Preprint

Long-term adaptation and distributed detection of local network changes

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 approach to distributed detection of local latency shifts in networked systems. For this purpose, response delay measurements are performed between neighbouring nodes via probing. The expected probe response delay on each connection is statistically modelled via parameter estimation. Adaptation to drifting delays is accounted for by the use of overlapping models, such that previous models are partially used as input to future models. Based on the symmetric Kullback-Leibler divergence metric, latency shifts can be detected by comparing the estimated parameters of the current and previous models. In order to reduce the number of detection alarms, thresholds for divergence and convergence are used.**

The method that we propose can be applied to many types of statistical distributions, and requires only constant memory compared to e.g., sliding window techniques and decay functions. Therefore, the method is applicable in various kinds of network equipment with limited capacity, such as sensor networks, mobile ad hoc networks etc. We have investigated the behaviour of the method for different model parameters. Further, we have tested the detection performance in network simulations, for both gradual and abrupt shifts in the probe response delay. The results indicate that over 90% of the shifts can be detected. Undetected shifts are mainly the effects of long convergence processes triggered by previous shifts. The overall performance depends on the characteristics of the shifts and the configuration of the model parameters.

***Index Terms*—change detection; adaptive monitoring; distributed probing; statistical modelling;**

I. INTRODUCTION

In networked systems, change detection is a difficult problem, as user behaviour, equipment and link quality varies over time [1]. Detection of temporal changes is often motivated by the interest of performing early or preventive actions to fault symptoms and anomalous behaviour (such as re-configuration of equipment), to maintain quality of service. Further, adaptation to various aspects of varying network behaviour is of particular interest, as it can be used to autonomously configure algorithm parameters, thereby reducing configuration efforts and improving algorithm performance [2].

In this paper, we address the problem of distributed change detection and long-term adaptation to observed link delays. Although we here focus on link delays, the method can be applied to other types of signals, such as packet loss, traffic load etc. The distributed approach is here based on statistical modelling of probe response delays measured between nodes (Figure 1). For each connection a statistical model is created via parameter estimation. Adaptation to changes in the expected probe response delay is done by using overlapping statistical models, such that older data is gradually forgotten as new data arrive. Changes are detected by comparing the underlying statistics of the current and previous models.

Essentially, the method offers reliable change detection with high certainty, as the statistical properties makes it less sensitive to outliers. Furthermore, the memory demands are small compared to other methods. In addition, the method is flexible as it can model signals based on different types of statistical distributions, without requiring rigorous modifications.

We have investigated the behaviour of the method when varying different parameters, such as the decay factor and the model size. In addition, we have tested the detection performance for different types of changes in network simulations.

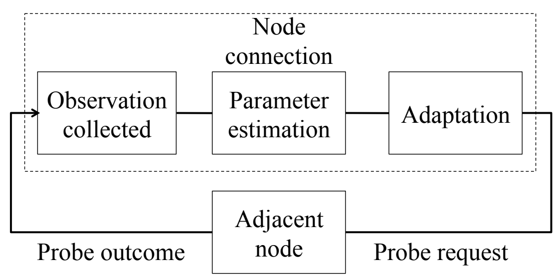

Fig. 1: Principal adaptation behaviour per node connection.

A. Related work

The simplest approach to long-term adaptation and change detection is the use of sliding windows, in which only recent data points are considered, and used for comparisons over time [3]. In turn, there are different methods for processing the data inside the windows, such as sketching (pairs of key-signal values) [4], buckets [5] or, statistics [6]–[8]. For the purpose of adjusting the importance of data points over time, decay functions are commonly applied to the data stream, as an alternative to simply discarding data out of scope [9].

The main drawback of both sliding window techniques and decay functions is that storage of data points is often required. In the worst case, space for the entire data stream is needed for exact tracking over time [9]. However, the memory demands can be significantly reduced from $O(N)$ to

$O(\log^2 N)$ when using e.g. aggregation techniques [5], [9]. In contrast to aforementioned methods, our statistical approach only requires a constant amount of memory, as data points are summed over time and used for parameter estimation.

