Distributed Fault Localization in Hierarchically Routed Networks

M. Steinder, A. S. Sethi
Computer and Information Sciences Department
University of Delaware
Newark, DE
USA
{*steinder,sethi*}*@cis.udel.edu*

Abstract

Probabilistic inference was shown effective in non-deterministic diagnosis of end-to-end service failures. To overcome the exponential complexity of the exact inference algorithms in fault propagation models represented by graphs with undirected loops, Pearl's iterative algorithms for polytrees were used as an approximation schema. The approximation made it possible to diagnose end-to-end service failures in network topologies composed of tens of nodes. This paper proposes a distributed algorithm that increases the admissible network size by an order of magnitude. The algorithm divides the computational effort and system knowledge among multiple, hierarchically organized managers. The cooperation among managers is illustrated with examples, and the results of a preliminary performance study are presented. [1]

1 Introduction

End-to-end network service failure diagnosis [19, 21] is a sub-task of fault localization [8, 10, 23] that isolates host-to-host services responsible for availability or performance problems associated with a communication between end-hosts. In [20, 21], probabilistic inference was applied to provide a non-deterministic solution to this problem. The probabilistic fault propagation model representing the problem of end-to-end service failure diagnosis is a bipartite directed graph, which contains undirected loops. To overcome the exponential computational complexity required by the exact inference algorithms in graphs with loops, Pearl's iterative algorithms for polytrees were used as an approximation schema [21]. The algorithm introduced in [21] diagnoses end-to-end service failures in network topologies composed of tens of nodes, but it does not scale well to topologies composed of hundreds of nodes.

This paper expands on the iterative algorithm proposed in [21] and introduces its distributed version that increases the admissible network size by an order of magnitude. The algorithm divides the computational effort and system knowledge involved in end-to-end service failure diagnosis among multiple, hierarchically organized managers. Each manager is responsible for fault localization within the network domain it governs, and reports to the higher-level manager which

[1]Prepared through collaborative participation in the Communications and Networks Consortium sponsored by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

oversees and coordinates the fault localization process of multiple domains. With this organization, the technique is suitable for end-to-end service failure diagnosis in networks with hierarchical topologies, including IP networks.

The analysis within a domain is based on the topology information within this domain, with no information about other domains' configuration. The distribution of system knowledge among managers makes the fault localization model, which is crucial to the algorithms' operation, easier to obtain and maintain. In a given domain, fault localization aims at identifying faults that occurred in this domain based on symptoms pertaining to the domain. Faults that occur on communication links between domains are isolated by the higher-level manager, based on failures of end-to-end paths which span multiple domains. To resolve ambiguity resulting from intra-domain faults which affect multiple domains, the domain managers communicate their findings to the overseeing manager and a consensus decision is reached before any results are published.

The paper is structured as follows. In Section 2, we give background information on applying probabilistic inference to the problem of end-to-end service failure diagnosis. Section 3 introduces the distributed version of the algorithm. Section 4 presents illustrative examples of cooperation among managers. In Section 5, preliminary results of a simulation study are presented. The related work is described in Section 6.

2 Centralized end-to-end service failure diagnosis

Consider layer 2 or 3 topology in which communication between end-hosts is achieved through a network of bridges or routers. A failure of a host-to-host link in a given layer may cause a failure of an end-to-end service provided between two end-hosts along the path that includes the malfunctioning host-to-host link. To diagnose the end-to-end service problem, one needs to identify the faulty host-to-host link. We refer to this problem as the problem of end-to-end service failure diagnosis [19, 18]. Ability to isolate both availability and performance problems associated with host-to-host connections, which are responsible for the observed malfunctioning of end-to-end services in a given layer, is an important step toward a comprehensive solution to the problem of multi-layer fault diagnosis in communication systems.

End-to-end service failure diagnosis uses a directed bipartite graph as a fault propagation model, whose parentless vertices represent host-to-host failures (link failures), and childless vertices represent the resultant end-to-end service failures (path failures). To build such a graph, knowledge of network topology and current routing information is required. A short survey of techniques that allow such a model to be built automatically is presented in [17]. In this paper, we do not restrict ourselves to any of the possible methods [2, 13, 16]. In end-to-end service failure diagnosis, host-to-host and end-to-end service failures are considered faults and symptoms, respectively. The diagnosis of performance-related or upper layer end-to-end service problems requires a probabilistic fault model, in which parentless vertices (representing root causes) are labeled with independent failure probabilities and the directed edges are weighted with conditional probabilities representing the strengths of causal influences between host-to-host and end-to-end service failures. Typically [10, 11, 21], a noisy-OR probability model is used in which all alternative causes of the same effect are independent and combined using the OR logical operator.

Diagnosing end-to-end service failures may be mapped into the problem of finding the most probable explanation of the observed evidence in belief networks [14]. While the problem is known to be NP-hard [3], a polynomial-time inference algorithm was proposed for a restricted class of

belief networks represented by singly-connected graphs [14]. The algorithm is commonly referred to as Pearl's iterative belief propagation. In [21], Pearl's algorithm is used as an approximation schema for end-to-end service failure diagnosis based on fault propagation models with loops. This section briefly summarizes these results.

Iterative belief propagation utilizes a message-passing schema, in which the belief network vertices exchange λ and π messages (Figure 1). Message $\lambda_X(v_j)$ that vertex X sends to its parent V_j for every valid V_j's value v_j=0, 1, denotes a posterior probability of the entire body of evidence in the sub-graph obtained by removing link $V_j \rightarrow X$ that contains X, given that V_j=v_j. Message $\pi_{U_i}(x)$ that vertex X sends to its child U_i for every valid value of X, x=0, 1, denotes a probability that X=x given the entire body of evidence in the sub-graph containing X created by removing edge $X \rightarrow U_i$. Based on the messages received from its parents and children, vertex X computes its $\lambda(x)$, $\pi(x)$, and $bel(x)$, where $bel(x)$ is the probability that X=x given the entire evidence. Messages $\lambda_X(v_j)$ and $\pi_{U_i}(x)$, and functions $\lambda(x)$, $\pi(x)$, and $bel(x)$ are calculated using the following equations [14].

