SLA-Driven Traffic Steering in B5G Systems with Network Slicing

C. Gijón, T. Mahmoodi, *Senior Member, IEEE*, M. Toril, S. Luna-Ramírez, J. L. Bejarano-Luque.

***Abstract*—In 5G and beyond wireless systems, Network Slicing (NS) feature will enable the coexistence of extremely different services by splitting the physical infrastructure into several logical slices tailored for a specific tenant or application. In sliced Radio Access Networks (RANs), an optimal traffic sharing among cells is key to guarantee Service Level Agreement (SLA) compliance while minimizing operation costs. The configuration of network functions leading to that optimal point may depend on the slice, claiming for slice-aware traffic steering strategies. This work presents the first data-driven algorithm for slice-aware traffic steering by tuning handover margins (a.k.a. mobility load balancing) per slice and adjacency with a proportional controller based on SLA criteria. The tuning process is driven by a novel indicator, derived from connection traces, showing the imbalance of SLA compliance among neighbor cells per slice. Performance assessment is carried out with a system-level simulator implementing a realistic sliced RAN offering services with different throughput, latency and reliability requirements. Results show that the proposed algorithm improves the overall SLA compliance by 9% in only 15 minutes of network activity compared to the case of not steering traffic, outperforming two legacy mobility load balancing approaches not driven by SLA.**

***Index Terms*—B5G, network slicing, self-optimization, traffic steering, service level agreement.**

I. Introduction

5G and beyond networks are envisioned to pave the way for a fully-connected world by supporting new applications in vertical industries, such as augmented and virtual reality, unmanned aerial vehicles, e-health or fully autonomous driving [1]. As a consequence, these Next-Generation Networks (NGNs) will serve different type of end-users (humans or machines), devices (e.g., smartphones, vehicles, wearables, sensors...) and services with highly diverging Quality-of-Service (QoS) requirements (e.g., energy efficiency, End-to-End –E2E– latency, peak data rate...) [2]. In this context, Network Slicing (NS) feature has been recognized as a key enabler to meet the performance, scalability, security and reliability requirements of such diversified applications in complex NGNs [3].

NS consists in building separate logical networks tailored for specific purposes on top of a common physical infrastructure [4]. An End-to-End (E2E) slice instance comprises hardware, software and radio resources together with a set of Virtualized Network Functions (VNFs). Such assets provide storage, processing and networking capabilities required to comply QoS, security, mobility and availability conditions specified in the Service Level Agreement (SLA) during slice operation [5]. A central MANager and Orchestrator (MANO) manages all slices operating in a network. Among other tasks, the MANO splits resources among slices, decides which VNFs (e.g., access control, HandOver –HO–...) are common to all/multiple slices and which VNFs are tailored or even omitted per slice, and sets up VNF parameters [6].

Cellular network conditions often change due to events in the network infrastructure (e.g., deployment of new cells) or in the area (e.g., social events). In Self-Organized Networks (SON), self-optimization solutions automatically tune Network Function (NF) parameters to guarantee optimal network performance in this changing environment [7]. A well-known self-optimization use case is load balancing (a.k.a. traffic steering), which aims to alleviate congestion problems by offloading traffic from congested to underutilized cells. Traffic steering in the Radio Access Network (RAN) can be addressed by adjusting antenna parameters such as transmit power or tilt angle [8] [9]. However, this approach may create coverage holes. Alternatively, most works tackle load balancing by optimizing mobility NFs (a.k.a. Mobility Load Balancing, MLB), driven by logical parameters than can be cost-effectively and immediately tuned [10]. Cutting-edge MLB algorithms perform traffic steering in non-sliced networks from a service-centric perspective through heuristic controllers [11] [12], analytical models [13], optimization algorithms [14] or Reinforcement Learning (RL) agents [15] [16] driven by data (e.g., traces, performance counters...) gathered in the Operations and Support System (OSS).

In NGNs, NS makes network management extremely complex, calling for slice-aware self-optimization solutions that ensure SLA compliance in each slice with different resource allocation and performance targets [17]. Several aspects must be considered when designing slice-aware MLB algorithms, such as: a) the possibility of tailoring NFs per slice, allowing to set a different optimal configuration per slice, b) the need for SLA-based solutions driven by indicators reflecting end-user performance (e.g., E2E latency, throughput...), which can be derived from connection traces, and c) the increase in network dynamism due to the activation, deactivation or redimensioning of slice instances, demanding a fast MLB operation [3]. To the author knowledge, slice-aware MLB schemes considering these aspects have not been proposed yet.

This work contributes to the design of slice-aware MLB

C. Gijón, M. Toril, S. Luna-Ramírez and J. L. Bejarano-Luque are with the Telecommunication Research Institute (TELMA), Universidad de Málaga, Málaga, Spain, Spain (e-mail: {cgm, mtoril, sluna, jlbl}@ic.uma.es).

T. Mahmoodi is with the Centre for Telecommunications Research, King's College London, London, United Kingdom (e-mail: toktam.mahmoodi@kcl.ac.uk)

solutions by showing the poor operation of legacy MLB schemes in NS scenarios. Then, the first slice-aware MLB algorithm is proposed. The algorithm aims to increase the percentage of users meeting SLA. For this purpose, traffic is steered by self-tuning intra-frequency HO Margins (HOMs) in a slice-aware HO scheme. The tuning process is driven by a novel indicator, derived from connection traces, reflecting the imbalance in SLA compliance per slice between neighbor cells. Moreover, the algorithm operates on a time resolution finer than the legacy 15-min Reporting Output Period (ROP) to deal with the high dynamism expected in B5G scenarios. Two versions of the algorithm are tested, differing in the frequency of parameter tuning per adjacency and the number of simultaneously optimized adjacencies.

The rest of the document is organized as follows. Section II outlines related work and details the main contributions of this work. Section III formulates the problem of performing slice-aware MLB. Section IV presents a novel slice-aware traffic steering algorithm, assessed via simulation in section V. Finally, section VI summarizes the main conclusions and proposes future research directions. To ease readability, Table I describes notation used in this work.

TABLE I: Table of notation.

Notation	Definition
N_u	No. users in a network
u	User index
$\mathcal{S}$	Set of slices operating in a network
N_s	No. slices operating in a network
s	Slice index
s_u	Slice serving user u
$\mathcal{C}$	Set of cells in a network
N_c	No. cells in a network
c	Cell index
i, j	Cell indexes of cells in an adjacency
$RSRP_u(c)$	RSRP received by user u from cell c [dBm]
$HOM(i,j)$	HOM from cell i to cell j [dB]
$HOM(i,j,s_u)$	HOM from cell i to cell j for slice s_u [dB]
N_n	No. relevant neighbors per cell
$\mathcal{A}$	Set of relevant adjacencies in a network
$\mathcal{N}(c)$	Set of relevant neighbors of cell c
$\mathcal{G}$	Set of adjacency groups
N_g	No. adjacency groups in $\mathcal{G}$
g_k	Adjacency group index
$N_a(g_k)$	No. adjacencies belonging to group g_k
$N_u(c,)$	No. users with relevant DL activity in cell c
$N_u(c,s)$	No. users from slice s with relevant activity in the DL of cell c
$SLA(u,c)$	Level of SLA compliance for user u belonging to slice s in cell c
$N_{KPI}(s_u)$	No. KPIs included in the SLA for slice s_u
$w_p(s_u)$	Factor weighting importance of KPI p for slice s_u
$SLA_p(u,c)$	Level of SLA compliance regarding KPI p for user u served by cell c
$KPI_p(u,c)$	Performance of KPI p for user u in cell c
$KPI_p^{tgt}(s_u)$	Performance target for KPI p in the SLA of slice s_u
SLA_{max}	Maximum level of SLA compliance
$\overline{SLA}_{dif}(i,j,s)$	SLA compliance imbalance between cells i and j for slice s
TH	Thourghput [kbps]
LR	Latency-reliability commitment
$V_{DL}(u)$	Data volume transmitted to user u in the DL at PDCP layer [kbits]
$t_{session}(u)$	Session duration [s]
$p(u)$	No. packets in the session of user u
$p_{succ}(u)$	No. packets in the session of user u fulfilling target E2E latency
$V_{DL}(u,c,TI)$	DL data volume transmitted to user u in cell c during TI TI [kbits]
$t_{TI}(u,c)$	Period of TI where user u is served by cell c [s]
$p(u,c,TI)$	No. packets transmitted from cell c to user u during TI TI
$p_{succ}(u,c,TI)$	No. packets transmitted from cell c to user u during TI TI fulfilling target E2E latency
SLA_{global}	Global percentage of users complying SLA [%]
$SLA_{bin}(u)$	Binary level of SLA compliance per user u
SLA_{global}^{norm}	Binary level of SLA compliance per user u normalized to baseline
SLA_i	Global percentage of users demanding service i complying SLA
$\overline{PRB_{util}}$	Final avg. PRB utilization across cells [%]
$\overline{\lvert\delta HOM^{(n)}\rvert}(s)$	Avg. absolute HOM deviation per slice from initial settings in optimization loop n
nHO^{norm}	Ratio between number of HOs in a simulation compared to the baseline

II. Related work and contributions

In the literature, several algorithms have been proposed for MLB through HOM tuning in cellular RANs. Most contributions tackle HOM optimization as a control problem, for which different types of controllers have been proposed. The earliest works relied on proportional controllers driven by heuristic rules to perform intra-frequency MLB in macrocellular scenarios. In [18], an incremental controller tunes HOMs in fixed steps when the load difference of adjacent cells exceeds a threshold. Cell load is measured considering Physical Resource Block (PRB) utilization ratio and QoS requirements. In [19], the controller estimates the impact of changing HOMs on network performance with an analytical model, and tunes HOMs to maintain all cells under a preset load threshold. A similar approach is proposed for small-cell scenarios in [20], where an adaptive load threshold is considered, so that MLB can also act in non-congested cells with unevenly loaded cell boundaries. In [21], HOMs in a femtocell scenario are tuned with a Fuzzy Logic Controller (FLC) fed by current HOM values and blocking statistics to equalize the blocking ratio among cells. All these approaches have been extended to tackle inter-frequency [22] or inter-radio-access-technology [23] MLB.

