Threshold Compression for 3G Scalable Monitoring

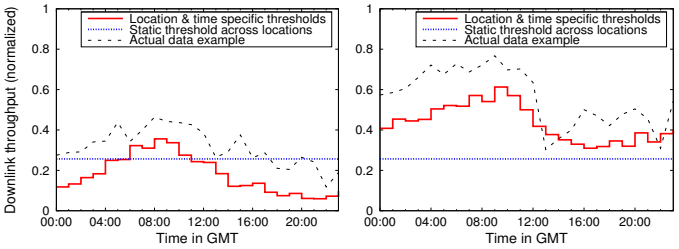
Suk-Bok Lee[1], Dan Pei[2], MohammadTaghi Hajiaghayi[3,5], Ioannis Pefkianakis[4], Songwu Lu[4]
He Yan[3], Zihui Ge[3], Jennifer Yates[3], Mario Kosseifi[6]
[1]Carnegie Mellon Univ. [2]IEEE Member [3]AT&T Labs [4]UCLA [5]Univ. of Maryland [6]AT&T Network Services

***Abstract*—We study the problem of scalable monitoring of operational 3G wireless networks. Threshold-based performance monitoring in large 3G networks is very challenging for two main factors: *large network scale* and *dynamics in both time and spatial domains*. A fine-grained threshold setting (e.g., per-location hourly) incurs prohibitively high management complexity, while a single static threshold fails to capture the network dynamics, thus resulting in unacceptably poor alarm quality (up to 70% false/miss alarm rates). In this paper, we propose a scalable monitoring solution, called *threshold-compression* that can characterize the location- and time-specific threshold trend of each individual network element (NE) with minimal threshold setting. The main insight is to identify groups of NEs with similar threshold behaviors across location and time dimensions, forming spatial-temporal clusters to reduce the number of thresholds while maintaining acceptable alarm accuracy in a large-scale 3G network. Our evaluations based on the operational experience on a commercial 3G network have demonstrated the effectiveness of the proposed solution. We are able to reduce the threshold setting up to 90% with less than 10% false/miss alarms.**

I. Introduction

In this paper, we focus on designing algorithms for scalable monitoring of operational 3G wireless networks. Monitoring of such a wide-area cellular network is challenging for two factors. First, both the network scale and the user population are quite large. The sheer volume of massive data collected from the large number of network elements (NEs) (e.g., Node B and sectors, etc) inside the 3G infrastructure can easily overwhelm a standard monitoring tool. Second, 3G networks exhibit richer dynamics in both temporal and spatial domains compared with their wired counterparts. User-perceived performance tends to vary over time and at different locations, reflecting the human activity over time and mobility-induced service diversity across geographic areas. Consequently, capturing sustained service impairments while ignoring inherent service fluctuations at each NE at runtime becomes important for 3G network monitoring.

The current practice for monitoring the health of a large-scale network is to use pre-defined thresholds of selected key performance indicator (KPI) metrics. However, direct application of such a threshold-based alarming model does not scale in 3G networks due to the two factors identified above. Consider Figure 1, which shows threshold examples of different NEs based on their historical data. A single representative static threshold per KPI fails to capture such spatial and temporal dynamics, leading to unacceptably poor alarm quality with nearly 70% false positives/negatives. On the other hand, a finer-grained location- and time-dependent threshold setting (e.g., each NE has its own thresholds at the given times of day) can capture network dynamics but incurs prohibitively high system management complexity. The number of thresholds to be maintained grows very large with the increasing number of NEs and monitoring time granularity. For example, given that one regional area has about 5,000 cells and 30 KPIs (using the statistics collected from one of the largest commercial 3G networks in the US), the per-NE hourly threshold scheme has as many as $5K \times 24 \times 30 = 3.6$ million thresholds in a single area. Considering that there are also other types of NEs (e.g., Node B, RNC, SGSN, GGSN) to monitor in 3G networks, it is increasingly difficult to monitor a large number of NEs with this fine-grained threshold based scheme. Therefore, naive pre-defined threshold scheme does not scale to operational 3G networks.



(a) Location 1 (b) Location 2

Fig. 1. Thresholding examples of different locations based on historical data. The observation below the threshold is considered alarming condition.

In this paper, we propose a scalable threshold-based solution, called *Threshold Compression*, which has both merits of a small number of used thresholds and accurate capturing of spatial-temporal network dynamics. Our threshold compression approach is motivated by two observations: (1) *certain groups of NEs exhibiting similar threshold behaviors*: The spatial dynamics is attributed to geographic locations of NEs and users in the corresponding region; (2) *stable/similar threshold trends over some period of time*: The temporal dynamics is characterized by the users' 3G usage pattern, thus likely following the diurnal pattern of human activity. Based on these observations, the main insight for our scalable solution is to identify such groups of NEs with similar threshold behaviors across locations and over time, forming spatial-temporal clusters to reduce the threshold settings while retaining acceptable alarm accuracy in a large-scale 3G network.

To this end, we first examine the fundamental tradeoff between the size of threshold settings and the resulting alarm accuracy. We then formulate the threshold compression problem taking the alarm quality into account. We show the hardness of the problems, and then devise a practical threshold compression solution that characterizes the tradeoff via

978-1-4673-0775-8/12/$31.00 ©2012 IEEE

intelligent threshold aggregation. We use real traces to show that our threshold-based solution scales very well with the large number of NEs, and delivers satisfactory performance of small threshold settings without much loss of alarm accuracy in a large 3G network. Overall, we have made three main contributions in this work:

- We examine different types of thresholding schemes for each KPI metric. We observe that some KPIs exhibit strong spatial and temporal dynamics (e.g., user throughput, number of active users, CPU load, etc), and for those KPIs, we find that the current static threshold schemes face severe scaling difficulties for 3G network monitoring. We also find that they work better on some KPIs that show relatively stable performance over time (e.g., accessibility, packet loss rate, etc), but their alarm quality is still far from the operational requirement.
- We observe the similar threshold behavior across certain groups of NEs. We formulate the threshold compression problem taking the alarm quality as well as the management-oriented requirements into account. We prove that this problem is not only NP-hard but indeed very hard to approximate. We develop a practical algorithm suite to identify the spatial-temporal clusters with similar threshold behaviors within the optimization framework.
- We report extensive performance evaluation results based on the operational experience on a commercial 3G network. We have confirmed the effectiveness of the proposed solution. Our solution can reduce the number of thresholds by 90% but still retains accuracy of less than 10% false and miss alarm rates. We also demonstrate the robustness of the solution in that our algorithm yields consistent spatial-temporal groups, rather than arbitrary grouping over time.

