# INFERENCE TOOL OF PRIORITY BASED PACKET FORWARDING ACROSS NETWORK VIA FEEDBACK

**D.V.Ramesh[1], D.Bujji Babu[2], S.Sreenivasulu[3]**

[1]*Student (M.Tech), Dept. of CSE, Prakasam Engineering College, AP, India,**rameshnet.it@gmail.com***
[2]*Associate Professor, Dept. of CSE, Prakasam Engineering College, AP, India,**bujji_bict@yahoo.com***
[3]*Head of the Department, Dept. of CSE, Prakasam Engineering College, AP, India, **sreenivasulusadineni@gmail.com***

**Abstract**

*Priority based packet forwarding in the network communication through routers is common mechanism now a days. This can have a significant impact on the accuracy of network measurements, the performance of applications and the effectiveness of network troubleshooting procedures. But There is no protocol to provide routing for end-to-end approach for forwarding packets based on priority. In this paper, We present an inference tool (end-to-end) approach for priority based packet forwarding inferred from feedback of experience through routers. We evaluated our approach via statistical analysis. It enables users to discover such network policies through measurements of packet losses of different packet types. In addition, we surveyed all related network operators and received responses for about half of them all confirming our inferences. out-of-order is unable to find many priority paths such as those implemented via traffic policing. We found it can detect existence of the mechanisms which induce delay differences among packet types such as slow processing path in the router and port-based load sharing.*

*Index Terms—Network inference, Packets, packet priority, Metrics, delay, loss of packet.*

---------- *** ----------

## 1. INTRODUCTION

Priority based Packet forwarding has been available in off-the shelf routers for quite a while, and various models from popular brands, such as Cisco and Juniper Networks offer support for it. These mechanisms are to be controlled for managing their networks, for example as a way of rate limiting certain classes of applications (e.g., peer-to-peer). Priority based Packet forwarding can have a significant impact on the performance of applications, on the accuracy of measurement tools' output, and on the effectiveness of network troubleshooting procedures.

Despite its potential impact, users, developers and most other network administrators have no information of such settings nor ways to procure it. In this paper, we present an end-to-end approach for priority based packet forwarding which inferred from routers experience by measuring the loss rate difference of different packet types. This approach can be used by the enterprises or end-users to discover whether their traffic are treated differently by the ISPs, and to determine whether the ISPs has fulfilled the contracts between them and the users. For example, after Comcast was found to treat BitTorrent with low priority in 2007, Comcast and BitTorrent reached an agreement to work together on network traffic in 2008 [4], [5].

There are a couple of challenges for designing and implementing packet forwarding effectively. First, background traffic fluctuations can severely affect the end-to-end inference accuracy of router properties. Second, probe traffic of a relatively large packet bursts are neither independent nor strong correlated. Most existing inference methods have to assume certain independence (i.e., i.i.d. processes) or strong correlation models for inference (e.g., back-to-back probe packets). However, a good mathematical model is needed to determine whether the loss rates difference between two packet types is the consequence of a random effect or being treated really differently. Third, more than two packet types are to be measured at the same time, so simply determining whether they are treated differently is not enough. To overcome these challenges, The mechanism takes the following three steps to infer priority based packet forwarding inference. First, it sends a relatively large amount of traffic to temporarily saturate the bottleneck traffic class capacity, which gives better resistance against background traffic fluctuations. Second, a robust *nonparametric* method is applied based on the *ranks* instead of pure loss rates. Thirdly, a rank-based metric is assigned to each packet type and hierarchical clustering method is used to group them when there are more than two packet types.

Our approach is validated via statistical analysis. As an extension of earlier paper [6], we compared this with the inference mechanisms based on other metrics with less overhead such as packet reordering (called *out-of-order (OOO)*) in this paper. OOO is unable to find many priority paths such as those implemented via traffic policing. On the other hand, it can detect existence of the mechanisms which induce delay differences among packet types such as slow processing path in the router and port-based load sharing.

### 1.1 Related Work

The efforts most closely related to this work are those identifying shared congestion [7]–[9]. Such efforts try to determine whether *two* congested flows are correlated and share a common congested queue along their paths. If the flows of different packet types are considered along a same path, the problem becomes to identify whether these flows do not share a common congested queue. While both problems are related clearly, we usually need to simultaneously consider a much larger number of packet types. Note that the correlation based method used for shared congestion identification methods requires back-to-back probing which, in our case, translates into pairs $\Omega(n^2)$ probing for n packet types. In addition, those efforts focused on flows which experience congestion (ignoring uncongested ones), so their probe traffic rate is low and not bursty. To identify priority based packet forwarding in routers, one must send relatively large amounts of traffic to temporarily force packet drops (by saturating the link). Thus, for better scalability and accuracy, our problem requires different measurement and statistical interference methods.

