Coordinating SON Instances: A Reinforcement Learning Framework

Ovidiu Iacoboaiea, Berna Sayrac, Sana Ben Jemaa
Orange Labs
Issy les Moulineaux, France
{ovidiu.iacoboaiea,berna.sayrac,sana.benjemaa}@orange.com

Pascal Bianchi
Telecom ParisTech
Paris, France
pascal.bianchi@telecom-paristech.fr

***Abstract*—In Long Term Evolution(LTE) networks one of the main focuses is on automating the network optimization. This is done through so called Self Organizing Network (SON) functions like Mobility Load Balancing (MLB), Mobility Robustness Optimization(MRO) and others. A SON instance is a realization of a SON function that governs (optimizes) one or a cluster of eNBs. The SON functions are built in a standalone manner, i.e. without considering the existence of other SON instances. So they do not necessarily operate in a coordinated fashion, especially in a network where different SON instances may come from different vendors. Thus we face a risk of generating conflicts and instabilities in the network and so this raises the need for a SON COordinator (SONCO) . The SONCO, built from an operator point of view, sees the SON instances as black boxes and has a very limited amount of information on them. In these conditions the SONCO has to solve conflicts and improve the network stability. In this paper we propose a Reinforcement Learning (RL) based solution for coordinating SON instances that run independently on neighboring eNBs and we provide results for a case study with MLB instances. We analyze the scalability of our solution and we provide numerical results showing how improvements in network stability can be obtained.**

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

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 2^{nd} which targets a run-time optimization of the network. In the rest of the paper, SON will refer to self-optimization.

SON functions, like Mobility Load Balancing (MLB), Mobility Robustness Optimization (MRO), Coverage and Capacity Optimization (CCO) and others, are meant to automate network tuning in order to obtain better performances throughout the network. A SON instance is a realization of a SON function that governs (optimizes) one or a cluster of eNBs. The SON functions have been built in a standalone manner, i.e. without considering the existence of other SON instances. So they do not necessarily operate in a coordinated fashion, especially in a multi-vendor network where different SON instances may come from different vendors. This may lead to conflicts and unstable network behavior. A SON COordination (SONCO) mechanism is then required to detect and prevent conflicts and instabilities. There are two dimensions that have to be considered for the SONCO functions:

- The SON instance dimension: where the SON instances of the same SON function (ex. MLB) need to be coordinated,
- The SON functionality dimension: where the coordination needs to be done between SON instances of different SON functions like MLB, MRO, CCO and others.

The interest for SON coordination has started fairly recent, but it is rapidly growing. SON conflict resolution started getting attention with Rel.10 of 3GPP [2]. In the literature, SON coordination has been addressed in several papers. In [3] and [4] the coordination of MRO and MLB is performed by attributing priorities to these SON functions. In [5] and [6], a decision tree is used for tuning the Remote Electrical Tilt (RET) and Transmission Power (TP) values. [7] focuses on conflict avoidance through creating restrictions for the value set of the parameters for MRO and MLB. In [8], the authors propose a per user optimization that attributes priorities to users for handovers based on the weights attributed to the SON functions (MRO and MLB). All these coordination approaches use the instantaneous network information and do not benefit from the past experiences. However, it is well-known that the past information constitutes a very important piece of information in finding the optimum course of action [9]. Therefore in this paper, we adopt an approach where the SONCO learns the optimal coordination decisions from the consequences of its past decisions. A very effective and well known technique to learn from past experience is Reinforcement Learning (RL) [9]. The efficiency of RL has been shown through its extensive use in designing self-optimization functions, e.g. for tuning relay-backhaul resource allocation parameters in [10], for tuning the CCO parameters in [11] etc.

An important point to highlight is that in this paper, we assume a "black-box" approach: the SONCO has no information on the characteristics/parameters of the SON instances that it coordinates (including the algorithms inside). In other words, the SON instances are "black boxes" for the SONCO. This is a plausible approach from an operator point of view, where the operator may not have access to the inside information of the SON instances. Note that this approach is different from other existing ("white box") approaches which assume to have

DOI: 10.1109/VTCFall.2014.6965923

the inside information on the SON instances ([12]).

