Wireless Pers Commun
DOI 10.1007/s11277-016-3843-2

Implementation of Machine Learning for Autonomic Capabilities in Self-Organizing Heterogeneous Networks

Plamen Semov[1] · Hussein Al-Shatri[2] · Krasimir Tonchev[1] · Vladimir Poulkov[1] · Anja Klein[2]

Abstract The 3GPP's self-organizing networks (SONs) standards are a huge step towards the autonomic networking concept. They are the response to the increasing complexity and size of the mobile networks. This paper proposes a novel scheme for SONs. This scheme is based on machine learning techniques and additionally adopting the concept of abstraction and modularity. The implementation of these concepts in a machine learning scheme allows the usage of independent vendor and technology algorithms and reusability of the proposed approach for different optimization tasks in a network. The scheme is tested for solving an energy saving optimization problem in a heterogeneous network. The results from simulation experiments show that such an approach could be an appropriate solution for developing a full self-managing future network.

Keywords Self-organizing networks · Machine learning · Energy saving · Self-managing network

✉ Vladimir Poulkov
vkp@tu-sofia.bg

Plamen Semov
p_semov@abv.bg

Hussein Al-Shatri
h.shatri@nt.tu-darmstadt.de

Krasimir Tonchev
k_tonchev@tu-sofia.bg

Anja Klein
a.klein@nt.tu-darmstadt.de

[1] Faculty of Telecommunications, Technical University of Sofia, Sofia, Bulgaria

[2] Institute for Telecommunications, Technical University Darmstadt, Darmstadt, Germany

Published online: 24 October 2016

1 Introduction

With the development of today's mobile networks towards next generation Heterogeneous Networks (HetNets), the Operation, Administration and Management (OAM) and network optimization issues are becoming more and more complex. Nowadays, the optimization of such networks does not only require deep understanding of the implemented technologies, but also a huge amount of man/hours for processing and analysis of the statistical and signaling data generated from the network before taking any decision and/or action. To overcome this problem an initiative was started called Autonomic Networking, with its ultimate target being to create self-managing networks, which will enable further growth with less or none human intervention [1]. The first response towards solving this problem in mobile networks is the development of the so called self-organizing network (SON) standards for the 3GPP 4th generation mobile networks. Even though that these standards are more or less technology specific, there is plenty of room for vendors of telecommunication equipment to implement their own algorithms. But in most cases the practically implemented algorithms are not applicable in multi-vendor network scenarios. In addition, it must be noted that the digitally recorded data, such as the signaling information between the base stations (BS) and mobile terminals and the signaling information inside the mobile network, is not used efficiently for the implementation of SON. On the other hand, with more and more available collected and recorded digital data, it becomes obvious that there are meaningful relations and information in the data archives that are way too big and too complex for the practical application of standard data processing approaches. Therefore, SON should evolve in a new concept called cognitive network (CN) or fully autonomic network (FAN) [1]. This can be achieved by using the power of artificial intelligence (AI) approaches or machine learning (ML) algorithms.

In this paper, we illustrate the implementation of a novel model for the introduction of autonomic capabilities in SON initially proposed in [2]. The idea is based on simplification of a complex task by splitting it into smaller ones that could be much more easily solved with ML learning techniques through the approaches of abstraction and modularity and adaptation via learning from the preprocessed data generated from the network. Based on this a scheme that is technology and vendor independent and reusable for other common optimization tasks is achieved.

The model is tested in a simulation scenario of a heterogeneous mobile network to solve an energy saving problem.

The rest of the paper is organized as follows. In the next section the state of the art related to the implementation of SON algorithms based on machine learning and the system model are presented. In Sect. 3, the proposed cognitive plane and machine learning scheme are described and how they are adopted for the implementation of an energy saving mechanism. The results are presented and analyzed in Sect. 4. Finally, we conclude the paper in Sect. 5.

2 Overview and System Model

2.1 Overview

In SON the algorithms for self-organization are control loops, which usually can be implemented in a mobile network by deploying them either in the control plane or in the management plane. Control plane SON algorithms in long term evolution mobile network (LTE) are built in the eNodeBs (eNB), which are sometimes denoted as on-line algorithms. Management plane SON solutions (off-line) are deployed in the network management system (NMS) and can be used from the telecom operator in the operation and maintenance center (OMC) by means of the application programming interfaces (APIs) [3]. Control plane SON functions are usually vendor specific algorithms, which will as was mentioned above hardly will work in a multi-vendor network.

These days there is a great interest in data mining and ML, because of the increased computational efficiency of the hardware. There are many papers, which present optimization solutions for SON scenarios, but rather few of them implement ML or data mining. In [4], a cognitive engine based on case-based reasoning and decision tree searches for improving coverage in 3G wireless networks is proposed. Again in [5], decision tree based unsupervised learning is used for network selection in wireless HetNets. The authors in [6] propose an algorithm for handover self-optimization using big data analytics. In [7] a support vector regression for autonomous parameter optimization of a heterogeneous wireless network aggregation system is introduced. In [8–13] reinforcement learning algorithms are used for coverage, capacity optimization and spectrum occupancy forecasting. K-means clustering and support vector machine learning (SVM) are other powerful algorithms, which are used in [14] for cooperative spectrum sensing in cognitive radio networks. By using traffic statistics in [15], an algorithm is proposed for self-optimization of the antenna configuration in LTE aiming at energy saving. For load balancing and handover optimization a reinforcement Q-learning technique is used respectively in [16, 17].

2.2 System Model

The simulated network was performed in a MATLAB environment. The simulation was initially developed by Aalborg University following 3GPP recommendations for network simulations. The simulation was used in papers [18, 19], but further modifications were made for the purposes of this paper with the support of the Institute for Telecommunications at the Technical University of Darmstadt.

The simulated system model represents an urban environment with 19 macro sites (MBS) with 3 macro cells per site as shown in Fig. 1. Additionally, three micro sites (mBS) per macro cell are distributed uniformly with omni antennas. The MBSs have a height of 30 m and an electrical down-tilt of 15° for each antenna as proposed by 3GPP in [20]. The path loss Model (PL) is the Model 2 proposed by 3GPP in [20] which considers different PL for line of sight (LOS) and non-LOS (NLOS) conditions. The MBS antenna gain pattern is as proposed in [20]. Equation (1) is for vertical gain, (2) for horizontal and (3) is the overall gain.

$$GainV = -min\left\{12 * \left(\frac{\theta - \varepsilon_{dtilt}}{\theta_{3\,\mathrm{dB}}}\right)^2, Gain_{min}^{\theta}\right\}, \tag{1}$$

where θ is the vertical angle, ε_{dtilt} is the electrical down-tilt of the antenna, $\theta_{3\ \mathrm{dB}}$ the value of the θ at 3 dB and $Gain_{min}^{\theta}$ the minimum vertical gain.