Our work relates somewhat to the approach described by Hajji [10], who proposes a finite mixture model for network traffic. Changes are detected by observing the log-likelihood ratio of model estimates. For adaptation purposes, the author makes use of an exponential decay factor. In our method, data is modelled using overlapping statistical models that are continuously created and discarded over time. Instead of decay factors, prior knowledge from previous models is used as input to new models. This way, the robustness between models is increased, subsequently reducing the risk of false alarms while adaptation to new data is achieved.

B. Contribution

We offer an alternative method to adaptively learn new models developing over time, that also can be used to detect short-term and long-term shifts. This is done by using overlapping models, each learning the distribution for a limited part of the data stream. The key idea is that rather than applying decay functions or discarding data out of scope, older observations are successively accounted for by using previous models as prior input to future models. In effect, older observations are gradually forgotten as models are created and discarded. Moreover, the approach is flexible in the sense that it can be used with various types of statistical distributions of the observed data. Finally, the method is a simple alternative to other, more complex methods (e.g., [7], [8]), as it is memory-efficient and requires few user parameters.

II. Statistical model

The statistical model that we use is based on the probability density function $P(t)$ of probe response delays, which is the type of distribution that matches the characteristics of the data. We assume that the probe response delays can be modelled as a Gamma distribution,

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

where α and β are the scale and shape parameters, respectively. This model of interarrival times is motivated by the assumption that the probe response delays are sums of exponential transmission delays, caused by the queueing times in processing nodes. The assumption is supported by previous work [11], [12] and empirical network latency tests [2].

A. Parameter estimation

From observed probe response delays, the Gamma parameters α and β are estimated based on the method of moments approach. For this purpose, the parameters are estimated from the first and second sample moments $s_1 = \frac{1}{n}\sum_i \Delta t_i$ and $s_2 = \frac{1}{n}\sum_i \Delta t_i^2$ (e.g. [13], [14]). Given that $\alpha\beta = s_1$ and $\alpha^2\beta(\beta+1) = s_2$, the estimates α^* and β^* are

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

These estimations are frequently performed in each node. The method of moments is mainly used for the benefit of computational efficiency, since the capacity of the nodes in practice may vary. Therefore we also accept the loss of precision in the parameter estimates, compared to when using more advanced methods such as maximum likelihood estimations. Note, that the parameter estimation requires only constant memory to store the sums, which makes it an attractive method for use in network equipment with limited memory capacity.

B. Overlapping models

Long-term network development is here accounted for by having each node modelling probe response delays as overlapping Gamma distributions. Observed data in the previous model is here partially summarised and used as prior input to the next model, such that older data successively decay while the sensitivity to new data points is reduced (Figure 2). The average of the T first samples of the observed data in the

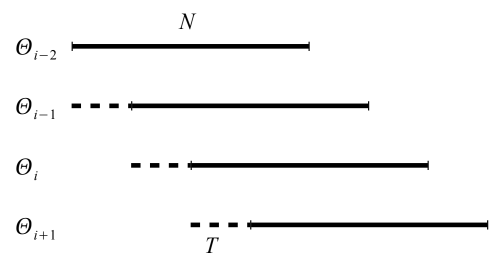

Fig. 2: Parameter estimation using overlapping models, with model size N and decay factor T.

current model is used as prior input to a new model (including the prior from the previous model). In order to take the priors into account, the moment estimations are modified to

$$s_1^{(i)} = \frac{\sum_j^n \Delta t_j^{(i)} + s_1^{(i-1)}}{n+1}, \quad s_2^{(i)} = \frac{\sum_j^n (\Delta t_j^{(i)})^2 + s_2^{(i-1)}}{n+1}, \quad (3)$$

such that the impact of earlier observations is gradually reduced as new models are created.

