Resource Allocation in Fog RAN for Heterogeneous IoT Environments based on Reinforcement Learning

Almuthanna Nassar, and Yasin Yilmaz, *Member, IEEE*
Electrical Engineering Department, University of South Florida, Tampa, FL 33620, USA
E-mails: {atnassar@mail.usf.edu; yasiny@usf.edu}

***Abstract*—Fog radio access network (F-RAN) has been recently proposed to satisfy the low-latency communication requirements of Internet of Things (IoT) applications. We consider the problem of sequentially allocating the limited resources of a fog node to a heterogeneous population of IoT applications with varying latency requirements. Specifically, for each service request, the fog node needs to decide whether to serve that user locally to provide it with low-latency communication service or to refer it to the cloud control center to keep the limited fog resources available for future users. We formulate the problem as a Markov Decision Process (MDP), for which we present the optimal decision policy through Reinforcement Learning (RL). The proposed resource allocation method learns from the IoT environment how to strike the right balance between two conflicting objectives, maximizing the total served utility and minimizing the idle time of the fog node. Extensive simulation results for various IoT environments corroborate the theoretical underpinnings of the proposed RL-based resource allocation method.**

***Index Terms*—Resource Allocation, Fog RAN, 5G, IoT, Markov Decision Process, Reinforcement Learning, Low-Latency Communication.**

I. Introduction

To tackle the challenge of massive Internet of Things (IoT) and the increasing amount of mobile traffic for better user satisfaction, cloud radio access network (C-RAN) architecture was proposed for 5G, in which a powerful cloud controller with pool of baseband units and storage pool supports large number of distributed remote radio units through high capacity fronthaul links [1]. The C-RAN is characterized by being clean as it reduces energy consumption and improves the spectral efficiency due to the centralized processing and collaborative radio. However, in light of the massive IoT applications and the corresponding generated traffic [2], [3], C-RAN structure places a huge burden on the centralized cloud controller and its fronthaul, which causes more delay due to limited fronthaul capacity and busy cloud servers in addition to the large transmission delays [4]. The latency limitation in C-RAN becomes a critical issue for IoT applications which are sensitive to large delays. To this end, an evolved architecture, Fog RAN (F-RAN) was introduced for 5G to extend the inherent operations and services of the cloud. In F-RAN, the fog nodes (FNs) are not only limited to perform RF functionalities but also empowered with caching, signal processing and computing resources. This makes FNs capable of independently delivering network functionalities to end users at the edge of the network, without referring the users to the cloud, to fulfill the low-latency demand [5]. IoT applications have various latency requirements. Some applications are satisfied by the traditional mobile broadband services of high throughput and capacity while some other IoT applications seek ultra-reliable low-latency communication [6]. Hence, especially in a heterogeneous IoT environment with various latency needs and limited F-RAN capacity, FN must allocate its limited and valuable resources in a smart way. In this work, we present a novel framework for resource allocation in F-RAN for 5G to guarantee the efficient utilization of limited FN resources while satisfying the low-latency requirements of IoT applications in various environments.

Recently, a large number of works in the literature focused on achieving low latency for IoT applications in 5G F-RAN. For instance, resource allocation based on cooperative computing at the edge to achieve low latency in F-RAN has been studied by [7], [8], [9], and [10]. Instead of utilizing the cloud server, edge mesh as a computing paradigm is proposed in [7]. To achieve ultra low latency, cooperative task computing across multiple F-RAN nodes, which utilize the current infrastructure of small cells and macro base stations, was considered in [8]. The number of F-RAN nodes and their locations have been investigated by [11]. The work in [10] proposed local small-cell clusters to balance the computing load for improved quality of experience. Content fetching is suggested in [1], [9] to maximize the delivery rate when the requested content is available in the cache of fog nodes. The congestion problem, when resource allocation is done based on the best signal quality received by the end user, is studied by [12]. To maximize the quality of experience, the work in [13] proposed soft resource reservation mechanism for the uplink scheduling. The model-free reinforcement learning approach is used in [14] to learn the optimal policy for user scheduling in heterogeneous networks to maximize the network energy efficiency. With regard to learning for IoT, [15] provided a comprehensive study about the advantages, limitations, applications, and key results relating to machine learning, sequential learning, and reinforcement learning. Multi-agent reinforcement learning was exploited in [16] to maximize

978-1-5386-8088-9/19/$31.00

network resource utilization in heterogeneous network by selecting the radio access technology and allocating resources for individual users.

