Spatio-temporal Patterns in Network Events

Ting Wang[†] Mudhakar Srivatsa[‡] Dakshi Agrawal[‡] Ling Liu[†]
[†]School of Computer Science, Georgia Institute of Technology
[‡]IBM T.J. Watson Research Center
[†]{twang, lingliu}@cc.gatech.edu [‡]{msrivats, agrawal}@us.ibm.com

ABSTRACT

Operational networks typically generate massive monitoring data that consist of local (in both space and time) observations of the status of the networks. It is often hypothesized that such data exhibit both spatial and temporal correlation based on the underlying network topology and time of occurrence; identifying such correlation patterns offers valuable insights into global network phenomena (e.g., fault cascading in communication networks). In this paper we introduce a new class of models suitable for learning, indexing, and identifying spatio-temporal patterns in network monitoring data. We exemplify our techniques with the application of fault diagnosis in enterprise networks. We show how it can help network management systems (NMSes) to efficiently detect and localize potential faults (e.g., failure of routing protocols or network equipments) by analyzing massive operational event streams (e.g., alerts, alarms, and metrics). We provide results from extensive experimental studies over real network event and topology datasets to explore the efficacy of our solution.

1. INTRODUCTION

A network, in its simplest form, can be modeled as a graph wherein nodes represent *network entities* and edges represent their *pairwise interactions*. It is known that simple, local interactions between network entities can give rise to complex, global network phenomena [5] (e.g., fault cascading in communication networks). Nevertheless, understanding and modeling such global phenomena based on local observations remains one key challenge in network science.

An operational network typically generates various monitoring data that essentially consist of local (in both space and time) observations on the state of dispersed network entities. Examples include SNMP (Simple Network Management Protocol) and syslog messages (e.g., ping failure, interface down, high CPU usage) in communication networks, and resource utilization alarms, SLA (Service Level Agreement) violations and threshold crossing alerts in datacenter networks.

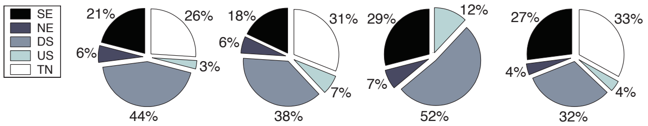

Figure 1: Percentage of fault-triggered events reported at faulty nodes (SE) and nodes with specific relationships to faulty ones (NE, DS, US, TN) in four real enterprise networks, with size of 2,514 nodes, 3,200 nodes, 141 nodes, and 12,444 nodes, respectively.

Such local observations (henceforth, called as *events*) provide a window into understanding the global network phenomena. However, the analysis of the raw monitoring data is inherently difficult due to its (1) *incompleteness*, e.g., important symptom events of a network fault may be missing due to incomplete monitoring or loss in transit; (2) *imprecision*, e.g., the recorded timestamps of individual events may be erroneous with respect to systemic causality relationships; and (3) *massive volume*, e.g., a single network fault often results in a burst of messages sent to the network management system (NMS) from the affected entities. Therefore, it can offer significant performance benefits to *succinctly and scalably identify relevant events occurring across the entire network*. Figure 1 motivates this with a concrete example from fault diagnosis in enterprise networks, using event data collected from four real enterprise networks. The fractions of relevant events (that correspond to a same fault) reported at the faulty node itself (SE), and nodes with specific relationships (neighboring - NE, down-streaming - DS, up-streaming - US, and tunneling - TN) to the faulty node are shown. Observe that those topologically related nodes account for an overwhelming 71-82% of the events.

It is often hypothesized that relevant network events exhibit both spatial and temporal correlation based on the underlying network topology and their time of occurrence, although the correlation extent may depend

ACM CoNEXT 2010, November 30 – December 3 2010, Philadelphia, USA.

on the accuracy of monitoring system. It is worth emphasizing that in complex, multi-layer networks (e.g., enterprise networks as targeted in this work), the network topology includes both *horizontal* (e.g., that between BGP peers) and *vertical* (e.g., that between layer L_1 and L_3 counterparts) relationships. We argue that identifying and leveraging such spatio-temporal patterns to correlate relevant events may offer both scalable (efficient in facing massive event volumes) and robust (resilient to incomplete and imprecise monitoring data) primitives for network-event analysis.

In this paper, we propose a new class of models suitable for learning, indexing and matching spatio-temporal patterns in network-event data. We exemplify our techniques with the application of detecting and localizing potential faults in enterprise networks. Conventional solutions, as adopted by most widely deployed NMSes in such networks, maintain a cache of "unresolved" events, and use rule or codebook based mechanisms (e.g., [23, 25]) to correlate each new event with all cached events to suppress dependent events and retain only the (unfixed) root-cause events in the cache. These approaches however suffer from both scalability (e.g., the computation complexity is quadratic in terms of the network size) and robustness (e.g., missing important symptom or root-cause events may result in a large number of unresolved events in the cache) issues.

We observe: (i) Events triggered by a fault are typically generated by a small, approximately constant subset[1] of network entities that are topologically related to the faulty entity within a limited time window; thus, for each new event arriving at the NMS, only those (potentially) topologically and temporally relevant events need to be considered. (ii) By aggregating a set of correlated events, one may be able to infer the root cause with high confidence, despite the possible loss of important symptom or root-cause events, and imprecision in individual events. To exploit these observations, we propose a new class of indexable *network signatures* that encode the temporal patterns of events as well as the topological relationships among the entities where these events occur. We present efficient learning algorithms to extract such signatures from noisy historical event data. With the help of novel space-time indexing structures, we show how to perform efficient online signature matching. We entitle the complete framework TAR (topologically-aware reasoning) that, to the best of our knowledge, is the first proposal to utilize topologically-aware event patterns to perform scalable, yet robust network root cause reasoning.

We implemented our solution on a large-scale testbed NMS, and empirically evaluated its efficacy in terms of diagnosis accuracy, scalability, predictive power, and error tolerance, demonstrating significant improvement over its previous counterpart techniques (e.g., reduce fault diagnosis time from 45 mins to 37 seconds for an event burst of size 15K).

2. PROBLEM CONTEXT AND SCOPE

This section describes a fundamental yet challenging task facing network operators when analyzing network monitoring data in enterprise networks, namely, detecting and localizing potential network faults, which motivates our study on spatio-temporal patterns.

2.1 Fault Diagnosis in Enterprise Networks

We target large-scale enterprise networks, which usually involve hundreds of thousands of network entities (e.g., routers), and are typically managed by a centralized management system that collects (local) observations from network entities at different layers of the protocol stack. The aim of fault diagnosis is to quickly detect potential faults (e.g., failures of routing protocols or network equipments) for a given set of symptom events, and localize the possible network entities responsible for the faults, such that corrective measures can be directed at the root cause, as opposed to merely addressing the immediate symptoms.

Even though operators today have a myriad network monitoring data at their disposal, it is still non-trivial to efficiently identify and extract root-cause events from massive event storms, attributed to the inherent incompleteness and imprecision of the monitoring system, and the cascading nature of network failures, namely, failure of a single network entity triggers a large burst of events from affected entities all over the network (see example shown in Figure 1). For instance, large networks in 2007 are faced with the challenge of monitoring over 100,000 network entities and cope with event bursts of over 10,000 events per second [17]. Evidently, *scalability* and *robustness* have become two key issues. The pairwise comparison approaches adopted by most widely deployed NMSes in enterprise networks, as we have discussed in Section 1, suffer from both scalability and robustness issues.

2.2 TAR: Topologically-Aware Reasoning

It is observed, however, that two events may be correlated only if the underlying nodes (at which these events were triggered) are topologically related.