With the latest advances in information technology, cutting-edge load balancing solutions rely on artificial intelligence. [10] surveys machine-learning-based load balancing schemes. Although some works are based on supervised learning (e.g., multiple linear regression [24]), most MLB schemes rely on RL. Initially, RL was used to enhance the ability of classical controllers to adapt to changing environments. For instance, [25] improves the solution proposed in [21] for femtocell scenarios by using a Q-learning agent that customizes IF-THEN rules of the FLC driven by information from trial-and-error interactions with the network. In [26], this fuzzy Q-learning approach is tested in macrocellular scenarios, revealing the potential of readjusting FLC rules with constant exploration and exploitation to capture changes in network conditions. As an alternative, RL can also be used to drive the control process. For instance, in [27], a Q-learning agent takes decisions per adjacency to equalize cell load from information about PRB utilization and cell-edge users.

All the above traffic steering algorithms are driven by simple

performance indicators obtained from the aggregation of all connections in a cell. In legacy networks, where voice calls were the predominant service, these approaches achieved the best user performance. However, field trial results in [28] point out that balancing cell load in Long Term Evolution (LTE) networks and beyond supporting services of very different requirements does not guarantee the best overall end-user Quality of Experience (QoE). As an alternative, more recent works tackle MLB from a QoE perspective. In [12], the QoE of neighbor cells in a single-layer macrocellular scenario is balanced by tuning service-specific HOMs with a FLC. Other QoE-based works rely on optimization instead of using heuristic control rules. For instance, in [13], the impact of tuning HOM on system QoE is estimated with an analytical model adjusted with network data, and optimality is ensured with a gradient ascent scheme. In [14], an algorithm based on dynamic particle swarm optimization is centrally applied, which optimizes the overall QoE and reduces the number of users with poor QoE. QoE aspects have also been considered in RL-based MLB solutions as part of state [29] and/or reward [15] [16].

Additionally to the need for service-oriented traffic steering, in NGNs, NS makes network management more complex, requiring novel self-optimization solutions to guarantee both tenant and end-user satisfaction in slices with different radio capacity and performance targets [17]. [30] showed the potential of slice-aware self-optimization to improve network performance, focusing on mobility robustness optimization. To this end, an algorithm tunes HOMs per slice with an actor-critic agent based on twin delayed deep deterministic policy gradient to improve service- and mobility-related performance metrics. Nonetheless, in the literature, most contributions on slice management focus on designing policies for new network functions such as slicing provisioning or slice admission control. To automate resource management in these processes, machine learning algorithms are often used [31]. Recent contributions propose traffic forecasting and classification with deep supervised learning to anticipate future resource needs per slice [32] [33], admission control with Deep RL (DRL) to grant access to new slices/users [34] [35] or dynamic slice provisioning with DRL for fine-grained resource allocation [36] [37] [38]. However, unexpected temporary changes in the spatial traffic distribution of certain slices in localized areas can lead to SLA violations. Although these issues could be solved by redimensioning the degraded slice, redistributing spectrum among tenants may degrade other slices and/or lead to an inefficient use of network resources. Alternatively, traffic served from a specific slice can be redistributed among cells (i.e., slice-aware traffic steering) to make the most of slice assets, leaving slice dimensioning for extreme cases where optimal slice traffic sharing among cells does not warrant SLA compliance.

The above-mentioned legacy traffic steering schemes are not suitable for sliced B5G RANs since: a) they lack of slice awareness, and b) they do not take into account SLA compliance in the tuning process. To the authors' knowledge, the task of performing slice-aware MLB has not been covered yet. Such a problem is addressed in this work. Specifically, the main contributions are:

a) Proving that legacy (i.e., slice-unaware) traffic steering schemes degrade SLA compliance in a realistic network slicing scenario offering enhanced Mobility BroadBand (eMBB) and ultra-Reliable Low-Latency (uRLLC) services, thus justifying the need for novel slice-aware solutions.This contribution aims to warn operators that enabling legacy self-optimization algorithms in sliced RANs will potentially lead to SLA violations.
b) Showing the potential of steering traffic per slice to improve SLA compliance for different slices based on their traffic patterns and performance requirements for the first time. Whereas dynamic slice spectrum allocation optimizes resource allocation in a given cell, slice-aware traffic steering takes advantage of underutilized cells to deal with non-uniform traffic distributions across the network.
c) Presenting a novel SLA-driven slice-aware MLB algorithm. The main contribution here is a novel driver indicator, derived from connection traces, reflecting the imbalance on SLA compliance between neighbor cells. This indicator is suitable for any slice definition in terms of performance indicators or target values for those indicators. Consequently, it can be used as a driver for other use cases requiring comparing the performance of different slices (e.g., dynamic slice capacity brokers), no matter the considered optimization policy (e.g., heuristic rules, RL...).
d) Analyzing the impact of key settings in the MLB process, such as time resolution (i.e., frequency of parameter changes per adjacency) or parallelization (i.e., simultaneous parameter changes in several adjacencies), on system performance. These aspects have a strong impact on network performance and determine the computational capacity required to run the algorithm on a live network. The latter aspect is critical for optimization schemes running on a second time horizon, such as the proposed MLB solution.

III. Problem formulation

Consider a cellular network with NS where a set of N_s slices, denoted as $\mathcal{S}$, operate simultaneously. In the RAN, the network comprises N_c cells, denoted as C, working at the same frequency band, so that every cell $c \in C$ may serve users from all active slices $s \in \mathcal{S}$. As typical in live networks, intra-frequency mobility is handled through power-budget HOs [11]. Thus, the HO of a user u from serving cell i to a neighbor cell j is triggered by HO event A3 based on Reference Signal Received Power (RSRP), i.e.,

$$RSRP_u(j) \geq RSRP_u(i) + HOM(i,j)\,, \tag{1}$$

where where $RSRP_u(i)$ and $RSRP_u(j)$ are pilot signal levels received by user u from the serving cell i and neighbor cell j, respectively, and $HOM(i,j)$ is the intra-frequency HOM, defined on a per-adjacency basis.

In this scenario, a legacy MLB algorithm would adjust $HOM(i,j)$ to steer traffic from congested to underutilized cells so that cell load is balanced. With the above HO scheme,

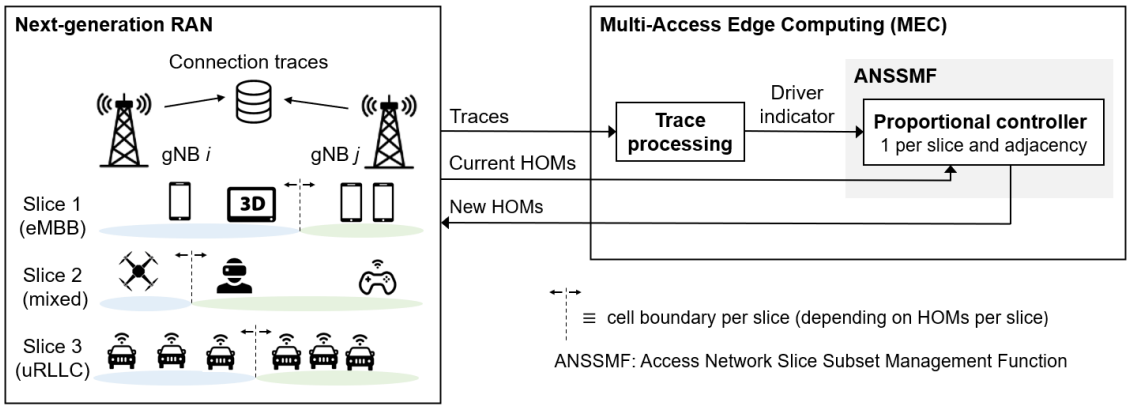


Fig. 1: Operation of proposed MLB algorithm.

parameter self-tuning can be performed on a per-adjacency basis. However, note that a certain HO set-up in a given adjacency may not lead to the same performance for all slices due to: a) the different traffic characteristics (e.g., user spatial distribution, mobility, performance requirements in SLA...) among slices, and b) the capacity broker, which may underestimate/overestimate resources required by a particular slice in a particular cell (or area), but not for others. This fact suggests the need for slice-aware MLB algorithms. For this purpose, a slice-aware HO scheme must be set first. The triggering equation for slice-aware RSRP-based HO event A3 can be expressed as

$$RSRP_u(j) \geq RSRP_u(i) + HOM(i, j, s_u)\,, \qquad (2)$$

where s_u is the slice to which user u belongs. With this new HO scheme, HOM tuning can be performed per adjacency and slice.

The aim of slice-aware MLB algorithms must be guaranteeing SLA compliance (and thus both tenant and end-user satisfaction) for all slices. However, as stated in [28], an evenly loaded scenario does not ensure that all cells offer the same performance (e.g., due to different radio link conditions). This behavior is expected to worsen in sliced RANs. In these networks, equalizing global load of neighbor cells can have a negligible (or even negative) impact on SLA compliance for slices offering services with low rate, whose performance is jeopardized by eMBB slices with a larger radio resource allocation. Moreover, even for eMBB slices, each slice may access to a different amount of PRBs in each cell. Consequently, the difference in spare PRBs per slice among neighbor cells may be different from the global load imbalance. To capture these differences, slice-aware MLB strategies must be SLA-driven (and not load-driven, as legacy solutions).

When designing slice-aware SLA-driven MLB algorithms, it should be taken into account that optimizing network performance globally sometimes compromises cell-edge users (i.e., those with the highest risk of violating SLA). Such an issue is circumvented by approaches that equalize performance among cells, as that presented in [12]. Likewise, a high dynamism is expected in sliced 5G networks due to slice activation, deactivation and resource reallocation [39]. As a consequence, slice-aware traffic steering must operate in a time resolution finer than legacy schemes, where HOMs are tuned based on performance counters updated every 15 minutes at most. Due to such dynamism, indicators driving the MLB process must reflect slice performance in the last few seconds, which can only be obtained by processing connection traces. All these aspects are considered by the slice-aware MLB algorithm proposed here.

IV. Traffic steering strategy

In this section, a novel slice-aware MLB algorithm is presented. The algorithm aims to equalize the level of SLA compliance (i.e., the percentage of users meeting SLA) per slice across the scenario by steering traffic among cells working at the same frequency band. For this purpose, $HOM(i, j, s)$ in the slice-aware HO scheme presented in (2) is iteratively self-tuned on a per-adjacency-and-slice basis.

