

# Statistical Detection of Enterprise Network Problems. \*

**Marina Thottan and Chuanyi Ji**

Department of Electrical, Computer, and Systems Engineering

Rensselaer Polytechnic Institute, Troy, NY 12180

e-mail: thottm@ecse.rpi.edu, chuanyi@ecse.rpi.edu

*Running Header: Statistical Problem Detection.*

Office purposes contact: Marina Thottan

Dept. of ECSE, JEC 6003, Rensselaer Polytechnic Institute

110, 8th street, Troy, NY 12180.

tel:(518)276-8424; fax:(518)276-2433.

## **Abstract**

The detection of network fault scenarios was achieved using an appropriate subset of Management Information Base (MIB) variables. Anomalous changes in the behavior of the MIB variables was detected using a sequential Generalized Likelihood Ratio (GLR) test. This information was then temporally correlated using a duration filter to provide node level alarms which correlated with observed network faults and performance problems. The algorithm was implemented on data obtained from two different network nodes. The algorithm was optimized using five of the nine fault data sets and it proved general enough to detect three of the remaining four faults. Consistent results were obtained from the second node as well. Detection of most faults occurred in advance (at least 5 mins) of the fault suggesting the possibility of

---

Supported by DARPA under contract number F30602-97-C-0274

prediction and recovery in the future.

*Key Words: MIB variables, hypothesis testing, AR, GLR, change detection, duration filter.*

## 1 Introduction

Fault management comes under the bigger umbrella of network management. Fault situations could arise from defective components, their transient failure, software failures or from operational errors [1]. As the demand for and dependence on enterprise networks increases, the variety in network devices and software also increases. Along with the variety come the problems of compatibility and coexistence of the network's different pieces. These problems make fault management a critical part of providing network reliability and Quality of Service (QoS) guarantees. The goal of this work was to detect potential network problems through network traffic measurements using the Management Information Base (MIB) variables [2].

Several network management software packages are commercially available; however these at best can only detect severe failures or performance issues such as a broken link or a loss of link capacity [3]. These methods do not capture the subtle changes in network traffic that are indicative of many common network problems (ex.file server crashes). Rule based methods have also been developed to detect certain subsets of faults. Unfortunately these methods require a data base of fault scenarios and rules for detection which often rely heavily on the expertise of the network manager . The rules thus developed are too specific to characterize all network fault scenarios that evolve with time. Thus most schemes based on Artificial Intelligence suffer from being dependent on prior knowledge about the fault conditions on the network and the rules developed do not adapt well to a changing network environment [4]. Research in the field of fault management is primarily focussed on fault localization by alarm correlation [5][6]. Fault localization is usually based on the assumption that the alarms provided by the network nodes are true and the relevance of temporal information in the alarms is ignored or assumed accurate. The issue

of temporal information has been partially addressed by introducing TMIBs which are time correlated MIB variables [7]. However, to the best of our knowledge the generation of time correlated alarms related to a fault at a network node is still an open problem.

The potential use of MIB variables in fault management has been explored previously by Hood and Ji [8]. In the present work a new, simpler fault detection algorithm has been developed which takes advantage of the interrelationships between MIB variables through a simple combination scheme, generating time correlated node level alarms. An appropriate subset of MIB variables was chosen based on their relevance to the network traffic. The alarms were generated proactively (i.e) before the fault actually occurred (refer Section 9). The problem of fault characterization was avoided by utilizing the statistical properties of the signal (specifically using the AR parameters). The simplicity of the algorithm and its adaptability to topological changes that are common in enterprise networks make it an attractive approach for online implementation. Since the processing is done locally, the method is expected to lend itself easily to a distributed implementation.

An adaptive GLR test was used to detect changes that occur in the traffic data collected from the network. The MIB variables represent a cross section of this traffic at the different layers of the protocol stack that defines the network. The changes detected in the signal (MIB variables) at the different protocol layers were then correlated so as to reflect the propagation of the change through the protocol stack. This correlated information was then used to declare an alarm at the node level. With very high probability the alarms obtained were found to be indicative of an impending fault.

The algorithm has been successfully implemented on a campus LAN network. Data was obtained from the two routers (internal router and gateway) associated with the LAN. The configuration and the nature of traffic of this campus network is rich enough to provide a small scale model of an enterprise network. As the algorithm developed does local processing of the MIB variables we believe that the

methods should scale well to the larger scale enterprise network environment.

## 2 Model

The network management system we developed is based on a network management protocol working in a client - server paradigm. The network manager is the client and the agent providing the data is the server. The network used was the Internet and the associated management protocol was SNMP [9]. The agents were different entities on the network that are capable of collecting MIB variables. The SNMP provided an organized structure to these variables (MIB) as well as a mechanism for communication. However, these variables by themselves were not sufficient to detect faults. The variables had to be further processed in order to obtain an indicator on the occurrence of network problems.

The node level alarm generation system consisted of two stages. These stages are depicted in Figure 1. In the first stage, the change points for each individual variable was detected. The data processing unit divided the time series into piecewise stationary segments. The piecewise stationary segments of the data were modelled using an AR (auto regressive) process. The data from the processing unit was analysed by the change detector which determined the generalized likelihood ratio. A sequential hypothesis test based on the power of the residual signals in the segments was performed to determine if a change had occurred. Each of these change points were then compared to the first two moments of the normal data. An alarm was raised at the variable level if the first two moments of the changed time series segment was significantly different from the normal data. The details of this approach are explained in Section 5. The first stage produced a series of alarms corresponding to changes observed in each of the individual MIB variables. These alarms were candidate points for fault occurrences.

In the second stage the variable level alarms were combined using the a priori information about the relationships between the MIB variables. This aspect is further discussed in Section 6. The time

correlated alarms corresponding to the faults were obtained as the output of the second stage.

