Anomaly Detection and Diagnosis for Automatic Radio Network Verification

Gabriela F. Ciocarlie[1], Christopher Connolly[1], Chih-Chieh Cheng[1], Ulf Lindqvist[1], Szabolcs Nováczki[2], Henning Sanneck[2], and Muhammad Naseer-ul-Islam[2]

[1] SRI International
{gabriela.ciocarlie,christopher.connolly,chih-chieh.cheng,ulf.lindqvist}@sri.com
[2] Nokia
{szabolcs.novaczki,henning.sanneck,muhammad.naseer-ul-islam}@nsn.com

Abstract. The concept known as Self-Organizing Networks (SON) has been developed for modern radio networks that deliver mobile broadband capabilities. In such highly complex and dynamic networks, changes to the configuration management (CM) parameters for network elements could have unintended effects on network performance and stability. To minimize unintended effects, the coordination of configuration changes before they are carried out and the verification of their effects in a timely manner are crucial. This paper focuses on the verification problem, proposing a novel framework that uses anomaly detection and diagnosis techniques that operate within a specified spatial scope. The aim is to detect any anomaly, which may indicate actual degradations due to any external or system-internal condition and also to characterize the state of the network and thereby determine whether the CM changes negatively impacted the network state. The results, generated using real cellular network data, suggest that the proposed verification framework automatically classifies the state of the network in the presence of CM changes, indicating the root cause for anomalous conditions.

Key words: network automation, self-organized networks (SON), SON verification, anomaly detection, diagnosis

1 Introduction

Modern radio networks for mobile broadband (voice and data) are complex and dynamic, not only in terms of behavior and mobility of users and their devices, but also in terms of the many elements that make up the network infrastructure. Network degradations that cause users to experience reduced or lost service could have serious short- and long-term impact on the operator's business, and must therefore quickly be resolved as part of network management. Effective management of complex and dynamic networks requires some form of automated detection of problems such as performance degradation or network instability, and also automation support for timely and effective diagnosis and remediation of the detected problems. There are many factors that could cause degradations

in radio network performance: hardware faults, software faults, environmental conditions (like weather and changes in the infrastructure), but degradations can also stem from unintended effects of network configuration changes.

The need for adaptive, self-organizing, heterogeneous networks becomes pressing given the explosion of mobile data traffic with increased use of smartphones, tablets, and netbooks for day-to-day tasks. Expectations for mobile networks have grown along with their popularity, and include ease of use, high speed, and responsiveness. Heterogeneous Networks (HetNet) can offer these capabilities, providing virtually "unlimited" capacity and "ubiquitous" coverage. However, a high level of distribution in the network infrastructure introduces higher complexity, which requires additional mechanisms such as Self-Organizing Networks (SON) concepts.

In order to prevent network-level degradation, actions that change network-element configurations must either be coordinated *a priori* or their effects must be verified (or both approaches could be used complementarily). Verification is a hard problem for several reasons: actions can be the result of automated or human decisions; networks for mobile broadband are very complex; some parameters such as user behavior cannot be controlled; and no established simple indicator of "system health" exists.

1.1 SON Verification

Before a system reaches the verification process, it may undergo a pre-action SON coordination process. Based on rules provided by human experts or automatically determined, SON coordination aims to reduce the risk of processing actions that lead to conflicts and degraded states [14]. Performance Management (PM) data is continuously collected from network cells in the form of Key Performance Indicators (KPIs), which are a set of selected indicators used for measuring network performance and trends: call-drop statistics, channel-quality-indicator statistics, handover statistics, throughput, etc. Based on the KPIs collected from all cells in the network or domain, the operation of a SON-enabled system in a certain network domain should be verified to ensure that the new actions improved the network performance rather than negatively impacting it.

The SON verification process must occur as quickly as possible in order to correlate detection results based on PM history with the history of Configuration Management (CM) changes. If verification indicates that the network entered a normal state, then the SON coordination process leverages knowledge of acceptable actions and combinations of actions given the configuration history and the system's current state. In addition, it is desirable to determine and learn which actions and combinations of actions are unacceptable, given the same history and state, along with enabling action reversal where applicable.

1.2 Contributions

This paper proposes a novel SON verification framework using anomaly detection and diagnosis techniques that operate within a spatial scope larger than

an individual cell (e.g., a small group of cells, a geographical region like a town section, an existing administrative network domain, etc.). The aim is to detect any anomaly which may point to degradations and eventually faults caused by an external or system-internal condition or event. CM changes (which reflect actions, e.g., by human operators or SON functions or optimization tools), and KPIs (which reflect the impact of the SON actions on the considered part of the network) are analyzed together to characterize the state of the network and determine whether it was negatively impacted by CM changes. Main contributions include:

- automatically identifying different network states using the KPI measurements from all the cells/network in scope,
- using intrinsic knowledge of the system to automatically classify the states as either normal or abnormal,
- determining the most likely explanation for the performance degradation.

