Modeling Residual-Geometric Flow Sampling

Xiaoming Wang, Xiaoyong Li, and Dmitri Loguinov

***Abstract*—Traffic monitoring and estimation of flow parameters in high speed routers have recently become challenging as the Internet grew in both scale and complexity. In this paper, we focus on a family of flow-size estimation algorithms we call *Residual-Geometric Sampling* (RGS), which generates a random point within each flow according to a geometric random variable and records all remaining packets in a flow counter. Our analytical investigation shows that previous estimation algorithms based on this method exhibit certain bias in recovering flow statistics from the sampled measurements. To address this problem, we derive a novel set of unbiased estimators for RGS, validate them using real Internet traces, and show that they provide an accurate and scalable solution to Internet traffic monitoring.**

I. Introduction

Recent growth of the Internet in both scale and complexity has imposed a number of challenges on network management, operation, and traffic monitoring. The main problem in this line of work is to scale measurement algorithms to achieve certain objectives (e.g., accuracy) while satisfying real-time resource constraints (e.g., fixed memory consumption and per-packet processing delay) of high-speed Internet routers. This is commonly accomplished (e.g., [5], [6], [7], [8], [9], [10], [11], [14], [15], [17], [21], [18], [19], [20], [22], [26], [32]) by reducing the amount of information a router has to store in its internal tables, which comes at the expense of deploying special estimation techniques that can recover metrics of interest from the collected samples.

In this paper, we study two problems in the general area of *measuring flow sizes* – 1) determining the number of packets transmitted by "elephant" flows [11], [15], [17], [21], [20], [22] and 2) building the distribution of flow sizes seen by the router in some time window [7], [18], [32] – coupled in a single measurement technique. The former problem arises in usage-based accounting and traffic engineering [6], [11], [12], [13], [27], while the latter has many security applications such as anomaly and intrusion detection [1], [23], [16].

Our interest falls within the family of *residual sampling*, which selects a random point A within each flow and then samples the remainder R of that flow until it ends. Denoting by L the size (in packets) of a random flow, sampled residuals R are simply $L - A$. Stochastically larger A results in fewer flows being sampled and leads to lower overhead in terms of both CPU and RAM consumption. Besides reduced overhead arising from omission of many small-size flows from counter tables, residual sampling guarantees to capture large flows with probability $1 - o(1)$ as their size $L \to \infty$. This allows ISPs to determine "heavy-hitters" and charge the corresponding customers for generated traffic.

While in P2P networks residual sampling distributes the initial point A uniformly within user lifetimes [31], flow-based estimation [11], [17] usually employs geometric A since it can be easily implemented with a sequence of independent Bernoulli variables. We call the resulting approach *Residual-Geometric Sampling* (RGS) and note that it has received some limited analytical attention in [11], [17]; however, unbiased estimation of individual flow sizes, analysis of the resulting error, asymptotically accurate recovery of flow-size distribution $P(L = i)$ from sampled residuals R, and analysis of space-CPU requirements in steady state have not been explored. We overcome these issues below.

A. Single-Flow Usage

We start with the problem of obtaining sizes of individual flows for accounting purposes. Since residual sampling requires an estimator to convert residuals into the metrics of interest, our first task is to define proper notation and desired properties for the estimation algorithm. Assume that for a flow of size L the sampling algorithm produces residual R_L, where both L and R_L are random variables. We call an estimator $e(R_L)$ *unbiased* if its expectation produces the correct flow size, i.e., $E[e(R_L)|L = l] = E[e(R_l)] = l$. Unbiased estimation allows one to average the estimated size of several flows of a given size l and accurately estimate their total contribution. We further call an estimator *elephant-accurate* if ratio $e(R_l)/l$ converges to 1 in mean-square as $l \to \infty$. Elephant-accuracy ensures that the variance of $e(R_l)/l$ tends to zero as $l \to \infty$, which means that the amount of relative error between $e(R_l)$ and l becomes negligible for large flows.

Prior work on RGS [11], [17] has suggested the following estimator:

$$e(R_L) = R_L - 1 + 1/p, \tag{1}$$

where $0 < p \leq 1$ is the parameter of geometric variable A. To understand the performance of (1), we first build a general probabilistic model for residual-geometric sampling and derive the relationship between flow size L and its residual R_L. Using this result, we prove that:

$$E[e(R_l)] = \frac{l}{1 - (1-p)^l}, \tag{2}$$

which indicates that (2) is generally biased and on average tends to overestimate the original flow size by a factor of up to $1/p$. To address this problem, we derive a different estimator:

$$\hat{e}(R_L) = R_L - 1 + 1/p - \frac{(1-p)^{R_L}}{p} \tag{3}$$

and prove that it is both unbiased and elephant-accurate. We also derive in closed-form the mean-square error $\delta_l =$

A shorter version of this paper appeared in IEEE INFOCOM 2011.

Xiaoming Wang is with Amazon.com, Seattle, WA 98101 USA (xmwang@gmail.com).

Xiaoyong Li and Dmitri Loguinov are with Texas A&M University, College Station, TX 77843 USA ({xiaoyong, dmitri}@cse.tamu.edu).

$E[(\hat{e}(R_l)/l-1)^2]$ for finite l, which can be used to determine when (3) approximates the true flow size with accuracy sufficient for billing purposes.

B. Flow-Size Distribution

Our second problem is estimation of the original flow-size probability mass function (PMF), which we assume is given by $f_i = P(L = i), i = 1, 2, \ldots$ We call PMF estimator q_i *asymptotically unbiased* if it converges in probability to f_i for all i as the number of sampled flows $M \to \infty$. One may be at first tempted to compute this distribution based on the values produced by either (1) or (3) for each observed flow; however, we show that such q_i almost always differ from the original distribution f_i and the bias persists as sample size $M \to \infty$. The reason for this discrepancy is that $e(.)$ and $\hat{e}(.)$ both estimate the sizes of flows *that have been sampled* by the algorithm, which are not representative of the entire population passing through the router. Since longer flows are more likely to be selected by residual sampling, this approach severely overestimates their fraction and thus skews the PMF towards the tail.

Denote by M_i the number of sampled flows with $R_L = i$ and define a new estimator:

$$\tilde{q}_i = \frac{M_i - (1-p)M_{i+1}}{Mp + (1-p)M_1}. \tag{4}$$

Using the general model of RGS derived later in the paper, we prove that $\tilde{q}_i$ tends to f_i in probability as $M = \sum_i M_i \to \infty$ and obtain the amount of error $|\tilde{q}_i - f_i|$ for finite M. We also provide asymptotically unbiased estimators for the total number of flows n:

$$\tilde{n} = M + \frac{1-p}{p} M_1 \tag{5}$$

and the number of flows n_i with exactly i packets:

$$\tilde{n}_i = \frac{M_i - (1-p)M_{i+1}}{p}, \tag{6}$$

where $\tilde{n}/n \to 1$ and $\tilde{n}_i/n_i \to 1$, both in probability, as $M \to \infty$. We call the resulting combination (3),(4)-(6) *Unbiased Residual-Geometric Estimators* (URGE).

C. Implementation and Evaluation

We finish the paper by discussing an efficient implementation of the above algorithms and evaluating their accuracy/performance using several Internet traces. Prior work has not discussed how residual sampling should be implemented or its overhead in steady-state, which prompts a fairly detailed exposition below.

We assume URGE uses a chain-linked hash table of size K, which keeps individual flow counters. Each linked list is sorted according to the flow ID and is traversed linearly until a match is found or an ID larger than the one being sought is encountered. Keeping the list sorted (as opposed to FIFO) reduces the lookup delay by half for flows not already in the table. To reduce RAM overhead, we remove flows from the table if they have completed (i.e., FIN, RST packets detected) or if no packets from these flows arrive within some timeout τ. To keep the overhead manageable, the removal process is run over the entire table on the timescale of seconds or even minutes.

As before, assume that the router sees a total of n flows in window $[0, T]$. Then, denote by $N(t)$ the number of *active* flows at time t and by $M(t)$ the number of them sampled by the router. It then follows that memory consumption $W_R(t)$ and lookup delay is $T_R(t)$ are both functions of $M(t)$. Under certain mild assumptions, we obtain a simple result on $E[M(t)]$ and show that even as the total number of flows $n \to \infty$, both RAM usage and CPU overhead of RGS remain constant.

We then explore how to satisfy the tradeoff between three design objectives – memory consumption, processing speed, and accuracy – using parameters K and p. Given upper bounds on memory usage W_0 and per-packet processing delay T_0, we propose a technique for deciding K based on the above analysis such that $W_R(t) \leq W_0$ and $T_R(t) \leq T_0$ are satisfied, while maximizing p at the same time (i.e., achieving the best accuracy within the constraints).

We finish the paper by evaluating URGE with real Internet traces obtained from NLANR [24] and CAIDA [3]. Our experiments reveal that the proposed algorithm produces very accurate estimation of flow metrics and thus allows one to perform more aggressive sampling (i.e., smaller probability p) of the monitored traffic. With $p = 0.01$, we find that $E[M(t)]$ is $40 - 4000$ times smaller than n and $3 - 100$ times smaller than $E[M]$, with most lookups requiring just 1-2 RAM hits. We also discover in the experiments with small traces that URGE does not degrade significantly in terms of accuracy even for small sample sizes, which makes it suitable for monitoring individual customer networks and certain protocols.

