Data-Driven Configuration of Interference Coordination Parameters in HetNets

Jose A. Ayala-Romero, Juan J. Alcaraz, and Javier Vales-Alonso

***Abstract*—The heterogeneous networks (HetNets) performance is highly dependent on the interference coordination among cells since both macrocells and small cells share the same spectrum. This paper focuses on the configuration of eICIC parameters: almost blank subframe ratio and cell range expansion bias defined for LTE-A. We propose an online learning mechanism based on a big data-driven framework and multiarmed bandit algorithms that retrieves data from the network to learn efficient configurations of eICIC parameters without requiring any previous knowledge about the network (e.g., traffic load, topology, scheduling algorithm). Our numerical results show that our approach attains a significant improvement with respect to the state of the art of online learning algorithms in networks under stationary and variable conditions (e.g., number of small cells, traffic load).**

***Index Terms*—Interference coordination, heterogeneous networks, data-driven, model-free, online learning.**

I. Introduction

THE amount of data produced in cellular network by mobile phones and other smart devices is increasing exponentially [1]. This creates an opportunity for network operators to extract relevant information using big data techniques and to apply algorithms that exploit this information for real-time decision making in a wide range of applications. Specifically, we address the interference coordination management problem in heterogeneous networks (HetNets) using a data-driven approach. Thus, we propose to use the data retrieved from the network in order to learn efficient configurations of the interference coordination parameters and, attending to the dynamic nature of the network, perform a real-time adjustment of the interference management parameters according to variations in network conditions.

We consider a typical HetNet scenario with small cells overlapping the macro cell coverage area. Since all cells share the same spectrum, the interference coordination management is essential, specially where the small cell deployment density increases. We apply our proposal to the enhanced Inter Cell Interference Coordination (eICIC) technique defined by 3rd Generation Partnership Project (3GPP) for Long Term Evolution Advanced (LTE-A) Network. The eICIC parameters are Cell Range Expansion (CRE) bias and Almost Blank Subframe (ABS) ratio.

The eICIC technique performs a resource allocation among macro cells (macro eNB in 3GPP terminology) and pico cells (pico eNBs) in the time domain. The optimal configuration of eICIC parameters depends on the dynamically varying network conditions (traffic load, user demands an positions, number of active pico eNBs, etc) [2]. However, most previous works consider a static situation of the network [3]–[11]. To overcome this limitation, we formulate the problem as an online learning problem where the proposed mechanism configures the eICIC parameters in real-time, during the operation of the network.

Another limitation of previous works is that their approaches rely on mathematical models of the network [3], [6], [12]–[14]. In general, even the most complex network models may not comprise all the relevant aspects of a real operating network. In contrast to previous works, our approach is data-driven instead of model-driven. Thus, it operates without making any assumption about the network.

As a part of our proposal, we propose a novel Multi-armed bandit (MAB) algorithm for a balanced exploration/exploitation decision making in eICIC parameter configuration. The objective of MAB algorithms is to maximize an expected reward over time, given a set of actions whose individual rewards are initially unknown and possibly random (as in our case). Our proposal is capable of adapting to variable network conditions, continuously adjusting the eICIC parameters to configurations that are efficient in the current conditions.

Finally, we evaluate our proposal against two families of benchmark online algorithms: stochastic gradient ascent (SGA) algorithms and MAB algorithms. We consider two settings: stationary and variable conditions in the network.

The rest of the paper is organized as follows. In Section II the related work and contribution summary are given. In Section III we describe the proposed big data framework and detail the eICIC configuration parameters. The network model and the formulation of the problem are given in Section IV. In Section V we describe the proposed mechanism. The benchmark algorithms are described in Section VI. Finally, the numerical results are given in Section VII and the conclusions are summarized in Section VIII.

II. Related Work and Contribution

A. Related Work

With recent advances in big data analytics, data-based cellular network optimization has attracted the attention of the scientific community [1], [2], [19]. The authors in [1] propose a frame-

Manuscript received May 16, 2017; revised October 23, 2017 and February 16, 2018; accepted April 1, 2018. Date of publication April 12, 2018; date of current version June 18, 2018. This work was supported by a Project Grant AEI/FEDER TEC2016-76465-C2-1-R (AIM). The work of J. A. Ayala-Romero was supported by a personal Grant FPU14/03701. *(Corresponding author: J. A. Ayala-Romero.)*

The authors are with the Department of Information and Communications Technologies, Technical University of Cartagena, Cartagena 30202, Spain (e-mail: josea.ayala@upct.es; juan.alcaraz@upct.es; javier.vales@upct.es).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2018.2825606

0018-9545 © 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

TABLE I
COMPARISON OF RELATED WORKS

	[4]	[15], [16]	[13]	[8]	[9], [10]	[3], [12]	[11]	[17]	[14]	[18]	DDeC
Controlled Parameters	CRE	ABS	ABS	ABS	CRE	both	both	both	both	both	**both**
Online Operation	Yes	Yes	Yes	No	No	No	No	Yes	Yes	Yes	**Yes**
Coordinated Solution	No	Yes	Yes	No	No	Yes	No	No	No	No	**Yes**
Black box approach	No	No	No	No	No	No	No	No	No	Yes	**Yes**
Approach	L	H	O	L	H	O	O	H	O	L	**L**

(O): Optimization, (H): heuristic, (L): learning.

work for big data-driven mobile network optimization and also suggest that it can be used for interference management.

Some works like [4], [8], [18] address this problem using learning algorithms. The authors in [8] use a reinforcement learning (RL) approach, specifically a fuzzy Q-learning algorithm, to learn the optimal ABS ratio considering a fixed CRE bias. In [4], Q-learning algorithms are also proposed for ABS ratio and CRE bias learning. However, in [4], [8] the learning algorithms are applied to static network situation, i.e., constant traffic intensity and UEs with fixed locations. In contrast, in [18] and our work, the variable nature of the network is considered. The main issue with RL approaches is that they require offline training before operating in the network [18]. This training is usually carried out in a network simulator introducing the additional requirement of reproducing the network in a simulator model. If the simulator model is not accurate the RL policies may be inefficient in practice. In contrast, our proposal operates without offline training or any previous knowledge about the network.

Most related works address a static network situation [3]–[11], not considering explicitly the real-time adaption to a dynamic network conditions. Although some works consider the dynamic nature of the network [13], [15]–[17], they differ from ours in the following aspects: First, the authors of [15]–[17] propose a heuristic approach which is scenario dependent. For example, in [15] they assume the use of Proportional Fair (PF) scheduling in the HetNet using the PF metric as a basic element of the heuristic. Second, the approaches proposed in [13], [15], [16] only configure the ABS ratio ignoring that both ABS ratio and CRE bias have a joint impact in the performance metric [17]. In contrast, our proposal is able to jointly configure these two parameters regardless of the particular scenario, only using data retrieved from the network.

Let us discuss the difference between model-driven schemes and our data-driven, model-free approach. Model-driven approaches (e.g., [3], [6], [12]–[14]) need to collect key parameters from the network (such as the location of all UEs or channel gains) to feed the model which necessarily contains assumptions and/or simplifications of the real network making it suitable for an optimization algorithm. However, the optimal configuration for the model might not necessarily be the optimal configuration of the real network due to the simplification and/or assumptions in the network model. In contrast, a data-driven approach relies on direct observations of the network performance obtained from the real network (without assumption or simplifications). This allows the decision algorithm to explore configurations aiming at finding the most efficient ones. For example, the scheme in [12] aims at optimizing a weighted sum of the logarithms of the user throughputs. The algorithm parameters are the interference graph of the network and the data rates that each user u would obtain if attached to the nearest macro station (r_u^m) of to the nearest pico station (r_u^p). The associated signaling overhead of this scheme is higher than ours, but the main drawback is the implicit assumption of considering these data rates, (r_u^m , r_u^p) independent of the eICIC configuration. Note that this assumption is a simplification/approximation, since the rate at each UE is determined by the SINR, which, in turn, depends on the eICIC configuration, and therefore r_u^m and r_u^p cannot be accurately known a priori. This assumption, among others, is useful in setting up a convex problem, but may imply that the optimal configuration for this model does not match the optimal configuration of the real system, especially if the desired performance metric is not captured by the objective function. In the numerical results of that paper it is shown that the 5th percentile throughput of some fixed configurations is practically equal to the proposed algorithm, which suggests that a search over the configuration set (as the one proposed in our paper) would eventually find a configuration at least as good as the one found by this algorithm.

TABLE II
SIMULATION MODEL COMPARISON IN RELATED WORKS

UE locations	Static [3], [4], [7]–[11]	Dynamic [12]–[15], [17], [18], **Ours**
Traffic Model	Full Buffer [3], [4], [7]–[12]	FTP [13]–[15], [17], [18], **Ours**

Some works like [20], [21] address the problem using stochastic geometry. This approach models the location of macro eNBs as a homogeneous poisson point process (PPP). It implies that the location of each macro eNB does not depend on others. In a real network the deployment of each macro eNB is generally planed and depends on the position of the surrounding macro eNB, the terrain, attenuation profile, etc. The authors in [20] also assume Round Robin as a scheduling policy, not appropriate for HetNets. In contrast, our mechanism relies on real data from the operating network avoiding the simplifications introduced by mathematical models.

Finally, our proposal also differs from previous works in terms of computation and signaling overhead, not imposing a noticeable burden in any of them. For example, the algorithm in [12] requires information about network topology and also computes at each iteration the following data: the data-rate achievable by each UE if it were connected to best macro, the data-rate achievable by each UE if it were connected to best pico in ABS and non-ABS subframe and the mean throughput of each UE. In a similar way, [3] and [22] require the interference graph of the network to operate.

We provide a detailed comparison between our work and related works in Tables I and II.

