An Adaptive Observation Window for Verifying Configuration Changes in Self-Organizing Networks

Tsvetko Tsvetkov
Department of Computer Science
Technische Universität München
tsvetko.tsvetkov@in.tum.de

Janne Ali-Tolppa
Nokia Networks
Munich, Germany
janne.ali-tolppa@nokia.com

***Abstract*—The automatic verification of Configuration Management (CM) changes is an important step towards a highly-optimized Self-Organizing Network (SON). A verification mechanism operates in three steps: based on the CM changes it divides the network into verification areas, assesses those by using an anomaly detection algorithm, and generates CM undo requests for the abnormally performing ones. To successfully fulfill those tasks, it has to sample the network for a certain time period, called an observation window. However, if the mechanism is timed improperly, it may generate too many false positive undo requests and may even prevent SON functions from reaching their set goals.**

To overcome this issue, we calculate for each cell a Cell Verification State Indicator (CVSI). It is based on the deviation from the expected performance and is updated using exponential smoothing. A verification area continuously reporting low CVSI values is considered as degraded and processed by the verification mechanism. The presented approach is evaluated in a simulated environment and compared it with other verification strategies. The results show that we are able to get a better result when we consider the CVSIs instead of the absolute performance values of the areas.

I. Introduction

The Self-Organizing Network (SON) concept as we know it today has been developed to deal with the complex nature of standards like Long Term Evolution (LTE) and LTE-Advanced. The main idea behind it is to optimize the operation of the network, supervise the configuration and auto-connectivity of newly deployed Network Elements (NEs), and enable automatic fault detection and resolution [1]. A SON-enabled network is, therefore, managed by a set of autonomous SON functions performing specific network management tasks. Those functions are designed as control loops, which monitor Performance Management (PM) and Fault Management (FM) data, and based on their goals, change Configuration Management (CM) parameters. For example, the Coverage and Capacity Optimization (CCO) function has been developed to optimize the coverage within a cell by changing the antenna tilt or the transmission power.

Despite the closed-loop automation of SON functions, there is a need to verify the combined performance impact of the deployed CM changes. Therefore, the concept of SON verification has been developed [2]–[4]. It is seen as a special type of anomaly detection that implements a so-called *verification process*. The outcome is either to accept the deployed CM changes or to revert them back to a previous stable state, which is also known as a *CM undo request*, or undo request for short. The verification process itself operates in three phases: (1) it divides the network into verification areas according to the CM changes, (2) runs an anomaly detection algorithm for each area, and (3) marks the changes for an undo that are most likely responsible for causing a degradation. Finally, it schedules CM undo requests for deployment.

Despite the progress made in the development of SON verification, it still has shortcomings in LTE. In particular, a verification mechanism experiences difficulties if it is timed improperly. In order to detect an anomalously performing area, it has to sample the network for a certain time, also referred to as an observation window. However, if this window is too short, a verification mechanism may try to rollback short, transient performance degradations, and therefore, generate false positive undo requests. That is, SON functions trying out different CM parameter settings, which induce temporary and short performance drops, may get interrupted as they try reaching their goals. Furthermore, having numerous undo requests implies that a lot of verification areas require performance assessment, which may delay the complete verification process.

In this paper, we propose an approach that overcomes those issues. The proposed solution dynamically adapts the length of the observation duration based on a so-called Cell Verification State Indicator (CVSI). The latter one is computed based on exponential smoothing of the deviation from the expected performance. Further, the factor used to update the CVSI depends on this deviation as well. The expected performance itself is an aggregation of the expected values of Key Performance Indicators (KPIs) a cell reports. A verification area, whose cells are reporting low CVSIs, is marked as ready to be further processed by the verification process. On the contrary, an area consisting of cells having high CVSIs is skipped by the verification process.

The remainder of this paper is organized as follows. Section II describes how the verification process is realized today. In Section III we present the challenges SON verification still has in LTE. In Section IV we describe our concept, whereas in Section V we evaluate it in a simulated LTE environment. Section VI is devoted to the evolution of SON verification for future standards, like the fifth generation of mobile communications (5G). Further, our paper includes a description of the related work in Section VII and a summary in Section VIII.

II. Verification process

1) Composition of verification areas: A verification area consists of the reconfigured cell, also called the *target cell*, and a set of cells impacted by the CM change(s), also referred to as a *target extension set* [5]. The selection of the latter one is based on the neighbor relations between cells. One possible way is to take the reconfigured cell and all of its direct neighbors to which User Equipments (UEs) can be handed over. Furthermore, if we have SON function activities in the network, we may define the verification area as the impact area [6] of the SON function that has been recently active. We may also enlarge a verification area based on its location [7], e.g., more cells are added if they are part of known trouble spots or dense traffic.

2) Detecting anomalies: An anomaly is understood as "something that deviates from what is standard, normal, or expected" [8]. In this paper, however, we focus on detecting abnormal cell behavior, i.e., the performance of a cell has notably degraded. Usually, it is done by profiling [9] the network behavior, which requires analyzing the network performance indicators and specifying how the network should typically behave. For instance, in [4] an anomaly detection technique for cellular networks is introduced. It is based on the extended version of the incremental clustering algorithm Growing Neural Gas (GNG) which partitions the input data into smaller groups with a similar behavior. The presented method is able to identify unusual behavior in the time domain. An example is a cell remaining a whole week in a state that represents the weekend.