Kuzmanovic and Knightly proposed a framework for enabling network clients to measure a system's multiclass mechanisms and parameters [10]. The basic idea is similar to ours, i.e., to inject multiclass traffic into the system and use a statistical method to infer its scheduling types and parameters based on the output. However, the technique did not consider cross-traffic effects and only simulation results were presented. Priority based Packet forwarding inference also has some goals in common with efforts on network tomography [11]–[13]. However, unlike in network tomography where loss information and topology information are combined to infer link losses, we are identified of different packet types (based on protocol or port numbers) experience different loss rates. In addition, while probes used for network tomography are always nonintrusive in order to get accurate link loss/delay, saturate links are made in order to uncover the configuration of the routers. Finally, Reis *et al.* presented a content-based method to detect the middleboxes which modify the contents of web pages. The method in this paper detect the middleboxes which generate packet loss differences but do not modify the contents of the packets.

## 2. INFERRING PRIORITY BASED PACKET-FORWARDING

### 2.1. Background on Priority based Mechanisms

Network administrators can enforce priority/link-sharing mechanisms in a router by defining a traffic class (usually IP protocol and TCP/UDP port number) and associating with it a particular queuing/scheduling mechanism [15]. Some of the commonly available mechanisms are as follows.

• *Priority Queuing (PQ).* This allows users to assign arbitrarily defined packet classes to queues with different priorities. Since queues are served based on their priority, this allows specified packet types to be always sent before other packet types.

• *Proportional Share Scheduling (PSS).* With PSS each traffic class is given a weight. Bandwidth is allocated to classes in proportion to their respective weights. There is no strict priority difference between classes. There are different ways to implement this scheduling mechanism, e.g., Weighted Fair Queuing (WFQ), Weighted Round-Robin (WRR). In Cisco routers, the CBWFQ is Class-Based WFQ and the Custom Queuing is WRR based.

• *Policing.* This restricts the maximum rate of a traffic class. Traffic that exceeds the rate parameters is usually dropped. The traffic class cannot borrow unused bandwidth from others.

Only the first mechanism sets absolute priorities between traffic classes. There is no absolute priority in the other two classes, and the loss experienced by one class depends on whether its traffic rate exceeds its allocated bandwidth.

### 2.2 Choosing Inference Metric

Three basic end-to-end performance metrics, loss, delay and out-of-order, can all be used as inference metrics. This is because these metrics of different packet types can become different when a router is configured to treat them differently. Consider Pand *Q* of two priorities, where the high priority queue is always served first. Low priority packets will experience larger loss rates and longer queuing delays than the high priority packets. Besides, a low priority packet may arrive earlier than a high priority packet but leave after it while the contrary will never happen. The reordering events between them are *asymmetric*. Here, the loss, delay, and reordering can

all be used as a metric to infer priority settings. In this paper, The loss, delay and Out-Of-Order (OOO) based methods are used to name the inference methods from the experience at router . Essentially, the delay and reordering metrics are equivalent because when a packet gets lagged behind another packet, its delay should be larger than the other.

In the following, the *pros* and *cons* between loss metric and the other two metrics are discussed and the reason we choose packet loss eventually.

**2.2.1. The probe overhead of packet loss metric is larger than the other two.**

Obviously, loss rates difference will not become evident until the associated link (or a sublink for a traffic class) is saturated and begins to drop packets. This simple observation defines the basis of loss-based inference approach:

In order to reveal packet-forwarding priorities, one needs to saturate the path available bandwidth for a given class to produce loss rates difference among different classes. On the other hand, packet reordering and delay differences can be observed as soon as queue begins to build up. Sending of packets as much traffic as the loss-based approach for the reordering and delay-based approach is not needed .

**2.2.2. Loss difference can be observed for all kinds of QoS mechanisms while the other two cannot.**

Although using delay and reordering metrics can result in less probe overhead, they can not detect certain router QoS mechanisms simply because those mechanisms do not generate different delays at all. According to our test on a real Cisco router, *Policing* does not generate any packet reordering nor delay differences. However, any kinds of router QoS mechanisms will ultimately generate loss rates differences because that is the purpose of configuring such mechanisms.

