Energy Optimization in Ultra-Dense Radio Access Networks via Traffic-Aware Cell Switching

Metin Ozturk, Attai Ibrahim Abubakar, João Pedro Battistella Nadas, Rao Naveed Bin Rais, Sajjad Hussain, *Senior Member, IEEE*, and Muhammad Ali Imran, *Senior Member, IEEE*

***Abstract*—We propose a reinforcement learning-based cell switching algorithm to minimize the energy consumption in ultra-dense deployments without compromising the quality of service (QoS) experienced by the users. In this regard, the proposed method can intelligently learn which small cells (SCs) to turn off at any given time based on the traffic load of the SCs and the macro cell. To validate the idea, we used the open call detail record (CDR) data set from the city of Milan, Italy, and tested our algorithm against typical operational benchmark solutions. With the obtained results, we demonstrate exactly when and how the proposed method can provide energy savings, and moreover how this happens without reducing QoS of users. Most importantly, we show that our solution has a very similar performance to the exhaustive search, with the advantage of being scalable and less complex.**

***Index Terms*—5G, reinforcement learning, cell switching, energy consumption, cellular networks.**

I. Introduction

SMALL cells (SCs) are low power nodes which are deployed to boost the capacity at hotspot zones, the busiest areas of a deployment. BSs are the major energy consumers, accounting for about 60%–80% of the total energy consumption in cellular networks [1]. Since the traffic load of cellular networks exhibit temporal and spatial variations, the traditional technique of keeping the BS always ON even when it is not serving any user results in energy wastage. Therefore, load adaptive network operation, where BSs are turned off or operated in low power modes during periods of low or no traffic in order to save energy, has been the focus of many studies [1]–[7]. Nevertheless, it is not always feasible to completely switch off SCs in the conventional heterogeneous network (HetNet) architecture because it often creates coverage holes, which in turn degrade the Quality of service (QoS) of users initially covered by the inactive SCs. In addition, sleeping BSs do not transmit pilot signals needed by the user equipment (UE) for cell discovery, channel estimation, and subsequent connection. Hence, in the conventional HetNet architecture certain components of the SCs need to be left always ON even in sleep mode, resulting in sub-optimal energy savings [8], [9].

Handling the aforementioned challenges of BS switching in conventional HetNets requires a paradigm shift towards a Control/Data Separated Architecture (CDSA) [8]. In CDSA, the macro cells (MCs)—also known as control BS (CBS)—maintain constant coverage and provide signalling functionalities and low data rate services. The SCs—also known as data BS (DBS)—provide high data rate services and are connected to the MCs through the backhaul. This architecture provides support and flexibility for dynamic cell switching operations as the MC always ensures constant coverage for both idle and active users. It is also responsible for switching BSs off/on as well as associating users to the SCs. As a result, it is possible to completely switch of the DBSs under the CDSA when there are no users associated to them [8].

Cell switching with traffic offloading has been identified as one of the techniques for reducing the energy consumption of MCNs. Several methods have already been proposed for scheduling cell switching in the literature using analytical modelling and heuristic algorithms [2]–[5], [10]. However, it is very difficult to develop accurate analytical models for network optimization when network dimensions become very large due to network complexity and high computational overhead [11]. On the other hand, heuristic algorithms are difficult to generalize and adapt to dynamic environment, such as ultra-dense network deployment scenarios that have been envisioned in 5G [11], [12]. They also mostly employ exhaustive search techniques, which makes them computationally demanding and could lead to degradation in the QoS of users. Recent works in [13]–[20] applied conventional reinforcement learning (RL) techniques for cell switching because of its ability to adapt to a dynamic network environment through learning.

However, conventional RL algorithms are very challenging to implement when the network dimensions become huge because it often results in very large state-action space, which is computationally demanding to learn. In addition, a considerable amount of memory is required to store the action-value table (Q-table). In an attempt to solve the curse of dimensionality problem facing conventional RL algorithms, they

Manuscript received July 7, 2020; revised November 6, 2020 and January 16, 2021; accepted January 23, 2021. Date of publication February 2, 2021; date of current version May 20, 2021. This work was supported in part by the Engineering and Physical Sciences Research Council (GCRF) funds under Grant EP/P028764/1 and in part by the Ajman University. The work of Metin Ozturk was supported by the Republic of Turkey Ministry of National Education. The editor coordinating the review of this article was H. Tabassum. *(Corresponding author: Attai Ibrahim Abubakar.)*

Metin Ozturk is with the Department of Electrical and Electronics Engineering, Ankara Yildirim Beyazit University, 06010 Ankara, Turkey.

Attai Ibrahim Abubakar, João Pedro Battistella Nadas, Sajjad Hussain, and Muhammad Ali Imran are with the James Watt School of Engineering, University of Glasgow, Glasgow G12 8QQ, U.K. (e-mail: abubakar.1@research.gla.ac.uk).

Rao Naveed Bin Rais is with the Department of Electrical and Computer Engineering, Ajman University, Ajman, UAE.

Digital Object Identifier 10.1109/TGCN.2021.3056235

were combined with value function approximation (VFA) to estimate the optimal policy. In this regard, RL with linear function approximators and deep RL approaches were proposed in [21]–[24]. Deep RL algorithms have the ability to accommodate large state-action space resulting from large scale networks deployment scenarios. However, training such models can be computationally demanding and energy consuming. Hence, they should only be considered when no simpler solutions exist or the complexity of the network requires the application of a non-linear function approximator to estimate the optimal policy.

We propose a RL-based cell switching and traffic offloading methodology for ultra-dense radio access network (RAN), and employ a RL algorithm with linear function approximation known as SARSA with VFA [25]. The well-known RL with VFA algorithm is carefully designed according to the requirements of the problem at hand in such a way that all the states need not be visited as in [21] before the optimal policy is learnt. In addition, the algorithm exhibits quick convergence, and it is simpler to implement compared to deep RL approaches. The learning algorithm is implemented at each MC and it has the ability to learn the optimal switching pattern even when a large number of SCs are deployed under the coverage area of the MCs.

A. Related Work

The authors in [2] developed a load-based dynamic SC switching scheme for ultra-dense networks. Their goal is to minimize the signalling overhead resulting from user traffic offloading during the cell switching process as well as to optimize the energy savings. Two heuristic algorithms were proposed in their work. In [3], the problem of user association and cell sleeping in multi-tier ultra-dense SC networks was formulated as a complex integer programming. Then, two low complexity heuristic algorithms were employed to determine the optimal user association and the cell switching pattern. A greedy heuristic algorithm was proposed in [4] to determine the switching off/on pattern of SCs in a green ultra-dense network. The optimization objective is to maximize the network energy efficiency while considering traffic load of the SCs and service requirements of users. Heuristic algorithms often employ exhaustive search, which is often slow and computationally demanding, to find the optimal solution, hence they are only suitable for small network deployment. On the other hand, the ultra-dense 5G network scenario will involve massive deployment of SCs which would make it practically impossible to adopt such heuristic algorithms. Also, it would result in huge computational overhead and degradation in QoS.

An alternative solution for finding optimal switching pattern for ultra-dense deployment scenarios is to employ RL techniques. The authors in [15], [16], [21] proposed Q-learning-based cell switching techniques for energy optimization. The authors in [15] proposed a Q-learning algorithm to determine the duration of time that the BS can spend at a particular sleep level in order to optimize energy consumption of the network. BS activation latency and the service requirements of users were considered as constraints in their problem formulation. A Q-learning framework was developed in [16] in order to determine the optimal switching and traffic offloading strategy in a two-tier heterogeneous network with separation architecture. Nonetheless, the works in [13]–[17] only considered small to medium network deployment scenarios, where the state-action space is suitable for the implementation of conventional RL algorithms.

The authors in [21] proposed a centralized and decentralised Q-learning algorithm with compact state representation (QC-learning) for traffic offloading and cell switching for HetNet to minimize energy consumption. The QC-learning is a compact representation of the state-action pair using linear VFA when it becomes practically impossible to explicitly store each state-action pair in a look-up table. Moreover, even with the compact state representation of the Q-value lookup table, as the number of SCs in the network becomes very large, the action set also grows dramatically, thereby making it difficult to implement a centralized cell switching scheme. As a result, they went further to develop a decentralized multi-agent-QC-learning scheme. In the decentralized scheme, the MCs learn in a cooperative manner and take joint traffic offloading and cell switching actions by exploiting the previous cell switching strategies used by other MCs. Moreover, in developing decentralized multi-agent-QC-algorithm, the authors [21] assumed that all the MCs under the controller have similar or stationary network states, which is a requirement for implementing joint cell switching and traffic offloading strategy. However, this might not be the case in real networks, as networks' states may vary from one MC to another due to temporal and spatial variations in user traffic demands [26]. In addition, the problem of increased state-action pair also arises in multi-agent Q-learning, since each agent also includes its own state-action pair to the joint state action space [26]. This increases the computational complexity as well as the memory required for storing the joint state-action space at the controller.

B. Contribution

In this article, we propose an intelligent cell switching and traffic offloading framework using a RL technique known as SARSA with VFA [25] in order to reduce the energy consumption of ultra-dense RAN. In the CDSA RAN, a cell switching and traffic offloading mechanism is implemented in a locally centralized manner at each MC. This is because the MC is responsible for scheduling the switching off/on pattern of all the SCs deployed under its coverage. The proposed algorithm provides a compact form of representing the action-value function like the QC-learning algorithm in [21]. However, all the states need not be visited as in [21] before the optimal strategy is obtained. The contributions of this work are as follows:

- We propose a scalable traffic-aware cell switching method based on RL with VFA to find the optimal policy in terms of energy minimization for controlling SC ON/OFF status, without compromising the QoS.
- We prove mathematically that turning a SC off is not always profitable in terms of energy savings, and identify such situations where it is not profitable to switch off SCs.