2 Anomaly Detection and Diagnosis for SON Verification

Our approach is comprised of two main steps: 1) detecting anomalies for a group of entities (i.e., cells) using topic modeling combined with 2) diagnosing any anomaly using Markov Logic Networks (MLNs), which rely on probabilistic rules to discern between different causes.

Topic modeling [16] is a type of statistical model that allows efficient topic training and inference of mixing proportion of topics. It was initially used for discovering topics in documents, where a topic is defined as a probability distribution over words. The benefit of topic modeling is that each topic is individually interpretable and characterizes a coherent set of correlated terms.

Markov Logic Networks (MLN) [13] are an approach for probabilistic reasoning using first-order predicate logic. As with traditional first-order logic, hypotheses can be described in terms of supporting evidence and conclusions about network properties. In contrast to first-order logic, MLNs permit rule weighting and an explicit representation of the probabilities associated with logical statements. As a result, one can think of MLNs as providing the infrastructure of a probabilistic knowledge base, which can operate using noisy or incomplete data and can incorporate diverse data sources into a common framework for analysis.

2.1 Overall Framework

Topic modeling is applied to the training KPI data from all the cells in scope, leading to the construction of topic modeling clusters [1] (indicators of the state of the network), which are further classified by semantic interpretation as either normal or abnormal. Using the labeled clusters, the KPI data under test (i.e.,

[1] Given that we apply topic modeling to KPI data, for clarity, we will refer to topics as clusters.

subject to detection) leads to the mixture of weights for the different clusters indicating the overall state of the network. For real deployment, testing data can span any time or geographic scope (as a subset of the training geographic scope), and may or may not overlap with the training data. This component is well suited for this application domain, as we do not have any *a priori* notion of what the different network states could be or how many types of states there could be. Here, we generalize the concept of topic modeling to identify the states (topics) of a system (i.e., the radio network). Furthermore, using semantic information, states can be interpreted as either normal or abnormal, enabling the detection of degradation in system's state (see Section 2.3). Moreover, incremental topic modeling can be used to capture new states over time [8].

In case of abnormal behavior, the diagnosis component is triggered. The MLN inference is achieved by using CM change history information or any other external information in form of an event sequence, along with the MLN rules and their associated weights. The MLN rules are specific to this application domain and are typically generated based on human expert knowledge. Rule weights can be estimated or learned during operation, as new cases arise.

Figure 1 presents the detailed verification approach:

- Initially, for a given period of time, the KPI measurements of the group of cells/network in scope are selected as the training dataset (D1) for generating the topic modeling.
- The topic modeling clustering (M1) is applied to the training dataset (D1).
- The result of (M1) is a set of clusters representing the different states in which the network can exist (D2). Each cluster has an associated weight corresponding to the percentage of the network in the state represented by that cluster.
- Given the set of clusters (D2), the semantics of the KPIs are used to further interpret the semantics of each cluster (M2).
- The result of (M2) is a set of labeled clusters that indicate if the network state is either normal or abnormal (D3).
- The labeled clusters (D3) and the KPI measurements for test for the group of cells in scope are used in a testing phase against the clusters (M3) to generate the weight mixture indicating how normal or abnormal the network is.
- The result of (M3) is the weight mixture (D4) indicating the current state of the network.
- The diagnosis component is triggered only if the cells in scope are abnormal. Principal Components Analysis (PCA) (M4) is applied to the training dataset (D1) to generate similar groups of cells. The result of (M4) contains groups of cells (D5) that behave similarly; MLN inference is applied primarily on these groups. Cell grouping is used to reduce MLN complexity.
- The groups of cells along with the CM change information (D5), external events (e.g., weather event feeds) and MLN rules (generated either manually based on human expert knowledge or automatically from other sources) are used to generate the diagnosis information based on the MLN inference (M5).
- The result of (M5) is the diagnosis information for the abnormal cells within the scope (D6).

