Trajectory Sampling With Unreliable Reporting

Nick Duffield, *Fellow, IEEE*, and Matthias Grossglauser, *Member, IEEE*

***Abstract*—We define and evaluate methods to perform robust network monitoring using trajectory sampling in the presence of report loss. The first challenge is to reconstruct an unambiguous set of packet trajectories from the reports on sampled packets received at a collector. In this paper we extend the reporting paradigm of trajectory sampling to enable the elimination of ambiguous groups of reports, but without introducing bias into any characterization of traffic based on the surviving reports.**

Even after the elimination, a proportion of trajectories are incomplete due to report loss. A second challenge is to adapt measurement based applications (including network engineering, path tracing, and passive performance measurement) to incomplete trajectories. To achieve this, we propose a method to join multiple incomplete trajectories for inference, and analyze its performance. We also show how applications can distinguish between packet and report loss at the statistical level.

***Index Terms*—Bloom filters, network traffic measurement, packet loss, packet sampling.**

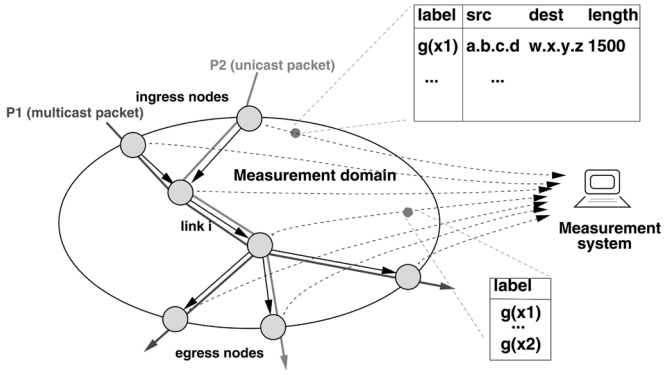

Fig. 1. Schematic representation of trajectory sampling. A measurement system collects packet *labels* from all the links within the domain. Labels are only collected from a pseudorandom subset of all the packets traversing the domain. Both the decision whether to sample a packet or not, and the packet label, are a function of the packet's invariant content.

I. Introduction

A. Motivation

TRAJECTORY SAMPLING (TS) has recently been proposed as a method to directly measure the spatial flow of traffic through an IP network at the packet level [4]. This is achieved by sampling a subset of packets consistently: each packet is sampled either at all routers it encounters, or at none. A router sends a report on each sampled packet to a collector. The reports contain sufficient information to distinguish different packets (with high probability); the collector is able to reconstruct the trajectory that the sampled packet took through the network.

The ability to reconstruct trajectories is impaired if reports are lost in transit; this consequently impairs the operation of measurement-based applications that exploit knowledge of the measured trajectories, either individually or through their statistical properties. In this paper we describe enhancements to reporting and reconstruction that enables measurement based applications to function when reports are subject to loss.

B. Elements of Trajectory Sampling

TS is realized through hash-based selection of packets. While processing each packet, routers calculate a hash over a domain within the invariant portion of the packet, i.e., that part that does not change from hop to hop. Fields can include, for example, source and destination IP address, protocol, IP identification, and fields form the transport header, and the payload if available. Fields to be excluded include the Type of Service field, which may be changed in transit, the Time to Live field in the IP packet header, and the IP header checksum, which is recalculated at each hop. The packet is selected for reporting if its hash falls within a set known as the selection range. When all routers use the same hash function, domain and selection range, the selection decision for each packet is the same at all routers: the packet is sampled either everywhere or nowhere; see Fig. 1.

Although hash-based selection is deterministic on packet content, selection can appear only weakly correlated with any field of the hash domain. This requires two things. Firstly, the hash function should be strong in the sense that small changes in the input (flipping a bit) generates large changes in the output. Secondly, there should be large variability in the content of each field in the hash domain. Under these conditions, hash values cover the range of the hash function nearly uniformly: hence the average sampling probability is the fraction of the range that is covered by the selection range. The selection range is to be configurable in the router; by setting it the router administrator controls the packet sampling rate.

Choice of selection hash function must trade off strength and computational complexity against the requirement that it be simple enough to compute on each packet at line rate. Specific hash functions are evaluated in [8]. Hash-based packet selection is being standardized in the Packet Sampling (PSAMP) Working Group of the Internet Engineering Task Force (IETF) [7]. The standard will include recommendation of hash function [14]. A discussion of the feasibility of implementing hash-based selection can be found in [3].

Manuscript received August 3, 2004; revised September 19, 2006; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor M. Krunz. A preliminary version of this paper was presented at IEEE INFOCOM 2004, Hong Kong, March 7–11, 2004.

N. Duffield is with AT&T Labs–Research, Florham Park, NJ 07932 USA (e-mail: duffield@research.att.com).

M. Grossglauser is with the School of Computer and Communication Sciences, EPFL, 1015 Lausanne, Switzerland (e-mail: matthias.grossglauser@epfl.ch).

Digital Object Identifier 10.1109/TNET.2007.899066

C. Applications of Trajectory Sampling

A strength of TS is that since trajectories are measured directly, measurement-based applications do not need to join trajectory samples with network state data (e.g., routing tables) for interpretation. This eliminates uncertainties (e.g., due to routing table fluctuations) and can save significant computational and administrative cost associated with obtaining and joining with the routing data. Applications of TS include:

i) *Network Engineering:* Reconstructed trajectories enable direct mapping of traffic to network topology. The traffic intensity of any class of traffic is estimated by dividing the intensity of the sampled traffic in that class by the sampling rate.

ii) *Path Tracing:* the form of the trajectories themselves can be used to detect routing loops (manifest as self-intersecting trajectories) and to trace paths taken by network attack traffic when source address spoofing obscures its originating host.

iii) *Passive Performance Measurement:* this is one of the major new applications of TS: passively measuring loss and delay experienced by regular traffic, rather than that of probe traffic injected into the network. Trajectories that terminate before reaching their destination are interpreted as packet loss. (There is no confusion with a packet entering a tunnel provided TS enabled routers are able to look beyond encapsulation headers to locate the appropriate hash domain). If routers include synchronized timestamps in packet reports, the latency of packets between routers can be found by subtraction. Sampling based on packet content is the only technique available for performing such measurements [13].

D. Security Issues

A requirement for accurate measurement is that a network attacker not be able to construct packets for which the sampling decision could be determined in advance. Such packets could either evade measurement, or be used to overwhelm the collection infrastructure. Since hash functions for packet selection are being standardized they will be known. However, two more pieces of information would be needed by an attacker. A weak defense is that hash selection range would have to be known. Some guesswork might help an attacker here. For example, if the hash range takes the form $[0, 2^n - 1]$, administrators might simply use the selection range $[0, 2^m - 1]$ (here $m < n$) in order to target a sampling rate of 2^{m-n}. A stronger defense is that the hash functions under standardization are parameterizable, each parameter giving rise to a different hash function. Applying standard security measures as one would with passwords would minimize the chance of successful attacks. An attacker might repeat a single packet in the hope that it is selected, but will receive no indication whether it is selected or not. We mention that network wide changes of hash function parameters could be performed in an automated way using standard network management procedures. Since changes would not be exactly simultaneous there would be some short transient period in which two parameters are deployed in different parts of the network.

E. Reporting and the Reconstruction of Trajectories

The packet reports contain a *key* and/or a *label*. The key contains fields from the invariant portion of the IP and transport headers, similar to the key used to distinguish packets of different IP flows. An example is the standard 5-tuple for TCP and UDP packets, namely, source and destination IP address, source and destination TCP/UDP port, and protocol number. By definition, the key does not distinguish between packets of the same flow. If this is desired, the report also contains a label. The label is the result of computing a second hash (distinct from that used for selection) over invariant parts of the packet. We assume that the label hash acts on fields that do vary between packets of a flow, e.g., IP identification, TCP sequence numbers, or even payload if available.

Our previous work [4], [5] has shown that a label length of about 4 bytes enables packets to be distinguished with high probability even in large networks. On the other hand, keys are expected to represent a significant portion of the IP and transport headers; ingress reports may also include routing state associated with the packet, such as routing prefix, and source and destination Autonomous System (AS). As a rule of thumb, we might expect keys to be $\alpha = 10$ times larger than labels. Thus, use of labels offers considerable reduction in bandwidth consumed by trajectory samples. Consider the ideal situation of collision free hash and no report loss. Then it would be sufficient to associate the key with the label only once. This leads to the paradigm of Label Reporting, in which the ingress link reports both key and label, while core links report only labels. Core reports for which there is no ingress report with matching label are discarded at the collector.

Label reporting from a path of T hops consumes $T + \alpha$ units of bandwidth (measured in units of label size), as compared with $T(1 + \alpha)$ if keys and labels are reported. The ratio of key to label bandwidth, $T(1 + \alpha)/(T + \alpha)$ is increasing in T and α. For a long path (say $T = 30$) this ratio represents an order of magnitude difference.

Approaches to the collection and joining of individual packet reports in order to reconstruct trajectories are described in [5]. One of the main issues arising is collision of label hashes from different packets, and their resolution. A simple and robust approach that avoids introducing topological bias is to discard all reports within a given time window for which identical labels are observed at one or more ingress routers. Some renormalization of measured traffic intensities is then required in order to compensate for discarding measurements when estimating original traffic intensities.

A related problem is how to accommodate constraints on the bandwidth for reporting. As the label size increases, reporting bandwidth increases while the frequency of hash collisions decreases. Since labels do eventually repeat for different packets, a related question is how to group individual reports temporally in preparation for trajectory reconstruction.

F. The Need for Duplicate Elimination

We briefly comment on the need to eliminate all duplicate labels in a measurement period. Without duplicate elimination, if two (or more) packets happen to possess the same label, the corresponding trajectory appears as the composite of the individual

trajectories followed by these packets. If this occurs rarely, the reliability of some types of statistical estimators inferred from a set of measured trajectories may not be affected (e.g., a simple estimator of the rate of traffic between two routers). This might suggest that we should simply ensure that duplicate labels are rare, but tolerate the occasional composite trajectory that results. However, there are two reasons why we would like to ensure that composite trajectories never occur.

