

A scalable SON coordination framework for 5G

Diego Fernando Preciado Rojas
Integrated Communications Systems
Technische Universität Ilmenau

Ilmenau, Germany
diego-fernando.preciado-rojas@tu-ilmenau.de

Andreas Mitschele-Thiel
Integrated Communications Systems
Technische Universität Ilmenau

Ilmenau, Germany
mitsch@tu-ilmenau.de

Abstract—For the near future mobile network management is expected to be an intractable task. This is due to the coexistence of different radio access technologies, slice-specific Quality of Service (QoS) requirements, the combination of multiple conflicting goals in network functions, etc. In order to reduce the human interaction in that duty, Self-Organized Networks Functions (SF) are proposed to provide self-adaptation capabilities to mobile networks on different fronts: configuration, optimization and healing. Conventional design of these SFs was based on single objective optimization approaches where SFs were considered as standalone agents aiming at one very specific local objective. Therefore, inter-dependencies between SFs were somehow neglected, so when more than one function is acting on the network, conflicts are inevitable. A well-studied conflict happens when Mobility Load Balancing (MLB) and Mobility Robustness Optimization (MRO) functions are simultaneously set up: without coordination, performance degradation will happen because of the inter-dependencies between both SFs which rule the standard handover procedure. To cope with these underlying conflicts, we propose a new methodology to automatically derive the system's model, considering the dynamic of the selected SFs, to assist the network optimization task and to equip the network with self-management faculties.

Index Terms—Deep Learning, Elastic Net, CIO, hysteresis, Lasso Regression, MLB, MRO, Q-Learning, SOM, SON, Reinforcement Learning, Ridge Regression, TTT

I. INTRODUCTION

Self-Organizing Networks (SON) comprise a set of functions in charge of ruling the behavior of a mobile network based on specific optimization targets and resource constraints. Roughly speaking, every SF takes as input some metrics of the environment (from now on called Environmental Variables) and according to some policy tunes network parameters (Controllable Variables) in order to solve a SF-specific optimization task.

Two of these functions are Mobility Load Balancing (MLB) which is in charge of the load balancing across the cells, improving the network capacity and Mobility Robustness Optimization (MRO) whose goal is to reduce the number of Radio Link Failures (RLF), and thus improving the QoS. These SFs have a strong interdependency since both optimize handover parameters. Modeling the cross effects between MLB and MRO usually demands expert knowledge which limits the self-organizing properties of the network and makes the network management expensive.

Recent proposals for modeling standalone SFs focus on Machine Learning (ML) techniques, specifically Reinforcement Learning (RL). Those algorithms allow one agent to learn the environment's model through on-line interactions between the agent and the network. That approach has been widely used for MLB [1]–[5] and MRO [6]–[8]. Although RL works well for isolated SFs, given the complex relationships between cellular system performance and operational parameters, all SFs are highly susceptible to parametric or logical inter-dependencies. Those inter-dependencies can induce various types of conflicts undermining the optimal network operation. Thus isolated modeling of SFs is suboptimal, resulting in poor network performance and problems with system stability.

Self-coordination is proposed as a mechanism to guarantee interoperability between SFs and a conflict-free environment. According to literature, it is possible to group the available works related to Self-coordination in different research directions:

- Explicit SFs modeling. In [9] a specific human modeling work is presented, where each SF is modeled as linear ordinary differential equations and the coordination solution consists in finding a coordination matrix that stabilizes the system of SFs. The authors use traditional control theory to solve the problem. Although the method works properly, we claim it is better to deploy fully autonomous systems minimizing the human intervention for modeling.
- Stochastic games. In [10], the authors decompose the global SFs dynamics in a set of Markov Decision Processes that were solved independently by means of RL methods and their individual policies are combined to obtain a global policy. The coordination among the different actions selected by the conflicting use cases is achieved by means of a coordination game. Unfortunately, the available literature about the theoretical properties of those games is mainly based on two-players and zero-sum games.
- Performance prediction. In [11] fully distributed SFs located in each cell predict the quality of potential candidate parameter configurations via fast calculations without the need of any direct system feedback. It was shown that a self-learning approach is able to quickly find "pretty good" configuration parameters.

One constraint of that approach is the suboptimality in the solutions, so an additional fine-tuning step of the candidate solutions is needed.

- Decoupled SFs execution. In [12] a time-decoupled SFs framework is proposed where the main idea is to reschedule or delay the execution of a SF function until the effects of another conflicting executing SF function becomes visible in the system. A similar approach is proposed in [13] where Space-Time scheduling procedures allow for separating the execution of SFs at different points in space and time so as to minimize negative cross effects among the SFs. In this paper we compare the performance of our proposed model with one of these approaches.
- ML driven approach. As far as we know at the time of writing, only [14] proposes a tool for conflict resolution based on network performance predictions using ML. The authors propose a centralized coordinator where two main tasks are done: 1. Network performance prediction for each SF and 2. Multi-objective optimization process (using a population based meta-heuristic), which searches for a set of non-dominated solutions (those which cannot improve the performance of one SF without degrading the other one). However, we claim for the sake of optimality, it is better to do the KPI prediction at the network level, so dependencies between SFs located in different cells are considered. It is also possible to transfer the complexity of the optimization step to a ML model in order to get a simpler optimization task (e.g. as simple as a table lookup). We also claim it is better having a coordinator enforcing directly the optimal configuration set into the network (instead of SFs sending a request to the SON coordinator and waiting for an authorization in order to adjust their configuration parameters).

