TECHNICAL RESEARCH REPORT

A Framework for Supporting Intelligent Fault and Performance Management for Communication Networks

by Hongjun Li, John S. Baras

CSHCN TR 2001-13
(ISR TR 2001-20)

A Framework for Supporting Intelligent Fault and Performance Management for Communication Networks

Hongjun Li, John S. Baras

Center for Satellite and Hybrid Communication Networks

Department of Electrical and Computer Engineering

University of Maryland, College Park, MD 20742

{ hjli, baras}@isr.umd.edu

Abstract

In this paper, we present a framework for supporting intelligent fault and performance management for communication networks. Belief networks are taken as the basis for knowledge representation and inference under evidence. When using belief networks for diagnosis, we identify two questions: When can I say that I get the right diagnosis and stop? If right diagnosis has not been obtained yet, which test should I choose next? For the first question, we define the notion of right diagnosis via the introduction of intervention networks. For the second question, we formulate the decision making procedure using the framework of partially observable Markov decision processes. A heuristic dynamic strategy is proposed to solve this problem and the effectiveness is shown via simulation.

Keywords: Fault Management, Performance Management, Fault Diagnosis, Belief networks, Markov Decision Processes, Dynamic Programming

Submission area: Resource, performance and fault management

1 Introduction

Communication networks have become indispensable today and this trend will continue as more and more new technologies emerge. Due to the growing number of networks that have served as the critical components in the infrastructure of many organizations, interest in fault management has increased during the past decade [8][18][19][28].

In a communication network environment, we categorize the term fault as either *hard* or *soft*. Hard faults consist of hardware or software faults [28]. Hardware faults include incorrect or incomplete logic design, damage, wear or expiry, etc. Software faults usually come from incorrect or incomplete design and implementation. However, there are still some other important kinds of faults that need to be considered. For example, the performance of a switch is degrading or there exists congestion on one of the links. Another example is to model faults as deviations from normal behavior [29]. Since there might not be a failure in any of the components, we call such faults *soft* faults. Hard faults can be solved by replacing hardware elements or software debugging. Such diagnosis is called re-active diagnosis. Soft faults are in many cases indications of some serious problems and for this reason, the diagnosis of such faults is called pro-active diagnosis. Handling soft faults is typically part of the functionality of performance management [11][24] and in the sequel, we use the term fault to represent both hard and soft faults for convenience.

The task of fault management is to detect, diagnose and correct the possible faults during network operations. Fault detection can be thought of as an online process that gives indication of malfunctioning. To declare the existence of such malfunctioning, we need a model of "normal" behavior against which comparisons can be made. Such normal behavior could be specified as a finite state machine, as in the case of protocol or software testing [5][23][37]; or it could be a derived model according to operation status and/or statistical analysis, for example Auto Regressive (AR) model [12] and Generalized Likelihood Ratio (GLR) model [36]. Such indications of malfunctioning are manifested in the form of events, which must be correlated to diagnose the most likely fault(s) [8][15][22][32]. Finally, corrective actions are taken to restore the normal operations. In this paper, we focus on fault diagnosis issues.

Efficient fault management requires an appropriate level of automation. Knowledge based expert systems, as examples of automated systems, have been very appealing for communication networks fault diagnosis [20]. Usually, such systems are based on deterministic network models. A serious problem of using deterministic models is their inability to isolate primary sources of faults from uncoordinated network events. Observing that the cause-and-effect relationship between symptoms and possible causes is inherently nondeterministic, *probabilistic* models can be considered to gain a more accurate representation. As a natural and efficient model for human inferential reasoning, belief networks have emerged as the general knowledge representation scheme under uncertainty and key technology for diagnosis [10][16][30][34].

In communication networks fault management field, Hood and Ji [12] proposed a pro-active network fault detection scheme based on AR models and belief networks. Selected Management Information Base (MIB) variables were monitored and their normal behaviors are learned via AR modeling. Belief networks were used to compute certain posterior probabilities, given some deviations from the normal behavior. However, their belief network model is over simplistic in that there is only one root node, which will explain whatever anomalies as detected by the AR modeling. It estimates the network status in a snapshot; there is no further test suggested. In [13], Huard and Lazar used a more general belief network model with multiple root nodes as the candidate faults. They also presented a dynamic programming (DP) formulation for the network troubleshooting problem. However, single fault assumption was made, which limits the applicability. In this paper, we develop a framework that supports fault diagnosis for communication networks. General belief network models with multiple root nodes are chosen as the knowledge representation scheme. We handle multiple faults and formulate the fault diagnosis procedure as a Partially Observable Markov Decision Processes (POMDP) problem with optimal stopping. To help solve the problem, we introduce the notion of right diagnosis for optimal stopping and provide a dynamic, heuristic strategy for test sequence generation.

The rest of the paper is organized as follows. In section 2, we introduce belief networks and in section 3, we identify two problems when using belief networks for diagnosis. We introduce the concept of intervention networks and right diagnosis for the first problem in section 4, and the decision theoretic fault diagnosis strategies are studied in section 5. We run simulation in section 6, and conclude the paper in section 7.

2 Belief Network as the Probabilistic Fault Model

A belief network, also called a Bayesian network or a causal network, is a graphical representation of cause-and-effect relationships within a problem domain. More formally, a belief network $\mathcal{B} = (V, L, P)$ is a Directed Acyclic Graph (DAG) in which: The nodes V represent variables of interest (propositions); The set of directed links L represent the causal influence among the variables; The strength of an influence is represented by conditional probability tables (CPT). For any node in the DAG, given its parents, that node is conditionally independent of any other node that is not its descendent. This conditional independence makes a belief network model a compact representation of the joint probability distribution P over the interested variables. Belief networks can also serve as the inference engine, and can compute efficiently any queries over the variables modeled therein[14][16][30]. Let us look at one example.

Suppose we are handling the problem call failure and identify the possible causes as follows: Server, link and switch may fail, and there might be heavy traffic that causes the network congestion. Luckily, we have access to the alarms associated with link failure and switch failure. This scenario is modeled as a belief network, as shown in figure 1. Each node takes binary value and the table associated with it represents the conditional probability distribution, given its parent nodes' instantiations. Without any observations, the initial marginal probabilities of each node are shown in figure 2.

