IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 19, 2021, accepted May 17, 2021, date of publication May 25, 2021, date of current version June 3, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3083554

Machine Learning–Based Mobility Robustness Optimization Under Dynamic Cellular Networks

MINH-THANG NGUYEN[1] AND SUNGOH KWON[2], (Senior Member, IEEE)
[1]Department of Electrical Engineering, École de technologie supérieure, Montreal, QC H3C 1K3, Canada
[2]School of Electrical Engineering, University of Ulsan, Ulsan 44610, South Korea

Corresponding author: Sungoh Kwon (sungoh@ulsan.ac.kr)

This work was supported by the 2021 Research Fund of the University of Ulsan.

ABSTRACT In this paper, we propose a machine learning−based mobility robustness optimization algorithm to optimize handover parameters for seamless mobility under *dynamic* small-cell networks. Small cells can be arbitrarily deployed, portable, and turned on and off to fulfill wireless traffic demands or energy efficiency. As a result, the small-cell network topology dynamically varies challenging network optimization, especially handover optimization. Previous studies have only considered dynamics due to user mobility in a specific *static* network topology. To optimize handovers under dynamic network topologies, together with user mobility, we propose an algorithm consisting of two steps: topology adaptation and mobility adaptation. To adapt to a dynamic topology, the algorithm obtains prior knowledge, which presents a belief distribution of the optimal handover parameters, for the current network topology as coarse optimization. In the second step, the algorithm fine-tunes the handover parameters to adapt to user mobility based on reinforcement learning, which utilizes the knowledge obtained during the first step. Under a dynamic small-cell network, we showed that the proposed algorithm reduced adaptation time to 4.17% of the time needed by a comparative machine–based algorithm. Furthermore, the proposed algorithm improved the user satisfaction rate to 416.7% compared to the previous work.

INDEX TERMS Transfer learning, distributed reinforcement learning, small cell on/off, self-organizing network, handover optimization.

I. INTRODUCTION

The small cell technologies have been considered an essential approach to high network capacity and high spectrum reuse in order to accommodate traffic demands in the fifth-generation (5G) era and beyond 5G [1], [2]. The small cell is a low-power, low-cost, and ready-to-use prototype of a base station [3], thus network providers can activate their small cells in a planned or unplanned manner in crowded places (such as shopping malls, stadiums, and downtown streets [4]) to improve the user's quality of experience.

To reduce administrative expenses from frequent parameter optimizations, a small-cell network is equipped with an autonomous solution called self-organizing network (SON) [3], [5]. The necessary for implementing SON for small-cell networks is highlighted by potential features, such as coverage and capacity optimization, energy efficiency, interference reduction, mobility load balancing (MLB), mobility robustness optimization (MRO), automatic neighbor relations, and random access optimization [6], [7]. To ensure reliable and efficient operation of small cells, the handover procedure should be optimized by MRO to guarantee seamless mobility throughout the small-cell network. The objective of MRO is minimizing handover failures due to radio link failures (RLFs) and retaining the number of ping-pongs as low as possible. Since RLFs disturb the user experience and ping-pongs induce resource-consuming signaling in a short time, RLFs and ping-pongs should both be minimized by optimizing handover parameters.

Unfortunately, small-cell network topology is non-static due to the arbitrary operation of small cells, such as powering on and off by personal usage or by network operators, to adapt to traffic loads and energy-saving demands [8]. When network topology changes, wireless environments, such as cell boundaries and interference, also vary according to the network topology, thus affecting handover optimization and performance [9]. Fig. 1 shows an example of a variable network topology when cells are switched on and off. Because

The associate editor coordinating the review of this manuscript and approving it for publication was Tiankui Zhang.

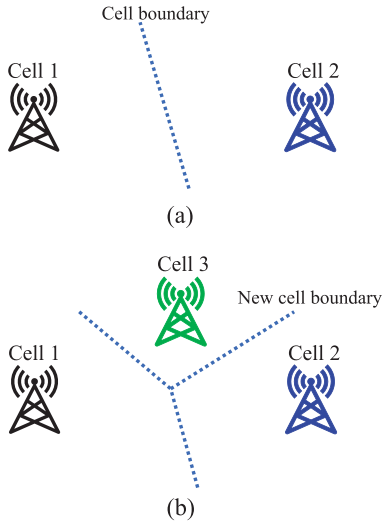


FIGURE 1. Dynamic topologies with cells switching on and off (a) two cells are switched on, and (b) three cells are switched on.

of cell 3's activation, the coverage of cells changes and handover parameters should be modified accordingly to adapt user mobility [9]. Thus, handover optimization for seamless mobility under dynamic topology and user mobility is a challenging issue in small-cell networks.

To optimize handover parameters, previous MRO algorithms have considered user mobility adaptation under *static* deployment of small cells [6], [10]–[13]. In [10]–[12], the authors improved handover performance by adjusting time-to-trigger (TTT) and handover offset. The algorithms are rule-based and take into account negotiation between RLFs and ping-pongs. A more flexible optimization policy that regulates cell individual offsets (CIOs), which considers local-specific handover optimization, was introduced in [13]. In [6], the authors proposed an algorithm that manipulates TTT, handover offset, and CIOs together to improve handover performance. The algorithm achieved an impressive handover performance without a negotiation between RLFs and ping-pongs.

To handle dynamic wireless environments, machine learning techniques have been applied to optimize handover parameters. In previous researches, reinforcement learning (RL) was applied to adjust handover parameters [14]–[17]. In the RL algorithm, the learning agent interacts with the outside environment to choose the optimal action based on a reward received from the environment [18]. Studies in [14]–[16] considered a model-free RL, which models the reward as a load level (for MLB) and handover performance (for MRO), and which chooses an action to modify handover parameters (TTT and hysteresis) for a given environmental state according to fixed policies. In [17], the authors utilized fuzzy-based policies with RL-based algorithms to change handover offsets in order to improve handover quality. Even though the algorithms in [14]–[16] adjusted handover parameters while considering dynamic user mobility, they were applied only to a *static* network topology. Furthermore, one cost of these algorithms is low convergence to adapt to the dynamic wireless environment, since they need a large number of training samples for experience. Therefore, optimizing handover parameters under dynamic small-cell networks, where network topology as well as mobility are both dynamic, still poses challenges for wireless network optimization.

In this paper, we propose a machine learning−based algorithm that includes a transfer learning–based algorithm to adapt to dynamic network topologies, along with a distributed reinforcement learning–based algorithm to adapt to time-variant UE mobility by optimizing handover parameters. The transfer learning−based algorithm harvests prior knowledge of optimization tasks, which includes an estimation or belief distribution of optimal handover parameters, for adapting to a variable topology. For adapting to user mobility, in each cell of a particular sub topology, the RL-based mobility adaptation algorithm utilizes the prior knowledge to optimize three handover parameters (TTT, hysteresis, and CIO) together. Via simulation and analysis, we verify our proposed algorithms under a dynamic topology, random mobility, and irregular deployment of small cells, and we show that the proposed algorithm can transiently optimize handover parameters to guarantee the target handover performance.

The remainder of this paper is organized as follows. Section II discusses handover models and issues in small-cell networks. Section III presents a framework of the proposed machine learning–based algorithm, which include a transferred learning–based algorithm and a reinforcement learning–based mobility adaptation algorithm. Section IV shows simulation results and comparisons with related works. Finally, we conclude this work in Section V.

II. SYSTEM MODELS

We consider a hierarchical SON that supervises small cells, as shown in Fig. 2. Two types of SONs are: the centralized SON (cSON) and the decentralized SON (dSON). While a cSON is located at network management for network-wide optimization, dSONs are implemented in the small cells for local adaptation. The cSON receives information from dSONs to monitor network-wide information in order to optimize the network globally [19]. When a new cell is activated, the dSON of this cell receives initial settings from the cSON, which includes cell identification (ID), handover parameters, neighboring cell information, and other control parameters.

To manage energy saving and the arbitrary on/off manner of small cells, dSONs send notifications to the cSON before the corresponding cells are deactivated. The cSON monitors the network topology in order to produce coarse adjustments. The cSON also communicates with smart traffic monitoring systems [20] to monitor the environmental context, such as average user speed and traffic flows on the streets. For mobility management, small cells communicate with their neighbors via Xn interface to share information [21]. Nearby small cells exchange user information, such as handover information and failure notification messages [21]. Based on

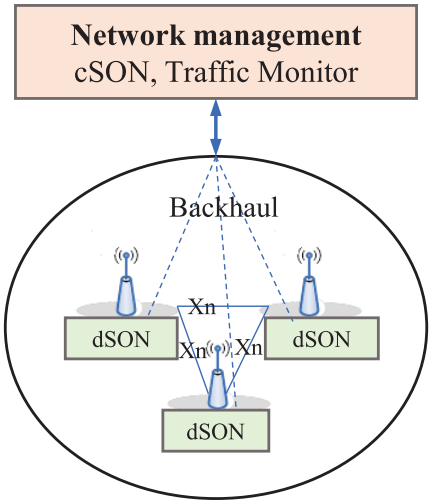


FIGURE 2. A scenario with a small-cell network and its SON.

the recorded data and the cSON's support, the dSON adapts to dynamic mobility by optimizing handover parameters locally.

Considering a nominated UE located at distance d_i (in meters) from cell i, the received power for such a distance can be modeled as

$$m_i = p_{\text{tx}} d_i^{-\alpha} \nu G_0^{-1},$$

where p_{tx} is the transmission power of a cell, G_0 is a reference value that accounts for a fixed propagation loss, α is the attenuation exponent characterizing the level of attenuation of a specific propagation environment, and ν is the fading factor. During a course of movement, the UE eventually or periodically sends signal measurements such as reference signal received power (RSRP) to the serving cell. The received measurement data are used for evaluating the quality of the wireless connection. The signal-to-interference-plus-noise ratio (SINR) is calculated based on the measurements. The SINR, γ, is defined as

$$\gamma = \frac{m_i}{\sum_{j \neq i} m_j + N_0}, \tag{1}$$

where m_i and m_j are linear values for the RSRP of the serving cell and other cells, respectively, as measured by the UE, and N_0 is thermal noise [9], [10], [22]. SINR is used by UEs to detect an RLF, which happens if the SINR remains below a predefined threshold, Q_{out}, for a certain period of time. Thus, the cell coverage is determined based on Q_{out}.