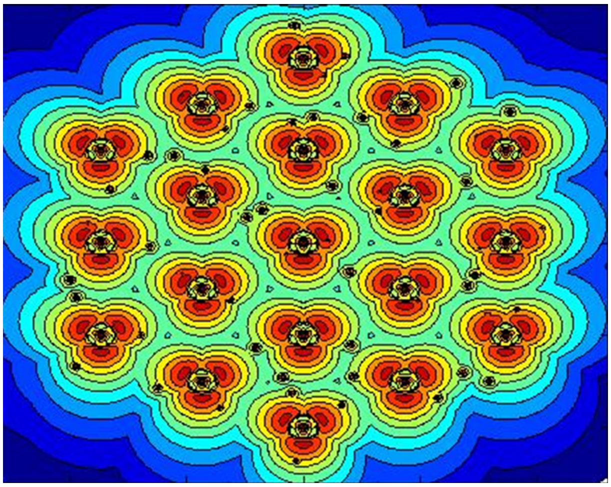


Fig. 1 Simulated network topology

$$GainH = -min\left\{12 * \left(\frac{\varphi}{\varphi_{3\,\mathrm{dB}}}\right)^2, Gain_{min}^{\varphi}\right\}, \tag{2}$$

where φ is the horizontal angle, $\varphi_{3\ \mathrm{dB}}$ the value of the φ at 3 dB and $Gain_{min}^{\varphi}$ the minimum horizontal gain. All angles are in degrees [°].

$$Gain_{Mcro} = -min\{-(GainV + GainH), Gain_{min}^{\varphi}\}. \tag{3}$$

The mBSs are modeled with omnidirectional antenna patterns, with Gainmicro = 0 [dB]. The signal-to-interference-plus-noise ratio (SINR) is given with Eq. (4):

$$SINR_{c,u}^{n} = \frac{P_{TX,c,u}^{n} \cdot \left|h_{c,u}^{n}\right|^2}{\sigma^2 + P_{micro,u}^{n} + P_{Macros,u}^{n}}, \tag{4}$$

where u is the user equipment (UE), c the Cell and n the Physical Resource Block (PRB), P_{TX} the transmission power, $|h|^2$ the channel gain between the user and the cell and σ^2 is the variance of the AWGN. $P_{micro,}$ and P_{Macros} are the powers received by the UE, from the micro cells and macro cells respectively. The achievable rate can be calculated, as expressed in Eq. (5):

$$R_u = \sum_n \log_2\left(1 + A_{t_{c,u}}^{n} * SINR_{c,u}^{n}\right), \tag{5}$$

where $A_{t_{c,u}}^{n}$ depicts the influence of a Rayleigh Fading Channel, giving this a more realistic approach. The Rayleigh fading channel, as proposed in [21], is simulated as a random variable that follows an exponential distribution with a mean value of 1. The cell load through the day is modeled as in [22] and it is shown on Fig. 2. The power consumption model for MBS and mBS is based on Eqs. (6) and (7) respectively [23]:

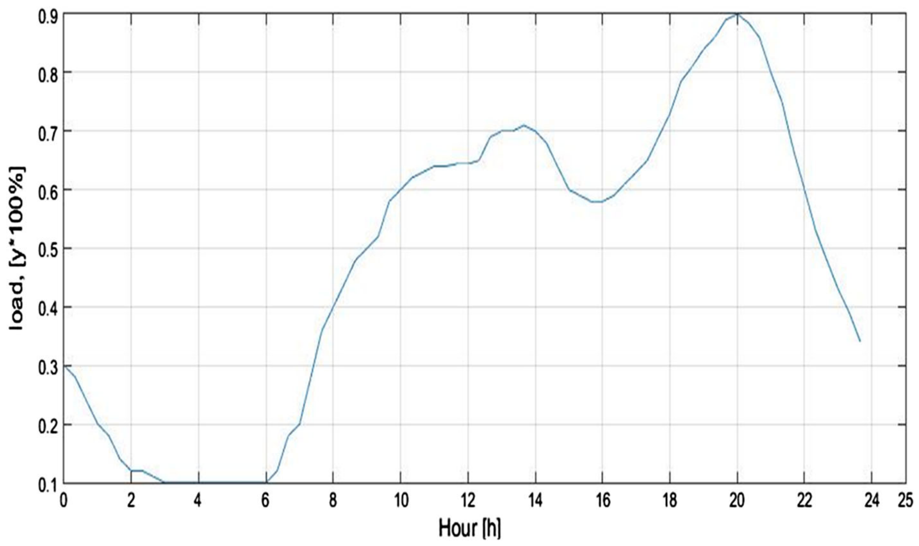


Fig. 2 Cell load pattern through a day

$$P_{MBS} = N_{sector} * N_{PApSec} * \left(\frac{P_{TX}}{\mu_{PA}} + P_{SP}\right) * (1 + C_c) * (1 + C_{PSBB}) \tag{6}$$

$$P_{mBS} = N_{sector} * N_{PApSec} * N_L * \left(\frac{P_{TX}}{\mu_{PA}}(1 - C_{TX,static}) * C_{TX,NL} + P_{SP,NL}\right) * (1 + C_{PS}), \tag{7}$$

where P_{MBS} and P_{mBS} are the power consumptions of a MBS and mBS respectively, N_{sector} is the number of sectors, N_{PApSec} is the number of power amplifiers per sector, P_{TX} is the transmit power, μ_{PA} is the power efficiency, P_{SP} is the signal processing overhead, C_c is the cooling loss, C_{PSBB} is the battery backup and power supply loss, N_L is the number of active links, $C_{TX,static}$ is the static transmit power, $C_{TX,NL}$ is the dynamic transmit power per link, $P_{SP,NL}$ is the dynamic power consumption caused by signaling processing per link, C_{PS} power supply loss.

3 Machine Learning Scheme

3.1 Motivation

Every communication network is generating signaling and statistical data and events, which are used from the OAM specialists for monitoring, troubleshooting and management of the network. Nowadays, when a problem in the network occurs, before taking any action, the necessary specific data from the network is analyzed based on the OAM engineers' knowledge and experience. As the amount of data increases exponentially with the increase in the complexity of communication technologies it becomes impossible for man to processes and analyze such data correctly. This is why ML schemes become an inherent tool for the development of autonomic processes in a communication network and to path the way towards the creation of a fully autonomous network. The logic of this autonomy will be placed or stored on a separate plane, called "*cognitive plane*". This is the plane which will interact and control different communication technologies and devices from different vendors.

3.2 Cognitive Plane and Machine Learning Scheme

For the development of a fully autonomic network we propose a novel cognitive plane and ML scheme based on abstraction, modularity and hierarchy, as shown in Figs. 3 and 4. Abstraction and modularity are well known concepts and could be seen in programming, in fabric production lines, in protocol stacks and etc. In Fig. 3 the cognitive plane is shown with its major functional blocks. Its main responsibilities are related to the simulation of human activities by monitoring and processing signaling data from the network plane and taking decisions on behalf of the OAM specialists or assist the OAM engineers to maintain the network by doing or processing repetitive events. The Cognitive plane will be developed separately from the network plane, which includes network devices, network protocols and etc. By implementing abstraction with the "standard based data and configuration model" block we can reuse the cognitive functions independently from the operating devices and used technologies.

We propose modularity and decision hierarchy also for the cognition block, as shown in Fig. 4. Every layer has its own and unique processes and can interact with the adjacent layers. The top layer will take high level decisions based on the created goals and policies. They are stored in the policy block (Fig. 3). The network itself cannot decide which are the tasks and goals that have to be pursued. That is why the tasks, goals and policies are developed by the mobile operator. For every problem a decision process will be created