Fig. 1 illustrates the MLB process in a given iteration. The self-tuning algorithm, run by the Access Network Slice Subset Management Function (ANSSMF), implements a specific proportional controller per slice and adjacency. Each controller is fed by a novel indicator reflecting the imbalance of SLA compliance per slice in neighbor cells. Such an indicator is computed in the Multi-Access Edge Computing (MEC) unit by processing connection traces collected by the gNBs of the adjacency since the previous optimization iteration.

When enabling the slice-aware intra-frequency HO scheme in (2), an initial value of $HOM(i, j, s)$=3 dB is set $\forall\ i, j, s$, as starting point for traffic steering. To prevent ineffective parameter changes, the MLB algorithm operates on a subset of adjacencies denoted as $\mathcal{A}$, comprising a limited number of relevant adjacencies per cell. Moreover, to avoid that changing several HOMs for a cell simultaneously leads to excessive reduction/increase of cell area for a slice, HOM tuning is not performed simultaneously for all adjacencies in $\mathcal{A}$.

For clarity, the adjacency selection and clustering strategy is first detailed, the self-tuning algorithm is presented later and the computational complexity is finally discussed.

A. Stage 1. Adjacency selection and clustering

The subset of adjacencies in the whole network where MLB will operate, $\mathcal{A}$, is created as follows. For every cell c in the

Algorithm 1 Adjacency clustering algorithm.

Input: a set of adjacencies $\mathcal{A}$
Output: a set of adjacency groups $\mathcal{G}$
function groupAdjacencies($\mathcal{A}$)
  $\mathcal{G} \leftarrow \{\}$
  $k \leftarrow 1$
  while $\mathcal{A} \neq \{\}$ **do**
    $g_k \leftarrow \{\}$
    for all adjacencies $a = (i \leftrightarrow j) \in \mathcal{A}$ **do**
      if none adjacency $a' \in g_k$ contains cell i **or** j **then**
        Add adjacency a to group g_k
        Remove adjacency a from $\mathcal{A}$
      end if
    end for
    Add group g_k to $\mathcal{G}$
    $k \leftarrow k$+1
  end while
  return $\mathcal{G}$

scenario, a fixed number of relevant neighbors cells, N_n, is selected. That set of N_n neighbors per cell c, denoted as $\mathcal{N}(c)$, includes: a) all co-sited cells, and b) the most interfering cells in the Down Link (DL) from nearby sites. Then, bidirectional adjacencies $(c \leftrightarrow j)\ \forall\ j \in \mathcal{N}(c)$ are included in $\mathcal{A}$. After repeating this process for all cells in the scenario, the number of adjacencies in $\mathcal{A}$ is $N_c \times N_n$. Then, duplicated adjacencies in $\mathcal{A}$ (if any) are removed.

Next, adjacencies in $\mathcal{A}$ are divided into a set of disjoint groups, $\mathcal{G}$. Clustering is performed with the heuristic scheme presented in Algorithm 1, inspired in [40]. Groups are created sequentially. For each group g_k, a random adjacency from $\mathcal{A}$ is first selected as seed and removed from $\mathcal{A}$. Then, another adjacency in $\mathcal{A}$ is randomly selected to be added to group g_k if it does not include any cell in the adjacency previously added to group g_k. More adjacencies are sequentially added to group g_k until no adjacency in $\mathcal{A}$ comprises disjoint cells with all adjacencies already in the group. Then, a new group g_{k+1} is created. This process is repeated until $\mathcal{A}$ becomes empty.

The subsequent MLB iteratively tunes HOMs. In each iteration k, only HOMs from adjacencies in group g_k are modified. As a consequence, the number of groups in $\mathcal{G}$, N_g, determines how often parameters change per adjacency. Since N_g grows with N_n, to ensure fast and optimal convergence, N_n must have the lowest value allowing to include all relevant adjacencies per cell in $\mathcal{A}$. Nonetheless, N_g may vary in different executions if the random seed changes. It is recommended to perform multiple runs of the clustering algorithm with different seeds before optimization starts, and select the solution providing the lowest N_g.

It should be pointed out that HOM tuning reshapes cell serving area, and thus DL interference may change once the tuning process begins (e.g., due to cell load changes). However, it is strongly recommended to perform the above adjacency clustering process with a stable HOM set-up and redefine it only after a significant event altering radio link performance in the network (e.g., deployment of a new cell).

B. Stage 2. SLA-driven HOM tuning

Once adjacency groups have been created, a slice-aware self-tuning algorithm is executed. For clarity, the indicator driving the tuning process is described first and the control algorithm is presented later.

a) Description of the driver: The average level of SLA compliance for slice s in the DL of a given cell c during a certain period of time can be expressed as

$$\overline{SLA}(c,s) = \frac{1}{N_u(c,s)} \sum_{u=1}^{N_u(c,s)} SLA(u,c)\,, \tag{3}$$

In the equation above, $N_u(c,s)$ is the number of users from slice s with relevant activity in the DL of cell c, i.e., those with data to be transmitted in at least 5% of transmission time intervals during the considered time period (in [41], all services similar to those considered here showed higher DL activity ratios). Likewise, $SLA(u,c)$ is the level of SLA compliance for user u belonging to slice s in cell c. $SLA(u,c)$ is computed as

$$SLA(u,c) = \sum_{p=1}^{N_{KPI}(s_u)} w_p(s_u) SLA_p(u,c)\,, \tag{4}$$

where $N_{KPI}(s_u)$ is the number of Key Performance Indicators (KPIs) included in the SLA for slice s_u to which user u belongs, $w_p(s_u)$ is a weight factor showing the relative importance of KPI p for the performance of slice s_u, and $SLA_p(u,c)$ is the level of SLA compliance related to KPI p for user u served by cell c. $w_p(s_u)$ ranges from 0 to 1, so that $\sum_{p=1}^{N_{KPI}(s)} w_p(s) = 1\ \forall\ s$. Besides, $SLA_p(u,c)$ is calculated as

$$SLA_p(u,c) = \min\left(\frac{KPI_p(u,c)}{KPI_p^{tgt}(s_u)},\ SLA_{max}\right)\,, \tag{5}$$

where $KPI_p(u,c)$ denotes performance of KPI p for user u in cell c, $KPI_p^{tgt}(s_u)$ is the performance target for KPI p in the SLA of slice s_u, and SLA_{max} is a maximum level of SLA compliance to avoid that users exceeding the SLA conceal those with worse performance in (3).

The indicator driving HOM tuning is the difference of SLA compliance levels for slice s in the two cells i and j of an adjacency, $\overline{SLA}_{dif}(i,j,s)$, defined as

$$\overline{SLA}_{dif}(i,j,s) = \overline{SLA}(j,s) - \overline{SLA}(i,s)\,. \tag{6}$$

A negative value of $\overline{SLA}_{dif}(i,j,s)$ indicates that, on average, the level of SLA compliance for slice s is better in cell i than in cell j, whereas a positive value of $\overline{SLA}_{dif}(i,j,s)$ indicates the opposite. The HO point for a balanced scenario is given by the condition $\overline{SLA}_{dif}(i,j,s)$=0. At that point, on average, the level of SLA compliance for slice s is similar in both cells i and j.

b) Control algorithm: Algorithm 2 outlines the operation of the self-tuning algorithm, designed as a set of proportional controllers (one per adjacency and slice) that iteratively modify $HOM(i,j,s)$ based on the value of $\overline{SLA}_{dif}(i,j,s)$ indicator.

The algorithm is executed a predetermined number of optimization loops. A loop comprises N_g iterations. The inter-iteration time (hereafter referred to as Tuning Interval, TI) must be short enough to reflect the current (and not past) network state, but long enough to get reliable computations of SLA compliance for services with bursty traffic (e.g., in this work, TI=5 s). In each iteration k, the HOM value for adjacencies in group g_k is tuned incrementally on a per-adjacency-and-slice basis as illustrated in Fig. 2. Specifically, the increment/decrement in HOM, $\Delta HOM(i,j,s)$, is computed from the value of $\overline{SLA}_{dif}(i,j,s)$ as

$$\Delta HOM(i,j,s) = \begin{cases} 2 & \overline{SLA}_{dif}(i,j,s) < \alpha_1 \,, \\ 0 & \alpha_1 \leq \overline{SLA}_{dif}(i,j,s) \leq \alpha_2 \,, \\ -2 & \overline{SLA}_{dif}(i,j,s) > \alpha_2 \,, \end{cases} \quad (7)$$

where α_1 and α_2 are thresholds for triggering HOM changes so as to eliminate random actions due to small fluctuations of the driver indicator. These parameters must be set to provide an adequate trade-off between optimality and convergence speed (in this work, $\alpha_2 = -\alpha_1 = 0.05$). Larger absolute values reduce the number of optimization loops required to reach equilibrium at the expense of deteriorating network performance slightly, since the algorithm converges before $\overline{SLA}_{dif}(i,j,s) = 0$.

Then, the new value of $HOM(i,j,s)$ is computed as

$$HOM^{(k+1)}(i,j,s) = HOM^{(k)}(i,j,s) + \Delta HOM^{(k)}(i,j,s) \,. \quad (8)$$

To guarantee adequate HO performance, $HOM(i,j,s)$ values are limited to the range $[-6, 12]$ dB. Finally, to avoid ping-pong effect, in all cases, a 6-dB hysteresis area is maintained by jointly setting HOMs in both directions of an adjacency so that $HOM(i,j,s) + HOM(j,i,s)$=6 dB.Calculating $\Delta HOM(i,j,s)$ has negligible computational complexity.

Note that $\overline{SLA}(c,s)$ for empty cells (i.e., $N_u(c,s)$=0) must be set to a value higher than SLA_{max} + max($|\alpha_1|, |\alpha_2|$) to ensure that traffic is offloaded to the empty cell. It is also remarkable that a step larger than typical 0.5-dB HOM resolution has been chosen since user density decreases when considering only traffic from a slice, and hence a higher change in HOM is required in slice-aware HO schemes to offload traffic from congested cells.

Thanks to α_1 and α_2 parameters, as optimization progresses, HOMs are modified only in those adjacencies with the highest initial SLA compliance imbalance. The algorithm converges when no HOM changes are performed in any adjacency and slice. To guarantee convergence, a large number of optimization loops must be executed.