III. HANDOVER PROCEDURE PROBLEM FORMULATION

In the 3rd Generation Partnership Project (3GPP) standards, a handover procedure starts based on measurement reports from a UE to a serving cell for a handover decision. UEs periodically measure RSRPs of all discovered cells, and eventually or periodically send measurement reports to the serving cells when certain conditions hold. There are six intra-frequency event measurements, designated from A1 to A6, in the 3GPP standards [23]. Each event measurement is used for a specific application of a SON, like mobility management and neighbor discovery [6], [7], [24].

For triggering a handover, the A3 event is selected since it is based on the better relative signal quality between cells [6], [9]. The A3 event is triggered when the RSRP of a neighboring cell becomes better than that of the serving cell, based on a certain offset. The UE sends report measurements to the serving cell for a handover after the TTT timer has expired. The condition for triggering the measurement report is

$$\mathrm{M}_j + \mathrm{Ocn}_{ij} + \mathrm{Ofn}_{ij} > \mathrm{M}_i + \mathrm{Ocp}_{ij} + \mathrm{Ofp}_{ij} + \mathrm{Off} + \mathrm{Hys}, \tag{2}$$

where M_i and M_j (on the decibel (dB) scale) are the measured RSRPs of serving cell i and neighboring cell j, respectively, i.e., $\mathrm{M}_i = 10 \log m_i$ and $\mathrm{M}_j = 10 \log m_j$. Hys is the hysteresis parameter to prevent oscillation of condition (2) due to fading, and Off is A3Offset for this event. Offsets Ocn_{ij} and Ocp_{ij} are individual CIOs of cell i for cell j. While the Hys and A3Offset affect handovers to all neighbors, the CIOs assign different handover offsets to each neighbor. In this paper, we set the CIOs' Ofn_{ij} and Ofp_{ij} to zero, since we consider intra-frequency handover. Condition (2) is rewritten as

$$\mathrm{M}_j + \Delta_{ij} > \mathrm{M}_i, \tag{3}$$

where Δ_{ij} is the equivalent CIO of serving cell i for neighboring cell j, which is given as $\Delta_{ij} = \mathrm{Ocn}_{ij} - \mathrm{Ocp}_{ij} - \mathrm{Off} - \mathrm{Hys}$. In practice, cell i has many neighboring cells; hence, we denote the set of neighboring cells of cell i as H_i.

To mathematically model the handover procedure, we use geometry elements: a straight line for the user trajectory, and Apollonian circles for A3-event coverage and transmission ranges [9]. An example of a successful handover from cell i to cell j is shown in Fig. 3(a). User positions (projected from the user trajectory to the x-axis) and time stamps are denoted as an ordered pair (t_n, x_n) indicating where and when measurement events occur. The trajectory is characterized by slope angle θ and x-intercept b. Fig. 3(b) depicts a user trajectory and its intersects with Apollonian circles of A3 and RLF measurement events of two cells [9]. Specifically, C, D, E, and F are the intersections of the user trajectory and the A3-event circle of cell i, the RLF-event circle of cell i, the A3-event circle of cell j, and the RLF-event circle of cell j, respectively. The A3-event circle for the handover from cell i to cell j is adjustable according to Δ_{ij}, while the RLF-event circles of cells are different, depending on the network topology [9]. To scrutinize handover problems, we project C, D, E, and F onto the line segment between cell i and cell j, which are denoted x_{C}, x_{D}, x_{E}, and x_{F}, respectively, and derive various conditions leading to handover problems. We note that x_1 and x_2 in Fig. 3(a) are x_C and x_D in Fig. 3(b), respectively.

In Fig. 3(a), when condition for A3 event in (3) is satisfied within the TTT period at (t_3, x_3), the UE sends a measurement report to cell i triggering a handover from cell i to cell j. The measurement report carries RSRPs of all discovered cells

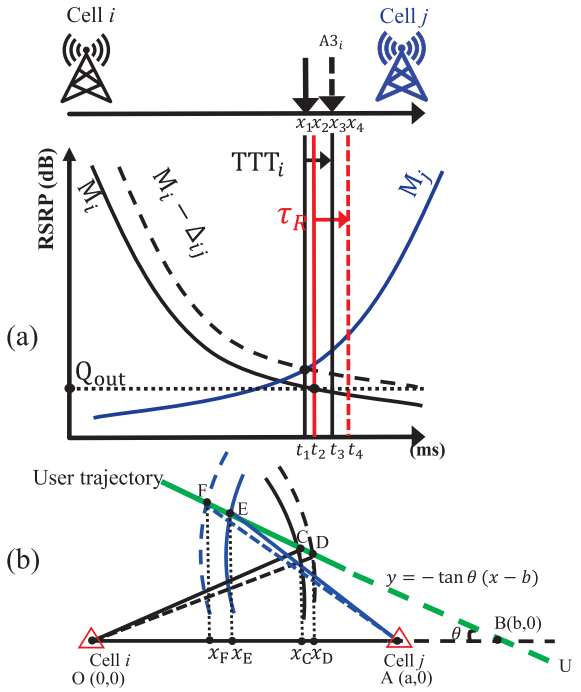


FIGURE 3. A successful handover.

where the RSRPs satisfies condition (3) with regard to cell i. Then, cell i chooses the best neighboring cell based on the reported RSRPs and triggers a handover to the chosen cell. However, if the SINR is lower than threshold Q_{out} for time duration τ_R, which is the timing length for RLF detection [9] that starts at (t_2, x_2), the wireless links are dropped leading to an RLF at time (t_4, x_4), and the handover fails.

A. UNDESIRABLE HANDOVER CONDITIONS

There are two types of undesirable handover: RLFs and ping-pongs. While ping-pongs are similar to repeated successful handovers among cells, an RLF causes failed handover attempts based due to one of three reasons: a too-late, a too-early, or a wrong-cell handover [6], [21]. Utilizing the user trajectory and parameters in Fig. 3(b), we explain conditions for undesirable handovers. A too-late handover occurs when the moving depth of the UE during TTT_i with average velocity v in the network, which is $x_{\text{C}} + v\text{TTT}_i \cos\theta$, is longer than the coverage that cell i allows for τ_R, which is $x_{\text{D}} + v\tau_{\text{R}} \cos\theta$. After an RLF, the UE tries to reconnect to cell j, which is the best neighbor of cell i for the UE. Cell j notifies cell i about the RLF via the Xn interface, and cell i recognizes a handover that was too late. The condition under which a too-late handover happens is expressed by

$$x_{\text{C}} + v\,(\text{TTT}_i - \tau_{\text{R}})\cos\theta > x_{\text{D}} \quad \text{(C1)},$$

A too-early handover happens when a UE joins target cell j too early after a successful handover, and the connection is immediately dropped because of a poor SINR ($\gamma \le Q_{\text{out}}$). This failure occurs shortly after a successful handover from cell i to cell j, and the UE camps to cell i again because it is still within cell i coverage. Therefore, the total moving depth during TTT_j and τ_R, which is $x_{\text{C}} + v\left(\text{TTT}_j + \tau_{\text{R}}\right)\cos\theta$, is too short to enter the coverage of cell j, which is at x_{F}. In addition, a duration for TTT_j that is longer than τ_R causes an RLF in cell j. After the RLF, the source cell recognizes the too-early handover based on handover history. The condition under which a too-early handover occurs is

$$x_{\text{C}} + v\left(\text{TTT}_j + \tau_{\text{R}}\right)\cos\theta \le x_{\text{F}} \text{ (C2), and } \text{TTT}_j > \tau_{\text{R}},$$

where x_{F} is the projection of F onto the line segment between cell i and cell j, as depicted in Fig. 3(b). A wrong-cell handover is detected when an RLF occurs shortly after a successful handover to the target cell, and then, the UE reconnects to another cell that is neither the serving cell nor the target cell. Since a wrong-cell handover is identical to a too-early handover except for the reconnection, we merge the wrong-cell problem into the too-early problem.

In contrast to RLFs, a ping-pong maintains the link connection. However, it repeats the handover from cell i to cell j multiple times or even among multiple cells within a short time [24]. To cause a simple ping-pong (i.e., the pattern i–j–i, which counts as one ping-pong) the moving depth during TTT_i and TTT_j, which is $x_{\text{C}} + v\left(\text{TTT}_i + \text{TTT}_j\right)\cos\theta$, should not exceed the A3-event coverage of cell j, which is at x_{E}, to guarantee a handover back to cell j again; and TTT_j must not be greater than τ_R to avoid an RLF at cell j. The condition for a ping-pong is described as

$$x_{\text{C}} + v\left(\text{TTT}_i + \text{TTT}_j\right)\cos\theta \le x_{\text{E}} \text{ (C3), and } \text{TTT}_j \le \tau_{\text{R}},$$

where x_{E} is the projection of E onto the cell i-cell j line segment, as shown in Fig. 3(b). The rapid pace of handovers wastes system resources, such as time and signaling procedures [5]. Even though ping-pongs do not cause RLFs, a regular requirement for handover optimization always includes minimizing ping-pongs as much as possible. The scenario for a ping-pong looks similar to the too-early handover problem, but ping-pongs do not cause RLFs.

B. OPTIMAL HANDOVER PARAMETERS

As explained in the conditions for undesirable handovers, when handover offset Δ_{ij} increases, too-late handovers happen less often and too-early handovers occur more often. Similarly, too-late handovers happen more often and too-early handovers occur less often when handover offset Δ_{ij} decreases. Therefore, there is a lower bound on Δ_{ij} to prevent too-late handovers, and an upper bound on Δ_{ij} to eliminate too-early handovers.

The lower bound of Δ_{ij}, $\Delta_{ij}^{\dagger}$, is obtained by solving

$$\Delta_{ij}^{\dagger} = \underset{O_{\min} \le \Delta_{ij} \le O_{\max}}{\arg\min} \Delta_{ij} \quad \text{s.t. } x_C + v\,(TTT_i - \tau_R)\cos\theta \le x_D, \qquad (4)$$

where $O_{\min}$ and $O_{\max}$ are the maximum and minimum values of Δ_{ij}, respectively. The upper bound, Δ_{ij}^{*}, of Δ_{ij}, is obtained

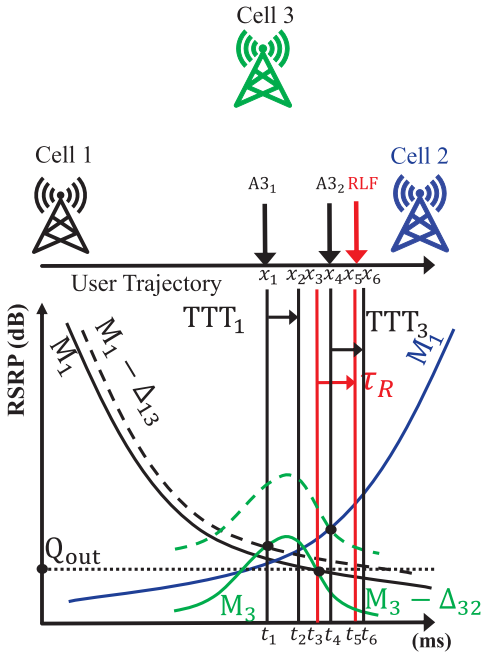


FIGURE 4. Examples of dynamic small-cell network handovers.

