Efficient Context-Aware Barring Scheme for Low-Latency 2-Step RACH in 5G Networks

Dawei Nie[1], Wenjuan Yu[1], Chuan Heng Foh[2], Qiang Ni[1], Luan Chen[3], Sara Berri[3], Arsenia Chorti[3], Hongjian Sun[4]

[1]School of Computing and Communications, InfoLab21, Lancaster University, Lancaster, UK

[2]5GIC & 6GIC, Institute for Communication Systems, University of Surrey, Guildford, Surrey, UK

[3]ETIS, UMR 8051 CY Cergy Paris University, ENSEA, CNRS, F-95000, Cergy, France

[4]Department of Engineering, Durham University, Durham, U.K.

Emails: {d.nie, w.yu8, q.ni}@lancaster.ac.uk, c.foh@surrey.ac.uk, {luan.chen, sara.berri, arsenia.chorti}@ensea.fr, hongjian.sun@durham.ac.uk

***Abstract*—The random access channel (RACH) procedure is critical for uplink synchronization and connection establishment in massive machine-type communication (mMTC). While 3GPP Release 16 introduced the 2-step RACH to reduce signaling overhead, latency incurred due to collisions remains a significant challenge in latency-critical mMTC scenarios. This paper proposes an enhanced 2-step RACH framework that integrates access class barring (ACB) with an action space-reduced contextual multi-armed bandit (AR-CMAB) agent to optimize performance. The proposed scheme dynamically adjusts the barring rate based on real-time traffic conditions, significantly reducing collisions, avoiding backoff delays, and minimizing overall access latency. Simulation results demonstrate that the AR-CMAB agent converges much faster than the benchmark scheme while achieving near-optimal latency, outperforming existing methods under varying traffic conditions.**

***Index Terms*—Two-step random access channel, contextual multi-armed bandit, access class barring.**

I. Introduction

The 3GPP Release 17 extends 5G connectivity beyond terrestrial boundaries by integrating non-terrestrial networks (NTNs), including unmanned aerial vehicles (UAVs), high altitude platforms (HAPs) and satellites, to leverage their extensive coverage capabilities [1]. To achieve seamless integration between terrestrial and non-terrestrial networks in 5G and beyond, the random access (RA) procedure[1], primarily used for uplink synchronization and connection establishment, needs to be studied. In this work, we focus on enhancing the random-access channel (RACH) procedure to lay the foundation for seamless integration. In 5G, massive machine-type communication (mMTC) aims to connect millions of devices per square kilometer [2]. However, achieving low-latency performance in such dense environments, which is essential for real-time applications like collaborative autonomous driving, remains a critical challenge for cellular systems [3].

Traffic bursts in mMTC significantly increase random access attempts to the base station (BS), causing collisions and prolonged latency [4]. User equipments (UEs) that fail to access the network normally backoff (BO) and retransmit, further contributing to latency and resource consumption. Effective solutions address these challenges by either managing access traffic or improving the performance of the access procedure. To manage traffic, LTE-A introduced access class barring (ACB), which stabilizes the RACH procedure by distributing UE access attempts over time [5]. To enhance access efficiency in 5G mMTC scenarios, 3GPP Release 16 proposed the 2-step RACH procedure. Compared to the conventional 4-step RACH, the 2-step RACH procedure reduces overhead and latency, thereby significantly improving UE access performance [6], [7].

Many studies have investigated strategies to reduce access latency and mitigate BO issues caused by access failures. In [8], Kim and Lee proposed a BO scheme to lower collision probability and improve access efficiency in IoT systems, providing insights of optimizing BO strategies for dense connectivity scenarios. Li *et al.* [9] introduced a QoS-based dynamic mechanism that prioritizes preamble allocation for delay-sensitive devices while adaptively adjusting ACB parameters to increase access success rates. Reinforcement learning (RL) approaches, such as Q-learning, have also been

This work was supported by HORIZON-MSCA-2022-SE-01 under the project TRACE-V2X (grant agreement ID 101131204), and in part by the EPSRC CHEDDAR TMF uplift - SustainAIRA6G project under award number EP/Y037421/1.

