User and resource allocation in latency constrained Xhaul via reinforcement learning

MOHSAN NIAZ CHUGHTAI[1], SHABNAM NOOR[1], IGNAS LAURINAVICIUS[1], PHILIPPOS ASSIMAKOPOULOS[1], NATHAN J. GOMES[1], HUILING ZHU[1], JIANGZHOU WANG[1], XI ZHENG[2], AND QI YAN[2]

[1] *Division of Computing, Engineering and Mathematical Sciences, University of Kent, Canterbury, Kent, CT2 7NZ*
[2] *Huawei Technologies, Shenzhen, Guangdong, China*
[*] *Corresponding author: mc978@kent.ac.uk*

Compiled February 20, 2023

FlexE is envisioned for the provisioning of different services and hard slicing of the Xhaul in 5G and beyond networks. For efficient bandwidth utilization in the Xhaul, traffic prediction for slot allocation in FlexE calendars is required. Further, if coordinated multipoint (CoMP) is used, the allocation of users to remote units (RUs) with an Xhaul path of lower latency to distributed/central unit (DU/CU) will increase the achievable user bit rate. In this paper, the use of multi-agent deep reinforcement (DRL) learning for optimal slot allocations in a FlexE-enabled Xhaul, for traffic generated through CoMP, and for offloading users among different RUs is explored. In simulation results, the DRL agent can learn to predict input traffic patterns and allocate slots with the necessary granularity of 5 Gbps in the FlexE calendar. The resulting gains are expressed in terms of the reduction of mean over-allocation of slots in the FlexE calendar in comparison to the prediction obtained from an autoregressive moving average (ARIMA) model. Simulations indicate that DRL outperforms ARIMA-based prediction by up to 11.6% © 2023 Optica Publishing Group

http://dx.doi.org/10.1364/ao.XX.XXXXXX

1. INTRODUCTION

The network architecture of future 5G networks incorporating Xhaul and coordinated multipoint beamforming (CoMP) is shown in Figure 1. The network segment between the remote unit (RU) and distributed unit (DU) is termed fronthaul while the segment between DU and central unit (CU) is termed mid-haul [1] [2]. The processing functions performed by each of these units depend on the functional split being deployed, with different tradeoffs between the requirements of latency, data rate and coordination gain [3],[4],[5]. Generally, and especially in cases where there is some converged deployment, this part of the transport network is referred to as the Xhaul [1]. Further, the Xhaul will also be a vital part of the future distributed and cell-free MIMO networks [6]. Under this paradigm, an RU is envisioned as an access point (AP) and the DU/CU is envisioned as the central processing unit (CPU). The variation in the bit rate in the Xhaul will depend on factors such as the type of beamforming being used, user mobility, user allocation to RUs and thus to different paths in the Xhaul, and resource allocation in both the radio access network (RAN) and Xhaul. Delays in the Xhaul will influence the achievable beamforming gains and achievable user bit rate [7],[8].

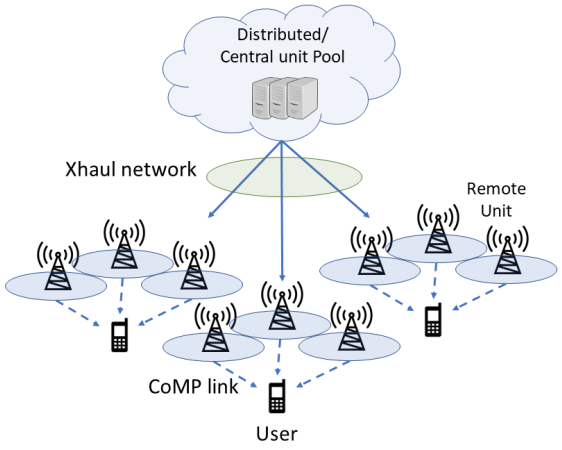

Fig. 1. 5G network architecture with Xhaul and CoMP.

If the transport segment of the Xhaul networks comprises

Flexible Ethernet (FlexE) [9], then the available bandwidth in the physical layer (PHY) links can be allocated in steps of 5Gbps to media access control (MAC) clients having different data rates. The PHY links would normally be standard Ethernet link PHY rates e.g., 10Gbps to 400 Gbps. The bandwidth allocation with 5Gbps granularity results in improved bandwidth efficiency as well as isolation of multiple input streams/sources in hardware [10], [11].

With FlexE [9] incorporated into the Xhaul, the achievable user bit rate will depend on the over and under-allocation of slots in the FlexE calendar, as this will affect serialization and buffering delays and, in turn, the Xhaul's total latency. The Xhaul latency affects the delay in feeding back-channel state information via uplink to the DU/CU where beamforming vectors are computed, especially if coordinated multi-point (CoMP) is employed [8], [12]. Therefore, to maximize the achievable user bit rate, a resource allocation scheme that can anticipate the traffic in advance and appropriately allocate resources is required. This will enable efficient utilization of the available bandwidth and minimization of packet loss in the Xhaul. Further, some coordination of control between RAN and Xhaul is desirable to maximize achievable user bit rates through the selection of low-latency paths over the Xhaul. For this, a mechanism to offload users from RUs with higher Xhaul latency to those with lower latency will contribute to the maximization of the achievable user bit rate.

Recently there have been several studies that have addressed the latency in the Xhaul (in fronthaul, midhaul or both) as a constraint for optimising the operational expenditure (OPEX) [13], [14] as well as finding optimal routes through the Xhaul [15]. In [13] and [14], the model for latency only considered the queuing delays in RUs. The research mainly focused on using integer linear programming (ILP) and mixed integer linear programming (MILP) techniques for optimisation. Such optimisation techniques require full knowledge of the values of the environment variables as well as the constraints. In contrast to these limitations, in deep reinforcement learning (DLR), the agents observe the environment's variable values and then take actions to achieve a reward defined by the user. Therefore, agents do not require any prior knowledge of environment variable values. Exploiting the benefit of this autonomy, DRL has been demonstrated for optimal selection of functional splits and routes in fronthaul [16] and including fronthaul with software-defined networking control [17], under the constraints of latency. DRL has also been used for resource allocation in an ultra-reliable low latency (uRLLC) network [18] and in Fog RAN networks [19], [20] under the constraints of latency.

The aforementioned studies omitted any mechanism for handling the allocation of users to RUs under the constraints of latency of their paths in the Xhaul, as a means of maximization of their achievable bit rate. Also, the models for latency in these studies were based either on propagation delay, serialization delay, mean queuing delay or a priori end-to-end delay, and thus did not consider the combined effect of all possible sources of delays in their models. They further omitted any consideration of percentile delays due to buffering of data at the destination nodes of Xhaul paths. None of the aforementioned studies considered a resource allocation scheme for Xhaul, that is enabled by FlexE for efficient bandwidth utilization or its influence on the latency constraint.

So, in this paper, a combined RAN/Xhaul network is modelled for which achievable user bit rate is maximized while employing efficient bandwidth allocation in the Xhaul, comprising

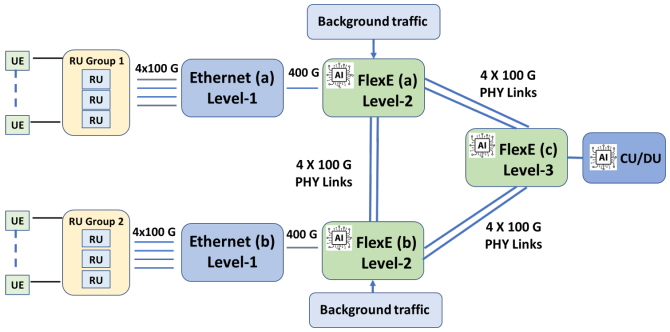

Fig. 2. Network architecture for DRL-based optimisation of RAN.

