ICE-CREAM: Multi-agent Fully Cooperative Decentralized Framework for Energy Efficiency in RAN Slicing

Hnin Pann Phyu, Diala Naboulsi, Razvan Stanica

ICE-CREAM: multI-agent fully CooperativE deCentRalizEd frAMework for Energy Efficiency in RAN Slicing

Hnin Pann Phyu, *Student Member, IEEE,* Diala Naboulsi, *Member, IEEE,* Razvan Stanica

***Abstract*—Network slicing is one of the major catalysts proposed to turn future telecommunication networks into versatile service platforms. Along with its benefits, network slicing is introducing new challenges in the development of sustainable network operations, as it entails a higher energy consumption compared to non-sliced networks. Using a sliced architecture, which includes guaranteeing the communication and computation requirements for each slice, is essential for operators to provide a satisfying user quality of service (QoS) in a multi-service network. At the same time, building sustainable mobile networks, with the least amount of resources used, is crucial today, for both economic and environmental reasons. As a result, mobile operators need to find a middle ground between these two objectives – a tough nut considering they are both antithetical and important. In this light, we investigate a joint slice activation/deactivation and user association problem, with the aim of minimizing energy consumption and maximizing the QoS. The proposed multI-agent fully CooperativE deCentRalizEd frAMework (ICE-CREAM) addresses the formulated joint problem, with agents acting at two different granularity levels. Not only all the agents can access the shared information with their direct neighbors, but also they are trained with one global reward, which is an ideal approach in multi-agent cooperative settings. We evaluate ICE-CREAM using a real-world dataset that captures the spatio-temporal consumption of three different mobile services in France. Experimental results demonstrate that the proposed solution provides more than 30% energy efficiency improvement compared to a configuration where all the slice instances are always active while maintaining the same level of QoS. From a broader perspective, our work explicitly shows the impact of prioritizing the energy over QoS, and vice versa.**

***Index Terms*—5G, Network Slicing, Energy Efficiency, QoS**

I. Introduction

The telecommunication industry accounts for approximately 2% of total global carbon emissions in the world [1]. By 2030, 8% of the projected global electricity demand is projected to come from the information and communications technology (ICT) sector as a whole, even in the best-case scenario [2]. Although 5G equipment is more energy efficient than in 4G [3], with the data traffic volume increasing tremendously along with 5G services, overall energy consumption will increase too. In fact, the energy consumption of a 5G base station is estimated as three times higher than that of a 4G network, when both are considered at a full load [4].

Beyond 5G and 6G networks are envisioned to serve a wide variety of services, with heterogeneous traffic, through network slicing [5]. This is done by forming, on one physical network, multiple virtual networks on a per-service basis, i.e., slices. That said, slice requirements need to be met, including performance isolation. Guaranteeing these requirements and the additional virtualization layer come with some overhead, which produces higher energy consumption when compared to non-sliced network deployments [6]. One of the key objectives in the field is to deliver the various types of services demanded by the users while aiming to reduce the associated CO_2 emissions. Indeed, energy efficiency is no longer an option but a necessity in beyond 5G and 6G networks.

Further delving into this topic, we observe that today the largest part of energy consumption comes from the radio access network (RAN), which accounts for approximately 70% of the overall network energy utilization [7]. To deal with this issue, several research works consider base station sleep schemes to further optimize the energy consumption in 5G networks [8]–[10]. While such techniques show feasible and effective results, they are more challenging to be applied directly in the case of multi-service network slicing environments. That is mainly because slice instances can exhibit quite different temporal traffic demand patterns. Completely shutting down or putting the entire base station into sleep mode could thus notoriously impact the quality of service (QoS) of users in specific slice instances.

This motivates us to introduce a new approach, in which slice instances are dynamically activated and deactivated, according to their traffic patterns, thereby enhancing the overall base station energy efficiency. Technically, a slice is a set of network functions, applied to specific users or to a specific type of traffic. Therefore, deactivating some slice instances to reduce energy consumption implies shutting down some of these functions, and it can potentially degrade the QoS of users (i.e., lower user satisfaction and higher blocking probability). On the other hand, activating all slices all the time, in order to reach the highest possible level of QoS, significantly increases energy consumption. Accordingly, the energy minimization objective shall be coupled with a QoS maximization objective [11].

To manage the trade-off between the two objectives, op-

H. Pann Phyu and D.Naboulsi are with the Département de Génie Logiciel et des Technologies de l'information, École de Technologie Supérieure, Université du Québec, Montreal, QC H3C 1K3, Canada (e-mail: hnin.pann-phyu.1@ens.etsmtl.ca, diala.naboulsi@etsmtl.ca).

Razvan Stanica is with INSA Lyon, Inria, CITI, Villeurbanne, France (e-mail: razvan.stanica@insa-lyon.fr).

This work was supported by the École de Technologie Supérieure (ÉTS), the National Natural Sciences and Engineering Research Council of Canada (NSERC) through research grant RGPIN-2020-06050. This work was partially funded by the CHIST-ERA ECOMOME project and the ANR COCO5G (ANR-22-CE25-0016) project.

erators may consider deactivating some slice instances and redirecting their users either to another slice on the same base station (where some required network functions might not be available) or to the same slice on a neighboring base station (most likely with a lower communication channel quality). In the first case, a possible approach implies using a slice instance with only the bare minimum resources and network function requirements, which we denote as an EcoSlice. Specifically, the EcoSlice is designed to be up and running $24/7$ (or as long as the base station is switched on) to provide network coverage and a bare minimum service to certain users. However, assigning too many users to the EcoSlice on the same base station is not sustainable in terms of QoS. In the second case above, a sudden increase in traffic load on the neighboring base stations can be observed, thereby impacting the QoS of their already present users [12]. To mitigate these challenges and optimize the overall network performance, it is essential to intelligently coordinate not only the activation and deactivation of slices (a network-level problem) but also the user association across different base stations (a user-level problem).

Considering these two dimensions, one can hypothesize a hierarchical RAN slicing problem formulation [13], considering both the network level and the user level, using optimization methods or solving it with heuristics. Nevertheless, these approaches do not fully align with a zero-touch network management paradigm, primarily due to their limitations in rapid changes in the RAN environment. To address this issue, machine learning (ML) techniques, particularly reinforcement learning (RL) algorithms, have demonstrated unprecedented performance in solving RAN slicing problems that were previously too challenging [14]. This opens the door to full automation in the management of RAN slicing operations [15].

From the solution design perspective, centralized settings are usually considered in mobile networks, where an operator has total control over the network equipment. However, in sliced architectures, this paradigm shifts towards multiple players, which can be in charge of different slices and parts of the network [14]. Therefore, coordination through a centralized controller and data training might not be possible, for business and technical reasons [16]. In this regard, decentralized multi-agent approaches with coordination among neighboring base stations exhibit several benefits over centralized approaches in terms of computation cost and robustness, as well as better scalability and higher learning speed compared to single-agent approaches [17].

All things considered, we design a multi-agents fully collaborative RL-driven framework that coordinates not only the slice activation/deactivation but also the user association decision. In this respect, our agents collaborate to achieve the same objective: reduce energy use and enhance user satisfaction. Our approach is guided by the idea that, if a slice is to be activated on a base station, its available resources can be shared with other nearby base stations that require them, hence enabling a more efficient utilization. The unique aspects of our work can be summarized as follows:

- We formulate a joint slice activation/deactivation and user association problem, to minimize base station energy consumption and maximize user QoS. This combined problem involves two levels of time granularity: large timescale, in the order of minutes, and small timescale, in the order of seconds. Our work is the first of its kind to address both slice activation/deactivation and user association in sliced mobile networks.
- We rely on the decentralized partially observable Markov decision process (Dec-POMDP) modeling to deal with the partial observability of the agents. Embracing the uniqueness of each state (i.e. states are independent of each other), we formulate the equivalent state-aware multi-armed bandit (MAB) problem based on this Dec-POMDP model.
- To solve the defined problem in a scenario with multiple base stations and slices, we design a multI-agent fully CooperativE deCentRalizEd frAMework (ICE-CREAM), in which each agent shares the information with its corresponding nearby agents, and all of them are trained with one global reward encouraging cooperation. Our code [1] is openly available for the community.
- We evaluate the performance of the proposed approach using a real-world traffic and user dataset. We assess the quality of the obtained solutions, as well as their computational cost, based on large-scale network scenarios. In view of the dynamic and heterogeneous demand of mobile users, the proposed solution leads to better decisions when compared to different benchmarks.

The rest of the paper is organized as follows. Section II discusses the related work in the field of energy efficiency in network slicing. Then, the system model and the utility models are laid out in Section III. Section IV includes the formulation of the problem, along with its reformulation in terms of state-aware MAB. In Section V, we present the detailed design of our proposed system architecture. We articulate our results in Section VI and conclude the paper in Section VII.

II. Related Work

Numerous research works have proposed solutions (both ML-based and non-ML techniques) in the context of energy efficiency in mobile networks. These approaches can be broadly categorized into network-centric, user-centric, and hybrid approaches. This section reviews some of the notable research works in each of these categories.

i) Network-centric: In network-centric solutions, the focus is on optimizing and managing network resources from the network infrastructure point of view. With the aim of enabling energy efficiency, several approaches consider optimizing the allocation of network slice resources (e.g., computing needs, communication bandwidth, and transmission power) in the different mobile network domains (i.e., RAN, core network, and end-to-end network).

At the RAN slicing level, [18] optimizes the energy consumption and computation cost in a slicing-based Cloud-RAN (C-RAN) setting, using a twin-delayed double-Q soft Actor-Critic (TDSAC) approach. Their agent performs the up/down scaling of computing and beamforming power resources. Their work outperforms other baseline RL models in terms of overall

[1] https://github.com/HninPannPhyu/icecream_MARL.git

network energy and computing cost. Similarly, [6] designs a slice energy consumption model based on the C-RAN architecture. An optimization problem is solved per slice, with the objective of minimizing the overall network energy cost, jointly considering communication and computation resources. This approach improves the energy efficiency over a baseline focused only on radio resources.

Furthermore, in [19], the authors combine deep learning (DL) and RL on a distributed framework to efficiently allocate radio and transmission power resources. More precisely, they use stacked and bidirectional long short-term memory (SBiLSTM) neural networks to predict the per slice resources demand on a large timescale and rely on asynchronous advantage actor-critic (A3C) to allocate resources to users on a small timescale. Their framework achieves higher energy efficiency than counterparts using a static power allocation model.

Focusing on core network (CN) slicing, the authors in [20] formulate a security-aware network slicing optimization problem to enhance the energy efficiency of CN nodes. They limit themselves to static resource allocation. Their proposed solution provides more power savings than a greedy approach.

ii) User-centric: In user-centric solutions, the primary focus is on catering to the individual needs and requirements of users within the network. In this light, several studies explored the problems of user association and task offloading, for the sake of energy efficiency while ensuring dynamic user demands.

In [21], the authors propose a device association scheme for RAN slicing, based on hybrid federated deep reinforcement learning (HDRL) to improve network throughput and reduce hand-off cost. The scheme exploits two levels of aggregation for the device association problem and calculates Shapley values to evaluate the importance of different global access features. Moreover, the HDRL framework increases the training sample efficiency, and the number of training on each device can be reduced, thus improving energy efficiency in each device. The proposed HDRL framework is shown to outperform existing methods in terms of network throughput, hand-off cost reduction, and communication efficiency.