Improving the network stability (i.e. eliminating unnecessary parameter changes) could be done by forbidding any parameter changes, but this would prevent SON functions from doing their job. We need a mechanism that finds the optimal parameter configuration and steers the network configuration into that direction. The major contributions of this paper can be summarized as follows:

- We use a RL-based SONCO solution which finds the best parameter configuration (the one that maximizes a predefined reward) and eliminates changes that divert us from this configuration, while still exploring occasionally other potentially good configurations.
- The SON instances coordinated by the proposed SONCO are considered as black-boxes (i.e. no information is passed from the SON instances to the SONCO, except the requests). This represents an operator-centric solution.
- We analyze the scalability issue by comparing distributed and centralized implementations.

The performance of the proposed SONCO is evaluated using MLB instances over a network segment consisting of 12 eNBs.

The remainder of this paper is structured as follows: Section II provides the system description presenting the scenario and the SON and SONCO instances; Section III outlines the Markov Decision Process (MDP) underlying the RL solution; Section IV presents the RL algorithm; simulation results are included in Section V and Section VI concludes the paper.

II. System Description

We consider a network segment composed of N cells. On each cell $i \in \mathcal{N} = \{1, ..., N\}$, we have only one independent SON instance running (see Fig. 1) thus the index i also identifies the SON instance.

The SON instances are governed by M SONCO instances. A SONCO instance $j \in \mathcal{M} = \{1, ..., M\}$ governs the SON instances running on N_j eNBs (in our case we only have one SON instance per eNB). We define $\mathcal{N}_j = \left\{i_1^j, \ldots, i_{N_j}^j\right\}$ the set of eNBs governed by SONCO j and we consider the following: $\bigcup_{j \in \mathcal{M}} \mathcal{N}_j = \mathcal{N}$ and $\mathcal{N}_{j_1} \cap \mathcal{N}_{j_2} = \emptyset$, $\forall j_1 \neq j_2 \in \mathcal{M}$, in other words all eNBs are governed by one and only one SONCO instance.

We consider all the SON instances, no matter under which coordinator they are, to be synchronized, i.e. they do their KPI evaluation within a time interval T and they send their requests simultaneously to the governing SONCO instance (Fig. 1) at the end of the time interval . The SONCO instances have the same time granularity T. They decide which requests will be accepted and which will be denied. The accepted requests are assumed to be immediately executed.

The output of SON instance i is a request to modify the parameter P (e.g. Cell Individual Offset (CIO)) of cell i: $U_{t,i} \in \{P\uparrow, P\downarrow, P\updownarrow\}$ where $P\uparrow$,$P\downarrow$ and $P\updownarrow$ means increase, decrease and maintain the value of P_i respectively. We denote $P_{t,i}$ the value of parameter P on cell i at time t and $\mathcal{P}$ the set of possible parameter values (we consider the

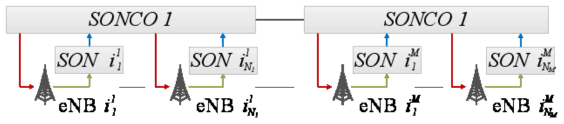

Figure 1. Functional Block Diagram: SONCO↔SON (MLB) interactions

same set for all cells). This can be easily extended to a set of parameters instead of only one parameter.

In this paper we use a coordination scheme in which the operator-centric SONCO j ($\forall j \in \mathcal{M}$) takes the decisions based on the observation of the current parameter values ($P_{t,i}$,$\forall i \in \mathcal{N}_j$) and the current update requests of the SON instance ($U_{t,i}$,$\forall i \in \mathcal{N}_j$). It does not know the optimization algorithms inside the SON instances (thresholds, inputs etc.).

The purpose of the SONCO instances is to find the most rewarding configuration (the reward is defined later on), to steer the network towards this configuration and to prevent as much as possible diverting from it.

The RL algorithm attributes a value function to the network states (parameter configurations). The value function of a state is a measure of how rewarding that state is expected to be given some underlying policy. Particularly, by using the Temporal Difference (TD) learning method [9], we are able to have online updates and continuously improve the estimate of the value function. The MDP underlying the RL algorithm is describe in the sequel.

III. Markov Decision Processes

A. General framework

Consider one SONCO instance governing all cells ($|\mathcal{M}| = 1$). We denote $P_t = (P_{t,i})_{i \in \mathcal{N}} \in \mathcal{P}^N$ and $U_t = (U_{t,i})_{i \in \mathcal{N}} \in \mathcal{U}^N$. We can now state the MDP model underlying our algorithm.