### 3 Experimental Network

The experiments were conducted on the Local Area Network (LAN) of the Computer Science (CS) Department at Rensselaer Polytechnic Institute. The network topology is as shown in Figure 2. The CS network forms one subnet of the main campus network. The network implements the IEEE 802.3 standard. Within the CS network there are seven smaller subnets and two routers. All of the subnets use some form of CSMA (Carrier Sense Multiple Access) for transmission. The routers implement a version of the Dijkstra's algorithm. One router (router 2 called node 1) is used for internal routing and the other serves mainly as a gateway (router 1 called node 2) to the campus backbone. The external router or gateway also provides some limited amount of internal routing. The internal router has 6 interfaces and the external router has 4 interfaces with the CS subnets. The back bone is an FDDI (Fiber Distributed Data Interface) ring. There are 133 host addresses and 2 file servers. The majority of the host machines are SPARC stations and there are also a few Ultras and PCs. The network spans 2 buildings. It is a well designed network in the sense that there are no recurrent problems and the utilization is within the limitations of the bandwidth. The majority of the traffic is file transfers or web traffic which involves the workstations accessing the file servers. The speed of the network is 10Mb/s. The external gateway and the internal router are SNMP agents. The Management Information Base on these agents were polled (using a PERL script) every 15 seconds to obtain the measurement variables. Data was collected from the interface of subnet 2 with both the routers and at the routers themselves. Queries to the MIB were sent out from a machine (denoted as nm in Figure 2) on subnet 3. As there were no dedicated machines for data collection purposes the issues of storage space and the impact of polling on the network traffic dictated the choice of the polling frequency [7][10].

There were no network management measures in place on this network. Each machine on the network

ran the UNIX *syslog* function. This function generated messages that related to problems associated with the network, applications, or the specific machine itself. These *syslog* messages were used to identify network problems. One of the most common network problems was NFS server not responding. The *syslog* messages only reported that the file server was not responding, but was unable to identify the cause of the problem. Possible reasons for this problem are unavailability of network path or that the server was down. Although not all problems could be associated with *syslog* messages, those problems which were identified by *syslog* messages were accurately correlated with fault incidents. A description of the data sets used is provided in Table 1.

#### 4 Choice of Variables

The Management Information Base maintains 171 variables [11]. These variables fall into the following groups: System, Interfaces, Address Translation (*at*), Internet Protocol(*ip*), Internet Control Message Protocol (*icmp*), Transmission Control Protocol (*tcp*), User Datagram Protocol (*udp*), Exterior Gateway Protocol (*egp*), and Simple Network Management Protocol (*snmp*). Each group of variables describes a specific functionality of the network. Depending on the type of node monitored an appropriate group of variables was considered. In this work the nodes being monitored were the routers and a specific interface with subnet 2 (refer Figure 2). The *if* group of variables and the *ip* group of variables describe the traffic characteristics at a particular interface and at the network level respectively. Therefore we investigated the *if* and *ip* groups.

Within a particular MIB group there exists some redundancy. For example consider the variables interface Out Unicast packets (*ifOU*), interface Out Non Unicast packets (*ifONU*) and interface Out Octets (*ifOO*). The *ifOO* variable contains the same traffic information as that obtained using both *ifOU* and *ifONU*. In order to simplify the problem, redundant variables were not incorporated. Some of the variables, by virtue of their standard definition [2], were not relevant towards the detection of a

fault and therefore excluded. The relationships between the MIB variables of a particular group can be represented using a Case diagram [12]. The Case diagram for the *if* and *ip* variables are shown in Figure 3.

Five MIB variables were selected for our algorithm. In the *if* layer, the variables *ifIO* (In Octets) and *ifOO* (Out Octets) were used to describe the characteristics of the traffic going into and out of that interface from the router. Similarly in the *ip* layer three variables were used. The variable *ipIR* (In Receives), represents the total number of datagrams received from all interfaces of the router. *ipIDe* (In Delivers), represents the number of datagrams correctly delivered to the higher layers as this node was their final destination. *ipOR* (Out Requests), represents the number of datagrams passed on from the higher layers of the node to be forwarded by the ip layer.

The variables chosen correspond to what are called the filter counters [12]. A filter counter is a MIB variable that measures the level of traffic at the input and at the output of each layer. The five MIB variables chosen are not strictly independent. However the relationships between these variables are not obvious and depend on parameters of the traffic such as source and destination of the packet, processing speed of the agent and the actual implementation of the protocol. Some of these dependencies have been incorporated at the combination stage (refer Section 6).

## 5 Change Detection

The detection algorithm employed at the node level was based on the premise that the statistical properties of the MIB variables change in response to fault conditions. It has been experimentally shown that changes in the statistics of traffic data can be used to detect faults [1][3][13]. The detection algorithm was implemented independently on each MIB variable. The increments in the MIB counters were obtained every 15 seconds and the data thus generated constituted a time series. Representative

time series data from both the *ip* and the *if* layers are shown in Figure 4. These two data traces reveal each variables' behavior over a two hour period. Note that the data exhibits a high degree of nonstationarity. Piecewise stationary auto-regressive models have been used to successfully describe such nonstationary stochastic time series signals [14]. Thus the MIB data was divided into 2.5 minute (10 time lags) piecewise stationary windows. Within a time window of size  $N$ , ( $N=10$ ), the MIB data was linearly modelled using a first order AR process. Using these piecewise stationary windows a sequential hypothesis test was performed using the GLR [15] [16]. Non overlapping windows were used in order to obtain uncorrelated residuals within each window.

