Preprint

Towards Distributed and Adaptive Detection and Localisation of Network Faults

Rebecca Steinert and Daniel Gillblad
Industrial Applications and Methods Lab (IAM)
Swedish Institute of Computer Science (SICS)
SE-164 29 Kista, Sweden
Email: {rebste, dgi}@sics.se

Abstract—We present a statistical probing-approach to distributed fault-detection in networked systems, based on autonomous configuration of algorithm parameters. Statistical modelling is used for detection and localisation of network faults. A detected fault is isolated to a node or link by collaborative fault-localisation. From local measurements obtained through probing between nodes, probe response delay and packet drop are modelled via parameter estimation for each link. Estimated model parameters are used for autonomous configuration of algorithm parameters, related to probe intervals and detection mechanisms. Expected fault-detection performance is formulated as a cost instead of specific parameter values, significantly reducing configuration efforts in a distributed system. The benefit offered by using our algorithm is fault-detection with increased certainty based on local measurements, compared to other methods not taking observed network conditions into account. We investigate the algorithm performance for varying user parameters and failure conditions. The simulation results indicate that more than 95% of the generated faults can be detected with few false alarms. At least 80% of the link faults and 65% of the node faults are correctly localised. The performance can be improved by parameter adjustments and by using alternative paths for communication of algorithm control messages.

Index Terms—Adaptive probing; distributed fault-detection; fault-localisation.

I. INTRODUCTION

Approaches to fault-detection and localisation in networked systems can be sorted into two categories; centralised and distributed methods [1]. Centralised methods are based on collecting network data for analysis in one or several dedicated network modules. Such methods are typically useful for analytical applications, in which it is of interest to identify network behaviour patterns, or deviating events in various parts of the network. Distributed methods locally process data collected in the immediate neighbourhood, which allow for e.g., fast detection of network faults and flexibility to varying topologies. In this paper, we present a statistical, distributed approach to adaptive fault-detection and localisation. Specifically, we attempt to detect network behaviour that deviates from normal observations, i.e., symptoms of physical or logical network faults, rather than finding a particular type of fault.

The approach is based on *probing* for two purposes. First, probes are sent between nodes in order to measure response delay and drop rate on each connection. Second, adaptive *probe tests* are performed in each node for detection of

abnormal network behaviour. Note that here the term *response delay* refers to the probe reply time or round-trip time.

For each connection parameter estimation is performed to model the probability distribution of observed delays, such that the expected response delay can be computed. Based on the expected response delay and drop rate, probe tests and probing intervals are autonomously adapted to the observed behaviour of individual connections. Probe tests are performed in each node to test the availability of adjacent nodes and links. If a probe test on a connection fails, a *symptom* of a network fault has been detected and a fault-localisation process based on node collaboration is initiated (see Figure 1).

Fig. 1: Example of algorithm functionality.

The fault-detection approach is designed to meet the following three requirements. First, the autonomous adaptation of algorithm parameters should significantly reduce the efforts on manual configuration. This is achieved by having probing mechanisms be specified as the cost of sending probes, rather than in terms of e.g., time intervals. Second, the autonomous configuration of the probing mechanisms should allow for improved efficiency of bandwidth usage, compared to conventional monitoring based on fixed interval probing. Third, the algorithm should, without rigorous modification, run on different types of networks and network layers.

This is the first step towards a fully adaptive method for fault-detection in distributed systems. Here, we focus on autonomous probing and detection mechanisms. Further development includes adaptation to long-term changes and detection of probe response delays deviating from normal observations.

A. Related work

In a paper by Yu et al., one of the main risks of centralised approaches identified is increased link load, close to the

central collection point [2]. Further, the authors point out that neighbour-coordination for fault-detection reduces communication overhead. In addition, network faults can be detected with increased certainty using neighbour-collaboration. They also mention the importance of simple management and flexibility, specifically in sensor-networks where focus is on energy-preservation. These arguments summarise some of the principles, under which our algorithm is designed.

Different strategies for fault-detection have been investigated by Zhuang et al. [3]. The authors divide fault-detection into passive and active methods. Passive approaches eavesdrop on packets for monitoring the status of other nodes, whereas active monitoring methods are based on end-to-end transactions between nodes. Further, two categories of fault-localisation approaches are identified; nodes can either independently or collaboratively decide the status of a neighbouring node. The approach that we apply relates to the second category. The results presented in [3] show that algorithms based on neighbour-collaboration and information-sharing can reduce the detection time, but increases the control overhead.