B. Contribution

The first contribution of this paper is to present a specific application of the general idea of data-driven network management, described in [1], [23], as a proof-of-concept. We show how this approach can be applied to the dynamic configuration

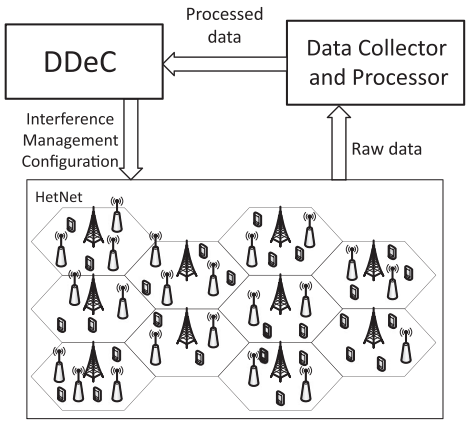

Fig. 1. Illustration of the big data-driven framework. *Data Collector and Processor* processes the raw data from the network. *DDeC* uses the processed data to learn efficient configurations of the interference coordination parameters to be used in the HetNet. The new configurations provided by *DDeC* affect the raw data used as feedback, an so forth.

of eICIC parameters in HetNets in our proposed mechanism: the Data-Driven eICIC Configurator (*DDeC*).

Our proposal finds efficient configurations of the eICIC parameters using only data retrieved from the network. This data-driven nature provides two main benefits to DDeC with respect to previous approaches: first, *DDeC* can operate without any previous knowledge of the underlying network (e.g. topology, scheduling algorithms); second, *DDeC* can optimize any measurable performance metric, even those that are analytically intractable such as the 5th percentile throughput.

DDeC is based on a novel online learning algorithm of the Multi-Armed Bandit (MAB) type. Compared to previous MAB algorithms, our proposal comprises the following novelties: 1) it uses a local exploration strategy which notably increases the algorithm's efficiency, 2) it uses only a reduced set of actions on each learning/decision stage, 3) it can operate under non-stationary network conditions by means of a novel module that detects if the optimal action changes. This addresses a major challenge for online learning algorithms. In consequence, our numerical experiments show that *DDeC* outperforms previous proposals. Besides, it does not introduce additional computational complexity compared to previous approaches.

III. Big Data Framework and Application Scenario

A. Big Data Framework

Our proposal is built upon the big data-driven framework proposed in [1] and depicted in Fig. 1. This framework integrates big data analytics and network optimization with the objective of improving the user quality of service. It can be applied to resource management in HetNets such as network planning optimization, energy saving, etc. However, our work is, to the best of our knowledge, the first to apply this framework to interference coordination in HetNets.

The framework, adapted to our eICIC functionality, consists on the following data flow: *Data Collector and Processor (DCP)* collects raw data from the network and processes it to obtain suitable data to feed the *DDeC* (Data-Driven eICIC Configurator). According to this processed data, *DDeC* selects the eICIC configuration for the operating network. This new eICIC configuration affects the raw data collected henceforth which is used as a feedback. The successive iterations of this process allows *DDeC* to find efficient configurations.

Regarding the data collection, the raw data is computed at eNBs and sent to the *DCP* to be processed. Thus, DCP obtains a network statistic or performance sample from the whole network.

In our scenario, the time varying nature of the network is a relevant aspect, since it affects, in general, to the efficiency of the eICIC configurations. Therefore, *DDeC* incorporates the capability of detecting statistical variations in the network conditions, in real time, to enable a fast adaptation of the interference management configuration.

B. Interference Management in LTE-A: eICIC

The eICIC is an interference management technique defined by 3GPP in Release 10 (LTE-A) [24] for HetNet environments. To prevent inter-cell interference, eICIC allows macro eNBs and pico eNBs to use radio resources in different time periods (subframes). The main features of eICIC are: *Cell Range Expansion (CRE)* and *Almost Blank Subframe (ABS)*.

Pico eNBs are intended to enhance the spatial frequency reuse of the network. However, since the transmission power of macro eNBs is higher than the transmission power of pico eNBs, a UE close to a pico eNB could associate with the macro eNB with high probability, losing the benefits provided by pico eNBs. It occurs because LTE networks implement association based on received signal reference power (RSRP), i.e., UEs associate with the highest reference signal that every cell broadcast. Therefore, it leads to the underutilization of the pico eNBs and the overload of the macro eNBs. To cope with that problem, CRE allows the UE to associate to a pico eNB even when its RSRP is lower than the RSRP from the macro eNB. For that purpose, the UEs add the CRE bias to pico RSRPs extending the pico eNBs footprint. Thus, the UE associates with the eNB with maximum (corrected) RSRP. Nevertheless, the UEs located at pico eNB extended region experience a poor Signal to Interference and Noise Ratio (SINR) due to the high power received from the macro eNB.

To alleviate this problem ABS is introduced. It allows the macro eNBs to mute all data symbols in certain subframes referred to as Almost Blank Subframes. In these protected subframes the UEs with poor channel conditions can boost its SINR due to the absence of macro interference in data symbols. Although a LTE frame comprises 10 subframes, the ABS patterns has a periodicity of 8 subframes. Thus, it is necessary to configure the ratio of protected subframes over conventional subframes $(0/8, \ldots, 8/8)$. We assume ABS ratio mixtures are also possible (e.g., $2/8$, $3/8$ with probability 0.5 each) obtaining another ABS ratio value (e.g., $2.5/8$). As a result, the ABS ratio can take values in $[0, 1]$.

As in [20], [25], [26], we consider synchronized muting and common CRE bias value.

IV. Problem Formulation

A. Network Performance Metric

Consider a network operator who wants to optimize a certain performance metric F which depends on the interference coordination parameters of the network. In our scenario, these parameters are the ABS ratio (γ) and CRE bias (ϕ). Note that,

for any configuration $x = (\gamma, \phi)^T$ and under stationary network conditions (i.e., traffic intensity, number of active pico eNBs), the value of $F(x)$ is a random variable due to the stochastic nature of the network (i.e., variable UE positions, stochastic shadow fading).

The performance metric F can be related, for example, to the physical layer (signal to interference and noise ratio, SINR) or to upper layers (data throughput). Since a mathematical characterization of $F(x)$ for a real operating network without simplifications is unavailable in general, our approach aims at finding efficient configuration based on performance observation from the network. Thus, *DCP* (Fig. 1) is responsible for processing the raw data collected from the network in order to obtain an observation of the random variable $F(x)$ for the configuration x. We consider *DCP* as a black box and its operation is out of the scope of this paper.

Although *DDeC* operates independently of the selected performance metric F, in our case, we consider the 5th percentile throughput over the downlink to be the performance metric. Specifically, $F(x)$ is the MLE estimator (and thus a random variable) of the 5th percentile throughput using the configuration x. This metric is proposed by the 3GPP [27] to evaluate the performance of LTE networks using non-full buffer traffic model. In addition, multiple related works [12], [15], [25] use this metric as well. We define a throughput sample as the quotient between the size of a downloaded file and the time needed to download it [27]. The 5th percentile throughput is defined as the value below which 5 percent of the throughput samples may be found. Thus, given a set of throughput samples retrieved under configuration x, we can obtain a sample of the 5th percentile throughput which is a random variable denoted by $F(x)$. The set of values $E[F(x)]$ for all x is referred to as the *system response*. Fig. 3 shows an example of systems response for especific network conditions. Note that when the network conditions varies, the system response generally changes.

Regarding the big data-driven framework depicted in Fig. 1, in our case, the *raw data* retrieved by *DCP* are throughput samples obtained by UEs. *Processed data* refers to 5th percentile throughput samples computed by *DCP* from throughput samples.

B. Online Problem Formulation

In this section, we formulate the interference management problem as an online learning problem. In this type of problem, in contrast to offline approaches, the algorithm has to select configurations and sample their performances from the network as feedback. Since changing the configuration actually affects the network performance, we face the challenge of selecting configurations allowing us to infer the system response, while avoiding poor performing ones. In addition, in our case, we must face the non-stationarity of the network conditions, which causes variations in the system response.

The performance samples are obtained in countable time stages $k = 1, 2, \ldots$. At each stage k a configuration x_k is selected and a performance sample $y_k = F(x_k)$ is obtained. The next configuration x_{k+1} is selected according to the knowledge obtained by past samples. All configurations $x_0 \ldots x_k$ should be selected from the set $\mathcal{P} = \left\{x = (\gamma, \phi)^T : 0 \leq \gamma \leq 1, 0 \leq \phi \leq \phi_{\max}\right\}$.

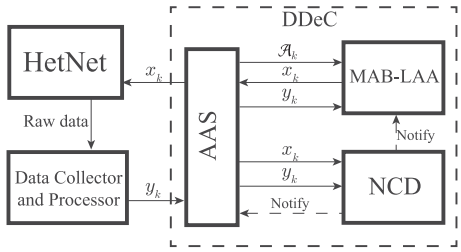

Fig. 2. Detailed system scheme with *DDeC* submodules.

The *pseudo-regret* up to stage k of an algorithm is defined by:

$$kE[F(x^*)] - \sum_{i=1}^{k} E[F(x_i)] \tag{1}$$

where

$$x^* = \arg\max_{x \in \mathcal{P}} E\left[F(x)\right]. \tag{2}$$

The proposed mechanisms are aimed at *minimizing* the pseudo-regret (hereinafter referred to as regret) which is the most used metric in online settings [28], [29]. Although the problem is formulated over the continuous set $\mathcal{P}$ of configurations, our proposal addresses it only selecting actions from a discrete set $\mathcal{A}$ as we detail in the next section.

Note that we have defined two performance metrics: first, the *5th percentile throughput* (F) which is the performance metric that our proposal uses and is obtained directly from the network data. Second, the *regret*, which shows how good is an algorithm in an online setting. The regret is computed from samples of 5th percentile throughput. Specifically, it captures the accumulated loss of not selecting the optimal configuration $F(x^*)$ at each sample, and therefore shows how the algorithm behaves with respect of the optimal performance. It is worth to remark that, since the optimal x^* is unknown in advance the regret cannot be computed in a real network. However, it can be estimated by exhaustive search in numerical experiments. We detail in Section VII-B how we address regret computation.

Note that the optimal configuration x^* can vary over time making it more challenging for the algorithms to select efficient configurations. Nevertheless, since low regret implies that the loss associated to a suboptimal configuration is also low, the aim of an online algorithm is to minimize this metric over time. For that purpose, there is a broad literature about SGA and MAB algorithms addressing regret minimization in online learning problems [28], [30]–[35]. The following section presents a novel mechanism showing a significant improvement with respect to SGA and MAB state of the art algorithms.