First, we can envision applications of TS that check whether a particular condition *ever* happens in the network, e.g., a routing loop. Such applications could be very sensitive to the occurrence of even a single unfortunate overlap of two or more trajectories (which could easily result in the appearance of a "phantom" routing loop). More generally, composite trajectories can be "physically impossible" for a single packet; this complicates the design of algorithms in measurement applications.

Second, allowing composite trajectories also complicates storing trajectory samples, because it will lead to a combinatorial explosion of the space of possible trajectories. In a trajectory database, significant compression and efficiency gains result from many packets following the same trajectory, which suggests data structures that either break out the observed trajectories into a separate table or memory structure, or explicitly enumerate the possible trajectories [5]. This table will grow significantly due to this combinatorial explosion.

G. The Case Against Reliable Reporting

Our previous work assumed that report packets are transported reliably from the observation points in the measurement domain (typically routers) to the collector. However, while this simplifies the task of the collector of reconstructing trajectories, this reliable transport has some disadvantages.

First, reliable transport requires the observing device be addressable for feedback (ACKs or NACKs); while this is usually not a problem if the device is a router, it precludes transparent devices that simply inject report packets into the network without being addressable themselves (such a device might sit on a router's linecard, for example).

Second, it is necessary in the reliable scenario to match the report generation rate to the available transport rate, as any excess packets have to be buffered by the measurement device until they can be delivered. This in turn requires a well-designed outer control loop to quickly adjust the sampling rate in case a mismatch exists. In the unreliable scenario, the device has the additional option to react to a short-term overload condition simply by dropping some packets itself. Appropriate congestion control is still required; we simply argue that in the unreliable case, we have more leeway to design this control (e.g., by averaging over longer timescales).

H. Scenarios for Information Loss

The most challenging scenario for TS is the export of reports across a wide area network (WAN) that offers only best effort service. In this case report loss may be highly variable and essentially uncontrolled. A tamer scenario is to use a two stage export procedure. First, routers located in a routing center export reports to co-located staging servers. The main function of staging servers is to receive and buffer the reports before exporting them reliably over the wide area internet. Even though the initial export from router to staging server may use an unreliable transport such as UDP, loss of export packets can be controlled by using dedicated management networks, or in-band over relatively tightly controlled network links in which loss is rarer than in the WAN. The staging servers may also play a larger role in distributed analysis, for example by performing local analysis. We refer the reader to [6] for description of such a multistage data collection infrastructure.

For the present work, we observe that even with good management of transport resources, data loss may still occur due to expected changes in traffic load, or resource contention within the routers or other devices in the measurement infrastructure. For these reasons, trajectory reconstruction analysis must be robust with respect to report loss.

I. Complications for Reconstruction and Unreliable Reporting

Unreliable reporting complicates trajectory reconstruction and statistical inference from the packet reports. The methods must be well adapted to the requirements of the applications described in Section I-C.

The first problem is how to eliminate duplicate labels in the presence of report loss. With label reporting, loss of reports from ingress routers leaves "orphan" label-only reports from the core that cannot generally be distinguished from reports on other packets with the same label. This exacerbates the problem of disambiguation of reports from distinct packets with the same label that is already present with reliable reporting, as described in [4].

This motivates a more robust method of duplicate elimination that degrades gracefully under report loss. Traffic volumes by path and class are an important input to network engineering. In estimating volumes, issues that arise during duplicate elimination are (a) how to avoid topological biasing against subsets of trajectories during elimination; and (b) how to renormalize the surviving measurements in order to estimate the original traffic volumes that gave rise to the samples.

These issues are further complicated in an environment where routers do not have perfectly synchronized clocks, and where both measured packets and reports to the collector are subject to random delays. The challenge for the collector is to ensure that duplicates are correctly eliminated even if reports arrive out of order, and without the inclusion in such reports of timestamps with respect to a network-wide reference time.

Once duplicates have been eliminated, there still remains the problem of adapting applications that perform analysis of packet trajectories to the occurrence of gaps in the reconstructed trajectories due to report loss. For path tracing, the problem is that measured trajectories may be incomplete due to report loss. One way to obtain complete paths is to overlay trajectories of multiple packets that are expected to follow the same path. When routing is stable, packets sharing a common IP destination (or even prefix) will have this property.

For passive performance measurement, the problem is to distinguish report loss from packet loss. This is not always possible at the level of individual packets. For example, loss of a packet at a given link e of a path will produce the same set of packet

reports at the collector as the loss of reports from all links subsequent to e on the path. Instead, we have to distinguish packet and report loss at the statistical level, employing trajectory samples from multiple packets.

J. Alternatives To Trajectory Sampling

We discuss alternatives to TS, and their drawbacks.

1) Ingress Packet Marking: In TS, a packet's hash value signals implicitly to the router whether the packet should be sampled. A alternative is to explicitly mark them for sampling on ingress, by randomly set a bit in the packet. Marked packets are selected for reporting at all routers they encounter. But this approach has two disadvantages. First, it requires allocating a bit for marking in the IP packet header. However, all bit positions in the IP4 are currently allocated, notwithstanding some proposals to overload header fields for path tracing applications [11]. Even if a small set of bits in the header were available as sample flags, this would put a strict limit on the number of concurrent sampling processes in a measurement domain, because each sampling process would require its own flag. This is a serious limitation for large networks, where many different, independent measurement tasks may operate at the same time. TS does not have such a limitation, because no explicit information needs to be carried in packets, regardless of the number of instances of TS running concurrently.

Second, a domain that used packet marking to signal selection would have to filter the mark for all incoming packets. Otherwise, it would be possible to overload the measurement subsystem by injecting marked packets. Sealing the network against this attack would require all edge routers to have this filtering capability. Making such a change would be a formidable task in a large multi-vendor environment. We saw in Section I-D that this is not an issue for TS if a strong parameterizable hash function is used, making it exceedingly difficult to craft streams of packets that would be selected. Also, TS can be deployed incrementally, enabling TS-based applications to operate for the logical overlay network spanned by TS enabled routers. For example, TS deployed between a subset of network edge routers would enable passive performance measurement between those points. Measuring to the network edge is currently a challenge for active measurement, since active measurement hosts are usually located in router centers, rather than the network edge.

2) Independent Sampling (IS): This entails routers selecting packets in an uncoordinated manner, each selecting some proportion of the packets that pass through it, e.g., by periodic or simple random sampling. IS destroys the trajectory semantic, since a given packet is unlikely to be sampled at all points on its trajectory: passive performance measurement of individual packets becomes practically impossible. Furthermore, it is not generally effective to substitute statistical performance measures. Section VI shows that for likely packet loss rates in the Internet, statistical estimation of loss in a link of transmission rate q incurs a variance roughly $2/(1-q)$ times larger for IS than for TS, e.g., 50 times larger for 1% loss.

K. Outline of the Paper

We describe the TS architecture and record concepts, our model, and notation in Section II. In Section III we describe a method to deal with report loss that enables trajectory reconstruction to be performed in an unbiased manner. Our approach is for ingress nodes to record the presence of labels in Bloom filters [1], which are periodically transmitted to the collector. A Bloom filter is a compressed set membership function, where the only possible error is a false positive, i.e., we falsely conclude that a test element is in the set. The collector checks every received label from any network link against the Bloom filters from all ingress points; if a label matches more than one Bloom filter, it is possible that several packets have produced the same label, and that label is eliminated as a duplicate. False positive matches give rise to some unique labels being eliminated as well. However, elimination is unbiased, and robust with respect to partial loss of the Bloom filter in transit. This enables unbiased inference of original traffic intensities. This is done transparently, using an *effective sampling rate* that is the product of the TS target sampling rate with the rate of duplicate elimination.

The ingress nodes export their Bloom filters to a collector in a sequence of packets: a Packetized Bloom Filter (PBF). The PBF is subject to loss in transmission, and part of our challenge is to perform unbiased duplicate elimination in the presence of spatially heterogeneous loss of PBF packets.

In Section IV, we assume that no network-wide reference time is available, and that both sampled packets and report packets from routers to the collector are subject to random delays. We extend the methods for trajectory reconstruction and for duplicate elimination to deal with these additional sources of uncertainty. Our proposed method essentially amounts to using a timer for every trajectory being assembled to ensure that all available reports are included, and to check this trajectory against every Bloom filter that might have caught a duplicate label that could have interfered with this trajectory.

Even after duplicate elimination, other applications must be adapted to report loss. Section V addresses the reconstruction of network paths from incomplete trajectories reconstructed from multiple packets in the same flow. Provided transmission rates are not identically zero, multiple packet reports that take the same network path eventually cover the path, in the sense that at least one report is received from each router on the path. We analyze the mean number of packets that must be reported to attain this coverage.

Section VI shows how link loss rates can be inferred even in the presence of report loss. The main idea is that the collector can infer the loss rates of report packets if the reports include sequence numbers. In both Sections V and VI we compare the performance of TS with applying the same methods with reports from independently sampled packets. In both cases, the performance is noticeably better for TS, and particularly so in the estimation of loss rate. We conclude in Section VII.

II. Overview and Notation

In this section, we give an overview of the proposed system architecture and methods we propose to deal with report loss. We first define some notation.

The *measurement domain* is a directed graph $G(V, E)$, where we refer to vertices as *routers* and to edges as *links*. The set of vertices comprises a set of *external routers*, a set of *edge routers*, and a set of *core routers*. External routers connect to edge routers through an *ingress link*. Links that are not ingress links are called *core links* or *internal links*; see Fig. 2.

