Proactive Network Fault Detection

Cynthia S. Hood and Chuanyi Ji[†]

Department of Computer Science and Applied Mathematics
Illinois Institute of Technology, Chicago, IL 60616

[†]Department of Electrical, Computer, and Systems Engineering
Rensselaer Polytechnic Institute, Troy, NY 12180

ABSTRACT

The increasing role of communication networks in today's society results in a demand for higher levels of network availability and reliability. At the same time, fault management is becoming more difficult due to the dynamic nature and heterogeneity of networks. We propose an intelligent monitoring system using adaptive statistical techniques. The system continually learns the normal behavior of the network and detects deviations from the norm. Within the monitoring system, the measurements are segmented, and features extracted from the segments are used to describe the normal behavior of the measurement variables. This information is combined in the structure of a Bayesian network. The proposed system is thereby able to detect unknown or unseen faults. Experimental results on real network data demonstrate that the proposed system can detect abnormal behavior before a fault actually occurs.

1. Introduction

High-speed communication networks play an increasingly important role in today's society. A key challenge in fulfilling this role, maintaining network availability and reliability, is the responsibility of network management. To meet this challenge, network management tools and methodologies must evolve to meet the needs of current and future communication environments [18].

Fault management is the part of network management responsible for detecting and identifying faults in the network. Interest in fault management has increased over the past decade due to the growing number of networks that have become a critical component of the infrastructure of many organizations, making faults and downtime very costly. In addition, as computer networks evolve from providing only "best effort" service to providing a range of service guarantees to accommodate real-time applications (e.g. video), higher levels of reliability are required. By preserving network reliability, fault management lays the foundation for the stringent Quality of Service (QoS) requirements placed on networks by real-time applications.

As the fault management problem becomes more important, it has also become more difficult. This can be traced primarily to the dynamic nature and heterogeneity of current networks. Fundamental changes to the network occur much more frequently due to the growing demands on the network and the availability of new, improved components and applications. With network components and applications developed in an open environment, a network can be configured by mixing and matching several vendors' hardware and software. While this allows the network to utilize the latest technologies and be customized to the needs of the users, it also increases the risk of faults or problems [18].

Current implementations of the fault management part of network management systems generally rely on a human network manager to transfer his or her network expertise into a set of rules. These rules are eventually translated into threshold levels on the measurement variables being collected. When some prescribed set of one or more thresholds is exceeded, an alarm is sent to the network manager. The fault management system must correlate the alarms to identify the problem and take corrective action. Alarm correlation is difficult, due largely to the lack of accurate temporal information available to the fault management system [11]. Additionally, as the rate of change in networks increases, it becomes more difficult for a human network manager to maintain a sufficient level of expertise on the behavior of the network.

Previous research in fault management has covered approaches such as expert systems [7], Finite State Machines (FSMs) [13], advanced database techniques [17], and probabilistic approaches [3]. A review of communication network fault detection and identification can be found in [8]. The approaches mentioned above require specification of the faults to be detected. This limits the performance of these approaches since it is not feasible to specify all possible faults. In addition, changes in network configuration, applications and traffic can change the types and nature of faults that may occur, making modeling faults more difficult and in many cases impractical. Research using learning machines to detect anomalies [11] addresses the issue of fault modeling, but does not provide a method for correlating the information collected in space or time.

The problem we will tackle is automated fault detection without specific models of faults. We propose an adaptive learning system for network monitoring. The system learns the normal behavior of each measurement variable. Deviations from the norm are detected and the information gathered is combined in the probabilistic framework of a Bayesian network. Benefits from this approach include the ability to detect unknown faults, the ability to correlate information in space and time, and the ability to detect subtle changes occurring before the actual failure. This allows faults to be detected when they are developing so the network manager has time to take corrective action to prevent outages or downtime. In addition, this approach requires minimal amounts of network specific information, so it can be generalized across network nodes and types of networks. Our approach is tested on a computer network. We monitor the Management Information Base (MIB) variables collected within the Simple Network Management Protocol (SNMP) framework. No specialized hardware is required for monitoring.

