Increasing robustness of fault localization through analysis of lost, spurious, and positive symptoms

Małgorzata Steinder, Adarshpal S. Sethi

Abstract—**This paper utilizes belief networks to implement fault localization in communication systems taking into account comprehensive information about the system behavior. Most previous work on this subject performs fault localization based solely on the information about malfunctioning system components (i.e., negative symptoms). In this paper, we show that positive information, i.e., the lack of any disorder in some system components, may be used to improve the accuracy of this process. The technique presented in this paper allows lost and spurious symptoms to be incorporated in the analysis. We show through simulation that in a noisy network environment the analysis of lost and spurious symptoms increases the robustness of fault localization with belief networks. We also demonstrate that belief networks yield high accuracy even for approximate probability input data and therefore are a promising model for non-deterministic fault localization.**[1]

Index Terms—**Fault localization, belief networks**

I. Introduction

FAULT localization in communication systems is a process of isolating root causes of a system disorder based on the observed indications of the disorder (symptoms). In the past, fault localization focused on diagnosing low-level resource availability-related problems such as a broken cable or an inactive interface. Recently, the scope of fault localization has been expanded to include the diagnosis of performance problems in higher layers of the communication system, such as the transport and application layers [1], [2]. For this purpose, non-deterministic reasoning needs to be incorporated in the fault localization process [3].

This paper utilizes belief networks [4] to perform fault localization in communication systems whose failure propagation model may be described by bipartite causality or dependency graphs. As explained in Section II, bipartite graphs may be used to describe a wide range of fault localization problems.

The paper expands on our previous research on applying belief networks techniques to fault localization [5] (Section IV) by taking into account comprehensive information about the system behavior. Most previous work on this subject [6], [7] performs fault localization based solely on the information about malfunctioning system components (i.e., negative symptoms). In this paper, we show that positive information, i.e., the lack of any disorder in some system components, may be used to improve the accuracy of fault localization (Section V).

In real-life communication systems, an observation of network state is frequently disturbed by the presence of lost and/or spurious symptoms (usually referred to as observation noise). Although many researchers have suggested [6], [8] that fault localization should be resilient to the existence of spurious and/or lost alarms, we are aware of only one technique [8] that incorporates lost and spurious symptoms into deterministic fault localization. In this paper, we propose a technique that allows lost and spurious symptoms to be incorporated in the non-deterministic analysis (Section VI). We prove that reasoning with positive and noisy observations does not increase a theoretical complexity bound of the algorithms introduced in [5]. We also show through simulation that in a noisy network environment the analysis of lost and spurious symptoms increases the robustness of fault localization with belief networks (Section VII). Moreover, we demonstrate that belief networks yield high accuracy even for approximate probability input data and therefore are a promising model for non-deterministic fault localization (Section VIII).

II. Bipartite models of network faults

Fault localization in communication systems distinguishes different types of events. Faults – root problems of system disorder – are those events that need to be handled directly to eliminate the undesirable system behavior. A failure is a discrepancy between the observed and expected system states or behaviors. It results from a fault and may not be directly corrected. When a failure is detected by the management system, an alarm is generated. Alarms delivered to the management console are called symptoms. In this paper, we denote by $\mathcal{F}$ the set of all faults that may occur in the considered system. We use $\mathcal{S}$ to denote the set of all possible symptoms, and $\mathcal{S}_O$ to denote the set of all observed symptoms. All network events (including faults and symptoms) will be denoted by $\mathcal{E}$. In general, $\mathcal{F} \cap \mathcal{S} \neq \emptyset$.

In a communication system, failures propagate among system components. A failure of a system entity that occurs in layer L on host H may affect a dependent entity in layer $L+1$ on host H (and recursively, in all layers above). This failure propagation pattern is referred to as vertical propagation. A failure of a system entity that occurs in layer L on host H may also affect a system entity in layer L on host H' that communicates with host H (horizontal propagation). Fault management systems model fault propagation by representing either causal relationships among network events [7], [8], or dependencies among communication system entities [1], [6], [9].

In the past, researchers focusing on fault localization frequently assumed a deterministic fault propagation model, in

Computer and Information Sciences Department, University of Delaware, Newark, DE, {steinder,sethi}@cis.udel.edu

[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.

which one can predict with certainty whether a failure of an entity will cause a failure of a dependent entity [7], [8]. The deterministic model is usually sufficient to model availability-related faults (such as a broken cable or an inactive interface), since their impact on other network components is easily predictable. Recently, more attention has been devoted to diagnosing performance related failures in higher layers of the protocol stack including the transport and application layers [1], [2], [3]. In this application, the deterministic model is not sufficient because the impact of performance-related failures in one entity on dependent entities is not easy to predict. Also, the dependencies among system entities change so frequently that they may not be deterministically captured by the fault-propagation model. Non-determinism is introduced into the fault propagation model by labeling graph edges with probabilities.

Since relationships among system entities tend to be very complex, so is the fault propagation model that captures them. Reasoning about faults with such complex models has been shown to be NP-hard [6]. To deal with the complexity, researchers develop heuristics that allow the problem to be simplified. One simplification involves reducing a complex graph to a bipartite graph by computing its transitive closure using serial and parallel reduction operators. [2] Such a simplification has been applied in the popular codebook approach [8].

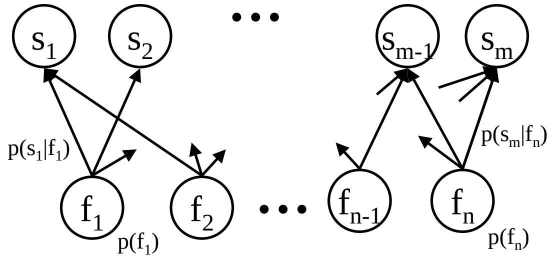

Fig. 1. A bipartite causality graph

The fault localization problem may be also simplified by dividing it into smaller sub-problems, each focusing on a subgraph of the original model, which typically represents a level of abstraction. In this approach, a solution to one sub-problem is fed as input to another, typically lower-level, sub-problem. For some of these sub-problems, the fault propagation model may be represented by a bipartite graph. As an example, 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 [3], [10]. End-to-end service failure diagnosis uses a bipartite graph, whose parentless nodes represent host-to-host failures (root problems), and childless nodes represent the resultant end-to-end service failures (symptoms). To build such a graph, the knowledge of network topology and current routing information are required.

In our work on fault localization [5], [10], we have been using the latter approach to simplifying the fault localization task, because it allows the system to respond easily to the system configuration changes. On the other hand, when the original fault propagation model is reduced to a bipartite graph, every configuration change necessitates repeating the reduction.

In the following sections, we present an algorithm that utilizes belief networks to calculate the most probable explanation of the observed set of symptoms. The algorithm allows the solution to be built in an event-driven fashion based on both positive and negative information about the system state. The technique is resilient to the inaccuracies in the symptom data (such as lost and spurious symptoms). In addition, as shown in Section VIII, the technique does not require the accurate knowledge of symptom-given-fault probability distributions.

III. Belief Network Concepts

A *belief network* [11], [4] is a directed acyclic graph [11] (DAG), in which each node represents a random variable over a multivalued domain. We will use terms “node” and “random variable” interchangeably, and denote them by V_i. The set of all nodes is denoted by V. The domain of random variable V_i will be denoted by symbol D_i. The set of directed edges E represents causal relationships among the variables, and the strengths of these influences are specified by conditional probabilities. Formally, a belief network is a pair (G, P), where G is a DAG, $P{=}\{P_i\}$, and P_i is the conditional probability matrix associated with a random variable V_i. Let $Par(V_i){=}\{V_{i_1}, \ldots, V_{i_n}\}$ be the set of all parents of V_i. P_i is a $(|Par(V_i)|{+}1)$-dimensional matrix of size $|D_i|{\times}|D_{i_1}|{\times}\ldots{\times}|D_{i_n}|$, where $P_i(v_i, v_{i_1}, \ldots, v_{i_n}){=}P(V_i{=}v_i|V_{i_1}{=}v_{i_1}, \ldots, V_{i_n}{=}v_{i_n})$. We will denote by $\mathcal{A}{=}\{V_1{=}v_1,\ldots,V_n{=}v_n\}$ an assignment of values to variables in set V, where each $v_j{\in}D_j$. We will use $v_j^{\mathcal{A}}$ to denote the value of variable $V_j \in V$ in assignment $\mathcal{A}$. Given a subset of random variables $U_k{=}\{V_{k_1}, \ldots, V_{k_m}\}{\subseteq}V$, we will denote by $U_k^{\mathcal{A}}{=}\{V_{k_1}{=}v_{k_1}^{\mathcal{A}}, \ldots, V_{k_m}{=}v_{k_m}^{\mathcal{A}}\}$ an assignment of values to variables in set U_k that is consistent with assignment $\mathcal{A}$. An evidence set e is an assignment $U_o^{\mathcal{A}}$, where $U_o{\subseteq}V$ is a set of variables whose values are known, and for each $V_{o_j}{\in}U_o$, $v_{o_j}^{\mathcal{A}}$ is its observed value.