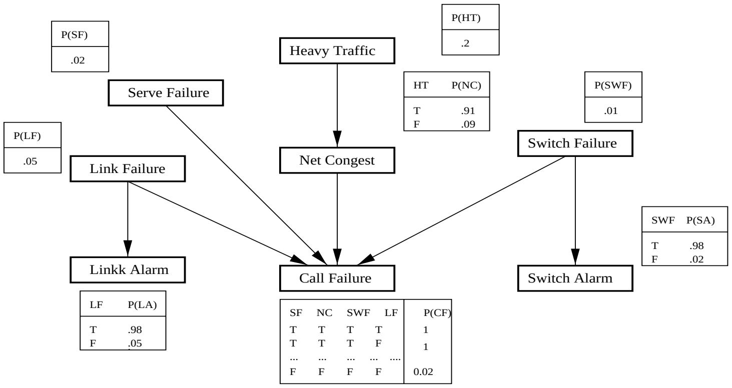

Figure 1: An example Belief Network

Now suppose we observe that there are call failures. We wish to infer the most probable cause for this symptom from the belief network model. To do this, we input this evidence, execute the belief propagation,

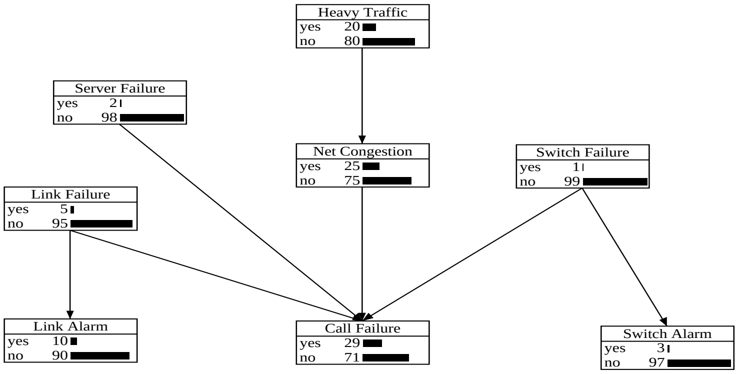

Figure 2: Initial marginal probabilities

and obtain the updated beliefs of each non-evidential node, as shown in figure 3. Note that for each candidate fault node, namely Link Failure, Server Failure, Heavy Traffic and Switch Failure, the probability of being faulty is increased. If we also observe link alarms, then we hope that this extra information could help locate the most probable fault, see figure 4. As we would have expected, the evidence of link alarms distinguishes Link Failure as the most probable fault, which also "explains away" other possible candidates in the sense that their updated beliefs are decreased, as compared with those in figure 3.

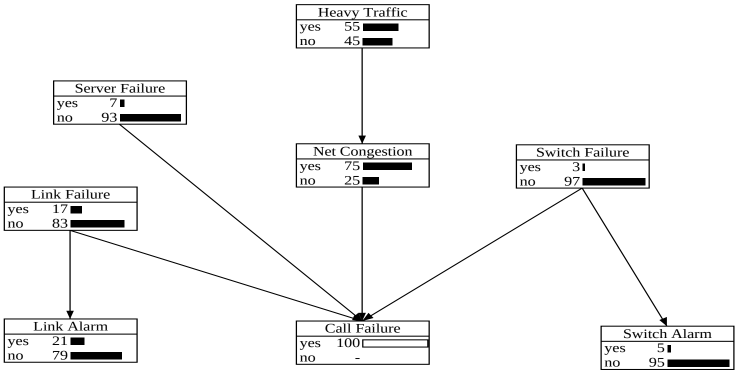

Figure 3: Updated beliefs after observing Call Failure

The above example shows the sketch of doing diagnosis using belief networks: obtain evidence, update beliefs, obtain evidence again, and so on. This is *active* diagnosis in that we are seeking more information on the fly during diagnosis. We will discuss this process in more details in the following sections.

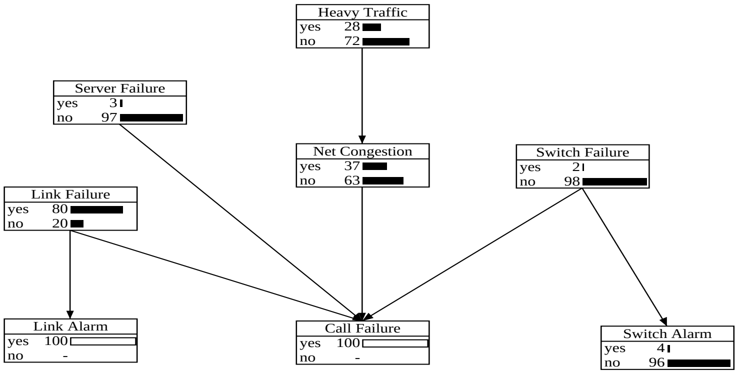

Figure 4: Updated beliefs after observing Call Failure and Link Alarm

3 Fault Diagnosis Problems using Belief Networks

In communication networks, probes are attached to some hardware/software components to get operation status. Typically the raw data returned from the probes will be grouped into vector form $\mathbf{d} \in \mathbf{R^n}$ and then processed to get some aggregated values (e.g. average, peak value, etc.). A statistics is a function from $\mathbf{R^n}$ to $\mathbf{R}$ that maps the raw data vector $\mathbf{d}$ to a real number. Such statistics will usually be quantified and represented using discrete values. We use 0 to represent normal status, and other positive integers to represent abnormal status with different level of severity. A node v in a belief network model $\mathcal{B} = (V, L, P)$ is called observable if and only if it represents the health status of a statistics, or corresponds to a user report. The set of observable nodes is denoted by O. The non-observable set is simply $\tilde{O} = V \setminus O$. We restrict these observable nodes to be leaf nodes only, and vice versa. The regular evidence set R contains those nodes that we observe during regular network monitoring operations. Each $r \in R$ is called a symptom node. The test set ST contains all other observable nodes that are not currently in R, namely $ST = O \setminus R$. The fault set F is the set of root nodes, and they are not observable, $F \subseteq \tilde{O}$. We restrict that all root nodes are binary valued. The hidden node set H contains all nodes in $\tilde{O}$ but not in fault set F, $H = \tilde{O} \setminus F$. Hidden nodes are intermediate nodes between faults and symptoms and we don't usually put queries on them during diagnosis.

