Reinforcement Learning Based Energy Efficient LTE RAN

Joan Meseguer Llopis
Orange Polska
Warsaw
Poland
+48 797 318 373
joan.llopis@orange.com

Lukasz Rajewski
Orange Polska
Warsaw
Poland
+48 519 310 854
lukasz.rajewski@orange.com

Slawomir Kuklinski
Orange Polska and Warsaw University of Technology
Warsaw
Poland
+48 601 213 013
slawomir.kuklinski@orange.com

ABSTRACT
Reducing power consumption in LTE networks has become an important issue for mobile network operators. The 3GPP organization has included such operation as one of SON (Self-Organizing Networks) functions [1][2]. Using the approach presented in this paper the decision about turning Radio Access Network (RAN) nodes off and on, according to the network load (which is typically low at night), is taken into account. The process is controlled using a combination of Fuzzy Logic and Q-Learning techniques (FQL). The effectiveness of the proposed approach has been evaluated using the LTE-Sim simulator with some extensions. The simulations are very close to real network implementation: we used the RAN node parameters that are defined by 3GPP and simulations take into account the network behaviour in the micro time scale.

General Terms
Algorithms, Network Management, Measurements, Performance, Design, Experimentation, Theory.

Keywords
Energy savings; LTE; SON; machine learning; reinforcement learning.

1. INTRODUCTION
The new generations of mobile systems offer high bitrate streams to end users. In order to cope with such high traffic demands, the relatively small size of cells in latest mobile systems (LTE) is used, and as the consequence, the number of users per cell is limited. Small cell size causes the increased number of cells in certain area of operator network, typically in cities. The increased number of cells makes cells deployment and operation troublesome and increases the overall power consumption by RAN. In order to cope with the problem of RAN management and having in mind that human-centric management is slow and error prone, a Self-Organizing Networks concept has been developed [3]. This concept enables automation of certain RAN management functions including handover, coverage optimization, load balancing, etc. One of the SON functions, the Energy Efficient RAN (EE RAN), seamlessly turns off some radio nodes (eNodeBs), sector's carriers or node's internal blocks in order to reduce power consumption when the network load is low (typically at night). In fact the radio network is dimensioned to cope with peak traffic hours whereas it can be under-utilized in off-peak traffic time. Switching a cell (or cell sectors) off is possible if network load is low and the neighbouring cells can compensate the coverage. In case of the traffic growth the switched-off cell has to be dynamically switched on.

GECCO'14, July 12–16, 2014, Vancouver, BC, Canada.

http://dx.doi.org/10.1145/2598394.2605694

A holistic approach to energy savings in RAN done by 3GPP is described in [1]. In this technical report, a case study based on different scenarios is made. Some possible solutions, which depend on the given RAN topology, are also proposed in the report. The document does not describe how the EE RAN mechanisms should be implemented. Typically it is assumed that for each RAN node the decision of switching off some cells can be driven by a pre-programmed policy, which takes into account time of day (week) and network traffic patterns. Such simple approach can lead to the degradation of offered services or to low efficiency of the EE RAN mechanism. There is no doubt that the mechanism based on real traffic measurements with appropriate decision algorithm will be much more effective. In this paper, we propose to use a cognitive algorithm for EE RAN. We described the usage of an algorithm for Inter-eNodeB energy saving scenario of [1]. This scenario is based on the so-called capacity limited network, which is homogeneous (composed of a group of cells with similar size) that is presented in Figure 1. For the sake of simplicity we will consider eNodeBs with omnidirectional antennas moreover only switching off of the whole node (eNodeB) is taken into account.

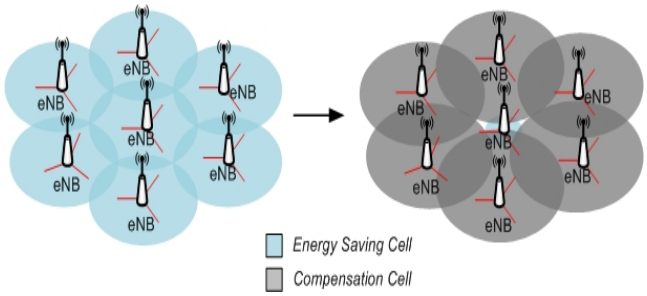

Figure 1. Inter-eNodeB energy saving use case [1]

The usage of machine learning technique automates the EE process and minimizes the degradation of offered services when the network will learn its proper behaviour. There are two situations when the cognitive approach can be applied in EE RAN. The first one is a prediction of the time when a cell can go to the dormant mode. The second one is appropriate prediction when the dormant cell has to be waked-up. If this procedure starts too early the cell will be forced again to go into the sleep state. If the cell will wake-up too late, the serious network degradation will be perceived by the users. The improper behaviour of the algorithm that is responsible for the decision will result in higher power consumption and increased signalling load – in some cases

degraded QoS can be observed as well. The switching events should happen not more often than several times per hour and this means that the switch off process does not have to be fast. On the other hand, in case of high mobility of users, the switching-on phase has to be fast.

The EE process is composed of four phases. Phase 1 lies in a continuous monitoring by each eNodeB to check if it is a candidate for switching off – potentially a preliminary decision about switching off a cell can be taken. During cell monitoring the number of active users, the cell load, time of day and other parameters related to cell activity are collected. The decision on-to-off is based on a cognitive algorithm. Phase 2 is related to the execution of the decision about switching off a cell by appropriate compensation of radio coverage by the neighbouring cells. Phase 3 is related to the monitoring of the load in the area that is compensated by the surrounding cells in order to detect a need for switching on of the sleeping cell. The set of monitored parameters related to cell activity is the same as in Phase 1. The decision about switching a cell on is taken by the cognitive algorithm likewise in Phase 1. Phase 4 concerns of restoration of the normal state of the network, i.e. turning-on the sleeping cells and does not require cognitive techniques. In our approach we used a combination of the well-known Fuzzy Logic and Q-Learning technique (FQL) to take the decision in Phases 1 and 3.