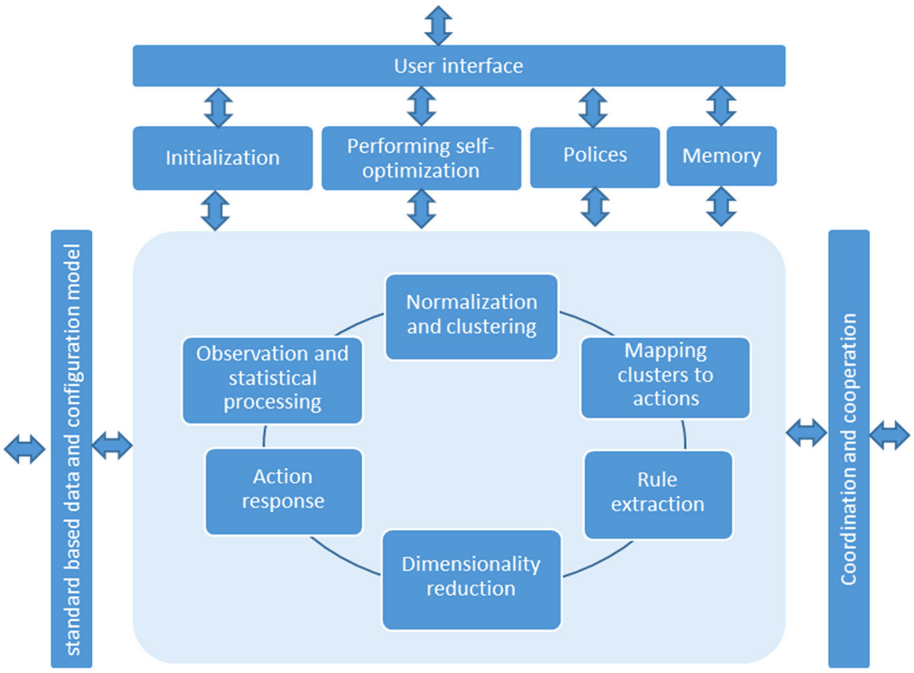

User interface
Initialization
Performing self-optimization
Polices
Memory
Normalization and clustering
Observation and statistical processing
Mapping clusters to actions
Action response
Rule extraction
Dimensionality reduction
standard based data and configuration model
Coordination and cooperation

Fig. 3 Abstracted cognitive plane

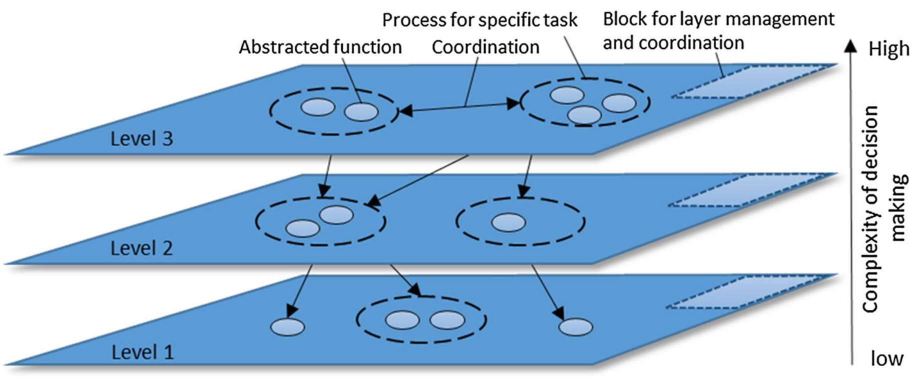
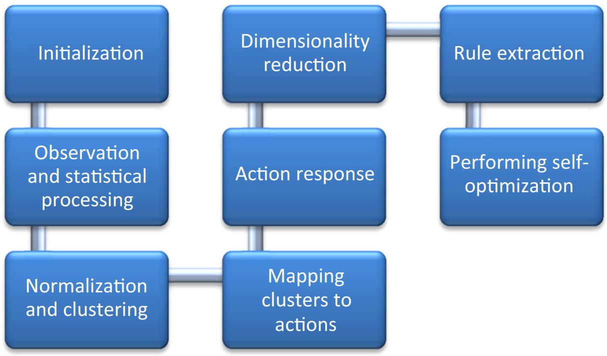

Process for specific task
Block for layer management and coordination
Abstracted function
Coordination
High
Level 3
Level 2
Level 1
Complexity of decision making
low

Fig. 4 Modular and hierarchical cognition

using one or more functional blocks from the given layer. Decisions are getting less complex going from the top layer down to the lower ones. For taking a decision, a given layer can use the functionalities from the layer below. For example, the lowest layer can have the functionality to gather data from the sensors in a network and the layer above has to decide based on the gathered information how to represent the data for further analysis in the next higher layer. Decisions from a top layer will be translated or divided into less complicated tasks or goals for the layer below. A layer will create decision processes depending on the tasks/goals given from the top layer. A decision process from a layer can use one or many decision processes from the layer below.

In Fig. 5 the proposed ML scheme and the creation of the basic cognition flow, which resides in the "*cognition*" block are illustrated. It is divided into many individual functional steps, which are modular and abstracted from one another. Some of the steps in the proposed scheme can be skipped. For example, the "***dimensionality reduction***" step can be removed if in the "***initialization***" step are picked only those features which are important for the given task. More or less the cognition functionality in a network is related to the way how humans are taking decisions. A man does not have a particular algorithm for every task, but tools like ML algorithms, which are used and shaped while searching for a solution, do have. For example when a man is learning how to ride a bike or drive a car, he has at the beginning a statistical information, which will help him to create some states. Then he will use supervised or unsupervised learning methods to find the best action for

Fig. 5 Proposed machine learning flow


Initialization
Dimensionality reduction
Rule extraction
Observation and statistical processing
Action response
Performing self-optimization
Normalization and clustering
Mapping clusters to actions

every state depending on the situation (learning with someone or by himself). This human learning behavior can be described as an abstracted program shaped by the data gathered from the environment by the man. So, our cognition flow will mimic this kind of behavior. We will create the flow as a program, which will be shaped by the signaling data in the network.

Having this kind of ML scheme will give us the following advantages:

- *Decreased complexity* One complex optimization problem or task could be solved by many small and simple ML functional steps. In some cases it is impossible to create an optimization task, because a full knowledge of the problem is required, such as parameter dependencies, ranges and etc. Even assumptions are made to simplify the problem solving scenario. The end result is a complex algorithm difficult to understand with high level of determinism and low scalability. By splitting the problem into smaller optimization tasks and solving each one of them by ML algorithms we acquire a solution, which is easy to follow, understand, shaped and changed by the data generated from the network;
- *Reusability* Usually complex problems have similarities. If these similarities are solved with ML, we can reuse the algorithm for solving another problem;
- *Flexibility and scalability* Cognitive algorithms must be scalable and flexible. Having functional steps with abstracted ML algorithms it will be easy to manipulate the cognitive flow in time.

The cognition flow steps and blocks from the abstracted cognitive plane are as follows:

1. "***Initialization***" Here the input features, the actions, the constraints are defined and initiated. The target parameters, which need to be optimized are set as input;
2. "***Observation and statistical processing***" Actual learning data is gathered from the network (signaling information, key performance indicators and etc.). Useful information is extracted from the data for the learning process;
3. "***Normalization and clustering***" In order the features to have an equal impact to the decision making and similar value ranges, normalization is used. Through grouping the samples with rather equal values into clusters a minimization of the number of the states in which the network can reside is achieved. The clusters determine the states, in which the network can be;
4. "***Mapping clusters to actions***" (i.e., the 'learning process') An action is mapped to a cluster, which maximizes the value of the target parameter or parameters, which have to be optimized without violating the constraints;
5. "***Action response***" This step predicts the potential future state of the network, if a particular action will be taken. This is done by gathering the input and output states for every action during learning. The transitions are stored and an interpolation is used for every feature to find the transition surface. This transition surface will show how the values of the features will change, while taking a particular action;
6. "***Dimensionality reduction***" Features, which do not contribute to the decision making are discarded;
7. "***Rule extraction***" After the learning process is completed, a decision boundary is extracted, which can be exchanged/shared between agents to speed up the learning process;
8. "***Performing self-optimization***" Based on the information obtained from the previous steps, a simple algorithm is derived for the optimization of the target parameter;

9. "***User interface***" This is an interface, which allows interaction with the autonomous learning model;
10. "***Polices***" High level abstracted and non-engineering rules are created to control the behavior of the AML model;
11. "***Memory***" All knowledge and gathered data are stored there;
12. "***Standard base data and configuration model***" The data, which is coming from a network device is standardized and abstracted;
13. "***Coordination and cooperation***" In a multi user scenario data has to be shared and actions among the users need to be coordinated.