Belief networks are used to make four basic queries given evidence set e: (1) belief assessment, (2) most probable explanation, (3) maximum a posteriori hypothesis, and (4) maximum expected utility [11]. The first two queries are of particular interest in the presented research. The *belief assessment* task is to compute $bel(V_i{=}v_i){=}P(V_i{=}v_i|e)$ for one or more variables V_i. The *most probable explanation* (MPE) task is to find an assignment $\mathcal{A}_{\max}$ that best explains the observed evidence e, i.e., $P(\mathcal{A}_{\max}){=}\max_{\mathcal{A}} \Pi_{i=1}^{n} P(V_i{=}v_i^{\mathcal{A}}|Par(V_i)^{\mathcal{A}})$ [11]. It is known that these tasks are NP-hard in general belief networks [12]. Moreover, some approximation schemes have been proven to be NP-hard [13]. A belief-updating algorithm, polynomial with respect to $|V|$, is available for *polytrees*, i.e., directed graphs without undirected cycles [4]. However, in unconstrained polytrees, the propagation algorithm still has an exponential bound with respect to the number of a node’s neighbors.

In this paper, we focus on a class of belief networks representing a simplified model of conditional probabilities called

[2] Note that, such a reduction is, in general, an NP-complete problem on its own. Frequently, an assumption is made with respect to the graph shape that allows the reduction process to be completed in polynomial time.

noisy-OR gates [4]. The simplified model contains binary-valued random variables. The noisy-OR model associates an inhibitory factor with every cause of a single effect. When some causes are present, the effect is absent only if all inhibitors corresponding to these causes are activated. The model assumes that all inhibitory mechanisms are independent [4]. This assumption of independence is ubiquitous in probabilistic fault localization approaches reported in the literature [6], [14]. It indicates that all alternative causes of the same effect are independent. This simplification helps avoid exponential time and memory otherwise needed to process and store conditional probability matrices associated with random variables in the belief network. Furthermore, belief assessment in polytrees with the noisy-OR model has polynomial complexity, which makes it attractive to use with our problem as an approximation.

Noisy-OR model may be insufficient for some fault localization problems. In some cases, a model may be more suitable that introduces the AND relationship between event causes, or the inverse causal relationships among events. Such models are easily incorporated into the belief network algorithms and allow the Bayesian inference to be equally efficient. However, this paper focuses on the noisy-OR model as the most frequent model in fault localization.

The noisy-OR fault propagation model may be built based on the knowledge of the dependencies between system components and their associated faults and symptoms. Techniques of building a fault propagation model based on this information were proposed in [14] and [5]. Building fault propagation models may be automated by applying belief network learning methods [15]. These methods being very complex, the design of such automated techniques remains an open problem.

IV. Iterative belief propagation in fault diagnosis

In this section, we introduce a fault localization algorithm based on iterative propagation in belief networks. Recall from Section III that in singly-connected networks (polytrees) representing the noisy-OR-gate model of conditional probability distribution, Bayesian inference (belief updating) may be computed in polynomial time using the algorithm presented in [4]. However, a bipartite fault propagation model usually contains undirected loops and therefore, is not singly-connected.

Networks with loops violate certain independence assumptions based on which the local computation equations were derived for polytrees. Nevertheless, successful applications of the iterative algorithm to calculating queries in loopy belief networks have been reported [16]. We adapt iterative belief propagation for calculating the most probable explanation of the set of observed symptoms [5].

A. Iterative belief propagation concepts

Iterative belief propagation utilizes a message passing schema in which network nodes exchange λ and π messages (Fig. 2). Message $\lambda_X(v_j)$ that node X sends to its parent V_j for every valid V_j's value v_j, 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 node X sends to its child U_i for every valid value of X, x, denotes probability that $X = x$ given the entire body of evidence in the sub-graph containing X created by removing edge $X \rightarrow U_i$. In a noisy-OR polytree, let us denote by q_{XU_i} the probability of activating the inhibitor controlling link $X \rightarrow U_i$. Every random variable assumes values from $\{0, 1\}$, where 1 denotes occurrence of the corresponding event and 0 means that the event did not occur. The probability that U_i occurs given X occurs is $c_{XU_i} = 1 - q_{XU_i}$. Based on the messages received from its neighbors, node X computes $\lambda(x)$, $\pi(x)$, and $bel(x)$ as follows [4]:

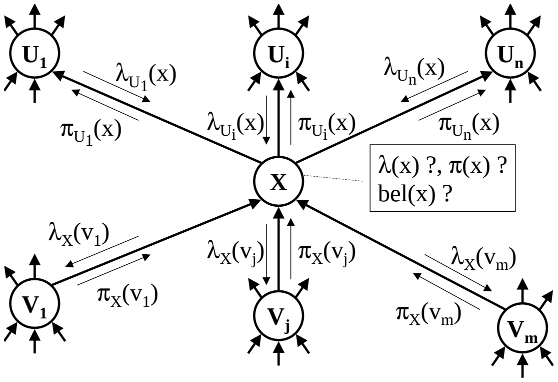

Fig. 2. Message passing in Pearl's belief propagation

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

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

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

In the above equations, $\pi_{jX} = \pi_X(v_j)$ for $v_j = 1$, and α is a normalizing constant. Let β represent any constant. The messages $\lambda_X(v_j)$ and $\pi_{U_i}(x)$ are computed using the following equations [4]:

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

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

In the initialization phase, for all observed nodes X, $\lambda(x)$ is set to 1 if x is the observed value of X. For other values of x, $\lambda(x)$ is set to 0. For all unobserved nodes $\lambda(x)$ is set to 1 for all values of x. Parentless nodes have their $\pi(x)$ set to the prior probabilities.

The belief propagation algorithm in polytrees starts from the evidence node 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 node. The complete description of the iterative algorithm for polytrees including expressions for $\lambda_X(v_j)$, $\pi_{U_i}(x)$, and $bel(x)$ along with some illustrative examples may be found in [4]. In noisy-OR gate belief networks, functions $\lambda_X(v_j)$, $\pi_{U_i}(x)$, and $bel(x)$ may be evaluated in linear time with respect to the number of node X's neighbors. In reported applications of iterative belief updating to loopy graphs, several iterations are performed in which the entire graph is searched according to some pre-defined ordering [16], [17].

B. Application of belief propagation to fault localization

In the domain of fault localization, observations of network disorder (symptoms) are considered as belief network evidence. Symptom observation is represented by assigning the corresponding belief network node to 1, which means that the symptom did occur. Belief network nodes whose corresponding symptoms were not observed are left unassigned (i.e., their $\lambda(0){=}\lambda(1){=}1$). The fault localization task is to find an assignment $\mathcal{F}^{\mathcal{A}_{\max}}$. For $f_i{\in}\mathcal{F}$, assignment $f_i{=}1$ indicates that fault f_i occurred, and $f_i{=}0$ indicates that fault f_i did not occur. The fault localization algorithm introduced in [5] starts with a belief network whose all evidence nodes are unassigned. Then, the algorithm proceeds iteratively, after every symptom observation applying one iteration of belief updating that starts from the observed evidence node and visits all belief network nodes in a breadth-first order. Note that, contrary to other approaches to fault localization [6], [14], [18], which delay symptom analysis until all symptoms are collected, Algorithm 1 does not require all symptoms to be observed before their analysis may be started. On the contrary, it analyzes a symptom independently of other symptom observations. The knowledge resulting from analyzing a symptom is stored for the next iterations in the belief network nodes in the form of λ and π messages, allowing Algorithm 1 to utilize time more efficiently. Moreover, for every fault the algorithm continuously provides the probability of its occurrence given the symptoms observed so far.

