Statistical Online Monitoring of Network Operations for Fast Detection of Faults Performance Problems

HASSAN HAJJI, BEHROUZ H. FAR
Department of Information and Computer Sciences
Saitama University, Urawa-shi, 338-8025, JAPAN
Email:hajji(far)@cit.ics.saitama-u.ac.jp

This paper addresses the problem of network monitoring, for fast detection of performance problems. The network behavior is modeled as clusters of dependent objects of the Management Information Base-II (MIB-II). Each cluster is modeled as finite mixture of simple regression models. Network baseline parameters are identified from routine operation data. An online residual generation method, based on successive parameter identification, is introduced. Residuals are shown to be stationary, with mean zero under normal operation. Performance problems are characterized by sudden jumps in the mean. Detection is formulated as an online change point problem, where the task is to process residuals and raise alarms as soon as anomalies occur. An analytical expression of false alarm rate allows us to choose the threshold, automatically. Experimental results on a Saitama university and Tokyo international university networks showed that the monitoring agent is able to detect even slight changes in the characteristics of the network, while maintaining a low false alarm rate.

統計手法に基づいたネットワーク運用のオンライン監視と故障とパフォーマンスの低下の早期検出への応用

ハジ ハサン、 B.H.ファー
埼玉大学情報システム工学部
浦和市 338-8025
Email: hajji (far)@cit.ics.Saitama-u.ac.jp

本論文では、ネットワークの故障やパフォーマンス低下の自動検出について述べる。ネットワーク、MIB-II を用いて、混合分布としモデル化します。ネットワークの正常な運用パラメータを、通常の運用データから、推定を行う。そして、逐次的なパラメータの推定をもとに残差を生成する新たな手法を提案する。その残差の平均は正常な場合はゼロであり、異常が発生するとその平均はゼロではない値に跳躍するという特性を持つ。よって、ネットワークの異常検出は Sequential Analysis で知られている変化点検出（change point detection）として定式化することに着目し、誤報率（false alarm rate）の解析解を定めることによって、設計の閾値は自動的に調整される。埼玉大学と東京国際大学のネットワークでの実験によるの結果，提案方式は誤報率を最小化し、高精度で異常検出が行え、提案手法の有効性が示された．

2 Introduction

Networks and distributed processing systems have become an important substrate of modern information technology. The rapid growth of these systems throughout the workplace has given rise to a discontinuity in expertise of human operators to manage them. There is a need for automating the management functions to reduce network operations and management cost.

Detection of network problems is a crucial step in automating network management. It has a direct impact on the accuracy of fault, performance and security management functions. From a control viewpoint, well designed fault and performance problems detection algorithms enhance the network control capability, by providing timely indication of network incipient problems. The possibility of early detection of performance degradation can alleviate the constant fire-fighting of network managers. Early warnings from the monitoring agent can trigger preventive actions, and serious and expensive outages can be avoided.

In this paper, we address the problem of performance problems detection in IP-Networks, where the knowledge about the problems to be detected is not required. The emphasis is on fast detection with minimal human supervision – an important requirement for reducing potential impact of problems on network services users. We propose a model of the network operations in terms of MIB objects dependencies, and we show that the parametric characterization of this dependency can be described as a finite mixture of simple regression models. Model parameters are identified from routine operation data, using the expectation maximization algorithm.

A new method for residual generation, based on successive parameter identification, is introduced. The residuals are shown to be approximately multivariate Normal, with mean zero under normal operations, and sudden jumps in this mean are characteristics of abnormal conditions. The detection problem is formulated as a *change point problem*. A real-time online change detection algorithm is designed to processes, sequentially, the residuals and raise an alarm as soon as the anomaly occurs. We motivate this formulation through a real problem scenario that occurred in Saitama university network. The proposed approach requires neither the set of faults and performance degradation nor the thresholds to be supplied by the user. Experimental results showed the effectiveness of the method on real data. A very low false alarm rate and a high detection capability has been demonstrated.