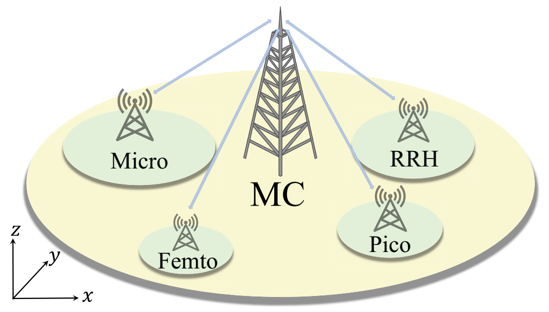


Fig. 1. A HetNet with CDSA comprising of a MC (CBS) and a dense deployment of various types of SCs (DBSs).

- We evaluate the proposed method using a traffic model based on real world data, making the solution more reliable and realistic.

Due to its fast convergence and learning ability, RL is employed in this work to make real-time, accurate, and efficient switch off/on decision at each time slot. Moreover, VFA is utilized in the developed RL algorithm, since the state space expands exponentially with increasing number of SCs in the network. Even though this could be handled by conventional RL algorithms to some extent, it becomes infeasible to manage once the network size becomes very large. In addition, due to the careful and proper design of the action set in the proposed method, there is no need to include all the possible switching combinations, thus annihilating the need for cooperative learning. Lastly, we tested the proposed method in a realistic scenario, where all types of SCs given in [27] with their diverse characteristics are included.

II. System Model

A. Network Model

As mentioned before, in this work we consider an ultra-dense RAN with a CDSA architecture [8]. The network model, as illustrated in Fig. 1, consists of a dense network, where SCs, acting as DBSs, are deployed under the coverage area of a MC, which acts as the CBS. Moreover, SCs and MC operate on dedicated frequency channels and SCs are connected to the MC via optical fibre links.

The MC is responsible for constant coverage, control signalling, and data services, while SCs handle only data services and user specific requests. Furthermore, the MC coordinates the traffic offloading and switching off/on of all SCs under its coverage. This task involves observing their traffic loads and deciding which set of SCs should be turned off during periods of low traffic intensity, while taking the available capacity of the MC into consideration.

It should noted that since an ultra-dense RAN with CDSA normally comprise of many MC-SCs configurations, only one of such configurations is depicted in the system model in Fig. 1.

B. Network Power Consumption

Following the Energy Aware Radio and neTwork tecHnologies (EARTH) power consumption model [27], P_j, the instantaneous consumption of a *j*th BS, B_j, is given by [28]

$$P_j = \begin{cases} P_{\mathrm{o},j} + \eta_j \lambda_j P_{\mathrm{T},j} & 0 < \lambda_j < 1, \\ P_{\mathrm{s},j} & \lambda_j = 0 \end{cases} \tag{1}$$

where $P_{\mathrm{o},j}$ and $P_{\mathrm{s},j}$ are the operational and sleep circuit power consumption, respectively, η_j is the power amplifier (PA) efficiency, λ_j is the load factor, and $P_{\mathrm{T},j}$ is the transmit power. All SCs of the same type are considered to have identical hardware, such that their PA efficiencies, and circuit power consumptions are the same. Also, power allocation is not considered, hence, each type of BS has a fixed transmit power that is constant among BSs of that type.

Lastly, $\boldsymbol{P}$, the instantaneous power consumption for the CDSA network, is expressed as

$$P = \sum_{j=1}^{s+1} P_j, \tag{2}$$

where s is the number of SCs in the deployment.

III. Problem Formulation

Considering the architecture described above, the aim of this article is to find the best policy, in terms of energy savings, which offers a required QoS to the users. A policy $\mu = \{\delta_1, \delta_2, \ldots, \delta_{s+1}\}$ is defined by which SCs should be ON at a given time *t*, where $\delta_j \in \{0, 1\}$ indicates the state of B_j, 1 for ON and 0 for OFF. B_1 is the MC, and thus δ_1 is always 1, as it is always ON.

Considering $j > 1$, when δ_j changes from 1 to 0 at time *t*, the MC allocates its users:

$$\lambda_{1,t} = \lambda_{1,t-1} + \phi_j \lambda_{j,t-1} \tag{3}$$

$$\lambda_{j,t} = 0, \tag{4}$$

where $\lambda_{j,t}$ corresponds to the load of B_j at time *t*, and ϕ_j is the relative capacity of B_j with respect to B_1, such that

$$\phi_j = \frac{C_j}{C_1}, \quad j > 1, \tag{5}$$

where C_j indicates the total resources available in B_j. Conversely, when δ_j switches from 0 to 1 at time *t*, the MC offloads some of its traffic to B_j, such that

$$\lambda_{j,t} = \frac{\tau_j}{C_j} \tag{6}$$

$$\lambda_{1,t} = \lambda_{1,t-1} - \phi_j \lambda_{j,t}, \tag{7}$$

where τ_j corresponds to the resources used by users served by B_j. Note that $C_j \geq \tau_j$.

Therefore, we can formally write the problem as

$$\min_{\mu} \; P(\mu) = \sum_{j=1}^{s+1} \big(P_{\mathrm{o},j} + \eta_j \lambda_j P_{\mathrm{T},j}\big)\delta_j + P_{\mathrm{s},j}(1 - \delta_j)$$
$$\text{s.t.} \;\; \lambda_1 \leq 1. \tag{8}$$

Note that the only constraint in the problem is to ensure that the capacity of the MC is not exceeded, this takes care of the QoS requirement. In other words, the MC only offloads users from a SC if it can maintain the QoS of any user associated to it. Moreover, this ensures that once a solution is implemented,

$$T_{\mathrm{r},1} \leq T_{\mathrm{m},1}, \tag{9}$$

where $T_{\mathrm{r},1}$ and $T_{\mathrm{m},1}$ are the required and maximum provided throughputs by the MC, respectively.

It should be noted that the problem formulation in (8) is not as simple as it appears as it takes into account the load status of both the MC and the SCs under its coverage. In addition, it considers the power consumption of the 4 different types of SCs (Table II) that are used in this work (even though this cannot be explicitly seen in (8)) when making a switching decision. As a result, it is more difficult to determine the optimal switching strategy in our work due to the more complex scenario considered.

Theorem 1: If we draw a random SC B_j from $\mathbb{B}$, the set of all possible BSs, the probability of $\delta_j = 1$ integrating the optimal policy is 1 if $j>1$ and

$$\frac{P_{\mathrm{o},j} - P_{\mathrm{s},j}}{\phi_j \eta_1 P_{\mathrm{T},1} - \eta_j P_{\mathrm{T},j}} < \lambda_j, \tag{10}$$

when $\phi_j \eta_1 P_{\mathrm{T},1} - \eta_j P_{\mathrm{T},j} > 0$.

Proof: See Appendix B for the proof. ■

Based on Theorem 1, we can see that the optimal policy will tend to have more SCs turned on when they are more loaded, as the inequality in (10) will be more easily met. The reasoning behind this is that the numerator of (10) represents the saving on the static power consumption when a SC is put into a sleep mode (which is by definition kept constant regardless of the load factor), while the denominator represent the loss on the power consumption due to offloading the load from the SC to the MC. The lesser load will incur more gain on the power consumption, and thus the designed algorithm will be prone to switch off a SC with lesser load factor. Similarly, we can also see that when the transmit power of SCs is smaller, it becomes more advantageous to use them, and the same can be said if the MC is not very efficient (η_1 is small). Moreover, as one would expect, when the power consumption of a sleeping BS is higher, the less likely it will be that turning it off would result in energy savings. Lastly, we can clearly see that when (10) occurs for any BS in the network, there will be a situation, where the most optimal policy has SCs turned on and consumes less energy than keeping only the MC operational.

IV. Proposed Solution

A. Value Function Approximation

The goal of this article is to find the best policy, that is, the set of SCs to switch off/on per time, in order to optimize the energy consumption of the network. However, since the complexity of the problem is exponentially increasing, in terms of possible configurations, its solution is non-trivial. With that in mind, we propose an intelligent solution, based on RL, in order to minimize the energy consumption of the network without compromising the QoS.

However, traditional RL techniques rely on a state-action matrix, which has to contain an entry for each possible state-action pair. This is an issue in this case because of the aforementioned dimensionality of the problem, which results in a very large matrix, and this in turn makes the algorithm infeasible. To circumvent this issue, we use RL alongside VFA, which is capable of estimating the state based on a set of features, and therefore does not have to maintain a huge structure with all the possibilities [25]. This not only alleviates the computational burden caused by the algorithm implementation,[1] it also results in saving in the memory consumption due to the fact that the proposed methodology needs to store a matrix containing the selected features, whereas traditional RL algorithms need to store a matrix that contains all the states and actions. The size of the matrix, in the former case, grows linearly with the network size, while the matrix size grows exponentially with the network size in the latter case.