Ethernet and FlexE aggregation nodes. The efficient bandwidth allocation is performed using Deep-Q learning on each FlexE aggregation node to enable predictions of input traffic and to allocate slots accordingly. These allocations attempt to minimize over-allocation as well as under-allocation which would increase Xhaul latency and reduce the achievable user bit rate. On the RAN side, achievable user bit rate maximization is performed by using Deep-Q learning for offloading users from RUs with higher Xhaul latency to those with lower latency, based on a target for achievable user bit rate maximization. End-to-end latency constraints on Xhaul paths are modelled in terms of 99.9 percentile delays. The use of multiple DRL agents (in the Xhaul and RAN) results in a multi-agent-based optimisation system for RAN [21].

In Section 2 of this paper, the network model under consideration is described. In Section 3, the wireless channel model incorporating CoMP, and the model for latency in Xhaul are presented. In Section 4, the simulation parameters for the wireless model and the model for latency based on percentile delay, the hyperparameters of the deep-Q network and the reward functions for the two DRL agents are described. In Section 5, simulation results for the predictions of input traffic patterns are discussed. These are compared with predictions from a more classical estimation technique based on autoregressive moving average (ARIMA) [22]. In Section 6, the results for the maximization of the achievable user bit rate by offloading UEs between RUs are discussed. Finally, Section 7 presents conclusions and discussion for future work.

2. NETWORK MODEL

For analysis of the achievable user bit rate affected by the latency of the Xhaul, the envisioned network architecture based on Figure 1 is shown in Figure 2. In the network, we assume that CU and DU are co-located, so the portion of Xhaul under analysis in the paper is only between RU and DU. In this network, users are connected to RUs via CoMP beamforming [7],[8]. Each RU is located at a different geographical location on the (x,y) coordinate plane. Additionally, due to user mobility, users may enter or leave the range of RUs, possibly resulting in a better rate if associated with the closer RU. The RUs in one group are connected to the same Ethernet aggregation node. The traffic is then transported over the Xhaul. The traffic in the Xhaul is higher than user traffic due to RAN protocol overheads added by the RUs (these will be dependent on functional split) [22] and background traffic from other sources in the network. The traffic from RUs is further aggregated by Ethernet aggregation nodes at level-1 (labelled a, b in Figure 2) in the Xhaul. The link rate

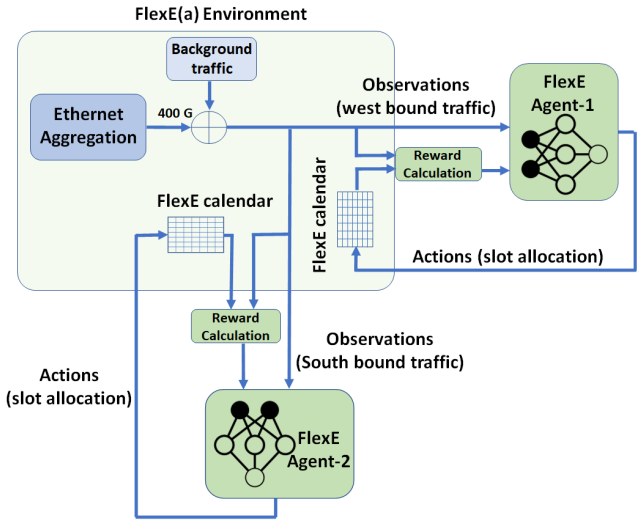

Fig. 3. Agents and environment in FlexE(a) node in Figure 2.

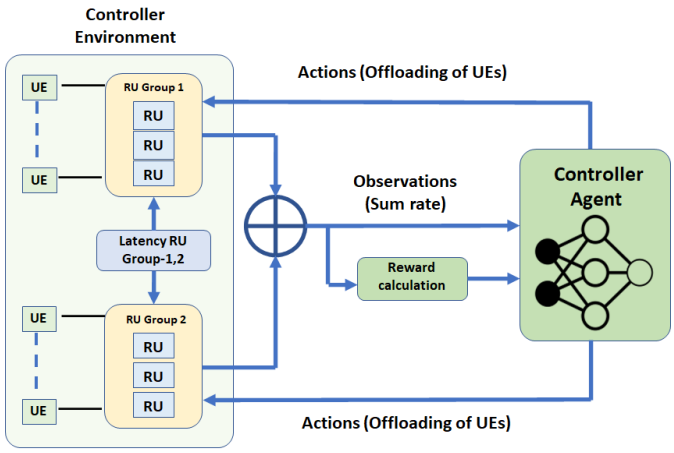

Fig. 4. Agent and environment for DU/CU in Figure 2.

between the RUs and these Ethernet nodes is assumed to be 4 x 100 Gbps. Further, background traffic from other sources in the network is aggregated with the RU traffic by FlexE aggregation nodes at level-2 (also labelled a, b in Figure 2). These FlexE aggregation nodes need to allocate slots in their calendars sufficient for the incoming traffic, in an efficient manner. Then, the traffic from the level-2 FlexE aggregation node is aggregated by a level-3 FlexE aggregation node (labelled c in Figure 2), which connects to the DU/CU. The FlexE aggregation nodes are linked by 4 x 100 Gbps PHY links.

Each RU will be equipped with Ethernet interfaces. So, the RUs and Ethernet aggregation nodes in Xhaul will introduce queuing and serialization delays to the user traffic being aggregated. It is assumed that the Ethernet aggregation nodes operate with a FIFO scheduling scheme in the queues. The Ethernet and FlexE nodes are equipped with buffers to hold packets when there is port contention at the output ports. These buffers will result in delay variations. In the FlexE aggregation nodes, the buffers would hold the packets if the rate allocated in the FlexE calendar is less than the input traffic [23]. For the uplink, the destination DU/CU will also be equipped with buffers to absorb the delay variations. The size of these buffers will be determined by the percentiles of delays that are absorbed by the buffers. Large buffer sizes will ensure a higher percentile delay variation absorption but increased buffering delay [24].

Within each FlexE aggregation node of Figure 2, there is a separate calendar for traffic flows in different directions, as shown in Figure 3. A FlexE aggregation node, therefore, has a separate DRL agent for the slot allocation for each calendar. The FlexE agents in Figure 3 receive the traffic from the Ethernet aggregation nodes and the background traffic from other sources in the network as input observations. Based on the observations and reward calculations, the FlexE agents perform actions in the form of slots with a granularity of 5Gbps. The actions and observations are updated every 4ms which is the delay for the reconfiguration of the FlexE calendar. The slot allocations for the incoming traffic are calculated using Deep-Q learning from past actions and rewards in an experience buffer [25]. Deep-Q learning was used since it observes input traffic both from RAN and other sources in the network. This traffic has large variations making it impossible for conventional Q-table-based reinforcement learning agents to relate observations to actions. In this way, the FlexE agent pre-configures the calendar in the aggregation node, to help reduce any buffering delay of packets due to the under-allocation of 5Gbps slots, while maintaining efficiency.

Note that if it is assumed that the FlexE aggregation nodes are equipped with bandwidth variable transponders (BVTs) [26], which can adjust the bandwidth of the PHY links dynamically then the current DRL solution would also be able to operate with that (with very minor changes to reflect any different granularity).

