Sharing End-user Negative Symptoms for Improving Overlay Network Dependability

Yongning Tang
School of Information Technology
Illinois State University
Normal IL 61790
ytang@ilstu.edu

Ehab Al-Shaer
School of Computing
DePaul University
Chicago IL 60604
ehab@cs.depaul.edu

Abstract

The dependability of overlay services rely on the overlay network's capabilities to effectively diagnose and recover faults (e.g., link failures, overlay node outages). However, overlay applications bring to overlay fault diagnosis new challenges, which include large-scale deployment, inaccessible underlying network information, dynamic symptom-fault causality relationship, and multi-layer complexity. In this paper, we develop an evidential overlay fault diagnosis framework (called DigOver*) to tackle these challenges. Firstly, the* DigOver *identifies a set of potential faulty components based on shared end-user observed negative symptoms. Then, each potential faulty component is evaluated to quantify its fault likelihood and the corresponding evaluation uncertainty. Finally, the* DigOver *dynamically constructs a plausible fault graph to locate the root causes of end-user observed negative symptoms.*

1. Introduction

Overlay networks have emerged as a powerful and flexible platform for developing novel network services [1, 3, 6]. The dependability of overlay services rely on the overlay network's capabilities to diagnose and recover faults, which may result from underlay and overlay layers. These faults occur for a variety of reasons.

Faults in the IP layer may be due to: (1) hardware problem (e.g., physical link disconnection, malfunctioned router interface), (2) software errors (e.g., IOS bugs) and (3) network misconfiguration (e.g., BGP misconfiguration). Overlay nodes inherit the same fault natures. For example, a recent query over 600 PlanetLab [3] nodes via CoMon [13] shows that the status of these overlay are highly unreliable, reflected by their uptime varying from 2.4 hours to 115.2 days with the average of 42.7 days. In addition to the underlay faults, overlay services may introduce new fault possibilities. First, virtualization technique that is commonly adopted on overlay nodes, may increase the chance of resource (e.g., CPU time or bandwidth) depletion. Second, overlay nodes may conduct more complicated functions (e.g., caching, probing) than regular routers. For a physically reliable overlay node, it may still become faulty. Third, overlay nodes may experience higher security risk that may lead to various problems. Thus, overlay networking increases the possible fault causes. Moreover, overlay networking brings the following new challenges to overlay fault diagnosis:

- *Scalability*: An overlay infrastructure is typically a large-scale distributed network involving with a large number of overlay and underlay nodes. For example, the current PlanetLab has more than 900 overlay nodes distributed over 400 networks across 45 countries. All overlay links may traverse more than 16,000 different routers [16]. Thus, it is difficult for an overlay service provider (OSP) to extensively monitor all possible faulty components.

- *Inaccessibility*: Overlay services are provisioned and operated by OSPs over opaque underlying ISP networks. ISPs are generally unwilling to share with OSPs their network statistics (e.g., prior fault probabilities of their routers), which are critical information for fault diagnosis.

- *Uncertainty*: Since comprehensively monitoring an overlay network is infeasible, overlay fault reasoning has to depend on incomplete and inaccurate information, which unavoidably results to uncertainty in the overlay fault diagnosis.

- *Dynamic symptom-fault causality*: The flexibility and dynamics of overlay nodes and links make the interaction between overlay and underlying networks unpredictable.

- *Multi-layer complexity*: Performance degradation may be due to malfunctioned overlay or underlay components.

Distinguishing faults between overlay and underlay components has significant impact on overlay fault recovery.

In this paper, we focus on the problem of locating faulty overlay components (including overlay nodes and underlay routers) when only end-user observed negative symptoms can be received by an OSP, and propose an evidential overlay fault diagnosis framework called *DigOver*. Firstly, the *DigOver* identifies a set of potential faulty components based on shared end-user observed negative symptoms. Then, each potential faulty component is evaluated to quantify its fault likelihood and the corresponding uncertainty via a basic belief assignment (bba) function. Finally, the *DigOver* dynamically constructs a plausible fault graph to locate the most likely root causes of end-user observed negative symptoms.

Though many fault diagnosis techniques were proposed as shown in Section 2, they are either based on unrealistic assumptions or present various limitations. The main technical challenge in the development of the *DigOver* is fault likelihood evaluation and the representation of the corresponding evaluation uncertainty. We introduce Dempster-Shafer theory, an uncertainty reasoning framework in Section 3. Then, we design a basic belief assignment function in Section 4 to analyze the correlation of end-user observations, evaluate the fault likelihood for overlay components and the corresponding evaluation uncertainty. In Section 5, we show the feasibility of diagnosing overlay faults based on end-user observations. The simulation and Internet experiments presented in Section 6 show that the *DigOver* can diagnose overlay faults effectively and accurately.

In conclusion, we make two main contributions in this paper: (1) we propose an evidential overlay fault diagnosis framework that only requires end-user observations; (2) we present the method for analyzing the correlation of end-user observations, the method for representing and handling reasoning uncertainty, and the method for combining and aggregating diagnosis results from multiple agents.

2 Related Work

There are many existing end-to-end based fault diagnosis schemes, which can be broadly classified into passive monitoring-based, active probing-based, and user observation-based approaches.

Passive Monitoring-based Approach. For an enterprise network, [5] attempts to achieve link-level diagnosis granularity by passively monitoring at least one network path that contains exactly one of every pair of links. Several approximation algorithms are proposed in [5] to minimize the number of required monitoring probes and monitored network paths. However, this technique is neither applicable nor scalable for overlay networks, where network links become overlay links and overlay paths are dynamically formed. In contrast, the *DigOver* regards end-users as well-distributed probes. Bayesian Belief Network (BBN) is widely adopted in passive approaches to associate end-to-end observable symptoms with their root causes [15, 18]. In such approach, a bipartite directed acylic Symptom-Fault casuality graph is used to provide a vector of correlation likelihood measure to bind a fault to a set of its related symptoms. In [10, 19], The BBN approach is also adopted to infer the faults in overlay networks. However, an overlay network profile or underlying network statistics are assumed to construct Symptom-Fault causality graph, which is not scalable for large-scale dynamic overlay networks. In [11], the Markov Chain Monte Carlo algorithm is used to identify lossy links in the network with tree topology based on passive traffic observation at a server.

