Objective-driven coordination in self-organizing networks

Christoph Frenzel, Bernhard Bauer, Christoph Schmelz, Henning Sanneck

Angaben zur Veröffentlichung / Publication details:

Frenzel, Christoph, Bernhard Bauer, Christoph Schmelz, and Henning Sanneck. 2015. "Objective-driven coordination in self-organizing networks." In *Proceedings of the 2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), 30 Aug.-2 Sept. 2015, Hong Kong, China*, edited by Victor O. K. Li and Wen Tong, 1453–58. Los Alamitos, CA: IEEE. https://doi.org/10.1109/pimrc.2015.7343527.

Objective-Driven Coordination in Self-Organizing Networks

Christoph Frenzel and Bernhard Bauer
Department of Computer Science
University of Augsburg, Germany
Email: {frenzel, bauer}@informatik.uni-augsburg.de

Christoph Schmelz and Henning Sanneck
Nokia Networks, Munich, Germany
Email: {christoph.schmelz, henning.sanneck}@nokia.com

***Abstract*—The operation of a Self-Organizing Network (SON) such that it achieves the objectives of a cellular network operator is a complex task. The previously presented SON objective manager enables the automatic management of SON functions based on formalized operator objectives on network Key Performance Indicators (KPIs). However, this approach initially did not consider SON coordination which resolves runtime conflicts between concurrently executed SON functions. We present an objective-driven SON coordination which adapts the SON objective manager approach to SON function conflict resolution. Therefore, the SON objective manager concept is based on multiattribute utility theory and the conflict resolution is modeled as a constraint optimization problem. As a result, this approach coordinates the execution of SON functions such that the operator objectives are satisfied as quickly as possible.**

I. Introduction

The Self-Organizing Network (SON) paradigm automates mobile network operations by introducing self-configuration, self-optimization, and self-healing features [1]. With the introduction of Long Term Evolution (LTE) or the future 5th Generation Mobile Networks (5G), the complexity of radio networks increases considerably making SON a crucial component of future network management [1], [2]. SON defines a set of use cases that are automatically performed by autonomous SON functions. For self-optimization, this means that a single function, e.g., Coverage and Capacity Optimization (CCO), Mobility Robustness Optimization (MRO), or Mobility Load Balancing (MLB), optimizes the performance of each single cell with respect to one or a few network Key Performance Indicators (KPIs), e.g., the Channel Quality Indicator (CQI), handover ping-pong rate, or cell load, by adjusting one or a few network configuration parameters, e.g., Remote Electrical Tilt (RET) or Cell Individual Offset (CIO).

Operators of mobile networks have specific objectives on the network performance that can be expressed as KPI target values [3]. Based thereon, SON operations is the process of instrumenting the SON functions such that the mobile network is optimized with respect to these objectives. This comprises two tasks: SON management and SON coordination [1].

SON management configures SON functions such that they jointly optimize the network regarding the objectives. In [3], [4], the SON objective manager is presented that enables operators to manage a SON with machine-readable objectives. These objectives are rules that, given an operational context, define desired values and importance for each KPI. For instance, `IF location=rural THEN load ≤ 0.8 WITH 0.5` defines that the cell load in a rural area should be below 80% with medium importance. Since the operator typically does not know the functional details of SON functions, the vendor of a SON function is supposed to provide a SON function model that defines the expected values of the KPIs that are affected by the SON function, depending on the configuration of the function. For instance, the MLB function configuration MLB_1 with the effect $(\text{load}, \leq 0.8)$ means that MLB_1 is expected to keep the cell load below 80%.

Since a SON function is not observe other functions, their concurrent execution can lead to conflicts resulting in inferior performance [1]. Such a conflict happens if, e.g., two SON functions attempt to simultaneously change the same network parameter. SON coordination detects and resolves conflicts by controlling the deployment of configuration changes. Therefore, SON functions request changes at SON coordination which accepts or rejects the requests such that the accepted changes are conflict-free. Previously presented conflict resolution approaches [5]–[8], thereby, do not consider the operator objectives for their decision making.

We present a new, objective-driven conflict resolution approach that builds on the SON objective manager, specifically formalized operator objectives and SON function models. The contributions of the paper are:

- we design a SON coordination that resolves SON function conflicts such that the satisfaction of operator objectives is maximized as quickly as possible,
- we embed the coordination into the SON objective manager system,
- we generalize the SON objective manager concepts using multiattribute utility theory by representing the objectives as utility functions and by providing probabilistic semantics for the SON function models,
- we model and solve conflict resolution as a constraint optimization problem, and
- we present an evaluation showing the advantages.

As a result, the operator can control SON coordination without detailed knowledge of the SON functions by setting the objectives and letting the system optimize the network accordingly.