There are a number of active methods based on probing. In general the goal is to determine the best probing action given certain conditions, for the purpose of reducing communication overhead while achieving reliable fault-handling. One such method is based on logical trees for determining a candidate node for probing [4]; in another approach, the authors solve the NP-hard problem of computing a minimal set of probe messages to be transmitted by the stations for fault-isolation and latency measurements, applying a polynomial-time greedy approximation algorithm [5]. Other techniques for active probing are based on statistics and information-theoretic approaches; for example probabilistic reasoning is performed to select the most informative probe test [6]. As a final example, Tang et al. propose a combination of passive and active methods for isolating faults via probing, involving heuristic fault-reasoning and fidelity measures for decision making [7]. Compared to these approaches, the design of our method is aimed at reducing communication overhead by using two types of individually set probe intervals on each link, based on local end-to-end transactions between adjacent nodes.

The probing method that we use relates to that described by Andersen et al. [8], in which two different probing frequencies are applied and used for outage detection. Their probing approach is based on fixed time intervals applied to all connections in the network. In contrast, probing intervals are here set by taking into account variations in probe response delays and packet loss, individually measured for each connection. Moreover, the detection mechanism that they use is based on the number of lost probe responses, whereas we apply a probabilistic detection threshold in order to achieve reliable fault-detection with few false alarms.

B. Contribution

Our contribution is a statistical and relatively simple method that reliably and with high certainty can detect and localise faults, based on locally observed measurements. We see that

distributed probing is the simplest and most flexible approach for networks e.g., under churn, compared to centralised methods. In addition, protocols needed to run the monitoring algorithm are already implemented in most network equipment of today. By taking into account the drop rate and the variance in probe response delays, increased reliability and robustness can be achieved compared to other probing methods, in which these factors are not considered. Further, probing parameters are here set autonomously based on parameter estimation for each link, which facilitates and reduces manual configuration efforts while only a small amount of link load is produced.

Section II and III describe our approach to fault localisation. Section IV contains algorithm descriptions, followed by an overview of the simulation environment, experimental results, and concluding remarks in sections V, VI, VII and VIII.

II. FAULT-DETECTION USING ADAPTIVE PROBING

The mechanisms for detecting and isolating faults are based on the adaptation of algorithm parameters related to probing. Specifically, the intervals between probes and the number of probes needed to detect a fault are adjusted to the locally observed probe response delay and packet drop rate.

In run-time, observations of probe response delays are continuously collected via probing, forming a two-parameter Gamma probability density function (PDF), from which the parameters α and β are estimated [9]:

$$P(t) = t^{(\beta-1)} \frac{e^{-t/\alpha}}{\alpha^\beta \Gamma(\beta)}. \quad (1)$$

The choice of model is motivated by the assumption that the response delay is a sum of independent exponential transmission delays caused by e.g., queueing times in processing nodes. Empirical tests indicate that the Gamma PDF matches real-world probe response delays quite well [10]. Similar conclusions about network traffic delays (on different network levels) matching Gamma, or other exponential distributions, have been made in several papers, such as [11], [12].

The fault-detection approach also involves the probability of packet drops $P(D)$. In order to increase the certainty of a suspected fault without specifying fixed detection conditions, we observe the joint probability of $P(t)$ and $P(D)$ (see section II-C). Further, we assume independence between the response delay and the drop rate, since packet loss is mainly related to malfunctioning equipment or link quality, rather than to transmission delays in processing nodes.

A. Parameter estimation

The probability of packet drops $P(D)$ is computed as the rate between dropped probes and the number of sent probes. Further, a method of moments approach is applied for estimation of the Gamma distribution parameters α and β from the first and second sample moments $s_1 = \frac{1}{n} \sum_i t_i$ and $s_2 = \frac{1}{n} \sum_i t_i^2$. Given that $\alpha\beta = s_1$ and $\alpha^2\beta(\beta+1) = s_2$, the estimates α^* and β^* are [9], [13]:

$$\alpha^* = \frac{s_2 - s_1^2}{s_1}, \quad \beta^* = \frac{s_1^2}{s_2 - s_1^2}. \quad (2)$$

This approach produces parameter estimates with less precision than maximum likelihood estimations, but requires less computational resources. Since these estimations are frequently performed in each node (which in practice may have limited computational resources) we accept less precision for the benefit of computational efficiency.

