I.J. Information Technology and Computer Science, **2014, 01, 93-100**
Published Online December 2013 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2014.01.11

A New Fault Detection Method Using End-to-End Data and Sequential Testing for Computer Networks

Mohammad Sadeq Garshasbi
Department of Computer Engineering, Germi branch, Islamic Azad University, Germi, Iran
E-mail: ms.garshasbi@gmail.com

Shahram Jamali
Computer Engineering Department, University of Mohaghegh Ardabili, Iran
E-mail: jamali@iust.ac.ir

***Abstract*—** Fault localization, a central part of network fault management, is a process of deducing the exact source of a failure from a set of observed failure indications. in the network, end systems and hosts communicate through routers and links connecting them. When a link or a router faces with a fault, the information sent through these components will be damaged. Hence, faulty components in a network need to be detected and repaired to sustain the health of the network. In this paper we introduce an end to end method that detect and repair the faulty components in the network. The proposed method is a heuristic algorithm that uses the embedded information retrieved from disseminated data over the network to detect and repair faulty components. Simulation results show that our heuristic scheme only requires testing a very small set of network components to detect and repair all faults in the network.

***Index Terms*—** Network, Sequential Testing, Faulty Routers, Fault Detection

I. Introduction

Network components are prone to a variety of faults such as packet loss, link cut or node ruin. To prevent the faulty components from hindering network applications, it is important aspect to diagnose (i.e., detect and localize) the components that are the root cause of network faults. To diagnose (but not repair) network faults, recent approaches like [11, 12] use all network nodes to collaboratively achieve this. For instance, in hop-by-hop authentication [12], each hop inspects packets received from its previous hop and reports errors when packets are found to be corrupted.

For computer networks, fault diagnosis procedure includes two steps: fault detection and fault localization. Fault detection is to identify whether all and any fault has happened. In this step, detection tools are adopted to detect the presence of network faults. If there exist faults, then fault localization is triggered to identify the fault reason and location then repair. Hence, fault detection is the first key step to perform fault diagnosis for computer networks. To ensure networks normal operation, quick and accurate fault detection methods are needed[13].

Faults components affect the normal operation of the network, and hence should be detected and corrected/repaired. Existing methods to detect and repair fault components of computer networks are active and passive methods. Active method means that information is sent over the network only for detect faulty components. Active measurement incurs additional monitoring traffic (a node needs to monitor itself or its neighbors, and transmit the monitoring results locally or to a centralized server), which consumes precious resources of sensor nodes, and may reduce the lifetime of the network[3]. On the other hand, it has the advantage that it can exactly pinpoint the faults.

Passive method uses existing end-to-end data inside the network: if end-to-end data indicate faulty end-to-end behaviors, then some components in the network must be faulty [1, 2, 3]. In these methods, introduces no additional traffic into the network, and hence is an attractive approach for energy-stringent sensor networks. On the other hand, it poses the challenge of fault inference accurate inference from end-to-end data (i.e., locating all faults with low false positives) is not always possible because end-to-end measurements can have inherent ambiguity[3, 14].

Classical fault detection tools are usually passive methods. These methods use the alarms sent by the failure components to monitor the operational state of the network and detect the presence of faults[14]. After the faults are detected, then they will be further localized based on fault information [15]. Because of their non-intrusive characteristics, passive methods for faults detection have been widely used [15, 16]. The main limitation of passive methods lies in their requirement the support of network components [17]. Those passive methods require that each network component should send out an alarm when it fails or faults. In real applications, however, many network

devices have no such a function, and the alarms may also be interrupted or lost during transmission. Meanwhile, the complexity of modern networks (larg scal and etc) could make the relationships between fault alarms and real faults complicated, so that it is difficult to identify fault locations correctly [17, 18].