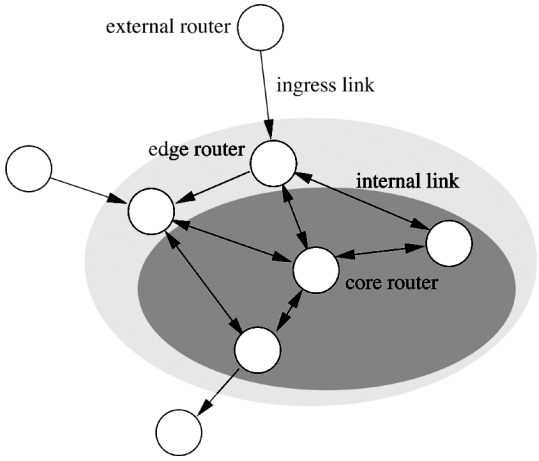

Fig. 2. A *measurement domain* consists of a set of routers under administrative control of a network operator, plus a set of external routers that act as sources and sinks for all traffic entering and leaving the network. The internal routers are further subdivided into *edge* and *core* routers, where edge routers are those having an incoming link (called *ingress link*) from an external router. In general, core routers only collect labels from their incoming links, while edge routers collect additional information from ingress links.

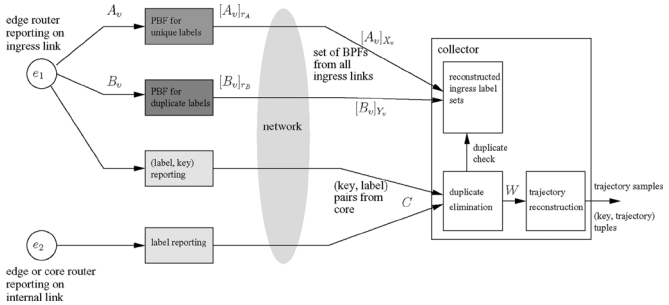

Fig. 3. The type of report generated for a sampled packet depends on the link on which a packet was observed. For an ingress link, a report contains the packet's label along with other information of interest (the key); separately, the label is reported as part of a set A_v or B_v, depending on whether the label was unique or duplicate. Both A_v and B_v are encoded as packetized Bloom filters $[A_v]_{X_v}, [B_v]_{Y_v}$ for transport. The collector tests every label l received from internal links against the PBFs, eliminating every label from consideration that has been observed multiple times.

A *trajectory* is a path,[1] i.e., an ordered set of links $(e_1, \ldots, e_n)$ for which the head vertex of e_{i+1} is the tail vertex of e_i, in the measurement domain G. $\mathcal{T}$ denotes the set of all valid trajectories. A *trajectory subset* is a set of links $\{e_1, \ldots, e_n\}$ that do not necessarily form a path, but which is a subset of at least one trajectory $\tau \in \mathcal{T}$. A trajectory subset arises when a trajectory is reported as a set of reports from each link it traverses, and some reports are lost.

A *trajectory sample* is a trajectory subset that is reconstructed by the collector based on reports received from the links on a packet's trajectory. In general, trajectory samples are not used in raw form, but are aggregated into higher-level statistics as outlined in the introduction. Examples include inference of the traffic and path matrices (see Section III-D); passive measurement of packet loss rates (see Section VI) and passive measurement of one-way packet delay (see e.g., [13]).

The concepts of *label graph* and *lossy label graph* record how many times a particular label l has been observed on every link in the network. Specifically, a label graph C_p^l is a mapping $E \to \mathbb{Z}$; it denotes how many times a label l has been observed *for each link* $e \in E$, where p is the sampling probability. For the case where labels are transported unreliably to the collector, where q_v is the probability that a label from a router v is lost, we call the resulting label graph a *lossy label graph* and denote it with $C_{p,\mathbf{q}}^l$, where $\mathbf{q} = (q_1, \ldots q_{|V|})$ contains all the report loss rates for every router.

A *path matrix* $\mathrm{PM}(k, \tau)$ is a function that maps a *key* k and a path τ to a volume of traffic. The key k is itself a property of packets, e.g., the source or destination IP address. Note that if the key is trivial (i.e., it takes the same value k_0 for each packet), then the path matrix $\mathrm{PM}(k_0, \tau)$ is simply the total traffic volume along path τ. The path matrix provides the most fine-grained spatial representation of the traffic that flows through a measurement domain over some time interval of interest. TS can be viewed as estimating the path matrix, the key being any function of the packet.

We discuss some of the assumptions underlying the description of the proposed method for inference from lossy reporting.

Duplicate elimination for one set of Packetized Bloom Filters. Initially, in Section III, we focus on the packet reports and Packetized Bloom Filters (PBFs) generated by a set of packet ingressing the network. We assume each ingress router generates one PBF from all the sampled packets. Reports carrying identical labels may have been generated by the same packet, or from different packets. The main challenge is to reconstruct the set of trajectories of packets sampled.

Duplicate elimination from series of PBFs. Section IV generalizes to the more realistic case that routers construct PBFs over successive epochs, a PBF being dispatched to the router at the end of each epoch. We do not assume that the epochs are aligned temporally among different routers, although we assume that the epochs have a common length. In this case the challenge for the collector is to determine which PBFs must be inspected for label collisions. Timeout based mechanisms for accumulating packet reports into label subgraphs were discussed in [5]. Here we must extend these methods to the accumulation of the PBFs for duplicate elimination.

The role of quasi-randomness. The performance analysis assumes that hash functions are *perfect*, i.e., a hash function computed over some a set of packets generates a set of i.i.d uniform random variables over the range of the function. This assumption is justified by the properties of hash functions and by our earlier work [4], where we show that there is enough entropy, i.e., variability from packet to packet, to ensure that sampling decisions and labels essentially appear random.

The basic TS architecture that is able to cope with report loss shown in Fig. 3. First, for every link in the measurement domain (internal and ingress links), reports are generated from sampled packets and transported to a collector. Multiple reports may be aggregated into a single report packet for efficiency, although time-sensitive applications will require the report on a

[1]Strictly speaking, a trajectory could more generally be a *walk*, which allows for repetitions of edges and vertices. This may arise, for example, because of a routing loop. However, such repetitions should be the exception, and we adopt the term *path* for simplicity.

given packet to be dispatched within a specified latency tolerance. When there is no danger of confusion, we do not make this distinction between report and the report packet explicit.

TS reports include one of both of the following:

- *Key:* fields from the invariant portion of the IP and transport headers. This is the standard notion of a flow key: for flows of packets defined by this key, the key value is by definition the same for all packets of the flow.
- *Label:* a hash calculated over an invariant part of the packet. We assume that the hash acts on fields that vary between packets of a flow (e.g., IP identification, TCP sequence numbers) in order the packets may be distinguished. This is important for applications such as passive delay measurement that must distinguish individual packets.

The basic reporting paradigm for TS is:

- *Label Reporting:* The ingress link reports both key and label, while core links report only labels. Core reports for which there is no ingress report with matching label are discarded at the collector.

The above is identical to the reporting paradigm in the reliable case, as proposed in [4]. We now add a new type of report that allows the collector to eliminate all duplicate labels, i.e., all reports from multiple packets that happened to generate identical labels. This is necessary because the collector has no guarantee to receive all the reports of these packets, with a possibility of missing these duplicates. The details of this process are explained in the next section.

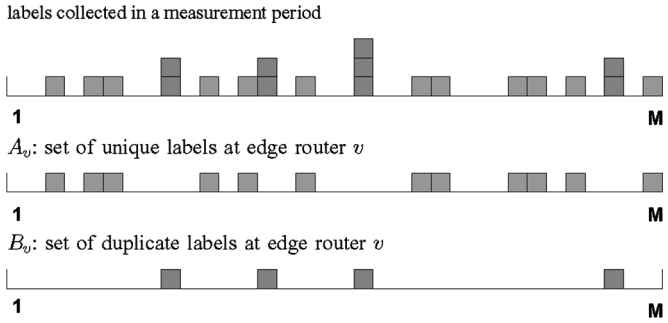

Fig. 4. At every ingress router v, the set of labels is partitioned into a set of unique labels A_v and a set of duplicate labels B_v, which are then separately encoded using a packetized Bloom filters.

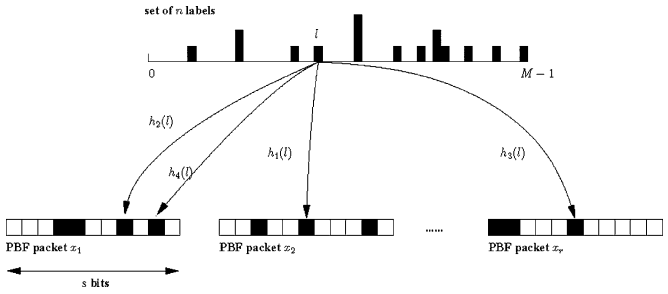

Fig. 5. A *packetized Bloom filter (PBF)* encodes the set of labels received at edge links into r packets of length s each.

III. Unbiased Duplicate Elimination Under Loss

A. Challenges for Unbiased Duplicate Elimination

As we have mentioned previously, there is a nonzero probability that two or more packets produce identical labels because the hash function is many-to-one [4], [5]. When more than one packet has the same label, it would sometimes be possible to disambiguate them. However, this disambiguation is costly, and may introduce bias into estimators if care is not taken. Therefore, in [5] we have proposed a different approach: eliminating all duplicate labels, whether they can be disambiguated or not. While this is slightly suboptimal because we ignore useful information, it greatly simplifies reconstructing trajectories and avoids bias in estimators.

If we assume that reports are carried reliably from routers to the collector, we can use the labels collected from ingress routers to detect and discard duplicate labels: if a label is observed multiple times on an ingress router, it is a duplicate.

In the unreliable case, this approach cannot be used directly because ingress reports may be lost, which can result in undetected duplicates. Consider the set of labels received from some ingress node over a time period of interest. If we simply transfer these labels as normal reports, i.e., as sequences of labels, then in the case where one or several reports are lost, we have no idea what the lost labels were. We could try to FEC-encode the set of labels using erasure codes in order to tolerate a certain loss rate, but this is computationally expensive, and would only work if the actual loss rate is smaller than the predefined target that the code was designed for. Another option would be to send the complement of the observed labels, i.e., all the label values *not* observed at each ingress link. However, this approach is very wasteful, as the complement set is much larger than the label set.