Active Probing-based Approach. For overlay networks, [7] proposes an approach to monitor a minimal set of linearly independent paths so that the behavior (loss rates and latency) of all paths can be inferred with the granularity up to each virtual link. This approach requires full participation of all overlay nodes to provide an initial network topology. Even using the proposed dynamic algorithm for topology changes (e.g., single path added or deleted), [7] is not designed for overlay networks with high churn rate (i.e., overlay nodes join/leave frequently). In [8], multicast or unicast-base network tomography is proposed to send probing traffic to a set of end hosts to infer network link properties by exploiting the transmission correlation. But such techniques only work well when the two back-to-back probes are always both lost or are both transmitted successfully (i.e., perfect transmission correlation). PlanetSeer [20], taking the Multiple Vantage Point Approach locates Internet faults by selectively and periodically invoking "traceroute" from multiple vantage points. The measurement model is manually managed and only matches the application domain directions of data flow.

User Observation-based Approach In [12], an user observations "Comparison" based fault detection method is proposed for overlay multicast applications. In this method, a recipient node compares its received packets to the others received packets to verify if they are identical. To facilitate the packet comparison, Bloom filter is adopted to reduce the amount of compared information. However, it is not clear that how the compared node was selected and how such selection was maintained if overlay is highly dynamic. Moreover, the exchange of Bloom filters among nodes is intrusive to the network. Several end-user empirical function based diagnosis schemes have been proposed. For example, in [17], a collaborative overlay fault diagnosis framework

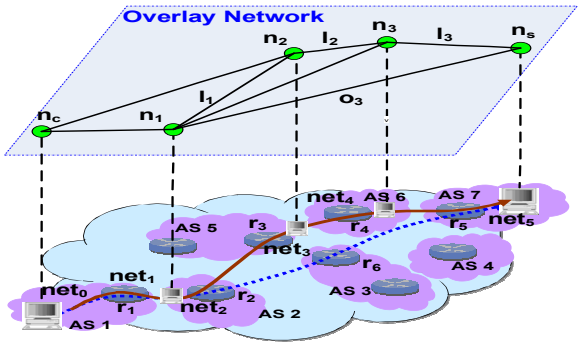

Figure 1. Overlay Network Model

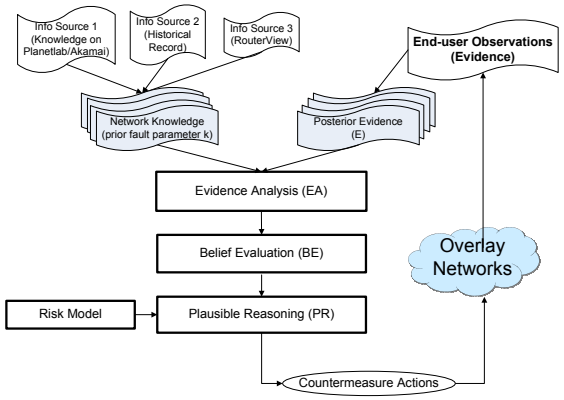

Figure 2. Evidential Overlay Fault Diagnosis.

has been presented, and an empirical likelihood function has been used to evaluate the status of network components. However, all these empirical approaches do not provide the insights behind different user empirical functions.

3 System Overview

We model an overlay network as a directed graph $G =< V, O >$, comprising a set of overlay nodes V joined by a set of bidirectional overlay links O as shown in Fig. 1. An *overlay link* is a default network path between two overlay nodes n_i and n_j $(n_i, n_j \in V)$ such as n_1 and n_2 in Fig. 1. However, an *overlay path* o_i may consist of multiple overlay links, which traverses a sequence of overlay components. Here, an *overlay component* could be a router, an overlay node, a network or an autonomous system (i.e., AS). Thus, an *overlay path* can be represented as a sequence of overlay components at different granular level. For example, the components contained by the overlay path o_3 between (n_1, n_s) can be represented in a router-level as $\{n_1, r_2, r_3, n_2, r_4, n_3, r_5, n_s\}$, in a network level as $\{net_2, net_3, net_4, net_5\}$, in an AS-level as $\{AS_2, AS_5, AS_6, AS_7\}$ or in a hybrid level as $\{r_2, AS_5, net_4, net_5\}$. In an overlay network, each end-user observation can be associated with a related overlay path (e.g., o_3 in Fig. 1), and thus can be represented as a set of components along the overlay path.

Definition 1. An end-user observed negative symptom is called an *evidence* e_i, which can be represented as a set of components along the corresponding overlay path. Thus, $e_i = \{c_1, c_2, \cdots, c_n\}$, where c_j is an overlay component.

We denote a relevant evidence set E_c for a component c as the set of different end-user evidence that contains the component c, i.e., $\forall e_i \in E_c, c \in e_i$. We denote the component visibility as the number of end-user evidence in E_c (i.e., $|E_c|$). For each observed negative symptom e_i (e.g., high packet loss or latency), there is at least one component c_j ($c_j \in e_i$) that is faulty. The occurrence of e_i can be also explained by any component c_j contained in e_i. Two evidences e_i and e_j are correlated if $e_i \cap e_j \neq \emptyset$.

3.1 Evidential Fault Diagnosis Overview

The *DigOver* includes three functional components: Evidence Analysis (*EA*), Belief Evaluation (*BE*), and Plausible Reasoning (*PR*) as shown in Fig. 2.

The evidence analysis (EA) component can identify a set of potential faulty components (C) based on end-user observed negative symptoms (E). *EA* can optionally use coarse-grained (general) network knowledge if available (e.g., a particular PlanetLab machine is known to be unreliable) to improve potential faulty components identification.

The belief evaluation (BE) component is to quantify the likelihood that a component c ($c \in C$) is faulty or non-faulty. As shown in Sec. 4, the fault likelihood of c is determined by its prior fault probability. However, the non-fault likelihood of c depends on the correlation of all c related end-user observations (i.e., E_c). The more chance that a set of different components $\{c_j\}$ (called a fault scenario)($c_j \in E_c, c_j \neq c$) can be used to explain E_c, c is more likely non-faulty (in good status). Finding all fault scenarios is an NP-complete problem (Shown in Sec. 4). The corresponding evaluation uncertainty is also measured at this stage.

The fault likelihood evaluation of all investigated components with their corresponding evaluation uncertainty are sent to the plausible reasoning (*PR*) component. By choosing an appropriate uncertainty handling strategy specified in the selected risk model, the total belief metric of each investigated component (denoted as Ψ_c) is calculated. Then, a plausible fault graph (*PFG*) is dynamically constructed to represent the correlation between each potential faulty component c and its related user evidence e_i ($e_i \in E_c$). Finally, *PR* locates the most likely faulty components based on their total belief metrics to explain end-user observations, which is proven an NP-complete problem.