At the end of the analysis cycle, i.e., after sufficient number of symptoms have been received, sufficient amount of time elapses, or system administrator decides so, the algorithm creates a complete, possibly multi-fault, hypothesis. In fault selection phase, several belief propagation iterations are performed. In each iteration, the most probable fault is chosen and its corresponding node is assigned to 1. Then, belief propagation is initiated from this node. This process continues until all symptoms are explained or until there are no faults with sufficiently high probability of their occurrences. Observe that an inherent property of Algorithm 1 is the capability to isolate multiple simultaneous faults even if their symptoms overlap. Formally, the fault localization algorithm is defined as follows.

Algorithm 1—MPE through iterative belief updating:

```
Inference iteration starting from node $Y_i$:
  let $o$ be the breadth-first order starting from $Y_i$
  for all nodes $X$ along ordering $o$ do
    if $X$ is not an unobserved path node then
      for all $X$'s parents, $V_j$ and for all $v_j \in \{0,1\}$
        compute $\lambda_X(v_j)$
      for all $X$'s children, $U_i$ and for all $x \in \{0,1\}$
        compute $\pi_{U_i}(x)$
Symptom analysis phase:
  for every symptom $S_i \in \mathcal{S}_O$ do
    mark $S_i$ as observed to have value of 1
    run inference iteration starting from $S_i$
  for every node $V_i$, $v_i \in \{0,1\}$ compute $bel(v_i)$
Fault selection phase:
  while $\exists$ link node $V_j$ for which $bel(1){>}0.5$
      and $\mathcal{S}_O{\neq}\emptyset$ do
      take $V_j$ with the greatest $bel(1)$
      remove all $S_i$ such that $V_j$ may cause $S_i$ from $\mathcal{S}_O$
      mark $V_j$ as observed to have value of 1
      run inference iteration starting from $V_j$
      for every node $V_i$, $v_i \in \{0,1\}$ compute $bel(v_i)$
```

The worst-case computational complexity of Algorithm 1 is $\mathcal{O}(|\mathcal{S}_{\mathcal{O}}|\max(|\mathcal{F}|d_{\mathcal{F}}, |\mathcal{S}|d_{\mathcal{S}})) \subseteq \mathcal{O}(|\mathcal{S}|\max(|\mathcal{F}|\, d_{\mathcal{F}}, |\mathcal{S}|\, d_{\mathcal{S}}))$, where $d_{\mathcal{F}}$ is the maximum out-degree of a fault node, and $d_{\mathcal{S}}$ is the maximum in-degree of a symptom node. In [5], we applied Algorithm 1 to the problem of end-to-end service failure diagnosis, and proved that the algorithm's complexity in a network composed of n nodes (such as routers, bridges, or switches) is $\mathcal{O}(n^5)$. Algorithm 1 was evaluated through simulation on a randomly generated set of spanning tree networks, and compared to the accurate, but exponential, bucket-tree elimination algorithm [11]. Algorithm 1 proved to significantly outperform the optimal algorithm in terms of its running time while preserving almost optimal accuracy (using Algorithm 1 we are able to detect 1-2% fewer faults than with the optimal algorithm).

This paper discusses difficulties involved in applying Algorithm 1 to a real-life network environment and presents solutions that allow the accuracy and robustness of iterative belief updating in such an environment to be improved. The solutions presented in this paper preserve the computational complexity of Algorithm 1.

V. Analysis of positive information

Algorithm 1 presented in Section IV calculates the most probable fault hypothesis based on the observed indications of network disorder. It does not take into account that some possible indications of network disorder have not been observed. As many researchers point out [8], [19], the fact that many of its possible symptoms have not been observed should decrease our confidence in the fault occurrence. In the realm of fault localization, an observation of network disorder is called a *negative symptom*. The lack of such an observation, or observation otherwise are considered *positive symptoms*. The inclusion of positive symptoms into the fault localization process may allow a more accurate fault hypothesis to be created.

In a belief network, a positive symptom is represented by assigning 0 to the symptom's corresponding belief network node. In the extreme case, all belief network nodes which represent symptoms could be assigned 0 if their corresponding alarms were not observed, and they could be assigned 1 if their corresponding alarms were observed. This approach is valid if all potential alarms included in the fault propagation model are *observable*. However, whether or not a potential alarm is observable may depend on a current configuration of the management system, which may change the set of observable alarms according to current management system objectives. Thus, a more general approach should allow some of the potential alarms to be unobservable, and it should allow the set of observable alarms to be easily modified.

As an example, consider the problem of end-to-end service failure diagnosis. To detect end-to-end service failures, the management system may utilize one or more monitoring nodes which monitor the availability and quality of end-to-end connections between a monitoring node and a chosen set of other network nodes. If such a scenario is applied, the observable set of alarms includes those triggered by failures of end-to-end services between a monitoring node and any node belonging to the set of chosen nodes. The set of chosen nodes and the set of monitoring nodes may change during system operation.

Let $\mathcal{S}_O$ be the set of all observable alarms. The ratio $|\mathcal{S}_O|/|\mathcal{S}|$ will be called *observability ratio* (OR). We will denote by $\mathcal{S}_N$ and $\mathcal{S}_P$ the sets of all negative and positive symptoms, respectively, where $\mathcal{S}_N \cup \mathcal{S}_P = \mathcal{S}_O$. Note, that in Algorithm 1 we assumed that $\mathcal{S}_N = \mathcal{S}_O$, and $\mathcal{S}_P = \emptyset$.

To include the analysis of positive symptoms in the fault localization process, we enhance Algorithm 1 as follows. Initially, all observable alarms are considered positive symptoms, and their corresponding belief network nodes are assigned 0. The observation of a negative symptom is represented by changing the assignment of the corresponding belief network node to 1.

Algorithm 2—MPE including positive symptoms:

```
Initialization phase:
    for every symptom $S_i \in \mathcal{S}_P$ do
        mark $S_i$ as observed to have value of 0
        run inference iteration starting from $S_i$
            as defined in Algorithm 1
Symptom analysis phase:
    run symptom analysis phase of Algorithm 1
        substituting $\mathcal{S}_N$ for $\mathcal{S}_O$
Fault selection phase:
    run fault selection phase of Algorithm 1
        substituting $\mathcal{S}_N$ for $\mathcal{S}_O$
```

The worst-case computational complexity of Algorithm 2 is $\mathcal{O}(|\mathcal{S}_O| \max(|\mathcal{F}| d_{\mathcal{F}}, |\mathcal{S}| d_{\mathcal{S}}))$, which is consistent with the computational complexity of Algorithm 1. However, the actual run-time of Algorithm 2 may be significantly higher than that of Algorithm 1, because $|\mathcal{S}_N|$ ($=|\mathcal{S}_O|$ in Algorithm 1) is typically much smaller than $|\mathcal{S}_O|$ used in Algorithm 2. The run-time of Algorithm 2 may be reduced by making two observations.