B. Packetized Bloom Filters

We instead propose a data structure based on *Bloom filters* to encode the set of labels observed on ingress links. A *Bloom filter* [1], [2] is a data structure to compress a set membership function. We essentially encode the set of labels as a Bloom filter and transport it to the encoder as a sequence of separate packets that we refer to as a *packetized Bloom filter (PBF)*.

More formally, let A be a set of labels out of an alphabet of size M, and let $n = |A|$ be its size. We denote by $[A]_r$ the r-packet PBF for A obtained as follows. The PBF uses a set of hash functions $\{h_i, i = 1, \ldots, k\}$, where h_i:$\{1, \ldots, M\} \rightarrow \{1, \ldots, rs\}$, with r the number of PBF packets generated from A (cf. Fig. 5) and s the size of a PBF packet. Specifically, each PBF packet j is a bit array x_j of length s, where $x_j(y - (j - 1)r) = 1$ if and only if $h_i(l) = y$ for at least one $i \in \{1, \ldots, k\}$ and any $l \in A$. Each packet also includes some control information (timestamp, j, etc.). In essence, this amounts to first generating a large Bloom filter of size rs, which we then transmit as r individual packets of size s each.

To check whether a candidate element l is in the compressed set, we check whether all the bit positions $\{h_1(l), \ldots, h_k(l)\}$ are set to one. Thus, a Bloom filter achieves compression of a set at the expense of false positives, but no false negatives. The bit positions corresponding to an element l present in the set are guaranteed to be set to one, while there is no guarantee that at

least one bit position corresponding to an element l' not in the set is set to zero.

Now note that we can ensure only false positives with any received subset of packets of a PBF, if we simply replace each missing packet with a vector of length s of only 1's. Obviously, the probability of false positives increases with the fraction of lost PBF packets. This property ensures that despite the lossy transport of the PBFs from ingress routers, any duplicate labels are *guaranteed* to be eliminated by the collector, because we cannot miss a label that has been observed more than once. Some unique labels are also eliminated due to false positives, and the probability of elimination increases with the number of lost PBF packets. However, we show below that these false positives can easily be compensated for.

C. Duplicate Elimination and the Elimination Rate

We use the PBF in our proposed architecture in the following way. Consider an edge router v, and the set of packets sampled on the ingress links connected to v. This set of packets generates a set of labels. We partition the set of labels into two subsets A_v and B_v, where A_v contains the set of *unique* labels at v (i.e., only a single packet gave rise to each label), and B_v contains the set of packets with *duplicate* labels at v (i.e., multiple packets gave rise to each label); see Fig. 4.

We now generate two PBFs $[A_v]_{r_A}$ and $[B_v]_{r_B}$ from these sets and transmit them unreliably to the collection system. As some of the packets may be lost, only a subset X_v, Y_v of the original packets is received. The collector replaces each missing packet with a sequence of 1's of the same length. This ensures that the partial PBF still produces only false positives, albeit with a higher probability. We denote the resulting partial Bloom filter by $[A_v]_{X_v}$ and $[B_v]_{Y_v}$, and we write $l \in [A_v]_{X_v}$ to denote a test for membership of l in the PBF $[A]_X$, which means either $l \in A$ or a false positive.

Once the collector has received $([A_v]_{X_v}, [B_v]_{Y_v})$ for all edge routers v as well as explicit label reports (packets containing sets of labels) from core routers, we assume for now that it equalizes the false positive probability for all Bloom filters $\{[A_v]_{X_v}\}$.[2] It can achieve this by matching the false positive probability of all the received PBFs to the highest among them. This can be done, for example, by boosting the density of 1's by randomly setting additional bits in the received PBF to one. This equalization ensures that the duplicate elimination probability does not depend on what edge router v a label has been observed on.[3]

For every label l received explicitly from a core or edge router, the collector eliminates l

$$\text{if} \quad l \in [B_v]_{Y_v} \text{ for some } v \tag{1}$$

or

$$\text{if} \quad l \in [A_{v_1}]_{X'_{v_1}} \cup [A_{v_2}]_{X'_{v_2}} \text{ for some } v_1 \neq v_2, \tag{2}$$

In other words, any label l that has either been observed more than once at a single ingress router, and/or been observed at multiple ingress routers, is eliminated from the pool of labels. We call C the set of explicit labels received from internal links by the collector, and W the set of labels after duplicate elimination (cf. Fig. 3).

Note that because of the possibility for a Bloom filter to produce false positives, some globally unique labels can also be eliminated. However, we next show that the elimination process is unbiased. This implies that the set of labels left after the duplicate elimination can be regarded as having been produced by a label assignment process that assigns a *unique* label to every sampled packet, but at a lower sampling rate.

Consider a fictitious system in which labels are a-priori unique (e.g., by selecting them out of a very large alphabet, or through some global coordination). We denote by $C^*_{p,\mathbf{q}}$ the label subgraph obtained with sampling rate p and report loss probabilities $\mathbf{q}$.

Theorem 1: Assume a network $G(V, E)$ and a set of packets with associated trajectories. Then the label subgraph $C^l_{p,\mathbf{q}}$ for every received label l satisfies

$$\mathsf{P}\left[C^l_{p,\mathbf{q}}\right] = \mathsf{P}\left[C^*_{\beta p,\mathbf{q}}\right] \tag{3}$$

where $\beta = 1 - \mathsf{P}[l \text{ eliminated}]$.

Proof: Partition the set of labels of sampled packets into two sets U and V, where U denotes the set of unique labels, and where V denotes the complement. Note that a label $l \in U$ occurs only in the set $A_{v(l)}$, where $v(l)$ is the ingress router where the corresponding packet entered.

Consider an arbitrary duplicate label $l \in V$. By definition, this label either (a) occurs in the set $A_{v(l)}$ and at least one other set A_w or B_w with $w \neq v(l)$, or (b) it occurs in the set $B_{v(l)}$ and possibly one or more sets A_w or B_w with $w \neq v(l)$. This implies that a label in $l \in V$, by the fundamental property of Bloom filters (no false negatives), will be correctly eliminated by the criteria (1) and (2).

Next, consider an arbitrary unique label $l \in U$. Define the following events. For an ingress node w, $E_w = \{l \in [A_w]_{X'_w}\}$, and $F_w = \{l \in [B_w]_{Y_w}\}$. Note that $E_{v(l)}$ is always true, while E_w, $w \neq v(l)$, and F_w can be true only due to a false positive match in the corresponding PBF.

It follows from the perfect hashing assumption for the sets of hash functions $\{h_i\}$ used to compute the PBFs that for any $w \neq v(l)$, the events $\{E_w{:}w \neq v(l)\}$ and $\{F_w\}$ are independent of each other and of l. Furthermore, the $\mathsf{P}[E_w]$ depend only on $|X'_w|$, which by definition are equal. Both $\cup_{w \neq v(l)} E_w$ and $\cup_w F_w$ are independent of each other and do not depend on $(l, v(l))$. Therefore, the events $\{l\text{eliminated}\}$ for all $l \in U$ are equiprobable and mutually independent.

Given the perfect hashing assumption on the label hash l, the label l of each packet is independent of the packet itself, and it follows that every packet is eliminated independently with equal probability $1-\beta$. This is equivalent to sampling the packet population with sampling probability βp. ■

[2]Note that this procedure is adopted for simplicity, although it has the disadvantage of increasing the duplicate elimination probability. An alternative approach would consist in not equalizing the PBF lengths, but to renormalize the weights of different trajectories to avoid bias in estimators.

[3]Note the subtle point that equalization is not necessary for the PBFs $\{B_v\}$. This is because the elimination probability in the test (1) below of a unique label l observed at ingress v is the same for all routers v even without equalization, because any match of l against any of the Bs, including B_v, is due to a false positive. On the other hand, a match with A_v in (2) is guaranteed, and a false positive results if l also matches at least one other $A_{v'}$. This therefore depends on v.

D. Effective Sampling Rate and Applications

The above theorem implies that the set of labels W after duplicate elimination can be regarded as resulting from a TS process that avoids label collisions altogether, i.e., where every label is unique. This simplifies the statistical inference of estimators, such as the loss rates on a set of links, as there is no need to explicitly account for the possibility of label collisions to avoid bias in constructing these estimators. Rather, we consider sampling to have taken place with the *effective sampling rate* βp. We give two examples of inference using the effective sampling rate:

- *Inference of Packet Loss Rates:* An example is provided in Section VI, where we estimate link loss rates: we can simply work on the set W and assume a-priori unique labels when constructing an estimator. The only correction is to assume that the sampling rate was βp.
- *Inference of Path and Traffic Matrix Elements:* Let $\mathrm{PM}^{\mathrm{TS}}(k,\tau)$ denote the path matrix element of trajectory sampled traffic after duplicates have been eliminated. The corresponding path matrix element of the original traffic is estimated by dividing by the effective sampling rate:

$$\widehat{\mathrm{PM}}^{\mathrm{TS}}(k,\tau) = \mathrm{PM}^{\mathrm{TS}}(k,\tau)/(\beta p) \tag{4}$$

The traffic matrix elements are derived from the path matrix by aggregation. Let K_{in} be a packet input key, i.e., some function of the packet key that does not depend explicitly on the destination IP address or destination TCP/UDP port numbers. An example would be source Autonomous System (AS). Likewise, let K_{out} be a packet output key, i.e., some function of the packet key that does not depend explicitly on the source IP address or source TCP/UDP port numbers, e.g., the destination AS. The traffic matrix element corresponding to values k_{in} and k_{out} of K_{in} and K_{out} is

$$\mathrm{TM}(k_{\mathrm{in}},k_{\mathrm{out}}) = \sum_{\tau\in\mathcal{T}} \mathrm{PM}((k_{\mathrm{in}},k_{\mathrm{out}}),\tau) \tag{5}$$

Depending on the application, $\mathcal{T}$ might be the set of all paths in a domain for which there is a non-zero measured path matrix, or the set of all paths in the domain that connect specific ingress and egress links, and have non-zero measured path matrix. Combining (4) and (5), the original traffic matrix elements are estimated from those of the sampled traffic after duplicate elimination through division by the effective sampling rate βp.