In general, the sensitivity to new data and the decay rate are trade-offs between the model size N and the decay factor T (see section IV). This means that the adaptation time and the robustness between models can be adjusted by varying these model parameters. By using prior input between models, a robust transition (less sensitive to new data) from one model to another is achieved, increasing the detection reliability.

III. Detecting latency shifts

In order to detect latency shifts, the current estimated model $\Theta_i^*(\alpha_i, \beta_i)$ is frequently compared to the previous model $\Theta_{i-1}^*(\alpha_{i-1}, \beta_{i-1})$ (Figure 2) using the symmetric Kullback-Leibler (KL) divergence $\mathcal{D}(\Theta_i^*, \Theta_{i-1}^*)$ as a metric,

$$\mathcal{D}(\Theta_i^*, \Theta_{i-1}^*) = \mathcal{D}_{KL}(\Theta_i^* || \Theta_{i-1}^*) + \mathcal{D}_{KL}(\Theta_{i-1}^* || \Theta_i^*), \quad (4)$$

where $\mathcal{D}_{KL}(\Theta_i^*||\Theta_j^*)$ is the divergence (or relative entropy) for Gamma distributions [15]:

$$\mathcal{D}_{KL}(\Theta_i^*||\Theta_j^*) = \psi(\beta_i)(\beta_i - \beta_j) - \beta_i + \log\frac{\Gamma(\beta_j)}{\Gamma(\beta_i)} + \beta_j \log\frac{\alpha_j}{\alpha_i} + \frac{\alpha_i\beta_i}{\alpha_j}. \quad (5)$$

The divergence metric is used for measuring the symmetric difference between current and previous models without regarding one of the models as 'true', which is commonly the case when measuring the divergence asymmetrically. The symmetric divergence is an effective and robust metric for change detection, and provides an intuitive interpretation of the difference between models.

Changes in the observed latency on the link are detected when the $\mathcal{D}(\Theta_i^*, \Theta_{i-1}^*)$ is higher than a divergence threshold η_{div}. Until convergence, $\mathcal{D}(\Theta_i^*, \Theta_{i-1}^*)$ can vary heavily around η_{div} which may cause several alarms for the duration of a shift. In order to reduce the amount of repeated reports, a convergence threshold $\eta_{conv} << \eta_{div}$ is used. When $\mathcal{D}(\Theta_i^*, \Theta_{i-1}^*) < \eta_{conv}$, latency shifts can again be reported.

IV. Experiments

As explained in section II-B, the sensitivity to new observations is a trade-off between model size and decay rate. The characteristics of the temporal development (such as gradual, abrupt etc.) are also significant factors to take into account when setting the model parameters. We have performed two experiments, in which model sensitivity and detection performance are investigated. In the first experiment, we tested the behaviour of the method for linear development while varying the model size and decay rate. In the second experiment, the detection performances for temporary stepwise shifts and linear changes were tested in network simulations. The results are presented in section V.

A. Model sensitivity and detection delay

In this experiment, we tested the impact on the sensitivity and detection delay for varying model size N and decay factor T between models, when the underlying structure of the observed probe response delays started to drift. The drift was produced by linearly varying the α and β parameters for simulated probe response delays, during a limited time period.

In the experiment, a probing mechanism immediately sent a new probe upon the reception of a probe response. Probe response delays Δt were randomly drawn from a Gamma distribution with parameters $\Theta(\alpha = 2.5 \times 10^{-3}, \beta = 44)$. After a certain time point t_{start}, the parameters were linearly increased with simulation time t, such that $\Theta_{i+1} = \Theta + k(t_i - t_{start})$ with the rate of change $k = 10^{-4}$. The effect of this parameter manipulation is that the simulated probe response delays start to drift in both mean and variance. The parameters drifted during a time period of 7200 time units, after which the Gamma parameters were set to the latest linear increment and were kept stationary until the end of the simulation. The simulation ended when $\mathcal{D}(\Theta_i^*, \Theta_{i-1}^*) < \eta_{conv}$, i.e., when the estimated parameters Θ^* were fully adapted to the new regime. The experiments were performed for the model parameters $N = \{1, 3, 5, 7\} \times 10^3$, decay factor $T = \{10, \ldots, 750\}$ and thresholds $\eta_{div} = 5 \times 10^{-3}$ and $\eta_{conv} = 10^{-5}$.