The paper is organized as follows. Section 2 provides background material on Bayesian networks. Our intelligent monitoring approach is described in Section 3. Detailed information about the data we collected is given in Section 4, and Section 5 contains results and comparisons. Conclusions and areas for further investigation are discussed in Section 6.

2. Bayesian network background

A Bayesian network, also called a belief network or a causal network, is a graphical representation of relationships within a problem domain. More formally, a Bayesian network is a directed acyclic graph (DAG), where certain conditional independence assumptions hold [7]. The nodes of the DAG represent random variables. The conditional independence assumptions are as follows: Given a DAG $G = (N,E)$, where $n \in N$ is a node in the network and $e \in E$ is a directed arc. For each $n \in N$, let $p(n) \subseteq N$ be the set of all parents of n, and $d(n) \subseteq N$ be the set of all descendents of n. For every subset $W \subseteq N - (d(n) \cup \{n\})$, W and n are conditionally independent given $p(n)$. In other words, for any node in the DAG, given that node's parents, that node is independent of any other node that is not its descendent. Figure 1 illustrates the independence assumptions for a Bayesian network similar to the one we will use in our monitoring system.

These assumptions allow us to estimate the conditional probabilities of any of the nodes (or random variables) in the Bayesian network given the observed information or evidence. The strength of Bayesian networks is that they provide a theoretical framework for combining statistical data with prior knowledge about the problem domain. Therefore, they are particularly useful in practical applications.

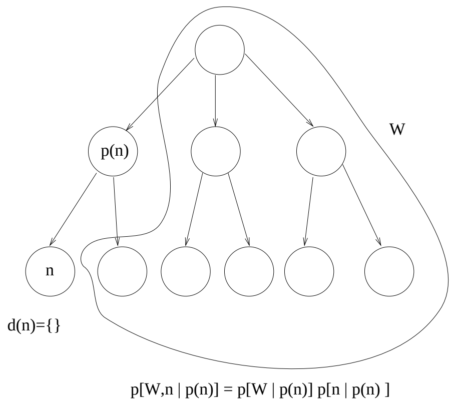

Figure 1 Example of independence assumptions

Bayesian networks have been widely used for medical diagnosis [15] [4], troubleshooting [5], and in the communication network field, they have been proposed to diagnose faults in Linear Lightwave Networks [6]. In [6] other methods have been used for detection and the Bayesian networks are used for diagnosis only. In this work, we propose using a Bayesian network as a mechanism to combine information from different variables for the purpose of detecting anomalies.

3. Monitoring system

We propose an automated monitoring system that is able to detect anomalies without specific models of the behavior to be detected. The premise of this approach is that anomalous or unusual network behavior is an indication of a fault within the network. The monitoring system components are shown in Figure 2.

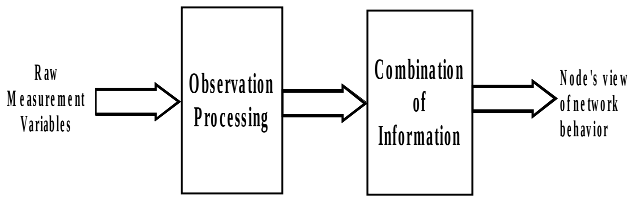
Raw Measurement Variables
Observation Processing
Combination of Information
Node's view of network behavior

Figure 2 Monitoring system

The system resides locally, allowing each node in the network to compose a picture of the network's health. To get this picture, measurement information must be combined with prior knowledge about the network. To accomplish this, the monitoring system has two main components; observation processing and combination of information. The raw measurement variables are processed to estimate the probability of each measured variable at a given time. The probabilities are then combined using a Bayesian network to provide a broader picture of the network's behavior. By doing this locally, we can correlate this information in time and space. This allows the central network manager to receive a more complete, less noisy picture of each node's view of network health. This can ease the alarm correlation problem. It also allows the node to take corrective actions if necessary.