II. Preliminaries

In this section, we first present the data sets we use in this paper, then describe the current practice for pre-computing thresholds with different levels of monitoring granularity.

A. Data Sets

The 3G network keeps a large number of counters that log various network events at each NE level, among which are chosen (or combined with multiple counters) as key performance indicators (KPI) to monitor the health of the network.We collected these KPI data from one of the largest commercial 3G service providers in the United States. Our data sets contain 30 KPIs recorded from June 2010 to October 2010 in a single regional area covering thousands of sectors, hundreds of Node Bs, tens of RNCs, and several SGSNs. We note that the data that we use for this study does not contain personally identifiable information.

End-to-End KPI. Some KPIs are categorized as end-to-end (E2E) KPIs. These E2E KPIs are the end-user perceived performance metrics including downlink user throughput, packet loss rate, round trip delay, dropped call rate, etc. They are reported at each NE level in an aggregated manner in the 3G infrastructure hierarchy.

In-network KPI. There are also various types of event measurements collected inside the 3G infrastructure, which are called in-network KPIs. These KPIs include the number of users in cell, average CPU load, RNC utilization, connection setup success rate, paging success rate, retainability rate, etc. They are also reported at each NE level in an aggregated manner in the 3G infrastructure hierarchy.

Measurement normalization. For proprietary reasons, all KPI values presented in this paper are normalized by an arbitrary constant. Normalization does not change the dynamic range represented in figures.

B. Pre-computing Thresholds

The current practice for monitoring the network is to set statistically-derived threshold levels, allowing alarming conditions to be identified as a function of the mean and standard deviation of the historical data [13]. The mean defines what might be considered a normal reading for a given KPI, and the standard deviation gives a way of determining the probability that the KPI will vary from the mean, so as to help differentiate normal variations from an abnormal event.

Removing major anomalies. Before calculating the mean and standard deviation, the data should be reviewed for the presence of outliers that are unusually low or high and are obviously not part of a normal data scheme. Such anomalies, if not removed, can lead to a biased mean and standard deviation, thus contributing to misleading threshold levels. We filter out major anomalies from our data sets via Holt-Winters (HW) forecasting method [14] that can handle trend as well as seasonality, so that diurnal, weekly, and seasonal patterns of individual NEs are considered accordingly.

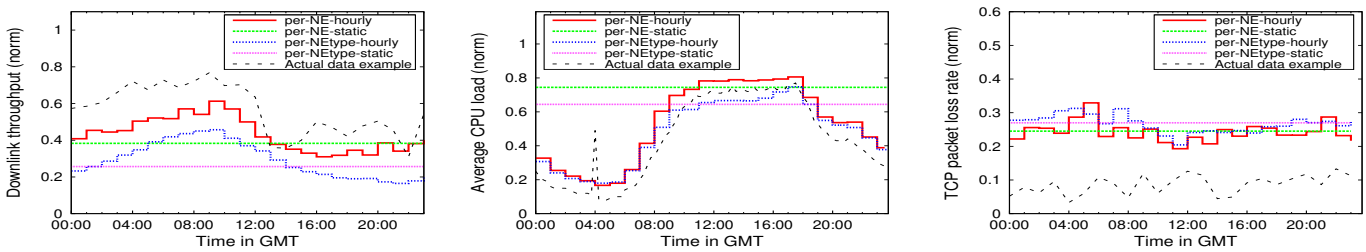
Mean and standard deviation. Based on the data without anomalies, the mean (μ) and standard deviation (σ) are calculated. The threshold value (T) is defined according to the alarming direction (dip or spike)[1] of each KPI:

$$T = \mu - c \times \sigma, \qquad \text{for "dip" KPIs}$$
$$T = \mu + c \times \sigma, \qquad \text{for "spike" KPIs}$$

where c is a control-limit parameter that determines the threshold below ("dip" KPIs) or above ("spike" KPIs) which the observation is considered alarming condition. It is common practice to use two (/one) standard deviations from the mean as critical (/warning) alarm levels. Setting thresholds at the two standard deviation level gives the 95% confidence interval for a normal distribution. In this study, we report the results based on $c = 2$ as the critical alarm level unless otherwise stated.

Thresholding granularity. There are several possible thresholding schemes with different monitoring granularity. We consider four representative schemes: (1) per-NE-hourly: The above threshold pre-computation is performed for each

[1]"Dip" KPIs include user throughput, connection success rate, etc. "Spike" KPIs include packet loss rate, average CPU load, etc.



(a) DL-throughput KPI (dip) (b) CPU-load KPI (spike) (c) Packet-loss KPI (spike)

Fig. 2. Different threshold schemes on a certain NE. Dip (/spike) indicates threshold direction: observation below (/above) is considered alarming condition.

Threshold scheme	#thresholds	FPR	FNR
per-NE-hourly	**25320**	-	-
per-NE-static	1055	**31.1%**	**51.8%**
per-NEtype-hourly	24	**51.2%**	**47.5%**
per-NEtype-static	1	**53.2%**	**58.0%**

TABLE I
THRESHOLDING ON DL-THROUGHPUT KPI.

Threshold scheme	#thresholds	FPR	FNR
per-NE-hourly	**39144**	-	-
per-NE-static	1631	**57.9%**	**76.7%**
per-NEtype-hourly	24	**71.2%**	**80.3%**
per-NEtype-static	1	**73.5%**	**87.4%**

TABLE II
THRESHOLDING ON CELL-USER-COUNT KPI.

Threshold scheme	#thresholds	FPR	FNR
per-NE-hourly	**25320**	-	-
per-NE-static	1055	**10.2%**	**37.2%**
per-NEtype-hourly	24	**11.0%**	**50.3%**
per-NEtype-static	1	**11.3%**	**51.2%**

TABLE III
THRESHOLDING ON PACKET-LOSS KPI.

individual NE for each hour,[2] so as to capture the performance trends on specific location and hour; (2) per-NE-static: The thresholds are computed individual-NE based (with aggregating all hours) and thus, each NE has a single (location-specific) threshold value per KPI; (3) per-NEtype-hourly: The thresholds are computed for each hour (with aggregating all NEs of the same type) such that hourly thresholds are applied to all NEs (e.g., all Node Bs); (4) per-NEtype-static: The threshold is computed via aggregating all hours and all NEs of the same type, thus resulting in a single threshold per KPI.