[1]This work focuses solely on the contention-based procedure; contention-free procedures are beyond the scope of this study.

applied to optimize ACB mechanisms, as shown in [10] and [11], effectively mitigating congestion and reducing random access latency. Several studies have also focused on enhancing the 2-step RACH procedure. For instance, Jones *et al.* [12] proposed the RAPID random access procedure for delay-sensitive devices, utilizing a Markov chain model to analyze random access load and latency and determine optimal preamble allocation. Subsequent research [13] further examined the 2-step RACH, highlighting its benefits in high-traffic scenarios.

Despite extensive research addressing the latency challenges in mMTC, many studies evaluated access latency by analyzing individual steps in isolation [3], [14]. Unlike previous studies, this work conducts a comprehensive evaluation of access latency, accounting for BO and complete iterations of the ACB and RACH procedures. Furthermore, while RL algorithms have advantages in dynamic decision-making, they often face challenges in training efficiency, convergence, and performance when dealing with large action spaces. To address these issues, this paper introduces an action space reduction method by leveraging insights from the relationship between barring rate and access latency. Finally, this work proposes an enhanced 2-step RACH procedure that integrates adaptive ACB with an intelligent agent. The primary contributions of this paper are summarized as follows:

- This paper proposes an enhanced 2-step RACH procedure that integrates adaptive ACB with an action space-reduced contextual multi-armed bandit (AR-CMAB) agent. The AR-CMAB agent dynamically optimizes the barring rate to reduce collisions, avoid unnecessary BOs and minimize access latency. Simulation results demonstrate its superior performance compared to conventional methods.
- To tackle the challenge of slow convergence caused by a large action space in barring rate selection, we introduce an action space reduction technique. This method achieves a 50% faster convergence speed compared to the baseline scheme, while maintaining superior performance in latency minimization.
- To evaluate its adaptability, simulations are designed to model varying UE traffic patterns in real-world scenarios. Results demonstrate that the agent effectively adjusts the barring rate in real-time, achieving near-optimal latency performance despite fluctuations in system loads and traffic dynamics.

II. System Model

In this section, we first introduce the ACB mechanism, followed by an overview of the two-step RACH procedure. The complete process, including ACB, RACH, and BO, is then described, along with the barring agent and the formula for access latency.

ACB is a UE access control mechanism that regulates access attempts, determining whether a UE can participate in the RACH procedure. It begins with the BS broadcasting a barring rate, b, constrained to the range $[0, 1]$, as part of the system information. Each UE who attempts to access generates a random number between 0 and 1. If it is smaller than b, the UE can participate in RACH by selecting and transmitting a preamble to the BS. If the generated number is larger than b, the device is barred and waits for the specified barring time. As one can see, b plays a key role here. A smaller b may reduce collisions but increase the waiting time to pass barring, potentially causing under-utilization of resources. Conversely, a larger b allows more devices to transmit, which can result in higher collision rates and increased latency. Therefore, determining the optimal b is crucial for system performance. By regulating access attempts, ACB ensures that the RACH process operates efficiently, especially during high-load scenarios.

A. 2-step RACH Procedure

Upon passing the ACB check, a UE is granted permission to initiate the 2-step RACH procedure. It can be viewed as a condensed version of the 4-step RACH, as illustrated in Fig. 1. Despite its reduced steps, the 2-step RACH involves more complex logic. First, a UE transmits Msg A, including the preamble and payload, over the physical random access channel (PRACH) and the physical uplink shared channel (PUSCH), respectively. In the 2-step RACH, the PUSCH resource for payload transmission is determined by a mapping sequence based on the selected preamble ID.