B. Adjustments of probing intervals

From the inverted cumulative distribution of $P(t)$ the time interval between probes is computed. Two types of intervals are used and controlled by parameters τ and θ , both adjusted to the observed response delays and the costs c_τ and c_θ of sending a probe:

$$\tau = c_\tau f_{cdf}^{-1}(l), \quad \theta = c_\theta f_{cdf}^{-1}(l). \quad (3)$$

The function $f_{cdf}^{-1}(l)$ is the inverted cumulative density function of $\int_0^{\Delta t} P(t)$, and l is a fraction that is used to determine the corresponding probe interval. The fraction l and the costs c_τ and c_θ are set manually by the user as a trade-off between the detection performance and the amount of probing traffic.

The parameter τ controls the interval between probe tests, whereas parameter θ determines the time interval between individual probes (see section IV). These time intervals are adjusted for each update of the Gamma parameter estimates; in performed simulations the parameters and the intervals are updated for each new response delay observation, but if needed this can be done at a sparser level.

The interval parameter θ is significantly smaller than τ . This way the link load caused by probing traffic is reduced during normal network behaviour, while being somewhat increased when a network fault is about to be detected. Compared to ordinary monitoring with fixed intervals, the use of two probing intervals that are autonomously set based on measurements in the network can reduce the total link load caused by probing.

C. Decision model

We assume that the probe response delay and the drop rate are mutually independent [14]. The probability of receiving a probe response $R_{\Delta t}$ within delay Δt is then computed as:

$$P(R_{\Delta t}) = (1 - P(D)) \int_0^{\Delta t} P(t; \alpha^*, \beta^*) dt. \quad (4)$$

The fault-detection mechanism relies on *probe tests*, i.e., series of probes sent with autonomously set time intervals. The purpose of sending several probes is to increase the certainty about a detected fault and to reduce the amount of false alarms.

To decide if a fault truly has been encountered, we assume that the joint probability of *not* receiving *any* response R given a set of statistically independent [14] probes in a probe test is

$$P(\neg R | \Delta t^{(1)}, \Delta t^{(2)}, \dots, \Delta t^{(n)}) = \prod_{i=1}^n (1 - P(R_{\Delta t}^{(i)})) \quad (5)$$

The probe test is stopped either when a probe response is obtained or when $P(\neg R | \Delta t^{(1)}, \Delta t^{(2)}, \dots, \Delta t^{(n)})$ reaches below the detection threshold ψ , subsequently triggering the fault-localisation process. The number of probes needed to reach

below the detection threshold ψ is thus adapted to $P(R_{\Delta t})$ in eq. 4. Smaller values on ψ increase the certainty about a fault but at the cost of increased probe traffic. Hence, the detection performance is a trade-off between communication overhead and the amount of false alarms.

III. COLLABORATIVE FAULT-LOCALISATION

The fault-localisation process involves collaboration between nodes in order to localise the origin of the abnormal network behaviour. The algorithm is designed to distinguish between symptoms of node faults and link faults.

A. Collaboration scheme

Each node n has a list of all adjacent nodes and their neighbours (protocols to obtain such information are easily implemented). The rate, at which each node n will probe a neighbour $\hat{n}$, is determined locally as described. When a probe test from node n to a node $\hat{n}$ fails, node n initiates the fault-localisation process. This involves collaboration with the neighbouring nodes of $\hat{n}$, i.e., $\tilde{n} = \{\tilde{n}_1, \tilde{n}_2, \dots, \tilde{n}_i\}$, in order to test the connection to $\hat{n}$ and report back to n . If at least one node $\tilde{n}_i$ reports a successful probe response, a link fault is indicated (Figure 2a). If none of the nodes in $\tilde{n}$ receive a probe response, a node fault in $\hat{n}$ is indicated. The outcome of the fault-localisation is reported to the operator by n .