User task offloading has recently attracted much attention. Exploiting the benefits of edge computing, the authors in [22] propose to optimize the decision of offloading users' tasks. Specifically, they attempt to minimize the processing time of the users' tasks associated with different slices, thereby ensuring the energy efficiency of users. To this end, they rely on a heuristic approach to provide near-optimal solutions. Similarly, the authors in [23] investigate energy-efficient joint task offloading and scheduling of enhanced mobile broadband (eMBB) and ultra-reliable low latency communications (URLLC) users to minimize energy consumption and maximize the achievable data rate of the eMBB users, subject to various constraints. The authors use a block coordinate descent (BCD) algorithm to solve the problem.

iii) Hybrid: Hybrid approaches that combine the benefits of both methods (i.e. network-centric and user-centric) can be used to achieve the right balance between network efficiency and user satisfaction. In this vein, our prior work [24] investigated the slice activation problem, with possible user offloading to an EcoSlice. Specifically, multi-armed bandit approaches were explored to determine the near-optimal slice activation/deactivation decision while respecting the QoS of individual users. Considering this obvious trade-off, it is sensible to couple user QoS maximization and energy consumption minimization. Therefore, while considering end-to-end network slices, the authors in [25] aim to maximize energy efficiency while respecting service level agreement (SLA) constraints. To this end, they rely on the statistical federated learning (stFL) framework. Their federated local agents coordinate and predict per-slice network metrics, without transferring datasets to a central unit, and largely outperforming other federated learning and centralized solutions. Also, the study in [26] tackled the problem of switching off small cell base stations by considering the four distinct power-saving modes. They developed a strategic sleeping scheme tailored for both static and dynamic traffic models. These proposed models further improved the energy efficiency compared to the baseline random sleeping policy.

Overall, the large majority of previous works have focused either on the global network energy efficiency or satisfying user QoS in the context of RAN slicing. Specifically, network-centric approaches (as the works in [6], [18]–[20]) ignore the variety of user requirements in each application scenario. Conversely, the user-centric approaches (as the works in [21]–[23]) tend to overlook the overall network performance, as the focus is primarily on catering to specific user demands, potentially leading to sub-optimal network efficiency. As for hybrid approaches, our preliminary work [24] introduced an approach that operates over individual base stations, without accounting for the potential impact on the neighboring base stations. On the other hand, although it presents a joint objective, the user side is not defined clearly in [25]. The work in [26] is close to our work in a sense that it optimizes small cell on/off switching while balancing energy efficiency and QoS. However their approach does not account for RAN slicing and depends on predefined parameters, which are less effective in today's dynamic and heterogeneous network environments. To the best of our knowledge, no contemporary research has jointly addressed the problem of minimizing energy consumption and maximizing user QoS in a sliced architecture, while enabling coordination among neighboring base stations.

III. System Model

In this section, we present our system model. First, we introduce the two timescales over which we derive slice activation/deactivation decisions and user association decisions, respectively. We then cover the network model, as well as the energy consumption model and the user QoS model, needed as part of our defined problem.

A. Two-Timescale Granularity

We consider a time-slotted system with operations over two different timescales: a large timescale and a small timescale. We define τ as the large timescale interval, or the slice activation/deactivation interval (SADI), where we consider that a slice is active or inactive continuously throughout that

interval. In particular, slice activation/deactivation decisions are made at the end of every τ for the upcoming $\tau + 1$. We also use Δt to refer to the duration of a SADI.

Each SADI τ consists of multiple small timescale intervals t, over which user associations are determined. In particular, given the knowledge of slice activation/deactivation decisions from the large timescale interval τ, each user is navigated, at each small timescale interval t, to a specific slice instance, on a specific base station, to be served. Simply put, base station configuration related to slice instances activation or deactivation is updated at every τ, while user association decisions occur at every t. Figure 1 illustrates the two timescale intervals.

We define T and χ as a set of large timescale intervals and a set of small timescale intervals, respectively, such that $\tau \in T$ and $t \in \chi$. With these notations, χ_τ represents a set of small timescale intervals t within τ period. The values of T, χ, τ, t and Δt can be defined based on the mobile operator policies.

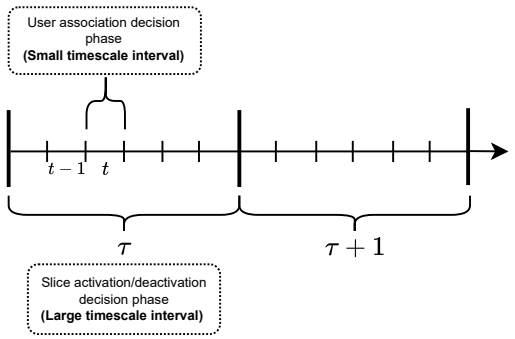

Fig. 1: Illustration of two timescale intervals.

The reasoning behind this model is that the two mechanisms (user association and slice activation) have different dynamics. While user association can change more often, by following the handover decision process, we expect slice (de)activation to happen less frequently, as the overall traffic demand varies.

B. Network Model

We consider a standard mobile network model consisting of multiple base stations, and each base station is associated with different slice instances and has a different coverage based on its location (i.e. rural or urban areas).

We denote a set of deployed base stations in a geographical area of interest as $\mathcal{B}$. Some of these base stations are interconnected with high-speed back-haul links. In this work, we consider four different types of slices: three of them, namely eMBB, URLLC, and Massive Machine-Type Communication (mMTC), provide services with different QoS levels, and they can be activated/deactivated as needed, while the EcoSlice is always up and running. These four slice instances are associated with each base station b. Therefore, we define $\mathcal{I}_b$ as a set of slice instances (including an EcoSlice i_{b_e}) attached to base station $b \in \mathcal{B}$. We denote the overall set of deployed slice instances in the network as $\mathcal{I}$. The bandwidth allocated to each slice i_b is BW_{i_b}. Accordingly, the total bandwidth BW_b of base station b can be calculated as $BW_b = \sum_{i_b \in \mathcal{I}_b} BW_{i_b}$. Moreover, we assume each base station b has a set of possible configurations $\mathcal{K}_b$. Each configuration $k \in \mathcal{K}_b$ implies activation/deactivation decisions for slices. In detail, $k = \left\{c^\tau_{i_b} | i_b \in \mathcal{I}_b\right\}$. Here, $c^\tau_{i_b} = 1$ if slice instance i_b is active at b during τ, and 0 otherwise. As explained, for the EcoSlice, $c^\tau_{i_{b_e}} = 1$ for any τ.

We denote $\mathcal{U}^t_{i_b}$ as a set of users that is served by the slice i_b, distributed under the coverage of base station b at time t. We also define $\mathcal{N}_b$ as a set of neighboring base stations capable of hosting users served by b. This can expressed as:

$$\mathcal{N}_b = \{b' \in \mathcal{B} |\ d_{b,b'} \leq d_{thr}, b \neq b'\} \tag{1}$$

where $d_{b,b'}$ is the distance between base stations b and b', and d_{thr} is a predefined distance threshold for two base stations to be considered as neighbors. We note that this simple model should be better refined in real operational conditions, where a simple distance threshold is not sufficient to define neighboring relations between base stations.

The service requirement of user u at time t can be described by (l^t_u, δ^t_u), where l^t_u is the traffic load in bits, δ^t_u is the maximum tolerable delay in milliseconds. In the optimal slice instance activation scheme, underutilized slice instances are switched off to save energy when certain conditions are met. As such, the user-perceived delay is prone to deteriorate. In this light, we deliberately consider user delay as an indicator of the QoS, but any other metric could be easily integrated instead. To demonstrate this, in the evaluation section below, we also consider a scenario where different QoS metrics are used for the different slice instances.

Given the bandwidth BW_{i_b} of each slice, the achievable data rate $r^t_{i_b,u}$ for user u, served by slice i_b at time t, is computed as:

$$r^t_{i_b,u} = \frac{BW_{i_b}}{|\mathcal{U}^t_{i_b}|} log_2(1 + SINR^t_{b,u}) \tag{2}$$

with $SINR^t_{b,u}$ being the signal to interference and noise ratio for user u of base station b at time t. Here, $r^t_{i_b,u}$ accounts for both the received signal strength of user u associated with slice i_b of base station b and the load level of slice i_b [27].

We compute the SINR as below:

$$SINR^t_{b,u} = \frac{P^t_{b,u} |g^t_{b,u}|^2}{\sum_{b' \in \mathcal{N}_b} P^t_{b',u} |g^t_{b',u}|^2 + N_0} \tag{3}$$

where $P^t_{b,u}$ represents the transmit power allocated to user u at base station b at time t and N_0 is an additive white Gaussian noise (AWGN). The channel gain $g^t_{b,u}$ is a function of the path loss between the base station b and the user u. The path loss in our model is an empirical model proposed by the International Telecommunications Union (ITU) for an urban environment, accounting in an average way for shadowing and other radio propagation phenomena. It is computed as $PL(d_{u,b}) = 128.1 + 37.6 log(d_{u,b})$, and it is based on the distance (in km) between the user u and the base station b [28]. With this, we obtain the channel gain as $g^t_{b,u} = 10^{-PL(d_{u,b})/10}$ [29].

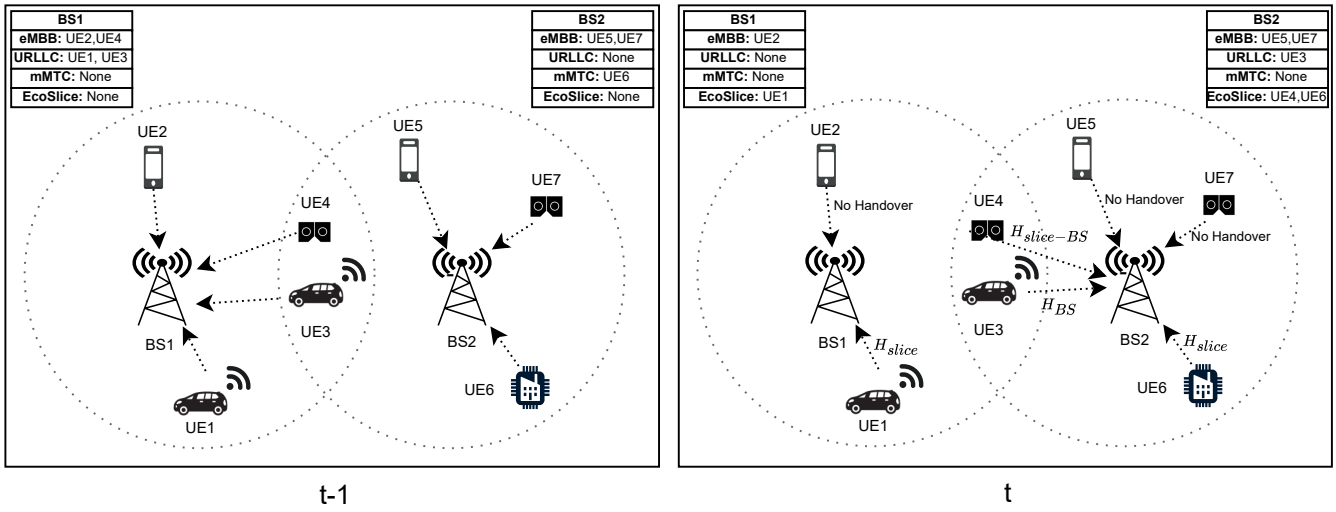

Fig. 2: Illustration of user handover from time **t-1** to **t**. At time **t-1**, UE1 (URLLC), UE2 (eMBB), UE3 (URLLC) and UE4 (eMBB) are linked to their corresponding slice instances associated with BS1, while UE5 (eMBB), UE6 (mMTC) and UE7 (eMBB) are connected to their respective slice instances associated with BS2. Each user can incur a handover cost at time **t** based on their new associations. For example, $H_{slice-BS}$ cost is considered for UE4 as it changes both BS and slice, while no handover cost is incurred for UE5 and UE7 at time **t**, and so forth.