Within the DU/CU there is a separate agent, termed the "controller agent", as shown in Figure 4. Among the different functions of a controller agent in the DU/CU, such as routing traffic in the Xhaul network, the focus in this paper is the control over offloading of users among RU-groups to improve the total achievable user bit rate. In Figure 4, the controller agent receives the total achievable user bit rate and its variations, of the RU-groups as an observation. If the path between the RU and the DU/CU causes large delays, this has the effect of reducing the achievable bit rate of the RU-group due to the reduced accuracy of channel state information (CSI). Thus, the action of the controller agent should be to offload users from the RU-group with higher Xhaul latency to the group with lower latency, in its attempt to maximize the achievable user bit rate. Like the FlexE agent, the controller agent was also trained using Deep-Q learning due to the large observation space of the RU bit rate and its variations. This would further help the agent, to generalize the transition from state to action. The actions are the number of UEs to be offloaded.

3. COMP AND LATENCY MODEL.

For modelling the achievable user bit rate allocated to RUs incorporating CoMP, the signal-to-interference noise (SINR) γ_k of a user is modelled as follows,

$$\gamma_k = \frac{\sum_j \left| h_{j,k}^H f_{j,k} \right|^2}{\sum_j \sum_{i \neq k} \left| h_{j,k}^H f_{j,i} \right|^2 + \sigma^2} \quad \textbf{(1)}$$

where $h_{j,k}^H$ is the known channel information for user k from RU, j delivered to the distributed/central unit (DU/CU) for

processing, and $f_{j,k}$ is the resulting beamforming vector given as follows,

$$f_{j,k} = \left(h_{j,k}^{\mathrm{H}} h_{j,k}\right)^{-1} h_{j,k}. \tag{2}$$

In Equation 1, each user is equipped with a single antenna and Zero-Forcing (ZF) beamforming is used to reduce interference between beams generated by the RUs [19]. As in [27], channel estimation is modelled with delay effects, that is, a perfect channel is not known but is instead estimated and denoted as $\hat{h}(t)$ with error $\eta(t)$, which satisfies a complex normal distribution $\mathcal{CN}\left(0, \sigma_\eta^2\right)$, with zero mean and variance σ_η^2 given as follows,

$$\sigma_\eta^2 = \frac{1}{(SNR)\, d} + 2\left[\pi f_D T_s \left(\frac{2L - d - 1}{2}\right)^2\right]. \tag{3}$$

Estimation error has the same dimension as $h(t)$; if SNR during channel estimation is decreased, channel accuracy is impacted negatively, which can be mitigated by an increased pilot training length d; this, in turn, affects the total length of payload L. Other significant terms are the Doppler frequency f_D which depends on the parameters of user mobility, and sampling time T_s of the receiver [28]. The complete channel estimate $h(t)$is given as follows,

$$h(t) = \hat{h}(t) + \eta(t). \tag{4}$$

Due to fronthaul and computational delay D, the channel estimate in Equation 3 is altered, given as follows,

$$\hat{h}(t) = \rho \hat{h}(t - D) + \epsilon(t) \tag{5}$$

where $\rho = J_0(2\pi f_D D)$ is the channel correlation vector affected by user mobility, and $\epsilon(t)$ is the $\mathcal{CN}\left(0, \sigma_\epsilon^2\right)$ error vector, with variance σ_ϵ^2 given as follows,

$$\sigma_\epsilon^2 = \sigma_{\hat{h}}^2 \left(1 - |\rho|^2\right). \tag{6}$$

Then the channel model in Equation 5 is substituted in Equation 1 for the estimation of SINR. The estimated SINR is then used to estimate the resulting achievable bit rate of a user R_{ZF} in the uplink via the Shannon capacity model [29], is given as follows,

$$R_{ZF} = \sum_{k=1}^{K} log_2\left(1 + \frac{\left|\left(\sqrt{\rho}\hat{h}_k^T + \sqrt{\sigma_{\epsilon,k}^2 + \sigma_{\eta,k}^2}\right) f_k\right|^2}{\sum_{i \neq k}\left|\left(\sqrt{\rho}\hat{h}_k^T + \sqrt{\sigma_{\epsilon,k}^2 + \sigma_{\eta,k}^2}\right) f_i\right|^2 + \sigma_n^2}\right). \tag{7}$$

It is also assumed that the users join and leave the RU-groups with arrival modelled by a Poisson distribution. The traffic generated by users is first received at RUs that add encapsulation overheads dependent on the split option being used. Then further overheads are added by the Ethernet and FlexE-based aggregation nodes in Xhaul. It was assumed that an intra-PHY functional split option 7.2 [3] in Xhaul was assumed that resulted in upscaling or an increase in the user data rate. The functional split upscaling factor F_{split} is calculated using the relation given as follows [30],

$$F_{split} = \frac{2 * N_{ant} * N_{bit} * \frac{N_{sc}}{T_s}}{N_{ant} * log_2(M) * BW} \tag{8}$$

where N_{ant} is the number of antenna elements, N_{bit} is the sampling width (no of bits per sample), N_{sc} is the number of data subcarriers comprising the OFDM symbol, M is the modulation order, BW is the bandwidth of the signal transmitted by a UE and T_s is the OFDM symbol length in time. Note that the calculations in Equation 8 ignore the encapsulation overheads.

The achievable bit rate of the users is reduced due to the latency in Xhaul discussed in Section 2 and indicated by the factor of D in Equation 5. For the modelling of latency, percentile delay was selected as a benchmark. This was done to model the worst-case scenario of latency instead of mean queuing delay which is orders of magnitude (1 and 2) less than percentile delays [31]. Also, as mentioned in Section 2, the destination nodes in Xhaul will be equipped with buffers to absorb the percentile delays. The latency, defined in terms of percentile delay of 99.9%, is then substituted as the delay term 'D' in Equation 5 to the wireless channel model. Percentile delays smaller than 99.9% were not estimated due to the limited statistics that could be gathered in network simulations.

For the estimation of percentile delays of a path in the Xhaul, first, the mean queuing delay of packets $T_{q,x}$, in RUs Ethernet interface and Ethernet aggregation nodes are estimated using the G/G/1 queueing model given as follows [32],

$$T_{q,x} \approx \frac{1}{\mu_r} \frac{\rho_{est}}{(1 - \rho_{est})} \frac{{C_a}^2 + {C_r}^2}{2} \tag{9}$$

where x denotes the node in the Xhaul, ρ_{est} is the estimated load on aggregation nodes, λ_t is the arrival rate of packets, μ_r is the packet service rate, C_r is the coefficient of variation of the service time, and C_a is the coefficient of variation of the inter-arrival time. The mean of the delay $\bar{T}$ of a path in the Xhaul in Figure 2 is given as follows,

$$\bar{T} = T_{s,RU} + T_{s,E} + 2\,T_{s,F} + T_{s,F} + T_{q,RU} + T_{q,E} \tag{10}$$

where $T_{s,RU}$, $T_{s,E}$ and $T_{s,F}$ are the fixed serialization delays in the RU's Ethernet interface, Ethernet aggregation node and two FlexE aggregation nodes in a path in Xhaul. $T_{q,RU}$ and $T_{q,E}$ are the waiting times in the RU's Ethernet interface and Ethernet aggregation node of the path in Xhaul. The distribution of the total delay of a path is then modelled by a log-normal distribution using the total mean delay in Equation 10. This distribution was selected after analysis of packet traces via simulations in NetSim®. From the log-normal distribution, the 99.9 percentile delay is extracted. The fabrication delays are ignored in the analysis.