V. Mechanism Description

This section describes *DDeC* depicted in Fig. 1 and further detailed in Fig. 2. This module sends at the beginning of each stage k the configuration x_k to be used in the network. Then, the network operates with this configuration for the whole duration of the stage k sending the raw data (in our case throughput samples) to *DCP* which processes it in order to obtain a performance sample $y_k = F(x_k)$. This performance sample is sent to *DDeC* which uses it to select the next configuration x_{k+1}.

Fig. 3 shows the expectation of $F(x)$ for all $x \in \mathcal{A}$ (system response). We can verify in this figure that the expected

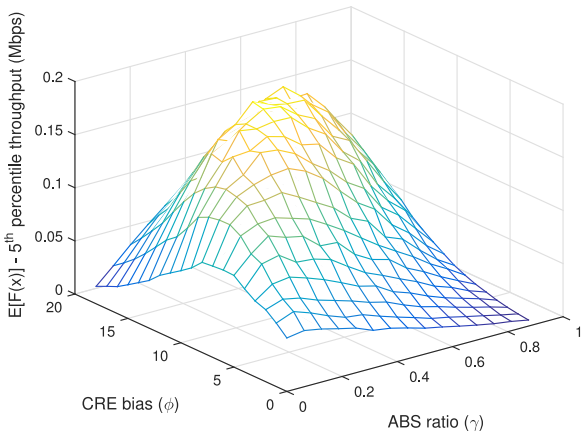

Fig. 3. Expected value of $F(x)$ for all $x \in \mathcal{A}$ in a HetNet scenario with 4 pico eNBs per sector and 75 Mbps of offered traffic per sector.

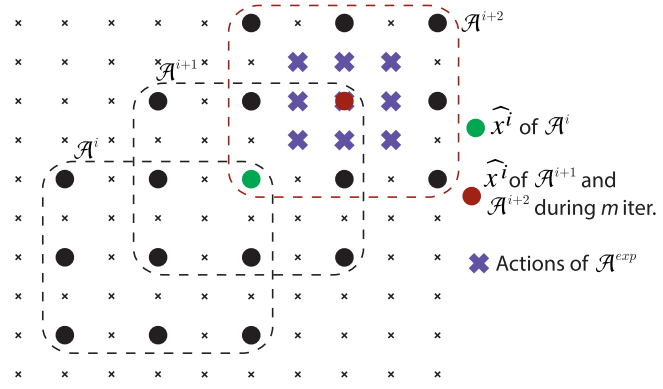

Fig. 4. Example of algorithm evolution from *AAS* submodule point of view. In the search period, the set $\mathcal{A}^{i+1}$ is created from the best local action of iteration i (highlighted in green). Idem for $\mathcal{A}^{i+2}$, created from the best local action of iteration $i+1$ (highlighted in red). The red action is the best local action during the next m iterations which indicates the beginning of the exploitation period. The actions remarked with crosses ($\mathcal{A}^{\text{exp}}$) are the available action during the whole exploitation period.

performance of every action is correlated with the expected performance of the actions of its surrounding, which means that the region of the optimum configuration can be reached from any other configuration following a sequence of actions with strictly increasing performance. This characteristic was verified in a wide range of scenarios with different configurations (traffic load, number of pico eNBs, etc.) by means of extensive simulation and is in line with the results obtained in [20]. Our proposal exploits this structure in the system response making local explorations and selecting the best local actions in order to determine the next set of actions for local exploration. Thus, we can approach the optimum action quickly and without exploring all the available actions in $\mathcal{A}$.

DDeC has two objectives: first, to find efficient actions by exploiting the above described structure of the system response, and second, to detect changes in the network conditions affecting this response. In order to accomplish these objectives, it is divided into three submodules: the Action Availability Selector (*AAS*), the Multi-Armed Bandit with limited action availability (*MAB-LAA*) and the Network Change Detector (*NCD*). These three submodules can be seen as three black boxes operating transparently from the others and only using the input/output data detailed in Fig. 2. The tasks of each submodule are summarized as follows:

- *AAS* generates at each stage a discrete set $\mathcal{A}_k$ of configurations x (also called actions) that can be selected by our mechanism.
- *MAB-LAA* selects which configuration x to use at each stage from the set of available actions $\mathcal{A}_k$ given by *AAS*.
- Finally, *NCD* tracks the values of the observations in order to notify to other submodules when it detects a variation on the network conditions.

A. Action Availability Selector (AAS)

The aim of *AAS* submodule is selecting the subset $\mathcal{A}_k \subset \mathcal{A}$ of actions that can be chosen by our proposal at each stage k, from the set $\mathcal{A}$ of all available actions. The operation of *AAS* comprises two periods: *search period* and *exploitation period*. The purpose of the search period is to rapidly approach the region of the optimal action x^* from any initial action x_0. The exploitation period performs a fine-tuned action exploration in the region of x^* in order to find actions with better performance.

The **search period** is divided in iterations. We define an iteration as a set of τ stages where the set of available actions does not change. Let $\mathcal{I}^i = \{i : k \leq i \leq k+\tau\}$ be the set of τ consecutive stages belonging to iteration i. For each iteration, *AAS* selects a set of available actions $\mathcal{A}^i \subset \mathcal{A}$ such that $\mathcal{A}_k = \mathcal{A}^i$ for all $k \in \mathcal{I}^i$. Fig. 4 depicts an example of $\mathcal{A}$ (all possible actions with x 2-dimensional) and the evolution of three consecutive iterations of search period. Note that, for each iteration i, all actions in the set $\mathcal{A}^i$ wrap around the so-called *central action* (x_c^i). Thus, at first iteration ($i=0$), we choose an initial *central action* x_c^0 and 8 additional actions wrapping around x_c^0 to define the set $\mathcal{A}^0$. The set $\mathcal{A}^i$ is selected by *AAS* which sends it to *MAB-LAA* at each stage k.

AAS stores in $\mathcal{Y}_x^i$ the number of times that each action $x \in \mathcal{A}^i$ has been selected during the iteration i. At the end of i-th iteration, *MAB-LAA* determines the *best local action*, given by

$$\widehat{x^i} = \arg\max_{x \in \mathcal{A}^i} \mathcal{Y}_x^i. \tag{3}$$

Then, $\widehat{x^i}$ is used as the *central action* for next iteration ($x_c^{i+1} = \widehat{x^i}$). The configuration of parameter τ will be discussed later in Section VII-F.

When the *best local action* of iteration i matches the current *central action* ($\widehat{x^i} = x_c^i$), the set $\mathcal{A}^{i+1}$ will contain the same elements of $\mathcal{A}^i$. The **exploitation period** starts when the set of available actions does not change during $m \geq 2$ consecutive iterations, that is, $x_c^i = x_c^{i-1} = \ldots = x_c^{i-m+1}$ (illustrated in Fig. 4 at iteration $i+2$ and in Algorithm 1 from line 16 onwards). Then, a new set of available actions $\mathcal{A}^{\text{exp}}$ is used, containing configurations closer to the last *best local action*. Note that $\mathcal{A}_k = \mathcal{A}^{\text{exp}}$ during the whole exploitation period.

The exploitation period benefit is twofold: (i) the set $\mathcal{A}^{\text{exp}}$ is generated with actions closer to x_c, which is the action with the highest estimated performance so far. This aims at exploring the optimal region in depth to find actions with higher performance than x_c. (ii) one of the main drawbacks of SGA algorithms is that they deviate from the optimal region after approaching that due to stochastic nature of $F(x)$. In contrast, in the exploitation period of DDeC the set of available actions $\mathcal{A}^{\text{exp}}$ remains fixed

Algorithm 1: AAS framework.

1: **Input parameters:** x_c^0, τ
2: **Initialize:** $k = 0$
3: **for** $i = 0, 1, 2, \ldots$ **do**
4: Generate the set $\mathcal{A}^i$ from x_c^i
5: Generate the set $\mathcal{I}^i = \{i : k \leq i \leq k + \tau\}$
6: **while** $k \in \mathcal{I}^i$ **do**
7: $\mathcal{A}_k = \mathcal{A}^i$
8: Send $\mathcal{A}_k$ to *MAB-LAA*
9: x_k is received and sent to the HetNet
10: y_k is received and sent to the *MAB-LAA*
11: Update $\mathcal{Y}_x^i$ with x_k and y_k
12: $k = k + 1$
13: **end while**
14: Compute $\widehat{x^i} = \arg\max_{x \in \mathcal{A}^i} \mathcal{Y}_x$
15: $x_c^{i+1} = \widehat{x^i}$
16: **if** $x_c^{i+1} = x_c^{i-1} = \ldots = x_c^{i-m+2}$ **then**
17: Generate the set $\mathcal{A}^{\text{exp}}$ from x_c^{i+1}
18: **while** Notify from *NCD* is not received **do**
19: $\mathcal{A}_k = \mathcal{A}^{\text{exp}}$
20: Send $\mathcal{A}_k$ to *MAB-LAA*
21: x_k is received and sent to the HetNet
22: y_k is received and sent to the *MAB-LAA*
23: Send x_k and y_k to *NCD*
24: $k = k + 1$
25: **end while**
26: **end if**
27: **end for**

until receiving a NCD notification, avoiding the aforementioned deviation. The operation of the *AAS* submodule is summarized in Algorithm 1.

B. Multi-Armed Bandit with Limited Action Availability (MAB-LAA)

This submodule selects the action x_k to use at each stage k from the set $\mathcal{A}_k$ which is provided by *AAS* submodule at each stage. Algorithm 2 summarizes the operation of *MAB-LAA* submodule. The input parameter $\bar{n}$ and how the actions are selected (line 5) are determined by the selected *MAB policy*.

Although any MAB policy can be used in the *MAB-LAA* framework, we use the Thompson sampling normal policy [30] (described in Section VI-B) because it obtains better performance compared to other MAB policies in our numerical evaluations.

C. Network Change Detector (NCD)

NCD submodule is responsible for detecting changes in the system response by using the samples of $F(x)$. This detection is only active in the exploitation period of *AAS*. When a change in the systems response is detected, the submodules *AAS* and *MAB-LAA* are notified by *NCD* (Fig. 2) and, as a consequence, *AAS* switches to search period and *MAB-LAA* resets its knowledge about the random variable $F(x)$ for all x. Then, the *central action* of the first iteration of *AAS* is set to the *best local action* in the exploitation period.

The notification mechanism of *NCD* is based on a hypothesis test which checks if there is a statistically significant change

Algorithm 2: MAB-LAA framework.

