Linear UCB for Online SON Management

Tony Daher, Sana Ben Jemaa
Orange Labs
44 Avenue de la Republique 92320 Chatillon, France
Email:{tony.daher,sana.benjemaa}@orange.com

Laurent Decreusefond
Telecom ParisTech
23 avenue d'Italie, 75013 Paris, France
Email:laurent.decreusefond@mines-telecom.fr

***Abstract*—Policy Based SON Management (PBSM) is the process of orchestrating the deployed Self-Organized Network (SON) functions, so that the network responds as a whole to the operator objectives. This process is based on the configuration of the SON functions in order to steer their actions in the network towards certain operator objectives. The PBSM ensures an automated translation of these objectives into configurations of the SON functions. An approach has been recently proposed to empower the PBSM with cognitive capabilities (C-PBSM), using a Multi-Armed Bandit algorithm, namely the UCB1. The C-PBSM learns the optimal SON configurations based on network feedback. In this paper we propose an alternative approach, based on the LinUCB algorithm, that is able to learn the optimal SON configuration much faster than the previous approach. The speed of convergence is a critical factor that has to be thoroughly considered in the deployment of online learning processes. Results are shown using an LTE-A simulator that considers real-like network topology and parameters, and accurate ray tracing based propagation.**

I. Introduction

Network management is one of the main and crucial challenges that network operators are facing, especially with the expected increase in future networks complexity, namely the 5G networks. A first step towards autonomic and intelligent network management was achieved with the 3GPP introduction of the Self-Organizing Networks (SON) functions in its release 8. These functions include self-configuration, self-optimization and self-healing functionalities [2]. We are particularly interested in the self-optimization SON functions for radio access networks. Their objective is to continuously optimize the network's parameters during its operation e.g. mobility, load balancing, inter cell interference *etc*. These functions have different objectives, that may not be compatible with each other if deployed simultaneously. Deploying such various functions in a network, without any kind of orchestration and coordination, will hardly lead to an optimal operation of the network in overall.

The work in this paper is focused on autonomous SON management through a Policy Based SON Management entity (PBSM) [3]. In this framework, we consider that the SON functions are provided by the SON vendors to the operator as black box functions. The operator has few information about the running algorithm because of proprietary issues. SON Configuration Value (SCV) sets represent a mean for the operator to control and orchestrate the SON functions deployed in its networks (an SCV set is a collection of threshold values, parameter range, step size ... of the SON algorithm). In fact, the network's Key Performance Indicators (KPIs) that are targeted by the function will be driven to a value or another depending on its configuration and also on the network context (radio environment, network topology, traffic conditions, *etc*). The objective of the PBSM is hence to translate autonomously the high level operator objectives into SCV sets, so that the network responds properly as a whole to the operator objectives (figure 1).

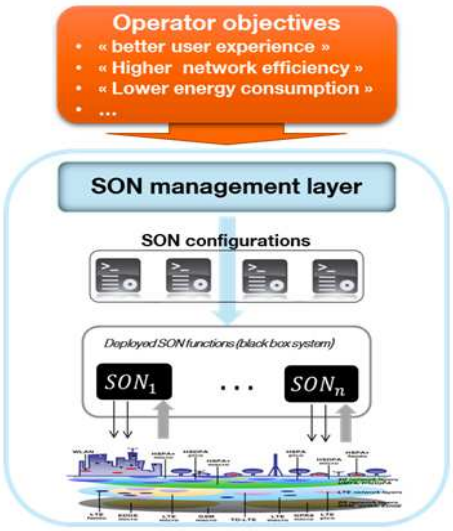


Fig. 1: PBSM functional description

The operator no longer needs to figure out a configuration for each SON function individually; it only has to define its KPI targets, and the PBSM will find the best SON configurations to achieve these targets. This improves the optimality of SON networks management, bringing the automation to a higher level and reducing considerably the human effort in the process.