4. SIMULATION PARAMTERS

Simulink MATLAB® was used to model wireless communications based on CoMP as described in Section 3. The parameters for the wireless system are listed in Table 1. The CoMP model was initialized by assuming 15 users in each RU-group (a total of 30 users in the two groups). Each RU was modelled to transport 16 independent layers/beams of user data in Xhaul. The locations of users were uniformly distributed over the 200x200m grid, shown in Figure 5, to maintain a balance of achievable bit rate between the RU-groups. The locations of RUs were in a hexagonal formation with a radius of 50m so that the RUs are equidistant from each other and on average equidistant from UE locations.

Equal background traffic of 110 Gbps was added into the paths from both RU-groups to DU/CU, resulting in a balanced

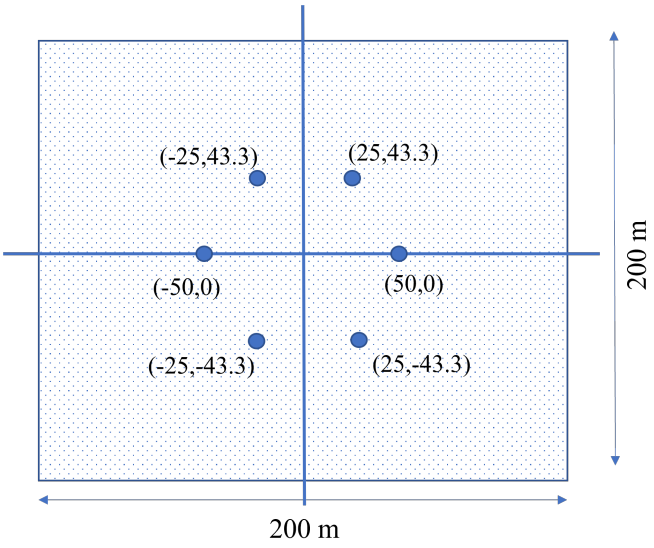

Fig. 5. RU coordinates in x,y plane (Case-1).

Table 1. Simulation Parameters.

Number of users per RU-group	15
Avg bit rate of RU-groups	117.04 Gbps
Functional split factor	10.5
Avg background traffic	110.7 Gbps
Number of RUs Groups	2
Number of RUs per Group	3
SNR per User	13 dB
Antennas per RU	16
Power Constraint per RU	40 W
Bandwidth per user	50 MHz
User Mobilities	6 m/s
Rate of Arrival	10^{-6}
Rate of Departure	$4x10^{-6}$

Table 2. Functional split upscaling parameters.

Number of antenna elements N_{ant}	15
Number of bits per symbol N_{bit}	8
Number of subcarriers N_{sc} in 50 MHz	250x12
Modulation level (M)	16
Interval of the OFDM symbol T_s	66.7 μs

Table 3. Deep-Q Network Parameters.

Learning Rate	0.01
Gradient Threshold	1
Number of Hidden Layers	6
Number of Neurons per Layer	98
Sample Time	4×10^{-3}
Experience Buffer length	40 x 6000
Mini batch size	2048
Epsilon	1
Epsilon Decay factor	0.005
Number of steps to look ahead	
FlexE agent	8
Controller agent	32
Discount factor	
FlexE agent	0.7
Controller agent	0.99

impact of latency in the two paths. The total achievable bit rate for both RU-groups was found to be 22.3 Gbps. This data rate was upscaled by a factor of 3.5 using Equation 8 and parameters from Table 2. Further upscaling by a factor of 3 was carried out to accommodate the fact that there are three RUs in an RU-group. This upscaling resulted in a load of 234.089 Gbps in the Xhaul.

The underlying background traffic pattern was extracted from the WIDE MAWI project packet traces repository [33] specifically for the date of 08/04/2020. WIDE MAWI project is an online repository of packet traces that have been collected by monitoring/measuring the traffic from a backbone optical network operated in Japan. The traces contain information about packets such as packet types, packet header and data lengths, packet source and destination IP addresses, and packet arrival and departure times at the node. The traces were collected at a node within the network with a 10 Gbps link. The traces were also used in [34], for the prediction of traffic patterns as well as slot allocations. The average bit rate of the traffic was 6 Gbps which is presumed less than the expected aggregate background and user traffic in future 5G Xhaul networks. Therefore, the traffic patterns were increased by multiplication with discrete upscaling factors (5∼11) to emulate a high load on the Xhaul aggregation nodes.

The parameters for the deep-Q network, given in Table 3, were fine-tuned, adjusting the discount factor, number of steps lookahead, mini-batch size, experience buffer length, number of hidden layers and neurons per layer. The discount factor for the FlexE agent was 0.7, the number of steps lookahead was 8 and the experience buffer length spanned over the total number of steps of 40 episodes. These parameters ensured that the earned average reward increased monotonically with each training episode and stabilized until the end of the training process. The large experience buffer ensured that the agent did not suffer from catastrophic forgetting which may lead it to take steps that earn negative rewards after earning positive rewards [35]. On the other hand, the discount factor for the controller

agent was 0.99 and the number of steps to look ahead was 32. Also, the minibatch size parameter for both the FlexE agent and controller agent was set to 2048 as a compromise between simulation speed and accuracy of convergence of the deep-Q network [36].

The parameters for modelling the latency using the G/G/1 queuing model are listed in Table 4. It was assumed that the average packet length was 735 bytes, considering the minimum and maximum size of Ethernet frames. It was also assumed that the length of Ethernet frames had a Gaussian distribution with a standard deviation of 300 bytes. The inter-arrival times of packets at source RUs had a Gaussian distribution with a mean set to give the required load and standard deviation of 300μs. Normal distribution was selected since its standard deviation is independent of mean interarrival times, which were varied to vary the load.

Table 4. Queuing Delay Parameters.

Average packet length	735 bytes
Standard deviation of packet length	300 byes
Output links rate	400 Gbps
Standard deviation of inter-arrival time of packets	300μs
Distance of optical links	1 km

A. Reward Function and training of FlexE Agent

For the FlexE agent, it was assumed that the action space for the FlexE agent is discrete and limited to 80 slots with a granularity of 5 Gbps since it was assumed that each FlexE node will support a total of 400 Gbps output line rate. In any DRL-based system, designing the reward functions is the most significant challenge. Different reward functions will force the RL system either to converge to the desired performance or force it to converge to a local minimum, which may not result in the desired performance [37]. Both discrete and continuous rewards were compared in terms of performance and convergence [38]. Of the different reward functions that were tested for the FlexE agent, the reward function achieving the least standard deviation of the difference between input traffic and allocated traffic is given as follows,

$$R_{FlexE} = \begin{cases} -20 + ((S - R_i) - 20) & \forall\ 0 \leq S - R_i \leq\ 20 \\ -1000\left(1 - e^{-1*(S-R_i)}\right) & \forall\ S - R_i > 0 \\ -1000\left(1 - e^{(S-R_i)}\right) & \forall\ S - R_i < 0 \end{cases} \quad \textbf{(11)}$$

where S is the bit rate allocated in the FlexE calendar and R_i is the observed input traffic bit rate. The reward function for the FlexE agent is also plotted in Figure 6.

