SON Coordination for parameter conflict resolution: A reinforcement learning framework

Ovidiu Iacoboaiea, Berna Sayrac, Sana Ben Jemaa
Orange Labs
38-40 rue du General Leclerc 92130
Issy les Moulineaux, France
Email: {ovidiu.iacoboaiea,berna.sayrac, sana.benjemaa}@orange.com

Pascal Bianchi
Telecom ParisTech
37 rue Dareau 75014
Paris, France
pascal.bianchi@telecom-paristech.fr

***Abstract*—Self Organizing Network (SON) functions are meant to automate the network tuning, providing responses to the network state evolution. An instance of a SON function can run on one cell (distributed architecture) or can be built to govern a cluster of cells (centralized/hybrid architecture). From the operator point of view, SON functions are seen as black boxes. Several independent instances of one or multiple SON functions running in parallel are likely to generate conflicts and unstable network behavior. At a higher level, the SON-COordinator (SONCO) seeks to solve these conflicts. This paper addresses the design of a SONCO. We focus on coordinating two distributed SON functions: Mobility Load Balancing (MLB) and Mobility Robustness Optimization (MRO). Thus on each cell we will have an MLB and an MRO instance. The MLB instances will tune the Cell Individual Offset (CIO) parameter and the MRO instances will tune the HandOver (HO) Hysteresis parameter together with the CIO parameter. The task of the SONCO is to solve the conflicts that will appear on the CIO parameter. We propose a Reinforcement Learning (RL) framework as it offers the possibility to improve the decisions based on past experiences. We outline the tradeoff between configurations through numeric results.**

***Index Terms*—SON; Coordination; MLB; MRO;SON instances; LTE; reinforcement learning; TD;**

I. Introduction

The continuous growth of traffic demand is forcing operators to improve their network capabilities by technological upgrades (LTE Advanced [1]), network densification and introduction of Heterogeneous Networks (HetNets). These upgrades lead to increased CAPital EXpenditures (CAPEX) and OPerational EXpenditures (OPEX). In order to reduce the costs, Release 8 of 3GPP has introduced the Self Organizing Networks (SON) that replace the costly human intervention with automated mechanisms. These mechanism are usually classified into 3 categories Self-Configuration, Self-Optimization and Self-Healing. We focus on the 2nd which provides algorithms that offer a run-time optimization of the network. In the rest of the paper, SON will refer to self-optimization.

In a real network you may find more than one SON function (e.g. Mobility Load Balancing (MLB) and Mobility Robustness Optimization (MRO)) that are trying to optimize some Key Performance Indicators (KPIs) by tuning a set of parameters and by doing so they may impact the performance of one another. Even having just one SON function (e.g. MLB) can raise problems as we may have several of its instances running on neighboring cells, unaware of each other so this can lead to instabilities; it may usually happen in a multi-vendor environment. We therefore need a SON COordination (SONCO) mechanism to deal with conflicts and instabilities. Note that we also use the term enhanced Node B (eNB) to refer to a cell.

The SONCO concept has been introduced fairly recent with Release 10 of 3GPP [2]. Until now the work on coordination strategies has mainly focused on solving conflicts based on the instantaneous network conditions ignoring the outcome of the past decisions. A coordination mechanism between MRO and MLB which attributes priorities to these SON functions can be found in [3] and [4]. Decision trees are used for tuning the Remote Electrical Tilt (RET) and Transmission Power (TP) values in [5] and [6]. In [7] the solution is based on restrictions on the value set of the MRO and MLB parameters. A per user optimization, is proposed in [8] where priorities are attributed to users for handovers based on the weights attributed to the SON functions (MRO and MLB). These approaches ignore an important piece of information which is the outcome of past decisions. Since the SON functions are considered as black-boxes (the operator does not know the algorithm inside), there will be an inevitable uncertainty on the impact of the SONCO's decisions on the SON functions and on the KPIs. In order to assure network stability and KPI performance, this uncertainty must be minimized. This encourages us to use the Reinforcement Learning (RL) [9] framework where we can use to some extent the information regarding the impact of our past decisions. RL has been widely used in self-optimization problems. For example in [10] it has been used for tuning relay-backhaul resource allocation parameters, and in [11] for tuning the CCO parameters. In this paper we target to evaluate the benefits of a RL mechanism in SON coordination. To the best of our knowledge, RL has not yet been used for the coordination of multiple SON functions.

We focus on coordinating several MLB and MRO instances that run on neighboring eNBs. We use a centralized SON coordinator whose target is to coordinate the SON (functions') instances by arbitrating their requests. The SON instances will not directly execute the desired parameter changes on the

DOI: 10.1109/WCNCW.2014.6934885

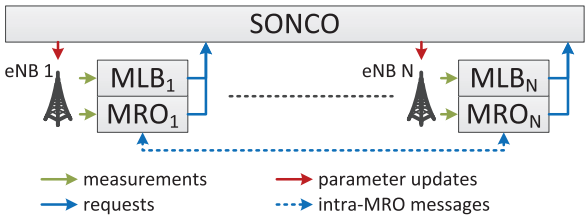

Figure 1. Functional Block Diagram: SONCO↔MLB/MRO interactions

network, instead they will formulate requests to the SONCO and it is the SONCO that decides which request will be executed and which will not.