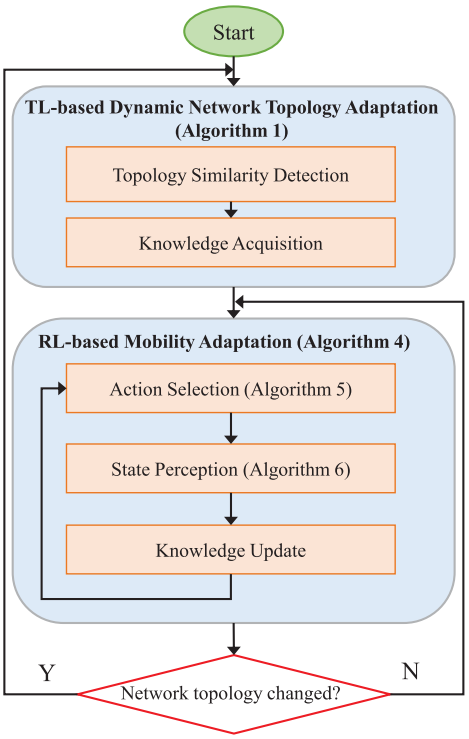


FIGURE 5. The proposed machine learning–based algorithm for dynamic small-cell networks.

as follows

$$\Delta_{ij}^{*} = \underset{O_{\min} \le \Delta_{ij} \le O_{\max}}{\arg\max} \quad \Delta_{ij}$$
$$\text{s.t.} \;\; x_{\mathrm{C}} + v\left(\mathrm{TTT}_j + \tau_{\mathrm{R}}\right)\cos\theta > x_{\mathrm{F}},$$
$$x_{\mathrm{C}} + v\left(\mathrm{TTT}_i + \mathrm{TTT}_j\right)\cos\theta > x_{\mathrm{E}}. \quad (5)$$

The optimal range for Δ_{ij} exists when $\Delta_{ij}^{\dagger} \le \Delta_{ij}^{*}$. The condition for the existence of the optimal range of handover parameters was explained in Lemma 1 in [9]. If the optimal range exists (i.e., $\Delta_{ij}^{\dagger} \le \Delta_{ij}^{*}$), optimal handover parameters is chosen in between the lower bound $\Delta_{ij}^{\dagger}$ and the upper bound Δ_{ij}^{*}. Otherwise (i.e., $\Delta_{ij}^{\dagger} > \Delta_{ij}^{*}$), we would probably choose a value based on a predefined policy. In this paper, we chose the average of the lower bound and the upper bound of the optimal handover offset as the approximate optimal value, which is $(\Delta_{ij}^{\dagger} + \Delta_{ij}^{*})/2$, regardless of the existence of the optimal range.

C. PROBLEM FORMULATION FOR DYNAMIC NETWORK TOPOLOGY

The dynamic on/off switching characteristic of small cells affects handovers, since the wireless environment (e.g., cell boundaries, topology, and interference) changes accordingly. In Fig. 4, we show an example of a handover failure where a new cell (cell 3) is powered on for a given network with two small cells. In Fig. 3(a), the UE should be handed over from cell 1 to cell 2 if there are only two cells. Due to the activation of cell 3, the UE is handed over to cell 3 at (t_1, x_1). However, at (t_6, x_6), the UE cannot be handed over successfully from cell 2 to cell 3. This is because handovers among cell 1, cell 2, and cell 3 are no longer optimized due to the different topology.

We define the topology $\mathcal{T}$ of a small-cell network as a set of cells that are identified by cell ID as well as cell location. For instance, $\mathcal{T} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ means that there are 10 cells with IDs from 1 to 10 deployed in the small-cell network. Also, we define a cluster of neighboring cells, $\mathcal{T}_i$, as a sub-topology of cells in the small-cell network such that $\mathcal{T} = \cup_{i=1}^{K} \mathcal{T}_i$ if there are K sub-topologies. For example, $\mathcal{T}_1 = \{1, 2, 3\}$ means that sub-topology $\mathcal{T}_1$ contains three neighboring cells: cell 1, cell 2, and cell 3.

We formulated a handover optimization problem considering the dynamic network topology as:

$$\begin{aligned} &\text{minimize } f(\mathrm{TTT}, \mathrm{Ocn}, \mathrm{A3Offset}, \mathcal{T}) \\ &\text{subject to } 0\,\mathrm{ms} \le \mathrm{TTT} \le 5120\,\mathrm{ms}, \\ &\qquad -24\,\mathrm{dB} \le \mathrm{Ocn} \le 24\,\mathrm{dB}, \\ &\qquad -15\,\mathrm{dB} \le \mathrm{A3Offset} \le 15\,\mathrm{dB}, \end{aligned}$$

where $\mathcal{T}$ is the network topology, and f is a cost function of handover optimization. In this paper, the cost function takes into account a statistic of undesirable handovers (e.g., too-late, too-early, and ping-pong handovers).

IV. MACHINE LEARNING-BASED MRO

To minimize undesirable handovers in dynamic small-cell networks, we propose a machine learning–based MRO algorithm. The flowchart of the proposed algorithm is described in Fig. 5. The algorithm consists of two steps: topology

adaptation at cSON and mobility adaptation at dSONs. A cSON obtains prior knowledge of optimization tasks. The prior knowledge can be an estimation or a belief distribution of the optimal handover parameters for the new network topology. The cSON dispatches the prior knowledge to the dSONs to speed up convergence of the learning processes at the dSONs. Then, the dSONs fine tune the handover parameters to quickly adapt to mobile environments.

A. TRANSFER LEARNING–BASED DYNAMIC NETWORK TOPOLOGY ADAPTATION ALGORITHM

To adapt to a dynamic network topology, we apply the concept of transfer learning, which exploits prior knowledge about one learning task to solve related tasks in order to achieve better optimization and fast convergence. The TL concept is defined as follows. Given source domain D^{s}, learning task L^{s}, a target domain D^{t}, and the learning task L^{t}, transfer learning enhances the accuracy of the target predictive model in D^{t} using the knowledge from D^{s} and D^{t} when $D^{\mathrm{s}} \neq D^{\mathrm{t}}$ and/or $L^{\mathrm{s}} \neq L^{t}$ [25]. To apply TL, we choose network topology $\mathcal{T}$ as the domain that varies dynamically, and the learning task is identical for all network topologies, which optimizes handover parameters to adapt to the dynamic wireless environment. Our ultimate goal is to transiently optimize handovers for a new network topology by utilizing knowledge about previous topologies. The TL-based algorithm addresses three questions: *when* prior knowledge is needed, *how* that prior knowledge is established, and *what* prior knowledge is transferred between different learning tasks.

1) TOPOLOGY SIMILARITY DETECTION

Prior knowledge is transferred *when* a network topology changes due to cells switching on and off. When the network topology changes, cell borders are different changing the handover situation. Our proposed algorithm utilizes prior knowledge to transiently optimize parameters of dynamic networks while previous algorithms focused on a static network topology and neglected prior knowledge. At the cSON, the TL-based algorithm extracts *features* from the old topology, $\mathcal{T}^{\mathrm{old}}$, which are similar to the new topology, $\mathcal{T}^{\mathrm{new}}$. The similarity between the old and new topologies is determined based on small cells' locations and their neighbors. A cell in the new topology is matched to a cell in the old topology if their neighboring cells are similar; otherwise, mismatched despite of the same cell identifications.

To determine *how* similar topology $\mathcal{T}^{\mathrm{new}}$ is to topology $\mathcal{T}^{\mathrm{old}}$, the proposed algorithm uses the locations and neighbors of the recent on-and-off cells, which are stored in set $\mathcal{U}_{\mathrm{ON/OFF}}$. For each cell i in $\mathcal{U}_{\mathrm{ON/OFF}}$, the algorithm groups cell i with its neighboring cells in topology $\mathcal{T}^{\mathrm{new}}$ to form a sub-topology ($\mathcal{T}_i$). The dissimilar parts between $\mathcal{T}^{\mathrm{old}}$ and $\mathcal{T}^{\mathrm{new}}$ are sub-topologies that contain the recent on/off cells, whereas the similar parts are sub-topologies that do not have neighboring relations to the recent on/off cells.

Algorithm 1 TL-Based Dynamic Network Topology Adaptation Algorithm

```
Select cells that were recently on and off in T^old to make set U_ON/OFF.
for cell i ∈ U_ON/OFF do
    Group cell i with its neighboring cells to obtain sub-topology T_i
    if T_i ∈ T_DB then
        Transfer prior knowledge to cells in T_i
    else
        Transfer the estimated optimal handover parameters to cells in T_i (Algorithm 3)
        Store T_i in T_DB
    end if
end for
Retain the current handover parameters and knowledge for the remaining cells in T^new
Go to the mobility adaptation algorithm by dSONs (Algorithm 4)
```

Algorithm 2 Geometry-Based Computation for the Upper Bound and Lower Bound of CIO

```
Input: TTT_i, Δ_ji, TTT_j, b, θ, v, and locations of cells
Output: Δ_ij^† and Δ_ij^*
Compute positions D and F with γ_min, and E with Δ_ji
for each Δ_ij value in a learning range do
    Compute position C with Δ_ij
    if (C1) is satisfied then
        Δ_ij^† ← Δ_ij and break
    end if
end for
for each Δ_ij value in a learning range do
    Compute position C with Δ_ij
    if (C2) and (C3) are satisfied then
        Δ_ij^* ← Δ_ij and break
    end if
end for
```

2) KNOWLEDGE ACQUISITION

When the similarities and dissimilarities between $\mathcal{T}^{\mathrm{old}}$ and $\mathcal{T}^{\mathrm{new}}$ are obtained, to determine *what* knowledge should be transferred to cells in $\mathcal{T}^{\mathrm{new}}$, Algorithm 1 searches sub-topologies $\mathcal{T}_i$ that contain the recent on/off cells throughout the topology database $\mathcal{T}_{\mathrm{DB}}$. Database $\mathcal{T}_{\mathrm{DB}}$ contains the structures of sub-topologies that have already appeared, and prior knowledge, which are belief distributions of optimal handover parameters for each stored sub-topology. If $\mathcal{T}_i$ is in $\mathcal{T}_{\mathrm{DB}}$, the cSON sends the prior knowledge to each cell in $\mathcal{T}_i$. Otherwise, the cSON forwards estimated optimal handover parameters to each cell in $\mathcal{T}_i$ and stores $\mathcal{T}_i$ in $\mathcal{T}_{\mathrm{DB}}$ for future queries. For similar parts between $\mathcal{T}^{\mathrm{old}}$ and $\mathcal{T}^{\mathrm{new}}$, current parameters and knowledge are kept unchanged.