With the motivation of satisfying the low-latency requirements of heterogeneous IoT applications through F-RAN, we provide a novel framework for allocating limited resources to low-latency users, that guarantees efficient utilization of limited FN resources. In this work, we develop a Markov Decision Process (MDP) formulation for the considered resource allocation problem and provide a Reinforcement Learning (RL) algorithm for learning the optimum decision-making policy adaptive to the IoT environment.

The remainder of the paper is organized as follows. Section II introduces the system model. The proposed MDP formulation for the resource allocation problem is given in Section III. Optimal policy and the related RL algorithm are discussed in Section IV. Simulation results are presented in Section V. Finally, we conclude the paper in Section VI.

II. System Model

We consider the F-RAN structure shown in Fig. 1, in which fog-nodes (FNs) are connected through the fronthaul to the cloud controller, where a massive computing capability, centralized baseband units and cloud storage pooling are available. FNs are equipped with caching, computing and signal processing capabilities to deliver network functionalities at the edge. However, these resources are limited and therefore, need to be utilized efficiently. An end user attempts to access the network by sending a request to the nearest FN. The FN takes a decision whether to serve the user locally at the edge using its own computing and processing resources or refer it to the cloud. We consider the FN's computing and processing capacity is limited to N slots. User requests arrive sequentially and decisions are taken quickly, so no queuing occurs. We assume that the time taken to fill the slots is much shorter than the average user serving time, i.e., no in-use slot becomes available while making decisions to fill the slots.

The quality of service (QoS) requirements of a wireless user are typically given by the latency requirement and throughput requirement. IoT applications have various levels of latency requirement, hence it is sensible for the FN to give higher priority to the low-latency applications. To differentiate between similar latency requirements we also consider the risk of failing to satisfy the throughput requirement. This risk is related to the ratio of the achievable throughput to the throughput requirement. The achievable throughput is characterized by the signal-to-noise ratio (SNR) through Shannon channel capacity. Shannon's fundamental limit on the capacity of a communications channel gives an upper bound for the achievable throughput, as a function of available bandwidth (B) and SNR, $C = B\,log_2(1 + \text{SNR})$. Hence, we define the utility of an IoT user request to be a function of latency requirement, l (in milliseconds), throughput requirement, τ (in bits per second), and channel capacity, C (in bits per second), i.e., $u = f(l, \tau, C)$. Since the utility should be inversely proportional to the latency requirement, and directly proportional to the achievable throughput ratio, $\mu = C/\tau$, we define utility as

$$u = \kappa(\mu^{\alpha}/l^{\beta}), \tag{1}$$

where $\kappa, \alpha, \beta > 0$ are mapping parameters. This provides a flexible model for utility. By selecting the parameters κ, α, β, a desired range of u and importance levels for latency and throughput requirements can be obtained. Since F-RAN is intended for satisfying low-latency requirements, typically, more weight should be given to latency by choosing larger β values.

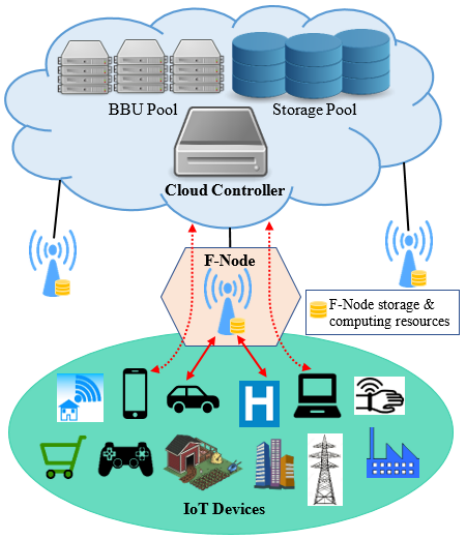

Fig. 1. Fog-RAN system model. The FN serves heterogeneous latency needs in IoT environment, and is connected to the cloud through the fronthaul links. Solid lines represent local service by FN to satisfy low-latency requirements, and dashed lines represent referral to the cloud to save limited resources.