1: **Input parameters:** $\mathcal{A}_k$, $\bar{n}$
2: **for** each stage k **do**
3: Generate the set $\mathcal{A}_{ini} \subset \mathcal{A}_k$ of actions selected less than $\bar{n}$ times
4: **if** $\mathcal{A}_{ini}$ is empty **then**
5: Select an action $x_k \in \mathcal{A}_k$ according to the **policy**
6: **else**
7: Select an action x_k randomly from the set $\mathcal{A}_{ini}$
8: **end if**
9: Obtain the observation y_k from the configuration x_k
10: Update the algorithm knowledge with the values (x_k, y_k).
11: **end for**

in the samples of $F(x)$ assuming normal distribution of this random variable. Let $\mathcal{R}^x$ be the *reference set* composed by the first ν samples obtained in the exploitation period using the configuration x. Let $\mathcal{T}^x$ be the *temporal set* composed by the last ν samples obtained in the exploitation period using the configuration x. Note that there is a *reference set* and a *temporal set* for each available action in the exploitation period. Let us denote by μ_x^r and μ_x^t the average of the sets $\mathcal{R}_x$ and $\mathcal{T}_x$, respectively. A hypothesis test is carried out for each $x \in \mathcal{A}^{\text{exp}}$ considering the following null hypothesis:

$$H_0 : \mu_x^r = \mu_x^t \quad \forall x \in \mathcal{A}^{\text{exp}} \tag{4}$$

Note that the hypothesis test for action x is performed every time that ν new observations of action x are taken. In order to minimize the loss produced by a false positive in the hypothesis test (false detection of a meaningful change in the network conditions), we consider that the system response has changed if 5 consecutive tests with significance level equals to 0.01 fail. The operation of *NCD* module is detailed in Algorithm 3. Table III summarizes the most relevant parameters of the proposal.

D. Signaling and Computation Overhead

As we detail at the beginning of this section, the *DDeC* sends a configuration x_k to the network at each stage k, that is, the stage duration defines the time period between two consecutive configurations. Therefore, this is a design decision which has to be properly tunned according to the change rate of network conditions.

We define a stage as the time needed to collect enough data from the network to obtain one 5th percentile throughput sample. The duration of one stage can be variable depending on the amount of data generated by the network. Considering that the number of throughput samples needed by *DCP* to compute one 5th percentile throughput sample with statistic guarantee is fixed, the duration of one stage is a design decision depending of:

- The time required to obtain a throughput sample. Since the throughput is defined as the length of the packet divided by the time required to download it, we have to define the *throughput packet length*. For example, given 2 Mbytes of downloaded data, we obtain either 4 or 20 throughput samples for a throughput packet length of 0.5 Mbytes or 100 Kbytes, respectively.

TABLE III
NOTATION TABLE

General	x_k	Action or configuration selected at stage k
	y_k	Performance sample (5$^{\text{th}}$ percentile throughput) obtained from stage k
	$\mathcal{A}$	Set of all possible actions or configurations
AAS module	τ	Number of stages of each iteration
	$\mathcal{A}^i$	Set of actions of iteration i
	$\mathcal{I}^i$	Set of τ stages corresponding to iteration i
	x_c^i	Central action of iteration i
	$\widetilde{x^i}$	Best local action of iteration i
	$\mathcal{A}^{\text{exp}}$	Set of actions of exploitation period
MAB-LAA module	$\mathcal{A}_k$	Set of available actions for stage k
	$\bar{n}$	Minimum number of times that any action has to be selected in the algorithm initialization
NCD module	ν	Size of $\mathcal{D}_x^r$ and $\mathcal{D}_x^t$
	$\mathcal{D}_x^r$	Reference set, first ν performance samples of action x in the exploitation period
	$\mathcal{D}_x^t$	Temporal set, last ν performance samples of action x in the exploitation period
	μ_x^r, μ_x^t	Average of the sets $\mathcal{D}_x^r$ and $\mathcal{D}_x^t$, respectively

Algorithm 3: NCD operation.

Initial Variables: $\mathcal{R}^x = \emptyset$, $\mathcal{T}^x = \emptyset$, counts $= 0$
for each stage k **do**
  if Exploitation period active **then**
    Receive $x_k \in \mathcal{A}^{\text{exp}}$ and y_k
    if length($\mathcal{R}^{x_k}$) $< \nu$ **then**
      Append y_k to $\mathcal{R}^{x_k}$
    else
      Append y_k to $\mathcal{T}^{x_k}$
      if length($\mathcal{T}^{x_k}$) $= \nu$ **then**
        if HypothesisTest($\mathcal{R}^{x_k}$, $\mathcal{T}^{x_k}$) fails **then**
          counts $=$ counts $+ 1$
        else
          counts $= 0$
        end if
        Empty the set $\mathcal{T}^{x_k}$
      end if
      if counts $= 5$ **then**
        Send *Notification* to *NCD* and *MAB-AAS*
        Empty the sets $\mathcal{R}^x$ and $\mathcal{T}^x$ $\forall x \in \mathcal{A}^{\text{exp}}$
      end if
    end if
  end if
end for

- The traffic intensity. Considering that the network is not saturated, the more UEs in the network, the more throughput samples we will get per second.
- The size of the network. Similarly to the previous item, a larger network contains more UEs which will provide more throughput samples per second.

Thus, adjusting the throughput packet length as a function of the current traffic intensity and the network size, the algorithm can operate at the desired adaptation rate and signaling overhead.

Regarding the size of the network, it is divided into clusters of several sectors sharing the same ABS pattern (synchronized muting[1]). It implies that we only need one control per cluster of macro eNBs, i.e., we need as many algorithm instances as clusters we have. These clusters have to be defined by the network operator, grouping together all macro eNBs with homogeneous (similar) traffic profile, e.g., the city center, the outskirts of the city, etc. Note that the input of the algorithm is fixed regardless of the size of the network. Given a fixed input, the number of operation is always bounded $(O(1))$. In terms of storage overhead, our proposal only needs to store three elements for each configuration in $\mathcal{A}^i$, i.e., a total of $3 \cdot |\mathcal{A}^i|$ scalars, where the cardinality of $\mathcal{A}^i$ is fixed for all i.

VI. BENCHMARK ALGORITHMS

In this section we present the benchmark algorithms used for a comparative evaluation of our proposal. We consider only algorithms that, like ours, use an online approach.

When evaluating a benchmark, we replace *DDeC* module (Figs. 1 and 2) with the corresponding benchmark algorithm. Note that the input/output of benchmark algorithms match with that of *DDeC*.

A. Stochastic Gradient Ascent (SGA) Algorithms

The operation of a SGA algorithm can be summarized in three steps: 1) Take performance samples around the current action, 2) compute an estimation of the gradient using these samples, 3) move to a new location following the direction of the gradient estimation using the projector operator $\Pi_{\mathcal{P}}$ over the set $\mathcal{P}$ (i.e., $\Pi_{\mathcal{P}}(x)$ returns the configuration in $\mathcal{P}$ closest to x).

Algorithm 4 shows the general framework for SGA algorithms. These algorithms consider x to be a continous variable confined in the set $\mathcal{P}$. Note that the operation of the framework is divided in iterations denoted by t. Each iteration is composed of $|\mathcal{S}^{(t)}|$ stages where $\mathcal{S}^{(t)}$ denotes the set of actions to sample at iteration t. The differences among the SGA algorithms presented below are basically two: how the set $\mathcal{S}^{(t)}$ is generated and its size, and how the estimation of the gradient is computed.

One Sample Gradient (OSG) [31] uses an approximation of the gradient that is computed using a single random performance sample $\mathcal{S}^{(t)} = \{x^{(t)} + \delta u^{(t)}\}$ of the metric F, where $u^{(t)}$ is a random unit vector and δ is a constant determining how far of

[1]Some previous works [15], [16] have considered unsynchronized muting, i.e., that the ABS pattern can be different on each macro sector, leading to an uncertain interference profile, and hence to a more complex resource allocation problem. Since 3GPP has shown [26] that the use of synchronized muting implies a significant performance gain over unsynchronized one, we considered synchronized operation in our simulation scenario.

Algorithm 4: Stochastic Gradient Ascent Algorithms Operation.

1: **Input parameters:** Learning rates η_t, initial point x_0
2: **for** $t = 1, 2, \ldots$ **do**
3: Generate the set of configurations to sample $\mathcal{S}^{(t)}$
4: Obtain the performance samples of the metric F at $\mathcal{S}^{(t)}$
5: Compute the gradient $\tilde{g}^{(t)}$
6: Update $x^{(t+1)} = \Pi_{\mathcal{P}}(x^{(t)} + \eta_t \tilde{g}^{(t)})$
7: **end for**

$x^{(t)}$ the observation is taken. Then, the gradient is estimated by

$$\tilde{g}^{(t)} = \frac{2}{\delta} F(x^{(t)} + \delta u^{(t)}) u^{(t)}. \quad (5)$$

Multi-Sample Gradient with 2 samples (MSG2) [32] takes two performance samples $\mathcal{S}^{(t)} = \{x^{(t)} \pm \delta u^{(t)}\}$ and the gradient is estimated by

$$\tilde{g}^{(t)} = \frac{1}{2}(F(x^{(t)} + \delta u^{(t)}) - F(x^{(t)} - \delta u^{(t)}))u^{(t)}. \quad (6)$$

Multi-Sample Gradient with 3 samples (MSG3) [32] considers $\mathcal{S}^{(t)} = \{x^{(t)}, x^{(t)} + \delta e_1, x^{(t)} + \delta e_2\}$ where e_i are the unit coordinate axes. The gradient is estimated using

$$\tilde{g}^{(t)} = \frac{1}{\lambda} \sum_{i=1}^{2} \left(F(x^{(t)} + \delta e_i) - F(x^{(t)})\right) e_i. \quad (7)$$

Response Surface Methodology (RSM) [33] considers $\mathcal{S}^{(t)} = \{x^{(t)},\ x^{(t)} + (\delta_1, \delta_2)^T,\ x^{(t)} + (-\delta_1, \delta_2)^T,\ x^{(t)} + (\delta_1, -\delta_2)^T,\ x^{(t)} + (-\delta_1, -\delta_2)^T\}$. Using the elements of $\mathcal{S}^{(t)}$, we define the following matrices:

$$W^{(t)} = \begin{pmatrix} 1 & x_1^T \\ 1 & x_2^T \\ \vdots & \vdots \\ 1 & x_5^T \end{pmatrix}, y^{(t)} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_5 \end{pmatrix} \quad (8)$$

where y_i are the observation corresponding to the point x_i. Then, we estimate the gradient by

$$\tilde{g}^{(t)} = \left(\left(W^{(t)}\right)^T W^{(t)}\right)^{-1} \left(W^{(t)}\right)^T y^{(t)}. \quad (9)$$

B. Multi-Armed Bandits (MAB) Algorithms

This type of algorithms consider a finite space of actions. We will use the same discrete set $\mathcal{A}$ considered in our proposal. Algorithm 5 details the general framework of a MAB algorithm. The differences among the algorithms are the initialization parameter $\bar{n}$ and the policy to select the next action. Note that each iteration of the for loop in Algorithm 5 corresponds to one stage.