This MLB algorithm follows a fixed policy consisting of tuning HOMs iteratively (+/- 2 dB) according to rules in (7) until equilibrium is reached (i.e., $\overline{SLA}_{dif}(i,j,s) \approx 0$). Since the controller does not rely on supervised learning, model training (or retraining) is not required. Additionally, as the controlled system (RAN) is non-linear, providing a detailed theoretical convergence analysis is not trivial. Thus, there is no guarantee that the system converges in a limited number of iterations. In practice, to guarantee convergence, a large number of optimization loops must be executed. Thanks to α_1 and α_2 parameters, HOMs are modified only in those adjacencies with significant SLA compliance imbalance in each loop. Convergence is reached when no HOM changes are introduced in any adjacency and slice. A faster convergence can be achieved by increasing the gain of the feedback loop at the expense of potential instabilities. Alternatively, α_1 and α_2 can be increased to enlarge the range of $\overline{SLA}_{dif}(i,j,s)$ values where the output of the proportional controller is zero (no change).

Algorithm 2 SLA-driven slice-aware self-tuning algorithm.

Inputs: a set of adjacencies $\mathcal{A}$, a set of current HOM values in these adjacencies $\mathcal{H}$, and a set of adjacency groups $\mathcal{G}$
Output: Updated $\mathcal{H}$
function tuneHOM($\mathcal{A}, \mathcal{G}, \mathcal{H}$)
  repeat
    for all adjacency groups $g_k \in \mathcal{G}$ **do**
      Wait for TI
      Collect connection traces
      for all slices $s \in \mathcal{S}$ **do**
        for all adjacencies $a = (i \leftrightarrow j) \in g_k$ **do**
          Compute $\overline{SLA}_{dif}(i,j,s)$
          if $\overline{SLA}_{dif}(i,j,s) < \alpha_1$ **then**
            $\Delta HOM(i,j,s) \leftarrow 2$
          else if $\overline{SLA}_{dif}(i,j,s) > \alpha_2$ **then**
            $\Delta HOM(i,j,s) \leftarrow$ -2
          else
            $\Delta HOM(i,j,s) \leftarrow 0$
          end if
          $HOM(i,j,s) \leftarrow HOM(i,j,s) + \Delta HOM(i,j,s)$
          $HOM(j,i,s) \leftarrow HOM(j,i,s) - \Delta HOM(i,j,s)$
        end for
      end for
    end for
  until a predefined no. of optimization loops is reached

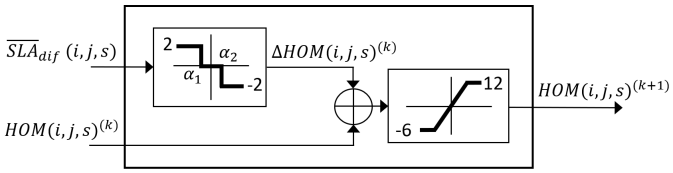


Fig. 2: Proportional controller.

C. Computational complexity

The proposed traffic steering strategy comprises two stages: a) creating adjacency groups, and b) executing the slice-aware MLB process. The time required for adjacency clustering (Algorithm 1) depends on the number of adjacencies where optimization will be carried out, given by the number of cells in the network and the number of relevant adjacencies per cell considered, i.e., $O\{N_c \times N_n\}$. Recall that this process is executed before parameter tuning starts and repeated only after events altering radio link performance in the network

significantly (e.g., deployment of a new cell), which are rare events. Thus, its contribution to the total computational complexity is negligible.

Regarding MLB process (Algorithm 2), the execution time for a given adjacency comprises the time to process connection traces from cells i and j, calculate $\overline{SLA}_{dif}(i,j,s)$ indicator driving the tuning process per slice, and compute HOM increment per slice, $\Delta HOM(i,j,s)$. Trace processing entails decoding events, creating connections and computing traffic descriptors. In live networks, this process is the most time-consuming task, with $O\{N_u(i)+N_u(j)\}$, where $N_u(c)$ is the number of active users in cell c during the tuning interval. The time to compute $\overline{SLA}_{dif}(i,j,s)$ indicator grows linearly with the number of active users in the slice in cells i and j, $N_u(i,s)$+$N_u(j,s)$. Finally, the MLB algorithm relies on a set of simple proportional controllers, and hence calculating $\Delta HOM(i,j,s)$ has negligible computational complexity. In each tuning interval k, this process must be performed for all slices in the $N_a(g_k)$ adjacencies belonging to adjacency group g_k. Thus, the total worst-case complexity is $O\{N_a(g_k)\times(N_u(i)+N_u(j)+N_s\times(N_u(i,s)+N_u(j,s)))\}$ for a centralized implementation (for a distributed implementation, $N_a(g_k)$ term can be omitted).

V. Performance assessment

This section presents the validation of the proposed slice-aware MLB algorithm. In the absence of commercial 5G networks with NS, method assessment is carried out via simulation. For clarity, the considered simulation tool is introduced first. Then, assessment methodology is detailed and results are presented later. Finally, execution time is outlined.

A. Simulation tool

Experiments are carried out with the dynamic system-level simulator used in [42], which emulates the DL activity of a realistic LTE-Advanced network with NS functionality. The main parameters and NS implementation in the simulator are briefly introduced next.

1) General description: Table II presents the main simulation parameters. The scenario, illustrated in Fig. 3, consists of 108 irregular cells located in urban and sub-urban areas covering 11×23 km^2. Cells work at 2.1 GHz in frequency division duplexing mode with a 10-MHz bandwidth. 5G numerology with $\mu = 0$ is set. Modulation and Coding Schemes (MCSs) in the 4-bits Channel Quality Indicator (CQI) table in [43] are used. A link abstraction model maps radio link performance to Block Error Rate (BLER) on a certain MCS [44]. For each user, the most efficient MCS guaranteeing a service-specific BLER is selected. The simulation tool models latency in data transmission due to packet scheduling (for computational efficiency, a 10-ms time resolution is considered). However, additional delays in the decision-making process of the proposed traffic steering strategy (e.g., those from transferring and processing traces) are not considered. The reader is referred to [45] for more information on simulation parameters.

Network users demand four services, namely file download via FTP (FTP), live video streaming (VIDEO), haptic communications (HAPTIC) and autonomous driving (DRIVING). The combination of these four traffic profiles could represent a typical automated driving scenario, with infotainment and map update services. Table III describes traffic models for these services. As in live networks, users can be static or move with a speed set per service.

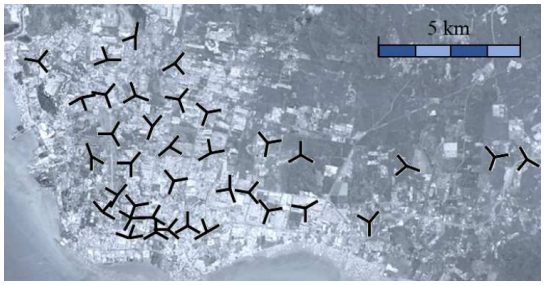


Fig. 3: Simulated scenario [45].

TABLE II: Main simulation parameters.

Parameter	Description
Time resolution	10 ms
Cell bandwidth	10 Mhz
Transmission mode	Frequency division duplexing
5G numerology (μ)	0
Propagation model	Path loss: Hata, COST-231 [46]
	Slow fading: log-normal σ_{SF}=8 dB, d_c=50 m
	Fast fading: ETU model [47]
Base station model	Tri-sectorized antennas, MIMO 2x2, transmit power from real base stations ([47.8-49] dBm), no beamforming
Packet scheduler	Classical exponential/proportional fair [48]
Link adaptation	CQI-based, MCS selected to guarantee a target BLER defined per service
Service model	FTP, VIDEO, HAPTIC and DRIVING
Traffic distribution	Non-uniform spatial user distribution and service mix
User mobility	Constant speed at 0 km/s (static), 3 km/s (pedestrian) or 50 km/h (car) and constant random direction
HO set-up	Intra-frequency HO triggered by RSRP-driven event A3
	$HOM(i,j,s)$=3 dB and time-to-trigger of 256 ms for all adjacencies and slices
NS implementation	Slicing at packet scheduling level, adaptive capacity broker

2) Network slicing implementation: NS implementation in the simulation tool is thoroughly described in [42]. The RAN is sliced at packet scheduling level, i.e., each slice has exclusive access to a set of PRBs that may differ in number and PRB index per cell [51]. The set of PRBs allocated to each slice is determined by an adaptive capacity broker. Thus, intra-cell slice isolation is guaranteed, but not inter-cell slice isolation.

The contracted SLA is defined in terms of performance targets for the expected traffic in a given area. Two performance KPIs are considered, computed on a session level. The first KPI is DL session throughput, TH, defined as

TABLE III: Traffic models in simulation tool.

Service	Traffic model
FTP	File size: log-normal (avg. [15, 85] MB)
VIDEO	Packet arrival process and file size from H.264/MPEG-4 AVC real trace with 720p resolution
	Call duration: uniform [30, 300] s
	Chunk size: 5 s of video content (initial burst), 2 s of video content (rest)
HAPTIC	Multi-point haptic traffic model in [49]
	Three components: globe, position tracker and actuators
	Packet size per component: fixed (min. 72 B, max. 442 B)
	Inter-packet time per component: Gaussian (avg. [10.87, 12.95] ms, std. dev. [1.98, 2.49] ms)
	Call duration: uniform [300, 600] s
DRIVING	Packet size=201 B
	Inter-packet arrival time of 100 ms (derived from lane merge use case data [50])
	Call duration: uniform [300, 600] s

$$TH(u) = \frac{V_{DL}(u)}{t_{session}(u)}\,, \tag{9}$$

where $V_{DL}(u)$ is the total data volume transmitted to user u in the DL at packet data convergence protocol layer, and $t_{session}(u)$ is session duration. The second KPI is latency-reliability commitment, LR, defined as the ratio of packets transmitted in a session with an E2E latency below a predefined threshold [52], i.e.,

$$LR(u) = \frac{p_{succ}(u)}{p(u)}\,, \tag{10}$$

where $p(u)$ is the total number of packets in the transmission buffer during the session of user u and $p_{succ}(u)$ is the number of those packets fulfilling target E2E latency for the slice to which user u belongs. In the simulation tool, it is assumed that: a) a packet is a block of information to be transmitted, and b) E2E latency is the time from the packet arrives to transmission buffer until it is scheduled.