3.3 Machine Learning Flow for Energy Saving

The goal of energy saving is to minimize the total power consumption in the network, thus decreasing the CO_2 emissions. In "Energy Saving" mode the algorithm finds the optimum time periods in which cells with low cell loads can offload their traffic to the neighboring cells. Measurements such as cell load (CL), reference signal received power (RSRP), measurement reporting events (MRE) in a mobile terminal and etc. are used to create the state spaces. Key performance indicators (KPI) such as handover failure rate (HFR), handovers per call (Ping Pong Handover), The radio link failure ratio (RLFR) will assess the effectiveness of the action taken in a particular state.

3.3.1 Simulated Energy Saving Scenario

In the proposed scenario every macro cell acts as an agent and is controlling many micro/pico cells. The cell load pattern is given in Fig. 2 and will fluctuate every day. A particular agent has to decide when and which of the micro/pico cells is justified to be turned off or on. The "on/off" pattern must be such, so that the consumed energy is less, but without handover failures and/or dropped traffic. First, for each cell under its control, the agent has to find out the better state ("off" or "on") and then to use this information for the multi-cell scenario.

Let the vector $x = [x_1, x_2, t]$ have three features. The load of cell 1 is represented by the feature x_1, x_2 represents load of cell 2 and t is the time. For simulation purposes samples of the load of cell 1 and 2 will be taken with a sampling step of 20 min. For the sake of simplicity we will assume that cells 1 and 2 have the same capacity. The cell load value is normalized so that the value changes from 0 to 1.

The energy saving optimization problem can be defined as follows:

$$MIN(P_{total})\ subject\ to\ \boldsymbol{action\,space}, \boldsymbol{KPI\,thresholds}, \qquad (8)$$

where P_{total} is the total power consumption in the network, ***action space*** are all possible actions for the network, which minimize the total power consumption, ***KPI thresholds*** are the network performance indicators thresholds requirements, which must be met while minimizing the power consumption.

The constraints in the energy saving scenario, are to deactivate cells without increasing the handover failure rate or overloading the recipient cell. Let cell 1 is the recipient cell and cell 2 is the donor cell, which has to be turned off. Cell 1 will command cell 2, if it is needed to be turned off or not. The possible actions will be: [***turn off cell 2***; ***turn on cell 2***]. It is obvious that when the total load of cell 1 and cell 2 is higher than one ($x_1 + x_2 > 1$)

we will have call failures or call drops, but this information will not be given to the recipient cell. The latter has to learn it.

Instead creating a complex solution for this optimization problem, we will use some of the functional steps from the proposed ML scheme. Every step from the cognition flow has its own task so that at the end of the flow in the "optimization" step we can write a simple algorithm in the form of:

$$\begin{gathered}\text{Pick best action}\,\boldsymbol{A}\,\text{for state}\,\boldsymbol{N}\,\text{when in time period}\,\boldsymbol{T}\\ \text{while}\,\boldsymbol{constraint}_1 < \text{value},\ \boldsymbol{constraints}_2 < \text{value}\end{gathered} \tag{9}$$

Action ***A*** is found in step 4, state ***N*** is created in step 3, time period ***T*** is measured in step 2 and 3 and ***constraints*** are set in step 1.

The steps for achieving energy saving are explained as follows:

Step 1 Initialization

Here we define the possible actions for cell 1, constraints, input features and the target parameter for optimization, which in this case is the total power consumption.

Step 2 Observation and statistical processing

If a mobile network is not reconfigured, it is expected all the functional data that is generated to have a repetitive pattern with periodicity of one or up to several days. This means that if we observe (or monitor) the network continuously for a number of consecutive days and gather enough long-term statistical data we could determine a time depending function for every feature. For this we apply a Fourier regression model and the function representation is (10):

$$f(t) = a_0 + a_1 \cdot \cos(t \cdot w) + b_1 \cdot \sin(t \cdot w) + \cdots + a_p \cdot \cos(p \cdot t \cdot w) + b_p \cdot \sin(p \cdot t \cdot w), \tag{10}$$

where t is time, w is the sampling rate, a_p and b_p coefficients, $\mathrm{p} \in \{0\} \cup \mathrm{N}$.

Step 3 Clustering

The result from the previous step is the collection of a large number of samples in time. One efficient way to process such data samples is to group them into clusters, which will represent a particular state in which we can take a given action. The most promising unsupervised algorithm for this kind of a scenario is the K-means algorithm [24]. The application of this technique gives us groups or clusters with samples, which have similar characteristics. The K-means algorithm divides a set of **M** samples into **K** disjoint clusters **C**, each described by the mean μ of the samples in the cluster. The means are commonly called the cluster "centroids". In general, they are not points from x, although they live in the same space. The K-means algorithm aims to choose centroids that minimize the inertia, or within-cluster sum of squared criterion:

$$\sum_{(i=0)}^{n} \min\left(\left\|x^i - \mu_k\right\|^2\right), \tag{11}$$

where μ_k is mean value in cluster k, x^i data sample, $\mathrm{n} \in \{0\} \cup \mathrm{N}$.

To find the optimal number of clusters we need to compare the variance of the clusters for a different size of **K**. The following measure represents the sum of intra-cluster distances between the points in a given cluster C_k, containing n_k points:

$$D_k = 2n_k \sum_{x_i \in C_k} x^i - \mu_k^2. \tag{12}$$

Adding the normalized intra-cluster sums of squares gives a measure of the compactness of the clustering:

$$W_k = \sum_{k=1}^{K} \frac{1}{2n_k} D_k. \tag{13}$$

This variance quantity W_{k} (or weight) is the basis of a procedure used to determine the optimal number of clusters: the elbow method [25]. Initially the clusters will add more information with a high variance (distance between data point and the centroid inside a cluster), but with increasing the number of clusters, at some point the marginal gain will drop, giving an angle in the graph. The number of clusters are chosen at this point, hence the "elbow criterion" applied. The number **K** can be automatically determined using the following rule:

$$\text{if} \quad W_k - W_{k+1} < \mathit{threshold} \quad \text{then clusters} = \mathrm{K}. \tag{14}$$

If we reduce our dimensionality from 3 to 2 by discarding the time feature we will have all the samples distributed on the $[x_1, x_2]$ plane. A representation of the samples as a result from a simulation of the system model described in point II, after the "***observation and statistical processing***", and the "***normalization and clustering***" steps, is shown in Fig. 6.

Step 4 Mapping clusters to actions

For the mapping of actions to states simple reinforcement learning (RL) is used. A RL agent interacts with its environment in discrete time steps. At each time t, the agent receives an observation O_t, which typically includes the reward rt. It then chooses an action from the set of actions available, which is subsequently sent to the environment. The environment moves to a new state s_{t+1} and the reward r_{t+1} associated with the transition (s_t, a_t, s_{t+1}) is determined. The best action for a given state will be the action giving maximum reward. As we have introduced some constraints in the form of thresholds, the best action has to comply with them. The reward will be the value of the parameter that must be optimized. In the case of energy saving the reward will be the amount of energy saved for the period in which a given action is taken. The update for the action value function is defined by Eq. (6) [26]:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \delta\left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)\right], \tag{15}$$

where $Q(s_t, a_t)$ is the last updated action value function; r_{t+1} is the reward for taking an action a_t in a state s_t; $\max_{\mathrm{a}} \mathrm{Q}(\mathrm{s}_{t+1}, \mathrm{a})$ is the maximum action value, which can be assumed in a state s_{t+1}; δ is the update rate and γ is the discount rate.