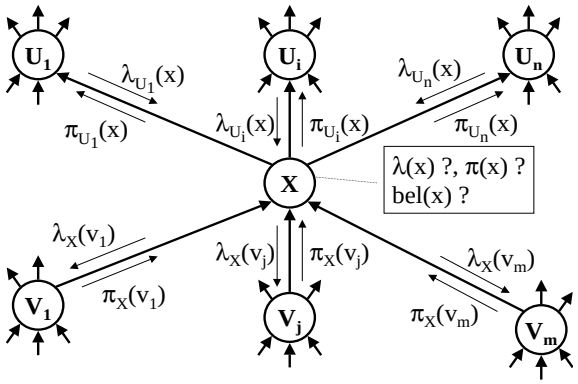

Figure 1: Pearl's belief propagation

$$\lambda_X(v_j) = \beta\Big(\lambda(1) - q_{V_jX}^{v_j}(\lambda(1) - \lambda(0)) \prod_{k \neq j} (1 - c_{V_kX}\pi_{kX})\Big) \quad (1)$$

$$\pi_{U_i}(x) = \alpha \prod_{k \neq i} \lambda_{U_k}(x)\pi(x) \quad (2)$$

$$\lambda(x) = \prod_{i=1}^{n} \lambda_{U_i}(x) \quad (3)$$

$$\pi(x) = \begin{cases} \alpha \prod_{j=1}^{m}(1 - c_{V_jX}\pi_{jX}) & \text{if } x = 1 \\ \alpha(1 - \prod_{j=1}^{m}(1 - c_{V_jX}\pi_{jX})) & \text{if } x = 0 \end{cases} \quad (4)$$

$$bel(x) = \alpha\lambda(x)\pi(x) \quad (5)$$

In the above equations, $c_{XU_i} = 1 - q_{XU_i}$ is the probability that U_i occurs given X occurs, α is a normalizing constant, and β is any constant.

The belief propagation algorithm in polytrees starts from evidence vertices and propagates the changed belief along the graph edges by computing $bel(x)$, $\lambda_X(v_i)$'s and $\pi_X(u_i)$'s in every visited vertex. The technique introduced in [21] (Algorithm 1) adapts the iterative belief propagation algorithm to the problem of fault localization with fault models represented by bipartite graphs with undirected loops. In this event-driven technique, one traversal of the entire graph is performed for every observed symptom. For each symptom a different ordering is defined that is equivalent to the breadth-first order started in the vertex representing the observed symptom. The set of all observed symptoms is denoted by $\mathcal{S}_O$.

Algorithm 1 (MPE through iterative belief updating)
Inference iteration starting from vertex Y_i:
let o be the breadth-first order starting from Y_i

for all vertices X such as X is not an unobserved path vertex, along ordering o
compute $\lambda_X(v_j)$ for all X's parents, V_j, and for all $v_j \in \{0,1\}$
compute $\pi_{U_i}(x)$ for all X's children, U_i, and for all $x \in \{0,1\}$

Symptom analysis phase:
for every symptom $S_i \in \mathcal{S}_O$ run inference iteration starting from S_i
compute $bel(v_i)$ for every vertex V_i, $v_i \in \{0,1\}$

Fault selection phase:
while $\exists$ link vertex V_j for which $bel(1)>0.5$ and $S_O \neq \emptyset$ do
take V_j with the greatest $bel(1)$ and mark it as observed to have value of 1
run inference iteration starting from V_j
remove all S_i such that V_j may cause S_i from S_O
compute $bel(v_i)$ for every vertex V_i, $v_i \in \{0,1\}$

The computational complexity of the algorithm is bound by $\mathcal{O}(|\mathcal{S}_O||E|) \subseteq \mathcal{O}(n^5)$, where E represents the set of belief network edges, and n represents the number of communication network nodes. It is shown through simulation that the algorithm offers close to the optimal accuracy [21]. The approximation significantly improves the feasibility of fault localization over the exact (but exponential) algorithm and allows the end-to-end service failure diagnosis to be efficiently performed in networks composed of tens of nodes. Additionally [20], the algorithm does not require the accurate knowledge of conditional probability distribution being able to retain high accuracy when a few confidence intervals are used instead of the exact conditional probabilities. It is also resilient to lost and spurious symptoms and allows positive symptoms to be incorporated without increasing the algorithm's computational complexity [20].

3 Distributed end-to-end service failure diagnosis

Centralized management has several well known shortcomings, which include:

- Single point of failure
- Inflexibility – the same management strategy is applied to the entire system even though particular subsystems may have different requirements
- Inefficiency
- Vulnerability to security breaches resulting from maintaining the management information of the entire system in a central location
- Infeasibility – when subsystems are in different administrative domains, obtaining management information, such as topology, routing, or internal state, may be impossible outside of the domain

In this paper, we propose a distributed fault management technique of end-to-end service failure diagnosis, which takes advantage of the domain semantics of real-life communication systems. The management domains considered by the technique correspond to administrative or routing network domains. We adopt the hierarchical organization of the management system. Although multiple levels of hierarchy are possible, for the sake of clarity (but without loss of generality) we will describe a two level management system in which domain managers (DM) report to the global manager (GM) overseeing the entire network. We assume that the following conditions are met:

1. Management domains are disjoint; clearly, this requirement is met when domains correspond to IP subnetwoks identified by their network IP address and mask.

2. No path which begins and ends in the same domain includes nodes from another domain. This requirement is met whenever hierarchical routing is used.

3. A domain manager is able to obtain topology and routing information in the domain it manages. Consequently, for every path that begins and ends in the managed domain, it is able to obtain a set of links within the domain through which the path service is provided. This information may be non-deterministic.

4. The global manager is able to obtain the list of all links whose beginning and ending nodes are located in different domains. In addition, for any pair of management domains, the global manager identifies the set of inter-domain links used to provide a communication service between the two domains.

We introduce the following notation.

$\mathcal{D}_1, \ldots, \mathcal{D}_n$	The set of management domains within the managed system
DM_i	Manager of domain $\mathcal{D}_i$
d_i	A unique identifier of domain $\mathcal{D}_i$, e.g., a network IP address and mask.
n_k	An identifier of a node in domain $\mathcal{D}_i$, e.g., its IP host address. Node identifiers are unique within a domain.
$d_i.n_k$	Network-wide unique identifier of node $n_k \in \mathcal{D}_i$.
$d_i.n_k \rightarrow d_j.n_l$	A directed link from node $d_i.n_k$ to node $d_j.n_l$.
$d_{j_1}.n_{p_1} \stackrel{*}{\rightarrow} d_{j_m}.n_{p_m}$	A directed, possibly multihop path from node $d_{j_1}.n_{p_1}$ to node $d_{j_m}.n_{p_m}$ consisting of links $d_{j_1}.n_{p_1} \rightarrow d_{j_2}.n_{p_2}$, …, $d_{j_{m-1}}.n_{p_{m-1}} \rightarrow d_{j_m}.n_{p_m}$.
$s_v{:}d_i.n_k \stackrel{*}{\rightarrow} d_j.n_l$	A symptom associated with path $d_i.n_k \stackrel{*}{\rightarrow} d_j.n_l$
$f_w{:}d_i.n_k \rightarrow d_j.n_l$	A fault associated with link $d_i.n_k \rightarrow d_j.n_l$
$d_i \stackrel{*}{\rightarrow} d_j$	The set of all paths which begin in domain $\mathcal{D}_i$ and end in domain $\mathcal{D}_j$, i.e., $d_i \stackrel{*}{\rightarrow} d_j = \{d_i.n_k \stackrel{*}{\rightarrow} d_j.n_l \mid n_k \in \mathcal{D}_i \text{ and } n_l \in \mathcal{D}_j\}$.
$s_z{:}d_i \stackrel{*}{\rightarrow} d_j$	A symptom associated with the set of paths $d_i \stackrel{*}{\rightarrow} d_j$, which indicates that at least one $s_v{:}d_i.n_k \stackrel{*}{\rightarrow} d_j.n_l$ occurred such that $n_k \in \mathcal{D}_i$ and $n_l \in \mathcal{D}_j$.