As an option, general network knowledge if available can be used in the *DigOver* to improve the accuracy and efficiency of overlay fault reasoning. However, the *DigOver* does not rely on such network knowledge to work. In contrast, sufficient end-user observations can rectify the inaccurate knowledge about the investigated components.

3.2 Overview of Dempster-Shafer Theory

In this section, we introduce Dempster-Shafer theory (DST) as the theoretical background for the *DigOver* to represent and handle uncertainty.

For a given question in the DST, let Θ be a finite set of exhaustive and exclusive hypotheses to the question, called a *Frame of Discernment*. Let 2^Θ be the set of all subsets of Θ called the power set of $\Theta : 2^\Theta = \{A|A \subseteq \Theta\}$. The subset A represents a statement that the correct hypothesis lies in A. The *Dempster-Shafer Theory* is formulated in terms of a *basic belief assignment* function (called *m-function*) $m : 2^\Theta \rightarrow [0, 1]$ such that $m(\emptyset) = 0$ and $\sum_{A\subseteq\Theta} m(A) = 1$. The value $m(A)$ represents the degree of evidential support that a specific hypothesis of Θ belongs to A, but should not be included by any subset of A because the current evidence is not strong enough. Every set $A \subseteq \Theta$ for which $m(A) \neq 0$ is called a *focal element*. Associated with each basic assignment $m(\cdot)$ is a pair of measures: a belief measure (*Bel*) and a plausibility measure (*Pl*), which are determined for all sets $A \in 2^\Theta$ by the following equations [14]:

$$Bel(A) = \sum_{B\subseteq A} m(B) \text{ and } Pl(A) = \sum_{B \bigcap A\neq\emptyset} m(B) \quad (1)$$

In the DST, different beliefs based on independent evidence can be combined. Suppose m_1 and m_2 are two basic belief assignment (bba) functions used by two agents, Dempster's rule of combination provides a mechanism to calculate the joint bba as the following:

$$m_{12}(A) = \frac{\sum_{B \bigcap C=A} m_1(B) \times m_2(C)}{1 - \sum_{B \bigcap C=\emptyset} m_1(B) \times m_2(C)} \quad (2)$$

Developing an evidential reasoning technique using *Dempster-Shafer theory* typically includes three steps: (1) to define *Frame of Discernment* for a question; (2) to develop basic belief assignment function (m-function) for the specific problem related to the question; and (3) to design reasoning algorithm based on the belief evaluation and uncertainty measurement. In the next section, we show how to apply *Dempster-Shafer Theory* to the *DigOver* framework.

4 Evidential Overlay Fault Reasoning

In this section, we present the three components of the *DigOver*: Evidence Analysis, Belief Evaluation and Plausible Reasoning.

4.1 Evidence Analysis

It is impractical and unnecessary to evaluate all end-user observed negative symptoms because: (1) the probability of simultaneous faults on a network path is low [8,15], (2) for a component c_i with $|E_{c_i}| = 1$, its fault likelihood is certainly less than another component c_j on the same network path with $|E_{c_i}| \geq 1$, and (3) the diagnosis performance may be significantly dropped down if excessive processing resource is wasted on generating unnecessary evaluations.

In the *DigOver*, the following components are identified as potential faulty components to investigate:

- Shared Component: For a component c, the more related evidence e_c ($e_c \in E_c$) is received, the more likely the component c is faulty. Thus, we consider all shared components as potential faulty components.
- Overlay components: Overlay nodes are generally less reliable than typical network components, even for physically reliable overlay nodes (e.g., Akamai servers).
- Historically unreliable network components: The historical status records of network components if available can be used to estimate their prior fault probabilities.

4.2 Belief Evaluation

4.2.1 Defining Frame of Discernment

As used in boolean tomography methods [8], for each evidence e_i, we assign a *bad* label to the corresponding overlay path; for each *bad* overlay path, there is at least one *bad* (i.e., faulty) component. A component is labeled *good* otherwise.

Definition 2. The working status of a component c is called the condition of c. c is called in bad condition if c is faulty; otherwise, c is called in good condition.

The essential technique of overlay fault diagnosis is the evaluation on the conditions of each potential faulty component. Thus, the question of interest in the *DigOver* is "Given E_c, in which condition is a potential faulty component c?". Accordingly, we define the Frame of Discernment about the conditions of component c as the following.

Definition 3. The Frame of Discernment of a component c is the set of possible conditions of c denoted as Θ_c.

$$\Theta_c = \{B_c, G_c\}. \quad (3)$$

Here, B_c represents that c is in bad condition; and G_c represents that c is in good condition. It should be noted that B_c and G_c are the only two possible and mutually exclusive conditions of the component c.

4.2.2 Basic Belief Assignments

Basic belief assignment function (m-function) is to evaluate the fault likelihood of the component c based on E_c

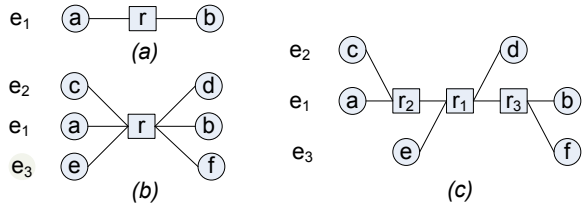

Figure 3. Fault Scenario Graph.

($\forall e_i \in E_c, c \in e_i$). Since Θ_c only contains two possible conditions: *bad* and *good*, we need to find the corresponding negative or positive indications from E_c to decide if a given E_c supports $\{B_c\}$, $\{G_c\}$, or $\{B_c, G_c\}$ (i.e., uncertain evidence), and how strongly E_c supports each.

Definition 4. The negative and positive indicators I_c^- and I_c^+ are the likelihood measures that a component c is in bad and good conditions respectively. The component condition ratio $\Re$ is the log-likelihood [9] ratio between I_c^- and I_c^+:

$$\Re_c = \log(\frac{I_c^-}{I_c^+}) = \log(I_c^-) - \log(I_c^+). \quad (4)$$

A component c is potentially faulty if $I_c^- \geq I_c^+$ or $\Re_c \geq 0$. In the following, we use two case studies to illustrate how to measure I_c^- and I_c^+ based on E_c.