In recent years, active methods have received many attentions [19]. These methods detect network faults by sending data packets called probes into networks and inferring the health condition of networks according to the probing results [19, 20]. Thus, they have the advantages of little support from network components and capability to infer network performance quickly and accurately [21]. Active probing detection methods should consume network's bandwidth resources, so most researchers have focused on probe scheduling methods to minimize the bandwidth consuming [19-21]. Meanwhile, some other researchers also propose active probing techniques to solve network black holes problem and the all optical network fault diagnosis problem [19, 20].

Pervious studies on fault detection based on active probing focus on optimal algorithms that select a set of probes to test the whole network components [16, 17]. When the network is small or with simple topology, these methods can work efficiently; but as the network getting large and complex, these methods may be computationally infeasible [22], and will consume too much network bandwidth. To overcome this shortcoming of existing methods, Barford et al. [22] and Zhao et al. [23, 25] proposed some different strategies to monitor network operation. Barford et al. adopted a simple algorithm to select a set of probes to cover a part of the network each time, which could reduce the traffic overhead at each detection stage and cover all nodes after several rounds of detections. Zhao et al. scheduled path measurements in multiple rounds and monitored a part of the network links in each round. These two approaches provided a new way to fault detection in large scale networks[25].

Nevertheless, these two approaches require the complete information situation about networks, which means that the relationships between nodes and probes are supposed to be deterministic[25]. In real applications, however, the information about networks is not deterministic because networks can be interfered by many factors [24]. For example, the probe may be influenced by the environment noise. Thus, using complete information assumption to model networks is unsuitable for some cases[25].

We formulate a problem of optimal sequential testing guided by end-to-end data. This problem determines an optimal testing sequence of network components problem that minimizes the total testing cost[3]. it picks the first component to be tested based on the test result (i.e., it is faulty or not faulty) and the end-to-end data (passive measurements, which indicate potential faults), then it determines the next component to be tested. This sequential testing continues until the detected faulty components have explained all end-to-end faulty behaviors. Since these detected faults may not have included all faults in the network. So we identify all faults by solving the optimal sequential testing problem in iterations. In an iteration, based on end-to-end data in this iteration, we solve the optimal sequential testing problem to identify a set of faulty components. We then repair all the identified faulty components and start the next iteration. The iteration repeats until all end-to-end behaviors are normal. At this time, all faulty components have been detected and repaired.

In this paper, we proposed a novel passive method to detect faulty components so then repair. According to this method, bad paths detected based on end-to-end behavior and then theses information used to detect faulty routers and faulty links. It considers all the routers one by one and then tests them from point of view faulty behavior. If a router is marked as a faulty router, it will be repaired. The iteration repeats until all end-to-end behaviors are normal. At this time, all faulty components have been detected and repaired. The aim is to minimize the total testing cost over all the links and nodes of the network. We have compared the proposed method with the basic method proposed in reference [3].

Simulation results show that our heuristic scheme only requires testing a very small set of network components to detect and repair all faults in the network. These results demonstrate the benefits of our approach. Our approach also outperforms two approaches [3] that identify and repair all faults in a network.

This study is divided into the following sections. In section 2 an overview of the problem is given along with brief description of the solution methodology. Section 3 provides a detailed proposed method. The results are analyzed in section 4 and section 5 presents the concluding results.

II. Problem Definition

Consider a network where sent and receive data from clients to a server [3]. We assume the amount of end-to-end data can be used to detect faults in the network:

- Insufficient amount of data indicates faults.
- Sufficient amount of data indicates that the network is operating normally.

The status of a component (i.e., whether faulty or not) can be tested through active measurements, e.g., by monitoring the component locally, or looking into the internal states of relevant components. This test incurs a testing cost, which accounts for personnel wages when human is involved, or network bandwidths used to transfer the monitoring results to the server[4][6].

A fault in a network can be of various forms: fault links and fault routers (nodes).