The remainder of the paper is organized as follows. We review prior work on traffic monitoring in Section II. We then develop a probabilistic model for residual-geometric sampling in Section III, analyze previous methods in Section IV, and propose the new estimators in Section V. We explore the implementation of the suggested framework in Section VI, evaluate its performance in Section VII, and conclude the paper in Section VIII.

II. Related Work

In this section, we review several sampling algorithms in the area of traffic monitoring. In particular, we classify existing work into two categories: *packet sampling* and *flow sampling*, where the former makes per-packet and the latter per-flow decisions to sample incoming traffic.

A. Packet Sampling

Sampled NetFlow (SNF) [26] is a widely used technique in which incoming packets are sampled with a fixed probability p. The general goal of SNF is to obtain the PMF of flow sizes; however, [14] shows that it is impossible to accurately recover the original flow-size distribution from sampled SNF data. Estan *et al.* [10] propose *Adaptive NetFlow* (ANF), which adjusts the sampling probability p according to the size of

the flow table; however, ANF's bias in the sampled data is equivalent to that in SNF and is similarly difficult to overcome in practice.

Instead of using one uniform probability for all flows as in [10], [26], another direction in packet sampling is to compute $p_i(c)$ for each flow i based on its currently observed size c. This approach has been studied by two independent papers, *Sketch-Guided Sampling* (SGS) [20] and *Adaptive Non-Linear Sampling* (ANLS) [15]. A common feature of these two methods is to sample a new flow with probability 1 and then monotonically decrease $p_i(c)$ as c grows. Both methods must maintain a counter for each flow present in the network and are difficult to scale due to the high RAM/CPU usage.

B. Flow Sampling

In *flow thinning* [14], each flow is sampled independently with probability p and then all packets in sampled flows are counted. Hohn *et al.* [14] show that flow thinning is able to accurately estimate the flow size distribution; however, this method typically misses $1-p$ percent of elephant flows and thus does not support applications such as usage-based accounting and traffic engineering [6], [11], [12], [13], [27]. For highly skewed distributions with a few extremely large flows and many short ones (which is typical for Internet links), this method may also take a long time to converge.

To address these problems of flow thinning, Estan *et. al.* [11] introduce a size-dependent flow sampling algorithm called *Sample-and-Hold* (S&H), which is proposed to identify elephant flows. For each packet from a new flow, the algorithm creates a flow counter with probability p; once a flow is sampled, all of its subsequent packets are then counted. It is easy to verify that S&H samples a flow with size l with probability $1-(1-p)^l$, which quickly approaches 1 as l grows. Creating a unifying analytical model for this approach and understanding the properties of samples it collects is the main topic of this paper.

Another direction of size-dependent flow sampling has been explored by Duffield *et al.* in [5], [6], [8], which present another size-dependent flow measurement method called *Smart Sampling*. Their approach selects each flow of size L with probability $p(L)=\min(1, L/z)$, where z is some constant. Since this method requires flow size L before deciding whether to sample it or not, it can only be applied off-line.

Kompella *et. al.* [17] examine a method called *Flow Slicing* (FS), which combines SNF and S&H with a variant of smart sampling. Other non-sampling methods include *exact counting* [25], [28], [30], [33] and *lossy counting* [18], [22], which are orthogonal to our work.

III. Underlying Model

In this section, we build a general probabilistic model of Sample-and-Hold [11] and establish the necessary analytical foundation for the results that follow.

A. Sample-and-Hold

Consider a sequence of packets traversing a router and assume that its flow-measurement algorithm checks each packet's flow identifier x in some RAM table. If x is found in the table, the corresponding counter is incremented by 1; otherwise, with probability p a new entry for x is created in the table (with counter value 1) and with probability $1-p$ the packet is ignored.

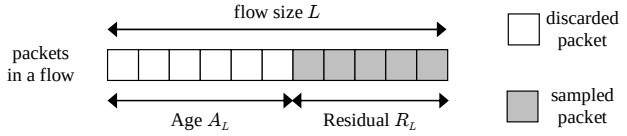


Fig. 1. Residual-geometric sampling of a flow with size L.

To model this process, we first need several definitions. Assume that flow sizes are i.i.d. random variables and define *geometric age* A_L to be the number of packets discarded from the front of a flow with size L before it is sampled (see Fig. 1). Let G be a shifted geometric random variable with success probability p, i.e., $P(G=j)=(1-p)^j p$. It thus follows that A_L is simply:

$$A_L = \min(G, L). \tag{7}$$

Now define *geometric residual* R_L to be the final counter value of a flow of size L conditioned on the fact that it has been sampled (i.e., $A_L < L$):

$$R_L = L - A_L, \tag{8}$$

which is also illustrated in Fig. 1. From the perspective of traffic monitoring in this paper, geometric residual R_L is the only quantity collected during measurement and available to an estimation algorithm. Since this approach belongs to the class of residual-sampling techniques [31] and specifically uses geometric age, this paper calls S&H by a more mathematically-specific name *Residual-Geometric Sampling* (RGS).

Assume that L has a PMF $f_i = P(L=i)$, where $i = 1, 2, \ldots$, and denote by $p_s = P(A_L < L)$ the probability that a random flow is sampled. Then, we have the following result.

Lemma 1: Probability p_s that a flow is selected by RGS is:

$$p_s = E[1-(1-p)^L] = 1 - \sum_{i=1}^{\infty} f_i (1-p)^i. \tag{9}$$

Proof: Observe that for a fixed flow size $L=l$, we have $P(A_l < l) = 1-(1-p)^l$. Unconditioning L, we immediately get (9). ∎

Next, let $h_i = P(R_L = i)$ be the PMF of geometric residual R_L. The following lemma expresses h_i in terms of f_i.

Lemma 2: The PMF of geometric residual R_L is:

$$h_i = \frac{p \sum_{j=i}^{\infty} f_j (1-p)^{j-i}}{p_s}. \tag{10}$$

Proof: Using (8), we have:

$$\begin{aligned} h_i &= P(R_L = i) = P(L - A_L = i | A_L < L) \\ &= \frac{P(L - A_L = i \cap A_L < L)}{p_s}, \end{aligned} \tag{11}$$

where $p_s = P(A_L < L)$. Substituting (7) into (11) and combining the fact that $L - G = i \geq 1$, we establish:

$$h_i = \frac{P(L-G=i)}{p_s} = \frac{\sum_{j=0}^{\infty} P(G=j-i) f_j}{p_s}, \tag{12}$$

which gives the desired result in (10) by substituting the PMF of G into (12). ∎

The result of Lemma 2 is fundamental as most of the results in this paper are conveniently derived from (10).

B. Fixed Flow Size

We next analyze a special case of residual sampling where the original flow size is fixed at $L = l$. Note that residuals are now R_l instead of R_L since the original flow size is no longer a random variable. Recall that the goal of single-flow size estimation is to obtain l from R_l for each sampled flow. The next corollary follows from (10) and gives the distribution and expectation of geometric residual R_l.

Corollary 1: Given flow size $L = l$, the PMF of R_l is:

$$P(R_l = i) = \frac{(1-p)^{l-i}p}{1-(1-p)^l} \tag{13}$$

and its expectation is:

$$E[R_l] = \frac{l}{1-(1-p)^l} + 1 - 1/p. \tag{14}$$

Proof: For $L = l$, we have $f_l = 1$ and $f_i = 0$ for all $i \neq l$. Writing $p_s = 1-(1-p)^l$, we get from (10):

$$P(R_l = i) = \frac{\sum_{j=i}^{\infty} f_j (1-p)^{j-i}p}{1-(1-p)^l} = \frac{(1-p)^{l-i}p}{1-(1-p)^l}, \tag{15}$$

which is exactly (13).

We next derive expectation $E[R_l]$, which can be expanded into:

$$E[R_l] = E[l - A_l | A_l < l] = l - E[G|G < l]. \tag{16}$$

Recall that for any non-negative discrete random variable Y taking values over the integer set $\{0, 1, \ldots\}$, its expectation is given by $E[Y] = \sum_{y=0}^{\infty} P(Y > y)$. It thus follows that (16) reduces to:

$$\begin{aligned} E[R_l] &= l - \sum_{j=0}^{l-1} P(G > j | G < l) \\ &= \sum_{j=0}^{l-1} P(G \leq j | G < l) = \frac{\sum_{j=0}^{l-1} P(G \leq j)}{P(G < l)}. \end{aligned} \tag{17}$$

Substituting $P(G \leq j) = 1-(1-p)^{j+1}$ into (17), we have:

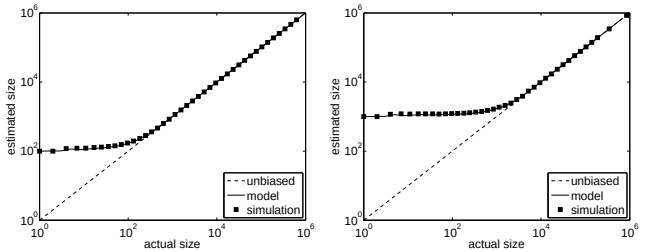
$$\begin{aligned} E[R_l] &= \frac{\sum_{j=0}^{l-1}[1-(1-p)^{j+1}]}{1-(1-p)^l} \\ &= \frac{l - (1-p)(1-(1-p)^l)/p}{1-(1-p)^l}, \end{aligned} \tag{18}$$

which can be simplified to (14). ∎

Next, we apply the results obtained in this section to analyze existing estimation methods that have been proposed for RGS.