This paper is arranged as follows: Section 2 introduces our proposed model of the network normal behavior, and the learning algorithm for parameter identification from routine operation data. Section 3 introduces our proposed approach for residual generation, and the formulation of the network problem detection. In Section 4, we present results of our experiments in a real network. We conclude in section 5.

3 Normal Operations Baselining

MIB objects are designed to reflect the activity of the network at each protocol layer entity. The goal of this section is to characterize network normal behavior using these objects, and to identify network model parameters from routine operation data.

3.1 Network Model

To be able to identify the network operation parameters from operation data, we have to define a parametric model for network operations. Our approach to network model parameterization is to view each cluster (X, Y) as switching between different regimes, where each regime is a simple linear regression model. This is a form of what referred to in the literature as *switching regression* [10]. The observations (x_i, y_i) are generated by one of the K linear regression, as shown in the following equations:

$$y_i = b'_k x_i + \epsilon_{ki} \quad k = 1, \ldots, K \tag{1}$$

$$p(y_i|x_i) = \sum_{i=1}^{K} \frac{\pi_k}{\sqrt{2\pi}\sigma_k} \exp \frac{-(y_i - b'_k x_i)^2}{2\sigma_k^2} \tag{2}$$

For the case of single variables such as the number of broadcasts *IfInNUcastPkts*, the model reduces to the finite mixture model, given by:

$$y_i = m_k + \epsilon_{ki} \quad k = 1, \ldots, K \tag{3}$$

$$p(y_i) = \sum_{i=1}^{K} \frac{\pi_k}{\sqrt{2\pi}\sigma_k} \exp \frac{-(y_i - m_k)^2}{2\sigma_k^2} \tag{4}$$

The errors ϵ_k are assumed to be Gaussian, with mean 0 and variance σ_k. The column vector b_k is made of the slope and the intercept for the regime k. Abusing the notation slightly, the integer K denotes the number of regimes, for both the mixture of regression and Normal distributions. Each regime k has a mixing probability, denoted by π_k.

Finite Gaussian mixture models are general enough to approximate any continuous function with a finite number of discontinuities, under appropriate regularity conditions [12]. For any cluster of variables (X, Y) that are linearly related, or even locally linear with slowly time-varying parameters, an adaptive algorithm with suitably chosen *forgetting factor* can track the model parameters. In general, however, breaks in the linear relationship are normal, and the idea then is to explicitly accommodate these breaks in the network model. The resulting model is, then, a mixture of regimes, where each regime describe a given mode of network operations.

The network normal behavior is then characterized by the parameters of the finite mixture model. In the next subsection, we show how these parameters are identified.

3.2 Learning Model Parameters

Identifying the network normal operations from routine operation data amounts to estimate the parameter θ of the

switching regression. The vector θ consists of the vectors $b_1, \ldots, b_K$, the variances $\sigma_1, \ldots, \sigma_K$ and the mixing probabilities $\pi_1, \ldots, \pi_K$. Given a training set of N independent and identically distributed data points (x_i, y_i), the Maximum Likelihood (ML) estimator is the vector $\hat{\theta}$ that maximizes the likelihood function $L(\theta)$, given by:

$$\hat{\theta} = \arg\max_{\theta} L(\theta) \tag{5}$$

$$L(\theta) = \sum_{i=1}^{N} \sum_{k=1}^{K} \pi_k f_{ik} \tag{6}$$

$$f_{ik} = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left(\frac{-(y_i - b_k' x_i)^2}{2\sigma_k^2}\right) \tag{7}$$

For the case of mixture of Normal distributions, f_{ik} is given by:

$$f_{ik} = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left(\frac{-(y_i - m_k)^2}{2\sigma_k^2}\right) \tag{8}$$

Viewing the separation variable z, that assigns each observation to its corresponding regime, this estimation can be formally identified with the Expectation Maximization (EM) algorithm [3]. The EM algorithm [2, 11] estimate the unknown parameter θ by iteratively maximizing the expected log likelihood $Q(\theta|\theta')$, starting from an initial parameters θ^0.