FNs should be smart to learn how to decide (serve/refer to the cloud) for each request (i.e., how to allocate its limited resources), so as to achieve the conflicting objectives of maximizing the average total utility of served users over time and minimizing its idle (no-service) time. One approach to deal with this resource allocation problem is to apply a fixed threshold on the user utility. For instance, we can define a threshold rule, such as "serve if $u > 5$", if we classify all applications in an IoT environment into ten different utilities $u \in \{1, 2, ..., 10\}$, 10 being the highest utility. However, such a policy is sub-optimum since the FN will be waiting for a user to satisfy the threshold, which will increase the idle time. The main drawback of this policy is that it cannot adapt to the dynamic IoT environment to achieve the objective. For instance, when the user utilities are almost uniformly distributed, a very selective policy with a high threshold will stay idle most of the time, whereas an impatient policy with a low threshold will in general obtain a low average served utility. A mild policy with threshold 5 may in general perform better than the extreme policies, yet it will not be able adapt to different IoT environments.

TABLE I
STATE TRANSITIONS OF 2-SLOT FN

S	a	r	S'
S_0	$Wait$	η	S_0
S_0	$Serve$	u	S_1
S_1	$Wait$	η	S_1
S_1	$Serve$	u	S_2

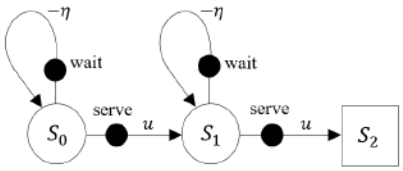

Fig. 2. State transition graph for MDP with $N = 2$. States are labeled by S_n where n is the number of filled slots.

A better solution for the F-RAN resource allocation problem is to use RL techniques which can continuously learn the environment and adapt the decision rule accordingly.

III. MDP PROBLEM FORMULATION

We formulate the Fog-RAN resource allocation problem in the form of infinite-horizon Markov decision process (MDP). The state S of the FN at any time is defined as the number of available slots at that time. The future state is independent of past states given the current state, hence we have Markov states. For an FN that has N slots of resources, there are $N + 1$ states, $S \in \{S_0, ..., S_n, ..., S_N\}$, where n denotes the number of occupied slots. Hence, the initial state of FN is S_0, while S_N represents the terminal state with all slots are occupied. At every time step t, the FN receives a request from a user with utility u_t, and FN takes an action $a_t \in \{\text{serve, wait}\}$. For tractability, we consider quantized utility values, such as $u_t \in \{1, 2, ..., 10\}$. Based on its decision, the FN receives an immediate reward r_t and moves to a successor state: either moves to a new state or remains at the current state. When the FN takes the action $a_t = \text{serve}$, then it will gain the user's utility value as a reward $r_t = u_t$ and one slot of the FN's resources will be occupied. Otherwise, for the action $a_t = \text{wait}$ (i.e., refer the user to the cloud), the FN will maintain its available resources and it will get a reward of $r_t = -\eta$, where η is the penalty for waiting, whose role is to encourage less idle time. The MDP terminates at time T when the terminal state S_N is reached. State transitions for the 2-slot case ($N = 2$) are shown in Table I. Being at state S, and taking the action a will result in getting an immediate reward r and moving to the successor state S'. All possible combinations of states and actions are shown in Table I.

We can also draw the dynamics of the MDP using a state transition graph as shown in Fig. 2, in which non-terminal states and terminal states are represented by circle and square, respectively; small filled circles represent actions, and arrows show the transitions with corresponding rewards.

We define the return G_t as the total discounted rewards received from time t till termination, $G_t = \sum_{j=0}^{\infty} \gamma^j r_{t+j}$, where $\gamma \in [0, 1]$ is the discount factor which represents the weight (i.e., importance) of future rewards with respect to the immediate reward, and after termination $r = 0$. Specifically, $\gamma = 0$ ignores the future rewards, whereas $\gamma = 1$ means that the future rewards are of the same importance as the immediate reward. Starting at the initial state $S = S_0$, the objective is to find the optimal decision policy which maximizes the expected initial return $\mathbb{E}[G_0]$.

The state-value function $V(S_0)$, where $V(S)$ is shown in (2), is equal to the objective function $\mathbb{E}[G_0]$. $V(S)$ represents the long-term value of being in state S in terms of the expected return which can be collected starting from this state onward till termination. Hence, the terminal state has zero value. The state value can be viewed also in two parts: the immediate reward from the action taken and the discounted value of the successor state where we move to.