The problem domain is said to be working in normal status with respect to regular evidence set R if and only if every node in R takes value 0, or vector $\mathbf{r} = \mathbf{0}$, where $\mathbf{r} = (\mathrm{r}_1, \mathrm{r}_2, \ldots, \mathrm{r}_{|\mathrm{R}|})$. The problem domain is said to be working in abnormal status with respect to regular evidence set R if and only if there is at least one $r \in R$ whose value is other than 0. There might be cases when multiple symptom nodes in R take nonzero values. The syndrome with respect to regular evidence set R is simply the nonzero vector $\mathbf{r}$. Any syndrome can trigger the diagnosis process.

After fault diagnosis is triggered, the initial evidence is propagated and the posterior probability of any $f \in F$ being faulty can be calculated. It would be ideal if we can locate the fault with efforts up to this. But most of the time, similar to what happens in medical diagnosis, we need more information to help pinpoint the fault. So naturally, we identify two important problems associated with belief network based fault diagnosis: When can I say that I get the right diagnosis and stop? If right diagnosis has not been obtained yet, which test should I choose next? We address these two problems in the next sections. In our work, we postulate that all the observations and tests are constrained within the belief network model.

4 Right Diagnosis via Intervention

Consider what a human usually think during diagnosis. After obtaining one possible reason, one may naturally ask, for example, “Will the problematic circuit work normally if I replace this suspicious component with a good one?” He/she then goes ahead and sees what will happen after the replacement. If the syndrome disappears, one can claim that he/she actually found and trouble-shooted the fault. If the problem domain is tiny, not very complex, and the replacement burden is light, this paradigm will work well. But for communication networks, the story is totally different. We would like to do intelligent diagnosis via *computation*, rather than brutal replacement before we are very confident what the fault is.

To do this, we need to distinguish between two kinds of semantics for the instantiation of a node in a belief network: passive observation and active setting. All the instantiations of nodes we have talked about so far are passive observations, and we would like to know the consequences of, and the possible causes for such observations. The alternative semantics is that we can also *set* the value of a node via active experiment. One example is the above question, where external reasons (the human diagnoser) explain why the suspicious component becomes good and thus all the parent nodes for this node should not count as causes during belief updating. Other belief updating like evaluating consequences, however, are not influenced by this active setting. This external force is called *intervention* in [31].

With this *set* semantics, we could do virtual replacement in our belief network model. For simplicity, we assume here that the single symptom node is $S1$. For each node in F, we could get its posterior probability of being faulty given $S1 = 1$. Let $f = argmax_{g \in F} P(g = 1|S1 = 1)$, and we would evaluate $P(S1 = 0|setting(f = 0))$. Other nodes in F are treated as background variables and they keep at the same status as what has just been updated. In our work, we introduce the so-called intervention belief network to help this virtual replacement.

DEFINITION 4.1. An *intervention belief network* $\widetilde{\mathcal{B}} = (V, L, P, S, Fs)$ is obtained from the original belief network $\mathcal{B} = (V, L, P)$ with the same V, L, P. S is the symptom set and $Fs \in F$ is the set of suspicious nodes. We compute for each $s \in S$ the probability $P(s = 0|setting(Fs = \mathbf{0}))$ using $\widetilde{\mathcal{B}}$.

For our particular example above, the virtual replacement procedure is as follows. First, in $\mathcal{B} = (V, L, P)$, update for each node $f_i \in F$ the probability $p_i \overset{\triangle}{=} P(f_i = 1|S1 = 1)$. Suppose $f_1 = argmax_{g \in F} P(g = 1|S1 = 1)$. Then in intervention belief network $\widetilde{\mathcal{B}} = (V, L, P, S1, f_1)$, set node $f_1 = 0$, and with $P(f_i = 1) = p_i$, $i = 2, \cdots, |F|$, compute $P(S1 = 0|setting(f_1 = 0))$. To determine whether or not this virtual replacement has led $S1$ to an acceptable status, we need a reference value for the computed $P(S1 = 0|setting(f_1 = 0))$ to compare with. Without any evidence input, the belief network model $\mathcal{B}$ itself gives the marginal probability of each leaf node to be normal. We use these values as the reference in our work.

DEFINITION 4.2. Given a small number ϵ, we say that node $S1$ becomes ϵ *-normal* via intervention on f_1 if and only if $P(S1 = 0) - P(S1 = 0|setting(f_1 = 0)) < \epsilon$.

Note that during diagnosis process, some of the testing nodes chosen may already manifested themselves as values other than "normal". These nodes should also be included in intervention network $\widetilde{\mathcal{B}}$.

DEFINITION 4.3. A nonempty set of suspicious nodes Fs is called the explanation or right diagnosis if and only if every node in set S, including both initial and newly-found symptoms, becomes ϵ-normal if we set every node in Fs to normal in the intervention belief network $\widetilde{\mathcal{B}} = (V, L, P, S, Fs)$. It is when Fs explains the set S that we terminate the diagnosis process.

5 Decision Theoretic Fault Diagnosis Strategies

We formulate the test selection procedure as a partially observable Markov decision processes (POMDP) problem with optimal stopping. At each decision epoch, we could either choose a node to test or stop there. Test is rarely free, and termination incurs some costs. The goal is to find a good test sequence and the right time to stop. We will show that by choosing termination cost appropriately, the optimal stopping rule matches our notion of right diagnosis.