- **Case 1:** In Fig. 3-a, $E_r = \{e_1\}$, here $e_1 = \{a, r, b\}$, reporting a negative symptom from the node a to b via r. Since all faults occur independently, we have $I_r^- = p_r$, and $I_r^+ = 1 - (1 - p_a)(1 - p_b)$, which is the probability that at least one component between a and b is *bad*. Here, p_i is the prior fault probability for a component i ($i \in \{a, r, b\}$). It should be noted that without e_1 that implies "at least one of $\{a, r, b\}$ is *bad*", we would have $I_r^+ = 1 - p_r$. In Fig. 3-b, $E_r = \{e_1, e_2, e_3\}$, here $e_1 = \{a, r, b\}$, $e_2 = \{c, r, d\}$, and $e_3 = \{e, r, f\}$. We still have $I_r^- = p_r$, but $I_r^+ = [1 - (1 - p_a)(1-p_b)][1-(1-p_c)(1-p_d)][1-(1-p_e)(1-p_f)]$, which is the probability that at least one component of $\{a, b\}$, one of $\{c, d\}$, and one of $\{e, f\}$ are *bad* simultaneously.

- **Case 2:** In Fig. 3-c, r_1 is the shared component contained by three evidences. $E_{r1} = \{e_1, e_2, e_3\}$, where $e_1 = \{a, r_1, r_2, r_3, b\}, e_2 = \{c, r_2, r_1, d\}$, and $e_3 = \{e, r_1, r_3, f\}$. In this case, we have a more general topological relationship among the evidences and their components. By the same reason that all faults occur independently, we have $I_{r_1}^- = p_{r_1}$. However, in order to measure $I_{r_1}^+$, we need to consider all possible conditions of other shared components (i.e., r_2, r_3) besides r_1. In *Case 2*, we have $I_{r_1}^+ = A + B$, where $A = p_{r_2}[p_{r_3} + (1-p_{r_3})(1-(1-p_e)(1-p_f))]$, is the probability that r_1 is *good* when r_2 is *bad* (p_{r_2}); and $B = (1-p_{r_2})(1-(1-p_c)(1-p_d))[p_{r_3} + (1-p_{r_3})(1-(1-p_a)(1-p_b))(1-(1-p_e)(1-p_f))]$, is the probability that r_1 is *good* when r_2 is also *good*.

The corresponding topological relationship among all components contained by E_c is called a fault scenario graph as shown in Fig. 3. Since $I_c^- = p_c$, in the following, we develop a method to calculate I_c^+ given an arbitrary set of evidence E_c. When calculating I_c^+, we assume c is in *good* condition and exclude it from E_c. The relationship between E_c and all related components contained in E_c can be represented by a boolean function $f(\cdot)$ as Eq. 5, which means at least one component (c_j) from each evidence (e_i) should be selected as a faulty component to collectively explain all observed evidence E_c.

$$f(E_c) = \bigwedge_{e_i \in E_c} f(e_i) = \bigwedge_{e_i \in E_c} \{ \bigvee_{c_j \in e_i} f(c_j) \}. \quad (5)$$

It is important to notice that this framework assumes that: (1) the fault diagnosis uncertainty of the network components is quite high (i.e., the prior faulty probability is low, which is typically the case in real networks), and (2) the probability of simultaneous faults on a network path is low. Thus, I_c^+ actually represents the creditability or likelihood measure of the hypothesis that a component c is in good condition considering the likely conditions of the other correlated components (i.e., the likelihood of other components being bad, assuming c is good). Consequently, this is determined based on the received evidences and prior fault probability of other components (but not the prior probability of the diagnosed component). This approach offers robust evidential reasoning as it gives more value to the received evidences and limits the inaccuracy effect due to prior fault probability estimation. Therefore, the DigOver diagnosis is effective only when there is significant uncertainty (i.e., low prior probability) in fault determination in which evidences will be useful to reason about faulty components.

Definition 5. Let E_c be a set of end-user observations containing the component c. A fault scenario is a set of faulty components (C) such that they can collectively explain all $e_i \in E_c$. This can be formally defined as:

$$\forall e_i \in E_c, \exists\, c_i \in C,\ \textit{explain}(c_i, e_i) \Longleftrightarrow FS(C, E_c). \quad (6)$$

where "*explain*" and "FS" are two predicates. *explain*(c_i, e_i) means c_i explains e_i. $FS(C, E_c)$ means C is a fault scenario of E_c.

Theorem 1. *Finding a fault scenario in E_C is a SAT problem.*

The proof can be found in [16]. To calculate I_c^+, we need to find all possible fault scenarios in Eq. 5 to explain E_c. This specific problem in *SAT* is called *SATALL*. We show in [16] that for this specific SATALL problem, there are eight clauses, 14 literals of each clause and 25 variables on average in the boolean expression as Eq. 5, which can be

solved in less than $0.1s$ using a regular computer (e.g., a PC with P4 CPU and 1GB memory).

Based on I_c^- and I_c^+, we can calculate $\Re_c$ to further develop an *m-function* for $\Theta_c = \{B_c, G_c\}$. As shown in [16], when $\Re_c = 0$, E_c equally supports both hypotheses $\{B_c\}$ and $\{G_c\}$. When $\Re_c$ approaches $-\infty$, E_c provides the maximal evidential support to $\{G_c\}$. When $\Re_c$ approaches $+\infty$, the evidence provides the maximal evidential support to $\{B_c\}$. Thus, we choose $\Re_c = 0$ as the threshold (denoted as $\Re_c^{TH}$). When $\Re_c \geq 0$, E_c as the evidence supports $\{B_c\}$; when $\Re_c < 0$, E_c supports $\{G_c\}$.

In the following, we denote $m(\{B_c\})$ as $m^-(c)$, $m(\{G_c\})$ as $m^+(c)$ and $m(\{B_c, G_c\})$ as $m^{\mp}(c)$. Theoretically, $\Re_c$ can approach $-\infty$ or $+\infty$, which is not required in practice to make a credible decision on the selection of hypotheses. Instead, we define $-\gamma$ and γ as thresholds to represent the target minimal ($\Re_c^{min}$) and maximal ($\Re_c^{max}$) evidential support. For example, if we set $\gamma = 3$, it means that the target maximal ($\Re_c^{max}$) can be reached if $I_c^- \geq 1000 I_c^+$. Accordingly, we define *m-function* for Θ_c as follows.