II. Concept

The goal of objective-driven SON coordination is to make coordination decisions guided by operator objectives. In order

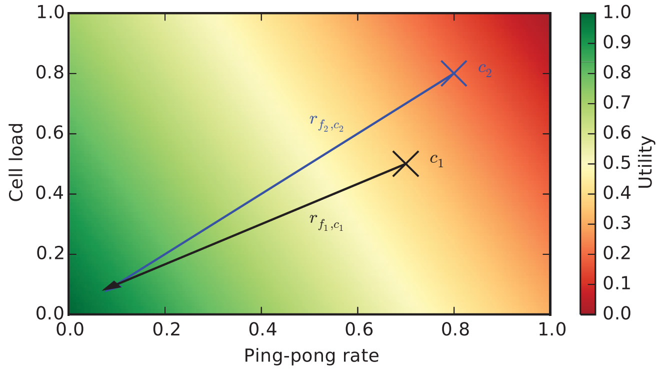

Figure 1. Performance of the cells c_1 and c_2 (indicated by the color) and expected effects of the conflicting SON function requests r_{f_1,c_1} and r_{f_2,c_2}.

to outline this goal, consider the simple situation depicted in Figure 1: it shows the performance of two network cells, c_1 and c_2, regarding the two KPIs handover ping-pong rate, i.e., the ratio of quickly reverted handovers from a cell, and the cell load, i.e., the utilized resources of the cell [1]. Thereby, c_1 performs better than c_2 since the lower the ping-pong rate or the load, the better. Consider the two SON function requests r_{f_1,c_1} and r_{f_2,c_2} targeting c_1 and c_2 as conflicting. Faced with this situation, it is reasonable to concentrate on the worse performing cell first, however, only if the expected overall improvement is reasonably high. In other words, the operator shall compare the estimated performance improvement with respect to the objectives for the two requests and select the one with the higher gain. The goal of objective-driven SON coordination is to automate such decision making.

In the following, we assume a mobile network with C cells $\mathcal{C} = \{1, \ldots, C\}$, F SON functions $\mathcal{F} = \{1, \ldots, F\}$, and K KPIs $\mathcal{K} = \{1, \ldots, K\}$ each with the domain $\mathrm{dom}(k)$, $k \in \mathcal{K}$. On each cell $c \in \mathcal{C}$, there is an independent instance f_c of each SON function $f \in \mathcal{F}$ executed. Triggered by certain events, the SON objective manager configures each SON function instance f_c with a configuration, referred to as SON Function Configuration Parameter Value (SCV) set, $s_{f,c} \in S_f$ according to the SON function model, the objective model, and the operational context [4]. We assume batch action coordination also referred to as synchronized execution [8]. Hence, a SON function instance f_c monitors the performance of c and sends a request $r_{f,c}$ to the SON coordination if it is necessary to change the parameters of c. The requests are collected into the set R which is processed by coordination in periodic time intervals. After that, each request $r_{f,c} \in R$ is either accepted, allowing f_c to perform the parameter change, or rejected.

The objective-driven SON coordination is embedded into a SON that is managed by the SON objective manager as depicted in Figure 2. The SON coordinator is configured together with the SON function instances by passing the current configuration of all SON functions for all cells, referred to as SON configuration $SC = \{s_{f,c} : c \in \mathcal{C}, f \in \mathcal{F}\}$. Furthermore, the SON coordinator is given the objective model, the SON function models, and access to the operational context, i.e., Configuration Management (CM), Performance Management (PM), and Failure Management (FM) data, of every cell in the network. Finally, a conflict detection model is required allowing the recognition of potential SON functions conflicts.

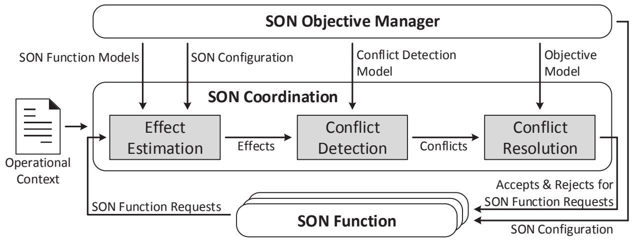

Figure 2. Functional overview of the objective-driven SON coordination.

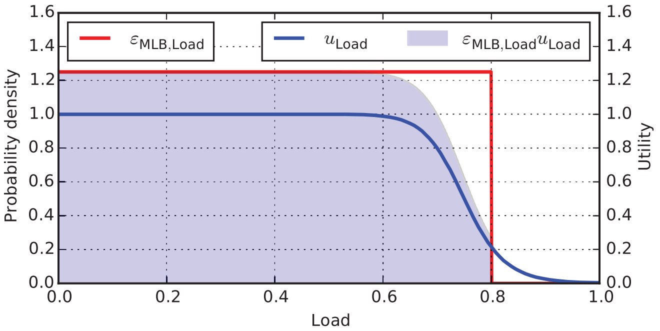

Figure 3. Expected KPI utility for KPI effect $\varepsilon_{\text{MLB,Load}}$ and utility u_{Load}.