Example 1. In the network shown in Figure 2, the failure of router n_6 may trigger events at both n_4 and n_7 since they establish a tunnel containing n_6.

Hence, an incoming event needs to be correlated with only a subset of events that occurred at topologically related network entities, rather than on the scale of the entire network. Also, the time of their occurrences tends

[1] The size of this subset depends on factors such as degree distribution of the network, and is independent of the size of the entire network.

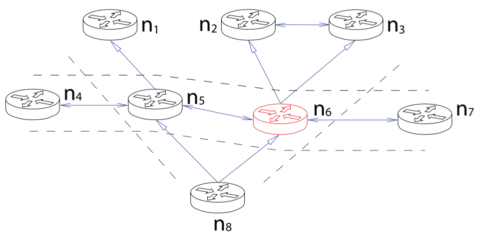

Figure 2: Correlation of topologically relevant events for root-cause analysis.

to follow the order of fault cascading, within a limited time window. We hypothesize that a network fault may be characterized by the spatio-temporal patterns of its triggered events; leveraging such patterns facilitates efficient diagnosis. Further, the imprecision and incompleteness of individual events may be overcome by aggregating correlated events, which capacitates us to infer or even predict network faults with high confidence, in facing of incomplete or imprecise information.

Exploiting such topological and temporal correlation, however, requires addressing a set of non-trivial challenges: First, how to select a set of patterns that best capture the essential spatio-temporal correlation in fault-triggered network events, from the myriad semantic structures, especially for complex, multi-layer enterprise networks? Second, how to concisely represent such spatio-temporal patterns? Third, how to efficiently extract (parameterize) the patterns from noisy historical data? Finally, how to support scalable online pattern matching against high-volume event streams?

This paper presents the design, implementation and evaluation of Tar, the first known complete framework for extracting, indexing and identifying spatio-temporal patterns in network monitoring data. The cornerstone of Tar is a class of indexable *network signatures* that concisely encode the temporal and topological patterns of events triggered by a root cause. Tar includes solutions to automatically extract such signatures from historical datasets and to efficiently match such signatures against high-volume network-event streams.

3. DESIGN OF TAR

This section introduces the basic concepts and notations used throughout the paper, then describes the design of network signature, fundamental to Tar, and finally gives an overview of the architecture of Tar.

For simplicity of exposition, we start with single-layer networks (e.g., L_3 layer network), and will discuss the extension to multi-layer networks in Section 4.4. We have the following assumptions. The **network** is modeled as a graph with each node representing a manageable logical entity ("entity" for short) uniting one or more physical devices (e.g., routers), and each edge corresponding to a network link (e.g., BGP peering). Also we assume the network configurations to be static. The network management system (NMS) consists of a set of monitoring **agents** and a **sink**. Deployed at dispersed entities, the agents collect and send monitored **events** (alarms, performance, alerts) to the sink that is responsible for root cause analysis. Each event is a tuple of the form $\langle v, e, t\rangle$, where v represents the network node generating the event, e the event type, and t the timestamp of the event. Events come in as an online **stream**. The goal of Tar can be summarized as: *by analyzing the event stream and exploiting the network topology information, detect the potential faults and localize their root-cause entities in real-time.*

relationship	description
selfing	u and itself
neighboring	u and v are directly connected
containing	u contains v as a sub-component (e.g., a router and its interfaces)
down-streaming	u is at v's down-stream side (route from sink to u contains v)
tunneling	u is on a tunnel (a special type of network connection) with v as one end

Table 1: Topological relationships and descriptions (u and v as two network entities under consideration).

3.1 Network Signature

The concept of *network signature* is central to Tar. In designing network signature, we strive to achieve a set of objectives: *expressive* - it should be semantically rich, capturing both topological and temporal features of correlated events; *compact* - yet, it should be structurally simple, thus easy to be matched against incoming events; and *indexable* - it should be amenable to indexing, thus can be encoded in space-efficient structures for online matching.

Intuitively, we construct network signature based on the following two fundamental observations. First, when a fault occurs at an entity u, correlated events are typically triggered at affected entities that are topologically related to u. Second, the triggered event at an affected entity v depends on the topological relationship between u and v, in terms of its event type and time delay.

Example 2. In Figure 2, the failure of n_6 may lead to the event of "*OSPF neighbor down*" at n_3 since n_3 is a direct neighbor of n_6, while n_4 may observe the event of "*failed connection attempt*" since the tunnel between n_4 and n_7 involves n_6.

We considers a set of relationships $\mathcal{R} = \{$*selfing, neighboring, down/up-streaming, containing/contained, tunneling*$\}$, with brief descriptions listed in Table 1. Note that the relationship *down/up-streaming* is referred from the view of sink, i.e., u is at v's down-stream side if the route from the sink to u contains v. We refer to the set of entities with a specific relationship r to v as a **topo-set**, denoted by $\mathcal{N}(v, r)$. Each $r \in \mathcal{R}$ (except *selfing*) is also associated with a *reverse* relationship, e.g., down-streaming to up-streaming, denoted by $\bar{r}$.

Note that in general when a fault occurs, a sequence

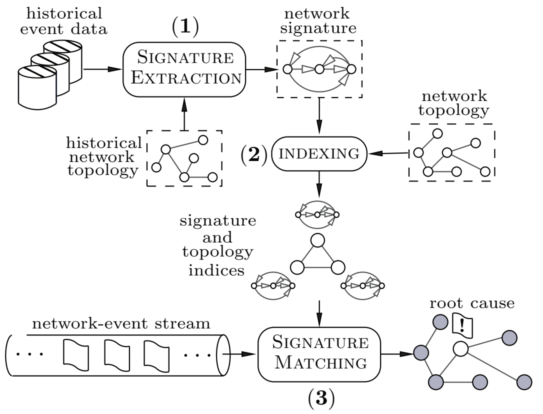

Figure 3: Main architecture of Tar.

of events could be triggered at a specific entity, which tend to follow certain statistical models (e.g., Markov chain [18, 19]), which could be manually encoded based on expert domain knowledge, or learned using temporal data mining techniques (e.g., [21]), with concrete techniques orthogonal to the scope of this work; in this paper, we consider such event sequences as *meta-events*, encoded in the set $\mathcal{E}$ of known event types.

We consider finite sets of faults $\mathcal{F}$ (that may be unknown, and the concrete faults considered in our implementation is discussed in detail in Section 5), event types $\mathcal{E}$, and topological relationships $\mathcal{R}$. We define network signature as below.

Definition 1 (Network Signature). *For a specific fault type $f \in \mathcal{F}$, its signature $\mathsf{sig}(f)$ is a series of tuples $\langle e, r, t, \mathsf{prob}(e, r, t|f)\rangle$, where $e \in \mathcal{E}$, $r \in \mathcal{R}$, t is a discretized time measure, and $\mathsf{prob}(e, r, t|f)$ denotes the probability of observing an event e at an affected node with topological relationship r to the faulty node where f occurs, with time delay t.*

fault type	relationship	event
interface down	tunneling	failed connection attempt
	neighboring	OSPF neighbor down
	· · ·	· · ·

Table 2: Signature example.

Example 3. Recall Example 2. We may have the signature for the fault "interface down" as in Table 2.

This definition is structurally intuitive to interpret in that it simply encodes the association between a fault and a set of symptoms; it is yet semantically rich in that the encoded association includes both the temporal correlations of symptom events, and the topological relationships among the entities that generate them.