1) For an unobservable symptom node X, and for any parent node of X, V_j, $\lambda_X(V_j{=}1){=}\lambda_X(V_j{=}0){=}\lambda(X{=}1){=}1$. Since $\lambda_X(v_j)$ does not depend on the received values of functions $\pi_X(v_k)$, node X does not propagate evidence between its parents. As a result, the iterative belief updating need not continue past an encountered unobservable symptom node.
2) For a positive symptom node X, and for any parent node of X, V_j:
$$\lambda_X(v_j) = \beta q_{V_j X}^{v_j} \prod_{k \neq j} (1 - c_{V_k X} \pi_{kX})$$
Since β is any constant, assignment $\beta{=}\alpha{=}1/(\lambda_X(V_j{=}1) + \lambda_X(V_j{=}0))$ leads to the following equation.
$$\lambda_X(v_j) = \begin{cases} 1/(1 + q_{V_j X}) & \text{if } v_j = 0 \\ q_{V_j X}/(1 + q_{V_j X}) & \text{if } v_j = 1 \end{cases}$$
Since $\lambda_X(v_j)$ does not depend on the received values of functions $\pi_X(v_k)$, node X does not propagate evidence between its parents. As a result, the iterative belief updating need not continue past a positive symptom node.

Note that, initially, the belief network contains solely unobservable and positive symptom nodes. Thus, the initialization phase may be reduced to calculating $\lambda_X(v_j)$'s for all symptom nodes, and then computing λ and λ_X values in the belief network nodes which correspond to faults. Moreover, the set of observable alarms may be easily modified during the fault localization process by redefining the values of function λ of nodes whose observability status has changed.

In Section VII we describe the application of Algorithm 2 to the problem of diagnosing end-to-end service failures, and evaluate benefits resulting from the inclusion of positive symptoms in the fault localization process.

VI. Dealing with noisy observations

In real-life communication systems, an observation of network state is frequently disturbed by the presence of lost and/or spurious symptoms (usually referred to as observation noise).

In a management system, alarms may be lost as a result of using an unreliable communication mechanism to transfer alarms from their origin to the management node. For example, since SNMP protocol [20] exploits an unreliable transport layer protocol (UDP), SNMP traps [20] issued by an SNMP agent are not guaranteed to be delivered to the destination. Management/monitoring agents on network devices monitor values of various performance metrics, and issue alerts when the monitored metrics violate pre-set threshold values. Too liberal threshold values may prevent an existing problem from being reported, thereby causing the alarm loss. Given the popularity of unreliable management protocols such as SNMP and the difficulty of calculating the correct threshold values, the probability of an alarm loss is not negligible. When the fault localization algorithm relies on positive information to create the most likely fault hypothesis, alarm loss, if ignored by the algorithm, could lead to an incorrect solution.

Another frequent disturbance in an observation of network state is due to spurious alarms, which are caused by intermittent network faults or by overly restrictive threshold values. Intermittent faults are the ones that result in observable symptoms, but no longer exist at the time of the symptoms' correlation. As with all fault types, the symptom patterns triggered by the intermittent faults and their probability of occurrence may be identified through the analysis of historical data such as alarm log files. When such identification is possible, the interrelation among spurious symptoms may be taken into account. In this case, the intermittent faults can be modeled similar to ordinary faults. Therefore, no change of either the bipartite fault propagation model, or the fault localization algorithm is needed. Oftentimes, in particular for spurious alarms due to overly restrictive detection mechanisms, the determination of spurious symptoms interdependence may be impossible. The method proposed in this section focuses on symptoms for which interdependence information may not be learned.

We address the problem of lost and spurious alarms by augmenting the bipartite belief network model presented in Section II. To model loss, we introduce unobservable failure nodes $e_1, \ldots, e_m$ as replacement for symptom nodes $s_1, \ldots, s_m$, respectively. Then, we add directed edges $e_i{\rightarrow}s_i$, $i{=}1\ldots m$. With every directed edge $e_i{\rightarrow}s_i$ we associate the probability of causal relationship between e_i and s_i equal to $1{-}p_{\text{loss}}(s_i)$, where $p_{\text{loss}}(s_i)$ is the probability that alarm indicating failure

e_i is lost. The values of $p_{\rm loss}(s_i)$ may be obtained, for example, by analyzing packet loss rate in the network used to transport symptom s_i from its origin to the management station.

To model spurious symptoms, we introduce nodes s^*_j, $j{=}1\ldots m$. Then, we add directed edges $s^*_j{\rightarrow}s_j$, where $j{=}1\ldots m$. With every s^*_j we associate prior belief $p_{\rm spurious}(s_j)$ that represents the cumulative probability of events (other than persistent faults) that trigger alarm s_j. The value of $p_{\rm spurious}(s_j)$ may be learned by analysing historical alarm log files. Every directed edge $s^*_j{\rightarrow}s_j$ is labeled with $1{-}p_{\rm loss}(s_j)$.

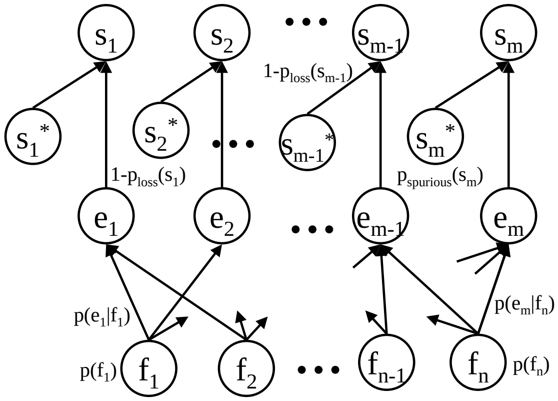

Fig. 3. A belief network modeling lost and spurious symptoms ($\mathcal{BN}(\mathcal{S}_O,p_{\rm loss},p_{\rm spurious})$)

The resultant belief network, which will be denoted by $\mathcal{BN}({\rm S}_O,p_{\rm loss},p_{\rm spurious})$, is presented in Fig. 3. Observe that, when $p_{\rm loss}(s_i){=}0$, edges $e_i{\rightarrow}s_i$ (for $i{=}1\ldots m$) are redundant, and nodes e_i and s_i may be considered identical ($e_i{=}s_i$). Also, when $p_{\rm spurious}(s_j){=}0$, nodes s^*_j are redundant and may be reduced ($s^*_j{=}s_j$). Thus, $\mathcal{BN}(\mathcal{S}_N,0,0)$ is equivalent to the bipartite belief network described in Section IV (used in Algorithm 1), and $\mathcal{BN}({\rm S}_O,0,0)$ is equivalent to the model used in Section V. When the existence of either lost or spurious (but not both) alarms may be neglected, one should use $\mathcal{BN}(\mathcal{S}_O,0,p_{\rm spurious})$ or $\mathcal{BN}(\mathcal{S}_O,p_{\rm loss},0)$, respectively.

When the augmented belief network, $\mathcal{BN}(\mathcal{S}_O,p_{\rm loss}$, $p_{\rm spurious})$, is used as a fault propagation model, fault localization may be performed using Algorithm 2 defined in Section V. In Section VII, we present an application of solutions introduced in this section to the problem of end-to-end service failure diagnosis. We also evaluate the impact of incorporating lost and spurious alarms on the accuracy of the fault localization process in a noisy environment.

VII. Simulation study

In this section, we describe the simulation study performed to evaluate the techniques presented in Sections IV, V, and VI. Algorithms 1 and 2 were implemented in Java. As a real-life application domain we chose end-to-end service failure diagnosis. Recall from Section II that the diagnosis of end-to-end service failures aims at isolating host-to-host services that caused the observed end-to-end service failures. In end-to-end service failure diagnosis, the set of all host-to-host service failures, the set of all end-to-end service failures, and the set of all symptoms are denoted by $\mathcal{F}$, $\mathcal{E}$, and $\mathcal{S}$, respectively. $\mathcal{F}$ contains only failures of services provided by host-to-host links that belong to some end-to-end route. The dependencies between end-to-end services and host-to-host services are determined using routing information. There exists a bijective mapping of $\mathcal{E}$ onto $\mathcal{S}$.

The simulation study presented in this paper uses tree-shaped network topologies, which result, for example, from the usage of the Spanning Tree Protocol [21] as the data-link layer routing protocol. The usage of tree-shaped topologies greatly simplifies their random generation, while not having any significant impact on the accuracy of the results presented in this section. We focus on diagnosing performance problems, e.g., excessive delay or high loss rate, which necessitates the usage of a non-deterministic fault model and fault localization algorithm.