Note that in practice, β can be easily derived from auxiliary information in report packets giving the size of the sets of labels before (C) and after (W) elimination. It follows from the law of large numbers that the ratio $|W|/|C|$ converges a.s. to β when the number of received labels grows large.

E. Parameter Settings for the PBF

Finally, we discuss parameter settings in the PBF. First, assume that all the labels are observed at a single point, where they are encoded in a PBF. We encode both sets A_v and B_v into PBFs; note that in practice $|A_v| \gg |B_v|$, therefore the cost is dominated by r_A.

Assume a fraction q of the r_A report packets is received, i.e., the report loss rate is $1-q$. When k is chosen optimally [2], a bit in the Bloom filter is zero or one with equal probability $1/2$, and the false positive probability is given by

$$f = (p_1)^k = (1-q/2)^k = (1-q/2)^{r_A s \ln 2/n} \tag{6}$$

where $p_1 = (1-q/2)$ is the probability that a bit in the *received* PBF is one. Therefore, to ensure a reasonably small error probability, a set of n elements has to be encoded into $r_A s = O(n/q)$ bits. As s should reasonably lie within the range of 10^3 to 10^4 bits in IP, this determines the number r_A of PBF packets that should be used to encode the labels received during an epoch. A similar reasoning gives r_B.

In our previous work [4], we computed the optimal number of sampled labels n^* and the optimal alphabet size M^*, given a constraint c on the total number of bits to be collected from the network in one measurement period. We showed that $M^* = c\log 2$ and $n^* = M^*/\log M^*$. Therefore, $r_A s = O(c/\log c)$, which is considerably cheaper than explicitly sending the complement of observed labels, which costs $O(c)$.

Labels are usually observed at multiple ingress links. If r_A and r_B are set to the same value for all ingress links, then this value has to be chosen according to the link with the highest packet rate. This has the advantage of simplicity, in particular for the equalization of the received PBFs, as discussed earlier. However, depending on the number and distribution of ingress links, this may be inefficient, because many links will send underpopulated PBFs.[4] Another approach would therefore consist in choosing r_A and r_B per ingress link, depending on its speed. This is more bandwidth efficient, but equalization would have to be achieved explicitly through (6), with p_1 the density of 1's of the received PBF. Also, this would require the use of different hash functions $h_i(\cdot)$, as their domain depends on the length of the PBF.

In summary, duplicate elimination described in this section ensures that duplicates are guaranteed to be eliminated. The Robustness to report loss was bought at the expense of the loss of a small number of unique labels. Appropriate dimensioning of the PBF sizes ensures that this additional loss rate remains small. The overhead of collecting PBF from ingress links is small with respect to the total overhead due to label collection from the entire network. Most importantly, the duplicate elimination process can be treated as simply subsampling the set of sampled packets. Therefore, the possibility of duplicates can be ignored by statistical estimators.

IV. Subgraph Accumulation and Trajectory Reconstruction

A. Issues With Domain-Synchronized Measurement Intervals

The previous section analyzed measurements (packet reports and PBFs) collected across the network from a single set of packets. This suggests a possible implementation of the measurement system in which measurements are collected and analyzed during each of a sequence of adjoining measurement intervals whose start and end times are aligned across all routers.

[4]We note that such underpopulated PBFs can be compressed, e.g., through runlength encoding.

However, there are two issues with this approach: the need to synchronize the intervals between different routers, and the fact that packet trajectories can span multiple windows.

Synchronization presents additional hardware requirements at router locations, such as the installation of receivers for GPS signals. Arranging alignment of measurement epochs across multiple domains may carry a large administrative overhead. To avoid these costs it is attractive to let routers report independently on measurement intervals without requiring alignment across routers, and attribute time to measurements according to their time of receipt at the collector. The disadvantage of this approach is that the partial overlap of measurement intervals must be accounted for at the collector during analysis.

The ramifications of the arrangement of measurement windows are different for packet reports as compared with PBFs. A packet's report is assumed to be dispatched within some bounded latency from the time the packet arrived at the router, independently of the arrangement of measurement windows. In this section we propose a timeout governed mechanism to group reports carrying the same label into label subgraphs. By choosing the timeout to be sufficiently long relative to transmission latencies in the network, we aim to group all reports generated by a given packet. But the timeout must be short enough for the label collision rate to be acceptably small.

Whereas packet reports can be dispatched immediately and independently from routers to the collector, PBFs are constructed progressively at each router during the measurement period. Finding label collisions in a given subgraph is more complex than described in Section III, since the duration of a label subgraph (i.e., the period between its first and last arriving packets) may span multiple measurement periods at routers. In this section we will show how to modify the approach of the previous section accordingly.

B. Accumulation of Label Subgraphs

Packets experience transmission, propagation, and queueing delays as they travel through the network. Reports are subject to various delays: labels are delayed in the router's label buffer if a measurement packet can transport reports on multiple packets; once sent out, the measurement packet experiences delay during the transport from the router to the collection system. Here we focus on how to accumulate reports bearing the same label into label subgraphs, when reports arrive at the collection system subject to random delays; see Fig. 6.

We make the following assumptions. First, we assume that the transit time through the domain for any packet is bounded above by a quantity T_{net}. Second, we assume that the time between arrival of a packet at a router, and receipt of a packet report at the collector is bounded above by a quantity T_{coll}. This latency has components due to buffering of the reports at the measuring router, and end-to-end transmission delay between the router and the collector. Even without buffering delay, T_{coll} could be larger than T_{net} if the collector is not located in the network domain from which measurements are collected. Third, we assume that all routers construct the individual PBFs over intervals of duration S, at the end of which they are transmitted and available at the collector after a maximum delay T_{pbf}, which includes buffering at the routers, transmission to the collector, and timeout at the collector for grouping the packets of the PBF. Any (or all) packets missing from the PBF at the end of the PBF timeout are replaced by a vector of 1's, as described in Section III-B. For simplicity we will assume that $S > T_{\mathrm{pbf}}$. We now describe a scheme called *subgraph-based timeout* under which to accumulate reports. We associate a timer with every possible label value l. (In practice, the timer would only have to be instantiated once a report carrying label l actually arrives). The timer is started whenever a label l is received at the collection system if no timer has already been instantiated for l. All arriving labels l are then accumulated into a subgraph, until the timer times out.

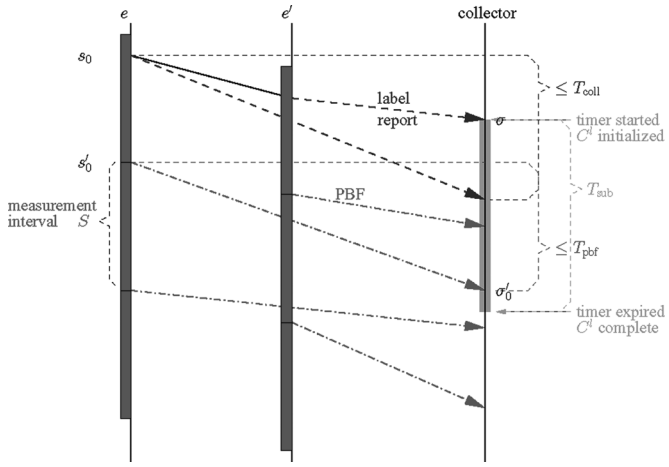

Fig. 6. Components of the delay experienced by samples from a single packet.

Let us examine what timeout value T_{sub} should be used to ensure that all reports from a given packet are accumulated. Consider a packet passing through the domain; let t_e denote the time of arrival of the packet at the source node of link e, and τ_e denote the time of arrival of the corresponding packet report (if it exists) at the collector. Let e and e' be two edges on the path. Then $\tau_e - t_e \in [0, T_{\mathrm{coll}}]$ and similarly for e', while $|t_e - t'_e| \leq T_{\mathrm{net}}$. Hence, the possible receipt times of reports for the two packet satisfy $|\tau_e - \tau_{e'}| \leq T_{\mathrm{coll}} + T_{\mathrm{net}}$. Consequently, in the subgraph-based timeout scheme, setting a timeout at least as large as $T_{\mathrm{sub}} = T_{\mathrm{coll}} + T_{\mathrm{net}}$ captures all reports on the packet in question. Fig. 6 illustrates the delay components.

C. Duplicate Elimination in Unaligned Measurement Intervals

In order to eliminate duplicates, we must test for membership, in the sense of (1) and (2), of the label l in all PBFs that could have recorded any sampled packet with label l whose report could have reached the collector during the timeout period. Let us determine the range of possible arrival times of such PBFs at the collector. Suppose that a report arrives at a time σ within a timeout interval $[0, T_{\mathrm{sub}}]$. The router encountered the packet producing the report at time $s \in [\sigma - T_{\mathrm{coll}}, \sigma]$ and entered the network at a time $s_0 \in [s - T_{\mathrm{net}}, s]$. The measurement interval at the ingress router during which the packet arrived end at a time $s'_0 \in [s_0, s_0 + S]$ and the corresponding PBF became available at the router at a time $\sigma'_0 \in [s'_0, s'_0 + T_{\mathrm{pbf}}]$. Combining these intervals we find that

$$\sigma'_0 \in [-T_{\mathrm{sub}}, T_{\mathrm{sub}} + T_{\mathrm{pbf}} + S] \tag{7}$$

The actual number of PBFs available in the interval (7) can vary according to the originating ingress router, due to variability in network conditions, and the relative phases of the measurement intervals. The maximum number of arrivals occurs if a sequence of PBFs is sent such that the first PBF is delayed maximally by $T_{\rm pbf}$, and the last PBF incurs no delay. It follows that the maximum number of arrivals is no larger than $2(T_{\rm sub}+T_{\rm pbf})/S+1$.

Thus, in the proposed scheme, duplicate elimination with nonaligned measurement intervals works as follows. We test for duplicates at the collector by checking every arriving label report against all the PBFs that might possibly have captured a label report falling into the same label subgraph. The following list shows how the collector reacts to an event occurring at time t to implement this scheme.