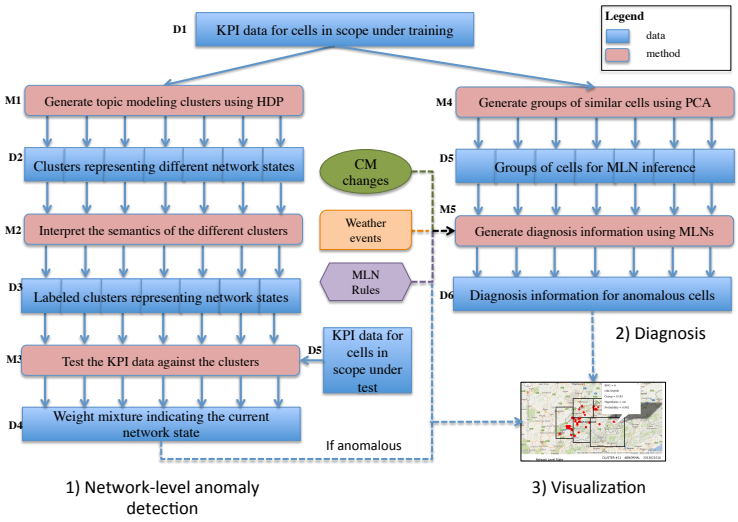

Fig. 1. Overall approach of the SON verification method applied to the group of cells in scope. Data is depicted in blue and methods in pink. The dashed lines indicated that an event is triggered in the presence of new evidence/data.

2.2 Anomaly Detection for SON Verification

The family of algorithms known as topic modeling [16] is well suited to the SON verification domain because topic modeling can discover and infer themes (topics) that constitute a hidden structure in a data set. In the set of KPI data from multiple cells in a network, we do not have any *a priori* notion of what the different network states could be or how many types of states there could be. From the KPI data, topic models learn clusters of network states, and output the codebook of the clusters and the mixing proportion of the clusters at each time stamp. A codebook records a profile for each cluster (the average of the cluster, called a *centroid*), and the clustering of an unknown query usually depends on its similarity to the cluster's profile.

Each cluster represents a state, characterized by its centroid. The mixing proportion of clusters at each timestamp can be seen as a type of trigger for the overall state of cells in scope. For our implementation, we used the Hierarchical Dirichlet Process (HDP) algorithm [18]. As the name suggests, HDP is a hierarchical graphical model which extends the Latent Dirichlet Allocation (LDA) [4], the most common topic modeling approach. For our context, the LDA model represents a collection of M timestamps of KPI data from N cells. At each time stamp, the KPI data is a mixture of K clusters, and the weight of each cluster is represented by θ. For each cell n at time m, the KPI feature w can be classified into one of the K clusters, determined by the hidden variable z.

Therefore, the goal of LDA models is to infer $p(z|w) = \frac{p(w,z)}{\sum_z p(w,z)'}$, which is usually intractable due to a complicated marginalization of hidden variables.

Several solutions exist for approximating the inference, including variational inference [4] and Gibbs sampling [10]. We considered the Gibbs sampling.

The inputs to LDA models are a collection of vectors containing feature values derived from KPI data of all cells. The outputs of LDA models are a codebook for K clusters, and a set of cluster mixture weights θ for each timestamp m.

By default, LDA can only be applied to a single KPI feature (i.e., it considers only one KPI feature value from cells in the network, and does clustering based on it). However, our framework needs to consider multiple KPIs as a whole to determine network status. We extend the model to accommodate multiple features by replacing the single feature w with multiple features w_i, and associate each feature to a codebook β_i. We denote this model as multi-variate LDA (m-LDA). Note that each cluster contains a histogram for every feature (KPI). The histogram represents the expected histogram of that feature value for the given scope (network or group of cells) under one cluster. The major difference between LDA and HDP is that for LDA both the number of topics at each timestamp and the number of profiles are fixed, while for HDP they are automatically determined. The inputs and outputs for HDP are the same as for LDA.

2.3 Cluster Interpretation

Using visual inspection, the topic modeling centroids can be characterized as either normal or abnormal, but an automated interpretation module is necessary. Consequently, we introduce a simple classifier for the centroids that considers the characteristics of the KPIs. This classification can be achieved only for KPIs that are not supposed to increase (e.g., drop call rate) or decrease (e.g., call success rate) within certain bounds. To characterize each KPI for a given centroid (represented as a normalized histogram with B bins, where p_i is the proportion allocated to bin i), we calculate its expected value as $E[X] = \sum_{i=1}^{B} p_i * i$.