B. Performance in network simulations

Based on network simulations in OMNeT++ [16], we tested the detection performance of stepwise and linear changes in the Θ parameters used for modelling simulated probe response delays. Further, we used the Abilene core network topology from 2003, consisting of 11 nodes and 14 symmetric links. In the simulation, each node modelled each connection by probing adjacent nodes in a fully distributed manner. Whenever $\mathcal{D}(\Theta_i^*, \Theta_{i-1}^*) > \eta_{div}$, the detected shift was centrally logged and reported the adjacent node, in order to reduce the number of alarms. No further reports were sent or logged until $\mathcal{D}(\Theta_i^*, \Theta_{i-1}^*) < \eta_{conv}$.

In two separate experiments, the detection performance was tested for linear latency shifts (as described in previous experiment) and temporary stepwise shifts. The rate k of linear development was randomly selected in the interval of $\pm 5 \times 10^{-7}$ and $\pm 1 \times 10^{-3}$ for the α and β parameters. Furthermore, the scale and shape were allowed to vary in the interval of $[10^{-4}, 10]$ and $[5, 100]$, respectively, in order to limit the simulated probe response delays to a reasonable level. In the case of stepwise shifts, the scale parameter α was temporarily multiplied by a random factor in the interval $[1.5, 10]$, for a random duration. The model parameters were set to $N = 500$ and $T = 75$ with thresholds $\eta_{div} = 1.5 \times 10^{-2}$ and $\eta_{conv} = 10^{-7}$. Simulated probe response delays (in milliseconds) were set symmetrically based on randomly drawn parameter values from a Gaussian distribution, with $\{\mu = 2.5 \times 10^{-3}, \sigma = 5 \times 10^{-4}\}$ for parameter α and $\{\mu = 30, \sigma = 6\}$ for parameter β. Moreover, a symmetric drop rate on each link was randomly set based on a Gaussian distribution with $\mu = 2.5 \times 10^{-2}$ and $\sigma = 5 \times 10^{-3}$. The detection performance was tested for the increasing number of expected shift events $\lambda = \{1, \ldots, 5\}$ drawn from a Poisson distribution. The experiments were performed for a duration of 16 days of simulated time. The shifts were triggered on uniformly selected links in each period of 24h. The duration of each event varied randomly from 1h up to 4h.

The probing intervals $\tau = cF(l)^{-1}_{cdf}$ were autonomously adjusted for each connection based on the inverted cumulative density function of $F(\Delta t) = \int_0^{\Delta t} P(t)dt$ (see eq. 1), the fraction $l = 0.9$ and cost $c = 10^5$. This allows for probing intervals that are adapted to local network behaviour, which can reduce the link load caused by probing [2].

V. Results

We here examine the results from the two experiments described in the previous section. The performance rates are based on data extracted from simulation logs. For statistical significance, the results are shown as the average of 40 runs.

A. Model sensitivity and detection delay

Two aspects of the model behaviour are examined. First, we investigate the sensitivity between models for varying model parameters in terms of the average divergence, measured throughout the simulation. Second, we investigate the detection delay in terms of the number of samples.

We observe in Figure 3, that the average divergence increases with the decay factor. Naturally, higher degrees of decay increase the overall divergence between the models. Further, the divergence varies increasingly between the averaged measurements as less data is preserved. In practice, this kind of variation or noise can complicate configuration of the thresholds η_{div} and η_{conv}, possibly leading to increased alarm rates or greater amounts of undetected shifts. The results indicate that the sensitivity to new data points is mainly determined by the decay factor T, rather than the model size N.