III. SCALING LIMITATIONS OF THRESHOLD METHODS

We use the threshold settings and the resulting alarm quality based on our KPI data (from one regional area in an operational 3G network) to elaborate the scaling difficulties of each threshold solutions introduced above.

Per-NE-hourly thresholding is ideal for monitoring the dynamic nature of 3G network characteristics (see Figure 1), thus enabling to detect an abnormal event from the location- and time-specific normal variations. However, the number of thresholds to be maintained per KPI grows very large ($\times 24$) with the number of NEs in the monitoring area. Tables I, II, and III show the number of thresholds required for each schemes on three different NodeB-level KPIs when monitoring a single area that covers nearly thousand Node Bs; per-NE-hourly requires more than 25K thresholds per NodeB-level KPI. Note that there are also other types of NEs each associated with 30 different KPIs. Our dataset (nearly 6K NEs in total) indicates that per-NE-hourly can have as many as $6K \times 24 \times 30 = 4.3$ million thresholds in a single area, thus making it increasingly difficult to manage a large number of NEs with this fine-grained thresholding approach.

On the other hand, aggregate-based threshold schemes (per-NE-static, per-NEtype-hourly, per-NEtype-static) have small threshold settings, but all result in very poor alarm quality. We evaluate the alarm accuracy of those schemes based on the alarm statistics of per-NE-hourly.We observe that those schemes lead to unacceptably high false positive rate (FPR) and false negative rate (FNR). The tables clearly show the fundamental trade-off relationship between cost (i.e., the size of threshold setting to be maintained) and gain (i.e., alarm accuracy) in 3G network monitoring.

The main reason for such high false/miss alarm rate is that, *simple aggregate-based thresholds fail to capture 3G network dynamics, i.e., location and time specific behavior of each NE at a given time*. Figure 2 depicts the threshold settings by the different schemes at a certain Node B. Neither of static threshold schemes (per-NE-static, per-NEtype-static) can appropriately react to the temporally abnormal situations. For example in Figure 2(b), abnormal spike at 4 am is not detected (i.e., false negative) and furthermore, per-NEtype-static falsely raises an alarm for the normal high-load observations between 10 am to 6 pm (i.e., false positive). Temporal-only thresholding (per-NEtype-hourly) is better, but its alarm accuracy is still poor due to lack of location-specific trends.

Nevertheless, one interesting observation is that, for packet-loss KPI in Figure 2(c), threshold settings of all different schemes somewhat overlap over time. We also observe similar trend in some other KPIs such as accessibility, retainability that show stable performance over time. This can explain the relatively better alarm quality (lower FPR and FNR in Table III especially when per-NE-static is applied). We however note that this alarm quality is still far from the operational requirement as detailed in Section VI.

In summary, the fine-grained thresholding that captures well the 3G network dynamics imposes significantly high management complexity, while the simple aggregate-threshold schemes result in too poor alarm accuracy to be employed in the operational 3G monitoring system. This tradeoff relationship is a key challenge for large-scale 3G network monitoring.

[2]In this study, we use hourly bin, since going with finer granularity (e.g. 15-min) offers marginal benefit but increases complexity.

IV. Overview of Threshold Compression

The main insight for our solution is that, although each NE has its own spatial and temporal thresholding behavior, such dynamic trend is not completely unique across locations and across time. In other words, a certain group of NEs (or some period of times) show quite similar threshold behavior. Taking the fine-grained thresholds (per-NE-hourly) as input (see Figure 3), threshold-compression identifies such similar groups of NEs and hours, and forms spatial-temporal clusters to have a small threshold setting while maintaining good alarm quality for large-scale 3G network monitoring.

Case for similar threshold behavior. Our threshold compression approach is motivated by two key observations: (1) threshold behavior similarity among a certain group of NEs, and (2) stable/close threshold trends over some period of time. Figure 4 shows the cumulative distribution of how many other NEs (proportional to total NE count) have close threshold values (difference within 10%) to that of each individual NE on Downlink-throughput-KPI. We observe a quite high spatial similarity in the NE-pairwise comparison. For 50% NE cases, each bears at least 10 hours of similar thresholds to other 25% NEs; 15 hours of similarity to other 17% NEs. We also find some NE-pairs that have similar threshold behavior across all hours. Figure 6 presents such example NE-pairs. This spatial similarity is attributed to the geographic locations of NEs and the user population in the corresponding area. For example, NEs in the same metropolitan area are likely to have similar high demand (i.e., competing cell capacity with more users).

Time-domain similarity is more evident and easier to understand. Figure 5 depicts the hourly pattern of the mean number of active users in a cell (averaged across NEs) via cell-user-count KPI. We observe that there are primarily two stable periods in time. There are a lot of active users between 12:00 GMT and 22:00 GMT and a relatively small number of users between 02:00 GMT and 07:00 GMT, which correspond to a day time and a night time in North America. This clear daily trend is attributed to the users' 3G usage pattern, likely following the diurnal pattern of human activity. Such trend is also observed in other KPIs: similar high (/low) demand during peak (/sleep) hours. Although their diurnal shape is not as apparent as the cell-user count (due to the fact that user-population is just one of the impacting factors), we still observe ample opportunity to exploit temporal similarity. For example in Figure 6, each NE-group shows very stable threshold behavior during peak hours between 11:00 GMT and 22:00 GMT, which can form a temporal-domain cluster.

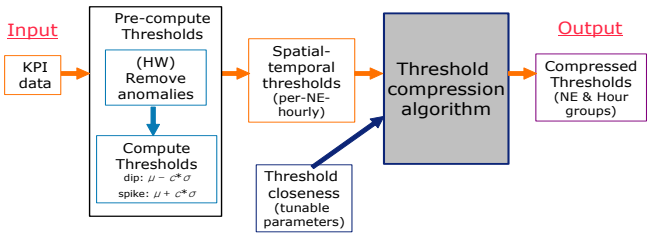


Fig. 3. Threshold compression solution overview.