Similarly, we define the action-value function $Q(S, a)$ as the expected return that can be achieved after taking the action a at state S, as shown in (3). The action-value function tells how good it is to take a particular action at a given state. The expressions in (2) and (3) are known as the Bellman expectation equations for state value and action value, respectively [17],

$$V(S) = \mathbb{E}[G_t|S] = \mathbb{E}[r_t + \gamma V(S')|S], \quad (2)$$

$$Q(S, a) = \mathbb{E}[G_t|S, a] = \mathbb{E}[r_t + \gamma \max_{a'} Q(S', a')|S, a], \quad (3)$$

where a' denotes the successor action at the successor state S'.

The objective of the FN in the presented MDP is to utilize the N resource slots for high-utility IoT applications in a timely manner. This can be achieved by maximizing the value of initial state $V(S_0) = \mathbb{E}[G_0]$. To this end, an optimal decision policy is required, which is discussed in the following section.

IV. OPTIMAL POLICY

A policy can be defined as the set of probabilities of taking a particular action given the state, i.e., $\pi = \{P(a|S)\}_{a,S}$ for all possible state-action pairs. The policy π is said to be optimal if it maximizes the value of all states, i.e., $\pi^* = \arg\max_{\pi} V_\pi(S), \forall S$. Hence, to find the optimal policy we need to find the optimal state-value function $V^*(S) = \max_{\pi} V_\pi(S)$, which selects the best action at each state. Defining the optimal action-value function $Q^*(S, a) = \max_{\pi} Q_\pi(S, a)$, from (2) and (3), we can write the optimal state-value function as,

$$V^*(S) = \max_a Q^*(S, a) = \max_a \mathbb{E}[r_t + \gamma V^*(S')|S, a]. \quad (4)$$

The notion of optimal state-value function $V^*(S)$ greatly simplifies the search for optimal policy. Since the goal of maximizing the expected future rewards is already taken care of the optimal value of the successor state, $V^*(S')$ can be taken out of the expectation in (4). Hence, the optimal policy is given by the best local actions at each state,

$$a^* = \arg\max_a \mathbb{E}[r_t|S, a] + \gamma V^*(S'|S, a). \quad (5)$$

In our problem, firstly the user arrives, then we make a decision to serve or wait (refer to the cloud), meaning that the reward u for serving and the reward $-\eta$ for waiting are known at the time of decision making. Thus, from (5), the optimal action at state S_n is given by

$$a_n^* = \begin{cases} \text{serve if } u + \gamma V^*(S_{n+1}) > -\eta + \gamma V^*(S_n), \\ \text{wait otherwise,} \end{cases} \tag{6}$$

for $n = 1, ..., N-1$, where $V^*(S_N) = 0$. Hence, the optimal decision rule is a thresholding on u, given by

$$a_n^* = \begin{cases} \text{serve if } u > h_n, \\ \text{wait otherwise,} \end{cases} \tag{7}$$

where $h_n = -\eta + \gamma[V^*(S_n) - V^*(S_{n+1})]$ denotes the optimal threshold at state S_n. The optimal state values, required by the optimal policy, as shown in (7), can be computed through the value iteration technique. The procedure to learn the optimal policy from the IoT environment using the Monte Carlo algorithm is given in Algorithm 1. Since our focus in this paper is to promote the RL-based approach against the conventional fixed-threshold approach, we do not elaborate on the choice of the RL method. The choice of the Monte Carlo method in this work is due its simplicity and straightforward convergence according to the law of large numbers. Other RL methods, such as Q-learning and SARSA, can be used as well to learn the optimal policy, but the comparison results with respect to the fixed-threshold approach (see Section V) would not change. Note that since the dimensionality of the state space is tractable in the considered problem, the tabular RL methods works well, thus there is no need for function-approximation methods, such as gradient-descent and deep RL methods [17].

Algorithm 1 Learning Optimum Policy for MDP

1: Select: $\gamma \in [0, 1]$, $\eta \in \mathbb{R}$;
2: Input: *Returns(S)*: is an array to save states' returns in all iterations;
3: Initialize: $V(S) \leftarrow 0$, $\forall S$;
4: **for** iteration $= 0, 1, 2, ...$ **do**
5: Initialize: $S \leftarrow S_0$;
6: Generate an episode: Take actions using (7) until termination;
7: $G(S) \leftarrow$ sum of discounted rewards from S till terminal state for all states appearing in the episode;
8: Append $G(S)$ to *Returns(S)*;
9: $V(S) \leftarrow$ average(*Returns*(S));
10: **if** $V(S)$ converges for all S **then**
11: **break**
12: $V^*(S) \leftarrow V(S)$, $\forall S$;
13: **end if**
14: **end for**
15: Use the estimated $V^*(S)$ to find optimal actions using (7).