From requirements 2 and 3 it is clear that a domain manager is able to diagnose all intra-domain path failures observed within the domain it manages. Thus, for intra-domain fault localization the system model and fault localization algorithm should be similar to those used in the centralized management.

In a multi-domain environment, a fault in one domain may cause symptoms in other domains. For example, a fault causing a failure of path $d_i.n_k \stackrel{*}{\rightarrow} d_i.n_l$, in domain $\mathcal{D}_i$ may be observed as a failure of an inter-domain path $d_{j_1}.n_{p_1} \stackrel{*}{\rightarrow} d_{j_m}.n_{p_m}$, such that $d_i.n_k, d_i.n_l \in \{d_{j_1}.n_{p_1}, \ldots, d_{j_m}.n_{p_m}\}$ and $d_i.n_k$ precedes $d_i.n_l$. Such a path $d_i.n_k \stackrel{*}{\rightarrow} d_i.n_l$ will be referred to as an intra-$\mathcal{D}_i$ segment of inter-domain path $d_{j_1}.n_{p_1} \stackrel{*}{\rightarrow} d_{j_m}.n_{p_m}$.

In the technique proposed in this paper, an inter-domain symptom $s_{t_1}{:}d_{j_1}.n_{p_1} \stackrel{*}{\rightarrow} d_{j_m}.n_{p_m}$ is handled by the GM, which is able to identify intra-domain path segments which might have caused s_{t_1}. GM delegates the task of intra-domain path segments' diagnosis to the corresponding domain managers. Thus, after symptom s_{t_1} is observed, GM will delegate the diagnosis of path segment

$d_i.n_k \xrightarrow{*} d_i.n_l$ to DM_i. It does so by simply creating and reporting symptom s_{t_2}:$d_i.n_k \xrightarrow{*} d_i.n_l$ to DM_i. With every such symptom a high level of uncertainty is associated: since s_{t_1} might have been caused by path segments located in domains other than $\mathcal{D}_i$, symptom s_{t_2} passed to DM_i is likely to be spurious. While forwarding s_{t_2} to DM_i the GM includes the value of the belief with which the symptom should be considered spurious in $\mathcal{D}_i$, $p_s(s_{t_2})$, and the information on a path between which two domains the failure occurred. The DM_i correlates symptoms received from the GM with those reported internally in domain $\mathcal{D}_i$.

A failure of an inter-domain path $d_{j_1}.n_{p_1} \xrightarrow{*} d_{j_m}.n_{p_m}$ may also be caused by a failure of an inter-domain link $d_{j_s}.n_{p_s} \rightarrow d_{j_{s+1}}.n_{p_{s+1}}$, such that $d_{j_s}.n_{p_s}, d_{j_{s+1}}.n_{p_{s+1}} \in \{d_{j_1}.n_{p_1}, \ldots, d_{j_m}.n_{p_m}\}$. Failures of inter-domain links have to be isolated by the GM, since no domain manager has sufficient knowledge about inter-domain connectivity to make this determination. During the process of diagnosing inter-domain symptoms, which might have been caused by both an inter-domain link failure and an intra-domain path-segment failure, the GM must collaborate with the domain managers. The higher the probability that a fault occurred in domain $\mathcal{D}_i$, which might have caused the failure of an intra-$\mathcal{D}_i$ segment of an inter-domain path, the lower the probability that the inter-domain path failure has been caused by any inter-domain link. Thus, before reporting an inter-domain link as a possible cause of a failure of inter-domain path $d_{j_1}.n_{p_1} \xrightarrow{*} d_{j_m}.n_{p_m}$, the GM requests from the manager of each domain $\mathcal{D}_i$ traversed by path $d_{j_1}.n_{p_1} \xrightarrow{*} d_{j_m}.n_{p_m}$, the failure probability of an intra-$\mathcal{D}_i$ path segment of $d_{j_1}.n_{p_1} \xrightarrow{*} d_{j_m}.n_{p_m}$, which is independent of any inter-domain symptom observations. To ensure the independence, the intra-domain path-failure probability is calculated by DM_i based solely on the symptoms observed by DM_i, excluding those that DM_i receives from GM. As new evidence, inaccessible to GM, is observed and analyzed within $\mathcal{D}_i$, the value of this probability changes.

In the following section, we present a more detailed description of the proposed technique, including the presentation of a distributed fault localization model, domain managers' and GM's algorithms, and the cooperation between GM and the DMs.

3.1 Distributed fault propagation model

In a distributed technique, the responsibility for maintaining the fault localization model is shared among DMs and the GM. Domain managers maintain fault propagation models for the domains they manage. A fault propagation model built by manager DM_i represents causal relationships among path and link services provided within domain $\mathcal{D}_i$. Such a model may be built in advance and/or dynamically extended as symptoms related to particular paths are observed. For every monitored path $d_i.n_{p_1} \xrightarrow{*} d_i.n_{p_m}$, DM_i's model contains vertices $V_p, V_{l_1}, \ldots$, and $V_{l_{m-1}}$ labeled s_p:$d_i.n_{p_1} \xrightarrow{*} d_i.n_{p_m}$, f_{l_1}:$d_i.n_{p_1} \rightarrow d_i.n_{p_2}$, …, $f_{l_{m-1}}$:$d_i.n_{p_{m-1}} \rightarrow d_i.n_{p_m}$, respectively. It also contains causal edges, each originating from one of $V_{l_1}, \ldots$, and $V_{l_{m-1}}$, which end at V_p. Link vertices are also labeled with prior failure probabilities, and causal edges are weighted with the probability of the causal influence taking place.