For a given MIB variable consider two adjacent time windows  $R(t)$  and  $S(t)$  of lengths  $N_R$  and  $N_S$  respectively as in Figure 5 ( $N_R = N_S = 10$ ). Let us first consider  $R(t)$ :

$$R(t) = \{r_1(t), r_2(t), \dots, r_{N_R}(t)\} \quad (1)$$

Here we can express any  $r_i(t)$  as  $\tilde{r}_i(t)$  where  $\tilde{r}_i(t) = r_i(t) - \mu$  and  $\mu$  is the mean of the segment  $R(t)$ . Now  $\tilde{r}_i(t)$  was modelled as an AR order  $p$  process ( $p=1$ ) with a residual error  $\epsilon_i$ ,

$$\epsilon_i(t) = \sum_{k=0}^p \alpha_k \tilde{r}_i(t-k), \quad (2)$$

where  $\alpha_R = \{\alpha_1, \alpha_2, \dots, \alpha_p\}$  are the AR parameters. From Equation(2) and assuming that each time sample is drawn from an  $N(0, \sigma_R^2)$  distribution [15][17], the joint likelihood of the residual time series was obtained as

$$p(\epsilon_{p+1}, \dots, \epsilon_{N_R} / \alpha_1, \dots, \alpha_p) = \left( \frac{1}{\sqrt{2\pi\sigma_R^2}} \right)^{\dot{N}_R} \exp \left( \frac{-\dot{N}_R \hat{\sigma}_R^2}{2\sigma_R^2} \right), \quad (3)$$

where  $\sigma_R^2$  is the variance of the segment  $R(t)$ , and  $\dot{N}_R = N_R - p$ , and  $\hat{\sigma}_R^2$  is the covariance estimate of  $\sigma_R^2$ . We obtained a similar expression for the segment  $S(t)$ . Now the joint likelihood  $l$  of the two segments  $R(t)$  and  $S(t)$  is given as,

$$l = \left( \frac{1}{\sqrt{2\pi\sigma_R^2}} \right)^{\dot{N}_R} \left( \frac{1}{\sqrt{2\pi\sigma_S^2}} \right)^{\dot{N}_S} \exp \left( \frac{-\dot{N}_R \hat{\sigma}_R^2}{2\sigma_R^2} \right) \exp \left( \frac{-\dot{N}_S \hat{\sigma}_S^2}{2\sigma_S^2} \right) \quad (4)$$

The expression for  $l$  is a sufficient statistic and was used to perform a binary hypothesis test. The two hypotheses were  $H_0$ , implying that no change was observed between segments, and  $H_1$ , implying that a change was observed. Under the hypothesis  $H_0$  we have,

$$\alpha_R = \alpha_S, \quad (5)$$

$$\sigma_R^2 = \sigma_S^2 = \sigma_P^2. \quad (6)$$

where  $\sigma_P^2$  is the pooled variance. Under hypothesis  $H_1$ , we have,

$$\alpha_R \neq \alpha_S, \quad (7)$$

$$\sigma_R^2 \neq \sigma_S^2. \quad (8)$$

Using the conditions in Equations (6 and 8) we obtained the likelihood ratio as,

$$\lambda = \sigma_P^{-(N_R+N_S)} \sigma_R^{N_R} \sigma_S^{N_S} \exp \left( \frac{-\hat{\sigma}_P^2 (N_R + N_S)}{2\sigma_P^2} + \frac{1}{2} \left[ \frac{N_R \hat{\sigma}_R^2}{\sigma_R^2} + \frac{N_S \hat{\sigma}_S^2}{\sigma_S^2} \right] \right). \quad (9)$$

Furthermore on using the maximum likelihood estimates for the variance terms, we get the log likelihood ratio to be,

$$-\ln \lambda = N_R (\ln \hat{\sigma}_P - \ln \hat{\sigma}_R) + N_S (\ln \hat{\sigma}_P - \ln \hat{\sigma}_S) \quad (10)$$

The log likelihood ratio ( $-\ln \lambda$ ) was compared with an optimally chosen threshold  $h$  ( $h=5-20$ , depending on the MIB variable). Segment boundaries where the threshold was exceeded were considered to be change points. That is,

$$-\ln \lambda > h \implies H_1, \quad (11)$$

$$\leq h \implies H_0. \quad (12)$$

( $\ln \lambda$ ) can also be interpreted as a measure of the information loss incurred by accepting the hypothesis  $H_0$  [18]. A second hypothesis test was conducted for all change points detected by the first hypothesis test. In the second hypothesis test the mean  $\mu_P$  and variance  $\sigma_P$  of the pooled segment were compared with the normal mean  $\mu_0$  and variance  $\sigma_0$  using a likelihood ratio and a second threshold  $\eta$ . The normal

mean and variance were computed from the normal data over a twenty four hour period. The two hypotheses considered were  $H_0$  with a distribution of  $N(\mu_0, \sigma_0^2)$  implying that the change was normal and  $H_1$  with a distribution of  $N(\mu_P, \sigma_P^2)$  implying an abnormal change had occurred. Abnormal changes were noted as variable level alarms. Our algorithm depends on the following parameters:

- \* the order  $p$  of the AR process,
- \* the threshold values  $h$  and  $\eta$
- \* the normal mean  $\mu_0$  and variance  $\sigma_0$ .

The choice of these parameters are discussed in Section 7.

## 6 Combination

The variable level alarms obtained in phase 2 of the detection algorithm were combined using a duration filter. The duration filter was implemented on the premise that a change observed in a particular variable would propagate into another variable that was higher up in the protocol stack. For example, in the case of the *ifIO* variable, the flow of traffic is towards the *ipIR* variable. Using the relationships from the Case diagram representation shown in Figure 3, the possible transitions between the chosen variables were determined (see Figure 6). The duration filter was designed to detect all four transition types. The time interval between transitions from one variable to another represents the duration filter. The length of the duration filter for each transition was experimentally determined and these values are denoted on the arcs of the transitions in Figure 6. Transitions that occur within the same protocol layer (*ipIR* to *ipIDe*) required a duration filter of length 15 seconds <sup>1</sup>. However, for transitions that occurred between the *if* and the *ip* layers a significantly longer duration filter of 20 to 30 min was required.