The paper is structured as follows: in section 2, the four Phases of the EE approach are explained in more detail, though the description of cognitive part of the process (i.e. phases 1 and 3) is left for the section 3. In section 4 the effectiveness of the approach is evaluated and realistic simulation results are presented. Finally, section 5 concludes the paper and suggests some further work for the improvement of proposed approach.

2. THE EE PROCESS

For switching the radio stations off and on we use two independent instances of the reinforcement learning (RL) technique combined with Fuzzy Logic in order to cope with continuous input parameters space. RL learns how to act given monitoring information regarding the environment. The parameters that are collected from each eNodeB as the input of the algorithm for all four phases of EE mechanism are presented in Table 1. In this table NAU is the number of active users [4]; the AVGTHR parameter defines the sum of each user's average throughput (downlink) and is normalized and cell's load history (LoadHist) is the aggregated load of eNodeB for the whole day.

In our EE RAN implementation a cell can be in three states [2]: NORMAL, SLEEP and COMPENSATE. When a cell has the state flag equal to NORMAL, it means that this cell is not participating in any energy saving process. The SLEEP mode indicates that the cell is turned off or restricted in physical resource usage. Finally, when the cell is in COMPENSATE mode it means that it has modified power transmission in order to provide coverage in the vicinity of the turned off cell. In our case study, QoS (Quality of Service) refers to the network performance parameters: Handover Failure (HF), Packet Loss Ratio (PLR) and Averaged Packet Delay (PDL). Handover procedure is defined in [5]. When too low received signal level is detected by the user equipment (UE) during the handover execution or due to the lack of resource at the target cell to which UE is handed over the handover failure happens. PDL parameter determines the packet delay in the downlink channel averaged during time period T, which is usually defined by the equipment vendor [4]. PLR is the number of dropped downlink packets ratio to all transmitted packets [4]. The Radio Link Failure (RLF) happens when the terminal detects radio link problems – a broken transmission [6]. Users' RxLev list is a list of measured RSRP level per each cell. Cell's RSRP is defined as the linear average over the power contributions (in Watts) of the resource elements that carry cell-specific reference signals within the considered measurement frequency bandwidth. Moreover, the following data has to be provided for proper operation of the proposed approach: RAN network topology with list of neighbors, transmitted power defaults for all RAN nodes and parameters reporting period.

Table 1. Network monitoring and configuration parameters used

Parameter	Phase	Acronym
The number of active users	1, 3	NAU
Cell load expressed in average throughput	1, 3	AVGTHR
Time of day and load history	1, 3	[Time, LoadHist]
Cell mode (NORMAL, SLEEP, COMPENSATE) with timer of the state	1, 3	[StateTimer(min), StateFlag]
Cell QoS vector: [HF, PLR, PDL]	2, 3, 4	[HF, PLR, PDL]
Radio Link Failure	2, 4	RLF
Users' RxLev (RSRP) list.	2, 4	UERxLevList
Transmission Power	2, 4	TxPower

2.1 Phase 1: Normal state

In this phase each eNodeB collects monitoring data and sends periodically the processed and averaged reports to the node that executes the EE algorithm. According to monitoring data, configuration parameters (network topology), policy parameters (network operator preferences) and history of previous decision the algorithm takes the new decision. The decision is binary and indicates a cell that should be switched off. It also indicates which cells should take part in the cell compensation procedure. The decision about switching off the cell is taken using a model-free RL (Q-Learning). The results can be positive only if all the neighbouring cells are in normal (non-compensating) state. In case when the interval time between switching a cell off and on is under a defined threshold the previous decision is seen as a failure.

2.2 Phase 2: Switching-off a cell

The phase is related to the execution of the decision about switching a cell off by appropriate compensation of radio coverage by the neighbouring cells. The process is relatively simple but has to be smooth, i.e. switching the base station off has to be preceded by proper increase of coverage of active stations. The quality of the process is evaluated by pathological network

behaviour during the transition (radio link failure ratio increase, interference level increase, etc.). In the proposed approach, in order to obtain the proper compensation, a special network pilot terminal (a kind of a mobile phone) is collocated with each eNodeB. Its role is to monitor the level of the neighbouring cells signals when the cell is switched off – it helps in proper cell compensation. This terminal is useful in the case when there are no real users under the coverage of compensated area.

The switching-off operation and the consequent coverage compensation is based on a multistep procedure described below.

2.2.1 First step

This step is composed of a modification of cell state flags of the compensating cells, (i.e. NORMAL to COMPENSATE) and gradual modification of maximum power transmission of EE compensating cells and EE candidate cell. This phase can actually be divided into several sub-steps. Each sub-step determines a new value of the transmitted power (TxPower) for cells involved in the EE procedure. This procedure ensures a gradual change of the topology and thus the users should not perceive any decrease of their QoS. Number of sub-steps to apply depends on delivered QoS in terms of radio link failures (RLF), handover failures (HF) and the received signal level (RxLev) of users in the area of the dormant cell candidate as well as of the cell's pilot terminal. The first step will finish as soon as the number of users under the EE candidate cell will be equal to zero. It is important to note that a new sub-step is performed as soon as new monitoring information is received, which is determined by the network monitoring reports collection frequency.