**2.2.3. Packet delay difference can be caused by many other mechanisms than QoS.**

The root cause of packet reordering is the existence of parallel packet forwarding paths. Such paths can be in a router, parallel links between two routers, or different routes over several hops. When packets are split to these parallel paths according to their packet types and these paths have different delays, we'll observe asymmetric packet reordering and delay differences among different packet types. we show that many paths showing differences in packet delay and reordering were not caused by router QoS mechanisms, but the various forms of parallel forwarding paths such as slow processing inside routers and port-based load sharing.

To sum up, although the loss-based method has larger probe overhead, it can detect all kinds of QoS mechanisms and is not likely to be affected by parallelism in the forwarding paths as the other two. Hence, we use packet loss as the metric for inference. we addressed several interesting challenges as follows.

**a). The accuracy of end-to-end inference of router properties can be severely affected by background traffic fluctuations.**

Clearly, if one's probing introduces relatively small additional traffic, whether the link is saturated or not depends solely on the amount of background traffic. To make our approach more resistant to background traffic fluctuations we opt for sending relatively large amount of traffic to temporarily saturate bottleneck traffic class capacity, which increases the probability of observing loss rates difference. To note, the sender may not be able to saturate the bottleneck link due to limited resources, which is an inherit limitation of this method.

**b). Probe traffic of a relative large packet bursts are neither independent nor strongly correlated.**

Once the loss rate for each packet type is obtained, it is to be determined that whether the loss rates difference among them is large enough to conclude that they are treated differently. When packet losses can be described with a good mathematical model, e.g., independent and identical distribution (i.i.d) process, we can determine if the loss rates of different packet types were evidently different or not by comparing all, the loss rate of packet type, using parametric statistical methods. However, our probe packets are sent in large packet bursts. Packet losses in one burst are not independent but correlated. Hence, a *nonparametric* method is employed based on *ranks*, which is independent to underlying packet loss model and insensitive to loss correlations.

c). **Grouping is needed for multiple packet types probing.**

If two packet types are probed at one time, simply determining whether they are treated differently is enough. However, we sometimes probe more than two packet types and need to group them based on their priorities. Here, a rank-based metric is assigned to each packet type and use hierarchical clustering method to group them.

In summary, This approach saturates the link with relatively large amount of traffic and clusters packet types based on their

loss ranks. Such an approach gives better resistance against background traffic fluctuations, allows it to cope with the inherent characteristics of its measurement traffic, and enables it to measure more than two packet types at one time.

## 3. DESIGN OF INFERENCE TOOL

In this section, we present the design of the tool which has three steps:
1) Probing the path;
2) Deriving ranks; and
3) Partitioning packets with different forwarding priorities to different groups based on their ranks.

### 3.1 Probing the Path

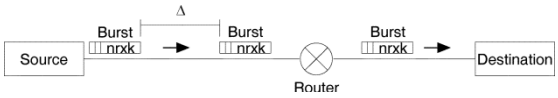

Figure 1. A burst consists of $n_r$ X k Packets

Figure. 1 illustrates our link probe method. Let k be the number packet types to be tested for transmission. The tool sends several bursts ($n_b$) from a source to a destination. The interval between bursts is ε. Each burst consists of $n_r$ rounds, in which k packets, one for each packet type studied, are interleaved in random order. So, there are $n_r$ X k back-to-back packets in each burst. There are three parameters ε, $n_r$ and $n_b$ for the probe method. In order to achieve independence between bursts, i.e., to ensure the router's queuing busy period caused by one burst does not interfere with the following one, ε should not be too small. On the other hand, in order not to experience large background traffic fluctuation duration the probe, we need to keep the whole probe duration within a relatively short period. In practice, ε is set to one to ten seconds to keep overall probe duration within several minutes.

The probe overhead is comparable with current available bandwidth measurement tools such as Pathload. In our experiment, $n_b$=32, $n_r$=40 and ε=10 . Usually, the user only needs to compare two or three packet types. Hence when k=2, 2560 packets are sent in one probe during 320 s.

### 3.2 Deriving Ranks

For every burst, loss rate ranks are computed by first sorting packet types in ascending order according to their packet loss rates in that burst and then assigning ranks in order, i.e., the packet type with the largest loss rate has rank 1, the one with the second largest loss rate has rank 2 and etc.