1) Arrival of a label report l from edge e: add e the to the label subgraph C^l; if $\texttt{labeltimer}(l)$ is not running, $t_{\rm start}(l):=t$, start $\texttt{labeltimer}(l)$, timeout $=T_{\rm sub}$, and start $\texttt{PBFtimer}(l,C^l)$, timeout $=T_{\rm sub}+T_{\rm sub}+S$.
2) Arrival of a PBF: put in PBF buffer for next step. PBF stays in buffer for $|\sigma_0'|$ time units.
3) Timer $\texttt{labeltimer}(l)$ expires for label l: stop accumulating label reports for C^l.
4) Timer $\texttt{PBFtimer}(l,C^l)$ expires for label l: check l (according to (1) and (2)) against all PBFs in buffer whose arrival time falls into $\sigma_0'+t_{\rm start}(l)$. If C^l is not eliminated, store it as a successful trajectory sample.

In analogy to Theorem 1 for the synchronous scenario, we argue that our scheme for asynchronous duplicate elimination is equivalent to subsampling the set of sampled packets.

Theorem 2: Assume a network $G(V,E)$ and a set of packets with associated trajectories, and a collector accumulating label subgraphs according to the algorithm above. Then the label subgraph $C^l_{p,\mathbf{q}}$ for every received label l (i) contains exactly one trajectory, and (ii) it satisfies

$$\mathsf{P}\left[C^{\boldsymbol{l}}_{p,\mathbf{q}}\right]=\mathsf{P}\left[C^{*}_{\beta p,\mathbf{q}}\right] \tag{8}$$

where $\beta=1-\mathsf{P}[l\ \text{eliminated}]$.

Proof: Property (i) follows from the fact that all label reports from the packet that started timer $\texttt{labeltimer}(l)$ will be included in the label subgraph. This is a consequence of the choice of $T_{\rm sub}$. Furthermore, any other packet that might generate an identical label that could arrive during the same timeout period would be eliminated by a PBF. This follows from the property of Bloom filters and the fact that we check all possible PBFs that might have seen the duplicate packet.

To show property (ii), we focus on a particular received label l, and we assume w.l.g. that the timer associated with this label was started at time 0. Denote by $\{\text{PBF}\}$ the set of PBFs that arrive at the collector within the interval σ_0'. By definition, every label l arriving between 0 and $T_{\rm sub}$ is checked against every element of $\{\text{PBF}\}$.

The rest of the proof proceeds as in Theorem 1, where U is interpreted as all the unique labels that were entered into $\{\text{PBF}\}$, and V as the corresponding set of duplicate labels. ■

We emphasize that the probability β that a sampled packet survives duplicate elimination is not constant from one label subgraph to the next. In fact, β depends on network conditions and traffic in complex ways, and may fluctuate over time. However, conditional on the β's of the sampled trajectories, the samples are indeed independent. Therefore, every trajectory sample produced by asynchronous trajectory reconstruction must be associated with the sampling probability βp that produced it, and the samples should be weighted according to their sampling probability in the construction of estimators. In order to simplify the discussion, we will not make this explicit in the remainder of this paper.

V. Path Coverage and Lossy Reporting

A. Coverage Count and Loss Model

In the introduction we stated that one of the new applications of TS is the ability to trace packet paths through a network. With lossy reporting, the set of links for which a packet is received may not form a contiguous path through the network. Nevertheless, when routing is stable, packets from a given traffic flow are expected to follow the same path (or set of paths if load balancing is used). Provided the report loss rate is not one on any link on a path, eventually a report will be received from every link on the path. Thus, taking the union of label subgraphs derived from multiple packets from the same flow, will eventually cover each link on the path or set of paths followed by the flows packets.

The covering approach requires that packets report a quantity that both identifies them as members of a flow, and also uniquely determines the packet's path. Here we will assume that the key performs this function. This is the case, for example, when packets are routed based on destination IP address which is reported in the key.

We shall assume that the set of packet reports has already undergone duplicate elimination as described in Section III. We derive expressions for the mean number of packets required to cover a path. Let τ be a trajectory, and let $T=\#\tau$ denote the number of links in τ. We assume that packets on the trajectory are sampled and reports dispatched to a collector from each link e in the trajectory. Consider a sequence of packets labeled by $n=1,2,\ldots$. The *coverage count* for τ is the smallest number of sampled packets needed to receive at least one packet report from each link on a path τ. For analysis we assume the following simple statistical model of TS: with probability p a packet is selected at all links on its trajectory; otherwise at none. We ignore transmission loss, and focus instead on report loss and assume that reports are independently successfully transmitted with probability q.

For this model, the average coverage count and its asymptotic behavior is characterized using the following result:

Theorem 3: Let $\{x_{ei}\colon e=1,\ldots,T;i=1,2,\ldots\}$ be i.i.d. indicator random variables with $\mathsf{P}[x_{ei}=1]=q$. Let $\hat{N}=\inf\{i\colon \min_e\max_{j\le i}x_{ej}=1\}$ abstract the coverage count.

i) $\mathsf{E}[\hat{N}]=F(T,q):=\sum_{k=1}^{T}\binom{T}{k}((-1)^{k-1})/(1-(1-q)^k)$
ii) $F(T,q)\to 1$ and $\partial F(T,q)/\partial q\to -T$ as $q\nearrow 1$.
iii) [(iii)] As $q\to 0$, $F(T,q)\sim F_1(T,q)=H_T/q$ where H_T is the harmonic number $\sum_{i=1}^{T}1/i$.
iv) $qF_1(T,q)\sim\gamma+\log T$ as $T\to\infty$, where γ is the Euler constant $\lim_{T\to\infty}(H_T-\log T)=0.577216\ldots$.

Proof: (i) $\hat{N} = \max_e \hat{N}^{(e)}$ where $\hat{N}^{(e)} = \inf\{i: x_{ei} = 1\}$. The $\hat{N}^{(e)}$ are i.i.d geometrically distributed random variables, with $\mathsf{P}[\hat{N}^{(e)} > n] = (1-q)^n$. Hence, $\mathsf{P}[\hat{N} > n] = 1 - (1-(1-q)^n)^T$ and $\mathsf{E}[\hat{N}] = \sum_{n\geq 0} \mathsf{P}[\hat{N} > n] = \sum_{n\geq 0} \sum_{k=1}^{T} \binom{T}{k} (-1)^{k+1} (1-q)^{nk} = F(T,q)$, after binomial expansion of $\mathsf{P}[\hat{N} > n]$; (ii) follows. (iii) $\lim_{q\to 0} qF(T,q) = \sum_{k=1}^{T} \binom{T}{k} (-1)^{k+1}/k = \int_0^1 dx(1-(1-x)^T)/x = \int_0^1 dx(1-x^T)/(1-x) = H_T$. (iv) follows from the definition of γ. ∎

B. Reporting Strategies and Mean Coverage

The mean coverage count for label reporting is calculated as follows. Only packets which generate a ingress label report that reaches the collector contribute. These occur at a rate pq relative to the original packet stream. Even if all other reports with the same label reach the collector, if the ingress report is loss, the other reports are discarded. For the substream of packets whose ingress report reaches the collector, the mean coverage count for reports from the $T-1$ core links is $F(T-1,q)$. Hence, the overall mean coverage count is

$$N_{\mathrm{TS,L}} = (pq)^{-1} F(T-1,q) \tag{9}$$

From Theorem 3, the dependence on the report loss rate q is

$$N_{\mathrm{TS,L}} \approx \begin{cases} \frac{1+T(1-q)}{p}, & q \approx 1 \\ H_{T-1}/(pq^2), & q \approx 0 \end{cases} \tag{10}$$

Note from (ii) that for nearly lossless reporting ($q \approx 1$) the mean coverage count grows affinely with the path length T. The rapid q^{-2} growth for small q can be tempered by adjusting the reporting strategy:

- *Key Reporting:* All routers report keys; labels may be reported as well if it is desired to distinguish different packets within a flow.

For key reporting, the mean coverage count is

$$N_{\mathrm{TS,K}} = p^{-1} F(T,q) \tag{11}$$

which behaves as $H_T/(pq)$ for small q, reducing the growth by q relative to label reporting. When report loss is small ($q \approx 1$), the behavior for small report loss ($q \approx 1$) is the same as for TS, to leading order in $(1-q)$.

C. Comparison With Independent Sampling

Routers which do not offer TS may still be able to sample packets; see e.g., [10]. With key reporting, trajectory coverage can then be performed at the collector. We model this with independent sampling (IS) of packets at probability p. The mean coverage count in this case is

$$N_{\mathrm{IS,K}} = F(T,pq) \tag{12}$$

We compare with key reporting for TS in two regimes. Suppose there is no report loss: $q = 1$. By Theorem 3:

$$\frac{N_{\mathrm{IS,K}}}{N_{\mathrm{TS,K}}} = pF(T,p) \approx H_T \tag{13}$$

since p is assumed small. A typical path might take T between 10 and 20, while for the longest paths observed in the wide area Internet we take $T = 30$; see [9]. The corresponding ratios are $H_{10} = 2.9$, $H_{20} = 3.6 H_{30} = 4.0$.

In the lossy regime $q \to 0$, then by Theorem 3,

$$\frac{N_{\mathrm{IS,K}}}{N_{\mathrm{TS,K}}} = \frac{pqF(T,pq)}{qF(T,q)} \approx 1 \tag{14}$$

since p is small. The decreasing advantage of TS relative to IS as $q \to 0$ is because once a report on a given packet has been received from one link, reports are reasonably likely to have been received from other links.

A fuller comparison of the sampling methods should also take account of the reporting bandwidth. Recall labels have the advantage of being smaller than keys, and hence consume less bandwidth. Let $\alpha > 1$ denote the ratio of the size of a key to the size of a label. Label sizes of 4 bytes have been found to be sufficient to distinguish packets in TS. On the other hand, a flow key may include a significant proportion of the IP and UDP/TCP packet headers. We assume α to be about 10.