We designed the simulation described in this section according to the following model. Let OR represent the alarm observability ratio, i.e., $OR=|\mathcal{S}_O|/|\mathcal{S}|$. Let LR represent alarm loss rate, i.e., the ratio of the number of generated alarms that were lost to the number of all generated alarms. Let SSR represent spurious alarm rate, i.e., the ratio of the number of spurious alarms to the number of all observable alarms. The three ratios OR, LR, and SSR are parameters of the simulation model.

Given the simulation model with parameters OR, LR, and SSR, for a given network topology size n, where n represents the number of intermediate network nodes, such as bridges or switches, we design K_n simulation cases as follows:

- We create a random tree-shaped n-node network $\mathcal{N}_i$ ($1{\leq}i{\leq}K_n$). We denote $\mathcal{F}$, $\mathcal{E}$, and $\mathcal{S}$ for network $\mathcal{N}_i$ as $\mathcal{F}_i$, $\mathcal{E}_i$, and $\mathcal{S}_i$, respectively. For every link in $\mathcal{N}_i$, we create one $f_j{\in}\mathcal{F}_i$. Note that in an n-node tree-shaped network there are $n{-}1$ links, i.e., $|\mathcal{F}_i|{=}n{-}1$. For every end-to-end path in $\mathcal{N}_i$, we create one $e_l{\in}\mathcal{E}_i$ and one $s_l{\in}\mathcal{S}_i$. Note that $|\mathcal{E}_i|{=}|\mathcal{S}_i|{\in}\mathcal{O}(n^2)$.
- We randomly generate prior fault probability distribution p_f: $\mathcal{F}_i{\rightarrow}[0.001,0.01]$; $p_f(f_j)$s are uniformly distributed over the range $[0.001,0.01]$. We randomly generate conditional probability distribution p_{fe}: $(\mathcal{F}_i{\times}\mathcal{E}_i){\rightarrow}[0,1)$, defined as the probability that e_l occurs given f_j occurs. For all $f_j{\in}\mathcal{F}_i$ and $e_l{\in}\mathcal{E}_i$ such that the path corresponding to e_l includes the link corresponding to f_j, $p_{fe}(f_j,e_l)$s are uniformly distributed over the range $(0,1)$; otherwise, $p_{fe}(f_j,e_l){=}0$.
- We randomly generate the set of observable alarms $\mathcal{S}_{iO}{\subseteq}\mathcal{S}_i$ such that on average $\frac{|\mathcal{S}_{iO}|}{|\mathcal{S}_i|}{=}OR$.
- Given network $\mathcal{N}_i$, we build its belief network model $\mathcal{BN}_i(\mathcal{S}_{iO},p_{\rm loss},p_{\rm spurious})$. We choose $p_{\rm loss}{=}LR$ ($p_{\rm spurious}{=}SSR$) to perform fault localization that includes alarm loss (spurious alarms) in the analysis. We use $p_{\rm loss}{=}0$ ($p_{\rm spurious}{=}0$) to perform fault localization that disregards lost (spurious) symptoms. To utilize Algorithm 1 we initialize $\mathcal{BN}_i(\mathcal{S}_{iO},p_{\rm loss},p_{\rm spurious})$ as described in Section IV; to utilize Algorithm 2 we initialize $\mathcal{BN}_i(\mathcal{S}_{iO},p_{\rm loss},p_{\rm spurious})$ as described in Section V.

For i-th simulation case ($1{\leq}i\ {\leq}K_n$), we create M_S simulation scenarios as follows.

1) Using p_f we randomly generate the set of faulty links in network $\mathcal{N}_i$, $\mathcal{F}^k_{iC}$ ($1{\leq}k{\leq}M_S$), and create probability distribution p^k_e: $\mathcal{E}_i{\rightarrow}[0,1]$, where $p^k_e(e_j){=}$

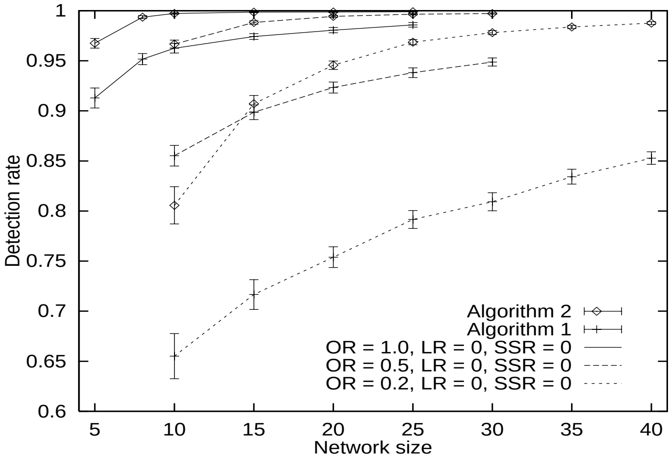

Fig. 4. Detection rate obtained with Algorithms 1 (disregarding positive symptoms) and 2 (including positive symptoms) for different observability ratios OR.

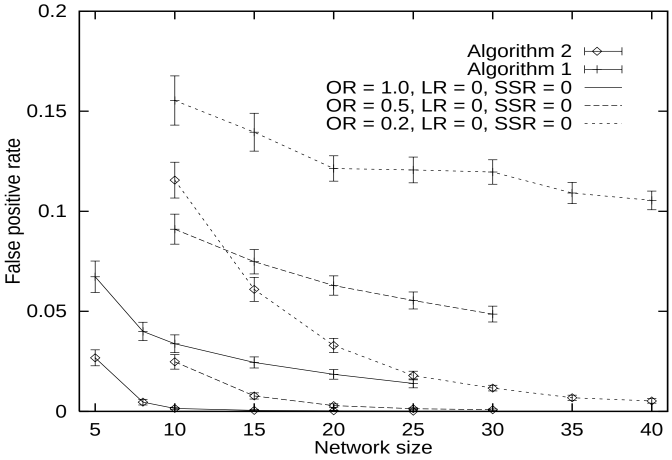

Fig. 5. False positive rate obtained with Algorithms 1 (disregarding positive symptoms) and 2 (including positive symptoms) for different observability ratios OR.

$P\{e_j \text{ occurs}|\text{all faults in } \mathcal{F}^k_{iC} \text{ occur}\}$.

2) Using p^k_e we randomly generate the set of events $\mathcal{E}^k_{iC}$ resulting from faults in $\mathcal{F}^k_{iC}$, and build $\mathcal{S}^k_{iC} \subseteq \mathcal{S}_i$, the set of alarms corresponding to events in $\mathcal{E}^k_{iC}$. We set $\mathcal{S}^k_{iG} = \mathcal{S}^k_{iC} \cap \mathcal{S}_{iO}$.
3) We randomly generate the set of spurious alarms $\mathcal{S}^k_{iS} \subseteq \mathcal{S}_{iO}$ such that $\frac{|\mathcal{S}^k_{iS}|}{|\mathcal{S}^k_{iO}|} = SSR$. We set $\mathcal{S}^k_{iG+S} = \mathcal{S}^k_{iG} \cup \mathcal{S}_{iS}$.
4) We create $\mathcal{S}^k_{iG-L}$ – the set of generated alarms that are not lost by the communication system – by randomly removing alarms from $\mathcal{S}^k_{iG+S}$ so that on average $\frac{|\mathcal{S}^k_{iG-L}|}{|\mathcal{S}^k_{iG+S}|} = 1 - LR$.
5) We set $\mathcal{S}^k_{iN} = \mathcal{S}^k_{iG-L}$; $\mathcal{S}^k_{iN}$ constitutes the set of negative symptoms observed in the simulation scenario k.
6) Using either Algorithm 1 or Algorithm 2 we compute $\mathcal{F}^k_{iD} \subseteq \mathcal{F}_i$, the most likely explanation of symptoms in $\mathcal{S}^k_{iN}$. We calculate *detection rate* (DR^k_i) and *false positive rate* (FPR^k_i) defined using the following equations.

$$\text{DR}^k_i = \frac{|\mathcal{F}^k_{iD} \cap \mathcal{F}^k_{iC}|}{|\mathcal{F}^k_{iC}|} \quad \text{FPR}^k_i = \frac{|\mathcal{F}^k_{iD} \setminus \mathcal{F}^k_{iC}|}{|\mathcal{F}^k_{iD}|}$$

For i-th simulation case we calculate the mean detection rate $\text{DR}_i = \frac{1}{M_S} \sum_{k=1}^{M_S} \text{DR}^k_i$ and mean false positive rate $\text{FPR}_i = \frac{1}{M_S} \sum_{k=1}^{M_S} \text{FPR}^k_i$. Then, we calculate the expected values of detection rate and false positive rate denoted by DR(n), and FPR(n), respectively. In our study, we used K_n=100, and M_S=100 or 200 depending on the variability of DR^k_i. We varied n from 5 to 45.