Figure 2 also outlines the process performed by objective-driven coordination when processing of the collected requests is triggered. First, for all $r \in R$ the effects are estimated. Second, R is analyzed for possible conflicts. Third, the effects of conflicting requests are evaluated based on the operator objectives and a conflict-free subset of actions that maximizes the satisfaction of the objectives is determined. The requests in this set are accepted and the others rejected.

A. Effect Estimation

The idea of effect estimation is to utilize the SON function models from the SON objective manager in order to estimate the performance of cell c after the execution of a SON function request $r_{f,c} \in R$, i.e., after the acceptance of the request.

1) SON Function Model: The SON objective manager concept assumes that it is possible to estimate the effects that a specific SON function configuration $s_{f,c}$ has on the performance of cell c [4]. Specifically, for each SON function $f \in \mathcal{F}$, the vendor of f is supposed to provide a SON function model SFM_f which allows to determine the effects of each possible configuration for f in a specific operational context. In [9], the authors outline an approach to create such a model based on simulations. Hence, we define $SFM_f(s_{f,c}) = \varepsilon_{f,c}$ to be the effect of the SON function configuration $s_{f,c}$ on the cell c in the current operational context. An effect $\varepsilon_{f,c} = \{\varepsilon_{f,c,k} : k \in \mathcal{K}\}$ is a set of expected values for each KPI k whereby the KPI effect denotes a range of expected values for k given the configuration $s_{f,c}$ in the current context.

In this paper, a KPI effect is defined as a probability density function over the KPI values: a KPI effect $\varepsilon_{f,c,k} \in \mathcal{F}_k \cup \{\perp\}$ can either be a probability density function $F_k \in \mathcal{F}_k$:

$\mathrm{dom}(k) \to \mathbb{R}^+$, with $\int_{\mathrm{dom}(k)} F_k(v)\,\mathrm{d}v = 1$, or $\perp$ indicating that $s_{f,c}$ does not affect k. This allows modeling of more complex expected effects, e.g., a normal distribution. Note that the expected value ranges used in [4] can be modeled as uniform probability densities that are 0 for unexpected KPI values. For instance, Figure 3 depicts the KPI effect $\varepsilon_{\mathrm{MLB,Load}}(v) = \begin{cases} 1.25 & v \leq 0.8 \\ 0 & v > 0.8 \end{cases}$ on the load by an MLB request $r_{\mathrm{MLB},c}$ with the SON function configuration $s_{\mathrm{MLB},c}$ as a red line. It means that the load is expected to be below 80%.

2) Effect combination: Based on the SON function configuration $s_{f,c}$ and the SON function model SFM_f, the SON coordinator can estimate the effects $\varepsilon_{f,c}$ of each SON function $f \in \mathcal{F}$. Therefore, it is assumed that $\varepsilon_{f,c}$ also represents the expected future effects of the execution of a respective SON function request $r_{f,c} \in R$. Although the effects typically do not manifest after the execution of one $r_{f,c}$, in the long-run, i.e., after multiple executions of $r_{f,c}$, this assumption does hold.

A SON function might not affect all KPIs, i.e., for some KPI k, $\varepsilon_{f,c,k} = \perp$. In such cases, the value of k will be the same before and after the execution of $r_{f,c}$. Hence, the KPI effect $\varepsilon_{r_{f,c},k} : \mathcal{F}_k$ of a request $r_{f,c}$ on the KPI k is

$$\varepsilon_{r_{f,c},k} = \begin{cases} \varepsilon_{f,c,k} & \text{if } \varepsilon_{f,c,k} \neq \perp \\ \delta_{\chi(c,k)} & \text{otherwise,} \end{cases} \tag{1}$$

with $\chi(c,k)$ denoting the current monitored value of k in cell c and δ_η denoting a Dirac delta function, i.e., a probability density function assigning the probability 1 to η, thereby representing a deterministic value in a probabilistic model. For instance, if an MLB request $r_{\mathrm{MLB},c}$ only affects the load, then expected CQI is $\varepsilon_{\mathrm{MLB,CQI}}(v) = \delta_{\chi(c,\mathrm{CQI})}$ and the expected handover ping-pong rate $\varepsilon_{\mathrm{MLB,PiPo}}(v) = \delta_{\chi(c,\mathrm{Pipo})}$.

B. Conflict Detection

Research has identified numerous conflict categories between two SON functions which might also depend on spatial and temporal characteristics of the functions [1], [6]. Objective-driven SON coordination focuses on conflict resolution and, hence, we draw on known approaches for conflict detection, e.g., [1], [5], [7]. Independent of the concrete implementation, conflict detection provides a conflict relation κ for the collected request R:

$$\kappa = \{(r_1, r_2) \in R \times R : r_1 \text{ and } r_2 \text{ are in conflict}\}. \tag{2}$$

κ is irreflexive and symmetric, but not necessarily transitive.

