A service-centric Q-learning algorithm for mobility robustness optimization in LTE

María Luisa Marí-Altozano[1], Stephen S. Mwanje[2], Salvador Luna-Ramírez[1], Matías Toril[1], Henning Sanneck[2], Carolina Gijón[1]
Email: {mlma[1], sluna[1], mtoril[1], cgm[1]}@ic.uma.es, {stephen.mwanje[2], henning.sanneck[2]}@nokia-bell-labs.com
[1]Department of Communication Engineering, University of Málaga, 29071, Málaga, Spain.
[2]Nokia Bell Labs, Munich, Germany.

***Abstract*—Due to the diversity of mobile services and rising user expectations, mobile network management has changed its focus from Quality of Service (QoS) to Quality of Experience (QoE). As a consequence, classical network optimization procedures must be updated accordingly. One of these optimization procedures is Mobility Robustness Optimization (MRO), whose aim is to improve HandOver (HO) performance by reducing HO failures. In this work, a novel QoE-aware MRO algorithm is proposed considering a multi-service scenario. Unlike previous approaches, whose aim is to increase successful handover rates, the optimization aim in this work is two-folded: to improve cell edge QoE while improving successful handover rates in the whole network. For this purpose, the handover trigger point, defined by the pair of HO control parameters HO margin and Time to Trigger, are tuned on a per-adjacency basis according to QoE and HO failure measurements. Method assessment is based on a dynamic system-level simulator implementing a realistic LTE scenario with multiple services. Results show that the proposed QoE-aware MRO algorithm improves cell edge QoE throughout the network while increasing the percentage of successful handovers compared to traditional approaches.**

***Index Terms*—Long Term Evolution, self organizing networks, neural network, Q-learning, quality of experience.**

I. Introduction

The size and complexity of current mobile communication networks makes it very difficult for cellular operators to manage their networks. Such a problem will increase in future 5G systems due to terminal and service diversity. Thus, network management will remain as one of the more challenging tasks in cellular networks for the coming years. To deal with mobile network complexity, Self-Organization Network (SON) frameworks are developed, which make the most of network data assets by self-configuration, self-optimization and self-healing features [1] [2].

One of the most important processes to manage in a mobile network is the HandOver (HO) procedure. HO is in charge of providing mobile users with seamless connectivity while they move across the network. If HO parameters are not well configured, instabilities arise, causing unnecessary HOs (Ping-Pong effect, PP) or Radio Link Failures (RLF) (i.e., too early/late HOs). To avoid these issues, Mobility Robustness Optimization (MRO) is a self-optimization feature that automatically tunes HO parameters to improve HO performance (i.e., minimize PP and RLF) [3].

Increased terminal and network capabilities have raised mobile user expectations. These changes will continue in the coming years with the deployment of 5G systems, which will introduce new appealing use cases [4]. Such changes have forced operators to shift their focus from network performance to end user opinion (a.k.a. *Quality of Experience*, QoE). QoE is defined as the overall satisfaction of a service as subjectively perceived by the user [5].

With recent advances in information technologies, big data analytics can be used for automated QoE management [6] [7]. In this context, Machine Learning (ML) algorithms can help to convert network data into actionable insights. Reinforcement Learning (RL) and Artificial Neural Networks (ANN) are two of the most popular ML tools used for self-optimization in mobile networks [8]. With these techniques, legacy reactive self-tuning algorithms, based on threshold comparison and simple heuristic rules, can be substituted by intelligent schemes that autonomously find the best tuning policies and proactively change network settings to quickly adapt to changing environments [9]. To this end, ML techniques have already been applied to MRO [10] [11] [12] [13] [14]. However, even if these advanced schemes can potentially improve user QoE, to the best of authors' knowledge, no MRO algorithm in the literature explicitly takes QoE into account.

In this work, a novel QoE-aware ML-based MRO algorithm is proposed for LTE systems. This work is a follow up of [11]. As in [11], the proposed adaptive HO scheme aims to reduce the number of PP and RLF in the network by displacing HO trigger location through HO Margin (HOM) and Time-To-Trigger (TTT) parameter modifications, widely used in many previous works. Likewise, adaptation is achieved by combining well-known RL and ANN techniques. However, unlike previous works, the scheme proposed here adds QoE concerns to MRO by also considering the QoE of cell edge users affected by the HO process as an input. The resulting scheme is validated in a dynamic system-level simulator implementing a realistic macrocellular LTE scenario.

The rest of the work is organized as follows. Section II reviews related work. Section III discusses the limitations of classical MRO schemes in terms of QoE. Section IV describes the system model used in this work. Section V describes the proposed QoE MRO algorithm, Section VI presents algorithm assessment and, finally, Section VII summarizes the main conclusions.

II. Related work

Previous studies on MRO have proven that HO performance can be improved with many different techniques. A first set of approaches [15] [16] [17] [18] [19] use an analytical model to find the optimal HO parameter settings by formulating the tuning problem as a multi-objective optimization problem, whose objective function includes the number of HOs per call, the outage probability, the cell-edge spectral efficiency or RLF rates. A second set of approaches use self-tuning methods during network operation to adjust the parameters of an existing HO scheme based on threshold crossing [20] [21]. Changes are made by iterative control algorithms driven by heuristic rules taken from expert knowledge, which can be adapted to outdoor [20] or indoor [21] environments. Such an approach can be improved by changing HO parameters in a proactive manner based on RLF prediction [22]. However, with heuristic rules, it is not guaranteed that optimal HO settings are reached in steady state. A third set of approaches redesign the HO scheme that process instantaneous signal measurements to decide when to trigger a HO for each individual user [13] [11]. These adaptive HO schemes constantly improve by analyzing their past behavior, avoiding the need for an expert and becoming a powerful tool for network optimization. Their main drawback is that they require updating vendor equipment, whereas the analytical and self-tuning approaches can still be used with existing infrastructure.

Several works have included QoE aspects when optimizing cellular network parameters, differing in the decision variables tuned, performance indicators that drive the tuning process and/or network models. As an example, in [23] [24], the authors define a self-tuning algorithm for a classical multi-service packet scheduler aiming to balance and optimize QoE across services by re-prioritizing users in a LTE cell. In [25] a 5G-QoE framework is proposed in order to adapt UHD (*Ultra High Definition*) video flows in a QoE manner. Alternatively, other schemes aim to improve user QoE through Mobility Load Balance (MLB) techniques. In [26] and [27] two QoE-based MLB approaches are proposed. The aim of the former is to reach cell QoE balance throughout a LTE network using a fuzzy-based QoE-driven algorithm, while the aim of the latter is to reach maximum overall system QoE using an ascent gradient algorithm. Likewise, in [28], a novel indicator derived from connection traces is developed to drive the tuning of inter-system handover parameters to optimize QoE in a multi-carrier LTE network. In the context of MRO, it is clear that a bad configuration of HO parameters not only degrades global network performance but also individual user experience [29]. A survey of machine learning techniques applied to self-organizing cellular networks is presented in [8]. MLB is one of the first SON use cases where machine learning has been applied. A RL algorithm based on Q-learning (QL) is presented in [30] to adapt a fuzzy logic controller for adjusting HO margins to balance the load between cells with heuristic rules by properly selecting consequents in the fuzzy inference engine. In [31], QL is used to find the best step for tuning HO margins. More sophisticated approaches combine RL and ANN with multiple layers (a.k.a., deep RL) to build adaptive MLB schemes that find the optimal MLB policy in complex system states by taking advantage of the generalization capability of ANN [32] [33]. ML techniques have also been extensively used for MRO. In [13], QL is used to adaptively change parameters in a classical HO scheme to reduce PP and call dropping. Likewise, [12] uses QL to update fuzzy rules to reduce the number of unsuccessful HOs and call dropping ratio by adjusting only HOM, while TTT is fixed. Similarly, an adaptive HO scheme based on QL is proposed in [11] to reduce RLF and PP by explicitly including both indicators as inputs to the learning process. Thus, it is possible to find the best HO settings per cell depending on user speed in the area. More sophisticated adaptive HO schemes with MRO are implemented with ANN. For instance, in [10], the conventional hysteresis rule is substituted by an ANN that performs received signal power pattern recognition and decides whether or not perform HO in the hope that the probability of HO failure is reduced. In [14], ANN is used to build a network performance model relating carried traffic, signal strength, signal quality and call dropping/blocking ratios, which can then be used to lead operator actions. However, none of these ML-based MRO schemes is driven by QoE issues. This work proposes an evolution of the legacy QL-based MRO algorithm described in [11]. The latter updates the Q-table based only on early, late and ping-pong handover counters. In contrast, a first variant of the algorithm proposed here explicitly considers QoE as a key handover performance metric. Moreover, the contribution is enriched with the proposal of a second variant of the proposed algorithm, which combines QL with an ANN.

The main contributions of this work are: a) uncovering the limitations of traditional MRO schemes from a QoE perspective, b) the inclusion of QoE criteria in an adaptive HO scheme to increase user QoE at cell edge while decreasing RLF and PP by modifying handover margins and Time To Trigger, and c) the validation of the algorithm via simulations in a realistic macrocellular LTE scenario.

III. Problem formulation

The HO process ensures a seamless connection between neighbor cells when the user moves. In the basic scheme, known as power budget HO, a HO is triggered at time t_0 when

$$\mathrm{P}_{rx}(j) - P_{rx}(i) \geq HOM(i,j) \ \ \forall t \in [t_0 - TTT(i,j), t_0]\,, \quad (1)$$

where $P_{rx}(j)$ is the pilot signal level received from neighbor cell j, $P_{rx}(i)$ is the pilot signal level received from the serving cell i, and $HOM(i,j)$ and $TTT(i,j)$ are the HO margin and Time-To-Trigger from cell i to j. Both HOM and TTT are defined on a per-adjacency basis.

A. Handover performance events

If HOM and TTT are not well configured, one of the following events can occur.

1) RLF Too Late HandOver (LHO): it happens when a user is moving from a serving cell i to a target cell j, but HO is triggered too late (or even not triggered at all). In this case, the pilot signal level from cell i drops below a certain threshold during a specific time window and a RLF occurs in cell i. As a result, the user is disconnected from cell i and reconnected to cell j some time later.
2) RLF Too Early HandOver (EHO): it takes place when a user is moving from a serving cell i to a target cell j, but HO is triggered too early. After HO, the pilot signal level from cell j drops below a certain threshold during a specific time window and a RLF occurs. As a result, the user is disconnected from cell j and reconnected to cell i (or another cell).
3) Ping-Pong (PP): when a user is moving from a serving cell i to a target cell j, but HO is triggered early, the corresponding HO algorithm will try to hand over the user back to cell i within a particular time window after the first HO took place. This is an unnecessary HO causing useless increase in signaling load.

Please note that time windows for RLFs and PP are different from $TTT(i,j)$ window. HO-related timers and thresholds are defined by vendors to classify a HO as a LHO, EHO or PP [34]. In this work, it is assumed that a RLF is detected by the user equipment when the Reference Signal Received Power (RSRP) is below -100 dBm for 200 ms (timer T310 [34]). Likewise, the time window for considering two consecutive HOs as PP is set to 5000 ms (timer T311 [34]).