In particular, our goal is to detect and repair fault routers that are used in routing. These faults can have several causes. (e.g., hardware faults, program bugs and etc.)[3]. We determine whether a router is fault or not based on its average loss rate or reception rate (defined as one minus the average loss rate). Router r is fault or bad if its average reception rate lies below a threshold, t_r. Otherwise, r is not fault or good. We assume the threshold, t_r, can clearly separate good and bad routers, i.e., good routers have reception rates much larger than t_r while bad routers have reception rates much lower than t_r. Furthermore, if a router (node) is good (or bad) on one path, then it is good (or bad) on all paths that use the router[3, 7].

Consider two settings of the problem:

1. We know complete path information, i.e., we know the path used by a client at any point of time.
2. We only know probabilistic path information, i.e., we only know the set of paths that are used by a client and the probability using each path. Here consider we only know probabilistic path information.

When only knowing probabilistic path information, we define client-server reception rate as the probability that a packet is sent/receive from a client to the server successfully. It can be estimated from end-to-end data: when n packets are sent/receive from a client to the server, and m packets arrive successfully, it is estimated as m/n[8]. We again assume that there exists a threshold so that at least one router used by a client-server pair is faulty if and only if the client-server reception rate is below this threshold. Again, we say a client-server pair (or simply a pair) is faulty or bad if its reception rate is below the threshold; otherwise, it is not faulty or good.

The above assumptions imply that all the routers on a good path/pair are good, and a bad path/pair contains at least one bad router[3, 9, 10]. Therefore, the potential bad routers are the ones that are used by bad paths/pairs, excluding those used by good paths/pairs.

For example, table 1 shows the routers that are used by clients for transmit information, there are client=3, server=1 and router=3.

Table 1 shown that the client 1 used from router 1,2 and 3 for transmit information to the server and client 2 used only from router 2 for transmit information to the server and client 3 used from router 1 and 2 for transmit information to server. For example, if router1 is faulty then faulty paths are *Path 1* and *Path 3* (shown table 2).

If router 2 is faulty then faulty paths are *Path 1, Path 2* and *Path 3* (shown table 3). If router 3 is faulty then faulty path is *Path 1* (shown table 4). If both router 1 and router 2 are faulty then faulty paths will *Path 1,Path 2* and *Path 3* (shown table 5).

For example, we assume that router 1 and router 3 are faulty. So faulty paths are *Path 1* and *Path 3*. First, we detect that router 1 is faulty using the proposed method (Described in section III) and we repair this router, so faulty path is only *Path 1* then will be test and detect router 3 is faulty using the proposed method and we repair router 1.

Aim is detect faulty routers with minimum test cost. So this problem is NP-Complete. In this paper, we introduce a method for minimizing the test cost. Our approach can run in iterations until all faulty routers are detected and repaired. our approach requires a similar or lower number of iterations and a much lower testing cost.

Table 1: routers that are used by clients for transmit information

Client 1	Router 1	Router3	Router2
Client 2	Router 2	-	-
Client 3	Router 1	Router2	-

Table 2: Faulty paths when router1 is faulty

Path 1	Router 1	Router 3	Router 2
Path 3	Router 1	Router 2	-

Table 3: Faulty paths when router2 is faulty

Path 1	Router 1	Router3	Router2
Path 2	Router 2	-	-
Path 3	Router 1	Router2	-

Table 4: Faulty path when router3 is faulty

Path 1	Router 1	Router3	Router2

Table 5: Faulty paths when router1 and router2 are faulty

Path 1	Router 1	Router3	Router2
Path 2	Router 2	-	-
Path 3	Router 1	Router2	-

III. The proposed Method

In this section, we develop a heuristic algorithm to solve the problem. The algorithm chooses a sequence of routers for test. In each step a router is selected for the test, if the router is good then next router selected for the test else if router is bad then repair router and the topology and bad paths re-configuration. We next describe the proposed method in detail. We aim is reduce test cost.

3.1 Difference between Good Paths and Bad Paths

First step our method is identify difference between good paths and bad paths, we find the difference

between good paths and bad paths, the difference is based on routers used at each path.