As an example, consider a three-domain network presented in Fig. 2. To diagnose symptom s_4:$2.17 \xrightarrow{*} 2.19$, given path $2.17 \xrightarrow{*} 2.19$ consists of links $2.15 \rightarrow 2.19$ and $2.17 \rightarrow 2.15$, DM_2 creates vertices labeled s_4:$2.17 \xrightarrow{*} 2.19$, f_2:$2.15 \rightarrow 2.19$ and f_4:$2.17 \rightarrow 2.15$, and connects vertices f_2:$2.15 \rightarrow 2.19$ and f_4:$2.17 \rightarrow 2.15$ to vertex s_4:$2.17 \xrightarrow{*} 2.19$. DM_2's belief network presented in Fig. 3 models fault propagation for symptoms s_1:$2.15 \xrightarrow{*} 2.16$, s_2:$2.15 \xrightarrow{*} 2.19$, s_3:$2.15 \xrightarrow{*} 2.18$, and s_4:$2.17 \xrightarrow{*} 2.19$.

This model is dynamically extended as new symptoms are received from the GM. Recall that the GM delegates a part of the fault diagnosis task initiated by an inter-domain symptom to DMs of domains traversed by the corresponding path. Thus, for an inter-domain path $d_{j_1}.n_{p_1} \xrightarrow{*} d_{j_m}.n_{p_m}$, which includes path segment $d_i.n_k \xrightarrow{*} d_i.n_l$, the GM forwards to DM_i the following information: symptom s_t:$d_i.n_k \xrightarrow{*} d_i.n_l$, the probability that s_t is spurious in $\mathcal{D}_i$, $p_s(s_t)$, and a pair (d_{j_1}, d_{j_m}). DM_i creates (if it does not yet exist) a new vertex V_p labeled with $\bar{s_p}$:$d_{j_1} \xrightarrow{*} d_{j_m}$ and then creates a causal edge from V_p to vertex V_t labeled with s_t:$d_i.n_k \xrightarrow{*} d_i.n_l$. Vertex V_p, which represents all possible causes of symptom s_t that are external to $\mathcal{D}_i$, is additionally labeled with $p_s(s_t)$. The edge from V_p to V_t is assigned weight 1.0. In addition, for every symptom vertex the following function is defined: `state-of`: $V \rightarrow \{$`UNOBSERVED`, `OBSERVED-INTERNAL`, `OBSERVED-EXTERNAL`$\}$.

As an example, consider link $2.15 \rightarrow 2.19$ in domain $\mathcal{D}_2$ in Fig. 3, which belongs to the shortest path between domains $\mathcal{D}_1$ and $\mathcal{D}_0$. Its failure may cause a failure of inter-domain path $1.22 \xrightarrow{*} 0.28$ consisting of links $1.22 \rightarrow 1.20$, $1.20 \rightarrow 1.24$, $1.24 \rightarrow 2.15$, $2.15 \rightarrow 2.19$, $2.19 \rightarrow 0.25$, and $0.25 \rightarrow 0.28$. The GM identifies the following possible causes of the symptom: intra-$\mathcal{D}_1$ path segment $1.22 \xrightarrow{*} 1.24$, link $1.24 \rightarrow 2.15$, intra-$\mathcal{D}_2$ path segment $2.15 \xrightarrow{*} 2.19$, link $2.19 \rightarrow 0.25$, and intra-$\mathcal{D}_0$ path segment $0.25 \xrightarrow{*} 0.28$. DM_2 receives from the GM symptom s_2:$2.15 \xrightarrow{*} 2.19$, $p_s(s_2)$, and a pair $(1, 0)$. It creates a vertex labeled $\bar{s_5}$:$1 \xrightarrow{*} 0$ and connects it to the vertex labeled s_2:$2.15 \xrightarrow{*} 2.19$ (Fig. 3). The failure of link $2.15 \rightarrow 2.19$ may also cause problems in data transfer from domain $\mathcal{D}_1$ to some of the nodes in $\mathcal{D}_2$, and from some of the nodes in $\mathcal{D}_2$ to domain $\mathcal{D}_0$. The DM_2 fault propagation model presented in Fig. 3 includes vertices and edges which are created when failures of paths $1.22 \xrightarrow{*} 0.28$, $1.23 \xrightarrow{*} 2.18$, $2.15 \xrightarrow{*} 0.28$, and $1.20 \xrightarrow{*} 2.16$ occur.

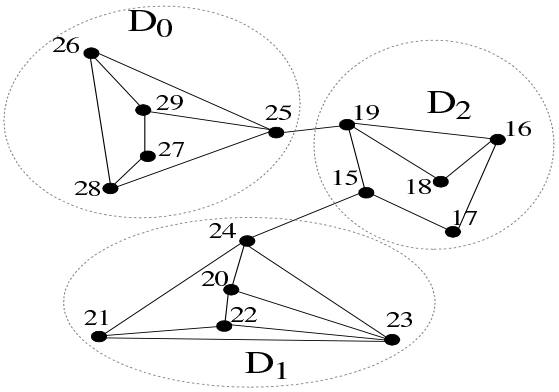

Figure 2: Multi-domain network topology

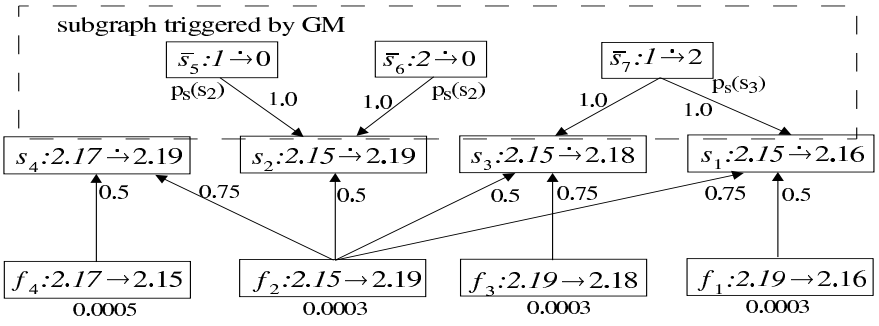

Figure 3: DM_2's belief network

A fault propagation model built by the global manager is concerned with connectivity among domains rather than particular nodes. It contains three types of vertices: those representing inter-domain links, inter-domain path sets, and domains. Suppose connectivity between domains $\mathcal{D}_i$ and $\mathcal{D}_j$ is provided using an inter-domain link $d_{j_s}.n_{p_s} \rightarrow d_{j_t}.n_{p_t}$ and intra-domain path segment