In this paper we propose a canonical and centralized solution which learns the model of the SF's dynamics based on the history (which means it is a data-efficient and model-free approach). We will show that it reduces the time needed for optimization space or State-Action Space (SAS) exploration, it is scalable to any number of SFs, it simplifies the underlying optimization process, it is able to generate compact and portable rules or representations to be applicable under different network conditions.

The rest of the paper is organized as follows: Section II explains our point of view of the problem, the mathematical formulation and proposed solution. Section III shows a study case with the conflict between MRO and MLB as well as the feature engineering process and the SAS generation. The simulation parameters, results and comparisons with other schemes are presented in Section IV. We conclude with general comments and remarks in Section V.

II. SYSTEM MODEL

A. Interdependencies between SFs and "learning" paradigm

Every SF is modeled as a learning agent locally interacting within one cell, so the whole network can be seen as a multi-agent system, where each learner's decision has impacts on another learners in the same cell (intra-domain dependencies) and probably on the neighborhood (inter-domain dependencies) as shown in Fig. 1.

This decentralized approach bring us many challenges: the interactions between SFs depend on the specific environment, so human intervention is mainly expected to assist the modeling of this dynamics. On the other hand, the knowledge obtained by each agent is locally generated and not shared with other entities, so, although the learners become simpler in logic, a strong effort is needed to model the interaction between them and to solve the joint optimization problem.

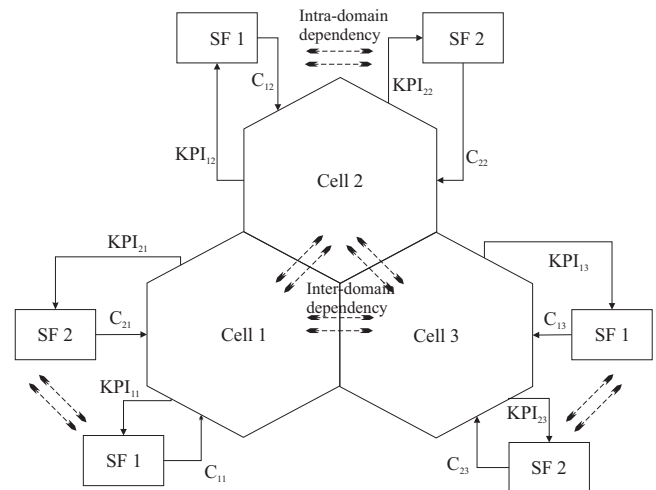


Fig. 1. Inter and Intra-domain conflicts.

In this paper we let all agents collect data from the environment and send it to a Machine Learning Sandbox: an environment in which ML models can be trained, tested and their effects on the network evaluated, designed to prevent a ML application from affecting the real network [15]. We enable intra/inter-domains observability (cf. Fig. 1) and we claim this will allow us to gain simplicity in the final optimization process. Since global decisions are made, concurrent modifications of parameters take place and these are explicitly signaled to all the agents. Every agent is in charge of the local configuration enforcement. In this manner, the locally-gathered knowledge is exploited to understand (in a centralized way) the dynamics between all agents. However, a drawback is expected: the SAS becomes much larger and more sparsely populated, at least at the beginning of the process, which means to face a slow learning process or "cold start" problem. This could generate temporary sub-optimal solutions. To cope with this limitation, we propose an idea for speeding the "learning" up, as shown in the next subsection.

B. Problem formulation and proposed solution

Consider a network with s number of cells, let P_{im}^c be the value of the i th parameter of cell c in the data sample¹ m and let E_{jm}^c define the j th environmental variable of cell c in the data sample m . From the network perspective, it is possible to define a global KPI (O_m) related to overall network performance considering all the deployed SFs as in (1).

$$O_m = g(P_{1m}^1, P_{2m}^1, \dots, P_{im}^1, E_{1m}^1, E_{2m}^1, \dots, E_{jm}^1, \dots, P_{1m}^c, P_{2m}^c, \dots, P_{im}^c, E_{1m}^c, E_{2m}^c, \dots, E_{jm}^c, \dots, P_{1m}^s, P_{2m}^s, \dots, P_{im}^s, E_{1m}^s, E_{2m}^s, \dots, E_{jm}^s) \quad (1)$$

We claim that it is feasible, using historical information and regression techniques to derive the system model $g(\cdot)$ which is the linkage function between KPI and controllable parameters plus environmental variables the cell is exposed to. Once we know $g(\cdot)$, it is possible to extrapolate the unknown combinations, that means to estimate the KPI values beyond conditions for which the model was trained, and thus to speed up the "learning" on never-seen-before data, therefore reducing the sparsity of the SAS. Thus, we can quickly create a huge knowledge database with the linkage between dependent and independent variables.² That is the first step in our proposed solution, as depicted in Fig. 2.