5.1 POMDP Formulation

State Space $\mathcal{S}$

The state is the status of the root nodes $F = \{F_1, \dots, F_{|F|}\}$, and for a particular $s \in \mathcal{S} = 2^{|\mathrm{F}|}$, $s = \{f_1, \dots, f_{|F|}\}$. We use S_k to denote the state at time k. In our diagnosis case, the current state, which is unobservable, does not change regardless what tests will be chosen. The goal of diagnosis is to *identify* this state by using initial symptoms and subsequent test results. So here we have

$$P(S_{k+1}|S_k) = \begin{cases} 1 & \text{if}\, S_{k+1} = S_k \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

History Process

If we choose one test per decision epoch, the time step set is defined as $N = \{1, 2, \dots, |ST|\}$. The active evidence set AE contains the nodes that are instantiated during the process of diagnosis. Initially $AE = R$ and it expands as more test nodes in ST are added into it. Nodes in AE are not to be considered for future use. The candidate test set C_{st} contains the nodes in ST that are available to be chosen and tested. Initially $C_{st} = ST$ and it shrinks as instantiated nodes are removed from it. The action set $A = C_{st} \cup \{STOP\}$. Let Z_{a_t} denote the value obtained by observing a_t, and we define the history process up to time k as $I_k = (Z_0, (a_1, Z_{a_1}), \dots, (a_k, Z_{a_k}))$, where $Z_0 = \big((r_1, Z_{r_1}), \dots, (r_{|R|}, Z_{r_{|R|}})\big)$ represents the regular evidence set and corresponding instantiations. I_k grows with diagnosis and obviously, $I_k = (I_{k-1}, (a_k, Z_{a_k}))$, the Markov property. We can simply take I_k as the *state* at time k and obtain a completely observable Markov decision problem. But the growing state process makes this approach impractical.

Belief / Information State

Given I_k, we define $b_k = P(\mathbf{F}|\mathrm{I_k})$ as the probability distribution of states in $\mathcal{S}$. It is proven that b_k is a sufficient statistics that contains all information embedded in the history process for control, and we call it belief or information state [1][3]. Using Bayes rule, we can easily verify that the process $\{b_k\}$ is also Markov. If we choose b_k as the state at time k, we avoid the growth of the state space; but now, the state space is *continuous,* and we call it $\mathcal{B}_\mathrm{c}$. In our case, if we are given $I_k, a_k, \mathrm{and}\, Z_{a_k}$, the next belief state b_{k+1} is uniquely determined via belief network propagation, and we define $\Psi(b_k, a_k, Z_{a_k}) \triangleq Pr(b_{k+1}|b_k, a_k, Z_{a_k})$. If we let $\mathcal{X} = \mathcal{B}_\mathrm{c} \cup \{\mathrm{T}\}$ and x_k be the state at time k, then the augmented states evolve according to

$$x_{k+1} = \begin{cases} \Psi(x_k, a_k, Z_{a_k}) & \mathrm{if}\, x_k \neq T \,\mathrm{and}\, a_k \neq STOP \\ T & \mathrm{if}\, x_k = T \,\mathrm{or}\, (x_k \neq T \,\mathrm{and}\, a_k = STOP) \end{cases} \tag{2}$$

The observation model for $a_k \neq STOP$ is $P(Z_{a_k}|I_k, a_k) = Pr(a_k = Z_{a_k}|I_k)$.

Choosing Suspicious Nodes

After we obtain x_k, it will not suffice to give out this probability distribution directly as the result. What is needed is the explanation. To see if we could obtain the explanation as defined above, we need to extract from x_k the suspicious nodes. However, it is always an important issue to determine how many suspicious nodes we should choose from the fault set F. In our belief network model and the parallel intervention model, we should be discreet in choosing multiple nodes. If we simply choose all nodes in F and do the intervention, the symptom nodes will all become ϵ-normal for sure. But clearly, calling every node in F as faulty is not acceptable; One of the most important aspects of fault diagnosis in general is to bias among the many possible faults and locate the real one(s)! In our work, we tried two schemes.

In the first scheme, we compute for each node in F the probability of being faulty given I_k, and sort them in a descending order, say $p_1 \geq p_2 \geq \cdots \geq p_{|F|}$. We only choose the first j nodes such that $\sum_{k=1}^{j} p_k / \sum_{k=1}^{|F|} p_k \geq \eta$, where $\eta \in (0, 1)$. It should not be close to 1, since in that case, we would have to choose almost all nodes in F. In our work, we choose 0.4 initially. If we could not find the right diagnosis, we increase η by a small amount so that more root nodes could emerge as the candidates. By doing this, we are not limiting ourselves to the single fault scenario. This scheme is intuitive, I_k suffices to provide information for each node in F, and it is not necessary to calculate x_k. However, it is very hard to choose a good η that works well without knowing in advance how many faults there might be.

The second scheme makes use of x_k. We first compute the belief state and get a table that contains the joint distribution of the root nodes given I_k. Then we choose the largest entry from the table and mark the index of the entry. The suspicious nodes are obtained from the index. For example, if we only have four root nodes and the binary string corresponding to the index of the largest entry is 0101, then the second and fourth nodes are chosen. In this scheme, there is no need to find a good η, and it adapts to multiple causes easily. The drawback is that extra storage space is needed for the joint distribution table. If the number of root nodes is small, this is a preferable scheme.

Cost Structure

There is an immediate cost associated with each $s_i \in ST$. The cost function $C(s_i, t)$ entails careful deliberation about many factors like the difficulty and time to be consumed for the test, etc. Here we assume that the cost function is of form $C(s_i)$. This is usually the case in that the cost is normally associated with the test itself only, and the test itself does not usually change with time. Also, we wish to diagnose promptly and we penalize on diagnosis steps. If $a_k = STOP$ at time k, no penalty. Otherwise, we penalize this extra step using function $g(k)$. Here, we simply take $g(k) = 1$ for all k. At time k with state $x_k \neq T$, if we choose $a_k = STOP$, we incur $t(x_k)$ as the termination cost. Note that $t(T) = 0$. Given $x_k \neq T$ and suspicious node set Fs, we compute $t(x_k)$ as follows. First, in original belie network, let $K = F \setminus F_s$ and compute for each node in K the probability of being faulty as $q_i \overset{\triangle}{=} Pr(K_i = 1|I_k)$. Second, in intervention network, set the root nodes that correspond to those in K with the same probabilities as those in $\{q_i\}$, and set the root nodes that correspond to those in F_s to state "normal". Finally, in intervention network for each node S_i in the active symptom set S, and for some given small ϵ, define $\Delta = P(S_i = 0) - P(S_i = 0|\text{Setting root nodes as above})$. If $\Delta < \epsilon$, $t_{S_i}(x_k) = 0$, else $t_{S_i}(x_k) = CONST\,[\Delta - \epsilon]$, where $CONST$ is a constant to make $t_{S_i}(x_k)$ large. The total cost is $t(x_k) = \sum_{S_i \in S} t_{S_i}(x_k)$. So, the immediate cost of choosing action a_k at time k with state $x_k \neq T$ is

$$g_k(x_k, a_k) = \begin{cases} c(a_k) + g(k) & \text{if } a_k \neq STOP \\ t(x_k) & \text{otherwise} \end{cases} \tag{3}$$