---

<sup>1</sup>sampling rate of the MIBs

## 7 Implementation Issues

The implementation steps of the algorithm are shown in the flow chart in Figure 7. The broad arrows indicate that a set of parameters corresponding to the five chosen MIB variables are being passed to the subsequent stage. The narrow arrows denote a single output.

*AR order ( $p$ ):* Adequate representation of the signal and parsimonious modelling [14] were competing requirements; hence a trade off between these two issues was necessary. The accuracy of the model was measured in terms of Akaike's final prediction error (FPE) criterion [19]. The order corresponding to a minimum prediction error was the one that best modelled the signal. However due to singularity issues, there was a constraint on the order  $p$  [15] expressed as:

$$0 \leq p \leq 0.1N, \quad (13)$$

where  $N$  is the length of the sample window. Furthermore, if we wish to detect a change segment of length  $Q$  or longer, then the sample window size has an upper limit [15],

$$N \leq 0.7Q \quad (14)$$

In our algorithm we set  $Q=15$  time lags (3.75 min, approx duration of faults, see Table 1) and  $N=10$  time lags (2.5 mins). Using these values for  $N$  and  $Q$  and subject to the constraint Equations ( 13) and ( 14), we found the only appropriate order for  $p$  that minimizes the FPE for each of the MIB variables to be 1.

*Threshold  $h$  and  $\eta$ :* The threshold  $h$  was optimised experimentally on data sets 3 and 6. The same values were used on all the other data sets, both as  $h$  in the first stage and as  $\eta$  in the second stage. The optimized values worked well on the other data sets (refer section 9). Since the  $ip$  variables represent traffic at a lower resolution (in units of datagrams), and the  $if$  variables at a higher resolution (in units of octets), the thresholds used for the  $ip$  variables were half that of the  $if$  variables.

*Normal mean and variance:* The normal values of mean  $\mu_0$  and variance  $\sigma_0$  used in stage 2 of the detection algorithm were computed over a 24 hour period for each variable. These values were obtained from data sets 3 and 6 and they worked well for the remaining data sets (refer Section 9).

## 8 Performance Analysis

The performance of the algorithm was assessed in terms of the probability of detection  $P_D$ , the probability of false alarm  $P_F$  and the average number of alarms generated in an hour. These measures obtained for the two nodes on different data sets are shown in Tables 3 and 4. The probability of detection  $P_D$  was calculated as the ratio of the total number of correct matches to total number of known faults within the data set [1]. All alarms generated within one hour before or 15 minutes after the fault occurred were considered to be true alarms. The probability of false alarm  $P_F$  was calculated as the ratio of the total number of false alarms generated to the total number of data samples available in the data set [1]. By recent proactive alarm, we refer to the alarm time that was closest to the actual fault. The probability of false alarm was computed conservatively. The effects due to the hysteresis of the fault and the persistence of alarms before a fault occurs, are included in the computation of the false alarm rate.

The performance of the algorithm was also studied under different sets of thresholds, and the corresponding values for the probability of detection and probability of false alarm are shown in Table 5. Thresholds for each of the MIB variables were set to be five above and five below the optimal values. It was observed that as the thresholds were increased the probability of detection increased along with the false alarm rate. The optimal threshold chosen was cost-benefit related. The cost refers to the frequency of false alarms and the benefit is the accuracy of fault prediction.

The algorithm is based on a linear model, rendering it feasible for online implementation. The complexity

as a function of the number of model parameters is  $O(4N)$ , where  $N$  is the number of input MIB variables. For the case under consideration we have  $N = 5$ . The computational complexity expressed in terms of the number of floating point operations performed is 2495 per cycle, which is approximately 15 floating point operations per sec. One cycle refers to the processing done to generate one node level alarm.

## 9 Results and Discussion

The alarm generation system was implemented on both nodes ( Routers 1 and 2 in Figure 2). The average number of alarms generated per hour on each variable was computed over all the data sets of node 1. The results of the variable level alarm generation are summarized in Table 2. Representative outputs of stage 1 are shown in Figure 8. All of the chosen variables showed the ability to detect changes associated with fault scenarios. However, not all the variables were capable of detecting all the faults. The *ip* level variables detected more faults than the *if* variables, suggesting that at the router level the *ip* variables alone are sufficient for fault detection. But in order for the router to detect problems which occur locally within a subnet, it is necessary to know the total throughput to and from that interface (represented by the *ifIO* and *ifOO*).

The duration filter helped to reduce the node level alarms as compared to the variable level alarms. Using the duration filter we were able to reduce the average number of alarms per hour to 1.4. The results after the combination stage on both nodes 1 and 2 are shown in Tables 3 and 4. Note that seven of the nine faults were detected. In six out of seven faults detected, the alarms were generated before the fault actually occurred. Despite the lack of information from the *if* variables of subnet 3 (data set 6) our algorithm was able to detect one of the two faults on that subnet. Since these faults were reported by only 2 or 3 machines outside subnet 3, we did not expect a significant effect on the *ip* variables. The detection of the first fault was attributed to its repeated occurrence. The weakness of the duration

filter however, is an additional parameter (length of the duration filter) that must be experimentally optimised.

The ability of the algorithm to generalize to a second node was also studied. The results obtained for the second node are summarized in Table 4. We were unable to obtain data from this node for data sets 2 and 5. For node 2, six out of the seven faults were detected. The one fault that went undetected was the second fault on data set 6 which occurred on subnet 3 (see above for explanation).