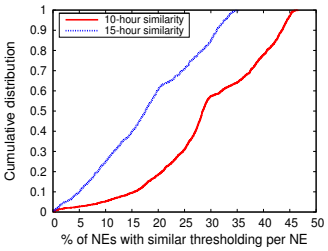


Fig. 4. CDF: NE-pairwise threshold similarity

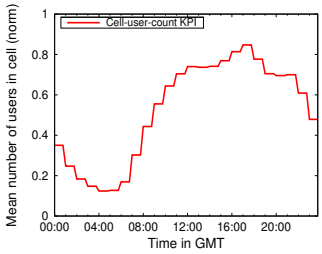
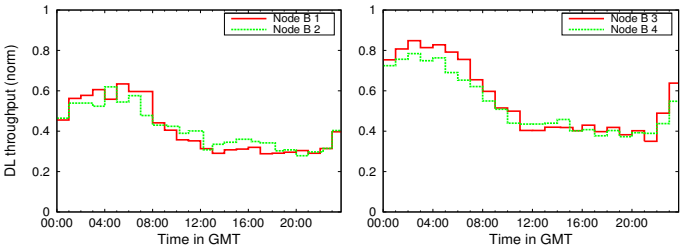


Fig. 5. Active user count in cell (averaged across NEs)



(a) NodeB1 and NodeB2 (b) NodeB3 and NodeB4

Fig. 6. DL-throughput KPI: similar threshold behavior of different Node Bs.

Desirable properties of threshold compression. To ensure scalable monitoring performance as well as practical threshold management, threshold-compression should have the following properties: (1) High compression gain: The resulting threshold setting should remain small even with a large number of NEs; (2) Low false alarm rate: The compressed thresholds must result in good alarm quality, i.e, low false positive and false negative rates, and thus, we use a concept of *threshold closeness* to approximate the per-NE-hourly thresholds. Figure 7 illustrates the threshold closeness with two input parameters α and β that define the bounds at which false positives and false negatives are equal to α and β proportions to the total number of historical data points (at each NE and hour). According to the parameters, each per-NE-hourly threshold $T^{i,j}_{orig}$ has lower ($T^{i,j}_{lower}$) and upper ($T^{i,j}_{upper}$) bound of (NE i and hour j), creating a *permissible interval* for corresponding threshold-compression threshold $T^{i,j}_{comp}$. Note that each α and β takes a value between 0 and 1, and they are the tunable input parameters for threshold-compression algorithm (see Figure 3); (3) Management-oriented policy: The spatial-temporal clusters must be easy to manage and update in the monitoring system. To this end, we employ a consistent NE grouping policy where each NE can belong to only one NE group (but there can be multiple hour groups within an NE group), hence a two-level hierarchical clustering structure.

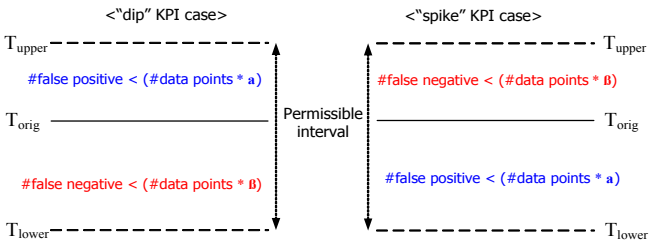


Fig. 7. Threshold-closeness defined by the number of false positives and false negatives based on input parameters α and β applied to historical data.

V. Threshold Compression Algorithm

In this section we present the detailed procedure of threshold-compression that provides an optimization of the desirable properties described in the previous section. To this end, we first formulate the threshold compression problem taking the alarm quality as well as the required clustering policy into account. We then show the hardness of the problem. Finally, we present a practical algorithm suite that can intelligently identify spatial-temporal clusters of similar threshold behaviors and generate the associated compressed thresholds.

A. Problem Formulation

We first introduce the following notations. Let N and H be the set of NEs in the region and the set of time steps (e.g., hours), respectively. We use spatial-temporal block, $\text{STB}_{i,j}$ to denote NE $i \in N$ of time step $j \in H$ in two-dimensional spatial and temporal space. We let $C(i,j)$ represent the cluster ID to which threshold-compression assigns $\text{STB}_{i,j}$. Each cluster x is given a compressed threshold $T_{comp}(x)$ that is shared by all member $\text{STB}_{i,j}$ in x, thus $T^{i,j}_{comp} = T_{comp}(x)$ such that $C(i,j) = x$. We formulate the threshold compression objective as the following optimization problem.

Objective function:

The goal is to find the minimum number of spatial-temporal clusters (or equivalently the minimum threshold setting) from a given fine-grained threshold setting.

$$\min \ |\{C(i,j) : i \in N, j \in H\}|$$

Constraints:

(**C1**) Each compressed threshold must be within the permissible threshold interval of each cluster member $\text{STB}_{i,j}$.

$$\forall i, \forall j, C(i,j) = x \quad \Rightarrow \quad T_{comp}(x) \leq T^{i,j}_{upper}$$
$$\forall i, \forall j, C(i,j) = x \quad \Rightarrow \quad T_{comp}(x) \geq T^{i,j}_{lower}$$

(**C2**) NE grouping must be consistent across time. In other words, each NE can belong to only one NE group, which however can have multiple hour groups.

$$\forall i, i', C(i,j) = C(i',j) \ \Rightarrow \ \forall j, C(i,j) = C(i',j)$$

(**C3**) (Optional rule): Each cluster must consist of continuous time steps.

$$\forall j \leq j', C(i,j) = C(i,j') \ \Rightarrow \ j \leq k \leq j', \forall i, C(i,j) = C(i,k)$$

We explore the fundamental nature of this threshold compression problem by seeking an efficient algorithm that minimizes the cluster count while satisfying constraints (1)-(3).

B. Hardness Result

We show that unfortunately, the threshold compression problem is not only NP-hard (regardless of the continuity optional rule) but indeed very hard to approximate as well.

Theorem 1: Threshold compression problem is inapproximable within $\Omega(n^{1-\epsilon})$ for any $\epsilon > 0$, unless ZPP = NP, where n is the number of NEs.

The proof is omitted due to space constraints, and can be found in our technical report [8].