3.1. Observation processing

The goal of the observation processing part of the monitoring system is to take the raw measurement variables and transform them into a set of measures indicating the behavior of each variable. Each measurement variable is a time series. Many of the measurement variables are representative of network traffic. To date, the characterization of network traffic signals is an active research area [9]. Therefore, the signals (i.e. measurement variables) to be processed are not considered to be well understood and as such there is not an optimal, or even standard method to characterize the behavior of these signals.

In processing the information we use a change detection methodology. Since the behavior of the network is dynamic, the behavior of the measurement variables change frequently. As most changes are related to network traffic, simply detecting that a change has occurred is not enough. The goal is to try to recognize the changes that are important in terms of fault detection. We do this by characterizing the behavior of the measurement variables.

3.1.1 Segmentation. One of the challenges presented by the network dynamics is the non-stationarity of the observations. Since our goal is to extract pertinent information, we need to group the time series data in some way so that features can be calculated. To do this we segment the data into variable length pieces. Each piece contains a portion of the time series that is statistically similar.

There are two primary benefits realized from segmentation: (1) the statistics calculated from each segment are more representative of the signal, and (2) signal processing techniques requiring a stationary signal can be used within each segment. In terms of monitoring, the segmentation provides the benefit of temporally correlating the observations. Since many of the network signals are bursty, the temporal correlation can help distinguish between a burst and a change in the nature of the signal. The sequential segmentation algorithm described in [1] is used. Once the observations have been grouped into segments, the pertinent information must be captured from each segment.

3.1.2 Feature extraction. Before our approach to feature extraction is discussed, we first need to examine the shortcomings of commonly used methods. Thresholds are the primary method currently used in both practice [10] and research [3] for detecting abnormal behavior. The feature is not the value of the threshold itself, but the information on whether or not the threshold has been exceeded by a particular measurement variable. There can be both upper and lower thresholds, in which case the feature would be the information on whether the variable is within the thresholds. One of the difficulties with thresholds is properly setting the threshold level, since they are highly dependent on the traffic level. Improperly set, the thresholds may never be exceeded, thereby letting problems go undetected, or they may be exceeded too often, flooding the network manager with false alarms.

While properly set thresholds do a good job of detecting large rises and falls in a measurement variable, more subtle behavior changes are missed. For example, a change in the variance of a signal, or a subtle change in the mean will not be detected using thresholds. These types of changes may be symptoms of something problematic in the network - this behavior is unusual for the variable. Detection of the more subtle signs of problems may allow corrective action to be taken to avoid a bigger problem. Identification of the problem also becomes easier with a more complete description of the symptoms rather than just the extreme cases.

Our goal is to extract information that will help determine whether the behavior of the measurement variable is normal or abnormal. Ideally we want to capture information in the signal that will change or become abnormal only when a problem is occurring. To do this we would need a feature that is invariant to the network traffic patterns and other influences that cause the non-stationarities. This is an open problem, so we choose a feature that changes along with the network and continually adapt the model of normal behavior.

To detect the more subtle changes in the nature of the measured variables, we use the parameters of an AR(2) or second-order autoregressive process as features. The AR(2) process is defined as

$$y(t) = a_1 y(t - 1) + a_2 y(t - 2) + \in(t),$$

where $y(t)$ is the value of the signal at time t, a_1 and a_2 are the AR parameters that we use as features, and $\in(t)$ is white noise. Additional features have been investigated in [6].

3.1.3 Learning the behavior. The features provide the ability to detect subtle changes, but they must first be used to establish a description of normal behavior. The description of the behavior is in the form of a probability distribution. Changes can then be detected relative to the distribution.

Ideally, for each sample we would like to estimate its likelihood given that, (1) the network is operating normally, and (2) there is a fault. To do this, we need to know exactly when the network as a whole, and each of its functions is operating normally. This type of information is not typically available. We are able to get some information about the health of the network from the tools currently used to report problems (i.e., log files). The information available contains reports on certain types of serious network problems that have occurred. From a learning perspective, we can use these reports as labels.