For example if we assume that router 1 and router 3 are faulty base on table 6. So bad paths are *Path 1* and *Path 3* (shown table 7). Good path is *Path 2* when router 1 and router 3 are faulty base on table 6 (shown table 8). Therefore, table 9 shown difference between good paths and bad paths. In fact, we are every one of the good paths minus with bad paths.

Table 6: Routers used at different paths

Path 1	Router1	Router 2	Router 3	Router 5
Path 2	Router 2	Router 4	-	-
Path 3	Router 1	Router 2	Router 3	-

Table 7: Faulty paths when router1 and router3 are faulty base on table 6

Path 1	Router1	Router 2	Router 3	Router 5
Path 3	Router 1	Router 2	Router 3	-

Table 8: Good path when router1 and router3 are faulty base on table 6

Path 2	Router 2	Router4

Table 9: difference between good paths and bad paths

Path1 - Path2	Router 1	Router 3	Router 5
Path3 - Path2	Router 1	Router 3	-

Table 10: An example For greedy algorithm desription paths

Path1	Router2	Router1	
Path2	Router3	Router1	
Path3	Router4	Router1	Rouer5

3.2 Select the router for the test

In this step (second step), algorithm pick the router for test in difference table (difference table between good paths and bad paths) based on the following equation:

$$H = \left(N_k \times P_k\right) / C_k \qquad (1)$$

Algorithm picks the router with the highest H, N_k is the number of paths that use Router R_k. Intuitively, it favors links with high values of $N_k \times P_k$. Furthermore, it favors links that are used by more paths. This is because, intuitively, knowing the status of such routers may provide more information. C_k is the cost for testing router R_k. P_k is the probability that a router R_k is faulty.

We use an example to illustrate this scheme. The paths are shown in table 6, where three client send/receive data to a server via the paths are shown in table 6. There is five routers, *R1*,..., *R5*. We assume that routers *R1* and *R3* are bad. The cost for testing a router is 1 unit. The probability that a router is faulty is p = 0.1.

We assume that we only know probabilistic path information. In this case, end-to-end data indicate that *Path 1* and *Path 3* are faulty. We consider two client-server pairs, all identified as bad from end-to-end data. Therefore difference between good paths and bad paths shown in table 9. Since *R1* and *R3* are more used in difference table between good paths and bad paths, we will test link *R1* or *R3* first and find it is Faulty. Next it is repaired[3, 4, 7]. We assume *R1* detected and repaired.

In the next iteration, compute difference table between good paths and bad paths again and detect *Path 3* is faulty. In this iteration detect the *R3* is faulty and repaired. To summarize, for this example, the proposed algorithm uses two tests to identify all faulty links.

Steps of our method are:

a) compute difference table between good paths and bad paths.

b) pick the router for test in difference table based on the (1) equation.

c) Test

d) Repair

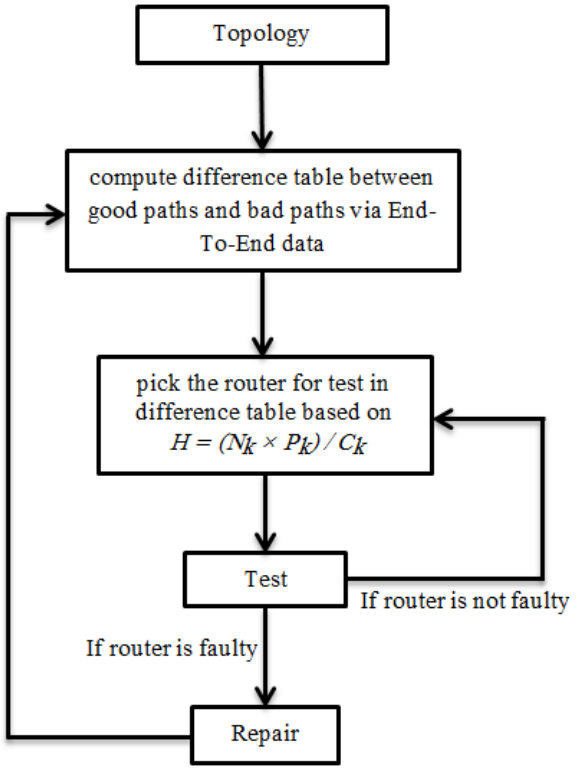

Fig. 1: our method chart

Steps run until difference table between good paths and bad paths is none (difference table = 0).