3.2 High-Level Design

Conceptually, Tar entails three major phases (see Figure 3): (1) *signature extraction*, abstracting network signatures from the (possibly noisy) historical event data and associated underlying topological information; (2) *indexing*, organizing the network signatures and the topological information of targeted network into space efficient indexing structures; and (3) *signature matching*, performing root cause analysis by matching indexed signatures against incoming event streams.

Signature extraction. The main input to this phase is the (typically noisy) historical network-event data and the associated network topology information, which is expected to be comprehensive to cover a large number of combinations of the variables (e.g., event type, topological relationship, time delay). Most network monitoring infrastructures today can typically provide such datasets [2, 1]. By applying unsupervised learning techniques (details in Section 4.1), Tar extracts the statistical relationships among the set of relevant variables, and encode them into a set of network signatures.

Indexing. Leveraging the learned network signatures requires "compressing" them into indexing structures so as to support fast lookup queries over spatio-temporal patterns. Specifically, in the application of fault diagnosis, one needs to efficiently correlate symptom events possibly caused by a common fault, a symptom-event-to-root-cause lookup structure that encodes *inverted signatures* is desired. Meanwhile, to determine the topological correlation amongst a set of events, the underlying network topology needs to be frequently queried. Tar employs a set of novel topology indexing structures that support efficient spatial intersection queries (details in Section 4.2).

Signature matching. Equipped with network signatures and topology indices, one is now able to perform scalable signature matching against high-volume monitoring event streams. Intuitively, the matching process correlates topologically and temporally relevant events, and reasons about the underlying root cause. We propose a novel *signature matching tree* structure to enable fast fault localization (details in Section 4.3).

4. IMPLEMENTATION

In this section, we describe in detail the implementation issues of our signature-based fault diagnosis solution. Further, we discuss how to extend our model to the case of multi-layer networks.

4.1 Signature Extraction

Before performing signature extraction, it is necessary to prepare the training data from typically noisy archive data. Here "noise" indicates events irrelevant to faults (e.g., heartbeat messages) or events with imprecise timestamps. We achieve this in two steps: (1) aggregate events into *meta-events* (a sequence of events at a specific network entity caused by a single root cause) by applying the temporal mining algorithm in [21], and (2) separate meta-events caused by faults from those triggered by regular network operations, based on their occurrence *frequency* and *periodicity*. Intuitively, events with extremely low frequency are usually the noise components while over-frequent or periodical events typically correspond to regular network operations.

Let $\mathcal{B}$ denote the resulting training data. For clarity of presentation, we assume that each event[2] in $\mathcal{B}$ is stored in the form (v, e_v, t_v) where v is the network entity generating the event, $e_v \in \mathcal{E}$ the event type, and t_v the timestamp of its occurrence.

The process of signature learning is divided into two main phases, *partitioning* and *extracting*. In partitioning, we select a subset of the archive $\mathcal{B}$, $\mathcal{B}'$, such that the events in $\mathcal{B}'$ are organized into disjoint partitions $\{B\}$, each corresponding to a common root cause with high probability. We discard the rest events $\mathcal{B} \setminus \mathcal{B}'$ as noise. In extracting, we summarize from each partition B a candidate signature P, and cluster the set of candidate signatures $\{P\}$ into the final set of signatures. For clarity, we conceive a candidate signature P as 3-dimensional matrix wherein each element $P(e, r, t)$ represents the likelihood of observing event e at node with relationship r to the faulty node with time delay t.

We assume a time window ω that specifies the maximum delay between observing the first and last events triggered by a single fault. Also all the time measures have been discretized, and all the timestamps have been normalized relative to the corresponding time windows.

```
Algorithm 1: SIGNATURE EXTRACTION
   Input: event archive $\mathcal{B}$, maximum window size $\omega$
   Output: set of network signatures
   // partitioning
1  $\mathcal{B}' \leftarrow \emptyset$;
2  for each subset $B$ of $\mathcal{B}$ within $\omega$ do
3      if $\mathsf{cohe}(B) > \lambda$ then continue;
4      if $B$ overlaps with $B' \in \mathcal{B}'$ then
5          if $\mathsf{cohe}(B) \geq \mathsf{cohe}(B')$ then continue;
6          $\mathcal{B}' \leftarrow \mathcal{B}' \setminus \{B'\} \cup \{B\}$;
   // extracting
7  for each partition $B \in \mathcal{B}'$ do
8      $\{r_v^*\}_{(v,e_v) \in B} = \arg\min_{r_v} |\bigcap_{(v,e_v) \in B} \mathcal{N}(v, r_v)|$;
9      for each $r \in \mathcal{R}$, $e \in \mathcal{E}$, and $t \in \omega$ do
10         $P(e, r, t) = \frac{\sum_{(v,e_v,t_v) \in B} \mathbf{1}(r_v^* = \bar{r}, e_v = e, t_v = t)}{\sum_{(v',e_{v'},t_{v'}) \in B} \mathbf{1}(r_{v'}^* = \bar{r})}$;
   // determining number of faults
11 $|\mathcal{F}| = \arg\min_{|\mathcal{F}|} 2|\theta| - 2\log[\mathsf{like}(\{P\}|\theta)]$;
12 apply $K$-means ($K = |\mathcal{F}|$) clustering to $\{P\}$;
13 set the cluster centers as fault signatures;
```

Algorithm 1 sketches our learning algorithm.

(i) In the event archive $\mathcal{B}$, we examine the events within each time-window ω as a partition B, i.e., the events in the same partition are possibly triggered by a single fault. We intend to narrow down the set of candidate faulty nodes. We achieve this by leveraging the following observation: for each node v appearing in an event $(v, e_v) \in B$, the faulty node must lie in one topo-set of v, $\mathcal{N}(v, r_v)$ (with topological relationship r_v); hence, it must also appear in the non-empty intersection of such topo-sets $\{\mathcal{N}(v, r_v)\}$. We consider all such non-empty intersections, and pick the minimum one as the candidate set. The intuition behind is the *minimum explanation principle*: the smallest candidate set is considered to provide the best explanation about the fault. We can then measure the quality of B using the size of this minimum candidate set, as formalized in the metric of *coherence*:

$$\begin{aligned} \mathsf{cohe}(B) &= \min_{r_v} |\cap_{(v,e_v) \in B} \mathcal{N}(v, r_v)| \\ \text{s.t.} &\quad \cap_{(v,e_v) \in B} \mathcal{N}(v, r_v) \neq \emptyset \end{aligned}$$

We discard those partitions with coherence above a threshold λ. If two selected partions overlap, we pick the one with better quality (line 2-6).

(ii) In each selected partition B, for each involved node v, one identifies the topological relationship r_v^* that leads to the minimum non-empty candidate set. All the events $\{\langle v, r_v^*, e_v, t_v \rangle\}$ in B are then used to compute a potential signature P (line 8-10). Note that here we simply use the frequency of tuples to compute P, while a prior distribution can be readily incorporated.

(iii) The number of fault types $|\mathcal{F}|$ (which is assumed unknown) essentially controls the complexity of the statistical model. We apply the *Akaike's information criterion* [3] to select $|\mathcal{F}|$.

Specifically, we assume that the candidate signatures corresponding to a common fault follow a Gaussian distribution. The information criterion of a model is given by: $\mathsf{aic}(\theta) = 2|\theta|$ - $2\log[\mathsf{like}(\{P\}|\theta)]$, where $|\theta|$ is the number of parameters to be estimated which captures its complexity (i.e., expressiveness), and $\mathsf{like}(\{P\}|\theta)$ is the likelihood of observing the set of candidate signatures under the model (i.e., fitting to the data). A trade-off is made between these two terms. The setting of $|\mathcal{F}|$ leading to a minimum $\mathsf{aic}(\theta)$ is considered optimal.