C. Threshold Compression Algorithm Suite

We now present our practical solution to the threshold compression problem. We take a two-staged approach. We first decouple the spatial NE grouping from the original two-dimensional clustering problem, then further proceed with temporal-domain clustering within each identified NE group. Our two-staged approach is not only motivated from the inherent problem complexity, but also driven by the consistent NE grouping policy (hence a two-level hierarchical structure) described in Section IV.

Our key strategy for clustering is to combine STBs if they (i) have common intersection in their permissible intervals represented by $T^{i,j}_{lower}$ and $T^{i,j}_{upper}$, and (ii) meet the consistent NE grouping rule. Note that having common intersection among the cluster members ensures the satisfying alarm quality. By setting a compressed threshold within the common intersection, the operator expects to have the desired low false alarm rate specified by input parameters α and β.

1) NE grouping: Greedy coloring approach: The first stage identifies NE groups each showing similar threshold behavior each hour among its members. Here, the concept of NE group is a *logical* one. As the first-level of clustering hierarchy, each NE group, in fact, consists of 24 hour-groups, which will be compressed further in the next stage via time-domain clustering. In other words, those 24 hour-groups are associated with the same set of NEs when we refer to the NE group. As a pre-processing step, the permissible threshold intervals $T^{i,j}_{lower}$ and $T^{i,j}_{upper}$ of all $\text{STB}_{i,j}$ are first calculated based on per-NE-hourly thresholds by applying α and β to the historical measurement data. Then, a group of NEs who have common intersection in their threshold intervals each of 24 hours form an NE group.

The NE grouping problem thus naturally reduces to the graph coloring that asks the minimum number of colors (NE groups) assignable to each vertex (NE) such that no edge (common intersection) connects two identically colored vertices (group members). This graph coloring instance is NP-hard as well, and we employ a greedy coloring heuristic, which works quite well in practice. Specifically we apply the Welsh-Powell algorithm [12] that uses at most $\max_i \min\{d(v_i)+1, i\}$ colors, that is at most one more than the maximum degree of the graph.

We first convert our problem instance to a graph $G(V,E)$, where each NE corresponds to a vertex in G. For each vertex pair v_i and $v_{i'}$, we put an edge between them if their counterpart NEs i and i' have disjoint threshold intervals in any hour. Then the vertices colored γ (by the greedy coloring algorithm) can be readily transformed to NE-group γ in our problem. We note that such (multiple) NEs among which do

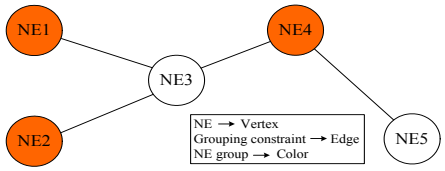


Fig. 8. Transforming NE grouping to coloring.

not share an edge in G (e.g., NE1, NE2, and NE4 in Figure 8) indeed have *common* intersection in their intervals every hour, even though we construct G via pairwise operations.

Lemma 1: If a group of NEs, when converted into the graph coloring instance, do not have an edge among them, they all must have common interval intersection every hour.

Proof: Consider NEs $v_1, v_2, \ldots, v_\ell$ that do not have an edge. It means for every hour, any two of them have a non-empty intersection of their corresponding threshold intervals. We show that indeed in every hour, all $v_1, v_2, \ldots, v_\ell$ have one common non-empty intersection of their corresponding intervals. First, note that by a simple induction on h, one can show common intersection of h intervals is just one interval. So it only remains to show that this interval is not empty, i.e., has at least one point in it. This follows since the minimum upperbound among all pair-wise intersection intervals is such a point which belongs to all intervals. ■

Once identified, each NE group γ defines its own permissible threshold interval $\Phi^{\gamma,j}_{lower}$ and $\Phi^{\gamma,j}_{upper}$ (for each hour j) to reflect each member's interval:

$$\Phi^{\gamma,j}_{lower} = \max_{i \in C_\gamma}\{T^{i,j}_{lower}\},\ \Phi^{\gamma,j}_{upper} = \min_{i \in C_\gamma}\{T^{i,j}_{upper}\}$$

Setting the group threshold interval to the common intersection among the members makes the next-stage clustering procedure to keep control on the resulting alarm quality. Recall that each NE group retains 24 hour-groups until the time-domain clustering.

The running time of the first-stage NE grouping algorithm is bounded by the initial graph conversion process that has $O(|N|^2|H|)$, as the time complexity of the Welsh-Powell coloring algorithm is proven to be $O(|N|^2)$, and the subsequent NE-group interval setting takes $O(|N||H|)$.

2) Hour grouping: Minimum cover selection: As the next level of the clustering hierarchy, the time-domain clustering takes the NE grouping result as input to perform the hour grouping for each identified NE-group. Here, we focus our description on the hour grouping on a certain NE group γ, as it is an identical procedure for all other NE groups. Within NE group γ, there are initially 24 hour-groups, each of which we simply refer an hour. Then each hour j is represented by its threshold interval $\Phi^{\gamma,j}_{lower}$ and $\Phi^{\gamma,j}_{upper}$ (i.e., the common intersection among all members at hour j) as a result of NE grouping.

Given the set of intervals, the hour grouping problem is to find the minimum number of interval groups such that (i) each interval belongs to one of the interval groups, and (ii) there is common intersection in each interval group. We use a simple greedy algorithm that leads to an optimal solution to this problem. The algorithm is as follows (Figure 9 illustrates the process). We first sort all the interval endpoints ($\forall j \in H$: $\Phi^{\gamma,j}_{lower}, \Phi^{\gamma,j}_{upper}$) in ascending order of their values. We scan the list (in ascending order) until first encountering an upperbound point $\Phi^{\gamma,j'}_{upper}$. We then put all intervals containing this point (i.e., all hours $j : \Phi^{\gamma,j}_{lower} \leq \Phi^{\gamma,j'}_{upper}$) into a new interval group C'_h, and delete them from the list. We repeat this process until there is no interval in the list. At first glance, it is certainly not obvious that this simple greedy rule returns an optimal set of interval groups. We show that our greedy rule indeed finds the minimum number of interval groups.

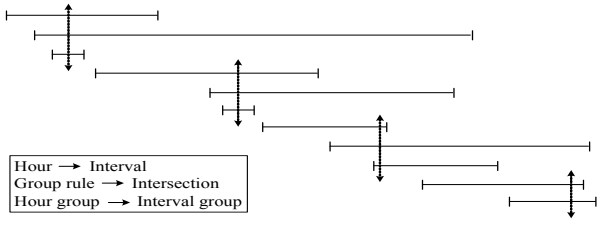


Fig. 9. Minimum interval group selection example. Intervals (horizontal lines) are sorted by start points. Vertical lines indicate the interval groups found by the optimal greedy algorithm.