4 Online Network Problems Detection

The previous section showed how the dynamics of the networks are identified. In this section we turn to discuss how the residuals are generated, and our formulation of the problem of anomaly detection.

4.1 Residual Generation

The residual generation method we propose is based on the algorithm for parameter estimation. After convergence of the learning algorithm (Sec. 2.2), we keep updating the slopes of the models, assuming that only one iteration of the EM algorithm is used for each new observation (x_n, y_n). It can be shown, that:

$$b_k^n = b_k^{n-1} + (X_n^T W_{kn} X_n)^{-1} w_{kn} x_n (y_n - b_k^{n-1} x_n) \tag{9}$$

$$b_k^0 = \hat{b}_k \tag{10}$$

For the case of mixture of Normal distributions, the mean m_k of each regime k is updated as follows:

$$m_k^n = m_k^n + (\sum_{i=1}^{n} w_{ki})^{-1} w_{kn} (y_n - m_k^{n-1}) \tag{11}$$

$$m_k^0 = \hat{m}_k \tag{12}$$

Where b_k^n and m_k^n are the estimate of the slope parameter b_k and mean m_k at iteration n, respectively. X_n is a $n \times 1$ vector made of the regressor x_i, and X_n' is its transpose vector. W_{kn} is the $n \times n$ diagonal matrix, made of the weights w_{ki}. The initial parameters b^0 and μ^0 is the estimates obtained using the batch EM (Sec. 2.2). It is worth nothing to note that Equation (9) is very similar to the recursive parameter estimation proposed in [?], where parameters are updated online for each new observation.

Under normal conditions, the difference between the successive values b^n and b^{n-1} is expected to fluctuate around zero. This difference should not drift constantly in a fixed direction. On the other hand, if this difference drifts systematically over long duration, then the new observations are generated by a different model, and the recursion in Equation (9) will alter the parameter b to its new value. The idea, then, is to generate the residuals based on the K-variate random variable $(b^n - b^{n-1})$. The mean value of this difference is a good indicator of the health of the network.

There is two major advantages of the residuals generated this way. First, successive identification of the parameters allows the model to adaptively track local changes in its parameters. It is unrealistic to assume that the model parameters will remain exactly the same over all the operating times of the network. Second, the difference $(b^n - b^{n-1})$ does not depend on the "true" value of the parameter b. This is very important since, in practice, we do not know this "true" value, and the only available information is the value $\hat{b}$, estimated from the data. Approximating b with $\hat{b}$, and studying the difference $(b^n - \hat{b})$ is possible, but our experiments showed that this approach is inefficient. Figure 1 compares both differences for a duration of one hour under the same network conditions. Results are shown only for one of the two regimes of the cluster, denoted by k. It can that the difference $(b_k^n - \hat{b}_k)$ is not symmetric around zero, while the difference $(b_k^n - b_k^{n-1})$ is both symmetric and very close to zero under normal conditions.

Under appropriate regularity conditions, we can show that the K-variate residuals e_n given by:

$$e_n = (b^n - b^{n-1})^T \Lambda^{-1} (b^n - b^{n-1}) \tag{13}$$

$$\Lambda = \text{diag}\left(\frac{\sqrt{w_{kn}} x_n \hat{\sigma}_k}{X_n^T W_{kn} X_n}\right) \tag{14}$$

For the case of mixture of Normal distributions, the residuals e_n are given by:

$$e_n = (m^n - m^{n-1})^T \Lambda^{-1} (m^n - m^{n-1}) \tag{15}$$

$$\Lambda = \text{diag}\left(\frac{\sqrt{w_{kn}} \hat{\sigma}_k}{\sum_{i=1}^{n} w_{ki}}\right) \tag{16}$$

are approximately Normal, with mean zero under network normal conditions. Approximating the variance matrix of e_n with Λ involves an informal argument about the asymptotic distribution of the univariate residuals e_{kn}. Note that e_n given in Equation (13) (respectively Equation (15)) is simply the difference $(b^n - b^{n-1})$ (respectively $(m^n - m^{n-1})$), scaled such that its variance-covariance matrix becomes Identity. Figure 2 shows the behavior of the residuals e_{kn}

