Alarm Prioritization and Diagnosis for Cellular Networks

Gabriela F. Ciocarlie[1(✉)], Eric Yeh[1], Christopher Connolly[1], Cherita Corbett[1], Ulf Lindqvist[1], Henning Sanneck[2], Kimmo Hatonen[3], Szabolcs Nováczki[4], Muhammad Naseer-Ul-Islam[2], and Borislava Gajic[2]

[1] SRI International, Menlo Park, CA, USA
{gabriela.ciocarlie,eric.yeh,christopher.connolly,
cherita.corbett,ulf.lindqvist}@sri.com
[2] Nokia Networks Research, Munich, Germany
{henning.sanneck,muhammad.naseer-ul-islam,borislava.gajic}@nokia.com
[3] Nokia Networks Research, Espoo, Finland
kimmo.hatonen@nokia.com
[4] Nokia Networks Research, Budapest, Hungary
szabolcs.novaczki@nokia.com

Abstract. Alarm events occurring in telecommunication networks can be an invaluable tool for network operators. However, given the size and complexity of today's networks, handling of alarm events represents a challenge in itself, due to two key aspects: high volume and lack of descriptiveness. The latter derives from the fact that not all alarm events report the actual source of failure. A failure in a higher-level managed object could result in alarm events observed on its controlled objects. In addition, alarm events may not be indicative of network distress, as many devices have automatic fallback solutions that may permit normal network operation to continue. Indeed, given the amount of equipment in a network, there can be a "normal" amount of failure that occurs on a regular basis; if each alarm is treated with equal attention, the volume can quickly become untenable. To address these shortcomings, we propose a novel framework that prioritizes and diagnoses alarm events. We rely on a priori information about the managed network structure, relationships, and fault management practices, and use a probabilistic logic engine that allows evidence and rules to be encoded as sentences in first order logic. Our work, tested using real cellular network data, achieves a significant reduction in the amount of analyzed objects in the network by combining alarms into sub-graphs and prioritizing them, and offers the most probable diagnosis outcome.

Keywords: Network automation · Self-organized networks (SON) · Alarm events · Anomaly detection · Diagnosis · Prioritization

1 Introduction

One of the key challenges faced by telecommunication network operators is the management of alarm events issued by managed objects in the network.

R. Agüero et al. (Eds.): MONAMI 2015, LNICST 158, pp. 28–42, 2015.
DOI: 10.1007/978-3-319-26925-2_3

As network failures can lead to a loss of revenue for the network operator, being able to identify and rectify these failures in a timely fashion is clearly a priority. However, alarm management faces two challenges: volume and descriptiveness. First, the number of discrete alarm events can easily reach tens of thousands per day. This high volume of events makes responding to every alarm a difficult, or more likely, impossible task for most operators. This is also compounded by the fact that there is commonly a level of alarm activity that occurs regularly within a network, and not all of it is indicative of actual network distress. Second, alarms themselves frequently do not describe the actual cause for a failure. For example, alarms may be a result of topological masking, where failures in a high-level or otherwise related component can result in alarms on downstream objects [5]. A failure in a basestation could result in a failure on a cell managed by that basestation, resulting in an alarm event on that cell. Attempting to rectify the problem at that cell would be pointless, as the true cause lies at the basestation level. To make matter worse, these alarms could be reflections of events outside of the network itself. For example, a third-party leased line connecting a set of basestations on the network could unexpectedly break down. The drop in connectivity between these basestations would trigger alarm events indicating failure to connect to each other. However, as the network data model does not describe the leased line, the alarms taken individually would not be able to identify that as the true cause of the failure.

This paper addresses these core needs of organizing and prioritizing alarm events in a network. Current techniques to perform this form of alarm correlation have fallen into two separate bins: data-driven [5,10] and rule-based [10] methods. The former are adaptable to networks, and are intended to facilitate discovery of causes of failure, whether they are repeating patterns of events or a possible point of failure. However, they offer little in the way of concrete diagnostic or prioritization capabilities, as they operate solely on derived statistics and do not incorporate knowledge about the network itself. Rule-based methods offer strong diagnostic capabilities as they do leverage background knowledge about a given network, but are inflexible to new networks or updates to the underlying network itself. Accommodating these changes requires an expert to construct new rules, which historically has proven to be an expensive task. What is needed is a mix of both: a method that can flexibly adapt to new networks while retaining as much background knowledge, and diagnostic capability, as well.