At the last step N, the terminal cost $g_N(x_N)$ is defined as

$$g_N(x_N) = \begin{cases} t(x_N) & \text{if } x_N \neq T \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Note that both $g_k(x_k, a_k)$ and $g_N(x_N)$ are deterministic functions. Now we have the finite horizon problem

$$\min_{a_k, k=0,\ldots,N-1} \left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, a_k) \right\}. \tag{5}$$

5.2 Solution for the Problem

Define $J_k(x_k)$ as the cost-to-go at state x_k and time k [3]. At termination state T, $J_k(T) = 0, \forall k = 0, \ldots, N-1$. For $x_k \neq T$, we have the dynamic programming algorithm:

$$J_N(x_N) = g_N(x_N) \tag{6}$$

$$J_k(x_k) = \min \left[t(x_k), \min_{a_k \in A_k} \left[c(a_k) + g(k) + \sum_j P(a_k = j | I_k) J_{k+1}(T(x_k, a_k, Z_{a_k} = j)) \right] \right] \tag{7}$$

So the optimal stopping policy is: Choose STOP if

$$t(x_k) \leq \min_{a_k \in A_k} \left[c(a_k) + g(k) + \sum_j P(a_k = j | I_k) J_{k+1}(\Psi(x_k, a_k, Z_{a_k} = j)) \right], \tag{8}$$

at current state x_k and time k. If we choose $t(x_k)$, as shown above, such that $t(x_k) = 0$ in the case of right diagnosis and let $t(x_k)$ be very large otherwise, then the optimal stopping policy is: STOP if and only if we obtain the right diagnosis. Now let us look at the test selection strategies.

To solve the problem (5) using the dynamic programming update (7), the continuous state space is the major obstacle. It would be very desirable if we could find some structures for the value function or optimal policy. In one class of problems [21], the optimal value function for the finite horizon problem is piecewise linear and concave. Thus we could represent the value functions using a set of discrete vectors and avoid the direct handling of the continuous space. Unfortunately, we don't have this good property in our problem, and we need to seek to approximate methods. As discussed above, we need to extract from state $x_k \neq T$ the suspicious node set F_s. We ignore those root nodes that are not very fault-prone and his is our first approximation. Now, given that F_s does not explain the current active symptoms, we need some heuristics to help choose the next test. Let us begin with a simpler problem for intuition.

Suppose the concern here is to locate the single faulty component. There are symptoms indicating the malfunction (e.g. car doesn't start) and for each possible faulty component there is a *direct* test associated with it. The cost for testing component i is c_i. Based on the symptoms, we obtain P_i, the probability that component i is in failure, for every component. We are supposed to test those components one at a time. As soon as one component fails its associated test, we claim that we find the single fault and stop. By interchange argument [3], it is easy to see that in an optimal strategy, all elements must be in non-decreasing sequence of c/P values, see also [17].

Our problem is different from this scenario in the following aspects. It tackles failures while our problem integrates both hard and soft faults. It assumes the existence of direct test while we don't have that luxury. For a communication network environment which is distributed, complex and heterogeneous, it is impossible to predefine and store a direct test for each possible cause. Actually one of the goals here is to *generate* dynamically the test sequence on the fly. In our setup, right diagnosis is determined through computation, rather than brutal replacement. Finally, our algorithm should be able to tackle multiple faults.

But the c/P algorithm does provide insight in that it reflects the following observation: in order to minimize the total cost, people are more likely to try those more fault-prone, cheaper components before the less-probable, expensive ones. In our diagnosis algorithm, we wish to find an appropriate test node st if Fs could not explain the active symptom set S. In particular, we would like to choose the test node from candidate test set C_{st} that is cheapest and most relevant to Fs. To achieve this, we need a measure for relevance between a test node in C_{st} and a fault node in F_s.

Definition 5.1. Given I_k, the relevance of random variable Y relative to random variable X is defined as

$$R(X;Y|I_k) = \frac{I(X;Y|I_k)}{H(X|I_k)},$$

where $H(X|I_k) = -\sum_{x\in\mathcal{X}} p(x|I_k)\log p(x|I_k)$ is the conditional entropy of a random variable X, $I(X;Y|I_k) =$

$\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x, y|I_k) \log \frac{p(x,y|I_k)}{p(x|I_k)p(y|I_k)}$ is the conditional mutual information between random variable X and Y [6]. $R(X; Y|I_k) \in [0, 1]$ indicates to what extent Y can provide information about X. $R(X; Y|I_k) = 1$ means that Y can uniquely determine X, while $R(X; Y|I_k) = 0$ indicates that Y and X are independent, given current I_k. Note that $R(X; Y|I_k) \neq R(Y; X|I_k)$. More generally,

Definition 5.2. Given I_k, the relevance of random variable Y relative to a set of random variables $\mathbf{X}$ is

$$R(\mathbf{X}; \mathrm{Y}|\mathrm{I_k}) = \frac{\mathrm{I}(\mathbf{X}; \mathrm{Y}|\mathrm{I_k})}{\mathrm{H}(\mathbf{X}|\mathbf{I_k})},$$

where $H(\mathbf{X}|\mathbf{I_k})$ and $I(\mathbf{X}; \mathrm{Y}|\mathrm{I_k})$ are defined similarly as above.

With the relevance measure, our next test node given I_k at time k is simply

$$st = argmax_{g \in C_{st}} R(\mathbf{Fs}; \mathrm{g})/\mathrm{c(g)}, \tag{9}$$

and our fault diagnosis process is summarized as follows, also shown in figure 5.

- Step 1. Initialization
    - Set time step $tp = 0$, $AE = R$, $C_{st} = ST$.
    - Input evidence by setting the nodes in set AE according to current active values ae.
- Step 2. Belief Propagation in belief network $\mathcal{B}$ and get the set of suspicious nodes F_s according to scheme one or two.
- Step 3. Set the root nodes in $\widetilde{\mathcal{B}} = (V, L, P, S, Fs)$ accordingly, and execute the intervention. If Fs explains S, update total cost and TERMINATE.
- Step 4. Get next testing node
    - If $C_{st} = \Phi$, update total cost and give out the set F_s and say "Didn't find the right diagnosis, but here is the list of possible faults in decreasing order".
    - Else: Get node st according to (9).
- Step 5. Observing test node st and get observation Z_{st}
    - Input this evidence $st = Z_{st}$ to original belief network $\mathcal{B}$. Update tp, C_{st}, and AE.
    - Goto Step 2.