3) Generating CM undo requests: An undo request is generated for each verification area that has been marked as degraded. The request itself has three data fields: one that contains the target cell identifier, another that lists all cell identifiers of the target extension set, and the requested CM settings for the target cell. Those settings can be a complete snapshot of a stable configuration the cell has previously had. It can also be a partial list, for example, if we rollback changes made by SON functions.

III. SON verification challenges

A. Consequences of improper timing

Even today, a drop in performance does not necessarily mean that we are obligated to immediately undo CM changes. Many of today's SON functions require not only one step, but several steps during which they observe whether they have moved more closely towards achieving their goal. By doing so, SON functions may induce a temporal performance decrease in the network. For instance, in LTE the CCO function monitors the impact of its last deployed antenna tilt or transmission power change, and corrects that if required [1], i.e., it may try out different CM settings.

Verification mechanisms, however, observe verification areas for a *fixed* time period, also called an *observation window*. As shown in Figure 1, the window is split into several time slots during which PM data is observed and the performance of the areas is assessed. To detect anomalies, though, the length of that time window has to be properly set. If we select a too short one, almost any transient performance decrease

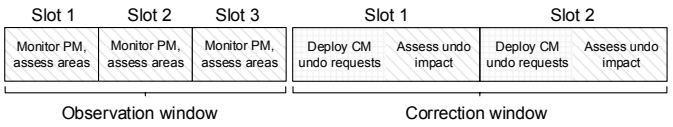


Figure 1. Observation and correction window

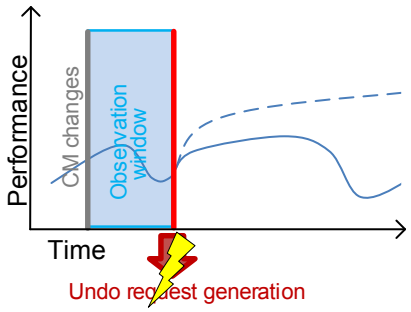


Figure 2. Inability to see the potential of reaching a better network performance (dashed lines)

results in the generation of one or more unnecessary undo requests. In other words, we may fail to find a better network configuration even when such exists. An example is given in Figure 2. For the selected observation window, the verification mechanism forces the performance of the cells to be stuck at the local optimum by starting generating undo requests, instead of searching for a better solution by allowing the remaining changes to be applied.

Of course, we may increase the length of the observation window, however, the question that arises here is for how long. On the one hand, the longer the observation window, the more difficult it becomes to find out which changes have actually caused a degradation. On the other hand, at some point in time the verification process has to enter the second time window, called the *correction window* (Figure 1). It is used for the deployment of undo requests and assessing their impact on the network performance. Simply shifting it is not always possible. For instance, in a highly populated area we might have the desire to restore the network performance as fast as possible.

B. Impact of undo requests on the verification process

In general, having several undo requests waiting to be deployed delays the verification process. It is mainly caused by verification collisions [2] or, more precisely, the time it takes to resolve them.

A collision occurs when at least two verification areas share common anomalous cells, i.e., the corresponding two undo requests impact two overlapping sets of cells. An example for such an event is given in Figure 3. The neighbors of cell 1 as well as cell 3 are 2 and 4, and the neighbor of cell 5 is cell 4. For simplicity reasons, let us assume that a single CM parameter has been changed within cells 1, 3, and 5. If we compute the verification area by taking the reconfigured cell and its direct neighbors, and cells 2 and 4 degrade, we will have three overlapping verification areas. As a result, we have an uncertainty which changes to accept and which to rollback.

To eliminate this uncertainty, the verification process cre-

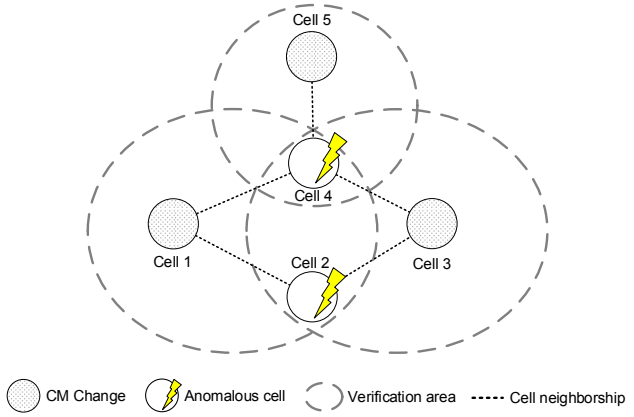


Figure 3. Verification collision problem

ates a deployment plan which assigns each undo request to a correction window time slot. The plan itself must have the properties of being (1) collision-free and (2) degradation-aware. The first property implies that undo requests assigned to the same correction window slot are not in collision with each other. The second property ensures that the deployment order maximizes the probability of returning the network performance to the expected state as quickly as possible. That is, undo requests for CM changes that are most likely causing a degradation are allocated to the first time slot.

Nevertheless, it might not always be possible to guarantee that the plan is collision-free due to the lack of available correction window time slots. In such a case, the verification process has to group undo requests while minimizing the probability of rolling back changes that did not harm the network performance [2].

IV. Concept

A. Verification state of a cell

Each cell is supplied with a state value, also referred to as a Cell Verification State Indicator (CVSI). On the one hand, a cell reporting a high CVSI is more likely to accept intermediate optimization steps, which may lead to temporarily reduced performance. On the other hand, a cell reporting a low CVSI is seen as ready for verification. That is, it is unlikely to see further optimization steps that would lead to a better performance.