C. Conflict Resolution

The conflict resolution resolves the conflicts among the SON function requests by accepting and rejecting them based on their calculated utility, i.e., the expected performance improvements with respect to the operator objectives. Requests that are not in conflict can be immediately accepted.

1) Objective Model: In [4], the objective model OM defines the KPI objectives for all KPIs that are applicable in a specific operational context. That means that each cell $c \in \mathcal{C}$ has its own KPI objectives $o_{c,k}$ for all KPIs $k \in \mathcal{K}$. A KPI objective $o_{c,k} = (t_{c,k}, w_{c,k})$ is a pair of a KPI target $t_{c,k} \subseteq \mathrm{dom}(k)$, i.e., a set of desired KPI values, and a weight $w_{c,k} \in [0,1]$ representing the importance of the target.

In this paper, this objective model is extended and based on multiattribute utility theory [10]. This theory is based on the von Neumann-Morgenstern expected utility theory which allows to formalize decision making in probabilistic settings if the preferences over stochastic outcomes of some decision satisfy an ordering, independence, and continuity assumption: let p and q be probability distributions over the possible outcomes of the variable X, then there exists a real-valued utility function $u(\cdot)$ over X such that the decision option inducing p is preferred to the option inducing q, denoted as $p \succ q$, if and only if the expected utility of p is greater than the expected utility of q, i.e., $\mathrm{E}_p[u] > \mathrm{E}_q[u]$ with $\mathrm{E}_p[u] = \int_X p(x)u(x)\,\mathrm{d}x$ and $\mathrm{E}_q[u] = \int_X q(x)u(x)\,\mathrm{d}x$. Furthermore, if the possible outcomes of X are characterized by m attributes, i.e., $X = \prod_{i=1}^m X_i$, then, given additive utility independence, $u(\cdot)$ can be decomposed into an additive, multiattribute utility function $u(x) = \sum_{i=1}^m k_i u_i(x_i)$ with $x \in X$, $x = (x_1, \ldots, x_m)$, which is based on utility functions $u_i(\cdot)$ and trade-off weights k_i for each criterion X_i.

In this paper, a KPI objective $o_{c,k} = (u_{c,k}, w_{c,k})$ is a pair consisting of a utility function $u_{c,k} : \mathrm{dom}(k) \to [0,1]$ mapping a KPI value to a real number between 0 and 1, and a normalized weight $w_{c,k} \in [0,1]$ with $\forall c \in \mathcal{C}.\ \sum_{k \in \mathcal{K}} w_{c,k} = 1$. The form of the utility function depends on the preferences of the operator. For instance, Figure 3 depicts the KPI objective $u_{\mathrm{Load}}(v) = 1 - \frac{(v/0.75)^{20}}{1+(v/0.75)^{20}}$ on the KPI load as a blue line. It means that the load should be below 80%. Refer to [11] for a comparative study of typical functions in the area of mobile networks. In order to model a desired value range as in [4], the utility function can be defined as a piecewise linear function that is 1 for the desired KPI values, and linear decreasing to 0 for the undesired values, whereby the further away a KPI value is from the desired value range, the smaller the utility.

2) Utility Calculation: Based on multiattribute utility theory, a conflict between two SON function requests is resolved by considering the operator preferences expressed in the objective model. The expected utility of the effect $\varepsilon_{r_{f,c}} = \prod_{k=1}^K \varepsilon_{r_{f,c},k}$ of a SON function request $r_{f,c}$ is calculated as the weighted sum of the expected utilities per KPI:

$$\begin{aligned} \mathrm{E}_{\varepsilon_{r_{f,c}}}[u_c] &= \mathrm{E}_{\varepsilon_{r_{f,c}}}\Big[\sum_{k \in \mathcal{K}} w_{c,k} u_{c,k}\Big] = \sum_{k \in \mathcal{K}} w_{c,k}\, \mathrm{E}_{\varepsilon_{r_{f,c},k}}[u_{c,k}] \\ &= \sum_{k \in \mathcal{K}} w_{c,k} \int\limits_{\mathrm{dom}(k)} \varepsilon_{r_{f,c},k}(v) u_{c,k}(v)\,\mathrm{d}v \end{aligned} \tag{3}$$

Figure 3 outlines this calculation for the KPI effect $\varepsilon_{\mathrm{MLB,Load}}$ and the KPI objective u_{Load} as a blue area. The resulting expected KPI utility for the cell load is $\mathrm{E}_{\varepsilon_{\mathrm{MLB,Load}}}[u_{\mathrm{Load}}] = 0.93$.