The rest of this paper is organized as follows: Section II provides the system description presenting the scenario with the MLB and MRO instances. Section IV outlines the characteristics of the SONCO together with the RL algorithm. Simulation results are included in Section IV and Section V concludes the paper.

II. System Description

We consider a network segment composed of N cells. On each cell i ($i = 1, ..., N$), we have one MLB instance and one MRO instance running (see fig. 1). The MLB instance tunes the Cell Individual Offset (CIO) parameter to optimize the cell load, and the MRO instance tunes the Hysteresis parameter to optimize 4 metrics: the number of Too Early (E) HandOvers (HOs), the number of Too Late (L) HOs, the number of HOs to a Wrong Cell (W) and the number of Ping-Pongs (P) [12]. The CIO and the Hysteresis are two parameters that are used in mobility management as follows:

- In IDLE mode, a User Equipment (UE) is camped on the eNB that satisfies the following condition:

$$camping\ eNB = \underset{i\in\{1,...,N\}}{\text{argmax}}\ (RSRP_i + C_i) \quad (1)$$

  where $RSRP_i$ is the Reference Signal Received Power from eNB i and C_i is the CIO of eNB i.
- In CONNECTED mode, a UE attached to eNB i will perform a HO to eNB j if $j \neq i$ and:

$$j = \underset{k=\{1,...,N\}}{\text{argmax}}\ \left(RSRP_k + C_k + H_i\mathbb{I}_{\{k=i\}}\right) \quad (2)$$

  where H_i is the Hysteresis of eNB i and $\mathbb{I}_{\{\bullet\}}$is the indicator function ($\mathbb{I}_{\{true\}} = 1$,$\mathbb{I}_{\{false\}} = 0$). Note that we are using *per cell* CIO and Hysteresis parameters.

The algorithm inside the SON function is unknown to the network operator. For simulation purposes, we design the MLB and MRO functions as presented in the next subsections.

In the sequel we consider the following notations: Let X_i be the value of parameter X for cell i. We define $X_i\downarrow$, $X_i\uparrow$ and $X_i\updownarrow$ to mean decrease, increase and maintain X_i. We propose a SON design where the request $U_{t,i}^{SON(X)}$ sent at time t by the instance of the *SON* function (MLB or MRO) running on cell i and tuning the parameter X has 2 components:

- the parameter change: $U_{t,i}^{SON(X)}[1] \in \{X_i\downarrow,\ X_i\uparrow,\ X_i\updownarrow\}$
- an indication of how happy the instance is with the current value of parameter X_i: $U_{t,i}^{SON(X)}[2]$. The smaller the value is the more critical the change is. This value ranges between $[-1; 0]$ except the case where we are cumulating the effect of a group of cells and the range will in this case be $[-N; 0]$.

A. Mobility Load Balancing

For the MLB SON function, the input (i.e. the optimized metric) is the cell load which is given by the resource utilization (average number of occupied Physical Resource Blocks - PRBs) of the hosting cell (the cell on which the MLB instance runs). The tuned parameter is the CIO (C) of the hosting cell. For cell i, the update request of the MLB instance at time t can be expressed as:

$$U_{t,i}^{MLB(C)} = \left(U_{t,i}^{MLB(C)}[1]; U_{t,i}^{MLB(C)}[2]\right) = \begin{cases} \left(C_i\downarrow; -\varphi\left(L_{t,i}; g_{MLB}^{C\downarrow}, m_{MLB}^{C\downarrow}\right)\right) & \text{, if } L_{t,i} > \mathbb{T}_{load}^{H} \\ \left(C_i\uparrow; \varphi\left(L_{t,i}; g_{MLB}^{C\uparrow} m_{MLB}^{C\uparrow}\right) - 1\right) & \text{, if } L_{t,i} < \mathbb{T}_{load}^{L} \\ (C_i\updownarrow; 0) & \text{, otherwise} \end{cases} \quad (3)$$

where C_i is the current CIO of cell i, $L_{t,i}$is the load on cell i at time t, $\mathbb{T}_{load}^{H}$ and $\mathbb{T}_{load}^{L}$ are fixed thresholds used by the MLB instance to trigger CIO modification requests ($\mathbb{T}_{load}^{H} > \mathbb{T}_{load}^{L}$); g is the steepness and m the center of a strictly increasing S-shaped function of x with values in $[0; 1]$: $\varphi(x; g, m) = 1/\left(1 + e^{-g\cdot(x-m)}\right)$.