To compute a CVSI, we model the behavior of each cell as an anomaly vector $\mathbf{a} = (a_1, a_2, \ldots, a_n)$ that is element of $\mathbb{R}^n$. The value of n equals $|K|$, where K is the set of KPIs that allow us to determine the cell performance status. Each element $a_k \in \mathbb{R}$, where $k \in [1; n]$, represents the deviation of a KPI from the expected value, also known as a *KPI anomaly level*. To calculate it, we define a training phase during which we collect samples $p_1 \ldots p_t$ for each given KPI, where t marks a training period. During this phase the network has to show an expected behavior. Then, we measure the current value of the KPI, denoted as p_{t+1}. The collected data, i.e., $p_1 \ldots p_t, p_{t+1}$, is standardized by computing the *z-score* of each data point. The KPI anomaly level is the z-score of p_{t+1}. The z-score itself represents the distance between the given point and the sample mean in units of the standard deviation. It is negative when the raw score is below the mean, and positive when above.

To give an example suppose that a cell has reported a Handover Success Rate (HOSR) of 99.9 %, 99.9 %, 99.1 %, 99.7 %, 99.8 %, and 99.6 % during the training phase. Moreover, let us assume that 90.2 % is the current HOSR. After normalizing the samples, we get the following outcome: 0.44, 0.44, 0.21, 0.38, 0.41, 0.35, and -2.26. The cell HOSR anomaly level is -2.26, which is the z-score of last data point.

An important remark should be made here. In general, we can distinguish between success KPIs (e.g., HOSR) and failure KPIs (e.g., handover ping-pong rate, call drop rate). As a result, a negative z-score is an indication for a drop in performance only in the case of success KPIs. For failure KPIs we would require a positive z-score to state a performance decrease. Thus, to keep things consistent, we negate the z-score of failure KPIs.

Next, we aggregate all elements a_k of a vector $\mathbf{a}$ into one single z-score that represents the overall cell performance. We compute that value by taking the arithmetic average of all a_i, as defined in Equation 1. The outcome is denoted as χ_t, where $t \in \mathbb{N}_0$ marks the time.

$$\varphi(\mathbf{a}) = \frac{1}{n} \sum_{k=1}^{n} a_k \tag{1}$$

Finally, we compute the CVSI τ by applying exponential smoothing, as follows:

$$\tau_0 = \chi_0 \tag{2}$$

$$\tau_t = \alpha \chi_t + (1 - \alpha)\tau_{t-1}, \; t > 0 \tag{3}$$

Here, $\alpha \in [0; 1]$ is the smoothing factor, and τ_t the CVSI measured at time t. The latter one is a simple weighted average of the current observation χ_t and the previous smoothed CVSI τ_{t-1}. Further, we call α the *state update factor* that determines the impact of the current χ_t on the overall τ_t. Its selection depends on χ_t, more precisely, the range χ_t falls in. In general, we distinguish between the following three intervals:

- $|\chi_t| \in [0; 1)$, i.e., the cell is up to one standard deviation away from the expected performance.
- $|\chi_t| \in [1; 2)$, i.e., the cell is between one and two standard deviations away from the expected performance.
- $|\chi_t| \in [2; \infty)$, i.e., the cell is more than two standard deviations away from the expected performance.

Typically, the more unexpected and unusual the behavior of a cell is, the higher the impact on the resulting CVSI has to be.

B. Dynamically adapting the observation window

Since a verification mechanism assesses verification areas, we need to define as a next step when an area is ready to enter the verification process. For this purpose, let us denote the set of all cells as Σ and the set of all verification areas as V. Furthermore, let us define an extraction function f_e that returns the cells of a verification area, i.e., $f_e \colon V \to \mathcal{P}(\Sigma) \setminus \{\varnothing\}$. An area $v \in V$ is considered at time t as ready for verification if and only if $\frac{1}{|f_e(v)|} \sum_{i=1}^{|f_e(v)|} \tau_t(\sigma)_i < T$, where $\tau_t(\sigma)$ is the CVSI of a cell $\sigma \in f_e(v)$ and T a *verification threshold*.