The way VFA can be used to enhance traditional RL is by estimating the action-value function with an approximator. It works by finding a set of weights for a known function by solving an optimization problem based on given examples. In particular, the value function denoted by $Q_\pi(S, A)$, which is obtained by following a policy π, is made the subject for an approximation denoted by $\hat{Q}(S, A, \vec{\theta})$; such that $Q_\pi(S, A) \approx \hat{Q}(S, A, \vec{\theta})$ [25]. Although various methods can be used for approximation, the main objective is to minimize the error between $Q_\pi(S, A)$ and $\hat{Q}(S, A, \vec{\theta})$, such that

$$\min_{\vec{\theta}} \left[Q_\pi(S, A) - \hat{Q}(S, A, \vec{\theta}) \right]^2, \tag{11}$$

where S and A are the sets of all states and actions, respectively. $\vec{\theta}$ is a vector of the weights.

Moreover, $\hat{Q}(S, A, \vec{\theta})$ can be expressed by

$$\hat{Q}(S, A, \vec{\theta}) = f(\vec{\theta}), \tag{12}$$

where $f(\cdot)$ is a known function, which is also referred to as hypothesis.

Even though any function can be used as the hypothesis, some render (11) either too hard or infeasible. Linear functions, shallow neural networks, and deep neural networks are commonly used as $f(.)$, and the choice depends on the type of problem. With regards to the optimization, a popular strategy is to use gradient descent (GD) or stochastic GD (SGD) to find $\vec{\theta}$ based on the known examples [25].

In this work, we use a linear hypothesis, meaning that $\hat{Q}(S, A, \vec{\theta})$ is approximated by a linear combination of input features [25], in other words,

$$\hat{Q}(S, A, \vec{\theta}) = \vec{X} \vec{\theta}^{T}, \tag{13}$$

where $\vec{X}$ and $\vec{\theta}$ are ψ-dimensional vectors—where ψ is the number of features selected—containing input features and weights, respectively. Note that $\vec{\theta}^{T}$ indicates $\vec{\theta}$ transposed. Furthermore, using SGD and taking the SARSA case, $\vec{\theta}$ is updated according to [25], [29]

$$\begin{aligned} \vec{\theta}_{t+1} = \vec{\theta}_t + \alpha \Big[& R_{t+1} + \varphi Q_\pi(S_{t+1}, A_{t+1}) \\ & - \hat{Q}(S_t, A_t, \vec{\theta}_t) \Big] \cdot \nabla_\theta \hat{Q}(S_t, A_t, \vec{\theta}_t), \end{aligned} \tag{14}$$

where α is the learning rate for SGD optimization (i.e., the step-size of the gradient descent), and ∇_θ represents the gradient with respect to $\vec{\theta}$. S_t and S_{t+1} are the current and

[1]See Section IV-C4 for a more detailed discussion.

next states respectively, while A_t and A_{t+1} are the taken and next actions respectively. Note that $\vec{\theta}$ is initialized with an arbitrary value, such as zero.

The agent observes a different example after each iteration and updates $\vec{\theta}$ accordingly using (14), which in turn help minimize the objective function given in (11). Therefore, as the number of iterations increases, $\vec{\theta}$ converges with the recursion in (14), and subsequently can be used to find the action-value function, which in turn can guide the policy.[2]

B. SARSA With VFA

For the readers benefit, we have included a small summary of SARSA in this section. The algorithm works by observing the cost of taking actions and updating its estimate of $\vec{\theta}$ at every iteration and then choosing the action based on an ϵ-greedy policy ($\epsilon > 0$) that facilitates the balance between the exploration and exploitation phases, such that the agent continues exploring with a probability of ϵ. Furthermore, ϵ decaying—where the value of ϵ is slightly reduced over time, is employed during the implementations, since it can yield asymptotic convergence to the optimal policy [25]. Therefore, although the policy remains the same over different episodes (i.e., time slots in this case), the probability of exploration is decayed in order to boost the convergence probability. Algorithm 1 [25] contains a pseudo code implementation of SARSA with VFA. Note that lines 22 to 29 in Algorithm 1 corresponds to the stopping criteria, where j_{min} is the minimum number of iterations to take before the stopping criteria come into the effect. C_{min} and C_{max} are the minimum and maximum cost functions observed up to that iteration, respectively. Lastly, Ω is the threshold for the feature scaled cost, while j_{rep} defines the number of iterations to be repeated (the conditions on lines 23 and 24 are kept satisfied) before stopping.

Algorithm 1: Proposed SARSA With VFA

```
1  for Every episode do
2      Initialize the current state, $S_t$;
3      for All actions in $\vec{A}$ do
4          Get features, $\vec{X}$;
5          Estimate value of $Q$ through (13);
6      end
7      Choose action, $A_t$, according to policy;
8      Set $j_{\text{it}} = 0$;
9      for Each iteration do
10         Update $j_{\text{it}} \leftarrow j_{\text{it}} + 1$;
11         Take the action $A_t$;
12         Observe cost, $C$, via (19);
13         Move to next state;
14         for All actions in $\vec{A}$ do
15             Get features, $\vec{X}$;
16             Estimate value of $Q$ through (13);
17         end
18         Choose next action, $A_{t+1}$, according to policy;
19         Update the weights, $\vec{\theta}_t$, using (14);
20         $S_t \leftarrow S_{t+1}$;
21         $A_t \leftarrow A_{t+1}$;
22         if $j_{\text{it}} > j_{min}$ then
23             if $\frac{C - C_{min}}{C_{max} - C_{min}} \leq \Omega$ then
24                 if $C$ is the same for $j_{rep}$ iterations then
25                     Stop executing;
26                     Jump to the next episode;
27                 end
28             end
29         end
30     end
31 end
```

Although the SARSA and *Q*-learning algorithms seem quite similar to each other, there are some key differences that provide characteristic features to each. They are both model-free methods—meaning that they do not require apriori knowledge for the environment model—; however, while *Q*-learning is an off-policy algorithm, SARSA is regarded as on-policy. Hence, *Q*-learning follows different policies when selecting the next action and updating the action-value function, but SARSA follows the same policy (i.e., ϵ-greedy) for both action selection and action-value function update. In addition, there are evidences that *Q*-learning is less suitable for function approximation, as divergence can happen in some cases [25], and thus the SARSA algorithm is preferred in this work in which linear function approximation is implemented.

C. Traffic Aware Energy Optimization via Cell Switching

Leveraging the framework described above, we propose a linear VFA solution to solve (8). Since we are looking to accomplish a globally optimal solution, our proposed framework is centralized and computed at the MC. As the total number of policies increases exponentially with *s*, it would not be scalable to consider any policy as an action. Therefore, we propose a reduced action space as follows.

1) Actions: The actions for the proposed VFA consist of switching OFF/ON different SCs in the network. However, because there are so many possibilities, we propose an alternative representation which allows the algorithm to sample several different possibilities by taking different actions.

This representation is done as follows. First, the status of the SCs in the network are converted to a binary number, such that the SCs that are ON are treated as binary 1, while the SCs that are OFF are considered binary 0. In this regard, the status of the network at time *t* is

$$\vec{\chi}(t) = \{\chi_i(t) | i \in \{1, 2, \ldots, s\}\}, \tag{15}$$

where $\chi_i \in \{0, 1\}$ is the state of the i^{th} SC in the network.

Next, $\vec{\chi}$ is represented by a binary number χ_{b} with *s* digits, such that the status of each SC represents one of its digits and thus $\chi_{\text{b}} = \chi_1\chi_2\cdots\chi_s$.

Within the proposed representation, the set of possible actions is defined as

$$A = \Big\{0, \pm\xi^0, \pm\xi^1, \ldots, \pm\xi^s\Big\}, \tag{16}$$

where ξ is a decimal constant number, defining the inter-space between two consecutive actions.

[2]Although the number of iterations/examples needed for convergence depends on various factors, such as the environment, the size of the problem, etc., this phenomenon is observed and discussed in Section V-E for our problem. In particular, the learning/training phase is clearly observed in Figs. 3(a) and 3(b), in which the designed method learns (i.e., performs (11) and (14)) in the first few episodes and then converges to the optimal value, which is produced by the exhaustive search in this case.

In this case, taking an action at time *t*, means to perform

$$\chi_{\mathrm{d}}(t+1) = \chi_{\mathrm{d}}(t) + A_z, \tag{17}$$

where A_z is an entry[3] from $\vec{A}$ and χ_{d} is the decimal representation of χ_{b}. In other words, an action consists of turning on/off a number of BSs, depending on the current status and on A_z. This ensures that, instead of checking all possible statuses of SCs, we are taking only some samples of the entire set.

To illustrate the proposed idea, consider the following example. In a network with 4 SCs at time *t*, let $\vec{\chi}(t) = \{1,1,0,1\}$. In this case, $\chi_{\mathrm{d}}(t) = 13$. Now, let us assume that $\xi = 2$ and that the action selected is $+\xi^1$. In this case, the next status will be $\chi_{\mathrm{d}}(t+1) = \chi_{\mathrm{d}}(t) + \xi^1 = 15$, or in other words $\vec{\chi}(t+1) = \{1,1,1,1\}$, which implies that all BSs are turned on at time *t* + 1.