$d_u.n_k \xrightarrow{*} d_u.n_l$. The model should contain a vertex V_p labeled with $s_p{:}d_i \xrightarrow{*} d_j$, which represents the set of all paths that begin in $\mathcal{D}_i$ and end in $\mathcal{D}_j$, vertex V_l labeled with $s_l{:}d_{j_s}.n_{p_s} \rightarrow d_{j_t}.n_{p_t}$, and vertex V_d representing domain $\mathcal{D}_u$ labeled with $D{:}d_u$. The causal edges point from vertices V_l and V_d to vertex V_p. Vertex V_d is additionally labeled with prior failure probability equal to 1.0. The edge between V_d and V_p is labeled with $p_c(d_u, d_i, d_j)$, i.e., the probability that as a result of faults in $\mathcal{D}_u$, any intra-$\mathcal{D}_u$ path segment is affected that belongs to at least one path in the path-set $d_i \xrightarrow{*} d_j$. Observe that each vertex labeled with $d_i \xrightarrow{*} d_j$ in GM's model is connected to at least two vertices representing a domain, i.e., these labeled with $D{:}d_i$ and $D{:}d_j$. Similarly to DMs, GM provides function `state-of`: $V \rightarrow \{$`UNOBSERVED`, `OBSERVED`$\}$, which is defined for each symptom vertex. Fig. 4 presents an example belief network created by a global manager of the network in Fig. 2.

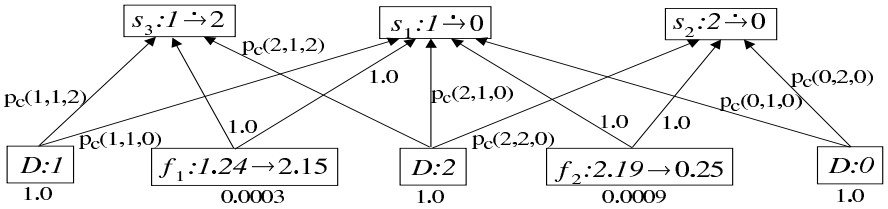

Figure 4: GM's belief network

3.2 Global manager's fault localization algorithm

We begin describing the distributed fault localization algorithm proposed in this paper by presenting the algorithm executed by the global manager (Algorithm 2). The GM's algorithm is composed of three phases, which may be interleaved. However, for the sake of clarity, they are presented as separate components.

Model synchronization phase aims at initializing or adjusting conditional probability values assigned to the causal edges between vertices representing domains and inter-domain paths as defined in Section 3.1. This phase must be first executed when the model is initialized and then repeated before final fault selection is made. The purpose of the repeated model synchronization is to update the values of the conditional probabilities assigned to edges between domain and path vertices, which have changed during fault localization process as a result of intra-domain symptoms analysis. When model synchronization is repeated in the fault selection phase, the `for` loop iterates only through observed inter-domain path vertices. The other vertices are ignored, as any modifications to causal probabilities assigned to their inbound edges would have no effect on the overall result.

Symptom analysis phase, given an inter-domain symptom, performs one iteration of probabilistic inference proposed in Algorithm 1 using the GM's internal fault propagation model. Then, through reporting a symptom, it requests that managers of domains traversed by the inter-domain path perform the same operation using their internal models based on symptoms received from the GM. Observe that in GM's model, vertex V_p labeled with $s_p{:}d_i \xrightarrow{*} d_j$ is assigned the value of 1 as soon as the first symptom referring to a path belonging to the set $d_i \xrightarrow{*} d_j$ is observed. Subsequent failures of paths from this set are ignored by the GM, which significantly reduces the amount of computation performed by the GM. Also observe that intra-domain path segments of an inter-domain path are easy to determine by scanning the sequence of inter-domain links used to provide connectivity from the domain of origin to the destination domain.

Algorithm 2 (Global Manager's Fault Localization Algorithm)
Model synchronization phase:

for every vertex V_p labeled with $d_i \xrightarrow{*} d_j$
  for every vertex V_d labeled with D:d_u such that V_d and V_p are connected
    obtain $p_c(d_u, d_i, d_j)$ and label edge (V_d, V_p) with $p_c(d_u, d_i, d_j)$.
    run inference iteration starting from V_d

Symptom analysis phase:

for every observed symptom s_u:$d_i.n_{k_i} \xrightarrow{*} d_j.n_{k_j}$ such that $d_i \neq d_j$
  let V_p be the vertex labeled with s_p:$d_i \xrightarrow{*} d_j$
  if `state-of`(V_p) =`UNOBSERVED`
    set `state-of`(V_p) =`OBSERVED` *and $V_p = 1$*
    run inference iteration starting from V_p
  for every intra-domain path segment of $d_i.n_{k_i} \xrightarrow{*} d_j.n_{k_j}$, $d_u.n_m \xrightarrow{*} d_u.n_l$
    create s_t:$d_u.n_m \xrightarrow{*} d_u.n_l$ and calculate $p_s(s_t)$
    forward $\{s_t, p_s(s_t), (d_i, d_j)\}$ to DM_u

Fault selection phase:

run model synchronization phase for only those path-vertices V_p
  for which `state-of`(V_p) =`OBSERVED`
compute $bel(v_i)$ for every vertex V_i, $v_i \in \{0, 1\}$
run fault selection phase of Algorithm 1

To complete the description of GM's algorithm, it remains to explain the calculation of $p_s(s_t)$. Value $p_s(s_t)$ sent to DM_u along with symptom s_t is defined as the probability that the observed inter-domain symptom in path set s_p:$d_i \xrightarrow{*} d_j$ is caused by links or domains other than domain $\mathcal{D}_u$. In Pearl's probabilistic inference, probability that $V_l = 1$ causes $V_p = 1$ is represented by $c_{V_l,V_p}\pi_{V_l}(v_p = 1)$ (Eq. 2) received by vertex V_p. Thus, assuming that vertex V_u is the one labeled with D:d_u and $Par(V_p)$ denotes the set of all parents of V_p, $p_s(s_t)$ may be calculated using the following equation:

$$p_s(s_t) = 1 - \prod_{V_l \in Par(V_p); V_l \neq V_u} \Big(1 - c_{V_l,V_p}\pi_{V_l}(v_p = 1)\Big) = 1 - \frac{1 - \pi(v_p = 1)}{1 - c_{V_u,V_p}\pi_{V_u}(v_p = 1)} \quad (6)$$

3.3 Domain manager's fault localization algorithm

Domain manager in domain $\mathcal{D}_i$ receives symptoms from two sources: from nodes in $\mathcal{D}_i$ and from the global manager. When a symptom from the global manager is observed, DM_i first updates the model as described in Section 3.1. Then it marks the symptom vertex as observed outside domain $\mathcal{D}_i$ and runs one inference iteration starting from this vertex. When an internal symptom is observed it overrides any previous external observations of the same symptom. The causal relationships between the symptom vertices and vertices representing causes outside of $\mathcal{D}_i$ are removed, as the symptom is now known to have been caused by a fault in domain $\mathcal{D}_i$. Until the internal symptom is cleared, no future external observations of the same symptom are taken into account.