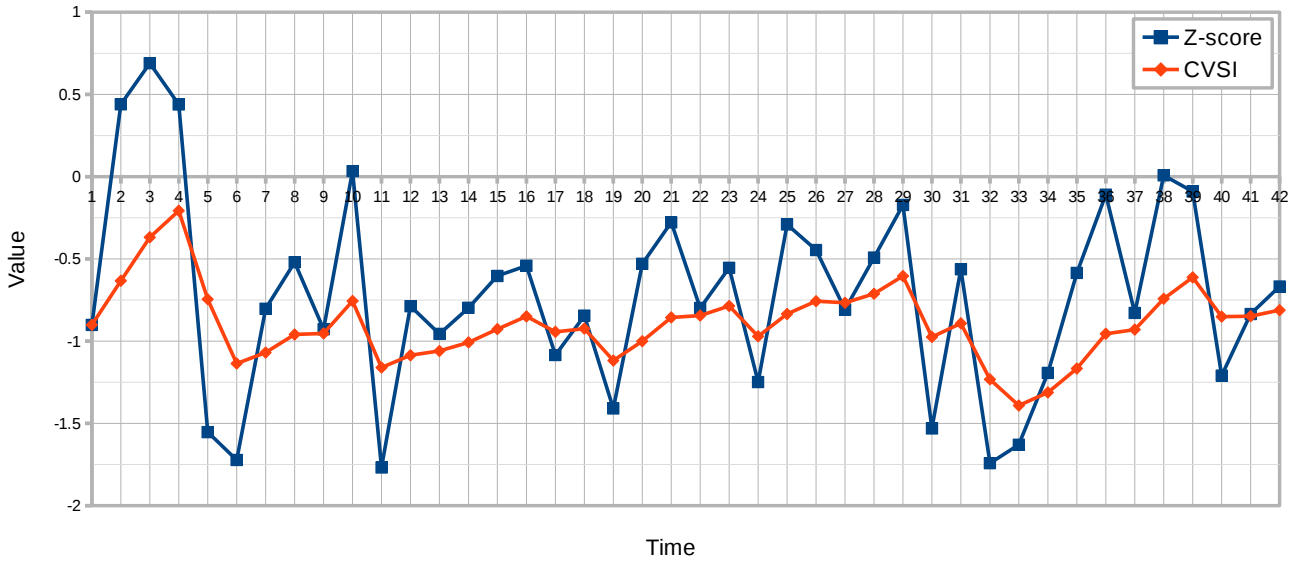


Figure 4. Concept example

C. Example

In Figure 4 we give an example of our concept. For simplicity reasons, let us assume that the verification area consists of one cell and that the threshold T equals -1.25. The example starts with an increase of the cell's performance, i.e., an increase of the z-score, which also causes the CVSI to rise. This behavior can be monitored between time ticks 1 and 4. Then, we have a sudden decrease of the z-score which also affects the cell's CVSI. As we can see, the very low z-score at time tick 5 and 6 leads to a rapid CVSI fall, however, it does not cross the threshold. Hence, the SON functions have the opportunity to stabilize the cell. Their impact can be observed at time tick 10, where we have a z-score of 0.033.

Until time tick 31, we can monitor similar z-score and CVSI changes. However, the CVSI does not fall below the threshold which allows the functions fulfill their optimization goal. They always manage to produce a z-score where it is supposed to be, namely near zero. At time tick 32 and 33, though, the performance drop leads the CVSI to fall below T. As a result, the observation is interrupted and the area is assessed by the verification process. In this particular example, undo requests have been generated which positively affect the z-score (time tick 34-36).

However, if we had not used the CVSI and had taken raw z-score instead, we would have processed the area immediately after the first degradation at time tick 5 which could have generated unnecessary undo requests. In other words, we would have prevented the functions from achieving their optimization objective.

V. Evaluation

A. Simulation environment

The simulation environment, called the SON Simulation System (S3) [5], consists of an LTE radio network simulator and a SON function coordinator [10]. In addition, the Mobility Robustness Optimization (MRO), Remote Electrical Tilt (RET), and Transmission Power (TXP) functions, as specified in [1], are deployed. A verification mechanism implementing the concept introduced in Section IV is active as well.

The LTE simulator, as part of the SON simulator/emulator suite [11], performs continuous simulation by tracking the changes in the network over time. The simulation time is divided into time slices, called simulation rounds, each corresponding to 100 minutes in real time. At the beginning of a round, the simulator configures the network as defined by the CM parameter setup. At the end of a round, cell PM data is exported which is fed into the deployed SON functions as well as the verification mechanism. During a simulation round, 1500 uniformly distributed users follow a random walk mobility model (speed of 6 km/h) and actively use the mobile network over an area of 50 km^2. The network covering the area is an LTE macro cell network consisting of 32 cells. The carrier frequency is set to 2000 MHz whereas the channel bandwidth is set to 20 MHz. Furthermore, a handover occurs immediately when a UE crosses the hysteresis threshold of 2.0 dB. A radio link failure is detected based on a signal-to-interference-plus-noise ratio comparison to a threshold of -6.0 dB.

B. Parameter selection

The training period t for computing a KPI anomaly level lasts a simulation round. The training data is collected during a separate test run, lasting 70 rounds, during which optimal network settings are used. In addition, the KPIs we consider for computing the anomaly vector are the HOSR, the Channel Quality Indicator (CQI), and the handover ping-pong rate, as specified in [1]. It should be noted that the CQI is calculated as the weighted harmonic mean of the CQI channel efficiency. The efficiency values are listed in [12].

Furthermore, a verification area consists of the target cell and its direct neighbors. An area is considered as potentially degraded when the CVSI of one cell falls in the range $(-\infty; -2.0]$. The verification threshold T is set to -1.75 whereas the state update factor α depends on χ_t as follows: $\alpha = 0.2$ if $|\chi_t| \in [0; 1)$, $\alpha = 0.4$ if $|\chi_t| \in [1; 2]$, and $\alpha = 0.8$ if $|\chi_t| \in [2; \infty)$.