2) Features: Feature selection is one of the fundamental aspects while designing a SARSA VFA-based solution, since it has a great impact on both the convergence and the performance of the developed solution. In almost every decision and optimization problems, there are key factors (or determinants), each of which have impacts on different scale, used while determining the optimality. For instance, if we want to buy a house with a minimum price, then we need to observe the key determinants of the price, such as the year of construction, location, number of bedrooms, etc. Similarly, in our problem given in (8), the primary objective is the minimization of the total power consumption of the network, and thus we need to include the determinants of the total power consumption in the feature selection. It can be observed from (1) that the load factor is a very important factor in determining the power consumption of a BS, the total load factor of each SC is included as features in the designed algorithm.

Moreover, as it is recommended in [25] that it would be better to include the interaction between the selected determinants in order to approximate well, we capture the interactions between the features as follows: 1) the total power consumption of the network (*P*): the result of the load factors of all the SCs and 2) the resultant load factor of the MC (λ_1): the combination of the load factor of the MC itself (before offlaoding) and the load factors of switched off SCs. The inclusion of the resultant load factor of the MC in the feature matrix is also important in a way because it indicates whether the constraint in (8) is violated or not. Therefore, the features used by the MC in order to find $\hat{Q}$ are the total power consumption of the network and the total load factor of each BS, such that

$$\vec{X} = [P, \lambda_1, \lambda_2, \ldots, \lambda_{s+1}]. \tag{18}$$

3) Cost: The cost, or penalty, that we propose is described as the total power consumption plus a penalty—which is proportional to the number of SCs and the MC load—if the load of the MC is exceeded. We can formally define it as

$$C = P + s\sigma\kappa\lambda_1, \tag{19}$$

where κ is a penalty factor, and $\sigma \in \{0,1\}$ indicates whether or not the MC is overloaded. The total power consumption of the network, *P*, is included in (19) in order to demotivate the agent from taking actions that result in more power consumption, since the primary objective of this work is to minimize the power consumption of the network through cell-switching.

4) Complexity: Since the problem given in (8) is an $\mathcal{NP}$-hard problem,[4] which does not have any deterministic polynomial-time solution yet, its optimal solution can be found with an exhaustive search (or brute-force search), which is quite demanding in terms of computational complexity. Therefore, with the proposed VFA-based solution, we intend to produce a sub-optimal solution that greatly reduces the complexity of finding a good operating point by compromising on exact optimality. Although the operating point that the proposed algorithm finds is not guaranteed to be the optimal point, which exhaustive search guarantees, the results—discussed in Section V-E—confirm that the proposed solution follows the optimal solution very closely, especially after the initial training phase.

In this regard, since there are *s* SCs which could be switched off/on in any combination, an exhaustive search approach would have complexity $O(2^s)$, while our proposed approach only keeps track of 2*s* actions, and therefore has a complexity *O*(*s*). However, note that our solution does not strictly guarantee the constraint in (8). This is important to give the RL algorithm the chance to explore different actions and learn to make the right decision. By incorporating $\sigma\kappa$ into the cost, we can influence the RL algorithm not to violate the constraint, and thus satisfy the QoS requirements of the users whilst seeking the best policy regarding energy consumption.

V. Performance Evaluation

The proposed cell switching framework is implemented in a locally centralized manner at each MC. As a result, the proposed SARSA with VFA model needs to be trained for only one MC-SCs configuration and then the trained model is shared by all other configurations throughout the network. The simulation parameters are provided in Table I, while Table II presents the power consumption characteristics of the BSs used in the simulations. Note that ξ is taken as 2 in this work, as it is more convenient when compared to another selection due to the fact that the cell-switching problem is formulated as a binary problem. Thus, $\xi = 2$ guarantees to visit all the states when the number of iterations is sufficiently large, which may not be provided by any other selection of ξ.

A. Data Set

In order to calculate the power consumption through (1), λ is required for each BS. In this regard, to obtain λ values, we use a real call detail records (CDR) data set provided by Telecom Italia, in which the city of Milan is divided into 10,000 square-shaped grids with a dimension of 235 × 235 meters. Then, within each grid, user call, text message, and Internet activity levels were recorded with 10-minutes resolution for a two months period (November and December 2013).

[3]Not all entries of $\vec{A}$ can be selected at every time instant: the only valid actions are those which render $0 \leq \chi_{\mathrm{d}} \leq 2^s - 1$.

[4]This can be proved by formulating the problem in terms of Knapsack problem, which is a well know $\mathcal{NP}$-hard problem [30], however due to the page limitations, we are unable to demonstrate this here.

TABLE I
SIMULATION PARAMETERS

Parameter	**Value**
SARSA with VFA	
Chance of choosing random action, ϵ	0.8
Learning rate, α	10^{-7}
Discount factor, γ	0.9
Inter-space between consecutive actions, ξ	2
Min. number of iterations for stopping, j_{min}	10
Threshold for the feature scaled cost, Ω	5×10^{-2}
Consecutive iterations for stopping, j_{rep}	10
Maximum number of iterations	100
General	
Number of time slots	144
Number of days	1
Number of grids considered for the MC	2
Number of grids considered per SC	1
Bandwidth for the MC	20 MHz
Bandwidth per SC	20 MHz

Even though the data set consists only of unitless activity level values and there is no information provided regarding the data processing phase, the activity levels can be interpreted as grid-wise relative traffic loads, since they represent the volume of user-mobile network operator interaction at each time slot. In the data processing phase of this work, first, we combine aforementioned separated activity levels (i.e., call, text message, and Internet). Then, we randomly pick two grids for the MC and one grid for each SC. Note that the activity levels at the two grids selected for the MC are further combined to create a traffic load data. After that, all the activity levels are normalized together, and the obtained values are treated as traffic loads for each cell.

Based on the resolution of the data set, the interval between one switching instance and another is set to 10 minutes in this work. This is required in order to account for the activation and deactivation latency of the SCs. Moreover, since the network traffic is assumed to be static during this period in our simulation, the switching interval is not expected to be long in order not to negatively impact the performance of the network.

B. Benchmarking

Four different benchmark methods are used for comparison purposes, and they will be elaborated individually in the following paragraphs.

1) Sorting: Inspired by [31], [32], the sorting algorithm is developed to compare the results of the proposed solution. In this method, the SCs are sorted in ascending order based on their load factors, λ. Then, they are switched off sequentially until there is no available capacity left at the MC, and the rest of the SCs are kept ON. Given the power consumption profile in (1) and the characteristics of different types of BSs in Table II, the MC consumes more power than the SCs for the same value of λ. As such, it is wiser to switch off a SC with smaller traffic load in order to save more energy. This concept lies at the heart of the sorting algorithm, since it aims at minimizing the energy consumption of the network. On the other hand, as the SC switching off is performed only when there is enough capacity at the MC, this method also guarantees the service of the users after the offloading process.

TABLE II
POWER PROFILES FOR BSS [27]

BS Type	Efficiency	Power Consumption [W]		
		Transmit	Operational	Sleep
	η_j	$P_{\text{T},j}$	$P_{\text{o},j}$	$P_{\text{s},j}$
Macro	4.7	20	130	75
RRH	2.8	20	84	56
Micro	2.6	6.3	56	39
Pico	4.0	0.13	6.8	4.3
Femto	8.0	0.05	4.8	2.9

2) All-ON Method: There is no switching implemented in this method, meaning that all the SCs are always kept ON. Accordingly, no offloading is needed as well in this case provided that none of the SCs are switched off at any time. Therefore, it can be inferred that there is no concern of QoS in this method, since all the users are served by the BSs (either MC or SC) that they were associated in the first place.

3) All-OFF Method: In this methods, the SCs are always kept switched off and their data traffics are offloaded to the MC. However, this method is performed blindly, meaning that the data traffic of the SCs are offloaded to the MC regardless of its available capacity. This means that the users, which are normally served by SCs, are vulnerable to service disruptions, since there is no guarantee that they will be served by the MC. Even if the service is provided by the MC, the QoS would be reduced in case there are more users than the available capacity, and in that case the MC reduces the available resources for each user by certain amount in order to keep all the users served.

4) Exhaustive Search: Exhaustive search is a method that tries to find the best policy among the set of all possible switching options consisting of the OFF/ON states of the SCs. In particular, given the available capacity of the MC as a constraint, this method searches for the option with the least energy consumption. Hence, this method guarantees the service for each user in the case of offloading, which in turn prevents the QoS of the users from being violated. Note that exhaustive search returns the optimum policy, and thus the objective of any algorithm should be to mimic it as much as possible.

Conventional RL methods such as *Q*-learning and SARSA are not used as benchmarks as they are not able to compute the value function for all state-action pairs since the number of state space involved in our work is very large. In addition, the use of RL with linear VFA have been proven to be more data and computation efficient compared to deep RL techniques [25], [33]. Moreover, the way that our proposed algorithm framework has been carefully designed, linear VFA is more suitable for solving this problem, as new algorithm framework design would be required in order to apply deep RL methods. Hence, deep RL technique is not used as one of the benchmarks in this work.

C. Performance Metrics

In this section, the metrics, which are used to evaluate the performance of the proposed solution and the benchmark methods are presented.

1) Gain: In this work, we are interested in the percentage gain on the total energy consumption compared to the all-ON method. It is calculated as $G(\%) = (E_{\text{on}} - E_x)/E_{\text{on}}$, where E_{on} and E_x are the total energy consumption in joules with all-ON method and with one of the other methods, such that $E_x \in \{E_{\text{on}}, E_{\text{es}}, E_{\text{sort}}, E_{\text{vfa}}\}$ where E_{off}, E_{es}, E_{sort}, and E_{vfa} are the total energy consumption in joules with all-OFF, exhaustive search, sorting, and the proposed VFA-based methods, respectively.