One way of using the labels is to learn the probability distribution of each measurement variable, given that its related network function is abnormal. This sounds promising, but in fact is extremely difficult due to the sparse nature of abnormal data. With so few examples of problems, we do not have the variety (i.e., many examples of different problems) or the depth (i.e., several examples of the same type of problem) to effectively learn the distribution [2].

Instead we learn only the likelihood of the sample given the network function is normal. We define normal behavior to be the behavior of the variable during the time period when the distribution is learned. This time period will be referred to as the learning window. Since what we are learning is the usual behavior of the variable during the learning window, there is no guarantee that this always corresponds to the network or network functions being healthy. If there is a problem in the network that impacts the behavior of the network for a significant portion of the learning window, the problematic behavior will be learned as the usual behavior. We assume that this is rarely the case.

We still need to estimate the likelihood of the sample given the network function is abnormal. Since we are unable to estimate this from the data, we treat this distribution as unknown as in [14], and assume a uniform distribution over the range of the AR(2) parameters. If the AR(2) process is stationary

$$a_1 + a_2 < 1,$$
$$a_2 - a_1 < 1,$$
$$-1 < a_2 < 1.$$

Any sample falling outside the range is assumed to be abnormal.

3.2 Combination of Information

The goal of this part of the monitoring system is to combine the processed measurement variable information into higher-level measures of network behavior. These higher-level measures provide the monitored node's view of the network behavior. These measures can be used to trigger local control actions or a message to a centralized network manager.

Each measurement variable measure is combined in the probabilistic framework of a Bayesian network. The Bayesian network is used because it provides a method for estimating probabilities and it allows observed information to be combined with prior knowledge.

We begin by defining the random variables or nodes in the Bayesian network. There are two types of variables, those that are observed and those that are not observed and thus need to be estimated. We will call the variables that are not observed the internal variables. The observed variables directly correspond to the MIB variables. The internal variables are defined to be *Network Interface (IF), IP,* and *UDP*. The IF, IP, and UDP variables correspond to the MIB groups. Logically they represent different types of network functionality. The MIB variables within a group are the measurement variables for that network function (nf).

The Network variable is defined to correspond to all of the network functionality. In this work, we assume that the network is comprised only of the IF, IP, and UDP functions since these were the functions being used at the router where we were monitoring. Other network functions can easily be added. The structure of the Bayesian network is shown in Figure 3. The arrows between the nodes in Figure 3 go from cause to effect.

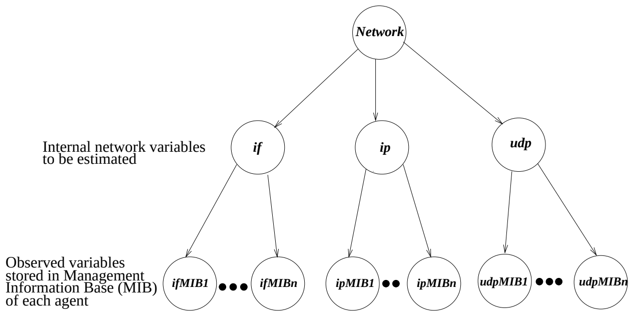
Network
Internal network variables to be estimated
if
ip
udp
Observed variables stored in Management Information Base (MIB) of each agent
ifMIB1
ifMIBn
ipMIB1
ipMIBn
udpMIB1
udpMIBn

Figure 3 Bayesian network for fault detection

In our model, the health of the network is the most general information estimated and can be considered to be an underlying influence on the rest of the nodes in the Bayesian network. The overall health of the network directly influences the health of the three functions of the network (i.e., IF, IP, and UDP). This is indicated in our model by the arrows from the network random variable to the IF, IP, and UDP random variables. Likewise, for each network function, the health of that function directly influences the values of the individual measurement or MIB variables for that function. One way of interpreting this is that a problem in the network will cause symptoms to occur within the network functions, and likewise, problems within a network function will cause symptoms to show up in the MIB variables for that function.