Contributions. This paper proposes novel techniques for determining how anomalous or fault effects are observed through the network using a priori information about the managed network structure, relationships, and fault management practices. The aim is to alleviate operator load by organizing and prioritizing alarm events in a mobile broadband network, and to produce diagnostics when relevant. Main contributions include:

– automatically generating alarm sub-graphs, based on correlated network objects using alarm event temporal windows, FM event data, and managed object and adjacency,

– featurizing the sub-graphs themselves for determining both prioritization and diagnostic information,
– using Markov Logic Networks to easily incorporation background knowledge, in the form of first order logic, into a probabilistic reasoning system.

2 Alarm Prioritization and Diagnosis

We rely on a priori information about the managed network structure, relationships, and fault management practices to determine how the effects of faults are observed through the network. We also alleviate operator load by organizing and prioritizing alarm events, and produce diagnostics when relevant. We model the network as a graph based on hierarchical and adjacency relationships. Unlike prior work that also derived graphs solely from managed object hierarchy information [1], we extract features from alarm sub-graphs, which are network objects correlated together based on managed objects, adjacency information and temporal occurrences of alarms.

2.1 Overall Framework

Figure 1 illustrates the overall approach. Network graph data (1) and alarm event data (2) are both collected by the target broadband network. Our Alarm Sub-graph Generation component (3) performs alarm correlation, incorporating network graph data as well as temporal alarm information to create alarm sub-graphs (4). Sub-graphs are interconnected network objects exhibiting overlapping alarm events.

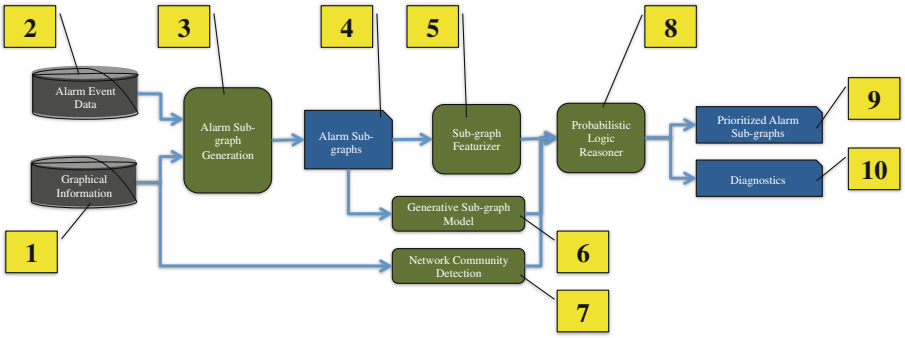

Fig. 1. Overall approach of the alarm prioritization and diagnosis approach. Raw data is depicted in grey, processed data in blue and methods in green.

These alarm sub-graphs are passed to a sub-graph featurizer (5), a novel component that assesses each sub-graph and produces features, i.e. weighted predicates describing characteristics of that alarm sub-graph. Features are derived

from a variety of sources, such as the distribution of object types in the sub-graph, as well as characterizations obtained using probabilistic measures derived over the data, such the rarity of a sub-graph. Additionally, we construct a probabilistic generative model of the alarm sub-graphs (6) to characterize their rarity, and use community detection algorithms (7) to identify clusters of objects that may have a stronger influence on each other. To the best of our knowledge, both of these are novel in their application in the network communications domain.

The components above are used as inputs to a probabilistic logic engine that allows evidence and rules to be encoded as sentences in first order logic (8). Using Markov Logic Networks (MLN) [13] as probabilistic logic engines allows us to incorporate this type of background knowledge more readily than purely statistics-driven approaches. MLN rules are encoded in first order logic, allowing experts to more easily express this knowledge. As MLNs perform probabilistic inference, we can accommodate uncertainty and inconsistencies that plague prior rule-based systems [10]. Rules and observations are assigned weights, and probabilistic inference allows the system to determine marginal probabilities associated with hypotheses of interest (where the marginal probability of a hypothesis is the probability of being true irrespective of the other hypotheses). This enables background knowledge to be merged with evidence in a probabilistic fashion, allowing for incomplete and uncertain observations. The set of rules that operates over the alarm sub-graph evidence is used to derive a probabilistically-motivated ranking of sub-graphs by priority (9), as well as diagnostic information (10) when applicable. While output in the form of prioritized alarm sub-graphs and diagnosis information relates to the state of the art, the processes of generating both the prioritization and diagnosis information are novel.

2.2 Alarm Sub-Graph Formulation and Featurizer

The basis for our alarm correlation and diagnostic system consists of alarm sub-graphs, which are groups of network objects connected by network graph relationships that have exhibited alarm events. Network graph relationships encompass both hierarchical relationships (e.g., between a basestation and its constituent cells) and planned adjacencies. To identify the set of sub-graphs from a graph of a given network and the alarm events within a given time span, we first identify edges between objects that have had at least one temporally overlapping alarm event. We then extract sub-graphs by finding constellations of objects that are not connected to each other.