ε-greedy policy [34] selects with probability $1 - \varepsilon$ the action with higher average performance and, with probability ε, a randomly chosen action. The probability ε is a fixed value and the initialization parameter $\bar{n} = 1$.

Similarly, **ε-greedy descent** policy uses $\varepsilon = 1/k$ and **ε-greedy logarithmic descent** policy uses $\varepsilon = 1/\log(k)$ where k is the number of the current stage.

Algorithm 5: Multi-Armed Bandit Algorithms Operation.

1: **Initialization:** Select $\bar{n}$ times each action $x \in \mathcal{A}$.
2: **for** $k = 1, 2, \ldots$ **do**
3: Select an action $x_k \in \mathcal{A}$ according to the policy
4: Obtain the observations y_k of the configuration x_k
5: Update the algorithm knowledge with the values (x_k, y_k).
6: **end for**

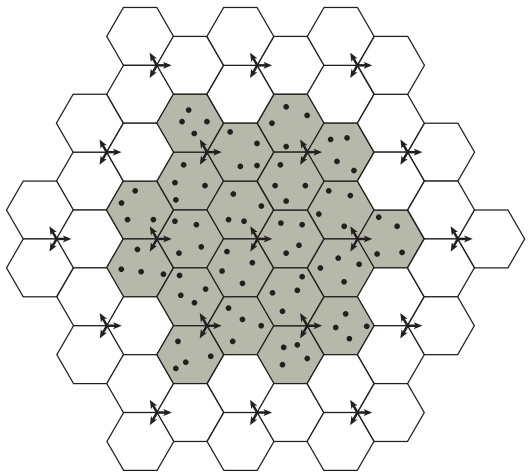

Fig. 5. Simulated scenario considering 4 pico eNBs per sector (dots) and highlighting the evaluation sectors (dark grey).

Softmax policy [35] selects each action a with probability $p_a = e^{\mu_a/\xi} / \sum_{a \in \mathcal{A}} e^{\mu_a/\xi}$ where μ_a is the average performance for the action a and ξ is a positive real parameter called the temperature. As a natural improvement of softmax, **Softmax logarithmic descent** considers $\xi = \xi_0/\log(k)$ where k is the current stage and ξ_0 is the initialization value.

UCB (Upper Confidence Bounds) normal policy [28] selects at each stage the action with the higher upper confidence bound considering that the performance samples are gaussian distributed with unknown mean and variance. When sampling an action the estimators of the parameters (mean and variance) of the distribution of this action are updated.

Thompson sampling normal policy [30] selects the maximum among the samples drawn from posterior beliefs of the mean estimator of every action.

For the precise formulation of UCB normal and Thompson sampling normal consult [28], [30].

VII. Numerical Evaluation

In this section we compare our proposal against the benchmark algorithms both under stationary and variable network conditions. Moreover, we address the adjustment of the configurable parameter τ.

A. Description of the Simulated Scenario

The simulated scenario is based on the 3GPP guidelines for LTE performance evaluation [27]. The wireless channel is modeled with deterministic pathloss attenuation and random shadow fading. The simulation layout is shown in Fig. 5, comprising 57 sectorial macro eNBs of 120 degrees (19 macro eNBs), and the

TABLE IV
SIMULATION PARAMETERS

Network layout	19 macro eNBs, 57 sectorial macro eNBs, 500 m ISD, 2, 4, 12 pico eNBs per sector for scenario 1, 2 and 3, respectively
System bandwidth	10 MHz
LTE frame duration	Subframe 1 ms, Protected-subframe pattern 8 ms, Frame 10 ms
Transmit power	Macro eNB 46 dBm, pico eNB 30 dBm
Macro sector antenna pattern	$A_H(\phi) = -min[12(\frac{\phi}{\phi_{3dB}})^2, A_m]$, A_m = 70 degrees A_m = 25 dB
Pico antenna pattern	Omnidirectional
Antenna gains	macro: 14 dBi; pico: 5 dBi; UE: 0 dBi
Macro to UE path loss	128.1 + 37.6 · $\log_{10}(R[\text{Km}])$ where R is the macro eNB to UE distance
Pico to UE path loss	149.7 + 36.7 · log10(R[Km]) where R is the pico eNB to UE distance
Shadow fading	Lognormal distribution with 10 dB standard deviation
Thermal noise	-176 dBm
Scheduling algorithm	Proportional Fair (PF)
Traffic model	File Transfer Protocol (FTP)
File size	0.5 Mbytes
λ [UEs/s] (Offered traffic load [Mbps])	18.75 UEs/s (75 Mbps) per sector
Minimum distances	Macro - pico: 70 m; Macro - UE: 35 m; Pico - pico: 40 m; Pico - UE: 10 m

simulation parameters are shown in Table IV. Several pico eNBs overlaps each eNB sector (shown as dots in Fig. 5), and pico eNBs can be (de)activated over time due to, for example, energy saving mechanisms.

The simulations are executed for the central 21 sectorial macro eNBs (dark grey zone in Fig. 5), while the rest of macro eNBs emulate the interference of a larger network. The total interference at each UE receiver is the aggregation of all interfering eNBs in the sector (macro and picos) plus the interference from the four nearest sectorial macro eNBs.

In the simulation, UEs arrive following a Poisson process of rate λ arrivals per second on each sector. When considering dynamical network conditions, λ varies during simulation time. According to the 3GPP FTP traffic model [27], each incoming UE downloads one file, and then leaves the network. In addition, each UE is dropped uniformly over the macro eNB coverage area with probability $\frac{1}{3}$, or over a pico eNB coverage area with probability $\frac{2}{3}$.

B. Evaluation Procedure

In order to evaluate the algorithms, we follow the next procedure:

- First, the associated performance $F(x)$ is evaluated for each configuration $x \in \mathcal{A}$. We consider that the set $\mathcal{A}$ has a total of 285 components, as a consequence of the combination of 15 and 19 values for γ and ϕ, respectively. Evaluation is performed by Monte Carlo, running 40 trials for each configuration.
- A Normal distribution is assumed to characterize $F(x)$ at each point x. Its mean $\mu_{F(x)}$ and variance $\sigma^2_{F(x)}$ are obtained from their MSE estimators from the samples obtained above.
- For configurations x outside the set $\mathcal{A}$, mean and variance are approximated by the MSE polynomial fits of degree 4 to the results obtained in the previous step. Henceforth, let $\mathbb{P}\mu$ and $\mathbb{P}\sigma^2$ denote these polynomial fits.
- The optimal configuration is determined as point x^* (see[2]) maximizing $\mathbb{P}\mu$.
- Then, to evaluate performance for a given algorithm, the states trajectory $x_1, x_2, \ldots, x_k, \ldots$ is determined by: (i) sampling the Gaussian process $F(x_k) \sim N(\mathbb{P}\mu(x_k), \mathbb{P}\sigma(x_k))$ to obtain sample y_k, (ii) determining x_{k+1} from x_k and y_k following the corresponding algorithm. For each stage k, the regret is updated by adding $\mathbb{P}\mu(x^*) - \mathbb{P}\mu(x_k)$. The total regret is computed by performing 5 independent trajectory runs each comprising 5000 stages.

[2] x^* is determined to allow regret computation within the simulator, but it is not required in a real operating network.

C. Stationary Network Conditions

In this setup we assume that network conditions (traffic intensity and number of active pico eNBs) do not change during evaluation, but UE generation is still a stochastic process as described above. Besides, initial state x_1 is set $(1.5, 1.5)$.

To optimize performance, SGA algorithms have been configured with a decreasing step-size following the function $c/\sqrt{k}$, where c is a constant and k is the current stage. The value of c has been set to minimize the regret in Scenario 2, and has been obtained by exhaustive simulation. Moreover, for all algorithms $\delta = 1$ except RSM, which uses $\delta = 0.25$.

Regarding the configurable parameters of the MAB algorithms, ε-greedy is configured with $\varepsilon = 0.15$, softmax with $\xi = 0.005$, softmax logarithmic descent with $\xi_0 = 1$ and Thompson Sampling with uniform prior for σ [30]. Our proposal is evaluated considering a fixed $m = 4$ and the best value of τ obtained by exhaustive simulation for each scenario.