In the case a link fault is concluded, information about the detected fault is conveyed from n to $\hat{n}$ via $\tilde{n}$ to prevent $\hat{n}$ triggering a second fault-localisation process. Similarly, neighbouring nodes affected by a node fault in $\hat{n}$ are informed by n to avoid triggering several fault-localisation processes.

The exchange of information between the detecting node and the collaborative nodes depend on timers, which control the duration that the nodes wait before returning to normal operation. In each node the timers are based on the expected probe response delay, the number of neighbours N to $\hat{n}$ and a cost c , such that $T = cN f_{cdf}^{-1}(l)$ (section II-B). If the timer of a detecting node expires before receiving all probe test results (possibly due to communication faults or packet loss), the fault is reported as undecidable. Similarly, the collaborating node returns to normal operation if the timer expires while waiting for the final result from the detecting node.

Fig. 2: Figure a) node n sends requests to $\tilde{n}_i$, $\tilde{n}_j$ to test the connection of $\hat{n}$. The outcome of the probe test between $\tilde{n}_i$ and $\hat{n}$ is successful, and the fault is reported as *link fault*. Figure b) communication fails between node n , $\hat{n}$ and $\tilde{n}$ - node n reports the fault as undecidable. Figure c) the fault detected by node n is reported as undecidable, as control messages on the path to node $\tilde{n}$ are lost.

B. Special cases

We assume that in-band signalling is used, in the sense that the network links and nodes under supervision are also the ones used for transmission of control messages of the algorithm. In some cases, this means that the origin of a detected fault is undecidable. This problem occurs if there are no other routes available to communicate control messages to a collaborating neighbour $\tilde{n}$ of the possibly failed node (see Figure 2b). Similar situations occur if any of the connections toward a collaborating node $\tilde{n}$ has failed (see Figure 2c). Thus, if control messages can be communicated via alternative paths, these problems become much less prominent. In practice this can be achieved in e.g., wireless networks sharing the same channel and in virtualised networks.

IV. FORMAL ALGORITHM DESCRIPTION

In this section, the subroutines forming the fault-detection process are described in algorithms 1, 3 and 2 shown below. Let n be a node in the network. Each node needs to keep track of the set of neighbouring nodes, N_n , as well as the sets of neighbours to each of these neighbours i , N_n^i .

Let each node n store an *error state* S_n^i for each neighbour i . Each S_n^i represents the current state of n_i as viewed from n , and can be assigned one of four different error states, namely *no fault*, *link fault*, *node fault* and finally, *link or node fault* (used when the cause of the detected fault is undecidable).

Algorithm 1 Monitor node $\hat{n}$ from node n

```

Require:  $\hat{n} \in N_n$ 
repeat
  if Test node  $\hat{n}$  from  $n$  fails then
    if  $S_n^{\hat{n}} = \text{No fault}$  then
      for all  $\tilde{n} \in N_n^{\hat{n}}$  do
        Confirm fault of  $\hat{n}$  for  $n$  in  $\tilde{n}$ 
      end for
    if Any  $\tilde{n} \in N_n^{\hat{n}}$  report success then
       $S_n^{\hat{n}} \leftarrow \text{Link fault}$ 
      for all  $\tilde{n} \in N_n^{\hat{n}}$  do
        Inform  $\tilde{n}$  about failed link
      end for
      Report failed link from  $n$  to  $\hat{n}$ 
    else if All  $\tilde{n} \in N_n^{\hat{n}}$  report fault then
       $S_n^{\hat{n}} \leftarrow \text{Node fault}$ 
      for all  $\tilde{n} \in N_n^{\hat{n}}$  do
         $S_{\tilde{n}}^{\hat{n}} \leftarrow \text{Node fault}$ 
      end for
      Report failed node  $\hat{n}$ 
    else
       $S_n^{\hat{n}} \leftarrow \text{Link or node fault}$ 
      Report link or node fault
    end if
  end if
end if
else
  if  $S_n^{\hat{n}} \neq \text{No fault}$  then
     $S_n^{\hat{n}} \leftarrow \text{No fault}$ 
    if  $S_n^{\hat{n}} = \text{Node fault}$  then
      for all  $\tilde{n} \in N_n^{\hat{n}}$  do
         $S_{\tilde{n}}^{\hat{n}} \leftarrow \text{No fault}$ 
      end for
    end if
    Report working link from  $n$  to  $\hat{n}$  and node  $\hat{n}$ 
  end if
end if
Wait  $\tau$  s
until  $\hat{n}$  disconnects