The FlexE agent was trained using deep-Q learning in the reinforcement learning toolbox in MATLAB®. It was trained to predict the combined traffic patterns from the WIDE project repository [33] (with an upscaling factor of (5∼11) and the user traffic patterns from each RU-group. The FlexE agent was trained for 40 episodes. Each episode spanned 24 seconds with a step size of 4ms, (reconfiguration time frame of the FlexE calendar) [39]. After training, the agent having the highest reward was deployed, which in our case was after 34 episodes.

B. Reward Function and training of FlexE Agent

The 'controller agent' was trained simultaneously with the FlexE agents, also using deep-Q learning in the reinforcement learning

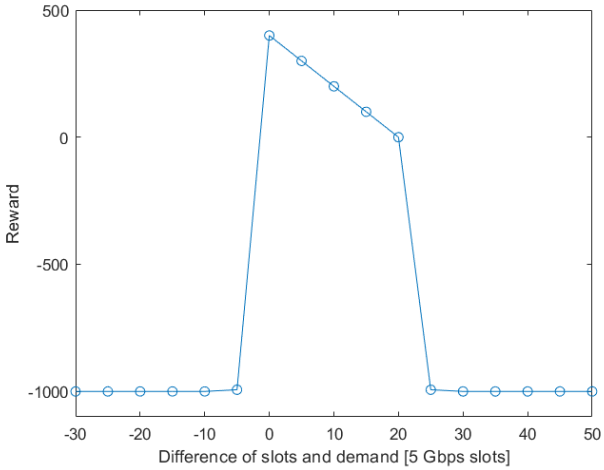

Fig. 6. Reward vs difference of allocated slots and input traffic demand.

toolbox in MATLAB®. The controller agent aims to maximize the total achievable bit rate of the two RU-groups by offloading users from a path with higher latency to a path with lower latency in the network. The actions of the controller agent are the offloading commands for each RU group, of integer values ranging from 0 to 5. In any training step, the number of users offloaded was limited to 5 users. This resulted in a monotonically increasing average reward earned by the controller agent during the training. The reward function for the controller agent is given as follows,

$$R_{ctr} = 500\left(\bar{R}_{sum}\right)^2 + \left(-5x10^5\right)\left(\bar{R}'_{sum}\right) \quad \textbf{(12)}$$

where $\bar{R}_{sum}$ is the averaged value over 5 samples of the total achievable bit rate for the two RU-groups and $\bar{R}'_{sum}$ is the difference in the bit rate in consecutive timesteps of 4ms, indicating the variations in the total achievable bit rate of the RU-groups. It is given as follows,

$$\bar{R}'_{sum} = \bar{R}_{sum}\left(t\right) - \bar{R}_{sum}\left(t - 0.004\right). \quad \textbf{(13)}$$

Due to the second term in Equation 12 involving $\bar{R}'_{sum}$ the controller agent is negatively penalized. This also triggers the agent to carry out offloading of users to compensate for the resulting decline in the average value of the total achievable bit rate $\bar{R}_{sum}$ and reward R_{ctr}.

The training process of the controller agent for the RU position in Figure 5 is depicted in Figure 7(b). It shows the first eight seconds of a training episode which spans over twenty-four seconds in the simulations. The resulting total achievable bit rate of the two RU-groups is shown in Figure 7(a). From Figure 7 (b), it can be observed that the controller agent constantly swaps the users from one RU-group to another, as it learns to maximize the total achievable bit rate of the two RU-groups. It can also be observed in Figure 7(a), that the total achievable user bit rate decreases when the difference in the number of users in the two RU-groups increases. Also, there are variations in the total number of users in the two RU groups, which is an artefact of the random arrival and departure in the number of users in the two RU-groups.

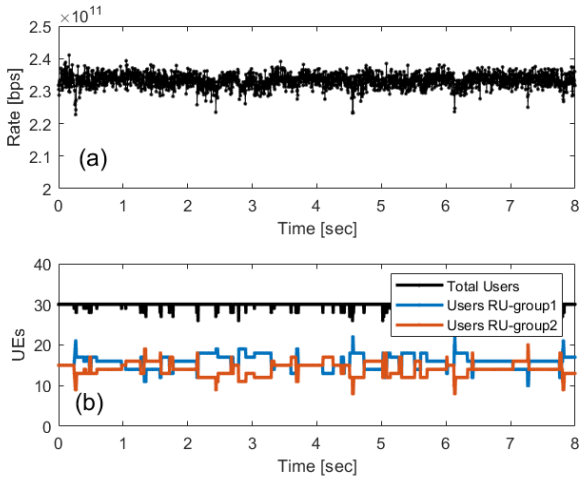

Fig. 7. Training results of controller agent, total achievable bit rate (a), number of users in RU-groups (b) for RU-group positions in Figure 5.

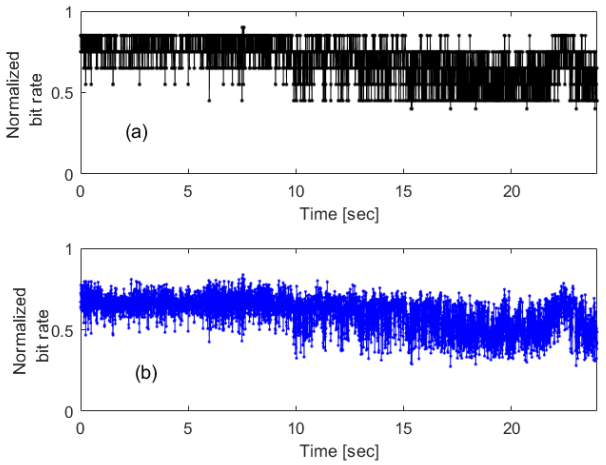

Fig. 8. FlexE Agent slot allocation (a), input traffic (b).

5. RESULTS FOR FLEXE AGENT

The results for the deployment of the trained FlexE agent are shown in Figure 8. The input traffic pattern is upscaled by a factor of 10 and normalized to the output rate of the FlexE aggregation node. Figure 8(b) shows the input traffic and Figure 8(a) shows the number of slots allocated to the input traffic. From the analysis of Figure 8(a), it can be observed that the agent attempts to track the input traffic and allocated slots.

The performance of the FlexE agent was also analyzed, for various upscaling factors of the input traffic patterns with predictions from the autoregressive integrated moving average (ARIMA(8,1,1)) model approach [34] and fixed allocations based on the highest traffic demand. The coefficients for the (ARIMA (8,1,1)), were calculated using the same training patterns for which the FlexE agent was trained. Since ARIMA estimates the average values based on past observations, this will result in under-allocation in the FlexE calendar. To overcome this shortcoming, the predictions from ARIMA $(\hat{R}_{ARMA})$ were normalized by a multiplication factor $(\alpha\hat{R}_{ARMA}: \alpha > 1)$, using a similar approach in [34]. This normalization also ensured that the predictions from ARIMA resulted in the same cumulative probability of under allocation (CUA) $(Pr\,(S - R_D) < 0)$ as that from the FlexE agent's predictions, for the input traffic that was upscaled by a factor of 10.