B. Mobility Robustness Optimization

We consider a distributed implementation of the MRO where the MRO instances running on neighboring eNBs communicate with each other. At time instant t, the MRO instance running on cell i has the following metrics as input: the number of HO ping pongs ($N_{t,i}^{P}$), the number of too late HOs ($N_{t,i}^{L}$), the number of too early HOs ($N_{t,i}^{E}$) and the number of HOs to a wrong cell ($N_{t,i}^{W}$) which all originate from cell i. Since the MRO tunes two parameters, the CIO (C) and the Hysteresis (H), the MRO of cell i at time t sends 2 simultaneous requests: one for each parameter. The one regarding the Hysteresis can be expressed as follows:

$$U_{t,i}^{MRO(H)} = \left(U_{t,i}^{MRO(H)}[1]; U_{t,i}^{MRO(H)}[2]\right) = \begin{cases} (H_i\downarrow; \emptyset), & \text{if } q_{t,i}^{1} = 1 \\ (H_i\uparrow; \emptyset), & \text{if } q_{t,i}^{2} = 1 \\ (H_i\updownarrow, \emptyset), & \text{otherwise} \end{cases}$$
$$q_{t,i}^{1} = \mathbb{I}_{\{N_{t,i}^{L} > \mathbb{T}_{L}^{H}\}}\mathbb{I}_{\{N_{t,i}^{E} < \mathbb{T}_{E}^{L}\}}\mathbb{I}_{\{N_{t,i}^{W} < \mathbb{T}_{W}^{L}\}}\mathbb{I}_{\{N_{t,i}^{P} < \mathbb{T}_{P}^{L}\}}$$
$$q_{t,i}^{2} = \mathbb{I}_{\{N_{t,i}^{L} < \mathbb{T}_{L}^{L}\}}\left(1 - \mathbb{I}_{\{N_{t,i}^{E} > \mathbb{T}_{E}^{H}\}}\mathbb{I}_{\{N_{t,i}^{W} > \mathbb{T}_{W}^{H}\}}\mathbb{I}_{\{N_{t,i}^{P} > \mathbb{T}_{P}^{H}\}}\right) \quad (4)$$

where H_i is the hysteresis of cell i and Ø means that this parameter is void (does not have any meaningful/practical value). $\mathbb{T}_{(\cdot)}^{H}$ and $\mathbb{T}_{(\cdot)}^{L}$ are fixed thresholds used to trigger events ($\mathbb{T}_{(\cdot)}^{H} > \mathbb{T}_{(\cdot)}^{L}$) .

The request regarding the CIO can be expressed as follows:

$$\begin{array}{l} U_{t,i}^{MRO(C)} = \left(U_{t,i}^{MRO(C)}\left[1\right]; U_{t,i}^{MRO(C)}\left[2\right]\right) = \\ \begin{cases} \left(C_i \uparrow; \sum_{j\in\mathcal{N}(i)} U_{t,j\to i}^{MRO(C)}\left[2\right]\right) & \text{, if } q_{t,i}^{3}=1 \\ \left(C_i \updownarrow; 0\right) & \text{, otherwise} \end{cases} \\ q_{t,i}^{3} = \mathbb{I}_{\left\{\exists j\in\mathcal{N}(i) \text{ s.t. } U_{t,j\to i}^{MRO(C)} = C_i\uparrow\right\}} \end{array} \tag{5}$$

where $\mathcal{N}(i) = \{1, ..., N\} \setminus \{i\}$ is the index set of the neighboring cells of cell i. $U_{t,j\to i}^{MRO(C)}$ is an *intra-MRO message*. The MRO instance running on eNB j sends to its neighboring MRO instances ($k \in \mathcal{N}(j)$) a message informing them about its state of *happiness* with their current CIO value (C_k) and requesting an increase if needed:

$$\begin{array}{l} U_{t,j\to k}^{MRO(C)} = \left(U_{t,j\to k}^{MRO(C)}\left[1\right]; U_{t,j\to k}^{MRO(C)}\left[2\right]\right) = \\ \begin{cases} \left(C_k \uparrow; \varphi'\left(N_{t,j}^{L}, N_{t,j}^{E}, N_{t,j}^{W}, N_{t,j}^{P}\right) - 1\right) & \text{, if } q_{t,j\to k}^{4}=1 \\ \left(C_k \updownarrow; 0\right) & \text{, otherwise} \end{cases} \\ q_{t,j\to k}^{4} = \mathbb{I}_{\left\{q_{t,j}^{1}=0\right\}}\mathbb{I}_{\left\{q_{t,j}^{2}=0\right\}}\mathbb{I}_{\left\{q_{t,j}^{5}=0\right\}}\mathbb{I}_{\{k=\xi(j)\}}, \\ q_{t,j}^{5} = \mathbb{I}_{\left\{N_{t,j}^{L}\le\mathbb{T}_{L}^{H}\right\}}\mathbb{I}_{\left\{N_{t,j}^{E}\le\mathbb{T}_{E}^{H}\right\}}\mathbb{I}_{\left\{N_{t,j}^{W}\le\mathbb{T}_{W}^{H}\right\}}\mathbb{I}_{\left\{N_{t,j}^{P}\le\mathbb{T}_{P}^{H}\right\}} \end{array} \tag{6}$$

where $\xi(j)$ is the cell towards which cell j has the biggest number of too late HOs (in case there are several such cells, it picks one randomly) and:

$$\begin{array}{l} \varphi'(x,y,z,t) = \varphi\Big(\max\left\{\frac{x}{\mathbb{T}_{L}^{H}} - 1; 0\right\} + \max\left\{\frac{y}{\mathbb{T}_{E}^{H}} - 1; 0\right\} + \\ \max\left\{\frac{z}{\mathbb{T}_{W}^{H}} - 1; 0\right\} + \max\left\{\frac{t}{\mathbb{T}_{P}^{H}} - 1; 0\right\}; g_{MRO}^{C\uparrow}, m_{MRO}^{C\uparrow}\Big) \end{array} \tag{7}$$

is a dedicated function that first sees by how much the evaluated KPIs are relatively above their associated "high" threshold.: $\frac{N_{t,j}^{(\cdot)}}{\mathbb{T}_{(\cdot)}^{H}} - 1$, next the negative values are replaced by 0 using the $\max\{\cdot, 0\}$ function and finally it does the sum-up and transposes the result in a value in the $[0, 1]$ interval using $\varphi'(.)$.