IV. Performance Evaluation

This section, We evaluate the performance of our heuristic algorithm whit greedy algorithm that the propose in [3] in a network. We used MATLAB for simulations. In the following, we describe the greedy algorithm.

4.1 Greedy Algorithm

In each step, this algorithm picks the router that provides the highest gain. The gain from knowing the status of a router is defined as the cost savings (from the routers that do not need to be tested due to this knowledge) subtracted by the testing cost of this router.

More specifically, for router R_k, let θ_k denote the expected gain from knowing the status of router R_k, let θ_{kb} denote the cost savings when knowing R_k is bad, and let θ_{kg} denote the cost savings when knowing R_k is good. Then $\theta_k = P_k\theta_{kb} + (1\text{-}P_k)\theta_{kg} - C_k$, where P_k is the probability that R_k is bad, and C_k is the cost of testing R_k[3].

For greedy algorithm desription we assume there are paths as well table 10. The topology of table 10 is as well figure 2.

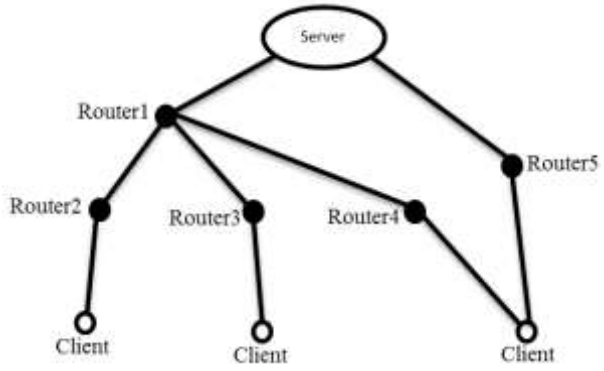

Fig. 2: An example to illustrate the greedy algorithm testing schemes

When we assume know probabilistic path information, the three lossy paths shown in table10. We calculate the gains from testing the various routers as:

$$\theta 1 = 4p + 2(1 - p) - 1 = 2p + 1 = 1.4$$

$$\theta 2 = \theta 3 = 3(1 - p) - 1 = 2 - 3p = 1.4$$

$$\theta 4 = \theta 5 = -1$$

The expected gain θ_1 is calculated as above because if Router 1 is bad, then *Path 1, Path 2* and *Path 3* are explained and hence we do not need to test links Router 2,..., Router 5, leading to a saving of 4; if Router 1 is good, then Router 2 and Router 3 must be bad, leading to a saving of 2. expected gain $\theta 2$ is $3(1-p)-1$ since if Router 2 is bad, then it leads to no savings; if Router 2 is good, then Router 1 must be bad, which makes testing Route r3, Router 4 and Router 5 unnecessary, leading to a saving of 3. The gain $\theta 3$ is obtained in a similar manner. Knowing the status of Router 4 or Router 5 does not lead to savings, and hence the gain is -1. Since testing links Router 1, Router 2, and Router 3 provide the same highest gain, we break the tie arbitrarily. Suppose we choose to test Router 2. We find that Router 2 is good, and hence Router 1 must be bad, which explains all paths. After Router 1 is repaired, the second iteration is similar to that under the ordering algorithm. To summarize, for this example, the greedy algorithm uses two iterations and two tests to identify all lossy links[3].

4.2 Tradeoffs between the Our and Greedy Schemes

The greedy scheme has a higher complexity than our scheme [3]. We compare the performance our schemes in more general settings where we, however, do not know the prior probability that a router is faulty, and assume that the prior probabilities of all the routers are the same. Simulation results shows that our heuristic scheme is outperform compare to greedy scheme.