Our algorithm was capable of detecting faults that occurred at different times of the day. Representative outputs from stage 2 of the algorithm are shown in Figures 9 and 10. The algorithm is capable of detecting changes that persist for lengths greater than 15 time lags (3.75 mins). To be able to increase the sensitivity of the algorithm to track even more abrupt changes, the length of the sample window  $N$  ( $=10$ ) has to be further reduced. However this would require an AR order  $p < 1$ . This problem can be circumvented using a higher polling rate. As can be seen in Figures 9 and 10 there was a relatively large number of alarms in the early part of the day. However by using the time of day as an additional feature these alarms can be further reduced.

Furthermore, once a fault occurs the system requires time to return to normal. This duration was also detected as change points, although they do not necessarily correlate to a fault. Alarms that are generated at these times can be avoided by allowing a renewal time immediately after a fault has been detected. Thus the addition of hysteresis will help reduce the false alarms.

## 10 Conclusions

This work shows that statistical methods complemented with a good understanding of network protocols can be used to address the problem of network fault detection. The advantages of this method are its independence from specific fault descriptions and ease of implementation. In addition our fault detection

scheme is capable of categorising network problems based on the type of transition that detects it. Future efforts will be directed towards diagnosis of faults using this information.

The algorithm is also an improvement over simple thresholding methods since it takes into account the relationships between the measurement variables that reflect the properties of the network. Implementation on a second node suggests the scalability of our algorithm to larger heterogeneous networks. Efforts are currently under way to increase the specificity of the algorithm through more sophisticated combination schemes and thereby reduce the false alarm rate.

The effectiveness of the detection can be evaluated only with the aid of a large number of test samples. As most faults cannot be reproduced accurately in a controlled environment there is a need to obtain simulated fault sets. This would require accurately modelling the normal behavior of the different MIB variables and then altering the statistical behavior to incorporate the fault. This is part of our future work.

Our algorithm has managed to detect network problems on a fully functional network with complex traffic patterns. Since the network that was considered can be related to any enterprise network both in complexity and in terms topology, our algorithm can be readily applied to enterprise networks.

## 11 Acknowledgments

The authors would like to thank Cindy Hood, Dave Hollinger, Nathan Schimke and Roddy Collins for their generous help with the data collection. We thank Ken Vastola and Joe Hellerstein for helpful discussions on the topic and the reviewers for their comments and suggestions. The support of DARPA (F30602-97-C-0274) and the NSF ((CAREER) IRI-9502518) is gratefully acknowledged.

## References

- [1] F. Feather and R. Maxion. Fault detection in an ethernet network using anomaly signature matching. In *ACM SIGCOMM*, volume 23, Sept 1993.
- [2] K. McCloghrie and M. Rose. Management information base for network management of tcp/ip-based internets: Mib 2. *RFC1213*, 1991.
- [3] R.E. Moore. Problem detection, isolation and notification in systems network architecture. In *Proc. IEEE INFOCOM*, pages 377–381, 1986.
- [4] Kormann Franceschi and Westphall. Performance evaluation for proactive network management. In *Proc. IEEE ICC*, 1996.
- [5] I. Rouvellou and G.W. Hart. Automatic alarm correlation for fault identification. In *Proc. IEEE INFOCOM*, pages 553–561, 1995.
- [6] I. Katzela and M. Schwarz. Schemes for fault identification in communication networks. *IEEE/ACM Trans.Networking*, 3:753–764, 1995.
- [7] T.K. Apostolopoulos and V.C. Daskalou. Temporal network management model, concepts and implementation issues. *Computer Communications*, 20:694–708, 1997.
- [8] C.S. Hood and C. Ji. Proactive network fault detection. *Proceedings of INFOCOM, Kobe, Japan*, 1997. Also available from <http://neuron.ecse.rpi.edu/>.
- [9] W. Stallings. *SNMP, SNMPv2, and CMIP The practical guide to Network Management Standards*. Addison-Wesley Publishing Company, 5 edition, 1994.
- [10] M. Thottan, M. Hulber, K. Vastola, and C. Ji. Analysis of data collection. *Computer Networks and Adaptive Systems Laboratory Technical Report:1*, 1998. Also available from <http://www.rpi.edu/~thottm>.

- [11] M.T. Rose. *The Simple Book: An Introduction to Internet management*. Prentice Hall Series in Innovative Technology, 2 edition, 1996.
- [12] J.D. Case and C. Partridge. Case diagrams: A first approach to diagrammed management information bases. *Computer Communication Review*, 19:13–16, Jan 1989.
- [13] R. Maxion. A case study of ethernet anomalies in a distributed computing environment. *IEEE transactions on Reliability*, 39(4), Oct 1990.
- [14] Box and Jenkins. *Time Series Analysis, Forecasting and Control*. Holden Day Series, 1976.
- [15] U. Appel and A.V. Brandt. Adaptive sequential segmentation of piecewise stationary time series. *Information Sciences*, 29:27–56, 1983.
- [16] P.V. Desouza. Statistical tests and distance measures for lpc coefficients. *IEEE trans on Acoustics, Speech and Signal Processing*, 25(6), Dec 1977.
- [17] H.B. Mann and A. Wald. On the statistical treatment of linear stochastic difference equations. *Econometrica*, 11, 1943.
- [18] A.V. Brandt. An entropy distance measure for segmentation and clustering of time series with application to eeg signals. In *Sixth International Conference on Pattern Recognition Munich*, 1982.
- [19] H. Akaike. A new look at statistical model identification. *IEEE trans. on Automatic Control*, 19(6):716–723, Dec 1974.

Marina Thottan received her M.S. in Physics at P.S.G. College of Arts and Science, Coimbatore, India. She received her M.S. in Biomedical Engineering at Rensselaer Polytechnic Institute where she is currently working towards a Ph.D in Electrical and Computer Systems engineering.