During experiments, the level of SLA compliance per user is computed in two different ways: a) per session, as a Figure of Merit (FoM) to assess algorithm performance, and b) per session, cell and TI, to calculate $\overline{SLA}_{dif}(i,j,s)$ indicator driving the HOM tuning process. In the latter case, with the above SLA definition, equation (4) can be particularized as

$$\begin{aligned} SLA(u,c) = {} & w_{TH}(s_u) SLA_{TH}(u,c) + \\ & w_{LR}(s_u) SLA_{LR}(u,c)\,. \end{aligned} \tag{11}$$

Note that $SLA_{TH}(u,c)$ and $SLA_{LR}(u,c)$ must reflect SLA compliance in terms of TH and LR per user, cell and TI. In $SLA_{TH}(u,c)$ calculation, $TH(u,c)$ is computed as

TABLE IV: Simulation set-up for assessing MLB strategies.

Slice	Service	Speed	$TH^{tgt}(s)$	$LR^{tgt}(s)$
1	FTP VIDEO	3 km/h Static	1 Mbps	1 s for 90% of packets
2	HAPTIC	Static	400 kbps	10 ms for 99.9% of packets
3	DRIVING	30 km/h	16 kbps	10 ms for 99.9% of packets

$$TH(u,c) = \frac{V_{DL}(u,c,TI)}{t_{TI}(u,c)}\,, \tag{12}$$

where $V_{DL}(u,c,TI)$ is the DL data volume transmitted to user u in cell c during the corresponding TI, TI, and $t_{TI}(u,c)$ is the time period of TI where user u is served by cell c. Similarly, when computing $SLA_{LR}(u,c)$, $LR(u,c)$ only considers packets that arrive to the transmission buffer and are sent or dropped within TI, i.e.,

$$LR(u,c) = \frac{p_{succ}(u,c,TI)}{p(u,c,TI)}\,. \tag{13}$$

For the above calculations, radio connection traces should be processed in a live environment.

B. Assessment methodology

Table IV summarizes NS set-up for method assessment. Three slices operate simultaneously in the network. Slice 1 serves FTP and VIDEO users, with the highest TH requirement ($TH^{tgt}(1)$=1 Mbps), but a relaxed target LR ($LR^{tgt}(1)$=1 s for 90% of packets). FTP users are pedestrians moving at 3 km/h, whereas VIDEO users are static. Slice 2 serves HAPTIC traffic, generated from static users demanding a moderate TH (400 kbps) with stringent LR requirements (10 ms for 99.9% of packets). Finally, slice 3 serves DRIVING users moving at 30 km/h demanding a low rate (16 kbps), but with the same stringent LR requirements as HAPTIC users.

The proposed traffic steering algorithm, referred to as SLA-driven MLB over Slice-Aware HO scheme (SAHO+SLA) is compared with other three MLB strategies. Since no RL-based slice-aware traffic steering algorithm has been proposed yet, all tested solutions rely on heuristic rule-based controllers, but differing on the driver indicator, slice awareness and/or operation mode (i.e., parallel vs. sequential tuning in all adjacencies). The first, referred to as Load Balancing over Legacy HO scheme (LHO+LB), is a classical MLB algorithm that tunes HOMs per adjacency on a legacy (i.e., slice-unaware) HO scheme, whose aim is to balance PRB utilization across cells. The second, referred to as Load Balancing over Slice-Aware HO scheme (SAHO+LB), steers traffic on a per-adjacency-and-slice basis to balance PRB utilization of those PRBs assigned to each slice between adjacent cells. To justify the need for adjacency clustering in SAHO+SLA, a third strategy referred to as SLA-driven MLB with fast convergence over Slice-Aware HO scheme (SAHO+SLAfast) is considered, which applies the proposed slice-aware self-tuning algorithm, but omitting adjacency clustering (i.e., HOMs for all adjacencies in $\mathcal{A}$ are tuned every TI). A simulation without MLB, referred to as No MLB, is also run as a benchmark.

For each of the above MLB strategies, 14 optimization loops (a total of 15 minutes of network activity) are simulated. In the starting point (i.e., TI=0), the adaptive capacity broker has already reached steady state. Therefore, resource allocation per slice remains fixed during the optimization process. The number of relevant neighbors per cell, N_n, is set to 6, with a total of 427 adjacencies ($i \leftrightarrow j$) in the network to be optimized per slice. The adjacency clustering algorithm results in N_g=13 groups of adjacencies. TI is set to 5 s. Recall that, in LHO+LB and SAHO+LB and SAHO+SLA, HOM is tuned once per optimization loop for each adjacency. With the above set-up, a loop lasts for 5×13=65 s, which is a reasonable time to adapt to rapid changes in network conditions. Finally, SLA_{max}=1.2 and $w_{TH}(s) = w_{LR}(s) = 0.5 \ \forall \ s \in \mathcal{S}$.

The main FoM to assess algorithm performance is the percentage of users complying SLA in terms of both TH and LR, SLA_{global}, computed as

$$SLA_{global} = \frac{100}{N_u} \sum_u SLA_{bin}(u) \quad [\%] \ , \tag{14}$$

where N_u is the number of users in the scenario and $SLA_{bin}(u)$ is a binary variable that equals '1' if slice-specific TH and LR targets are fulfilled for user u, and '0' otherwise. $SLA_{bin}(u)$ is computed from information in connection traces (i.e., numerical performance metrics per user) as

$$SLA_{bin}(u) = \mathcal{H}\left(\frac{TH(u)}{TH^{tgt}(s_u)}\right) \cdot \mathcal{H}\left(\frac{LR(u)}{LR^{tgt}(s_u)}\right) \ , \tag{15}$$

where $\mathcal{H}(\cdot)$ denotes Heaviside function. SLA_{global} is constrained to the range [0, 100] to ease the interpretation of results.

From an operator perspective, maximizing SLA_{global} implies improving SLA compliance for all tenants, which is the final objective of the self-tuning process. As a counterpart, optimal network spectral efficiency (bits/Hz) cannot be guaranteed since the performance of the best users may be degraded (although still complying SLA) in favor of cell-edge users. This FoM is analyzed in absolute terms (SLA_{global}) and relative to that obtained with the benchmark (SLA^{norm}_{global}), i.e.,

$$SLA^{norm}_{global} = \frac{SLA_{global}}{SLA_{global\ baseline}} \ . \tag{16}$$

The overall SLA compliance per service, SLA_i $\forall \ i \in \{FTP, VIDEO, HAPTIC, DRIVING\}$ is similarly computed.

Five secondary FoMs are also considered. The first is the final $\overline{SLA}_{dif}(i,j,s)$ averaged for all the tuned adjacencies, showing the capacity of MLB strategies to balance SLA compliance among neighbor cells. The second is the final PRB utilization ratio across cells in the scenario, $\overline{PRB_{util}}$, as a proxy of resource usage. The third is the average absolute HOM deviation per slice from initial settings in the tuned adjacencies, $\overline{|\delta HOM^{(n)}|}(s)$, computed as

$$\begin{aligned} \overline{|\delta HOM^{(n)}|}(s) = \frac{1}{N_a} \sum_{(i,j)\in\mathcal{A}} |\delta HOM^{(n)}(i,j,s)| = \\ \frac{1}{N_a} \sum_{(i,j)\in\mathcal{A}} |HOM^{(n)}(i,j,s) - HOM^{(0)}(i,j,s)| \ , \end{aligned} \tag{17}$$

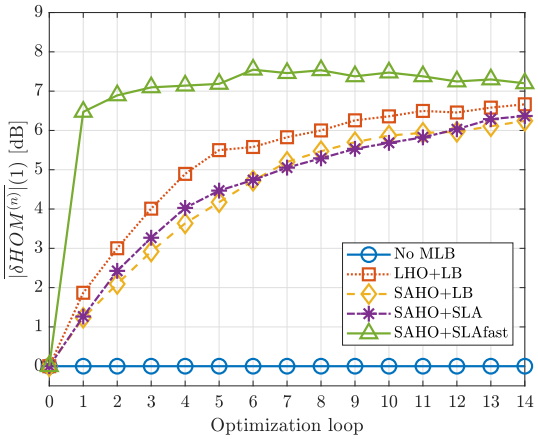
where n denotes optimization loop index, N_a is the number of adjacencies in $\mathcal{A}$, and $HOM^{(0)}(i,j,s)$ is the initial intra-frequency HOM value (i.e., in TI=0). Finally, the ratio between the number of HOs in a simulation compared to the baseline, nHO^{norm}, is also considered as a measure of the increase in signaling load caused by MLB.

C. Results

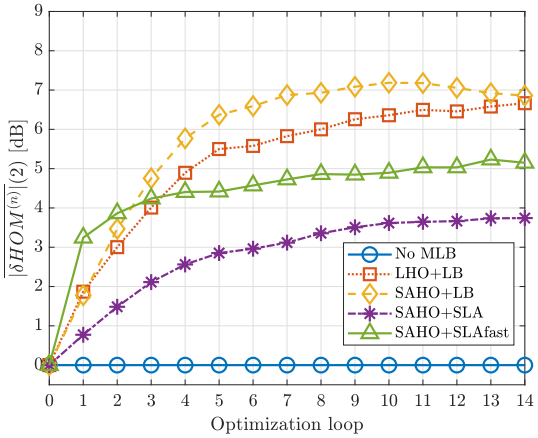
To analyze algorithm convergence, Fig. 4.a)–c) show the evolution of $\overline{|\delta HOM^{(n)}|}(s)$ per slice across the optimization process obtained for all the tested MLB strategies. The (almost) stable level observed in the last optimization loops in all curves confirms that all algorithms converge for all slices. As expected, SAHO+SLAfast shows the fastest convergence, since it tunes HOMs for all adjacencies in $\mathcal{A}$ simultaneously every TI (i.e., 5 s). It should also be pointed out that, since LBO+LB relies on a slice-unaware HO scheme, red curves in Fig. 4.a) to c) are identical. For the remaining approaches, the evolution of HOM settings significantly differs per slice. This observation suggests that, at the beginning of the tuning process, performance (i.e., load for SAHO+LB, and SLA compliance for SAHO+SLA and SAHO+SLAfast) in neighbor cells varies per slice. This phenomenon may be due to: a) the different spatial traffic distribution per slice, or b) a poor capacity broker performance for some slices in certain cells.