Upon receiving Msg A, the BS (or gNB) proceeds with detecting the preamble from PRACH and decoding the payload from PUSCH. Meanwhile, the UE waits for a response within a configured window W_{RAR}. This response, referred to as Msg B, varies depending on the outcomes of preamble detection and payload decoding. Different cases arise based on the outcomes of preamble detection (PBD), preamble reception time configuration (RTC), and payload decoding (PLD), as detailed below.

Case 1 Upon successful detection and decoding of the PRACH preamble and PUSCH payload, the BS sends Msg B, which includes a successful random access response (RAR) and a Timing Advance (TA) command. The UE, upon receiving Msg B, completes the 2-step RACH procedure.

Case 2 If the PRACH preamble is detected but PUSCH payload decoding fails due to channel errors or PUSCH resource contention [13], the BS uses

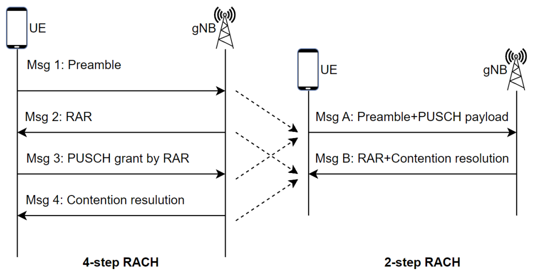

UE
gNB
Msg 1: Preamble
Msg 2: RAR
Msg 3: PUSCH grant by RAR
Msg 4: Contention resulution
4-step RACH
UE
gNB
Msg A: Preamble+PUSCH payload
Msg B: RAR+Contention resolution
2-step RACH

Fig. 1: Procedures for 4-step and 2-step RACH.

the preamble's reception time to send a fallback RAR in MsgB. The handshake then reverts to the third and fourth steps of the 4-step RACH.

Case 3 If the PRACH preamble is detected but the BS cannot determine its reception time, typically due to identical preambles from multiple UEs [13], [15], the preamble is deemed collided. The BS transmits a BO indication, instructing the UEs to retry random access after BO.

Case 4 If no response is received within the RAR window W_{RAR}, the UE assumes that MsgA was unsuccessful and therefore no RAR transmission occurs between the BS and the UE. The UE subsequently initiates a BO procedure.

The flowchart of the 2-step RACH procedure is presented in Fig. 2. The UE that successfully receives the contention resolution (CR) completes the random access procedure[2]. If the random access procedure fails, UEs enter the BO state and retransmit MsgA. If MsgA retransmissions reach the threshold m, the 2-step RACH procedure is deemed unsuccessful.

B. *Access Latency and Agent Integration*

As illustrated in Fig. 2, each UE attempting to establish a connection with the network may undergo multiple rounds of ACB and RACH procedures. Each step incurs processing time and contributes to overall latency. Let t_w, t_T and $\overline{t_{\text{BO}}}$ denote the barring waiting time, preamble transmission time, and the average BO time including RAR window, respectively. Additionally, t_{RAR} and W_{CR} represent the RAR transmission time and CR time window, respectively. Finally, let t_2 represent the UE processing delay combined with the transmission time of the radio resource control (RRC) message in Case 2. It is assumed that the UE transmits packets continuously over time. From a statistical perspective, the UE is expected to traverse all procedures with certain probabilities. The average latency can thus be expressed

[2]Any UE reaching the CR step is assumed to complete it successfully in this work.

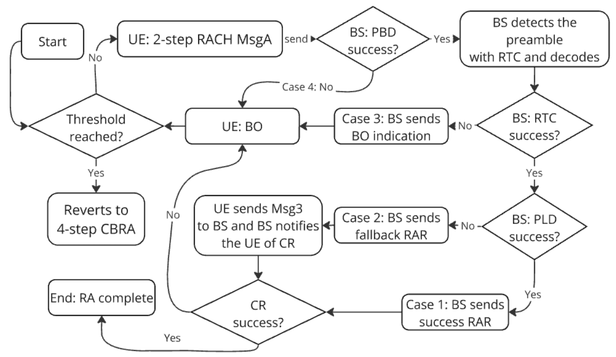


Fig. 2: Flowchart of the 2-step RACH procedure.