2.2.2 Second step

Once the number of users attached to EE candidate cell is equal to zero, and the RxLev list of pilot terminals reflects normal RSRP levels from compensation cells, the state flag is set to SLEEP and the neighbouring cell list of surrounding cells of EE candidate cell must be updated in order to avoid conflicts in next handover (HO) procedures. In case of reaching this step and RSRP list of pilot terminal reflects power levels below the minimum required, the transmission power of these cells is increased by the difference of this level and the current RSRP level.

2.2.3 Third step

The EE candidate is switched off. Here two options are considered. First one, the whole eNodeB is turned off or moved to the standby mode. Second option is just the result of reducing the transmitted power to zero. This will force all the UEs to handover to neighbouring cells. In consequence the power consumed by the eNodeB's radio unit will be zero.

2.3 Phase 3: Sleeping state monitoring

This phase is related to the monitoring of the area that is compensated in order to detect a need for switching on of the sleeping cell. All cells that are compensating cells of the sleeping cell are monitored. The set of monitored parameters is the same as in Phase 1. The decision about switching a cell on uses cognitive techniques, the same algorithmic technique as used in Phase 1. Improper behaviour of the algorithm will lead to subsequent turning off the station recently turned on, i.e. the EE ping-pong effect will occur. Another measure of the success is a non-degraded QoS offered to users in the compensated area. In the case, when the interval time between switching off and on is under a defined threshold, the previous decision is seen as a failure.

2.4 Phase 4: Normal state restoration

The parameters of the switched on cells, TxPower and neighbouring cell list as well as their flags to NORMAL are restored. However, similarly to Phase 2, a gradual decrease of compensate cells' TxPower parameter is necessary as well as a gradual increase of TxPower parameter of the switched off cell. The process is non-cognitive one.

3. THE ALGORITHM

3.1 Q-Learning and Fuzzy Logic

We used Reinforcement Learning (RL) algorithm for the switching on/off of an eNodeB. From many available RL variants we have chosen the Q-learning approach due to its simplicity and its proven convergence [7]. In Q-learning, an agent (i.e. in our case the EE RAN agent) switches from one state to another by performing an action. At each state transition a reward is received which measures the quality of the action taken. The goal of the algorithm is to find the best action for each state S_t while maximizing the long-term reward. In order to achieve this, Q-learning keeps a q-value for each state-action pair. This q-value is also known as the estimation of the real-valued function *Q(S,a)*. This value function estimates the expected cumulative discounted reward of performing an action being in the state S and then following the optimal policy. At each state, the estimated q-value is updated using the formula:

$$Q_{t+1}(s_t, a_t) \leftarrow Q_t(s_t, a_t) + \alpha\left[r_{t+1} + \gamma Max_{|a} Q_t(s_{t+1}, a) - Q_t(s_t, a_t)\right] \quad (1)$$

where $\alpha \in [0,1]$ is the learning rate that defines by how much the new value is updated for the current state-action pair. It, therefore, determines the learning speed of the agent. $\gamma \in [0,1]$ is the discount factor and it influences how much the agent considers the immediate reward. The closer γ is to 1 the more it looks after the future payoffs. The reward r_{t+1} measures the quality of the action taken in a certain state (S_t) and it is calculated when the agent arrives to the next state (S_{t+1}).

In a discrete state space environment the state and action spaces are counted and each of them keeps a q value in a so called lookup table. This becomes very complex and impractical when these spaces are continuous. For this reason, the Fuzzy Logic [8] is applied to discretize the continuous input values. To keep the algorithm simple we use only three input parameters to define the cell load state S: number of active users (NAU), cell's averaged throughput (AVGTRGHP) and cell's load history (see Table 1).

$$S = [NAU, AVGTHR, LoadHist] \quad (2)$$

As mentioned before, in order to discretize the continuous space of the input variables, the data obtained from eNodeBs are fuzzified. For the fuzzification, a finite number of fuzzy labels were defined over the domain of each input variables. For the input vector, each variable will have three fuzzy labels (*Low, Medium* and *High*). A membership function that maps the same input values to a degree of truth value in the real range from 0 to 1 is assigned to each label. In this case the triangular (Medium Label) and the trapezoidal (Low and High Labels) membership function have been used with overlapping areas so that the sum of the membership functions at any point of the input space will be equal to 1.

$$\mu_{LOW}(x) = \begin{cases} 1, & x < a \\ \frac{x-b}{a-b}, & a \le x \le b \\ 0, & x > b \end{cases} \quad (3)$$

$$\mu_{MEDIUM}(x) = \begin{cases} 0, & x < a \\ \frac{x-a}{b-a}, & a \le x \le b \\ \frac{x-c}{b-c}, & b < x \le c \\ 0, & x > c \end{cases} \quad (4)$$

$$\mu_{HIGH}(x) = \begin{cases} 1, & x < a \\ \frac{x-b}{c-b}, & a \le x \le b \\ 1, & x > c \end{cases} \quad (5)$$

3.2 Selecting the candidate cell

All the parameters used by the algorithm are defined in Table 1. In order to get the best candidate cell to switch-off (Phase 1) we apply the input [NAU, AVGTHR] of each cell in a particular cell cluster to a Fuzzy Rule-Based System [9] to get a value on the output parameter., the *Quality of Candidate*, *QoC*. Likewise for the cell load related parameters, in this fuzzy system output parameter we have three fuzzy labels: Low, Medium and High. Each label is also assigned to a membership function (eq. 3-5) that maps the same input values to a degree of truth value in the real range from 0 to 1. The Table 2 consists the conjunctive rules of inference based on the observations of the network.