TABLE II
UTILITY DISTRIBUTION IN VARIOUS IOT ENVIRONMENTS

	φ_1	φ_4	φ_7	φ_{10}	φ_{15}	φ_{19}
$P(u=1)$	0.015	0.012	0.01	0.008	0.004	0.001
$P(u=2)$	0.073	0.062	0.05	0.038	0.019	0.004
$P(u=3)$	0.365	0.308	0.25	0.192	0.096	0.019
$P(u=4)$	0.292	0.246	0.2	0.154	0.077	0.015
$P(u=5)$	0.205	0.172	0.14	0.108	0.054	0.011
$P(u=6)$	0.014	0.057	0.1	0.142	0.214	0.271
$P(u=7)$	0.013	0.051	0.09	0.129	0.193	0.244
$P(u=8)$	0.011	0.046	0.08	0.114	0.171	0.217
$P(u=9)$	0.009	0.034	0.06	0.086	0.129	0.163
$P(u=10)$	0.003	0.012	0.02	0.029	0.043	0.055
$\rho = P(u>5)$	5%	20%	35%	50%	75%	95%
$\mathbb{E}[u]$	3.82	4.4	4.97	5.55	6.5	7.27

V. SIMULATIONS

We next provide simulation results to compare the performance of the FN when implementing the proposed RL-based resource allocation algorithm, given in Algorithm 1, with the FN performance when a fixed thresholding algorithm is employed. We consider that the FN is equipped with computing and storage resources of five slots, i.e., $N = 5$. We evaluate the performances in various IoT environments with different compositions of latency requirements. For brevity, we do not consider the effect of ratio of the achievable throughput to the throughput requirement in assessing the utility of a service request. Specifically, we consider 10 utility classes with different latency requirements to exemplify the variety of IoT applications in an F-RAN setting. That is, we consider $\alpha = 0, \beta = 1, \kappa = 1$ in equation (1), and discretize the latency-based utility to 10 classes. The utility values $1, 2, ..., 10$ may represent the following IoT applications, respectively: smart farming, smart retail, smart home, wearables, entertainment, smart grid, smart city, industrial Internet, autonomous vehicles, and connected health. By changing the composition of utility classes, we generate 19 scenarios, 6 of which are summarized in Table II. Higher percentages of high-utility users make the IoT environment richer.

Denoting an IoT environment of particular statistics with φ, in Table II we show the statistics of φ_1, φ_4, φ_7, φ_{10}, φ_{15}, and φ_{19}. The last two rows in Table II show the probability ρ of utility being greater than 5, and the expected value of u, respectively. The first 10 rows in the table provide detailed information given by the probability of each utility value in an IoT environment. In the considered 19 scenarios, ρ increases by 0.05 from 5% to 95% for $\varphi_1, \varphi_2, ..., \varphi_{19}$ respectively. The remaining 13 scenarios have statistics proportional to their ρ values. We started with a general scenario given by φ_7, and changed ρ to obtain the other scenarios. Considering the MDP formulation for the IoT environment given by scenario φ_7 and an FN with 5 slots of resources Fig. 3 shows how the FN learns the optimal policy using Algorithm 1 with $\eta = 1$ and $\gamma = 1$. By interaction with the environment, the FN updates the state-value function which converges to the optimum policy. It is seen in Fig. 3 that the optimal policy is learned quickly as

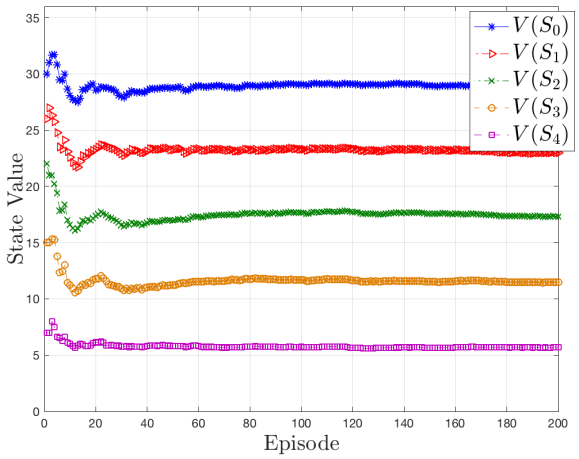

Fig. 3. Learning optimum policy of MDP, with $N = 5$, $\eta = 1$ and $\gamma = 1$, by applying RL algorithm given by Algorithm 1 to obtain the optimal state values.