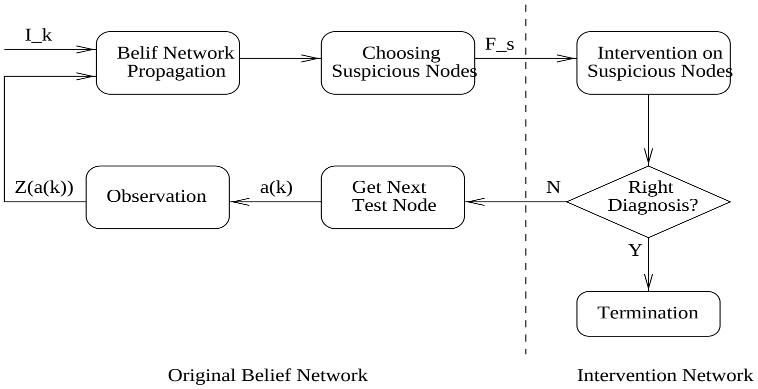

Figure 5: Illustration of the diagnosis process using intervention belief network

6 Simulation

To illustrate the effectiveness of our fault diagnosis algorithm, consider the example network in figure 6. Two switches $SW1$ and $SW2$ are connected via link $L1$. We have a probe a hooked at the end of $SW2$ to measure the traffic throughput going out of $SW2$. Suppose the information we could obtain during network operation include whether or not: $SW1$ alarm is normal, A could connect $SW2$, B could connect $SW2$, A could connect C, C could connect $SW1$, throughput at probe a is normal, and D could connect $SW1$. The possible faults are identified as: $SW1$ works normal or not, $L1$ normal or congested, $SW2$ normal or not, and source pumped from C to $L2$ is normal or not. We set up a belief network model for such situations, and figure 7 shows the structure and initial probability distributions.

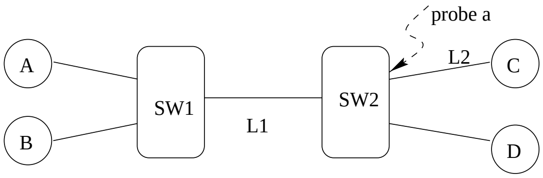

Figure 6: Example Network

Let us look at one diagnosis scenario. Suppose we observe that A_Conn_SW2 goes wrong, and we get the updated distribution as shown in figure 8. We see that $SW1$ is the suspicious node and the intervention result is $P(A\_Conn\_SW2 = yes|Intervention) = 0.78$. Initially, $P(A\_Conn\_SW2 = yes) = 0.83$, and we have not yet got the right diagnosis for $\epsilon = 0.4$. Based on our test selection scheme, node $SW1\_Indicator$

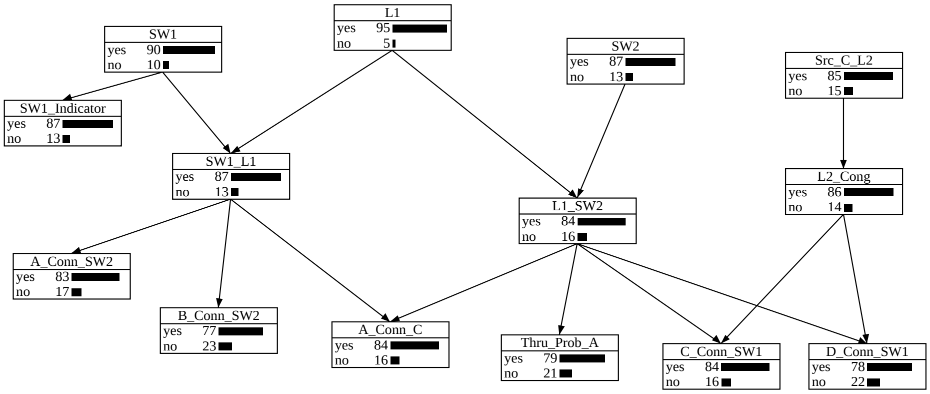

Figure 7: Belief Network for Example Network

is chosen and the observation of it is "normal". The updated distribution is shown in figure 9. Again, $L1$ is intervened and no right diagnosis is obtained. The next node selected this time is A_Conn_C and the observation is "abnormal". We got the updated distribution again in figure 10. If we intervene node $L1$, we have $P(A\_Conn\_SW2 = yes|Intervention) = 0.87 > 0.83$, and we obtain the right diagnosis!

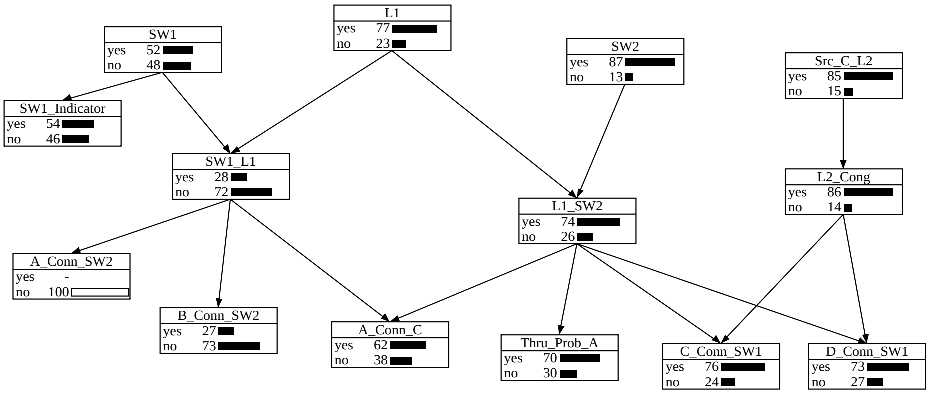

Figure 8: After A_Conn_SW2 Goes Wrong

As a comparison to our node selection scheme, we use the random scheme meaning that each time we need a test node, we simply choose one uniformly from all current available nodes in C_{st}. In our simulation, the outcome of chosen test node st is uniformly generated as either 0 or 1. The costs for testing each leaf node is shown in Table 1, with 40 as the penalty for not being able to find the right diagnosis. Table 2 shows for three scenarios the comparisons of the two test generation schemes with 2000 runs, which take only about 40 milliseconds per run for each scenario on a SUN Ultra2 running Solaris 8. We see that node