IV. Analysis of Existing Methods

In this section, we examine prior approaches [11], [17] to estimating single-flow usage and whether their results can be generalized to recover the PMF of L.



(a) $p = 0.01$ (b) $p = 0.001$

Fig. 2. Expectation of estimator (19) in simulations and its model (20).

A. Single-Flow Usage

To evaluate single-flow estimators, we use the following definition that is commonly used in statistics [2].

Definition 1: Estimator $e(R_l)$ is called *unbiased* if $E[e(R_l)] = l$ for all $l \geq 1$.

Unbiased estimation is a key property of an estimator as it allows accurate estimation of the total contribution from a sufficiently large pool of flows (e.g., one customer network). However, since large flows are typically rare, one commonly faces an additional requirement to estimate their size *with just a single sample* $e(R_l)$, which is formalized in the next definition.

Definition 2: Estimator $e(R_l)$ is called *elephant-accurate* if $e(R_l)/l \to 1$ in mean-square as $l \to \infty$.

Elephant-accuracy guarantees that the amount of relative error between $e(R_l)$ and l decays to zero as $l \to \infty$. As before, suppose that a flow of size l produces a counter with value R_l. Recall that [11], [17] suggest the following estimator:

$$e(R_l) = R_l - 1 + 1/p, \tag{19}$$

where p is the probability of residual-geometric sampling. The next result directly follows from (14).

Theorem 1: Expectation $E[e(R_l)]$ is given by:

$$E[e(R_l)] = \frac{l}{1-(1-p)^l}. \tag{20}$$

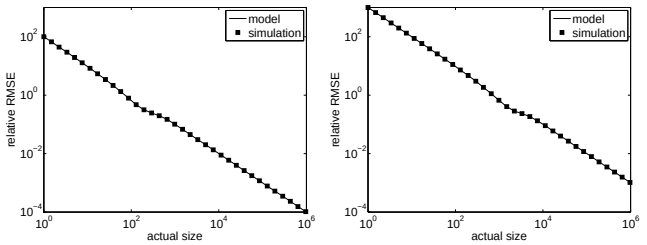
Proof: Taking the expectation of (19), we have:

$$E[e(R_l)] = E[R_l] - 1 + 1/p, \tag{21}$$

which immediately leads to (20) using (14). ∎

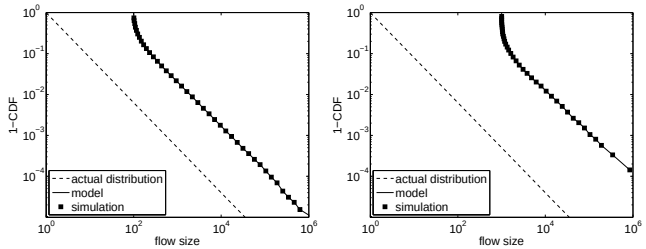
Note that (20) indicates that (19) is generally biased, especially when lp is small. Indeed, for $lp \approx 0$, we have $1-(1-p)^l \approx lp$ and $E[e(R_l)] \approx 1/p$ regardless of l, which shows that in such cases $E[e(R_l)]$ *carries no information about the original flow size*. However, as $l \to \infty$, it is straightforward to verify that the bias in $e(R_l)$ vanishes exponentially, which is consistent with the analysis in [17], which has only considered the case of $l \to \infty$.

To see the extent of bias in (19) and verify (20), we apply residual-geometric sampling to flows of size l ranging from 1 to 10^6 packets, feed the measured sizes to (19), and average the result after 1000 iterations for each l. Fig. 2 plots the obtained $E[e(R_l)]$ along with model (20). The figure indicates that (20) indeed captures the bias and that (19) tends to overestimate the size of short flows *even in expectation*, where smaller sampling probability p leads to more error.



(a) $p = 0.01$ (b) $p = 0.001$

Fig. 3. RRMSE of (19) in simulations and its model (23).



(a) $p = 0.01$ (b) $p = 0.001$

Fig. 4. Distribution $\{q_i\}$ in simulations and its model (24).

To quantify the error of individual values $e(R_l)$ in estimating flow size l and to understand elephant-accuracy, denote by $Y_l = e(R_l)/l$ and define the *Relative Root Mean Square Error* (RRMSE) to be:

$$\delta_l = \sqrt{E[(Y_l - 1)^2]}. \tag{22}$$

Note that $\delta_l \to 0$ indicates that $Y_l \to 1$ in mean-square and thus implies elephant-accurate estimation. The next result derives δ_l in closed form. We omit the rather tedious derivations for brevity.

Theorem 2: The RRMSE of (19) is given by:

$$\delta_l = \sqrt{\frac{1 - p - l(l-1)p^2(1-p)^l - (1-p)^{l+1}}{l^2 p^2 (1 - (1-p)^l)}}. \tag{23}$$

Observe from (23) that for flows with size $l = 1$, the relative error is $\sqrt{1-p}/p$, but as $l \to \infty$, $\delta_l \to 0$ and the estimator is elephant-accurate. Fig. 3 plots (23) against simulations, indicating a close match. The figure also shows that the RRMSE starts from $1/p$ and decreases towards zero as $\Theta(1/l)$ as $l \to \infty$.

B. Flow-Size Distribution

We now investigate whether $e(R_L)$ defined in (19) can be used to estimate the actual flow-size distribution $\{f_i\}_{i=1}^{\infty}$. Denote by $q_i = P(e(R_L) = i)$ the PMF of estimated sizes among the sampled flows. To understand our objectives with approximating the PMF of L, a definition is in order.

Definition 3: An estimator $\{q_i\}_{i=1}^{\infty}$ of PMF $\{f_i\}_{i=1}^{\infty}$ is called *asymptotically unbiased* if q_i converges in probability to f_i for all i as the number of sampled flows $M \to \infty$.

The next theorem follows directly from (10).

Theorem 3: The PMF of flow sizes estimated from (19) is given by:

$$q_i = \frac{\sum_{j=y(i)}^{\infty} f_j (1-p)^{j-y(i)} p}{p_s}, \tag{24}$$

where $y(i) = \lceil i + 1 - 1/p \rceil$ and p_s is in (9).

The result in (24) indicates that each q_i is different from f_i regardless of the sampling duration and thus cannot be used to approximate the flow-size distribution. We verify (24) with a simulated packet stream with 5M flows, where flow sizes follow a power-law distribution $P(L \le i) = 1 - i^{-\alpha}$ for $i = 1, 2, \ldots$ and $\alpha = 1.1$. Fig. 4 plots the CCDF of random variable $e(R_L)$ obtained from simulations as well as model (24), both in comparison to the tail of the actual distribution. The figure shows that (24) accurately predicts the values obtained from simulations and that PMF $\{q_i\}$ is indeed quite different from $\{f_i\}$.

So far, our study of existing methods in residual-geometric sampling has shown that they are not only generally biased, but also unable to recover the flow-size distribution from residuals R_L. This motivates us to seek better estimation approaches, which we perform next.

V. URGE

This section proposes a family of algorithms called *Unbiased Residual-Geometric Estimators* (URGE), proves their accuracy, and verifies them in simulations.

A. Single-Flow Usage

For estimating individual flow sizes, we first consider an estimator directly implied by the result in (14). Notice that solving (14) for l and expressing flow size l in terms of $E[R_l]$, we get:

$$l = u - \frac{1}{\log(1-p)} W\Big(u(1-p)^u \log(1-p)\Big), \tag{25}$$

where $u = E[R_l] + 1/p - 1$ and $W(z)$ is Lambert's function (i.e., a multi-valued solution to $We^W = z$) [4]. Thus, a possible estimator can be computed from (25) with $E[R_l]$ replaced by the measured value of geometric residual R_l.

However, there are two reasons that (25) is a bad estimator of flow sizes. First, Lambert's function $W(z)$ has no closed form solution and has to be numerically solved using tools such as Matlab. Second, it can be verified (not shown here for brevity) that (25) is not an unbiased estimator. Instead, we define a new estimator:

$$\hat{e}(R_l) = R_l - 1 + 1/p - \frac{(1-p)^{R_l}}{p}. \tag{26}$$

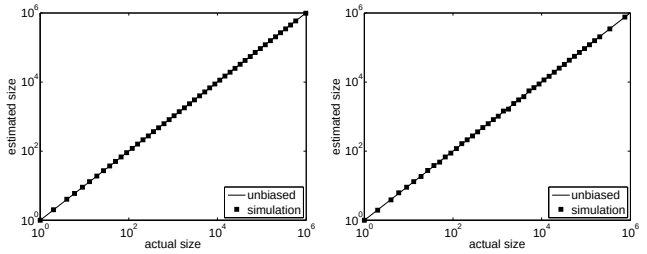
and next show that it is unbiased.