Fig. 4.a)–c) also give insight into how the tested algorithms operate in different slices. It is observed that slice 1 presents the most similar final HOM settings across algorithms. In this slice, eMBB traffic requires a high PRB allocation per cell. Hence, slice 1 performance strongly impacts cell PRB utilization ratio. As a consequence, LHO+LB may perform similarly to SAHO+LB. Moreover, since LR target for this slice is not too tight, the level of SLA compliance mainly depends on TH performance. As throughput is related to PRB utilization, load-based and SLA-based slice-aware approaches tend to tune HOMs in the same direction. Nonetheless, HOM set-up per strategy varies in many adjacencies, leading to different SLA_{FTP} and SLA_{VIDEO} FoMs, as will be shown later.

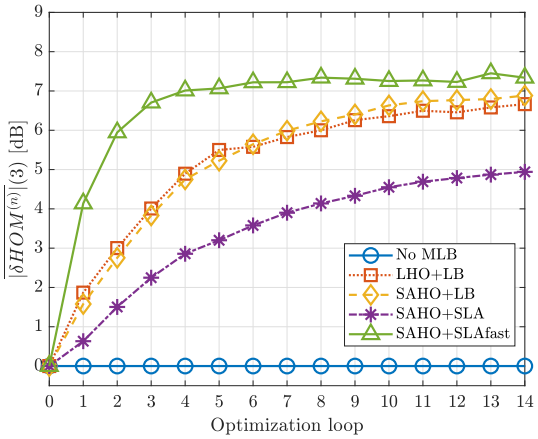
To illustrate the impact of MLB on network performance, Fig. 5 shows the evolution of SLA^{norm}_{global} for all the tested algorithms. It can be observed that, surprisingly, legacy LHO+LB algorithm presents the worst level of SLA compliance, even below the baseline case (i.e., SLA^{norm}_{global}<1). In contrast to LHO+LB, all the remaining algorithms (i.e., SAHO+LB, SAHO+SLA and SAHO+SLAfast) outperform the baseline case in terms of SLA^{norm}_{global} across the whole tuning process (i.e., curves over 1 in Fig. 5). This behavior confirms the potential of slice-aware MLB schemes to improve SLA compliance in NS scenarios. It is remarkable that SAHO+SLA approach presents unstable SLA^{norm}_{global} evolution, with the best initial



(a) Slice 1 (FTP + VIDEO).



(b) Slice 2 (HAPTIC).



(c) Slice 3 (DRIVING).

Fig. 4: Evolution of absolute HO margin deviation from default values in tuned adjacencies per slice.

results due to fast HOM tuning followed by an undesirable strong performance degradation, compensated later.

For a deeper analysis, Table V summarizes the value of all the considered FoMs at the end of the tuning process (i.e., average FoM values in TIs belonging to the last optimization loop), computed globally and broken down per slice. SLA FoMs per slice reveal that LHO+LB has very poor performance in slice 3 (DRIVING), with a $SLA_{DRIVING}$ degradation of 12.24% in absolute terms compared to No MLB. Note that the low target TH for DRIVING users leads to a reduced PRB allocation per cell to slice 3, which therefore has a negligible impact on cell PRB utilization that drives the tuning process in LHO+LB. In contrast, for slices 1 (FTP+VIDEO) and 2 (HAPTIC), with a higher PRB allocation per cell, LHO+LB outperforms the baseline, with SLA_{FTP}, SLA_{VIDEO} and SLA_{HAPTIC} higher than those of No MLB. Thus, it can be stated that balancing cell load in NS scenarios offloads congested cells only for slices accessing a significant number of PRBs. Even so, LHO+LB performance for slices 1 and 2 is still the worst among the tested MLB schemes.

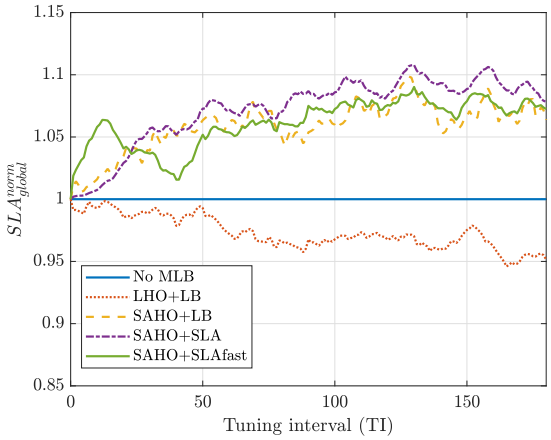


Fig. 5: Evolution of the overall SLA compliance in the scenario.

Regarding slice-aware algorithms, PRB utilization values in Table V reveal that the improvement in SLA_{global}^{norm} shown in Fig. 5 comes along with a higher usage of radio resources due to the fact that traffic is offloaded from congested to underutilized cells. SAHO+SLA shows the best SLA_{global}, with a final improvement of 8.78% in relative terms compared to No MLB (i.e., 72.61% vs. 66.75%), followed by SAHO+SLAfast, with a SLA_{global} improvement of 7.34% compared to No MLB. These results prove that $\overline{SLA}_{dif}(i,j,s)$ indicator is more powerful than PRB utilization ratio as a driver for MLB in sliced networks. Per-service SLA FoMs show that SAHO+LB is competitive to SAHO+SLA only for HAPTIC users served by slice 2, with SLA_{HAPTIC}≈79%. Although the tested algorithms provide significantly different HOM settings for this slice (shown in $\overline{|\delta HOM^{(14)}|}(2)$ values), the high TH and LR requirements lead to moderate SLA improvements in all cases, with a maximum SLA_{HAPTIC} increase of 2% in absolute terms compared to No MLB.

To understand how SLA-driven algorithms obtain the above results, Fig. 6 shows the Cumulative Distribution Function (CDF) of the final SLA compliance per cell in slice 1 for SAHO+SLA (solid line) and SAHO+SLAfast (dashed line), compared to No MLB (dotted line). Both MLB schemes show better SLA compliance in the worst cells at the expense

TABLE V: Performance comparison of MLB strategies in a NS scenario.

Slice	FoM	No MLB	LHO+LB	SAHO+LB	SAHO+SLA	SAHO+SLAfast
Global	SLA_{global} [%]	66.75	63.54	70.43	72.61	71.65
	$\overline{PRB_{util}}$ [%]	56.25	59.71	62.07	62.28	63.71
	nHO^{norm}	1	3.32	3.39	2.41	3.82
Slice 1 (FTP+VIDEO)	SLA_{FTP} [%]	39.16	45.41	50.95	57.37	59.07
	SLA_{VIDEO} [%]	56.52	58.87	65.71	68.09	67.95
	Avg. $\overline{SLA_{dif}}(i,j,1)$	0.39	0.36	0.29	0.25	0.25
	$\overline{\lvert\delta HOM^{(14)}\rvert}(1)$ [dB]	0	6.67	6.24	6.37	7.20
Slice 2 (HAPTIC)	SLA_{HAPTIC} [%]	77.67	78.80	79.02	79.04	79.76
	Avg. $\overline{SLA_{dif}}(i,j,2)$	0.29	0.29	0.26	0.25	0.25
	$\overline{\lvert\delta HOM^{(14)}\rvert}(2)$ [dB]	0	6.67	6.86	3.74	5.15
Slice 3 (DRIVING)	$SLA_{DRIVING}$ [%]	74.84	62.60	73.33	76.18	73.02
	Avg. $\overline{SLA_{dif}}(i,j,3)$	0.23	0.30	0.18	0.17	0.16
	$\overline{\lvert\delta HOM^{(14)}\rvert}(3)$ [dB]	0	6.67	6.88	4.94	7.34

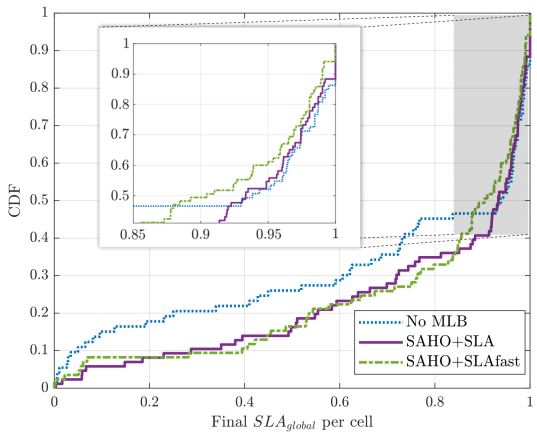


Fig. 6: Cumulative distribution of final SLA compliance per cell for slice 1 (FTP + VIDEO).

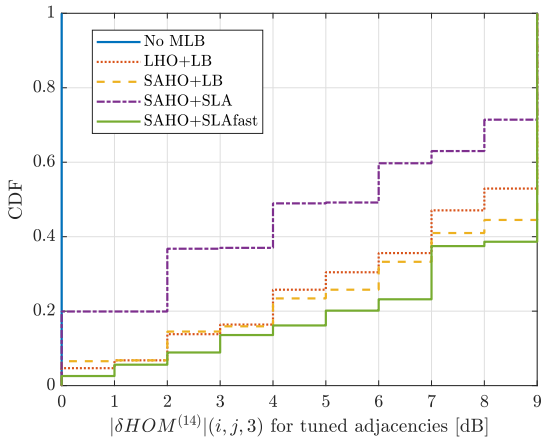


Fig. 7: Cumulative distribution of final HOM deviation from initial setting in tuned adjacencies for slice 3 (DRIVING).

of a slight performance degradation in the best cells. This behavior, also present in slices 2 and 3, is typical on self-tuning algorithms that balance a FoM across the scenario. In fact, according to average $\overline{SLA_{dif}}(i,j,s)$ figures in Table V, SAHO+SLA and SAHO+SLAfast provide the best equilibrium of SLA compliance in neighbor cells, with a relative reduction of 35.9%, 10.4% and 26.1% in $\overline{SLA_{dif}}(i,j,s)$ compared to No MLB for slices 1 to 3, respectively. Thus, balancing SLA compliance among cells on a per-adjacency-and-slice basis improves the overall system SLA compliance in NS scenarios.