2) Power Consumption: Power consumption in watts during the simulation time are obtained for each method. This is a beneficial metric to evaluate the performance of the methods, since it reflects the variations in power consumption for different times of a day. Moreover, given that the gain is calculated on the energy consumption by accumulating the power consumption during the simulations, which can also be interpreted as upsampling, the detailed behaviours of the developed methods are kind of lost. Thus, power consumption is also a utilitarian metric that paves the way for detailed behavioural observations.

3) Average Network Throughput: The total required RAN throughput, $T_{\text{T,r}}$, is calculated by combining the throughput required from each cell as follows:

$$T_{\text{T,r}}(t) = \sum_{i=1}^{s+1} T_{\text{r},i}, \tag{20}$$

where $T_{\text{r},i}$ is the required throughput from B_i, and is calculated as follows:

$$T_{\text{r},i}(t) = T_{\text{u},i}(t) N_{\text{u},i}(t), \tag{21}$$

where $T_{\text{u},i}(t)$ is the average throughputs for users allocated by B_i and where $N_{\text{u},i}(t)$ is the number of users served by B_i at time *t*.

However, there is one caveat that since the backhaul capacity of the cells is limited by the installed backbone, B_i penalizes the throughput for each user by Υ_i when the combined demand of the users exceeds $T_{\text{p},i}$, the maximum installed capacity of B_i, such that

$$\hat{T}_{\text{u},i}(t) = T_{\text{u},i}(t) - \Upsilon_i(t), \tag{22}$$

where $\hat{T}_{\text{u},i}(t)$ is the average throughput for users allocated by B_i at time *t* after penalization. This also ensures the condition in (9).

The throughput penalty, Υ_i, is calculated as

$$\Upsilon_i(t) = \begin{cases} \dfrac{T_{\text{r},i}(t) - T_{\text{m},i}}{N_{\text{u},i}(t)}, & \text{if } T_{\text{r},i}(t) > T_{\text{m},i} \\ 0, & \text{otherwise,} \end{cases} \tag{23}$$

Next, as explained in Appendix A, normalized throughput is represented by the load factor in this work, thereby the provided normalized network throughput is given as follows:

$$\tilde{T}_{\text{T,p}}(t) = \sum_{i=1}^{s+1} u(-\lambda_i(t) + 1)\lambda_i(t) + u(\lambda_i(t) - 1), \tag{24}$$

where $\tilde{T}_{\text{T,p}}(t)$ is the normalized throughput of the network and $u(\cdot)$ is the unit step function.

D. Scenarios

The developed benchmark methods and the proposed VFA-based switching algorithm are tested in two different scenarios, namely Scenario A and Scenario B.

1) Scenario A: This is a simplistic scenario, where there is only one type of SC (micro) in the network. Moreover, the sleep mode power consumption for the SCs are assumed to be zero, such that the SCs are not contributing to the total power consumption at all when they are off.

2) Scenario B: This is a complex scenario, comprising four different types of SCs, e.g., micro, remote radio head (RRH), pico, and femto, are deployed in the network, and the number of SCs are distributed in these four types almost[5] equally. Moreover, the sleep mode power consumption is not assumed to be zero in this scenario, instead the values in Table II are used. Therefore, this scenario is more realistic than Scenario A, as in real networks, there are heterogeneous combinations of SCs and the sleep mode power consumption is not zero.

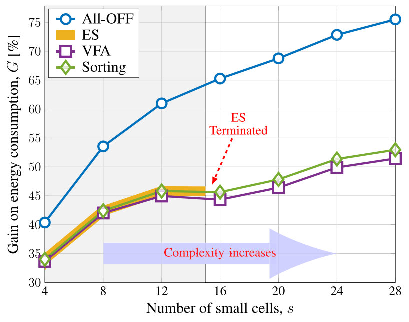
E. Results

We first analysed the convergence of the developed algorithm, as the convergence plays an important role in determining the performance and stability of the algorithm. For this, we plotted the penalty calculated through (19) for each iteration and episode, followed by averaging them out over the episodes. We observed that, for any network size (e.g., any value of *s*) that is considered during the simulations, the developed algorithm manages to converge after around 20 iterations (note that the maximum number of iterations allowed is 100). However, we couldn't include such graph in this article due to the page limitations.

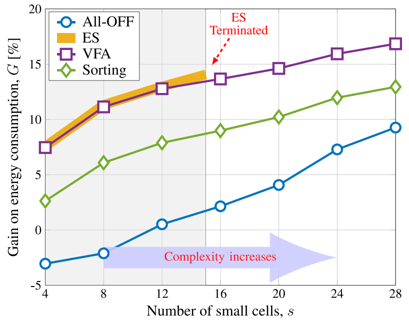
As explained in Section V-C1, Fig. 2 demonstrates how much gain in energy consumption is obtained when all-OFF, exhaustive search, sorting, and proposed VFA-based methods are compared to all-ON method. Fig. 2(a) shows the gain results for Scenario A, while the results for Scenario B are presented in Fig. 2(b). Note that since the exhaustive search method is computationally demanding with $O(2^s)$, where it doubles the elapsed time when *s* is incremented by 1, it is allowed to run only until $s = 15$ for both Fig. 2(a) and Fig. 2(b). The idea in Fig. 2 is that we first compare the performances of all-OFF, sorting, and the proposed method to the exhaustive search, which is optimum, in a smaller network due to the time requirement for exhaustive search when *s* increases. Then, once we have an idea about the performance of the methods compared to the optimum, we terminate the exhaustive search and keep the other methods running in order to observe further behaviours in larger scale networks.

It can be seen in Fig. 2(a) that the sorting method is working exactly same with the exhaustive search, while the proposed method follows them quite closely. These outcomes can lead to a conclusion that the proposed method is outperformed by a more simpler and straightforward algorithm, and one can question the validity of the proposed method. However, when

[5]It is not always possible to distribute them equally, since the amount of SCs are sometimes not divisible by four.



(a) Results for Scenario A when $\kappa = 20$.



(b) Results for Scenario B when $\kappa = 10$.

Fig. 2. Percentage gain performances compared to the all-on method for Scenario A and Scenario B.

Fig. 2(a) is reconsidered together with Fig. 2(b), it can be inferred that the better performance of the sorting algorithm is not generalized. While it gives promising results in simplistic Scenario A, it starts under-performing in the complex and realistic Scenario B. On the other hand, given that the proposed VFA-based method gives very close results to the exhaustive search in both scenarios, it seems quite immune to the changes in the scenario with generalized good performance. In other words, although there might be simpler alternatives to the proposed method in simple scenarios, which are mostly unrealistic, the proposed method takes the advantage of being capable of generalization and works properly even when the scenario becomes more complicated and realistic owing to the nature of the developed VFA-based RL algorithm. This, in turn, makes the proposed method work properly regardless of the conditions, while the sorting method, for example, relies on the simplicity of the scenario, making it impractical for realistic scenarios, where the assumptions in Scenario A are no more valid.

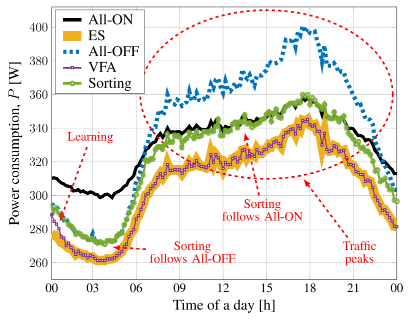
Another point about Fig. 2 is that the gain decreases significantly when the scenario is switched from A to B. For the proposed method, for instance, the gain drops from around 52% to 17% when $s = 30$, which yields around 67% reduction. This is mainly due to more heterogeneity of Scenario B, where there are four types of SCs. Scenario A includes only micro cell, which is the second most energy consuming SC after RRH according to Table II, making the SCs in Scenario A consume a considerable amount of energy. For Scenario B, on the other hand, there are four types of SCs with distinctive power profiles, and thus total power consumption decreases because of the inclusion of pico and femto cells, which consume small amount of energy, in the network. Besides, while SCs still consume energy when they are switched off in Scenario B, the sleep mode power consumption is assumed to be zero for Scenario A, making it consume overall less energy when switching is performed.

The last point that is worthy to discuss about the findings in Fig. 2 is related to the all-OFF method. While it outperforms all the other method in terms of gain in Fig. 2(a), it becomes the worst-performing method in Fig. 2(b). The reason behind this phenomenon is again the characteristic diversity between Scenarios A and B. There is only micro cell, which is demanding in energy, in Scenario A, and thus switching off SCs almost always result in less energy consumption, whereas, due to the heterogeneity of Scenario B, the optimal policies for switching off are different from each other for each type of SC. In particular, (10) holds for larger λ_j, $j>1$ values when the type of SC goes from femto cell to RRH. This means that the number of cases, where switching off is profitable, is larger for micro cell than that of femto and/or pico cells. Therefore, since the all-OFF method switches off all the available SCs regardless of their types, the overall process becomes less profitable in Scenario B when compared to Scenario A.