Hence, the transmission delay at base station b for user u, served by slice i_b at t, can be computed as:

$$\delta^t_{i_b,u} = \frac{l^t_u}{r^t_{i_b,u}} \tag{4}$$

C. Energy Consumption Model

We define the function $f(.)$ to refer to the overall energy consumption of a base station. This energy model can be further extended based on specific operator resource management policies and energy-saving features activated at the RAN level. In our model, $f(.)$ is composed of the sum of energy consumption resulting from its individual deployed slice instances $E^\tau_{i_b}$ and a static energy value E^{static}_b, related to functions such as cooling and circuit power:

$$f(c^\tau_{i_b}, \mathcal{I}_b) = \sum_{i_b \in \mathcal{I}_b} c^\tau_{i_b} \cdot E^\tau_{i_b} + E^{static}_b \tag{5}$$

with

$$E^\tau_{i_b} = \sum_{t \in \chi_\tau} E^t_{i_b} \tag{6}$$

where $E^t_{i_b}$ represents the energy consumption of slice i_b during time t and it is computed as follows:

$$E^t_{i_b} = \underbrace{\psi_{i_b} \cdot \rho^t_{i_b} \cdot P^{load}_b \cdot t}_{load-dependent} + \underbrace{\psi_{i_b} \cdot P^{fixed}_b \cdot t}_{load-independent} \tag{7}$$

As indicated in the Equation (7) above, $E^t_{i_b}$ encompasses a load-dependent power consumption component (the first term of the sum in Equation (7)), and a load-independent power consumption component (the second term in Equation (7)). Both these terms are regulated by ψ_{i_b}, a power consumption impact factor of slice instance i_b.

Regarding the load-dependent power consumption component in Equation (7), this is obtained using a static power transmission value per quantity of information, denoted as P^{load}_b, and $\rho^t_{i_b}$, the traffic load ratio of the associated slice instance i_b during t. This parameter $\rho^t_{i_b}$ is obtained by dividing the traffic load over the slice $\mathbf{l}^t_{i_b}$ by the maximum traffic load $\mathcal{L}_{max,b}$ registered by base station b, as below:

$$\rho^t_{i_b} = \frac{\mathbf{l}^t_{i_b}}{\mathcal{L}_{max,b}} \tag{8}$$

where we consider $\mathcal{L}_{max,b}$ to be a predefined parameter (i.e., the base station capacity) and known by an operator.

We note here that, in practice, not all slice instances are designed equally [30]. Their required network functions and signaling traffic are different [31], and they result in very different energy consumption patterns.

For instance, a URLLC service consumes higher energy, because it requires specific network functions at all the protocol levels to offer high reliability and low latency [32], which implies that any sleep time is removed in the network. In short, the more stringent the latency requirements of the service, the higher its consumed energy [33]. On the other hand, an mMTC service requires high connection density to connect tens of thousands of devices, which in turn leads to more energy consumption in the transmission downlink [34], which needs to coordinate all these devices. Finally, the eMBB service demands high data capacity, meaning that it needs more allotted resources (i.e. resource blocks for communication or processing power for computing) [31]. However, regardless of their requirements for energy, both mMTC and eMBB have flexibility in terms of latency, and it is sensible to assume that they present a lower ψ_{i_b} value than URLLC. Therefore, it is fair to assume that, even under the same amount of traffic load, the energy consumption of each service is different [35], which explains the use of the parameter ψ_{i_b} in Equation (7).

We also consider a specific type of slice, denoted as EcoSlice. The EcoSlice is deployed with relatively low energy consumption, as only basic network functions are active and long sleep timers are used. The EcoSlice is meant to accommodate traffic with very low QoS requirements, where

energy consumption can be prioritized without too much inconvenience. However, under a low traffic load, the EcoSlice can also satisfy the QoS requirements of more complex types of traffic, but it can not scale up with traffic load, because the required network functions are not activated.

As for the load-independent component in Equation (7), it is worth stressing that a slice instance consumes some power even at zero traffic load, in order to instantiate and run its corresponding network functions. Thereupon, P_b^{fixed} is independent of the traffic load, but still specific to the slice type, hence still regulated by ψ_{i_b}.

D. User QoS Model

As explained, our objective is coupled with ensuring the user QoS level. Accordingly, we define η_b^τ as the overall QoS of a base station:

$$\eta_b^\tau = \frac{\sum_{i_b \in \mathcal{I}_b} \eta_{i_b}^\tau}{|\mathcal{I}_b|} \tag{9}$$

where $\eta_{i_b}^\tau$ is an average QoS metric of slice instance i_b at τ. It is expressed as:

$$\eta_{i_b}^\tau = \frac{\sum_{t \in \chi_\tau} \eta_{i_b}^t}{|\chi_\tau|} \tag{10}$$

Again, $\eta_{i_b}^t$ during t can be computed by taking into account the QoS of all users associated with slice instance i_b, such as:

$$\eta_{i_b}^t = \frac{\sum_{u \in \mathcal{U}_{i_b}^t} \eta_u^t}{|\mathcal{U}_{i_b}^t|} \tag{11}$$

where each user QoS is defined in terms of delay η_u^t as:

$$\eta_u^t = \frac{1}{1 + e^{-\theta(\delta_u^t - \delta_{i_b,u}^t)}} \tag{12}$$

with θ defined as a constant deciding the steepness of the user satisfaction curve.

Moreover, in Equation (12), a handover penalty cost is integrated, in order to model the impact that changing the user association (an operation known as handover) temporarily has on the user QoS. This impact comes from the fact that the handover mechanism implies supplementary control traffic and buffering of incoming packets at the level of the CN.

TABLE I: Handover and QoS relationship for user u.

$(x_{u,b'}^{t-1}, b')$**vs**$(x_{u,b}^t, b)$	η_u^t	**Handover**
$b'=b$, $x_{u,b'}^{t-1}$ =1, $x_{u,b}^t$ =1	$\frac{1}{1+e^{-\theta*(\delta_u^t - \delta_{i_b,u}^t)}}$	No handover
$b'=b$, $x_{u,b'}^{t-1}$ =1, $x_{u,b}^t$ =0	$\frac{1}{1+e^{-\theta*(\delta_u^t - \delta_{i_b,u}^t)}} - H_{slice}$	Slice only
$b' \neq b$, $x_{u,b'}^{t-1}$ =1 ,$x_{u,b}^t$ =1	$\frac{1}{1+e^{-\theta*(\delta_u^t - \delta_{i_b,u}^t)}} - H_{BS}$	Base station only
$b' \neq b$, $x_{u,b'}^{t-1}$ =1, $x_{u,b}^t$ =0	$\frac{1}{1+e^{-\theta*(\delta_u^t - \delta_{i_b,u}^t)}} - H_{BS-slice}$	Both base station and slice

As it can be seen in Table I, we define $x_{u,b}^t$ to represent the user slice association decision at t. More precisely, $x_{u,b}^t = 1$ if user u, who is connected to base station b, is associated with his requested slice. $x_{u,b}^t = 0$ if user u, who is connected to base station b, is associated with the EcoSlice. At each time interval t, a user association decision is taken, meaning that u could change his slice instance or even base station. Acknowledging the potential impact of such handovers on the user QoS, we consider three types of scenarios, depicted in Figure 2: *i)* switching slice instances only, in which handover cost H_{slice} is incurred, *ii)* switching base stations only, in which handover cost H_{BS} is applied and *iii)* switching base station and slice instances, where a handover cost denoted as $H_{BS-slice}$ is applied. Each of these scenarios impacts the user QoS differently, and we assume that $H_{slice} < H_{BS} < H_{BS-slice}$ [36]. These handover costs are applied in our model as a QoS penalty whenever the user association is changed between two consecutive time slots (similarly to [21], [37]).

IV. Hierarchical Multi-Agents Problem Formulation

In this work, as an extension of our preliminary work in [24], we investigate the multi-agent joint slice activation/deactivation and user association problems, where the overall policy can be denoted as $\pi = \{\pi_\mathcal{B}, \pi_\mathcal{I}\}$. Specifically, slice activation/deactivation decision: $\pi_\mathcal{B} = \bigcup_{b \in \mathcal{B}} \pi_b$ is made at the large timescale τ, with π_b being the policy of base station b. Based on the slice activation/deactivation decisions, the user association decision $\pi_\mathcal{I} = \bigcup_{b \in \mathcal{B}} \bigcup_{i_b \in \mathcal{I}_b} \pi_{i_b}$ is determined at each small timescale t, where π_{i_b} is the policy related to slice instance i_b. Since the environment is partially observable, traditional Markov decision processes (MDPs) are not well-suited to this specific work. Indeed, due to having multiple independent base stations and slice instances in the system, it becomes necessary to employ a multi-agent setting. With this, we formulate the problem as a Dec-POMDP model and redesign it after that as a contextual MAB problem (also known as context/state-aware MAB), inspired by similar approaches in related problems [38].

A. Decentralized Partially Observable Markov Decision Process (Dec-POMDP)

Our work applies a cooperative multi-agent approach, where agents take actions based on local observations and then receive a shared global reward. Ideally, this setting is formalized as a decentralized POMDP. A Dec-POMDP, in fact, extends the single-agent POMDP by incorporating the coordination of actions and observations among multiple agents [39].

A Dec-POMDP is defined by a tuple $(\mathcal{N}, \mathcal{S}, \mathcal{O}, \mathcal{A}, P, \mathcal{R})$, where $\mathcal{N}$ is the total number of agents, $\mathcal{S}$ represents the (finite) set of network states of all the agents and $\mathcal{O} = \mathcal{O}_1 \times \mathcal{O}_2 \times \cdots \times \mathcal{O}_\mathcal{N}$ is the set of joint observations by agents. In particular, at each time step, $s \in \mathcal{S}$ is the global state of the network environment while the observation of each agent z is $o_z \in \mathcal{O}_z$, which is a subset of s, i.e. $o_z \subset s$. Simplifying matters, each agent can solely observe their own local observation o_z, i.e. $s = o_1 \cup o_2 \cup \cdots \cup o_\mathcal{N}$. We denote $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2 \times \cdots \times \mathcal{A}_\mathcal{N}$ as the set of joint actions and $\mathcal{A}_z$ as the set of actions available to agent z, which can be different for each agent. Here, $a_z \in \mathcal{A}_z$ represents the action of agent z at any point, and a_{-z} denotes the action of the remaining agents except agent z, i.e. $a = a_z \cup a_{-z}, a \in \mathcal{A}$.