as the weighted sum of the probabilities of the UE encountering each procedure and the corresponding time consumption of that procedure.

$$\begin{aligned} T_{\text{ave}} = (&P_w t_w + P_T t_T + P_2(t_2 + W_{\text{CR}}) + P_{\text{BO}}\overline{t_{\text{BO}}} \\ &+ P_1(t_{\text{RAR}} + W_{\text{CR}}))S_{\text{ave}}. \end{aligned} \tag{1}$$

Here, P_w, P_T, and P_{BO} denote the probabilities of a UE undergoing barring waiting, preamble transmission, and BO procedures, respectively. Likewise, P_1 and P_2 represent the probabilities of a UE encountering Case 1 and Case 2. S_{ave} represents the average steps for a UE to complete one single round of random access procedure. Fig. 3 illustrates an example of the time structure for a UE performing random access under high-traffic conditions. It can be observed that the time spent in the BO state constitutes the majority of the overall latency. Optimizing the barring rate and reducing collisions can improve the success probability of random access, thereby reducing the time spent in the BO state and ultimately reducing the overall latency.

In real-world systems, traffic load fluctuates in real time. To minimize latency, it is essential to design an intelligent agent capable of dynamically adjusting the barring rate and responding quickly to fluctuating load conditions based on feedback. The model of the proposed scheme, which integrates an intelligent agent with ACB and the 2-step RACH, is illustrated in Fig. 4.

III. Agent Design

A. *CMAB Configuration*

In this section, we introduce the design of the proposed agent, beginning with an introduction of the multi-armed bandit (MAB) framework, a suitable approach for dynamically adjusting the barring rate in ACB. By using MAB, decisions are made from an action space $\mathcal{A}$, where each action, referred to as an "arm", generates a stochastic reward observable after the action is chosen. The primary objective is to maximize cumulative

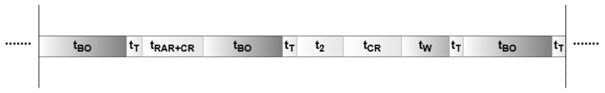


Fig. 3: Example of the time structure for a typical UE.

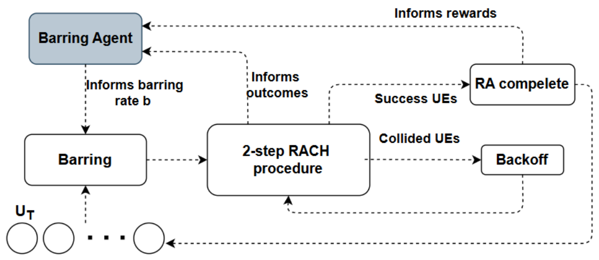


Fig. 4: System model.

rewards by identifying the optimal arm. In this work, we adopts contextual MAB (CMAB), which uses contextual information for decision-making. To capture relevant environmental details, the context $\mathcal{C}$ in this study includes the current preamble outcomes and the barring rate b.

As one can note, the choice of context in CMAB is critical, as it reflects the environment and directly influences decision-making. Let us now justify the rationale behind our context design by providing a theoretical analysis of preamble selection and transmission. The system begins with U_T incoming UEs undergoing ACB. Let U_s represent the number of UEs that successfully pass ACB; each of these UEs independently selects and transmits one preamble from M available preambles. Meanwhile, some UEs that failed in previous rounds of RACH and experienced BO are rejoining the process. Assume there are U_{BO} such UEs. These UEs, along with newly arriving ones, will rejoin the RACH procedure by re-transmitting MsgA. Thus, the total number of UEs involved in preamble transmission is given by $\bar{N} = U_s + U_{BO}$. Since each UE randomly selects a preamble without knowledge of others' selections, preamble collisions may occur when multiple UEs choose the same preamble and transmit it simultaneously. The probability that a specific preamble is transmitted by only one UE is given by $\binom{\bar{N}}{1}\frac{1}{M}(1-\frac{1}{M})^{\bar{N}-1}$. Let N_s represent the expected number of successful UEs and N_c represent the number of collided UEs. N_s can be calculated as $N_s = \bar{N}\left(1-\frac{1}{M}\right)^{\bar{N}-1}$. Since there are $\bar{N}$ UEs in total, the number of collided UEs is $N_c = \bar{N} - N_s$. To statistically characterize the system's operating conditions at steady state, we define the tuple $\bar{\mathcal{N}} = \{N_s, N_c, \bar{N}\}$, which reflects the system load. This tuple and the barring rate form the context for the CMAB algorithm, enabling adaptation to UE traffic variations.