For each sub-graph, we generate a set of features that characterize that sub-graph. Features are propositions that describe some characteristic of the sub-graph. Example features determined based on sub-graph statistics include:

- All objects are of one type (e.g., basestation, cell);
- The majority of objects (determined as the maximum count per object type) are of a certain type (e.g., basestations, cells);
- The sub-graph is a singleton, containing only one object;
- In the case of singletons, the type of the object;

- The sub-graph is heavily connected, or exhibits a large number of edges (determined by the standard deviation from the mean) between its objects;
- The sub-graph is large, medium, or small, determined by the size distribution over the dataset;
- The sub-graph is rare or frequent, as determined by a probabilistic alarm sub-graph generation model.

2.3 Data-Derived Generative Probabilistic Model

In prior studies, networks have been shown to exhibit a certain background level of alarm activity. However, the alarm background may not necessarily be indicative of a major event. Although every alarm should be resolved, in practice, operators' attention is a limited resource and therefore some alarms have greater priority than others, particularly those that are unusual or rare. To account for this, we construct a probabilistic graphical model that uses statistics from a given dataset to arrive at a score for a given alarm sub-graph, computed as a quantitative measure of priority or likelihood of indicating a root cause. We use this score to rank sub-graphs as frequent versus unusual, with the latter category given higher priority. We formulate a given alarm sub-graph as a Markov model, with the likelihood of observing that sub-graph as:

$$P(O, M, N) = \prod_{o} \prod_{m} P(o'|o, m) P(m|o) P(o|N) P(N) \tag{1}$$

where o represents the type of managed object (e.g., basestation, cell) in the graph, m represents the edge degree of the node, O and M represent the vector of object type assignments and edge degrees for all of the managed objects in the graph, N represents the number of objects in this sub-graph, $P(o'|o, m)$ represents the probability of observing an edge starting with an object with type o and ending with an object type o' (conditioned on the edge degree of the starting object), $P(m|o)$ represents the edge degree for the given object type, and $P(o|N)$ represents the probability of seeing a node with the given object type given the size of the sub-graph.

To simplify the model, we make additional independence assumptions:

- The probability of an edge occurring is independent of the node's degree as well as the number of nodes in the graph, or $P(o'|o, m) = P(o`|o)$, $P(o'|o, N) = P(o'|o)$;
- Edges from node o are determined independently of each other, and are also independent of the number of nodes in the graph, or $P(o'|o) = \prod_{o'} P(o'|o)$, $P(o'|o, N) = P(o'|o)$;
- The node types are determined independently of all the other nodes, or $P(O|N) = \prod_{o} P(o|N)$.

Note that this formulation would double count edges, as we currently do not consider directionality in the model. However, this is unlikely to have negative effects given that our goal is to obtain a ranking between alarm sub-graphs.

To account for possible sparsity, we apply Laplacian noise modeling. Furthermore, if a background corpus of alarm information was available from another network, it can be incorporated via Dirichlet smoothing. In this formulation, the probability estimate is reframed as:

$$P(o'|o) = \frac{I(o'|o) + \mu P(o'_c|o_c)}{E + \mu} \tag{2}$$

where $I(o'|o)$ represents the frequency of edge transition o to o', E is the number of edges, $P(o'_c|o_c)$ representing the probability derived from the background corpus and μ represents a weighting factor, with larger values giving more weight to the background corpus.

Note that an inherent property of this model is that the more objects there are, the lower the likelihood of observing the graph. We view this as a desirable property, as larger alarm sub-graphs would be considered rarer events compared with other sub-graphs, and thus could very well be indicative of network distress.

2.4 Network Community Generation

One of the key techniques in graph analysis is the identification of communities [2], i.e. groups of objects that are more interconnected with each other than one would expect. Objects that are connected would be more likely to "influence" each other, or, in other words, to exhibit any anomalies or faults together, due to underlying phenomena such as topological masking. In our case, we wish to identify groups of nodes that are more likely to influence each other, given their proximity in the influence network. To identify these communities, we apply two methods: modularity analysis and force-directed layout. Modularity analysis generates a "hard" clustering, and assigns nodes to discrete communities based upon how much more strongly they are connected with each other than if edges were assigned by chance. Force-directed layouts are a way to project the relationships between nodes and edges into a lower dimensional space, and are most often used to make tightly connected nodes more apparent in visualizations. This class of algorithms assigns spatial locations for nodes based on an attraction and repulsion algorithm. Specifically, all nodes exhibit mutual repulsion; those with edges between them have an attractive force. A node layout solution that minimizes the overall energy resulting from these forces generally results in strongly connected communities being closer to each other to produce an intuitively understandable visual layout.