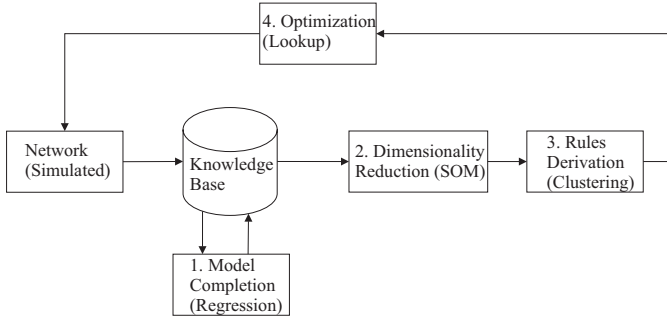


Fig. 2. Global view of the proposed scheme.

The variance of a regression model³ is inversely proportional to the difference between the number of independent variables n and the number of observations/samples m . That means that when n is getting closer to m , the variance of the model is pretty high. According to right-hand side of (1), the number of independent variables is expected to be high (it depends on the number of cells, the number of activated SFs, the number of controllable and environmental variables per SF), so reducing the variance in our regression models is

¹A data sample is observed and collected according to an observation period that is usually defined by the network operator.

²The terms dependent variable, target, output and KPI are used interchangeably in this paper as well as independent variables, inputs, features and regressors.

³In regression problems, there are two critical metrics: the bias and the variance of the estimators. The bias is the difference between the real value and the expected estimation, so it measures the accuracy of the estimates. On the other hand, variance measures the spread or uncertainty in these estimates.

a paramount challenge to face. In this paper, we propose to use regularization techniques to get over variance issues as we will see in the next section.

Once we have a realization of the model in the form of a table, the optimization problem could be seen as a simple lookup process inside it. However, the dimensionality of the structure is pretty high and the operational cost to make such a fast scan process is elevated. So we propose to apply dimensionality reduction without losing the linkage function between the target and regressors variables. For this, we use an approach based on semi-supervised learning theory using Supervised Self-Organizing Maps (SOM), as it is shown in the second step of Fig. 2. SOM was chosen because it can scale up to hundreds of input variables and also because this model is able to reconstruct the relationship between target and regressor's layers, hence the name semi-supervised model.

A traditional SOM (a.k.a Kohonen map) is a sort of neural network that converts high dimensional data into low dimensional representation (typically 2D). The map is made up of several nodes that cluster similar input samples. SOM retain the original topology of the data in the map, that means two data points that are close in the original input feature plane, will be mapped to have small distance in the resulting map. A SOM could have more than one layer of feature representation [16]. We use two layers, one for the regressors (controllable and environmental variables) and another one for the target variable. This is done in order to gain directionality in the map, that means get a clear idea of how to traverse it for the sake of reaching the optimal configuration under specific environmental conditions (i.e. simplicity in the optimization step). If additional information regarding the training and prediction processes for supervised SOM is needed, the authors recommend the work in [17], [18].

After having the dynamics mapped in a lower dimensional representation, we learn high-level/high-value behavioral rules that can be applied in complete sets of network conditions. This way we ensure the scalability and portability of the "learning" as well as simplicity in the optimization process. To create portable models, we allow the reuse of the same parameter sets for multiple environmental conditions. So, by an additionally clustering stage using only the environmental variables (step 3 in Fig. 2) we find excursion zones where an optimization task is almost as simple as a lookup process.

In step 4 in Fig. 2, the learner finds out the current state of the network which is associated with an unit in the SOM (step 2) and the corresponding cluster (step 3), then it selects the optimal unit inside the current cluster changing the controllable parameters. The new optimal configuration is selected based on the euclidian distance between the current configuration vector and the candidates of configuration sets. That means, we choose the configuration vector exhibiting the lowest number of changes with respect to the current configuration.

So far we have kept the proposed solution as general as possible, because we think it could be applicable to any network management problem where the involved variables can

be mapped in three categories as we have seen: controllable variables (or tunable parameters), environmental quantities (not under the operators's control) and performance metrics (in line with the operator's policies). In order to show the feasibility of our framework, we have applied it to two SFs: MLB and MRO which interact with each other through the handover procedure parameters, as it is shown below.

C. Handover Procedure and MLB-MRO conflict

The handover procedure (HO) was designed to preserve the connection of the User Equipment (UE) when it moves across the network due to new radio conditions or even load balancing schemes. The 3GPP standards have defined a set of trigger events for intra-system HO being A3 condition the most common. Roughly, A3 triggering condition causes the execution of a HO if the neighbor cell becomes offset better than the serving cell during a specific time period determined by the TTT parameter. It could be expressed (neglecting the frequency specific offsets) as in (2):

$$\overline{RSRP}_t > \overline{RSRP}_s + Hys + Oc_s - Oc_t \quad (2)$$

where $\overline{RSRP}_s$ and $\overline{RSRP}_t$ are the values of Reference Signal Received Power (RSRP) for the serving and target cell respectively, Hys is the hysteresis, Oc_s and Oc_t are the cell specific offsets for the serving and target cell respectively.

Typical implementations of MRO consider TTT and Hys as input parameters whereas MLB uses the cell individual offsets (CIO): Oc_s and Oc_t . Because two different SFs modify parameters related with the same network procedure, conflicting commands could take place. For instance, if MLB detects one cell is overloaded, it could try to shrink the cell size, transferring edge users to the neighborhood. Therefore the overloaded cell modifies the CIO for some of its neighbor cells (advancing some HO procedures towards the tier-one neighbors). However, if the radio conditions in the new cells are not good enough for the users that were handed over, some RLF (because of early handovers) could occur, so the MRO function in the target cells will hand the users back to the previously overloaded cell. With no coordination, the edge users could be bouncing between both cells affecting their QoS and also draining the signaling resources of the network, so orchestration between both SFs is essential.