A final score is computed as:

$$score = \begin{cases} \frac{|1-E[X]|}{B-1}, \text{if it should not increase} \\ \\ \frac{|B-E[X]|}{B-1}, \text{if it should not decrease} \end{cases} \quad (1)$$

The following qualifiers are generated for the different score values:

$$label = \begin{cases} \text{VERY GOOD, if } score < \tau_1 \\ \text{VERY BAD, if } score > 1 - \tau_1 \\ \text{GOOD, if } \tau_1 <= score < \tau_2 \\ \text{BAD, if } 1 - \tau_2 <= score < 1 - \tau_1 \\ \text{RELATIVELY BAD, otherwise} \end{cases} \quad (2)$$

where τ_1, $\tau_2 \in [0, 1]$ are the thresholds that determine the classification and are empirically determined such that the quality of the (VERY) GOOD clusters is high (i.e., $\tau_1 = 0.05$ and $\tau_2 = 0.15$). A cluster that has at least one BAD (any type) histogram is considered an abnormal centroid; otherwise it is normal.

2.4 Diagnosis for SON Verification

MLNs are well suited for diagnosis in this application domain because they can be used to compute the most likely explanation for an event given data that is noisy, incomplete or even contradictory. Probabilistic parameter values (weights) can be learned through experience and user feedback. We approach the problem of diagnosis in terms of expressing multiple hypotheses within the MLN rule set, running the inference engine, then querying for the most likely explanations given the observed conditions. We use the Probabilistic Consistency Engine (PCE) [1], a very efficient MLN solver under continuous improvement.

We apply MLNs by reasoning over groups of cells (where groups can be geographically or behaviorally defined) at different times. PCA is applied to identify groups of cells that behave similarly over all KPIs. By reasoning over groups, we can improve the efficiency of the inference process, reducing the number of entities (from cells to group of cells) on which the inference is performed. The PCE input language consists of the following elements:

- Definition of types (also called sorts)
- Enumeration of the sets corresponding to each type
- Declarations of the predicates to be used, and the types of each argument
- A set of weighted clauses comprising the probabilistic knowledge base
- Assertions (predicate forms that express information that is known to be true)

Each clause is an expression in first-order logic. MLNs search for the most likely explanation for the knowledge base in terms of the assignments of variables to predicate arguments. MLNs accumulate the probabilities that each clause is true given a particular variable assignment. One can also query the knowledge base and ask for the probability that a specific predicate is true under a specific variable assignment or ask how often a predicate is true in general.

Figure 2 presents an example of a PCE input specification, which includes three types (sorts), called $Time\_t$, $Group\_t$ and Mag_t. In particular, the $Group\_t$ sort refers to PCA-derived groups of cells. The `const` declaration defines 486 cell groups that we can reason over. We also have anomaly conditions derived from the network-level anomaly detection component described above. In the MLN excerpt here, we see two out of several hundred anomaly conditions. These two declare that anomalies were observed in groups $G39$ and $G46$ at time interval $T2$. Finally, we see three `add` statements. These are rules that link weather, anomaly, and configuration information with hypotheses about the reasons for network degradation. The final statements in the PCE input are `ask` statements that query the state of the network for the probabilities of different hypotheses.

When applying MLNs to temporal data, decomposition of the timeline into intervals or atomic units (individual timestamps or samples) is generally useful. In some cases, rules might be needed to define temporal order, especially for attempts to represent causality and delayed response to disruptive events. Time (or sample number) can be applied as an extra argument in certain predicates. In general, MLN solution times depend polynomially on the number of observations in the data, but will grow exponentially with the number of arguments to

```
sort Time_t;
sort Group_t;
sort Mag_t;
 . . .
const G1, G2, G3 … G486: Group_t;
 . . .
assert anomaly(G39,T2);
assert anomaly(G46,T2);
 . . .
add [C, T] (snowdepth(C,T,HEAVY) and anomaly(C, T))
             implies weather_event(C,T) 5.0;
add [C, T] cm_correlation(C,T) and anomaly(C,T)
             implies cm_event(C, T) 5.0;
add [C, T] anomaly(C,T) and (not weather_event(C,T))
            and (not cm_event(C,T))
             implies hw_event(C,T) 5.0;
….
ask [x,y] cm_event(x,y) 0.7;
ask [x,y] hw_event(x,y) 0.7;
ask [x,y] weather_event(x,y)  0.7;
ask [x,y] normal(x,y)  0.7;
```

Fig. 2. Example of a PCE input specification

predicates. Therefore, the argument count should be kept low for all predicates and, if necessary, the problem should be decomposed to maintain a low argument count for all predicates used by the knowledge base.

When specifying an MLN, we normally start with rules and weight estimates that represent a subject matter expert's (SME's) understanding of causes and effects that determine network behavior. Moreover, the weights associated with the MLN rules can be learned over time to provide a more accurate probabilistic model for the observed situations. Several weight learning schemes exist, most of which take the form of maximum-likelihood estimation with respect to a training set. As more relevant training data is available, the MLN weights can be modified to tune the probabilistic knowledge base to generate the best answers for a given situation. Realistically, a SME might not be able to account for all possibilities in advance. As unexpected cases arise, there will be a need to add rules during the training phase to accommodate these cases. The new rules' weights will also need to be trained. Weight adjustment can be an ongoing process, but weight stability will be the primary indicator that the MLN is working as expected.