As you can see the request formulated by the MRO instance for the CIO sums the "happiness" of the neighboring eNBs regarding its own CIO value. By increasing the CIO value of one eNB we target to reduce the number of too late HOs to that eNB from the neighboring eNBs. The requests concerning the hysteresis do not contain any cell happiness indication because we will choose to accept and execute all of them.

III. SON Coordination

A. SON coordination description

We consider that all the MLB and MRO instances are synchronized, i.e. they do their KPI evaluation within a time window T and they send the requests simultaneously to the SONCO at the end of the time window (fig. 1) . The SONCO has the same time granularity T. It will decide which requests it will accept (and immediately execute) and which it will deny. Although the operator sees the SON instances as black-boxes, it can know their inputs and outputs. Therefore as we can see in the MLB and MRO description the CIO is a parameter on which their instances may have a conflict. The task of the SONCO is to provide a reasonable conflict resolution.

The indicator on how happy a SON instance is with the current parameter settings contains an important piece of information. In order to properly solve a conflict the SONCO has to know how critical the request is. For example: a request to offload ($C_i \downarrow$) should have a higher priority when it is done at a load of 99% as compared to when it is done at a load of 90%. On the same note if we consider 2 SON instances pulling the parameter change in opposite directions than the SONCO should compare their *strength* (in other words evaluating their happiness) and decide which one should win. We can also bias the decision by attributing weights to the SON instances.

Having the information on how happy the SON instances are with the current configuration cannot reflect how happy they will be in another state. For this we use RL. RL allows us to keep track of our past decisions through value functions which reflect how satisfied we were with a configuration. Temporal Difference (TD) learning is one of the RL techniques that provides the means of updating the parameter estimates every time we receive a new piece of information.

B. Markov Decision Process

Consider $i \in \{1, \ldots, N\}$ to be the index of an arbitrary eNB. Let $C_{t,i}$ and $H_{t,i}$ be the CIO and hysteresis of cell i at time instant t. Next we define the request summaries built using the requests from the SON instances (eq. 3, 4 and 5). They reflect if there exist any requests to increase/decrease (from MLB and/or MRO instances) the CIO and Hysteresis:

$$\begin{array}{l} C_{t,i}^{+} = 1 - \mathbb{I}_{\left\{U_{t,i}^{MLB(C)}[1]\neq C_i\uparrow\right\}}\mathbb{I}_{\left\{U_{t,i}^{MRO(C)}[1]\neq C_i\uparrow\right\}}, \\ C_{t,i}^{-} = \mathbb{I}_{\left\{U_{t,i}^{MLB(C)}[1]= C_i\downarrow\right\}}, \\ H_{t,i}^{+} = \mathbb{I}_{\left\{U_{t,i}^{MRO(H)}[1]= H_i\uparrow\right\}}, H_{t,i}^{-} = \mathbb{I}_{\left\{U_{t,i}^{MRO(H)}[1]= H_i\downarrow\right\}} \end{array} \tag{8}$$

Let $\mathcal{C}$ be the set of possible CIO values and $\mathcal{H}$ the set of possible Hysteresis values (for practical reasons we assume them to be sorted in an ascending order). Accepting to increase/decrease one of the parameters simply means to set the parameter to the next upper/lower value. Note that $\mathcal{C}$ and $\mathcal{H}$ are finite sets; therefore all the requests to go outside these sets will be replaced by a request to maintain that parameter: $(C_i \updownarrow; 0)$ for C_i and $(H_i \updownarrow; \emptyset)$ for H_i.

We denote: $C_t = (C_{t,i}, \ldots, C_{t,N})$, $H_t = (H_{t,1}, \ldots, H_{t,N})$, $C_t^{+} = \left(C_{t,1}^{+}, \ldots, C_{t,N}^{+}\right)$, $C_t^{-} = \left(C_{t,1}^{-}, \ldots, C_{t,N}^{-}\right)$, $H_t^{+} = \left(H_{t,1}^{+}, \ldots, H_{t,N}^{+}\right)$ and $H_t^{-} = \left(H_{t,1}^{-}, \ldots, H_{t,N}^{-}\right)$. Next we can define the underlying Markov Decision Process (MDP):

- **State space** $\mathcal{S} = \mathcal{C}^N \times \mathcal{H}^N \times \{0,1\}^{4N}$. A state $s \in \mathcal{S}$ is composed of 6 N-uplets $s = (c, h, c^+, c^-, h^+, h^-)$: the current CIOs, Hystereses and request summaries. We refer to the state of the MDP at time t as the random variable (r.v.) $S_t = \left(C_t, H_t, C_t^+, C_t^-, H_t^+, H_t^-\right)$.
- **Action space** $\mathcal{A} = \{-1, 0, 1\}^{2N}$. An action $a \in \mathcal{A}$ is composed of 2 N-uplets $a = \left(a^C, a^H\right)$ where $a^C, a^H \in \mathcal{A}^{\dagger} = \{-1, 0, 1\}^N$. We define the r.v. $A_t = \left(A_t^C, A_t^H\right) =$