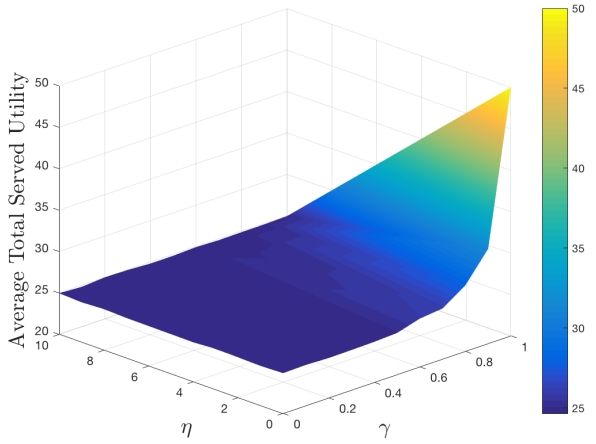

Fig. 4. The average total served utility by the FN with $N = 5$ when applying Algorithm 1 for different combinations of η and γ in MDP.

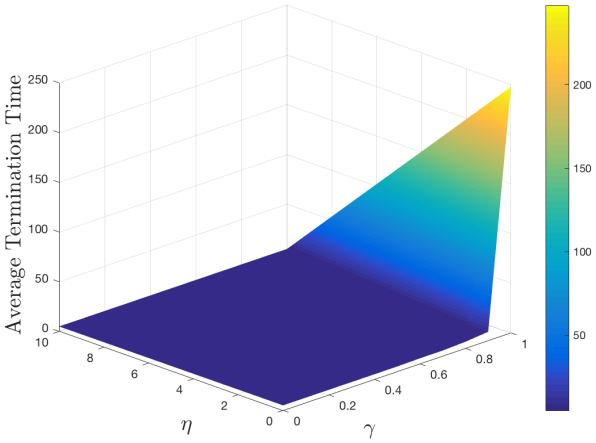

Fig. 5. The average termination time, in time-steps, starting from the initial state for FN with $N = 5$ when applying Algorithm 1 for different combinations of η and γ in MDP.

the state values converge after around 50 episodes.

As shown in Figs. 4 and 5, we next test the effect of changing η and γ on the FN performance when applying Algorithm 1 in the IoT environment φ_7. We consider $\gamma \in \{0, 0.1, 0.2, ..., 1\}$ and $\eta \in \{0, 1, 2, ..., 10\}$. Starting from the initial state S_0, the average total served utility for various combinations of η and γ is shown in Fig. 4. Putting less weight for future rewards, represented by smaller γ, the FN is encouraged to serve regardless of the waiting penalty. It maintains an average total served utility of about 24.85, which is about five times the expected utility of φ_7 given in Table II due to the available 5 slots of resources. The corresponding average termination time T is 5, as shown in Fig. 5, which means the FN serves all the time irrespective of the received utility. Similar results of average total served utility and expected termination time are experienced for large waiting penalty η regardless of γ since serving is encouraged also in this case. The average total served utility increases by more than 35% to about 33.8 for $\gamma = 1$ and $\eta = 1$, with a corresponding average termination time of about 10.27. A maximum average total served utility of 50 is achieved when $\eta = 0$ and $\gamma = 1$ as there is no penalty for waiting with maximum weight for future rewards. However, in this case, the FN waits too long to serve the maximum utility ($u = 10$) users, hence average T exceeds 200 as shown in Fig. 5.