Since two conflicting requests r_{f_1,c_1} and r_{f_2,c_2} might be executed on different, maybe neighboring cells, the utility of r_{f_1,c_1} also needs to consider the missed performance improvement by r_{f_2,c_2}. To outline this, consider the scenario in Figure 1: the requests r_{f_1,c_1}, which aims to improve the performance of cell c_1, and r_{f_2,c_2}, which does the same for cell c_2, are in conflict. Both requests are expected to improve the respective cell's performance to an equal state (marked by the tips of the arrows). Hence, assuming the same objectives, the expected resulting performance for both requests is equal. However, since the performance of c_2 was much worse than c_1's before the optimization (indicated by the color), it is obvious that an operator shall prefer r_{f_2,c_2}. Following this argument, the conflict resolution needs to be based on the difference in the utility of the cell before and the expected utility after the execution of the SON function request as

$$\Delta u_{r_{f,c}} = \mathrm{E}_{\varepsilon_{r_{f,c}}}[u_c] - u_c(\chi_c) \tag{4}$$

whereby $u_c(\chi_c) = \sum_{k \in \mathcal{K}} u_{c,k}(\chi_c(k))$ denotes the utility of the current performance of cell c, and $\chi_c(k)$ denoting the current value of KPI k in c.

3) Request Selection: The request selection computes a conflict-free subset of the collected SON function requests $R_{\overline{\kappa}} \subseteq R$ that maximizes the satisfaction of the operator objectives. These requests $r \in R_{\overline{\kappa}}$ are accepted, i.e., the corresponding SON function can perform the configuration changes, whereas the requests $r \in R \setminus R_{\overline{\kappa}}$ are rejected.

Just like the estimated performance of one SON function request is an outcome with K attributes, $R_{\overline{\kappa}}$ can be seen as an action that produces an outcome with $|R_{\overline{\kappa}}|$, i.e., the cardinality of $R_{\overline{\kappa}}$, attributes. So, assuming that the operator's preferences regarding the performance of a network cell are additive utility independent (cf. Section II-C1) of the performance of the other cells, the overall utility of $R_{\overline{\kappa}}$ can be decomposed into a weighted sum of the utilities of the requests $r \in R_{\overline{\kappa}}$. Consider w_c to be an operator-provided weight of cell c representing the importance of c. Consequently, the utility of the conflict-free requests is $u_{R_{\overline{\kappa}}} = \sum_{r_{f,c} \in R_{\overline{\kappa}}} w_c \Delta u_{r_{f,c}}$. Notice that this summation overestimates the actual utility if two accepted SON function requests $r_{f_1,c}$ and $r_{f_2,c}$ affect the same KPI k in cell c, i.e., $\varepsilon_{r_{f_1,c},k} \neq \perp$ and $\varepsilon_{r_{f_2,c},k} \neq \perp$. In this case the utility of the improvement of k would be counted twice although it will only be optimized once. However, such a case should not occur since two requests that affect the same KPI in the same cell are typically seen as conflicting [1].

Since the conflict relation does not need to be transitive, the conflicts cannot be resolved by iteratively selecting the action with the highest utility for each transitive closure of the conflicts. In order to outline this, consider three SON function requests $R = \{r_1, r_2, r_3\}$ with the utilities $\Delta u_{r_1} = 0.5$, $\Delta u_{r_2} = 0.4$ and $\Delta u_{r_3} = 0.3$, as well as the conflicts $\kappa = \{(r_1, r_2), (r_1, r_3)\}$. The action with the highest utility in both conflicts is r_1, however, the conflict-free subset of the requests R with the maximal utility is $\{r_2, r_3\}$.

The optimization problem of conflict resolution can be mapped to a constraint optimization problem [12]. Although such problems are, in general, NP-hard, modern solvers can exploit specific problem structures to solve it more efficiently.

$$\max u_{R_{\overline{\kappa}}} = \sum_{r_{f,c} \in R} w_c \Delta u_{r_{f,c}} var_{r_{f,c}} \tag{5}$$

$$\text{subject to } \forall (r_i, r_j) \in \kappa.\ var_{r_i} = 0 \vee var_{r_j} = 0 \tag{6}$$

$$\forall r_i \in R.\ var_{r_i} \in \{0, 1\} \tag{7}$$

Equation 7 defines a set of binary variables, one for each SON function request $r_i \in R$, that is optimized by the solver. Thereby, $var_{r_i} = 1$ is indicating acceptance of r_i and $var_{r_i} = 0$ is indicating rejection of r_i. Equation 6 poses a constraint for each conflict $(r_i, r_j) \in \kappa$ that the conflict partners cannot both be accepted, i.e., at least one of the variables must be 0. Finally, Equation 5 defines the optimization target as to maximize the weighted sum of the utilities of the accepted SON function requests. After solving this problem, the set of accepted SON function requests can be computed as $R_{\overline{\kappa}} = \{r_i \in R : var_{r_i} = 1\}$. Note that $R_{\overline{\kappa}}$ is conflict-free with respect to κ, i.e., $\forall r_i, r_j \in R_{\overline{\kappa}}.\ (r_i, r_j) \notin \kappa$.

III. Evaluation

In the following, we compare the performance of objective-driven SON coordination with a policy-based SON coordination scheme in a realistic network simulation.