Step 5 Action response

If we want to predict what will be the next state after taking an action, the Nearest-neighbor interpolation algorithm can be used [27]. This step will be needed for a multi cell energy saving scenario when the dependencies between the actions have to be discarded.

In a multi cell scenario the action will be multidimensional in the form of vector $\mathrm{A}[a_2, a_3, \ldots, a_n]$, where a_n is the action taken for cell n. It must be noted that there is no action for

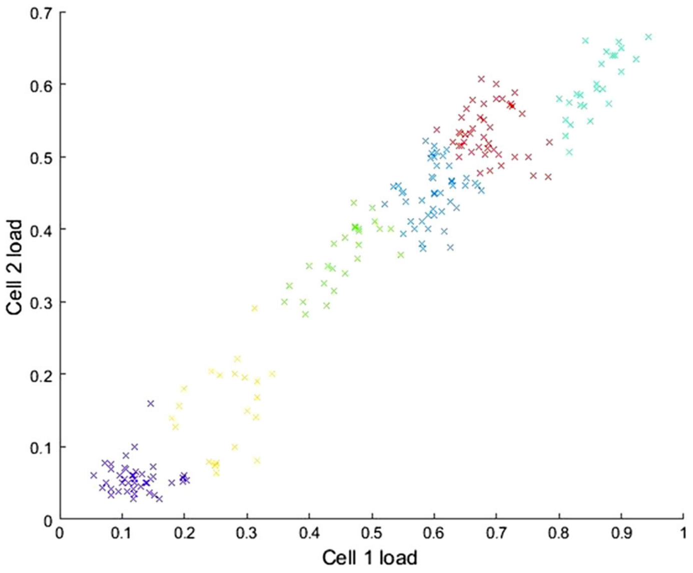


Fig. 6 Sample distribution and clustering

cell 1, because the cell acts as a decision maker for the rest of the cells. For two possible actions for a given cell [‘*turned on*’, ‘*turned off*’] and a number of N cells the possible variations of the action vector will be 2N − 1. Every action for a cell is dependent from the actions taken for the rest of the cells. It will be inefficient to try every possible variation of the action vector in a given state. That is why this dependency from the decision process must be discarded. This is done through the implementation of feature prediction for a particular action and a target cell. Analyzing the data before and after taking every single action, a response function is created which is used for future decisions. For a four cell scenario we have a three dimensional action vector. Also we have six response functions divided into three groups, one for every target cell. In the response function groups we have the action response for the feature ‘*load of cell 1*’ when action ‘*turn on*’ or ‘*turn off*’ is taken only for the target cell. For example the response function $\boldsymbol{F_{cell1\ load}}$[action − ‘*turn off*’, state {$\boldsymbol{load_{cell1}}$, $\boldsymbol{load_{cell2}}$}] shows how the load of cell 1 will change after taking action ‘*turn off*’ for cell 2 in the given state with features $\boldsymbol{load_{cell1}}$ and $\boldsymbol{load_{cell2}}$. This is illustrated on Fig. 9.

First the algorithm will decide the action for one target cell. Then it will determine the potential next state from the action response function for this cell with the decided action, which will be used for the decision of the next action with the next target cell in queue. After the second decision the next potential state will be calculated and so on. In this case a decision-prediction chain is created. This chain will stop, when a prediction state enters an area, where the constraints are not met. In this way instead trying 2^{N-1} possible actions for every state, we can create $2 \cdot (N - 1)$ action response functions. If the learning rate is 1 day per cell, with action response functions it will take only $2 \cdot (N - 1)$ days to complete the learning stage in multi cell scenario. Because the action response nature for a given feature is the same among all target cells we can minimize the learning period even more by combining action response data from all cells into one.

In multivariate nearest-neighbor interpolation, the algorithm assigns to some point P(x_1, x_2, x_3, …, x_n) the value of the closest observed/given data point $P^*(x_{1*}, x_{2*}, \ldots, x_{n*})$ to P, which minimizes the objective function:

$$\delta\left(x_i^1, x_i^2, \ldots, x_i^n\right) := \sqrt{\left(x_1 - x_i^1\right)^2 + \left(x_2 - x_i^2\right)^2 + \ldots + \left(x_n - x_i^n\right)^2} \tag{16}$$

for all i = 1, 2, …, N, where (x_i^1, x_i^2, …, x_i^n) are the given data points.

Step 6 Rule extraction

When the mapping is ready, we can extract some rule/boundary how the classes/actions are separated. For multidimensional input space the decision boundary will be a hyper plane. For the learning classification (linear boundary) a logistic regression or a support vector machine learning (SVM) with linear kernel can be used. For non-linear classification (non-linear boundary) a SVM with Gaussian kernel can be used [28]. Rule extraction will be useful also, when we want to share the knowledge among cells, which are responsible for taking actions.

For the simulated scenario, the mapping of clusters into actions and the rule extraction step is illustrated in Fig. 8. In green are the samples in which the action '*turn off*' is taken for a given cell, while in red are the samples, in which the action "turn on" is chosen to be better.

Step 7 Optimization

The flow is in learning stage from step 1 through step 6. When the learning stage is done, in step 7 the learned features from the previous steps are exploited in the form of simple commands as shown in (9). In a multi cell scenario the algorithm needs to decide which cells are more efficient to be '*turned off*' or '*turned on*'. The decision is based on the calculation of the reward over action cost efficiency ratio. Reward is the saved energy and action cost is the reciprocal value of the distance between the present/predicted state and the decision boundary for the cell agent and target cell. If a state is near to the decision boundary (distance is low, action cost is high), there is a higher chance for the next state to be in the zone where the next target cell cannot be turned off. The cells with higher efficiency ratio will be turned off first.

4 Simulation Results

Following, we present not only the end results, but illustrate all intermediate steps from the machine learning flow.

In Fig. 6 are shown the statistics gathered for 3 days. Also the '***normalization***' and '***clustering***' steps are applied. Because cell traffic has periodicity with 1 day we can see that the cell load samples have a distribution pattern. This information can be extracted and used for learning purposes. Also we can see the optimum number of clusters, which later will be used for the '***mapping actin to states***'.

In Fig. 7 the variation of the cell load and the transition of the clusters in time are shown. Again through statistics we can extract useful information about the clusters or states transitions in time and how long a cell or cells remains in a particular state.

The outcomes from '***mapping actions to states***' and '***rule extraction***' are illustrated in Fig. 8. If the decision boundary falls within a cluster, the latter will be split during the learning process. There is little difference between the actual and learned boundary, but this gap is expected to diminish in time as a result of the learning process. In this simulation the learning process has a duration of 3 days. If the learning process was longer, the cluster would be split into two parts. The part, which is between actual boundary and learned boundary on Fig. 8 will be classified in the other region.

In Fig. 9 the action response function is shown. It can be seen how the load of cell 1 will change after the action "*turn off*" is applied for cell 2. It is visible that for an area where the data distribution is compacted, the prediction is more accurate than for an area where no data was observed. In these areas the error prediction from the actual value can be significant. This error can be minimized with more gathered data in step 2 and with the increase of the learning time, with the use of a better algorithm for multivariate interpolation. As mentioned above, the cell load is normalized to the maximum load that a cell can have in the network.

From Fig. 10 the load of cell 1 during learning (day 1) and after learning (day 2) could be seen. It is illustrated what the actual load of cell 1 will be without shutting down cell 2 and what is the sum of the loads of the two cells. In Fig. 11 the normalized power consumption of cell 2 and call drops values are shown. There is a clear difference between the learning stage and the exploitation stage. In the learning stage we have three periods where a wrong decision was taken, where cell 2 was shut down and cell 1 did not have the capacity to receive the load of cell 2. The result is that the number of dropped calls exceeded the predefined normalized threshold. In the exploitation stage the agent (cell 1) already learned that in these periods of time the state will be in the zone where action "turn on" is better to be applied. After taking the correct decision higher values of call drops were not perceived.