Table 2. Fuzzy System Rules

Input		Output
NAU	AVGTHR	QoC
L	L	H
L	M	H
L	H	M
M	L	M
M	M	L
M	H	L
H	L	L
H	M	L
H	H	L

We use the well-known max-min Mandami implication method of inference [8], which is denoted as follows:

$$\mu_{QoC}(QoC) = \max_{k}\{min[\mu_{NAU^k}(NAU), \mu_{AVGTHR^k}(AVGTHR)]\} \quad (6)$$

The result of this method is a fuzzy set so that it is yet to be calculated a final QoC discrete value. This process is known as defuzzification and in our case the so-called *centre of gravity* procedure has been applied [8].

3.3 Decision making: FQL

As mentioned before the output action is binary {0;1}. We assign the value 1 to the action of switching off and 0 for the action of switching on. When the cell is already switched on, the action 0 then means to keep the cell in the same operating state (Phase 1), i.e. the "*do-nothing"* action. Similarly, if the cell is switched off, the action 1 refers to the "*do-nothing"* action as well. In FQL, states and actions are defined using fuzzy membership functions. The continuous state variables are first transformed into a finite number of fuzzy variable membership functions. This process is called fuzzification. From these fuzzy variables, the corresponding output is calculated based on the Fuzzy Inference System (FIS). Finally the fuzzy output of the FIS is mapped back to the continuous output variable through the process of defuzzification. In general, the rule based on FIS consists of arbitrary number of different rules constructed out of AND and OR operators. A FIS has a rule based consisting of *N* rules, and the input vector *X* has *n* elements representing the input space. It typically expresses an inference as a conclusion (consequent) [9]. In our particular case we have three input variables [NAU, AVGTHR, LoadHist (time of day)] (see Table 1) with three possible labels for each one (L∈[L,M,H]) and two consequents denoted by O_p. Hence, the total number of rules is 27. Each of these rules has the following form:

$$Rule\ i:\ if \begin{bmatrix} \text{NAU} \\ \text{AVGTHR} \\ \text{LoadHist} \end{bmatrix} = \begin{bmatrix} L_1^i \\ L_2^i \\ L_3^i \end{bmatrix} then[O_p]\ is \begin{cases} 0\ with\ q_{value} = q_1 \\ 1\ with\ q_{value} = q_2 \end{cases} \quad (7)$$

As mentioned above, for each rule, O_p represents the action to be taken, and in each rule we will have 2 q-values for each action. At the beginning, all q-values will be initiated to 0. Its value will be updated after the action taken and the reward computed.

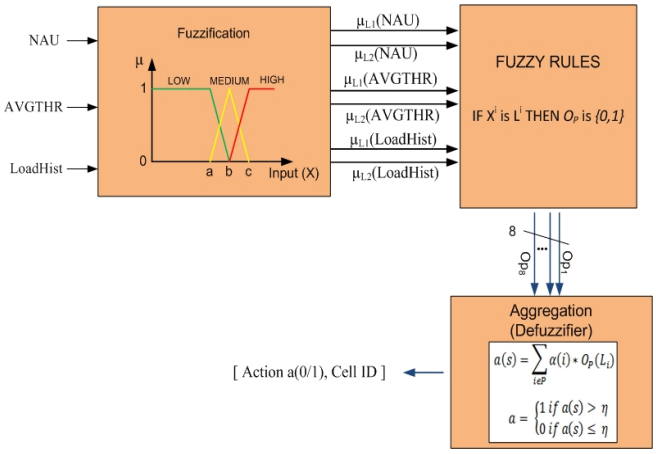

Figure 2. EE RAN Fuzzy Inference (based) system

The general structure of the developed fuzzy inference system (see Figure 2) consists of three conceptual blocks or components: a rule base, which contains a selection of fuzzy rules; a dictionary, which defines the membership functions used in the fuzzy rules; and a reasoning mechanism, which performs the inference procedure upon the rules and given facts to derive a reasonable output or conclusion. However, sometimes it is necessary to have a crisp output. In such a case the reasoning mechanism has to include the defuzzification mechanism to extract a crisp value that represents the output fuzzy set in the best way. In our case, the entire space of rule will be 27 for all the possibilities state input vector. Moreover, each input will have maximum two values of membership function (L&M or M&H). As a result, the maximum number of activated rules (with result different than zero) will be 8 for a particular input vector. Therefore, we will have 8 output values which will be fed to the defuzzification component in order

to get a crisp value indicating the action to be taken. All q values, which are associated to a rule and action pair, are kept in a table in the following form:

Table 3. The Q-Learning table template

Rule	Output Action O_p	q-value
R1	1	v_{11}
	0	v_{12}
R2	1	v_{21}
	0	v_{22}
R3	1	v_{31}
	0	v_{32}
…	…	…