So far, for ease of exposition, we only consider faulty routers. The heuristic schemes is also applicable to the more general scenarios where both nodes (routers) and links can be faulty.

4.3 Simulation Results

In this section, we evaluate the performance of our heuristic algorithm with greedy algorithm In the network.

There are two types of routing: static and dynamic routings. Under static routing, the paths from the clients to the server are fixed, and we know the complete path information. Under dynamic routing, there are several paths from client to server and we assume a path report service that reports path information periodically to the server. We assumed dynamic routing in simulations.

For testing costs, there are two models of cost: homogeneous and heterogeneous cost models, respectively. In the homogeneous cost model, all the routers have the same testing costs of 1 unit. In the heterogeneous cost model, the testing cost of a router is proportional to its distance to the server. More specifically, a router that is k hops away from the server has a testing cost of k units; a link that is adjacent to the root has a testing cost of 1 unit, and k hops away from the server has a testing cost of k units. We assumed homogeneous model in simulations.

The prior probability that a router is faulty can be estimated from historical data, if a network has been operating for a long time. It can also be estimated online from end-to-end data. This technique is, however, computational intensive, and we were not able to obtain results in a reasonable amount of time for our simulation scenarios. In the following, for simplicity, we assume that the prior probabilities are unknown, and all routers have the same probability of being faulty. Under this assumption, the proposed testing scheme does not depend on the exact value of p. For our scheme sequential testing scheme, we set $p = 0.1$.

We compare our sequential testing schemes with existing studie [3] that also localize and repair all faults in a network.

First, we consider a network that has one server and 50 client, this network uses of 100 routers to Transmission information. Figure 3 shown plots the results when clients = 50 and routers = 100, The results of greedy and our method testing schemes are plotted in the figure 3. Number of router use via clients is randomly assigned to clients.

Figure 3 shown that our propose method is better than greedy algorithm for find faulty routers. It shown that the testing cost in our propose method is less than greedy algorithm when number of the bad routers are 10. Number of routers used via clients is between 1 and 20 and randomly assigned to clients.

In other status, we consider a network that has one server and 200 client, this network uses of 500 routers to transmission information. Figure 4 shown plots the results when clients = 200 and routers = 500. The results of greedy and our method testing schemes are plotted in the figure 4. Number of routers used via clients is between 1 and 50 and randomly assigned to clients.

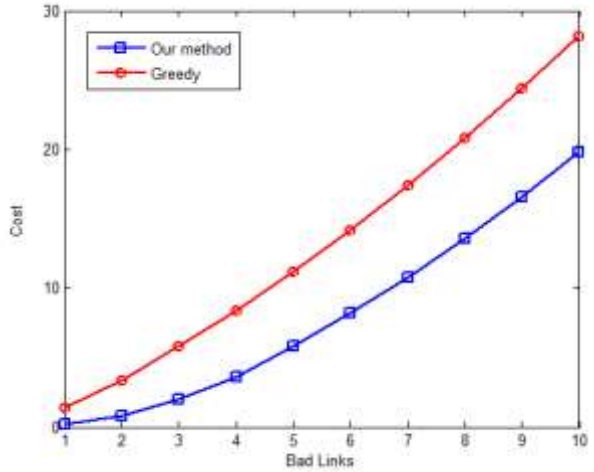

Fig. 3: testing cost plot. faulty routers scenario, simulation results when client = 50 and router = 100

Figure 4 shows the proposed model is better than greedy algorithm and reduce total testing cost too. It also indicates that the proposed method for small and large scale networks have similar results. Also, the complexity of out method is less compare to greedy algorithm and other methods. This results when routers are faulty, results under the links are faulty have similar trends as those under the routers are faulty.