In Fig. 12 the total load of cell 1 is shown in a simulated multi-cell energy saving scenario. Around 1 AM cell 1 has enough free capacity and the algorithm, embedded in cell 1, decides to switch off a cell, thus the total load of cell 1 starts to increase. Half an hour later another cell is switched off. After 6 h as a result of the increased overall traffic, some cells are turned on, thus creating a saw-edged look of the total load of cell 1.

For the energy saving optimization problem there are different approaches as the ones proposed in [29–33]. Taking into account their simulation setup, it could be concluded that the results are similar to the above. By making a general comparative analysis between the

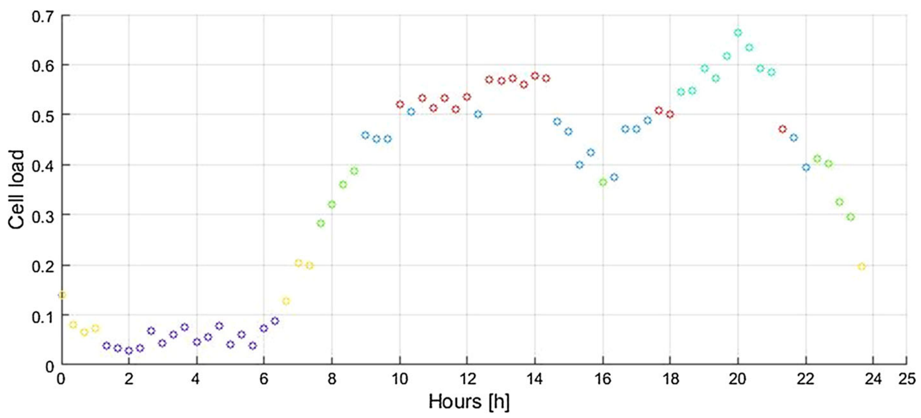


Fig. 7 Cluster transitions in time

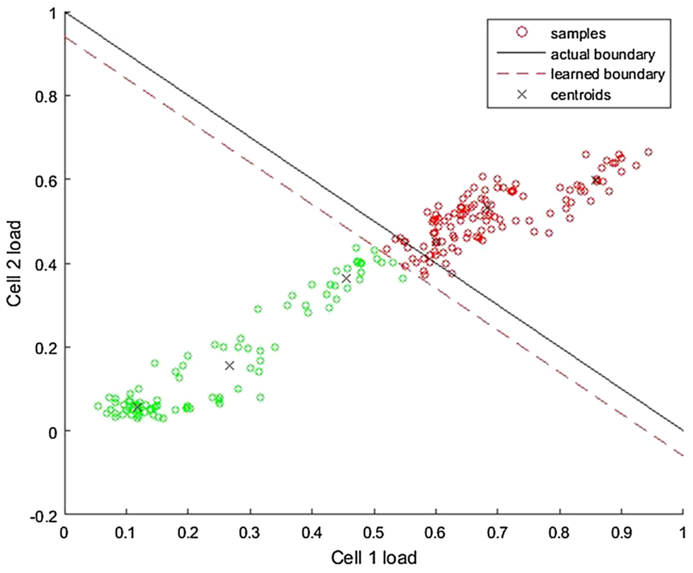


Fig. 8 Learned boundary

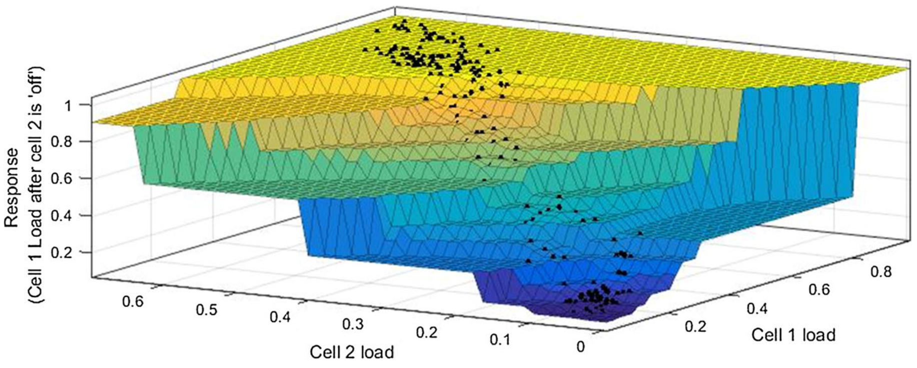


Fig. 9 Action response—prediction of cell 1 load after shutting down cell 2

proposed machine learning scheme and other algorithms for energy saving the following main differences can be observed:

- *Scalability* Some of the papers are having too much assumptions, which will lead to difficulties in large a scale deployment scenario;
- *Complex algorithms* It seems that many of the solutions are very complex for this kind of a problem.
- *'Fail protection'/validation* Some of the algorithms do not propose protective mechanisms with KPIs, which will validate the efficiency of the taken action for turning 'off' or 'on' a cell;
- *Statistical data* Only small number of papers are using the collection of statistical data for their algorithms.