Lemma 3: Estimator $\hat{e}(R_l)$ in (26) is unbiased, i.e.,

$$E[\hat{e}(R_l)] = l. \tag{27}$$

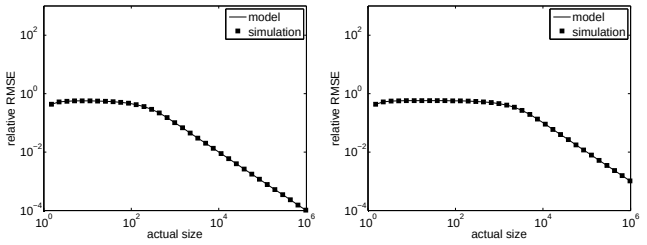
Proof: We prove (27) by deriving such function $\psi(R_l)$ that satisfies $E[\psi(R_l)] = l$. First, it follows from (13) that:

$$\begin{aligned} E[\psi(R_l)] &= \sum_{j=1}^{l} \psi(j) P(R_l = j) \\ &= \frac{\sum_{j=1}^{l} \psi(j)(1-p)^{l-j} p}{1 - (1-p)^l}. \end{aligned} \tag{28}$$



(a) $p = 0.01$ (b) $p = 0.001$

Fig. 5. Expectation of estimator (26) in simulations.



(a) $p = 0.01$ (b) $p = 0.001$

Fig. 6. RRMSE of (26) in simulations and model (31).

For $E[\psi(R_l)] = l$ to hold, we must have:

$$\sum_{j=1}^{l} \psi(j)(1-p)^{-j} = \frac{l\left(1-(1-p)^l\right)}{p(1-p)^l}. \tag{29}$$

Writing (29) twice for l and $l-1$ and subtracting the two equations from each other, we get:

$$\psi(l)(1-p)^{-l} = \frac{1+p(l-1)-(1-p)^l}{p(1-p)^l}. \tag{30}$$

Simplifying (30), we obtain (26). ■

We plot in Fig. 5 simulation results obtained from (26). The figure indicates that $\hat{e}(R_l)$ accurately estimates actual sizes for all flows in both cases of p. Next, we derive the RRMSE of URGE.

Theorem 4: The RRMSE of (26) is given by:

$$\hat{\delta}_l = \sqrt{\frac{1-p+lp(p-2)(1-p)^l-(1-p)^{2l+1}}{l^2p^2(1-(1-p)^l)}}. \tag{31}$$

It is easy to verify from (31) that URGE has zero RRMSE for $l = 1$ or $l \to \infty$, confirming its elephant-accuracy. We plot $\hat{\delta}_l$ obtained from simulations along with the model in Fig. 6, which shows that (31) accurately tracks the actual relative error. From Figures 5-6, it is clear that $\hat{e}(R_l)$ significantly improves the accuracy of estimating small flow sizes compared to $e(R_l)$. In practice, (31) can be used to determine threshold l_0, which leads to desired bounds on error for all $l \geq l_0$ and allows ISPs to use $e(R_l)$ instead of l.

B. Flow-Size Distribution

It is worth mentioning that while (26) produces unbiased estimation of flow sizes, $\hat{e}(R_L)$ is not suitable for computing the flow-size distribution, as we show below. Denote by $\hat{q}_i = P(\hat{e}(R_L) = i)$ the PMF of $\hat{e}(R_L)$. Then, we have the following result.

Lemma 4: PMF of $\hat{e}(R_L)$ is given by:

$$\hat{q}_i = \frac{1}{p_s} \sum_{j=y(i)}^{\infty} (1-p)^{j-y(i)} f_j p, \tag{32}$$

where p_s is in (9), function $y(i)$ is:

$$y(i) = \lceil i + 1 - 1/p - \omega \rceil, \tag{33}$$

and $\omega = W\left(-(1-p)^{i+1-1/p}\log(1-p)\right)$.

Proof: We first solve

$$R_L + 1/p - 1 - (1-p)^{R_L}/p = i, \tag{34}$$

for R_L and express it in terms of i, i.e., $R_L = y(i)$, where $y(i)$ is given by (33), ignoring approximate round-offs to the nearest integer. Combining with (10), we have:

$$\hat{q}_i = P(R_L = y(i)) = h_{y(i)}, \tag{35}$$

where h_i is in (10). This directly leads to (32). ■

Notice from (32)-(33) that distribution $\hat{q}_i$ does not even remotely approximate the original PMF f_i. This problem is fundamental since residual sampling exhibits bias towards larger flows and even if we could recover L from R_L exactly, the distribution of sampled flow sizes would not accurately approximate that of all flows passing through the router.

We thus explore another technique for estimating the flow-size distribution. Before doing that, we need the next lemma.

Lemma 5: The flow size distribution f_i can be expressed using the PMF of geometric residuals $\{h_i\}$ in (10) as:

$$f_i = \frac{h_i - (1-p)h_{i+1}}{p + (1-p)h_1}. \tag{36}$$

Proof: From (10), we obtain that:

$$h_i - (1-p)h_{i+1} = \frac{p}{p_s} f_i. \tag{37}$$

It then immediately follows that f_i is given by:

$$f_i = \frac{p_s(h_i - (1-p)h_{i+1})}{p}. \tag{38}$$

Notice that p_s in (9) is a function of $\{f_i\}$, which are unknown from the measurement perspective. The last step of the proof is to express p_s in terms of known quantities $\{h_i\}$, which can be accomplished by applying the normalization condition $\sum_{i=1}^{\infty} f_i = 1$. It is easy to verify that:

$$\sum_{i=1}^{\infty} h_i = 1 \quad \text{and} \quad \sum_{i=1}^{\infty} h_{i+1} = 1 - h_1. \tag{39}$$

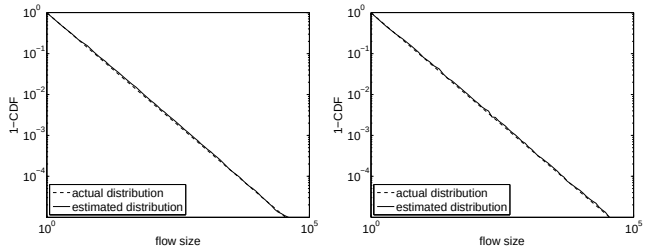
Then, summing up both sides of (38) for i from 1 to infinity gives us:

$$p_s = \frac{p}{\sum_{i=1}^{\infty}(h_i - (1-p)h_{i+1})} = \frac{p}{p + (1-p)h_1}, \tag{40}$$

which together with (38) establishes (36). ■

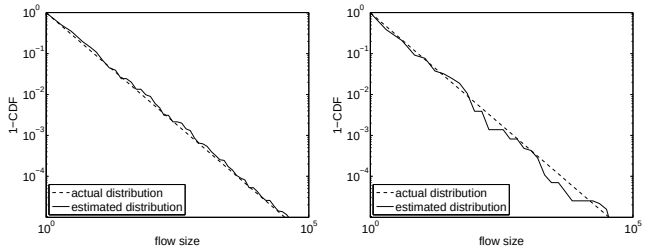
This result leads to a new estimator for the flow-size distribution:

$$\tilde{q}_i = \frac{M_i - (1-p)M_{i+1}}{Mp + (1-p)M_1}, \tag{41}$$



(a) $p = 0.01$, $M = 194,208$ (b) $p = 0.001$, $M = 26,233$

Fig. 7. Estimator (41) in simulations.



(a) $p = 10^{-4}$, $M = 3,090$ (b) $p = 10^{-5}$, $M = 337$

Fig. 8. Estimator (41) in simulations with very small p.

where M is the total number of sampled flows and M_i is the number of them with the geometric residual equal to i. Since $M_i/M \to h_i$ in probability as $M \to \infty$ (from the weak law of large numbers), we immediately get the following result.

Corollary 2: The estimator in (41) is asymptotically unbiased.

We next verify the accuracy of $\tilde{q}_i$ in simulations with 5M flows in the same setting as in the previous section. We plot in Fig. 7 the CCDF estimated from (41) along with the actual distribution. The figure shows that $\tilde{q}_i$ accurately follows the actual distribution for both cases of p.

C. Convergence Speed

We next examine the effect of sample size M on the convergence of estimator $\tilde{q}_i$. To illustrate the problems arising from small M, we study (41) with $p = 10^{-4}$ and 10^{-5} in simulations with the same 5M flows. The estimator obtained $M = 3,090$ flows for $p = 10^{-4}$ and just $M = 337$ for $p = 10^{-5}$. Fig. 8 indicates that while the estimated curves under both choices of p still approximate the trend of the original distribution, they exhibit different levels of noise. As the next result indicates, small p leads to a small sample size M and thus more noise in the estimated values.

Corollary 3: Suppose that M flows are selected by residual-geometric sampling from a total of n flows. Then, the expected value of M is given by:

$$E[M] = np_s = nE[1 - (1-p)^L]. \tag{42}$$

Proof: This result follows from the fact that $E[M] = nP(A_L < L) = np_s$, where p_s is given by (9). ∎

To shed light on the choice of proper p for RGS, we show how to determine the minimum M that would guarantee a certain level of accuracy in $\tilde{q}_i$. Define $\tilde{h}_i = M_i/M$ to be an estimate of $h_i = P(R_L = i)$. The next lemma follows from Lemma 5 and Corollary 2 and indicates that the accuracy of $\tilde{q}_i$ directly depends on whether $\tilde{h}_i$ approximates h_i accurately.

Lemma 6: Suppose that $|\tilde{h}_j - h_j| \le \eta h_j$ holds with probability $1-\xi$ for $j \in [1, i+1]$, where η and ξ are small constants. Then, there exists a constant ζ:

$$\zeta = \frac{\eta(p + 2\eta(1-p)h_1)}{p + (1-p)(1-\eta)h_1} \tag{43}$$

such that $\zeta \to 0$ as $\eta \to 0$ and $P(|\tilde{q}_i - f_i| \le \zeta f_i) = 1 - \xi$.