The model has been designed based on these intuitive relationships from the structure of the MIB. The Bayesian network model requires the conditional independence assumptions described in Section 2. These assumptions are reasonable since each of the network functions represent independent functional components of the network. These components may fail independently, although there is a relationship between the functions, and serious problems in one component can eventually impact the other components. Since propagation of a fault through the functional components depends on the type and location of the fault (i.e., faults may propagate from low-level functions to high-level functions, or vice versa) [16], it is difficult to incorporate a propagation structure into the model that will accommodate all types of propagation.

In addition, the relationships between the nodes of the Bayesian network (network functions and MIB variables) are complex and not well understood. Therefore, as a starting point, we have proposed a simple model where no a priori relationship between the network functions is assumed, given the overall health of the network is known. Therefore we have not assumed a fault propagation structure in our model. Alternative Bayesian network structures that assume fault propagation structures have been investigated in [6].

Each of the internal variables is defined to be discrete with two states, *normal* and *abnormal*. Each of the observable variables is defined to be continuous. We are interested in estimating the following posterior probabilities:

$$p(network = normal / abnormal \mid MIBs) \qquad (1)$$

and

$$p(nf = normal / abnormal \mid MIBs). \qquad (2)$$

Due to the tree or singly connected structure of the Bayesian network, these probabilities can be calculated efficiently either using Pearl's algorithm [12] or directly. The equations for the direct calculation can easily be derived using the conditional independence assumption stated above.

The following quantities are necessary to calculate the probabilities in (1) and (2):

$$p(network = normal / abnormal), \qquad (3)$$

$$p(nf = normal / abn \mid network = normal / abn), \qquad (4)$$

and

$$p(MIB\ variable \mid nf = normal / abnormal). \qquad (5)$$

The prior probabilities in (3) and the conditional probabilities in (4) are estimated using prior knowledge of the network behavior gained from observations and conversations with the network managers. These probabilities remain constant throughout the monitoring process. The conditional probabilities in (5) are estimated using the observed MIB variables. Section 3.1 contains the details of how these are obtained.

Since we are monitoring locally, all of the evidence or probabilities estimated from the observed MIB variables is available to the system. This enables the system to calculate the desired posterior probabilities using a complete and current set of observations.

4. Data Collection

Data for this work was collected from the RPI Computer Science Department network. The network as shown in Figure 4 is comprised of 7 subnetworks, or subnets, and two routers. The individual nodes (e.g., workstations, printers, etc.) on the subnets are not shown. Router 1 is the gateway between this network and the campus network, with all the traffic to and from the campus and the outside world flowing through this router. Router 2 mainly routes the local traffic flowing between the subnetworks. A large portion of this traffic is access from workstations to the fileservers. Data was collected from Router 2, the internal router.

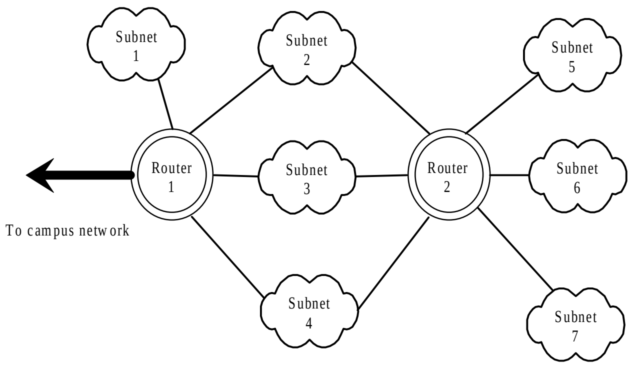

Figure 4 Configuration of the monitored network

The data was collected by polling the router using SNMP queries. The router was polled every 15 seconds. All of the available router MIB variables were collected. Of these variables, we studied the 14 that were active. The other variables changed infrequently and during the times of recorded faults, they rarely provided information that was not provided already by the active variables.