D. State-Action Space definition

For MLB, the environmental variables considered in this paper are: the border's user distribution uD (defined as the relationship between the number of users in the cell's border over the total number of users in a cell) and the load in the cell (ρ), defined as the Physical Resource Block (PRB) usage as in (3):

$$\rho = \frac{\sum_{n=1}^{NoUsers} N_{PRB_n}}{BW_{net}/BW_{PRB}} \quad (3)$$

where N_{PRB_n} is the number of Physical Resource Blocks (PRB) required for the n th user, $BW_{PRB} = 180KHz$ and $BW_{net} = 10MHz$.

Cell overload occurs when the offered load ρ exceeds a preset threshold for the given cell bandwidth. Accordingly, ρ can be greater than 1, representing the case when the total required PRBs exceed the maximum PRBs available within for BW_{net} .

As we mentioned before, for MLB, the controllable variable, is the Cell Individual Offset (CIO). The main KPI for MLB is the rate of unsatisfied users N_{uu} which is defined as the relation between the number of un-satisfaction events in cell/second and the number of users in the cell. One un-satisfaction event occurs when the user's total achieved data rate in the continuous 1 second period is less than the Guaranteed Bit Rate (GBR).

For MRO, the environmental variable is the mean user speed, the controllable variables are TTT and Hys , and the main metric is the Handover Aggregate Performance (HOAP), defined as in (4):

$$HOAP = \frac{(w_1 * PP + w_2 * RLFE + w_3 * RLFL)}{Users_{cell}} \quad (4)$$

where PP is the Ping-Pong rate. A Ping-Pong HO is registered when a successful HO from a cell t to another cell s occurs in a time less than the "PP time" after another successful HO had already occurred from s to t . The PP rate measures the rate of HO oscillations per user per second. $RLFE$ and $RLFL$ components in (4) are related to RLF rate. A RLF occurs if the UE Signal to Interference & Noise ratio (SINR) stays below a threshold for a duration equivalent to the critical time (Timer T310 [19]). The RLFs are differentiated between those due to HOs being triggered too early (second term) and those due to HOs being triggered too late (last term).

In order to avoid conflicts between objective goals of MLB and MRO, we defined a combined metric as the weighted Average of $HOAP$ and N_{uu} :

$$O_m = w_4 * HOAP + w_5 * N_{uu} \quad (5)$$

This metric is the same global KPI previously defined in (1) and it is the quantity we must predict using the environmental and controllable variables. Wrapping up, the Table I summarizes the proposed SAS:

TABLE I
STATE-ACTION SPACE

SON Function	Env. Var.	Cont. Var	KPI
MLB	uD, ρ	CIO	N_{uu}
MRO	$Speed$	TTT, Hys	$HOAP$

III. EVALUATION

A. Baseline algorithms

For the sake of performance comparison of our method, we use as benchmark some of the proposed schemes in [13] where MLB and MRO are modelled as decentralized agents running a model-free Reinforcement Learning algorithm, namely Q-Learning (QL). The greedy RL implementation of MLB and

MRO are called QLB and QMRO respectively. QLH refers to a scheme where both MLB and MRO are working simultaneously without explicit coordination framework.

In [13] each SF acts as a control agent that: observes the network to find out its current state (both environmental and controllable variables), takes an action (exploration process) and gets feedback based on the effect of that action at the end of a monitoring period (called the SON interval). The action is the adjustment of some controllable variables. Based on the effect of this action, the Q-learner receives an immediate reward and the environment undergoes a transition into a new state. The objective of the learner is to discover an action policy that maximizes the cumulative rewards over time. One scheduling mechanism proposed in [13] is also considered as benchmark, namely Temporal Separation during Learning (TSL). This scheme is intended to reduce intra-domain conflicts. During the learning period there is a time separation between the SFs. Thereafter all the SFs are simultaneously executed.

B. Simulation Environment and Parameters

Simulation studies were done using a C++ event based LTE downlink system level simulator based on software libraries provided by Nokia Bell Labs and the University of Stuttgart's Institute of Communication Networks and Computer Engineering [20]. The network scenario consists of 7 eNBs each with 3 cells, where each cell has a system bandwidth $BW_{net} = 10MHz$. The simulator implements a wraparound of the network for reliable interference and SINR calculations, see Fig. 3.

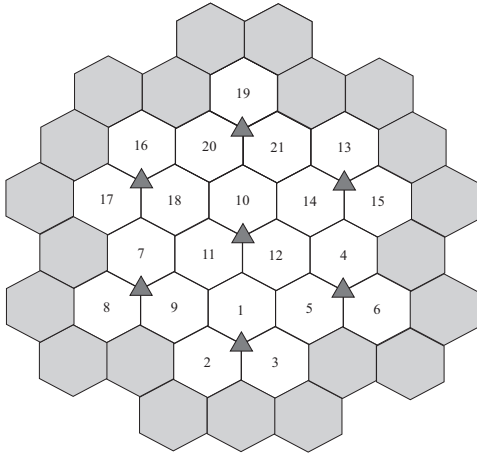


Fig. 3. Network layout.