Proof: We prove the result by deriving ζ that satisfies $|\tilde{q}_i - f_i| \le \zeta f_i$ given that $|\tilde{h}_j - h_j| \le \eta h_j$. From (36) and (41), we have:

$$|\tilde{q}_i - f_i| = \frac{|a_1|}{a_2}, \tag{44}$$

where

$$a_1 = p(\tilde{h}_i - h_i) + p(1-p)(\tilde{h}_{i+1} - h_{i+1}) + (1-p)\times (h_1\tilde{h}_i - \tilde{h}_1 h_i) + (1-p)^2(h_1\tilde{h}_{i+1} - \tilde{h}_1 h_{i+1}) \tag{45}$$

and

$$a_2 = (p + (1-p)h_1)(p + (1-p)\tilde{h}_1). \tag{46}$$

From the condition $|\tilde{h}_j - h_j| \le \eta h_j$, we bound $|a_1|$ and a_2 as follows:

$$\begin{aligned} |a_1| &\le \eta p h_i + 2\eta(1-p)h_1 h_i + \eta p(1-p)h_{i+1} \\ &\quad + 2\eta(1-p)^2 h_1 h_{i+1} \\ &= \eta(h_i + (1-p)h_{i+1})(p + 2\eta(1-p)h_1), \end{aligned} \tag{47}$$

and

$$a_2 \ge (p + (1-p)h_1)(p + (1-p)(1-\eta)h_1). \tag{48}$$

It thus follows from (36) and (47)-(48) that $|\tilde{q}_i - f_i| \le \zeta f_i$, where constant ζ is given by:

$$\zeta = \frac{\eta(p + 2\eta(1-p)h_1)}{p + (1-p)(1-\eta)h_1}, \tag{49}$$

and that $\zeta \to 0$ as $\eta \to 0$. ∎

Next, we obtain a bound on M from the requirement that $\tilde{h}_i$ be bounded in probability within a given range $[h_i(1-\eta), h_i(1+\eta)]$.

Theorem 5: For small constants η and ξ, $|\tilde{h}_i - h_i| \le \eta h_i$ holds with probability $1 - \xi$ if sample size M is no less than:

$$M \ge \frac{(1-h_i)}{h_i \eta^2} \left(\Phi^{-1}(1 - \xi/2)\right)^2, \tag{50}$$

where $\Phi(x)$ is the CDF of the standard Gaussian distribution $\mathcal{N}(0,1)$.

Proof: Notice that M_i is a random variable whose distribution is given by Binomial(M, h_i) and that $\tilde{h}_i$ can be approximated by a Gaussian random variable with mean $\mu_i = h_i$ and variance $\sigma_i^2 = h_i(1-h_i)/M$. Define

$$Z = \frac{\tilde{h}_i - \mu_i}{\sigma_i}, \tag{51}$$

which is a standard Gaussian random variable with mean 0 and variance 1. It follows that:

$$P(|Z| \le z) = 2\Phi(z) - 1, \tag{52}$$

where $\Phi(.)$ is the CDF function of the standard Gaussian distribution $\mathcal{N}(0,1)$. Therefore, we establish that:

$$P(|\tilde{h}_i - h_i| \leq z\sigma_i) = 2\Phi(z) - 1. \tag{53}$$

We can guarantee target accuracy by setting $z\sigma_i = \eta\phi_i$ and $2\Phi(z) - 1 = 1 - \xi$, which gives the following equality:

$$\frac{\eta h_i}{\sigma_i} = \Phi^{-1}\left(1 - \xi/2\right). \tag{54}$$

Substituting $\sigma_i = \sqrt{h_i(1-h_i)/M}$ into the above equation and solving for M, we obtain (50). ∎

For example, to bound $\tilde{h}_i$ within 10% percent of h_i (i.e., $\eta = 0.1$) with probability $1 - \xi = 95\%$ for all $h_i \geq 10^{-2}$, the following must hold:

$$M \geq \frac{(1 - 10^{-2}) \times 1.96^2}{10^{-2} \times 0.1^2} \approx 3.8 \times 10^4, \tag{55}$$

which indicates that $M = 38$K flows must be sampled to achieve target accuracy. If we reduce η to 1%, increase $1-\xi$ to 99%, and require the approximation to hold for all $h_i \geq 10^{-3}$, then M must be at least 66M flows. Converting η into ζ using (43), one can establish similar bounds on the deviation of $\tilde{q}_i$ from f_i.

D. Estimation of Other Flow Metrics

Besides flow sizes and the flow-size distribution, URGE also provides estimators for the total number of flows and the number of them with size i. Before introducing these estimators, we need the next lemma.

Lemma 7: The expected number of flows with sampled residuals $R_L = i$ is:

$$E[M_i] = E[M]h_i = nh_ip_s, \tag{56}$$

where h_i is the PMF of geometric residuals and p_s is given by (9).

Proof: Writing:

$$\begin{aligned} E[M_i] &= nP(A_L < L \cap R_L = i) \\ &= nP(R_L = i | A_L < L)P(A_L < L), \end{aligned} \tag{57}$$

notice that (56) follows from the fact that $P(R_L = i | A_L < L) = h_i$ and $P(A_L < L) = p_s$. ∎

Based on this, we next develop two estimators and prove their accuracy. Let $\tilde{n}$ be an estimator of the total number of flows n observed in the measurement window $[0, T]$:

$$\tilde{n} = M + \frac{1-p}{p}M_1 \tag{58}$$

and $\tilde{n}_i$ be an estimator of the number of flows n_i with size i:

$$\tilde{n}_i = \frac{M_i - (1-p)M_{i+1}}{p}. \tag{59}$$

Then, the next result shows that both of these estimators are asymptotically unbiased.

Lemma 8: Ratios $\tilde{n}/n$ and $\tilde{n}_i/n_i$, for all i such that $f_i > 0$, converge to 1 in probability as $M \to \infty$.

Proof: To prove convergence in probability, it suffices to show that $E[\tilde{n}/n] = 1$ and $Var[\tilde{n}/n] \to 0$ as $n \to \infty$. From (58), we have:

$$E[\tilde{n}] = E[M] + \frac{1-p}{p}E[M_1]. \tag{60}$$

Applying (42) and (56), we get:

$$E[\tilde{n}] = np_s\Big(1 + \frac{(1-p)h_1}{p}\Big), \tag{61}$$

which simplifies to $E[\tilde{n}] = n$ using (40).

To tackle the variance of $\tilde{n}/n$, first notice that M can be represented as a sum of n i.i.d. Bernoulli variables (i.e., $M = \sum_{j=1}^{n} A_j$), each with fixed probability p_s. Therefore:

$$Var\Big[\frac{M}{n}\Big] = \frac{1}{n^2}\sum_{j=1}^{n} Var[A_j] = \frac{p_s(1-p_s)}{n}, \tag{62}$$

where the last term is bounded by $1/n$. Applying similar reasoning to M_1, we obtain that $Var[\tilde{n}/n] \leq 1/np$. Since we assumed that the number of sampled flows $M \to \infty$, this implies that $np_s \to \infty$ and thus from (40) that $np \to \infty$, which establishes that $Var[\tilde{n}/n] \to 0$. Convergence in probability immediately follows (in fact, an even stronger convergence in mean-square holds, but this distinction is not essential in our context).

For the second part of the theorem, define $X_n = \tilde{n}_i/n$ and $Y_n = n_i/n$. We first prove that both X_n and Y_n converge in probability to f_i. We then argue that their ratio X_n/Y_n converges to 1, also in probability.

Using (56), (40), and finally (36), we have:

$$\begin{aligned} E[X_n] &= \frac{E[M_i] - (1-p)E[M_{i+1}]}{np} \\ &= \frac{p_s(h_i - (1-p)h_{i+1})}{p} \\ &= \frac{h_i - (1-p)h_{i+1}}{p + (1-p)h_1} = f_i. \end{aligned} \tag{63}$$

Since n_i is the number of flows with size i, its expectation is $E[n_i] = nP(L = i) = nf_i$ and thus $E[Y_n] = f_i$. Using reasoning similar to that in the first half of this proof, we obtain that $Var[X_n] \to 0$ and $Var[Y_n] \to 0$, which shows convergence of these variables to f_i in probability.

For the final step, consider two sequences $\{X_n\}$ and $\{Y_n\}$ that converge to the same positive constant $f_i > 0$. Then, simple manipulation shows that their ratio converges to 1 in probability. We leave details to the reader. ∎

Note that [17] provided a similar estimator as (58) and proved $E[\tilde{n}] = n$ using a different approach from ours; however, our results are stronger as they show convergence in probability and additionally address estimation of n_i. Simulations verifying (58)-(59) are omitted for brevity.

VI. Implementation

In this section, we implement URGE and examine its memory consumption and processing speed.

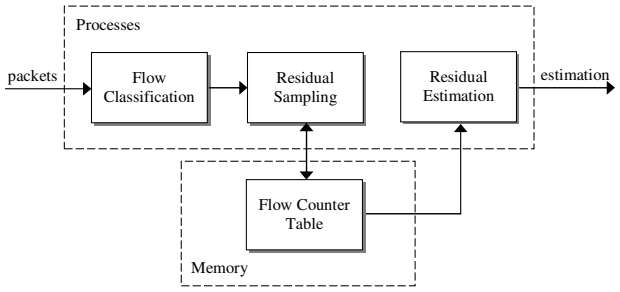


Fig. 9. The URGE framework.