Recall that the FN's objective is to maximize the expected total served utility and minimize the expected termination time. Hence, to compare the performance of Algorithm 1 with the fixed threshold algorithm, which does not learn from the interactions with the environment, we define an objective performance metric R as

$$R = \mathbb{E}\left[\sum_{m=1}^{M} u_m - \theta(T - M)\right], \tag{8}$$

where a served utility is denoted with u_m, the cost of waiting is denoted with θ, and the number of served requests in an episode is denoted with M. In the proposed resource allocation algorithm with $\eta = \theta$ and $\gamma = 1$, R corresponds to the average return starting from the initial state S_0, i.e., $\mathbb{E}[G|S = S_0]$. For $\theta = 1$, we compare the performance of the proposed algorithm, in terms of R, with the fixed-threshold algorithm, which uses the same threshold regardless of the environment, in the 19 IoT environments. For the proposed algorithm, we set the parameters as $\eta = 1$ and $\gamma = 1$, and for the fixed threshold algorithm we consider all possible thresholds $1, 2, ..., 10$.

As shown in Figs. 6 and 7, the proposed RL-based algorithm exhibits the best performance as it adaptively learns how to balance early termination with higher utilities. It never terminates too early or too late ($T \approx 9$ for all environments as seen in Fig. 7), as opposed to the fixed-threshold algorithm which is not adaptive to the environment. As seen in Fig. 6, the performance of the fixed-threshold algorithm with thresholds $1, 2, 3, 8, 9$ are steadily below that of the RL algorithm. Threshold 4 has a comparable performance

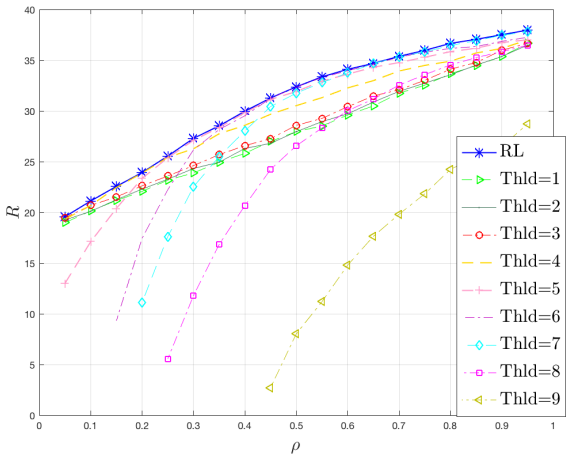

Fig. 6. The performance of FN with $N = 5$ in various IoT environments when applying Algorithm 1 with $\eta = 1$, $\gamma = 1$, and the fixed-threshold algorithm with different thresholds.

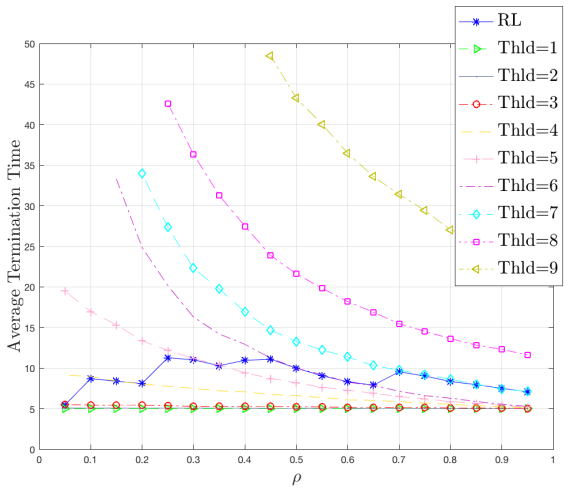

Fig. 7. The average termination time, in time-steps, for FN with $N = 5$ in various IoT environments when applying Algorithm 1 with $\eta = 1$, $\gamma = 1$, and the fixed-threshold algorithm with different thresholds.

to RL for the environments with $\rho \leq 25\%$, after which its performance starts to decline. Although thresholds $5, 6, 7$ have good performances close to RL for environments with medium to high ρ, they perform far from RL for IoT environments with small ρ. The R values for threshold 10 are negative for all environments due to the long termination time which exceeds 90, thus it does not appear in Figs. 6 and 7.

VI. Conclusions

We formulated the resource allocation problem for F-RAN in a heterogeneous IoT environment as an infinite-horizon Markov Decision Process (MDP) problem. Then, we provided the optimum solution (decision policy) for the MDP problem through a Reinforcement Learning (RL) algorithm. Various IoT environments were considered in the simulations to test the performance of the proposed RL-based resource allocation algorithm. The numerical results corroborated the fact that the RL method adapts to the environment by learning the optimum policy from experience. We showed that the proposed RL-based method always outperforms the fixed-threshold method, which does not learn from the environment, irrespective of the IoT environment. Moreover, the fixed-threshold algorithm cannot automatically select the best threshold for the environment.