Mobile users are randomly placed in the coverage area and move around following a random walk mobility model. We have introduced an asymmetric load distribution via 30 static users in the cell number 12. The simulation runs multiple batches each lasting a specified batch period and within which snapshots are taken. A single snapshot represents the aggregated effects of all the events that would have happened in the network since the last snapshot was taken. At the

TABLE II
SIMULATION PARAMETERS

Type	Parameter	Value
System specifics	Inter-site distance	500 m
	Pathloss	$128.1 + 37.6 \log_{10} [\max(d[km], 0.035)]$
	Shadowing	Standard deviation = 6 dB; Decorrelation distance = 50 m
	eNB Tx power	46dBm
	eNB Tx antennas	gain 15 dBi, height = 32 m
Simulation specifics	Batch duration	180 s
	Time between snapshots	10 ms
User specifics	Number of users	240 mobile, 30 static (cell 12)
	User's speed	10-120 Km/h
	Mobility model	Random Walk
	Positions of users	random, uniform distribution
	UE receive antennas	gain 2 dBi, height = 1.5m
	User data rate	256-1024 kbps
Algorithm specifics	PP Time	5 seconds
	T310	0.2s
	TTT	0-1280 ms
	Hys	0-15 dB steps: 0.5 dBs
	Weights in (4)	$w_1 = 0.3 w_2 = 0.2 w_3 = 0.5$
	Weights in (5)	$w_4 = 0.5 w_5 = 0.5$

beginning of each batch, all users are re-introduced in the network following the random deployment model. Further details of the simulation parameters are given in Table II.

C. Results

To learn the model $g(\cdot)$ in (1) and overcome the "cold-start" issue, a supervised learning approach is proposed (step 1 in Fig. 2). The idea is to estimate the expected KPI based on 150 regressors: 21 values per cell for the variables Hys , TTT , $Speed$, uD , ρ and 45 values for CIO (one value per adjacency). Different techniques are compared including: regression trees ($rTree$), random forest (rf), deep learning structures (dl) and models with regularization, specially useful when n is close to m , namely: Ridge regression ($ridge$), Lasso regression ($lasso$), elastic nets ($eNet$) and Partial Least Squares regression (pls). For each model, a grid search was carried out to find out the optimal value of the associated hyperparameters. The best model's hyperparameters are given in Table III. A deep explanation of the meaning of every parameters is out of the scope of this paper. If further details are needed, [21] could be a starting point.

The performance of the models is also shown in Table III.

The best model is a *deep learning* structure which exhibits the smallest RMSE and highest correlation factor, followed by the *pls* model. Satisfactory results are also obtained using *ridge* regression, therefore one important finding is the valuable impact of the regularization techniques on this specific problem, as we expected.

In Fig. 4 we plot the predicted and real values for O_m given

TABLE III
MODEL'S HYPARAMETERS

Model	Hyperparameters	RMSE	Corr. Factor
<i>rt</i>	maxdepth = 6	0.075	0.86
<i>rf</i>	ntree=200, mtry=7	0.054	0.93
<i>dl</i>	units=150/130/50/11/1, act.=RELU	0.04	0.95
<i>ridge</i>	$\alpha=0$, $\lambda=0.1$	0.05	0.94
<i>lasso</i>	$\alpha=1$, $\lambda=0.1$	0.13	0.80
<i>eNet</i>	$\alpha=0.5$, $\lambda=0.1$	0.09	0.86
<i>pls</i>	number of components=31	0.05	0.95

by the *dl* model in the test set⁴. We can see that the points are located around the perfect correlation curve (where the real value is exactly as the predicted one) with low dispersion.

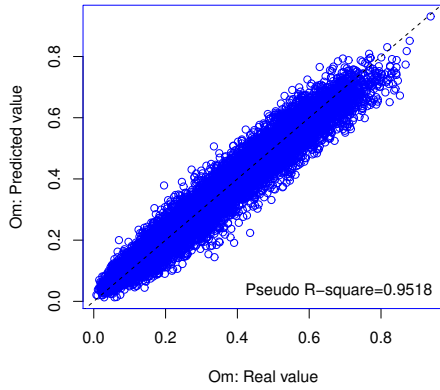


Fig. 4. Predicted against real values for deep learning regression method.

Because of the high dimensionality of the model (150 regressors and 1 KPI) we propose to translate the optimization problem into a lower dimensional space. Using Supervised SOM, we get the standard side-by-side representation of the regressor map (Fig. 5) and the respective KPI map (Fig. 6). Every unit in the regressor map contains an unique "spectrum" revealing the expected variable profile for all the samples within that unit. These profiles become the node's individual signature and every node is able to collect new data samples into it as long as the new incoming point has similar variable's profile.

On the other side, looking at the KPI map in Fig. 6 we realize the worst value for the metric in (5) lies in the top left-hand corner (the bigger the red circle is, the higher the KPI and the lower the network performance). The best value is diagonally opposite.

These two layers of SOM in Fig. 5 and Fig. 6 are linked. That means, it is possible to estimate the value of the KPI using the regressor's values and the SOM model. Actually, Fig. 7 shows the predicted and real values for O_m given by

⁴We have used the traditional cross-validation approach where the training set is used to create a base model, the validation set is used for fine tuning and the test set to face the model to never-seen-before data and assess its performance.