```

Algorithm 2 Test node $\hat{n}$ from n

```

Require:  $\hat{n} \in N_n$ 
repeat
  Send test transaction to  $\hat{n}$ 
  Wait  $\theta$  s
until Any response or  $\prod_i (1 - P(R_{\Delta t}^{(i)})) < \psi$ 
if Any response then
  return Success
else
  return Fault
end if

```

Algorithm 3 Confirm fault of $\hat{n}$ for n in $\tilde{n}$

```

Require:  $\hat{n} \in N_{\tilde{n}}, n \in N_{\tilde{n}}^{\hat{n}}$ 
 $t \leftarrow$  Test node  $\hat{n}$  from  $\tilde{n}$ 
Report  $t$  to  $n$ 

```

V. SIMULATION ENVIRONMENT AND IMPLEMENTATION

We have implemented the algorithm in the discrete event simulator environment OMNET++ [15], and used it to simulate link delays, fault events (i.e., communication faults) and drop rates. In performed simulations, randomly selected Gamma parameters drawn from a normal distribution were used to symmetrically simulate probe response delays on each link. Further, fault events were randomly generated over the whole population of nodes and links, drawn from a Poisson distribution with parameter λ , specifying the expected number of fault events within a given time period. Finally, the drop rate was symmetrically set on all links and randomly drawn from a Gaussian distribution with mean ξ and deviation $\sigma = 0.2\xi$.

VI. EXPERIMENTS

We have investigated the algorithm performance with respect to different parameter settings and varying network conditions. The results were obtained by performing two series of experiments on two types of network topologies.

In the first series of experiments, the algorithm performance was tested when varying the parameters ψ and c_τ in $\tau = c_\tau f_{cdf}^{-1}(l)$, while holding the expected number of fault events $\lambda = 5$ and the drop rate $\xi = 0.025$ fixed. In the second series of experiments, the algorithm performance under varying network conditions was tested for different values of ξ and λ while $c_\tau = 2^8$ and $\psi = 10^{-4}$ were held fixed.

In all the experiments we assumed that in each period of 4 hours the expected number of λ fault events was generated on uniformly selected network elements. Further, the fault duration was randomly set up to 1 hour. Simulated response delays in each direction were based on randomly drawn parameter values from a Gaussian distribution, with $\mu = 2.5^{-3}$, $\sigma = 5^{-4}$ for the scale parameter α and $\mu = 30$, $\sigma = 6$ for the shape parameter β . The probing interval θ between individual probes was set to $\theta = f_{cdf}^{-1}(0.8)$. During initialisation, each node sent 200 probes to obtain preliminary estimates of α and β . For statistical significance, all results are based on 4 days of simulated time and shown as the mean of 10 runs.

A. Network topologies

The experiments were performed on a synthetically generated scale-free network, and on a real-world network topology. The synthetic network consists of 30 nodes and 81 undirected links, and was generated using the Barabási-Albert method, starting with a small random network of 5 nodes and 3 links added at each iteration [16]. Scale-free networks resemble to some degree the structures of real-world topologies. To achieve a slightly more realistic topology (such as nodes with single connections), 10% of the links were randomly removed. The real-world network topology consists here of 172 nodes and 381 undirected links, extracted from original network topology data from a European ISP (1755-EBONE) [17].

VII. RESULTS

As performance metrics we investigated the localisation rate, detection rate, false positives rate and the probe rate. The localisation rate is the number of faults that was correctly localised to a link or node relative the number of generated fault events of each type. The detection rate is based on the number of detected symptoms relative the total number of generated fault events. The rate of false positives is the number of detected symptoms caused by drop rates and other factors not related to generated fault events, relative the total number of detected fault symptoms. The probing rate is the number of probes needed to detect abnormal behaviour, and is here normalised by the largest number of probes for each series of experiments. Note that the probing rate is mainly used to show the probing behaviour for different parameter settings, rather than showing the number of actually sent probes.