When comparing SAHO+SLA and SAHO+SLAfast results in Table V, it is observed that HOM deviations reached at the end of the tuning process are significantly different, even if both schemes have the same goal (i.e., equalizing SLA compliance per slice between neighbor cells). The highest variation appears in slice 3, with a difference of 2.4 dB in $\overline{|\delta HOM^{(14)}|}(3)$ obtained with SAHO+SLAfast and SAHO+SLA. For a deeper analysis, Fig. 7 depicts the CDF of final absolute HOM deviation per adjacency in slice 3, $|\delta HOM^{(14)}|(i,j,3)$, in the tuned adjacencies for all the tested algorithms. It is observed that SAHO+SLA follows the most conservative tuning, leaving 20% of HOMs with the initial value. In contrast, SAHO+SLAfast performs the most aggressive HOM changes, with extreme HOM values in approximately 60% of adjacencies. According to Table V, the conservative strategy followed by SAHO+SLA turns into the lowest increment in HOs triggered due to traffic steering, with nHO^{norm}=2.41 (for all the remaining strategies, nHO^{norm}>3). Thus, SAHO+SLA causes the lowest signaling overload and likelihood of dropped connections due to failures in the HO process.

The distinct HOM set-up of SAHO+SLA and SAHO+SLAfast lead to a different final performance. SAHO+SLAfast only outperforms SAHO+SLA in more than 1% in absolute terms for FTP users, with SLA_{FTP} of 57.37% vs. 59.07% for SAHO+SLA and SAHO+SLAfast, respectively. Not shown in Table V is the fact that SAHO+SLAfast dramatically increases the number of HOs for slice 1 (nHO^{norm}=27.16), which does not pay off. On the contrary, SAHO+SLA outperforms SAHO+SLAfast in slice 3, with $SLA_{DRIVING}$ of 76.18% vs. 73.02% for SAHO+SLA and SAHO+SLAfast, respectively. More importantly, SAHO+SLAfast degrades performance for this slice compared to No MLB case. Actually, SAHO+SLA is

the only strategy outperforming the baseline for slice 3. The poor LHO+LB performance, due to a reduced PRB allocation per cell, has been discussed above. This problem should be solved by SAHO+LB, taking into account slice-specific PRB utilization measurements. However, DRIVING users have a bursty traffic profile consisting of small data chunks sent periodically that must be scheduled immediately. For a given cell-slice bandwidth, if data must be simultaneously transmitted to all DRIVING users, LR SLA may be violated even if PRB utilization remains low, since no data arrives to the transmission buffer until the next period. In contrast, if data bursts for DRIVING users in a cell must be transmitted at different time instants, average PRB utilization will be higher, but LR SLA is more likely be complied. Hence, PRB utilization is not representative of SLA compliance for slices with low TH but stringent LR requirements. Finally, to understand the bad performance of SAHO+SLAfast in slice 3, note that DRIVING users move fast during long connections (unlike the other considered services). Due to the larger distance traveled, their radio conditions are subject to a wider range of variability. Consequently, for these users, aggressive cell area changes caused by the simultaneous modification of several HOMs in SAHO+SLAfast can lead to very poor radio conditions that temporarily prevent data transmission. For services with high latency and reliability requirement such as DRIVING, not transmitting a single packet strongly impacts the level of SLA compliance.

The above results confirm that the slice-aware MLB algorithm with adjacency clustering proposed in this work (SAHO+SLA) is the best option to enhance the level of SLA compliance while equalizing end-user satisfaction across the scenario and keeping a low increase in the number of HOs due to traffic steering.

D. Execution time

In this work, experiments have been performed on a personal computer with Intel Core i7-8700 processor working at 3.2 GHz with a RAM of 16 GB. The simulation tool runs in Matlab 2022a. For the considered scenario (i.e., N_c=108 cells and N_n=6 relevant neighbors per cell), adjacency clustering takes 43.5 ms. Likewise, on average, the MLB process (excluding trace processing, not required in simulations) takes 4.8 ms per adjacency. These times are negligible for the proposed application, where decisions are taken on a second-scale resolution. In a live network, the time required to process connection traces and exchange data between network equipment must also be considered. To reduce delay, the computation of $\overline{SLA}(i,s)$ and $\overline{SLA}(j,s)$ (and the associated trace processing) can be parallelized.

VI. Conclusions

In 5G and beyond systems with network slicing, new slice-aware self-optimization solutions are required to guarantee SLA compliance. In this work, a novel slice-aware MLB algorithm has been proposed to adjust intra-frequency handover margins on an adjacency and slice basis with SLA criteria. Performance assessment has been carried out in a realistic simulator with slices serving traffic from eMBB and uRLLC services.

Results have shown the poor performance of slice-unaware MLB techniques in NS scenarios, especially for slices delivering services with a low data rate, neglected by legacy load balancing schemes. Moreover, the proposed algorithm has outperformed a slice-aware load-driven MLB scheme, showing the potential of the novel SLA-based indicator to drive the tuning process. Additionally, it has been proved that, even with the proper driver indicator, tuning parameters too often (every 5 s) and in many adjacencies simultaneously dramatically increases the number of HOs, leading to signaling overload and possible dropped calls. In 15 minutes of network activity, the proposed algorithm has improved the overall SLA compliance by up to 8% compared to the case of not performing any optimization. This improvement has been obtained with a significantly different final HOM set-up per slice.

The solution proposed in this paper improves SLA compliance by equalizing slice performance among cells. Results have shown good performance for different slice types in a realistic environment. However, the optimal handover settings from a SLA perspective may be different for some adjacencies and/or slices, which can be learned by a DRL agent. Moreover, such agents can be retrained to adapt to changes in the network affecting slice performance (e.g., change in capacity broker), leading to a faster (and maybe better) convergence. Thus, the design of DRL-based slice-aware traffic steering solutions is a promising research line. An interesting option is using a collaborative multi-agent approach, as done in [53] for the capacity broker, capturing slice peculiarities and inter-slice performance relationships. Nonetheless, in the DRL approach, some practical limitations must also be addressed that do not affect the algorithm proposed here, such as: a) the reluctance of network operators to randomly tune parameters by DRL, which may lead to unsafe network states during exploration stage, b) time complexity required for agent training and inference, determining the minimum tuning interval, and c) data privacy issues preventing tenants from sharing data with other tenants and/or with the central MANO. These issues may be alleviated by using federated learning [54].

Acknowledgment

This work has been funded by the Spanish Ministry of Science and Innovation (RTI2018-099148-B-I00 and PID2021-122217OB-I00) and Universidad de Málaga.

References

[1] M. Giordani, M. Polese, M. Mezzavilla, S. Rangan, and M. Zorzi, "Toward 6G networks: Use cases and technologies," *IEEE Communications Magazine*, vol. 58, no. 3, pp. 55–61, 2020.

[2] Z. Zhang, Y. Xiao, Z. Ma, M. Xiao, Z. Ding, X. Lei, G. K. Karagiannidis, and P. Fan, "6G wireless networks: Vision, requirements, architecture, and key technologies," *IEEE Vehicular Technology Magazine*, vol. 14, no. 3, pp. 28–41, 2019.

[3] S. Zhang, "An overview of network slicing for 5g," *IEEE Wireless Communications*, vol. 26, no. 3, pp. 111–117, 2019.

[4] NGMN Alliance, "Description of network slicing concept," in *NGMN 5G P1: Requirements Architecture Work Stream End-to-End Architecture*, 2016.

[5] 3GPP, "Management and orchestration; Concepts, use cases and requirements," in *TS 28.530*, version 17.1.0, 2021.

[6] A. Devlic, A. Hamidian, D. Liang, M. Eriksson, A. Consoli, and J. Lundstedt, "NESMO: Network slicing management and orchestration framework," in *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1202–1208, IEEE, 2017.

[7] J. Ramiro and K. Hamied, *Self-organizing networks: self-planning, self-optimization and self-healing for GSM, UMTS and LTE*. John Wiley & Sons, 2011.

[8] K. Lee, S. Kim, S. Lee, and J. Ma, "Load balancing with transmission power control in femtocell networks," in *13th International Conference on Advanced Communication Technology (ICACT2011)*, pp. 519–522, IEEE, 2011.

[9] S. Musleh, M. Ismail, and R. Nordin, "Load balancing models based on reinforcement learning for self-optimized macro-femto LTE-advanced heterogeneous network," *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 9, no. 1, pp. 47–54, 2017.

[10] E. Gures, I. Shayea, M. Ergen, M. H. Azmi, and A. A. El-Saleh, "Machine Learning Based Load Balancing Algorithms in Future Heterogeneous Networks: A Survey," *IEEE Access*, 2022.

[11] C. Gijón, M. Toril, S. Luna-Ramírez, and M. L. Marí-Altozano, "A data-driven traffic steering algorithm for optimizing user experience in multi-tier LTE networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 10, pp. 9414–9424, 2019.

[12] M. L. Marí-Altozano, S. Luna-Ramírez, M. Toril, and C. Gijón, "A QoE-driven traffic steering algorithm for LTE networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 11271–11282, 2019.

[13] M. L. M. Altozano, M. Toril, S. Luna-Ramírez, and C. Gijón, "A self-tuning algorithm for optimal QoE-driven traffic steering in LTE," *IEEE Access*, vol. 8, pp. 156707–156717, 2020.

[14] R. Fang, G. Chuai, and W. Gao, "Improve quality of experience of users by optimizing handover parameters in mobile networks," in *Proc. of the 4th International Conference on Computer Science and Application Engineering*, pp. 1–7, 2020.

[15] P. E. Iturria-Rivera and M. Erol-Kantarci, "QoS-Aware Load Balancing in Wireless Networks using Clipped Double Q-Learning," in *2021 IEEE 18th International Conference on Mobile Ad Hoc and Smart Systems (MASS)*, pp. 10–16, IEEE, 2021.

[16] P. E. Iturria-Rivera and M. Erol-Kantarci, "Competitive Multi-Agent Load Balancing with Adaptive Policies in Wireless Networks," in *2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC)*, pp. 796–801, IEEE, 2022.

[17] I. Da Silva, G. Mildh, A. Kaloxylos, P. Spapis, E. Buracchini, A. Trogolo, G. Zimmermann, and N. Bayer, "Impact of Network Slicing on 5G Radio Access Networks," in *2016 European conference on networks and communications (EuCNC)*, pp. 153–157, IEEE, 2016.

[18] R. Kwan, R. Arnott, R. Paterson, R. Trivisonno, and M. Kubota, "On mobility load balancing for LTE systems," in *IEEE 72nd Vehicular Technology Conference Fall (VTC-2010-Fall)*, pp. 1–5, 2010.