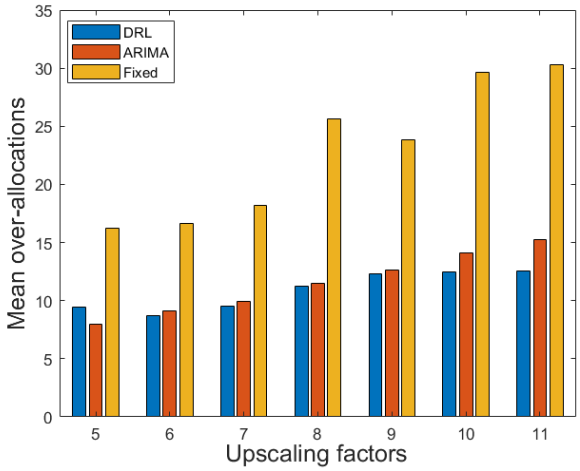

Fig. 9. MOAs for different upscaling factors and allocation schemes (for the same CUA at upscaling factor of 10).

Then, the results for the three approaches were compared in terms of mean over-allocation (MOA) of traffic as a measure of bandwidth over-consumption in the Xhaul. The mean-over allocations were calculated by averaging the positive difference between the allocated rate by the FlexE agent and input traffic $(\overline{(S - R_D) > 0})$ over 24 seconds. The mean over-allocations for upscaling factors of 5 to 11 of the input traffic are shown in Figure 9. It can be observed in Figure 9 that at an upscaling factor of 10, the mean over-allocations from DRL-based predictions are 11.6 % less than the mean over-allocation from ARIMA-based predictions and 58% less than from fixed allocations. For the other upscaling factors of the input traffic patterns, the mean over-allocation using DRL is also generally less than the mean over-allocations from ARIMA and fixed allocations (except for the case of an upscaling factor of 5, where ARIMA gave better performance).

Afterwards, to compare the performance of the three approaches in terms of CUA, the predictions from ARIMA $(\hat{R}_{ARMA})$ were normalized by a positive multiplication factor $(\alpha\hat{R}_{ARMA}: \alpha > 1)$ so that they result in the same MOA as that from the FlexE agent's predictions. The resulting CUAs for the upscaling factors of 5 to 11 of the input traffic are shown in Figure 10. The CUAs for fixed allocation are not shown in Figure 10 since they are very low in comparison to CUAs from DRL and ARIMA. It can also be observed in Figure 10 that at an upscaling factor of 10, the CUAs from DRL are 25% less than the CUAs from ARIMA-based predictions. Therefore, DRL can outperform ARIMA-based predictions both in terms of efficient bandwidth consumption and minimal delays in Xhaul.

6. RESULTS FOR CONTROLLER AGENT

The deployment results for the controller agent that was trained for 36 episodes are shown in Figure 11 with the RU positions shown in Figure 5. The average background traffic added to the paths of the two RU groups was the same and around 110 Gbps.

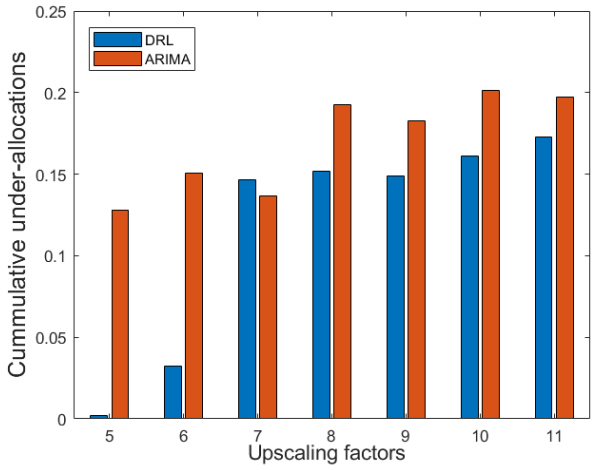

Fig. 10. CUAs for different upscaling factors and allocation schemes (for the same mean over-allocation at upscaling factor of 10).

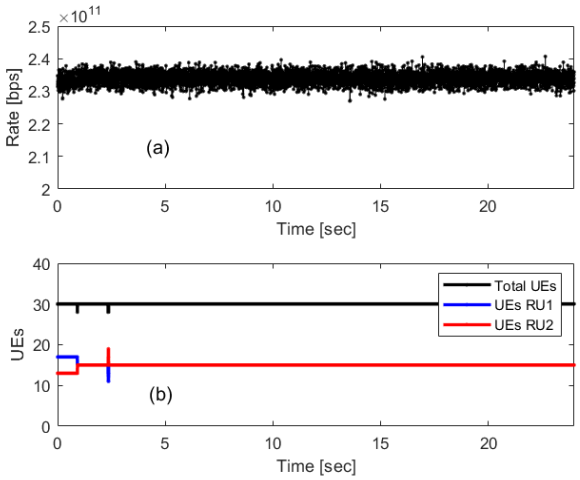

Fig. 11. Controller agent deployment results, total achievable bit rate (a), number of users in RU-groups (b) for RU positions in Figure 5).

Table 5. RU-group bit rates for different positions and background traffics.

RU positions		Background traffic (Gbps)		Number of users		Bit rate (Gbps)		Total bit rate (Gbps)
		RU-group		RU-group		RU-group		
		1	2	1	2	1	2	
1	a	110	110	15	15	117.04	117.04	234.1
	b	285	110	15	15	86.16	115.8	201.8
2	a	110	110	15	15	113.3	115.8	229.1
	b	114	111	14	16	107.8	121.2	229
	c	250	110.7	9	21	77.25	145	222.3
	d	285.5	105.67	7	23	62.9	153.5	216.5
	e	331.5	111.5	3	27	31.38	166.67	198.5
	f	113	265.26	19	11	125	100.2	225.2

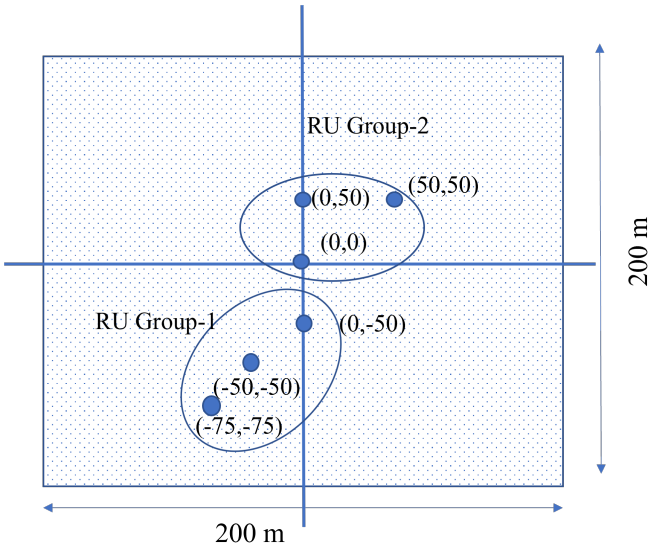

Fig. 12. RU coordinates in x,y plane (Case-2).

The results in Figure 11(a) show the total achievable bit rate of the two RU-groups is around 234.09 Gbps after applying the functional split factor of 10.5. The results in Figure 11(b) show the total number of users in two RU-groups, along with users in the individual RU-groups. It is apparent from Figure 11(b) that the initial number of users in the two RU-groups was 13 and 17 as a non-ideal starting distribution of users. The agent then offloads users so that both RU-groups have the same number of users. This is so because the total achievable bit rate is maximum when the two RU-groups have the same number of users.

The results in Figure 11 are also summarized in case-1(a) of Table 5. In case-1(b), higher background traffic of 285 Gbps was added in the path of RU-group-1. Then the trained controller agent in case-1(a) was deployed. It can be observed in case-1(b), that agent does not offload users from RU-group-1 to RU-group-2. This resulted in a reduction of the total bit rate of the RU-groups from 234.1 Gbps to 201.8 Gbps. This shows that the addition of higher background traffic results in higher latency which then causes the reduction of the total bit rate.