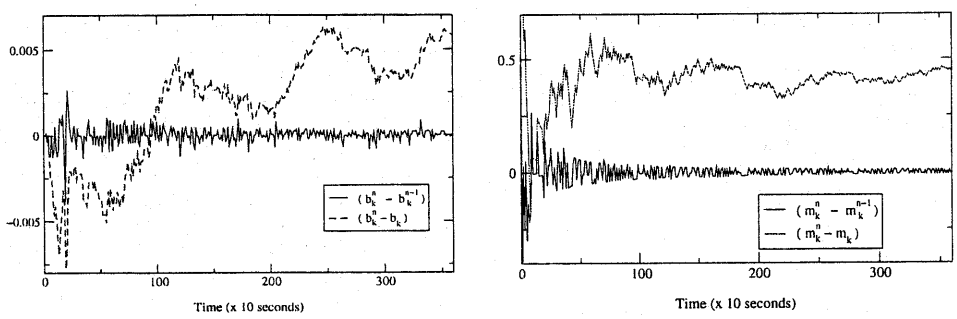

Figure 1: Behavior of the drifts of two typical clusters of objects (a) drifts $(b_k^n - b_k^{n-1})$ and $(b_k^n - \hat{b}_k)$ under the same network conditions, for the cluster *(IfInPkts, ipInReceives)* (b) drifts $(m_k^n - m_k^{n-1})$ and $(m_k^n - \hat{m}_k)$ under the same network conditions, for the variable *ifOutNUcastPkts*

under the same network conditions as in Figure 1. It can be seen that these residuals are stable, and their mean is very close to 0.

In summary, network operations are characterized by the distribution the residuals e_n. We showed that, under normal operations, the residuals e_n are Normal with mean zero and variance Identity matrix. The next section shows the behavior of the residuals under abnormal conditions, and how we formulate and solve the detection problem.

4.2 Anomaly Detection

Anomaly detection is determining the discrepancy between the normal behavior and the predicted behavior. Figure 3 shows the behavior of the residuals generated by the model under a real abnormal condition that affected Saitama university network, due to badly formatted packets. As shown in Figure 3-a, this abnormal condition causes a sudden jump in the mean of the residuals. Figure 3-b shows the behavior of the residuals just before the sudden jump in the mean. Interestingly, we notice that the sudden jump is preceded by a slight change in the mean of residuals. If the detection approach is designed to be sensitive to slight changes in the operating characteristics of the network, we could have predicted the problem of Figure 3 at least 19 minutes before it became serious. The problem could have been avoided, or at least addressed immediately after its occurrence. Obviously, not all problems presents signs to allow their prediction. In this case, we require our detection method to raise alarm as soon as change in the mean occurs.

Consider the residuals E_c^n obtained by observing sequentially the residuals e_i from time point c to n. Under the normal operations of the network, the sample of e_n follows a K-variate Normal distribution with mean 0 and Identity covariance matrix (Sec. 3.1). At some unknown time point c, a change happens in the model, and the new generated residuals shift to a new distribution. The goal is to find a *decision function* and a *stopping rule* that detects this change and raise an alarm as soon as possible, under a controlled false alarm rate. This formulation is known in sequential analysis literature as the *disruption problem.* The main difference with classical hypothesis testing is that the sample size is a function of the observations made so far (i.e. not fixed a priori), and the distribution of the residuals is known, when the process being monitored, is in control. The goal is to achieve fast detection of change, by using the minimum sample size possible to decide whether an alarm is to be raised or not.