We apply K-means (with $K = |\mathcal{F}|$) clustering algorithm to the set $\{P\}$; the centers of the clusters are regarded as the signatures for the $|\mathcal{F}|$ faults (line 11-13). Due to its sensitivity to the initial clustering centers, we run the clustering algorithm multiple times with randomly selected centers, and average over the results.

4.2 Indexing

TAR attempts to detect and localize faults as follows. At each affected entity v observing event e at time t, network-signatures are used to compute the probability $\mathsf{prob}(f, \bar{r}|e)$ ($\bar{r}$ is the inverse relationship of r) that the faulty entity incurred the fault f and has a topological relationship $\bar{r}$ to v. If this probability is greater than a system threshold, an *evidence* $\langle f, v, \bar{r}, t \rangle$ is formed, indicating with high confidence that a faulty entity exists among the set of entities with relationship $\bar{r}$ to v. All collected evidences within a time-window are correlated

[2] In following, we assume that the meta-events are encode in the set of known event types $\mathcal{E}$, and do not further distinguish the terms "event" and "meta-event".

to gradually narrow down the candidate faulty entity set and the fault f.

Example 4. Recall Example 2. Assume in Figure 2 n_3 observes event "OSPF neighbor down" while n_4 observes event "failed connection attempt". According to the signature in Table 2, one can infer that the faulty node lies in the topo-set with neighboring relationship to n_3 and in the topo-set with tunneling relationship to n_4, which uncovers n_6 as the faulty node.

To facilitate signature-based fault diagnosis, Tar employs two indexing structures: index of signatures for evidence computation, and index of network topological dependency for evidence correlation.

Signature Index We propose an inverted signature structure $\mathcal{I}_s$ to support fast symptom-root-cause lookup, by maintaining the association between symptom events and possible faults. Recall that the signature of a fault f is a series of tuples $\langle e, r, t, \mathsf{prob}(e, r, t|f)\rangle$, where e, r, and t represent a symptom event, a topological relationship, and time delay, respectively, and $\mathsf{prob}(e, r, t|f)$ is the probability distribution of observing e at an entity with relationship r to the faulty entity with delay t. Here we temporarily ignore t, and adopt a marginal version $\mathsf{prob}(e, r|f)$, because the underlying root cause is unknown, and the information of absolute time delay is missing. We will later use the relative time difference between symptom events to infer potential fault.

Corresponding to each signature, we create an entry in the index $\mathcal{I}_s$: $\langle f, \bar{r}, \mathsf{prob}(f, \bar{r}|e)\rangle$, where $\bar{r}$ is the inverse relationship of r, and $\mathsf{prob}(f, \bar{r}|e)$ is the posterior probability that f occurs at an entity with topological relationship $\bar{r}$ to a given entity observing e. Its computation is given by:

$$\mathsf{prob}(f, \bar{r}|e) = \frac{\mathsf{prob}(e, r|f) \cdot \mathsf{prob}(f)}{\sum_{f' \in \mathcal{F}} \sum_{r' \in \mathcal{R}} \mathsf{prob}(e, r'|f') \cdot \mathsf{prob}(f')}$$

where the prior probability of the occurrence of fault f, $\mathsf{prob}(f)$, can be derived from the overall statistics of historical event data. At an entity v observing event e, for each fault $f \in \mathcal{F}$, we select the set of topological relationships $\mathcal{R}_v$ that satisfy $\mathsf{prob}(f, \bar{r}|e)$ above system threshold for $\bar{r} \in \mathcal{R}_v$. We term such a tuple $\langle f, v, t, \mathcal{R}_v\rangle$ as an **evidence**.

Topology Index The incorporation of network topological information significantly boosts the precision of fault diagnosis, by correlating events according to their underlying topological relationships. Such improvement, however, requires space-efficient indexing structures that support fast retrieval of topological relationships among network entities.

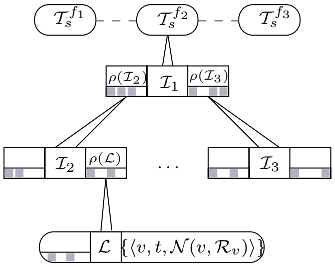

Figure 4: Signature matching tree $\mathcal{T}_s$.

As will be shown shortly, a key operation heavily involved in fault localization is computing the intersection of two topo-sets, e.g., joining the down-streaming neighbors of one entity and the direct neighbors of another; hence, for each indexing structure, we are particularly interested in minimizing the cost of retrieving (constructing) a topo-set from it. Due to space limitations, we focus our discussion on building indices for up/down-streaming relationships.

We construct our index based on the following two observations: (1) the shortest path routes from the sink to all the entities form a spanning tree rooted at the sink, i.e., a tree cover of the network; (2) the diameter ϕ of a typical management domain (as observed in four large enterprise networks shown in Figure 1) is about 3-7 hops. Therefore, the set of up-streaming neighbors (utmost ϕ) of an entity can be directly cached. We then traverse the routing tree in level order (breadth-first), assigning each entity a traversal-order number. The down-streaming neighbors of a given entity u can be summarized as ϕ intervals, $\{[l_i, r_i]\}_{i=1}^{\phi}$, where l_i (r_i) denotes the order number of its left-most (right-most) descendent on the ith level below u. Clearly, this structure achieves retrieval cost of $O(\phi)$, since the neighbors on the same level can be retrieved in one consecutive chunk, at the storage cost of $O(\phi)$ for each entity.

4.3 Signature Matching

With the help of topology indices, in each evidence $\langle f, v, t, \mathcal{R}_v\rangle$, $(v, \mathcal{R}_v)$ can be replaced with the union of the corresponding topo-sets $\bigcup_{r \in \mathcal{R}_v} \mathcal{N}(v, r)$ ($\mathcal{N}(v, \mathcal{R}_v)$ for short). We consider two evidences $\langle f, t, \mathcal{N}(u, \mathcal{R}_u)\rangle$ and $\langle f', t', \mathcal{N}(v, \mathcal{R}_v)\rangle$ (possibly) *correlated* if (1) $f = f'$, (2) the time of their occurrences is within a short window $|t - t'| \leq \omega$, and (3) $\mathcal{N}(u, \mathcal{R}_u) \cap \mathcal{N}(v, \mathcal{R}_v) \neq \emptyset$. This concept can be generalized to multiple evidences.

While checking conditions (1) and (2) is fairly straightforward, computing the intersection of $\mathcal{N}(u, \mathcal{R}_u)$ and $\mathcal{N}(v, \mathcal{R}_v)$ is costly. The complexity is $O(\min\{|\mathcal{N}(u, \mathcal{R}_u)|, |\mathcal{N}(v, \mathcal{R}_v)|\})$, even if both sets are stored in a hashing-table structure. Moreover, following the naïve pairwise comparison paradigm, each incoming evidence would be compared with all existing ones to detect relevance, and thus scales poorly with the event-stream rate.

Signature Matching Tree We devise a novel signature matching tree $\mathcal{T}_s$ that enables efficient correlation of evidences. Our design follows the one-pass clustering philosophy [11], which endows $\mathcal{T}_s$ with high throughput and scalability. Figure 4 shows the chief structure of

$\mathcal{T}_s$. It is a hierarchical structure, with the highest level containing $|\mathcal{F}|$ buckets, each corresponding to one fault type $f \in \mathcal{F}$. Within each bucket is a height-balanced tree $\mathcal{T}_s^f$, into which evidences of the form $\langle f, t, \mathcal{N}(v, \mathcal{R}_v) \rangle$ are inserted. Each leaf of $\mathcal{T}_s^f$ corresponds to a cluster of relevant evidences; and each non-leaf node represents the union of all the clusters in its subtree.