- **State space** $\mathcal{S} = \mathcal{P}^N \times \mathcal{U}^N$. A state $s \in \mathcal{S}$ is composed of 2 N-uplets $s = (p, u)$: the current parameter values and update requests .
- **Action space** $\mathcal{A} = \{0, 1\}^N$. An action $a \in \mathcal{A}$ is a N-uplet of binary variables. (1=accept, 0=deny)
- **Transition kernel**. $\mathcal{T}(s'|s, a)$ the probability of going to state s' when the current state-action pair is (s, a).
- **Reward.** $r(s, a)$ is the reward associated to the state-action pair (s, a)

A policy π is a transition kernel π on $\mathcal{S} \times 2^{\mathcal{A}}$, such that $\pi(s, \{a\})$ represents the probability to take action a when the current state is s.

Consider a stochastic process $(S_t, A_t)_{t \in \mathbb{N}} \in \mathcal{S} \times \mathcal{A}$ where S_t and A_t represent the state and action at time t respectively. Set $S_t = (P_t, U_t)$.

For any policy π introduce a probability $\mathbb{P}_\pi$ such that $(S_t, A_t)_{t \in \mathbb{N}}$ is a Markov chain under $\mathbb{P}_\pi$ thus:

$$\mathbb{P}_\pi (S_{t+1} = s' | S_t = s, A_t = a) = \mathcal{T}(s'|s, a), \tag{1}$$

$$\mathbb{P}_\pi (A_t = a | S_t = s) = \pi(s, a). \tag{2}$$

Note that we shall simply use the notation $\mathbb{P}$ instead of $\mathbb{P}_\pi$ in eq. (1) i.e. when the probability does not depend on π.

B. Assumptions

The transition kernel is not a general one. We know that the future value of the parameter (P_{t+1}) is a deterministic function of the current state-action pair (S_t,A_t). Also we know the update requests (U_{t+1}) given the parameter configurations (P_{t+1}) are independent of past state (S_t) and action (A_t). We formalize this in the sequel.

Assumption 1 (kernel)**.**

1) *There exists $g : \mathcal{S} \times \mathcal{A} \to \mathcal{P}^N$ s.t. $P_{t+1} = g(S_t, A_t)$ with probability 1.*
2) $\mathbb{P}(U_{t+1} = u' | S_t = s, A_t = a, P_{t+1} = p') = \mathbb{P}(U_{t+1} = u' | P_{t+1} = p')$.

More formally it means we assume that the kernel $\mathcal{T}$ admits the following decomposition:

$$\mathcal{T}((p', u') | s, a) = \begin{cases} q(u'|p') & \text{if } p' = g(s,a) \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $q(u'|p') \overset{(def)}{=} \mathbb{P}(U_{t+1} = u' | P_{t+1} = p')$.

Given some state-action pair the reward is a function of the *happiness* of the SON instances reflected in the succeeding update requests (i.e. whether or not they request changes).

Assumption 2 (reward)**.** *There exists $\rho : \mathcal{U}^N \to \mathbb{R}$ such that $r(s,a) \overset{(def)}{=} \mathbb{E}[\rho(U_{t+1}) | S_t = s, A_t = a]$.*

We now provide a specific form of ρ, relevant for the SON coordination. Consider first ρ' a function which attributes a per cell reward based on the update request of each cell (e.g. a big reward for the requests to maintain the parameter values), then consider ρ to be a function returning the minimum of the mentioned per cell rewards (the target is to maximize the minimum of the per cell rewards). In other words for $u = (u_i)_{i \in \mathcal{N}}$ we have:

$$\begin{aligned} &\rho(u) = \min_{i \in \mathcal{N}} \rho'(u_i) \\ &\rho'(x) = r_{\updownarrow} \mathbb{I}_{\{x = P\updownarrow\}} + r_{\uparrow} \mathbb{I}_{\{x = P\uparrow\}} + r_{\downarrow} \mathbb{I}_{\{x = P\downarrow\}} \end{aligned} \tag{4}$$

where $r_{\updownarrow}, r_{\uparrow}, r_{\downarrow} \in [0,1]$ (the closer to zero the more the probability of receiving that request is reduced) and $\mathbb{I}_{\{\cdot\}}$ is the indicator function ($\mathbb{I}_{\{true\}} = 1$, $\mathbb{I}_{\{false\}} = 0$).