A. Scenario

The evaluation is performed using a network simulator for an urban LTE network in the city of Helsinki, Finland (presented in detail in [13]). The main technical parameters of the simulations are summarized in Table I.

To simplify the description, only three KPIs are considered: the weighted harmonic mean of CQI channel efficiency [14], the rate of handover ping-pongs (PiPo), and the cell load based on the average utilized Physical Resource Blocks (PRBs). The objectives for these KPIs, shown in Table I, are sigmoidal utility functions as proposed in [11] and can be interpreted as: the CQI should be above 0.6, the ping-pong rate should be below 5%, and the cell load should be below 0.8. The configuration of the network is optimized by three SON functions which are configured according to the objectives (cf. [1] for more detailed information on the algorithms):

CCO optimizes the coverage and capacity of a network cell, i.e., triggered by a low CQI, it adjusts the RET such that the CQI improves.

MRO optimizes the handover performance between two neighboring cells, i.e., triggered by a high rate of handover ping-pongs, it adjusts the CIO such that the ping-pongs are reduced.

MLB optimizes the load of a cell, i.e., triggered by a high load, it adjusts the CIO such that more users are handed over to neighboring cells and, thus, the load is reduced.

On every cell in the network, there is an instance of each SON function running. The execution of the SON function instances is performed in *rounds* of 100 simulated minutes, i.e., all instances are synchronously triggered at the end of every round. If an instance detects a problem, it sends a SON function request to the SON coordination component, which collects all requests, coordinates them, and finally notifies the instances about their acceptance or rejection. If a SON function request is accepted, the instance performs the configuration change at the beginning of the next round.

The conflict detection is based on the impact area of the SON function requests [1] and dependency rules (cf. Table I). For the conflict resolution, we compare the objective-driven coordination with the policy-based approach presented in [5] that prioritizes SON functions. For the latter, the CCO function is more important than the MRO function which, in turn, is more important than the MLB function. The SON function

Table I. SIMULATION SETUP

Category	Parameter	Value
Network	Carrier frequency / bandwidth	2 GHz / 20 MHz
	Path loss model	UMTS 30.03 [15]
	Number of cells / area	32 / 50 km^2
	User mobility model	6 km/h / random walk
	Users normal / hot spot	1500 / 150
	Data rate normal / hot spot	200 / 100 kbps Constant Bit Rate
	Handover hysteresis threshold	2.0 dB
	Radio Link Failures threshold	-6.0 dB
Conflict detection	(CCO, CCO/MRO/MLB)	overlapping impact area
	(MRO, MLB)	overlapping impact area
	(MLB, MLB) / (MRO, MRO)	none / none
	Impact area CCO	cell + 1st degree neighbors
	Impact area MRO, MLB	cell pair
SON function models (not normalized)	CCO: $\varepsilon_{\text{CCO,CQI}}(v)$	$1 \quad : v \geq 0.6$ $0 \quad : v < 0.6$
	MRO: $\varepsilon_{\text{MRO,PiPo}}(v)$	$1 \quad : v \leq 0.05$ $0 \quad : v > 0.05$
	MLB: $\varepsilon_{\text{MLB,Load}}(v)$	$1 \quad : v \leq 0.8$ $0 \quad : v > 0.8$
Objective model	CQI: $u_{\text{CQI}}(v)$	$\frac{(v/0.5)^{20}}{1+(v/0.5)^{20}}$
	Ping-pong rate: $u_{\text{PiPo}}(v)$	$1 - \frac{(v/0.075)^{15}}{1+(v/0.075)^{15}}$
	Cell load: $u_{\text{Load}}(v)$	$1 - \frac{(v/0.9)^{40}}{1+(v/0.9)^{40}}$
	Objective weights	$w_{\text{CQI}} = w_{\text{PiPo}} = w_{\text{Load}} = 1/3$
	Cell weights	$\forall c \in \mathcal{C}.\, w_c = 1$
SON function priorities		CCO $\succ$ MRO $\succ$ MLB

models for the objective-driven conflict resolution are derived from the configuration of the SON functions. Hence, each SON function affects only the KPI that it is supposed to optimize and the KPI effect is derived from the triggering threshold. For instance, a CCO function request only affects the CQI and produces a KPI value greater than 0.6 with uniform probability.

The simulation starts with a non-optimal network configuration in order to stimulate optimization by the SON functions. In the following, we concentrate on one three-sectored Base Station (BS): one cell has a too low CIO resulting in an increased ping-pong rate, another cell has a suboptimal RET setting leading to a reduced CQI, and the third cell covers a hot spot area rendering it overloaded. Since all three cells are neighbors, there will be conflicts between the CCO instance optimizing the RET, and the MRO and MLB instances optimizing the CIOs. Although this scenario has been chosen to exemplify the advantage of the objective-driven coordination approach, such a case can also happen in reality, e.g., if a new BS is introduced into the network.