For each leaf node (cluster) $\mathcal{L}$ containing a set of evidences, we maintain the intersection of their topo-sets, called its aggregation, $\rho(\mathcal{L}) = \bigcap_{\langle f,t,\mathcal{N}(v,\mathcal{R}_v)\rangle \in \mathcal{L}} \mathcal{N}(v, \mathcal{R}_v)$, which corresponds to the candidate faulty entity set; while for each non-leaf node (super cluster) $\mathcal{I}$, we maintain the union of the aggregations of all the leaves in its subtree, $\rho(\mathcal{I}) = \bigcup_{\mathcal{L} \in \mathcal{I}} \rho(\mathcal{L})$.

The signature matching tree $\mathcal{T}_s$ supports two basic operations, *insertion* and *deletion*. In insertion, an arriving evidence $\langle f, t, \mathcal{N}(v, \mathcal{R}_v) \rangle$ recursively descends down $\mathcal{T}_s^f$ by testing $\mathcal{N}(v, f, \mathcal{R}_v) \cap \rho(\mathcal{I})$ for each non-leaf $\mathcal{I}$ encountered, until being clustered into an appropriate leaf $\mathcal{L}$ that can absorb it. If no such leaf exists, a new one is created which solely contains this evidence; it then updates the aggregations of the nodes on the path from the leaf to the root of $\mathcal{T}_s^f$. Those evidences with timestamps out of the current time window are considered as expired. In deletion, expired evidences are removed from the tree, and the aggregations of the nodes on the paths from the affected leaves to the root are updated in a bottom-up manner.

4.4 Extension

Now we briefly discuss how to extend our model to support multi-layer networks. Typically, in a complex network (e.g., enterprise networks), the monitoring data is collected at different layers within the protocol stack, e.g., data may be available from networks entities at L_1 layer and L_3 layer. Different layers tend to expose fairly different connectivity structures and relationships. It is challenging to line up such heterogenous topology in an consistent manner.

In our current implementation, we employ the concept of *composite relationship*. Specifically, for two network entities u and v at two different layers, we may consider $u \times v$ as a composite entity if there exists a mapping $u \xrightarrow{m} v$ between them, e.g., a layer L_3 entity and its layer L_1 counterpart. Now consider two relationships $u' \xrightarrow{r_1} u$ and $v \xrightarrow{r_2} v'$ with u' at the same layer as u and v' at the same layer as v. We define the composite relationship between u' and v' as $u' \xrightarrow{r_1, m, r_2} v'$. We can then apply the concept of network signature as described above.

The drawback of this approach is the increased variable space, which may impose prohibitive overhead on both learning and applying the network signature model. For two layers with relationship types $\mathcal{R}_1$ and $\mathcal{R}_2$, the cardinality of the set of composite relationships is typically $|\mathcal{R}_1||\mathcal{R}_2|$. Note that, however, the combinatorial complexity may be largely reduced by domain expertise, since many composite relationships may be invalid.

attribute	description
IPAddress	address of the network entity generating the event
PollerIPAddress	address of the poller/monitor
Event-count	sequence number of the event
generic-trap	SNMP trap ID
specific-trap	enterprise specific SNMP trap ID
Raw Capture Timestamp	timestamp of the trap message

Table 3: Attributes of network event.

5. EMPIRICAL EVALUATION

This section presents an empirical evaluation of the efficacy of Tar in network fault diagnosis. The experiments are specifically designed to center around the following metrics: (1) the descriptive power of the network signature model in capturing real network faults; (2) the effectiveness of the learning algorithm in extracting the network signatures from historical data; (3) the scalability of Tar in detecting and localizing network faults facing high-volume monitoring data streams; (4) its fault predictive power by exploiting partial/incremental fault signature matching; and (5) its robustness against missing symptom events (e.g., due to packet losses in SNMP messages transported over UDP, or incomplete monitoring caused by configuration errors, etc.) and errors in the information regarding underlying network topology (e.g., due to the staleness in the discovered network topology). We start with describing the datasets.

5.1 Dataset

Our experiments mainly used two datasets from real-life network management systems. The first dataset is an archive of 2.4 million SNMP trap messages collected from a large enterprise network (spanning 7 ASes, 32 IGP networks, 871 subnets, 1,268 VPN tunnels, 2,068 main nodes, 18,747 interfaces, and 192,000 network entities) over several weeks in 2007. Event attributes of interest to us are listed in Table 3. The second dataset is a real European backbone network consisting of 2,383 network nodes (spanning 7 countries, 11 Ases, and over 100,000 entities). Based on its topology, we generate a synthetic monitoring data stream (with tunable failure rates) to quantify the efficacy and scalability of Tar. While the real event dataset collected in 2007 indicates event bursts (events arriving within a extremely short time window) of sizes up to 12,000 events, the synthetic dataset (generated by artificially increasing failure rates) includes event bursts of sizes up to 36,000 events. Our core libraries were implemented using Java. All experiments were conducted on a Linux workstation with modest computation power, running 1.6 GHz Pentium IV and 1G memory.

5.2 Descriptive Power

trap id	description
28	bsnIpsecEspAuthFailureTrap
79	bsnAPRegulatoryDomainMismatch
102	bsnAPBigNavDosAttack
104	bsnAPContainedAsARogue
124	bsnAPIfDown
130	bsnAPInterferenceProfileFailed
230	bsnTemperatureSensorFailure
378	csiErrorTrap

Table 4: SNMP trap ids (bsn - BackwardSequenceNumber, csi - Cisco SSL VPN Client Interface).

(network entity, event) set	support
(x.y.11.3, 124) (x.y.1.10, 124)	0.87
(x.y.8.29, 124) (x.y.9.163, 230)	0.86
(x.y.8.29, 124) (x.y.9.163, 230) (x.y.15.1, 378)	0.78
(x.y.9.33, 103) (x.y.15.1, 378) (x.y.1.10, 230)	0.75
(x.y.9.163, 104) (x.y.9.163, 230) (x.y.15.1, 378) (x.y.107.1, 102)	0.61

Table 5: Topology-agnostic signatures (only masked entity IP-addresses and trap message IDs are shown).

We contrast our fault signature model against a codebook based model [25], which we refer to as the topology-agnostic model. The topology-agnostic signature for a network fault is defined as a set of tuples of the form (network entity, symptom event), where the network entity is the concrete network node generating the event and the set of symptom events are shown in Table 4. A sample of topology-agnostic signature is show in Table 5. The co-occurrence of the tuples of a signature in the event stream indicates the potential occurrence of the corresponding fault. Note that such signatures are coupled with concrete network entities, and are thus inherently deterministic in that the correlation of symptom events are explicitly encoded.

To make the comparison accurate, we apply frequent pattern mining techniques to learn both topology-aware and -agnostic signatures. From both synthetic and real event datasets, we extracted a trace of 40,000 annotated events (root-cause events are marked with special tags) with 20,000 events used for extracting network signatures and the rest for measuring the efficacy of both signature models.