Finally we introduce a reward process $(R_t)_{t \in \mathbb{N}}$ such that $\mathbb{E}[R_{t+1} | S_{0:t}, A_{0:t}] = r(S_t, A_t)$, $\forall t$.

C. Optimal policy

For any policy π we introduce the value function:

$$V^{\pi}(s) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} | S_0 = s\right] \ , \tag{5}$$

where $0 \le \gamma < 1$ is the reward sum discount factor and $\mathbb{E}_{\pi}$ is the expectation given that the policy is π. If a policy π^* maximizes the value function $\forall s \in \mathcal{S}$ then it is said to be *optimal* [9].

Note that π^* is known to be a deterministic policy i.e. $\pi^* : \mathcal{S} \to \mathcal{A}$ and we have (see [9]): $\pi^*(s) = \text{argmax}_a Q^*(s,a)$ where Q^* satisfies the fixed point equation :

$$\begin{aligned} \forall (s,a), \ & Q^*(s,a) = r(s,a) + \\ & \gamma \mathbb{E}[\max_{a'} Q^*(S_{t+1}, a') | S_t = s, A_t = a]. \end{aligned} \tag{6}$$

The following lemma allows to simplify (6). For any $s \in \mathcal{S}$, we define $\Delta_s = \{g(s,a) | a \in \mathcal{A}\}$ as the set of parameter configurations that are accessible from s by means of some action $a \in \mathcal{A}$. Also we define $\overline{r}(p) = \mathbb{E}[\rho(U_{t+1}) | P_{t+1} = p]$.

Lemma 1. *There exists a function $W^* : \mathcal{P}^N \to \mathbb{R}$ s.t. for any $(s,a) \in \mathcal{S} \times \mathcal{A}$, $Q^*(s,a) = W^*(g(s,a))$. Moreover, W^* solves the following fixed point equation:*

$$\begin{aligned} \forall p, \ W^*(p) = \overline{r}(p) + \gamma \textstyle\sum_{u \in \mathcal{U}^N} \mathbb{P}[U_{t+1} = u | \\ P_{t+1} = p] \cdot max_{\ p' \in \Delta_{(p,u)}} W^*(p'). \end{aligned} \tag{7}$$

Proof: By (6) $Q^*(s,a) = r(s,a) + \gamma \mathcal{Q}$ where:

$$\begin{aligned} \mathcal{Q} \ &= \mathbb{E}[\max_{a'} Q^*((P_{t+1}, U_{t+1}), a') | S_t = s, A_t = a] \\ &= \mathbb{E}[\mathbb{E}[\max_{a'} Q^*((P_{t+1}, U_{t+1}), a') | P_{t+1}, S_t = s, \\ &\qquad A_t = a] | S_t = s, A_t = a] \\ &\overset{A1.2)}{=} \mathbb{E}[\mathbb{E}[\max_{a'} Q^*((P_{t+1}, U_{t+1}), a') | P_{t+1}] | \\ &\qquad S_t = s, , A_t = a] \\ &\overset{A1.1)}{=} \mathbb{E}[\max_{a'} Q^*((P_{t+1}, U_{t+1}), a') | P_{t+1} = p] \end{aligned} \tag{8}$$

where $p = g(s,a)$. Using Assumptions 1 and 2 we easily get $r(s,a) = \overline{r}(p)$ through similar manipulations. Finally:

$$Q^*(s,a) = \overline{r}(p) + \gamma \mathbb{E}[\max_{a'} Q^*((p, U_{t+1}), a') | P_{t+1} = p]. \tag{9}$$

Thus there exists a function W^* on $\mathcal{P}^N$ that for any (s,a): $Q^*(s,a) = W^*(g(s,a))$. The conclusion of Lemma 1 follows simply by replacing Q^* with W^* in (9). ■

The following proposition is an immediate consequence of Lemma 1:

Proposition 1. *The optimal policy is obtained as:*

$$\forall s, \ \pi^*(s) = argmax_a W^*(g(s,a)). \tag{10}$$

IV. Reinforcement learning

A. Algorithm

In order to find the optimal policy in Proposition 1 we have to solve the fixed point equation in Lemma 1, but as we have only partial knowledge on the transition kernel this is not possible. Instead we estimate W^* in (7) using a sequence $(W_t)_{t \in \mathbb{N}}$ defined $\forall p$ by the following Robbins-Monro recursion:

$$\begin{aligned} W_{t+1}(p) = W_t(p) + \alpha \left(R_t + \gamma \max_a W_t(g(S_t, a)) - \right. \\ \left. W_t(P_t)\right) \mathbb{I}_{\{P_t = p\}} \end{aligned} \tag{11}$$

which for constant α converges in average to W^* ([9]).