LHO, EHO and PP are mutually exclusive. If a HO is performed and none of the above occurs, HO is considered successful (denoted as SHO). Only A3 event-based HOs are considered in this work [35].

B. Mobility Robustness Optimization

MRO schemes modify HOM and TTT values to maximize the ratio of SHO (or, conversely, minimize LHO, EHO and PP). Ideally, the optimal configuration should ensure that HO occurs at that point where RSRP from the serving and target cells are comparable and both significantly good. This is achieved with low values of HOM and TTT. In practice, a hysteresis level must be enforced to avoid instabilities due to rapid fluctuations of propagation conditions. When tuning HOM and TTT, a trade-off exists between LHO, EHO and PP. The lower HOM and TTT, the faster the HO is triggered, ensuring that the user is always connected to the best cell, but the more likely that a EHO and PP occurs. In contrast, the larger HOM and TTT, the longer the HO delayed, avoiding EHO and PP, but the user remains connected to the source cell offering worse signal level than the target cell, which temporarily degrades link performance and might end up to a LHO.

C. Need for QoE-awareness

The above-mentioned signal impairments may decrease user throughput and increase packet delay, especially at cell edge, but these changes may not be perceived by the user. This fact points out that there is not a direct relationship between radio link performance and QoE [26]. Thus, minimizing HO failures and PP does not necessarily lead to an increase in the QoE of handed-over users. For the same reason, HO settings achieving optimal HO performance might not lead to the best overall user QoE at cell edge. Moreover, services are not affected the same when radio performance is degraded, thus requiring differentiated HO settings. Correspondingly, with these as the main hypotheses, this paper proposes means to learn the HO settings that are optimal not only for link robustness, but that concurrently maximize the QoE both before and after the handover.

IV. System model

This section outlines the traffic and QoE models of the mobile services covered in this work.

A. Traffic models

Four services have been considered in this work: progressive video streaming (VIDEO), file download service via File Transfer Protocol (FTP), web browsing (WEB) [36], and Voice over Internet Protocol (VoIP). Table I shows their main characteristics [26]. VIDEO, FTP and WEB are non-guaranteed bit rate (non-GBR) services. In contrast, VoIP is a guaranteed bit rate (GBR) service with low data rates. The Video service model (inspired in [37]) corresponds to buffered live video streaming with fixed quality (720 p) and variable bit rate. For this purpose, a simple model of the player's buffer at the client side is implemented. In live video streaming, content generation and playback request occur at the same time (unlike video on demand, where the whole content is available at the start of the session); thus, the video server starts sending frames to the client as they are generated, and frames are stored in the client buffer until reaching a minimum video content (3 seconds in this work). This is modeled as a fixed video playback start delay (i.e., initial buffering time). If the video buffer runs out of content during playback, the video stops (i.e., a stalling event takes place) and the player waits until the buffer is re-filled again. Obviously,

TABLE I: Traffic model parameters [26].

Service	Main features
VIDEO	H.264/MPEG-4 AVC VBR (*Variable bit rate*) 720p resolution, 25 frames per second. Video duration: uniform distribution between 0 and 540 s. Frame size according to real traces (avg. 9.2 MB). Connection dropped when stalling lasts for twice the video duration. $\lambda_{VIDEO} = 4 \cdot 10^{-3}$.
FTP	File size: log-normal distribution (avg. 20 MB) [36]. $\lambda_{FTP} = 2.5 \cdot 10^{-3}$.
WEB	Web page size: log-normal distribution (avg. 20 MB). No. pages per session: log-normal (avg. 4). Waiting time: exponential distribution (avg. 107 s) [36]. $\lambda_{WEB} = 3.7 \cdot 10^{-3}$.
VoIP	Coding rate 16 kbps Session time: exponential distribution (avg. 60 s). Call dropped after 1 s without resources. $\lambda_{VoIP} \simeq 0$.

videos of less than 3 s do not experience stalling. The duration of the Video sequence follows a uniform distribution between 0 and 540 s. Frame sizes are taken from a real H.264 video trace [38]. A video session drop criterion is also modeled, where the connection is terminated if session time is more than twice the video content duration. The other two data services FTP and WEB are best-effort services. FTP is a file download service and WEB consists of downloading several web pages with different sizes with a random reading time between them. VoIP is modeled to generate 20 bytes of voice every 10 ms, with a bit rate of 16 kbps.

Traffic connections are modeled as data bursts, and, therefore, new connections follow a Poisson distribution for any service [39] [40]. The Poisson distribution parameter, λ is shown in Table I for each service.

B. *QoE models*

QoE is measured using the *Mean Opinion Score* (MOS) scale, ranging from 1 (bad) to 5 (excellent). In the absence of surveys, QoE can be estimated from QoS measurements. For this purpose, QoE figures are obtained from QoS measurements gathered per-session by means of utility functions [41]. In the context of mobile networks, a utility function describes the relationship between key objective QoS performance indicators taken directly from the network and the subjective QoE perceived by the users of a service. Utility functions are service based (i.e, similar network performance usually leads to different service's quality perception), and provide an estimate of the user QoE, even if they miss contextual factors (e.g., social environment, location, time of day, $\cdots$). Thus, network operators can estimate user QoE by processing passive measurements of key performance indicators from individual connections [26].

Different utility functions are defined for each service. VIDEO utility function is defined as [23]:

$$QoE^{(VIDEO)} = 4.23 - 0.0672L_{ti} - 0.742L_{fr} - 0.106L_{tr} \ , \quad (2)$$

where $QoE^{(VIDEO)}$ is the MOS estimated for the video connection, L_{ti} denotes the initial buffering time (in seconds), L_{fr} is the average stalling frequency (s^{-1}) (i.e., number of times per second that the video player is paused due to a stalling event), and L_{tr} is the average stalling duration (in seconds) for the user connection under consideration. As observed in (2), the maximum QoE value for a video connection is upper limited to 4.23.

The utility function for FTP service characterized as [42]

$$QoE^{(FTP)} = \max(1, \min(5, 6.5 \cdot TH - 0.54)) \ , \quad (3)$$

where TH is the average user throughput in Mbps.

For WEB service, user QoE can be estimated as [42]

$$QoE^{(WEB)} = 5 - \frac{578}{1 + (\frac{TH+541.1}{45.98})^2} \ , \quad (4)$$

where TH is the average user throughput in kbps. Note that, $\max(QoE^{(WEB)}) = 5$. No dropping of web connections is implemented, therefore low MOS values for WEB are reached when TH is zero (i.e., $QoE^{(WEB)} = 1$ when $TH \simeq 0$ kbps).

Finally, The utility function for VoIP service is modeled as [43]

$$QoE^{(VoIP)} = 1 + 0.035R + R(R-60)(100-R)7 \cdot 10^{-6}, \quad (5)$$

where $QoE^{(VoIP)}$ is the MOS value for a VoIP connection, and R is a parameter representing the connection quality, with values from 0 (minimum) to 93 (maximum), that only depend on the mouth-to-ear delay experienced by VoIP packets. Note that $\max(QoE^{(VoIP)}) = 4.4054$ (when $R = 93$), i.e., MOS never reaches the highest value of 5, showing that even with the best possible network performance, some individuals may not perceive their experience as excellent. Likewise, QoE is set to the minimum (i.e., $QoE^{(VoIP)} = 1$) if the connection is dropped.

Note that the above-described QoE models do not depend on traffic model parameters (e.g., web page size, file size or video sequence duration) but on connection performance indicators.

V. QoE-aware MRO algorithm

In this section, a new QL-based QoE-aware MRO algorithm is presented. For clarity, the Q-learning framework is introduced first. Then, the baseline QL MRO algorithm only driven by HO performance indicators is explained, hereafter denoted as Quality-MRO (Q-MRO) [11]. Finally, the proposed algorithm including QoE criteria, referred to as Experience MRO (E-MRO), is described.

A. Q-Learning framework

QL is a model-free RL algorithm to solve learning problems. It is selected here due to its ability to learn and improve system performance through experience. A QL problem is defined by the triplet (X, A, r), where X and A are the sets of all possible system states and actions, respectively, and $r : X \cdot A \to R$ is the *reward* function, representing the reward (i.e., performance improvement) obtained by executing each action in a given system state. In this work, a state is defined by the combination of variables determining the specific radio environment of each HO event (e.g., user speed, cell load, interference...). Likewise, an action is a particular setting of HO parameters. Both states and actions can only take discrete values.

Every time some event n occurs at time t_n, a QL (a.k.a. learning) agent checks the benefit from executing some action $a_{t_n} \in A$ with the system in state $x_{t_n} \in X$ by computing the associated reward, $r(x_{t_n}, a_{t_n})$. From these observations, the learning agent aims to choose those actions that maximize its cumulative rewards over time. To this end, the learning agent uses a greedy policy π that will explore all possibilities and find the best action to take at every moment so as to maximize rewards along time. For this purpose, a value function, $Q(x, a)$, showing the expected reward from action a in state x is defined as

$$Q(x, a) = E_\pi \left[r(x_{t_n}, a_{t_n}) \right] . \qquad (6)$$

To infer $Q(x, a)$, a Q-table is constructed with as many rows as states and as many columns as actions. The aim of Q-table is to store and update Q-values, showing system performance for every state-action pair (x, a) experienced across time (i.e., for all events n). Reward values in the Q-table are updated everytime an event n (i.e., a HO) takes place at t_n by using the Bellman equation, as

$$Q^{(n+1)}(x, a) = (1 - \alpha)Q^{(n)}(x, a) + \alpha r(x_{t_n}, a_{t_n}) , \qquad (7)$$

where α is the learning rate. The learning speed of the Q-table is controlled byα. A high value of α gives more weight to instantaneous rewards (i.e., to the recent HO performance), which can easily lead to algorithm's divergence. In contrast, a low value of α makes QL rely more on previous cumulative rewards (i.e., aggregated HO performance for a period of time) than on instantaneous rewards, which improves system convergence. Note that superscript n denotes successive versions of the Q-table after every new event. At the end of the convergence process, the Q-table is considered as the Q function defined in (6), and the best action for every state, $a_{max}(x)$, can easily be determined as

$$a_{max}(x) = \max_a \left(Q^{(n_{end})}(x, a) \right) \quad , \qquad (8)$$

which is the aim of the QL process. Superscript n_{end} reflects the last event, when the update process has ended and no more events are considered.

The QL optimization algorithm is an iterative scheme, where actions, states and rewards are collected for some time, and best actions are then selected and applied onwards. This process is repeated until some convergence criterion for the best rewards, $a_{max}(x)$, is fulfilled. Such an iterative scheme requires a time division in slots, hereafter referred to as Action Intervals (AI). During an AI, (a, x, r) values are collected for every event and the Q-table is updated accordingly every time an event takes place. Once the AI ends, the best action for every state, $a_{max}(x)$, is selected and used for the next AI. Thus, an AI index, n_{AI}, denotes the iteration index. In this work, AI consists of 30 seconds of network time. Note that index n in (7) reflects events, while n_{AI} indicates the time slot when the best actions are calculated. Thus, a best action per state can be defined for every AI, $a_{max}(x, n_{AI})$.