In signature mining, we examine events in a narrow time window around an annotated fault event, and use the well-known *apriori* algorithm to identify event sets with high co-occurrence frequency across such time windows. This approach extracts event sets that are observed with high support when a fault occurs in the network. The topology-agnostic signature directly encodes the association between frequent symptom event sets and concrete network entities wherein these events occur; in contrast, our network signature abstracts such associations based on topological relationships among network entities, rather than concrete entities.

dataset	# signatures	time (mins)
real	44 (4,518)	3 (10)
synthetic	92 (12,645)	6 (21)

dataset	precision	recall
real	0.92 (0.82)	0.91 (0.83)
synthetic	0.88 (0.68)	0.87 (0.70)

Table 6: Comparison of topology-aware and -agnostic signature models (numbers for topology-agnostic model are within braces).

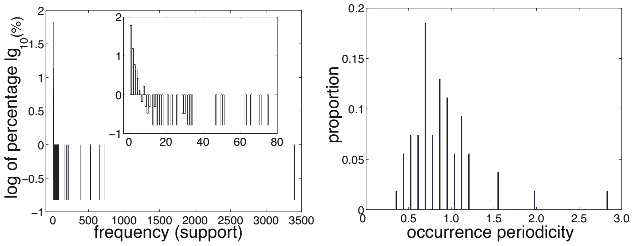

Figure 5: Distribution of frequency of meta events, and normalized histogram of periodicity of meta-events.

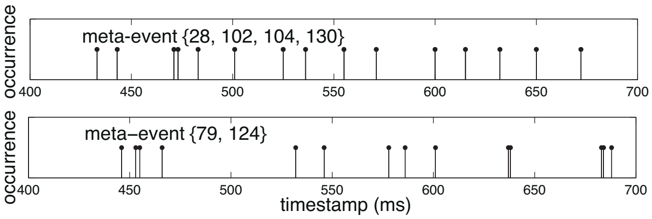

Figure 6: Occurrences of two sample meta events.

We compare these two models in terms of signature size and execution time, with results shown in the upper table of Table 6. Evidently, topology-aware model generates much more concise signatures than topology-agnostic model, with the size of signature set two orders of magnitude smaller than the latter on both real and synthetic datasets. This is mainly attributed to the nature of topology-aware signature: it may encode multiple topology-agnostic signature instances into one signature, given that they reflect the same correlation on the topological relationship level. Also, it is noticed that topology-aware signature leads to much higher efficiency of signature extraction; this is explained by that the introduction of topology-awareness significantly reduces the search space.

We further apply the extracted signatures to diagnose (classify) the faulty events in the test dataset, with accuracy (precision and recall) shown in the lower table of Table 6. Evidently, topology-aware signature demonstrates better descriptive power in capturing the essential characteristics of network faults. Note that, however, because of its strong dependency on topological information, topology-aware signature is inherently faced with the challenge of dealing with errors in the topological information (e.g., missing or stale topological information). We will discuss the robustness of TAR against such errors shortly.

5.3 Learning Effectiveness

We evaluate the effectiveness of the data preparation phase of our learning algorithm. In data preparation, we use frequency and periodicity as two criteria to distinguish fault-caused meta events from the rest. The normalized histogram of meta-events with respect to frequency (in logarithmic scale) is depicted in the left plot of Figure 5, which approximately follows a *power law* distribution. It is observe that more than 60% meta-events have fairly low frequency, e.g., below 5, which, as we confirmed by examining the definition of trap ids, are mainly caused by infrequent network operations, e.g., the event set {3} represents "the cisco NetReg server has started on the host from which this notification is sent", or certain non-recurrent faults, which are of modest interest for our purposes of leveraging existing diagnosis efforts. Meanwhile, the meta-events with significantly higher support than others are typically due to regular network operations, e.g., the event set {102, 104} which appears with support 348 indicates "data from the remote side is available for the TDM channel".

The distribution of the periodicity of meta-events is illustrated in the right plot of Figure 5. Observe that most of the meta-events demonstrate low deviation of occurrence intervals, i.e., they are resulted from normal network operations. We randomly selected two meta-events {28, 102, 104, 130} and {79, 124} with periodicity 0.43, and 1.09 (lower periodicity ⇒ more regular), respectively, and examined their implications. Figure 6 compares their occurrences. From the descriptions of the traps, it is confirmed that the meta-event {79, 124} indicates potential network faults, while the meta-event {28, 102, 104, 130} is caused by regular network operations, e.g., link mode sniffing.

We then evaluate the effectiveness of the signature extraction component using the meta-event size histogram. More specifically, after applying the unsupervised learning algorithm over the event dataset, by running Monte Carlo simulation, we derived the histogram of meta-event size from the learned signatures, and compared it against that extracted from the real data.

The upper plot of Figure 7 illustrates the comparison of these two histograms (normalization is applied). It is clear that the distribution of the model-generated data fits that of the underlying data fairly tightly. Furthermore, we analyzed the distribution of individual events for real data and model generated data, respectively. As shown in the lower plot of Figure 7, these two distributions demonstrate strong consistency, which empirically proves that our learning model can capture the essential features of the real data.

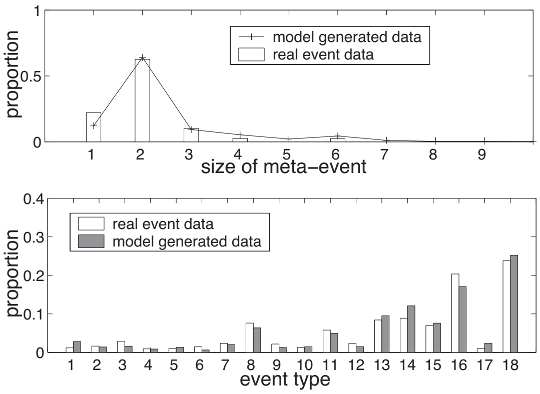

Figure 7: Histograms of size of meta events and individual events in real and model generated data.

5.4 Scalability

Next, we proceed to evaluating the scalability of Tar against the state-of-the-art approaches used by fault diagnosis engines in widely deployed NMSes. These approaches follow a pairwise event correlation paradigm: the engine maintains a cache of events seen in the recent past. A new event is compared (pairwise) against all the events in the cache to determine if it can suppress one or more dependent events in the cache or if it can be suppressed by any one in the cache; pairwise comparisons terminate when the new event gets suppressed. In such approaches, all symptom events will be eventually suppressed by the root-cause event (also known as the failure event). Indeed after processing all the events in an event burst, the only unsuppressed event in the cache has to be the root-cause event. Hence, one can identify root-cause events (and thus diagnose faults) by filtering unsuppressed events in the cache. Diagnosed faults are ultimately funneled to network operators for corrective actions.

However, the pairwise correlation paradigm suffers from the following drawbacks: (1) the complexity of comparison grows quadratic in the size of event burst, (2) caching all the unsuppressed events results in memory bounded operations (especially, since event burst sizes of over 10,000 events per second are not uncommon), and (3) it lacks sufficient predictive capabilities and robustness to missing events (especially, if the event corresponding to the root cause is missing). More recently, [4] proposed a divide-and-conquer approach to enhance the scalability of the pairwise event correlation approach by partitioning a large network into multiple management domains, each of which are independently monitored. They proposed a hierarchical event correlation architecture wherein, event correlation is first performed within each management domain; then, a selected subset of events is sent for cross-domain correlation to a super root-cause-analysis engine. In comparison, we use CRCA to denote (pairwise) *centralized root cause analysis*, and DRCA the distributed version of the same. Furthermore, we consider a simplified version of CRCA that ignores down-stream correlation (which captures the most common types of cascading faults, and thus requires network-wide event correlation in the absence of down-stream topology indices).