To evaluate the impact of including positive symptoms into the fault localization process, we set LR=0, and SSR=0 in the simulation model. Correspondingly, we used $\mathcal{BN}(\mathcal{S}_O, 0, 0)$ as the fault propagation model. We compared Algorithms 1 and 2 using observability ratios 1.0, 0.5, and 0.2. As shown in Fig. 4 and 5, the inclusion of positive symptoms in the fault localization process allows the detection and false positive rates to be substantially improved. The improvement is bigger for lower observability ratios; with high observability ratios (e.g., OR=1), the number of negative symptoms is typically large enough to allow quite accurate fault localization without considering positive symptoms. However, such high symptom observability is unlikely in real-life systems. One may also ob-

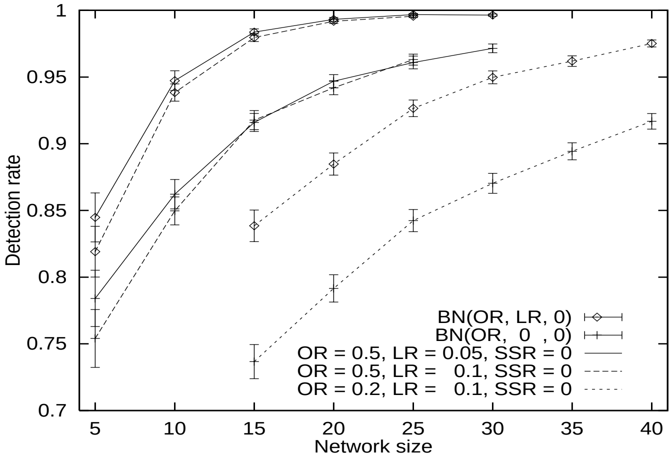

Fig. 6. Detection rate obtained with Algorithm 2 using fault propagation models $\mathcal{BN}(\mathcal{S}_O, LR, 0)$ and $\mathcal{BN}(\mathcal{S}_O, 0, 0)$ within statistically computed confidence intervals.

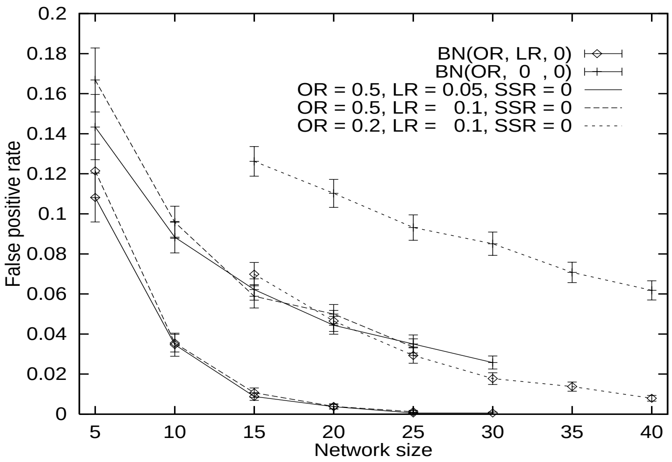

Fig. 7. False positive rate obtained with Algorithm 2 using fault propagation models $\mathcal{BN}(\mathcal{S}_O, LR, 0)$ and $\mathcal{BN}(\mathcal{S}_O, 0, 0)$ within statistically computed confidence intervals.

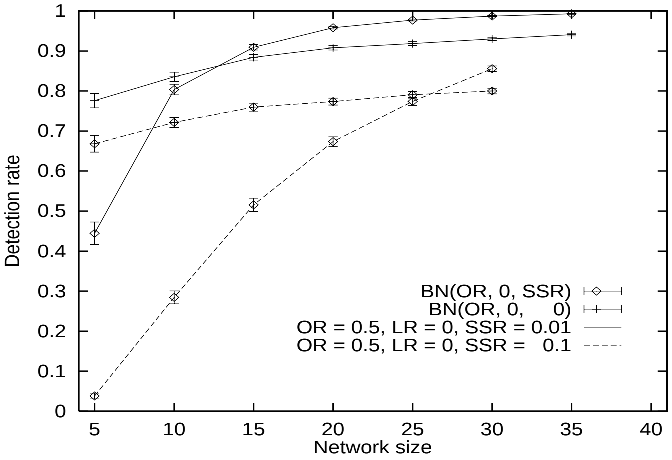

Fig. 8. Detection rate obtained with Algorithm 2 using fault propagation models $\mathcal{BN}(\mathcal{S}_O, 0, SSR)$ and $\mathcal{BN}(\mathcal{S}_O, 0, 0)$ within statistically computed confidence intervals.

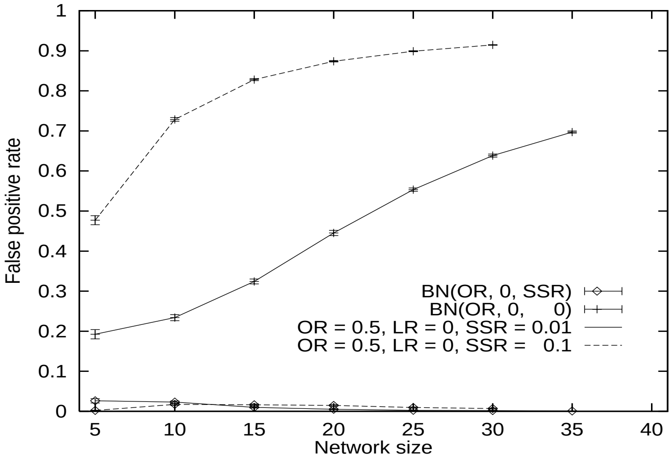

Fig. 9. False positive rate obtained with Algorithm 2 using fault propagation models $\mathcal{BN}(\mathcal{S}_O, 0, SSR)$ and $\mathcal{BN}(\mathcal{S}_O, 0, 0)$ within statistically computed confidence intervals.

serve that the accuracy of both algorithms depends on the observability ratio; higher OR means more information about faults, and, in consequence, higher accuracy.

In the next set of experiments, we isolated the impact of symptom loss on the accuracy of fault localization process. We set SSR=0 in the simulation model. Loss rate was set to either 0.01 or 0.1. We compared the accuracy of Algorithm 2 using belief networks $\mathcal{BN}(\mathcal{S}_O, LR, 0)$ and $\mathcal{BN}(\mathcal{S}_O, 0, 0)$, varying OR between 0.2 and 0.5. Fig. 6 and 7 show that, by including loss rate in the analysis, the detection (false positive) rate may be increased (decreased) by up to 10%. Moreover, given constant OR, this gain is insensitive to the value of LR.

The impact of including spurious symptoms in the fault localization process was evaluated by applying Algorithm 2 to fault propagation models $\mathcal{BN}(\mathcal{S}_O, 0, SSR)$ and $\mathcal{BN}(\mathcal{S}_O, 0, 0)$ using LR=0, and OR=0.5. We varied SSR between 0.01 and 0.1. As shown in Fig. 8, the inclusion of spurious symptoms in the fault localization process in small networks decreases detection rate. This is explained by the fact that in small networks (in particular, with small observability ratios), only a few symptoms are available to the fault localization process. When the possibility of spurious symptoms is taken into account, and the number of symptoms is small, the algorithm concludes that there is no sufficient evidential support for the existence of faults, and considers most of these symptoms spurious. When SSR=0.1, for small networks, the probability that all observed symptoms are spurious is frequently higher than the probability of fault occurrence. Therefore, the algorithm refuses to identify a fault thereby achieving very low detection rate. One can conclude that small networks need to be better instrumented (i.e., have higher OR) to allow fault localization to benefit from the analysis of spurious symptoms. In larger networks, the inclusion of spurious symptoms does not cause a decrease in the detection rate; in fact, as shown in Fig. 8, it allows the detection rate to be improved. Fig. 9 presents the impact of including spurious symptoms on the false positive rate. It shows that regardless of the network size, by taking spurious symptoms into account, the false positive rate of the fault localization process may be significantly decreased.