At the beginning of the optimization process, when just a few events have happened, network performance may not be representative of the global system behavior. To ensure an adequate search trajectory, at the end of each AI, the learning agent should also explore other actions, $a(x, n_AI)$, even if those do not lead to the maximum value in the Q-table (i.e., $a(x, n_{AI}) \neq a_{max}(x, n_{AI})$). The dilemma of how much effort to spend on testing new actions (probably suboptimal) or take those already explored (getting high reward) is known as the *exploration-exploitation* trade-off [44]. Such a trade-off is controlled by a decaying Epsilon-greedy (ε-greedy) policy that indicates to the learning agent how much to explore and how much to exploit, which evolves with time. Figure 1 illustrates a typical ε evolution with n_{AI}, where an initial exploration stage is configured, and, after a transition time, an exploitation stage is reached. In this decaying ε-greedy policy, $\varepsilon = 1$ indicates that the learning agent always selects new options for the next AI (i.e., $a_{max}(x)$ from Q-table is not considered at all). This value is used at the beginning of the optimization process for a certain number of AIs, $N_{explore}$ AIs. After that exploration time, ε starts decreasing with time, causing that only some random actions are selected with ε probability. Otherwise, the $a_{max}(x, n_{AI})$ actions are selected. This ε decreasing stage lasts for N_{train} AIs. Finally, in the exploitation stage, $\varepsilon = 0$ and the learning agent only exploits the best actions $a_{max}(x, n_{AI})$ calculated from the Q-table. Such an exploitation of the best actions, with no further testing of new actions, can only be done when network conditions do not change. In a live scenario, ε should never reach 0, but keep a relatively low value to react to changes in the network by exploring new actions. In this work, the length of this stage, $N_{exploit}$, is controlled to ensure that enough measurements are collected to assess the performance of the tested algorithms.

The basic structure of a generic QL scheme is summarized in Algorithm 1. The main loop represents iteration across AIs. In each iteration, ε is first updated, following the strategy shown in Figure 1. Secondly, information (i.e., actions and states) from all events in the n_{AI} iteration are collected and their rewards $r(x_{t_n}, a_{t_n})$ are calculated. With

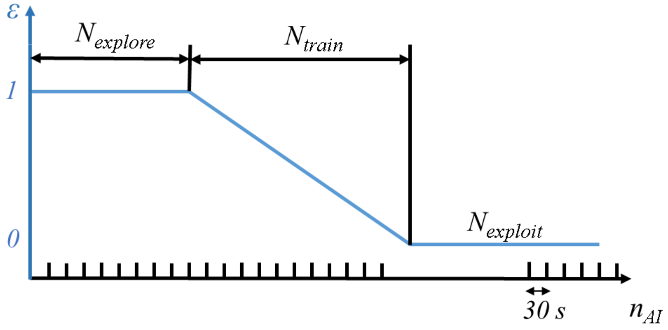

Fig. 1: Evolution of ε parameter in the QL scheme.

Algorithm 1 Structure of Q-MRO algorithm.

```
Input: states $x_{t_n}$ and actions $a_{t_n}$ $\varepsilon(n_{AI})$
Output: actions for next iteration $a(x, n_{AI}+1)$
For $n_{AI} = 1, 2, ...$
  calculate $\varepsilon(n_{AI})$
  FOR each event $n$ occurring in $t_n \in n_{AI}$
    calculate $r(x_{t_n}, a_{t_n})$ with (9)
    update element in Q-table $Q^{(n+1)}(x_{t_n}, a_{t_n})$ with (7)
  end
  FOR every $x \in X$
    IF rand () > $\varepsilon(n_{AI})$
      $a(x, n_{AI}+1)$ (8) = $a_{max}(x, n_{AI}+1)$ (8)
      $= \max_a \left(Q^{(n)}(x, a)\right)$
    else
      select a random action $a$ for $a(x, n_{AI}+1)$ (8)
    end
  end
  $n_{AI} = n_{AI} + 1$
end
```

these rewards, Q-table is updated following (7). Finally, depending on the ε value, actions for the next iteration are decided. Note that this structure is shared by all MRO algorithms tested in this work. Algorithms differ in the definition of the reward function, r, as will be presented in next sections.

B. Q-MRO algorithm

Q-MRO is a classical implementation of MRO using a Q-learning scheme as the one explained in Section V-A. The aim of Q-MRO is to reduce the number of EHO, LHO and PP by tuning HOM and TTT parameters in the HO process. This algorithm was already proposed in [11] and it is used here as a benchmark. The core of the algorithm is the definition of the reward function, r (i.e., how good or bad a particular HO event has performed), calculated as

$$r(x_{t_n}, a_{t_n}) = -w_{RLF} X_{EHO}(n) - w_{RLF} X_{LHO}(n) - w_{PP} X_{PP}(n) \,, \quad (9)$$

where $X_{EHO}(n) = 1$ if event n occurring in t_n is categorized as an EHO ($X_{EHO}(n) = 0$, otherwise). Analogously, $X_{LHO}(n) = 1$ or $X_{PP}(n) = 1$ if event n in t_n is a LHO or PP, respectively (0, otherwise). The values of those variables are obtained by analyzing the behavior of the connection around the handover event. These binary variables are weighted by constant coefficients, w_{RLF} and w_{PP}, to prioritize the reduction of RLF or PP in the parameter optimization process. In most cases, $w_{RLF} > w_{PP}$, since RLFs are less desirable than PPs. In this work, $w_{RLF} = 1$ and $w_{PP} = 0.5$. Note that only one binary variable can be true for the same event n. If none of the previous alternatives occur (i.e., a successful HO), $X_{EHO}(n) = X_{LHO}(n) = X_{PP}(n) = 0$ and $r(x_{t_n}, a_{t_n}) = 0$.

C. E-MRO algorithm

Unlike Q-MRO, E-MRO scheme not only aims to reduce RLF and PP, but also to optimize cell-edge user QoE. For this purpose, HOM and TTT are adjusted on an adjacency basis with the aim of increasing the average QoE of cell-edge users. In this work, the time window for evaluating the QoE of a user experiencing a HO event comprises 1 second before and after the HO trigger point (i.e., event). This short time window ensures that user QoE is only evaluated around cell edge. The resulting value is denoted as $QoE(n)$ (n for the n^{th} HO event). Note that event n is associated to the user u experiencing that n^{th} HO event, so that indexes n and u_n can be interchanged.

The reward function for E-MRO, differently to Q-MRO, is computed as the sum of three components as

$$r(x_{t_n}, a_{t_n}) = r_{radio}(x_{t_n}, a_{t_n}) + r_{QoE_{step}}(x_{t_n}, a_{t_n}) + r_{QoE_{prev}}(x_{t_n}, a_{t_n}) \,, \quad (10)$$

where r_{radio} is a reward related to radio robustness, $r_{QoE_{step}}$ is a reward due to the change in QoE caused by the HO event, and $r_{QoE_{prev}}$ is a negative reward (i.e., a penalty) due to a bad QoE before HO, defined as

$$\begin{aligned} r_{radio}(x_{t_n}, a_{t_n}) &= -w_{RLF} X_{EHO}(n) - w_{RLF} X_{LHO}(n) - w_{PP} X_{PP}(n) \,, \\ r_{QoE_{step}}(x_{t_n}, a_{t_n}) &= \max(-1, \min(1, (QoE^A(u_n) - QoE^B(u_n)))) \,, \\ r_{QoE_{prev}}(x_{t_n}, a_{t_n}) &= \max(-1, \min(0, (\frac{QoE^B(u_n)}{\overline{QoE^B(w,x)}} - \frac{\overline{QoE^B(w,x)}}{\overline{QoE^B(w,x)}}))). \end{aligned} \quad (11)$$

The first term, r_{radio}, showing the HO performance from the radio perspective, is identical to reward r defined in (9) for Q-MRO. The other two terms, $r_{QoE_{step}}$ and $r_{QoE_{prev}}$, are only included in E-MRO. $r_{QoE_{step}}$ focuses on maximizing the difference between the QoE experienced by the used after and before the HO, $QoE^A(u_n)$ and $QoE^B(u_n)$, respectively. Thus, $r_{QoE_{step}}$ rewards HOM andTTT settings causing that $QoE(u_n)$ in the target cell is better than $QoE(u_n)$ in the serving cell. The QoE difference is limited to the interval $[-1, 1]$ to ensure that the resulting reward is in a range similar to r_{radio}. By considering only r_{radio} and $r_{QoE_{step}}$ in (10), E-MRO could lead to some HO settings obtaining a high r value thanks to a large QoE step, $QoE^A(u_n)$-$QoE^B(u_n)$, due to a bad user performance in the serving cell (i.e., a low $QoE^B(u_n)$), and not caused by an improvement of the performance in the target cell, $QoE^A(u_n)$. The third component $r_{QoE_{prev}}$ in (10) avoids these wrong HO settings by penalizing situations when QoE before HO is lower than average, $\overline{QoE^B(w,x)}$. Specifically, $\overline{QoE^B(w,x)}$ is the average QoE before HO from any user w performing a HO between any cell pair in the scenario within the same state x as the state in the adjacency of user u_n. In such an average, only HOs in a time window comprising the initial period of the optimization process are considered. Thus, $r_{QoE_{prev}}$ in (10) penalizes HO settings degrading QoE in the source cell (i.e., before HO) compared to average user performance before HO with similar conditions (i.e., state).

Two variants of E-MRO approach are proposed, denoted as EQ-MRO and EN-MRO. EQ-MRO follows the above-described process of updating a Q-table with Q-values as described in (7), with a very conservative value of α to make subtle changes in Q-table (i.e.,$\alpha = 0.005$). With such a small value, it is intended to maximize the Q-value per state-action pair, $Q(x, a)$, in the long term.

Alternatively, EN-MRO replaces the computation of Q-values per action in each row of the Q-table (representing states) by an ANN as function approximator [45]. When the number of states and actions increase significantly, dealing with the Q-table is extremely difficult due to the large number of actions and states to be explored [46]. Replacing the Q-table by an ANN allows to have continuous state and action spaces, since it enables to estimate $Q(x, a, \theta)$ for non-explored state-action pairs. θ represents the trainable weights of the ANN [45]. This work uses a shallow ANN [1] with two hidden layers. This ANN is used to reduce the need for large training datasets and avoid overfitting. Specifically, a multi-layer perceptron (MLP) of 2 hidden layers is trained with a backpropagation algorithm (Levemberg-Marquardt), as a substitute of each row of the Q-table [46]. Although this technique allows the exploration of continuous states (i.e., any value for HOM and TTT parameters), discrete states and actions are maintained not only to minimize the complexity of the algorithm, which is aligned to the fact that HOM and TTT parameters are discretized in vendor equipment.