B. Results

Figure 4 and Figure 5 depict the results for the policy-based and the objective-driven coordination. The graphs show the means and standard deviations of the KPI utilities and accepted SON function requests for the three degraded cells over three simulation runs. From the stacked KPI utility means, one can easily derive the mean overall utility as the sum of the mean KPI utilities. In all simulations, the initial network configuration is equal but the user distribution is random.

The policy-based conflict resolution is driven by the function priorities. As shown in Figure 4, it is first accepting every CCO function request. All MRO and MLB requests are rejected until the CCO function is satisfied. This improves the utility of the CQI objective. Once there are no more requests

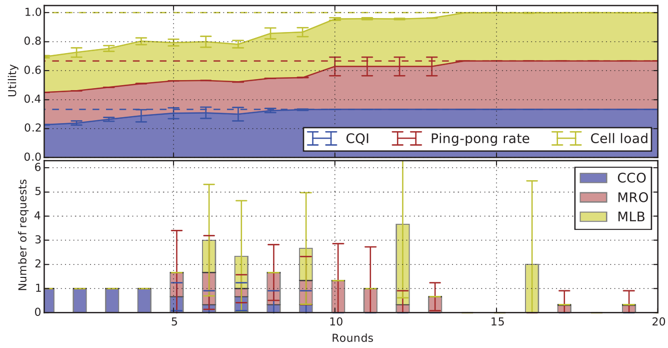

Figure 4. Simulation results of the policy-based SON coordination.

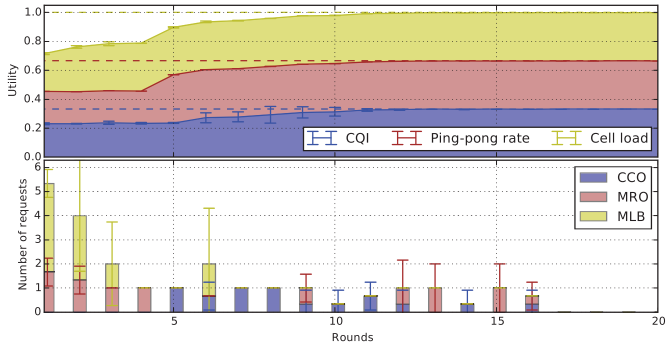

Figure 5. Simulation results of the objective-driven SON coordination.

by CCO starting in Round 5, it focuses on the improvement of the ping-pong rate: it accepts all MRO requests and rejects conflicting MLB requests. Just at Round 6, the first MLB requests, which are not conflicting with some still executed MRO function requests, are accepted. Based on the KPIs, it can be seen that the SON finally adapted the network configuration to the new situation around Round 14. Later requests are caused by random variations in the simulations.

The objective-driven coordination, shown in Figure 5, first analyses the possible improvements of every request, i.e., their effects. In this scenario, this can be seen as the difference of each KPI utility of the current system performance to the maximal KPI utility $\frac{1}{3}$ indicated by the dashed lines. Then, it determines which conflict-free subset of the requests improves the utility the most. Thus, it computes that the concurrent execution of the MRO and MLB requests yields a higher gain than the execution of the RET request. Consequently, the MRO and MLB requests are accepted in the beginning of the simulation. Just later on, when the CIOs are close to optimal in Round 5, the CCO requests are accepted and the CQI is improved. Considering the KPIs, the SON finally adapted network configuration to the new situation around Round 13.

Comparing policy-based and objective-driven coordination, it can be seen that both achieve full objective satisfaction after around 14 rounds. Thereby, small standard deviations in the KPI utilities indicate consistent behavior of both approaches. The little time difference between them is not surprising: in principle, the scenario requires a common number of accepted SON function executions, to adapt the network to the new situation. Furthermore, due to the common conflict-detection, there is theoretically the same overall selection of conflict-free SON function request sets both approaches can choose from. However, the objective-driven SON coordination achieves great utility increases already in the beginning of the simulation between Round 4 and 5 because it concentrates first on the important problems, with respect to the operator objectives, in the network. In contrast, the policy-driven coordination achieves these improvements later between Round 9 and 10. Of course, this result depends on the specific priorities of the SON functions in this scenario, and another order could lead to better performance. However, following this argument would require the operator to adapt the priorities for each and every coordination situation which, actually, is a manual implementation of the objective-driven conflict resolution. This summarizes the general goal of objective-driven coordination: focus on the worst problems regarding the operator objectives that can be overcome quickly.

IV. Related work

Most proposals for SON coordination are either based on a fixed policy, i.e., a set of rules, or on some simple evaluation of the estimated effects of a SON function. [5], [6], and the “Multi-priority event driven separation in time” in [7] belong to the former category. Thereby, the operator prioritizes the SON functions requiring detailed knowledge how each SON function affects the KPIs. Although the policy allows some adaptation to the operational context, this is typically limited to CM data, e.g., the network cell location, and does not consider KPI values. In contrast, the objective-driven coordination allows the operator to define KPI objectives and let the system automatically determine the best coordination decision based on them without human involvement. Note that objective-driven coordination with a policy requires the non-trivial calculation of the best action for all possible combinations of SON function requests in all possible KPI value combinations.