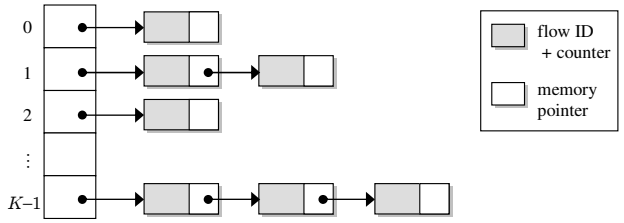


Fig. 10. Illustration of a chained hash table for maintaining flow counters.

A. General Structure

Fig. 9 illustrates a framework that implements the various URGE algorithms. This framework contains three processes – flow classification, residual-geometric sampling, and estimation – as well as one data structure containing the flow counter table.

Flow classification processes each incoming packet for flow ID and then forwards it to residual-geometric sampling. For each flow ID x arriving from flow classification, *residual-geometric sampling* first checks if the flow table has an existing entry for x and increment the counter by 1; if an entry does not exist, it is created with probability p and its counter is initialized to 1. The geometric estimation process collects counter values from the flow table and then uses URGE to estimate flow statistics.

The flow table keeps a mapping between flow IDs and associated counters. The table supports three operations: 1) $lookup(x)$ to retrieve the record of flow x; 2) $add(x)$ to insert a new entry for flow x in the table with the initial counter value 1; and 3) $increment(x)$ to add 1 to the counter of flow x. We display in Fig. 10 an implementation of the counter table, which is based on a *chained hash table*. Assume a hash function $hash(x)$ that produces an integer value in $[0, 1, \ldots, K-1]$. We assume that the generated hash values are uniformly distributed within interval $[0, K-1]$ and the implementation of function $hash(.)$ is fast enough. Efficient hardware hash functions can be found in [29].

We maintain an array A of size K and each entry $A[k]$ points to a liked list that keeps the set of flows whose IDs have the same hash value k. Each node in the list contains two fields: 1) *flow data* that keep the flow ID, the packet counter, and the timestamp of the last packet; and 2) a *pointer* to the next node. An important element of our algorithm is to ensure that the table keeps only *active* flows, which is accomplished by periodic sweeps through the table and removal of all flows that have completed using FIN/RST packets or have been idle for longer than τ time units. Upon removal, flow information is saved to disk (single-flow usage) or aggregated into a RAM-based PMF table (flow-distribution usage). Operations $add(x)$ and $increment(x)$ automatically modify the timestamps associated with each flow and allow timeout-based expulsion of dead flows.

Notice that the flow table is accessed by residual-geometric sampling upon each packet arrival. Therefore, the scalability of the measurement algorithm essentially depends on the access speed to the table. In what follows, we analyze the design of the flow table and quantify its two important properties: memory consumption and processing speed.

B. Active Flows

To understand how much benefit removal of dead flows provides to memory consumption, we next derive the expected number of active flows at any time t and their fraction sampled by the algorithm. Assume a measurement window $[0, T]$, where T is given in packets seen by the router. For each flow j, let inter-packet delays within the flow be given by a random variable Δ_j, which counts the number of packet arrivals from *other* flows between adjacent packets of j. Denoting by $\Delta = E[\Delta_j]$, we have the following result.

Lemma 9: Assuming stationary flow arrivals in $[0, T]$ and $T \to \infty$, the expected number of active flows $N(t)$ at time t is given by:

$$E[N(t)] = \Delta + 1. \tag{64}$$

Proof: Represent $N(t) = \sum_{j=1}^{n} A_j(t)$ as the sum of n indicator variables, where $A_j(t)$ is 1 if flow j is alive at t and 0 otherwise. Observe that:

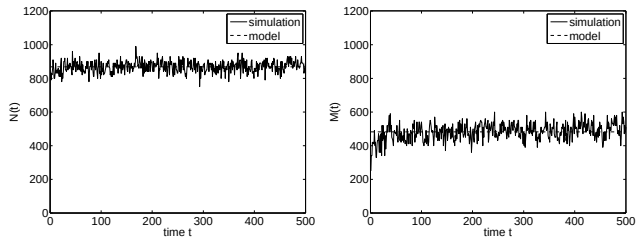
$$E[N(t)] = nE[A_j(t)] = nP(A_j(t) = 1) \tag{65}$$

and notice that each flow "exists" at the router for $\sum_{k=1}^{L}(\Delta_j^k + 1)$ packet units, where $\Delta_j^1, \Delta_j^2, \ldots$ are i.i.d. instances of variable Δ_j. Then, the probability that $t \in [0, T]$ lands within a given flow is simply the flow's expected footprint (in packets seen by the router) normalized by the window size:

$$E[A_j(t)] = \frac{1}{T} E\Big[\sum_{k=1}^{L}(\Delta_j^k + 1)\Big]. \tag{66}$$

Using Wald's equation, this simplifies to $E[A_j(t)] = \Delta E[L]/T$. Finally, since $nE[L]/T = 1$, we immediately obtain (64) using (65). ■

Our baseline reduction in flow volume comes from geometric sampling in previous sections and reduces the number of flows by a factor of $r_1 = n/E[M]$. Now additionally define ratio $r_2 = n/E[N(t)] = T/(\Delta + 1)E[L]$ and observe that longer observation windows (i.e., larger T), smaller flow sizes (i.e., smaller $E[L]$), and denser arrivals (i.e., smaller Δ) imply more savings of memory. In fact, $T \to \infty$ results in $r_2 \to \infty$ if the other parameters are fixed. However, even more reduction is possible by discarding dead flows in RGS. Denote by $M(t)$ the number of sampled flows that are still alive at t and consider the next result.



(a) alive flows (b) sampled flows

Fig. 11. Verifying models (64) and (67).

Lemma 10: Assuming the flow arrival process is stationary in $[0, T]$ and $T \to \infty$, the expected number of *active* sampled flows at time t is given by:

$$E[M(t)] = (\Delta + 1)\Big(1 - \frac{1-p}{pE[L]}p_s\Big), \tag{67}$$

where p_s in (9) is the fraction of all flows sampled by RGS.

Proof: Following Lemma 9, it suffices to derive the average packet footprint of flow j within window $[0, T]$. Dividing this footprint by T gives us the probability that current time t falls within the residual of the flow and multiplying the result by n produces the expected number of flows stored in RAM.

Condition on $L = l$ and define P_l as the number of packets counted by RGS from flow i:

$$P_l = \begin{cases} R_l & \text{flow sampled} \\ 0 & \text{otherwise} \end{cases}. \tag{68}$$

Then, the flow's footprint F_l is:

$$F_l = \sum_{k=1}^{P_l} (\Delta_j^k + 1), \tag{69}$$

where as before Δ_j^k are i.i.d. inter-packet delays induced by cross-traffic that do not depend on the size of flow j. Next, taking the expectation of F_l, we have:

$$\begin{aligned} E[F_l] &= E[P_l](\Delta + 1) \\ &= E[R_l|\text{sampled}]P(\text{sampled})(\Delta + 1). \end{aligned} \tag{70}$$

Using (14) and recalling that $P(\text{sampled}) = 1 - (1-p)^l$, we have:

$$E[F_l] = (\Delta + 1)\Big[l - \frac{1-p}{p}(1 - (1-p)^l)\Big]. \tag{71}$$

Unconditioning $L = l$, we have the expected footprint as:

$$E[F] = (\Delta + 1)\Big[E[L] - \frac{1-p}{p}p_s\Big], \tag{72}$$

where $p_s = E[1 - (1-p)^L]$ is the probability that a flow is sampled by RGS. Multiplying $E[F]$ by n, dividing by T, and taking $E[L]$ outside, we get (67). ∎

Define $r_3 = n/E[M(t)]$ as the expected reduction of space when tracking only active RGS flows compared to all seen flows at the router and notice that this ratio increases not only as T grows, but also when p decreases. Performing a self-check using Jensen's inequality, observe that $0 \le p_s/pE[L] \le 1$ and therefore $E[M(t)] \le E[N(t)]$, which means that the former indeed always results in more reduction in table size.

TABLE I
COMPARISON OF (64) AND (67) TO SIMULATION RESULTS

time t	$E[N(t)]$		$E[M(t)]$	
	simulation	model (64)	simulation	model (67)
100	867.1	866.6	487.0	486.8
200	866.4	866.6	487.0	486.8
300	866.7	866.6	486.5	486.8
400	866.3	866.6	486.4	486.8
500	866.9	866.6	486.8	486.8

We discuss numerical values of $r_1 - r_3$ in the next section and now focus on the accuracy of the obtained results.

We evaluate models (64) and (67) in simulations with $1,000$ iterations through window $[0, T]$ with randomly generated flows from the a distribution with flow-size CDF $F_i = 1 - i^{-\alpha}$, where $\alpha = 1.1$ and $p = 0.01$. Fig. 11 plots the evolution of $N(t)$ and $M(t)$ along with the expected values computed from the models. Table I compares the models with $E[N(t)]$ and $E[M(t)]$ computed in simulations, where each value is averaged using the same $1,000$ iterations of the traffic stream. Both indicate a very close match.