Element $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ captures the stochastic transition probability function to transition to s' from current state s based on action a, with $\sum_{s' \in \mathcal{S}} P(s'|s, a) = 1$ for all

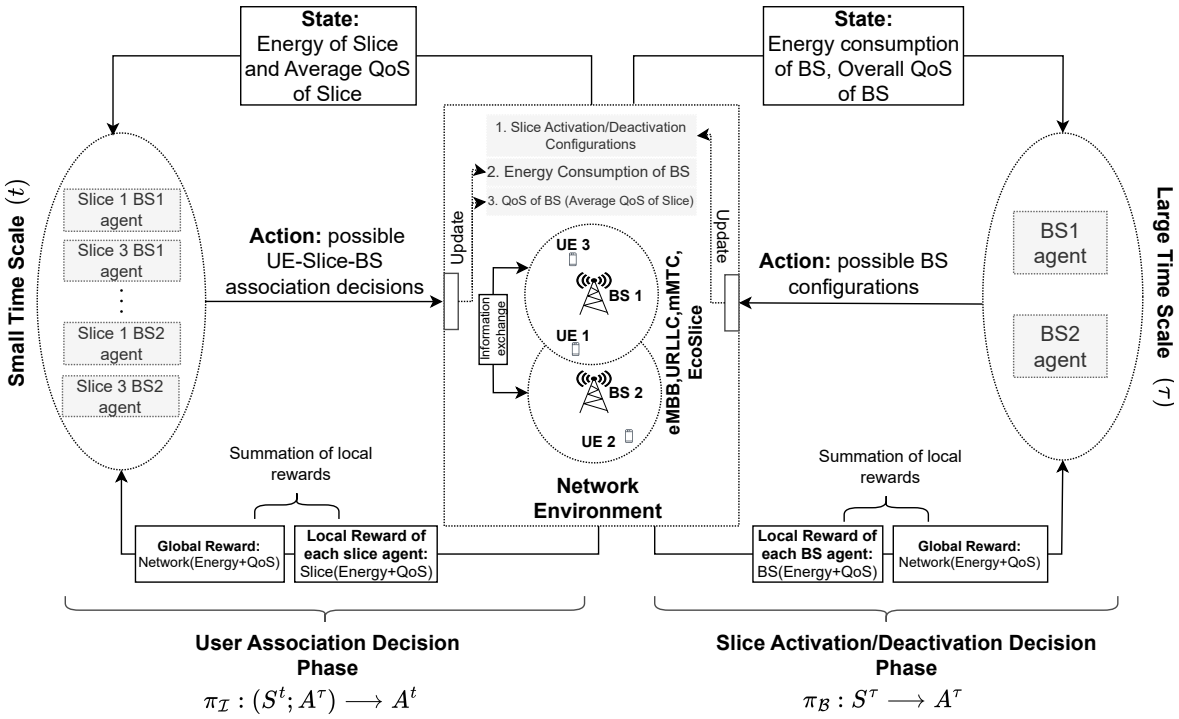

Fig. 3: ICE-CREAM architecture for energy efficiency in RAN slicing. For simplicity, state, action, and reward are represented for only one agent.

$s \in \mathcal{S}$ and $a \in \mathcal{A}$. P is unknown to an RL agent. However, it occurs that, if an agent knows the current state and the reward obtained in each iteration, it can still converge to the optimal solutions through RL approaches [40]. Similarly, the last term $\mathcal{R} = \mathcal{R}_1 \times \mathcal{R}_2 \times \cdots \times \mathcal{R}_\mathcal{N}$ is the set of joint reward functions. At each time step, for agent z, $\mathcal{R}_z$ gives the immediate global reward r_z, i.e. $r_z = \mathcal{R}_z(s, a_z, a_{-z}, s')$ [41].

Denoted by $\pi_z : \mathcal{O}_z \rightarrow \mathcal{A}_z$, the policy of agent z specifies how to select the actions based on the given context. At the base station level, the policy of the base station agent can be described as $\pi_b : \mathcal{O}_b \rightarrow \mathcal{A}_b$. Ideally, π_b is iteratively updated at every τ during the learning process, and we thus express the immediate policy of base station agent at τ by π_b^τ. Accordingly, the overall policy at the level of all base stations is $\pi_\mathcal{B} : \mathcal{S}^\tau \rightarrow \mathcal{A}^\tau$. Similarly, at the level of slice instances, the policy of a slice agent can be described as $\pi_{i_b} : \mathcal{O}_{i_b} \rightarrow \mathcal{A}_{i_b}$. While $\pi_{i_b}^t$ is the immediate policy of slice agent at t, the corresponding overall policy is denoted as $\pi_\mathcal{I} : (\mathcal{S}^t; \mathcal{A}^\tau) \rightarrow \mathcal{A}^t$.

B. State-aware Multi-armed Bandit

Motivated by the observations that the problem above can be seen as a simplified instance of an MDP, we formulate our joint slice instance activation/deactivation and user association problem as a state-aware MAB, an equivalent to the previously discussed Dec-POMDP. Our motivation for this reformulation arises from two key points that we observe in our prior modeling of the problem based on Markovian properties: *i)* the transition probability of the MDP can be simplified, such as $P(s'|s, a) \equiv P(s'|a)$, where states are independent of each other, and *ii)* unlike typical MDP [18], our reward function depends only on the current state and action, but not on the successor state. With this, our previous reward function could be simplified as $\mathcal{R}_z(s, a_z, a_{-z}, s') \equiv \mathcal{R}_z(s, a_z, a_{-z})$ for agent z. In the following, we intentionally omit a subscript z in the reward representation for the sake of simplification. We note that this problem reformulation as a MAB is not always possible, especially in problems where the reward is related to some kind of prediction of the future state. However, in our system, depicted in Figure 3, we do not rely on such predictions, and prefer to react to changes in our dynamic environment by using the RL process itself.

Formally, a multi-agent partially observable state-aware MAB (POMAB) is a tuple $(\mathcal{N}, \mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{R})$. Similar to our Dec-POMDP formulation, $\mathcal{N}$ is the total number of agents, $\mathcal{S}$ is the set of global states, $\mathcal{O}$ is the set of observations of all the agents, $\mathcal{A}$ is the set of joint actions and $\mathcal{R}$ is the set of joint reward functions. In fact, the detailed mathematical representations of states, observations and actions remain consistent with those presented in Section IV-A.

The only main difference lies in the reward function. In a classical MAB setting, each possible action is linked to an underlying reward distribution, denoted as $\hat{\mathcal{R}}(r|a)$. Since we consider the state-aware MAB (i.e., involving multiple states), our reward distribution is non-stationary, and changes based on the state s (also called context in the following). With this, for a given state, the distribution of reward for available actions can be defined as $\hat{\mathcal{R}}(r|s, a)$, i.e. $a \in \mathcal{A}, s \in \mathcal{S}$.

Subsequently, we elaborate in greater detail on the definition of the observation, action, and reward for the joint slice activation/deactivation problem (i.e. large timescale problem) and user association problem (i.e. small timescale problem).

1) Slice Activation/Deactivation Problem: It is worth noting that slice reconfiguration unfolds gradually over several minutes or even longer. In this context, the decision to activate or deactivate slices takes place on a much larger timescale, denoted as τ. This means that the constraint of a synchronized update coming from base station agents is very weak, as these updates can be transmitted over the duration of a SADI. The observation received by the base station agent at τ consists of the energy consumption and QoS of the base station in the previous SADI τ, which can be expressed as:

$$o_b^\tau = \left\{ f(c_{i_b}^{\tau-1}, \mathcal{I}_b), \eta_b^{\tau-1} \right\} \tag{13}$$

While the full state is not directly accessible to any individual agent, and it does not play a role in their decision-making, it can still be inferred from the collective observations made by all agents, as discussed in Section IV-A:

$$s^\tau = o_1^\tau \cup o_2^\tau \cup \cdots \cup o_{|\mathcal{B}|}^\tau \tag{14}$$

Each action a_b^τ is a configuration k, as defined in Section III-B, and $\mathcal{A}_b$ is the set of possible configurations $\mathcal{A}_b = \mathcal{K}_b$. It can be expressed as:

$$a_b^\tau = \left\{ c_{i_b}^\tau |\ i_b \in \mathcal{I}_b \right\}_{k \in \mathcal{K}_b} \tag{15}$$

The reward function is critical for the RL agent to be able to learn the optimal policy, and it usually boils down to the main objectives of the problem. The base station agent aims to maximize the energy efficiency and yet respect the QoS of users collectively. With this, the base station agent's immediate global reward function is formulated as below:

$$r_b^{global}(\tau) = \sum_{b \in \mathcal{B}} r_b^{local}(o_b^\tau, a_b^\tau) \tag{16}$$

$$r_b^{local}(o_b^\tau, a_b^\tau) = \frac{1}{f(c_{i_b}^\tau, \mathcal{I}_b)} + \beta \cdot \eta_b^\tau \tag{17}$$

where β is the QoS impacting factor. A larger β gives more weight to the QoS. $r_b^{global}(.)$ is the global reward, which is the summation of all the local rewards $r_b^{local}(.)$ of the base station agents. In fact, $r_b^{global}(.)$ fosters a spirit of cooperation among the base station agents. The one and only task of the base station agents is to find a policy π_b that maximizes the expected cumulative global reward. Mathematically, this can be expressed as:

$$\underset{\pi_b}{max}\ \mathbb{E}\left[\sum_{\tau \in T} r_b^{global}(\tau) \right] \tag{18}$$

2) User Association Problem: Unlike the slice activation/deactivation problem, user association happens on a smaller timescale, namely at every t. Based on the slice activation/deactivation status given at the large timescale τ and its own local observation, the slice agent makes the user association decision accordingly. The observation for the agent at t involves the energy consumption and QoS of slice instance i_b during the previous interval $t-1$, which can be expressed as below:

$$o_{i_b}^t = \left\{ E_{i_b}^{t-1}, \eta_{i_b}^{t-1} \right\} \tag{19}$$

Similarly to Equation (14), the global state for the user association problem can be expressed as:

$$s^t = o_1^t \cup o_2^t \cup \cdots \cup o_{|\mathcal{I}|}^t \tag{20}$$

The action of a slice agent at t is represented by the set of user association decisions taking into consideration the neighboring base stations of b that can potentially serve the users. It is expressed as:

$$a_{i_b}^t = \left\{ a_{U_{i_b,b'}^t} | b' \in \hat{\mathcal{N}}_b \right\} \tag{21}$$

$$a_{U_{i_b,b'}^t} = \left\{ (x_{u,m}^t, m) |\ m \in \{b, b'\}, x_{u,m}^t \in \{0,1\} \right\} \tag{22}$$

where $\hat{\mathcal{N}}_b$ is a set that includes base station b and its neighboring base stations, $x_{u,m}^t = 1$ if user u is served by their requested slice at base station m (and 0 if the user is served by the EcoSlice). Also, $U_{i_b,b'}^t$ is a subset of $\mathcal{U}_{i_b}^t$ (i.e. $U_{i_b,b'}^t \subseteq \mathcal{U}_{i_b}^t$). If $b \neq b'$, $U_{i_b,b'}^t$ is the set of users who can potentially be associated with the neighboring base station b'. When $b = b'$, $U_{i_b,b'}^t$ is the set of users who do not have other options and can only be associated with base station b.