When $\Re_c \geq 0$,

$$\begin{cases} m^-(c) & = \min(1, \Re_c/\gamma), \\ m^{\mp}(c) & = 1 - m^-(c). \end{cases} \tag{8}$$

When $\Re_c < 0$,

$$\begin{cases} m^+(c) & = \min(1, -\Re_c/\gamma), \\ m^{\mp}(c) & = 1 - m^+(c). \end{cases} \tag{9}$$

Given E_c, the evaluation result is either $\Re_c \geq 0$ or $\Re_c < 0$. This means that for a single agent, it will not create any conflicted evaluation results. Accordingly, we have $Bel(\{B_c\}) = m^-(c)$ and $Pl(\{B_c\}) = 1$ when $\Re_c \geq 0$; $Bel(\{G_c\}) = m^+(c)$ and $Pl(\{G_c\}) = 1$ when $\Re_c < 0$.

4.3 Collaborative Overlay Fault Diagnosis

Multiple monitoring agents are usually required to diagnose faults for a large-scale distributed overlay network. In the following, we show how multiple agents or an individual agent can improve the fault diagnosis credibility via belief combination or belief aggregation.

4.3.1 Belief Combination

When individually collected evidence is insufficient to find the credible hypotheses, combining the beliefs from multiple agents can effectively improve belief trustworthiness, and reduce the impact of spurious beliefs (e.g., based on incredible end-user observations). Multiple individual belief evaluations on the same component can be combined via Dempster's rule as Eq. 2. If only negative evidence considered as in this paper, individual or combined belief only contains the evaluation on $\{B_c\}$ and $\{B_c, G_c\}$. However, if the positive evidence (e.g., successful network probing) also considered, the combined beliefs may contain $\{B_c\}$, $\{G_c\}$ and $\{B_c, G_c\}$. If the majority agents produce the credible evaluations, the belief combination can effectively reduce the impact of spurious beliefs.

4.3.2 Belief Aggregation

If two components c_1 and c_2 logically belong to a higher level component C, c_1 and c_2 are called child components of their parent component C. For example, the higher level component C could be an *ISP* (identified as its ASN) consisting of two lower level child components, which are the networks c_1 and c_2. When the available evidence for specific component is insufficient, the *DigOver* system can aggregate the beliefs on the different but related child components to construct a belief on their common parent component. The parent component C is *bad* if any of its child component is *bad* (logical *OR* relationship), and C is *good* if all of its child components are *good* (logical *AND* relationship). Thus, adaptively aggregating beliefs can make tradeoff between the diagnosis accuracy and granularity. For instance, the router r_1 and r_2 both belong to a network N. We have $m^-(N) = \max(m^-(r_1), m^-(r_2))$, reflecting a "OR" relationship for aggregating the negative belief; $m^+(N) = \min(m^+(r_1), m^+(r_2))$, reflecting an "AND" relationship for aggregating the positive belief; and $m^{\mp}(N) = 1 - m^-(N) - m^+(N)$. In [16], we prove $m^-(N) + m^+(N) \leq 1$.

4.4 Plausible Reasoning

4.4.1 Risk Models

$m^{\mp}(c) = Pl(B_c) - Bel(B_c) = 1 - m^+(c) - m^-(c)$ represents the uncertainty on $\{B_c\}$. If we could receive enough evidence, $m^{\mp}(c)$ can be dissected as $\omega m^{\mp}(c)$ to support $m^-(c)$, and $(1-\omega)m^{\mp}(c)$ to support $m^+(c)$ ($0 \leq \omega \leq 1$). We define a new parameter Ψ called a total belief metric to combine the three belief components: negative belief $m^-(c)$, positive belief $m^+(c)$, and uncertain belief $m^{\mp}(c)$ together as shown in Eq. 7. Since $m_c^- + m_c^+ + m_c^{\mp} = 1$, $\Psi_c \leq 1$. The utility function max() is to keep $\Psi_c \geq 0$. By choosing three binary variables K_1, K_2 and ω, we can have five risk models as shown in the Table 1.

In Model 1, only negative belief $\oplus_{i=1}^n m_i^-(c)$ is considered ($\oplus$ means belief combination). In Model 2, the combined negative and positive beliefs are aggregated via the deduction $\oplus_{i=1}^n m_i^-(c) - \oplus_{i=1}^n m_i^+(c)$. Using this model, the spurious beliefs can be effectively controlled or minimized as long as the majority agents are reliable. In Model 3, the combined plausible (negative plus uncertain) belief

$$\begin{aligned}\Psi_c &= \max[\,0,\ \oplus_{i=1}^{n} m_i^-(c) - K_1 \oplus_{i=1}^{n} m_i^+(c) + K_2(\omega \oplus_{i=1}^{n} m_i^{\mp}(c) - (1-\omega) \oplus_{i=1}^{n} m_i^{\mp}(c))\,] \\ &= \max[\,0,\ \oplus_{i=1}^{n} m_i^-(c) - K_1 \oplus_{i=1}^{n} m_i^+(c) - K_2(1-2\omega) \oplus_{i=1}^{n} m_i^{\mp}(c)\,]\end{aligned} \tag{7}$$

Model	K_1	K_2	ω	Ψ_c
1	0	0	/	$\oplus_{i=1}^{n} m_i^-(c)$
2	1	0	/	$\oplus_{i=1}^{n} m_i^-(c) - \oplus_{i=1}^{n} m_i^+(c)$
3	0	1	1	$\oplus_{i=1}^{n} m_i^-(c) + \oplus_{i=1}^{n} m_i^{\mp}(c)$
4	1	1	1	$\oplus_{i=1}^{n} m_i^-(c) - \oplus_{i=1}^{n} m_i^+(c) + \oplus_{i=1}^{n} m_i^{\mp}(c)$
5	1	1	0	$\oplus_{i=1}^{n} m_i^-(c) - \oplus_{i=1}^{n} m_i^+(c) - \oplus_{i=1}^{n} m_i^{\mp}(c)$

Table 1. Risk Models.

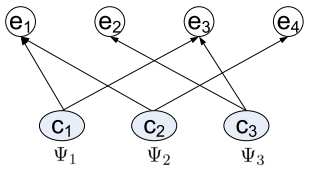

Figure 4. Plausible Fault Graph.