In practice, it is beneficial to use an ϵ-greedy policy, defined for some $\epsilon > 0$ (representing the exploration-exploitation tradeoff):

$$\pi_t(s, \{a\}) = (1 - \epsilon) \mathbb{I}_{\{a = \text{argmax}_b W_t(g(s,b))\}} + \frac{\epsilon}{2^N}. \tag{12}$$

The proposed algorithm is summarized in Alg. 1. For each SONCO instance **Function Init** should be called for the initialization and **Function SONCO** should be called every time after receiving the requests of the SON instances.

Algorithm 1 *SONCO instance*

```
Function Init:
    Initialize $W(p) = 0\ \forall p$, (considering the eNB set $\mathcal{N}_j$)
Function SONCO:
    Observe current state $s = (p, u)$ and
        calculate reward $r = \rho(u)$
    Calculate $a^{opt} = \underset{b}{argmax} W(g(s, b))$ and
        $c^{opt} = g(s, a^{opt})$
    $W(p) \leftarrow W(p) + \alpha(r + \gamma W(p^{opt}) - W(p))$
    Choose action $a$ using an $\epsilon$-greedy policy $\pi$,
    Take action $a$.
```

B. Complexity analysis

The optimal policy is usually obtained through Q-Learning ([9]) with one SONCO instance that governs all eNBs. According to the previous section the optimal policy can also be obtained by learning W^* in (7) (for $|\mathcal{M}| = 1$). This allows us to reduce the required state (and action) space from a size of: $|\mathcal{S}| \cdot |\mathcal{A}| = |\mathcal{P}|^N |\mathcal{U}|^N \cdot |\{0,1\}|^N$ ($\sim 3.7.10^{16}$ for $|\mathcal{P}| = 4$ and $N = 12$) to a size of $|\mathcal{P}|^N$ ($\sim 1.7 \cdot 10^7$ for the same conditions). Still, this solution scales exponentially with N, but as mentioned earlier we test also sub-optimal approaches: we divide the area that we want to coordinate into sub-areas which will be governed by independent SONCO instances and thus the complexity becomes $\sum_j |\mathcal{P}|^{N_j}$ ($\sim 8.1 \cdot 10^3$ for $|\mathcal{P}| = 4$, $|\mathcal{M}| = 2$ and $N_j = 6$, $\forall j \in \mathcal{M}$). We analyze the impact of this sub-optimal division in the results section.

V. Simulation Results

A. Simulation scenario

To demonstrate the concept we use the MLB function described in the sequel. The MLB instances tune the Cell Individual Offset (CIO) parameter in order to optimize the cell load [13]. The users that wish to transmit data get attached to cell $k = \text{argmax}_{i \in \mathcal{N}}(RSRP_i + C_i)$ where $RSRP_i$ is the Reference Signal Received Power from cell i and C_i is the CIO of cell i. The input of an MLB instance is the cell load ($L_{t,i}$) calculated as the average number of occupied Physical Resource Blocks (PRBs) of the hosting cell (the cell on which the MLB instance runs) during the time interval $(t-1, t]$. The output is a request to change the CIO of the hosting cell ($P_i \leftarrow C_i$): $U_{t,i} \in \{C\uparrow, C\downarrow, C\updownarrow\}$. For simulation purposes we use the following algorithm as a typical MLB implementation:

$$U_{t,i} = \begin{cases} C\downarrow & \text{, if } L_{t,i} > \mathbb{T}^H_{load} \\ C\uparrow & \text{, if } L_{t,i} < \mathbb{T}^L_{load} \\ C\updownarrow & \text{, otherwise} \end{cases} \tag{13}$$

where $\mathbb{T}^H_{load}$ and $\mathbb{T}^L_{load}$ are fixed thresholds used by the MLB instance to trigger CIO update requests ($\mathbb{T}^H_{load} > \mathbb{T}^L_{load}$). The MLB instance sends: *off-load* requests ($C\downarrow$) when the cell is overloaded and *on-load* requests ($C\uparrow$) to get back to the default configuration when the traffic is no longer imbalanced.