At this point, we want to highlight the importance of the regret in an online setting. The regret accumulates, at each stage, the distance from the expected performance of the current action to the expected performance of the best action (1). That is, the farther from the best performance, the higher the regret is. Besides, this metric gives us information about something similar to the convergence of the algorithm. Note that we cannot talk about convergence rigorously in a non-stationary online setting since the optimal action changes over time. For this reason, our objective is to be as close as possible to the optimal action at every stage. It can be seen in the regret: when the slope of the regret curve is close to zero, the algorithm is very close to the optimum. However, the higher the regret slope, the larger the distance between the actions

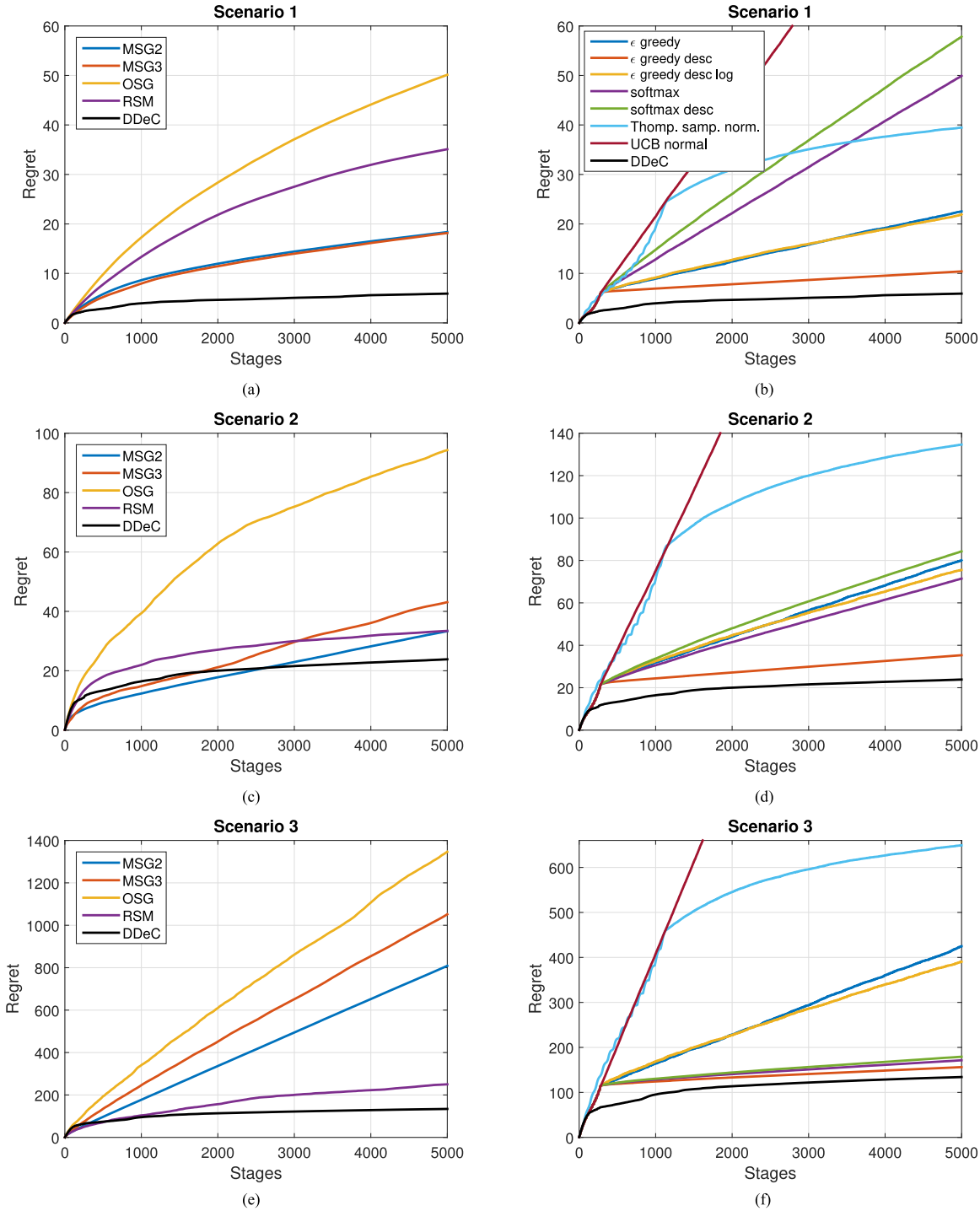

Fig. 6. Numerical evaluation in a network with fixed conditions. Three scenarios and two families of algorithms (SGA and MAB) are considered. (a) Stochastic gradient ascent algorithms, scenario 1. (b) Multi-armed bandit algorithms, scenario 1. (c) Stochastic gradient ascent algorithms, scenario 2. (d) Multi-armed bandit algorithms, scenario 2. (e) Stochastic gradient ascent algorithms, scenario 3. (f) Multi-armed bandit algorithms, scenario 3.

selected by an algorithm and the optimal action in terms of performance.

Fig. 6 shows the numerical results for Scenarios 1, 2 and 3 in term of regret and Fig. 7 shows a trace of the 5th percentile throughput in Scenario 1. Note that the performance of each algorithms with respect to others changes depending on evaluated scenario. That is because the system response and its variance changes with the scenario and therefore the optimal values of the configuration parameters of each algorithm such as the step-size, λ, ε, etc. See for example that RSM algorithm is not the best option in Scenario 1 but its performance increases in Scenario 3 with respect to the others. Something similar occurs with the Softmax policy. Finding the optimal parameter configuration for each algorithm in each scenario is a very complicated task. Even more when we consider a real operating network, where the scenario (system response) is continuously chang-

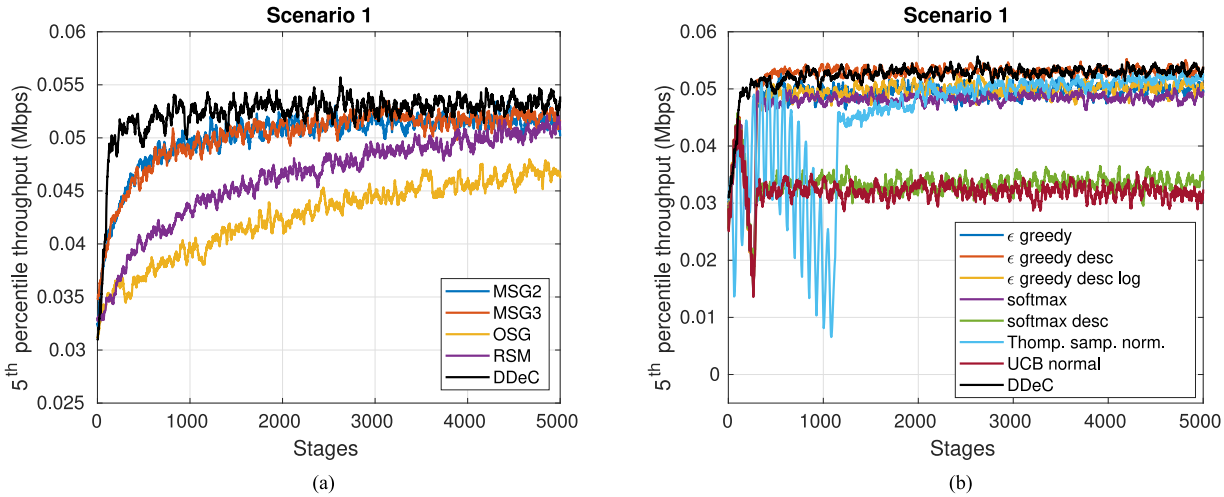

Fig. 7. Numerical evaluation in a network with fixed conditions. Performance in terms of 5th percentile throughput for Scenario 1. (a) Stochastic gradient ascent algorithms. (b) Multi-armed bandit algorithms.

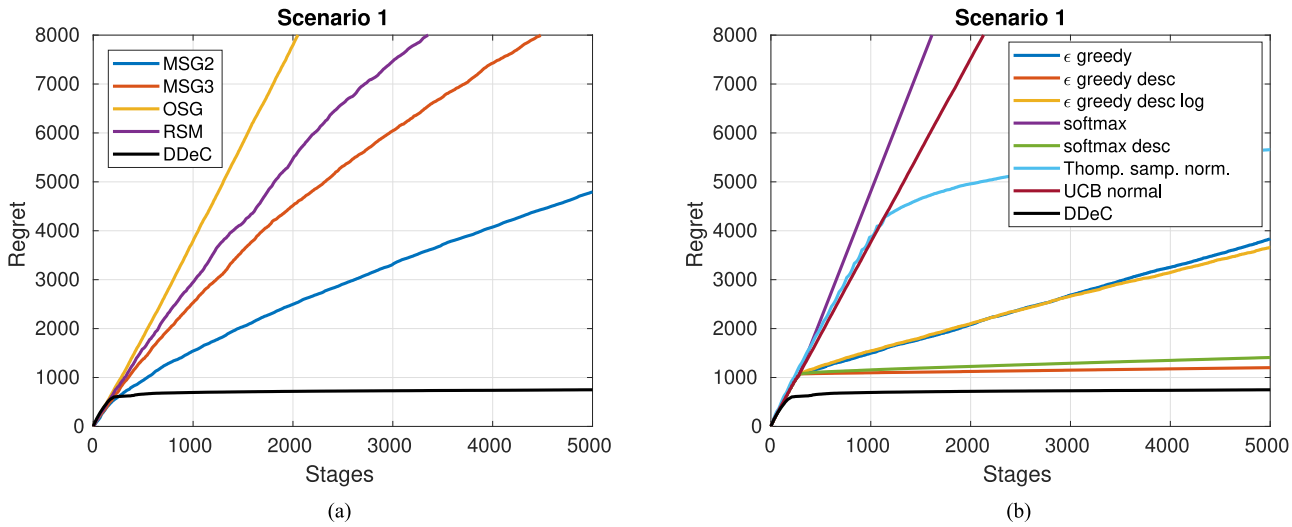

Fig. 8. Numerical evaluation considering the throughput as the objective performance metric. Performance in terms of regret for Scenario 1. (a) Stochastic gradient ascent algorithms. (b) Multi-armed bandit algorithms.

ing over time. Therefore, the parameter tunning dependency is a serious drawback of our benchmarks. In contrast, although our proposal operates with the optimal value of τ for each scenario, it is highly insensitive to changes in τ, as we discuss in Section VII-F.

Note that, for MAB algorithms, the simpler policies (ε-greedy and softmax) obtain better results than sophisticated ones (UCB normal and Thompson sampling normal) in our setting. In the case of the UCB Normal policy, it has to explore by definition each action at least $\lceil 8\log(k) \rceil$ times at stage k and, given the total number of actions in our setting ($|\mathcal{A}| = 285$), it behaves like a random policy. On the other hand, although the Thompson sampling policy finally finds the region of the optimal configuration (the slope of its regret tends to zero) it takes more stages than other algorithms. However, ε-greedy descent policy which is one of the simplest policies, obtains a low regret in all scenarios.

It is worth highlighting that, although our proposal considers only a finite set $\mathcal{A}$ of configurations, which in general does not contain x^*, it obtains lower regret than SGA algorithms which operates over the whole set $\mathcal{P}$ and could, in principle, converge to x^*. In other words, the configurations selected by our proposal are, in average, closer to the optimum than the configurations selected by SGA algorithms.