Theorem 2: The greedy interval selection rule produces the optimal minimum number of hour groups.
The proof can be found in our technical report [8].

Now, all hours in each identified interval group C'_h of NE group γ can form a spatial-temporal cluster C''_δ. In order to preserve the threshold-closeness property for all members, we compute the common intersection across all NEs $i \in C_\gamma$ and hours $j \in C'_h$ in the spatial-temporal cluster:

$$\chi_{lower} = \max_{i \in C_\gamma, j \in C'_h}\{T^{i,j}_{lower}\},\ \chi_{upper} = \min_{i \in C_\gamma, j \in C'_h}\{T^{i,j}_{upper}\}$$

Finally, we set the compressed thresholds $T_{comp}(\delta)$ within the common intersection, and we use the median point in this study:

$$T_{comp}(\delta) = (\chi_{lower} + \chi_{upper})/2$$

We again note that this compressed thresholds $T_{comp}(\delta)$ is shared by all NEs and hours in C''_δ, thus reducing the threshold setting while still preserving the location and time specific thresholds.

Continuous hour grouping. The above optimal greedy algorithm can be naturally extended to the continuous hour grouping. We sort the intervals by hours instead of their values, and group maximum possible contiguous hours having common intersection with the smallest hour in the list, then delete them in the list; we repeat the process until covering all intervals. It is not hard to see that this greedy algorithm also returns the optimal continuous hour grouping solution.

The hour grouping process is executed for each NE group, and therefore the running time of this second-stage algorithm is $O(|N||H|\log|H|)$ for both (dis)continuity rule cases. Therefore, the time complexity of the entire algorithm suite is still bounded by the first-stage graph conversion process $O(|N|^2|H|)$, as the first- and the second-stage algorithms run in sequence, and $|N|$ is much larger than $|H|$ in practice.

VI. Evaluation

We evaluate the performance of threshold-compression on our historical data obtained from the commercial 3G network. This training dataset contains 30 KPIs recorded from June 2010 to August 2010 in one regional area that covers several thousands of NEs. We show that the compression results on this training data are very positive, which will be further validated via operational experience in Section VII.

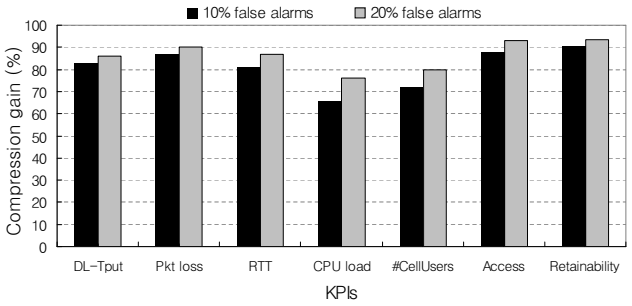


Fig. 10. Compression gain on different KPIs. Each compression gain value represents the highest gain when the resulting FPR and FNR are both within 10% and 20% ranges.

Threshold scheme	#thresholds	FPR	FNR
per-NE-hourly	**25320**	-	-
threshold-compression	3763	8.4%	2.7%
per-NE-static	1055	**31.1%**	**51.8%**
per-NEtype-hourly	24	**51.2%**	**47.5%**
per-NEtype-static	1	**53.2%**	**58.0%**

TABLE IV
THRESHOLDING ON DL-THROUGHPUT KPI.

Threshold scheme	#thresholds	FPR	FNR
per-NE-hourly	**27120**	-	-
threshold-compression	3969	10.2%	5.8%
per-NE-static	1130	**14.4%**	**23.5%**
per-NEtype-hourly	24	**41.0%**	**45.2%**
per-NEtype-static	1	**41.1%**	**46.0%**

TABLE V
THRESHOLDING ON RTT KPI.

Threshold scheme	#thresholds	FPR	FNR
per-NE-hourly	**32160**	-	-
threshold-compression	3538	6.1%	3.4%
per-NE-static	1340	**23.7%**	**35.8%**
per-NEtype-hourly	24	**36.2%**	**84.4%**
per-NEtype-static	1	**34.0%**	**86.3%**

TABLE VI
THRESHOLDING ON ACCESS-SUCCESS-RATE KPI.

A. Results

Figure 10 shows the threshold compression gain on different KPIs. The compression gain is defined as the threshold-setting reduction relative to the fine-grained per-NE-hourly setting. Each compression gain in the figure represents the highest threshold-compression gain observed when the resulting false/miss alarm rates FPR and FNR (based on the per-NE-hourly alarm statistics) are both within 10% (and 20%) range. We observe that, within 10% false/miss alarm condition, most KPIs show very high compression gain nearly 80–90%. Consulted by the operations team, we consider a 10–15% FPR still within the acceptable false alarm range.[3] Indeed, they are willing to accept FPR even up to 20%, since a moderate level of false positive alarms only creates some additional manpower to drill-down the events. On the other hand, the operations team puts more stringent control on false negatives (i.e., FNR less than 10%), because the consequence of miss alarms is more severe than that of false alarms. Thus, we set

[3]In this study, we use slightly different definitions of FPR = FP/(FP+TP) and FNR = FN/(FN+TP), to adapt them to the context where TP is much smaller than TN.

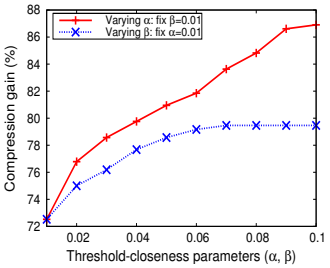


Fig. 11. Compression gain by threshold-closeness parameters α and β on DL-throughput KPI.

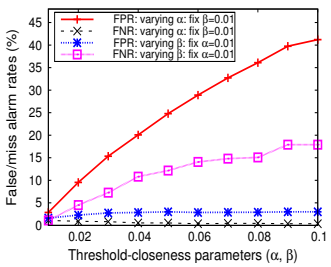


Fig. 12. False/miss alarm rates trend by input parameters α and β on DL-throughput KPI.

the target alarm accuracy within 15% and 10% for FPR and FNR respectively in our evaluation.