3 Performance Evaluation

This section analyzes the performance of our framework applied to a real network dataset. The experimental corpus consisted of KPI and CM data for approximately 4,000 cells, collected from 01/2013 to 03/2013. The KPIs have different characteristics; some of them are measurements of user traffic utilization (e.g. downlink or uplink data volume or throughput), while others are measurements of call control parameters (e.g. drop-call rate and successful call-setup rate).

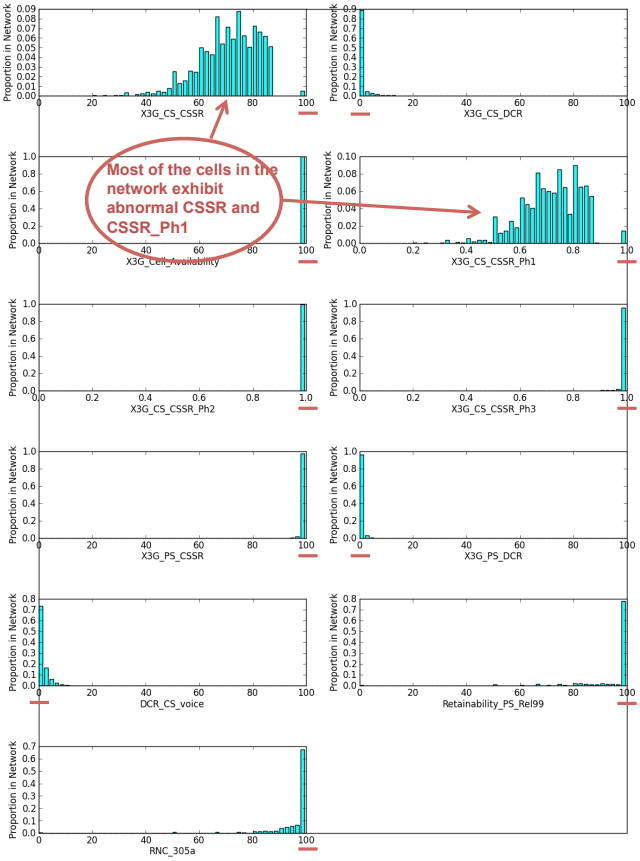

Fig. 3. Codebook of cluster #8 generated by the HDP experiment. The cluster is characterized by profiles of features, and each profile shows how the KPI value is distributed across the network within this cluster. The red underlines denote the normal value of each KPI. Under normal condition, most of the cells are supposed to be concentrated around normal values.

3.1 Cluster Analysis

We experimented with the HDP approach on all cells in the network with valid KPI data from 01/2013 to 03/2013. We trained the model on all timestamps from this time range to produce the cluster profiles, and tested on the same temporal and geographic scope to investigate the cluster semantics over the whole network. However, for real-time testing, data could span any time and geographic scope. We did not set the number of clusters, nor any constraints on how they were constructed, in advance. The final number of clusters automatically learned by our algorithm was 32. Figure 3 shows the codebook of cluster #8, with 11 KPIs. We use this particular cluster as an example because it shows one type of anomaly condition, corresponding to an abnormal network condition in mid-February. The predominant level of cluster #8 along the entire period of time

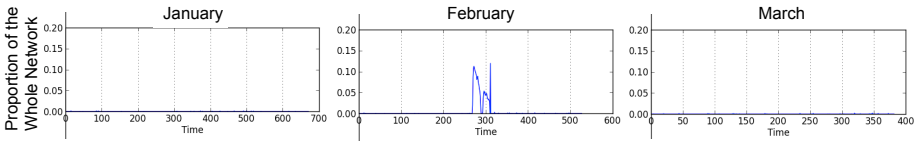

Fig. 4. The predominant level of cluster #8 as a time series.

is shown in Figure 4, which is generated by running the HDP model of Figure 3 on all cells from the whole time span.

3.2 Cluster Interpretation

Since HDP modeling is an unsupervised learning technique, the generated clusters have no associated semantics (normal or abnormal). We thus implemented a simple classifier for the centroids, taking into consideration the characteristics of the KPIs. This classification can be achieved only for KPIs that are not supposed to increase (e.g., drop call rate) or decrease (e.g., call success rate). A cluster that has at least one BAD (any type) histogram is considered an abnormal centroid; otherwise it is normal. After analyzing all the clusters (where $\tau_1 = 0.05$ and $\tau_2 = 0.15$), our tool deemed 15 centroids as normal and 17 as abnormal. Visual inspection confirmed the labels.