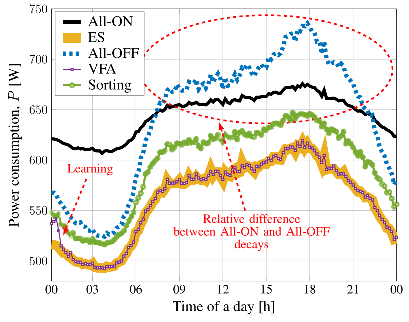
Fig. 3 reveals the power consumption behaviours of the developed methods for various numbers of SCs included in the network. Similar to the results in Fig. 2, for consistency, we present the power consumption results for integer multiples of four, where the types of SCs are distributed equally. Hence, s value is altered as $s = 4$, $s = 12$, and $s = 28$, respectively. Note that Fig. 3(c) does not include the exhaustive search method, since it is allowed to run until $s = 15$ for computational complexity reasons.

Fig. 3(a) demonstrates the power consumption performance of the developed methods when $s = 4$. The findings suggest that the proposed VFA-based method manages to mimic the exhaustive search almost perfectly apart from the initial learning phase. Given the computational complexities for the exhaustive search and the proposed method are $O(2^s)$ and $O(s)$, respectively, these results confirm that the proposed method performs quite well (i.e., producing near-optimal results) with drastic decrease in computational complexity.

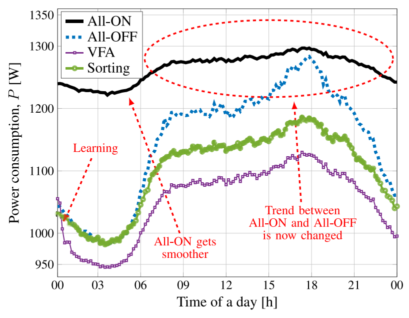
It is also worth discussing the learning phase of VFA algorithm, which is common in all the three cases of s. This behavior is expected as we propose an online learning framework, where VFA-based cell switching method is deployed without any prior knowledge, and thus it learns by interacting with the real environment. In other words, the promising performance after the initial learning phase is due to the experience obtained during the training. The slightly worse initial performance is due the fact that VFA takes more random actions in the beginning in order to increase the knowledge about the environment. This process is referred to as *exploration*. Then, after the exploration, the randomness in the actions taken decreases with the number of episodes in order to let the VFA start using the information it received, which is known as *exploitation*. One important point here is that the exploration process in the developed model takes a short amount of time, making the online implementation feasible, since longer learning phases would undermine the advantage



(a) $s = 4$



(b) $s = 12$



(c) $s = 28$

Fig. 3. Average power consumption performances of the developed methods for various number of SCs.

of the VFA based solutions. The reason behind preferring the online implementation over the offline one is that the former case is model-free, where it does not require any prior knowledge, while a full environmental knowledge is needed in the latter. This, in turn, renders the online implementation more practical. Therefore, even though the offline implementation is free from the possible negative impacts of the training process, it is comparatively less functional in real scenario, where full prior knowledge is often inaccessible.

Another interesting aspect that can be deduced from Fig. 3(a) is the behavior of the sorting method. It mirrors the all-OFF method in the beginning, where the data traffic is relatively low, whereas it starts following the all-ON method when the traffic load increases. This is because the sorting method manages to switch off most (even all) SCs, which is similar (same) behavior with the all-OFF method, when the traffic load is lighter, since there is more offloading opportunities in the MC owing to the low traffic patterns. Moreover, the amount of offloaded data is also comparatively less at these times. When the traffic volume becomes higher, on the other hand, since the amount of data to be offloaded and the occupancy of the MC increases simultaneously, the switching off becomes much harder, which is quite similar to the behavior of all-ON.

Fig. 3(b) presents the power consumption results when $s = 12$. Similar to Fig. 3(a), we observe the sub-optimal results of VFA during the training phase. However, other than being short in time, the appreciable thing about this training phase is that the results that VFA produces are still reasonable even though it is not optimal. Another interesting observation is the relative difference between all-ON and all-OFF methods shrinks compared to the one in Fig. 3(a). Taking into account the peak points, the relative difference between the all-OFF method and all-ON method decreases by around 25% when s increases from 4 to 12.

Two important questions arise from these findings. First, why does the all-OFF method result in more power consumption than the all-ON method? It is counter-intuitive to observe such results where switching all the SCs off causes more power consumption than always keeping all the SCs ON. Moreover, we observe that all-ON outperforms all-OFF especially when the traffic loads are higher. The reason behind this is that, as repeated previously, considering (10) together with Table II, it is usually non-profitable to switch off SCs when the traffic load is above some certain threshold, which is different for each type of SC. Therefore, it is not a rule of thumb that the switching off is always resulting in less power consumption. The outcomes in Fig. 2(b), where the all-OFF method gives negative gains, also confirm this conclusion. Nonetheless, this is not the only condition that makes the all-ON more favorable than the all-OFF method in terms of power consumption. It is also the intensity of the SCs in the network. Since s is not large enough in both Figs. 3(a) and 3(b), the contribution of the SCs to the power consumption is comparatively less than that of the MC, therefore, the overall energy saving resulting from switching off SCs cannot prevail against the loss caused by offloading traffic to the MC.

Second, why does the relative difference between the all-OFF and all-ON methods decay when s rises up from 4 to 12? The answer for this question is related to the last discussion for the previous question; since the intensity of the SCs increases in the network with increasing s, the dominance of the MC in the total power consumption scales down. This subsequently renders the gain resulting from switching off more significant, and thus the all-OFF method starts being more reasonable. Hence, the number of instances that all-ON outperforms the all-OFF method also decreases when s is increased from 4 to 12. The results in Fig. 2(b) again supports this conclusion, as the percentage gain enhances with increasing s.

Fig. 3(c) showcases the power consumption results when $s = 28$. It is again worth noting that exhaustive search is not included this time, since it is run until $s = 15$, owing to the computational complexity concerns. Unlike the results in Figs. 3(a) and 3(b), there is no point in Fig. 3(c), where all-OFF outperforms all-ON. Similar to the previous discussions on this topic, the distribution of the total power consumption among the SCs and the MC is an integral aspect of the performances of the all-ON and all-OFF methods. Given that the

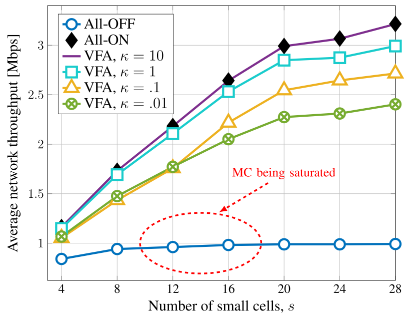


Fig. 4. Results for normalized network throughput against the number of SCs. The network throughput is calculated by averaging out the obtained throughput values at each time slot during the simulation period.

SCs now consume considerable amount of energy due to their increased number, the gain obtained from switching off the SCs, which are profitable to switch off according to (10), prevails over the loss incurred by switching off the SCs, which are non-profitable to switch off.

Another interesting point about the findings in Fig. 3(c) is that the power consumption for the all-ON method becomes smoother compared to the results in Figs. 3(a) and 3(b). The relative peak-to-peak difference, for example, was around 20% when $s = 4$, whereas it drops to 6.2% when s increased to 28. This is again because of the MC losing its dominance in the total power consumption. While, when fully-loaded, 72.3% of the power consumption comes from the load dependent part for the MC, this rate is around 28% on average (minimum: femto cell with 8.3%, maximum: RRH with 66.7%) for the SCs. In other words, the MC consumes more on the load dependent part, whereas SCs consume more on the static power. Therefore, for the smaller s values, the load dependent power consumption is higher as the MC is the main contributor to the total power consumption, while the load dependent power consumption becomes relatively less and the static power gets more significant for higher values of s.

Fig. 4 shows normalized average network throughput, which is calculated through (24), for various κ values of VFA, the all-ON and the all-OFF methods. Furthermore, the activity levels in the data set are assumed to be in Mbps after the pre-processing detailed in Section V-A. The objective of demonstrating these results is to highlight the impact of κ value on the performance of the proposed VFA-based switching algorithm. In addition, the findings also display the cost of switching off all the SCs without taking into account the available capacity at the MC. As such, the results suggest that there is an upper bound for the all-OFF method, since it only relies on the capacity of the MC, and after normalization, each BS (MC or SC) has a capacity of unity at maximum. Given that the MC is the only BS that is kept ON, all-OFF switches off all the SCs and offloads their traffic to the MC, meaning that it has one unit of capacity available in the network. Therefore, these results also confirm that having a blind policy, that is, acting without considering the environmental conditions and/or constraints, is not a wise idea, since it results in the degradation of the QoS of users as well as being more costly in power consumption on some occasions, as already proven in Figs. 2(b), 3(a), and 3(b). The purpose of presenting the results for the all-ON method is to demonstrate the upper bound that can be achieved in throughput.

As seen in Fig. 4, we obtained promising results for the proposed method. It gives quite close results to the all-ON method, proving a good performance as it is producing similar results to the best case. By considering the findings in Figs. 2, 3, and 4 together, it is easy to deduce that the proposed VFA-based switching algorithm performs outstandingly well in terms of both power consumption and the throughput, since it reduces the power consumption (similar to exhaustive search) without compromising on the QoS of the users (similar to the all-ON method). Provided that exhaustive search and all-ON are the best methods in terms of power consumption and QoS, respectively, the proposed method combines the advantages of both methods.