Several approaches have already been investigated for the PBSM in the scope of the Semafour project [3], based on the combination and comparison of simulated models for the SON functions, the network context and the operator's objectives. In [4], the authors propose to empower

the PBSM with cognitive capabilities, based on the UCB1 Multi-Armed Bandit (MAB) algorithm [6]. The Cognitive PBSM (C-PBSM) proposed in [4], that we henceforward refer to as baseline C-PBSM, has the advantage of using an online Reinforcement Learning (RL) algorithm, i.e. UCB1 algorithm, to find the optimal action (that is the optimal SCV set combination for the deployed SON functions) directly from the real network feedbacks, without relying on simulated models, that may not reflect entirely the actual network and hence be misleading. Furthermore, the baseline C-PBSM is able to dynamically adapt its strategies with the operator objective changes. However, before converging to the optimal action, the learning process has to explore the action space. During this exploration phase, the network KPIs will be suboptimal with respect to the operator objective. The convergence speed of such online learning processes is hence a critical factor that needs to be thoroughly considered, so that less time is spent being suboptimal while exploring the action space.

In this paper, we argue that a different MAB algorithm, namely the LinUCB [7], can be used to find the optimal actions with a better convergence speed. By considering a feature vector associated with each action, and linear rewards with respect to the feature vector, the LinUCB is able, at each iteration, to estimate the outcome of different actions by testing only one action. We perform simulations using a real network topology and measurements that show that the C-PBSM proposed in this paper is able to find the optimal action and converges faster than the baseline C-PBSM.

The rest of the paper is organized as follows. In the next section we present the baseline C-PBSM and compare it with the approach adopted in the paper. The scenario and the results are presented and discussed in sections III and IV receptively. Finally a conclusion and future works are given in section V.

II. C-PBSM Based on Multi-Armed Bandits

A. Problem Formulation

Consider a network section where several SON functions instances are deployed. Let S be the set of SON functions deployed in the network. N_s is the number of deployed instances of a SON function s and V_s is the set of possible configuration sets for SON s, $\forall s \in S$. Assuming that all instances of a certain SON function s have the same set of possible configuration sets V_s, we define the action set as:
$C = (V_{s_1})^{N_{s_1}} \times (V_{s_2})^{N_{s_2}} \times ... \times (V_{s_{|S|}})^{N_{s_{|S|}}}$ where $s_1, s_2, ..., s_{|S|} \in S$.
We also consider the following reward function:

$$r_c = \sum_i \omega_i . K_i \tag{1}$$

where K_i are KPIs and ω_i are weights set by the operator, they are positive and add up to one. They reflect the operator's priority to optimize the corresponding KPI. We also consider the following hypothesis:

- The traffic in the network is stationary
- $\forall c \in C$, all the SONs converge after a sufficient time

Under these 2 conditions, for the same configuration c, the observed rewards can be considered as i.i.d. [4]. Consequently, the optimal action becomes:

$$c^* = \underset{c \in C}{\operatorname{argmax}}(\mathbf{E}(r_c)) \tag{2}$$

The learning agent's objective is hence to find c^*.

B. Multi Armed Bandits

MAB belongs to the RL framework, where an agent has to learn an optimal behavior by interacting with a dynamic environment [5]. In the MAB framework, a learning agent has to find the optimal action among a set of actions called arms, while maximizing the cumulated reward obtained during the process [6]. The agent has to learn the optimal action through a sequential learning process. In the recent years, MAB algorithms have gained popularity in many telecommunication applications such as in [8] and [9]. Some of the notable applications of MAB in mobile networks include resource allocation [10], interference coordination [11] as well as cognitive radio [12].