Similar to packet loss rates, due to randomness of packet losses, the ranks of different packet types are like random arrangements over the all bursts when the packet types are treated equally. On the other hand, the ranks of certain packet types are always small when they are treated with low priority. However, the advantage of using ranks is that there is a theory to bound the variance of loss ranks caused by the random effects whereas we do not have that bound for loss rates when the loss model is unknown.

### 3.3. Partitioning Based on Ranks

Every packet burst can be regarded as an observation. Identifying whether there is consistent difference among k ranks over m observations is a well-known statistical problem called *problem of m rankings*. Classic nonparametric solutions such as the Friedman test [18] can find whether there is consistent difference, but they do not make partitions among packet types. Therefore, we proposed to use Average Normalized Ranks (ANR) to group packet types when there is consistent difference. The ANR is the average of the ranks for a packet type over all bursts. Our statistical method is as follows:

***Calculate ANR.*** Let rim = (1,2,….) denote the rank for packet type i in m th burst. The NRim Normalized Rank is rim / k. The range of NRim is between 1 / k and 1.
The ANRi for packet type i is

$$\text{ANRi} = \left(\sum_{m=1}^{n_b} N\, R_m^i\right)/n_b \quad (1)$$

We developed a mathematical model for ANR using Central Limit Theorem discussed in appendix.

According to this model, when R > , $\theta_{1-\alpha,k_j,n_b,k}$ those packets should belong to multiple groups.

**Partition groups based on ANR**. A hierarchical divisive partition approach is used to cluster ANRs. Initially, all packet types belong to one group is assumed first, and then the above criteria is used to judge this assumption. If, they are partitioned into two groups using the means clustering algorithm [20]. This procedure is applied recursively to all newly partitioned groups until or there is only one packet type in the group. The pseudocode of the partition procedure is as follows :

Procedure 1 parttion(anrs)
// This procedure is applied recursively to all newly partitioned groups until or there is only one packet type in the group by taking Average Normalized Ranks anrs as input parameter.
{
1. $k_j$ := length(anrs);

2. if $k_j = 1$ or $(\max(anrs) - \min(anrs) \leq \theta_{1-\alpha,k_j,n_b,k})$ then
3. return anrs;
4. else
5. $[anrs_1, anrs_2] :=$ kmeans(anrs,2);
6. return ([partition($anrs_1$), partition($anrs_2$)]);
7. endif
}

## 3.4 Performance Analysis on Priority Group Partitioning

### 3.4.1. Methodology:

We first simulate many sets of random rank values (given the number of priority groups and packet types) that satisfy the following two conditions:

1) Every packet type i in priority group $G_u$, and every packet type j in priority group $G_v$ , the loss rate rank $r_i > r_j$ when $G_u$has higher priority than $G_v$.

2) For packet types within the same priority group, their ranks are randomly permuted in each burst in order to simulate the effects of random losses.

We then analyze the ANR group partition performance using the generated rank values. Note that the above conditions do not consider the worst case when the rank of in higher priority group can be smaller than the rank of some packet type in a lower priority group . However, we do consider an extreme case that violates those two constraints due to severe traffic fluctuations in the end of this section.

### 3.4.2. Cluster Error Types:

Group partition gives three types of errors.

a) *overpartitioning*: Suppose the number of partitioned groups is more than that of the actual situation(anrs). This results from the type 1 error associated with the criteria > θ , if the packet types are not in one group but actually they are in one. According to statistical theory, the percentage of these errors is less than the confidence level α, calculation of θ can chosen.

b) *underpartitioning*: The number of partitioned groups is less than that of the actual situation (anrs) which results from the type 2 error of the criteria. Theorem 2 in the Appendix denotes that when is above certain value, i.e., larger than 12 for k= 32, the percentage of type 2 errors is zero.

c) *mispartitioning*: The number of partitioned groups is equal to that of the actual situation (anrs), but some packet types are partitioned to wrong groups.

### 3.4.3. Average Cluster Error Percentage:

As the basic operation of our cluster method is to split several packet types into two groups, we first analyze the case of two priority groups (J=2). There are 256 combinations ($k_1$, $k_2$) where $k_1+k_2 = k \leq 32$ and $k_1 \leq k_2$. 64 simulations for each of the combination are tried, i.e., Sixteen 384 simulations in total. Table II shows the percentage of the first two types of cluster errors for different under or 0.001. To note, we do not show the percentages for the third type of errors since they are all zero.