The Q table represents the memory of what the agent have learned through many experiences. Since Phase 1 and Phase 3 have independent FQL instances, each phase will have its own Q matrix with its own rules. Consequently we use two different rewards as well: RW_{ph1} (8) and RW_{ph3} (9) for the phases. In both cases we use a continuous reward function based on progress estimators. Progress estimators provide a measure of improvement towards an objective [9]. Moreover, the Gaussian function in both of them has been used in order to accelerate the learning process [10]. In order to update the q-value associated to the action chosen, the reward should be computed taking into consideration the action taken by the agent (O_p). Therefore, the reward functions for both Q-Learning instances are defined as follows:

$$RW_{ph1}(O_k) = \begin{cases} 1 - 2 \cdot e^{-0.5*\frac{(W_{it}*\nabla QoS_1)^2}{\sigma_{ph1}^2}}, & O_k = O_1 \\ 2 \cdot e^{-0.5*\frac{(W_{it}*\nabla QoS_1)^2}{\sigma_{ph1}^2}} - 1, & O_k = O_2 \end{cases} \quad (8)$$

$$RW_{ph3}(O_k) = \begin{cases} 1 - e^{-0.5*\frac{\nabla QoS_3^2}{\sigma_{ph3}^2}}, & O_k = O_1 \\ e^{-0.5*\frac{\nabla QoS_3^2}{\sigma_{ph3}^2}}, & O_k = O_2 \end{cases} \quad (9)$$

$$\nabla QoS = w_{HF} * HF + w_{PLR} * PLR + w_{PDL} * PDL \quad (10)$$

For both cases we have introduced the term $\nabla QoS \in \mathbb{R}^+$ (10). This term gathers all the Key Performance Indicators (KPIs) that symbolize the network's performance degradation: handover failure (HF), packet lost ratio (PLR) and averaged packet delay (PDL). The agent may switch off/on an eNodeB when the cell is under a load peak. In consequence, a global degradation of these KPIs is expected. ∇QoS_1 aggregates these KPIs during the Phase 2 and Phase 3, and ∇QoS_3 aggregates these KPIs during the Phase 3 and Phase 4. For the reward RW_{ph1} the maximum and minimum value are 1 and -1 respectively (i.e. $RW_{ph1} \in [-1,1]$). When ∇QoS_1 approaches 0, this reward will be closer to 1 for the switch-off action (O_2) and it will be near -1 for the switch-on action (O_1). On the other hand, when ∇QoS_1 is high, the switch-off action is punished with a reward close to -1 and with a value close to 1 for the switch-on action. As for the reward RW_{ph3} the maximum and minimum value are 1 and 0 respectively (i.e. $r_{t+1} \in [0,1]$). When ∇QoS_3 increases, for those active rules with action equal to O_1 and O_2, the value of RW_{ph3} will approach to 1 and 0 respectively. In the same way, when ∇QoS_3 decreases, for those active rules with action equal to O_1 and O_2, the value of RW_{ph3} will approach to 0 and 1 respectively. In equations (8) and (9), parameter σ determines the reward gradient influence area. The lower σ is the more sensitive the reward will be against an increase of ∇QoS. This parameter has been object of study during the algorithm's simulation phase since it has a big impact on its performance and convergence speed. The time window in which the reward of Phase 1 can punish the action of switching a cell off must be limited since the second FQL instance is the responsible to re-establish the normal state in case of any network performance degradation. Therefore, in equation (8), it has been added the coefficient W_{it} to the ∇QoS_1, which represents the QoS degradation impact factor for reward of phase 1 when a cell is turned off. This is shown in the following equation:

$$W_{it}(t_{sleep}) = 1 - \frac{1}{1+e^{-4*(t_{sleep}-RWimpTime)}} \quad (11)$$

The above function is the scale function with the transition from 1 to 0 at the cell sleep time (t_{sleep}) equal to $RWimpTime$. Therefore $RWimpTime$ represents the impact time of ∇QoS against the reward of Phase 1. In other words, this means that at any sleep time after $RWimpTime$ the value of this factor is 0 and in consequence value of ∇QoS will be irrelevant for the final reward, whose value will be 1 (from that time on) for the switch-off action and 0 for the switch-on action.

The first step of the FQL algorithm is to evaluate the current state of the network. A set of new values for the input variables will be provided and the degree of truth of each FIS rule i will be computed. The degree of true is the product of membership values of each input state label (L) for the specific rule i:

$$Algorithm\ Input \quad \begin{bmatrix} NAU \\ AVGTHR \\ \text{LoadHist} \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$$\alpha(i) = \mu_{L_i^1}(x) * \mu_{L_i^2}(y) * \mu_{L_i^3}(z) \quad (12)$$

The output action for each activated rule is denoted by O_p, and it can take only two possible values: 0 or 1. The way in which one of the actions is selected depends on whether the agent is on the learning stage (exploration stage) or exploitation stage. The exploration/exploitation policy (EEP) followed to decide which action to take is described via the equation (13). Where ε is the tradeoff between exploration and exploitation. In the exploitation phase (i.e. ε = 1), the agent has achieved the global optimal behaviour; thus the output chosen is the one with maximum q-value out of the two present the Q-Learning table.

$$o_p = \begin{cases} \text{argmax}_{k \epsilon K}\, q(L_p, o_p^k), & with\ probability\ \varepsilon \\ \text{random}_k\, o_p^k, & with\ probability\ 1-\varepsilon \end{cases} \quad (13)$$

If α is the degree of truth for each rule and P is the total number of activated rules, the final action will be:

$$a(s) = \sum_{i \epsilon P} \alpha(i) * O_P(L_i) \quad (14)$$

where i is the number of active rule. The final action given here belongs to the continuous space with minimum value 0 and maximum 1, i.e. [0,1]. However, a final decision (whether to

switch a cell off or on) is yet missing. To solve this, we have introduced a rule based on a variable threshold (η), that is:

$$a = \begin{cases} 1 \; if \; a(s) > \eta \\ 0 \; if \; a(s) \leq \eta \end{cases} \qquad (15)$$

Therefore, the action to be applied depends on the final action value $a(s)$, where η shows the strictness of the current condition given (in terms of cell load) that will allow the switching-off/on of a cell. Likewise σ, the value to assign to η has been a topic of study during the simulation phase of the algorithm.