We monitored the router for approximately seven months. The monitoring was continuous for most of the time. During this time the log files generated by the *syslog* function were also saved to label the faults within the data. As *syslog* is not targeted specifically for network errors, only a subset of all network problems will be reported by *syslog*. The network related problems reported by *syslog* are usually severe. In fact, the type of network problem we found to be most commonly reported by *syslog* was server not responding. This type of message indicates a severe problem that may be occurring for a number of reasons (e.g. server down, path unavailable). The *syslog* messages do not provide any information about the cause of the problem.

5. Experimental Results

The system proposed for anomaly detection was tested on a set of 10 faults observed on the network in Figure 4 between October 1995 and March 1996. Most of the faults (9 out of 10) have been recorded as server not responding. The remaining fault is a report of excess Ethernet collisions on one of the subnets. Due to the mechanism currently used to log faults on this network (*syslog*), we could only observe types of faults where a service provided by the network is not operational. This mechanism provided accurate reports of severe faults, but no account of less severe network faults.

Although nine of the faults studied were the same, we did not observe the same types of changes in the data from fault to fault. This can possibly be traced to the fact that the faults were caused by different sets of circumstances or root causes. Even if the root causes were similar, the problems could have manifested themselves differently due to the network environment at the time they occurred. Specific root cause information may not be available without knowledge of the implementation details of the nodes comprising the network.

The results along with comparisons with threshold methods are shown for one of the faults. The results for the remaining faults are summarized.

5.1. Our Results

One of the faults the system was tested on was a fault that was reported as *server not responding* between 6:33 am and 6:36 am on December 23, 1996. The fileserver that was not responding was on Subnet 2. A total of 13 machines reported this problem (7 on Subnet 2, 4 on Subnet 3, and 2 on Subnet 4).

The results for the posterior probabilities estimated:

p(network = abnormal | MIBs),
p(IF = abnormal | MIBs),
p(IP = abnormal | MIBs),
and p(UDP = abnormal | MIBs)

are shown in Figure 5.

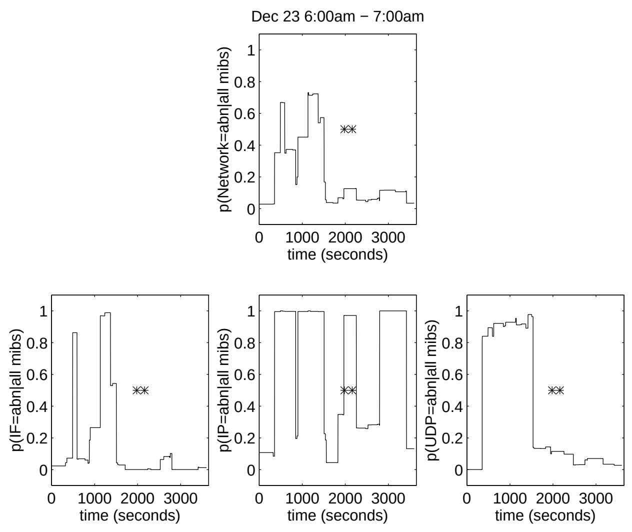

Figure 5 Results using 1 hour learning window

The asterisks denote the fileserver downtime period. Abnormal behavior is detected in the Network approximately 12 minutes before the server is reported unreachable. Anomalies are present before the problem in all three network functions, but only IP detects an anomaly during the crash. Similar results were obtained when a 4 hour learning window was used.

It is important to keep in mind the structure of the network in Figure 4. The router that we are monitoring continues to route all other traffic normally. The fileserver being down is not problematic to the router, but still we are able to detect the problem by monitoring the router. The router is able to detect that there is an anomaly in the network through changes to its MIB variables. Therefore, the results that we have are from the routers view of the network.