It is true that simpler reward functions based on signal quality indicators (e.g., average Signal-to-Interference-plus-Noise Ratio, SINR, or packet loss ratio) could be used to update Q-tables in both Q-MRO and E-MRO. However, valuable information would be missed, as the algorithm would neglect the fact that a certain parameter setting results in EHO, LHO or PP changes, which are ultimately the main drivers of the optimization process.

D. Algorithmic complexity

The time complexity of the Q-learning algorithm in EQ-MRO is proportional to the size of the Q-table, which is given by the product of the number of states, N_s, and the number of actions, N_a. The former is linear with the number of services. Thus, the theoretical worst-case time complexity of EQ-MRO is $O(N_s, N_a)$. For EN-MRO, complexity not only depends on the number of states and actions, but also on the ANN size. The theoretical worst-case time complexity for the backpropagation algorithm used to train the ANN in EN-MRO with N_i inputs, 1 output and N_{hl} hidden layers is $O(N_i, N_{hl}, N_{s-hl}, N_{it})$, where N_{s-hl} is the size of the hidden layers and N_{it} is the number of iterations in which the ANN is re-trained. Note that, from the above explanation, the number of services considered in the performance assessment scenario directly impacts on the algorithm complexity (through N_s) and convergence time (since a higher N_s leads to higher simulation times to collect a high number of events per action).

There are various proofs that Q-learning converges to the optimal Q function, provided that the right exploration policy and learning rate is selected [47]. Exploration needs to ensure that each state action is performed infinitely often. Likewise, the sum of the learning rates must tend to infinity (so that any value could be reached) while the sum of the squares of the learning rates is finite (to ensure convergence) [48]. Both conditions are ensured by the proposed learning scheme, which starts with a high learning rate to allow fast changes and lowers the learning rate as time progresses.

QL convergence in practice has been thoroughly studied in [49]. A key aspect to favor convergence is to use initial optimistic Q-values. Zero has been used as initial Q-value per state-action pair, which is an optimistic value since final Q-values are negative, as will be seen later. Additionally, the larger the number of events collected per state and action, the faster the convergence, since the network behavior is better known and, thus, better actions for the next AI are selected. Hence, a proper selection of AI duration is needed (30 seconds in this work). To increase the number of events per action, an action dropping strategy is followed in this work, which progressively reduces the space of possible state-action values, as will be explained in Section VI-C.

VI. Performance analysis

The above-described algorithms are tested in a dynamic system-level simulator. For clarity, the analysis set-up is presented first and results are shown later.

A. Experimental methodology

1) Simulation tool: Performance assessment is performed in a realistic LTE scenario implemented in a dynamic

[1] A shallow neural network has up to two hidden layers (opposite to deep neural networks, comprising multiple hidden layers).

TABLE II: Simulation parameters.

Time resolution	20 TTI (20 ms)
Propagation model	Pathloss COST 231 Okumura-Hata, slow fading (log-normal $\sigma = 8$ dB, $d_{corr} = 20$ m), fast fading (ETU model)
Base station model	Tri-sectorized antennas, MIMO 2x2, BW = 5 MHz (25 PRB), $f_{carrier} = 1850$ MHz, EIRP$_{max}$= 67 dBm.
Traffic model	Spatial traffic distribution and service mix based on live statistics collected on a cell basis
Mobility model	Random direction, constant speed, 30/70 km/h.
Radio resource management model	Scheduler: Classical exponential/proportional fair [57]
Link adaptation	CQI-based

LTE simulator built in Matlab [50]. Table II presents the configuration of the main simulation parameters. To reduce the computational load, only the downlink is simulated.

The propagation model includes pathloss, slow fading and fast fading. Pathloss is computed with COST-231 Hata model [51]. Slow fading is modeled statistically by a lognormal distribution, with zero mean and standard deviation of 8 dB, typical in urban macrocells [52]. Correlation distance d_{corr} quantifies the minimum distance between two statistically independent (i.e., uncorrelated) points in the scenario, which is needed for slow fading channel simulations. In this work, $d_{corr} = 20$ m, corresponding to an urban macrocellular environment [53]. Both pathloss and slow fading are computed with a 50-meter resolution. Fast fading is computed with a more detailed spatial grid with multiple realizations of a Wide-sense stationary uncorrelated scattering (WSSUS) channel. To generate that matrix, a narrow band fading grid is built following Clarke's model (i.e., a spatial bidimensional complex Gaussian variable is filtered by a bidimensional Doppler filter) [54] . Then, the model is extended by repeating the same procedure for every path in the power delay profiles corresponding to the Extended Typical Urban (ETU) channel [55]. Fast fading is only applied to the serving cell. All these terms are pre-computed to speed up computations.

From propagation losses, the signal level received by each user from every base station is computed. It is assumed that intracell interference is negligible, so that only co-channel intercell interference is considered. Average interference level is computed by considering neighbor cell load. Noise power at the terminal receiver is -112.44 dBm per Physical Resource Block (PRB). Then, link quality is determined by computing SINR, from which to derive Channel Quality Indicator (CQI). Link adaptation is modeled by a table mapping CQI to spectral efficiency, calculated by a truncated Shannon bound [56].

2) States and actions spaces: All the tested algorithms work over the same set of states and actions. Each state is modeled as a tuple $\{s,v,d,l\}$. Table III illustrates the meaning and possible values for each parameter.

TABLE III: Parameter defining state space

Parameter	Possible values	Index
Service s	{FTP,VIDEO,WEB}	{1,2,3}
User velocity v **[km/h]**	{30,70}	{1,2}
Inter-site distance d **[km]**	{≤1.25,>1.25}	{1,2}
Target cell load l **[%]**	{≤70,>70}	{1,2}

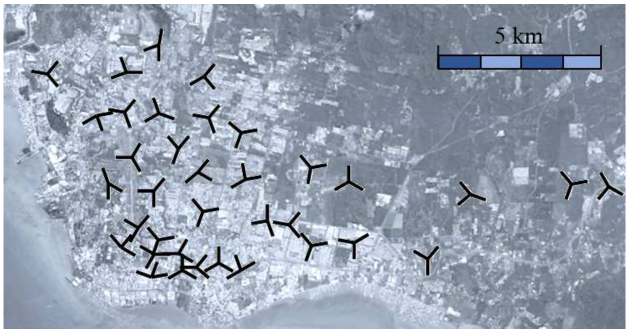

Fig. 2: Simulated scenario [26].

Service s indicates the type of service (i.e., FTP, Video or Web). Although VoIP traffic is present in the scenario, live network statistics used to tune the simulator showed that VoIP traffic is extremely low and scattered in a few cells. The inclusion of VoIP in the optimization scheme might cause unreliable RLF and QoE statistics for this service. For this reason, Q-MRO and E-MRO are not allowed to change HO settings for VoIP (i.e., $HOM(i,j,VoIP) = 0$ dB and $TTT(i,j,VoIP) = 40$ ms in all experiments). v denotes user speed, with 30 or 70 km/h as possible values. These values model users in a city or in highways. Lower speeds (e.g., pedestrian users) are not considered in this work since they hardly make HOs, and, thus, MRO techniques do not have a significant impact on those users. d is the Inter-Site Distance (ISD), which is used to differentiate between close and far neighbor cells. Specifically, a threshold of 1.25 km is used for labeling an adjacency as close or far. In the realistic scenario considered, shown in Figure 2, 954 adjacencies are labeled as close and 10710 as far. Finally, l is the target cell load, for which a threshold of 70 % is set to differentiate between congested and non-congested neighbors.

With the above configuration, the number of states is $3{\cdot}2{\cdot}2{\cdot}2 = 24$ states. Note that the aim of the optimization algorithm is to find the optimum HO settings for every state, for which many HOs are needed per state. A larger number of values per parameter s, v, d and l would improve the accuracy when characterizing the scenario, but it would also imply a significant increase in the number of HO events needed to learn the optimal system behavior, requiring a larger evaluation period. For an easier analysis, Table IV enumerates states with a variable x, ranging from 1 up to 24, together with the corresponding parameter labels.

For each state, a total of 45 possible actions are considered, corresponding to the combination of 15 possible values of HOM (from -7 dB to +7 dB in steps of 1 dB) and 3 values of TTT (40, 100 and 256 ms). For space reasons,

State x	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
s	1	1	1	1	1	1	1	1
v	1	2	1	2	1	2	1	2
d	1	1	2	2	1	1	2	2
l	1	1	1	1	2	2	2	2
State x	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
s	2	2	2	2	2	2	2	2
v	1	2	1	2	1	2	1	2
d	1	1	2	2	1	1	2	2
l	1	1	1	1	2	2	2	2
State x	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**
s	3	3	3	3	3	3	3	3
v	1	2	1	2	1	2	1	2
d	1	1	2	2	1	1	2	2
l	1	1	1	1	2	2	2	2

TABLE IV: State space.

TABLE V: Action space mapping.

Action a	**1**	**2**	**3**	**4**	**5**
HOM [dB]	-2	-1	0	1	2
TTT [ms]	40	40	40	40	40
Action a	**6**	**7**	**8**	**9**	**10**
HOM [dB]	-2	-1	0	1	2
TTT [ms]	100	100	100	100	100
Action a	**11**	**12**	**13**	**14**	**15**
HOM [dB]	-2	-1	0	1	2
TTT [ms]	256	256	256	256	256

Table V only presents a subset of 15 actions, a, with $HOM \in \{-2, -1, 0, 1, 2\}$ dB and $TTT \in \{40, 100, 256\}$ ms, which will be used later.

3) Experiments: Three experiments of increasing complexity are carried out. The first experiment intends to show Q-MRO limitations, i.e., how neglecting QoE issues in the parameter tuning process leads to bad QoE performance. Then, in the second experiment, the proposed E-MRO schemes are compared with traditional Q-MRO in a naive scenario of a single service. Finally, the third experiment compares the methods in a complete scenario with all the services.

In the first experiment, a simple experiment is defined with only one state in the network. The state is characterized by one service (s = FTP) and a fixed user mobility (v = 30 km/h), while ISD and target cell load are not classified. The action space A involves only one dimension, HOM values in the range -7 dB to 7 dB adjusted in steps of 1 dB. This results in 15 possible action values and a Q-table as a 1x15 array. This HOM range is chosen because, in most vendors, users with SINR < -7 dB are not given radio resources in the scheduling process. TTT takes a fixed value of 40 ms, which is the minimum non-zero value according to 3GPP [34]. A low TTT value allows the system to show the impact of different HOM settings. To simplify the analysis, users are uniformly distributed within every cell.

A classical Q-MRO algorithm is used in the first experiment with rewards defined as in (9). The initial exploration period is set to $N_{explore} = 100$ AIs (50 minutes of network time) followed by a training phase N_{train}= 360 AIs (3 hours) and an exploitation phase of $N_{exploit} = 240$ AIs (2 hours), long enough to ensure adequate performance assessment.