B. Reduction of Action Space

Typical MAB algorithms select actions either randomly or based on historical data; however, when known correlations exist between actions and rewards, such exploration wastes resources and slows convergence. Building on the ACB analysis in Section II, we propose a mechanism to reduce the action space for a given context, thereby improving the algorithm's convergence speed. This method, termed action space-reduced contextual multi-armed bandit (AR-CMAB), leverages the insight that there is a unique optimal barring rate minimizing access latency. By comparing the latencies of two recent barring rates, the mechanism narrows the selection direction and range for the next rate, refining exploration.

The action space of the agent, denoted as $\mathcal{B}$, consists of a set of barring rates, represented as $\mathcal{B} = \{S, S+\delta b, S+2\delta b, ..., E\}$, where S and E are the minimum and maximum barring rates, and δb is the step size. We now explain how the action space is updated based on the feedback. Assume that in two consecutive barring rounds, the agent selects barring rates b^{i-1} and b^i, resulting in latencies T^{i-1} and T^i, respectively. If $T^i > T^{i-1}$ when $b^i > b^{i-1}$, it indicates that the optimal barring rate does not lie in the range greater than b^i, and thus the upper limit of the action space should be updated to $E = b^i$. Conversely, if $T^i < T^{i-1}$ when $b^i > b^{i-1}$, it suggests that the optimal barring rate lies within the range greater than b^{i-1}, and the lower limit should be updated to $S = b^{i-1}$. Similarly, if $T^i > T^{i-1}$ when $b^i < b^{i-1}$, then $S = b^i$, while if $b^i < b^{i-1}$ and $T^i < T^{i-1}$, then $E = b^{i-1}$. We summarize the above strategies as $\pi(S, E \mid b^i, b^{i-1}, T^i, T^{i-1})$. The next barring rate is then chosen from the updated action space $\mathcal{B}$. By iteratively selecting barring rates and observing the corresponding latencies, the action space $\mathcal{B}$ is progressively narrowed, ultimately converging to the optimal barring rate.

C. AR-CMAB Agent Design

Let us now introduce the design of the AR-CMAB agent. Algorithm 1 provides an overview of the process for adjusting the barring rate b^i during the barring round i. In each round, the agent selects a barring rate from its action space $\mathcal{B}$ and executes the 2-step RACH procedure. Due to system delays, changes to the barring rate do not immediately affect the observed reward, resulting in a delay of at least δt. As a result, the agent evaluates changes in b^i over a period of δt or longer. Upon receiving the reward for the previously selected barring rate, the agent captures the current tuple $\bar{\mathcal{N}}^i = \{N_s^i, N_c^i, \bar{N}^i\}$, representing the observed system state for this barring round. This tuple is incorporated into the context matrix $\mathcal{C}$, where the context is primarily determined by $\bar{\mathcal{N}}$. Each unique $\bar{\mathcal{N}}$ is assigned an action space, denoted as $\mathcal{B}^{\bar{\mathcal{N}}}$.