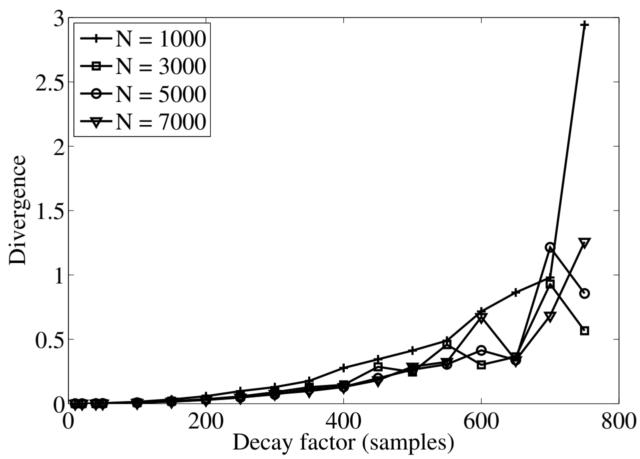

Fig. 3: Average divergence for different values of model size N and decay factor T.

In Figure 4, we see that the number of samples needed to detect drift for a fixed value of η_{div}, decreases as the decay factor T increases. The less information that is kept, the more noise occurs in the detection model, consequently leading to shorter detection time and possibly increased alarm rates. Further, we see that the detection delay varies with the model size, such that small models require fewer samples for detection than larger models. The results suggest that the number of samples required to detect changes is a trade-off between the degree of decay and model size.

The combined results in Figures 3 and 4, indicate that with a small decay factor, the statistical model is more robust to new data points. In effect, more data points are required to increase the certainty about a detected shift. By fine-tuning the model parameters, the detection performance of the investigated types of temporal development can be improved.

B. Performance in network simulations

We here use the detection rate for stepwise and linear shifts, respectively, to measure the performance relative the total number of events. Further, we have measured the rate of additional alarms relative the total number of reported shifts, and the number of spurious alarms relative the total number of alarms. The rate of additional alarms shows to which degree

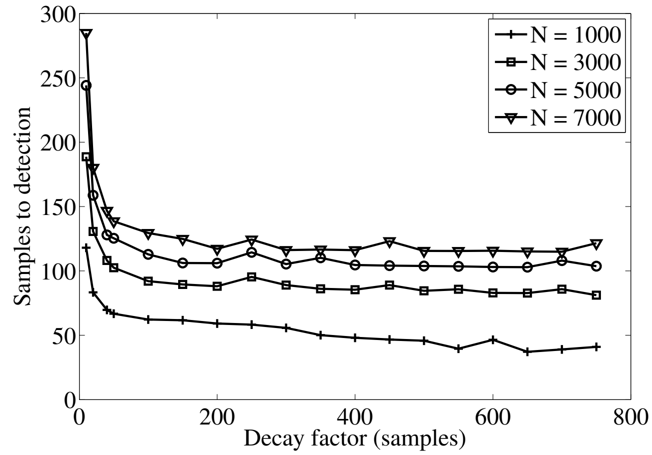

Fig. 4: Detection delay in terms of samples for varying model parameters N and T, using the detection threshold $\eta_{div} = 5 \times 10^{-3}$.

detected shifts are reported. Additional reports can occur from dropped reports between neighbouring nodes, coincidental detection between two nodes or, noisy convergence due to e.g., short intervals between events on one connection. The rate of spurious alarms is based on the number of alarms not related to any event (i.e., no event has yet been triggered on a certain link). Spurious alarms can occur as an effect of sensitive models and low detection thresholds.