C. C-PBSM based on UCB1

There are several types of MAB algorithms in the literature that depend on the formulation of the MAB problem [6]. For the C-PBSM, the SON management problem can be modeled as a stochastic MAB. In this case the learning process goes as follows: For $t = 0, 1, ..., T$:

- The agent selects an action (arm) $c_t \in C$ according to some policy
- The environment outputs a vector of rewards $\mathbf{r}_t = (r_{t,1}, r_{t,2}, ..., r_{t,n}) \in [0,1]^n$ where n is the number of arms.
- Agent observes only r_{t,c_t}

As stated previously, the agent's objective is to find the optimal action while maximizing the cumulative rewards in the process, or stated differently, while minimizing the cumulative regret. In the stochastic case, the optimal action is by definition the one that generates the highest reward in expectation and the cumulative regret is defined as:

$$\overline{R}_T = T\mu^* - \mathbf{E}[\sum_{t=0}^{T} \mu_{c_t}] \tag{3}$$

where μ_{c_t} and μ^* are respectively the average reward of action c_t and the average reward of the optimal action (namely c^*). The UCB1 (from Upper Confidence Bound) algorithm was first analyzed in [6] where the authors showed that it achieved an expected regret bound that matches with the lower bound that can be achieved with any other action decision strategy.

UCB1 Algorithm

$\alpha > 0$
for t=0,...,T
- select arm c_t that maximizes $\widehat{\mu}_{t-1,c_t} + \sqrt{\frac{\alpha ln(t)}{2N_{t-1,c_t}}}$
- observe reward r_{t,c_t}
- evaluate $\widehat{\mu}_{t,c_t}$
- $N_{t,c_t} = N_{t,c_t} + 1$
where $\widehat{\mu}_{t-1,c_t}$ is the empirical average and N_{t-1,c_t} the number of times arm c_t was pulled at iteration $t-1$

Considering C to be the set of arms and an associated reward as formulated in equation 1, UCB1 is able to learn and identify the optimal action. In practice the space action may easily explode because of its combinatorial aspect. However, the action space can be reduced through cell classification based on their context. The agent will hence configure the SON instances per cell class instead of the SON instances deployed in each cell [4, 13].

D. C-PBSM based on LinUCB

UCB1 is a good solution when the environment is stochastic and the arms are independent. However, faster strategies can be used, if a particular structure or relation between the arms can be identified. For example in [14] there is a set of super arms, each containing a subset of arms. At each iteration when a super arm is played, the environment reveals the outcome of all the arms belonging to the corresponding subset of arms. In [15] the authors consider dependencies between clusters of arms in the form of a generative model. In [7] each arm is constituted by a vector of features. The expected reward function is considered to be linear with respect to the features. MAB strategies can be adapted to each specific case of arms structure and dependence, so that the agent is able to perform better than in the general case where the arms are considered to be independent.

In our case, we are interested in the algorithm proposed in [7], namely the LinUCB. The reasons are motivated after we present the algorithm. Let C be the set of K arms. At each iteration t the agent picks an action c_t, corresponding to a features vector (known to the agent) $x_{c_t} \in \mathbb{R}^d$. The observed rewards r_{t,c_t} are independent random variables, for which the expectation is a linear combination of features:

$$\mathbf{E}[r_{t,c_t}|x_{c_t}] = x_{c_t}^T \theta^* \quad (4)$$

θ^* is an unknown parameter vector to be estimated. The definition of the best action and the regret is still the same as in equation 2 and 3. The idea behind the algorithm is to estimate the unknown parameter vector θ^* by applying a regression on trained data at iteration t. The trained data is in this case the previously seen feature vectors (arms) until iteration t.

LinUCB Algorithm

1:Inputs $\alpha > 0$, $K, d \in \mathbb{N}$
2: $A \leftarrow I_d$
3: $b \leftarrow \mathbf{0}_d$
4: for t=0,...,T
5: - $\theta_t \leftarrow A^{-1}b$
6: - Observe K features $x_{c^{(1)}}, x_{c^{(2)}}, ..., x_{c^{(K)}} \in \mathbb{R}^d$
7: - $\forall c \in C$ do
8: $\quad p_{t,c} \leftarrow \theta_t^T x_{t,c} + \alpha\sqrt{x_c^T A^{-1} x_c}$
10: - Choose action $c_t = argmax_{c \in C}(p_{t,c})$
11: - Observe reward $r_{t,c_t} \in 0, 1$
12: - $A \leftarrow A + x_{c_t} x_{c_t}^T$
13: - $b \leftarrow b + x_{c_t} r_{t,c_t}$
14: end for

I_d is a d by d identity matrix and $\mathbf{0}_d$ is a column vector with 0 values. The ridge regression is performed on line 5. In line 8 $p_{t,a}$ is composed of 2 expressions, the first is an estimate of the expected reward and the second is an upper confidence bound. The detailed analysis of LinUCB can be found in [7] where the authors showed that the algorithm achieved a regret bound of $O(\sqrt{Td\ln^3(KT\ln(T)/\delta)}$ with probability $1-\delta$. They also prove a matching lower bound for this framework $\Omega(\sqrt{Td})$.