Different figures of merit are monitored per AI during the optimization process. The first metric is the average cell edge QoE, defined as

$$\overline{QoE_{edge}} = \frac{1}{2N_u}\sum_{u}(QoE^A(u) + QoE^B(u)) \ , \qquad (12)$$

where N_u is the number of users experiencing a HO event (i.e, EHO, LHO, PP or SHO) during an AI.

Another important indicator is the average QoE for all users in the network $\overline{QoE}$, considering the whole connection, and at the end of the optimization process. $\overline{QoE}$ is defined as

$$\overline{QoE} = \frac{1}{N_u}\sum_{\forall u} QoE(u) \ . \qquad (13)$$

During the optimization process, user QoE, $QoE(u)$, must be continuously calculated around some slicing temporal window. $QoE(u)$ for FTP and Web services are calculated as in (3) and (4), which requires estimating average user throughput during the time window when QoE is assessed (i.e., 1 second around the HO event). Average user throughput at every simulation step n_{sim} in that time window is calculated by an Auto-Regressive (AR) filter as

$$TH_{n_{sim}}(u) = (1-\beta)TH_{n_{sim}-1}(u) + \beta \ TH_{n_{sim}}(u) \ , \quad (14)$$

with $\beta = 0.98$.

Additionally to QoE indicators, average user throughput, $\overline{TH}$, is also used as a performance indicator, and defined as

$$\overline{TH} = \frac{1}{N_u}\sum_{\forall u} TH(u) \ , \qquad (15)$$

where $TH(u)$ is the average throughput of user$u$. Note that index u in (13) and (15) denotes all users in the scenario, and not only those users experiencing a HO, as in (12). Thus, $\overline{TH}$ and $\overline{QoE}$ are global performance indicators, while $\overline{QoE_{edge}}$ assesses cell edge users. Also note that throughput figures are intermediate indicators, aiming to understand different algorithms' performance, while QoE indicators are defined to quantify final performance of MRO techniques.

Other four metrics are introduced. These metrics are the traditional MRO performance indicators reflecting the ratio of LHO, EHO, PP and SHO, as

$$LHO(n_{AI}) \ [\%] = 100 \frac{N_{LHO}(n_{AI})}{N_{events}(n_{AI})} \ , \qquad (16)$$

$$EHO(n_{AI}) \ [\%] = 100 \frac{N_{EHO}(n_{AI})}{N_{events}(n_{AI})} \ , \qquad (17)$$

$$PP(n_{AI}) \ [\%] = 100 \frac{N_{PP}(n_{AI})}{N_{events}(n_{AI})} \ , \qquad (18)$$

$$SHO(n_{AI})\ [\%] = 100\frac{N_{SHO}(n_{AI})}{N_{events}(n_{AI})} = 100\frac{N_{events}(n_{AI}) -}{N_{events}(n_{AI})}\ ,$$
$$\frac{-N_{LHO}(n_{AI}) - N_{EHO}(n_{AI}) - N_{PP}(n_{AI})}{N_{events}(n_{AI})}\ , \qquad (19)$$

where $N_{LHO}(n_{AI})$, $N_{EHO}(n_{AI})$, $N_{PP}(n_{AI})$ and $N_{SHO}(n_{AI})$ are the number of LHO, EHO, PP and SHO during the n_{AI} AI, and N_{events} is the number of events (i.e., HOs) during the n_{AI} AI.

As a result for the first experiment, the best actions (HOM values) found by Q-MRO will be selected as a reduced action space for the second experiment.

In a second experiment, the aim is to find the best settings for both HOM and TTT parameters on an adjacency basis in a simple scenario with a single service. To this end, the space of states is enlarged by including all possible values for v, d and l. Yet, only FTP service is still considered, so that only eight states are simulated (states from 1 to 8 in Table IV). Likewise, the action space is limited to 15 possible actions, defined by the 5 best HOM values in the first experiment and 3 possible TTT values (i.e., 40, 100 and 256 ms). With these states and actions, three MRO approaches are compared: Q-MRO, EQ-MRO and EN-MRO, with Q-MRO considered as a benchmark [11]. For this second experiment, $N_{explore}$ = 150 AIs (75 minutes), N_{train} = 480 AIs (4 hours) and $N_{exploit}$ = 960 AIs (8 hours). The larger exploitation time (8 hours vs 2 hours in the first experiment) is needed because of the higher number of actions to be tested (15 actions vs 5 actions in the first experiment). EQ-MRO uses a 8x15 Q-table (states·actions), whereas EN-MRO replaces the Q-table by a shallow ANN with two hidden layers of 4 neurons each. The ANN is trained for the first time after $N_{explore}$ AIs and re-trained every 40 AIs (i.e, every 20 minutes), since the system needs more time to collect brand new data to learn from. To speed up the learning process, one action, the one with the lowest Q-value, is dropped from the learning process every 40 AIs (20 minutes). This dropping process starts after a certain time (200 AIs, 200 minutes) in order to collect significant statistics for the dropping decision, and it is repeated 10 times to eliminate the worst 10 actions during the learning process. At the end of the learning process, the best 5 actions will remain. Thus, the outcome of the second experiment are the best HO settings for every adjacency in the network considering its peculiarities (e.g., user speed, inter-site distance, target cell load,...) in a single service scenario.

TABLE VI: Traffic indicators.

Indicator	Min	Avg.	Max
$N_u(i, VIDEO)/N_u(i)$ [%]	4.3	33.03	50.7
$N_u(i, FTP)/N_u(i)$ [%]	16.9	29.46	72.25
$N_u(i, WEB)/N_u(i)$ [%]	22.1	37.51	47.62
$U(i)$ [%]	4.72	58.18	95.27

In a third experiment, the complete system with all services (i.e., s = FTP, VIDEO and WEB) is considered. The optimization algorithm needs estimating user QoE before and after every HO event, using the same 1-second time window. Such an estimation is simple for continuous services (i.e. s = FTP), but not for bursty services (i.e., VIDEO and WEB), which alternate periods of information download and silence. If a HO event occurs when, for example, a web user is reading (i.e., no information is being downloaded), the QoE of that user is not taken into account in the HO optimization process. For video users, it is assumed that, at most, only one stalling will occur during the HO time window. As for the user throughput estimation, the average stalling duration (L_{tr}) is computed with an Auto-Regressive (AR) filter as

$$L_{tr,n_{sim}}(u) = \beta L_{tr,n_{sim}-1}(u) + sim_{step}\ , \qquad (20)$$

where $L_{tr,n_{sim}}$ is the average stalling duration at simulation step n_{sim}, $\beta = 0.98$ and sim_{step} is the duration of the simulation step (20 ms).

In the third experiment, all services are included, leading to 24 potential states and 15 possible actions per state, shown in Tables IV and V. Thus, longer simulation times are needed. Specifically, $N_{explore}$ = 300 AIs (150 minutes), N_{train} = 960 AIs (8 hours) and $N_{exploit}$ = 1320 AIs (11 hours). EQ-MRO uses a 24x15 Q-table to store the Q-value per state-action pair, whereas EN-MRO replaces the Q-table by the same shallow ANN described in the second experiment. The ANN is trained for the first time after $N_{changes}$ and re-trained every 75 AIs (i.e, 37.5 minutes) while $\varepsilon > 0.75$, or every 50 AIs (25 minutes) otherwise. To speed up the learning process, one action, the one with the lowest Q-value, is dropped from the learning process every 75 AIs (37.5 minutes) if $\varepsilon > 0.75$, or every 50 AIs (25 minutes) otherwise.

Regarding the service traffic distribution, Table VI describes some cell-level statistics extracted from the live scenario used for the experiment. The first three rows show the service mix by presenting the ratio of users of each service. $N_{us}(i, s)$ denotes the number of users demanding service s in cell i and $N_{us}(i)$ is the total number of users in cell i. The last row shows the average cell load, $U(i)$, measured as the average PRB utilization in the cell. Columns represent average and extreme values at cell level.

B. Experiment 1: Q-MRO limitations

Figure 3 shows the evolution of the different HO metrics across AIs. Recall that the initial period takes 100 AIs, while the learning interval takes another 360 AIs. Hence, the final performance can be extracted after n_{AI} = 460, approximately. Two curves are superimposed: the solid one, with high variability, represents the value for each AI, while the dashed curve is a Simple Moving Average (SMA) with a window of 100 samples. For comparison purposes, the first and last values of the SMA process are considered

TABLE VII: Q-MRO performance (Experiment 1).

	Initial	**End**
$\overline{EHO}$ [%]	1	0.1
$\overline{LHO}$ [%]	18.52	2
$\overline{PP}$ [%]	21.6	19
$\overline{SHO}$ [%]	58.88	78.9
$\overline{QoE_{edge}}$ [MOS]	1.28	1.48

as the initial and final values of the optimization process, respectively.

Table VII presents initial and final values for the different metrics in the first experiment. As shown in the table, LHOs are practically removed at the end of the optimization process ($\overline{LHO}$ = 2 %, compared to $\overline{LHO}$ = 18 % at the beginning). In contrast, EHOs were already low at the beginning of the process (1 %), due to the way the HO scheme is designed in the simulation tool. Nonetheless, EHOs are also reduced at the end of the process (0.1 %). Regarding PP, higher figures at the beginning are not significantly reduced at the end of the process (21.6 and 19 %, respectively). This is due to the minor weight of the PP metric in the reward function (9). Ultimately, SHO increase as a result of the improvement (i.e., decrease) in the other HO events.

Figure 4 illustrates the evolution of the Q-value for actions HOM = -2 dB, -1 dB, 0 dB, 1 dB and 2 dB. These are the best 5 actions out of the 15 actions tested in the experiment. It is observed that HOM≥0 dB performs better than those actions with $HOM<$ 0 dB, given the fact that negative HOM values largely increase PP events, which is the most frequent HO event. These 5 HOM values are selected for the next experiments.

Finally, Figure 5 shows the evolution of $\overline{QoE_{edge}}$ during the optimization process. Note that, in this experiment, no QoE metric is included in the reward function. Specifically, the initial and final values of $\overline{QoE_{edge}}$ are 1.48 and 1.28, respectively. Thus, the HO performance improvement is obtained at the expense of deteriorating the QoE of cell edge users by 0.2 MOS points, in line with the reward function in (9), which does not take QoE into account.

C. Experiment 2: QoE-aware algorithms (single service)

Table VIII shows the initial and final metrics in the second experiment. All methods share the same initial value and the best final value is highlighted for each metric. Q-MRO manages to reduce LHO (from 18.7 to 4.05 %), but QoE is unaltered ($\overline{QoE_{edge}}$ = 1.45). In contrast, EN-MRO halves PP (from 20.65 to 11.44 %), while LHO is increased (from 18.7 to 21.35 %), resulting in $\overline{SHO}$ = 66.68 %. EQ-MRO achieves the best SHO by improving all indicators (from 18.7 to 9.85 % for LHO and 20.65 to 17.84 % in PP). In this second experiment, cell edge QoE is slightly improved by both E-MRO schemes (from 1.45 up to 1.48 with EQ-MRO and up to 1.52 with EN-MRO).