After applying the final action, the agent moves to the next state s_{t+1} and the reward will be calculated. Then the next step is to update those elements of Q-Learning table whose pair rule-action coincide with those activated rules L_p and its corresponding action chosen O_p. This is done as showed with the following equation:

$$q_{t+1}(L_p, O_p) \leftarrow q_t(L_p, O_p) + \beta * \alpha_p(s_t) * \Delta Q \qquad (16)$$

First of all, before the agent applies the action, the q-value for the current state s_t must be calculated as follows:

$$Q(s_t, a(s_t)) = \sum_{p \epsilon P_s} \alpha_p(s_t) * q(L_p, O_p) \qquad (17)$$

Where P_s is the set of activated rules and $q(L_p, O_p)$ is the q-value for the pair label L_p and output O_p, which is taken from the Q-Learning table. In equation (16) the quantity ΔQ is defined as the difference between the old and new value of Q(x,a(x)):

$$\Delta Q = r_{t+1} + \gamma V_t(s_{t+1}) - Q(s_t, a(s_t)) \qquad (18)$$

In the above equation, r_{t+1} is the reward computed for the new state s_{t+1} (via monitoring the network status after the action is taken) and $V_t(s_{t+1})$ measures the aggregated maximum q-value achievable for the new state. The former is calculated following equations (8) or (9) and the latter is obtained from the formula shown below.

$$V_t(s_{t+1}) = \sum_{p \epsilon P_{s_{t+1}}} \alpha_p(s_{t+1}) * \text{argmax}\{q(L_p, O_p)\} \qquad (19)$$

Both factors, β and γ, have range value from 0 to 1 ($0 \leq \beta < 1$ and $0 \leq \gamma < 1$). β is the learning rate which determines the influence of new information on the previous knowledge at the moment of taking a decision. The closer β is to 0 means the less learning and the closer it is to 1 means the only newest information is considered. The closer γ is to 0 the more the agent will consider only immediate rewards. If γ is closer to 1, the agent considers future reward with grater preference, willing to delay the immediate payoff.

3.4 Exploration and exploitation phases

The essence of Q-Learning is that the agent learns through experience without external supervision. This agent will explore from state to state, from transition to transition, until it reaches the goal. Each exploration process is known as an episode. In one episode the agent will move from initial state to another and so on until the goal state. In our case the goal state will be the execution of EE without a degradation of delivered QoS. Once the agent completes one episode it starts a new one. This exploration phase is also known as the training session. The longer the training is, the more global optimal way the agent will take the decisions. Based on the action taken, each episode includes two possible paths per each FQL instance. For the FQL instance of phase 1, first path belongs to the whole cycle *Phase 1* → *Phase 2* → *Phase 3* → *Phase 4* → *Phase 1* (i.e. in time *t* the agent decides to switch-off the candidate cell), and second one to the cycle *Phase 1* → *Phase 1* (i.e. in time *t* the agent decides to not to switch-off the candidate cell). Once the agent finishes the execution of Phase 4, the reward (RW_{ph1}) is computed, the values of Q-table 1 are updated and the agent moves to Phase 1. Similarly to Phase 1, when the agent is in phase 3 and its FQL instance is used, the episode also includes two possible paths based on the action taken: first one belongs to the cycle *Phase 3* → *Phase 3* (i.e. in time *t* the agent decides to not to switch-on the switched-off cell) and the second one to the cycle *Phase 3* → *Phase 4* → *Phase 1* (i.e. in time *t* the agent decides to switch-on the switched-off cell). In both cases, whenever the agent reaches the Phase 3 or Phase 1 the reward (RW_{ph3}) is computed and the values of Q-table 2 are updated.

In both FQL instances the exploration happens by the selection of a lower than 1 value of ε in EEP (eq. 13). The parameter is increased exponentially in each iteration. Once ε reaches the value of 1 only the exploitation iterations are present.

4. SIMULATIONS AND DISCUSSION

For the evaluation of the presented algorithm the LTE-Sim simulator has been used [11]. Two main modifications have been made in order to facilitate the interaction of our EE RAN agent with the simulated network. First one is the enhancement of the simulator in order to obtain of the algorithm input parameters, such: NAU, State Flag, State Timer and RLF (see Table 1). Second one is the implementation of a new traffic and mobility patterns. The simulated network's topology is composed of 7 cells with omnidirectional antennas. Each one has a radio range of 400 meters and 12 mobile terminals when the simulation starts. The initial transmission power is set to 23 dBm. Simulations are divided by simulation slots, each one representing a typical working day (from Monday to Friday). Users' traffic patterns (TP) are deterministic. For each simulation run we have set 7 slots (i.e. week). As it is shown in Figure 3 the TP for one slot (second 1-200) it can be differentiated three parts in terms of traffic load: two traffic peaks (90-100%) and one off-traffic peak (0-5%). The off-peak traffic represents cell's activity at night (e.g. 10 pm-6 am).

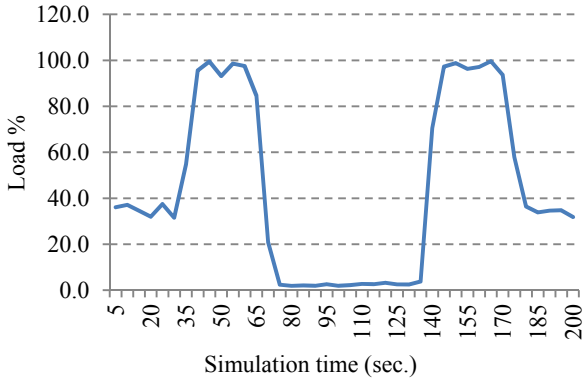

Figure 3. Users' traffic pattern

For all simulations VoIP (G.729 codec) and Video traffic models with 128Kb download transmission rate have been used. Each user receives one VoIP session and one video stream in the medium traffic intervals (approx. 0-30 and 170-200 sec.), one VoIP session in the off-peak traffic and 2 VoIP and 3 video applications at the traffic peak. User's mobility pattern is deterministic: at the peak traffic time, 90% of the users of one cell move towards some of its neighbouring cell to stay there until the time of the second traffic peak when they move back towards the initial position.