Our interest in the LinUCB algorithm is motivated by the fact that the actions of the C-PBSM, described in paragraph A, can be seen as a vector of features: if N is the number of different SON function instances deployed in the network, than an action $c \in C$ will be a vector of N categorical features. These features represent the configuration sets of the SON instances. We encode the categorical variables as binary vectors then we normalize the obtained vector to the unit [16]. In the following sections we evaluate and compare the performances of the 2 approaches.

III. Scenario Description and Evaluation

We consider an LTE-A system level simulator where we consider a heterogeneous radio access network as represented in figure 2. The macro cellular layer corresponds to a real network in central Paris with real-like network parameters and accurate ray tracing based propagation. The small cell layer is added using standard 3GPP propagation specifications [17]. We consider an unbalanced and stationary traffic distribution: users arrive according to Poisson processes with different arrival rates in different parts of the network. This results in some macro cells being highly loaded and others with low load. Additional traffic hotshots are served by the small cells. We only consider downlink traffic.

We consider a scenario with 3 distributed SON functions, each having several instances deployed in the network:

- Mobility Load Balancing (MLB): Deployed on each macro cell, it tunes the Cell Individual Offset (CIO)

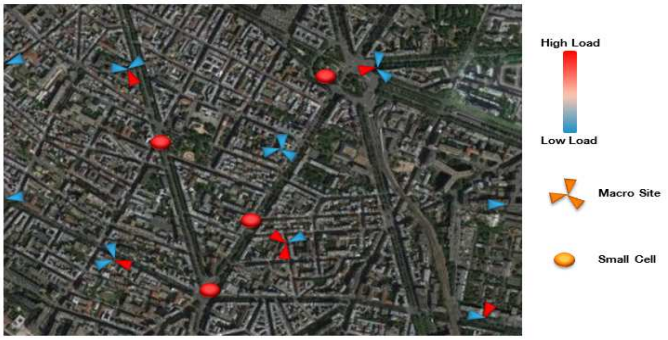


Fig. 2: Network Section

of macro cells. Its objective is to balance the traffic load between the macro cells.

- Cell Range Expansion (CRE): Deployed on each small cell, it tunes the CIO of the small cells to balance the load between small cells and the associated macro cell.
- Enhanced Inter Cell Interference Coordination (eICIC): Deployed on macro cells with small cells in their coverage area. eICIC manages Almost Blank Subframes (ABSF) transmissions of macro cells in order to protect small cell edge users from macro downlink interference.

We consider the SON configuration sets:

- MLB:
  - Off: Function is turned off
  - SCV1: Soft configuration
  - SCV2: Aggressive configuration
- CRE:
  - SCV1: Soft configuration
  - SCV2: Aggressive configuration
- eICIC
  - Off: Function is turned off
  - SCV1: Function is turned On

It is unfeasible and unnecessary to consider all the possible combinations of SON configurations because the action space would be huge and this would lead to a lot of combinations leading to similar outcomes and behavior. Therefore, we adopt the same approach as in [4] and consider cell classes as follows: a class of macro cells with high traffic and where small cells can be present in the coverage area (class C_1) and another class of small cells with low traffic (class C_2). We hence consider that all the instances of a certain SON function, deployed in the same class, will be configured with the same SCV sets. In other words, C_1 class would have 12 possible configurations: 3 for MLB instances, 2 for CRE and eICIC each, and C_2 would have 3 for MLB. This leaves us with an action space of 36 actions ($|C| = 36$). This corresponds to a binary feature vector of dimension 10.