As shown in this table, we should at least choose $n_b > 8$ for our experiments since both α have a large error percentage when $n_b = 8$. Besides, we will use $\alpha = 0.0001$ for our experiments since both $\propto = 0.01$ $\alpha = 0.0001$ and have 0% *underpartitioning* when $n_b \geq 16$ but $\alpha = 0.0001$ has smaller percentage of *overpartitioning*. The results also agree well with our theoretical analysis, i.e., there is no *underpartitioning* when $n_b \geq 16$ and the percentages of *overpartitioning* are close to the confidence level we set for $n_b \geq 32$. .

**TABLE I**

**KEY NOTATIONS**

| Notation | Meaning |
|---|---|
| $n_b$ | Number of bursts in a path measurement |
| nr | Number of rounds in one burst |
| Δ | Time interval between bursts in a path measurement |
| K,kj | Number of all tested packet types, number of tested packet types for class j |
| j | Number of queues/classes/groups |
| ANRi | The average normalized rank for packet type i over nb bursts |
| θ | Threshold used for comparing with ANR range |

**TABLE II**

**AVERAGE CLUSTER ERROR PERCENTAGE (%) FOR J = 2**

| α | $n_b$ / Type | 8 | 16 | 32 | 64 | 128 |
|---|---|---|---|---|---|---|
| 0.01 | Over partition | 8.5 | 2.52 | 2.23 | 2.52 | 2.42 |
| | Under partition | 0.20 | 0 | 0 | 0 | 0 |
| | sum | 8.7 | 2.52 | 2.23 | 2.52 | 2.42 |
| 0.001 | Over partition | 5.7 | 0.63 | 0.21 | 0.29 | 0.23 |
| | Under partition | 43.5 | 0 | 0 | 0 | 0 |
| | sum | 49 | 0.63 | 0.21 | 0.29 | 0.23 |

**TABLE III**

**AVERAGE CLUSTER ERROR PERCENTAGE (%) FOR J = 3, 4, 5,**

α=0.0001

| $n_b$ / J | 8 | 16 | 32 | 64 | 128 |
|---|---|---|---|---|---|
| 3 | 50.0 | 0.30 | 0.31 | 0.35 | 0.46 |
| 4 | 69.0 | 0.44 | 0.39 | 0.54 | 0.53 |
| 5 | 82.1 | 0.93 | 0.60 | 0.52 | 0.48 |

When j > 2, the number of ($k_1$,$k_2$,……,$k_j$) combinations grows dramatically for $\sum k_j \leq 32$ and $k_j \leq k_j + 1$ . we tested all combinations , but with reduced number of simulations for each combination to keep the total number of simulations about the same to that of J=2 . Table III shows the partitioning accuracy for J=3,4,5. our partitioning method consists of two basic operations , the thrsshold comparison and K means partition . When J increases, the number of such operations required for correct partitioning increases. As each operation may introduced error, the overall performance decreases as J increases, as shown in the table. However, the error percentages are all below 1% for $n_b \geq 16$.

### 3.2.4. Effects of Background Traffic Fluctuations

The background traffic may not be stable during the probe. Consider an extreme case where the background traffic is ON/OFF traffic. Suppose when a probe burst is sent during the ON period, the loss rates measured are well separated. When a probe is sent during the OFF period, no loss rate difference is observed (as the link is not saturated). The ranks in such bursts will be the same for all packet types. We fixed the total number of bursts to 32 and increased the number of bursts probed during the OFF period($n_0$). Fig. 2 shows how $n_0$ affects the cluster results. The error is the average performance of all ($k_1$, $k_2$) combinations for $k_1+k_2$=2,14,23,32,and $k_1 \leq k_2$, where we run 64 simulations for every combination. The cluster error increase suddenly when $n_0$ becomes larger than 13, 40% of 32 bursts. Below that value, the error percentage remains zero. This shows our clustering method does not require every burst to have loss rate differences, and that it is robust against quite large fluctuations on background traffic.

## 4. CONCLUSION

In this paper, we have demonstrated the tool , an end-to-end priority inference tool, is able to accurately infer the router's priority based packet forwarding. The contributions of this work are the findings over Internet as well as the methodology. In the experiments, the loss-based method detected several multi priority paths in the Internet. In searching for a method with less probe overhead than the loss-based method, we used packet reordering and delay as the inference metrics and found they were not as effective as loss in detecting packet forwarding priorities. They failed to flag many multi priority paths that were discovered by the loss-based method, and the paths flagged by them were mainly caused by the mechanisms which induce different delays among packets types. Hence, as for the packet forwarding priority, we believe packet loss metric is more suitable than the other two.