The modularity score of a graph, Q, measures how many more edges appear in a community than if the edges were assigned by chance. Q is given by:

$$Q = \frac{1}{2m} \sum_{v,w} [A_{v,w} - \frac{k_v k_w}{2m}] \delta(c_v c_w) \tag{3}$$

where m is the number of edges in the graph, the values v and w are node indices, A_{vw} indicates the actual number of edges between nodes v and w, k_v and k_w are

their edge degrees, and c_v and c_w their modularity class assignments. The term $\delta(c_v c_w)$ is an indicator variable, set to 1 if the modularity assignment for node v is equal to the one for node w. The term $\frac{k_v k_w}{2m}$ corresponds to the expected number of edges between the two nodes, if edges were assigned at random while respecting the original node degrees. As there are $2m$ possible edge assignments, the Q score represents the difference between the fraction of edges occurring within the assigned communities and the fraction that would have been expected in that group. Community assignments c that have an unusually large number of edges between their nodes would thus increase the Q score.

Both the assignments and the number of classes considered are determined by an iterative procedure until the modularity score meets a predetermined threshold, or until no more communities can be added without degrading the score. For our analyses, we used the default setting of 1.0.

For the force directed layout, we used the ForceAtlas2 algorithm [8], which was originally implemented in order to address deficiencies in existing force layout algorithms. Here, the attraction between two connected nodes, $F_a(n_1, n_2)$, is linearly correlated with their distance in the visualization:

$$F_a(n_1, n_2) = d(n_1, n_2) \tag{4}$$

Repulsion between two nodes, $F_r(n_1, n_2)$, is inversely proportional to their distance and directly proportional to the product of their degrees:

$$F_r(n_1, n_2) = \frac{(deg(n_1) + 1)(deg(n_2) + 1)}{d(n_1, n_2)} \tag{5}$$

The intent here is to allow leaves and other poorly connected nodes to move closer to hubs, which have a higher edge degree.

2.5 Markov-Logic Networks for Prioritization and Diagnostics from Alarm Sub-Graphs

Given an alarm sub-graph and a set of constituent features, we combine these against a backdrop of rules and heuristics to obtain prioritization and diagnostic information. Markov Logic Networks (MLN) [13] provide a formalism that allows rules to be expressed in first order logic, but with probabilistic inference. In contrast to approaches like Bayesian Networks [7] or Markov Random Fields [14], the use of first order logic allows a greater degree of flexibility, describing relationships in terms of variables over sets, instead of the instances themselves. Unlike existing rule-based systems, probabilistic inference grants the ability to tolerate noise and uncertainty. These rules are weighted, giving authors the ability to weigh them against each other. This also allows an existing ruleset with different levels of generality to derive hypotheses over a new network. Inference here is conducted using a Monte Carlo sampling algorithm with Boolean satisfiability solving techniques.

To obtain prioritization and diagnostic information from a set of sub-graphs, we first add their features as observations to the MLN inference engine (for our experiments we used the Probabilistic Consistency Engine (PCE) [12]). We then query the system to obtain the probability of occurrence for the set of observations. For prioritization, these probabilities are used to rank sub-graphs for receiving human attention. For diagnostics, they quantify whether a given diagnostic has a higher likelihood of explaining the observations than random chance.

The PCE input language, which is used to express MLNs, consists of the following elements:

1. Definition of types (also called "sorts"). In the alarm sub-graph case, these correspond to alarm sub-graphs.
2. Enumeration of the sets corresponding to each type. In this case, these are the observed sub-graphs.
3. Declarations of the predicates to be used, and the types of each argument. Here, a predicate would be $prioritize(sub-graph)$, which indicates the given alarm sub-graph should be prioritized.
4. A set of weighted clauses comprising the probabilistic knowledge base. An example would be $abnormal(sub-graph) \rightarrow prioritize(sub-graph)2.0$, meaning that if a sub-graph was deemed abnormal by an analysis component, its priority is raised by the given weight.
5. Assertions (predicate forms that express information that is known to be true). Here, these would be the set of observed alarm sub-graphs, along with any other derived observations generated by our analysis tools.

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. MLN solvers generally approach the problem by using a Monte Carlo sampling algorithm with satisfiability (SAT) solving techniques.

3 Experimental Evaluation

This section analyzes our framework applied to a real network dataset. The experimental corpus consists of alarm-event data for approximately 3,300 cells, collected from 11/25/2013 to 12/15/2013. The network elements in this dataset are represented as managed objects, for which relationships with the neighboring elements are given, and which are grouped into managed object classes. The managed object classes were tracked through a hierarchy of elements.

Using the 496 available hourly timeslices in our dataset, we automatically identified the alarmed sub-graphs using one-hour windows and generated a PCE [12] ruleset for each of these timeslices. We then ran inference over each of these timeslices, and queried both the priorities that should be assigned to each alarm sub-graph, and the diagnostic information, if available.