On the other hand we define the following KPIs:

- $L_{i,c,t}$ is the load of cell i
- $\overline{L}_{c,t}$ is the average load in the considered section
- $\overline{T}_{c,t}$ is the average user throughput in the considered section
- $\overline{T'}_{c,t}$ is the average small cell edge user throughput in the considered section

The reward becomes:

$$r_{c,t} = \omega_1(1 - \sigma_{c,t}) + \omega_2\overline{T}_{c,t} + \omega_3\overline{T'}_{c,t} \tag{5}$$

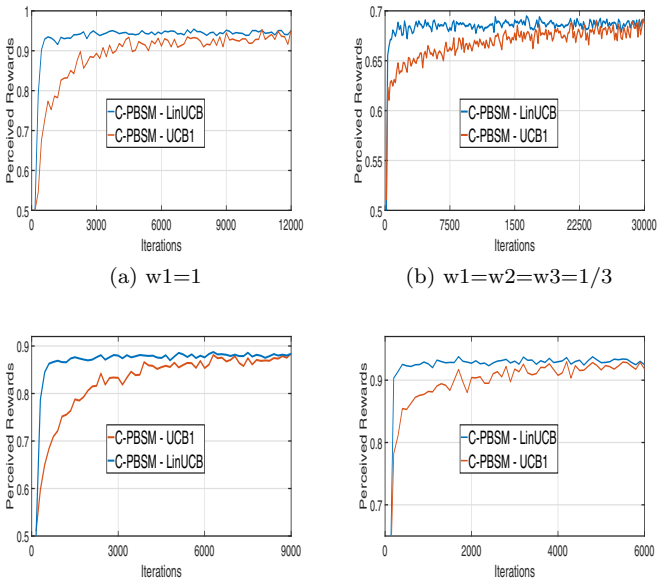
And the load variance is:

$$\sigma_{c,t} = \frac{\sum_{i=0}^{B}(L_{i,c,t} - \overline{L}_{c,t})^2}{B}$$

Where $c \in C$, t is the iteration and B is the number of cells in the considered section.

IV. Simulation Results

In the previously described scenario, we test the baseline C-PBSM based on UCB1 algorithm and the C-PBSM proposed in this paper, based on LinUCB. We run each C-PBSM for different operator objectives. The results are presented in figure 3.



(a) w1=1 (b) w1=w2=w3=1/3

(c) w1=w3=1/2 (d) w2=1

Fig. 3: Perceived Rewards Comparison

The results show that both algorithms converge and lead to the same performances in terms of generated rewards and hence KPI performances. However, the LinUCB shows a much faster convergence than the UCB1 based approach. This fast convergence is due to the linearity assumption in the LinUCB. As stated previously, unlike the UCB1 that assumes independent arms, the LinUCB assume a linear relation between the arms as shown in equation 4. In other words, this means that by testing a certain SCV combination in the network, the C-PBSM based on LinUCB is able to estimate the outcome of other SCV combinations without testing them, i.e. at each iteration, the LinUCB tries a single action but deduces and updates

the estimate of several actions according to the linear model. Whence the faster convergence.

However, the linearity hypothesis should be handled with caution when generalizing the approach to other scenarios. Mainly because in certain cases, the deployed SON functions and instances may by strongly correlated with each other in the sense that changing the SCV set of a SON instance might alter the behavior of other SON instances. In such cases, the linearity considered in the proposed approach is no longer valid, which may lead to misleading learning feedbacks or even prevent the convergence of the process. Consequently, a thorough analysis of the correlation and interaction between the deployed SON functions and instances for a certain scenario should be done before launching the proposed C-PBSM learning process. Studying deeply the correlation between SON functions may not be easy as they are designed as black boxes with limited information. The other solution would be to classify the cells into geographical clusters with low correlations, in order to guarantee the linearity of the model. This clustering would take into account the topology of the network as well as the traffic distribution and evolutions. The C-PBSM would hence configure the SON functions per cell cluster.