Additionally, the reward for the user association problem is formulated to resonate with our main objective: to strike a balance between the energy consumption of the slice instances and the QoS experienced by users. Note that our slice agents collaborate with one another, and thus they are trained based on a global reward accordingly. In this respect, the immediate global reward of the slice agent can be expressed as:

$$r_{i_b}^{global}(t) = \sum_{b \in \mathcal{B}} \sum_{i \in \mathcal{I}_b} r_{i_b}^{local}(o_{i_b}^t, a_{i_b}^t) \tag{23}$$

$$r_{i_b}^{local}(o_{i_b}^t, a_{i_b}^t) = \frac{1}{E_{i_b}^t} + \beta \cdot \eta_{i_b}^t \tag{24}$$

with β being the QoS impacting factor. $r_{i_b}^{global}(.)$ is the global reward, which is the summation of all the local rewards $r_{i_b}^{local}(.)$ of slice agents. Again, $r_{i_b}^{global}(.)$ ensures that all the slice agents are cooperative. The objective of a slice agent is:

$$\underset{\pi_{i_b}}{max}\ \mathbb{E}\left[\sum_{t \in \chi} r_{i_b}^{global}(t) \right] \tag{25}$$

V. Proposed Solution

In this section, we begin by presenting the ICE-CREAM architecture. After that, we provide details to give a better understanding of how the framework functions step by step.

A. ICE-CREAM Architecture

The architecture of the proposed multi-agent ICE-CREAM solution is depicted in Figure 3. Note that we only consider two base stations in this illustration for the sake of simplicity. Our proposed framework features two types of agents operating over two timescales: base station agent (large timescale) and slice agent (small timescale). On the one hand, a base

Algorithm 1: Base station b agent learning procedure for the large timescale τ

Input: Probability of selecting a random action ϵ, learning rate α, maximum time steps T
Output: $\widehat{R}^{*}_{\tilde{w}_b}(\tau)$

1 Initialize $\tilde{w}_b$ randomly
2 **for** $\tau \in T$ **do**
3 Observe partial context/state: o_b^τ - based on definition $o_b^\tau = \left\{f(c_{i_b}^{\tau-1}, \mathcal{I}_b), \eta_b^{\tau-1}\right\}$
4 Predict reward distribution for each action: $\left[\widehat{R}_{\tilde{w}}(a_b^\tau|o_b^\tau)\right]_{a_b^\tau \in \mathcal{A}_b^\tau}$
5 **if** *generate random probability:* $rand() < \epsilon$ **then**
6 Select a random action a_b^τ
7 **else**
8 Select $a_b^\tau = \underset{a_b^\tau \in \mathcal{A}_b^\tau}{argmax} \left[\widehat{R}_{\tilde{w}}(a_b^\tau|o_b^\tau)\right]$
9 **end if**
10 Obtain local reward $r_b^{local}(o_b^\tau, a_b^\tau)$ according to Equation (17)
11 Share slice configuration a_b^τ to the corresponding slice agents residing at $i_b \in \mathcal{I}_b$, as well as with every neighbor $b' \in \mathcal{N}_b$
12 Evaluate global reward $r_b^{global}(\tau)$ according to Equation (16)
13 Update new partial state $o_b^{\tau+1}$
14 Calculate the loss: $\mathcal{L}(\tilde{w}_b) \triangleq \left(r_b^{global}(\tau) - \left[\widehat{R}_{\tilde{w}}(a_b^\tau|o_b^\tau)\right]_{a_b^\tau}\right)^2$
15 Update the weights: $\tilde{w}_b \leftarrow \tilde{w}_b - \alpha \bigtriangledown \mathcal{L}(\tilde{w}_b)$
16 **end for**

Algorithm 2: Slice instance i_b agent learning procedure for the small timescale t

Input: Probability of selecting a random action ϵ, learning rate α, maximum time steps χ
Output: $\widehat{R}^{*}_{\tilde{w}_{i_b}}(t)$

1 Initialize $\tilde{w}_{i_b}$ randomly
2 **for** $t \in \chi$ **do**
3 Observe partial context/state: $o_{i_b}^t$ - based on definition $o_{i_b}^t = \left\{E_{i_b}^{t-1}, \eta_{i_b}^{t-1}\right\}$
4 **if** $t\%\tau = 0$ **then**
5 Obtain slice activation/deactivation decision a_b^t from the base station agent
6 Update action set $\mathcal{A}_{i_b}^t$
7 **end if**
8 Predict reward distribution for each action: $\left[\widehat{R}_{\tilde{w}}(a_{i_b}^t|o_{i_b}^t)\right]_{a_{i_b}^t \in \mathcal{A}_{i_b}^t}$
9 **if** *generate random probability:* $rand() < \epsilon$ **then**
10 Select a random action $a_{i_b}^t$
11 **else**
12 Select $a_{i_b}^t = \underset{a_{i_b}^t \in \mathcal{A}_{i_b}^t}{argmax} \left[\widehat{R}_{\tilde{w}}(a_{i_b}^t|o_{i_b}^t)\right]$
13 **end if**
14 Obtain local reward $r_{i_b}^{local}(o_{i_b}^t, a_{i_b}^t)$ according to Equation (24)
15 Evaluate global reward $r_{i_b}^{global}(t)$ according to Equation (23)
16 Update new partial state $o_{i_b}^{t+1}$
17 Calculate the loss: $\mathcal{L}(\tilde{w_{i_b}}) \triangleq \left(r_{i_b}^{global}(t) - \left[\widehat{R}_{\tilde{w}}(a_{i_b}^t|o_{i_b}^t)\right]_{a_{i_b}^t}\right)^2$
18 Update the weights: $\tilde{w_{i_b}} \leftarrow \tilde{w_{i_b}} - \alpha \bigtriangledown \mathcal{L}(\tilde{w_{i_b}})$
19 **end for**

station agent is deployed at the base station level and its main responsibility is to make a slice activation/deactivation decision over the large timescale τ. On the other hand, a slice agent is used at the level of each slice instance to make a user association decision at the smaller timescale t.

As shown on the right side of Figure 3, each base station agent is faced with different configuration options. The base station agent can observe their own environment and share the information (i.e. slice activation/deactivation configurations) with its neighboring base station agents. Each configuration is linked to a specific reward. The responsibility of the base station agent is to pick up the most suitable configuration at the end of every τ. Having the slice activation/deactivation decision from a base station agent, the slice agents linked to the corresponding base station then determine how users are associated with these slices, taking into consideration the user satisfaction and network energy consumption improvement, as shown on the left side of Figure 3.

B. Multi-agent Fully Cooperative Decentralized Framework

ICE-CREAM is composed of two types of agents: *1)* base station agent (as articulated in Algorithm 1) and *2)* slice agent (as articulated in Algorithm 2).

1) Base Station Agent: The inputs of the base station agent consist of the probability of selecting a random action ϵ, the learning rate α for the deep neural network (DNN), and the maximum time steps T to train the base station agent. The output is a trained model, which can predict a reward distribution $\widehat{R}^{*}_{\tilde{w}_b}(\tau)$ for the available actions of the associated partial state. With this, Algorithm 1 begins by initializing the weights $\tilde{w}_b$ with arbitrary values (Line 1). At each step τ, the base station agent observes a partial state o_b^τ. It then proceeds to predict the reward distribution for all the available actions (Line 4) based on the current partial state. Subsequently, an action a_b^τ is chosen, taking into account the exploration and exploitation trade-offs (Line 5 - Line 9). In particular, there is a probability ϵ of selecting a random action, while the action with the highest predicted reward is chosen otherwise. The selected action is then applied to the network environment, which generates an actual reward (calculated using Equation 17) (Line 10). In addition, the chosen action (i.e., slice activation/deactivation configuration) at base station b is shared not only with the corresponding set of slice agents $\mathcal{I}_b$, but also with its neighboring base stations $\mathcal{N}_b$ (Line 11).

The immediate global reward $r_b^{global}(\tau)$ is obtained according to Equation (16) (Line 12). Meanwhile, the new partial state $o_b^{\tau+1}$ is obtained based on the action a_b^{τ} (Line 13). Then, the difference between the predicted reward and the actual global reward (denoted as a loss function) is computed for model training purposes (Line 14). We rely on a gradient descent method to update the weights matrix of the DNN with a learning rate α (Line 15).

2) Slice Agent: A slice agent operates over the small timescale where t represents each epoch. Similar to the base station agent, the inputs of Algorithm 2 are the probability ϵ of selecting a random action, the learning rate α, and maximum time steps χ for training the agents. We note that, for simplicity, we use the same ϵ and α values as in Algorithm 1, but the two algorithms are in fact independent and different values could be used for these parameters.

Algorithm 2 starts by initializing the weights $\tilde{w}_{i_b}$ with arbitrary values (Line 1). At each small time step t, the slice agent observes a partial state $o_{i_b}^t$. At every τ period (i.e. $t\%\tau = 0$), the slice agent updates its available action set $\mathcal{A}_{i_b}^t$ (Lines 5-7). It then predicts the reward distribution for the available actions (Line 8). An action $a_{i_b}^t$ is then chosen, considering exploration and exploitation (Lines 9 to 13), where we use a probability ϵ for selecting a random action.

The chosen action is applied to the network environment to obtain an actual reward using Equation (24) (Line 14). Subsequently, the global reward is obtained using Equation (23) (Line 15). A new partial state denoted as $o_{i_b}^{t+1}$ is received (Line 16) and the loss is calculated (Line 17). Finally, the model is trained based on the gradient descent method (Line 18).

VI. Evaluation

In this section, we evaluate the performance of our proposed solution using a real-world dataset and compare it with established baselines. We start with a description of the dataset and network environment. Afterward, we explain the benchmark approaches and implementations that we employ. Finally, we engage in a comprehensive discussion of the results.

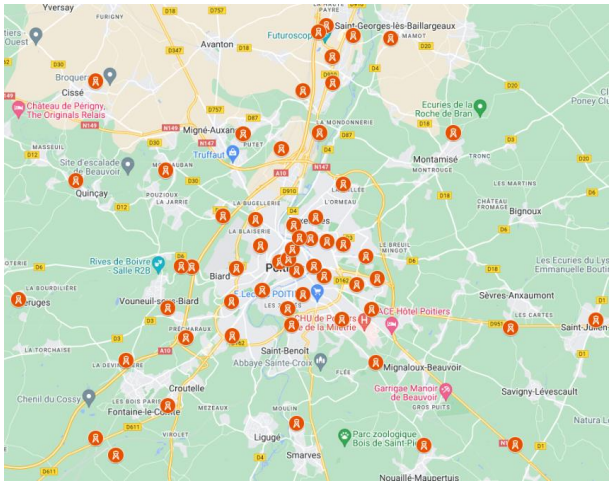

Fig. 4: 4G site locations in Poitiers, France.

A. Dataset and Simulation Behavior

With no commercial slice-based deployment from our knowledge, evaluating ICE-CREAM in a real environment is complicated. Therefore, we conduct a simulation-based study using real-world data collected from the Orange 4G network, in Poitiers, France. The dataset [42] provides, for each 1 minute time interval, the downlink and uplink traffic recorded at the base stations in multiple French urban areas, from which we select the city of Poitiers. The traffic information is provided separately for 68 different mobile services. As no slice information is available in the dataset, we assume slice instances are deployed on a service basis, i.e., one service maps to one slice instance. We note that this is a popular approach for slice deployment in the mobile network industry [14]. More precisely, in the following, Facebook, YouTube, and Google have been considered as three different types of slice instances attached to each base station. It is worth mentioning that these three different applications are used for very different purposes and exhibit different traffic demands, which is appropriate to be used for the simulation of network slicing [43]. Besides, we consider that users who access YouTube require the lowest delay, while Facebook and Google have higher flexibility in terms of delay. Accordingly, the assumed values for the user delay demand of each application are given in Table II.