Table I
SIMULATION PARAMETERS

Category	Parameter	Value
Network	Inter Site Distance	500 m
Channel modeling	Carrier frequency/ Bandwidth	2 GHz /10 MHz
	eNB TX Power	46 dBm
	Propagation Model	3GPP Case 1 [14]
	Channel Model	MIMO 2×2
	Radio Resource Control	3GPP [15]
SON	CIO values ($\mathcal{C}$ [dB])	$\{-9, -6, -3, 0\}$
	Time window T	2.5 min
	$(r_{\updownarrow}; r_{\uparrow}; r_{\downarrow})$	$(1; 0.9; 0)$
	$(\mathbb{T}^L_{load}; \mathbb{T}^H_{load})$	$(0.5; 0.8)$
SONCO	Learning rate α	0.05
	Discount factor γ	0.8
	Epsilon greedy parameter ϵ	0.1

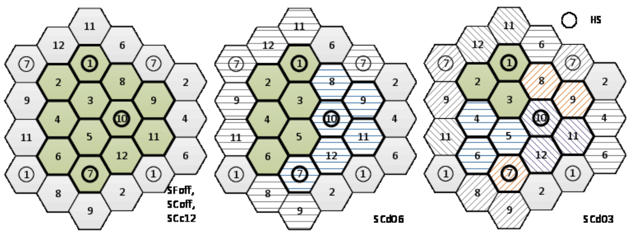

Figure 2. Network topology

The reward is calculated using (4) where the coefficients ($r\updownarrow$,$r\uparrow$ and $r\downarrow$) are fixed such that we favor the configurations where we do not receive *off-load* requests. Simulation details are summarized in Table I.

We use a hexagonal topology with $N = 12$ and wraparound (see Fig. 2), and we analyze 5 cases with varying degrees of clustering of the eNBs (or SON instances) with respect to the SONCOs, SCcX standing for a centralized SONCO deployment and SCdX for a distributed SON deployment with X coordinated SON instances:

SFoff) MLB instances are deactivated,
SCoff) MLB instances are activated, SONCO is off,
SCc12) $|\mathcal{M}| = 1$ and $\mathcal{N}_1 = \mathcal{N}$
SCd06) $|\mathcal{M}| = 2$, $\mathcal{N}_1 = \{1, .., 6\}$ and $\mathcal{N}_2 = \{7, .., 12\}$,
SCd03) $|\mathcal{M}| = 4$, $\mathcal{N}_1 = \{1, 2, 3\}$, $\mathcal{N}_2 = \{4, 5, 6\}$, $\mathcal{N}_3 = \{7, 8, 9\}$ and $\mathcal{N}_4 = \{10, 11, 12\}$.

These cases represent different degrees of scalability: the distributed ones are scalable whereas the centralized one is not. We consider a general background traffic arrival rate η_G [Mb/s] together with an additional hotspot (HS, see Fig. 2) arrival rate η_{HS} [Mb/s] (the resulting HS arrival rate is η_G+η_{HS}). The user arrivals rates per area unit can be easily obtained as: $\rho_{(\cdot)}[UE/s/m^2] = \eta_{(\cdot)}[Mb/s]/S_{(\cdot)}[m^2]/FS[Mb/UE]$ (S refers to the area and FS to the file size). We use Space Poisson Point Processes for the user arrivals. A user arrives in the network, transmits its file and then leaves the network. User scheduling is done in a Proportional Fair (PF) manner.