It is well-known that for known probability distribution after change, Page-Lorden cumulative sum (CUSUM) [9, 6] test is optimal, in the sense that it minimizes the delay to detection, among all tests with a given false alarm rate. However, in the present case of network anomaly detection, we do not have a priori knowledge about the distribution probability after change P_{θ_1}, and the change point c. The common extension of Page-Lorden CUSUM test consists of estimating the post-change distribution mean, and the change point from the data. This approach is known as the Generalized Likelihood Ratio (GLR) test [1]. That is, for the unknown parameter θ_1 of the distribution of $P_{\theta_1}(e_i)$ after change, and the change point c are estimated from data, using the maximum likelihood estimator. The resulting decision function is given by:

$$R_n = \sup_{1 \le c \le n} \sup_{\theta_1} \ln \frac{P(E_c^n|\theta_1, c)}{P(E_c^n|\theta_0)} \quad (17)$$

$$T_n = \inf \{n : R_n > \lambda\} \quad (18)$$

In our case, where pre-change and post-change distributions are Normal, the maximization problem of Equation (17) can be worked out explicitly. It has a simple form,

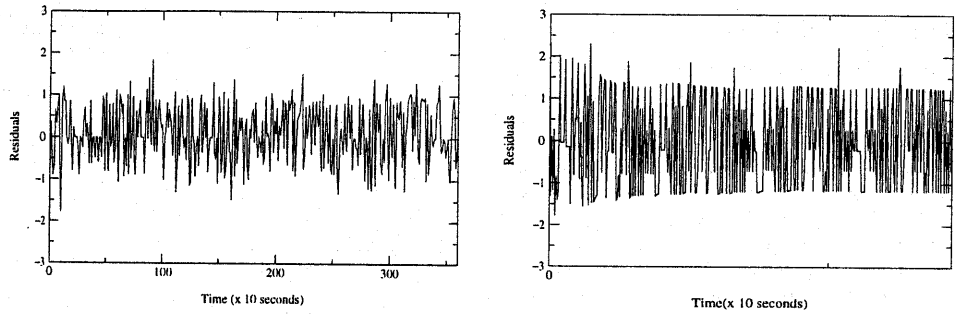

Figure 2: Residuals plots under the same network conditions as in Figure (1): (a) Univariate residuals e_{kn} for regime k of the switching regression, as given by Equation (13) (b) e_{kn} for finite Normal mixtures, as given by Equation (15)

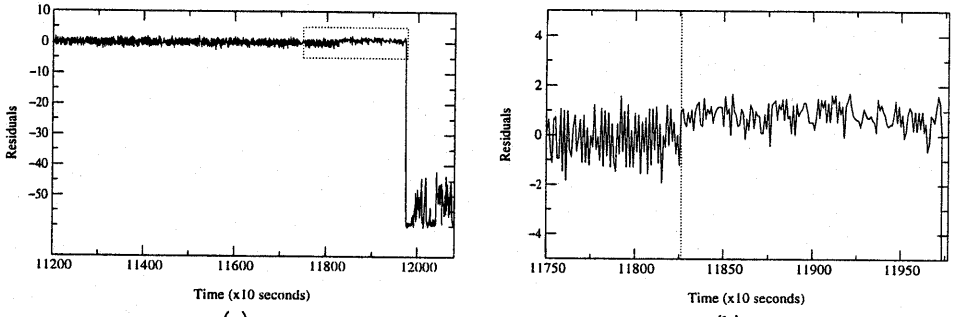

Figure 3: Residuals generated by one of the two regimes of *(ifInPkts,IpInReceives)*, under abnormal conditions: (a) structural break variables relationship (b) slight change in the mean of the residuals just before the break

given by:

$$S_0 = (0, \ldots, 0)^T \qquad S_n = \sum_{i=1}^{n} e_i \tag{19}$$

$$T_n = \inf\{n : \max_{0 \le c < n} \frac{\|S_n - S_c\|}{\sqrt{n-c}} > \lambda\} \tag{20}$$

The equation assumes that after change, the distribution of the residuals is still Normal, but with different mean. For the abnormal case, it is hard to obtain an unbiased fit of the post-change distribution $P_{\theta_1}(e_i)$. Fortunately such accurate estimation is not crucial. What is needed is that, when the an anomaly occurs, the closest Normal distribution, obtained by maximum likelihood estimation, has a mean significantly different from zero.