Other approaches are based on estimated effects of executed SON functions. The “Self-orchestration through Utility Predicates” [7] requires the SON functions to predict the utility of their actions. However, the actual utility calculation is not presented. In [8], a similar concept is presented that requires the SON functions to report a “happiness”, a simple technical performance indication, along with the requests. Using reinforcement learning, the approach learns which combination of accepted requests yields the highest expected improvement in the happiness. However, learning the happiness, i.e., the combined performance evaluation over several KPIs, renders the learned knowledge useless if the objectives, in this case simple KPI thresholds, change. In contrast to the objective-driven coordination, these approaches do not consider the operator defined, SON function independent KPI objectives for their decision making. Furthermore, they only consider simple SON function conflicts and do not provide an optimal solution for complex conflicts spanning several network cells.

V. Conclusion

This paper presented an objective-driven conflict resolution approach for Self-Organizing Network (SON) coordination that is based on the SON objective manager concept. The introduction of probabilistic SON function models, which estimate the expected performance effects of a SON function instance, and multiattribute utility theory-based objectives, which describe the operator preferences regarding the network performance, enable the valuation of change requests by SON function instances. Based on these utilities, it is possible to determine a conflict-free subset of the SON function requests that maximizes the satisfaction of the objectives by solving a constraint optimization problem. As a result, the operator can control SON coordination without detailed knowledge of the SON functions by solely setting the objectives and letting the system optimize the network to satisfy them. A simulation-based evaluation shows that the approach focuses the SON function activities to severe network problems first, thus, improving the overall network performance faster than related approaches. In the future, we plan to improve the effect estimation by the SON function models with machine learning.

References

[1] S. Hämäläinen, H. Sanneck, and C. Sartori, Eds., *LTE Self-Organising Networks (SON): Network Management Automation for Operational Efficiency*. Chichester, UK: John Wiley & Sons, Dec. 2011.

[2] NGMN Alliance, “5G White Paper,” Next Generation Mobile Networks, White paper, 2015.

[3] C. Frenzel, S. Lohmüller, and L. C. Schmelz, “Dynamic, context-specific SON management driven by operator objectives,” in *2014 IEEE Network Operations and Management Symposium (NOMS)*, Krakow, Poland, May 2014.

[4] ——, “SON management based on weighted objectives and combined SON Function models,” in *2014 11th International Symposium on Wireless Communications Systems (ISWCS)*, Barcelona, Spain, Aug. 2014.

[5] T. Bandh, R. Romeikat *et al.*, “Policy-based coordination and management of SON functions,” in *12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011)*, Dublin, Ireland, May 2011.

[6] H. Y. Lateef, A. Imran, and A. Abu-Dayya, “A framework for classification of Self-Organising network conflicts and coordination algorithms,” in *2013 IEEE 24th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, London, UK, 2013.

[7] K. Tsagkaris, A. Galani *et al.*, “Unified Management Framework (UMF) Specifications Release 3,” UniverSelf Project, Deliverable D2.4, Nov. 2013.

[8] O. Iacoboaiea, B. Sayrac *et al.*, “SON Coordination for parameter conflict resolution: A reinforcement learning framework,” in *2014 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, Istanbul, Turkey, Apr. 2014.

[9] S. Hahn and T. Kürner, “Managing and altering mobile radio networks by using SON function performance models,” in *2014 11th International Symposium on Wireless Communications Systems (ISWCS)*, Barcelona, Spain, Aug. 2014.

[10] J. S. Dyer, “Maut - Multiattribute Utility Theory,” in *Multiple Criteria Decision Analysis: State of the Art Surveys*, J. Figueira, S. Greco, and M. Ehrogott, Eds. New York, USA: Springer, 2005, ch. 7.

[11] Q.-T. Nguyen-Vuong, Y. Ghamri-Doudane, and N. Agoulmine, “On utility models for access network selection in wireless heterogeneous networks,” in *2008 IEEE Network Operations and Management Symposium (NOMS)*, Salvador, Brazil, Apr. 2008.

[12] F. Rossi, P. van Beek, and T. Walsh, *Handbook of Constraint Programming*, 1st ed. Amsterdam, Boston: Elsevier B.V., 2006.

[13] T. Tsvetkov, H. Sanneck, and G. Carle, “A Graph Coloring Approach for Scheduling Undo Actions in Self-Organizing Networks,” in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, Ottawa, Canada, 2015.

[14] 3GPP, “Evolved Universal Terrestrial Radio Access (E-UTRA); Physical layer procedures,” TS 36.213, 2015.

[15] ——, “Universal Mobile Telecommunications System (UMTS); Selection procedures for the choice of radio transmission technologies of the UMTS,” TR 101 112, 1998.