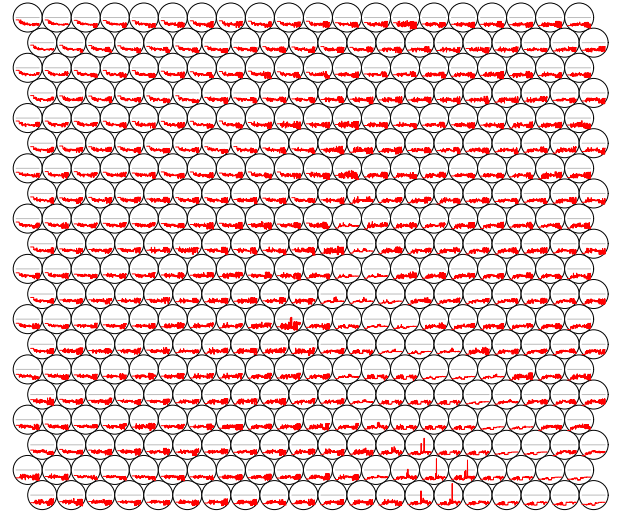


Fig. 5. Codes plot for the regressor's layer using a 20x20 nodes map.

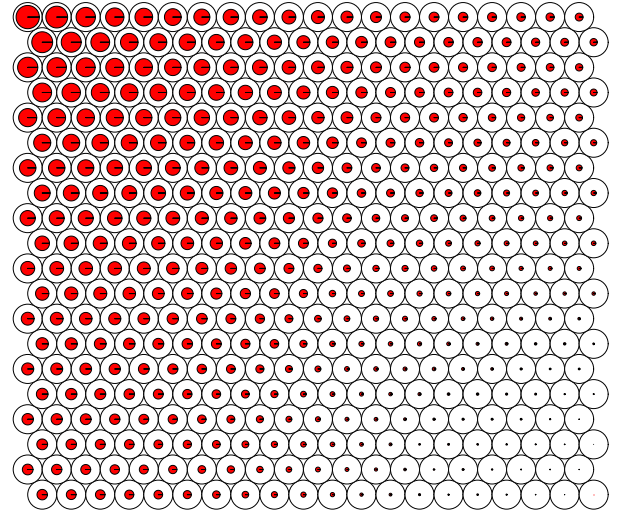


Fig. 6. Codes plot for the target's layer using a 20x20 nodes map.

the Supervised SOM using the test set. Once again, we can see that the points are located around the perfect correlation curve although the correlation factor is not as high as the one obtained with the *dl* model in Fig. 4. That is expected because we are trading the ability to explain the complete data variance for simplicity in the optimization process, as we will see below.

The optimization problem in this lower dimensional representation (2D) is raised as finding the best configuration for a given environmental condition. The degrees of freedom are limited only to the controllable variables: *Hys*, *TTT* and *CIO*. In step 3, by adding a clustering stage on top of regressor map using only the environmental variables, namely ρ , uD and *Speed* it is possible to find excursion zones where an optimization task is viable changing the configuration

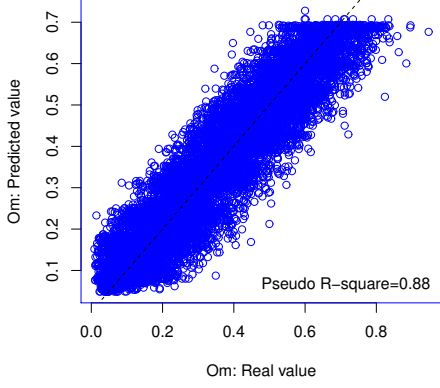


Fig. 7. Predicted against real values for Supervised SOM method.

parameters in order to get close to the bottom and rightmost corner (where the optimal KPI lies).

Fig. 8 depicts 24 regions (this number was chosen for the sake of explanation's simplicity). Inside every excursion zone, all the nodes have quite similar values in the environmental variables but different values in the controllable quantities, so starting from an initial working point, it is possible a parameter's fine tuning process trying to move the working point towards the bottom rightmost corner inside every excursion zone as it is depicted in Fig. 8, where one initial working point is shown and by changing the parameter accordingly, it is possible to reach the best possible value of the indicator. Inter-zones changes are only possible if the environmental conditions change (which of course, is not under the network operator control).

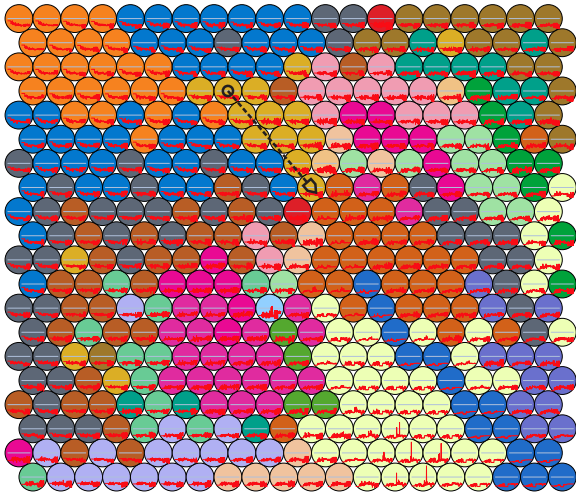


Fig. 8. Intra-cluster optimization.