To show the explicit benefits of threshold-compression over the existing schemes, we compare all the threshold-setting sizes and false/miss alarm rates produced by each thresholding scheme. Tables IV, V, and VI show the thresholding results on three different NodeB-level KPIs. As shown in the table, threshold-compression balances very well the problematic tradeoff relationship between the threshold setting and the alarm quality, while other schemes are unable to achieve both. We see that threshold-compression produces the comparable scale of threshold setting with per-NE-static but with much lower false/miss alarms. We point out that, for certain KPIs (e.g., RTT KPI in Table V), per-NE-static thresholds give somewhat relatively better alarm accuracy than the other three straightforward approaches, yet it is still far from the operational requirement.

Compression gain. Threshold-compression can achieve higher (or lower) compression gain by tuning the input parameters α and β that explicitly control the proportions of false and miss alarms to the historical data points. Increasing (/decreasing) those values extends (/reduces) the permissible threshold intervals in all NEs at all hours, thus implementing more strict (/relaxed) grouping rule. Figure 11 depicts the compression gain trend on DL-troughput KPI with these parameters. Here, we vary one parameter while fixing the other to 0.01. We see that (i) compression gain increases with each parameter and (ii) it diminishes with parameter β. This is because inherently the number of TP (true positives) is much smaller than that of TN (true negative) in our context, therefore limiting the FN-associated threshold range (which is controlled by β). We however note that high compression gain via increasing parameters may not be always desirable, since it comes with sacrificing alarm quality as we describe below.

False/miss alarm rates Figure 12 plots the corresponding FPR and FNR in the same experiment. We see that FPR grows with α while FNR stays low with fixed β, and vice versa. This trend clearly shows the direct relationship between the threshold-closeness parameters and the alarm accuracy; increasing α (or β) leads to a higher false (or miss) alarm rate but a constant miss (or false) alarm rate. Thus, although we can achieve high compression gain of 85% with α=0.08 and β=0.01 (as an example in Figure 11), it results in an unacceptably high FPR of 35% (in Figure 12). Fortunately,

KPI name	Comp.Gain	FPR	FNR
DL-throughput	75.2	15.6	9.4
Packet-loss	84.0	10.5	4.3
RTT	82.5	9.1	8.8
CPU-load	65.1	17.4	12.8
Cell-user-count	71.3	16.8	11.9
Iub-throughput	73.9	15.1	8.8
MAC-throughput	74.6	14.7	11.5
Accessibility	83.0	13.6	7.1
Retainability	81.6	13.4	8.5
Call-drop-rate	80.3	12.8	7.3

TABLE VII
VALIDATION RESULTS BY APPLYING THE COMPRESSED THRESHOLD SETTINGS (DERIVED FROM THE TRAINING DATA) TO REAL DATA ON VARIOUS KPIS. EACH ENTRY REPRESENTS A PERCENTAGE

the alarm quality trend in the figure gives us a clear idea of how α and β should be chosen according to our target alarm accuracy. By setting α=0.03 and β=0.04, we can meet the target FPR (<15%) and FNR (<10%), which turns out to lead to compression gain of 82%.

B. Summary

Our evaluation results demonstrate that our threshold-compression solution scales very well with a large number of NEs, and delivers good compression performance (i.e., the small threshold setting) with little loss of alarm accuracy. More specifically, most KPIs from our two-month training dataset achieve nearly 80–90% compression gain under the desired false and miss alarm rates within 15% and 10%, respectively.

VII. OPERATIONAL EXPERIENCE

In this section, we present our experiences in applying the threshold-compression solution on the real data collected from the operational 3G network over a two-month period, from August 16, 2010 to October 15, 2010. We directly apply the compressed threshold settings derived from the past two-month *training* dataset to the *real* KPI measurements of the corresponding NEs and hours. The goals of this validation are two-fold: (i) see whether the pre-generated compressed thresholds still retain the desired alarm quality when applied to the real system, and (ii) demonstrate the robustness of the solution by examining the grouping consistency between two datasets.

A. Alarm Accuracy

Table VII shows the alarm accuracy results (in terms of both false and miss alarm rates) of various KPIs when we employ the compressed threshold settings generated from the training data for monitoring the real measurement data. For each KPI, we use the threshold setting that satisfies our target accuracy (i.e., FPR and FNR within 15% and 10% respectively) in the training data, and its compression gain is also presented in the table. We compare these threshold-compression alarm results against the per-NE-hourly alarm statistics in the real measurement period.

The results are quite encouraging. We see that the resulting FPR and FNR are within (or very close to) the range of our

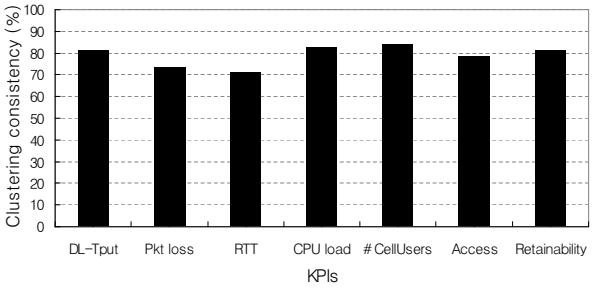


Fig. 13. Spatial-temporal clustering consistency between the training data and the monitoring data on various KPIs. Each result is obtained using the same parameters $\alpha = \beta = 0.01$.

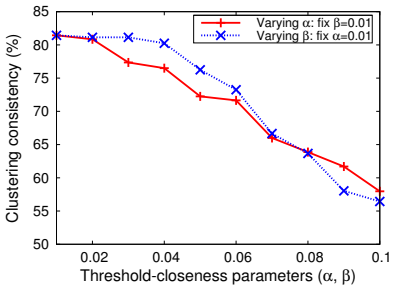


Fig. 14. Clustering consistency trend by input parameters α and β on DL-throughput KPI.

desired alarm accuracy for all KPIs, and we also obtain the similar results for other KPIs. This good alarming performance can be explained by two reasons: (i) *reflection of historical trends*: The compressed thresholds are generated from the original spatial-temporal thresholds, which are initially derived based on the historical data (via more sophisticated procedures described in Section II) to capture the normal variations on specific locations and hours. In this study we train our compression thresholds using two-month historical data that is long enough to encode such representative trends, enabling our solution to perform reasonably well in monitoring other time period; (ii) *stable clustering trend over time*: Although we use the grouping results of the training period, the members in each identified cluster still share the similar behavior in other time period as well. This is because each individual NE has its own spatial and temporal behavior as shown in Section IV. We verify such trend by showing that our threshold-compression algorithm is likely to produce the stable spatial-temporal groups, rather than arbitrary grouping over time as we present below.