As for $\overline{QoE_{edge}}$, $\overline{QoE}$ also improves with EQ-MRO and EN-MRO compared to Q-MRO (4.17 for both EQ-MRO and EN-MRO against 4.09 for Q-MRO). Likewise, $\overline{TH}$

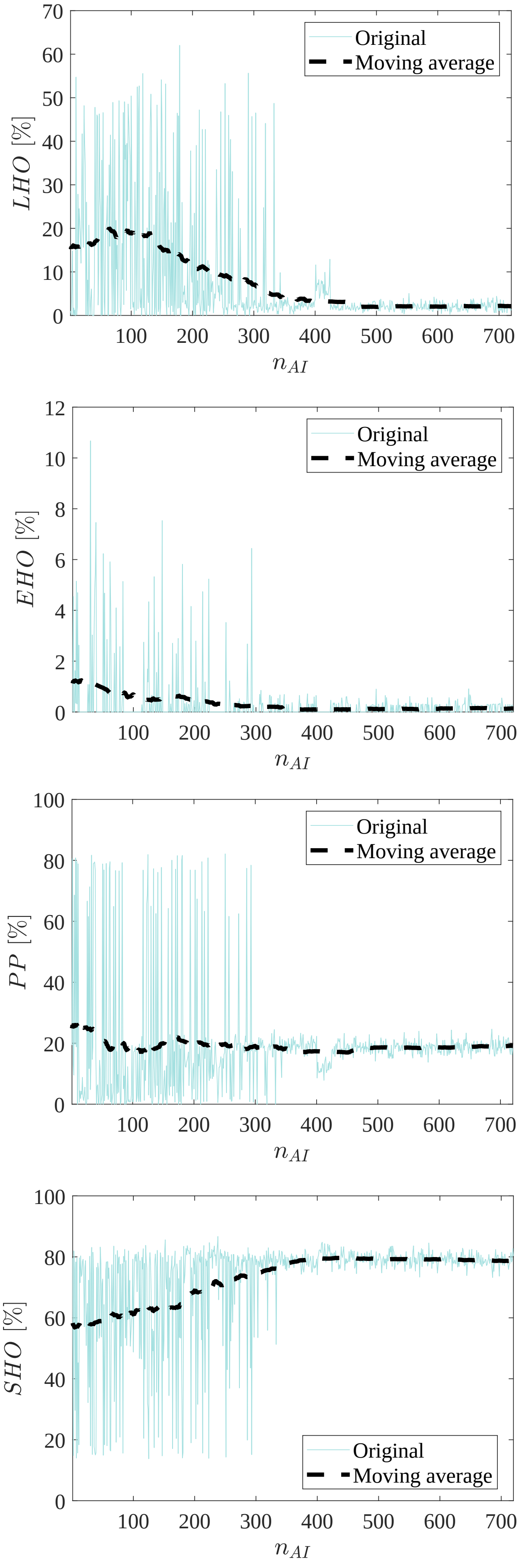

Fig. 3: Q-MRO performance (Experiment 1).

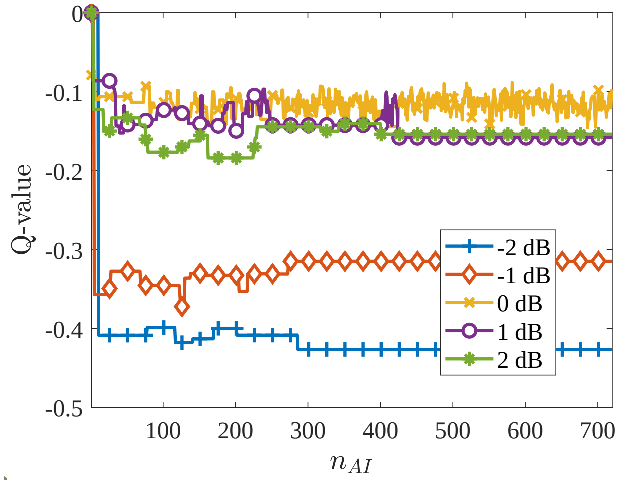

Fig. 4: Evolution of Q-value for the best 5 actions (Experiment 1).

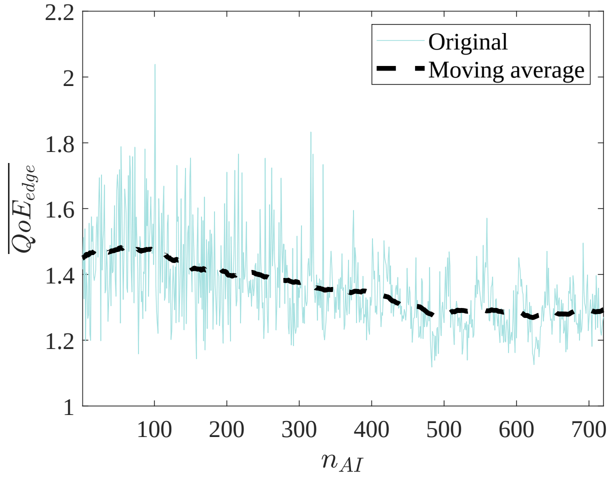

Fig. 5: $\overline{QoE_{edge}}$ over time.

improves (4.19 for EQ-MRO and 4.29 Mpbs for EN-MRO, compared to 4 Mbps for Q-MRO). These results show that E-MRO algorithms not only improve cell edge users, as it is reflected in the reward function, but also the global average QoE and throughput figures compared to Q-MRO.

A further analysis is carried out using a new indicator showing the overall impact of HOs on QoE by comparing the user QoE before and after the HO as

$$\overline{QoE_{diff}} = \frac{1}{N_u}\sum_{\forall u}(QoE^A(u) - QoE^B(u)) \,. \tag{21}$$

TABLE VIII: Method performance (Experiment 2).

	Initial	Q-MRO	EQ-MRO	EN-MRO
$\overline{EHO}$ [%]	0.46	1.12	0.47	0.53
$\overline{LHO}$ [%]	18.7	4.05	9.85	21.35
$\overline{PP}$ [%]	20.65	25.14	17.84	11.44
$\overline{SHO}$ [%]	60.19	69.7	71.84	66.68
$\overline{TH}$ [Mbps]	-	4	4.19	4.29
$\overline{QoE}$ [MOS]	-	4.09	4.17	4.17
$\overline{QoE_{edge}}$	1.45	1.45	1.48	1.52

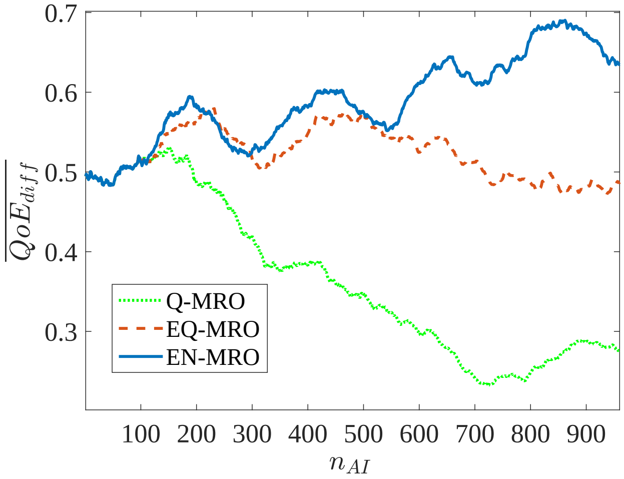

Fig. 6: Evolution of the change in QoE experienced by handed over users (Experiment 2).

Note that this indicator is $r_{QoE_{diff}}$ in the reward equation (10). Figure 6 shows the evolution of $\overline{QoE_{diff}}$ for this second experiment. It is observed that $\overline{QoE_{diff}}$ is always positive in all approaches (i.e., QoE after HO is better than QoE before HO). However, Q-MRO decreases the QoE difference of handed over users, EQ-MRO keeps the same QoE difference along the optimization process and EN-MRO manages to increase QoE difference of users performing HO.

Table IX details the best action, a, selected per state, x. As expected, Q-MRO selects those actions with the HOM setting that triggers the HO when the signal level received from the serving cell is the same as that from the target cell (i.e., action $a = 3$, with $HOM = 0$ dB and $TTT = 40$ ms) for 6 out of the 8 states in this second experiment (states $x = 3$ to 8). Only in states $x = 1$ and 2 (corresponding to adjacencies with small ISD and unloaded target cell), HO is delayed by selecting higher HOM values (actions 4 and 5, with $HOM = 1$ and 2 dB, respectively). In contrast, EQ-MRO labels as best actions those delaying the HO trigger (i.e, action 4 with $HOM = 1$ dB and $TTT = 40$ ms, action 5 with $HOM = 2$ dB and $TTT = 40$ ms and action 15 with $HOM = 2$ dB and $TTT = 256$ ms) in 6 out of the 8 states. Finally, EN-MRO delays HO trigger even more, since the best actions in 5 out of the 8 states show $TTT \geq 100\ ms$.

A more detailed analysis compares the best actions selected by EQ-MRO and EN-MRO for states with small ISD $\{1, 2, 5, 6\}$ and large ISD $\{3, 4, 7, 8\}$. The selected actions in the former group, comprising adjacencies between distant cells, delay the HO point (by increasing HOM, TTT or both) by a larger amount than the second group, comprising adjacencies between nearby cells (which choose lower HOM and/or TTT values). Such a behavior stresses the importance of considering ISD (i.e., parameter d) when

TABLE IX: Best actions selected per state (Experiment 2).

	State index x							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Q-MRO	4	5	3	3	3	3	3	3
EQ-MRO	15	5	5	8	5	5	4	3
EN-MRO	14	10	9	4	15	13	4	3

TABLE X: Method performance (Experiment 3).

	Initial	Q-MRO	EQ-MRO	EN-MRO
$\overline{EHO}$ [%]	1.27	0.64	0.43	0.43
$\overline{LHO}$ [%]	19.92	5.08	12.78	12.8
$\overline{PP}$ [%]	25.87	27.22	13.28	14.75
$\overline{SHO}$ [%]	52.94	67.06	73.51	72.02
$\overline{TH}$ [Mbps]	-	3.59	3.64	3.69
$\overline{QoE}$	-	4.08	4.12	4.12
$\overline{QoE_{edge}}$	2.04	2.07	2.09	2.14
$\overline{QoE_{edge}^{(VIDEO)}}$	2.05	2.11	2.1	2.11
$\overline{QoE_{edge}^{(FTP)}}$	1.58	1.55	1.59	1.66
$\overline{QoE_{edge}^{(WEB)}}$	2.38	2.43	2.46	2.56
$\overline{QoE_{diff}}$	0.6	0.4	0.75	0.75

selecting optimal HO settings . A similar analysis (not presented here) shows that EN-MRO tends to suggest actions with larger TTT values for the lower user speed of 30 km/h (states 1, 3, 5, 7).

D. Experiment 3: QoE-aware algorithms (multiple services)

Table X shows the main performance indicators at the end of the optimization process for all methods. For a more detailed analysis, QoE figures are broken down by services.