$\oplus_{i=1}^{n} m_i^-(c) + \oplus_{i=1}^{n} m_i^{\mp}(c)$ is considered. This model conservatively take all uncertain evaluation as negative indications. In Model 4, the positive belief is removed from the plausible belief. This model conservatively takes all the uncertain evaluation as the negative indications and also consider the potentially existing spurious (i.e., untrue) beliefs. In Model 5, both positive and uncertain beliefs are removed from the negative belief. This is the most optimistic belief function because the uncertainty $\oplus_{i=1}^{n} m_i^{\mp}(c)$ is treated positively. There is no single superior model. Depending on different applications and different network systems, one risk model may be more appropriate than the others. For example, if we know some fault diagnosis agents sometimes are not reliable but the majority is reliable, we should adopt Model 2 to reduce the impact of spurious beliefs.

4.4.2 Plausible Fault Reasoning

After the process of belief evaluation, a *Plausible Fault Graph* is constructed as the following to represent the dynamic relationship between the potential faulty components and the evidence: (1) for each evidence in E, to associate an evidence vertex e_i, (2) for each potential faulty component c, to associate a component vertex with its total belief metric Ψ_c, and (3) for each potential faulty component, to associate a link to all its related evidences E_c.

Comparing to the Symptom-Fault causality graph adopted in [15, 18], a *Plausible Fault Graph* has the following features: (1) dynamical creation, (2) evidence-driven model without relying on statistical information, (3) uncertainty measurement of a component's condition via a selected risk model (i.e., Ψ_c). A *plausible fault reasoning* problem is to find the minimal number of the most likely faulty components based on a *Plausible Fault Graph* that explain all observed evidence.

Theorem 2. *A plausible fault reasoning problem is NP-complete.*

The proof and a greedy heuristic algorithm for the plausible fault reasoning problem can be found in [16].

5 Correlation of End-user Observations

We show here how likely a network component can be observed by multiple end-users on the Internet. We deployed the *DigOver* system on 800 PlanetLab hosts and conducted the topology discovery experiment, in which each host ran *traceroute* to all other hosts simultaneously. From all *traceroute* results, we selected 450 hosts that can reach at least 90% other hosts with complete *traceroute* results. The set of reachable remote hosts from a given host h is called h's *Reachable Neighbor Set* or *RNS*. In our experiment, the average size of *RNS* is 723. The *traceroute* data includes 15,945 routers, 2,840 networks and 394 ASes. In our study, for a given *IP* address, we identify its network ID using both its *"/24"* subnetmask and its autonomous system number (*ASN*). We use a web service from *whois.cymrc.com* and the data provided by *Route Views* [4] in April 2008 to map an *IP* to its corresponding *ASN*. Only less than 1% IP addresses cannot be mapped.

We show the all identifiable components visibility at the *Router-level*, *Network-level*, and *AS-level* respectively, while varying the network size by including 10% to 100% of the total $450 \times 723 = 325,350$ network paths. In Fig. 5, when all network paths (100%) are selected, the percentages of the components with their visibility of four or more are 85%, 90% and 98% of the total components on the *Router-level*, *Network-level* and *AS-level* respectively. Even with 10% of the randomly chosen trace (i.e., network path) data, the percentages of the components with their visibility larger than four are 15%, 25% and 75% of the total components on the *Router-level*, *Network-level* and *AS-level* respectively. This result shows that there is inherent correlation among the end-user observations. However, it is evident that the component visibility closely relates to the number of different network traces, or user participation rate.

6 Evaluation with Simulation

In this section, we present our evaluation metrics, simulation methodology and simulation results.

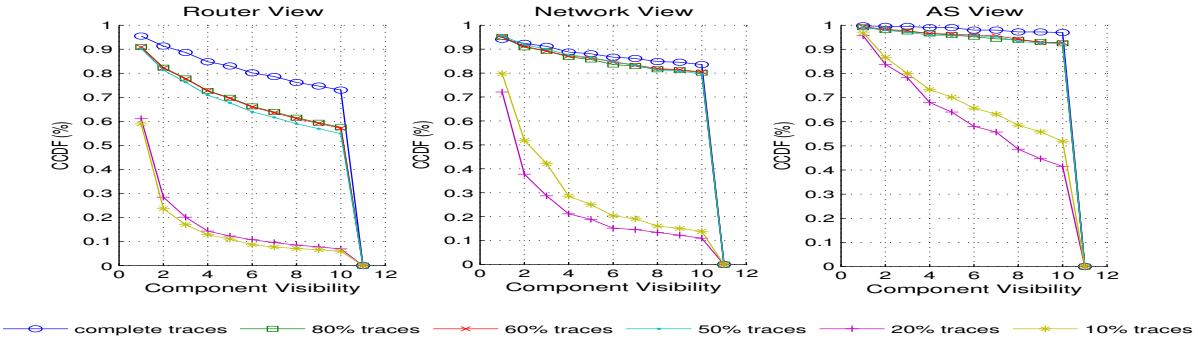

Figure 5. Network Correlation.

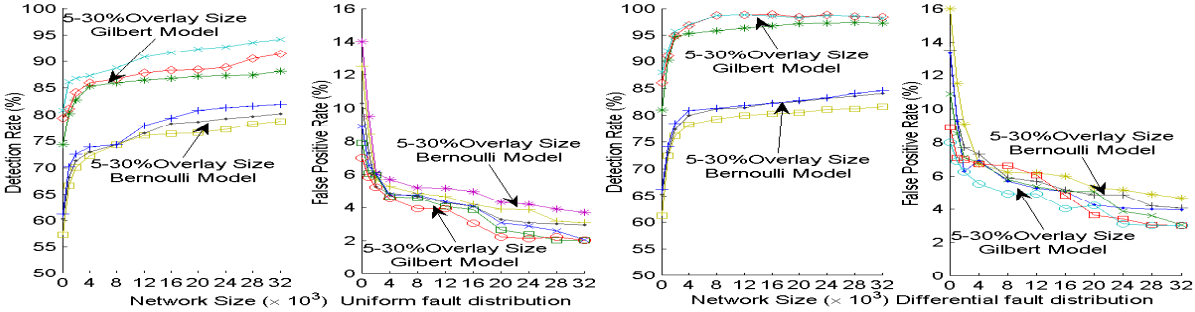

Figure 6. The Impact of Network Size and Loss Distribution with Hierarchical Topology.

6.1 Evaluation Metrics

The evaluation metrics include: (1) Detection Rate (DR), (2) False Positive Rate (FPR), and (3) Diagnosis Granularity (DG). Let C_f be a set of actual faulty components, and C_d be a set of detected faulty components. Detection Rate is defined as $DR = \frac{C_f \bigcap C_d}{C_f}$; and False Positive Rate is defined as $FPR = \frac{C_d - C_f}{C_d}$. Diagnosis Granularity (DG) is the ratio between $|C_d|$ and the number of components in the most fine-grained level (e.g., router-level).