Then, the positions of RUs were changed such that they have non-ideal positions and more non-uniform distances to the users distributed over the grid of 200x200m. The coordinates of the RUs are shown in Figure 12. This configuration resulted in a higher average distance to users from RU-group-1 than from RU-group-2. Note that the controller agent was retrained for the new configuration of RU positions.

The background traffic added to the paths of the RU-groups was also varied with their values listed in case-2(a∼f) of Table 5. These values represent the average background traffic over the deployment period of 24 seconds. The variation of background traffic in RU-group-1's path was higher than the RU-group-2 path from case-2(a) to case-2(f). The background traffic in the RU-group-2 path varied around 110 Gbps. The bit rate and the number of users in each RU-group from the deployment of the trained controller agent are also listed for case-2(a∼f). It can be observed that for case-2(a) with the same background traffic of 110 Gbps in the two paths, the users in RU-group-2 had a higher

bit rate than RU-group-1 since the users on average are closer to RU-group-2. Despite this, the agent did not offload users from RU-group-2 to RU-group-1. It only carried out the offloading from RU-group-1 to RU-group-2 when more background traffic was added in the path of RU-group-1, and increasingly so as shown from case-2(b) to case-2(e). It can also be observed that as the background traffic is increased, due to the increase in latency, the bit rate of RU-group-1 as well as the total bit rate of the two RU-groups also decreases. This indicates that the impact of latency in the terms of reduction of the total bit rate of RU-groups is more pronounced than the impact of unequal distance of users from RU-groups.

Also, by observing case-1(b) and case-2(d) which have the same background traffics in both paths of the RU-groups, the total bit rate in case-2(d) (216.5 Gbps) was higher than in the case-1(b) (201.8 Gbps) by up to 7.3 %. This shows that the trained controller through offloading users among RU groups can improve the achievable bit rate of the system. In case-2(f), more background traffic was added to RU-group-2's path. This resulted in offloading of users from RU-group-2 to RU-group-1 by the trained controller agent. This proves that the system has symmetry, and the trained controller agent can also learn to offload users in both directions

7. CONCLUSION

A combined mechanism for user and resource allocation based on DLR has been demonstrated as a promising way of achieving user bit rate maximization in future 5G or beyond networks. For this, a RAN using CoMP was implemented. The transport segment was assumed to comprise Ethernet and FlexE nodes. The achievable user bit rate was degraded by the latency in the Xhaul, as it added delay in the CSI sent to the DU/CU. The end-to-end latency in the Xhaul was modelled by 99.9 percentile delay instead of mean delay, to address the effect of buffering in the destination node in Xhaul, which in this case was DU/CU.

DRL was used for the prediction of traffic and slot allocation in FlexE calendars in the Xhaul. From the results it can be concluded that DRL can more efficiently allocate resources in FlexE-based Xhaul, outperforming more classical approaches based on ARIMA-based allocation schemes both in terms of under-allocation and over-allocation of available resources. Further, the FlexE agent could learn a wide variety of traffic patterns with different properties of mean and variance, to the least, and was able to predict underlying patterns in incoming traffic. While on the other hand, ARIMA-based methods would require recalculation of model parameters every time statistical properties of the input traffic patterns change.

It was also demonstrated that DRL can effectively circumvent the effects of latency in Xhaul via user offloading. The controller agent was able to constantly offload users from the RU group with the path having higher latency in Xhaul to the path having low latency. This offloading boosted the achievable user bit rate. The controller agent also learnt to symmetrically offload users in two directions depending on the load on the paths in Xhaul. Overall, both the controller agent and the FlexE agent were able to concurrently improve the system performance albeit without having a priori knowledge of the environment variable values. So it can be concluded that DRL can enable the network to autonomously improve its resource utilization as well as achievable user bit rate.

For future work, it is proposed that the computation time and memory resources for training the DRL agents can be modelled. Further to improve the utilization of memory the size of the experience buffers can be optimised by selecting a tradeoff between memory resources and the probability of catastrophic forgetting of the agents during the training process. Lastly, the research presented in this paper focused on a multi-agent system, in which the agents optimise the systems independently, so the performance of the system can be further improved via sharing of the learned experiences and earned rewards among agents.

FUNDING

This work at the University of Kent has been partially funded by the Networking and User Experience Lab, Huawei Technologies Co., Ltd., China.

REFERENCES

1. S. Sirotkin, *5G Radio Access Network Architecture: The Dark Side of 5G* (John Wiley & Sons, 2020). Google-Books-ID: _90IEAAAQBAJ.
2. RAD technologies, "Everything you need to know about 5g xHaul," .
3. 3GGP, "3GPP TR 38.801 V14.0.0 (2017-03): Study on new radio access technology: Radio access architecture and interfaces." (2016).
4. N. J. Gomes, P. Sehier, H. Thomas, P. Chanclou, B. Li, D. Munch, P. Assimakopoulos, S. Dixit, and V. Jungnickel, "Boosting 5G Through Ethernet: How Evolved Fronthaul Can Take Next-Generation Mobile to the Next Level," IEEE Veh. Technol. Mag. **13**, 74–84 (2018).
5. L. M. P. Larsen, A. Checko, and H. L. Christiansen, "A survey of the functional splits proposed for 5g mobile crosshaul networks," IEEE Commun. Surv. Tutorials **21**, 146–172 (2019).
6. S. Chen, J. Zhang, J. Zhang, E. Björnson, and B. Ai, "A survey on user-centric cell-free massive MIMO systems," Digit. Commun. Networks **8**, 695–719 (2022).
7. B. Akgun, M. Krunz, and D. Manzi, "Impact of beamforming on delay spread in wideband millimeter-wave systems," in *2020 International Conference on Computing, Networking and Communications (ICNC),* (2020), pp. 890–896.
8. B. E. Godana and D. Gesbert, "Coordinated beamforming in multicell networks with channel state information exchange delays," in *2013 IEEE 24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC),* (2013), pp. 713–718.
9. A. Eira, A. Pereira, J. Pires, and J. Pedro, "On the efficiency of flexible ethernet client architectures in optical transport networks [invited]," J. Opt. Commun. Netw. **10**, A133–A143 (2018).
10. T. Hofmeister, V. Vusirikala, and B. Koley, "How can flexibility on the line side best be exploited on the client side?" in *2016 Optical Fiber Communications Conference and Exhibition (OFC),* (2016), pp. 1–3.
11. N. Huin, P. Medagliani, S. Martin, J. Leguay, L. Shi, S. Cai, J. Xu, and H. Shi, "Hard-isolation for network slicing," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS),* (2019), pp. 955–956.
12. F. Zhou, G. Luo, Y. Liu, Y. Wang, and L. Fan, "Coordinated beamforming for heterogeneous small-cell networks with a non-ideal backhaul," IET Commun. **12**, 595–602 (2018).
13. A. Tzanakaki, M. P. Anastasopoulos, and D. Simeonidou, "Converged optical, wireless, and data center network infrastructures for 5g services," J. Opt. Commun. Netw. **11**, A111–A122 (2019).
14. N. Molner, A. de la Oliva, I. Stavrakakis, and A. Azcorra, "Optimization of an integrated fronthaul/backhaul network under path and delay constraints," Ad Hoc Networks **83**, 41–54 (2019).
15. M. Klinkowski, "Optimization of latency-aware flow allocation in ngfi networks," Comput. Commun. **161**, 344–359 (2020).
16. Z. Gao, J. Zhang, S. Yan, Y. Xiao, D. Simeonidou, and Y. Ji, "Deep reinforcement learning for bbu placement and routing in c-ran," in *2019 Optical Fiber Communications Conference and Exhibition (OFC),* (2019), pp. 1–3.