3.1 Alarm Sub-Graphs for Diagnostics and Prioritization

Using our network-wide graph-based visualization tool, we identified several interesting timeslices to analyze, based on the size and number of alarm events. One such interesting timeslice is the one for Dec 1st 2013, from 1700–1800 CET. Within this hour there were 673 alarms; after applying our sub-graph analysis we arrived at 28 alarm sub-graphs, providing a significant reduction in data that a human operator would need to investigate. 16 of these sub-graphs were singletons (only one network object), and the remaining were sub-graphs consisting of multiple objects. Figure 2 illustrates the overall network state when viewed from the topologically motivated point of view. An alternative view is

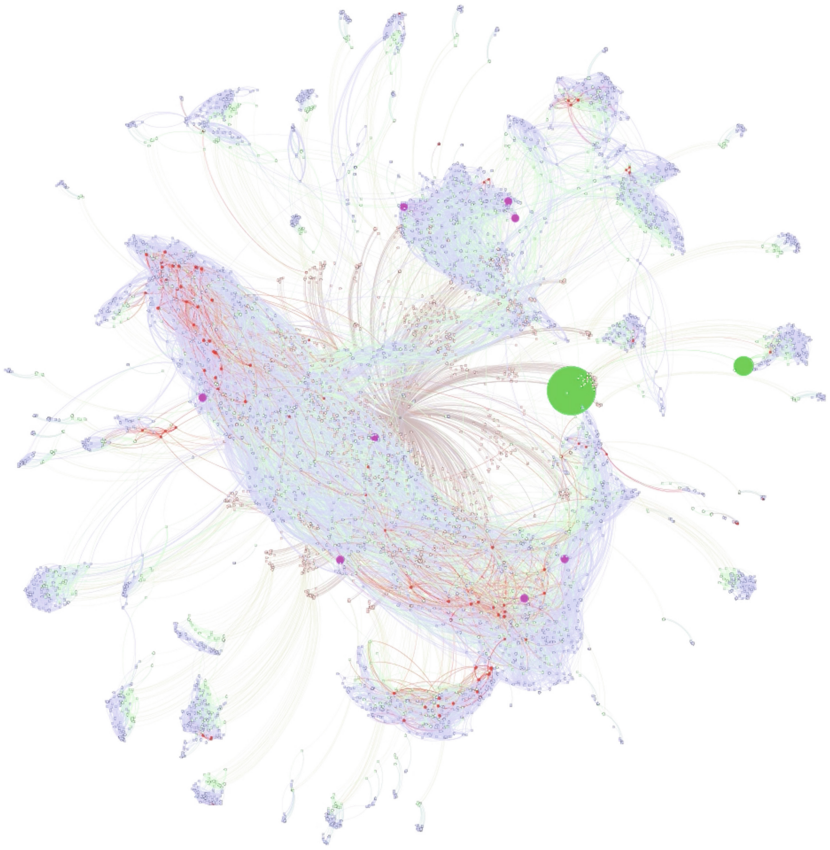

Fig. 2. Graph-level visualization of a portion of the network state at Dec 1st, 1700–1800 CET. Red objects correspond to alarmed objects in the network. Alarmed sub-graphs are highlighted by the red edges between their objects. Purple indicates the object exhibited a high probability of an anomaly. Green indicates a configuration management change was applied, with size indicating the number of changes applied in that window.

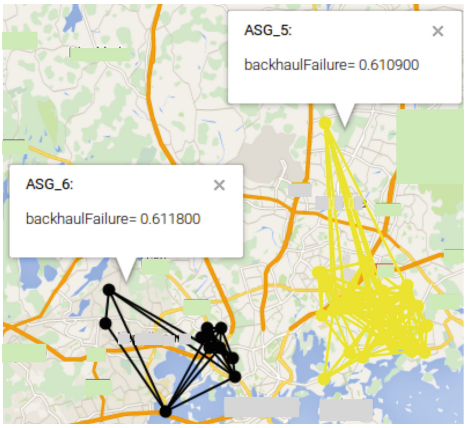

Fig. 3. Graph-level visualization of a portion of the network state at Dec 1st, 1700–1800 CET. The graph-level analysis was overlaid using the lat/long coordinates of the object's multi-radio basestation (MRBTS). Circles correspond to objects that had one or more alarm events within that time window, and edge objects indicate that these belonged to the same alarm sub-graph. As visualization is at the MRBTS level, edges represent planned adjacencies between the MRBTS objects themselves or their objects further down the control hierarchy.

provided by Fig. 3, which illustrates the latitude/longitude coordinates of the alarmed objects. Here, we show a subset of the network state, along with the alarmed objects. As lat/longs were only available for objects at the basestation level, edges are drawn only between basestations or controlled objects that have planned adjacencies with each other.

