator, which impacts UEs attempting the random access procedure in the subsequent RAO slot. To address this, we suggest a strategy that enables agents to automatically adjust the value of the backoff indicator based on the network state.

Simulation results from two reinforcement learning algorithms, Q-learning and DDPG, indicate that DDPG performs well in terms of rewards and success rate. However, there are concerns with the continuous increase of backoff indicators. In contrast, Q-learning maintains a stable backoff indicator, but its performance lags significantly behind DDPG. Thus, utilizing reinforcement learning to optimize the random access procedure is beneficial, but choosing the right algorithm and fine-tuning the reward mechanisms are crucial.

For future research stemming from this paper, we aim to investigate reinforcement learning algorithms to refine the reward mechanism and enhance the performance of random access procedures in even larger environments.

Acknowledgment

This work was supported by Institute for Information & communications Technology Planning & Evaluation(IITP) and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) grant funded by the Korea government(MSIT) (No. 2021-0-00368, Development of the 6G Service Targeted AI/ML-based autonomous-Regulating Medium Access Control (6G STAR-MAC), No. 2022R1A2C2005705, AI-MAC Protocol on Distributed Machine Learning for Intelligent Flying Base Station)

References

[1] A. Dogra, R. K. Jha, and S. Jain, "A survey on beyond 5G network with the advent of 6G: Architecture and emerging technologies," in IEEE Access, vol. 9, 2020, pp. 67512–67547.

[2] H. Song, J. Bai, Y. Yi, J. Wu, and L. Liu, "Artificial intelligence enabled Internet of Things: Network architecture and spectrum access", in IEEE Computational Intelligence Magazine, vol. 15, 2020, pp. 44–51.

[3] M. V. da Silva, R. D. Souza, H. Alves, and T. Abrao, "A NOMA-based Q-learning random access method for machine type communications," in IEEE Wireless Communications Letters, vol. 9, 2020, pp. 1720—1724.

[4] N. Jiang, Y. Deng, and A. Nallanathan, "Traffic prediction and random access control optimization: Learning and non-learning-based approaches," in IEEE Communications Magazine, vol. 59, 2021, pp. 16–22.

[5] N. H. Mahmood, N. Pratas, T. Jacobsen, and P. E. Mogensen, "On the Performance of One Stage Massive Random Access Protocols in 5G Systems," in Turbo Codes and Iterative Information Processing (ISTC), pp. 340–344, 2016.

[6] C. J. Watkins, and P. Dayan, "Q-learning", in Machine Learning, vol. 8, 1992, pp. 279–292.

[7] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in International Conference on Learning Representations (ICLR), 2015, arXiv preprint arXiv:1509.02971.

[8] 3GPP TR 37.868, "RAN Improvements for Machine-type Communications."