Similarly to Experiment 2, Q-MRO achieves the best performance in terms of LHO ($\overline{LHO} = 5.08$ %) compared to other algorithms. However, EQ-MRO and EN-MRO end up with a better SHO ratio (67.06 for Q-MRO vs 73.51 for EQ-MRO and 72.02 for EN-MRO) and better cell edge QoE. In particular, EN-MRO obtains the best QoE indicators, both aggregated and per service (i.e., $\overline{QoE_{edge}}$, $\overline{QoE_{edge}^{(VIDEO)}}$, ...). Likewise, the overall QoE figure for Q-MRO ($\overline{QoE} = 4.08$) is outperformed by EQ-MRO and EN-MRO in a similar amount ($\overline{QoE} = 4.12$). Finally, $\overline{TH}$ is also improved by EQ-MRO and EN-MRO ($\overline{TH} =$ 3.59 Mbps for Q-MRO, while $\overline{TH} = 3.64$ Mbps for EQ-MRO and 3.69 Mbps for EN-MRO). As expected, Q-MRO degrades $\overline{QoE_{diff}}$, while EQ-MRO and EN-MRO achieve similar QoE improvements for cell edge users ($\Delta\overline{QoE_{diff}} \approx 0.35$).

For a more detailed analysis of cell edge performance, Figure 7 shows the cumulative density function for individual users at the end of the optimization process. Users are ordered from worst to best QoE (i.e., from 1 to 5). The curves of EQ-MRO and EN-MRO are above the Q-MRO curve. Thus, both E-MRO methods not only improve cell edge QoE, but also users with medium/best QoE values. Specifically, when comparing 70 %-percentile, Q-MRO obtains 2.87 against 2.93 and 3.13 points for EQ-MRO and EN-MRO, respectively.

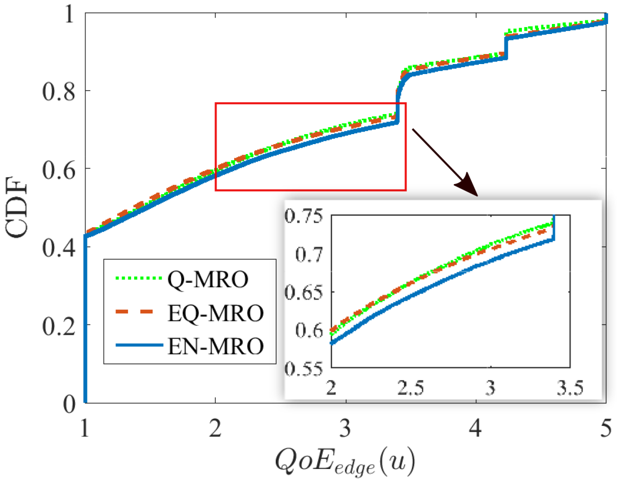

Fig. 7: Cumulative density function of optimized user QoE.

TABLE XI: Best actions per state (Experiment 3).

	State index x							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Q-MRO	3	3	3	2	4	3	3	3
EQ-MRO	5	10	4	3	9	10	4	4
EN-MRO	5	4	4	3	10	11	4	4
	State index x							
	9	**10**	**11**	**12**	**13**	**14**	**15**	**16**
Q-MRO	3	5	3	3	3	4	3	3
EQ-MRO	5	5	3	3	5	12	3	3
EN-MRO	5	5	4	4	4	9	4	3
	State index x							
	17	**18**	**19**	**20**	**21**	**22**	**23**	**24**
Q-MRO	3	4	3	4	3	3	3	3
EQ-MRO	5	5	5	4	10	5	5	5
EN-MRO	5	4	4	4	10	5	5	5

Table XI breaks down the best actions per state and algorithm at the end of the optimization process. Recall that states 1-8 refer to FTP users, states 9-16 refer to video users and states 17-24 correspond to web users, as shown in Table (IV). Thus, states $x = 1$, 9 and 17 show the same network state except for the service (FTP, video and web service, respectively). A detailed analysis (not presented here) shows that the best HOM, and TTT settings per state are similar to those in Experiment 2, with low positive values for both parameters.

To check the benefit of selecting different parameter settings per service, a detailed analysis of the final Q-value per action and service obtained by EN-MRO is carried out. Note that 24 Q-values are obtained per action (i.e., as much as the number of states). For an easier analysis, the Q-values of the 8 states corresponding to the same service are averaged, so that a single value is obtained per action and service as

$$Q - value^{(s)}(a) = \frac{1}{8}\sum_{x/s} Q - value(a,x) \tag{22}$$

where $s \in \{FTP, Video, Web\}$. The resulting averages (3 per action) are shown in Figure 8. At first glance, it is observed that the three services show their best (average) behavior with actions 3 to 5 (i.e., non-negative HOM

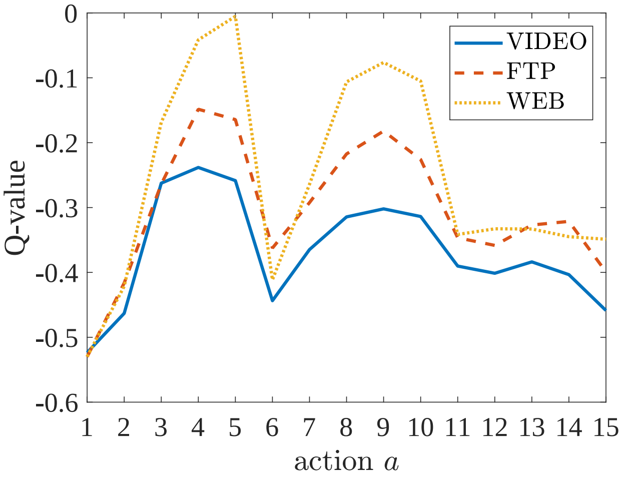

Fig. 8: Average Q-value per action for EN-MRO algorithm.

and minimum TTT, as observed in Table V). However, the best action for Web service is $a = 5$ ($HOM = 2$ dB), while the best for Video and FTP is $a = 4$ ($HOM = 1$ dB). Moreover, the vertical shift between the three curves (services) shows that rewards are different between services, which is a clear indication that improvements are different between services. Specifically, Web service experiences larger rewards than the other two services. This is aligned to cell edge QoE values per service reported in Table X, where it is observed that Web service is the one with the largest improvement obtained with EQ-MRO (from 2.38 to 2.56).

E. Discussion on execution concerns

The proposed EQ/EN-MRO algorithms are executed periodically (every 30 s of network time) until the best actions have been discovered. In terms of execution time, the most limiting factor is the large period to collect enough HO events to train the ANN in EN-MRO. This time grows linearly with the number of system states. In this work, EN-MRO is implemented with the Deep Learning Toolbox in Matlab. With that toolbox, training an ANN with 2 hidden layers of 4 neurons takes more time than updating a bidimensional Q-table matrix, even if the former is shallow (0.07 seconds per training operation). Specifically, the average execution time of one execution of EQ-MRO (Q-table) and EN-MRO (ANN) is 0.12 and 0.19 seconds, respectively, in a personal computer with a 3.6-GHz octa-core processor and 24 GB of RAM. Overall, EQ-MRO and EN-MRO take 2.6 and 4.2 minutes when 11 hours of network time are simulated (1320 executions, 1 per AI).

VII. Conclusions

In this paper, a novel QoE-aware mobility robustness optimization scheme for adjusting handover trigger points in a LTE network has been proposed. The aim of the algorithm is to improve QoE at cell edge while increasing the percentage of successful HOs. The proposed learning algorithm changes handover trigger points periodically (every 30 seconds) based on a Q-learning scheme. Two variants have been presented, depending on the way the expected Q-value per state-action pair is obtained: either with a Q-table or by training a neural network implemented with a multi-layer perceptron. Method assessment has been carried out in a dynamic system-level LTE simulator implementing a realistic macrocellular scenario. The proposed scheme is conceived to be implemented in the base station.

Results have shown that a legacy MRO optimization algorithm only driven by radio performance degrades QoE up to 0.2 MOS points in a simplified scenario with a single service and medium user speed. In contrast, the two variants of the proposed QoE-driven algorithm improve cell edge QoE, while also increasing the ratio of successful handovers. Compared to the legacy method, both variants improve the successful handover ratio by more than 5 % in absolute terms, while cell edge QoE of some services is increased by up to 0.13 MOS points. Web services experience larger improvements than FTP and Video users. Moreover, from the analysis of the parameter settings suggested by the algorithm, it has been deduced that the handover trigger point should be delayed more in adjacencies between distant cells and slow moving users.

Future work will consider the extension of the algorithm to enhanced mobile broadband services, requiring extremely large user throughput, and mission critical services, requiring ultra-reliable low-latency communications.

Acknowledgments

This work has been partially funded by the Spanish Ministry of Science, Innovation and Universities (RTI2018-099148-B-I00) and Junta de Andalucía (UMA18-FEDERJA-256).

References

[1] NGMN Use Cases related to Self Organising Network, Overall Description, 2008.

[2] O. G. Aliu, A. Imran, M. A. Imran, and B. Evans, “A Survey of Self Organisation in Future Cellular Networks,” *IEEE Communications Surveys Tutorials*, vol. 15, no. 1, pp. 336–361, 2013.

[3] S. Hämäläinen, H. Sanneck, and C. Sartori, “LTE Self-Organizing Networks (SON): Network Management Automation for Operational Efficiency Hardcover,” 2012.

[4] Ericsson AB, “Ericsson mobility report,” Nov. 2017.

[5] S. Barakovic and L. Skorin-Kapov, “Survey and Challenges of QoE Management Issues in Wireless Networks,” *Journal of Computer Networks and Communications*, vol. 2013, Mar. 2013.

[6] K. Zheng, Z. Yang, K. Zhang, P. Chatzimisios, K. Yang, and W. Xiang, “Big data-driven optimization for mobile networks toward 5G,” *IEEE Network*, vol. 30, no. 1, pp. 44–51, 2016.

[7] A. J. Garcia, M. Toril, P. Oliver, S. Luna-Ramirez, and R. Garcia, “Big Data Analytics for Automated QoE Management in Mobile Networks,” *IEEE Communications Magazine*, vol. 57, no. 8, pp. 91–97, 2019.

[8] P. V. Klaine, M. A. Imran, O. Onireti, and R. D. Souza, “A Survey of Machine Learning Techniques Applied to Self-Organizing Cellular Networks,” *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2392–2431, 2017.

[9] A. Imran, A. Zoha, and A. Abu-Dayya, “Challenges in 5G: how to empower SON with big data for enabling 5G,” *IEEE Network*, vol. 28, no. 6, pp. 27–33, 2014.

[10] R. Narasimhan and D. C. Cox, "A handoff algorithm for wireless systems using pattern recognition," in *Ninth IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (Cat. No.98TH8361)*, vol. 1, Sep. 1998, pp. 335–339 vol.1.

[11] S. S. Mwanje and A. Mitschele-Thiel, "Distributed cooperative Q-learning for mobility-sensitive handover optimization in LTE SON," in *2014 IEEE Symposium on Computers and Communications (ISCC)*, vol. Workshops, Jun. 2014, pp. 1–6.