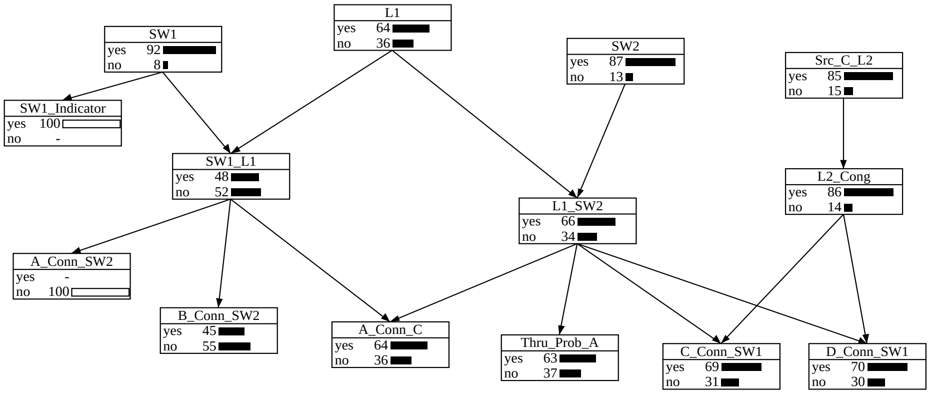

Figure 9: After *SW1_Indicator* Observed as *normal*

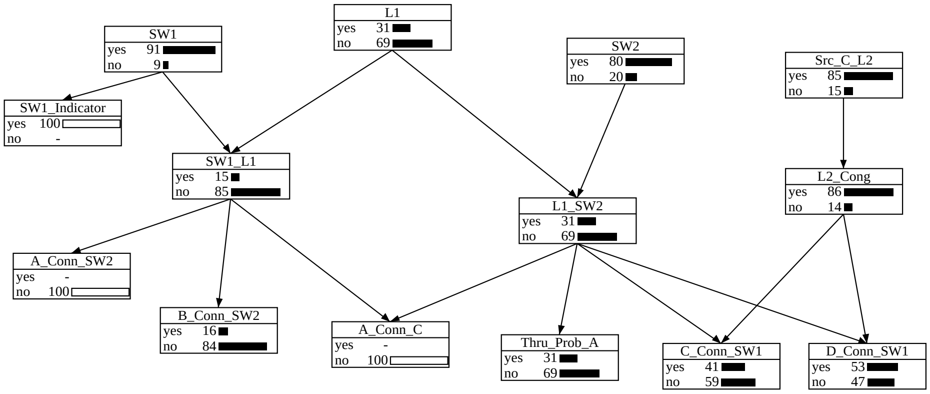

Figure 10: After *A_Conn_C* Observed as *Abnormal*

selection via relevance is much better than that via random selection.

Table 1: Cost for All Leaf Nodes

SW1_Indicator	A_Conn_SW2	B_Conn_SW2	A_Conn_C	Thru_Prob_A	C_Conn_SW1	D_Conn_SW1
2	1	7	1	3	1	3

7 Conclusions

In this paper, we presented a framework that supports intelligent fault and performance management for communication networks. We used belief networks as the knowledge representation scheme and inference engine for the problem domain. The optimal stopping problem is tackled by using the notion of right

Table 2: Comparison of Node Selection Schemes

Symptom Nodes	Random Selection		Relevance Selection	
	Avg. Cost	Success Rate	Avg. Cost	Success Rate
A_Conn_SW2	15.38	84.5%	9.13	94%
A_Conn_C	26.21	70.1%	14.22	88%
A_Conn_SW2 and A_Conn_C	24.68	67.8%	3	100%

diagnosis via intervention, and test selection is based on a heuristic dynamic strategy. Simulation shows that this scheme is much superior than a random selection scheme. Note that as evidence accumulates, we may input them one by one followed by a propagation right after each evidence-input, as we have shown in this paper, or we may input them once altogether and do only one propagation. This provides us the flexibility for either on-line diagnosis or off-line diagnosis/analysis.

This framework is quite general. The belief network model and the associated decision making algorithm could exist at any management station in a network management system. After a test node is chosen, the observation for this test may take advantage of the traditional SNMP paradigm by polling appropriate MIB variables; or, delegated (mobile) agents could be sent to the network elements to collect the data by using the management by delegation paradigm [7][9]. As one example of such an agent-based environment, the authors presented in [26] a couple of system designs for adaptive, distributed network monitoring and control, and in a sister paper also submitted to this conference [27], the authors discuss in more details the idea of change of logic. Further, the managed system could be divided into domains [33], and for each domain we could assign such an "intelligent" module that take charge of the fault and performance management for it [2][25].

The dynamic heuristic strategy could be improved via reinforcement learning [4][35], and in particular, Q-learning techniques [38]. The idea is that, by interacting with the environment, the decision making module could accumulate experience and improves its performance. We could use the above dynamic strategy as the starting point. One possible problem when using Q-learning is that we need to represent each state explicitly to store the associated learned value. As we discussed above, this might be intractable. However, by noticing that there will be many cases that our diagnosis could stop within only a couple of steps, we can truncate the expanding state space significantly. We will discuss the details in a forthcoming paper. Also, simulation on a more complex network environment is on the way.

References

[1] K. J. Astrom, "Optimal control of Markov decision processes with incomplete state estimation," *Journal of Mathematical Analysis and Applications,* vol. 10, pp. 174–205, 1965.