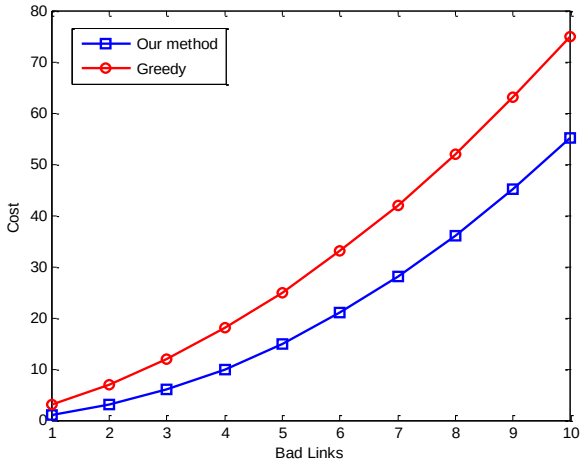

Fig. 4: testing cost plot. faulty routers scenario, simulation results when client = 200 and router = 500

V. Conclusions

In this paper we proposed a novel method to detect and repair all faulty links and nodes while minimizing the total test cost. The proposed method first detects bad paths based on the end-to-end path behavior and functionality and then the routers that are distinguished as fault node are repaired based on the passive method proposed in this paper. Simulation results show that our heuristic algorithms has only a few testing cost, since a small subset of components needs to be scanned to identify all faulty components in the network. These results demonstrate the benefits of our approach.

References

[1] A. Adams, T. Bu, R. Caceres, N. Duffield, T. Friedman, J. Horowitz F. L. Presti, S. Moon, V. Paxson, and D. Towsley. The use of end-to-end multicast measurements for characterizing internal network behavior. IEEE Communications Magazine, May 2000.

[2] Xin Zhang, Zongwei Zhou, Geoff Hasker, Adrian Perrig and Virgil Gligor, "Network Fault Localization with Small TCB" Network Protocols (ICNP), 2011 19th IEEE International Conference on.

[3] Bing Wang, Wei Wei, Hieu Dinh, Wei Zeng, "Fault Localization Using Passive End-to-End Measurements and Sequential Testing for Wireless Sensor Networks", IEEE TRANSACTIONS ON MOBILE COMPUTING, March 2012, 439 – 452.

[4] Ma łgorzata Steinder, Adarshpal S. Sethi, " A survey of fault localization techniques in computer networks ", Elsevier Science of Computer Programming Volume 53, Issue 2, November 2004, Pages 165–194.

[5] Dipt. di Ing. Elettron. e dell'Inf., Univ. of Perugia, Perugia, " Fault localization in data networks", Communications Letters, IEEE, March 2009.

[6] A.T. Bouloutas, S. Calo, A. Finkel, "Alarm correlation and fault identification in communication networks", IEEE Transactions on Communications, 42 (2–4) (1994), pp. 523–533.

[7] N. Duffield. Network tomography of binary network performancecharacteristics. IEEE Transactions on Information Theory, 52(12),December 2006.

[8] H. X. Nguyen and P. Thiran. Using end-to-end data to infer lossy links in sensor networks. In Proc. of IEEE INFOCOM, April 2006.

[9] K. R. Pattipati and M. G. Alexandridis. Application of heuristic search and information theory to sequential fault diagnosis. IEEE Transactions on Systems, Man and Cybernetics, 20(4):872–887, 1990.

[10] P. P. Lee, V. Misra, and D. Rubenstein. Toward optimal network fault correction in externally managed overlay networks. IEEE Transactions on Parallel and Distributed Systems, 21(3):354–366, March 2010.

[11] A. T. Mizrak, Y.-C. Cheng, K. Marzullo, and S. Savage. Fatih: Detecting and Isolating Malicious Routers. In Proc. of the IEEE Conference on Dependable Systems and Networks (DSN), June 2005.

[12] S. Q. Zhuang, D. Geels, I. Stoica, and R. H. Katz. On Failure Detection Algorithms in Overlay Networks. In Proc. of IEEE INFOCOM, March 2005.

[13] Natu Maitreya, Sethi Adarshpal S. Probabilistic fault diagnosis using adaptive probing. Lecture Notes in Computer Science 2007;4875:38–49.

[14] Jakobson G, Weissman M. Alarm correlation. IEEE Network 1993;7(6):52–9.