4.3 Tuning the Threshold λ

So far we have introduced the decision function and the stopping rule used for online detection of network faults and performance degradation. The remainder of our problem set-up concerns the choice of the design threshold λ.

It can be shown that the expectation of the stopping rule, under no change denoted by $E_\infty(T)$, is given by [13]:

$$E_\infty(T) \sim \frac{\Gamma(K/2)2^{K/2}\exp(\lambda^2/2)}{\lambda^K \int_0^\lambda x v^2(x)dx} \text{ as } \lambda \to \infty \tag{21}$$

$$v(x) = 2x^2 \exp\left(-2\sum_{1}^{\infty} n^{-1}\Phi(\frac{-xn^{1/2}}{2})\right), x > 0 \tag{22}$$

Where Φ denotes the Normal distribution function. For calculation, see [13] for an approximation of $v(x)$. Not surprisingly, Equation (21) turns out to be the mean time between false alarms. It follows that, given a desired false alarm rate, we can recover the design threshold λ, by solving Equation (21).

5 Evaluation and Results

The network monitoring algorithms described earlier has been implemented in a real networks. This section discusses how the data is collected, and the results that validate the agent capabilities.

5.1 Experimental Setting

The network traffic is monitored using the standard MIB-II information base. To ensure reliable data sets, the agent is implemented to fetch the network statistics directly from the OS kernel. The agent is written in C++, and currently runs on both Solaris 2.6 and AIX 4.1. To validate the agent results, we implemented a program to access the underlying data-link layer for fault injection. The program is written using the Data Link Provider Interface (DLPI), on Solaris 2.6 operating system. The goal is not to simulate all possible faults, but rather to prove that alarms generated by the agent are, effectively, due to abnormal network conditions.

5.2 Implementation Aspects

Unfortunately, Equation (20) can not be written recursively. Consequently, the number of residuals to be inspected can grow large. To circumvent this difficulty, we use a moving horizon of fixed length, where the starting point of the horizon moves one step forward as new observation are made available to inspection. For the mean false alarm rate (Sec. 3.3), it is fixed to 8640, which corresponds to 24-hours period, given that samples are collected every 10 seonds. Finally, for the number of components K, it was found empirically that at most 4 regimes are enough to describe the data satisfactorily. Work is underway to infer the number of components automatically from data.

5.3 Detection Capability

The agent detection capability is first illustrated using the network problem, introduced earlier in Section 3.2. The problem showed up as a streaming NIC card sending excessively badly formatted packets. Figure 4 shows the behavior of the residuals and test statistic as detected by the cluster (*ifInPkts, ipInReceives*). As shown in the figure, it takes approximately 30 samples (5 minutes) to detect the slight change in the residuals. In this particular case, the threshold is crossed 19 minutes before the sudden disruption. It could have been possible to address *proactively* the problem before it became serious, or at least draw the attention of network operators earlier, before the impact of the problem is felt by all network users.

Detection capability of the agent is tested with other problems. The obtained results are shown in Figure 5. For IP packet loss and excessive outbound broadcasts, appropriately assembled packets are sent every two seconds. For the remaining problems, assembled packets are injected every one second. It can be seen that the agent could detect all of these problems, with reasonably short delay.

We note that, apart from reasons given in Section 2.1 for defining network model as the parametric characterization of dependent objects, there is no MIB counter for ARP operations. Also, for the problem of packet loss induced by assembling Ethernet packets with non-existent protocol type, some kernels do not have any entry for this type of errors, even if this object is part of the standard MIB-II information base. For IP operations, one can easily check that *IpInDelivers* and *IpInReceives* can be sometimes very large, without noticing any change in IP related errors. This is true even when we take into consideration the fact that loopback packets are to be added to *IpInReceives*. With current under-instrumented networks, our approach to network modeling and performance problems detection can provide useful insights about incipient problems.