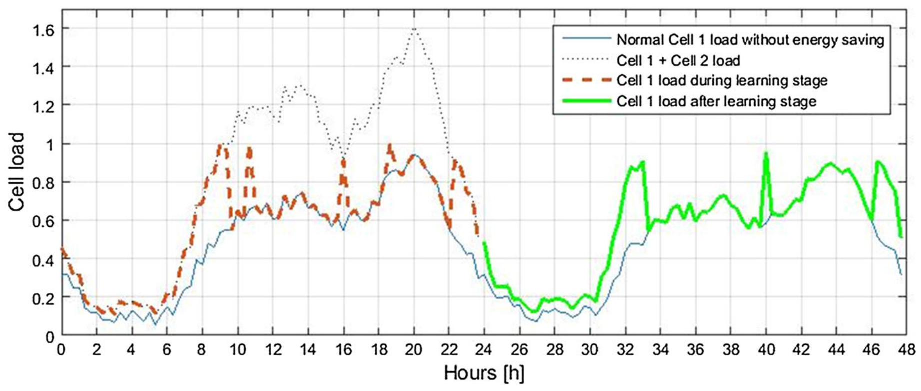


Fig. 10 Cell load of the recipient cell 1 during and after learning

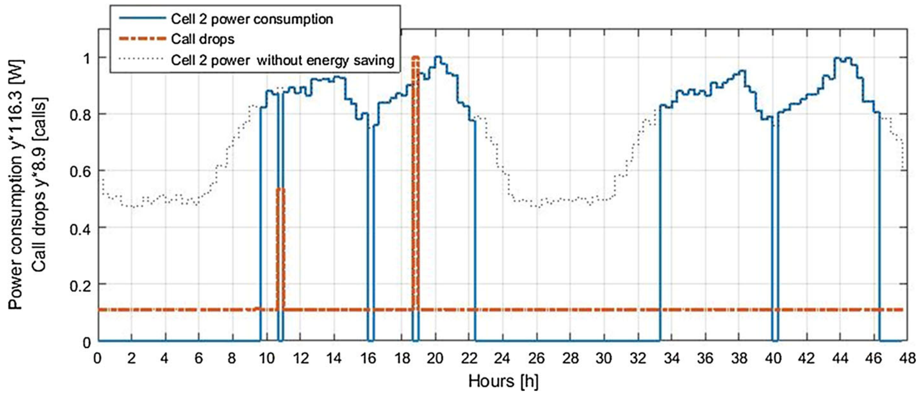


Fig. 11 Normalized power consumption of cell 2 and call drops values

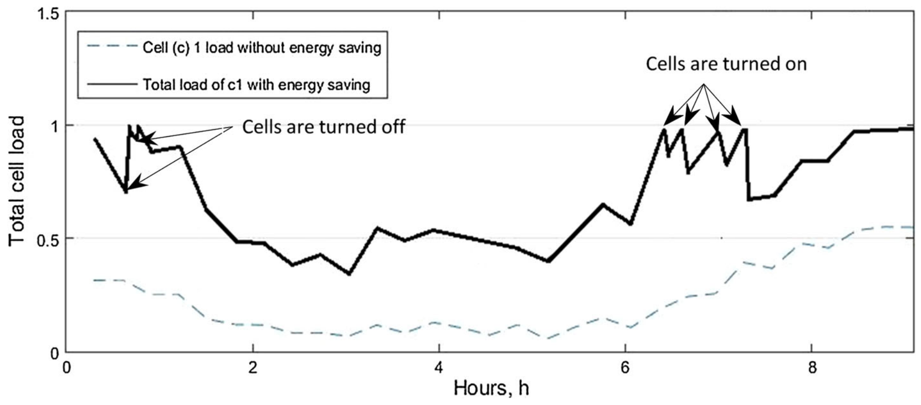


Fig. 12 Multi cell energy saving scenario

5 Conclusion and Future Work

In this paper a novel energy saving approach for self-organized HetNets is presented. By splitting an complex optimization problem into several modules, each one representing a more simple optimization task and the application of a ML scheme in which the principles of abstraction an modularity are incorporated, the advantages of this approach are outlined. They are mainly related to the lower complexity of the final definition of the optimization task and the flexibility and adaptation in the ML process, the latter based on the data generated and gathered from the heterogeneous network. Based on this, an algorithm is developed and simulated for the optimization of the energy efficiency in a HetNet. The results from the simulation experiments show that such an approach could be an appropriate solution for developing a full self-managing future network. One of the disadvantages of the proposed approach is the need for coordination between different autonomic processes during and after the learning process and in some cases the necessity of a longer time of the learning stage. They could be minimized, if the extracted knowledge is shared among the agents and a simple coordinating scheme is implemented. In this relation the future work will be related to the implementation of a simple coordination scheme among the different autonomic processes and to introduce some priorities, which reflect the mobile operator's needs. Also we will define and test a more simplified event driven ML scheme for problems which have a high computational complexity and cannot be easily solved with numerical methods.

References

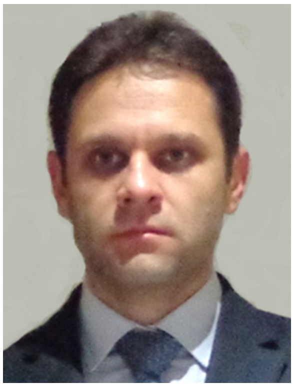
1. Hämäläinen, S., Sanneck, H., & Sartori, C. (2012). *LTE self-organising networks (SON)*. New York: Wiley.
2. Semov, P., Koleva, P., Tonchev, K., Poulkov, V., & Mihovska, A. (2016). Autonomous learning model for achieving multi cell load balancing capabilities in HetNet. In *Proceedngs of the IEEE BlackSeaCom 2016 Conference*. Varna, Bulgaria.
3. Altman, Z., Sallem, S., Nasri, R., Sayrac, B., & Clerc, M. (2014). Particle swarm optimization for mobility load balancing SON in LTE Networks. In *Wireless Communications and Networking Conference Workshops (WCNCW)* (pp. 172–177). IEEE. doi:10.1109/WCNCW.2014.6934881.
4. Morales-Tirado, L., Suris-Pietri, J. E., & Reed, J. H. (2009). A hybrid cognitive engine for improving coverage in 3G wireless networks. In *Communications Workshops, 2009. ICC Workshops 2009* (pp. 1–5). doi:10.1109/ICCW.2009.5208034.
5. Wang, Y., & Zhang, K. (2011). Decision tree based unsupervised learning to network selection in heterogeneous wireless networks. In *Consumer communications and networking conference (CCNC)* (pp. 1108–1109). IEEE. doi:10.1109/CCNC.2011.5766340.
6. Lee, C.-L., Su, W.-S., Tang, K.-A., & Chao, W.-I. (2014). Design of handover self-optimization using big data analytics. In *Network operations and management symposium (APNOMS), 2014 16th Asia-Pacific* (pp. 1–5). doi:10.1109/APNOMS.2014.6996546.
7. Kon, Y., Ito, M., Hassel, N., Hasegawa, M., Ishizu, K., & Harada, H. (2012). Autonomous parameter optimization of a heterogeneous wireless network aggregation system using machine learning algorithms. In *Consumer communications and networking conference (CCNC)* (pp. 894–898). IEEE. doi:10.1109/CCNC.2012.6181186.
8. Ul Islam, M. N., & Mitschele-Thiel, A. (2012). Reinforcement learning strategies for self-organized coverage and capacity optimization. In *Wireless Communications and Networking Conference (WCNC)* (pp. 2818–2823). IEEE. doi:10.1109/WCNC.2012.6214281.
9. Thampi, A., Kaleshi, D., Randall, P., Featherstone, W., & Armour, S. (2012). A sparse sampling algorithm for self-optimisation of coverage in LTE networks. In *Wireless communication systems (ISWCS)* (pp. 909–913). doi:10.1109/ISWCS.2012.6328500.