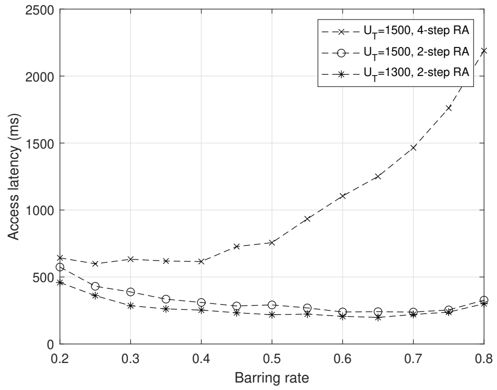


Fig. 5: Access latency of RACH procedures with ACB.

The selection of the next barring rate follows the action space reduction strategy outlined in Section III-B.When the action space $\mathcal{B}^{\bar{\mathcal{N}}}$ for a specific context converges to the optimal barring rate, the context is classified as *seen*. For seen contexts, the AR-CMAB agent utilizes the learned optimal arm. For unseen contexts, the agent initializes a new action space $\mathcal{B}^{\bar{\mathcal{N}}}$ and explores it. After sufficient exploration, if a new traffic condition matches a previously seen context, the agent directly selects the optimal barring rate, enabling faster decision-making and improved adaptability to dynamic traffic conditions. This algorithm can be effectively applied in real-world scenarios; however, δt must be adjusted according to the total number of users to avoid increased latency or diminished algorithm accuracy.

IV. Simulation Results

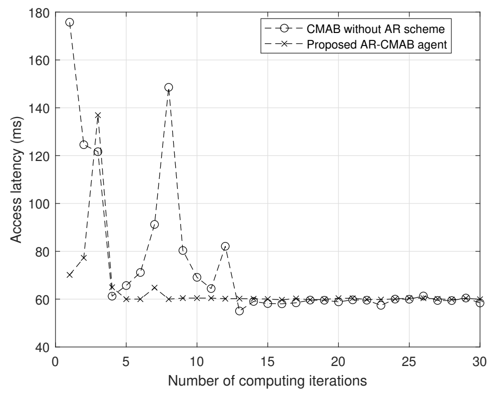
In this section, we present simulation results to evaluate the performance of the proposed scheme in comparison with benchmark approaches. The convergence behaviour and accuracy of the designed algorithm are also shown. For the latency simulations, the following parameters are assumed unless otherwise specified: a preamble transmission time of 1ms, a barring waiting time of 4ms, an average BO time of 25ms, a BO time threshold of 15ms, a RAR window size of 4ms, a Case 2 delay of 3ms, and a CR time window of 5ms.

Fig. 5 compares the average access latency of the proposed 2-step RACH scheme against a benchmark scheme (conventional 4-step RACH) under varying barring rates and different values of U_T. The results show that for a fixed barring rate, access latency increases with the number of incoming UEs. Furthermore, the latency initially decreases as the barring rate increases, reaching a minimum before rising again, thereby confirming the existence of an optimal barring rate that minimizes latency in the 2-step RACH. This behaviour arises because a low barring rate increases the likelihood of repeated barring attempts, leading to longer waiting times, while a high barring rate causes preamble collisions, resulting in higher transmission failures and BO delays. For comparison, the upper curve in Fig. 5 represents the access latency of the traditional 4-step RACH. The results clearly show that, across all barring rates, the proposed 2-step scheme achieves lower latency, primarily due to the reduced steps in the RA process.