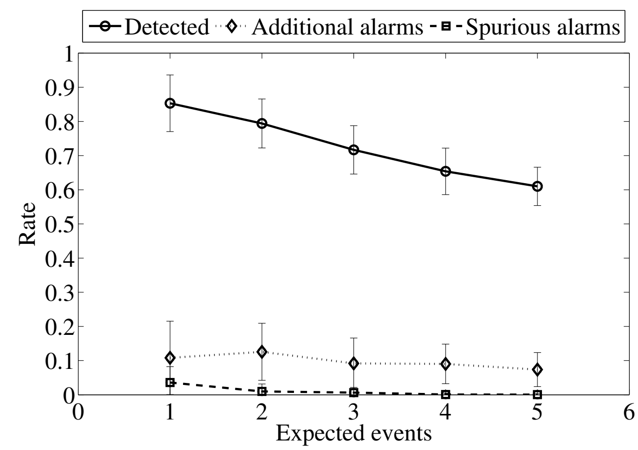

Fig. 5: Performance rates for detection of linearly increasing and decreasing changes.

We see in Figures 5 and 6 that the detection rates decrease with the number of expected events. At most 85.3% of the linear shifts (Figure 5) and 91.1% of the stepwise shifts (Figure 6) can be detected. Moreover, we observe for both types of temporal development, that the rate of additional alarms is relatively constant. In combination with the detection rate, the smaller amount of additional alarms (Figure 6) indicates that the detection of stepwise shifts is somewhat more robust compared to the detection of linear shifts (Figure 5). An examination of the data logs reveals that additional alarms mainly arise as an effect of shifts with complicated convergence, such that the divergence metric varies heavily between the η_{div} and

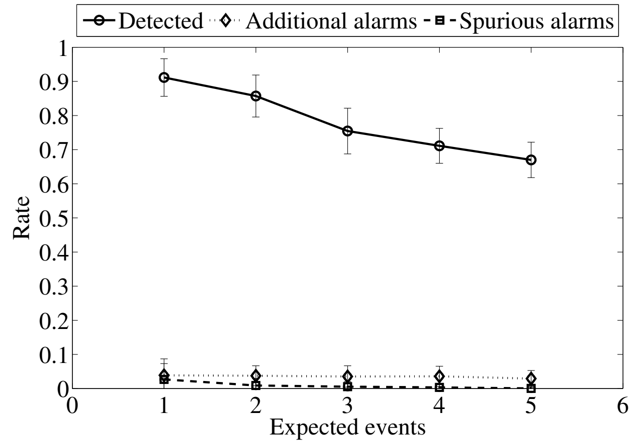

Fig. 6: Performance rates for detection of temporary stepwise changes obtained from network simulations.

η_{conv} thresholds. In the specific case of stepwise shifts, alarms are also raised as the response delay is shifted back to normal. As such alarms are triggered for the same event, these are here accounted for as additional alarms. In practice, adjusting the model parameters by observing the rate and duration of shifts can improve the detection performance and reduce the amount of additional alarms. Moreover, the rates of spurious alarms are in both cases relatively low, which means that links free from latency shifts occasionally report a spurious alarm. Again, adjusting the detection threshold can help reduce the amount of such alarms.

The results indicate that the performance varies for different types of temporal development, when using the same configuration of model parameters. Thus, the model parameters can be adjusted to a certain type of temporal development for improved performance.

VI. Conclusion and Future Work

We have presented an approach to change detection and adaptation of long-term temporal development of latency shifts, based on comparisons between overlapping statistical models. The models successively adapt to long-term changes, while early observations are gradually forgotten. The symmetric Kullback-Leibler divergence metric used for measuring changes between the models, allows for memory-efficient and reliable detection of signal shifts.

The performance of the method has been investigated for different types of temporal development. The results show that a majority of latency shifts can be detected. Some changes can remain undetected on a link if another shift starts before the model has converged from a previous shift. Further, the detection delay and model sensitivity to new data points are trade-offs between model size and decay rate. In general, it is indicated from the results that the overall performance can be improved by parameter adjustment and by taking the type of temporal development into account.