$\left(\left(A^C_{t,1},\ldots,A^C_{t,N}\right),\left(A^H_{t,1},\ldots,A^H_{t,N}\right)\right) \in \mathcal{A}$ ($A^C_t, A^H_t \in \mathcal{A}^\dagger$) as the action of the SONCO at time t. The impact on parameter $X_i \in \{C_i, H_i\}, \forall i \in \{1,...,N\}$, is:

$$\begin{cases} X_{t,i}\uparrow & \text{, if } \mathbb{I}_{\left\{A^X_{t,i}=1\right\}}\mathbb{I}_{\left\{X^+_{t,i}=1\right\}}=1 \\ X_{t,i}\downarrow & \text{, if } \mathbb{I}_{\left\{A^X_{t,i}=-1\right\}}\mathbb{I}_{\left\{X^-_{t,i}=1\right\}}=1 \\ X_{t,i}\updownarrow & \text{, otherwise} \end{cases} \quad (9)$$

- **Transition kernel**. We endow the MDP with a probability transition kernel $\mathbb{P}(S_{t+1}=s'|S_t=s,A_t=a)$ quantifying the probability of going to state s' when the current state-action pair is (s,a). The kernel is supposed to be unknown. However, as you can see from eq. (9) the future CIOs (c') and Hystereses (h') configuration are deterministic functions of the the current state and the action:

$$c' = \mathcal{T}_C\left(c, c^+, c^-, a^C\right) \text{ and } h' = \mathcal{T}_H\left(h, h^+, h^-, a^C\right) \quad (10)$$

- **Reward** $r(s,a)$. The aim of the SONCO is to solve conflicting requests between SON instances. We define a reward based on the happiness indicators of the SON instances:

$$r_t = w_{MLB}\sum_{i=1}^{N} U^{MLB(C)}_{t,i}\,[2] + w_{MRO}\sum_{i=1}^{N} U^{MRO(C)}_{t,i}\,[2] \quad (11)$$

where w_{MLB} and w_{MRO} are weights associated to the MLB and MRO instances respectively. Note that it is independent of the action, therefore in the sequel we use the notation $r(s)$.

Let π denote the probability transition kernel on $\mathcal{S}\times\mathcal{A}$, where $\pi(s,\{a\}) = \mathbb{P}_\pi(A_t=a|S_t=s)$ is the probability to take action a when the current state is s. For any policy π, RL defines a value function that measures the performance of the policy as:

$$V^\pi(s) = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty}\gamma^t r_t\,|S_0=s\right], \quad (12)$$

where $0<\gamma\leq 1$ is the reward sum discount factor. If a policy $\pi^\star$ maximizes the value function $\forall s\in\mathcal{S}$ then it is said to be *optimal* [9].

C. Q-Learning

The optimal policy is usually obtained through Q-Learning. Q-learning requires a huge state (and action) space with a cardinality of: $|\mathcal{S}|\cdot|\mathcal{A}| = |\mathcal{C}|^N\cdot|\mathcal{H}|^N\cdot|\{0,1\}|^{4N}\cdot|\{-1,0,1\}|^{2N}$. For example if $|\mathcal{C}|=|\mathcal{H}|=5$ and $N=7$ the state-action space size is $\sim 7.8\,.\,10^{24}$. This is not a feasible implementation. The literature provides a wide variety of techniques to reduce the complexity like function approximation, state space aggregation [9] etc.. In our approach we follow the later while employing a General Policy Iteration (GPI) [13], thus we reduce the state space cardinality to $|\mathcal{C}|^N = 78125$.

D. General Policy Iteration

We use an actor-critic based GPI. While the actor applies the policy the critic evaluates it and thus an actor-updater will enhance the policy in order to obtain better performance.

1) The critic: In order to evaluate the value function V^π for a fixed policy π we can use th following recursion:

$$V_{t+1}(s) = V_t(s) + \alpha\left(r_t + \gamma V_t(S_{t+1}) - V_t(S_t)\right)\mathbb{I}_{\{S_t=s\}} \quad (13)$$

where α is a fixed step-size. For α small enough V_t is known to converge in the mean to V^π [9].

In order to perform a *state aggregation* we denote by $\mathcal{F}$ the set of functions $V:\mathcal{S}\to\mathbb{R}$ such that for any $s=(c,h,c^+,c^-,h^+,h^-)$, $V(s)$ only depends on c, namely:

$$\mathcal{F} = \left\{\left(c,h,c^+,c^-,h^+,h^-\right)\mapsto W(c) : W\in\mathbb{R}^{\mathcal{C}}\right\} \quad (14)$$

Thus the update in equation 13 is replaced by:

$$W_{t+1}(c) = W_t(c) + \alpha\left(r_t + \gamma W_t(C_{t+1}) - W_t(C_t)\right)\mathbb{I}_{\{C_t=c\}} \quad (15)$$

where the sequence of estimates $\tilde{V}_t : (c,h,c^+,c^-,h^+,h^-) \mapsto W_t(c)$ converges in the mean to a fixed point of the so-called Bellman operator *projected onto* $\mathcal{F}$ [9].