C. Compared strategies

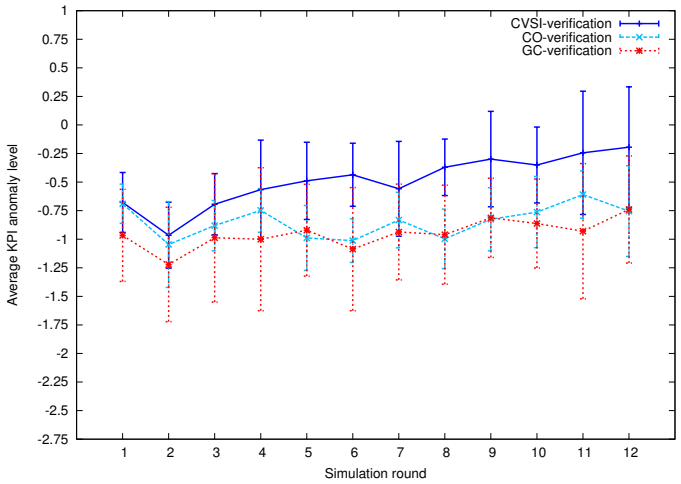
The idea of the simulation case study is not only the detection of a degradation and finding out which CM change was the trigger for this to happen, but mainly to observe the impact of the CVSI concept on the overall network performance. Furthermore, we are interested in the comparison between the introduced approach and other verification approaches that do not utilize the CVSI concept. In particular, we take into consideration the Graph Coloring (GC)-based method, outlined in [5], and the Constraint Optimization (CO)-based approach described in [2]. The first one uses minimum graph coloring to identify the sets of cells whose configuration can be safely rolled back. Two undo requests are called safe if and only if the corresponding verification areas are not participating in a verification collision. Note that his strategy takes the frequency of the used colors as the main criteria while selecting the execution order, i.e., the requests having the most frequently used color are scheduled at first place. The second approach makes use of constraint optimization techniques to identify which requests can be merged together in case we have an insufficient number of correction window slots. To do so, it utilizes constraint softening based on the KPI anomaly levels of the impacted cells.

D. Results

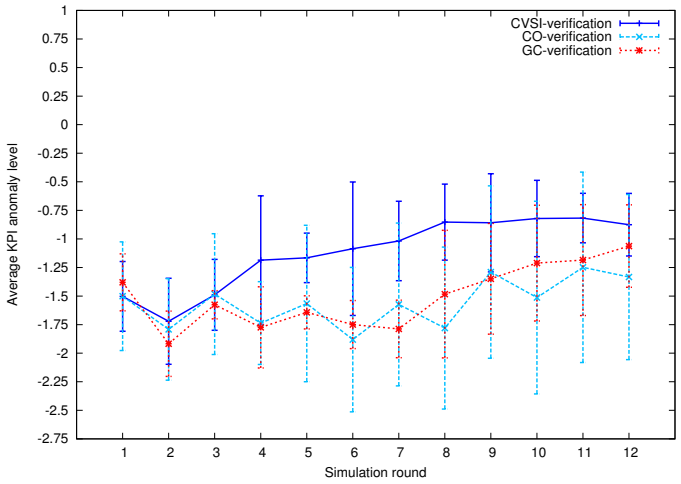
To perform the evaluation, we have selected nine cells and changed their coverage by using obsolete data. In particular, four cells have a tilt degree of 0.0, two cells a degree of 1.0, one cell a degree of 3.0, and two cells a degree of -4.0. In addition, their transmission power is set to the maximum of 46 dBm. As stated in [1], in reality we might get supplied with obsolete data since the environment may change from the assumptions made when the network was initially planned and set up. Typical causes are the construction or demolition of buildings, insertion and deletion of base stations, and season changes. As a result, the coverage can be reduced compared to what we could achieve with optimal settings.

Those obsolete settings are applied before starting a test run. In total, we have seven test runs, during each of which we let the SON functions try reaching their optimization goal. In particular, the coverage functions are changing the physical cell borders at first, which is later followed by an MRO adjustment of the handover parameters. Those functions, however, can be interrupted by the verification mechanism if it decides to generate an undo request, i.e., it has the highest priority. Furthermore, a test run lasts 12 rounds and the verification mechanism is allowed to observe the performance impact of the changes after the third simulation round.

During the test runs, we managed to spot two cells (having the ID 2 and 6) on which the optimization functions put their highest focus on. The resulting verification areas included five and six cells, respectively. Furthermore, the areas shared two cells, thus, creating the potential for verification collisions. In Figure 5 we have plotted the average KPI anomaly level of the two areas, impacted by the changes. Our observations show that the SON functions tried out different CM changes in order to achieve their goal. In particular, the RET function started to adjust the antenna tilt, which was followed by a Cell Individual Offset (CIO) adjustment by MRO. Those changes,



(a) Verification area of target cell 2



(b) Verification area of target cell 6

Figure 5. Average KPI anomaly level

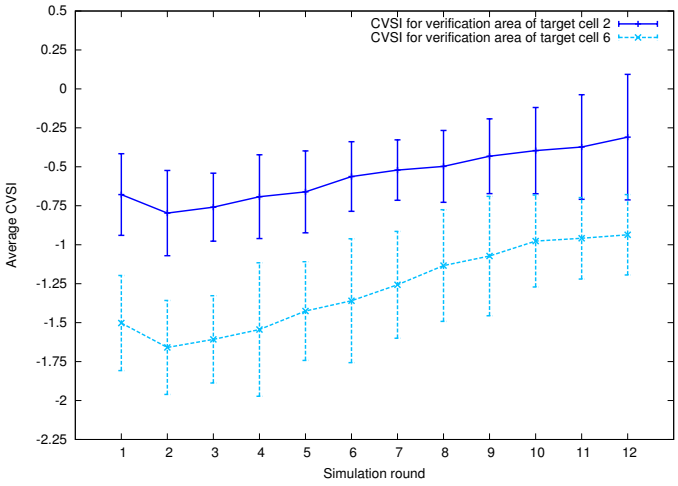


Figure 6. Average CVSI of the verification areas

though, induced temporal performance drops which resulted in the generation of undo requests during the CO and GC evaluation. As a result, the average KPI anomaly level of both areas fluctuates until round 12 and never reaches zero,

as presented in Figure 5(a) and 5(b).