Table 1 lists the 28 identified alarm sub-graphs with the distribution of managed object types, ranked by priority, which was generated based on the rules presented in Fig. 4. The top five prioritized sub-graphs consisted entirely of base stations (BTS), with multiple alarm events on each node. Of these, the top three were large sub-graphs containing 11–26 objects (note that the likelihood scores are determined based on more features than the number of objects). When querying for possible causes, each of these indicated backhaul failure as a stronger explanation (Fig. 3), given the set of rules that we proposed for diagnosis (Fig. 4).

The second tier of prioritized sub-graphs consisted mainly of clusters of alarmed cells belonging to separate basestations, but connected by planned adjacencies. A more likely possible cause, drawn from the test inventory, was a form of radio interference event. This was derived from the supplementary information field in the alarm data, which contained information such as "Configuration error: Invalid frequency channel for the BTS HW" or "Commissioning error: Invalid Configuration file". The last tier of sub-graphs consisted of singletons; these were prioritized by their position in the control hierarchy.

As severe as alarmed groups of base stations may seem, these alarms occurred frequently enough, when compared with other sub-graphs in the data. An example

Table 1. List of alarm sub-graphs identified in the given timeslice, sorted by the PCE derived priority. The numbers of each type of managed object are also presented (MRBTS=multi-radio basestation; BTS=basestation). Left side presents graphs with more than one node, while right side presents singletons.

ID	Priority	MRBTS	BTS	Cell	Total
ASG_6	0.9560	0	11	0	11
ASG_5	0.9558	0	26	0	26
ASG_0	0.9554	0	13	0	13
ASG_11	0.9027	0	4	0	4
ASG_8	0.9002	0	4	0	4
ASG_2	0.8920	0	0	3	3
ASG_9	0.8911	0	0	3	3
ASG_1	0.8904	0	0	3	3
ASG_10	0.8903	0	0	3	3
ASG_7	0.8767	0	0	2	2
ASG_4	0.8742	0	0	2	2
ASG_3	0.8579	0	1	3	4

ID	Priority	MRBTS	BTS	Cell	Total
ASG_25	0.7948	1	0	0	1
ASG_20	0.7916	1	0	0	1
ASG_12	0.7915	1	0	0	1
ASG_14	0.7907	1	0	0	1
ASG_21	0.7802	0	1	0	1
ASG_22	0.7798	0	1	0	1
ASG_19	0.7786	0	1	0	1
ASG_18	0.7775	0	0	1	1
ASG_23	0.7760	0	1	0	1
ASG_17	0.7754	0	0	1	1
ASG_15	0.7728	0	0	1	1
ASG_13	0.7723	0	0	1	1
ASG_24	0.7710	0	0	1	1
ASG_16	0.7706	0	0	1	1
ASG_27	0.7704	0	0	1	1
ASG_26	0.7700	0	0	1	1

of a rare alarm sub-graph occurred at Dec 13th from 1300–1400 CET. This event consisted of 24 alarmed basestations; the event was considered rare, given the size of the sub-graph. Inspection of the prioritization in that timeslice showed that the rare sub-graph was indeed assigned the highest priority of all identified sub-graphs. In Fig. 5, the graph-level analysis was overlaid using the lat/long coordinates of objects' basestations. We noticed that the basestations were in geographic proximity, indicating high chance of a significant impact in the area.

In addition, we conducted a preliminary evaluation of how well the sub-graph likelihood from the generative model can prioritize sub-graphs with significant network degradation. Using our Key-Performance-Indicator-based topic model [4], we identified 43 events for which cell performance degraded significantly and cells were objects in the sub-graphs. We then summed up the number of this type of events, considering the ones with sub-graph likelihood scores less than or equal to a given threshold, in essence retaining less-likely sub-graphs. Figure 6 presents the percentages of significant degradation events, alarm events, and sub-graphs against the threshold. Although there was a relatively small number of degradation events, the majority of these (39 out of 43) occurred on 25 % of the rare sub-graphs, indicating that prioritization based on likelihood is a viable strategy.