5.2. Comparisons

Since thresholds are commonly used to detect faults, we compared our results to those obtained using a feature set of an upper and lower threshold. The feature is whether the particular variable is within the thresholds or not. To combine the information from each MIB variable, we counted the total number of variables exceeding their thresholds at each time instance. The thresholds were calculated using learning windows of 1 hour, 4 hours and 1 week. The first two correspond to the windows used by the monitoring system. The third corresponds to the common practice of determining threshold levels using a large amount of data.

The results from all of these methods are shown in Figure 6. The asterisks denote the period where the fileserver was down. The results for 1 and 4 hour learning windows are identical. Both have small peaks where thresholds have been exceeded by 3 of the 14 MIB variables. The results using a 1 week learning window are essentially the same for the entire hour, thereby providing no useful information.

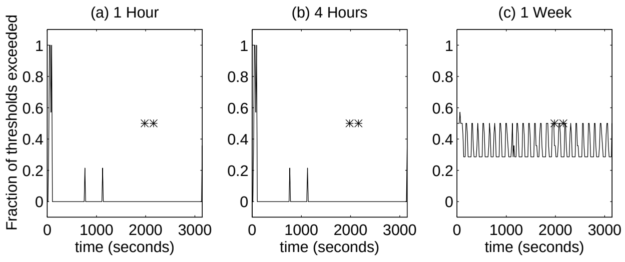

Figure 6 Results using thresholds

5.3 Composite results

More generally, 7 out of the 10 faults studied were detected using a 1 hour learning window and 5 out of 10 were detected using a 4 hour learning window. We considered a fault to be detected if the posterior probability that Network is abnormal is greater than 0.5.

These results need to be put into perspective. Ideally, we could calculate the number of times that a fault is detected when there is no fault (false alarms). Since we only have labels from the log file for the severe faults, it is not clear where other faults should or should not be detected. Therefore, to get some measure of the sensitivity we calculated the percentage of time that the posterior probability of the Network, IF, IP, and UDP is abnormal. The following figures indicate the percentage of abnormal time.

Network abnormal
- 6.29% of time using 1 hr learning window
- 3.85% of time using 4 hr learning window

IF abnormal
- 35.97% of time using 1 hr learning window
- 24.85% of time using 4 hr learning window

IP abnormal
- 42.03% of time using 1 hr learning window
- 33.32% of time using 4 hr learning window

UDP abnormal
- 42.24% of time using 1 hr learning window
- 30.29% of time using 4 hr learning window

Abnormalities at the highest level of our monitoring system (Network) are being detected a small percentage of the time. Therefore the detection of the test faults is significant. On the other hand, the faults that were not detected may not have had symptoms present at the router, or the features we used may not have captured adequate information to detect symptoms that were present.

6. Conclusion

We have shown that it is possible to use an adaptive learning machine to detect network faults without using models of specific faults. The Bayesian network provided a theoretical framework within which we were able to use prior knowledge to determine a structure and learn the normal behavior of the measurement variables. The system was tested on real data involving a fileserver crash on a computer network. It was able to successfully detect something abnormal approximately 12 minutes before the fileserver crashed. The detection was done from a router on the network which was operating properly. The router detected that something was wrong in the network from changes to its measurement variables.

With early detection, the network manager can be warned of impending failure, take corrective action and avoid the failure and costly downtime. Our approach is able to accomplish early detection by recognizing deviations from normal behavior in each of the measurement variables, correlating this information in time and then combining the information in a probabilistic framework.

This work is viewed as a first step in the direction of an automated fault management system that can generalize from network to network with minimal network specific information required a priori. The fault management problem is very complex and the nature of the problem evolves as networks evolve. Future work involves expanding the scope of the experiment, as well as further investigation of the methods.

The Bayesian formulation of this detection problem can be extended to incorporate more types of information or MIB groups. It can also be extended to combine the observations of several nodes in the network at the central network manager. With this extension, the Bayesian network's ability to estimate probabilities with incomplete information could be utilized. In addition, the use of learning to identify other possible structures for the Bayesian network is also an area for further investigation. Another area to investigate is better usage of the fault information that is available. This information can be used to help detect and diagnose known faults and to improve the feature extraction.