D. Comparative analysis

Performance comparison between the proposed scheme and the aforementioned benchmark was carried out at low-speed scenario where the user velocities are randomly distributed

around 30 kmh. In Fig. 9 it is shown the evolution over the time for the HOAP (after a smoothing process) for the considered models. It is evident that after a long exploration phase, the QMRO learns the optimal policy, whereas QLH or QLB can not reach the same optimality degree. On the other side, SOM outperforms the benchmark from the very beginning of the simulation because of the learning acceleration due to the regression task.

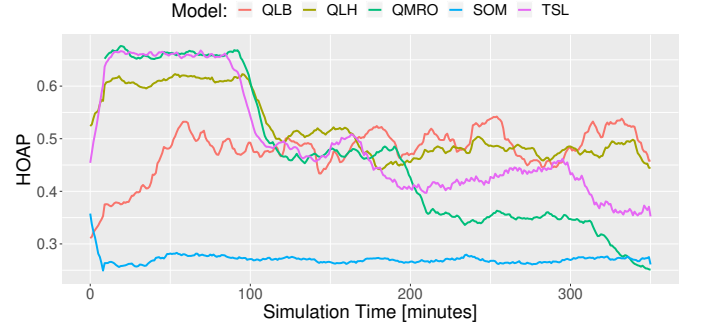


Fig. 9. HOAP evolution after removal of high frequency components.

The Rate of Unsatisfied Users is shown in Fig. 10. For this low-speed scenario QLB quickly reaches the optimality (it highly depends on the speed of the users). The mean performance of the SOM approach is quite stable across the simulation, however it is not as good as the one obtained with QLB. The worst performance is achieved with the QLH and QMRO algorithms.

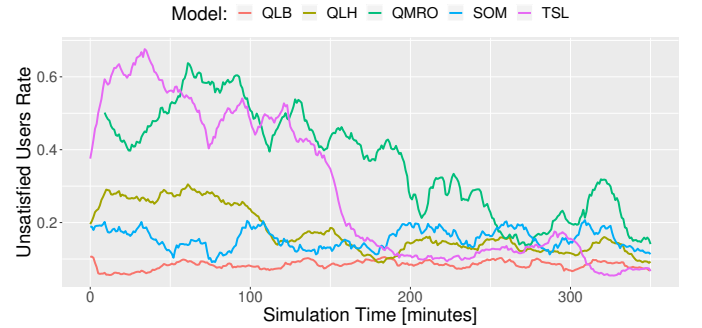


Fig. 10. Smooth Rate of Unsatisfied Users profile.

The profile of the global KPI defined in (1) is depicted in Fig. 11. It is possible to see the good results obtained with the proposed solution even from the beginning of the simulation.

The performance comparison in steady state is depicted in Fig. 12. The figure shows that the worst performance is achieved when no coordination is present and the best one using the SOM model.

IV. CONCLUSION

In this paper we have shown the applicability of ML techniques to get a higher degree of automation making the explicit human modeling of conflicts between SFs dispensable. Although our approach has been applied to the specific

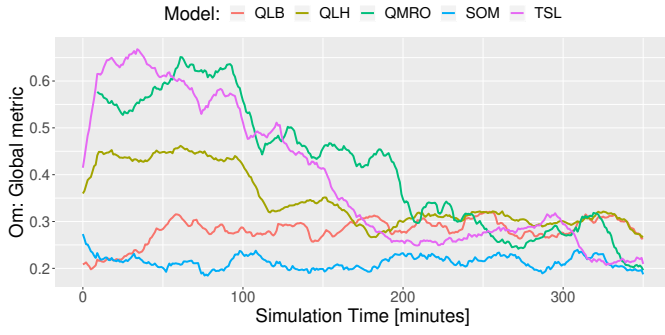


Fig. 11. Global KPI comparison.

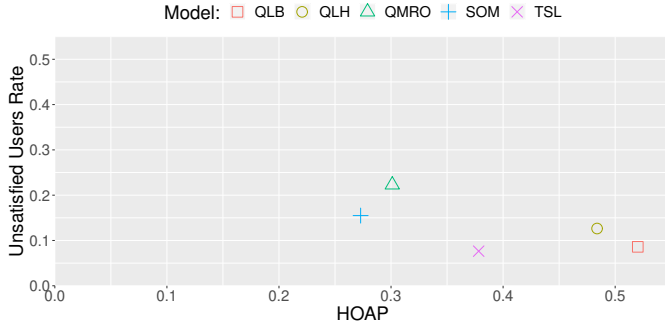


Fig. 12. Average steady state performance.

conflict between MLB and MRO we claim it can scale up to multiple SFs as long as a functional decomposition of the use cases into environmental, controllable and metric variables is conceivable.

Because of the higher degrees of freedom, the SAS is huge. Therefore, two main challenges must be handled: the time needed to learn the SAS and the dimensionality explosion. To cope with the first problem, a supervised learning approach was followed to speed up the "learning" process and inter/extrapolate unseen combinations among dependent and independent variables. It turned out that deep learning models were the most suitable choices for this task. These estimators shall change every time a new SF or cell is deployed. Therefore an outer loop for ML model's fine tuning should be present (which is fully compliant with [15]).