Given the outcome of the interpretation module, we further classified the overall state of the network for February 2013, the month with more interesting dynamics. Figure 5 presents the portion of the network in an abnormal state for February 2013. We observe that overall 5-10% of the network exhibits some abnormalities, and then for some periods of time a larger portion of the network exhibits abnormalities. The diagnosis component will further identify the causes for the observed anomalies.

3.3 Diagnosis Results

When we applied PCE to the February 2013 rule set, we asked for instances of three different hypothetical conditions: (1) normal behavior, (2) weather-related degradation, and (3) configuration changes. Our query results are limited to predicates with a greater than 70% probability of being true. The results are presented in Figure 7. The MLN used input regarding changes in the *wcel_angle* (antenna tilt angle) parameter for approximately 4,000 cells for each day in February, along with weather reports that covered the whole area of interest for February

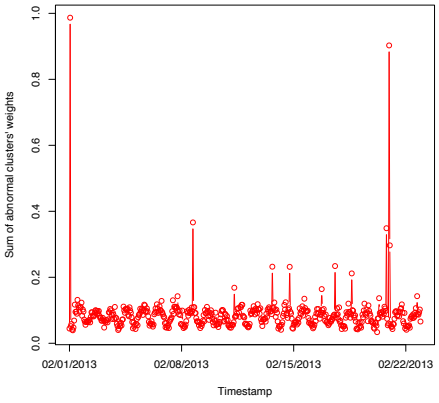

Fig. 5. Portion of the network in abnormal state for February 2013.

2013. The network found some anomalies for 10 February 2013. In Figure 7, we can observe that not all changes in *wcel_angle* served as anomaly triggers, since only a smaller portion of the cells were affected. In our investigation, we also noticed a very interesting trend in the change, which can indicate an automated action (Figure 6).

MLNs are generated in a semi-automated fashion for each timestamp. Figure 8 presents the input and output of PCE tool for February 10th 2013. We can observe that MLN receives input from topic modeling regarding the groups of cells that were deemed anomalous as well as input from the CM data as *wcel_angle* changes. Moreover, the MLN can also accommodate a visibility delay of up to n hours for which CM changes can propagate and affect cells ($n = 48$ in our experiments); hence, the time window in which cells are labeled as anomalous in Figure 7. The output of the PCE tool consists of groups of cells affected by CM changes and normal groups of cells (no cell group was affected by weather events for that day).

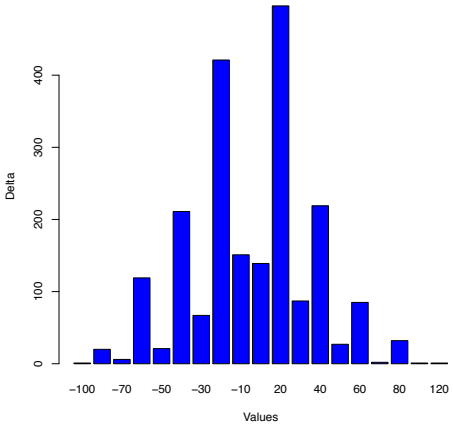

Fig. 6. Deltas between the after and before wcel_angle values for all the cells affected

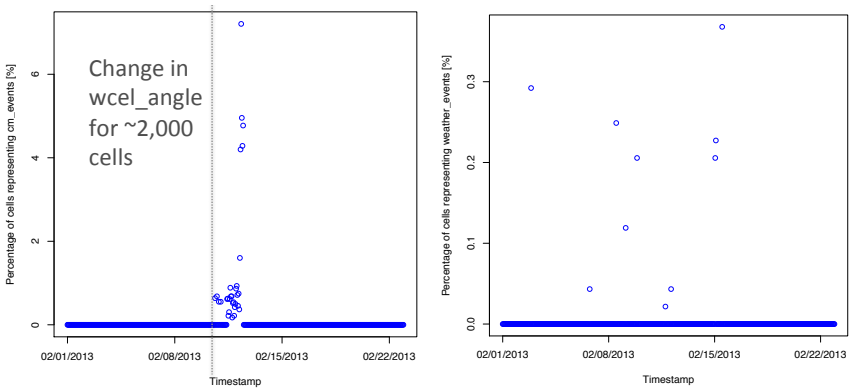

Fig. 7. Percentage of cells diagnosed as anomalous due to *wcel_angle* changes (left) and weather events (right). The dotted vertical line indicates when changes in *wcel_angle* started to occur.