Algorithm 3 (Domain Manager's Fault Localization Algorithm for Domain $\mathcal{D}_i$)
Symptom analysis phase:

for every observed symptom s_p:$d_i.n_k \xrightarrow{*} d_i.n_l$
  let V_p *be the vertex labeled with* s_p:$d_i.n_k \xrightarrow{*} d_i.n_l$
  if s_p *is received from GM in message* $\{s_p, p_s(s_p), (d_u, d_j)\}$
    if `state-of`(V_p) =`OBSERVED-INTERNAL` *then continue /* ignore* s_p **/*
    assign weight $p_s(s_p)$ *to edge* (V_s, V_p)*, where* V_s *is labeled with* s_s:$d_u \xrightarrow{*} d_j$
    set `state-of`(V_p) =`OBSERVED-EXTERNAL` *and* $V_p = 1$
    run inference iteration starting from V_p
  *else /** s_p *is internal */*
    assign weight 0 to every edge (V_s, V_p)*, where* V_s *is not a link vertex in* $\mathcal{D}_i$
    set `state-of`(V_p) =`OBSERVED-INTERNAL` *and* $V_p = 1$
    run inference iteration starting from V_p

Fault selection phase: *identical to Algorithm 1*

In addition to the above algorithm, DM_i calculates $p_c(d_i, d_j, d_u)$ required by the GM in model synchronization phase of its algorithm (Algorithm 2). Let $\mathcal{G}^i_{i,j}$ be a set of border nodes in $\mathcal{D}_i$ which are used as gateways accepting traffic from $\mathcal{D}_j$, and $\mathcal{G}^o_{i,u}$ be a set of gateways in $\mathcal{D}_i$ which forward traffic to $\mathcal{D}_u$. We define $I\mathcal{D}_i(d_j, d_u)$ as the set of intra-$\mathcal{D}_i$ paths whose failure may affect communication between $\mathcal{D}_j$ and $\mathcal{D}_u$ using the following equation.

$$I\mathcal{D}_i(d_j, d_u) = \begin{cases} \{d_i.n_k \xrightarrow{*} d_i.n_l | n_l \in \mathcal{G}^o_{i,u}, n_k \in \mathcal{D}_i\} & \text{if } d_j = d_i \\ \{d_i.n_k \xrightarrow{*} d_i.n_l | n_k \in \mathcal{G}^i_{i,j}, n_l \in \mathcal{D}_i\} & \text{if } d_u = d_i \\ \{d_i.n_k \xrightarrow{*} d_i.n_l | n_k \in \mathcal{G}^i_{i,j}, n_l \in \mathcal{G}^o_{i,u}\} & \text{otherwise} \end{cases} \quad (7)$$

Function $p_c(d_i, d_j, d_u)$ is interpreted as the probability that a failure occurs on at least one path in $I\mathcal{D}_i(d_j, d_u)$, which is independent of any external symptom observations. We also define $Child_I(V_l)$ as the set of all children of V_l whose `state-of`()= `OBSERVED-INTERNAL` and V_L as the set of all link vertices in $\mathcal{D}_i$. Function $p_c(d_i, d_j, d_u)$ is calculated using the following equations.

$$p_c(d_u, d_i, d_j) = 1 - \prod_{V_l \in V_L} \Big(1 - bel^*(v_l)(1 - \prod_{V_p \in I\mathcal{D}_u(d_i, d_j)} q_{V_l, V_p})\Big) \quad (8)$$

$$bel^*(v_l) = \alpha \lambda^*(v_l)\pi(v_l) \quad (9)$$

$$\lambda^*(v_l) = \prod_{V_c \in Child_I(V_l)} \lambda_{V_c}(v_l) \quad (10)$$

4 Examples

In this section, we illustrate the algorithms proposed in this paper with examples using the network presented in Fig. 2. We also assume that fault localization models of the GM and DM_2 are those presented in Figs. 4 and 3. Belief network models of DM_0 and DM_1 are not shown.

4.1 Inter-domain link failure example

In the first example, we consider a scenario in which a fault occurs in inter-domain link $1.24 \rightarrow 2.15$. As a result, three inter-domain symptoms are observed indicating failures of paths $1.20 \overset{*}{\rightarrow} 2.16$, $1.21 \overset{*}{\rightarrow} 0.26$, and $1.23 \overset{*}{\rightarrow} 2.19$. When the failure of path $1.20 \overset{*}{\rightarrow} 2.16$ is observed, the GM runs one iteration of belief updating starting from node labeled s_3:$1 \overset{*}{\rightarrow} 2$. This process identifies link $1.24 \rightarrow 2.15$ as a possible cause of the observed path failure. At the same time, the GM delegates the diagnosis of intra-domain path segments which may also be responsible for the failure of $1.20 \rightarrow 2.16$ to domain managers DM_1 and DM_2 by sending messages $\{1.20 \overset{*}{\rightarrow} 1.24, 0.003, (1, 2)\}$ and $\{2.15 \overset{*}{\rightarrow} 2.16, 0.003, (1, 2)\}$, respectively. Table 1 presents the list of inter- and intra-domain links whose $bel(1)$ exceeds 0.1 after this operation. It shows that neither DM_1 nor DM_2 is able to single out a fault meeting this criterion after the first symptom observation. The results of subsequent symptoms' analysis are presented in Table 1. Observe that the third symptom ($1.23 \overset{*}{\rightarrow} 2.19$) is ignored by the GM as it belongs to the same path set as one of the symptoms previously analyzed ($1.20 \overset{*}{\rightarrow} 2.16$). Observe also that, in this example, the model synchronization phase does not contribute any changes to the results of GM's fault diagnosis process, since no intra-domain symptoms have been observed.