If prior knowledge is unavailable for new sub-topologies, the estimation algorithm estimates the optimal handover

Algorithm 3 Optimal Handover Parameterd Estimation Algorithm

1: Input: TTT_i, Δ_{ji}, TTT_j, a range of θ, a range of b, and locations of cells
2: Output: $\hat{\Delta}_{ij}^{\text{opt}}$ and $\hat{\Delta}_{ji}^{\text{opt}}$
3: $\hat{\Delta}_{ji}^{\text{opt}} \leftarrow \Delta_{ji}$
4: **for** each value of θ and b in the specified ranges **do**
5: Compute Δ_{ij}^{*} and $\Delta_{ij}^{\dagger}$ for cell i given $\hat{\Delta}_{ji}^{\text{opt}}$ (Algorithm 2)
6: **end for**
7: Compute $\hat{\Delta}_{ij}^{\dagger}$ and $\hat{\Delta}_{ij}^{*}$ via (6) and (7)
8: $\hat{\Delta}_{ij}^{\text{opt}} \leftarrow \left(\hat{\Delta}_{ij}^{*} + \hat{\Delta}_{ij}^{\dagger}\right)/2$
9: **for** each value of θ and b in the specified ranges **do**
10: Compute Δ_{ji}^{*} and $\Delta_{ji}^{\dagger}$ for cell j given $\hat{\Delta}_{ij}^{\text{opt}}$ (Algorithm 2)
11: **end for**
12: Compute $\hat{\Delta}_{ji}^{\dagger}$ and $\hat{\Delta}_{ji}^{*}$ via (6) and (7)
13: $\hat{\Delta}_{ji}^{\text{opt}} \leftarrow \left(\hat{\Delta}_{ji}^{*} + \hat{\Delta}_{ji}^{\dagger}\right)/2$
14: Return to Line 4 until $\hat{\Delta}_{ij}^{\text{opt}}$ and $\hat{\Delta}_{ji}^{\text{opt}}$ converged, then round them to the nearest value standardized by 3GPP.

parameters of all cell pairs in the network while considering other cells' locations, because the optimal parameters vary according to cell locations [9]. Considering two nominated neighboring cells, cell i and cell j ($i, j \in \mathcal{T}_i$), to estimate optimal parameters for handovers by cell i to cell j, the lower bound ($\Delta_{ij}^{\dagger}$) and the upper bound (Δ_{ij}^{*}) of Δ_{ij} are estimated by solving optimization problems (4) and (5), respectively (Algorithm 2). Problems (4) and (5) are formulated with the condition (C1), (C2), and (C3), and thus, the solutions depend on user trajectory, cell locations and settings, and TTT.

To find the optimal range of a CIO, positions D, E, and F are computed for a given user trajectory determined by tuple (θ, b). Then, we can find $\Delta_{ij}^{\dagger}$ by trying each offset value (i.e., alternating position C) until condition (C1) is satisfied. Similarly, Δ_{ij}^{*} is find by trying each offset value until condition (C2) and (C3) are met. Bound values $\Delta_{ij}^{\dagger}$ and Δ_{ij}^{*} depend on various parameters, such as θ, b, Δ_{ji}, TTT_i, and TTT_j. Hence, we express $\Delta_{ij}^{\dagger}$ and Δ_{ij}^{*} as $f_{ij}^{\dagger}(\theta, b, \text{TTT}_i)$ and $f_{ij}^{*}(\theta, b, \Delta_{ji}, \text{TTT}_i, \text{TTT}_j)$, respectively, to represent such dependencies. Since a user trajectory varies according to (θ, b), we estimate the lower bound and the upper bound for the CIO of cell i for cell j as follows[1]

$$\hat{\Delta}_{ij}^{\dagger} = E\left[f_{ij}^{\dagger}(\Theta, B, TTT_i)\right], \tag{6}$$

$$\hat{\Delta}_{ij}^{*} = E\left[f_{ij}^{*}(\Theta, B, TTT_i, \Delta_{ji}, TTT_j)\right], \tag{7}$$

where E is the expectation, and random variables Θ and B, respectively, take values of θ and b following uniform distributions [9], [26]. The estimated optimal value of Δ_{ij} ($\hat{\Delta}_{ij}^{\text{opt}}$) is the average value of $\hat{\Delta}_{ij}^{\dagger}$ and $\hat{\Delta}_{ij}^{*}$, i.e., $\hat{\Delta}_{ij}^{\text{opt}} = \left(\hat{\Delta}_{ij}^{*} + \hat{\Delta}_{ij}^{\dagger}\right)/2$.

[1] A closed form of optimal CIO values were presented in Theorem 3 in [9]

Algorithm 4 Distributed Reinforcement learning−based Mobility Adaptation Algorithm

1: **if** prior knowledge are available **then**
2: Initialize handover parameters according to the transferred knowledge from cSON by (12)
3: **else**
4: Initialize handover parameters according to the estimated parameters
5: **end if**
6: Select action (Algorithm 5)
7: Receive data $N_{\text{L}_{ij}}$, $N_{\text{E}_{ij}}$, and $N_{\text{PP}_{ij}}$,$\forall j \in H_i$
8: **if** timer for receiving handover statistics has expired **then**
9: Compute $\mathcal{R}_{\text{L}_{ij}}$, $\mathcal{R}_{\text{E}_{ij}}$, RR_i, and $\mathcal{R}_{ij}$ by (9), (10), (8), and (11), respectively
10: Measure environment state s_{ij}, $\forall j \in H_i$ (Algorithm 6)
11: Update knowledge by (14)
12: **if** $RR_i > \text{Th}_{RR}$ **then**
13: Select action a_{ij}, $\forall j \in H_i$ (Algorithm 5)
14: **end if**
15: Reset the timer and flush all counters of cell i, then return to Line 6
16: **else**
17: **if** network topology changes **then**
18: Go to the TL-based algorithm (Algorithm 1)
19: **else**
20: Return to Line 6
21: **end if**
22: **end if**

To optimize handovers from cell j to cell i, the algorithm finds $\hat{\Delta}_{ji}^{\text{opt}}$ for cell j using a similar process similar to the given $\hat{\Delta}_{ij}^{\text{opt}}$. The processes for computing $\hat{\Delta}_{ij}^{\text{opt}}$ and $\hat{\Delta}_{ji}^{\text{opt}}$ are repeated until convergence. The results are rounded to the nearest values that follow the 3GPP standard [23]. Algorithm 3 estimates the optimal handover parameters for one cell pair. When the estimation is completed, the algorithm selects another cell pair in $\mathcal{T}_i$ until completing all the cell pairs in $\mathcal{T}_i$. Utilizing the prior knowledge, the distributed mobility adaptation algorithm adjusts handover parameters.

B. DISTRIBUTED REINFORCEMENT LEARNING-BASED MOBILITY ADAPTATION ALGORITHM

The mobile environment also varies due to UE mobility, and therefore, we propose an RL-based algorithm for dSONs (Algorithm 4) to fine-tune handover parameters given the knowledge transferred from the cSON, which is a estimation or a belief distribution of optimal handover parameters. Basically, the algorithm perceives the environment state based on handover optimization costs (e.g., too-late handovers, too-early handovers, and ping-pongs), and chooses an action to optimize handover parameters (e.g., TTT and CIOs) in accordance with the measured state by utilizing prior knowledge. The knowledge is updated based on optimization costs to reflect the belief distribution of optimal values for the given state.

At the dSON, the mobility adaptation algorithm initializes the handover parameters based on the knowledge transferred from the cSON (Algorithm 5). Then, the algorithm computes the optimization costs after the timer for handover statistics has expired. The costs are the ratios of the number of undesirable handovers (e.g., too-late and too-early handovers) to the total number of handover attempts. Based on the costs, the algorithm perceives the environment states (Algorithm 6) to choose an action in order to adjust parameters accordingly. Furthermore, with the computed costs and the environment state, the algorithm updates the knowledge to adapt to the environment.

To check whether parameter adjustments are necessary, the algorithm compares the RLF rate with the target performance Th_{RR}. The RLF rate for handovers from cell i to all neighboring cells, RR_i, is given as

$$RR_i = \frac{\sum_{j\in H_i} N_{\mathrm{TL}_{ij}} + N_{\mathrm{TE}_{ij}}}{\sum_{j\in H_i} N_{\mathrm{total}_{ij}}}, \tag{8}$$

where $N_{\mathrm{TL}_{ij}}$, $N_{\mathrm{TE}_{ij}}$, and N_{total_i} are the numbers of too-late handovers, too-early handovers, and all handover attempts from cell i to cell j, respectively. If RR_i exceeds Th_{RR}, Algorithm 5 selects optimal actions to adjust the handover parameters of cell i in order to handle mobility. After adjusting handover parameters, the adaptation algorithm resets the timer and flushes all the counters for handover statistics. If the network topology changes, the current optimization process switches to the TL-based algorithm (Algorithm 1). The mobility adaptation algorithm is described in details after the algorithm elements definitions.

1) ELEMENTS OF THE ALGORITHM

To adapt to mobility from cell i to cell j, the mobility adaptation algorithm includes four elements: environment state $s_{ij} \in \mathcal{S}$, which is a finite set of environment states; action $a_{ij} \in \mathcal{A}(s_{ij})$, which is a finite set of actions that depend on the current state, s_{ij}; database $Q(a_{ij})$, which provides the belief distribution of optimal parameters; and cost function $\mathcal{R}_{ij}$, which reflects the quality of an action in a given state. Basically, the algorithm chooses the optimal action in set $\mathcal{A}(s_{ij})$ for state s_{ij} to minimize the handover cost based on a belief distribution of optimal actions. The belief distribution of optimal actions is updated using optimization costs. Environment states, actions, and the cost function are defined step-by-step in the following paragraphs.