B. Simulation results

We use a file size of $FS = 16[Mb/UE]$ and the traffic arrival rates $\eta_G = 72[Mb/s]$ and $\eta_{HS} = 63[Mb/s]$. We collect

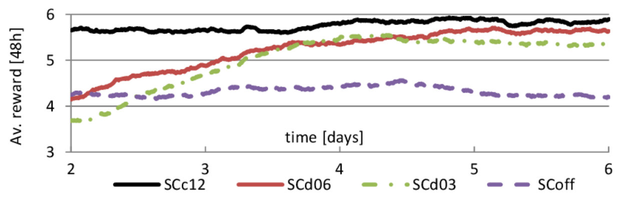

Figure 3. Average reward ($48h$ sliding window)

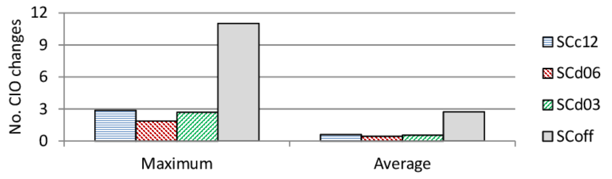

Figure 4. Average no. of CIO changes ([#/h]) over last 48h

data from 6 simulated days in order to evaluate the benefits and the scalability issue of the solution.

One can see in Fig. 3 that the average reward is increased when the SONCO is on, with the biggest gain being (as expected) in the centralized SONCO deployment (SCc12). We can see that the more distributed the system is the more we loose in terms of reward. This is the price to pay in order to gain in scalability. It is easy to foresee that if the cell clusters of the SONCO instances in the distributed cases were independent than the performance would have been the same as for the centralized case. In practice the cell clusters should be designed such that between them there is as less interaction as possible (the border cells have almost static parameter configuration), but as you can see good results can also be obtained if this is not possible.

Since our aim is to reduce the number of CIO changes we plot in Fig. 4 the maximum and the average (over all eNBs) of the time-averaged number of CIO changes. We can see a significant decreased (73-84%) eliminating most of the unnecessary configuration changes and the accompanying signaling. Also, by steering the CIOs configuration towards the most rewarding one, the frequency of overload events (when cells send off-load requests) is reduced by 30-46%.

In Fig. 5 we plot the maximum and the average (over all eNBs) of the time-averaged loads; one can see that the MLB instances are still fulfilling their task keeping the load well below the *off-load* threshold ($\mathbb{T}^H_{load}$). Note that the differences between all MLB on scenarios (SC*) are very small.

VI. Conclusions and Future Work

We have proposed a coordination mechanism for SON instances that do not communicate with each other and we have analyzed its performance in terms of network stability, convergence and scalability. The proposed framework is based on RL which learns from past decisions. We used MLB to demonstrate the concept but the work could be easily extended to other SON functions (MRO tuning the HO hysteresis, CCO tuning the transmission power or antenna tilt, etc.). Using SONCO instances to coordinate the SON (MLB) instances enables us to: provide an increased stability in the network (decrease parameter fluctuations by 73-84%) reducing also the associated signaling and still allow the SON instances to perform their duty (for MLB: keep the eNBs from becoming overloaded). We also observe that the proposed scheme trades off convergence (in terms of time and reward) with scalability. Future work includes the use of multiple SON functions and more advanced RL algorithms based on function approximation to overcome the scalability-convergence trade-off.

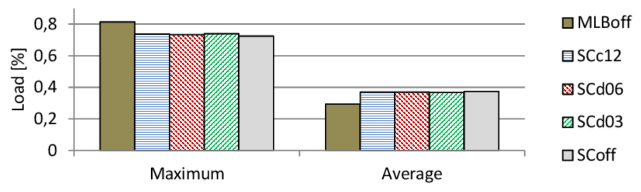

Figure 5. Average loads over last 48h

Acknowledgment

The research leading to these results has been carried out within the FP7 SEMAFOUR project [16] and has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no 316384.

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.
[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.
[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.
[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. N. ul Islam and A. Mitschele-Thiel, "Reinforcement learning strategies for self-organized coverage and capacity optimization," in *Wireless Communications and Networking Conference (WCNC), IEEE 2012*.
[12] R. Combes, Z. Altman, and E. Altman, "Coordination of autonomic functionalities in communications networks," *CoRR*, 2012.
[13] 3GPP, "LTE; E-UTRA and E-UTRAN; Overall description," 3rd Generation Partnership Project (3GPP), TS 36.300, 2013.
[14] ——, "E-UTRA; Further advancements for E-UTRA physical layer aspects," 3rd Generation Partnership Project (3GPP), TR 36.814, 2010.
[15] ——, "LTE; E-UTRA; Radio Resource Control; Protocol specification," 3rd Generation Partnership Project (3GPP), TS 36.331, 2012.
[16] SEMAFOUR project web page http://fp7-semafour.eu/.