Measuring in units of label size, key reporting consumes $T\alpha$ (a key of size α for each of T links) while label reporting consumes $T + \alpha$ without keys (T labels plus the ingress key), and $T + T\alpha$ with keys (label and key for each of T links). Comparing TS without keys and IS, the ratio of the mean bandwidth required to cover a trajectory of length T is

$$\frac{N_{\mathrm{IS,K}}}{N_{\mathrm{TS,L}}} \frac{T\alpha}{T+\alpha} = \frac{pqF(T,pq)}{F(T-1,q)} \frac{1}{T^{-1}+\alpha^{-1}} \tag{15}$$

With lossless reporting ($q = 1$) and small p this ratio is approximately $H_T/(T^{-1}+\alpha^{-1})$. With $\alpha = 10$ and taking $T = 10, 20$, and 30, yields ratios $15.$, $24.$, and $30.$ respectively. For lossy reporting, the ratio behaves as $q/(T^{-1}+\alpha^{-1})$. Thus, for sufficiently small q, IS can have a bandwidth advantage, i.e., the ratio becomes less than 1. This happens when $q < T^{-1} + \alpha^{-1}$, i.e., about 0.13 using above values for T and α. This would be an extremely low ambient transmission rate in the Internet, so in practice TS would be expected to have the bandwidth advantage over IS.

VI. Loss Inference in the Presence of Report Loss

A. Distinguishing Packet and Report Loss

Passive measurement of packet loss is one of the most attractive new applications of TS. With reliable reporting, packet loss is manifest by trajectories that terminate without the packet reaching its destination. Consider the case where traffic of a given key class is routed along a single path (i.e., stable routing and no load balancing). The loss rate for packets of that class at a link on that path is estimated as the proportion of packet reports for the class that terminate at that link.[5]

With unreliable reporting, reports may be lost. In order to estimate packet loss, we must disentangle the effect of report loss. Clearly this is not possible at the level of single packets. Loss of a packet at a given link e of a path will produce the same

[5]A potential alternative, namely comparing counts of arrived packets at each end of the link, requires counting in predefined traffic classes, and hence is not useful for ad hoc studies.

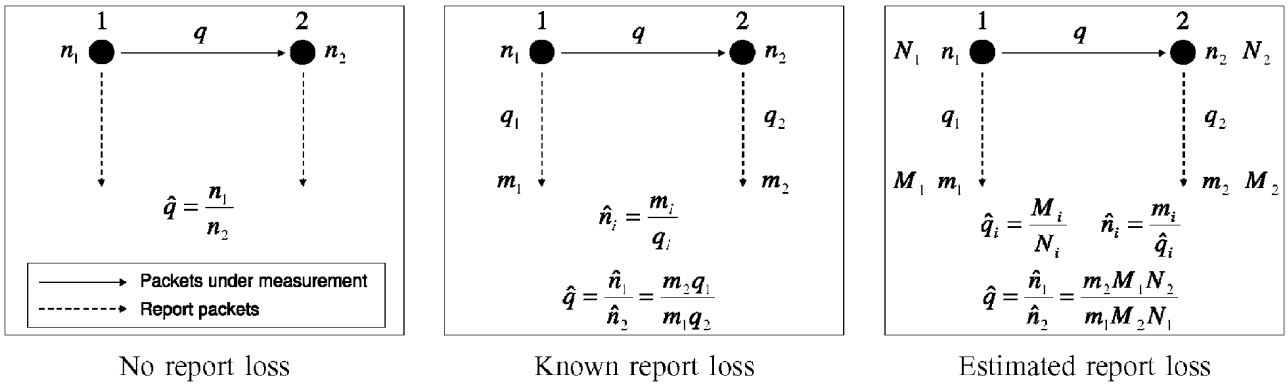

Fig. 7. Network and measurement collection configuration for loss estimation. Left: no report loss. Middle: known report loss. Right: estimated report loss.

set of packet reports at the collector as loss of reports from all links subsequent to e on the path. Instead, we have to distinguish packet and report loss at the statistical level, employing reports from multiple packets. In the following we shall assume that the set of packet reports has already undergone duplicate elimination as described in Section III.

B. Estimating Packet Loss With Known Report Loss

The configurations for loss estimation are shown in Fig. 7. We wish to estimate the packet loss rate on a path $1 \rightarrow 2$. Packet reports are collected from both nodes. We make no assumptions on the collection paths: they may have subpaths in common, and may encompass the paths $1 \rightarrow 2$ or $2 \rightarrow 1$.

Consider some number n of packets in a class of interest. Define an indicator variable $x_i^{(k)}$, taking the value 1 if packet k is present at node i, and 0 otherwise. The indicator variable $y_i^{(k)}$ takes the value 1 if a report on packet k is received at the collector from node i, and 0 otherwise. Thus, $m_i = \sum_k y_i^{(k)}$ is the number of reports reaching the collector from node i. Let q_i be the transmission rate of reports from node i to the collector. In a stochastic model, q_i then becomes the conditional probability for a report to reach the collector from node i, given that the underlying packet is sampled at i.

Let there be n_i sampled packets in the class of interest present at node i. In the special case of lossless reporting, the transmission rate q is estimated as $\hat{q} = n_2/n_1$. See Fig. 7 (left). Suppose, more generally, the transmission rates q_i for the packet reports were known; see Fig. 7 (center). Then we estimate the numbers n_i by $\hat{n}_i = m_i/q_i$ and the transmission rate q on the link $1 \rightarrow 2$ by

$$\hat{q} = \frac{n_2}{n_1} = \frac{m_2 q_1}{m_1 q_2} \tag{16}$$

This estimator is consistent for a stationary loss process, provided the law of large numbers holds for the numbers of packet and reports transmitted. This holds, for example, if the sequence of random variables $(x_1^{(k)}, y_i^{(k)})$ labeled by packet k sampled at node 1, forms a stationary and ergodic process. As $n_1 \rightarrow \infty$, then $n_2/n_1 \rightarrow q, m_i/n_i \rightarrow q_i$, and hence

$$\hat{q} = \frac{m_2}{n_2}\frac{n_2}{n_1}\frac{n_1}{m_1}\frac{q_1}{q_2} \rightarrow q. \tag{17}$$

The numbers of packets n_i in the class that are sampled at nodes i do not enter explicitly into the estimator $\hat{q}$.

C. Estimating Packet Loss With Unknown Report Loss

If the q_i are not known, they may be estimated from the streams of packet reports; see Fig. 7 (right). For example, assume that reports are transmitted individually to the collector, and that they carry transmission sequence numbers. Suppose M_i successive trajectory samples reach the collector from node i in a given period. We assume that the collector performs any reordering of samples with respect to transmission sequence number, if required. Adding 1 to the difference between the transmission sequence numbers of the first and last of these packets yields N_i, the number of trajectory samples transmitted between transmission of the first and last received samples. The transmission rate for trajectory samples from node i is estimated by $\hat{q}_i = M_i/N_i$.

The statistics of the packet process may potentially influence the transmission rate of reports. For example, burstiness in the packet stream of a traffic class can be inherited to some extent by the sampled packet stream. However, we expect sampling rates to be quite small, hence "taming" the burstiness by spacing out packets in the sampled stream, as compared with the original stream. Beyond this, there is no reason to assume that packet selection and transmission of packet reports will be coupled with packet content. For these reasons we assume that the transmission rate of packet reports for the class under study is the same as for all traffic. Thus, we are free to employ transmission sequence numbers applied to packet reports from the whole packet stream, rather than those from the traffic class under study. Thus, we estimate the transmission rate q by replacing q_i with $\hat{q}_i$ in (16):

$$\hat{q} = \frac{m_2 M_1 N_2}{m_1 N_1 M_2} \tag{18}$$

D. Loss Estimation Variance

Correlation between loss of reports does not affect estimator consistency, but it can affect estimator variance. We consider two mechanisms for correlation: spatial correlation of loss of different reports on the same packet, and temporal correlations through the common loss of multiple packet reports exported together in the same packet.

Now, correlation between loss reports is manifest as non-zero conditional covariance between $y_1^{(k)}$ and $y_2^{(k)}$, conditional on packet k having being sampled. But since terms $y_1^{(k)}$ appear in

the denominator, while terms $y_2^{(k)}$ appear in the numerator, positive correlations between loss reports actually reduce estimator variance. To show this, we compute the variance of the estimator (18), asymptotically for a large number of samples. For simplicity we ignore the variability in the numbers M_i of reports and packet N_i in all classes. This is a reasonable approach if the traffic under study forms a small proportion of the total, and is equivalent to treating the ratios M_i/N_i as fixed numbers q_i as in (16).

We analyze the asymptotic variance of $\hat{q}$ from (16), as $n \to \infty$, using the Delta method [12]. This derives the asymptotic behavior of functions of sums of random variables that obey the central limit theorem, as we now summarize:

Lemma 1: Let Z and Z_n be any vector-valued random variables such that $\sqrt{n}(Z_n - Z)$ has asymptotically, as $n \to \infty$, a multivariate Gaussian distribution with mean zero and covariance matrix C. Then for any real function f of the random variables, differentiable at Z, $\sqrt{n}(f(Z_n) - f(Z))$ has asymptotically, as $n \to \infty$, a multivariate Gaussian distribution with mean zero and covariance $c = v \cdot Cv$ where $v = \nabla f(Z)$ is the gradient of f at Z.