The data used in this study covers 3 days in May 2019. With this, for our simulation purposes, we consider T to be 3 days, SADI $\tau = 10$ minutes and $t = 1$ minute. Thus, T includes up to 432 τ and 4320 t intervals. We analyze the 57 base stations from the Poitiers city, where 3 different slice instances are associated. Thus, we end up with a total of 171 slice agents and 57 base station agents for the small timescale and the large timescale, respectively. The locations of the 57 cellular sites in the city are shown in Figure 4. The detailed simulation parameters are summarized in Table II.

TABLE II: LIST OF PARAMETERS

Parameter	Value
P_b^{static} [44]	18 Watts
P_b^{fixed} [45]	139 Watts
P_b^{load} [45]	742 Watts
N_0 [28]	-174dBm
d_{thr}	2km
BW_{i_b}[Facebook, YouTube, Google]	[15,20,15] MHz
$BW_{i_{be}}$	5MHZ
H_{slice}	0.00005
H_{BS}	0.0001
$H_{slice-BS}$	0.0002
$P_{b,u}^t$	19dBm-75dBm
ψ_i [Facebook, YouTube, Google, EcoSlice]	[1.5, 1.8, 1.6, 1]
η_u [Facebook, YouTube, Google]	[5, 1, 10] ms
Loss function	MSE
Learning rate (small and large timescale) α	0.001
β [QoS, Trade-off, Energy]	[5,1,0.8]
Maximum time steps T of large timescale τ	100,300
Maximum time steps χ of small timescale t	1000,3000

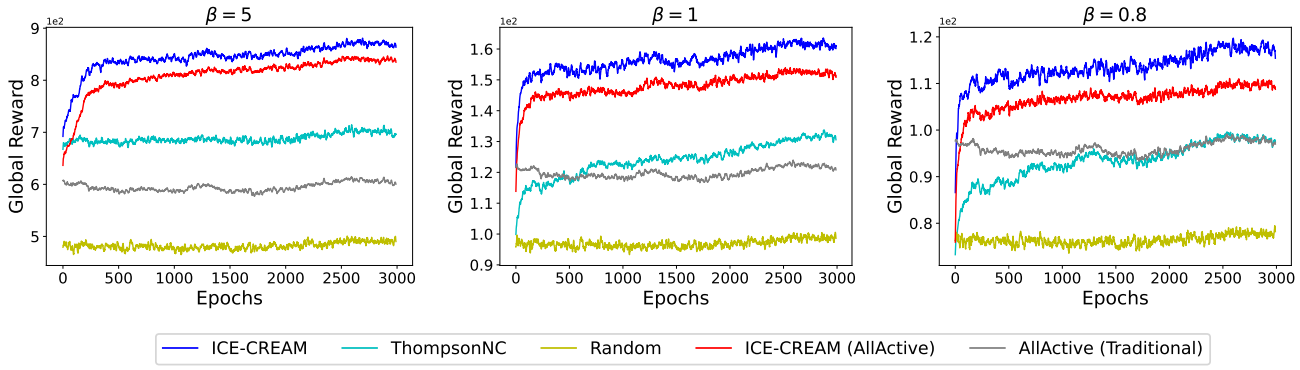

Fig. 5: Global reward obtained for different β values.

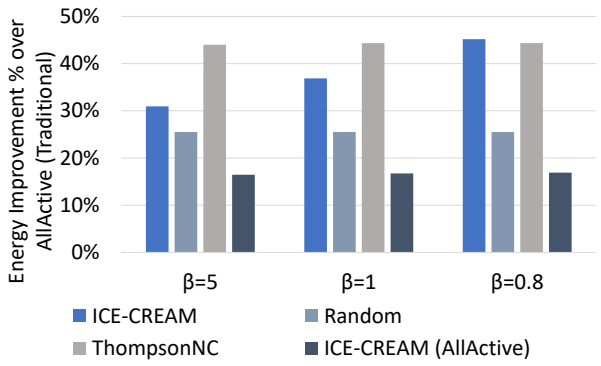

Fig. 6: Energy improvement over AllActive (Traditional) for different β values.

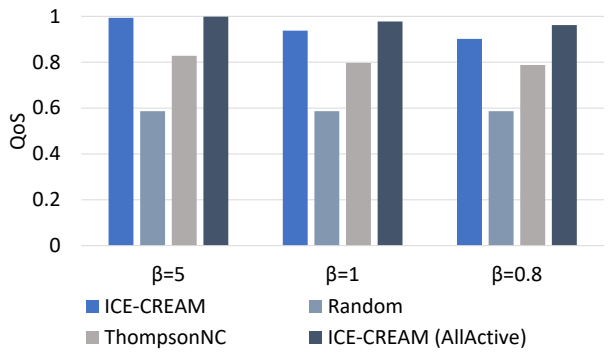

Fig. 7: Comparisons of achieved QoS for different β values.

B. Benchmarks and Implementation Setup

To validate the performance of our ICE-CREAM solution, we implement six counterparts as following:

- **Strategic Sleeping** [26]: Since their work is closely related to ours, we adapt their solution to suit our RAN slicing problem. Specifically, we make the same assumption that the slice sleep scheduling has knowledge of the number of users within the coverage area of each slice instance. Then, it sorts the number of associated users in descending order across all base stations to which a specific slice instance is associated, before deciding a percentage of slice instances to remain active. Accordingly, N_{act} represents the percentage of slice instances to be activated, and N_{deact} represents the percentage of instances to be deactivated.
- **Thompson Sampling Contextual (Thompson-C)**: This is a classical RL method where the Thompson-C agent adopts a statistical approach with the goal of achieving a proper estimation of the posterior distribution of expected reward for each action based on the observed state/context [24].
- **Thompson Sampling Non-Contextual (Thompson-NC)**: Unlike Thompson-C, no state/context information is considered in Thompson-NC [46].
- **ICE-CREAM (AllActive)**: ICE-CREAM (AllActive) conducts the same way as ICE-CREAM, except the fact that all the slice instances (not only the EcoSlice) remain active continuously.
- **AllActive (Traditional)**: All the slice instances are active all the time, and users are being served by their associated slice instances at the target base station, without the possibility to be navigated to neighboring base stations or on the EcoSlice.
- **Random**: As the name implies it, a random action is selected at each iteration for the Random approach.

For the implementation, we rely on the Pytorch framework for the ICE-CREAM agents. All the models (i.e. ICE-CREAM, Strategic Sleeping, Thompson-C, Thompson-NC, ICE-CREAM (AllActive), AllActive (Traditional), and Random) are implemented using a Python environment and trained on the high-performance Linux server provided by the Digital Research Alliance of Canada. The DNN of the ICE-CREAM agents (for both the slice agent and the base station agent) is composed of three fully connected layers of 100 neurons, each followed by a rectified linear unit (ReLU) activation function. The detailed parameters are summarized in Table II.

C. Performance of ICE-CREAM

1) Overall Agents Performance: First, to fully comprehend the performance of our agents, we study the trends of the global reward for $\beta = [5, 1, 0.8]$. We note that agents focus on QoS when $\beta = 5$, search for a trade-off between QoS and energy when $\beta = 1$, and stress on energy when $\beta = 0.8$.

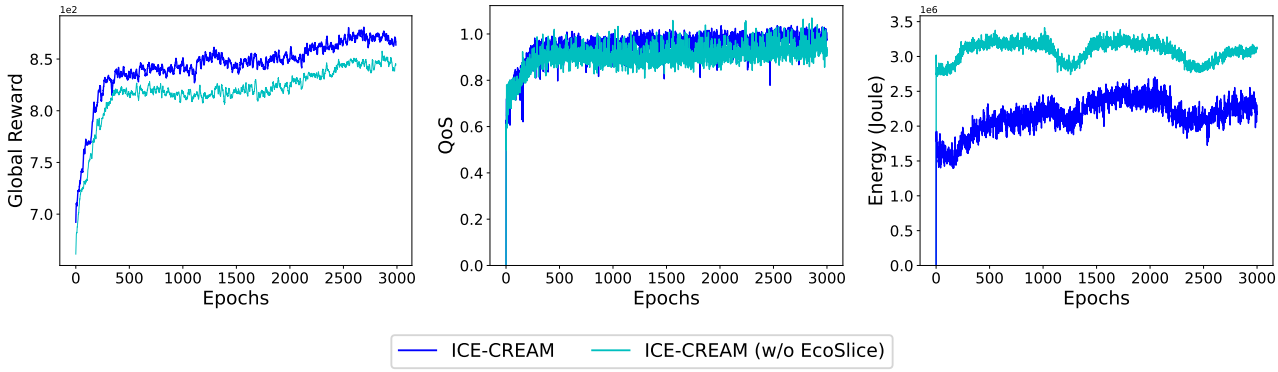

Fig. 8: Comparisons of ICE-CREAM Vs ICE-CREAM (w/o EcoSlice).

Through our results, we note that the evolution of the global reward on the large timescale aligns with that on the small timescale. Moreover, the small timescale also provides insights into a more granular level of detail. Therefore, for brevity, we deliberately present the global reward evolution only per small timescale in the following. The curves are smoothed by averaging within a rolling window of 10 iterations.

As we can see in Figure 5, the ICE-CREAM agents achieve consistently superior global rewards, even significantly better than ICE-CREAM (AllActive), followed by Thompson-NC and Random across different β values. While ICE-CREAM (AllActive) generally demonstrates a lower performance compared to our proposed solution, it still outperforms other approaches. These results show that collaborating with neighboring base stations and enabling users to be served by them has an important impact on network efficiency. This proves that our proposed solution performs well even in networks with limited (and even none) slice activation/deactivation functionalities (e.g., networks with a high reconfiguration cost). In addition, we note the significantly lower performance of Thompson-NC. On the other hand, agents with awareness of the state/context consistently outperform those that do not consider it in this particular problem. As anticipated, random approaches failed our proposed solutions in every scenario.

2) Roles of Agents in Energy and QoS: We next explicitly verify if our proposed solution is practical for energy optimization in RAN slicing, taking the AllActive (Traditional) strategy as a reference. Illustrated in Figure 6, the energy enhancement achieved by ICE-CREAM is notable, approximately 45%, 38%, and 31% for β values of 0.8, 1, and 5. In line with expectations, the energy gain deteriorates when the β value increases. While confirming the same phenomena as prior, we observe a limited change in energy gain for different β values for Thompson-NC, Random, and ICE-CREAM (AllActive). Overall, ICE-CREAM demonstrates a good performance when compared to other strategies in terms of reducing energy consumption. Yet, Thompson-NC has better energy gains than ICE-CREAM across all β values, and we will discuss the main reason behind this below.

Of course, optimizing energy consumption means compromising QoS to an extent. In this light, we visualize the QoS of all the agents in Figure 7. It is observed that our proposed solution delivers almost 100% QoS at $\beta = 5$. Therewith, it is at $\beta = 5$ where our agents make themselves stand out, as they deliver the same QoS as AllActive (Traditional), while providing significant improvement in energy efficiency. Furthermore, our suggested solution achieves a comparable QoS to ICE-CREAM (AllActive) for different β values while reaching approximately double the energy gains (we refer back to Figure 6). It is worth noting that Thompson-NC exhibits a significantly lower QoS performance for $\beta = [5, 1, 0.8]$, which explains and counterbalances its status as the top energy achiever. Lastly, the random approach shows the lowest QoS performance among the compared strategies.