VIII. Impact of probability estimation errors

One of the drawbacks of applying non-deterministic reasoning to the fault localization problem is the necessity to determine the probabilities assigned to nodes and edges in the fault propagation model. Researchers frequently state that these probabilities may be assigned by a human expert [6]. This process being error prone, it is likely that the probabilities assigned by the expert will differ from those describing the real system. In actuality, the expert assigns discrete confidence levels rather than the exact probabilities.

In this section, we analyze the impact of such probability estimation on the accuracy of Algorithm 1. For this purpose, we designed simulation experiments as described in Section VII. However, in the belief network, exact probabilities are replaced by c confidence levels; i-th confidence level (i=$1 \ldots c$) is represented by probability $\frac{i-1}{c} + \frac{1}{2c}$. Thus, probability p in the real system is estimated as probability $\frac{\lfloor pc \rfloor}{c} + \frac{1}{2c}$.

Fig. 10 and 11 compare the detection rate of Algorithm 1 having exact knowledge of the probability distribution with the detection rate achieved using one, two, and three confidence levels for various observability ratios. Similar comparison of false positive rates is shown in Fig. 12 and 13. It can be observed that probability estimation with three confidence levels allows fault localization to be almost as accurate as with the knowledge of exact probabilities. This observation has an important implication: it allows the expert to use a small set of meaningful qualitative probability assignments such as *unlikely*, *possible*, and *likely*, which correspond to confidence levels 1, 2, and 3, respectively.

Belief networks are therefore a promising model for non-deterministic fault localization, yielding high accuracy even for approximate probability input data.

IX. Related Work

In the past, various event correlation techniques were proposed including rule-based systems [22], [23], model-based reasoning systems [24], [25], model traversing techniques [26], case-based systems [27], fault propagation models [6], [7], and

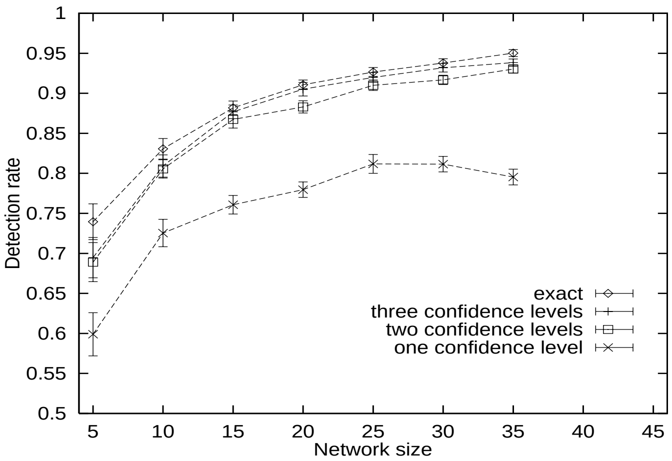

Fig. 10. Detection rate of Algorithm 1 with exact and approximate knowledge of the probability distribution for OR=0.5.

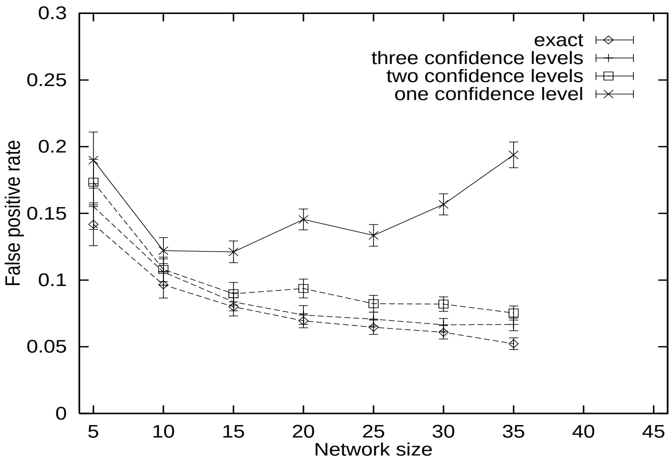

Fig. 12. False positive rate of Algorithm 1 with exact and approximate knowledge of the probability distribution for OR=0.5.

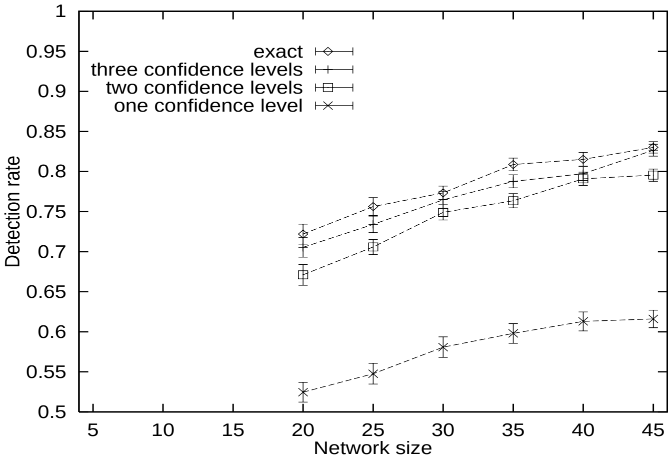

Fig. 11. Detection rate of Algorithm 1 with exact and approximate knowledge of the probability distribution for OR=0.2.

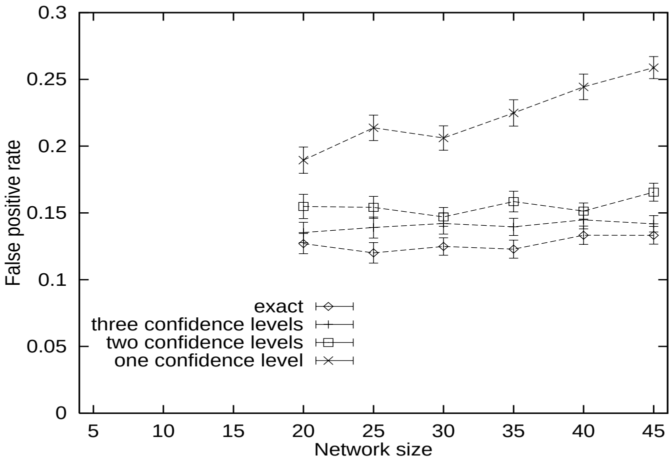

Fig. 13. False positive rate of Algorithm 1 with exact and approximate knowledge of the probability distribution for OR=0.2.

the code-book approach [8]. These approaches differ with respect to their knowledge representation, scalability, adaptability to configuration changes, ability to deal with lost and spurious symptoms, ability to solve novel problem, etc. Most of the above approaches use deterministic reasoning.

In the area of non-deterministic fault diagnosis, several approaches have been proposed. Katzela et al. [6] introduced an $\mathcal{O}(N^3)$ algorithm that finds the most probable explanation of a set of symptoms in an N-node dependency graph. The approach presented in [6] does not allow lost or spurious symptoms and relies on time-windows to collect alarms. In multilayer fault diagnosis, specifying time-windows may be very difficult. The solution presented in this paper addresses all the above issues achieving comparable computational complexity.

Kliger et al. [14] propose a probabilistic model to be used with the codebook approach. Unfortunately, they do not present the non-deterministic decoding schema. We believe that the approach of Algorithms 1 and 2 can be used for this purpose.