2) The actor: Using (10) we get

$$\tilde{V}_t(S_{t+1}) = W_t(C_{t+1}) = W_t\left(\mathcal{T}_C\left(C_t, C^+_t, C^-_t, A_t\right)\right) \quad (16)$$

At time t the best possible action is then:

$$A^*_t = \underset{a=(a^C;a^H)\in\mathcal{A}}{\arg\max}\; W_t\left(\mathcal{T}_C\left(C_t, C^+_t, C^-_t, a^C\right)\right) \quad (17)$$

Please note that the solution $A^*_t = \left(A^{C,*}_t; A^{H,*}_t\right)$ in (17) is independent of the argument component a^H. Thus we can rewrite (17) as:

$$A^*_t = \left(\underset{a^C\in\mathcal{A}^\dagger}{\arg\max} W_t\left(\mathcal{T}_C\left(C_t, C^+_t, C^-_t, a^C\right)\right); \forall a^H\in\mathcal{A}^\dagger\right) \quad (18)$$

Therefore we establish to always accept the requests on the Hystereses:

$$A^H_{t,i} = A^{H,\checkmark}_{t,i} = H^+_{t,i} - H^-_{t,i}, \forall i\in\{1,...,N\} \quad (19)$$

and for practical reasons to use an ϵ-greedy policy w.r.t. A^C_t ($\epsilon > 0$ defines a tradeoff between exploitation and exploration), so the policy can be expressed as:

$$\pi(S_t,\{a\}) = \left((1-\epsilon)\,\mathbb{I}_{\left\{a^C=A^{C,*}_t\right\}} + \frac{\epsilon}{|\mathcal{A}^C|}\right)\mathbb{I}_{\left\{a^H=A^{H,\checkmark}_t\right\}} \quad (20)$$

We are aiming to have a reward that reflects how satisfying the CIO configuration is, independent to some extent of the Hysteresis thus explaining our choice of the reward in (11).

The proposed algorithm is summarized in Alg. 1 where we have extended our framework to eligibility traces (see [9]). **Function Init** should be called for the initialization of the

Algorithm 1 $SONCO\,(\lambda)$

Function Init:

Initialize $W\,(c) = 0$ *and* $e\,(c) = 0$ *for all* $c \in \mathcal{C}$

Initialize $s = ((0,0,0),(0,0,0))$

Function SONCO:

Observe current state $s' = (c', u')$ *and*
calculate reward $r\,(s')$

$\delta \leftarrow r + \gamma W\,(c') - W\,(c)$

$e\,(c) \leftarrow e\,(c) + 1$

for all $\tilde{c} \in \mathcal{C}$

$W\,(\tilde{c}) \leftarrow W\,(\tilde{c}) + \alpha\delta e\,(\tilde{c})$

if $a'^C = a^{C,opt}$ *then* $e\,(\tilde{c}) \leftarrow \gamma\lambda e\,(\tilde{c})$

else $e\,(\tilde{c}) \leftarrow 0$

Choose action $a' = \left(a'^C, a'^H\right)$ *using policy* π,

Calculate $a^{C,opt}$,

Take action a',

$s \leftarrow s' (\equiv (c,u) \leftarrow (c',u'))$

algorithm and **Function SONCO** should be called every time after receiving the requests of the SON instances (MLB and MRO).

IV. Simulation Results

A. Simulation scenario

The scenario is built on $N = 7$ cells placed on 4 sites (fig. 2). The simulation details are summarized in Table I. For the user arrivals, we consider a general background traffic arrival rate η_G [Mb/s] together with an additional hotspot (HS) arrival rate η_{HS} [Mb/s] (such that the resulting arrival rate in the HS is η_G+η_{HS}). The user arrivals rates per area unit can be easily obtained as: $\rho_G\left[UE/s/m^2\right] = \eta_G[Mb/s]/S_G\left[m^2\right]/FS[Mb/UE]$ and $\rho_{HS}\left[UE/s/m^2\right] = \eta_{HS}[Mb/s]/S_{HS}\left[m^2\right]/FS[Mb/UE]$ (S refers to the area and FS to the file size). We use Space Poisson Point Processes for the user arrivals. A user will arrive in the network, transmit its file and then leave the network.

The user scheduling is done in a Proportional Fair (PF) manner with a sub-frame by sub-frame (1 ms) granularity, therefore only one user may be scheduled in a sub-frame. The simulator temporal granularity is 1ms. Inter-cell interference is calculated as follows: if a neighboring eNB has at least one user that can be scheduled for transmission (the user is not in the Connection Establishment/ Reestablishment/ Reconfiguration mode) than it is assumed that during the next sub-frame the interference received from that eNB is maximum. Thus we can assume perfect knowledge on the inter-cell interference. Practically this is impossible, however the users continuously inform the eNBs of the channel conditions through the Channel Quality Indicator (CQI) [14].