The main benefit of using the proposed method is that it requires constant memory compared to other methods, such as sliding windows and decay functions. This clearly reduces the memory demands of the network equipment, and allows for usage in various types of networks, in which the memory capacity is limited. We have here focused on detecting latency shifts based on Gamma distributions, but the approach can also be used for modelling signals with underlying structures matching other types of statistical distributions.

Future work includes the investigation of probabilistic divergence and convergence thresholds, for the purposes of improving detection performance and simplifying configuration. The thresholds that we currently use are difficult to set, and require some prior knowledge about the characteristics of the monitored signal in order to obtain a robust detection. Finally, we aim to test the detection performance using real-world network measurements as input.

References

[1] S. Wong, H. Yang, S. Lu, and V. Bharghavan, "Robust rate adaptation for 802.11 wireless networks," in *Proc. of the 12th Intl. Conf. on Mobile computing and networking*. ACM New York, USA, 2006, pp. 146–157.

[2] R. Steinert and D. Gillblad, "Towards distributed and adaptive detection and localisation of network faults," in *Proc. of the 6th Adv. Intl. Conf. on Telecommunications*, 2010, pp. 384–389.

[3] A. Bifet and R. Gavalda, "Learning from time-changing data with adaptive windowing," in *SIAM Intl. Conf. on Data Mining*, 2007.

[4] X. Li, F. Bian, M. Crovella, C. Diot, R. Govindan, G. Iannaccone, and A. Lakhina, "Detection and identification of network anomalies using sketch subspaces," in *Proc. of the 6th ACM SIGCOMM conf. on Internet measurement*. New York, NY, USA: ACM, 2006, pp. 147–152.

[5] M. Datar, A. Gionis, P. Indyk, and R. Motwani, "Maintaining stream statistics over sliding windows," in *Proc. of the 13th annual ACM-SIAM sym. on discrete algorithms*. Society for Industrial and Applied Mathematics Philadelphia, PA, USA, 2002, pp. 635–644.

[6] A. Gençata and B. Mukherjee, "Virtual-topology adaptation for WDM mesh networks under dynamic traffic," *IEEE/ACM Trans. Netw.*, vol. 11, no. 2, pp. 236–247, 2003.

[7] D. Kifer, S. Ben-David, and J. Gehrke, "Detecting change in data streams," in *Proc. of the 30th Intl. Conf. on Very Large Databases*. VLDB Endowment, 2004, pp. 180–191.

[8] T. Dasu, S. Krishnan, D. Lin, S. Venkatasubramanian, and K. Yi, "Change (Detection) You Can Believe in: Finding Distributional Shifts in Data Streams," *Advances in Intelligent Data Analysis VIII*, pp. 21–34, 2009.

[9] E. Cohen and M. Strauss, "Maintaining time-decaying stream aggregates," *Journal of Algorithms*, vol. 59, no. 1, pp. 19–36, 2006.

[10] H. Hajji, "Statistical analysis of network traffic for adaptive faults detection," *IEEE Trans. on Neural Netw.*, vol. 16, pp. 1053–1063, 2005.

[11] A. Mukherjee, "On the dynamics and significance of low frequency components of internet load," University of Pennsylvania, Philadelphia, USA, Tech. Rep. CIS-92-83, 1992.

[12] P. Ghosh, K. Basu, and S. Das, "Improving end-to-end quality-of-service in online multi-player wireless gaming networks," *Comp. Comm.*, vol. 31, pp. 2685–2698, 2008.

[13] P. Kumar, "Probability distributions conditioned by the available information: Gamma distribution and moments," *Lecture Notes in Computer Science*, vol. 2865, pp. 151–163, 2003.

[14] 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.

[15] R. Kwitt and A. Uhl, "Image similarity measurement by Kullback-Leibler divergences between complex wavelet subband statistics for texture retrieval," in *IEEE Intl. Conf. on Image Proc.*, 2008, pp. 933–936.

[16] 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.