[19] A. Lobinger, S. Stefanski, T. Jansen, and I. Balan, "Load balancing in downlink LTE self-optimizing networks," in *IEEE 71st Vehicular Technology Conference (VTC-2010-Spring)*, pp. 1–5, 2010.

[20] M. M. Hasan, S. Kwon, and J.-H. Na, "Adaptive Mobility Load Balancing Algorithm for LTE Small-Cell Networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 4, pp. 2205–2217, 2018.

[21] J. M. Ruiz-Avilés, S. Luna-Ramírez, M. Toril, and F. Ruiz, "Traffic steering by self-tuning controllers in enterprise LTE femtocells," *EURASIP Journal on Wireless Communications and Networking*, vol. 2012, no. 1, p. 337, 2012.

[22] L. C. Gimenez, I. Z. Kovács, J. Wigard, and K. I. Pedersen, "Throughput-based traffic steering in LTE-Advanced HetNet deployments," in *IEEE 82nd Vehicular Technology Conference (VTC-2015-Fall)*, pp. 1–5, 2015.

[23] P. Muñoz, D. Laselva, R. Barco, and P. Mogensen, "Dynamic traffic steering based on fuzzy q-learning approach in a multi-RAT multi-layer wireless network," *Computer Networks*, vol. 71, pp. 100–116, 2014.

[24] C. A. S. Franco and J. R. B. de Marca, "Load balancing in self-organized heterogeneous LTE networks: A statistical learning approach," in *2015 7th IEEE Latin-American Conference on Communications (LATINCOM)*, pp. 1–5, IEEE, 2015.

[25] P. Muñoz, R. Barco, J. M. Ruiz-Avilés, I. De La Bandera, and A. Aguilar, "Fuzzy rule-based reinforcement learning for load balancing techniques in enterprise LTE femtocells," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 5, pp. 1962–1973, 2012.

[26] P. Muñoz, R. Barco, and I. de la Bandera, "Optimization of load balancing using fuzzy Q-learning for next generation wireless networks," *Expert Systems with Applications*, vol. 40, no. 4, pp. 984–994, 2013.

[27] S. S. Mwanje and A. Mitschele-Thiel, "A Q-learning strategy for LTE mobility load balancing," in *2013 IEEE 24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 2154–2158, IEEE, 2013.

[28] J. M. Ruiz-Aviles, M. Toril, S. Luna-Ramírez, V. Buenestado, and M. Regueira, "Analysis of limitations of mobility load balancing in a live LTE system," *IEEE wireless communications letters*, vol. 4, no. 4, pp. 417–420, 2015.

[29] K. Attiah, K. Banawan, A. Gaber, A. Elezabi, K. Seddik, Y. Gadallah, and K. Abdullah, "Load balancing in cellular networks: A reinforcement learning approach," in *2020 IEEE 17th Annual Consumer Communications & Networking Conference (CCNC)*, pp. 1–6, IEEE, 2020.

[30] Q. Liao, T. Hu, and D. Wellington, "Knowledge Transfer in Deep Reinforcement Learning for Slice-Aware Mobility Robustness Optimization," *arXiv preprint arXiv:2203.03227*, 2022.

[31] H. P. Phyu, D. Naboulsi, and R. Stanica, "Machine learning in network slicing-a survey," *IEEE Access*, 2023.

[32] B. Mareri, G. O. Boateng, R. Ou, G. Sun, Y. Pang, and G. Liu, "MANTA: Multi-Lane Capsule Network Assisted Traffic Classification for 5G Network Slicing," *IEEE Wireless Communications Letters*, vol. 11, no. 9, pp. 1905–1909, 2022.

[33] H. P. Phyu, D. Naboulsi, and R. Stanica, "Mobile traffic forecasting for network slices: A federated-learning approach," in *2022 IEEE 33rd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pp. 745–751, 2022.

[34] G. Dandachi, A. De Domenico, D. T. Hoang, and D. Niyato, "An artificial intelligence framework for slice deployment and orchestration in 5G networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 2, pp. 858–871, 2019.

[35] H. Esmat and B. Lorenzo, "Deep reinforcement learning based dynamic edge/fog network slicing," in *IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, 2020.

[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, p. 102573, 2020.

[37] G. Sun, G. O. Boateng, D. Ayepah-Mensah, G. Liu, and J. Wei, "Autonomous resource slicing for virtualized vehicular networks with D2D communications based on deep reinforcement learning," *IEEE Systems Journal*, vol. 14, no. 4, pp. 4694–4705, 2020.

[38] W. Guan, H. Zhang, and V. C. Leung, "Customized slicing for 6G: Enforcing artificial intelligence on resource management," *IEEE Network*, vol. 35, no. 5, pp. 264–271, 2021.

[39] ITU-R, "IMT Vision – Framework and overall objectives of the future development of IMT for 2020 and beyond," in *Recommendation M.2083*, 2015.

[40] P. A. S. Ordóñez, S. Luna-Ramírez, and M. Toril, "A Computationally Efficient Method for QoE-Driven Self-Planning of Antenna Tilts in a LTE Network," *IEEE Access*, vol. 8, pp. 197005–197016, 2020.

[41] C. Gijón, M. Toril, M. Solera, S. Luna-Ramírez, and L. R. Jiménez, "Encrypted Traffic Classification Based on Unsupervised Learning in Cellular Radio Access Networks," *IEEE Access*, vol. 8, pp. 167252–167263, 2020.

[42] C. Gijón, M. Toril, and S. Luna-Ramírez, "Data-Driven Estimation of Throughput Performance in Sliced Radio Access Networks via Supervised Learning," *IEEE Transactions on Network and Service Management*, 2022.

[43] 3GPP, "New Radio (NR); Physical Layer procedures for data," in *TS 38.214*, version 16.6.0, 2021.

[44] K. Brueninghaus, D. Astely, T. Salzer, S. Visuri, A. Alexiou, S. Karger, and G.-A. Seraji, "Link performance models for system level simulations of broadband radio access systems," in *16th International Symposium on Personal, Indoor and Mobile Radio Communications*, vol. 4, pp. 2306–2311, IEEE, 2005.

[45] M. L. Marí-Altozano, S. S. Mwanje, S. Luna-Ramírez, M. Toril, H. Sanneck, and C. Gijón, "A service-centric Q-learning algorithm for mobility robustness optimization in LTE," *IEEE Transactions on Network and Service Management*, 2021.

[46] Y. Singh, "Comparison of Okumura, Hata and COST-231 models on the basis of path loss and signal strength," *International journal of computer applications*, vol. 59, no. 11, 2012.

[47] 3GPP, "Evolved Universal Terrestrial Radio Access (E-UTRA); Base Station (BS) radio transmission and reception," in *TS 36.104*, version 15.2.0, 2018.

[48] J.-H. Rhee, J. M. Holtzman, and D.-K. Kim, "Scheduling of real/non-real time services: adaptive EXP/PF algorithm," in *The 57th IEEE Semiannual Vehicular Technology Conference, 2003 (VTC-2003-Spring)*, vol. 1, pp. 462–466, IEEE, 2003.

[49] M. Abu-Tair and A. Marshall, "An empirical model for multi-contact point haptic network traffic," in *Proc. of the 2nd International Conference on Immersive Telecommunications*, pp. 1–6, 2009.
[50] O. Nassef, L. Sequeira, E. Salam, and T. Mahmoodi, "Building a lane merge coordination for connected vehicles using deep reinforcement learning," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2540–2557, 2020.
[51] O. Sallent, J. Perez-Romero, R. Ferrus, and R. Agusti, "On radio access network slicing from a radio resource management perspective," *IEEE Wireless Communications*, vol. 24, no. 5, pp. 166–174, 2017.
[52] P. Popovski, J. J. Nielsen, C. Stefanovic, E. De Carvalho, E. Strom, K. F. Trillingsgaard, A.-S. Bana, D. M. Kim, R. Kotaba, J. Park, *et al.*, "Wireless access for ultra-reliable low-latency communication: Principles and building blocks," *IEEE Network*, vol. 32, no. 2, pp. 16–23, 2018.
[53] I. Vilà, J. Pérez-Romero, O. Sallent, and A. Umbert, "A novel approach for dynamic capacity sharing in ulti-tenant scenarios," in *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*, pp. 1–6, IEEE, 2020.
[54] B. Brik and A. Ksentini, "On predicting service-oriented network slices performances in 5G: A federated learning approach," in *2020 IEEE 45th Conference on Local Computer Networks (LCN)*, pp. 164–171, IEEE, 2020.

Carolina Gijón received her B.Sc. degree in Telecommunication Systems Engineering and her M.Sc. Degree in Telecommunication Engineering from the University of Málaga, Spain, in 2016 and 2018, respectively. Since 2017, she is a research assistant in the Communications Engineering Department of the University of Málaga, where she received the Ph.D. degree in 2023. In 2021, she was visiting researcher at King's College London. Her research interests include self-organizing networks, machine learning and radio resource management.

Toktam Mahmoodi (Senior Member, IEEE) received the B.Sc. degree in electrical engineering from the Sharif University of Technology, Iran, in 2002, and the Ph.D. degree in telecommunications from King's College London, U.K., in 2009, where she is currently the Head of the Centre for Telecommunications Research with the Department of Engineering. Her research interests include mobile communications, network intelligence, and low-latency networking.

Matías Toril received his M.S in Telecommunication Engineering and the Ph.D degrees from the University of Málaga, Spain, in 1995 and 2007 respectively. Since 1997, he is Lecturer in the Communications Engineering Department, University of Málaga, where he is currently Full Professor. He has co-authored more than 130 publications in leading conferences and journals and 8 patents owned by Nokia or Ericsson. His current research interests include self-organizing networks, radio resource management and data analytics.

Salvador Luna-Ramírez received his M.S in Telecommunication Engineering and the Ph.D degrees from the University of Málaga, Spain, in 2000 and 2010, respectively.Since 2000, he has been with the department of Communications Engineering, University of Málaga, where he is currently Associate Professor. His research interests include self-optimization of mobile radio access networks and radio resource management.

Juan L. Bejarano-Luque received the B.S. degree in Telecommunications Engineering, the M.S. degree in Acoustic Engineering and the Ph.D. degree in Telecommunication Engineering from the University of Málaga, Málaga, Spain, in 2015 2016 and 2023, respectively. His research interests include optimization of radio resource management for mobile networks, location-based services and management, and data analytics and machine learning for networks management.