The behavior of the user in IDLE mode is not modeled but we assume them to be camped according to (1). The SON instances (MLB and MRO) and the SONCO are active during the entire simulation.

Table I
Simulation parameters

Category	Parameter	Value
Network	Inter Site Distance	1732 m
Channel modeling	Carrier frequency	2 GHz
	Bandwidth	10 MHz
	OFDM data symbols	11 out of 14
	eNB TX Power	46 dBm
	Propagation Model	3GPP Case 3 [15]
	Channel Model	MIMO 2×2
Mobility	HO Time To Trigger	160 ms
	H0 Hysteresis ($\mathcal{H}$[dB])	$\{0,3,6,9,12\}$
	CIO values ($\mathcal{C}$ [dB])	$\{-12,-9,-6,-3,0\}$
	Radio Resource Control	3GPP [14]
SON functions	Time window T	5 min
	$\left(\mathbb{T}^L_{load};\mathbb{T}^H_{load}\right)$	$(0.5;0.8)$
	$\left(\mathbb{T}^L_L;\mathbb{T}^H_L\right)$	$(1.25;5)$
	$\left(\mathbb{T}^L_E;\mathbb{T}^H_E\right) = \left(\mathbb{T}^L_W;\mathbb{T}^H_W\right)$	$(\infty;\infty) \equiv$ off
	$\left(\mathbb{T}^L_P;\mathbb{T}^H_P\right)$	$(2.5;10)$
	$\left(g^{C\downarrow}_{MLB};m^{C\downarrow}_{MLB}\right)$	$(30;0.9)$
	$\left(g^{C\uparrow}_{MLB};m^{C\uparrow}_{MLB}\right)$	$(20;0.4)$
	$\left(g^{C\uparrow}_{MRO};m^{C\uparrow}_{MRO}\right)$	$(20;0)$
SONCO	Learning rate α	0.2
	Discount factor γ	0.8
	Eligibility traces decay rate λ	$[0,1]$
	Epsilon greedy parameter ϵ	0.1

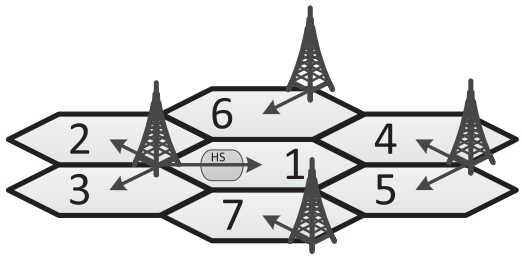

Figure 2. Network topology

B. Simulation results

The simulator parameters have been set to bring out the impact of different configurations of the SONCO. For this we consider a file size of $FS = 16[Mb/UE]$.The user arrival rates are $\eta_G = 11[Mb/s]$ and $\eta_{HS} = 33[Mb/s]$. Initially we set $\lambda = 0$. We try out several sets of weight pairs $w = (w_{MLB}, w_{MRO})$ to see the impact on the KPIs. The total simulation duration is 48 hours where the first 24 hours period is considered as a warmup period, and the the KPIs are calculated over the last 24 hours period.

In fig. 3 we plot the maximum and the average (over all eNBs) of the time-averaged loads (over the last 24 hours of the simulations). One can see how giving bigger weights to the MLB instances allows to reduce the maximum time-averaged load value, achieving a better load balancing. In our case eNB 1 is the overloaded cell and we reduce its load (significantly more when MLB has a bigger priority) at the expense of slightly increasing the loads of the other eNBs.

Fig. 4 shows the impact of different weights (w) on the time-averaged number of Too Late HOs : we look again at the maximum and the average values over eNBs 1 to N . As

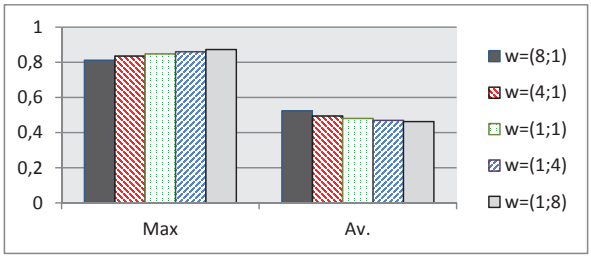

Figure 3. Average Load $SONCO\,(\lambda = 0.0)$

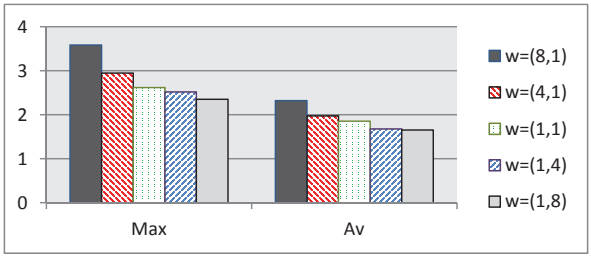

Figure 4. Average Number of Too Late HOs $[\#/\text{min}]$ $SONCO\,(\lambda = 0.0)$

expected, giving higher priorities to the MRO tends to better-solve the issue of the Too Late HOs: the maximum value is decreased because the MLB of eNB 1 is not allowed to offload as much as it would want (i.e. to decrease the CIO of eNB 1). Thus the HO borders to and from neighboring base stations are pushed *further away* from eNB 1. This reduces the number of Too Late HOs from neighboring base stations towards eNB1 at the cost of slightly increasing the number of Too Late HO from eNB1 towards its neighbors.