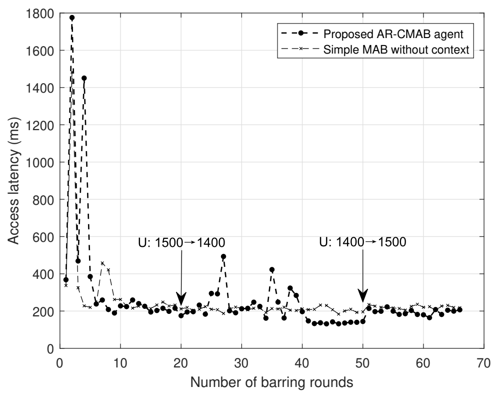
Algorithm 1 Barring Algorithm with AR-CMAB Agent

```
Input: $M$, $\delta b$, $\delta t$, $\mathcal{C} = \emptyset$, $i = 0$, $t_0 = 0$, $b^0 = 0.5$
 1: while RACH is on do
 2:     Barring round $i$ begins, set barring rate as $b^i$
 3:     Obtain $[N_c^i, N_f^i]$ of $\bar{\mathcal{N}}^i$ and $T^i$ at time $t_i + \delta t$
 4:     Set reward $R_i = -T^i$.
 5:     if $\bar{\mathcal{N}}^i \in \{[\bar{\mathcal{N}}, R, seen] | \mathcal{C}\}$ then
 6:         Set $i \leftarrow i + 1$, $t_i \leftarrow$ Current time
 7:         Set $b^i \leftarrow \underset{[\bar{\mathcal{N}}, R] \in \mathcal{C}, \forall \bar{\mathcal{N}} = \bar{\mathcal{N}}^i}{\arg\max} R$
 8:         continue
 9:     else if $i < 1$ or facing a new context then
10:         $S = 0$, $E = 1$
11:     else
12:         $\pi(S, E \mid b^i, b^{i-1}, T^i, T^{i-1})$
13:     end if
14:     $\mathcal{B}_{\bar{\mathcal{N}}^i} = \{S : \delta b : E\}$, $\mathcal{C} \leftarrow \mathcal{C} \bigcup \{[\bar{\mathcal{N}}^i, R^i]\}$
15:     if length$(\mathcal{B}_{\bar{\mathcal{N}}^i}) \leq 3$ then
16:         Mark the context of $\bar{\mathcal{N}} = \bar{\mathcal{N}}^i$ as seen.
17:         Set $\mathcal{C} \leftarrow \mathcal{C} \bigcup \{[\bar{\mathcal{N}}, R, seen]\}$
18:         Set $i \leftarrow i + 1$, $b^i \leftarrow \underset{[\bar{\mathcal{N}}, R] \in \mathcal{C}, \forall \bar{\mathcal{N}} = \bar{\mathcal{N}}^i}{\arg\max} R$
19:     else
20:         Set $i \leftarrow i + 1$, $b^i \leftarrow$ a random arm $\in \mathcal{B}_{\bar{\mathcal{N}}^i}$.
21:     end if
22:     set $t_i \leftarrow$ Current time
23: end while
```

To evaluate the effectiveness of the action space reduction, Fig. 6(a) illustrates the access latency convergence over multiple barring rounds. As different b are explored, the b corresponding to the minimum converged latency is determined to be the optimal solution. The benchmark scheme, classic CMAB without action space reduction, performs random action selection without replacement. Results show that the proposed scheme with action space reduction achieves over 50% faster convergence compared to the benchmark. To validate the AR-CMAB algorithm's ability to handle dynamic traffic variations, Fig. 6(b) presents a scenario where the number of UEs, U_T, decreases from 1500 to 1400 at barring round 20 and reverts to 1500 at round 50. During the transition from $U_T = 1500$ to 1400, the AR-CMAB agent encounters a new context and enters an exploration phase (rounds 20–40), characterized by



(a) Latency variation trends versus the number of barring rounds between the CMAB agent and the benchmark scheme.



(b) Access latency versus the number of barring rounds.

Fig. 6: Comparison between the proposed AR-CMAB agent and benchmark schemes.

latency fluctuations, before successfully converging to the optimal latency for $U_T = 1400$ in rounds 40–50. In contrast, the simple classic MAB scheme, which lacks context awareness, fails to adapt to the traffic change, resulting in higher latency in rounds 40–50 (after the AR-CMAB agent has converged). When traffic conditions revert to $U_T = 1500$ after round 50, the AR-CMAB agent efficiently recalls the previously explored optimal barring strategy, achieving a latency value closely aligned with the theoretical optimal latency shown in Fig. 5, without requiring additional exploration. This adaptability allows the AR-CMAB algorithm to outperform the benchmark scheme, which cannot dynamically adjust to such variations. Overall, the AR-CMAB algorithm demonstrates superior adaptability and achieves lower BO rates and latency under dynamic traffic conditions compared to the context-free MAB scheme.