17. E. H. Bouzidi, A. Outtagarts, and R. Langar, "Deep reinforcement learning application for network latency management in software defined networks," in *2019 IEEE Global Communications Conference (GLOBECOM),* (2019), pp. 1–6.
18. M. Elsayed and M. Erol-Kantarci, "Reinforcement learning-based joint power and resource allocation for urllc in 5g," in *2019 IEEE Global Communications Conference (GLOBECOM),* (2019), pp. 1–6.
19. A. Nassar and Y. Yilmaz, "Reinforcement learning for adaptive resource allocation in fog ran for iot with heterogeneous latency requirements," IEEE Access **7**, 128014–128025 (2019).
20. S. Sritharan, H. Weligampola, and H. Gacanin, "A study on deep learning for latency constraint applications in beyond 5g wireless systems," IEEE Access **8**, 218037–218061 (2020).
21. A. Feriani and E. Hossain, "Single and multi-agent deep reinforcement learning for ai-enabled wireless networks: A tutorial," IEEE Commun. Surv. Tutorials **23**, 1226–1252 (2021).
22. P. J. Brockwell and R. A. Davis, eds., *ARMA Models* (Springer New York, New York, NY, 2002), pp. 83–110.
23. S. Zhang, Q. Zhong, M. Zha, and T. Zuo, "Hybrid multiplexing over flexe group," in *2018 23rd Opto-Electronics and Communications Conference (OECC),* (2018), pp. 1–2.
24. P. Iovanna, G. Bottari, F. Ponzini, and L. M. Contreras, "Latency-driven transport for 5g," J. Opt. Commun. Netw. **10**, 695–702 (2018).
25. V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, and J. Pineau (2018).
26. M. Jinno, "Elastic optical networking: Roles and benefits in beyond 100-gb/s era," J. Light. Technol. **35**, 1116–1124 (2017).
27. N.-S. Kim and Y. H. Lee, "Effect of channel estimation errors and feedback delay on the performance of closed-loop transmit diversity system," in *2003 4th IEEE Workshop on Signal Processing Advances in Wireless Communications - SPAWC 2003 (IEEE Cat. No.03EX689),* (2003), pp. 542–545.
28. Q. Sun, D. Cox, H. Huang, and A. Lozano, "Estimation of continuous flat fading mimo channels," in *2002 IEEE Wireless Communications and Networking Conference Record. WCNC 2002 (Cat. No.02TH8609),* vol. 1 (2002), pp. 189–193 vol.1.
29. D. Tse and P. Viswanath, *Fundamentals of Wireless Communication* (Cambridge University Press, 2005).
30. V. Q. Rodriguez, F. Guillemin, A. Ferrieux, and L. Thomas, "Cloud-ran functional split for an efficient fronthaul network," in *2020 International Wireless Communications and Mobile Computing (IWCMC),* (2020), pp. 245–250.
31. G. O. Pérez, J. A. Hernández, and D. Larrabeiti, "Fronthaul network modeling and dimensioning meeting ultra-low latency requirements for 5g," J. Opt. Commun. Netw. **10**, 573–581 (2018).
32. J. F. C. Kingman, "The single server queue in heavy traffic," Math. Proc. Camb. Philos. Soc. **57**, 902–904 (1961).
33. Kenjiro Cho, Koushirou Mitsuya, and Akira Kato, "Traffic Data Repository at the WIDE Project," USENIX 2000 FREENIX Track,San Diego, CA (2000).
34. Y. Liao, S. A. Hashemi, H. ElBakoury, J. Cioffi, and A. Goldsmith, "Calendar allocation based on client traffic in the flexible ethernet standard," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC),* (2020), pp. 1–6.
35. W. Fedus, P. Ramachandran, R. Agarwal, Y. Bengio, H. Larochelle, M. Rowland, and W. Dabney, "Revisiting fundamentals of experience replay," in *Proceedings of the 37th International Conference on Machine Learning,* vol. 119 of *Proceedings of Machine Learning Research* H. D. III and A. Singh, eds. (PMLR, 2020), pp. 3061–3071.
36. X. Qian and D. Klabjan, "The Impact of the Mini-batch Size on the Variance of Gradients in Stochastic Gradient Descent," (2020). ArXiv:2004.13146 [cs, math].
37. R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, Adaptive Computation and Machine Learning series (A Bradford Book, Cambridge, MA, USA, 2018), 2nd ed.
38. MATHWORKS, "Define Reward Signals - MATLAB & Simulink," .
39. OIF, "Flex Ethernet 2.2 Implementation Agreement," (2021).

AUTHOR BIOGRAPHIES

Mohsan Niaz Chughtai (mc978@kent.ac.uk) received his M.Eng. from the Chalmers University of Technology, Sweden in 2009 and is currently pursuing his PhD in electronic engineering at the School of Engineering, University of Kent, Canterbury, U.K.

Shabnam Noor(S'16, M'19) (s.noor2@aston.ac.uk) received her PhD in 2019 from the School of Engineering, University of Kent, Canterbury, U.K., before working there as a research associate. She is currently a research associate at Aston Institute of Photonic Technologies, Aston University, Birmingham, U.K.

Ignas Laurinavicius(il79@kent.ac.uk) received his M.Eng. in 2018 and is currently pursuing his PhD in electronic engineering at the School of Engineering, University of Kent, Canterbury, U.K.

Philippos Assimakopoulos (S'09, M'16) (p.asimakopoulos@kent.ac .uk)received his PhD in 2012 and is currently a Lecturer in electronic systems with the School of Engineering, University of Kent, Canterbury, U.K.

Nathan J. Gomes(M'92, SM'06) (n.j.gomes@ucl.ac.uk, n.j.gomes@kent.ac.uk) is a Professor at the Department of Electronic and Electrical Engineering, University College, London, U.K. and an Honorary Professor at the School of Engineering, University of Kent, Canterbury, U.K. He was also the TPC Chair of ICC 2015.

Huiling Zhu (SM'17) (h.zhu@kent.ac.uk) received her PhD from Tsinghua University, Beijing, China and is currently a Reader (Associate Professor) at the School of Engineering, University of Kent, Canterbury, U.K. She also serves as an Editor for IEEE Transactions on Vehicular Technology.

Xi Zheng (zhengxi3@huawei.com) received her PhD from Tsinghua University, Beijing, China in 2019 and is currently a Research Engineer in Networking and User Experience Lab, at Huawei Technologies Co., Ltd., China.

Qi Yan (yanqi1@huawei.com) received his PhD from Xidian University, China and is currently a Senior Research Engineer in Networking and User Experience Lab, at Huawei Technologies Co., Ltd., China.

Jiangzhou Wang (M'91, SM'94, F'17) (j.z.wang@kent.ac.uk) is a Professor at the School of Engineering, University of Kent, Canterbury, U.K. He is also a Fellow of the Royal Academy of Engineering, U.K., and a Fellow of the IET. He was the Technical Program Chair of the ICC 2019 and the Executive Chair of ICC 2015.