B. Robustness

We examine how our solution performs on different time periods. We apply the threshold-compression algorithm on both the training data and the monitoring data, and compare their clustering results. Figure 13 shows the consistency results between two datasets on various KPIs. Here, the clustering consistency is defined by the ratio of the number of members that belong to the same spatial-temporal clusters in both datasets to the total member count in the data. We use the same parameters for comparison, and the results from $\alpha = \beta = 0.01$ are presented in the table. We observe that the grouping results

are quite stable. All KPIs show above 70% consistency in this particular setting. This stablility results verify the first point we stated in the previous subsection. Setting $\alpha = \beta = 0.01$ makes the permissible intervals close to the original thresholds, and therefore the clustering consistency results mainly depend on the threshold similarity between two different time periods. Note that both grouping results are from two-month datasets so that they can provide the representative thresholding trends on specific locations and hours.

We further investigate how the consistency changes by relaxing the threshold closeness constraint. Figure 14 plots the consistency trend on DL-throuhput KPI with varying the parameters. We see that the threshold-compression algorithm starts to produce somewhat different clustering results as we relax the grouping rule. We expect such behavior, since extending the permissible intervals tends to allow two (or multiple) members each with a quite different threshold to form a cluster as long as their intervals intersect. However, the results are still promising. A closer look at the figure shows that the clustering consistency remains still high up to a certain point, e.g., higher than 70% until α or β reaches 0.05. This result indicates that our solution indeed generates the stable grouping under the desired alarm quality. Recall that we set α=0.03 and β=0.04 to meet the acceptable target FPR ($<$15%) and FNR ($<$10%), which also gives good compression performance nearly 80%. We find that such case is also applied to most KPIs.

The stable grouping results above verify the second point we stated in the previous subsection, and explain the good alarm accuracy results when we apply the pre-generated compressed thresholds to the actual monitoring data. The clustering consistency results also confirm the similarity observations we made in Section IV. The grouping stability results can be interpreted as follows. The similar behaviors across locations are consistent over time. That is, the members in each identified cluster indeed behave very closely one another across time, just like one single entity, which is the key idea of our threshold-compression solution.

VIII. Related Work

There have been quite a few studies on the 3G networks. Most of them focus on the performance measurements of current 3G networks [2], [4], [9], whereas only few studies focus on monitoring a large-scale 3G network. Ricciato *et al.* [11] study the bottleneck detection via TCP monitoring in the UMTS core network. They use TCP parameters (e.g., RTT and retransmissions) as a set of bottleneck indicators. Khanafer *et al.* [5] present an automated troubleshooting by adopting the Bayesian model for UMTS network diagnosis. Our work differs from the previous approaches in that (i) we explicitly consider the 3G network specific spatial and temporal dynamics, (ii) show that such dynamics make the direct usage of the thresholds increasingly difficult to manage a large number of NEs, and (iii) validate our scalable thresholding approach in an operational 3G network.

The work by Laiho *et al.* [7] is probably the most closely related to our study in the sense that they focus on simplifying network analysis via visualizing similarly behaving cells. However, their grouping has different meaning from ours, as they group the cells based on their instantaneous performance. Such time-varying grouping results are hardly interpreted as representative clusters in the network. Moreover, they employ the conventional k-means algorithm [6], which is effective for large-scale data clustering in general. However, k-means algorithm performs very poor in our context. We indeed applied this algorithm as the initial candidate algorithm to our problem, but it leads to unacceptably poor alarm quality. This is because k-means algorithm cannot control the resulting false/miss alarm rates in the course of clustering process while our threshold-compression algorithm does.

IX. Conclusion

Threshold-based performance monitoring in large 3G networks is very challenging due to its strong dynamics in both time and spatial domains. There exists a fundamental tradeoff between the size of threshold settings and the alarm quality. Motivated by key observations of spatial-temporal threshold similarity, we have proposed a scalable monitoring solution, called threshold-compression that can characterize the location- and time-specific threshold trend of each individual NE with minimal threshold setting. Our experience with applying our threshold-compression solution in the operational 3G network monitoring has been very positive, and demonstrated the effectiveness of the proposed approach, e.g., threshold setting reduction up to 90% with less than 10% false/miss alarm rates.

References

[1] E. S .Gardner. Exponential smoothing: the state of the art. *Journal of Forecasting*, 4(1), 1-28, 1985.

[2] A. Gerber et al. Estimating achievable download speed from passive measurements. *ACM IMC*, 2010.

[3] U. Gupta, D. Lee, and Y. Leung. Efficient algorithms for interval graphs and circular-arc graphs. *Networks*, 1982.

[4] K. Jang et al. 3G wireless network performance measured from moving cars and high-speed trains. *ACM MICNET*, 2009.

[5] R. Khanafer et al. Automated diagnosis for UMTS networks using Bayesian network approach. *IEEE TVT*, 2008

[6] D. Arthur and S. Vassilvitskii. k-means++: the advantages of careful seeding. *ACM SODA*, 2007

[7] J. Laiho et al. Advanced analysis methods for 3G cellular networks. *IEEE TWC*, 2005.

[8] S.-B. Lee el al. Scalable monitoring via threshold compression in a large operational 3G network. *AT&T Technical Report*, 2011.

[9] X. Liu et al. Experiences in a 3G network: interplay between the wireless channel and applications. *ACM MOBICOM*, 2008.

[10] S. Makridakis et al. Forecasting: Methods and Applications. John Wiley & Sons, 1998.

[11] F. Ricciato et al. Bottleneck detection in umts via TCP passive monitoring: a real case. *ACM CoNEXT*, 2005.

[12] D. Welsh and M. Powell. An upper bound for the chromatic number of a graph and its application to timetabling problems. *Computer Journal*, 85–86, 1967.

[13] D. Wheeler and D. Chambers. Understanding statistical process control. SPC press, 1992.

[14] P. R. Winters. Forecasting sales by exponentially weighted moving averages. *Management Science*, 6:324-342, 1960.