To face the high dimensionality problem, a semi-supervised learning approach was proposed, and we showed that using Supervised SOM it was possible to reduce the dimensionality of the problem without losing the linkage between the combined KPI and its regressors. That allowed us to use an additional clustering step based only on the observable variables. Thus a simple final optimization step is sufficient, where the current cell's working point is moved towards the optimal corner inside every excursion zone based on configuration adjustments.

ACKNOWLEDGMENT

The authors would like to thank Nokia Bell Labs, Germany, for the support and the provision of the network simulator.

REFERENCES

- [1] P. Muñoz, R. Barco, and I. de la Bandera, "Load balancing and handover joint optimization in lte networks using fuzzy logic and reinforcement learning," *Computer Networks*, vol. 76, pp. 112–125, 2015.
- [2] P. Muñoz, R. Barco, and I. de la Bandera, "Optimization of load balancing using fuzzy q-learning for next generation wireless networks," *Expert systems with applications*, vol. 40, no. 4, pp. 984–994, 2013.
- [3] S. S. Mwanje and A. Mitschele-Thiel, "Minimizing handover performance degradation due to lte self organized mobility load balancing," in *Vehicular Technology Conference (VTC Spring), 2013 IEEE 77th*, pp. 1–5, IEEE, 2013.
- [4] J. Suga, Y. Kojima, and M. Okuda, "Centralized mobility load balancing scheme in lte systems," in *Wireless Communication Systems (ISWCS), 2011 8th International Symposium on*, pp. 306–310, IEEE, 2011.
- [5] P. M. Luengo, R. Barco, J. M. Ruiz-Avilés, I. de la Bandera, and A. Aguilar, "Fuzzy rule-based reinforcement learning for load balancing techniques in enterprise lte femtocells," *IEEE Trans. Vehicular Technology*, vol. 62, no. 5, pp. 1962–1973, 2013.
- [6] A. Klein, N. P. Kuruvatti, J. Schneider, and H. D. Schotten, "Fuzzy q-learning for mobility robustness optimization in wireless networks," in *Globecom Workshops (GC Wkshps), 2013 IEEE*, pp. 76–81, IEEE, 2013.
- [7] W. Qin, Y. Teng, M. Song, Y. Zhang, and X. Wang, "Aq-learning approach for mobility robustness optimization in lte-son," in *Communication Technology (ICCT), 2013 15th IEEE International Conference on*, pp. 818–822, IEEE, 2013.
- [8] S. S. Mwanje and A. Mitschele-Thiel, "Distributed cooperative q-learning for mobility-sensitive handover optimization in lte son," in *Computers and Communication (ISCC), 2014 IEEE Symposium on*, pp. 1–6, IEEE, 2014.
- [9] A. Tall, R. Combes, Z. Altman, and E. Altman, "Distributed coordination of self-organizing mechanisms in communication networks," *IEEE transactions on control of network systems*, vol. 1, no. 4, pp. 328–337, 2014.
- [10] J. Moysen and L. Giupponi, "Self-coordination of parameter conflicts in d-son architectures: a markov decision process framework," *EURASIP Journal on Wireless Communications and Networking*, vol. 2015, no. 1, p. 82, 2015.
- [11] I. Karla, "Resolving son interactions via self-learning prediction in cellular wireless networks," in *2012 8th International Conference on Wireless Communications, Networking and Mobile Computing*, pp. 1–6, IEEE, 2012.
- [12] H. Y. Lateef, A. Imran, M. A. Imran, L. Giupponi, and M. Dohler, "Lte-advanced self-organising network conflicts and coordination algorithms," *IEEE Wireless Communications Magazine*, vol. 22, no. 3, pp. 108–117, 2015.
- [13] S. S. Mwanje and A. Mitschele-Thiel, "Sts: space-time scheduling for coordinating self-organization network functions in lte," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pp. 357–362, IEEE, 2015.
- [14] J. Moysen, M. Garcia-Lozano, L. Giupponi, and S. Ruiz, "Conflict resolution in mobile networks: a self-coordination framework based on non-dominated solutions and machine learning for data analytics [application notes]," *IEEE Computational Intelligence Magazine*, vol. 13, no. 2, pp. 52–64, 2018.
- [15] ITU, "Architectural framework for machine learning in future networks including imt2020," Recommendation Y.3172 (06/2019), ITU-T Study Group 13, jun 2019.
- [16] R. Wehrens and J. Kruisselbrink, "Flexible self-organising maps in kohonen 3.0," *Journal of Statis*, 2018.
- [17] F. M. Riese and S. Keller, "Introducing a framework of self-organizing maps for regression of soil moisture with hyperspectral data," in *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*, pp. 6151–6154, IEEE, 2018.
- [18] F. M. Riese and S. Keller, "Susi: Supervised self-organizing maps for regression and classification in python," *arXiv preprint arXiv:1903.11114*, 2019.
- [19] E. U. T. R. Access, "Radio resource control (rrc)," *Protocol specification (Release 10)*, vol. 290, 2011.
- [20] J. Sommer and J. Scharf, "Ikr simulation library," in *Modeling and Tools for Network Simulation*, pp. 61–68, Springer, 2010.
- [21] G. Aurélien, "Hands-on machine learning with scikit-learn & tensorflow," 2017.