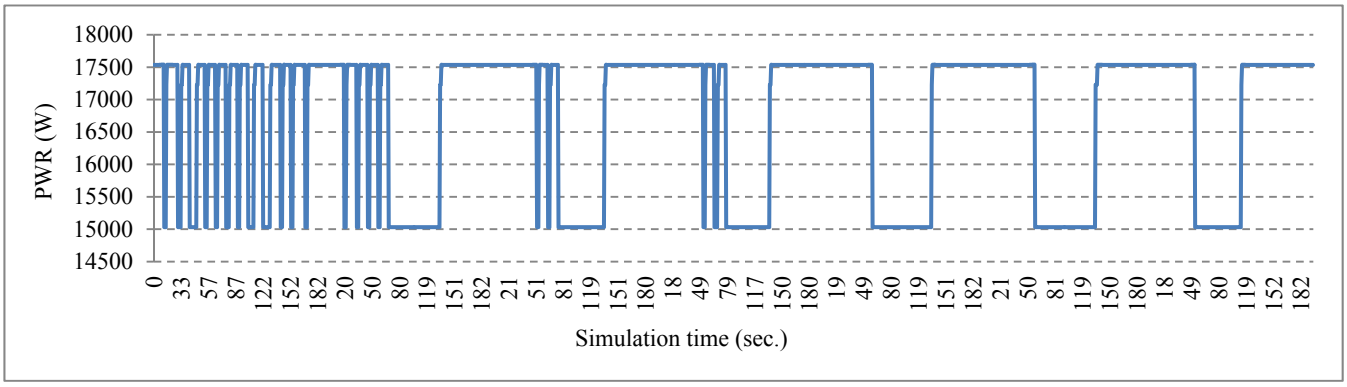

Figure 4. Power consumption for the whole network

In this way, within the simulated topology (see Fig. 1) the cell at centre would emulate the business area of a big city which usually has no users at night.

Both Q-learning instances of our algorithm (the EE RAN agents) should achieve the convergence not longer than in the first 2-3 time slots. Thus, the selection of some Q-Learning parameters (i.e. from the reward and the real-valued functions) is critical for the algorithm's convergence speed and performance. For this reason, the process of tuning of some parameters is necessary. In our case, in order to get the optimal learning parameters' values of the algorithm and robust algorithm behaviour the tuning process, which lied on changing of mutual importance of QoS parameters and learning parameters, has been very long one. The selected values of the parameters are shown in Table 4.

Table 4. Algorithm learning settings

Algorithm parameter	Parameter value
β: learning rate	0.7
γ: discount factor	0.9
ε: explotation/exploration trade-off	50% (beginning of slot 1) 100% (Slot 2-7)
W_{HF}	0.4
W_{PLR}	10.0
W_{PDL}	10.0

Table 5. Membership function thresholds

Input	a	b	c
NAU	4	12	14
AVGTHR (%)	10	50	80
QoC	0.5	0.6	0.7

Additionally, the membership functions described in equations 3-5 have been implemented using the values presented in Table 5. Algorithm performance is checked by the evaluation of the following indicators: averaged packet delay (PDL), packet loss ratio (PLR), handover failures (HF) and power consumption (PWR). Results are provided in form of charts with data changing over the simulation time (7 slots of 200 seconds each). In fact, four charts are being shown: first one shows the power consumption (in W) for the whole network, i.e. 7 cells (see Fig. 4). In this chart the EE episodes can be seen in form of down steps within the Power(t) function. The used power consumption model is described in [13]. The second, third and fourth charts (Fig. 5, 6 and 7) show the network performance via the KPIs: PDL, PLR and HOF respectively. These indicators show the network performance degradation as a consequence of the action taken by the algorithm (switch off/on the cell) when the cell is congested. If the algorithm achieves the convergence on its actions in a relatively short time, two key results are expected. First one is a decrease of switch-off events when the cell is congested while prolonging them at the time within the slot when the off-peak traffic is given. The second one is a decrease of the delivered QoS degradation along the simulation as a consequence of the less and less wrong decisions that are being taken.

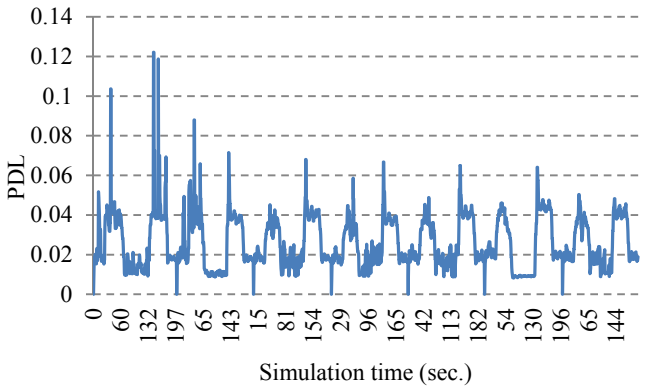

Figure 5. Averaged packet delay

Two main observations can be drawn from these charts. First one is related to the learning speed and convergence of both Q-learning instances of our agent. According to the expectations, from the third slot the agent achieves the coherence on its decisions about the proper time when a cell must enter and leave the dormant state. If we compare Fig. 4 and Fig. 3 we can see that EE episodes are aligned with the off-peak traffic interval. However, it is also noticeable that the Q-learning instance that decides about entering the dormant cell has a relatively slower learning in comparison to the instance that decides about leaving the dormant, whose convergence is achieved from second slot.