Environment state $s_{ij} \in \mathcal{S}$ is sensed through a perception stage based on the handover situation to minimize undesirable handovers. The set $\mathcal{S}$ consists of four states ($\mathrm{S}_{\mathrm{cio}^+}$, $\mathrm{S}_{\mathrm{cio}^-}$, $\mathrm{S}_{\mathrm{ttt}^+}$, and $\mathrm{S}_{\mathrm{ttt}^-}$) that indicate the situation when we need to increase the CIO, decrease the CIO, increase TTT, and decrease TTT, respectively, from their current values, to handle too-late and too-early handover issues.

Action $a_{ij} \in \mathcal{A}(s_{ij})$ adjusts the values of TTT and CIO to minimize undesirable handovers according to current state s_{ij}. Therefore, set $\mathcal{A}(s_{ij})$ varies based on state s_{ij}. The cost of too-late handovers from cell i to cell j, $\mathcal{R}_{\mathrm{L}_{ij}}$, is given as

$$\mathcal{R}_{\mathrm{L}_{ij}} = \frac{N_{\mathrm{TL}_{ij}}}{N_{\mathrm{total}_{ij}}}, \tag{9}$$

The cost of too-early handovers from cell i to cell j is defined as

$$\mathcal{R}_{\mathrm{E}_{ij}} = \frac{N_{\mathrm{PP}_{ij}} + N_{\mathrm{TE}_{ij}}}{N_{\mathrm{total}_{ij}}} \tag{10}$$

where $N_{\mathrm{PP}_{ij}}$ denotes the number of ping-pongs from cell i to cell j. The cost for a handover from cell i to cell j, $\mathcal{R}_{ij}$, taking into account RLFs and ping-pongs, is given as

$$\mathcal{R}_{ij} = \mathcal{R}_{\mathrm{L}_{ij}} + \mathcal{R}_{\mathrm{E}_{ij}}. \tag{11}$$

Cost $\mathcal{R}_{ij}$ is used to update the knowledge of the dSON at cell i for neighboring cell j.

The knowledge of cell i for cell j is stored in a Q-value database,[2] $Q(a_{ij})$, that maps a parameter value to a real value. There are one Q-database for TTT and at least one Q-database for CIO because more than one neighboring cell of a cell exists. For future usage of prior knowledge, a dSON dispatches the Q-databases to the cSON after updating them. In the next subsections, we explain how the adaptation algorithm selects optimal actions, determines environment states, and updates Q-databases.

Algorithm 5 Action Selection

```
Input: Environment state s_ij, ∀j ∈ H_i
Output: Optimal action a*_ij
if s_ij == S_ttt+ then
    Increase TTT from the current TTT action based on (13)
else if s_ij == S_ttt- then
    Decrease TTT from the current TTT action based on (13)
else if s_ij == S_cio+ then
    Increase Δ_ij from the current Δ_ij action based on (13)
else
    Decrease Δ_ij from the current Δ_ij action based on (13)
end if
```

2) ACTION SELECTION

At the beginning of the network topology, if prior knowledge is available, the optimal action a^*_{ij} of cell i for neighboring cell j is selected based on a greedy strategy, which is

$$a^*_{ij} = \arg\min_{a_{ij}} Q(a_{ij}) \tag{12}$$

[2] A Q-value database can be implemented as a table, a neural network (such as a deep neural network) or a convolutional neural network according to the complexity of the applications [18]. In this study, we apply the table-based Q-value database for small-size, discrete-state-space reinforcement learning, since we applied a state quantization method and proposed an algorithm for estimating optimal handover parameters. Future work can apply the neural network–based structure to handle a large or continuous state space, such as an application for prediction of mobility patterns for efficient handovers [27].

Algorithm 6 State Perception

Input: $\mathcal{R}_{L_{ij}}$ and $\mathcal{R}_{E_{ij}}, \forall j \in H_i$
Output: Environment state $s_{ij}, \forall j \in H_i$
for each neighboring cell $j \in H_i$ **do**
  if $\mathcal{R}_{L_{ij}} \geq \mathcal{R}_{E_{ij}}$ **then**
    $s_{ij} \leftarrow \mathrm{S}_{\mathrm{cio}^+}$
  else
    $s_{ij} \leftarrow \mathrm{S}_{\mathrm{cio}^-}$
  end if
end for
if $s_{ij} == \mathrm{S}_{\mathrm{cio}^+}, \forall j \in H_i$ **then**
  $s_{ij} \leftarrow \mathrm{S}_{\mathrm{ttt}^-}, \forall j \in H_i$
else if $s_{ij} == \mathrm{S}_{\mathrm{cio}^-}, \forall j \in H_i$ **then**
  $s_{ij} \leftarrow \mathrm{S}_{\mathrm{ttt}^+}, \forall j \in H_i$
end if

Otherwise, the dSON applies the estimated optimal handover parameters that were transferred by the cSON.

For forthcoming optimizations, an optimal action is chosen based on a softmax policy given the perceived environment state. An action is selected based on a belief distribution. An action with a higher probability is more likely to be selected because it is expected to bring better handover performance in the future. The probability of taking an action is calculated from the Gibbs (or Boltzmann) distribution [18]. For neighboring cell j of cell i, each action $a_{ij} \in \mathcal{A}(s_{ij})$ for state s_{ij} is assigned a probability of selection, $p(s_{ij}, a_{ij})$, as follows

$$p(s_{ij}, a_{ij}) = \frac{\exp\left(\frac{-Q(a_{ij})}{\tau}\right)}{\sum_{b_{ij} \in \mathcal{A}(s_{ij})} \exp\left(\frac{-Q(b_{ij})}{\tau}\right)}, \tag{13}$$

where τ is a positive parameter called *temperature*. A higher temperature causes the actions to have a more equal probability, which encourages state space exploration. A lower temperature leads to a greater difference in the selection probability for actions, which encourages utilizing prior knowledge. Note that, we apply negative Q-values for the softmax policy, since it is preferred to a parameter with a low Q-value rather than parameters with a higher Q-value. The lower the Q-value for a parameter, the better the handover performance the parameter is expected to achieve. After choosing optimal actions, the algorithm waits to obtain handover optimization costs and environment states from the mobile environment and updates the knowledge.

3) STATE PERCEPTION

The adaptation algorithm translates the optimization costs into environment states (Algorithm 6). When a too-late problem is dominant for handovers to neighboring cell j (i.e., $\mathcal{R}_{L_{ij}} \geq \mathcal{R}_{E_{ij}}$), the environment state for handovers to cell j is $\mathrm{S}_{\mathrm{cio}^+}$, which demands a CIO increment. When a too-early problem is dominant for handovers to cell j ($\mathcal{R}_{L_{ij}} < \mathcal{R}_{E_{ij}}$), the environment state for handovers to cell j becomes $\mathrm{S}_{\mathrm{cio}^-}$ to decrease the CIO. If the environment state for all the neighboring cells of cell i is $\mathrm{S}_{\mathrm{cio}^+}$, the environment state for each neighboring cell is $\mathrm{S}_{\mathrm{ttt}^-}$ for TTT decrement. This is because the TTT adjustment affects handovers to all the neighboring cells of a cell. If the environment state for all the neighboring cells is $\mathrm{S}_{\mathrm{cio}^-}$, the environment state for each neighboring cell is $\mathrm{S}_{\mathrm{ttt}^+}$ for increasing TTT.

4) KNOWLEDGE UPDATE

Q-databases are loaded with prior knowledge when a network topology changes and prior knowledge is available at the cSON for the current network topology. During mobility adaptation, to update the knowledge of a dSON about cell i for neighboring cell j, $Q(a_{ij})$, we apply a temporal difference method that considers observed environment state s_{ij}, current action a_{ij}, and optimization cost $\mathcal{R}_{ij}$ [18]. The knowledge update is as follows

$$Q(a_{ij}) = (1-\beta)Q(a_{ij}) + \beta(\mathcal{R}_{ij} + \lambda \max_{a'_{ij} \in A(s'_{ij})} Q(a'_{ij})), \tag{14}$$

where $\beta \in (0, 1]$ is the learning factor and $\lambda \in [0, 1]$ is the discount rate. A β learning factor of 1 means that the latest knowledge is considered, while the prior knowledge is ignored. A β of zero means there is no learning at all. Discount rate λ that is close to 1 increases the importance of the prior knowledge in the received cost. After Q-databases have been updated, they are dispatched to the cSON for future utilization with dynamic network topologies.

V. NUMERICAL RESULTS AND DISCUSSION

A. SIMULATION ENVIRONMENT

To evaluate the performance of the proposed algorithm under dynamic networks, we consider a network of 12 small cells [28], where the topology periodically changes, as shown in Fig. 6.[3] In practice, small cells can be turned on and off depending on specific strategies, such as energy efficiency and load balancing [29]–[31]. We show the cell numbers in Fig. 6(d) for readers to easily compare different network topologies in different periods, where each period is 20 minutes long. For user mobility, we applied a Manhattan grid mobility model with different user speed (5 km/h and 30 km/h). The probability of going straight is 0.5, and the probability of taking a left or a right is 0.25 each. The number of UEs in the simulation was 200. For the default settings, TTT, hysteresis, A3Offset, and Ocn were set to 256 ms, 3 dB, 0 dB, and 0 dB, respectively [26]. We considered the learning range for TTT and CIO to be from 0 ms to 480 ms and from -4 dB to 4 dB, respectively, because outliers easily lead to undesirable handovers [6], [9].

For a channel model, signal decaying factor α and G_0 were selected at 4.33 and $10 \log 14.74$ [32]. To model the fading effect, a log-normal random variable together with a Rayleigh factor was integrated into the propagation model. The fading

[3]For denser small-cell network scenarios, Algorithm 1 groups cells to make sub topologies to optimize handovers. Furthermore, Algorithm 4 fine tuning parameters to adapt mobility regardless of network topology. Thus, our algorithm is applicable to ultra-dense small-cell networks.