The literature on event correlation reports on the applications of belief networks to fault diagnosis. However, the approaches are limited to rather narrow applications. Deng et al. [28] present a polynomial time algorithm for updating belief in a restricted Bayesian network used as a model for fault diagnosis in linear light-wave networks. In [29] belief networks were applied to troubleshoot printing services. The belief network used for this purpose is tree-shaped, which enables exact inference using, e.g., Pearl's algorithms. Wang et al. [30] applied Bayesian theory to identifying faulty links in communication networks. They propose an approximation of *maximum a posteriori* query to find a link responsible for a failure of end-to-end connectivity between a management station and a group of other stations, given prior link failure probabilities. They do not include the representation of conditional probabilities, which makes the approach not suitable for the diagnosis of other than availability-related problems.

Other approaches to dealing with uncertainty in network fault diagnosis include an application of Dempster-Shafer theory to detect break faults in communications networks [31]. Similarly to [30], this technique is tailored specifically to diagnosing connectivity problems in networks with known and static topologies. This solution would not be sufficient for the purpose of diagnosing performance problems, or when the knowledge of the network topology is uncertain or incomplete. Statistical data analysis methods were used for non-deterministic fault diagnosis in bipartite-graphs in [18]. The solution was proposed to detect link failures in wireless and/or battlefield networks based on the observed set of broken end-to-end connections. The technique focuses on dealing with unknown and constantly changing network topologies, and it is not able to deal with lost or spurious symptoms. In addition, all symptoms have to be known before the fault localization process may be started.

The algorithm proposed in this paper performs probabilistic fault diagnosis in communication systems whose fault propagation may be modeled by bipartite graphs. The model of uncertainty is suitable for diagnosing various types of faults, including performance-related ones. The algorithm is event-driven, i.e., it allows a symptom to be analyzed as soon as it is received. Therefore, it has the ability to perform fault diagnosis faster than window-based algorithms [6], [14], [30], [31], [18]. In our work on fault localization we also proposed an incremental, event-driven algorithm based on bipartite causality graphs [10]. The algorithm has slightly lower complexity than the algorithms presented in this paper; however, currently it does not support positive, lost, or spurious symptoms.

X. Conclusions and Future Work

This paper utilizes belief networks to perform fault localization in communication systems taking into account comprehensive information about the system behavior. Most previous work on this subject performs fault localization based solely on the information about malfunctioning system components (i.e., negative symptoms). In this paper, we show that positive information, i.e., the lack of any disorder in some system components, may be used to improve the accuracy of the process. The technique presented in this paper allows lost and spurious symptoms to be incorporated in the analysis. We show through simulation that in a noisy network environment the analysis of lost and spurious symptoms increases the robustness of fault localization with belief networks. We also demonstrate that belief networks yield high accuracy even for approximate probability input data and therefore are a promising model for non-deterministic fault localization.

In our future research we plan to generalize the solution presented in this paper to fault localization with belief networks of arbitrary shape. We will also investigate the impact of positive lost and spurious symptoms on other non-deterministic fault localization techniques such as Incremental Hypothesis Update [10].[3]

References

[1] R. Gopal, "Layered model for supporting fault isolation and recovery," in *Proc. of Network Operation and Management Symposium*, Honolulu, Hawaii, 2000, pp. 729–742.

[2] K. Appleby, G. Goldszmidt, and M. Steinder, "Yemanja – a layered event correlation engine for multi-domain server farms," in *Integrated Network Management VII*, G. Pavlou, N. Anerousis, and A. Liotta, Eds. May 2001, pp. 329–344, IEEE.

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

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

[5] M. Steinder and A.S. Sethi, "End-to-end service failure diagnosis using belief networks," in *Proc. of Network Operation and Management Symposium*, Florence, Italy, 2002, (to appear).

[6] I. Katzela and M. Schwartz, "Schemes for fault identification in communication networks," *IEEE Transactions on Networking*, vol. 3, no. 6, pp. 733–764, 1995.

[7] M. Hasan, B. Sugla, and R. Viswanathan, "A conceptual framework for network management event correlation and filtering systems," In Sloman et al. [34], pp. 233–246.

[8] S. A. Yemini, S. Kliger, E. Mozes, Y. Yemini, and D. Ohsie, "High speed and robust event correlation," *IEEE Communications Magazine*, vol. 34, no. 5, pp. 82–90, 1996.

[9] S. Kätker, "A modeling framework for integrated distributed systems fault management," in *Proc. of the IFIP/IEEE Int'l Conference on Distributed Platforms*, C. Popien, Ed., Dresden, Germany, 1996, pp. 187–198.

[10] 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, AR, 2001, pp. 374–379.

[11] R. Dechter, "Bucket Elimination: A unifying framework for probabilistic inference," in *Proc. of the Twelfth Conference on Uncertainty in Artificial Intelligence*, E. Horvitz and F. V. Jensen, Eds., Portland, Oregon, Aug. 1996, Morgan Kaufmann Publishers.

[12] G. F. Cooper, "Probabilistic inference using belief networks is NP-Hard," Tech. Rep. KSL-87-27, Stanford University, 1988.

[13] P. Dagum and M. Luby, "Approximately probabilistic reasoning in Bayesian belief networks is NP-hard," *Artificial Intelligence*, pp. 141–153, 1993.

[14] S. Kliger, S. Yemini, Y. Yemini, D. Ohsie, and S. Stolfo, "A coding approach to event correlation," In Sethi et al. [32], pp. 266–277.

[15] R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter, *Probabilistic Networks and Expert Systems*, Statistics for Engineering and Information Science. Springer-Verlag, 1999.

[16] R. J. McEliece, D. J. C. MacKay, and J.-F. Cheng, "Turbo decoding as an instance of Pearl's "belief propagation" algorithm," *Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 140–151, Feb. 1998.

[17] F. R. Kschischang and B. J. Frey, "Iterative decoding of compound codes by probability propagation in graphical models," *Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 219–230, Feb. 1998.

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

[19] A. T. Bouloutas, S. Calo, and A. Finkel, "Alarm correlation and fault identification in communication networks," *IEEE Transactions on Communications*, vol. 42, no. 2/3/4, pp. 523–533, 1994.

[20] J. D. Case, K. McCloghrie, M. T. Rose, and S. Waldbusser, *Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)*, IETF Network Working Group, 1996, RFC 1905.

[21] R. Perlman, *Interconnections, Second Edition: Bridges, Routers, Switches, and Internetworking Protocols*, Addison Wesley, 1999.

[22] G. Liu, A. K. Mok, and E. J. Yang, "Composite events for network event correlation," In Sloman et al. [34], pp. 247–260.

[23] P. Wu, R. Bhatnagar, L. Epshtein, M. Bhandaru, and Z. Shi, "Alarm correlation engine (ACE)," in *Proc. of Network Operation and Management Symposium*, New Orleans, LA, 1998, pp. 733–742.

[24] G. Jakobson and M. D. Weissman, "Alarm correlation," *IEEE Network*, vol. 7, no. 6, pp. 52–59, Nov. 1993.

[25] Y. A. Nygate, "Event correlation using rule and object based techniques," In Sethi et al. [32], pp. 278–289.

[26] J. F. Jordaan and M. E. Paterok, "Event correlation in heterogeneous networks using the OSI management framework," In Hegering and Yemini [33], pp. 683–695.

[27] L. Lewis, "A case-based reasoning approach to the resolution of faults in communications networks," In Hegering and Yemini [33], pp. 671–681.

[28] R. H. Deng, A. A. Lazar, and W. Wang, "A probabilistic approach to fault diagnosis in linear lightwave networks," In Hegering and Yemini [33], pp. 697–708.

[29] D. Heckerman and M. P. Wellman, "Bayesian networks," *Communications of the ACM*, vol. 38, no. 3, pp. 27–30, Mar. 1995.

[30] C. Wang and M. Schwartz, "Identification of faulty links in dynamic-routed networks," *Journal on Selected Areas in Communications*, vol. 11, no. 3, pp. 1449–1460, Dec. 1993.

[31] N. Dawes, J. Aloft, and B. Pagurek, "Network diagnosis by reasoning in uncertain nested evidence spaces," *IEEE Transactions on Communications*, 1995.

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

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

[34] M. Sloman, S. Mazumdar, and E. Lupu, Eds., *Integrated Network Management VI*. IEEE Publishing, May 1999.

[3] The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U. S. Government.