V. Conclusion

In this paper we have introduced a new reinforcement learning approach to the PBSM (Policy Based SON Management) based on the LinUCB multi-armed bandit framework. This approach considers linearity in the expected rewards of actions with respect to their corresponding feature vector. The motivation behind this approach is to take advantage of the structure of the action space, that is the SCV sets combinations, in order to converge faster towards the optimal action, hence spending less time exploring the action space and being suboptimal with respect to the operator objectives. We compare the performances of the proposed approach with a baseline C-PBSM (Cognitive PBSM) using a simulator that considers a real network topology, configuration and measurements. Results confirm that the proposed C-PBSM converges much faster towards the optimum. The speed of convergence is a critical factor since the algorithm is learning the optimal action online and from the real network.

This work has been carried out in the context of the effort to introduce cognition and intelligence in the network SON management process. The future work would be to generalize the proposed approach and to consider real life traffic variation models and to deal with the practical challenges of deploying such algorithms in real networks.

References

[1] Cisco, Cisco Visual Networking Index. "Global Mobile Data Traffic Forecast Update, 2015-2020." *white paper*, 2016.

[2] S. Hämäläinen et al., *LTE self-organising networks (SON): network management automation for operational efficiency,* John Wiley & Sons, 2012.

[3] SEMAFOUR project, http://fp7-semafour.eu/, 2015.

[4] T. Daher, S. Jemaa and L. Decreusefond, "Cognitive Management of Self-Organized Radio Networks Based on Multi Armed Bandit," *IEEE Personal Indoor and Mobile Radio Communications (PIMRC)*, 2017.

[5] L.P. Kaelbling, M.L. Littman and A.W. Moore "Reinforcement learning: A survey," *Journal of artificial intelligence research,* vol. 4, pp. 237-285, 1996.

[6] S. Bubeck and N. Cesa-Bianchi, "Regret analysis of stochastic and non-stochastic multi-armed bandit problems," *Foundations and Trends in Machine Learning,* vol. 5, pp. 1-122, 2012.

[7] W. Chu et al., "Contextual bandits with linear payoff functions," *International Conference on Artificial Intelligence and Statistics,* pp. 208-214, 2011.

[8] X. Wang, X. Li and V.C. Leung, "Artificial intelligence-based techniques for emerging heterogeneous network: State of the arts, opportunities, and challenges," *IEEE Acess,* vol. 3, pp. 1379-1391, 2015.

[9] R. Li et al., "Intelligent 5G: When Cellular Networks Meet Artificial Intelligence," *IEEE Wireless Communications,* 2017.

[10] A. Feki and V. Capdevielle, "Autonomous resource allocation for dense lte networks: A multi armed bandit formulation," *IEEE Personal Indoor and Mobile Radio Communications (PIMRC)*, 2011.

[11] P. Coucheney, K. Khawam and J. Cohen, "Multi-armed bandit for distributed inter-cell interference coordination," *IEEE International Conference on Communications (ICC)*, 2015..

[12] W. Jouini et al., "Multi-armed bandit based policies for cognitive radio's decision making issues," *IEEE Signals, Circuits and Systems (SCS)*, 2009.

[13] S. Hahn et al., "Classification of Cells Based on Mobile Network Context Information for the Management of SON Systems," *IEEE Vehicular Technology Conference (VTC Spring)*, 2015.

[14] W. Chen, Y. Wang and Y. Yuan, "Combinatorial multi-armed bandit: General framework and applications," *International Conference on Machine Learning,* pp. 151-159, 2013.

[15] S. Pandey, D. Chakrabati and D. Agarwal, "Multi-armed bandit problems with dependent arms," *Proceedings of the 24th international conference on Machine learning,* pp. 721-728, 2007.

[16] W. Chu and S.T. Park, "Personalized recommendation on dynamic content using predictive bilinear models," *Proceedings of the 18th international conference on World wide web,* pp. 691-700, 2009.

[17] 3GPP TR 36.814, "Evolved Universal Terrestrial Radio Access (E-UTRA); Further advancements for E-UTRA physical layer aspects (Release 9)", v.9.0.0, March 2010.