[15] Steinder M, Sethi Adarshpal S. A survey of fault localization techniques in computer networks. Science of Computer Programming 2004;52(2):165–94.

[16] Mohamed Abduljalil, Basir Otman. Fusion based approach for distributed alarm correlation in computer networks. In: Second international conference on communication software and networks. Singapore; February 2010. p. 318–24.

[17] Tang Yongning, Al-Shaer Ehab. Overlay fault diagnosis based on evidential reasoning. In: Proceedings of the IEEE INFOCOM. Rio de Janeiro, Brazil; April 2009. p. 2596–600.

[18] Cheng Lu, Qiu Xuesong, Meng Luoming, Qiao Yan, Boutaba Raouf. Efficient active probing for fault diagnosis in large scale and noisy networks. In: Proceedings of the IEEE INFOCOM. San Diego, CA; March 2010. p. 1–9.

[19] Chu LW, Zou SH, Chen SD. Wang WD, Tian CQ. Internet service fault management using active probing in uncertain and noisy environment. In: 4th international conference on communications and networks in China. Xian, China; August 2009. p. 1–5.

[20] Nguyen Hung X, Teixeira Renata, Thiran Patrick, Diot Christophe. Minimizing probing cost for detecting interface failures: algorithms and scalability analysis. In: Proceedings of the IEEE INFOCOM. Rio de Janeiro, Brazil; April 2009. p. 1386–94.

[21] Harvey Nicholas JA, Patrascu Miai, Wen Yonggang, Yekhanin Sergey, Chan Vincent WS. Non-adaptive fault diagnosis for all-optical networks via combi- natorial group testing on graphs. In: Proceedings of the IEEE INFOCOM. Anchorage, AK; May 2007. p. 697–705.

[22] Barfod Paul, Duffield Nick, Ron Amos, Sommers Joel. Network performance anomaly detection and localization. In: Proceedings of the IEEE INFOCOM. Rio de Janeiro, Brazil; April 2009. p. 1377–85.

[23] Zhao Yao, Zhu Zhaosheng, Chen Yan, Pei Dan, Wang Jia. Towards efficient large-scale VPN monitoring and diagnosis under operational constraints. In: Proceedings of the IEEE INFOCOM. Rio de Janeiro, Brazil; April 2009. p. 531–9.

[24] Li Cheng, Zou Shihong, Chu Lingwei. Online learning based internet service fault diagnosis using active probing. In: International conference on network- ing, sensing and control. Okayama, Japan; March 2009. p. 773–8.

[25] Lu Lu,ZhengguoXu, WenhaiWang,YouxianSun, A new fault detection method for computer networks, Reliability Engineering and System Safety 114 (2013) 45–51.

Authors' Profiles

Mohammad Sadeq Garshasbi, born in 1989, He is MSc student in Islamic Azad University of Germi, Iran. He received his B.S. degree in computer software engineering from Sabalan College, Ardabil, Iran, in 2011. He is teacher in Sama technical and

professionals college of Khalkhal. He teaches on computer networks and operating systems. His research interests include computer networks, operating systems, and evolutionary algorithms.

Shahram jamali is an assistant professor leading the Autonomic Networking Group at the Department of Engineering, University of Mohaghegh Ardabili. He teaches on computer networks, network security, computer architecture and computer syetems performance evaluation. Dr. Jamali received his M.Sc. and Ph.D. degree from the Dept. of Computer Engineering, Iran University of Science and Technology in 2001 and 2007, respectively. Since 2008, he is with Department of Computer Engineering, University of Mohaghegh Ardabil and has published more than 100 conference and journal papers.

How to cite this paper: Mohammad Sadeq Garshasbi, Shahram Jamali,"A New Fault Detection Method Using End-to-End Data and Sequential Testing for Computer Networks", International Journal of Information Technology and Computer Science(IJITCS), vol.6, no.1, pp.93-100, 2014. DOI: 10.5815/ijitcs.2014.01.11