Clusters	Average alarm rate per hour
ifInNUcastPkts	0.33
ipInReceives, ifInPkts	0.50
ipInReceives, ipInDelivers	0.08
ipForwDatagrams, ipInReceive - ipInDelivers	0.04 [1]
ifOutPkts, ipForwDatagrams + ipOutRequests	0.16
ifOutNUcastPkts	0.00

Table 1: Average number of alarms per hour for each of the clusters of network model

5.4 Alarm rate

The final aspect we investigate in our proposed approach is the false alarms rate. Ideally, we would like to estimate this rate, given that the network is operating normally. Unfortunately, it is difficult to gain perfect knowledge about all the subtle changes in the network behavior. Instead, Table 1 shows the average alarm rate per hour, for data collected during a period of 24-hours.

Some comments are now in order. First, we note that ifOutNUcastPkts is very stable, recording a zero alarm rate per 24-hours. In general, alarms are raised only rarely. It would be more efficient, then, to restrict broadcast rate monitoring to only output packets only, and not both input and output broadcast packets. This way, not only detection accuracy and low false alarm rate are achieved, but more importantly, the diagnosis of broadcast problems is immediate. Second, the results in the Table 1 are not the minimum alarm rate that could be achieved. In times the network is almost idle, the alarm rate can be extremely low. In fact during experiments conducted in summer vacation (August 2000), only 8 alarms are raised for the 28956 samples collected by the cluster *ifInPkts-ipInReceives*. This makes an average of 0.09 false alarms per hour.