Note that the number of Too Early HOs and Wrong Cell HOs are neglected because there are very few events of these types. Next we check the maximum and the average number of ping pongs, which is plotted in fig 5. It would be expected that the number of ping pongs depends mostly on the Hystereses. This is true but in fig. 5 we can also see that the weights (and thus the CIO configuration) have also an impact on the number of Ping-Pongs. Reducing the number of Too Late HOs (by tuning the CIOs) allows a better optimization of the Hysteresis such that, better performances in terms of number of ping pongs may also be achieved indirectly.

We have obtained similar results for $\lambda \neq 0$. It has been proven that λ has a significant impact on the convergence time of the algorithm[9]. However, this issue is outside the scope of this paper.

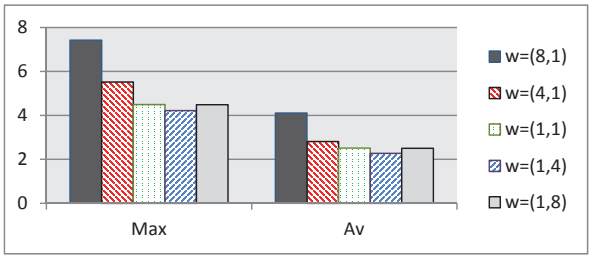

Figure 5. Average Number of Ping Pongs $[\#/\text{min}]$ $SONCO\,(\lambda = 0.0)$

V. Conclusions and Future Work

In dealing with the conflict resolution between the requests of the SON instances (MLB and MRO), RL proves to have the qualities that allow us to tune the arbitration in favor of one or the other with respect to priorities/weights defined by the operator. By using a SON Coordinator we were able to intelligently decide when to accept/deny the requests of the SON instances in order to reach the desired tradeoff between the two SON functions w.r.t. to their corresponding targets (avoiding over-loads for MLB versus decreasing the number of connection failures and ping-pongs due to mobility for MRO). Simulation results show that the resulting KPIs reflect the operator priorities. Future work will focus on analyzing and improving the scalability of our solution.

Acknowledgment

The presented work was carried out within the FP7 SEMAFOUR project [16] which is partially funded by the Commission of the European Union.

References

[1] S. Sesia, I. Toufik, and M. Baker, *LTE - The UMTS Long Term Evolution: From Theory to Practice 2nd Edition*. Wiley, 2011.

[2] S. Hämäläinen, H. Sanneck, and C. Sartori, *LTE Self-Organising Networks (SON): Network Management Automation for Operational Efficiency*. Wiley, 2011.

[3] L. Schmelz, M. Amirijoo, A. Eisenblaetter, R. Litjens, M. Neuland, and J. Turk, "A coordination framework for self-organisation in lte networks," in *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on*, 2011, pp. 193–200.

[4] A. Lobinger, S. Stefanski, T. Jansen, and I. Balan, "Coordinating handover parameter optimization and load balancing in lte self-optimizing networks," in *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*, 2011, pp. 1–5.

[5] T. Bandh, H. Sanneck, and R. Romeikat, "An experimental system for son function coordination," in *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*, 2011, pp. 1–2.

[6] T. Bandh, R. Romeikat, H. Sanneck, and H. Tang, "Policy-based coordination and management of son functions," in *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on*, 2011, pp. 827–840.

[7] Z. Liu, P. Hong, K. Xue, and M. Peng, "Conflict avoidance between mobility robustness optimization and mobility load balancing," in *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, 2010, pp. 1–5.

[8] J. Chen, H. Zhuang, B. Andrian, and Y. Li, "Difference-based joint parameter configuration for mro and mlb," in *Vehicular Technology Conference (VTC Spring), 2012 IEEE 75th*, 2012, pp. 1–5.

[9] R. Sutton and A. Barto, *Reinforcement Learning: an introduction*, ser. A Bradford book. A Bradford Book, 1998.

[10] R. Combes, Z. Altman, and E. Altman, "Self-organizing relays: Dimensioning, self-optimization, and learning," *Network and Service Management, IEEE Transactions on*, vol. 9, no. 4, pp. 487–500, 2012.

[11] M. ul Islam and A. Mitschele-Thiel, "Reinforcement learning strategies for self-organized coverage and capacity optimization," in *Wireless Communications and Networking Conference (WCNC), 2012 IEEE*, 2012, pp. 2818–2823.

[12] 3GPP, "LTE; E-UTRA and E-UTRAN; Overall description," 3rd Generation Partnership Project (3GPP), TS 36.300, 2013.

[13] C. Szepesvári, *Algorithms for Reinforcement Learning*, ser. G - Reference,Information and Interdisciplinary Subjects Series. Morgan & Claypool, 2010.

[14] 3GPP, "LTE; E-UTRA; Radio Resource Control; Protocol specification," 3rd Generation Partnership Project (3GPP), TS 36.331, 2012.

[15] ——, "E-UTRA; Further advancements for E-UTRA physical layer aspects," 3rd Generation Partnership Project (3GPP), TR 36.814, 2010.

[16] SEMAFOUR project web page http://fp7-semafour.eu/.