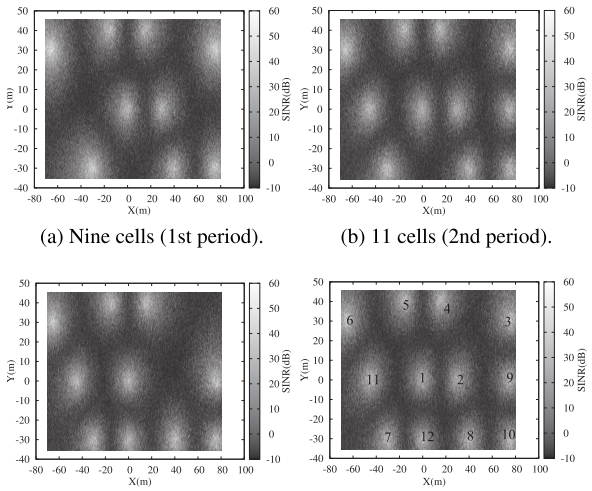


FIGURE 6. The dynamic topology of a small-cell network.

variable had a zero mean, and standard deviation σ was 6 dB, as in a typical environment [33]. An omni-directional antenna was used for all small cells, and transmit power p_{tx} was 23 dBm. For RLF detection, out-of-sync threshold Q_{out} was set at -4 dB, and RLF detection time τ_{R} was 500 ms [9]. For estimating the optimal handover parameters, we derived ranges for θ and b by uniformly sampling distributions $(0, \pi/3)$ and $(0, a)$, respectively, where a is the inter-site distance between two neighboring cells [9]. For the adaptation algorithm, β, λ, and τ were set at 0.1, 0.95, and 1, respectively [34], for utilization of the transferred knowledge.

To evaluate the proposed transfer learning–based MRO algorithm (MRO-ML), two base lines were considered: the MRO algorithm based on classification (MRO-ABC) [6], which adjusts TTT, CIO, and A3Offset to adapt to mobility; and a Q-learning-based MRO (MRO-Q) [16], which is a model-free RL algorithm for adjusting TTT and A3Offset. The target performance for the considered algorithms was set at 1 % [6].

B. HANDOVER PERFORMANCE UNDER DYNAMIC NETWORK TOPOLOGY

Fig. 7 shows the performance in terms of RLFs and ping-pongs in a 5 km/h environment. We observe that MRO-ML satisfies the target performance, improves handover performance more than MRO-Q and MRO-ABC, and optimizes RLFs and ping-pongs together. MRO-ML experiences the lowest rise in RLF at the beginning of each period. This is because MRO-ML utilizes prior knowledge of the optimal handover parameters for the new topology just before the new cells are switched on and off.

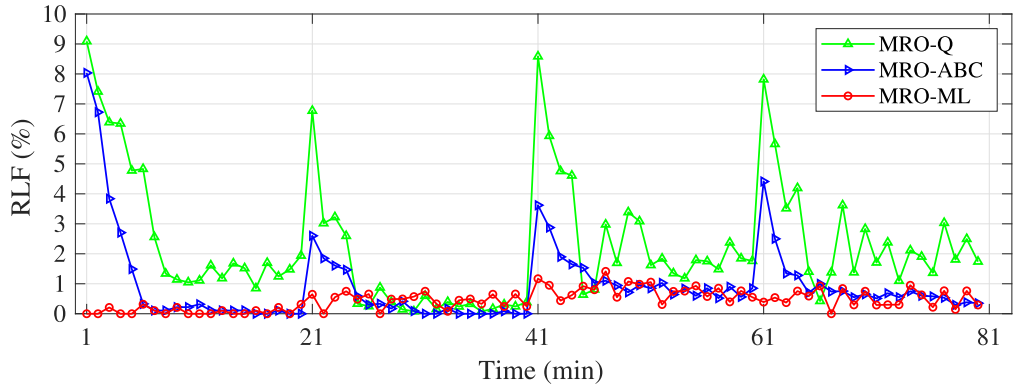
To evaluate convergence rate of the algorithms under dynamic a network topology, we computed the average adaptation time, which is from the beginning of a topology until the first time when the algorithm meets the target performance, for four considered topologies. The adaptation times of the algorithms are shown in Table 1. The results show that MRO-ML owns the shortest adaptation time, 0.25 minutes on average, to meet the target performance, while MRO-ABC and MRO-Q needed 4.75 minutes and six minutes on average, respectively. With respect to MRO-Q, which is based on the model-free RL, MRO-ML reduced the adaptation time to 4.17% of MRO-Q, while that of MRO-ABC was 79.17% of MRO-Q. MRO-ML achieved such a significant improvement thanks to the optimal parameters' estimation for a new topology, where prior knowledge is unavailable. Since MRO-ML achieved fast adaptation to topology changes, it is applicable to dynamic small-cell networks.

To examine how the algorithms fulfill the target handover performance, we calculated a satisfaction rate, which is the fraction of time in which handover performance is below the target performance. The results are summarized in Table 2. We observed that MRO-ML satisfies the handover performance requirements for all the topologies with the best satisfaction rate, 93.75% on average, while MRO-ABC and MRO-Q attained 72.5% and 22.5%, respectively, on average. Regarding MRO-Q, MRO-ML improved the satisfaction rate by 416.7%, while MRO-ABC improved it by 322.2%. MRO-ML attained such a remarkable improvement because it fine-tunes the handover parameters around the optimal parameters based on transferred knowledge, which provides information on optimal values. Unlike MRO-ML, MRO-ABC and MRO-Q adjust the handover parameters independent of topology changes. Regarding ping-pong performance, MRO-ML and MRO-ABC were stable, while MRO-Q fluctuated.

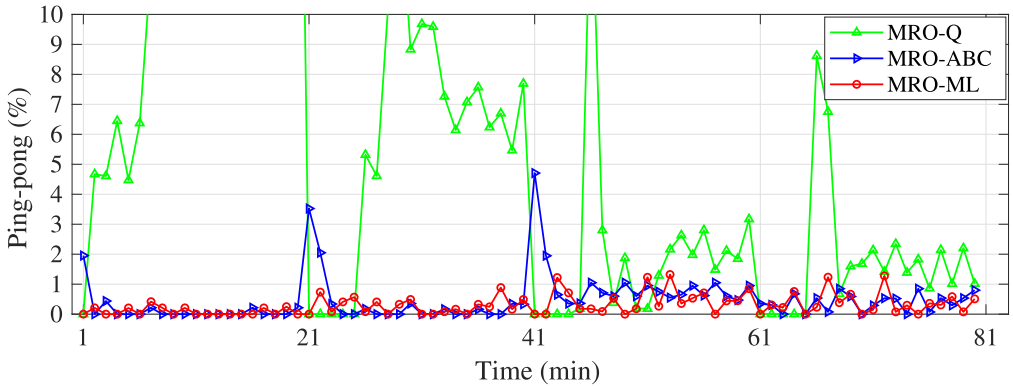
We also studied the impact of user speed to handover optimization in a dynamic topology. Fig. 8 visualizes the RLF and ping-pong rates of the algorithms when speed was 30 km/h. Adaptation time and satisfaction rate are presented in tables 3. MRO-ML owns the shortest adaptation time, 0.75 minutes on average, to meet the handover performance target, while MRO-ABC and MRO-Q spent 3.25 minutes and 2.25 minutes on average, respectively, to adapt to topology changes. For the satisfaction of UEs, MRO-ML achieved a notable satisfaction rate of 96.25%. Regarding ping-pong performance, MRO-ML was close to MRO-ABC, while MRO-Q had more fluctuations. With respect to the 5 km/h environment, the algorithms experienced fewer ping-pongs because the upper bound of the handover offset, which is to avoid too-early handovers and ping-pongs, increases when the user speed increases [9]. Overall, the results show that MRO-ML works well at a higher speed and under a dynamic network topology.

C. IMPACT OF THE USER MOBILITY MODEL

To verify the impact of a random mobility model on handover performance, we chose a random waypoint (RWP) mobility model to simulate a pedestrian environment. UEs moved at a speed of 5 km/h in the scenario of 12 active cells (Fig. 6(d)). Simulation time was 30 minutes long to check how the algorithms handle randomness in mobility.



(a) RLF.



(b) Ping-pong.

FIGURE 7. Handover performance for a speed of 5 km/h.

TABLE 1. Adaption times of the algorithms (in minutes) for a speed of 5 km/h.

	9 cells	11 cells	10 cells	12 cells	Average	Reduction (%)
MRO-Q	10	4	5	5	6.0	100
MRO-ABC	5	4	6	4	4.75	79.17
MRO-ML	0	0	1	0	0.25	4.17

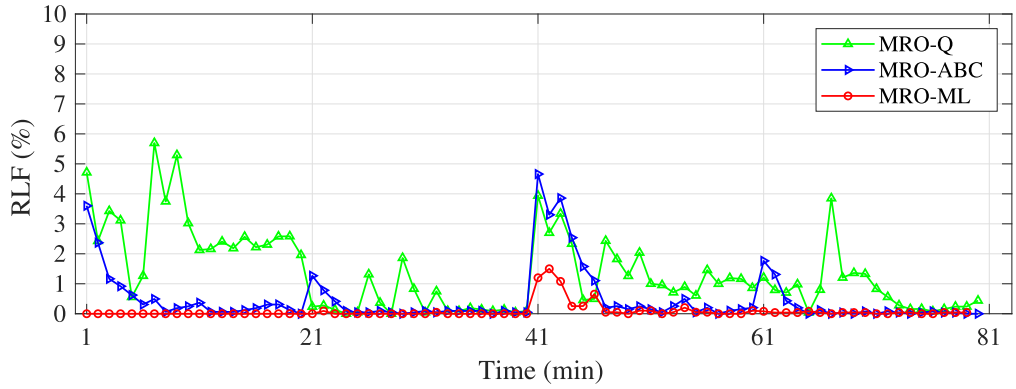
TABLE 2. Satisfaction rates of algorithms (%) for a speed of 5 km/h.