6.2 Simulation Methodology

We consider the following dimensions for simulation and use packet loss as end-user perceivable negative symptoms.

- Topology type: We use BRITE [2] to experiment with Top-down hierarchical and Router-level topology, and vary network size from $1,000$ to $30,000$. We choose 10-30% nodes with the least degree as overlay nodes.
- Loss rate distribution and loss model: We use the similar loss distribution model as in [15]. The independent failure probabilities of all components are distributed as: (1) uniform fault distribution in range $[0.001, 0.01]$ for all nodes; (2) differential fault distribution in range $[0.001, 0.01]$ for underlay nodes (e.g., routers), but $[0.01, 0.1]$ for overlay nodes. Once each link has been assigned a loss rate, we use either a Bernoulli or Gilbert model to simulate the loss processes as in [7, 11].
- Overlay end-user observation ratio: We vary overlay end-user observation ratio from 100% reduced to only 20% to evaluate the *DigOver* performance.

6.3 Simulation Results

6.3.1 The Impact of Network Loss Model

As shown in Fig. 6, there is about 10-15% increase on the detection rate in Gilbert model comparing to Bernoulli model no matter network size, topology and fault probability distribution. This is because Gilbert model has been shown more realistic to represent the packets loss distribution on the Internet [22], thus can better represent the adopted component condition model in the *DigOver* as discussed in Sec. 4.2.1. All simulations (Fig. 6) show that the *DigOver* system is applicable for a large-scale overlay network. The simulation results in Fig. 6 also show that with the increase of network size, the detection rate (DR) is increased by 5-8% and the false positive rate (FPR) decreased by 4-9%. This is due to the fact that the average component visibility increases with network size increases.

6.3.2 The Impact of End-user Observation Ratio

We use the real network topology of the *PlanetLab* to evaluate how the *DigOver* performs when the end-user observation ratio varies (details available in [16]). For each experiment, we change overlay nodes from 50 to 350, and let

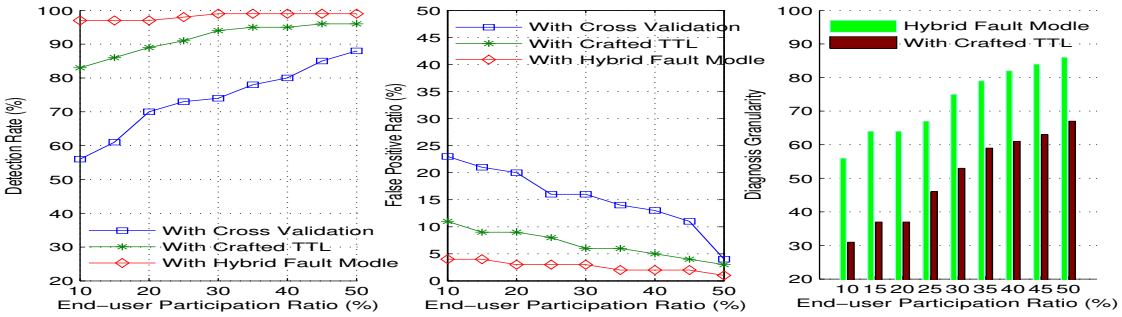

Figure 7. Internet Experiment with Different Fault Models and User Participation Ratio.

each selected overlay node to choose 5-30% from its Reachable Neighbor Set to connect. The result shows that when the end-user observation ratio increased from 50 nodes with 5% RNS (approximately $50 \times 5\% \times 723 \simeq 178$ paths) to 350 nodes with 30% RNS (approximately $350 \times 30\% \times 723 \simeq 75,915$ paths), the detection rate increases from 86% to 99%, and the false positive rate decreased from 18% to 3%.

7 Internet Experiments

In this section, we evaluate the *DigOver* on the Internet.

7.1 Methodology

We developed a test program called *DigOverTester*, and deployed it over the selected 800 PlanetLab hosts. *DigOverTester* is a multi-threaded program that can send and receive UDP packets simultaneously. Each UDP packet has 36 bytes including a 20-byte IP header, a 8-byte UDP header, and a 8-byte UDP payload with a sequence number and a timestamp. *DigOverTester* can also send UDP packet with crafted TTL. We choose 250ms as the sending interval such that the sending rate to each destined host is approximately $1kbps$, and the total sending rate is less than $750kbps$. We repeated 10 sessions and each session lasted 5 minutes, in which each host sent 1,200 36-byte UDP packets to every destined host. The receivers counted the number of received packets over 1,200 (the total number of sent packets) to calculate the packet loss rate for each end-to-end path. If the packet loss rate was larger than 3%, which is the user tolerance threshold as shown in [21], the receiver reported an end-user observation to the agent. Three types of experiments are developed to evaluate the *DigOver*:

- Type 1: *DigOverTester* is deployed to directly measure the packet loss over a set of network paths (denoted as PT). For an inferred *bad* overlay component, we directly validate the system logs of the corresponding *PlanetLab* host. For an inferred *bad* underlay component (e.g., a router), we use a cross validation method as follows. First, we associate each router r with all end-to-end network paths containing r (denoted as PT_r, $PT_r \subseteq PT$). If a path itself is reported as *bad*, we call that this path confirms the result that this component (i.e., r) is *bad*. Only if the ratio between all confirmed paths of r and the total number of paths in PT_r is larger than a threshold (90% in this case), we regard the hypothesis as a credible one.

- Type 2: A set of routers is selected from the discovered component set to investigate (called investigated components). For each router such as r, a list of network paths containing r is listed. For each path, let s be the source host, d be the destination host and T be the corresponding number of hops between s and r. For every source host s, a vector $< s, d, r, T >$ is generated and uploaded to s. Thus, all source hosts can be controlled by a central node to send UDP test packets with the corresponding crafted TTL. We repeat the same experiments as Type 1.

- Type 3: Hybrid faults are used such that in addition to the randomly introduced underlay faults using crafted TTL, the overlay faults are injected to the randomly selected PlanetLab hosts. For each injected overlay or underlay faults, a corresponding event is sent to a central agent and used for validating the reasoning results.

7.2 Experiment Results

The previous studies [7, 11] have shown the packet loss rarely occurred on the Internet and most occurred on the last mile networks. We have observed the same situation in our designed first type experiment. We define end-user participation ratio as the ratio of the participated PlanetLab hosts over the total 450 chosen PlanetLab hosts as discussed in Section 6.2. Every participated PlanetLab host randomly chose 100 other PlanetLab hosts from its RNS (Sec. 5).