C. Memory Consumption

The memory used by the flow table can be divided into two parts: one for the hash table, which contains an array of pointers, and the other for flow records, which are organized in a set of linked lists. Define w_p to be the number of bytes used by each memory pointer and w_f to be that needed for flow counter, timestamp, and flow ID. Then, the following theorem gives the memory required for the measurement algorithm.

Theorem 6: The average number of bytes required by URGE in steady-state is:

$$E[W_R(t)] = Kw_p + E[M(t)](w_c + w_f), \tag{73}$$

where $E[M(t)]$ is the average number of sampled active flows at time t given by (67).

From (73), observe that for n original flows with a given distribution of L, memory consumption $E[W_R(t)]$ can be reduced by lowering either $M(t)$ or K. As discussed in the previous section, $M(t)$ cannot be arbitrarily small as it would lead to lower accuracy. At the same time, small K leads to more conflicts in the hash table, longer linked lists, and thus may slow down the sampling process, which are the issues we study next.

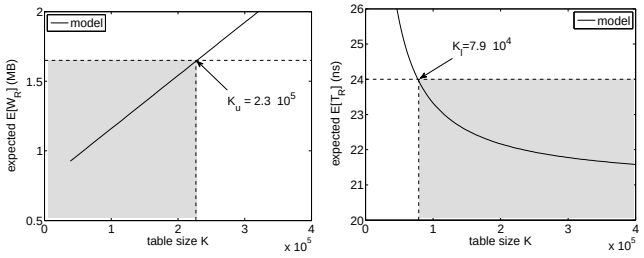
D. Processing Time

The time spent in processing each packet depends on how linked lists are built. We examine an approach that sorts flow entries of each linked list based on flow IDs. In this approach, function $lookup(x)$ returns a pointer to the entry of flow x if it exists in the table; otherwise, the function returns a pointer to where the new entry should be inserted.

For each packet with flow ID x, we perform the following steps in sequential order: 1) compute the $k = hash(x)$; 2) retrieve the linked-list head pointer $A[k]$ from the hash table; 3) iterate through the linked list until a flow record is matched or a flow with ID larger than x is reached; 4) if x is not found, a

TABLE II
CONSTANTS USED IN (73) AND (74)

RAM constant	value	CPU constant	value
w_p	4B	t_h	12ns
w_f	17B	t_p	9ns
W_0	1.65MB	t_c	3ns
		T_0	24ns



(a) $E[W_R(t)]$ (b) $E[T_R(t)]$

Fig. 12. Tradeoff: (a) memory consumption and (b) processing time with $E[M(t)] = 3.9 \times 10^4$. Gray areas display the acceptable ranges of K.

new entry for x is created with probability p and inserted to the location returned by $lookup(x)$. Notice that the fourth step is executed only when a new flow arrives and is sampled, which is much less frequent compared to the case of an existing flow. Thus, consider its contribution to the overall overhead negligible and omit it from analysis.

Denote by t_h the time spent in computing a hash, by t_p that of memory access, and by t_c that of each comparison of flow IDs. Define $T_R(t)$ to be the processing delay/latency of each incoming packet at time t. Then, noticing that the expected list length is $E[M(t)]/K$ entries and on average traversal stops in the middle of a list, we have the next result.

Theorem 7: The expected per-packet processing time is:

$$E[T_R(t)] = t_h + t_p + (t_c + t_p)\frac{E[M(t)]}{2K}. \tag{74}$$

The result in (74) indicates that both large hash table size K and small sample size $M(t)$ can contribute to a faster sampling process. Since larger K reduces (74), but increases (73), we next examine how to properly select K and p to simultaneously satisfy certain target constraints on $E[W_R(t)]$ and $E[T_R(t)]$ given their conflicting dependency on K.

E. Tradeoff Analysis

Now, we are ready to explore the design space of constants (K, p) to strike a balance between accuracy and scalability. Suppose that a router requires that $E[W_R(t)] \leq W_0$ and $E[T_R(t)] \leq T_0$. Further assume that the number of sampled flows $E[M(t)]$ is known and fixed (i.e., fixed p, window T, and flow-size distribution). Define two constants:

$$K_l = \frac{(t_c + t_p)E[M(t)]}{2(T_0 - (t_h + t_p))}, \tag{75}$$

and

$$K_u = \frac{W_0 - E[M(t)](w_c + w_f)}{w_p}. \tag{76}$$

Assuming $K_l \leq K_u$, it then follows from (73) and (74) that one can choose any value $K \in [K_l, K_u]$ to satisfy the two

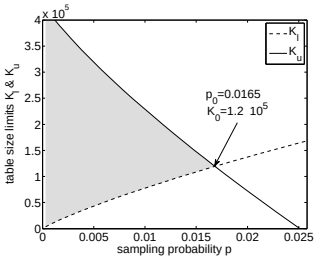


Fig. 13. Lower and upper bounds on table size K with varying probability p. Gray areas display the acceptable range of K and p.

constraints on memory and speed. We show below how to vary p in order to maximize accuracy while ensuring $K_l \leq K_u$.

To understand this better, consider the following example. Assume that the original traffic contains $n = 10^6$ flows with a power-law distribution $P(L \leq i) = 1 - i^{-1.1}$. With $p = 0.01$, residual-geometric sampling obtains $E[M(t)] = 3.9 \times 10^4$ sampled flows. Table II gives the constants we use to compute the expected memory consumption and processing time in (73) and (74). We also impose the following constraints on memory and delay: $W_0 = 1.65$MB and $T_0 = 24$ns.[1] Fig. 12 illustrates the acceptable ranges of table size K derived from the models. The figure indicates that table size K can be any value between $K_l = 7.9 \times 10^4$ and $K_u = 2.3 \times 10^5$ to simultaneously satisfy both requirements W_0 and T_0.

Note that for some values of $E[M(t)]$ it is possible that K_l is larger than K_u and thus the constraints cannot be met. Therefore, we next vary p to show how the choice of K will be affected. Fig. 13 plots K_u and K_l as functions of p, where both curves are obtained from the corresponding models. Notice from the figure that K_l monotonically *increases* and K_u monotonically *decreases* in p. This implies that interval $[K_l, K_u]$ eventually shrinks to a single point K_0, after which no feasible assignment of table size K exists.

Since larger p implies more accurate estimation (i.e., the router sees more flows M in the interval $[0, T]$ and thus estimates distribution $\{h_i\}$ more accurately), it is desirable to select the *maximum* p that allows the router to satisfy the space and speed constraints. This occurs in a single optimum point p_0 that corresponds to $K_l = K_u = K_0$. In our example, we get $p_0 = 0.0165$ and $K_0 = 1.2 \times 10^5$.

VII. PERFORMANCE EVALUATION

In this section, we evaluate our models using several Internet traces in Table III from NLANR [24] and CAIDA [3]. Trace FRG was collected from a gigabit link between UCSD and Abilene in 2006. We extracted from it additional traces with only Web, DNS, and NTP flows (also seen in the table). Additionally, we use three traces from CAIDA: LARGE – a one-hour trace from an OC48 link, MEDIUM – a one-minute trace from a OC192 link, and SMALL – a 7-minute trace from a gigabit link.

As the table shows, URGE typically sees a reasonably large number of flows M over the entire interval $[0, T]$; however,

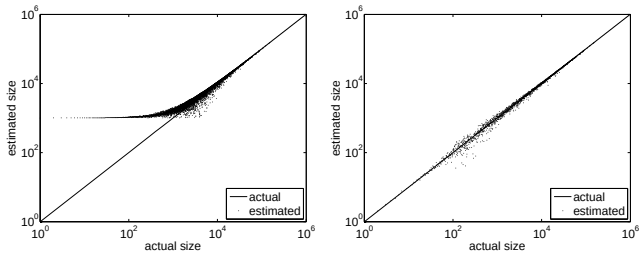
[1]These values allow to hold about 10^5 flow records (each with a flow ID and a counter) and process 1-Kbit packets at OC-768 rates (i.e., 40 Gbps).

TABLE III
REDUCTION IN THE NUMBER OF FLOWS USING RESIDUAL SAMPLING WITH $p = 0.01$ AND DIFFERENT TYPES OF PERIODIC REMOVAL OF DEAD FLOWS

source	trace	total flows n	total pkts $nE[L]$	sampling only		removal only		both	
				$E[M]$	r_1	$E[N(t)]$	r_2	$E[M(t)]$	r_3
NLANR	FRG	1,756,702	131,821,685	117,995	15	21,645	81	2,669	658
	Web	239,174	6,497,894	26,051	9	9,698	24	985	240
	DNS	120,446	292,977	2,073	44	600	152	19	4,797
	NTP	382,489	720,447	4,086	54	3,036	73	77	2,887
CAIDA	LARGE	9,653,609	117,250,415	519,144	19	262,525	37	21,590	447
	MEDIUM	2,317,369	43,837,666	139,316	17	281,137	8	53,903	43
	SMALL	200,910	2,179,574	12,862	16	44,414	5	5,948	34

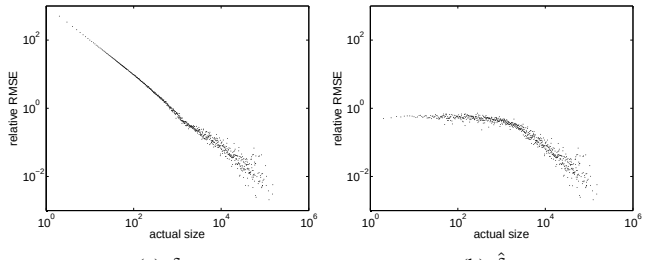
TABLE IV
PERFORMANCE OF URGE WITH $p = 0.001$ AND HASH TABLE SIZE $K = E[M(t)]$

source	trace	$E[W_R(t)]$	$E[T_R(t)]$	# of flows			# of size-one flows		
				actual (n)	estimated ($\tilde{n}$)	error	actual (n_1)	estimated ($\tilde{n}_1$)	error
NLANR	FRG	31KB	24.1ns	1,756,702	1,736,261	1.16%	768,742	749,958	2.44%
	Web	10KB	21.4ns	239,174	253,996	6.2%	13,686	13,922	1.72%
	DNS	257B	21ns	120,446	124,176	3.1%	76,607	78,045	1.88%
	NTP	752B	21.1ns	382,489	375,326	1.87%	281,370	279,096	0.8%
CAIDA	LARGE	132KB	28.1ns	9,653,609	9,717,315	0.66%	4,535,449	4,630,037	2.09%
	MEDIUM	341KB	23.7ns	2,317,369	2,278,984	1.66%	1,299,343	1,273,989	1.95%
	SMALL	23KB	21.2ns	200,910	202,604	0.84%	93,575	95,106	1.64%



(a) $E[e(R_l)]$ (b) $E[\hat{e}(R_l)]$

Fig. 14. Estimating single-flow usage in the FRG trace with $p = 0.001$.