Fig. 4 also showcases the impact of κ in (19) on the performance of the developed VFA model. The results suggest that the throughput performance of the proposed VFA decreases with decreasing values of κ. This is because higher values of κ will result in more cost been incurred for the case when the demanded capacity exceeds the available capacity at the MC. As explained in Section V-C, when the demanded capacity is higher than the available one, the network reduces the allocated bandwidth for each user in order to accommodate the demands of all users. In this regard, the agent refrains from switching off a SC, whose demanded capacity is larger than the available one at the MC, which in turn helps to keep the QoS above the required level. However, for the smaller values of κ, the agent starts following more relaxed policies on the given constraint, where it takes more actions that are against the above-mentioned demanded/available capacity criterion. Hence, the obtained throughput starts decreasing for lower values of κ. For the extreme scenario, where $\kappa = 0$, then (19) becomes $C = P$, meaning that the agent only focuses on the total power consumption and does not care about the constraint of available capacity at the MC. Intuitively, the agent would be reducing the total power consumption as much as possible at the expense of QoS degradation.

VI. Conclusion

In this article we presented a RL-based solution for cell switching, which is capable of learning the best policy in a dense HetNet environment in order to save energy and satisfy the QoS at the same time. We evaluated our solution using real data from Milan and compared it with various benchmark methods. The results in terms of power and energy consumption show that the proposed method can perform just as well as the exhaustive search method, which produces the optimum solutions, regardless of the complexity and size of the given scenario. Moreover, the proposed method resulted in much fewer computations than that of exhaustive search, meaning that it is a scalable method. Furthermore, network throughput was also measured, and we observed that the proposed method gave similar results with the all-ON method, which is the best in terms of the QoS given that it does not include any offloading. Therefore, the proposed VFA-based method resulted in a significant reduction in the network's energy consumption without much compromise on the QoS, thus making it suitable for practical application.

APPENDIX A
NORMALIZED NETWORK THROUGHPUT

After penalization, the provided throughput for B_i at time *t* can be expressed as the product of the average user throughput and the number of users it serves, as

$$T_{\mathrm{p},i}(t) = \hat{T}_{\mathrm{u},i}(t) N_{\mathrm{u},i}. \tag{25}$$

After that, using (22) into (25), it is obtained that

$$T_{\mathrm{p},i}(t) = \big(T_{\mathrm{u},i}(t) - \Upsilon_i\big) N_{\mathrm{u},i}. \tag{26}$$

When (23) is used in (26), $T_{\mathrm{p},i}(t)$ becomes

$$T_{\mathrm{p},i}(t) = \ T_{\mathrm{u},i}(t) N_{\mathrm{u},i} - \begin{cases} T_{\mathrm{r},i}(t) - T_{\mathrm{m},i}, & T_{\mathrm{r},i}(t) > T_{\mathrm{m},i} \\ 0, & \text{otherwise,} \end{cases} \tag{27}$$

after simplifying.

Then, using (21), (28) can be rewritten as

$$T_{\mathrm{p},i}(t) = \begin{cases} T_{\mathrm{m},i}, & T_{\mathrm{r},i}(t) > T_{\mathrm{m},i} \\ T_{\mathrm{r},i}(t), & \text{otherwise.} \end{cases} \tag{28}$$

Next, the throughput can be normalized with respect to the installed capacity, as

$$\lambda_i(t) = \frac{T_{\mathrm{r},i}(t)}{T_{\mathrm{m},i}}, \tag{29}$$

where $\lambda_i(t)$ is the load factor of B_i at time *t*, but as mentioned earlier, it is also treated as the normalized throughput of B_i at time *t*. Therefore, dividing (28) and using (29), the normalized throughput can be obtained as

$$\tilde{T}_{\mathrm{p},i}(t) = \begin{cases} 1, & \lambda_i(t) > 1 \\ \lambda_i(t), & 0 \le \lambda_i(t) \le 1, \end{cases} \tag{30}$$

Then, using the unit step function, (30) can be rewritten as

$$\tilde{T}_{\mathrm{p},i}(t) = u(-\lambda_i(t) + 1)\lambda_i(t) + u(\lambda_i(t) - 1). \tag{31}$$

Lastly, in order to calculate the total provided network throughput, a summation is performed over all BSs, arriving at (24).

APPENDIX B
PROOF OF THEOREM 1

Using (2), the difference in power consumption ΔP when considering changing δ_j to 1 can be expressed as

$$\Delta P = P_t - P_{t-1} = \sum_{i=1}^{s+1} P_{i,t} - \sum_{i=1}^{s+1} P_{i,t-1}, \tag{32}$$

where P_t is the total power consumption of the network at time *t* and $P_{i,t}$ is the power consumption of B_i at time *t*.

Next, (32) can be expanded as

$$\begin{aligned} \Delta P = {} & P_{1,t} + P_{j,t} + \sum_{i=2, i\neq j}^{s+1} P_{i,t} \\ & - \left(P_{1,t-1} + P_{j,t-1} + \sum_{i=2, i\neq j}^{s+1} P_{i,t-1} \right). \end{aligned} \tag{33}$$

Assuming $P_{i\notin\{1,j\},t} = P_{i\notin\{1,j\},t-1}$, meaning all other SCs are kept at their states, (33) becomes

$$\Delta P = P_{1,t} + P_{j,t} - P_{1,t-1} - P_{j,t-1}. \tag{34}$$

Next, using (1) and replacing (6) and (7) into (34) yields

$$\begin{aligned} \Delta P = {} & P_{\mathrm{o},1} + \eta_1\big(\lambda_{1,t-1} - \phi_j\lambda_{j,t}\big)P_{\mathrm{T},1} + P_{\mathrm{o},j} \\ & + \eta_j\lambda_{j,t}P_{\mathrm{T},j} - P_{\mathrm{o},1} - \eta_1\lambda_{1,t-1}P_{\mathrm{T},1} - P_{\mathrm{s},j}, \end{aligned} \tag{35}$$

which can be further simplified to

$$\Delta P = P_{\mathrm{o},j} + \eta_j\lambda_{j,t}P_{\mathrm{T},j} - \eta_1\phi_j\lambda_{j,t}P_{\mathrm{T},1} - P_{\mathrm{s},j}. \tag{36}$$

From (36), it is easy to see that $\Delta P < 0$ when

$$P_{\mathrm{o},j} + \eta_j\lambda_{j,t}P_{\mathrm{T},j} < \phi_j\eta_1\lambda_{j,t}P_{\mathrm{T},1} + P_{\mathrm{s},j}, \tag{37}$$

Note that, in order to isolate $\lambda_{j,t}$ in (37), we must divide both sides by $\phi_j\eta_1 P_{\mathrm{T},1} - \eta_j P_{\mathrm{T},j}$, and thus (10) is only valid for $\phi_j\eta_1 P_{\mathrm{T},1} - \eta_j P_{\mathrm{T},j} > 0$. Lastly, we remove the index *t* from (37) to make it general and solve for λ_j, yielding (10).

REFERENCES

[1] M. Feng, S. Mao, and T. Jiang, "Base station on-off switching in 5G wireless networks: Approaches and challenges," *IEEE Wireless Commun.*, vol. 24, no. 4, pp. 46–54, Aug. 2017.

[2] C. Luo and J. Liu, "Load based dynamic small cell on/off strategy in ultra-dense networks," in *Proc. 10th Int. Conf. Wireless Communi. Signal Process. (WCSP)*, Oct. 2018, pp. 1–6.

[3] Y. Sun and Y. Chang, "Energy-efficient user association and cell sleeping strategy for multi-tier ultra-dense small cell networks," *Electron. Lett.*, vol. 54, no. 12, pp. 787–789, Jun. 2018.

[4] J. Wu, J. Liu, and H. Zhao, "Dynamic small cell on/off control for green ultra-dense networks," in *Proc. 8th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, 2016, pp. 1–5.

[5] S. Buzzi, I. Chih-Lin, T. E. Klein, H. V. Poor, C. Yang, and A. Zappone, "A survey of energy-efficient techniques for 5G networks and challenges ahead," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 4, pp. 697–709, Apr. 2016.

[6] W. Yu, H. Xu, A. Hematian, D. W. Griffith, and N. Golmie, "Towards energy efficiency in ultra dense networks," in *Proc. IEEE 35th Int. Perform. Comput. Commun. Conf. (IPCCC)*, 2016, pp. 1–8.

[7] F. Ahmed, M. Naeem, W. Ejaz, M. Iqbal, A. Anpalagan, and M. Haneef, "Energy cooperation with sleep mechanism in renewable energy assisted cellular hetnets," *Wireless Pers. Commun.*, vol. 116, no. 1, pp. 105–124, 2021.

[8] A. Mohamed, O. Onireti, M. A. Imran, A. Imran, and R. Tafazolli, "Control-data separation architecture for cellular radio access networks: A survey and outlook," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 446–465, 1st Quart., 2016.

[9] O. Onireti, A. Imran, M. A. Imran, and R. Tafazolli, "Energy efficient inter-frequency small cell discovery in heterogeneous networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 9, pp. 7122–7135, Sep. 2016.

[10] J. Yang, Z. Pan, and L. Guo, "Coverage and energy efficiency analysis for two-tier heterogeneous cellular networks based on matérn hard-core process," *Future Internet*, vol. 12, no. 1, p. 1, 2020.

[11] F. Hussain, S. A. Hassan, R. Hussain, and E. Hossain, "Machine learning for resource management in cellular and IoT networks: Potentials, current solutions, and open challenges," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 1251–1275, 2nd Quart., 2020.