In Fig. 7, when the user participation rate is low (less than 10%), it also resulted in low visibility on many network components, especially those on the last mile network segment. In such situation, the *DigOver* only has 55% detection rate and 24% false positive rate in the first type experiment. However, with the same situation, the *DigOver* performs much better, with 83% and 97% detection rate and 11% and 4% false positive rate in case of crafted TTL and

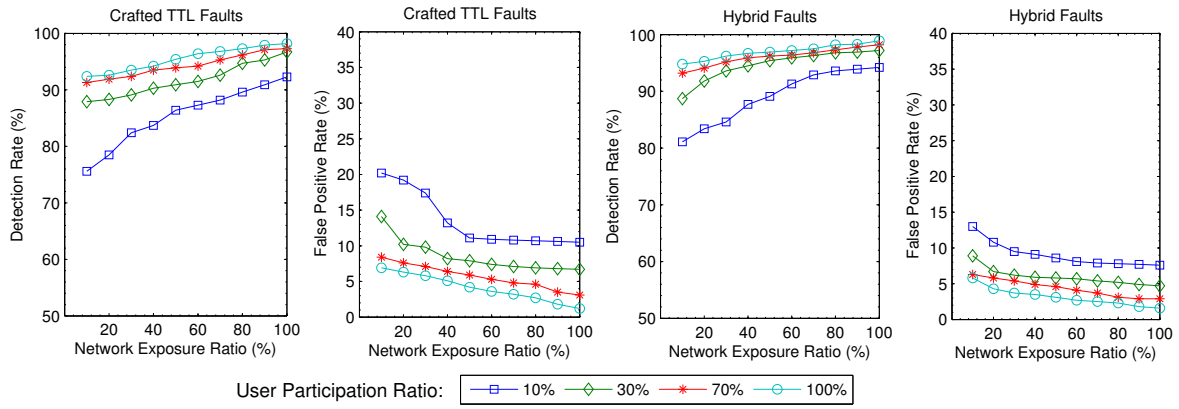

Figure 8. The Impact of Network Exposure Ratio with Crafted TTL Fault Model.

hybrid faults. When the end-user participation ratio significantly increased from 10% to 50%, the *DigOver* performs very well in all three types of experiments with detection rate increased from 84% to 99% and false positive ratio decreased from 5% to 2%. In Fig.7, we observe that the diagnosis granularity is also significantly increased from initial 30%, 55% to 63%, 86% (for Type 2 and 3).

In the case, when the user participation ratio is high (70% to 100%) but Network Exposure Ratio is low (5% and 20%), the scenario is closer to the real network environment. It means sufficient end-users participate into the *DigOver*, but for each end-user, the user network activities are usually moderate. In this case, as shown in Fig. 8, the detection rates for both fault models are 91% and 94%, and the false positive rates are relatively as high as 20% and 15%.

8 Conclusion and Future Work

In this paper, we propose an evidential overlay fault diagnosis framework (i.e., *DigOver*) to locate the most likely faulty overlay components based on end-user reported negative symptoms. Both extensible simulation and Internet experiments have shown that the *DigOver* can effectively and accurately diagnose overlay faults when end-user participation ratio is reasonable (e.g., 30%). In our future work, we plan to apply the *DigOver* to peer-to-peer (p2p) network to help p2p applications improve their performance.

References

[1] Akamai. http://www.akamai.com/en/html/technology.

[2] Brite. http://www.cs.bu.edu/brite/.

[3] Planetlab. http://www.planet-lab.org.

[4] Routeviews. http://www.routeviews.org.

[5] S. Agrawal, K. Naidu, and R. Rastogi. Diagnosing link-level anomalies using passive probes. In *INFOCOM*, 2007.

[6] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris. Resilient overlay networks. In *Proceedings of the 18th ACM SOSP*, October 2001.

[7] Y. Chen, D. Bindel, H. Song, and R. H. Katz. An algebraic approach to practical and scalable overlay network monitoring. In *In Proceeding of ACM SIGCOMM*, 2004.

[8] N. G. Duffield. Network tomography of binary network performance characteristics. In *IEEE Transactions on Information Theory*, volume 52, pages 5373–5388, 2006.

[9] J. Halpern. *Reasoning about Uncertainty*. MIT Press, 2003.

[10] G. J. Lee and L. Poole. Diagnosis of tcp overlay connection failures using bayesian networks. In *SIGCOMM 06 Workshops on Mining network data*, 2006.

[11] V. N. Padmanabhan, L. Qiu, and H. J. Wang. Server-based inference of internet link lossiness. In *INFOCOM*, 2003.

[12] V. Pappas, B. Zhang, A. Terzis, and L. Zhang. Fault-tolerant data delivery for multicast overlay networks. In *IEEE ICDCS*, March 2004.

[13] K. Park and V. S. Pai. Comon: A mostly-scalable monitoring system for planetlab. *Operating Systems Review*, 2006.

[14] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

[15] M. Steinder and A. S. Sethi. Probabilistic fault diagnosis in communication systems through incremental hypothesis updating. *Computer Networks*, 45(4):537–562, July 2004.

[16] Y. Tang and E. Al-Shaer. Plausible overlay fault reasoning. Technical report, DePaul University, October 2008.

[17] Y. Tang and E. Al-shaer. Towards collaborative user-level overlay fault diagnosis. In *INFOCOM MINIConference*, 2008.

[18] Y. Tang, E. Al-Shaer, and R. Boutaba. Active integrated fault localization in communication networks. In *IEEE/IFIP Symposium on Integrated Network Management*, 2005.

[19] Y. Tang, E. Al-Shaer, and R. Boutaba. Efficient fault localization using incremental alarm correlation and active investigation for internet and overlay networks. *IEEE Transactions on Network and Service Management (TNSM)*, 2008.

[20] M. Zhang, C. Zhang, V. Pai, L. Peterson, and R. Wang. Planetseer: Internet path failure monitoring and characterization in wide-area services. In *OSDI*, 2004.

[21] Y. Zhang, N. Duffield, V. Paxson, and S. Shenker. On the constancy of internet path properties. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop*, 2001.

[22] Y. Zhao, Y. Chen, and D. Bindel. Towards unbiased end-to-end network diagnosis. In *ACM SIGCOMM*, 2006.