We model TS as selecting trajectories independently with probability p, and apply the Delta method using

$$Z_n = n^{-1}\left(\sum_k z^{(k)} y_1^{(k)}, \sum_k z^{(k)} y_2^{(k)}\right) \quad (19)$$

$$Z = \lim_{n\to\infty} \mathsf{E} Z_n = (pq_1, pqq_2) \quad (20)$$

$$f(m_1, m_2) = m_2/m_1 \cdot q_1/q_2 \quad (21)$$

One finds $v = (-q/(pq_1), 1/(pq_2),)$, and C is the covariance matrix of (zy_1, zy_2) namely,

$$C = \begin{pmatrix} pq_1(1-pq_1) & pd_{12} + p(1-p)qq_1q_2 \\ pd_{12} + p(1-p)qq_1q & pqq_2(1-pqq_2) \end{pmatrix} \quad (22)$$

where d_{12} is the covariance of $y_1^{(k)}$ and $y_2^{(k)}$, conditional on packet k having been sampled. Summarizing:

Theorem 4: The distribution of $\sqrt{n}(\hat{q} - q)$ converges as $n \to \infty$ to a Gaussian random variable with mean 0 and variance $c = v \cdot Cv$, equal to

$$c^{\mathrm{TS}} = \frac{q(q_1(1-qq_2) + qq_2(1-q) - 2d_{12})}{pq_1q_2}. \quad (23)$$

This establishes the earlier claim that positive correlation between transmission of reports reduced estimator variance. Conversely, negative correlation increases estimator variance. Negative correlation may occur if the reports from different routers compete for transmission resources along a common path to the collector.

Correlations between different probes do not qualitatively change the results. Gaussian asymptotics still prevail, although the asymptotic covariances are in general different. For a given mode, e.g., the $(x^{(k)}, y^{(k)})$ form a Markov process, the covariances can be calculated using generalizations of the Central Limit Theorem for dependent variables.

Although we have estimated the loss rate on a single path, one could perform joint estimation of the loss rates on a number of (possibly) intersecting paths. It can be shown that the loss rate estimators remain consistent under the previous assumptions. Variance is calculated using a higher dimensional analog of the matrix (22), whose elements take into account potential correlations of loss between export from different routers, e.g., due to intersecting export paths.

Finally, we mention the effect of routers aggregating reports into the same packet for export to the collector. Let us take a simple (and probably worst) case that aggregates s packet reports together, and that boundaries between export packets are aligned to carry matching reports. In this case one finds that all variances are multiplied by a factor s. In practice, we expect correlations to be far weaker, because reports on packets of a given flow under study are interspersed with those from background traffic, which varies between routers. This reduces the likelihood that multiple reports on a given flow are lost in the same report, and also acts to randomize the group of packet reports for export.

E. Comparison With Independent Sampling

We now isolate the difference in estimator variance due to the sampling method. For reference, consider first TS with the simplifying assumption of no report loss: $q_i = 1, d_{12} = 0$, Then c^{TS} reduces to

$$c^{\mathrm{TS}} = q(1-q)/p. \quad (24)$$

For IS, it can be shown that the only change to the calculation behind Theorem 4 is to set the diagonal terms in (22) to 0, since independent selection renders report transmission independent under the assumption of no report loss. This yields asymptotic variance

$$c^{\mathrm{IS}} = q(1+q-2pq)/p. \quad (25)$$

Both variances are inversely proportional to p: fewer samples mean higher variance.

One sees easily that $c^{\mathrm{IS}} \geq c^{\mathrm{TS}}$, with equality only when $p = 1$ or $q = 0$. The expected physical regime is small sampling probability p and small report loss probability $1 - q$. In this regime, the expressions for variance simplify:

$$c^{\mathrm{TS}} \approx (1-q)/p, \quad \text{while} \quad c^{\mathrm{IS}} \approx 2/p. \quad (26)$$

The notable property is that the ratio of the variances of the two estimators is driven by the loss rate to be estimated, with $c^{\mathrm{IS}}/c^{\mathrm{TS}} \approx 2/(1-q) = 50$ when estimating a 1% loss.

F. Loss Estimation Under Load Balancing

The techniques of this section apply to estimation of loss on a point-to-point path. In practice, load balancing may give rise to point-to-multipoint paths. The above technique can be applied provided the problem can be reduced to estimation on a set of point-to-point paths. This is possible if TS is employed on both inbound and outbound interfaces at nodes in which load balancing takes place.

VII. Conclusion

In a network measurement system such as the one described in this paper, there exists a tradeoff between the complexity of the measurement devices and the complexity of the central collector. In our previous work, we assumed that measurement devices are capable of reliably exporting measurements to the collector, which simplifies the task of the collector in reconstructing a statistically representative set of trajectory samples. However, there are circumstances where such reliable export is either not desirable or not possible. Therefore, in this paper, we assume that measurement devices export measurement report packets unreliably, which relieves them of the burden of buffering and processing acknowledgments. But dealing with missing reports complicates the task of the collector. In this paper, we propose methods for the collector to deal with such loss.

The first aspect of trajectory reconstruction that is complicated by report loss is duplicate elimination, i.e., elimination of reports that map to identical labels. We wish to eliminate such duplicates in order to ensure that the final set of trajectories is not polluted by composite trajectories resulting from multiple packets. This may have various undesirable side effects when estimating quantities of interest from these trajectories and monitoring the correct network behavior.

With reliable reporting, a straightforward approach to eliminate duplicate labels is to rely on auxiliary information reported about sampled packets from ingress links, or to assume that a given packet should not be observed on ingress links more than once. With unreliable reporting, this approach may not catch all duplicates, if ingress reports are lost.

We have proposed an approach based on Bloom filters, a data structure that compresses a set membership function into a bit array. Bloom filters are appropriate because the only error they incur are false positives, which may lead to the elimination of some unique labels in addition to actual duplicates. While this represents a small loss of measurement data, the main property of this approach, given in Theorem 1, is that the duplicate elimination essentially behaves like subsampling the original set of packets. Therefore, by applying a correction factor, the resulting set of trajectories can be treated as if it had been obtained in a collision-free way. This insulates the estimation and detection procedures fed by trajectory samples from the intricacies of duplicate elimination.

Once duplicate labels have been eliminated, the resulting report stream can be passed to applications. In general, applications must be adapted to report loss. Path tracing applications must amalgamate reports from several packets in order to reconstruct complete trajectories. Passive loss measurement applications must distinguish report loss from packet loss by exploiting transmission sequence numbers in the reports to estimate report loss rates. The performance analysis of these applications shows that TS brings substantial advantages over independent packet sampling, reducing both estimator variance and reporting bandwidth.

References

[1] B. Bloom, "Space/time tradeoffs in hash coding with allowable errors," *CACM*, vol. 13, no. 7, pp. 422–426, 1970.

[2] A. Broder and M. Mitzenmacher, "Network applications of bloom filters: A survey," in *Proc. Allerton Conf.*, Monticello, IL, 2002.

[3] N. G. Duffield, (Ed.), "A framework for packet selection and reporting," work in progress, Jun. 2007 [Online]. Available: http://www.ietf.org/internet-drafts/draft-psamp-framework-12.txt

[4] N. G. Duffield and M. Grossglauser, "Trajectory sampling for direct traffic observation," *IEEE/ACM Trans. Networking*, vol. 9, no. 3, pp. 280–292, Jun. 2001.

[5] N. G. Duffield, A. Gerber, and M. Grossglauser, "Trajectory engine: A backend for trajectory sampling," in *Proc. Network Operations and Management Symp. (NOMS)*, Florence, Italy, Apr. 2002.

[6] N. G. Duffield and C. Lund, "Predicting resource usage and estimation accuracy in an IP flow measurement collection infrastructure," in *Proc. ACM SIGCOMM Internet Measurement Conf. 2003*, Miami Beach, Fl, Oct. 2003, pp. 27–29.

[7] IETF Packet Sampling Working Group Charter. IETF [Online]. Available: http://www.ietf.org/html.charters/psamp-charter.html

[8] M. Molina, S. Niccolini, and N. G. Duffield, "A comparative experimental study of hash functions applied to packet sampling," in *Proc. ITC-19*, Beijing, China, 2005.

[9] Packet Wingspan Distribution. NLANR [Online]. Available: http://www.nlanr.net/NA/Learn/wingspan.html

[10] P. Phaal, S. Panchen, and N. McKee, "InMon Corporation's sFlow: A method for monitoring traffic in switched and routed networks," RFC 3176, Sep. 2001.

[11] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical network support for IP traceback," in *Proc. ACM SIGCOMM 2000*, Aug. 2000, pp. 295–306.

[12] M. J. Schervish, *Theory of Statistics*. New York: Springer, 1995.

[13] T. Zseby, "Deployment of sampling methods for SLA validation with non-intrusive measurements," in *Proc. Passive and Active Measurement Workshop (PAM 2002)*, Fort Collins, CO, Mar. 2002.

[14] T. Zseby *et al.*, "Sampling and filtering techniques for IP packet selection," IETF, work in progress, Jun. 2007 [Online]. Available: http://www.ietf.org/internet-drafts/draft-ietf-psamp-sample-tech-10.txt

Nick Duffield (M'97–SM'01–F'05) received the Ph.D. degree from the University of London, London, U.K., in 1987.

He is a Distinguished Member of Technical Staff and an AT&T Fellow in the Network and Internet Systems Research Center at AT&T Labs–Research, Florham Park, NJ, where he has been since 1995. He previously held postdoctoral and faculty positions in Dublin, Ireland, and Heidelberg, Germany. His current research focuses on measurement and inference of network traffic. He was charter Chair of the IETF working group on Packet Sampling. He is a co-inventor of the Smart Sampling technologies that lie at the heart of AT&T's scalable Traffic Analysis Service.

Matthias Grossglauser (M'95) received the Diplôme d'Ingénieur en Systèmes de Communication from the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, in 1994, the M.Sc. from the Georgia Institute of Technology, Atlanta, in 1994, and the Ph.D. degree from the University of Paris VI, France, in 1998. He did most of his Ph.D. thesis work at INRIA, Sophia Antipolis, France.

From 1998 to 2002, he was with AT&T Labs–Research, Florham Park, NJ. Currently, he is an Assistant Professor at EPFL, Lausanne. His research interests are in mobile and wireless networking, social and complex networks, network traffic measurement and modeling, and resource allocation problems.

Dr. Grossglauser received the 1998 Cor Baayen Award from the European Research Consortium for Informatics and Mathematics (ERCIM) and the IEEE INFOCOM 2001 Best Paper Award.