[2] J. S. Baras, H. Li and G. Mykoniatis, "Integrated, Distributed Fault Management for Communication Networks", Technical Report, CSHCN TR 98-10, University of Maryland, 1998

[3] D. P. Bertsekas, *Dynamic Programming and Optimal Control, Vol. I and II* , Athena Scientific, Belmont, MA, 1995

[4] D. P. Bertsekas, and J. N. Tsitsiklis, *Neuro-Dynamic Programming,* Athena Scientific, 1996

[5] A. Bouloutas, G. W. Hart, and M. Schwartz, "Simple finite-state fault detectors for communication networks", *IEEE Trans. Commun.,* , vol. 40, pp. 477-479, Mar. 1992

[6] T. M. Cover, and J. A. Thomas, *Elements of Information Theory* , Wiley Interscience, 1991

[7] M. El-Darieby, A. Bieszczad, "Intelligent Mobile Agents: Towards Network Fault Management Automation", in *Proceedings of the Sixth IFIP/IEEE International Symposium on Integrated Network Management*, Boston, MA, May 1999

[8] Gabrijela Dreo, "A Framework for Supporting Fault Diagnosis in Integrated Network and Systems Management: Methodologies for the Correlation of Trouble Tickets and Access to ProblemSolving Expertise", PhD Dissertation, Department of Computer Science, University of Munich, 1995

[9] G. Goldszmidt, Y. Yemini, "Distributed Management by Delegation", in *Proceedings of 15th International Conference on Distributed Computing Systems*, 1995

[10] D. Heckerman, J. S. Breese, and K. Rommelse, "Decision-Theoretic Troubleshooting", *Communications of the ACM,* vol. 38, pp. 49-57, 1995

[11] H.G. Hegering, S. Abeck, and B. Neumair. *Integrated Management of Networked Systems: Concepts, Architectures, and Their Operational Application.* Morgan Kaufmann, San Francisco, CA, USA, 1999.

[12] C. S. Hood, and C. Ji, "Probabilistic Network Fault Detection", *GlobalCom,* pp. 1872-1876, 1996

[13] J. Huard, and A. A. Lazar, "Fault Isolation based on Decision-Theoretic Troubleshooting", Tech. Rep. TR 442-96-08, Center for Telecommunications Research, Columbia University, 1996

[14] http://www.hugin.dk

[15] G. Jacobson, M. Weissman, "Alarm Correlation", *IEEE Network,* Vol. 7, No. 6, 1993

[16] F. V. Jensen, *An Introduction to Bayesian Networks*, UCL, 1997

[17] J. Kalagnanam and M. Henrion, "A Comparison of Decision Analysis and Expert Rules for Sequential Diagnosis", in *Uncertainty in Artificial Intelligence 4,* (Amsterdam, The Netherlands), pp. 271-281, Elsevier Science Publishers B. V., 1990

[18] Irene Katzela and Mischa Schwarz, "Schemes for Fault Identification in Communication Networks", *IEEE/ACM Transactions on Networking,* Vol. 3, No. 6, pp. 753–764, Dec. 1995.

[19] I. Katzela, *Fault Diagnosis in Telecommunication Networks*, Ph.D. Thesis, Columbia University, 1996

[20] L. Kerschberg, R. Baum, A. Waisanen, I. Huang and J. Yoon, "Managing Faults in Telecommunications Networks: A Taxonomy to Knowledge-Based Approaches", IEEE, pp. 779-784, 1991

[21] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, " Planning and acting in partially observable stochastic domains", Artificial Intelligence, Vol 101, pp. 99-134, 1998

[22] S. Kliger, S. Yemini, Y. Yemini, D. Ohsie, and S. Stolfo. "A Coding Approach to Event Correlation." In Sethi, Raynaud, and Faure-Vincent, editors, *Integrated Network Management* , number 4, pp. 266-277. IFIP, Chapman and Hall, may 1995.

[23] D. Lee, A.N. Netravali, K.K. Sabnani, B. Sugla, and A. John, "Passive testing and applications to network management," *Proceedings of the 1997 International Conference on Network Protocols,* pp. 113-122, 1997

[24] A. Leinwand and K. F. Conroy, *Network Management, A practical perspective,* second edition, Addison-Wesley, 1996

[25] H. Li, J. S. Baras and G. Mykoniatis, "An Automated, Distributed, Intelligent Fault Management System for Communication Networks", *ATIRP'99,* 2-4 February, 1999

[26] H. Li, S. Yang, H. Xi, and J. S. Baras, "Systems Designs for Adaptive, Distributed Network Monitoring and Control", *IFIP/IEEE International Symposium on Integrated Network Management,* Seattle, Washington, May 2001, to appear.

[27] H. Li, S. Yang, and J. S. Baras, "On System Designs for Distributed, Extensible Framework for Network Monitoring and Control", submitted to *IFIP/IEEE International Conference on Management of Multimedia Networks and Services,* Chicago, October 2001.

[28] G. Mahamat, A. Das, G.V. Bochmann, "An overview of fault management in telecommunication networks", *Advanced Information Processing Techniques for LAN and MAN Management,* 1994 IFIP

[29] R. Maxion, "A case study of ethernet anomalies in a distributed computing environment", *IEEE Trans. on Reliability*, Vol. 39, No. 4, pp. 433-443, Oct 1990

[30] J. Pearl, *Probabilistic Reasoning In Intelligent Systems: Networks of Plausible Inference,* Morgan Kaufmann, 1988

[31] J. Pearl, *Causality,* Cambridge Press, 2000

[32] I. Rouvellou, and G. W. Hart, "Automatic alarm correlation for fault identification", In *Proc. IEEE INFOCOM*, page 553-561, 1995

[33] M. Sloman and K. Twidle. "Chapter 16. Domains: A Framework for Structuring Management Policy". In M. Sloman (Ed.). *Network and Distributed Systems Management,* pp. 433-453. Addison-Wesley, Wokingham, UK, 1994.

[34] D. Spiegelhalter, P. Dawid, S. Lauritzen, and R. Cowell. "Bayesian analysis in expert systems." *Statistical Science,* vol. 8, no. 3, pp. 219-282, 1993.

[35] R. S. Sutton, A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, 1998

[36] M. Thottan, C. Ji, "Adaptive Thresholding for Proactive Network Problem Detection," *Third IEEE International Workshop on Systems Management,* pp. 108-116, Newport, Rhode Island, April, 1998.

[37] C. Wang and M. Schwartz, "Fault detection with multiple observers," in *Proc. INFOCOM,* May 1992

[38] C.J.C,H. Watkins, P. Dayan, "Q-learning", *Machine Learning*, 8, pp. 279-292, 1992