(<http://www.rpi.edu/~thottm>)

Chuanyi Ji received the BS (with honors) from Tshinghua University, Beijing, China in 1983, an MS from University of Pennsylvania, Philadelphia in 1986, and a Ph.D from California Institute of Technology,

Pasadena, in 1992, all in Electrical Engineering. In November 1991, she joined the faculty at Rensselaer Polytechnic Institute, Troy, where she is now an Associate Professor of Electrical, Computer and Systems Engineering. Her research interests are in the areas of stochastic modeling, analysis and control of multi-media traffic, management of hybrid computer communication networks, and adaptive learning systems.

Dr. Ji is a recipient of NSF CAREER award in 1995.

(<http://www.ecse.rpi.edu/homepages/chuanyi>)

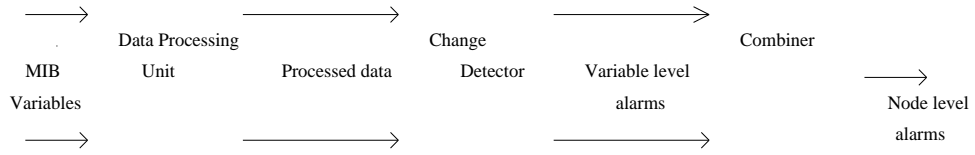


Figure 1:

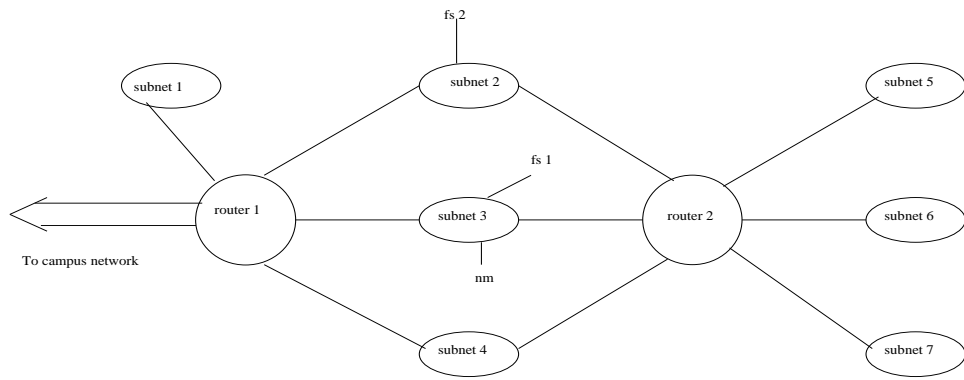


Figure 2:

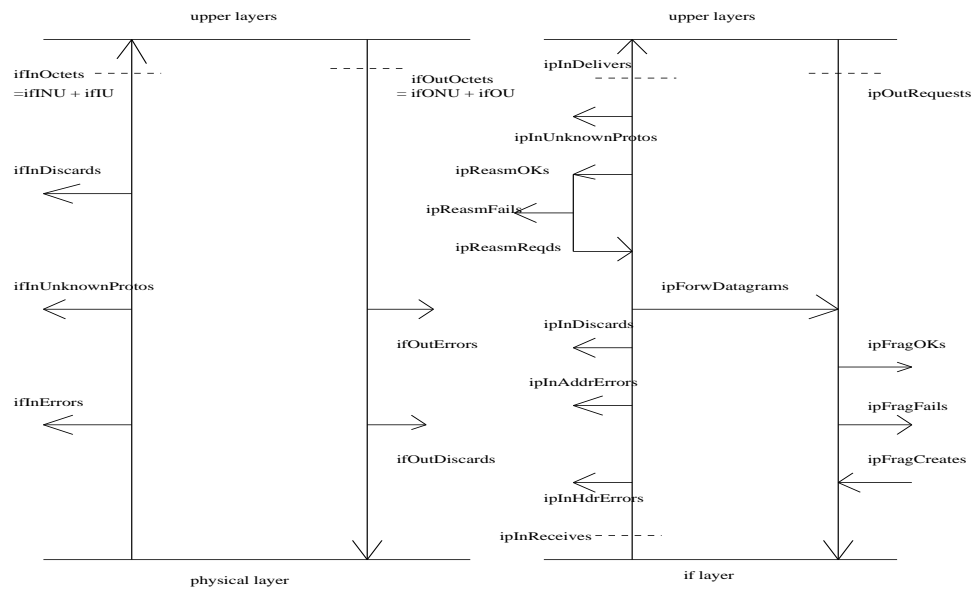


Figure 3:

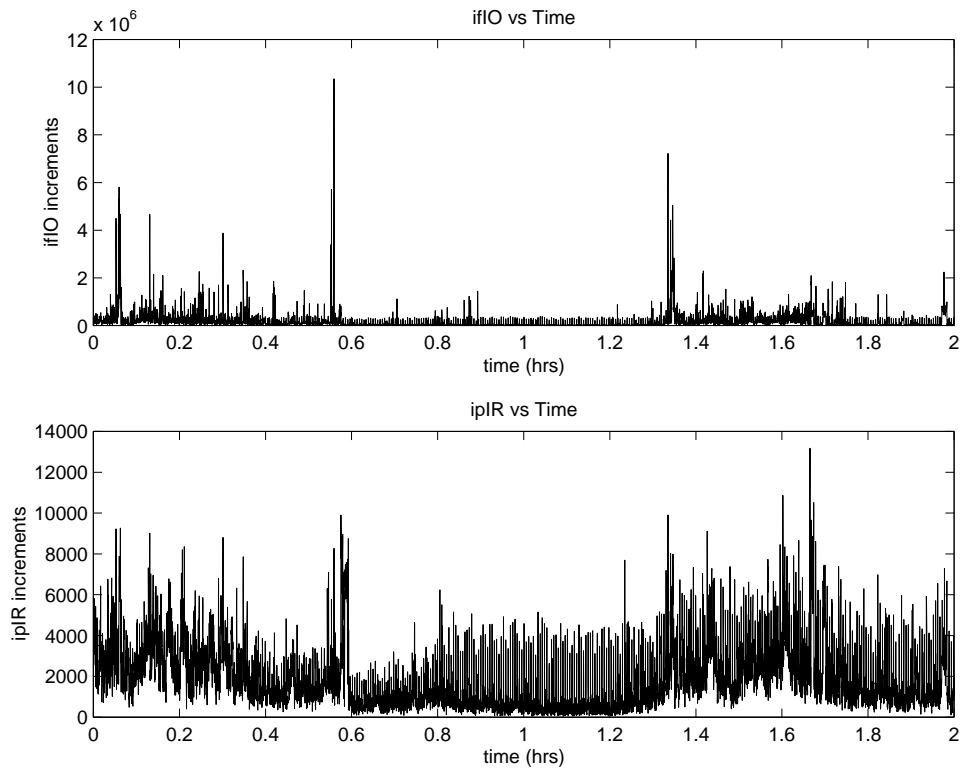


Figure 4:

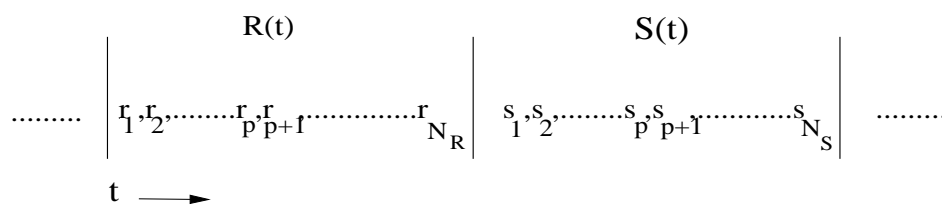


Figure 5:

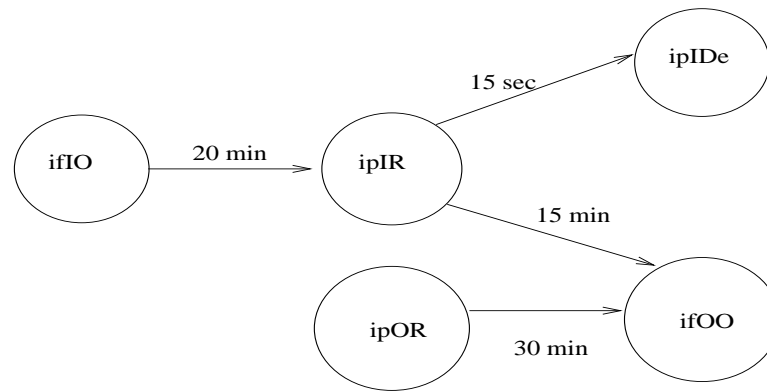


Figure 6:

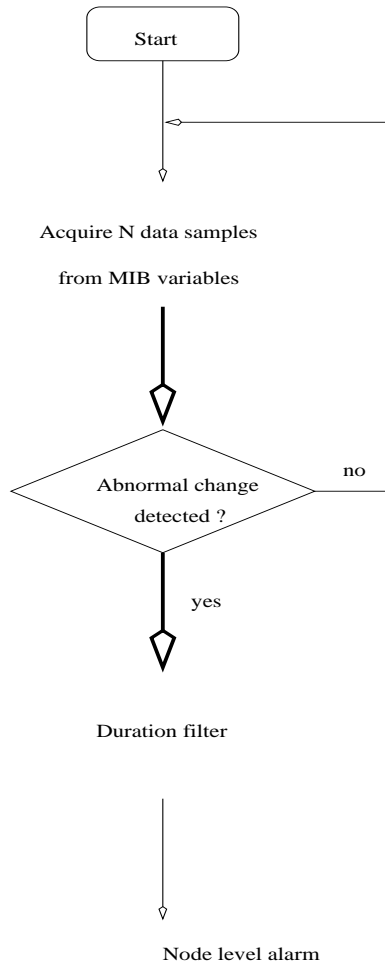


Figure 7:

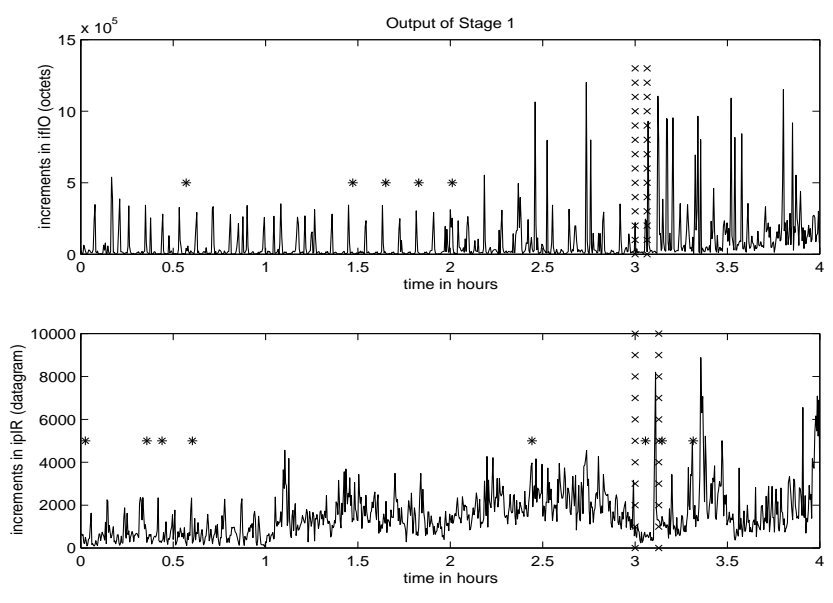


Figure 8:

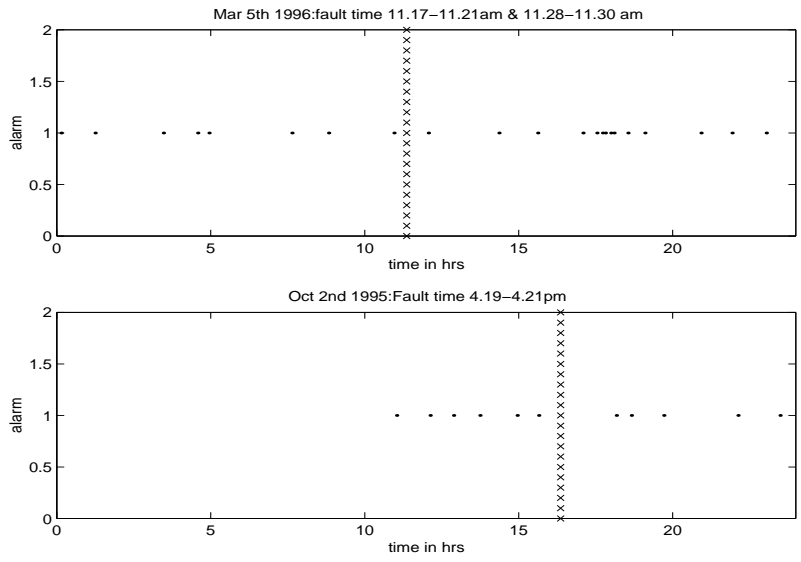


Figure 9:

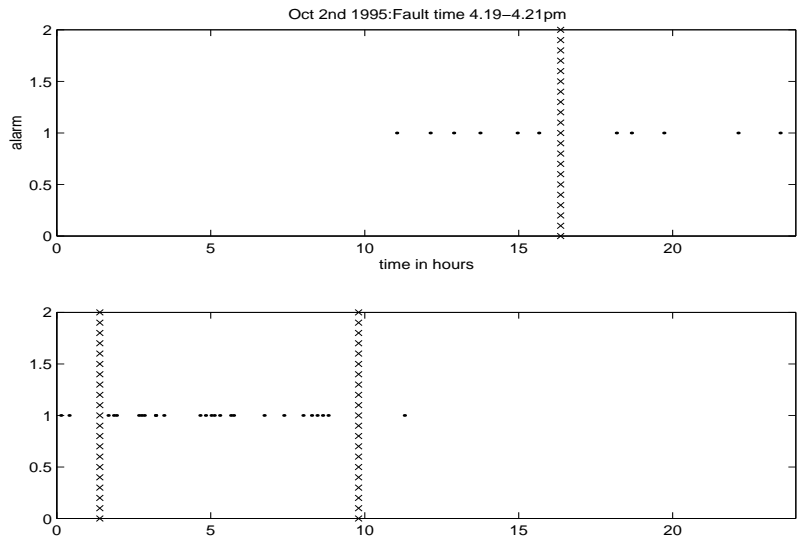


Figure 10:

Captions for Figures:

Figure 1: System Model

Figure 2: Configuration of the Monitored Network

Figure 3: Case Diagram for the *if* and *ip* variables

Figure 4: Representative Trace of *if* and *ip* Variables

Figure 5: Piecewise Stationary Segments

Figure 6: Transitions between MIB Variables

Figure 7: Flow Chart

Figure 8: Output at Stage 1 of Node 1

Figure 9: Output at Stage 2 of Node 1

Figure 10: Output at Stage 2 of Node 2

data set no	no of faults in data set	fault location (subnet)	Time of day/ duration of fault	no of machines affected on subnet 2	no of machines affected on subnet 3	no of machines affected out side 2 and 3
1	1	2	4.19 - 4.21pm	8	6	18
2	1	2	11.10 - 11.17am	12	8	16
3	3	2	8.23 - 8.26pm	5	6	8
		2	1.23 - 1.25am	2	0	0
		2	9.48 - 9.52am	9	6	9
4	1	2	6.33 - 6.36am	7	4	2
5	1	2	9.36 - 9.41pm	8	5	5
6	2	3	11.17 - 11.21pm	1	5	1
			11.28 - 11.30pm			
		3	3.22 - 3.26pm	1	8	2

Table 1:

MIB variable	avg no of alarms per hour	no: of faults detected
ifIO	0.91	4/9
ifOO	1.14	5/9
ipIR	3.97	7/9
ipFD	3.98	7/9
ipIDe	2.08	7/9
ipOR	1.19	9/9

Table 2:

data set no:	no: of known faults in data set	$P_D$	recent proactive alarm in min	$P_F$	avg no: of alarms per hour
1	1	1	40	0.0037	0.89
2	1	1	8.7*	0.0038	0.91
3	3	1	31.4 59.8 59.4	0.0071	1.74
4	1	1	18.4	0.0060	1.45
5	1	0		0.0065	1.55
6	2	0.5	31	0.0076	1.83

Table 3:

data set no:	no: of known faults	$P_D$	recent proactive alarm in min	$P_F$	avg no: of alarms per hour
1	1	1	43.8	0.0093	2.23
3	3	1	38.2 22.3 11.2	0.0075	1.86
4	1	1	8.2	0.0106	2.57
6	2	0.5	21.6	0.0040	0.96

Table 4:

threshold	$P_D$	$P_F$
lower	1	0.0134
optimal	0.75	0.0058
higher	0.305	0.0017

Table 5:

Captions for Tables:

Table 1: Description of Fault Sets

Table 2: Summary of Stage 1 Results for Node 1

Table 3: Summary of Stage 2 Results for Node 1>(\* detected after the fault)

Table 4: Summary of Stage 2 Results for Node 2

Table 5: Optimization of thresholds