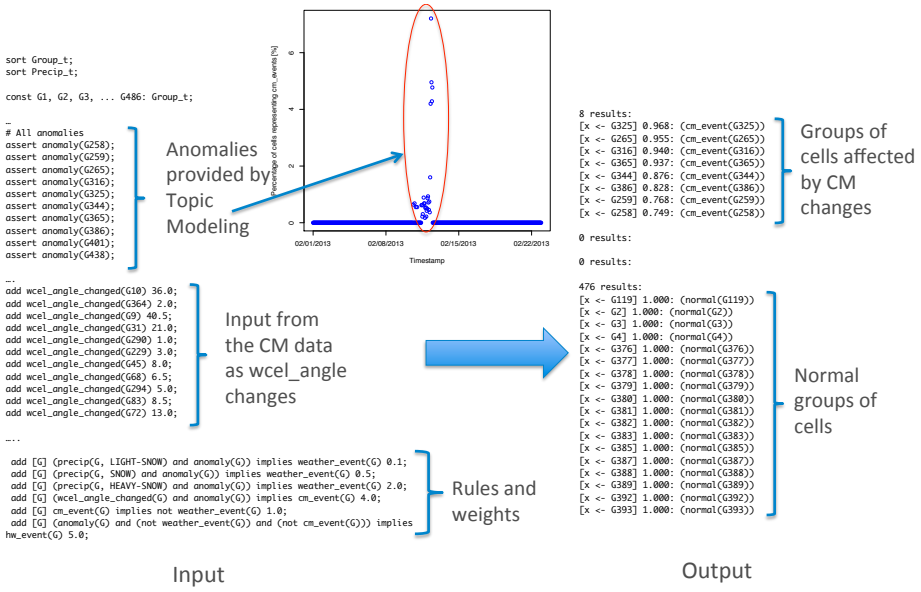

Fig. 8. Input and output of PCE tool for February 10th 2013

3.4 Computational Performance

We implemented the HDP models using the Gibbs sampling, which is a variant of Markov Chain Monte Carlo (MCMC) method that is based on previous sampling of hidden variables. Gibbs sampling is an iterative algorithm, which loops through all timestamps within the valid time range in each iteration. The number of iterations depends on the convergence of the algorithm, and is related to the number of available data. With the full dataset from January to March, the algorithm converged in 10 iterations. Let us denote the total number of timestamps as N, the average number of valid cells at each timestamp as M, the average number of topics at each timestamp as T, the number of global profiles as K, the number of KPIs as F, and the number of digitized bins for features as W. The complexity of topic sampling for each cell data is $O(KF)$, and the complexity of profile sampling for each topic is $O(KFW)$. Therefore, the total complexity for each Gibbs sampling iteration is $O(N(MKF + TKFW))$.

For the detection phase, the Gibbs sampling for computing the probabilities has the same complexity as training. However, the deployed system usually collects KPI data for one time point at a time (N=1) and calls the evaluation process. For a small set of test data like this, the number of iterations to reach convergence is less than the training one, and the HDP evaluation process can respond in real time. On a Linux system with 2.27GHz CPU, an iteration for the evaluation takes 2 seconds with a subset ($\sim$ 1000) of all cells. The evaluation process converged in 2 iterations.

An MLN description consists of "direct" predicates that correspond to observations and "indirect" predicates that are evaluated during the inference process. In contrast to conventional logic systems for which predicates can only be

true or false, MLN predicates can be true with some probability. PCE uses an MCMC approach to inference, relying on sampling to estimate the probabilities of different indirect predicates in the system. Despite the theoretical worst-case complexity of MLN inference, the MCMC approach has distinct advantages for practical application. Our experiments exhibit running times on the order of 1 minute for 486 PCA-derived cell groups and three segmented time intervals. In these experiments, the sampling parameter ranged from 1000 to 10000 runs, a range which appears appropriate for convergence.

4 Related Work

To the best of our knowledge, there are significant methods available for CM and PM analysis; however, none of them fully addresses the SON verification use case. Some of the existent work is only partially automated [2] with tool support for regular network reporting and troubleshooting, while others use disjoint analysis for PM and CM, requiring an additional linking mechanisms which is normally done manually. In terms of PM analysis, most work relates to detection of degradations in cell-service performance. If previous research addressed the cell-outage detection [11], cell-outage compensation [3] concepts and network stability and performance degradation [9, 5] without relying on PM data, more recently, detection of general anomalies has been addressed based on PM data [17, 12, 6, 7]. For CM analysis, Song et al. [15] propose formal verification techniques that can verify the correctness of self-configuration, without addressing the need for runtime verification.