[12] O. Alamu, A. Gbenga-Ilori, M. Adelabu, A. Imoize, and O. Ladipo, "Energy efficiency techniques in ultra-dense wireless heterogeneous networks: An overview and outlook," *Int. J. Eng. Sci. Technol.*, vol. 23, no. 6, pp. 1308–1326, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2215098619328745

[13] P.-Y. Kong and D. Panaitopol, "Reinforcement learning approach to dynamic activation of base station resources in wireless networks," in *Proc. IEEE 24th Annu. Int. Symp. Pers. Indoor Mobile Radio Commun. (PIMRC)*, 2013, pp. 3264–3268.

[14] A. El-Amine, M. Iturralde, H. A. Haj Hassan, and L. Nuaymi, "A distributed Q-learning approach for adaptive sleep modes in 5G networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, 2019, pp. 1–6.

[15] F. E. Salem, Z. Altman, A. Gati, T. Chahed, and E. Altman, "Reinforcement learning approach for advanced sleep modes management in 5G networks," in *Proc. IEEE 88th Veh. Technol. Conf. (VTC-Fall)*, 2018, pp. 1–5.

[16] A. I. Abubakar, M. Ozturk, S. Hussain, and M. A. Imran, "Q-learning assisted energy-aware traffic offloading and cell switching in heterogeneous networks," in *Proc. IEEE 24th Int. Workshop Comput. Aided Model. Design Commun. Links Netw. (CAMAD)*, 2019, pp. 1–6.

[17] A. El-Amine, H. A. Haj Hassan, M. Iturralde, and L. Nuaymi, "Location-aware sleep strategy for energy-delay tradeoffs in 5G with reinforcement learning," in *Proc. IEEE 30th Annu. Int. Symp. Pers. Indoor Mobile Radio Commun. (PIMRC)*, 2019, pp. 1–6.

[18] Q. Zhang, X. Xu, J. Zhang, X. Tao, and C. Liu, "Dynamic load adjustments for small cells in heterogeneous ultra-dense networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, 2020, pp. 1–6.

[19] H. Park and Y. Lim, "Reinforcement learning for energy optimization with 5G communications in vehicular social networks," *Sensors*, vol. 20, no. 8, p. 2361, Apr. 2020.

[20] A. E. Amine, P. Dini, and L. Nuaymi, "Reinforcement learning for delay-constrained energy-aware small cells with multi-sleeping control," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, 2020, pp. 1–6.

[21] X. Chen, J. Wu, Y. Cai, H. Zhang, and T. Chen, "Energy-efficiency oriented traffic offloading in wireless networks: A brief survey and a learning approach for heterogeneous cellular networks," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 4, pp. 627–640, Apr. 2015.

[22] J. Liu, B. Krishnamachari, S. Zhou, and Z. Niu, "Deepnap: Data-driven base station sleeping operations through deep reinforcement learning," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4273–4282, Dec. 2018.

[23] J. Ye and Y.-J. A. Zhang, "DRAG: Deep reinforcement learning based base station activation in heterogeneous networks," *IEEE Trans. Mobile Comput.*, vol. 19, no. 9, pp. 2076–2087, Sep. 2020.

[24] C.-W. Huang and P.-C. Chen, "Joint demand forecasting and DQN-based control for energy-aware mobile traffic offloading," *IEEE Access*, vol. 8, pp. 66588–66597, 2020.

[25] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT press, 2018.

[26] L. Busoniu, R. Babuska, and B. D. Schutter, "A comprehensive survey of multiagent reinforcement learning," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 38, no. 2, pp. 156–172, Mar. 2008.

[27] G. Auer *et al.*, "How much energy is needed to run a wireless network?" *IEEE Wireless Commun.*, vol. 18, no. 5, pp. 40–49, Oct. 2011.

[28] H. Wu, X. Xu, Y. Sun, and A. Li, "Energy efficient base station on/off with user association under C/U split," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, 2017, pp. 1–6.

[29] P. H. V. Klaine, "Self-organization for 5G and beyond mobile networks using reinforcement learning," Ph.D. dissertation, Sch. Eng., Univ. Glasgow, Glasgow, U.K., 2019.

[30] S. Coniglio, F. Furini, and P. S. Segundo, "A new combinatorial branch-and-bound algorithm for the knapsack problem with conflicts," *Eur. J. Oper. Res.*, vol. 289, no. 2, pp. 435–455, Mar. 2021.

[31] N. Saxena, B. J. R. Sahu, and Y. S. Han, "Traffic-aware energy optimization in green LTE cellular systems," *IEEE Commun. Lett.*, vol. 18, no. 1, pp. 38–41, Jan. 2014.

[32] N. Saxena, A. Roy, and H. Kim, "Traffic-aware cloud RAN: A key for green 5G networks," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 4, pp. 1010–1021, Apr. 2016.

[33] J. Montoya-Bayardo and C. Borgelt, "Wide and deep reinforcement learning for grid-based action games," in *Proc. 11th Int. Conf. Agents Artif. Intell.*, Feb. 2019, pp. 50–59. [Online]. Available: http://www.icaart.org/?y=2019

Metin Ozturk received the B.Sc. degree in electrical and electronics engineering from Eskisehir Osmangazi University, Turkey, in 2013, the M.Sc. degree in electronics and communication engineering from Ankara Yildirim Beyazit University, Turkey, in 2016, and the Ph.D. degree from the Communications, Sensing, and Imaging Research Group, James Watt School of Engineering, University of Glasgow, U.K., in 2020. He is currently a Lecturer with Ankara Yildirim Beyazit University where he previously worked as a Research Assistant from 2013 to 2016. His research interests include intelligent networking for wireless communication networks, with a focus on energy efficiency, mobility management, and radio resource management in cellular networks.

Attai Ibrahim Abubakar received the B.Eng. degree (First Class) in electrical and electronics engineering from the Federal University of Agriculture, Makurdi, Nigeria, in 2011, and the M.Sc. degree (Distinction) in wireless communication systems from the University of Sheffield, U.K., in 2015. He is currently pursuing the Ph.D. degree with the James Watt School of Engineering, University of Glasgow, U.K. His research interests include energy performance optimization of 5G and beyond heterogeneous networks, radio resource management, self-organizing networks, and application of machine learning algorithms to wireless communications.

João Pedro Battistella Nadas was born in Curitiba, Brazil. He received the B.Sc. degree in electronics and telecommunications engineering and the master's degree (third rank) in telecommunications engineering from the Federal Technological University of Paranà, Brazil, in 2014 and 2016, respectively. He is currently pursuing the Ph.D. degree from the University of Glasgow. During both his degrees he has simultaneously worked in the industry in various positions. His research interests include ultra-reliable and low latency communications, HARQ, machine learning applied to communications, cyber-physical systems, and wireless sensor networks.

Rao Naveed Bin Rais received the B.E. degree in computer systems from the National University of Science and Technology, Pakistan, in 2002, and the M.S. and Ph.D. degrees in computer engineering with specialization in Networks and Distributed Systems from the University of Nice, Sophia Antipolis, France, in 2007 and 2011, respectively. He has experience of more than 15 years in teaching, research and industry Research and Development. He has authored several of publications in internationally recognized peer-reviewed journals and conferences. His research interests include communication network protocols and architectures, next-generation communication network, information-centric and software-defined networks, machine learning algorithms, cloud computing, network virtualization, and Internet naming and addressing issues.

Sajjad Hussain (Senior Member, IEEE) received the master's degree in wireless communications from Supelec, Gif-sur-Yvette, in 2006, and the Ph.D. degree in signal processing and communications from the University of Rennes 1, Rennes, France, in 2009. He is a Senior Lecturer (Associate Professor) in Electronics and Electrical Engineering with the University of Glasgow, U.K. He has served previously with the Electrical Engineering Department, Capital University of Science and Technology, Islamabad, Pakistan, as an Associate Professor. He has over 10 years of academic experience with roles as the Principal/Co-Investigator in several funded projects. He has authored/coauthored over 60 journal and conference publications along with one authored book, and one edited book. His research interests include 5G self-organizing networks, industrial wireless sensor networks, machine learning for wireless communications and technology enhanced student learning techniques. He has been an Associate Editor for IEEE ACCESS, since 2019. He is a Fellow of Higher Education Academy and a Senior Fellow of Recognising Excellence in Teaching.

Muhammad Ali Imran (Senior Member, IEEE) is a Professor of Wireless Communication Systems. He heads the Communications, Sensing and Imaging CSI Research Group, University of Glasgow. He is an Affiliate Professor with the University of Oklahoma, USA, and a Visiting Professor with the 5G Innovation Center, University of Surrey, U.K. He has over 20 years of combined academic and industry experience with several leading roles in multi-million pounds funded projects. He has filed 15 patents, has authored/coauthored over 400 journal and conference publications, edited of three books, and authored more than 20 book chapters, and has successfully supervised over 40 postgraduate students at Doctoral level. He has been a consultant to international projects and local companies in the area of self-organized networks. His research interests includes self organized networks, wireless networked control systems, and the wireless sensor systems. He has previously served as an Associate Editor of IEEE COMMUNICATIONS LETTERS, IEEE ACCESS and *IET Communications*. He is the Speciality Chief Editor for the section IoT and Sensor Networks of *Frontiers in Communications and Networks* and an Associate Editor for IEEE TRANSACTIONS ON COMMUNICATIONS and IEEE ACCESS. He is a Fellow of IET and a Senior Fellow of HEA.