[1] Results are from an internal router of Saitama University network

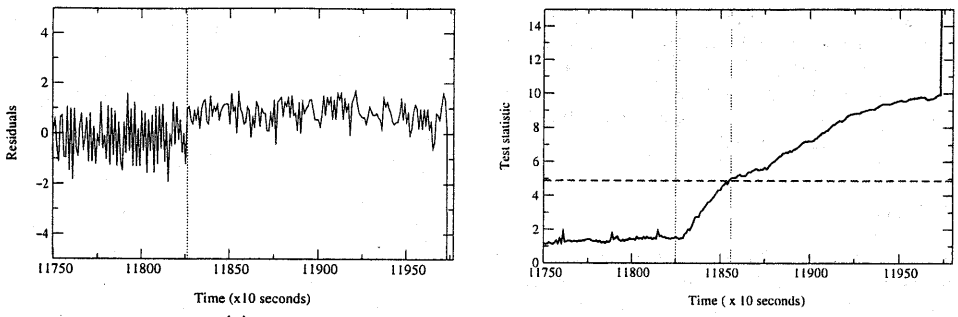

Figure 4: Behavior of the test static for the broadcast storm

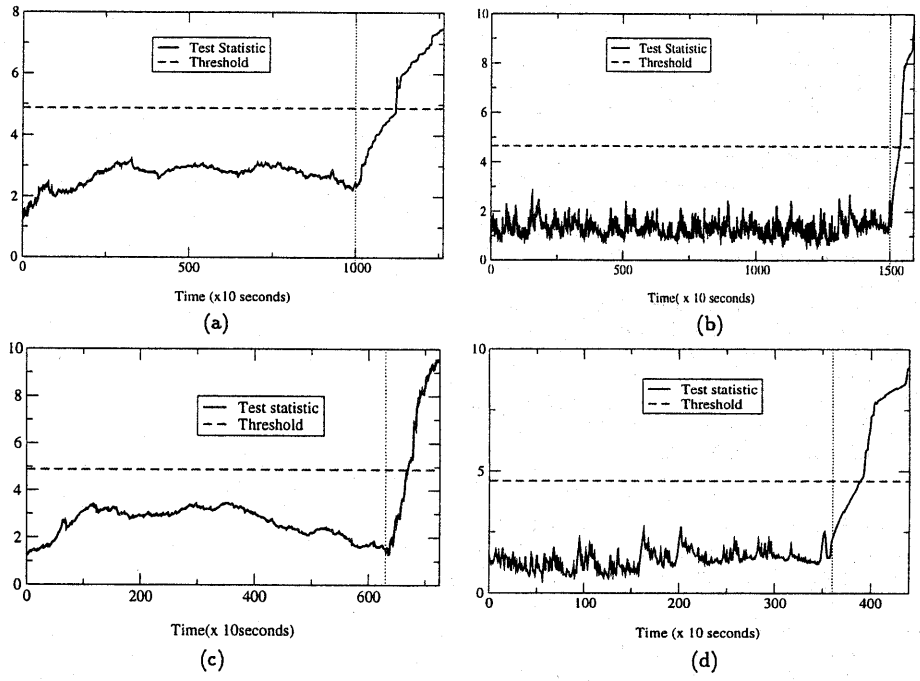

Figure 5: Behavior of the test statistic corresponding to excessive ARP packets, excessive inbound broadcasts, excessive outbound broadcasts, and IP packet loss problems: (a)IP packet discards (b) Outbound broadcast packets (c)Inbound broadcast packets (d) Inbound ARP packets

6 Conclusion

In this paper, we developed an online technique for fast detection of performance problems in IP-Networks. We proposed a model of the network operations in terms of MIB objects dependencies, and we showed that the parametric characterization of this dependency is amenable to a finite mixture model. Model parameters are identified from routine operation data, using the expectation maximization algorithm. A new method for residual generation, based on successive parameter identification, is introduced. The residuals are shown to be approximately Normal, with mean zero under normal operations, and sudden jumps in this mean are characteristics of abnormal conditions. A real-time online change detection algorithm is designed to processes, sequentially, the residuals and raise an alarm

as soon as the anomaly occurs. The proposed approach requires neither the set of faults and performance degradation nor the thresholds to be supplied by the user. Experimental results showed the effectiveness of the method on real data. A low false alarm rate and a high detection capability has been demonstrated.

References

[1] M. Basseville and I. V. Nikiforov. *Detection of Abrupt Changes : Theory and Application.* Prentice-Hall, 1993.

[2] A. Dempster, N. Laird and D.Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *J. R. Statist. Soc. B*, Vol. 39, pp:1-38, 1977.

[3] M. J. Hartley. Comment on "Estimating mixtures of Normal Distributions and Switching Regressions", *Journal of the American Statistical Association,* Vol. 73, pp:738-741, 1978

[4] S. C. Hood and C. Ji. Proactive Network Fault Detection. *Proceedings of the INFOCOM'97*, pp:1147-1155, 1997.

[5] A. Leinwand, K. Fang Conroy. Network management, a practical perspective. 2nd Edition. Addison-Wesley (1996).

[6] G. Lorden. Procedures for reacting to a change in distribution. *Annals of Mathematical Statistics*, Vol.42, pp:1897-1908, (1971).

[7] R. A. Maxion and F. E. Feather. A Case Study of Ethernet Anomalies in a Distributed Computing Environments. *IEEE Transactions on Reliability*, Vol. 39, No. 4, pp:433-443, 1990.

[8] K. McCloghrie, M. Rose. Management Information Base for Network Management of TCP/IP-based internets: MIB-II, RFC 1213, 1991.

[9] E. S. Page. Continuous Inspection Schemes. *Biometrika*, Vol. 41, pp:100-115, 1954.

[10] R. E. Quandt. A New Approach to Estimating Switching Regressions. *Journal of the American Statistical Association*, Vol. 67, No. 338, pp. 306-310, 1972.

[11] R. A. Redner and H. F. Walker. Mixture Densities, Maximum Likelihood and the EM Algorithm. *SIAM Review*, Vol. 26, No. 2, pp:195-239, 1984.

[12] B.D. Ripley. *Pattern Recognition and Neural Networks*, Cambridge University Press, 1996.

[13] D. Seigmund and E. S. Venkatraman. Using the Generalized Likelihood Ratio Statistic for Sequential Detection of a Change Points. *The Annals of Statistics*, Vol. 23, No.1, pp:255-271, 1995.

[14] M. Thottan and C. Ji. Proactive Anomaly Detection Using Distributed Intelligent Agents. *IEEE International Workshop on Systems Management*, 1998.