A. Algorithm performance for different user parameters

In general, we observe from the results obtained in the first series of the experiments that nearly all of the generated fault events can be detected (Figure 3). Further, we see that the localisation rates for node faults are lower for the real-world ISP topology (Figure 3c, d) compared to the synthetic network (Figure 3a, b). This can be explained by the characteristics of the topologies. The synthetic network has a fraction of 0.03 single connections whereas the fraction is 0.13 for the ISP topology. The single connections and the lack of alternative paths for communicating algorithm control messages generally causes lower localisation rates for node faults (see section III).

The results obtained for increasing values of c_τ , indicate that the rate of false positives can be reduced (fig. 3a, c). Further, we observe that the localisation and probing rates are relatively stable up to a certain point. When c_τ is set to very large values, the overall performance decreases as an effect of reduced reliability of the estimated parameters, caused by significantly fewer probing tests. In addition, we see in Figure 4 that the detection time increases with c_τ , as a result of fewer probe tests. Combined with the results in Figure 3a and Figure 3c, it is verified that by adjusting τ satisfactory performance can be achieved at relatively low levels of link load caused by probing traffic. Thus, c_τ is a trade-off between communication overhead and detection performance.

Fig. 3: Performance rates with varying probe test interval $\tau = c_\tau f_{cdf}^{-1}(0.8)$ and detection threshold ψ , obtained from a synthetic network (Figure 3a,b) and a real-world ISP topology (Figure 3c,d) when holding drop rate and expected number of fault events fixed.

Fig. 4: Shortest mean detection time for varying probe test interval $\tau = c_\tau f_{cdf}^{-1}(0.8)$, obtained from the synthetic network (Figure 4a) and the real-world ISP topology (Figure 4b), when holding drop rate and expected number of fault events fixed.

In Figure 3b and 3d, we see that the rate of false positives varies with increasing values of parameter ψ . Since the ability to correctly localise a fault is independent from the detection threshold, the localisation rates are relatively fixed for different values of ψ . For a fixed level of dropped traffic, the probing rate decreases linearly with increasing ψ . Probabilistically a larger ψ means that the confidence of a detected symptom is relaxed. In turn, this leads to a higher degree of false positives as fewer probes are used in order to detect and localise fault symptoms. Indeed, we see that up to a certain value of ψ , the probing rate can be reduced while the number of false positives are kept relatively fixed. Thus, our results indicate that the value of ψ is a trade-off between the probing rate and the rate of false positives.

B. Detection performance for varying network conditions

The results from the second series of experiments generally show fixed detection rates for varying ξ and λ (Figure 5). Further, we see that the localisation rates decrease with increasing drop rate ξ as a result of dropped control messages

for the fault-localisation processes (Figure 5a, c). In addition, the rate of false positives and the probing rate increase with the drop rate ξ . In this case when ψ is fixed, we see that the probing rate is autonomously adapting to the drop rate, while the rate of false positives is kept relatively low.

For increasing number of fault events λ , we observe that the probing rate and the rate of false positives remain fairly invariant (Figure 5b, d). The localisation rate, on the other hand, gradually decreases for increasing values of λ , as a result of unavailable network equipment needed in the fault-localisation processes. The actual impact of increasing λ relates directly to the size of the network. In the smaller synthetic network the degradation in localisation performance is more significant compared to the much larger ISP network, relative the number of fault events.

Fig. 5: Performance rates with fixed probe test cost c_T and detection threshold ψ , obtained from a synthetic network (Figure 5a,b) and a real-world ISP topology (Figure 5c,d) when varying the drop rate ξ and the expected number of fault events λ .

VIII. CONCLUSION AND FUTURE WORK

We have presented a first approach to distributed, adaptive fault-detection and localisation. The use of statistical modelling of observed network behaviour (here probe response delay and packet drop rate) allows for autonomous configuration of algorithm parameters, such as probing intervals and decision-conditions used for fault-detection.

The gains of distributed probing are quick adaptation and fault-detection on local network level, compared to centralised methods. On the other hand, probing can cause increased link load, if for example the intervals are fixed and based on simple assumptions (as in conventional heartbeat probing). The problem is here addressed by adjusting probing intervals to the expected probe response delay, and by using two different intervals for probe tests and individual probes on each link.