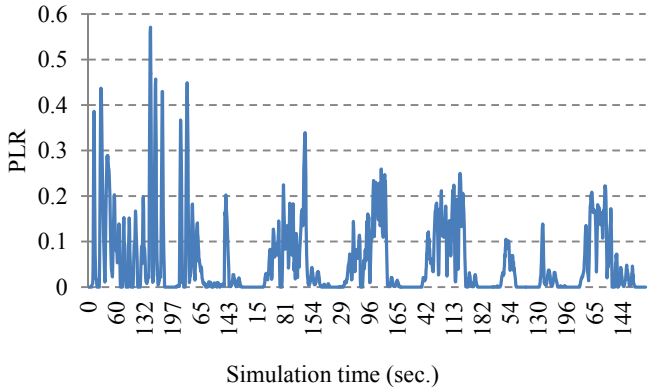

Figure 6. Packet Loss Ratio

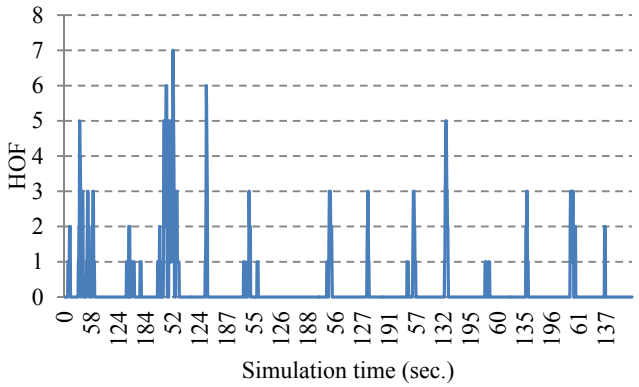

Figure 7. Handover failure

The second observation is depicted from the charts showing the network performance degradation indicators (i.e. Fig. 5-7). Here the progressive decrease of these indicators is observed. This is because of the reduction of the amount of switch-off actions during the peak traffic intervals at the time of agent's exploration.

5. CONCLUSIONS

In the paper a cognitive EE RAN approach based on machine learning has been presented. We have successfully applied the reinforcement learning to define the proper moments of switching off or on the mobile network radio stations. During simulations we have verified the proper behavior of the proposed approach via realistic simulations. In fact, the concept can be easily transferred to real networks, because it uses the network parameters that are used by real radio stations of LTE RAN as defined by 3GPP standards. Moreover, our simulations took into account all the processes which exist in real network (handovers, users' mobility, etc.). We decided to use the learning algorithm in order to have adaptive behavior of the EE RAN procedure and to avoid a design of an algorithm by an expert, however we noticed that the algorithm require a lot of efforts related to tuning it in order to obtain proper and robust behavior of the algorithm. The final result is satisfactory, but in real networks the initial learning can lead to the perceived by the users degradation of services. In the future work we intend to minimize the problem and make more simulations with differentiated users' traffic and behavior. Moreover we will use cells with directional antennas.

6. ACKNOWLEDGMENTS

The work presented in this paper has been supported by the Celtic COMMUNE project CP08-001 entitled Cognitive Network Management under Uncertainty [13]

7. REFERENCES

[1] 3GPP, "LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Potential solutions for energy saving for E-UTRAN (Release 11)", TR 36.927, 2012.

[2] 3GPP, "Technical Specification Group Services and System Aspects; Telecommunication management; Study on Energy Savings Management (ESM) (Release 10)", TR32.826, 2010.

[3] 3GPP, "Technical Specification Group Services and System Aspects; Telecommunication Management; Self-Organizing Networks (SON); Concepts and requirements (Release 10)", TS 32.500, 2010.

[4] 3GPP, "Evolved Universal Terrestrial Radio Access (E-UTRA); Layer 2 – Measurements (Release 11)", TS 36.314, 2012.

[5] 3GPP, "E-UTRA Radio Resource Control (RRC); Protocol specification (Release 11)", TS 36.331, 06.2012.

[6] 3GPP, "Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control (RRC); Protocol specification (Release 12) ", TS 36.331, 03.2014.

[7] R Sutton "Reinforcement learning: An introduction" MIT Press 1998.

[8] T. J.Ross, Fuzzy logic With Engineering Applications, 2010.

[9] Mataric, M.J.: Reward functions for accelerated learning. In: Proc. of the 11th ICML. (1994) 181–189.

[10] L. Matignon et all. Reward Function and Initial Values: Better Choices for Accelerated Goal-Directed Reinforcement Learning. Artificial Neural Networks – ICANN 2006. Lecture Notes in Computer Science Volume 4131, 2006, pp 840-849.

[11] G. Piro, L. A. Grieco, G. Boggia, F. Capozzi and P. Camarda, "Simulating LTE Cellular Systems: an Open Source Framework", online, http://telematics.poliba.it/publications/2010/TVT/PiroTVT2010.pdf

[12] W. Keating, "Reducing Energy Consumption in Access Networks", August 2011. School of Electronic Engineering, Dublin City University.

[13] http://www.celtic-initiative.org/Projects/Celtic-projects/Call8/COMMUNE/commune-default.asp.