Acknowledgments

The authors would like to thank Dave Hollinger, Roddy Collins, and Nathan Schimke for helping us with the data collection. We would also like to thank John Miller, Ken Vastola, and George Nagy for helpful discussions and suggesting related references. The support of the National Science Foundation (ECS-9312594 and (CAREER) IRI-9502518) is gratefully acknowledged.

References

[1] U. Appel, A.V. Brandt, "Adaptive Sequential Segmentation of Piecewise Stationary Time Series," *Information Sciences,* vol. 29, 1983, pp. 27-56.

[2] C. Cortes, L.D. Jackel, W-P Chiang, "Predicting Failures of Telecommunication paths: Limits on Learning Machine Accuracy Imposed by Data Quality," *Proceeding of the International Workshop on Applications of Neural Networks to Telecommunications 2,* Stockholm, 1995.

[3] R.H. Deng, A.A. Lazar, W. Wang, "A Probabilistic Approach to Fault Diagnosis in Linear Lightwave Networks," *IEEE JSAC,* vol. 11, no. 9, Dec. 1993, pp. 1438-1448.

[4] D. Heckerman, "A tractable algorithm for diagnosing multiple diseases," *Proceedings of the Fifth Workshop on Uncertainty in Artificial Intelligence,* Windsor, ON, 1989, pp. 174-181.

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

[6] C. Hood, "Intelligent Detection For Fault Management of Communication Networks," Ph.D. Dissertation, Rensselaer Polytechnic Institute, 1997.

[7] G. Jakobson, M.D. Weissman, "Alarm Correlation," *IEEE Network,* Nov. 1993, pp. 52-59.

[8] A.A. Lazar, W. Wang, R. Deng, "Models and Algorithms for Network Fault Detection and Identification: A Review," *ICC,* Singapore, Nov. 1992.

[9] W. Leland, M. Taqqu, W. Willinger and D. Wilson, "On The Self Similar Nature of Ethernet Traffic (extended version)," *IEEE/ACM Trans. Networking*, vol. 2, pp. 1-15, Feb., 1994.

[10] E.L. Madruga, L.M.R. Tarouco, "Fault Management tools for a Cooperative and Decentralized Network Operations Environment," *IEEE JSAC,* vol. 12, no. 6, Aug. 1994, pp. 1121-1130.

[11] R. Maxion, F. Feather, “A Case Study of Ethernet anomalies in a Distributed Computing Environment,” *IEEE Trans. on Reliability,* vol. 39, Oct. 1990, pp. 433-443.

[12] J. Pearl, *Probabilistic Reasoning in Intelligent systems: Networks of Plausible Inference.* San Mateo, CA: Morgan Kaufman, 1988.

[13] I. Rouvellou, “Graph Identification Techniques Applied to Network Management Problems,” Ph.D dissertation, Columbia University, 1993.

[14] P. Smyth, “Markov Monitoring with Unknown States,” *IEEE JSAC,* vol. 12, 1994, pp. 1600-1612.

[15] D.J. Spiegelhalter, A.P. Dawid, S.L. Lauritzen, R.G. Cowell, “Bayesian Analysis in Expert Systems,” *Statistical Science,* vol. 8, no. 3, 1993, pp. 219-288.

[16] Z. Wang, “Model of network faults,” Integrated Network Management I, B. Meandzija and J. Westcott (eds.), New York, NY, Elsevier Science Publishing Company, 1989.

[17] O. Wolfson, S. Sengupta, Y. Yemini, “Managing Communication Networks by Monitoring Databases,” *IEEE Transactions on Software Engineering,* vol. 17, no. 9, 1991.

[18] Y. Yemini, “A Critical Survey of Network Management Protocol Standards,” Telecommunications Network Management Into the 21st Century, S. Aidarous and T. Plevyak (eds.), New York, NY, IEEE press, 1994.