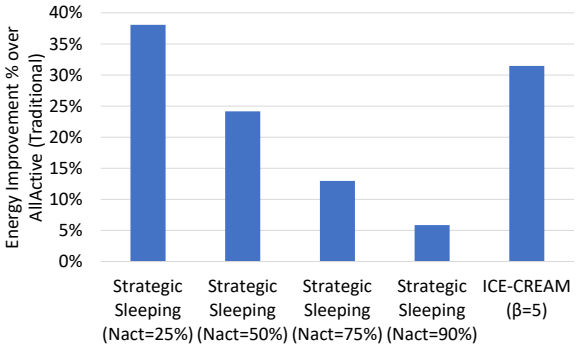

Fig. 9: Energy improvement over AllActive (Traditional) comparisons of Strategic Sleeping and ICE-CREAM

Next, we compare the performance of ICE-CREAM with Strategic Sleeping by setting $N_{act} = [25\%, 50\%, 75\%, 90\%]$ for comparison with ICE-CREAM ($\beta = 5$). By considering the different N_{act} values, we gain a better understanding of the behavior of Strategic Sleeping in relation to our proposed solution. As shown in Figures 9 and 10, lower N_{act} values lead to greater energy savings, while higher N_{act} values result in improved QoS. It is pretty obvious that ICE-CREAM outgrows Strategic Sleeping in finding the sweet spot between energy savings and QoS. It further proves the credibility of ICE-CREAM in coping with multi-objectives problem where it can efficiently explore the given environment and learn subtle cues to maximize the overall expected global reward.

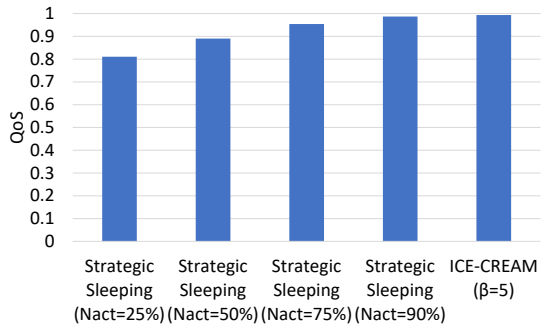

Fig. 10: Comparisons of achieved QoS for Strategic Sleeping and ICE-CREAM.

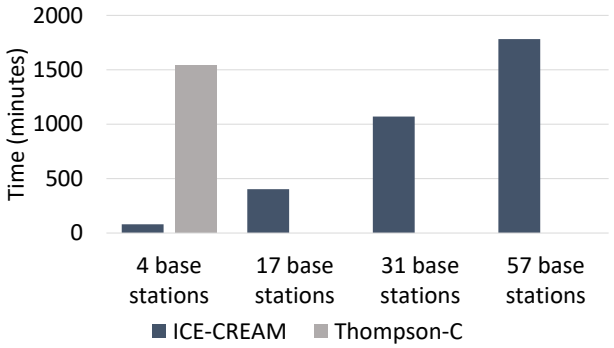

Fig. 12: Computing time comparison of ICE-CREAM and Thompson-C.

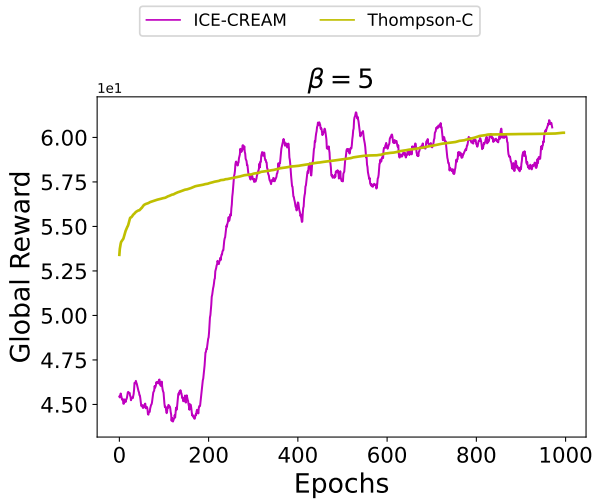

Fig. 11: Global reward comparison for ICE-CREAM and Thompson-C for four base stations.

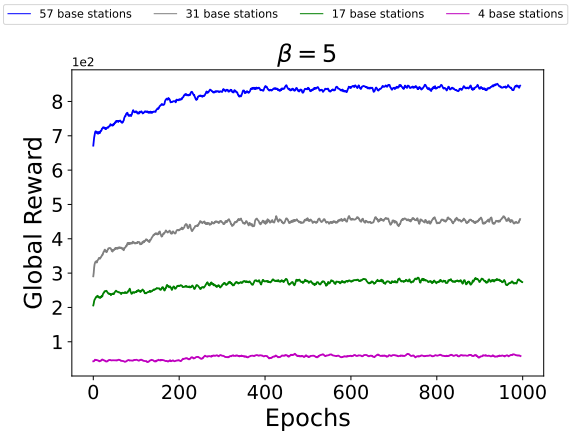

Fig. 13: Global reward comparison of ICE-CREAM for different network sizes.

3) Impact of the EcoSlice: To fully comprehend the benefits of using an EcoSlice, we examine in Figure 8 the performance of ICE-CREAM agents with and without an EcoSlice, in terms of global reward, QoS, and energy utilization. As we can observe, ICE-CREAM demonstrates better performance under the different metrics when compared to ICE-CREAM (w/o EcoSlice). Indeed, ICE-CREAM provides approximately 15% lower energy consumption in the presence of an EcoSlice than in its absence, while respecting the same level of QoS (this can be seen in the right-most and middle graphs of Figure 8). It is clear that EcoSlice significantly enhances the overall energy efficiency of the network by enabling operators to deactivate underutilized slice instances while still ensuring a certain QoS.

D. Scalability of ICE-CREAM

The key to a sound assessment of the scalability of ICE-CREAM is to investigate its ability to deal with increasing network size. In this light, we compare it with Thompson-C, which proved to be the best solution in our preliminary work [24], on a simpler problem.

We observe the global reward evolution in Figure 11 where ICE-CREAM and Thompson-C are compared for a set of four neighboring base stations. By studying the numerical results, one can easily find that our proposed solution achieves a similar global reward compared to Thompson-C. However, the computing time of Thompson-C is notably higher, as shown in Figure 12. In fact, the computing time for Thompson-C with four base stations nearly matches the computing time of ICE-CREAM for the entire city of Poitiers (i.e., 57 base stations).

Moreover, we observe the progression of the global reward for ICE-CREAM for a varying number of base stations (i.e., 4, 17, 31, and 57 base stations), as depicted in Figure 13. As anticipated, the global reward increases as the number of base stations increases. It is worth stressing that ICE-CREAM performs consistently across diverse network sizes. This demonstrates that our proposed solution is scalable and reliable with the increase in network size.

E. ICE-CREAM with Different QoS Functions

To further appreciate the performance of ICE-CREAM, we analyze its behavior when different slices present different QoS objectives. Considering this, we assume that the QoS for Facebook users is based on data rate, the QoS for YouTube users is determined by delay, and the one for Google users is also based on data rate, but less sensitive than Facebook. Specifically, we assume that Facebook users require a data rate

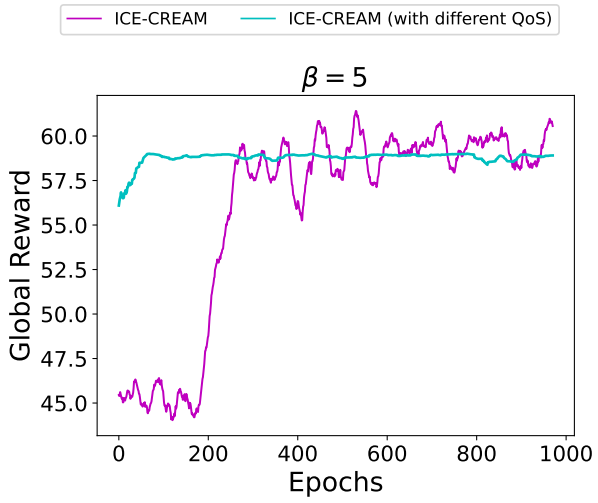

Fig. 14: Global reward comparison for ICE-CREAM with different QoS objectives.

of $r_u^t = 0.5Mbps$, YouTube users require a delay of $\delta_u^t = 1ms$, and Google users need a data rate of $r_u^t = 0.05Mbps$. Regarding data rate based QoS, similar to Equation (12), we can compute the QoS of each user as $\eta_u^t = \frac{1}{1+e^{-\theta(r^t_{i_b,u} - r^t_u)}}$. Besides considering the different QoS requirements per slice, all other parameters are kept the same as in the previous simulation. Figure 14 shows the behavior of ICE-CREAM with different QoS objectives in terms of global reward. This essentially demonstrates that ICE-CREAM performs well even in heterogeneous QoS settings. The intuitive explanation is that the ICE-CREAM framework is designed to make decisions per slice, with each agent receiving feedback specific to its corresponding slice, independent of other slices.

VII. Conclusion

This study is centered on enhancing energy efficiency in RAN slicing by addressing the joint slice activation/deactivation and user association problem. To tackle this, we introduce an ML-driven state-aware MAB approach, named ICE-CREAM, where multiple agents work collectively to activate the optimal slice instances (at a large timescale) and also navigate the associated users to the most suitable slice instances (at a small timescale), thereby minimizing the energy consumption while ensuring user QoS. Our results, based on a real-world dataset, demonstrate that the ICE-CREAM framework significantly reduces energy consumption at the base station level while preserving user QoS compared to numerous considered baselines. Moreover, we further showcase the adaptability of our solution to networks with constraints in terms of slice activation and deactivation. More than anything else, embracing the pivotal role of global coordination between base stations and slice instances, our approach carefully maps excessive demands on a given slice in an area with underutilized resources present in the neighborhood. All in all, our results provide valuable insights into how operators can further optimize their services in a sliced architecture, providing new services while keeping energy consumption under control.

As future work, we plan to further assess the performance of ICE-CREAM considering that the sets of available slice instances can vary from one base station to another, while they were assumed homogeneous in this study. Another path to explore is to evaluate ICE-CREAM under more realistic communication and energy models, e.g. by considering slices using specific waveform, or multiple-input multipe-output (MIMO) transmissions. Ideally, we plan to implement ICE-CREAM in an open RAN context and evaluate it in real deployments.

References

[1] B. Cubukcuoglu, "The Importance of Environmental Sustainability in Telecom Service Providers' Strategy," *Risk, Reliability and Sustainable Remediation in the Field of Civil and Environmental Engineering*, pp. 249–254, 2022.

[2] N. Jones, "The Information Factories," *Nature*, no. 561, pp. 163–167, 2018.

[3] NGMN, "Green Future Networks - Network Energy Efficiency," NGMN, Tech. Rep., 2021.

[4] Y. Fan, B. Wang, J. Wei, M. Tan, and H. Ran, "A Hierarchical Distributed Operational Framework for Renewables-Assisted 5G Base Station Clusters and Smart Grid Interaction," *Frontiers in Energy Research*, vol. 10, pp. 1–4, June 2022.

[5] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network Slicing in 5G: Survey and Challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 94–100, 2017.

[6] M. Masoudi, O. T. Demir, J. Zander, and C. Cavdar, "Energy-Optimal End-to-End Network Slicing in Cloud-Based Architecture," *IEEE Open Journal of the Communications Society*, vol. 3, pp. 574–592, April 2022.

[7] N. Piovesan, D. López-Pérez, A. D. Domenico, X. Geng, H. Bao, and M. Debbah, "Machine Learning and Analytical Power Consumption Models for 5G Base Stations," *IEEE Communications Magazine*, vol. 60, no. 10, October 2022.