3.2 Computational Performance

Sub-graphs are formed by assessing which objects in the topological graph have an alarm in that time period. We then use a simple iterative agglomerative clustering scheme to group connected objects until no more connections can be made. For a number of alarms A and a total number of distinct objects N, we have a worst case of $O(AN)$ steps for determining which objects have alarms

```
add [x] isAllLNBTS(x) => backhaulFailure(x) 100.0;
add [x] isAllLNCEL(x) => radioInterferenceEvent(x) 100.0;
add [x] isAllLNCEL(x) => interfaceFailure(x) 0.5;
add [x] isLNBTS_LNCEL(x) => basestationBlocked(x) 1.2;
add [x] isMRBTS_LNBTS_LNCEL(x) => issueAtMRBTS(x) 1.1;

# Prioritization strategies
add [x] isRare(x) => priority(x) 10.0;
add [x] isFrequent(x) => priority(x) 1.2;
add [x] isAllMRBTS(x) => priority(x) 4.0;
add [x] isAllLNBTS(x) => priority(x) 2.0;
add [x] isAllLNCEL(x) => priority(x) 1.0;
add [x] isSingleton(x) => priority(x) 0.05;
add [x] ~isSingleton(x) => priority(x) 1.0;

add[x] isHeavilyConnected(x) => priority(x) 1.0;
add[x] ~isHeavilyConnected(x) => priority(x) 0.5;
add[x] isSingleMRBTS(x) => priority(x) 0.4;
add[x] isSingleLNBTS(x) => priority(x) 0.2;
add[x] isSingleLNCEL(x) => priority(x) 0.1;

add [x] isLarge(x) => priority(x) 4.0;
add [x] isMedium(x) => priority(x) 1.0;
add [x] isSmall(x) => priority(x) 0.1;
```

Fig. 4. Example MLN rules applied to each timeslice. The top rules are used for diagnosis, while the rest are intended for prioritization.

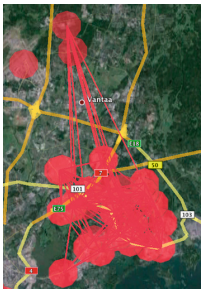

Fig. 5. An example of a rare alarm sub-graph, occurring on Dec 13th from 1300–1400 CET. The graph level analysis was overlaid using the lat/long coordinates of the objects' basestations. Red circles correspond to objects that had one or more alarm events within that time window, and edges objects indicate these belonged to the same alarm sub-graph.

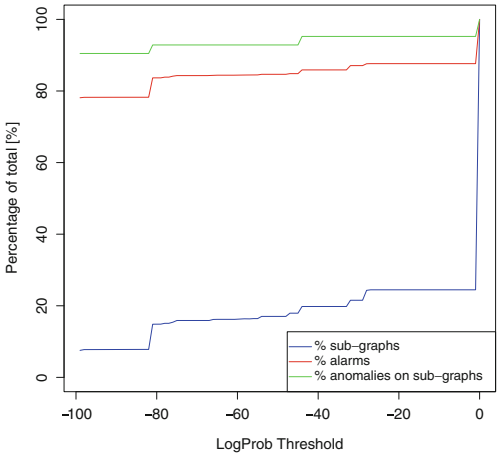

Fig. 6. Percentages of significant degradation events, alarm events, and sub-graphs against the threshold.

when using a strictly naïve analysis without indexing the alarm occurrences. In our case, each iteration in the clustering scheme is $O(A^2)$. Feature computation is strictly $O(FN)$, with F as the number of featurizers in the system. Each featurizer can be considered to run in constant time, due to its templated nature.

PCE uses a Markov Chain Monte Carlo approach for 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. The worst-case complexity of MLN inference is dominated by the predicates with the largest number of arguments. Inference is exponential in the number of arguments of these predicates, since the search space grows exponentially. However, careful design of the rule set can alleviate this problem. For our graph-analysis diagnosis, this growth has not presented a problem. Inference running times grow much more slowly in the number of observations (e.g., number of alarm sub-graphs), usually somewhere between linear and quadratic in the number of observation assertions.

4 Related Work

Alarm correlation utilizes rule-based approaches [10], statistical methods [5,10], and hierarchical and adjacency information between network objects [3,9]. Rule-based approaches, though accurate and explainable, tend to be brittle: generated rules are specific to a network and domain, and usually not generalizable to new networks. These approaches thus require a high degree of maintenance to adjust rules in order to accommodate network updates and changes. The formal methods used by these systems also have a difficult time incorporating uncertain and contradictory information. Statistical methods deal with this uncertainty by design, and can use learning algorithms to adapt to new networks, but cannot incorporate valuable rules describing background knowledge about the network.

Among the alarm correlation methods, one body of work does make use of what we have deemed network graph information [3,6,9]. These dependency-based models are similar to our approach in the use of observations of alarms between objects on the network as graphs. However, some of these methods [3,9] make an implicit assumption that faults originate within objects on the network itself, and much of their effort focuses on identifying a minimal set of objects deemed to be responsible for the observed alarms. However, experience shows that it is entirely possible for faults to be triggered by exogenous causes, and oftentimes information useful for diagnostic purposes must be drawn from background knowledge. Along these lines, Hatonen and Klemettinen [6] explore different adjacencies in the network and use different distance metrics between domain objects to reduce the amount of correlating alarm type combinations. Furthermore, prior work in the dependency-based and statistical methods offer limited capabilities for distinguishing between rare and frequent alarm events, or prioritizing between correlated groups of alarms.