Interestingly, the executed CM changes have the lowest impact on the CVSI approach. As we can see, it was able to provide the highest anomaly level result, i.e., it gave us the best network performance among all used verification strategies. The reason why the CVSI based method manages to perform like that is presented in Figure 6. After simulation round 4, the CVSIs of the areas simply do not fall below the threshold T. As a result, they are not further assessed by the verification mechanism and are let to be optimized by the SON functions.

VI. Evolution of SON Verification

The SON concept as it is today will be much further developed, especially in future standards like the fifth generation of mobile communications (5G). In 5G, advanced SON techniques will not only apply to physical NEs, but will enable operators to, for instance, balance load in a multi-radio-access technology environment, and support traffic steering as well as dynamic spectrum allocation [13]. Moreover, we will have a wider variety of use cases [14], e.g., broadband access in dense areas, higher user mobility, and extreme real-time communications, which may result in having a much higher number of deployed SON functions in the network. As a consequence, the assessment of the combined performance impact of changes triggered by SON functions will become more challenging. Hence, the question arises of how much a verification mechanism needs to change and which parts will remain as they are.

In general, the structure of a verification mechanism will resemble the one we currently have. As outlined in Figure 7, it realized as a central entity that collects CM and PM data from the network. Internally, it forwards the gathered information to a verification area resolver as well as an anomaly level assessor. Those two components implement the first two steps of the verification process as described before. In addition, the concept introduced in Section IV is implemented by the anomaly level assessor, i.e., we will still need a functionality that prevents unnecessary verification areas to be assessed by the verificaiton process. Should, however, we find an area with a low CVSI, the undo request assembler is triggered. This component implements the last step of the verification process, i.e., creating a deployment plan having the properties of being collision-free and degradation-aware, as discussed in Section II.

Exactly, this last step is where we see the highest potential for a change in a 5G deployment. In such an environment base stations will start having a range similarly to commonly used Wi-Fi routers [15]. Consequently, verification areas will get larger and include more entities requiring verification, which creates a potential for more area overlaps and collisions. Therefore, the scope of the verification process will no longer solely include verification areas, but form so-called *undo collision domains*. A collision domain is a section of the mobile network where verification collisions have taken place, i.e., areas being in a verification collision are part of such a domain. Areas that are not participating in collisions form their own collision domain.

Furthermore, a verification mechanism will extend its capabilities, besides assessing verification areas and generating

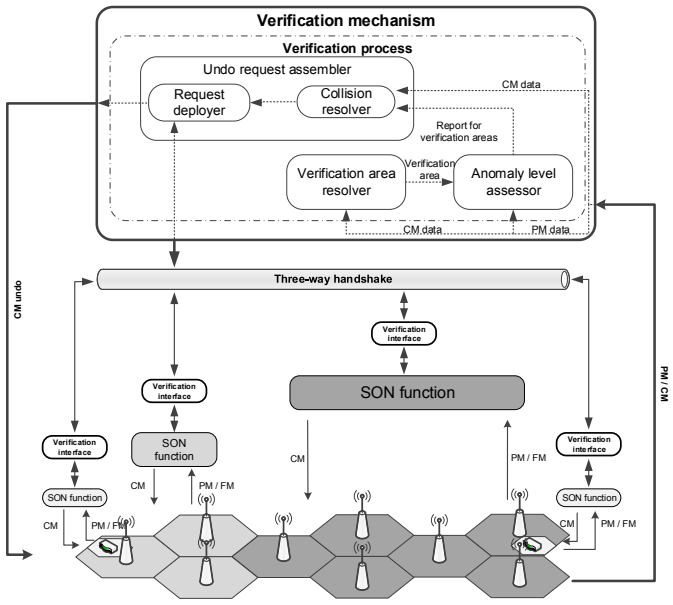


Figure 7. Architecture overview

undo requests. In our opinion, it needs a much more active interaction with the functions. In this paper, we presented a way of timing a verification mechanism, in particular, when to complete the observation period. Further, several strategies have been introduced that target anomaly detection [3], [9], [16], [17] as well as methods that deal with the deployment of undo requests [2], [5]. However, what we certainly do not do is to involve the SON functions into the undo decision making process. Therefore, we see the interaction with the deployed functions as an important part in the workflow of a 5G verification strategy. In order to meet the ultra-reliability requirements listed in [14], it would need to reliably negotiate all required verification parameters with all involved functions. A possible way of doing that is presented in Figure 8, where we have depicted communication flow as a *three-way handshake*.