References

[1] S.-H. Park, O. Simeone, and S. Shamai, "Joint optimization of cloud and edge processing for fog radio access networks," in *Information Theory (ISIT), 2016 IEEE International Symposium on*. IEEE, 2016, pp. 315–319.

[2] A. T. Nassar, A. I. Sulyman, and A. Alsanie, "Achievable rf coverage and system capacity using millimeter wave cellular technologies in 5g networks," in *Electrical and Computer Engineering (CCECE), 2014 IEEE 27th Canadian Conference on*. IEEE, 2014, pp. 1–6.

[3] A. I. Sulyman, A. T. Nassar, M. K. Samimi, G. R. MacCartney, T. S. Rappaport, and A. Alsanie, "Radio propagation path loss models for 5g cellular networks in the 28 ghz and 38 ghz millimeter-wave bands," *IEEE Communications Magazine*, vol. 52, no. 9, pp. 78–86, 2014.

[4] W. Wang, V. K. Lau, and M. Peng, "Delay-aware uplink fronthaul allocation in cloud radio access networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 7, pp. 4275–4287, 2017.

[5] Y.-Y. Shih, W.-H. Chung, A.-C. Pang, T.-C. Chiu, and H.-Y. Wei, "Enabling low-latency applications in fog-radio access networks," *IEEE network*, vol. 31, no. 1, pp. 52–58, 2017.

[6] P. Schulz, M. Matthe, H. Klessig, M. Simsek, G. Fettweis, J. Ansari, S. A. Ashraf, B. Almeroth, J. Voigt, I. Riedel *et al.*, "Latency critical iot applications in 5g: Perspective on the design of radio interface and network architecture," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 70–78, 2017.

[7] Y. Sahni, J. Cao, S. Zhang, and L. Yang, "Edge mesh: A new paradigm to enable distributed intelligence in internet of things," *IEEE access*, vol. 5, pp. 16 441–16 458, 2017.

[8] A.-C. Pang, W.-H. Chung, T.-C. Chiu, and J. Zhang, "Latency-driven cooperative task computing in multi-user fog-radio access networks," in *Distributed Computing Systems (ICDCS), 2017 IEEE 37th International Conference on*. IEEE, 2017, pp. 615–624.

[9] G. S. Rahman, M. Peng, K. Zhang, and S. Chen, "Radio resource allocation for achieving ultra-low latency in fog radio access networks," *IEEE Access*, vol. 6, pp. 17 442–17 454, 2018.

[10] J. Oueis, E. C. Strinati, and S. Barbarossa, "The fog balancing: Load distribution for small cell cloud computing," in *Vehicular Technology Conference (VTC Spring), 2015 IEEE 81st*. IEEE, 2015, pp. 1–6.

[11] E. Balevi and R. D. Gitlin, "Optimizing the number of fog nodes for cloud-fog-thing networks," *IEEE Access*, vol. 6, pp. 11 173–11 183, 2018.

[12] Y.-J. Liu, S.-M. Cheng, and Y.-L. Hsueh, "enb selection for machine type communications using reinforcement learning based markov decision process," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 12, pp. 11 330–11 338, 2017.

[13] M. Condoluci, T. Mahmoodi, E. Steinbach, and M. Dohler, "Soft resource reservation for low-delayed teleoperation over mobile networks," *IEEE Access*, vol. 5, pp. 10 445–10 455, 2017.

[14] Y. Wei, F. R. Yu, M. Song, and Z. Han, "User scheduling and resource allocation in hetnets with hybrid energy supply: An actor-critic reinforcement learning approach," *IEEE Transactions on Wireless Communications*, vol. 17, no. 1, pp. 680–692, 2018.

[15] T. Park, N. Abuzainab, and W. Saad, "Learning how to communicate in the internet of things: Finite resources and heterogeneity," *IEEE Access*, vol. 4, pp. 7063–7073, 2016.

[16] M. Yan, G. Feng, and S. Qin, "Multi-rat access based on multi-agent reinforcement learning," in *GLOBECOM 2017-2017 IEEE Global Communications Conference*. IEEE, 2017, pp. 1–6.

[17] R. Sutton, and A. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.