	9 cells	11 cells	10 cells	12 cells	Average	Improvement (%)
MRO-Q	0	80	5	5	22.5	100
MRO-ABC	75	80	60	75	72.5	322.2
MRO-ML	100	100	85	90	93.75	416.7

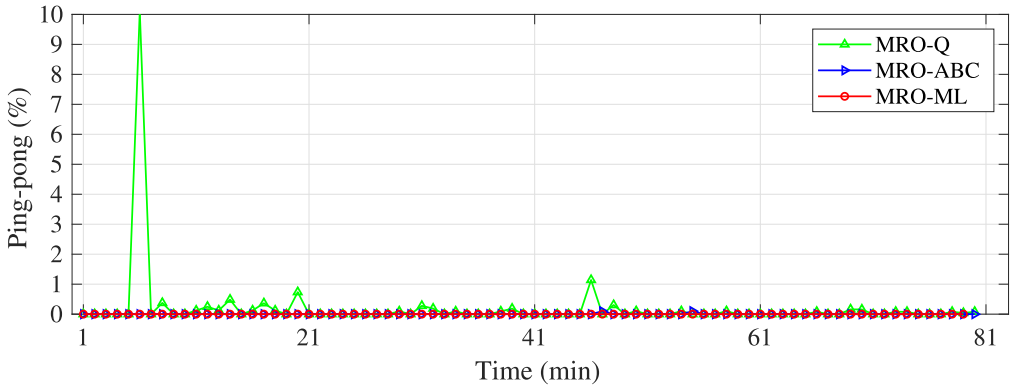
TABLE 3. Adaption times (in minutes) of algorithms for a speed of 30 km/h.

	9 cells	11 cells	10 cells	12 cells	Average	Reduction (%)
MRO-Q	4	0	4	1	2.25	100
MRO-ABC	3	1	6	2	3	133.33
MRO-ML	0	0	3	0	0.75	33.33

The simulation results for RLFs and ping-pongs are depicted in Fig. 9. We observed that handover performance fluctuated more than in the Manhattan environment due to the randomness in the RWP model. However, MRO-ML still improved handover performance more than the others, because it estimates the optimal handover parameters at



(a) RLF.



(b) Ping-pong.

FIGURE 8. Handover performance for a speed of 30 km/h.

TABLE 4. Satisfaction rates of algorithms (%) for a speed of 30 km/h.

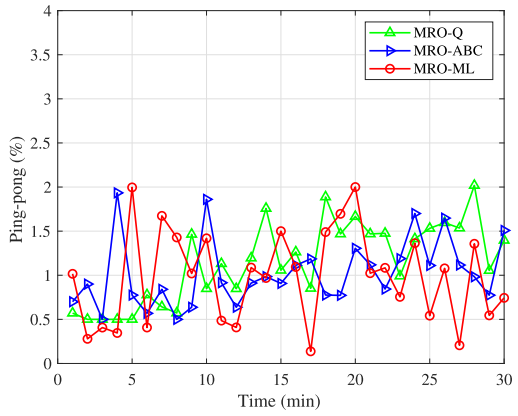
	9 cells	11 cells	10 cells	12 cells	Average	Improvement (%)
MRO-Q	5	90	30	80	51.25	100
MRO-ABC	85	95	70	90	85	165.9
MRO-ML	100	100	85	100	96.25	187.8

the very beginning, and adapts to the mobile environment with the estimated optimal parameters. MRO-ABC achieved the second-best performance because it fine-tunes the handover parameters according to undesirable handover classifications. MRO-ABC performance was close to MRO-ML in the long run, but in the beginning, MRO-ML was better because it estimates the optimal parameters before the topology changes. MRO-ML had better performance than MRO-Q, since MRO-ML utilizes prior knowledge and fine-tunes the parameters.

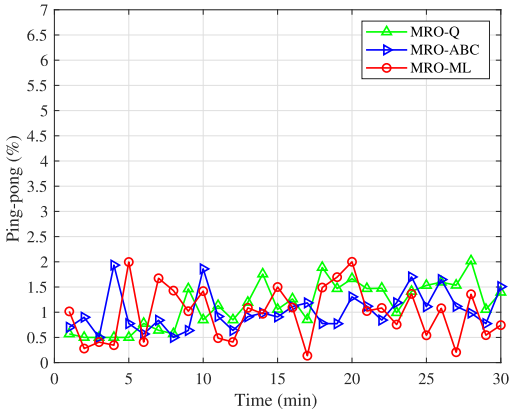
D. IMPACT OF CIO RESOLUTION ON HANDOVER PERFORMANCE

Handover performance is affected by the resolution of CIO because the optimal handover offset is a real value instead of an integer, as standardized by the 3GPP [23]. Fig. 10 shows the way that MRO-ML (specifically Algorithm 3) estimates the optimal handover parameters for two neighboring cells, called cell 1 and cell 2. The inter-site distance was 30 m. TTT_1, Δ_{21}, and TTT_2 were initialized at 256 ms, -2 dB, and 480 ms, respectively. Other settings, such as transmit power and antenna gain, were identical for both cells. The estimation of $\hat{\Delta}_{12}^{\mathrm{opt}}$ and $\hat{\Delta}_{21}^{\mathrm{opt}}$ took nine iterations to converge. The results show that $\hat{\Delta}_{12}^{\mathrm{opt}}$ and $\hat{\Delta}_{21}^{\mathrm{opt}}$ (in dB) were approximately 0.5 and -1, respectively. This result inspired us to evaluate handover performance for MRO-ML with a finer resolution for CIO.

To verify the impact of resolution on handover optimization, we investigated MRO-ML under the wireless scenario in Fig. 6(d) with a CIO resolution of 0.5 dB. The results are compared to those from a resolution of 1 dB, as depicted in Fig. 9. Fig. 11 shows that the resolution of 0.5 dB improved handover performance more than the resolution of 1 dB, and the outcome of the earlier resolution fluctuated less. This is because the adaptation algorithm provides finer tuning of the CIO around the optimal values, thus leading to better results.

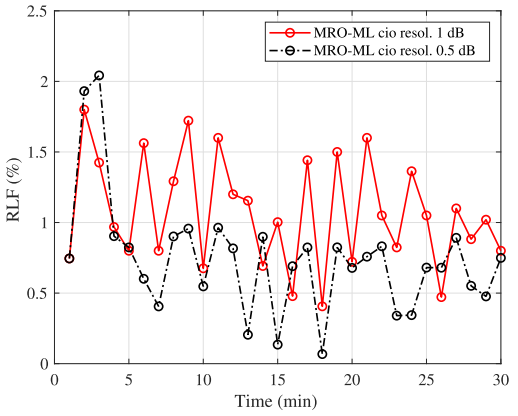


(a) RLF.

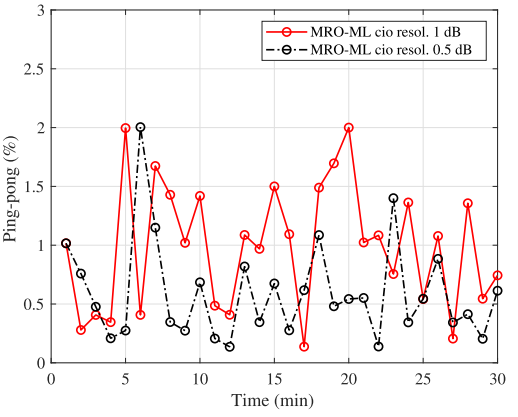


(b) Ping-pong.

FIGURE 9. Handover performance under a random mobility model.



(a) RLF.



(b) Ping-pong.

FIGURE 11. Handover performance with different CIO resolutions.

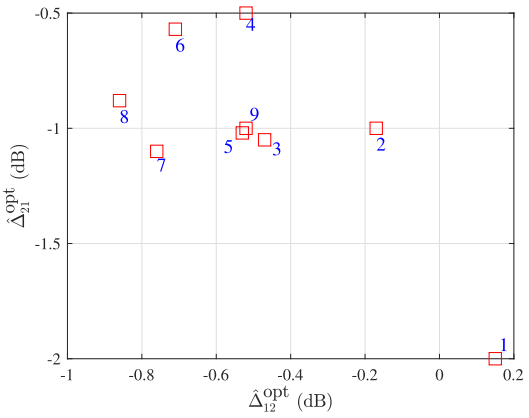


FIGURE 10. Examples of optimal handover parameter estimation. The number indicates the optimization step.

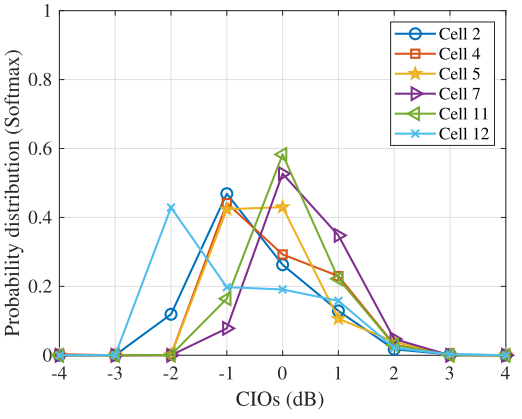
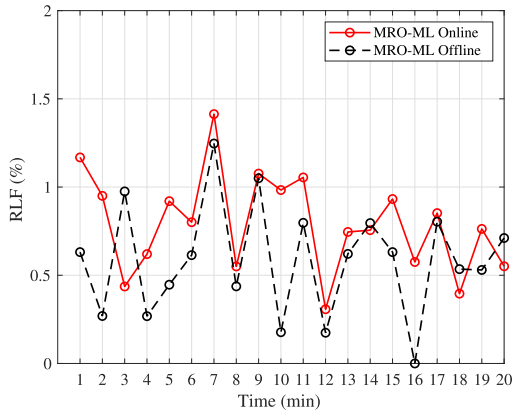


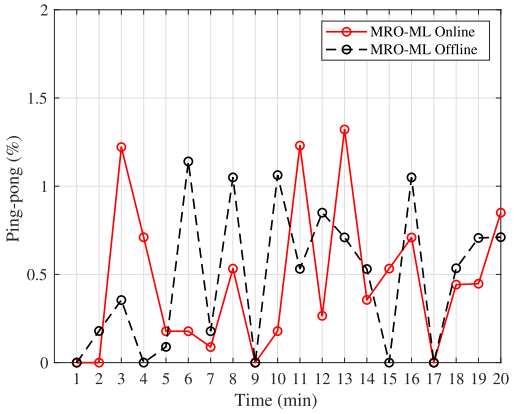
FIGURE 12. Prior knowledge for cell 1 with a softmax probability of $\tau = 1$.

E. OFFLINE LEARNING

MRO-ML utilizes prior knowledge to optimize handover parameters, thus the more prior knowledge the algorithm is provided, the better the performance it can achieve. The offline learning strategy is applied to create more knowledge for online learning processes. Thus, performance with the offline learning is an upper bound of that with the online learning only. To obtain more prior knowledge for MRO-ML, we kept simulating the environment of 10 cells in Fig. 6(c)for five hours and stored the knowledge of the dSONs in the database of the cSON. The cSON transferred the prior knowledge to the dSON at the beginning of the considered topology. We depict the belief distribution of the optimal CIOs in the dSON of cell 1 for its neighboring cells in Fig. 12. The



(a) RLF.