## APPENDIX

*Theorem 1:* When packets are in a same class j , the range of this class ($R=ANR_{max} - ANR_{min}$) for $n_b$ bursts at confidence level 1 - α is

$$\theta_{1-\alpha,k_j,n_b,k} = Q_{1-\alpha,k_j} \text{ X } \sqrt{k_j^2 - 1} \text{ / k } \sqrt{12n_b} \quad (2)$$

where

$\theta_{1-\alpha,k_j}$ is the 100(1-α) percentile of the range (of $k_j$ i.i.d. standard normals) distribution.

***Proof:***

Let k be the total number of packet types. When $k_j$ packet types are in same class j , the $NR_i$ of a packet type within this class has the discrete uniform distribution at $i_0$ / k, $i_0$ + 1 / k, …. ($i_0 + k_j - 1$) / k, where $i_0$ is the minimum NR for this class. Then, $\sigma^2(NR_i) = \frac{(k_j^2-1)}{12k^2}$ . According to the central limit

theorem, $ANR_i = \frac{1}{n_b}\sum_{i=1}^{n_b} NRi$ approximates to the normal distribution with mean $\mu = \mu(NRi)$ and variance $\sigma^2(ANR_i) = \sigma^2(NR_i)/n_b$. From the range of ANRs is $\theta_{1-\alpha,k_j,n_b,k} = Q_{1-\alpha,k_j} \text{ X } \sigma(ANR_i)$.

## ACKNOWLEDGMENT

We would like to express our sincere thanks to Sri. Dr. Kancharla Ramaiah Secretary and Correspondent, Prakasam Engineering College, Kandukur, A.P. India for his support with providing research environment . We are extremely thankful to our collogues , friends and family members who are cooperated in this work.

## REFERENCES

[1] "Cisco Ios Quality of Service Solutions Configuration Guide Release 12.2," Cisco Systems [Online].

[2] "Filter-based forwarding," Juniper Networks, 2001 [Online].

[3] P. Grant and J. Drucker, "Phone, cable firms rein in consumers' Internet use," 2005 [Online].

[4] J. Cheng, "Evidence mounts that Comcast is targeting Bittorrent traffic," 2007 [Online].

[5] V. Kumar, "Comcast, Bittorrent to work together on network traffic," 2008 [Online].

[6] G. Lu, Y. Chen, S. Birrer, F. E. Bustamante, C. Y. Cheung, and X. Li, "End-to-end inference of router packet forwarding priority," in Proc. IEEE INFOCOM, 2007, pp. 1784–1792.

[7] K. Harfoush, A. Bestavros, and J. Byers, "Robust identification of shared losses using end-to-end unicast probes," in Proc. IEEE ICNP, 2000, pp. 22–36.

[8] D. Rubenstein, J. Kurose, and D. Towsley, "Detecting shared congestion of flows via end-to-end measurement," IEEE/ACM Trans. Netw., vol. 10, no. 3, pp. 381–395, Jun. 2002.

[9] M. S. Kim, T. Kim, Y. Shin, S. S. Lam, and E. J. Powers, "A wavelet-based approach to detect shared congestion," in Proc. ACM SIGCOMM, 2004, pp. 293–306.

[10] A. Kuzmanovic and E. W. Knightly, "Measuring service in multiclass networks," in Proc. IEEE INFOCOM, 2001, pp. 1281–1289.

[11] A. Coates, A. Hero, III, R. Nowak, and B. Yu, "Internet tomography," IEEE Signal Process. Mag., vol. 19, no. 3, pp. 47–65, May 2002.

[12] T. Bu, N. Duffield, F. Presti, and D. Towsley, "Network tomography on general topologies," in Proc. ACM SIGMETRICS, 2002, pp. 21–30.

[13] M. Rabbat, R. Nowak, and M. Coates, "Multiple source, multiple destination network tomography," in Proc. IEEE INFOCOM, 2004, pp. 1628–1639.

[14] C. Reis, S. D. Gribble, T.Kohno, and N. C.Weaver, "Detecting in-flight page changes with Web tripwires," in Proc. NSDI, 2008, pp. 31–44.