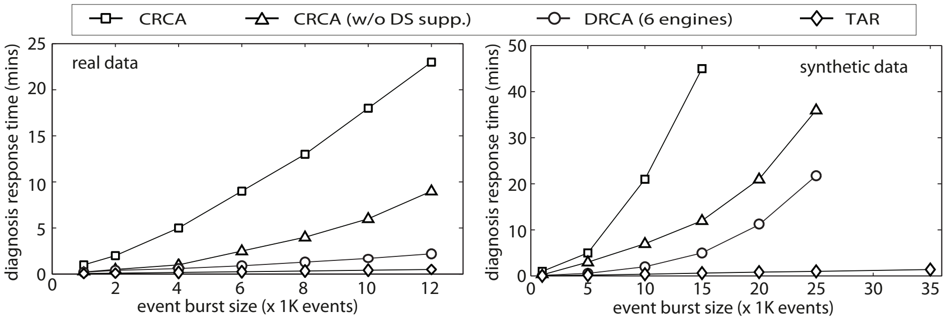

Figure 8: Scalability of multiple fault diagnosis approaches with respect to event burst size.

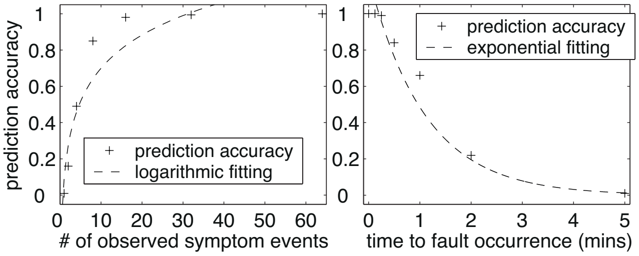

Figure 9: Prediction accuracy of TAR with respect to number of observed symptoms (synthetic dataset), and time to fault occurrence (real dataset).

Figures 8 presents the fault diagnosis response time of the multiple correlation approaches as a function of event burst size, using both real and synthetic event datasets. Figures show the ability of signature-based correlation approach to scale linearly with burst size; on the other hand, the pairwise event correlation approach is of quadratic complexity, and does not scale well to large-scale networks featuring high-volume event streams. It is noticed that among them, DRCA demonstrates better scalability than its centralized counterparts; however, it tends to pose much stricter computation power requirement than TAR, e.g., the memory required by cross-domain correlation easily hits the bound of the test platform (which causes the missing numbers for DRCA on the synthetic dataset).

Nevertheless, we note that the pairwise event correlation approach supports more generic event correlation than TAR, which only supports topological event correlation. Indeed in the case of network events, an event e_1 occurring on node n_1 and an event e_2 occurring on node n_2 are correlated only if the nodes n_1 and n_2 are topologically related. Hence, all event correlation rules used in the context of IP network could be readily translated into network signatures of TAR.

5.5 Predictive Power

For large communication networks, it is a desired feature of a fault diagnosis engine to raise certain warning ahead of the actual occurrence of the fault, i.e., before fully observing the symptom events, such that the network operators can quickly direct corrective efforts to the fault before its full-scale cascading.

Figures 9 shows the predictive capability of signature-based diagnosis approach. These results have been averaged over 121 real faults and 1,000 synthetic faults, respectively. Using a signature matching based approach allows us to partially match a network-event stream with a fault signature and predict the fault even before the failure event is actually observed (or received) by NMS. In our experiments with both synthetic and real event datasets, we ensured that the root-cause event (or the failure event) was withheld from the fault diagnosis engine, until after the engine predicts the fault.

Using synthetic event dataset, the left plot of Figure 9 shows the accuracy of fault prediction with respect to the number of observed symptom events. For a total set of more than 128 symptom events, observing only 15 symptom events gives prediction accuracy close to 1. Using real event dataset, the right plot of Figure 9 shows the fault prediction accuracy as a function of the time ahead of receiving (or observing) the failure event. The fitting curve shows that the predictive power grows exponentially as approaching the failure occurrence. In most cases (99.99%), it is possible to diagnose the fault accurately after the failure event is received; however, it is possible to predict failure about 30 seconds and one minute before the actual failure event is observed with accuracy 84% and 66%, respectively.

Predicting failures enables swifter recovery actions, and thus reduces SLA (service level agreement) violation costs (e.g., customer VPN tunnel failures). In such cases, predicting a failure event a couple of seconds in advance can help us reconfigure MPLS paths between two customer edge routers that are the end points of a (probably) faulty VPN tunnel. We also note that predicting a failure a minute before the actual failure is observed may be insufficient for other classes of failures (e.g., BGP route failures due to misconfiguration of BGP policies, whose ripple effects may propagate all over the network creating route stabilization problems).

The ability to withhold the failure event and yet diagnose the fault clearly shows that TAR can tolerate missing events (even if the missing event were the failure event itself). A pairwise event correlation approach that suppresses symptom events only after observing the suppressing event cannot tolerate missing root-cause

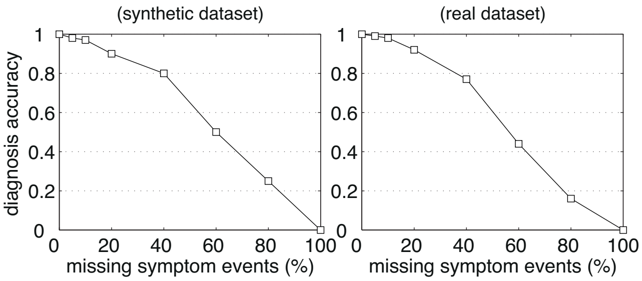

Figure 10: Fault diagnosis accuracy of Tar with respect to varying amount of missing symptom events.

events. In the absence of the root-cause event, several symptom events will be left unsuppressed; eventually such symptom events will (incorrectly) be funneled as failure events to network operators.

5.6 Error Tolerance

Above we have shown partially the robustness of Tar against missing symptom events. In this set of controlled experiments, we further explore in this direction. By varying the amount of withheld symptom events, we simulate random packet losses. The efficacy of fault diagnosis by Tar with respect to the amount of missing symptoms (averaged over 121 real faults and 1,000 synthetic faults) is shown in Figure 10. On both synthetic and real event datasets, Tar achieves an accuracy above 0.75, under missing 40% percent of symptom events. The result may seem at first glance in conflict with that in Figure 9, where Tar demonstrates prediction accuracy close to 1 after observing only 11.7% of symptoms. However, keep in mind that the symptom events arriving early at NMS tend to be topologically close to the faulty entity, thus bearing more information for fault diagnosis; while in this set of experiment, we simulate random package losses, irrespective of their arrival order (i.e., the information carried by the events), which explains the seeming inconsistency.

Figure 11 shows the efficacy of fault diagnosis in facing errors in the network topology information. We consider two models of topological errors, one by randomly modifying (adding or deleting) links in the network, and the other by modifying links according to a preferential attachment model [6]. As per the preferential attachment model, new links are likely to be added/deleted between a high degree node and a low degree node (degree of a node denotes the number of its neighbors). In such a model (which more accurately reflects the growth of a network and thus attributes to errors in network topology information [24]), updated links typically affect connectivity to the nodes that are in the periphery of the network, rather than the core nodes in the network. Such added/deleted network links tend to only locally affect network relationships; meanwhile, random addition/deletion of links may affect network relationships at the scale of the entire network itself. Consequently, Tar performs much better when topology errors are introduced as per the preferential attachment model than selected randomly.