As we claim in Section IV-A, our proposal can operate regardless of the selected performance metric $F(x)$. To illustrate that, we run our algorithm considering the average throughput instead of the 5th percentile throughput as our objective performance metric. We consider the same configuration parameters described in this section. Figs. 8 and 9 show the performance in terms of regret and throughput for Scenario 1, respectively. Note that, although MAB algorithms show a similar performance as in Fig. 6, SGA algorithms perform significantly worse. A parameter tuning of SGA algorithms may increase their

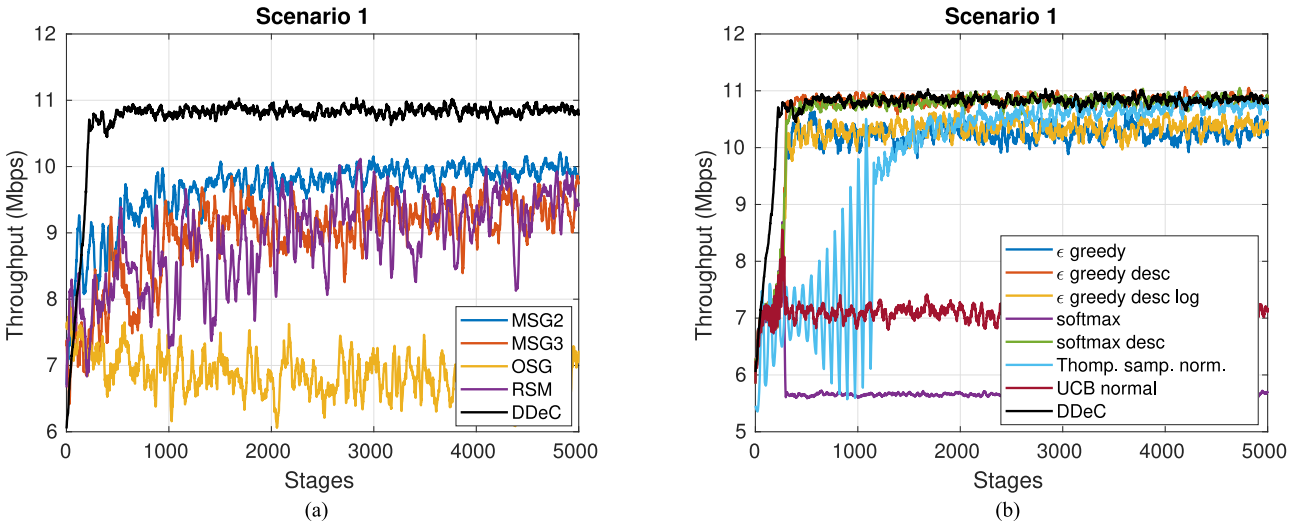

Fig. 9. Numerical evaluation considering the throughput as the objective performance metric. Performance in terms of throughput for Scenario 1. (a) Stochastic gradient ascent algorithms. (b) Multi-armed bandit algorithms.

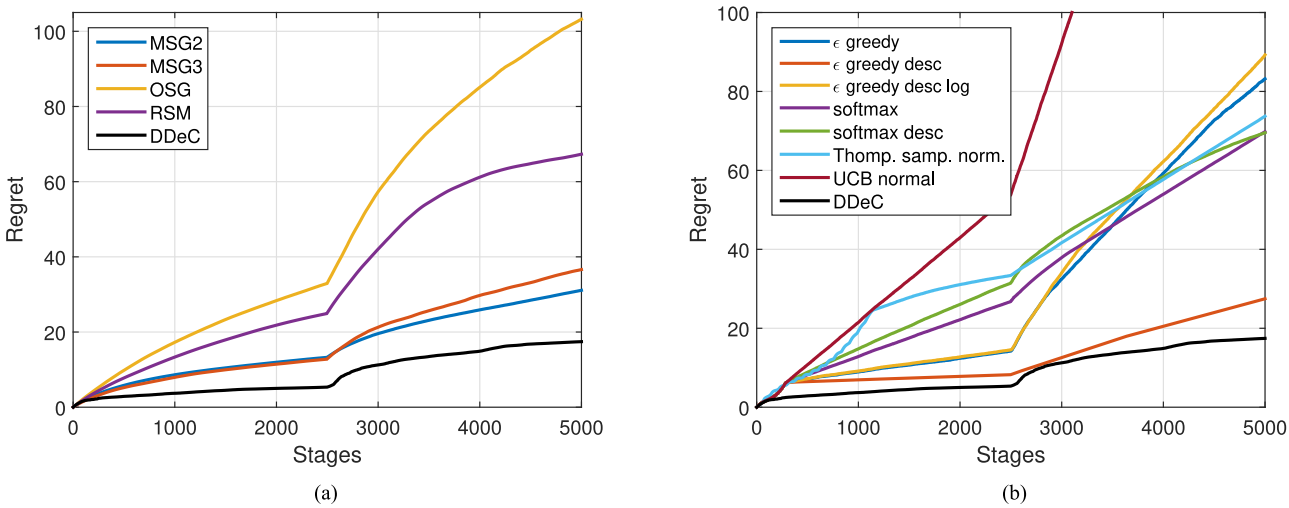

Fig. 10. Numerical evaluation in a network with changing conditions. We switch from Scenario 1 to Scenario 2 at stage 2500. (a) Stochastic gradient ascent algorithms. (b) Multi-armed bandit algorithms.

performance. However, our proposal obtains always the lowest regret regardless the scenario and the selected performance metric without tunning any parameter.

D. Dynamic Network Conditions

In this case, we assume changes in the network conditions in runtime: We switch from Scenario 1 to Scenario 2 (described in previous section) at stage 2500. This change would be produced, for example, by an energy saving mechanism which activates or deactivates pico eNBs depending on the network load. A different scenario implies, in general, a different system response, and therefore a change on the region where the most efficient configurations are located. Consequently, after stage 2500 we consider the new optimal configuration x^* in the regret computation. Regarding parameter configuration of the algorithms, we consider the same of previous section and $\nu = 5$ for the *NCD* submodule.

Fig. 10 shows that our proposal also obtains lower regret than other algorithms in a dynamic setting. Benchmark algorithms are not aware of the change in network conditions, which affects their performance negatively. For example, SGA algorithms diminish their step-size at each iteration. This assures convergence under stationary conditions, but if these conditions undergo a significant change, the step-size should be restarted. A usual solution to this issue is to use a fixed step-size which can help to track the changes in system response, in exchange for worse performance under stationary conditions.

MAB algorithms are also designed for stationary scenarios, in which they aim at finding the best performing action based on the observed history, as this history becomes sufficiently long. However, when switching from Scenario 1 to Scenario 2, past history becomes not only irrelevant but misleading for MAB algorithms, since they keep selecting the arms that were identified as optimal for Scenario 1 but perform poorly in Scenario 2 [see Fig. 10(b)]. One solution for MAB algorithms is to be restarted when the network conditions changes, forgetting all the information collected until this point. It implies that the MAB initialization phase will select $\bar{n}$ times all actions in $\mathcal{A}$.

In contrast, when our proposal detects a change in system response using the *NCD* module, the search period restarts in the *AAS* module, which only selects actions nearby the current one. In order to evaluate the system responsiveness, we measured empirically the number of stages that *NCD* needs in average to detect the variation on the network conditions. For that purpose, we evaluated multiple simulation runs these three scenarios obtaining always similar results, a quick adaption which takes in average 26 stages. Moreover, the number of false positives (detect a erroneous change in the scenario) was null in all the simulations.

E. Comparison to a Model-Driven Approach

The regret of an algorithm essentially gives us the performance gap of the algorithm with respect to the best possible configuration. However, in order to illustrate the benefits of our model-free approach, we compare our proposal to an existing model-based scheme: PF-ABS [15]. Table V shows the performance values attained by each algorithm in terms of 5th and 50th percentile throughput in three different scenarios characterized by the number of pico eNBs per sector (10, 14 and 18). *DDeD* outperforms PF-ABS in terms of 5th percentile throughput in all scenarios, while PF-ABS achieves better results in terms of 50th percentile throughput only for low density scenarios. These results are consistent with the fact that DDeC was configured to optimize the 5th percentile throughput, but its optimization objective can be changed to other metric (see Section VII-C). In contrast, PF-ABS is not able to change its optimization objective.

In addition, PF-ABS assumes PF as a scheduling algorithm and, using the PF metric, configures the ABS ratio of each eNB. As a result, PF-ABS uses local configurations. This allows a distributed implementation but, in general, it tends to perform worse than global configurations (see Section V-D). In order to compare both schemes fairly, we show the PF-ABS results using the optimal CRE bias for each scenario.

We would like to emphasize that our scheme does not necessarily replace model-based approaches. Instead, both approaches can complement each other. For example, in a plug-and-play scenario, a model-driven algorithm can be used to find an efficient configuration given the current network conditions. Then, our data-driven mechanism can use this configuration as an initial point to perform a fine parameter tuning and track the optimal configuration over time.

F. Evaluation of τ Parameter

As discussed in Section V-A, τ balances the accuracy and the adaption velocity of our algorithm. For example, a large value of τ implies a more accurate selection of the best local action but taking more time in each iteration, as a tradeoff.

The optimal value of this parameter (τ^*) is the one that minimizes the total regret of our proposal and depends on the system response (network conditions). By simulation, we found that the values of τ^* are 34, 31 and 29 for scenarios 1, 2 and 3, respectively. However, since the system response is a priori unknown and changes continuously over time, we are no able to find τ^* at every step, specially in a real operating network. For this reason, we evaluate the average relative loss[3] of using $\bar{\tau}$ (the average of the optimal values for these tree scenarios) as a fixed value instead of τ^* of each scenario.

Our numerical results show that, although the system response and the optimal configuration of the network changes with network conditions (e.g., traffic load, pico eNB topology), we can configure our algorithm with $\bar{\tau}$ as a fixed value not implying noticeable losses (0.0007% per stage).

[3]Between configurations using the optimal τ and that using $\bar{\tau}$.