Table 1: Fault diagnosis process of fault f_1:$1.24 \rightarrow 2.15$

	GM	DM_0	DM_1	DM_2
Symptom analysis phase:				
Path failure	$1.20 \overset{*}{\rightarrow} 2.16$			
Symptoms	$1 \overset{*}{\rightarrow} 2$		$1.20 \overset{*}{\rightarrow} 1.24$	$2.15 \overset{*}{\rightarrow} 2.16$
Suspect faults	$1.24 \rightarrow 2.15$			
Path failure	$1.21 \overset{*}{\rightarrow} 0.26$			
Symptoms	$1 \overset{*}{\rightarrow} 0$	$0.25 \overset{*}{\rightarrow} 0.26$	$1.21 \overset{*}{\rightarrow} 1.24$	$2.15 \overset{*}{\rightarrow} 2.19$
Suspect faults	$1.24 \rightarrow 2.15$			
Path failure	$1.23 \overset{*}{\rightarrow} 2.19$			
Symptoms, $p_s()$	*ignored*		$1.23 \overset{*}{\rightarrow} 1.24$	$2.15 \overset{*}{\rightarrow} 2.19$
Suspect faults, $bel()$	$1.24 \rightarrow 2.15$			
Model synchronization phase:				
Suspect faults, $bel()$	$1.24 \rightarrow 2.15$			
Fault selection phase:				
Identified faults	$1.24 \rightarrow 2.15$			

4.2 Intra-domain link failure

In the second example, we consider a scenario in which a fault occurs in intra-domain link $2.15 \rightarrow 2.19$. Since this link is used by the backbone route between domains 1 and 0, inter-domain symptoms may be generated as a result of the failure. The complete sequence of symptoms generated in the scenario and their diagnosis process are presented in Table 2. Before the model synchronization phase, confidence associated with failures of links $1.24 \rightarrow 2.15$, $2.19 \rightarrow 0.25$, and $2.15 \rightarrow 2.19$, is 0.29, 0.80, and 0.99, respectively. Thus, without this phase, two faults would be chosen by the algorithm: $2.19 \rightarrow 0.25$, and $2.15 \rightarrow 2.19$. Updating the model with the information learned by DM_2 from the internal symptoms it observed allows fault $2.19 \rightarrow 0.25$ to be eliminated.

Table 2: Fault diagnosis process of fault f_2:2.15→2.19

	GM	DM_0	DM_1	DM_2
Symptom analysis phase:				
Path failure	$1.22 \xrightarrow{*} 0.28$			
Symptoms	$1 \xrightarrow{*} 0$	$0.25 \xrightarrow{*} 0.28$	$1.22 \xrightarrow{*} 1.24$	$2.15 \xrightarrow{*} 2.19$
Suspect faults	$1.24 \rightarrow 2.15$			
	$2.19 \rightarrow 0.25$			
Path failure				$2.17 \xrightarrow{*} 2.19$
Suspect faults	$1.24 \rightarrow 2.15$			$2.17 \rightarrow 2.15$
	$2.19 \rightarrow 0.25$			$2.15 \rightarrow 2.19$
Path failure	$1.23 \xrightarrow{*} 2.18$			
Symptoms	$1 \xrightarrow{*} 2$		$1.23 \xrightarrow{*} 1.24$	$2.15 \xrightarrow{*} 2.18$
Suspect faults	$1.24 \rightarrow 2.15$			$2.15 \rightarrow 2.19$
	$2.19 \rightarrow 0.25$			
Path failure	$2.15 \xrightarrow{*} 0.28$			
Symptoms	$2 \xrightarrow{*} 0$	$0.25 \xrightarrow{*} 0.28$		$2.15 \xrightarrow{*} 2.19$
Suspect faults	$1.24 \rightarrow 2.15$			$2.15 \rightarrow 2.19$
	$2.19 \rightarrow 0.25$			
Path failure				$2.15 \xrightarrow{*} 2.18$
Suspect faults	$1.24 \rightarrow 2.15$			$2.15 \rightarrow 2.19$
	$2.19 \rightarrow 0.25$			
Model synchronization phase:				
Suspect faults				$2.15 \rightarrow 2.19$
Fault selection phase:				
Identified faults				$2.15 \rightarrow 2.19$

5 Preliminary Simulation Study

This section presents the preliminary results of a simulation study designed to verify concepts in this paper. We use Brite network topology generator [12] to build random, two-level network topologies similar to those of the Internet. Then we build a belief network representing relationships among end-to-end and hop-to-hop services in this communications network assuming a shortest-path routing algorithm. In the belief network, prior link failure probabilities are randomly generated as uniformly distributed random variables over the range $(0.0001, 0.001)$. Conditional probabilities in domain managers' models are randomly generated from range $(0, 1)$. We assume that all intra-domain path services are observable, i.e., if as a result of a fault a path failure occurs, the corresponding symptom is always observed by the DM. The observability ratio for inter-domain paths is related to the number of network domains. We vary the number of domains between 10 and 25, using observability ratios of 2% and $.5\%$, respectively. We vary the size of network domains from 5 to 25 nodes. Thus the overall experiment covers networks consisting of 50-1250 nodes.

The test scenarios are generated using the belief network model built by the managers. This technique of generating scenarios assumes that the fault propagation model accurately represents relationships among faults and symptoms. Two performance metrics are calculated: detection rate DR defined as a percentage of faults occurring in the network which are isolated by the technique, and false positive rate FPR defined as a percentage of faults reported by the technique that are not occuring in the network. The preliminary results of this simulation study are shown in Table 3.

We distinguish three types of experiments: those involving only intra-domain link failures, those

Table 3: Simulation study results (DR – detection rate, FPR – false positive rate)

No. of Domains	Nodes per domain	Total nodes	Symptom types: Inter-domain DR	Inter-domain FPR	Intra-domain DR	Intra-domain FPR	Mixed DR	Mixed FPR
10	5	50	.80	.06	.95	.04	.70	.06
	10	100	.90	.02	.95	.04	.80	.04
	15	150	.95	.02	.95	.04	.90	.02
	20	200	.90	.00	.95	.02	.85	.02
	25	250	.85	.00	.90	.02	.70	.02
50	5	250	.80	.06	.95	.06	.85	.06
	10	500	.95	.04	.90	.06	.85	.06
	15	750	.95	.04	.90	.04	.75	.04
	20	1000	1.0	.02	.85	.04	.70	.02
	25	1250	1.0	.02	.85	.02	.65	.02

involving only inter-domain link failures, and those involving both types of failures. Clearly, the mixed-failure scenarios are the most difficult to diagnose since they always involve at least two concurrent faults and the interpretation of their symptoms, which may overlap, leads to ambiguity. Irrespective of the scenario type used, we observe the relationship between the network topology size and the fault localization accuracy achievable with the distributed algorithm. This observation is consistent with the results of the simulation study utilizing the centralized algorithm [21]. This study shows that as the network size grows, as a result of the increasing number of possible failure suspects, the probability of proposing a highly probable, but incorrect or partly correct solution increases.