V. Conclusions

This paper proposed an enhanced 2-step RACH framework integrating adaptive ACB with a context-aware intelligent agent to address latency and collision challenges in mMTC scenarios. By introducing AR-CMAB model, the framework achieves faster convergence and dynamically optimizes the barring rate under varying traffic conditions. Simulation results demonstrated significant latency reductions and superior performance compared to benchmark methods. The proposed scheme provides a scalable and efficient solution for low-latency access in latency-critical mMTC scenarios.

References

[1] 3rd Generation Partnership Project (3GPP), "Release 17 Description; Summary of Rel-17 Work Items," 3GPP, Technical Report TR 21.917, 2022.

[2] W. Yu, C. H. Foh, A. U. Quddus, Y. Liu, and R. Tafazolli, "Throughput analysis and user barring design for uplink noma-enabled random access," *IEEE Transactions on Wireless Communications*, vol. 20, no. 10, pp. 6298–6314, 2021.

[3] S. Park, S. Lee, and W. Choi, "Markov chain analysis for compressed sensing based random access in cellular systems," in *2019 International Conference on Computing, Networking and Communications (ICNC)*, 2019, pp. 34–38.

[4] M. Grau, C. H. Foh, A. u. Quddus, and R. Tafazolli, "Preamble barring: A novel random access scheme for machine type communications with unpredictable traffic bursts," in *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*, 2019, pp. 1–7.

[5] 3GPP TS 22.011, "Technical specification group services and system aspects," 3GPP, Tech. Rep., Release 13, 2016.

[6] 3GPP RP-190711, "3GPP work item description, 2-step rach for NR," 3GPP, Tech. Rep., Sep. 2019.

[7] 3GPP TS 38.331, "NR; radio resource control (RRC); protocol specifi- cation," 3GPP, Tech. Rep., ver. 15.8.0, Jan. 2020.

[8] J. Kim and S. Lee, "Delayed response and random backoff first for low-power wide-area iot networks," *Sensors*, vol. 23, no. 23, p. 9556, 2023.

[9] L. Zhao, X. Xu, K. Zhu, S. Han, and X. Tao, "Qos-based dynamic allocation and adaptive acb mechanism for ran overload avoidance in mtc," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6.

[10] L. Tello-Oquendo, D. Pacheco-Paramo, V. Pla, and J. Martinez-Bauset, "Reinforcement learning-based acb in lte-a networks for handling massive m2m and h2h communications," in *IEEE Int. Conf. Commun (ICC)*, 2018, pp. 1–7.

[11] M. Jihun and L. Yujin, "A reinforcement learning approach to access management in wireless cellular networks," *Wireless Communications & Mobile Computing*, vol. 2017, pp. 1–7, 2017.

[12] J. Kim, S. Kim, T. Taleb, and S. Choi, "Rapid: Contention resolution based random access using context id for iot," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 7, pp. 7121–7135, 2019.

[13] J. Kim, G. Lee, S. Kim, T. Taleb, S. Choi, and S. Bahk, "Two-step random access for 5g system: Latest trends and challenges," *IEEE Network*, vol. 35, no. 1, pp. 273–279, 2021.

[14] Z. Li, L. Tian, Y. Yin, and W. Cao, "On contention-based 2-step random access procedure," in *2020 International Conference on Wireless Communications and Signal Processing (WCSP)*, 2020, pp. 771–776.

[15] 3rd Generation Partnership Project(3GPP), "3GPP TS 38.213 V16.6.0 (2021-06) Technical Specification," 3GPP; Technical Specification Group Radio Access Network, Technical Specification 38.213.