[12] J. Wu, J. Liu, Z. Huang, and S. Zheng, "Dynamic fuzzy Q-learning for handover parameters optimization in 5G multi-tier networks," in *2015 International Conference on Wireless Communications Signal Processing (WCSP)*, Oct 2015, pp. 1–5.

[13] Wencong Qin, Yinglei Teng, Mei Song, Yinghai Zhang, and Xiaojun Wang, "A Q-learning approach for mobility robustness optimization in Lte-son," in *2013 15th IEEE International Conference on Communication Technology*, Nov 2013, pp. 818–822.

[14] M. Ekpenyong, J. Isabona, and E. Isong, "Handoffs Decision Optimization of Mobile Celular Networks," in *2015 International Conference on Computational Science and Computational Intelligence (CSCI)*, Dec 2015, pp. 697–702.

[15] N. O. Tuncel and M. Koca, "Joint ICIC and Mobility Management Optimization in Self-Organizing Networks," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, March 2017, pp. 1–6.

[16] R. Vijayan and J. Holtzman, "Model for analyzing handoff algorithms," *Vehicular Technology, IEEE Transactions on*, vol. 42, pp. 351 – 356, 09 1993.

[17] O. Andrisano, M. Dell'Acqua, G. Mazzini, R. Verdone, and A. Zanella, "On the parameters optimization in handover algorithms," in *VTC '98. 48th IEEE Vehicular Technology Conference. Pathway to Global Wireless Revolution (Cat. No.98CH36151)*, vol. 2, 1998, pp. 1400–1404 vol.2.

[18] O. Andrisano, M. Dell'Acqua, G. Mazzini, and R. Verdone, "On the parameters optimization in handover algorithms," vol. 2, 06 1998, pp. 1400 – 1404 vol.2.

[19] G. Hui and P. Legg, "Soft Metric Assisted Mobility Robustness Optimization in LTE Networks," in *2012 International Symposium on Wireless Communication Systems (ISWCS)*, 2012, pp. 1–5.

[20] I. M. Bălan, B. Sas, T. Jansen, I. Moerman, K. Spaey, and P. Demeester, "An enhanced weighted performance-based handover parameter optimization algorithm for LTE networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2011, no. 1, p. 98, Sep 2011.

[21] V. Buenestado, J. M. Ruiz-Aviles, M. Toril, and S. Luna-Ramirez, "Mobility Robustness Optimization in Enterprise LTE Femtocells," in *2013 IEEE 77th Vehicular Technology Conference (VTC Spring)*, 2013, pp. 1–5.

[22] Y. Mal, J. Chen, and H. Lin, "Mobility robustness optimization based on radio link failure prediction," in *2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN)*, July 2018, pp. 454–457.

[23] P. Oliver-Balsalobre, M. Toril, S. Luna-Ramírez, and J. M. Ruiz Avilés, "Self-tuning of scheduling parameters for balancing the quality of experience among services in LTE," *EURASIP Journal on Wireless Communications and Networking*, vol. 2016, no. 1, p. 7, Jan. 2016.

[24] P. Oliver-Balsalobre, M. Toril, S. Luna-Ramírez, and R. G. Garaluz, "Self-Tuning of Service Priority Parameters for Optimizing Quality of Experience in LTE," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 4, pp. 3534–3544, Apr. 2018.

[25] J. Nightingale, P. Salva-Garcia, J. M. A. Calero, and Q. Wang, "5G-QoE: QoE Modelling for Ultra-HD Video Streaming in 5G Networks," *IEEE Transactions on Broadcasting*, vol. 64, no. 2, pp. 621–634, 2018.

[26] 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. 11 271–11 282, Nov 2019.

[27] M. L. Marí-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. 156 707–156 717, 2020.

[28] C. Gijón, M. Toril, S. Luna-Ramírez, and M. Luisa 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, Oct 2019.

[29] Z. Ali, N. Baldo, J. Mangues-Bafalluy, and L. Giupponi, "Machine learning based handover management for improved QoE in LTE," in *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, 2016, pp. 794–798.

[30] 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.

[31] 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)*, Sep. 2013, pp. 2154–2158.

[32] Y. Xu, W. Xu, Z. Wang, J. Lin, and S. Cui, "Deep Reinforcement Learning Based Mobility Load Balancing Under Multiple Behavior Policies," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, May 2019, pp. 1–6.

[33] Y. Xu, W. Xu, Z. Wang, J. Lin, and S. Cui, "Load Balancing for Ultradense Networks: A Deep Reinforcement Learning-Based Approach," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 9399–9412, 2019.

[34] 3GPP Technical Specification 36.331, "E-UTRA radio resource control (RRC) protocol specification (Release 8)," Dec. 2011.

[35] X. Zhang, *Mobility Optimization*, 2017, pp. 254–359.

[36] Scenarios, requirements and KPIs for 5G mobile and wireless system ICT-317669 METIS project, D6.1 v1-2, 2013.

[37] 3GPP Technical Report 25.892 v6.0.0, "Feasibility Study for Orthogonal Frequency Division Multiplexing(OFDM) for UTRAN Enhancement(Rel-6), pp. 62-63," 2004.

[38] P. Seeling and M. Reisslein, "Video Transport Evaluation With H.264 Video Traces," *IEEE Communications Surveys Tutorials*, vol. 14, no. 4, pp. 1142–1165, Apr. 2012.

[39] Y. Fang, I. Chlamtac, and Yi-Bing Lin, "Channel occupancy times and handoff rate for mobile computing and PCS networks," *IEEE Transactions on Computers*, vol. 47, no. 6, pp. 679–692, Jun. 1998.

[40] R. R. Tyagi, F. Aurzada, K. Lee, and M. Reisslein, "Connection Establishment in LTE-A Networks: Justification of Poisson Process Modeling," *IEEE Systems Journal*, vol. 11, no. 4, pp. 2383–2394, Dec. 2017.

[41] P. Reichl, B. Tuffin, and R. Schatz, "Logarithmic laws in service quality perception: where microeconomics meets psychophysics and quality of experience," *Telecommunication Systems*, vol. 52, no. 2, pp. 587–600, Feb. 2013.

[42] J. Navarro-Ortiz, J. M. Lopez-Soler, and G. Stea, "Quality of experience based resource sharing in IEEE 802.11e HCCA," in *2010 European Wireless Conference (EW)*, Apr. 2010, pp. 454–461.

[43] ITU-T G.114 Recommendation, "One-Way Transmission Time," 2003.

[44] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement Learning: A Survey," *Journal of Artificial Intelligence Research*, vol. 4, no. 1, pp. 237–285, May 1996.

[45] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[46] L. Ghignone and M. Barlow, "Shallow Network Training With Dynamic Sample Weights Decay - a Potential Function Approximator for Reinforcement Learning," in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2019, pp. 149–154.

[47] D. Bertsekas and J. Tsitsiklis, *Neuro-Dynamic Programming*, 01 1996, vol. 27.

[48] E. Even-Dar and Y. Mansour, "Learning rates for q-learning," *J. Mach. Learn. Res*, vol. 5, pp. 1–25, Jan. 2004.

[49] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. The MIT Press, 2018.

[50] P. Muñoz, I. de la Bandera Cascales, F. Ruiz, S. Luna-Ramírez, R. Barco, M. Toril, P. Lázaro, and J. Rodríguez, "Computationally-Efficient Design of a Dynamic System-Level LTE Simulator," *International Journal of Electronics and Telecommunications*, vol. 57, pp. 347–358, Nov. 2011.

[51] E. Bonek, "Tunnels, corridors, and other special environments," *in COST Action 231: Digital Mobile Radio Towards Future Generation Systems, C. E. Damosso, Ed. Brüssel: European Union Publications*, pp. 190–207, 1999.

[52] 3GPP Technical Report 38.901 V16.1.0, "Study on channel model for frequencies from 0.5 to 100 GHz (Release 16)," Dec. 2019.

[53] M. Gudmundson, "Correlation model for shadow fading in mobile radio systems," *Electronics Letters*, vol. 27, no. 23, pp. 2145–2146, 1991.

[54] J. Parsons, *The Mobile Radio Propagation Channel.* Pentech Press, Jan. 1992.

[55] 3GPP Technical Specification 36.101, "Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) Radio Transmission and Reception (Release 9)," Dec. 2009.

[56] A. Burr, A. Papadogiannis, and Tao Jiang, "Mimo truncated shannon bound for system level capacity evaluation of wireless networks," in *2012 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, 2012, pp. 268–272.

[57] J. 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, VTC 2003-Spring*, vol. 1, Apr. 2003, pp. 462–466 vol.1.

María Luisa Marí-Altozano received her M.S. degree in Telecommunication Engineering from the University of Málaga, Spain, in 2012. From 2013 to 2016, she was with Ericsson in a collaborative project with the University of Málaga. Since 2017, she has been working toward the Ph.D.degree with the Communication Engineering Department, University of Málaga. Her interests are focused on self-optimization of mobile radio access networks based on quality of experience.

Stephen S. Mwanje received the B.S. degree in electrical engineering from Makerere University, Kampala, Uganda, the M.S. degree in electrical engineering from the University of Rochester, Rochester, NY, USA, and the Dr. Ing. degree in multiagent co-ordination of cognitive self organizing network functions from the Integrated Communications Research Group, Ilmenau University of Technology, Ilmenau, Germany. He is a Senior Research Engineer with Nokia, where is he working on network management automation with a special focus on small cell ultra dense networks. He has more than 8 years of experience in mobile network operations, where he managed various projects on GSM/GPRS network planning, deployment, and optimization; microwave radio engineering and spectrum/interference management as well as fiber optic network planning, deployment and operations.

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.

Henning Sanneck received the Dr.-Ing. (Ph.D.) degree in electrical engineering from the Technical University of Berlin, with a thesis on voice over IP QoS in 2000. He heads the Cognitive Network Management Research Team with Nokia Bell Labs, Research, Munich, Germany. In 2001, he joined Siemens-Mobile Networks in Munich, as a Senior Research Engineer, and became a Technology Innovation Project Manager in the area of 3G Network Management in 2003. In 2007, at the formation of Nokia Siemens Networks, he head the Network Management Automation Team as a Research Manager driving self-organizing networks (SONs) concepts, IPR, and demos for LTE using analytics and policy-based management technologies. From 2014 to 2015, he acted as the Coordinator of SON research and standardization work in Nokia Networks, which has included the strategy development for the area. He has co-edited and authored a book entitled LTE SON. His current research interests are in 5G cognitive network management, in particular configuration, healing, and the operation of cognitive functions in virtualized networks.

Matías Toril received his M.S in Telecommunication Engineering and 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 150 publications in leading conferences and journals and 8 patents owned by Nokia and Ericsson. His current research interests include self-organizing networks, radio resource management and data analytics.

Carolina Gijón received her M.S. degree in Telecommunication Systems Engineering from the University of Málaga, Spain, in 2018. Currently, she is working towards the Ph.D. degree. Her research interests include self-organizing networks and radio resource management.