One of the more popular methods for alarm correlation in telecommunication networks is the use of Bayesian networks [11]. This class of techniques has been

shown to be effective for reasoning under uncertainty. However, these methods require probabilities for each of the possible conditions a network must reason about. These must be derived either from data, or in the frequent case that not enough data is available, must be generated from a human expert. The latter case can become a problematic knowledge engineering task, especially in complex scenarios, where a large number of outcomes must be assigned a valid probability value and assessed against each other. Furthermore, standard Bayesian networks are inherently propositional; thus, reasoning about generalizations is difficult.

Finally, most of the available alarm correlation techniques focus on discovering commonly repeating patterns, for the purposes of identifying persistent faults in the network. As a consequence, there has been little attention paid to prioritizing rare groups of alarm events, in particular methods for deriving this from network data. Given equipment in networks can fail on a regular basis, an argument could be made that many of these cases may generally be known and resolvable by automated measures. Thus alarm events that are unusual or novel would be more likely to represent an unaccounted-for failure, and thus would be likelier to require operator intervention.

5 Conclusions

This paper proposed a novel framework for prioritizing and diagnosing faults in broadband networks based on a priori information about the managed network structure, relationships, and fault management practices. Our system reduces the amount of analyzed objects by combining the alarming objects into subgraphs and prioritizing them, and can also derive the most probable cause for the observed alarms. The design was tested on a dataset collected from a real cellular network. We are planning to test our framework in a realtime setting and to adapt it to highly dynamic environments. We are also planning to expand our framework to more SON use cases and to identify other types of data that can be used in the diagnosis process, including configuration management information.

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

References

1. Bandh, T., Carle, G., Sanneck, H.: Graph coloring based physical-cell-ID assignment for LTE networks. In: International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly. ACM (2009)
2. Blondel, V.D., Guillaume, J., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech. Theor. Exp. **2008**, October 2008
3. Bouillard, A., Junier, A., Ronot, B.: Alarms correlation in telecommunication networks. [Research Report] RR-8321, p. 17 (2013)

4. Ciocarlie, G.F., Connolly, C., Cheng, C.-C., Lindqvist, U., Nováczki, S., Sanneck, H., Naseer-ul-Islam, M.: Anomaly detection and diagnosis for automatic radio network verification. In: Agüero, R., Zinner, T., Goleva, R., Timm-Giel, A., Tran-Gia, P. (eds.) MONAMI 2014. LNICST, vol. 141, pp. 163–176. Springer, Heidelberg (2015)
5. Hatonen, K.: Data mining for telecommunications network log analysis. PhD thesis, University of Helsinki (2009)
6. Hätönen, K., Klemettinen, M.: Domain structures in filtering irrelevant frequent patterns. In: Meo, R., Lanzi, P.L., Klemettinen, M. (eds.) Database Support for Data Mining Applications. LNCS (LNAI), vol. 2682, pp. 289–305. Springer, Heidelberg (2004)
7. Heckerman, D.: A tutorial on learning with Bayesian networks. In: Jordan, M. (ed.) Learning in Graphical Models. MIT Press, Cambridge (1999)
8. Jacomy, M., Venturini, T., Heymann, S., Bastian, M.: ForceAtlas2, a Continuous Graph Layout Algorithm for Handy Network Visualization Designed for the Gephi Software. PloS one (2014)
9. Katzela, I., Schwartz, M.: Schemes for fault identification in communication networks. IEEE/ACM Trans. Netw. **3**(6), 753–764 (1995)
10. Martin-Flatin, J.P., Jakobson, G., Lewis, L.: Event correlation in integrated management: lessons learned and outlook. J. Netw. Syst. Manage. **15**(4), 481–502 (2007)
11. Meira, D.M.: A Model for Alarm Correlation in Telecommunications Networks. PhD dissertation, Federal University of Minas Gerais, Belo Horizonte, Brazil (1997)
12. Probabilistic Consistency Engine. https://pal.sri.com/Plone/framework/Components/learning-applications/probabilistic-consistency-engine-jw
13. Richardson, M., Domingos, P.: Markov logic networks. Mach. Learn. **62**(1–2), 107–136 (2006)
14. Wu, C.-H., Doerschuk, P.C.: Cluster expansions for the deterministic computation of Bayesian estimators based on Markov random fields. IEEE Trans. Pattern Anal. Mach. Intell. **17**(3), 275–293 (1995)