5 Conclusions

This paper proposed a framework for SON verification that combines anomaly detection and diagnosis techniques in a novel way. The design was implemented and applied to a dataset consisting of KPI data collected from a real operational cell network. The experimental results indicate that our system can automatically determine the state of the network in the presence of CM changes and whether the CM changes negatively impacted the performance of the network. We are currently planning to expend our framework to more SON use cases such as troubleshooting and we are exploring other types of data that can be used in the diagnosis process. We envision that additional work is needed to integrate our framework with human operator input.

Acknowledgment We thank Lauri Oksanen, Kari Aaltonen, Kenneth Nitz and Michael Freed for their contributions.

References

1. Probabilistic Consistency Engine, https://pal.sri.com/Plone/framework/Components/learning-applications/probabilistic-consistency-engine-jw

2. Transparent network performance verification for LTE rollouts, Ericsson whitepaper, 2012, http://www.ericsson.com/res/docs/whitepapers/wp-lte-acceptance.pdf.
3. M. Amirijoo, L. Jorguseski, R. Litjens, and L.C. Schmelz, "Cell Outage Compensation in LTE Networks: Algorithms and Performance Assessment," 2011 IEEE 73rd Vehicular Technology Conference (VTC Spring), 15–18 May 2011.
4. D. Blei, A. Ng, and M. Jordan. Latent Dirichlet Allocation. Journal of Machine Learning Research, 3: 9931022. January 2003.
5. A. Bouillard, A. Junier and B. Ronot, "Hidden Anomaly Detection in Telecommunication Networks," International Conference on Network and Service Management (CNSM), Las Vegas, NV, October 2012.
6. Gabriela F. Ciocarlie, Ulf Lindqvist, Szabolcs Novaczki, and Henning Sanneck, Detecting Anomalies in Cellular Networks Using an Ensemble Method, 9th International Conference on Network and Service Management (CNSM), 2013.
7. Gabriela F. Ciocarlie, Ulf Lindqvist, Kenneth Nitz, Szabolcs Nováczki and Henning Sanneck, On the Feasibility of Deploying Cell Anomaly Detection in Operational Cellular Networks , IEEE/IFIP Network Operations and Management Symposium (NOMS), Experience Session, 2014.
8. Gabriela F. Ciocarlie, Chih-Chieh Cheng, Christopher Connolly, Ulf Lindqvist, Szabolcs Nováczki, Henning Sanneck and Muhammad Naseer-ul-Islam, Managing Scope Changes for Cellular Network-level Anomaly Detection, International Workshop on Self-Organized Networks (IWSON), 2014.
9. A. D'Alconzo, A. Coluccia, F. Ricciato, and P. Romirer-Maierhofer, "A Distribution-Based Approach to Anomaly Detection and Application to 3G Mobile Traffic," Global Telecommunications Conference (GLOBECOM) 2009.
10. Griffiths, T., and Steyvers, M.. Finding Scientific Topics. Proceedings of the National Academy of Sciences, 101 (suppl. 1): 52285235, 2004.
11. C. M. Mueller, M. Kaschub, C. Blankenhorn, and S. Wanke, "A Cell Outage Detection Algorithm Using Neighbor Cell List Reports," International Workshop on Self-Organizing Systems, 2008.
12. S. Nováczki, "An Improved Anomaly Detection and Diagnosis Framework for Mobile Network Operators," 9th International Conference on Design of Reliable Communication Networks (DRCN 2013), Budapest, March 2013.
13. Richardson, Matthew, and Pedro Domingos. Markov logic networks. In Machine learning, vol. 62, no. 1-2, 2006, pp. 107-136.
14. S. Hämäläinen, H. Sanneck, C. Sartori (eds.), LTE Self-Organising Networks (SON) - Network Management Automation for Operational Efficiency, Wiley, 2011.
15. JaeSeung Song; Tiejun Ma; Pietzuch, P., Towards automated verification of autonomous networks: A case study in self-configuration, IEEE International Conference on Pervasive Computing and Communications Workshops, 2010.
16. M. Steyvers and T. Griffiths. Probabilistic topic models, Handbook of Latent Semantic Analysis, Erlbaum, 2007.
17. P. Szilágyi and S. Nováczki, "An Automatic Detection and Diagnosis Framework For Mobile Communication Systems," IEEE Transactions on Network and Service Management, 2012.
18. Yee Whye Teh, Michael I. Jordan, Matthew J. Beal and David M. Blei (2006). Hierarchical Dirichlet Processes. Journal of the American Statistical Association 101, 476: 1566-1581.