TABLE V
COMPARISON TO A MODEL-DRIVEN APPROACH

	PF-ABS	
	5^{th} percentile throughput	50^{th} percentile throughput
Scenario 1	0.2714	6.2500
Scenario 2	0.6016	7.4627
Scenario 3	0.9141	7.4732
	DDeC	
	5^{th} percentile throughput	50^{th} percentile throughput
Scenario 1	0.5616	5.2808
Scenario 2	0.9350	6.7536
Scenario 3	1.1925	7.6280

VIII. CONCLUSION

In this paper, we have presented a Data-Driven eICIC Configurator (*DDeC*) for interference coordination in HetNets environments based on a big data-driven framework for network optimization. It finds efficient configurations of eICIC parameters (ABS ratio and CRE bias) operating without any previous knowledge of the network (e.g., traffic load, topology, propagation conditions) only using data retrieved from the network. This data-driven approach allows us to optimize metrics of practical interest that are analytically intractable, such as 5th percentile throughput. Our numerical results show that our proposal achieves a significant improvement with respect to the state of the art of SGA and MAB algorithms in networks with both stationary and variable conditions.

REFERENCES

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

[2] S. Samulevicius, T. B. Pedersen, and T. B. Sorensen, "MOST: Mobile broadband network optimization using planned spatio-temporal events," in *Proc. IEEE 81st Veh. Technol. Conf.*, 2015, pp. 1–5.

[3] A. Liu, V. K. N. Lau, L. Ruan, J. Chen, and D. Xiao, "Hierarchical radio resource optimization for heterogeneous networks with enhanced inter-cell interference coordination (eICIC)," *IEEE Trans. Signal Process.*, vol. 62, no. 7, pp. 1684–1693, Apr. 2014.

[4] M. Simsek, M. Bennis, and I. Güvenç, "Learning based frequency-and time-domain inter-cell interference coordination in HetNets," *IEEE Trans. Veh. Technol.*, vol. 64, no. 10, pp. 4589–4602, Oct. 2015.

[5] W. Jin *et al.*, "Joint user association and ABS proportion optimization for load balancing in HetNet," in *Proc. Int. Conf. Wireless Commun. Signal Process.*, Nanjing, China, Oct. 2015, pp. 1–6.

[6] A. Bin Sediq, R. Schoenen, H. Yanikomeroglu, and G. Senarath, "Optimized distributed inter-cell Interference coordination (ICIC) scheme using projected subgradient and network flow optimization," *IEEE Trans. Commun.*, vol. 63, no. 1, pp. 107–124, Jan. 2015.

[7] D.-H. Sung and J. S. Baras, "Utility-based almost blank subframe optimization in heterogeneous cellular networks," in *Proc. IEEE Global Commun. Conf.*, 2014, pp. 3622–3627.

[8] A. Daeinabi and K. Sandrasegaran, "A fuzzy Q-learning approach for enhanced intercell interference coordination in LTE-Advanced heterogeneous networks," in *Proc. 20th Asia-Pac. Conf. Commun.*, 2014, pp. 139–144.

[9] M. Al-Rawi, "A dynamic approach for cell range expansion in interference coordinated LTE-advanced heterogeneous networks," in *Proc. IEEE Int. Conf. Commun. Syst.*, 2012, pp. 533–537.
[10] S. Mishra, A. Sengupta, and C. S. R. Murthy, "Enhancing the performance of HetNets via linear regression estimation of range expansion bias," in *Proc. 19th IEEE Int. Conf. Netw.*, 2013, pp. 1–6.
[11] N. Trabelsi, L. Roullet, and A. Feki, "A generic framework for dynamic eICIC optimization in LTE heterogeneous networks," in *Proc. IEEE 80th Veh. Technol. Conf.*, 2014, pp. 1–6.
[12] S. Deb, P. Monogioudis, J. Miernik, and J. P. Seymour, "Algorithms for enhanced inter-cell interference coordination (eICIC) in LTE HetNets," *IEEE/ACM Trans. Netw.*, vol. 22, no. 1, pp. 137–150, Feb. 2014.
[13] S. Vasudevan, R. N. Pupala, and K. Sivanesan, "Dynamic eICIC-A proactive strategy for improving spectral efficiencies of heterogeneous LTE cellular networks by leveraging user mobility and traffic dynamics," *IEEE Trans. Wireless Commun.*, vol. 12, no. 10, pp. 4956–4969, Oct. 2013.
[14] M. S. Ali, P. Coucheney, and M. Coupechoux, "Load balancing in heterogeneous networks based on distributed learning in near-potential games," *IEEE Trans. Wireless Commun.*, vol. 15, no. 7, pp. 5046–5059, Jul. 2016.
[15] B. Soret and K. Pedersen, "Centralized and distributed solutions for fast muting adaptation in LTE-advanced HetNets," *IEEE Trans. Veh. Technol.*, vol. 64, no. 1, pp. 147–158, Jan. 2015.
[16] M. Al-Rawi, J. Huschke, and M. Sedra, "Dynamic protected-subframe density configuration in LTE heterogeneous networks," in *Proc. 21st Int. Conf. Comput. Commun. Netw.*, Munich, Germany, Jul. 2012, pp. 1–6.
[17] K. Pedersen, B. Soret, S. B. Sanchez, G. Pocovi, and H. Wang, "Dynamic enhanced inter-cell interference coordination for realistic networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 7, pp. 5551–5562, Jul. 2016.
[18] O.-C. Iacoboaiea, B. Sayrac, S. B. Jemaa, and P. Bianchi, "SON coordination in heterogeneous networks: A reinforcement learning framework." *IEEE Trans. Wireless Commun.*, vol. 15, no. 9, pp. 5835–5847, Sep. 2016.
[19] V. Chandrasekhar *et al.*, "Interference management and network performance optimization in small cells," U.S. Patent 20 160 080 949, Mar. 17, 2016.
[20] M. Cierny, H. Wang, R. Wichman, Z. Ding, and C. Wijting, "On number of almost blank subframes in heterogeneous cellular networks," *IEEE Trans. Wireless Commun.*, vol. 12, no. 10, pp. 5061–5073, Oct. 2013.
[21] Y. Wang, H. Ji, and H. Zhang, "Spectrum-efficiency enhancement in small cell networks with biasing cell association and eICIC: An analytical framework," *Int. J. Commun. Syst.*, vol. 29, pp. 362–377, 2016.
[22] L. Zhou *et al.*, "A dynamic graph-based scheduling and interference coordination approach in heterogeneous cellular networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 5, pp. 3735–3748, May 2016.
[23] W. Wang, A. Kwasinski, D. Niyato, and Z. Han, "A survey on applications of model-free strategy learning in cognitive wireless networks," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 1717–1757, Jul.–Sep. 2016.
[24] "Evolved universal terrestrial radio access (e-utra) and evolved universal terrestrial radio access network (e-utran); overall description (release 10)," Eur. Telecommun. Standards Inst., Sophia Antipolis, France, Tech. Spec. 3GPP 36.300 v10.5.0, 2011.
[25] B. Soret and K. I. Pedersen, "Macro transmission power reduction for HetNet co-channel deployments," in *Proc., IEEE Global Commun. Conf.*, Anaheim, CA, USA, Dec. 2012, pp. 4126–4130.
[26] "System performance of heterogeneous networks with range expansion," Eur. Telecommun. Standards Inst., Sophia Antipolis, France, Tech. Rep. 3GPP R1-100142, 2010.
[27] "Evolved universal terrestrial radio access (eutra); further advancements for e-utra physical layer aspects," Eur. Telecommun. Standards Inst., Sophia Antipolis, France, Tech. Rep. 3GPP TR 36.814, 2010.
[28] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, pp. 235–256, 2002.
[29] S. Bubeck and *et al.*, "Regret analysis of stochastic and nonstochastic multi-armed bandit problems," *Found. Trends R Mach. Learn.*, vol. 5, no. 1, pp. 1–122, 2012.
[30] J. Honda and A. Takemura, "Optimality of thompson sampling for Gaussian bandits depends on priors," in *Proc. 17th Int. Conf. Artif. Intell. Statist.*, 2014, pp. 375–383.
[31] A. D. Flaxman, A. T. Kalai, and H. B. McMahan, "Online convex optimization in the bandit setting: Gradient descent without a gradient," in *Proc. 16th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2005, pp. 385–394.
[32] A. Agarwal, O. Dekel, and L. Xiao, "Optimal algorithms for online convex optimization with multi-point bandit feedback." in *Proc. 27th Annu. Conf. Learn. Theory*, 2010, pp. 28–40.
[33] R. H. Myers, D. C. Montgomery, and C. M. Anderson-Cook, *Response Surface Methodology: Process and Product Optimization Using Designed Experiments.* Hoboken, NJ, USA: Wiley, 2016.
[34] V. Kuleshov and D. Precup, "Algorithms for multi-armed bandit problems," unpublished paper, 2014. [Online]. Available: https://arxiv.org/abs/1402.6028
[35] J. Vermorel and M. Mohri, "Multi-armed bandit algorithms and empirical evaluation," in *Proc. 16th Eur. Conf. Mach. Learn.*, 2005, pp. 437–448.

Jose A. Ayala-Romero received the B.Sc. degree in telematics engineering and the M.Sc. degree in telecommunication engineering from the Technical University of Cartagena, Cartagena, Spain, in 2012 and 2014, respectively, where he is currently working toward the Ph.D. degree with the Department of Information and Communications Technologies. His research interests include cellular networks and learning algorithms.

Juan J. Alcaraz received the M.Sc. degree in telecommunications engineering from the Polytechnical University of Valencia, Valencia, Spain, in 1999, and the Ph.D. degree in telecommunications engineering from the Technical University of Cartagena (UPCT), Cartagena, Spain, in 2007. Between 1999 and 2004, he was employed by several technology companies. In 2004, he joined UPCT, where he is currently an Associate Professor with the Department of Information and Communication Technologies. He was a Fulbright Visiting Scholar with the Electrical Engineering Department, University of California, Los Angeles, in 2013, and a Visiting Professor with the Department of Information Engineering, University of Padova, in 2017. His current research focuses on learning algorithms for wireless network management.

Javier Vales-Alonso received the M.Sc. and Ph.D. degrees in telecommunication engineering from the University of Vigo, Vigo, Spain, and the Technical University of Cartagena, Cartagena, Spain, in 2000 and 2005, respectively, and the M.Sc. degree in mathematics from the National University of Distance Education, Madrid, Spain, in 2015. His research interests include modeling and optimization.