The experimental results indicate that satisfactory performance, in terms of detected faults, can be achieved with

small rates of false positives for autonomously set probing parameters. As indicated earlier, the somewhat low localisation rates for node faults are due to the lack of alternative paths between nodes. Moreover, it has been verified that the number of probes needed to detect faults is autonomously adapted to observed network measurements. This property allows for fault-detection with high certainty and few false alarms.

Aiming for a fully adaptive method, future work include extensions for detection of drifting probe response delays, based on the same probabilistic model as described. Shifts in local network latencies can be symptoms of malfunctioning equipment, malicious activities, misconfiguration, varying user behaviour etc. For the purpose of capturing such shifts, we will investigate how to account for long-term network development, by estimating parameters from recently observed data while gradually forgetting about older observations. Finally, we will investigate the algorithm performance when using alternative paths for efficient communication of control messages.

REFERENCES

- [1] M. Steinder and A. Sethi, "A survey of fault localization techniques in computer networks," *Sc. Comp. Prog.*, vol. 53, no. 2, pp. 165–194, 2004.
- [2] M. Yu, H. Mokhtar, and M. Merabti, "A survey of fault management in wireless sensor networks," in *Proc. of PGNET Conference*, 2007.
- [3] S. Zhuang, D. Geels, I. Stoica, and R. Katz, "On failure detection algorithms in overlay networks," in *Proc. of the 24th Annual Joint Conf. of the IEEE Comp. and Comm. Soc.*, vol. 3, 2005, pp. 2112–2123.
- [4] P. Lee, V. Misra, and D. Rubenstein, "Toward optimal network fault correction via end-to-end inference," in *Proc. of the 26th IEEE Intl. Conf. on Computer Communications*, 2006, pp. 1343–1351.
- [5] Y. Bejerano and R. Rastogi, "Robust monitoring of link delays and faults in IP networks," *IEEE/ACM Trans. Net.*, vol. 14, no. 5, pp. 1092–1103, 2006.
- [6] I. Rish, M. Brodie, S. Ma, N. Odintsova, A. Beygelzimer, G. Grabarnik, and K. Hernandez, "Adaptive diagnosis in distributed systems," *IEEE Transactions on Neural Networks*, vol. 16, pp. 1088–1109, 2005.
- [7] Y. Tang, E. Al-Shaer, and R. Boutaba, "Active integrated fault localization in communication networks," in *Proc. of the 9th IFIP/IEEE International Symposium on Integrated Network Management*, 2005.
- [8] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient Overlay Networks," in *Proc. of the 18th ACM symposium on Operating systems principles*, New York, NY, USA, 2001, pp. 131–145.
- [9] P. Kumar, "Probability distributions conditioned by the available information: Gamma distribution and moments," *Lecture Notes in Computer Science*, vol. 2865, pp. 151–163, 2003.
- [10] R. Steinert and D. Gillblad, "An initial approach to distributed adaptive fault-handling in networked systems," Swedish Institute of Computer Science, SICS, Kista, Sweden, Tech. Rep. T2009:07, 2009.
- [11] J. Färber, "Network game traffic modelling," in *Proc. of the 1st workshop on Network and system support for games*. New York, NY, USA: ACM, 2002, pp. 53–57.
- [12] H. K. Choi and J. O. Limb, "A behavioral model of web traffic," in *Proc. of the Seventh Annual International Conference on Network Protocols*. Washington, DC: IEEE Computer Society, 1999, p. 327.
- [13] P. Huang and T. Hwang, "On new moment estimation of parameters of the generalized Gamma distribution using its characterization," *Taiwanese Journal of Mathematics*, vol. 10, no. 4, pp. 1083–1093, 2004.
- [14] J. Pearl, *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.
- [15] A. Varga and R. Hornig, "An overview of the OMNeT++ simulation environment," in *Proc. of the 1st Int. Conf. on Simulation tools and techniques for comm., networks, systems & workshops*, 2008, pp. 1–10.
- [16] M. E. J. Newman, "The structure and function of complex networks," *SIAM Review*, vol. 45, pp. 167–256, 2003.
- [17] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson, "Measuring ISP topologies with Rocketfuel," *IEEE/ACM Trans. Netw.*, vol. 12, no. 1, pp. 2–16, 2004.