10. Razavi, R., & Claussen, H. (2013). Improved fuzzy reinforcement learning for self-optimization of heterogeneous wireless networks. In *International conference on telecommunications (ICT)* (pp. 1–5). doi:10.1109/ICTEL.2013.6632073.
11. Rouzbeh, R., Siegfried, K., & Holger, C. (2010). A fuzzy reinforcement learning approach for self-optimization of coverage in LTE networks. *Bell Labs Technical Journal, 15*(3), 153–175. doi:10.1002/bltj.20463.
12. Semov, P. T., Poulkov, V., Mihovska, A., & Prasad, R. (2014). Increasing throughput and fairness for users in heterogeneous semi coordinated deployments. In *Wireless communications and networking conference workshops (WCNCW)* (pp. 40–45). IEEE. doi:10.1109/WCNCW.2014.6934858.
13. Baltiiski, P., Iliev, I., Kehaiov, B., Poulkov, V., & Cooklev, T. (2015). Long-term spectrum monitoring with big data analysis and machine learning for cloud-based radio access networks. *Wireless Personal Communications*. doi:10.1007/s11277-015-2631-8.
14. Thilina, K. M., Choi, K. W., Saquib, N., & Hossain, E. (2013). Machine learning techniques for cooperative spectrum sensing in cognitive radio networks. *IEEE Journal on Selected Areas in Communications, 31*(11), 2209–2221. doi:10.1109/JSAC.2013.131120.
15. Wu, R., Wen, Z., Fan, C., Liu, J., & Ma, Z. (2010). Self-optimization of antenna configuration in LTE-advance networks for energy saving. In *3rd IEEE international conference on broadband network and multimedia technology (IC-BNMT)* (pp. 529–534). doi:10.1109/ICBNMT.2010.5705146.
16. Mwanje Stephen S., & Mitschele-Thiel A. (2013). A Q-learning strategy for LTE mobility load balancing. In *24th international symposium on personal indoor and mobile radio communications (PIMRC)* (pp. 2154–2158). IEEE. doi:10.1109/PIMRC.2013.6666500.
17. Mwanje, S. S., & Mitschele-Thiel A. (2014). Distributed cooperative Q-learning for mobility-sensitive handover optimization in LTE SON. In *Symposium on computers and communication (ISCC)* (pp. 1–6). IEEE. doi:10.1109/ISCC.2014.6912619.
18. Eduardo, S., Mihovska, A., Rodrigues, A., Prasad, N., & Prasad, R. (2013). Cell load balancing in heterogeneous scenarios. In *Proceedings of GWS 2013*, June 24–28, 2013, Atlantic City, NJ.
19. Monteiro, N., Mihovska, A., Rodrigues, A., Prasad, N., & Prasad, R. (2013). Interference analysis in a LTE-A HetNet scenario: Coordination vs. uncoordination. In *Proceedings of GWS 2013*, June 24–28, 2013, Atlantic City, NJ.
20. GPP. Evolved Universal Terrestrial Radio Access (E-UTRA); Further advancements for E-UTRA physical layer aspects; (Release 9). *TR 36.814, 3rd Generation Partnership Project (3GPP)*, March 2010.
21. Li, J., Bose, A., & Zhao, Y. Q. (2005). Rayleigh at fading channels' capacity. In *Proceedings of the 3rd annual communication networks and services research conference* (pp. 214–217).
22. Oliver, B., Anton, A. Michael, W., & Ulrich, B. (2013). Energy efficiency of LTE networks under traffic loads of 2020. In *Proceedings of the tenth international symposium on wireless communication systems (ISWCS 2013)* (pp. 1–5).
23. Arnold, O., Richter, F., Fettweis, G., & Blume, O. (2010). Power consumption modeling of different base station types in heterogeneous cellular networks. In *Future Network and Mobile Summit* (pp. 1–8).
24. Zaki, M. J., & Meirajr, W. (2014). *Data mining and analysis*. Cambridge: Cambridge University Press.
25. Ketchen, D. J., Jr., & Shook, C. L. (1996). The application of cluster analysis in strategic management research: An analysis and critique. *Strategic Management Journal, 17*(6), 441–458. doi:10.1002/(SICI)1097-0266(199606)17:6<441::AID-SMJ819>3.0.CO;2-G.
26. Sutton, Richard S., & Barto, Andrew. (1998). *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press.
27. Surhone, L. M., Timpledon, M. T., & Marseken, S. F. (2010). *Nearest—Neighbor interpolation*. VDM Publishing. ISBN 9786131015762.
28. Suykens, J. A. K. (2001). Nonlinear modelling and support vector machines. In *IMTC 2001* (Vol. 1, pp. 287–294). doi:10.1109/IMTC.2001.928828.
29. Bousia, A., Kartsakli, E., Alonso, L., & Verikoukis, C. (2012). Energy efficient base station maximization switch off scheme for LTE-advanced. In *2012 IEEE 17th international workshop on computer aided modeling and design of communication links and networks (CAMAD)*. doi:10.1109/CAMAD.2012.6335345.
30. Roth-Mandutz, E., & Mıtschele-Thıel, A. (2013). LTE energy saving SON using fingerprinting for identification of cells to be activated. In *Future Network and Mobile Summit (FutureNetworkSummit)*. ISBN: 978-1-905824-37-3.
31. Hiltunen, K. (2013). Improving the energy-efficiency of dense LTE networks by adaptive activation of cells. In *2013 IEEE international conference on communications workshops (ICC)*. doi:10.1109/ICCW.2013.6649410.

32. Hiltunen, K. (2013). Utilizing eNodeB sleep mode to improve the energy-efficiency of dense LTE networks. In *2013 IEEE 24th annual international symposium on personal, indoor, and mobile radio communications (PIMRC)*. doi:10.1109/PIMRC.2013.6666707.
33. Ambrosy, A., Blume, O., Ferling, D., Jueschke, P., Wilhelm, M., & Yu, X. (2014). Energy SAVINGS in LTE macro base stations. In *Wireless and mobile networking conference (WMNC), 2014 7th IFIP*. doi:10.1109/WMNC.2014.6878872.

Plamen T. Semov has received his two M.Sc. degrees at the Technical University of Sofia and at the Alborg University. He has more than 3 years of work experience in leading mobile operators in Bulgaria in the field of fixed access network planning, radio planning and optimization of radio access network and base station design. He also has more than 1 year of expertise in network security operation. He has publications related to the development of algorithms for self-organization, power control and energy efficiency in heterogeneous networks. His major fields of scientific interest and expertise are related to machine learning, self-organizing networks, resource management in next generation networks. Currently he is a Ph.D. student at the Technical University of Sofia.

Hussein Al-Shatri received the B.Sc. degree in electronic and communications engineering from Hadhramout University, Yemen, the M.Sc. degree in communications engineering from Munich University of Technology, Germany, and the Ph.D. degree in electrical engineering from the University of Rostock, Germany, in 2003, 2008, and 2014, respectively. Between 2009 and 2014, he was assistant researcher with the Institute of Communications Engineering, University of Rostock. During that time, he was active in the topics of power allocation and interference alignment. Since August 2014, he is a Postdoctoral Researcher with Communications Engineering Laboratory, Technische Universität Darmstadt, Germany. His research interests include hierarchical signal processing, cloud radio access networks, distributed algorithms design, and user preferences analysis & integration in underlay wireless networks.



Krasimir Tonchev graduated M.Sc. in Telecommunications at the Technical University of Sofia, Bulgaria, in 2009. He is currently working on his second M.Sc. in Applied Mathematics and on his Ph.D. work in Computer Vision. He is also leading researcher at the Teleinfrastructure R&D laboratory at the Technical University of Sofia. His primary research interests include machine learning, computer vision, general data analysis and processing. He has published multiple papers on these topics. As an engineer, he has realized multiple projects, including image processing and computer vision systems deployed across the world.

Vladimir Poulkov has received his M.Sc. and Ph.D. degrees at the Technical University of Sofia. He has more than 30 years of teaching, research and industrial experience in the field of telecommunications, starting from 1981 as R&D engineer working for the telecommunication industry, and developing his carrier to a full professor at the Faculty of Telecommunications, Technical University of Sofia, Bulgaria. He has successfully managed and realized numerous industrial and engineering projects, related to the development of the telecommunication transmission and access network infrastructure in Bulgaria, many R&D and educational projects. His fields of scientific interest and expertise are related to interference suppression, resource management in next generation networks and IoT. He is author of more than 100 scientific publications and is leading B.Sc., M.Sc. and Ph.D. courses in the field of Information Transmission Theory and Access Networks. In the period 2007–2015 he was Dean of the Faculty of Telecommunications at the Technical University of Sofia. Currently he is head of the "Teleinfrastructure R&D" laboratory at the Technical University of Sofia. He is Chairman of the Bulgarian Cluster of Telecommunications, Senior IEEE Member and co-founder of the CONASENSE (Communication, Navigation, Sensing and Services) society.

Anja Klein received the diploma and Dr.-Ing. (Ph.D.) degrees in electrical engineering from the University of Kaiserslautern, Germany, in 1991 and 1996, respectively. From 1991 to 1996, she was a member of the staff of the Research Group for RF Communications at the University of Kaiserslautern. In 1996, she joined Siemens AG, Mobile Networks Division, Munich and Berlin. She was active in the standardization of third generation mobile radio in ETSI and in 3GPP, for instance leading the TDD group in RAN1 of 3GPP. She was vice president, heading a development department and a systems engineering department. In May 2004, she joined the Technische Universität Darmstadt, Germany, as full professor, heading the Communications Engineering Lab. Her main research interests are in mobile radio, including multi-antenna systems, radio resource management, interference management, relaying and multi-hop, cooperative communication, network planning, and cross-layer design. Dr. Klein has published over 290 refereed papers and has contributed to twelve books. She is inventor and co-inventor of more than 45 patents in the field of mobile radio. In 1999, she was inventor of the year of Siemens AG. Dr. Klein is a member of IEEE and of Verband Deutscher Elektrotechniker-Informationstechnische Gesellschaft (VDE-ITG).