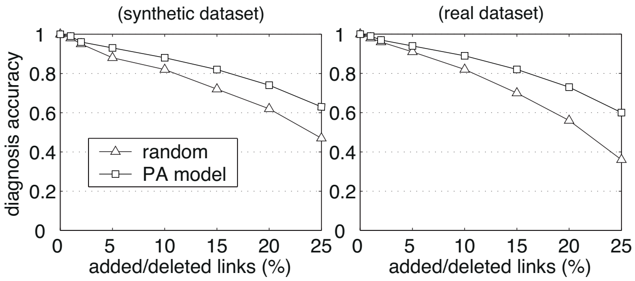

Figure 11: Diagnosis accuracy versus errors in topological information (introduced via changing links according to random or Preferential Attachment model).

6. RELATED WORK

Network management techniques have been evolving rapidly with advance in monitoring infrastructures, and gradual diversification of monitoring context.

Due to its wide spread availability, low-level metric data (e.g., network traffic or routing data) has been providing valuable information for network operators to monitor underlying network status. A plethora of work has been focused on static analysis of low-level metric data for disruption detection and trouble shooting in communication networks. Examples include analyzing BGP update messages using clustering to detect network disruption [10, 22], applying multivariate analysis to routing data to model normal network traffic and detect deviations [15, 12], and using wavelet-based clustering to detect abnormal routing behavior in routing data [27]. Moreover, disruption detection using historical metric data has also been an important topic for computing systems in general [8].

Many of today's network monitoring infrastructures can provide high-level, descriptive observations or symptoms (events). A line of research efforts have been dedicated to fault diagnosis from the set of observed symptoms. Existing solutions can be categorized roughly as expert-system or graph-theoretic techniques. The first category attempts to imitate the knowledge of domain experts, with examples including rule-based [23], cased-based [16], and model-based systems [19]; the second category relies on a graphical model of the system that describes the propagation for each specific fault, with examples as dependency graph [14], codebook [25], and belief-network [20]. The drawbacks of these techniques lie in the requirement for accurate dependency information amongst network entities (usually not available for large-scale enterprise networks), and the cost of fault inference (scale poorly with network size and complexity). In contrast, Tar only requires elementary topological information and network signatures to support online fault diagnosis over high-volume event streams.

With the emergence of more complex network contexts (e.g., information network, social networks, etc.), network management is exposed to monitoring data with increasingly richer semantics (e.g., email, social media, etc.). It poses great challenge to understand the information conveyed by monitoring data within the context of underlying network and yet, offers valuable insight into global network phenomena. For example, SybilGuard [26] was proposed to leverage underlying social network structures in interpreting nodes' voting messages, thus defending against Sybil attacks. We consider Tar as an initial effort towards understanding and modeling the interplay between monitoring data and underlying network context. Nonetheless, the network context setting in Tar is still fairly limited, e.g., we only consider topological relationships among network entities, and focus on pairwise interactions among them. It might be part of a temporary solution until more comprehensive models are proposed, and it might inform the design of these models.

7. CONCLUSION

This work advances the state-of-the-art in network monitoring data analysis by presenting Tar, a general framework of learning, indexing, and identifying topological and temporal correlation existing in network-event data, based on a novel class of network signatures. We present efficient learning algorithms to extract such signatures from noisy historical event data, and with the help of novel indexing structures, we show how to perform efficient, online signature matching against high-volume event streams. While focusing on topological-temporal patterns only is unlikely to capture the myriad semantic structures existing in network-event data, we show that it is a powerful primitive to support a range of applications. Our experiments of deploying Tar with a large-scale testbed NMS to perform fault diagnosis show that Tar is able to perform scalable, yet robust root cause analysis.

Acknowledgement

This work was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on. The first author and the last author are partially supported by grants from NSF CyberTrust program, NSF NetSE program, an IBM SUR grant, and a grant from Intel research council.

8. REFERENCES

[1] HP OpenView. http://www.openview.hp.com.

[2] IBM Tivoli Monitoring. http://www-01.ibm.com/software/tivoli/products/monitor/.

[3] H. Akaike. A new look at the statistical model identification. *IEEE Trans. Auto. Cont.*, 19(6), 1974.

[4] D. Banerjee, V. Madduri, and M. Srivatsa. A framework for distributed monitoring and root cause analysis for large ip networks. In *SRDS*, 2009.

[5] A.-L. Barabási. *Linked: The New Science of Networks*. Perseus Publishing, 2002.

[6] A.-L. Barabási and R. Albert. Emergence of Scaling in Random Networks. *Science*, 286(5439):509–512, 1999.

[7] E. Cohen, E. Halperin, H. Kaplan, and U. Zwick. Reachability and distance queries via 2-hop labels. *SIAM J. Comput.*, 32(5), 2003.

[8] I. Cohen, S. Zhang, M. Goldszmidt, J. Symons, T. Kelly, and A. Fox. Capturing, indexing, clustering, and retrieving system history. In *SOSP*, 2005.

[9] L. Fan, P. Cao, J. Almeida, and A. Broder. Summary cache: A scalable wide-area web cache sharing protocol. In *IEEE/ACM Trans. Netw.*, 1998.

[10] A. Feldmann, O. Maennel, Z. Mao, A. Berger, and B. Maggs. Locating internet routing instabilities. *SIGCOMM Comput. Commun. Rev.*, 34(4), 2004.

[11] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *SIGMOD*, 1984.

[12] Y. Huang, N. Feamster, A. Lakhina, and J. Xu. Diagnosing network disruptions with network-wide analysis. *SIGMETRICS Perform. Eval. Rev.*, 35(1), 2007.

[13] Internet Engineering Task Force. OSPF version 2. *http://www.ietf.org/rfc*.

[14] I. Katzela and M. Schwartz. Schemes for fault identification in communication networks. *IEEE/ACM Trans. Netw.*, 3(6), 1995.

[15] A. Lakhina, M. Crovella, and C. Diot. Mining anomalies using traffic feature distributions. *SIGCOMM Comput. Commun. Rev.*, 35(4), 2005.

[16] L. Lewis. A case-based reasoning approach to the resolution of faults in communication networks. In *IM*, 1993.

[17] J. P. Martin-Flatin, G. Jakobson, and L. Lewis. Event correlation in integrated management: Lessons learned and outlook. *J. Netw. Syst. Manage.*, 15(4):481–502, 2007.

[18] X. Meng, G. Jiang, H. Zhang, H. Chen, and K. Yoshihira. Automatic profiling of network event sequences: algorithm and application. In *IEEE INFOCOM*, 2008.

[19] Y. Nygate. Event correlation using rule and object based techniques. In *IM*, 1995.

[20] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., 1988.

[21] T. Wang, M. Srivatsa, D. Agrawal, and L. Liu. Learning, indexing, and diagnosing network faults. In *KDD*, 2009.

[22] J. Wu, Z. Mao, J. Rexford, and J. Wang. Finding a needle in a haystack: pinpointing significant BGP routing changes in an IP network. In *NSDI*, 2005.

[23] P. Wu, R. Bhatnagar, L. Epshtein, M. Bhandaru, and S. Zhongwen. Alarm correlation engine. In *NOMS*, 1998.

[24] K. Yamasaki, K. Matia, S. V. Buldyrev, D. Fu, F. Pammolli, M. Riccaboni, and H. E. Stanley. Preferential attachment and growth dynamics in complex systems. *Phys. Rev. E*, 74(3), 2006.

[25] S. Yemini, S. Kliger, E. Mozes, Y. Yemini, and D. Ohsie. High speed and robust event correlation. *Communications Magazine, IEEE*, 34(5), 1996.

[26] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman. Sybilguard: defending against sybil attacks via social networks. In *SIGCOMM*, 2006.

[27] J. Zhang, J. Rexford, and J. Feigenbaum. Learning-based anomaly detection in BGP updates. In *MineNet*, 2005.