[8] F. Han, Z. Safar, and K. J. Liu, "Energy-Efficient Base-Station Cooperative Operation with Guaranteed QoS," *IEEE Transactions on Communications*, vol. 61, no. 8, pp. 3505–3517, 2013.

[9] M. Feng, S. Mao, and T. Jiang, "Base Station On-Off Switching in 5G Wireless Networks: Approaches and Challenges," *Wireless Communications*, vol. 24, no. 4, p. 46–54, January 2017.

[10] L. Maggi, C. Mihailescu, Q. Cao, S. Aaltonen, R. Koblitz, M. Holma, S. Macchi, M. E. Ruggieri, I. Korenev, and B. Klausen, "Energy Savings under Performance Constraints via Carrier Shutdown with Bayesian Learning," *arXiv*, February 2023.

[11] A. Chatzipapas, S. Alouf, and V. Mancuso, "On the Minimization of Power Consumption in Base Stations using On/Off Power Amplifiers," *Proc. IEEE Online Conference on Green Communications (GreenCom)*, pp. 18–23, 2011.

[12] W. Guo, S. A. Wagan, D. R. Shin, I. F. Siddiqui, J. Koo, and N. M. F. Qureshi, "Periodic-Collaboration-Based Energy-Efficient Cell Dormancy in Heterogeneous Dense Networks," *Proceedings - 2022 IEEE 23rd International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM 2022*, pp. 527–534, 2022.

[13] Y. Sun, M. Peng, S. Mao, and S. Yan, "Hierarchical Radio Resource Allocation for Network Slicing in Fog Radio Access Networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3866–3881, 2019.

[14] H. P. Phyu, D. Naboulsi, and R. Stanica, "Machine Learning in Network Slicing — A Survey," *IEEE Access*, vol. 11, pp. 39 123–39 153, 2023.

[15] A. Abouaomar, A. Taik, A. Filali, and S. Cherkaoui, "Federated deep reinforcement learning for open ran slicing in 6g networks," 2022.

[16] J. Wang, W. Guan, Y. Huang, R. Schober, and X. You, "Distributed optimization of hierarchical small cell networks: A gnep framework," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 2, pp. 249–264, 2017.

[17] I. Vila Munoz, J. Perez-Romero, O. Sallent, and A. Umbert, "A Multi-Agent Reinforcement Learning Approach for Capacity Sharing in Multi-tenant Scenarios," *IEEE Transactions on Vehicular Technology*, pp. 1–1, jul 2021.

[18] F. Rezazadeh, H. Chergui, L. Christofi, and C. Verikoukis, "Actor-Critic-Based Learning for Zero-touch Joint Resource and Energy Control in Network Slicing," *Proc. IEEE International Conference on Communications (ICC)*, 2021.

[19] Y. Azimi, S. Yousefi, H. Kalbkhani, and T. Kunz, "Energy-Efficient Deep Reinforcement Learning Assisted Resource Allocation for 5G-RAN Slicing," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 1, pp. 856–871, 2022.
[20] O. Akin, U. C. Gulmez, O. Sazak, O. U. Yagmur, and P. Angin, "GreenSlice: An Energy-Efficient Secure Network Slicing Framework," *Journal of Internet Services and Information Security*, vol. 12, no. 1, pp. 57–71, 2022.
[21] Y.-J. Liu, G. Feng, Y. Sun, S. Qin, and Y.-C. Liang, "Device association for ran slicing based on hybrid federated deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 15 731–15 745, 2020.
[22] M. A. Hossain and N. Ansari, "Energy Aware Latency Minimization for Network Slicing Enabled Edge Computing," *IEEE Transactions on Green Communications and Networking*, vol. 5, no. 4, pp. 2150–2159, 2021.
[23] Y. K. Tun, D. H. Kim, M. Alsenwi, N. H. Tran, Z. Han, and C. S. Hong, "Energy efficient communication and computation resource slicing for eMBB and URLLC coexistence in 5G and beyond," *IEEE Access*, vol. 8, pp. 136 024–136 035, 2020.
[24] H. P. Phyu, D. Naboulsi, R. Stanica, and G. Poitau, "Towards energy efficiency in ran network slicing," in *2023 IEEE 48th Conference on Local Computer Networks (LCN)*, 2023, pp. 1–9.
[25] H. Chergui, L. Blanco, L. A. Garrido, K. Ramantas, S. Kuklinski, A. Ksentini, and C. Verikoukis, "Zero-Touch AI-Driven Distributed Management for Energy-Efficient 6G Massive Network Slicing," *IEEE Network*, vol. 35, no. 6, pp. 43–49, 2021.
[26] C. Liu, B. Natarajan, and H. Xia, "Small Cell Base Station Sleep Strategies for Energy Efficiency," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 3, pp. 1652–1661, 2016.
[27] T. Zhou, Z. Liu, J. Zhao, C. Li, and L. Yang, "Joint user association and power control for load balancing in downlink heterogeneous cellular networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 3, pp. 2582–2593, 2018.
[28] X. Wang and T. Zhang, "Reinforcement Learning Based Resource Allocation for Network Slicing in 5G C-RAN," *2019 Computing, Communications and IoT Applications, ComComAp 2019*, pp. 106–111, 2019.
[29] M. Jiang, M. Condoluci, and T. Mahmoodi, "Network slicing management & prioritization in 5G mobile systems," *European Wireless Conference 2016, EW 2016*, pp. 197–202, 2016.
[30] C. Marquez, A. Banchs, M. Gramaglia, C. Ziemlicki, M. Fiore, and Z. Smoreda, "Not All Apps Are Created Equal: Analysis of Spatiotemporal Heterogeneity in Nationwide Mobile Service Usage," *Proc. 13th International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, pp. 180–186, 2017.
[31] W. Guan, X. Wen, L. Wang, Z. Lu, and Y. Shen, "A Service-Oriented Deployment Policy of End-to-End Network Slicing based on Complex Network Theory," *IEEE Access*, vol. 6, pp. 19 691–19 701, 2018.
[32] A. K. Bairagi, M. S. Munir, M. Alsenwi, N. H. Tran, S. S. Alshamrani, M. Masud, Z. Han, and C. S. Hong, "Coexistence Mechanism between eMBB and uRLLC in 5G Wireless Networks," *IEEE Transactions on Communications*, vol. 69, no. 3, pp. 1736–1749, 2021.
[33] M. Bennis, M. Debbah, and H. V. Poor, "Ultrareliable and Low-Latency Wireless Communication: Tail, Risk, and Scale," *Proceedings of the IEEE*, vol. 106, no. 10, pp. 1834–1853, 2018.
[34] S. R. Sabuj, A. Ahmed, Y. Cho, K. J. Lee, and H. S. Jo, "Cognitive UAV-Aided URLLC and mMTC Services: Analyzing Energy Efficiency and Latency," *IEEE Access*, vol. 9, pp. 5011–5027, 2021.
[35] C. Marquez, M. Gramaglia, M. Fiore, A. Banchs, and X. Costa-Perez, "Resource Sharing Efficiency in Network Slicing," *IEEE Transactions on Network and Service Management*, vol. 16, no. 3, pp. 909–923, 2019.
[36] G. Sun, K. Xiong, G. O. Boateng, G. Liu, and W. Jiang, "Resource slicing and customization in RAN with dueling deep Q-Network," *Journal of Network and Computer Applications*, vol. 157, no. July 2019, p. 102573, 2020. [Online]. Available: https://doi.org/10.1016/j.jnca.2020.102573
[37] D. U. Kim, S. B. Park, C. S. Hong, and E. N. Huh, "Resource Allocation and User Association Using Reinforcement Learning via Curriculum in a Wireless Network with High User Mobility," *International Conference on Information Networking*, vol. 2023-Janua, pp. 382–386, 2023.
[38] K. Stylianopoulos, G. Alexandropoulos, C. Huang, C. Yuen, M. Bennis, and M. Debbah, "Deep Contextual Bandits for Orchestrating Multi-User MISO Systems with Multiple RISs," *IEEE International Conference on Communications*, vol. 2022-May, pp. 1556–1561, 2022.
[39] F. A. Oliehoek and C. Amato, *A Concise Introduction to Decentralized POMDPs*. Springer, 2016.
[40] Y. Shoham and K. Leyton-Brown, *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2009, pp. 215–216.
[41] Y. Xiao, Y. Song, and J. Liu, "Multi-Agent Deep Reinforcement Learning Based Resource Allocation for Ultra-Reliable Low-Latency Internet of Controllable Things," *IEEE Transactions on Wireless Communications*, vol. 22, no. 8, pp. 1–1, 2023.
[42] O. E. Martinez-Durive, S. Mishra, C. Ziemlicki, S. Rubrichi, Z. Smoreda, and M. Fiore, "The NetMob23 Dataset: A High-resolution Multi-region Service-level Mobile Data Traffic Cartography," Jul. 2023.
[43] H. P. Phyu, R. Stanica, and D. Naboulsi, "Multi-slice privacy-aware traffic forecasting at ran level: A scalable federated-learning approach," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2023.
[44] M. M. Aftab Hossain, C. Cavdar, E. Bjornson, and R. Jantti, "Energy Saving Game for Massive MIMO: Coping with Daily Load Variation," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 3, pp. 2301–2313, 2018.
[45] B. Debaillie, C. Desset, and F. Louagie, "A Flexible and Future-Proof Power Model for Cellular Base Stations," in *IEEE 81st Vehicular Technology Conference (VTC Spring)*, 2015, pp. 1–7.
[46] S. Agrawal and N. Goyal, "Near-Optimal Regret Bounds for Thompson Sampling," *Journal of the ACM*, vol. 64, no. 5, 2017.

Hnin Pann Phyu received the M.Sc. degree in communication networks and services from the Institut Mines-Telecom, France, in 2016, and the M.Eng. degree in telecommunications from the Asian Institute of Technology, Thailand, in 2016. From 2016 to 2020, she was a Network Strategist and Architect Engineer with Telenor, Myanmar. She is currently pursuing a Ph.D. degree in Software and IT Engineering with École de Technologie Supérieure University, Montreal, Canada. Her current research interests include next-generation mobile communication systems, machine learning, big data analytics, resource management and network slicing.

Diala Naboulsi is an Assistant Professor at the École de Technologie Supérieure (ÉTS), Canada. Before that, she held a Research Professional position in the Ultra-TCS research chair, at ÉTS, Canada, and a Research Associate position and a Postdoctoral Researcher position at Concordia University, Canada. She was also a Visiting Researcher at Ericsson, Canada. She held Course Lecturer positions at McGill University, Canada and Concordia University, Canada. She obtained the Ph.D. degree in Computer Science from INSA Lyon, France in 2015. As part of a double degree program, she received in 2012 the M.Sc. degree in Computer Science from INSA Lyon, France and the M.Eng. degree in Telecommunications from the Lebanese University, Lebanon. Her research interests are in mobile networks, virtualized networks, and wireless networks. She holds an NSERC Discovery Grant (2020-2026) on network slicing in future mobile networks. She has collaborations with several providers in communications technology such as Ultra Intelligence & Communications and Octasic.

Razvan Stanica is an associate professor at INSA Lyon, France, and a research scientist with the Inria Agora team of the CITI laboratory. He obtained a M.Eng. degree and a Ph.D. in computer science, both from INP Toulouse, France, in 2008 and 2011 respectively. His research interests include wireless mobile networks, with a special focus on communication networks in urban environments.