After detecting areas with low CVSIs, the verification mechanism contacts the SON functions of each collision domain over a *verification interface* by sending an *initiation message*. It serves the purpose to notify the functions within the verification areas about the spatial scope of the verification process. There are, however, two cases between which we would need to differentiate. The first one is when the verification mechanism notifies functions that are operating in a collision domain comprised of more than one verification area. In such a case, it will need to include not only the list of impacted cells and marked CM parameters for an undo, but also a parameter indicating the time required for resolving the collisions. In the second case, i.e., a collision domain consisting of a single verification area, it will send the same message, but without the duration parameter.

Upon reception of such a message, each function will respond with either a *verification-accept* or with a *verification-decline* message. By sending an accept message, the replying SON function is declaring its decision not to interfere as the CM undo deployment is taking place within the collision domain. However, a function may also decline the planned

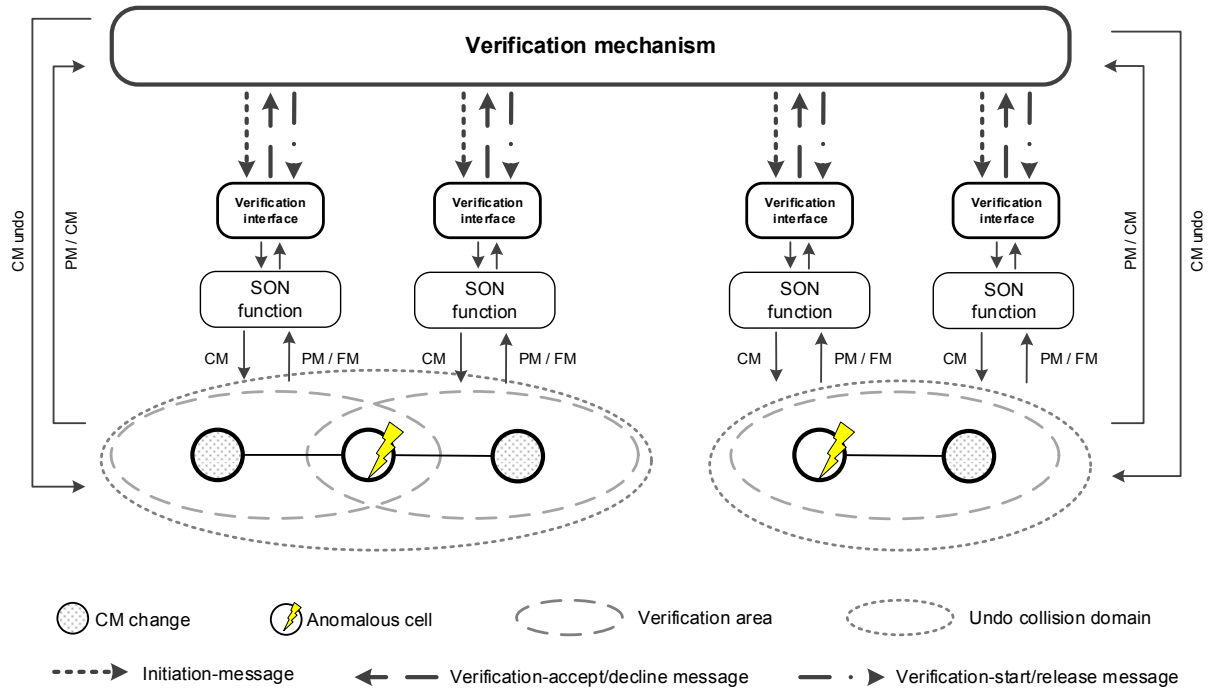


Figure 8. Undo collision domains and three-way handshake

undo process if it needs some additional steps to reach its target. Thereby, functions will be obligated to inform the verification mechanism about the CM parameters they are optimizing as well as the estimated time for achieving their objective.

After receiving all replies, the verification mechanism would send either a *verification-start* or a *verification-release* message to the functions for each collision domain. In case all functions have reported that they will freeze their operation for the suggested time, a start message is will be sent, which will be later followed by the first set of undo requests. However, if a function has reported that it is on the way of achieving its objective, the verification function will be obligated to notify the remaining functions within the domain about the expected duration of this process.

VII. Related work

In [18], a CM scoring method is proposed for the SON verification use case. It introduces a mechanism that assesses the CM changes made to cells over a certain time period. It categorizes them to zones based on their impact on the performance. For instance, if a CM change causes an abnormal cell behavior, it is assigned to a so-called red zone. Depending to which zone it has been assigned to, it is positively or negatively rated. However, the scoring itself is a simple procedure that either adds or subtracts a zone-dependent value from the overall one. Thus, changes continuously getting positive scores need many negative scores in order to be considered for a rollback. In other words, some anomalies may retain for a long time period, simply because they may not be immediately visible in the overall score.

The concept of pre-action SON coordination [6] can be seen as an alternative strategy to SON verification. Today, it is implemented as a pessimistic strategy that makes use of rules required for the detection and resolution of *known* conflicts between active SON function instances. However, it prevents conflicting functions from getting active, rather than observing the network performance after the deployment of CM changes. A typical example of such conflicts is the operation on shared CM parameters within the same physical area. Another example is when the activity of one function affects the input measurements of another one.

In [3], an anomaly detection and diagnosis framework has been introduced. It attempts to verify the effect of CM changes by monitoring the state of the network. The framework operates in two phases: (1) it detects anomalies by using topic modeling, and (2) performs diagnosis for any detected anomaly by using Markov Logic Networks (MLNs). In the latter case it makes use of probabilistic rules to differentiate between different causes. The proposed solution does not address the presented problems.