6 Related work

Many researchers have recognized the importance of distributed fault localization [1, 9, 23]. However, few distributed fault localization techniques have actually been proposed. The theoretical foundation for the design of such systems has been laid by Bouloutas et al. [1] and Katzela et al. [9], who investigate different schemes of non-centralized fault localization: decentralized and distributed schemes. The technique proposed in this paper has properties of both these schemes. Similarly to the decentralized scheme [9], we envision a hierarchy of managers with a central manager (GM) making the final fault determination. Unlike in the decentralized scheme [9], however, the GM not only arbitrates among solutions proposed by the domain managers, but also participates in the actual fault determination by proposing its own hypothesis composed of network faults that cannot be identified by the domain managers.

This paper utilizes Pearl's belief updating [14] as an approximation scheme [21, 20] in fault localization performed by the managers on all layers of the hierarchy. Other non-deterministic fault localization algorithms could be considered for this purpose: maximum mutual dependency heuristics [10], statistical methods [5], or the incremental algorithm proposed in [18]. Of those the maximum mutual dependency heuristics [10] would be difficult to apply to the problem of end-to-end service failure diagnosis as it relies on the causal relationships among network faults, while in end-to-end services model, all host-to-host services are independent of one another.

Belief networks have previously been applied to the problem of fault diagnosis, but the reported solutions are limited to rather narrow applications [4, 6, 22]. These solutions either assume a tree-shaped belief network model [22] or disregard uncertainty involved in causal relationships between

faults symptoms, i.e., conditional probabilities are 0,1-values [4, 6]. The approach proposed in this paper is more general in this respect.

7 Conclusion and future work

This paper introduces a distributed non-deterministic fault localization algorithm suitable for the diagnosis of end-to-end service problems in communication systems. It builds upon the previously proposed centralized algorithm [21], and increases the admissible network size by an order of magnitude. The algorithm divides the computational effort and system knowledge involved in end-to-end service failure diagnosis among multiple, hierarchically organized managers. With such an organization, the technique is suitable for end-to-end service failure diagnosis in networks with hierarchical topologies, including IP networks.

Future work will involve several important improvements to the proposed technique. The accuracy of the algorithm needs to be increased in scenarios involving mixed types of faults. The performance of the algorithm may be further improved by having the GM delegate the fault diagnosis task to the managers of only those domains that are the most likely to contain a faulty link. The theoretical analysis of signaling overhead of the algorithm is also required. Finally, an extensive simulation study will be conducted.

References

[1] A. T. Bouloutas, S. B. Calo, A. Finkel, and I. Katzela. Distributed fault identification in telecommunication networks. *Journal of Network and Systems Management*, 3(3), 1995.

[2] Y. Breitbart, M. Garofalakis, C. Martin, R. Rastogi, S. Seshadri, and A. Silberschatz. Topology discovery in heterogeneous IP networks. In *Proc. of IEEE INFOCOM*, 2000, pp. 265–274.

[3] G. F. Cooper. Probabilistic inference using belief networks is NP-Hard. Technical Report KSL-87-27, Stanford University, 1988.

[4] R. H. Deng, A. A. Lazar, and W. Wang. A probabilistic approach to fault diagnosis in linear lightwave networks. In Hegering and Yemini [7], pp. 697–708.

[5] M. Fecko and M. Steinder. Combinatorial designs in multiple faults localization for battlefield networks. In *IEEE Military Commun. Conf. (MILCOM)*, McLean, VA, 2001.

[6] D. Heckerman and M. P. Wellman. Bayesian networks. *Communications of the ACM*, 38(3):27–30, Mar. 1995.

[7] H. G. Hegering and Y. Yemini, eds. *Integrated Network Management III*. North-Holland, Apr. 1993.

[8] G. Jakobson and M. D. Weissman. Alarm correlation. *IEEE Network*, 7(6):52–59, Nov. 1993.

[9] I. Katzela, A. T. Bouloutas, and S. Calo. Comparison of distributed fault identification schemes in communication networks. Technical Report RC 19630 (87058), T. J. Watson Research Center, IBM Corp., Sep. 1993.

[10] I. Katzela and M. Schwartz. Schemes for fault identification in communication networks. *IEEE Transactions on Networking*, 3(6):733–764, 1995.

[11] S. Kliger, S. Yemini, Y. Yemini, D. Ohsie, and S. Stolfo. A coding approach to event correlation. In Sethi et al. [15], pp. 266–277.

[12] A. Medina, A. Lakhina, I. Matta, and J. Byers. BRITE: Universal topology generation from a user's perspective. Technical Report 2001-003, 1 2001.

[13] M. Novaes. Beacon: A hierarchical network topology monitoring system based in IP multicast. In A. Ambler, S. B. Calo, and G. Kar, eds, *Services Management in Intelligent Networks*, no. 1960 in Lecture Notes in Computer Science. Springer-Verlag, 2000, pp. 169–180.

[14] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, 1988.

[15] A. S. Sethi, F. Faure-Vincent, and Y. Raynaud, eds. *Integrated Network Management IV*. Chapman and Hall, May 1995.

[16] R. Siamwalla, R. Sharma, and S. Keshav. Discovering Internet topology. Technical report, Cornell University, 1998.

[17] M. Steinder and A. S. Sethi. Non-deterministic fault localization in communication systems using belief networks. Under preparation for journal submission.

[18] M. Steinder and A. S. Sethi. Non-deterministic diagnosis of end-to-end service failures in a multi-layer communication system. In *Proc. of ICCCN*, Scottsdale, AZ, 2001. pp. 374–379.

[19] M. Steinder and A. S. Sethi. The present and future of event correlation: A need for end-to-end service fault localization. In N. Callaos et al., ed, *World Multi-Conf. Systemics, Cybernetics, and Informatics*, vol. XII, Orlando, FL, 2001. pp. 124–129.

[20] M. Steinder and A. S. Sethi. Increasing robustness of fault localization through analysis of lost, spurious, and positive symptoms. In *Proc. of IEEE INFOCOM*, New York, NY, 2002. (to appear).

[21] M. Steinder and A.S. Sethi. End-to-end service failure diagnosis using belief networks. In *Proc. Network Operation and Management Symposium*, Florence, Italy, 2002.

[22] C. Wang and M. Schwartz. Identification of faulty links in dynamic-routed networks. *Journal on Selected Areas in Communications*, 11(3):1449–1460, Dec. 1993.

[23] S. A. Yemini, S. Kliger, E. Mozes, Y. Yemini, and D. Ohsie. High speed and robust event correlation. *IEEE Communications Magazine*, 34(5):82–90, 1996.