(b) Ping-pong.

FIGURE 13. Comparison between online and offline MRO-ML.

figure shows that the belief distributions of the optimal CIOs are various, cell-to-cell, due to the cell locations.

We compared the performance of MRO-ML in Fig. 6(c) referred to as MRO-ML Online, and against MRO-ML with the transferred five-hour knowledge, called MRO-ML Offline. The results are depicted in Fig. 13 showing that MRO-ML Online and MRO-ML Offline performances were close. Particularly, MRO-ML Online and MRO-ML Offline achieved satisfaction rates at 85% (13/20) and 90% (18/20), respectively. Since MRO-ML estimates the optimal handover parameters in the very beginning of a specific topology, and fine-tunes the parameters to adapt to user mobility with the transferred knowledge, the online algorithm is applicable to real-time handover optimization.

VI. CONCLUSION

In this paper, we proposed a machine learning–based MRO algorithm, MRO-ML, to minimize undesirable handovers (handover failures and ping-pongs) under a dynamic network topology as well as user mobility. To that end, MRO-ML has two steps: first, to adapt to the changing topology, at a cSON, a TL-based algorithm estimates the optimal handover parameters for the cells in the new topology, and transfers prior knowledge to dSONs. Then, at dSONs, an RL-based adaptation algorithm utilizes the prior knowledge and fine-tunes the handover parameters around the estimated optimal values to adapt to dynamic user mobility. We evaluated the proposed algorithms under a dynamic topology network with dynamic cell switching on/off scenarios and different mobility models. With the transfer learning–based algorithm, MRO-ML adapts to dynamic small-cell topologies in a short time. Simulation showed that the adaptation time of MRO-ML was only 4.17% of the baseline reference algorithm. Furthermore, by utilizing prior knowledge, MRO-ML adapted to user mobility better with a stable handover performance, and satisfied the performance target at such a significantly improved rate–416.7%–compared to the baseline algorithm. Finally, the online MRO-ML performed so closely to the offline one that the difference was negligible. Therefore, the online MRO-ML is applicable to real-time handover optimization for dynamic small-cell networks.

REFERENCES

[1] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. K. Soong, and J. C. Zhang, "What will 5G be?" *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1065–1082, Jun. 2014.

[2] I. F. Akyildiz, A. Kak, and S. Nie, "6G and beyond: The future of wireless communications systems," *IEEE Access*, vol. 8, pp. 133995–134030, 2020.

[3] A. J. Fehske, I. Viering, J. Voigt, C. Sartori, S. Redana, and G. P. Fettweis, "Small-cell self-organizing wireless networks," *Proc. IEEE*, vol. 102, no. 3, pp. 334–350, Mar. 2014.

[4] R. Webb, T. Wehmeier, and K. Dyer, "Small cells 2012 integration and optimisation," Alcatel Lucent, Boulogne-Billancourt, France, Tech. Rep. 2, 2012.

[5] *Self-Configuring and Self-Optimizing Network Use Cases and Solutions*, document TR 36.902, 3GPP, 2011.

[6] M. T. Nguyen, S. Kwon, and H. Kim, "Mobility robustness optimization for handover failure reduction in LTE small-cell networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 5, pp. 4672–4676, May 2018.

[7] M. M. Hasan, S. Kwon, and J.-H. Na, "Adaptive mobility load balancing algorithm for LTE small-cell networks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 4, pp. 2205–2217, Apr. 2018.

[8] T. S. Rappaport, Y. Xing, G. R. MacCartney, A. F. Molisch, E. Mellios, and J. Zhang, "Overview of millimeter wave communications for fifth-generation (5G) wireless networks—With a focus on propagation models," *IEEE Trans. Antennas Propag.*, vol. 65, no. 12, pp. 6213–6230, Dec. 2017.

[9] M.-T. Nguyen and S. Kwon, "Geometry-based analysis of optimal handover parameters for self-organizing networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 4, pp. 2670–2683, Apr. 2020.

[10] T. Jansen, I. Balan, J. Turk, I. Moerman, and T. Kurner, "Handover parameter optimization in LTE self-organizing networks," in *Proc. IEEE 72nd Veh. Technol. Conf. Fall*, Sep. 2010, pp. 1–5.

[11] K. Kitagawa, T. Komine, T. Yamamoto, and S. Konishi, "A handover optimization algorithm with mobility robustness for LTE systems," in *Proc. IEEE 22nd Int. Symp. Pers., Indoor Mobile Radio Commun.*, Sep. 2011, pp. 1647–1651.

[12] 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 J. Wireless Commun. Netw.*, vol. 2011, no. 1, p. 98, Sep. 2011.

[13] P. Munoz, R. Barco, and I. de la Bandera, "On the potential of handover parameter optimization for self-organizing networks," *IEEE Trans. Veh. Technol.*, vol. 62, no. 5, pp. 1895–1905, Jun. 2013.

[14] K. T. Dinh and S. Kuklinski, "Joint implementation of several LTE-SON functions," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2013, pp. 953–957.

[15] A. Klein, N. P. Kuruvatti, J. Schneider, and H. D. Schotten, "Fuzzy Q-learning for mobility robustness optimization in wireless networks," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2013, pp. 76–81.

[16] S. S. Mwanje, L. C. Schmelz, and A. Mitschele-Thiel, "Cognitive cellular networks: A Q-learning framework for self-organizing networks," *IEEE Trans. Netw. Service Manage.*, vol. 13, no. 1, pp. 85–98, Mar. 2016.
[17] P. Muñoz, R. Barco, J. M. Ruiz-Aviles, I. de la Bandera, and A. Aguilar, "Fuzzy rule-based reinforcement learning for load balancing techniques in enterprise LTE femtocells," *IEEE Trans. Veh. Technol.*, vol. 62, no. 5, pp. 1962–1973, Jun. 2013.
[18] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*. Cambridge, MA, USA: MIT Press, 2018.
[19] H. Sanneck, Y. Bouwen, and E. Troch, "Dynamic radio configuration of self-organizing base stations," in *Proc. 7th Int. Symp. Wireless Commun. Syst.*, Sep. 2010, pp. 716–720.
[20] F. Calabrese, M. Colonna, P. Lovisolo, D. Parata, and C. Ratti, "Real-time urban monitoring using cell phones: A case study in Rome," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 1, pp. 141–151, Mar. 2011.
[21] *LTE; E-UTRAN; Overall Description; Stage 2*, document TS 36.300, 3GPP, 2020.
[22] *Physical Layer; Measurements*, document TS 36.214, 3GPP, 2018.
[23] *Universal Mobile Telecommunications System (UMTS), Radio Resource Control (RRC) Protocol Specification*, TS 36.331, 3GPP, 2020.
[24] M. M. Hasan, S. Kwon, and S. Oh, "Frequent-handover mitigation in ultra-dense heterogeneous networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 1035–1040, Jan. 2019.
[25] L. Torrey and J. Shavlik. *Transfer Learning*. Accessed: Aug. 2009. [Online]. Available: http://pages.cs.wisc.edu/~shavlik/abstracts/torrey.handbook09.abstract.html
[26] D. Lopez-Perez, I. Guvenc, and X. Chu, "Mobility management challenges in 3GPP heterogeneous networks," *IEEE Commun. Mag.*, vol. 50, no. 12, pp. 70–78, Dec. 2012.
[27] P. Fazio, M. Tropea, F. D. Rango, and M. Voznak, "Pattern prediction and passive bandwidth management for hand-over optimization in QoS cellular networks with vehicular mobility," *IEEE Trans. Mobile Comput.*, vol. 15, no. 11, pp. 2809–2824, Nov. 2016.
[28] H. Elshaer, F. Boccardi, M. Dohler, and R. Irmer, "Downlink and uplink decoupling: A disruptive architectural design for 5G networks," in *Proc. IEEE Global Commun. Conf.*, Dec. 2014, pp. 1798–1803.
[29] L. Saker, S. E. Elayoubi, R. Combes, and T. Chahed, "Optimal control of wake up mechanisms of femtocells in heterogeneous networks," *IEEE J. Sel. Areas Commun.*, vol. 30, no. 3, pp. 664–672, Apr. 2012.
[30] E. Oh, K. Son, and B. Krishnamachari, "Dynamic base station switching-On/Off strategies for green cellular networks," *IEEE Trans. Wireless Commun.*, vol. 12, no. 5, pp. 2126–2136, May 2013.
[31] S. Samarakoon, M. Bennis, W. Saad, and M. Latva-aho, "Dynamic clustering and on/off strategies for wireless small cell networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 3, pp. 2164–2178, Mar. 2016.
[32] *Further Advancements for E-UTRA Physical Layer Aspects*, document TS 36.814, 3GPP, 2017.
[33] S. Sun, T. A. Thomas, T. S. Rappaport, H. Nguyen, I. Z. Kovacs, and I. Rodriguez, "Path loss, shadow fading, and line-of-sight probability models for 5G urban macro-cellular scenarios," in *Proc. IEEE GLOBECOM Workshops*, Dec. 2015, pp. 1–7.
[34] J. Zhu, Y. Song, D. Jiang, and H. Song, "A new deep-Q-learning-based transmission scheduling mechanism for the cognitive Internet of Things," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2375–2385, Aug. 2018.

MINH-THANG NGUYEN received the B.E. degree in electrical engineering from the Ho Chi Minh City University of Technology, Vietnam, in 2013, and the Ph.D. degree in electrical engineering from the University of Ulsan, South Korea, in 2021. Since 2021, he has been a Postdoctoral Fellow with the École de technologie supérieure (ETS), Montreal, QC, Canada. His research interests include self-organizing networks (SONs), network routing, large-scale optimization, graph theory, and machine learning in wireless communication networks.

SUNGOH KWON (Senior Member, IEEE) received the B.S. and M.S. degrees in electrical engineering from KAIST, Daejeon, South Korea, in 1994 and 1996, respectively, and the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, USA, in 2007. From 1996 to 2001, he was a Research Staff Member with Shinsegi Telecomm Inc., Seoul, South Korea. From 2007 to 2010, he was a Principal Engineer with Samsung Electronics Company Ltd., South Korea, where he developed LTE schedulers. Since 2010, he has been with the School of Electrical Engineering, University of Ulsan, South Korea, where he is currently a Professor. His research interest includes wireless communication networks.