VIII. Conclusion

In today's Self-Organizing Networks (SONs), the verification of Configuration Management (CM) changes is important step towards achieving an optimal network performance. Approaches that aim to verify such changes usually operate in three phases. At first, the network is divided into verification areas, then, those areas are assessed by an anomaly detection algorithm, and finally CM undo requests are generated for the abnormally performing ones. Those requests return the poorly performing areas to a previous stable configuration.

However, if the observation window length of the verification mechanism is not correctly set, for example, by observing

the network for a too short period, it may prevent the deployed optimization functions from reaching their optimization goal. A too short observation window leads to a cautious verification strategy, which may produce a high number of false positive undo requests. Those requests can also delay the verification process since they might be part of a verification collision. Collisions occur when the corresponding verification areas share anomalous cells, in which it is difficult to determine, which change caused the degradation and should be rolled back. Resolving such collisions is a time-consuming process which may violate the requirement for availability, reliability and efficiency in a current as well as in a future network generation.

To overcome this issue, we present an approach that allows us to dynamically select the observation period. We supply each cell with a Cell Verification State Indicator (CVSI) that defines whether it is ready for verification. The CVSI a cell has is computed by using exponential smoothing as well as how much its current performance deviates from the expectations. Furthermore, the factor based on which the CVSI is updated, depends on how far the performance of the cell is from the expected value.

The evaluation of the concept has been carried out in a simulated environment. We compare our approach with two other verification strategies: one that is based on minimum graph coloring and another that utilizes constraint softening. The results show that the CVSI method is able to successfully detect cells that a currently being optimized and prevent them from being further processed by the verification process. This effectively led to a better network performance.

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] T. Tsvetkov, C. Frenzel, H. Sanneck, and G. Carle, "A Constraint Optimization-Based Resolution of Verification Collisions in Self-Organizing Networks," in *IEEE Global Communications Conference (GlobeCom 2015)*, San Diego, CA, USA, Dec. 2015.

[3] G. Ciocarlie, C. Connolly, C.-C. Cheng, U. Lindqvist, S. Nováczki *et al.*, "Anomaly Detection and Diagnosis for Automatic Radio Network Verification," in *6th International Conference on Mobile Networks and Management (MONAMI 2014)*, Würzburg, Germany, Sep. 2014.

[4] B. Gajic, S. Nováczki, and S. Mwanje, "An Improved Anomaly Detection in Mobile Networks by Using Incremental Time-aware Clustering," in *IFIP/IEEE Workshop on Cognitive Network and Service Management (CogMan 2015)*, Ottawa, Canada, May 2015.

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

[6] T. Bandh, "Coordination of autonomic function execution in Self-Organizing Networks," PhD Thesis, Technische Universität München, Apr. 2013.

[7] Ericsson, "Transparent Network-Performance Verification For LTE Rollouts," White Paper, 284 23-3179 Uen, Sep. 2012.

[8] "The Oxford Dictionary of English," Revised Edition, Oxford University Press, 2005.

[9] S. Nováczki, "An Improved Anomaly Detection and Diagnosis Framework for Mobile Network Operators," in *9th International Conference on Design of Reliable Communication Networks (DRCN 2013)*, Mar. 2013.

[10] R. Romeikat, H. Sanneck, and T. Bandh, "Efficient , Dynamic Coordination of Request Batches in C-SON Systems," in *IEEE Veh. Technol. Conf. (VTC Spring 2013)*, Dresden, Germany, Jun. 2013.

[11] NSN, "Self-Organizing Network (SON): Introducing the Nokia Siemens Networks SON Suite - an efficient, future-proof platform for SON," White Paper, Oct. 2009.

[12] 3GPP, "Evolved Universal Terrestrial Radio Access (E-UTRA); Physical layer procedures," 3rd Generation Partnership Project (3GPP), Technical Specification 36.213 V12.1.0, Mar. 2014.

[13] Ericsson, "5G: what is it?" White paper, Oct. 2014.

[14] Next Generation Mobile Networks Alliance, "A Deliverable by the NGMN Alliance: NGMN 5G White Paper," Feb. 2015, final deliverable, version 1.0.

[15] S. Chen and J. Zhao, "The Requirements, Challenges, and Technologies for 5G of Terrestrial Mobile Telecommunication," *Communications Magazine*, Jan. 2015.

[16] G. Ciocarlie, U. Lindqvist, K. Nitz, S. Nováczki, and H. Sanneck, "On the Feasibility of Deploying Cell Anomaly Detection in Operational Cellular Networks," in *IEEE/IFIP Network Operations and Management Symposium (NOMS 2014)*, Krakow, Poland, May 2014.

[17] P. Kumpulainen, M. Särkioja, M. Kylväjä, and K. Hätönen, "Finding 3G Mobile Network Cells with Similar Radio Interface Quality Problems," in *Engineering Applications of Neural Networks, L. Iliadis and C. Jayne, Eds. Springer Berlin Heidelberg*, 2011, pp. 392–401.

[18] S. Nováczki, T. Tsvetkov, H. Sanneck, and S. Mwanje, "A Scoring Method for the Verification of Configuration Changes in Self-Organizing Networks," in *7th International Conference on Mobile Networks and Management (MONAMI 2015)*, Santander, Spain, Sep. 2015.