(a) δ_l (b) $\hat{\delta}_l$

Fig. 15. RRMSE of single-flow usage in the FRG trace with $p = 0.001$.

the number of active flows $N(t)$ and those constantly kept in memory $M(t)$ is much smaller. For the FRG trace, for example, $E[M]$ is 15 times smaller than n, while $E[N(t)]$ is 81 and $E[M(t)]$ is 658 times smaller. In general, NLANR traces benefit more from the removal of dead flows than CAIDA data, because former was collected over two consecutive days and thus had a larger observation window T, which led to larger ratios r_2 and r_3. The same reasoning also explains the fact that the LARGE trace exhibits much higher benefit from removing dead flows than MEDIUM or SMALL traces.

A. Memory and Speed

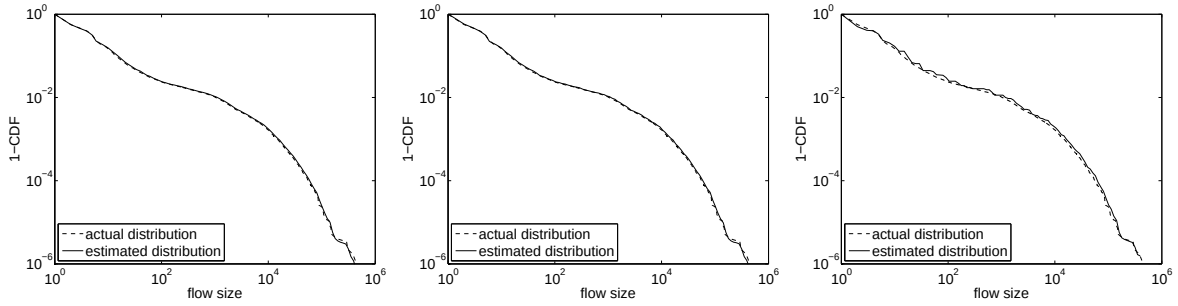
We use the settings of Table II to compute the amount of memory consumed by URGE according to (73). As shown in the third column of Table IV for $p = 0.001$ and $K = E[M(t)]$, the required memory size is small and rarely exceeds 40 KB. Even for the LARGE trace that has the most flows in this comparison, URGE only needs 132 KB of RAM, much smaller than roughly 120 MB required for keeping all flow counters. We also compute per-packet processing time from (74) based on Table II and show in the fourth column of Table IV that $E[T_R(t)] \leq 25$ ns in the majority of the studied cases.

B. Estimation Accuracy

First, we examine the problem of estimating the total number of flows n in $[0, T]$ and size-one flows n_1 in this interval. The seventh and tenth columns of Table IV list the absolute error of models (58) and (59), respectively. With the exception of the Web NLANR trace, these estimates are within approximately 3% of the correct value.

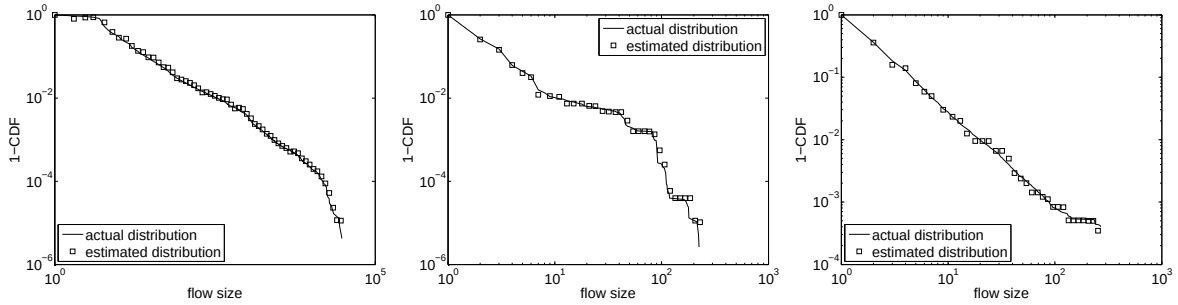
We next evaluate the performance of URGE in estimating single-flow usage. Fig. 14 plots the expectation of estimated flow sizes (averaged over 100 iterations) along with the actual values obtained from the FRG trace using $p = 0.001$. The figure shows that the estimator $e(R_l)$ from previous work tends to overestimate the sizes of small flows, while URGE's estimator $\hat{e}(R_l)$ accurately follows the actual values. We also compare the relative errors of the two studied methods in Fig. 15, which indicates that URGE has RRMSE bounded by 1 for all flows, while $e(R_l)$ exhibits very large δ_l for small and medium flows, which is an increasing function of $1/p$.

For the flow-size distribution, we first examine three values of p to compare its effect on the accuracy of URGE in the FRG trace. Fig. 16 indicates that estimation for all three values of p are very consistent and all of them follow the actual distribution accurately. In our experiments with $p = 0.0001$, URGE recovered the original PMF $\{f_i\}$ using only $M = 7,616$ total flows out of $n = 1.75$M.



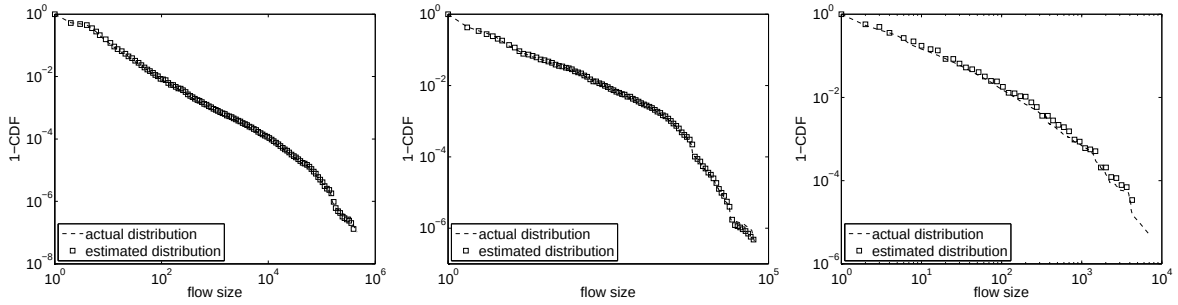
(a) $p = 0.01$ (b) $p = 0.001$ (c) $p = 0.0001$

Fig. 16. Estimating the flow size distribution using URGE in the FRG trace.



(a) Web (b) NTP (c) DNS

Fig. 17. Estimating the flow size distribution using URGE in NLANR traces with $p = 0.001$.



(a) LARGE (b) MEDIUM (c) SMALL

Fig. 18. Estimating the flow size distribution using URGE in CAIDA traces with $p = 0.001$.

Finally, we apply URGE with $p = 0.001$ to NLANR traces of different traffic types and plot in Fig. 17 the estimated distributions along with the actual ones. As the figure shows, the flow statistics of different applications can be accurately estimated by URGE. We observe a similar match in our experiments with three CAIDA traces as shown in Fig. 18.

VIII. Conclusion

In this paper, we proved that previous methods based on residual-geometric sampling had certain bias in estimating single-flow usage and were unable to recover the flow-size distribution from the sampled residuals. To overcome this limitation, we proposed a novel modeling framework for analyzing residual sampling and developed a set of algorithms that were able to perform accurate estimation of flow statistics, even under the constraints of small router RAM size, short trace duration, and low CPU sampling overhead.

References

[1] D. Brauckhoff, B. Tellenbach, A. Wagner, A. Lakhina, and M. May, "Impact of Traffic Sampling on Anomaly Detection Metrics," in *Proc. ACM IMC*, Oct. 2006, pp. 159–164.

[2] G. Casella and R. L. Berger, *Statistical Inference*, 2nd ed. Duxbury/Thomson Learning, 2002.

[3] Cooperative Association for Internet Data Analysis (CAIDA). [Online]. Available: http://www.caida.org/.

[4] R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth, "On the Lambert W Function," *Advances in Computational Mathematics*, vol. 5, pp. 329–359, 1996.

[5] N. Duffield and C. Lund, "Predicting Resource Usage and Estimation Accuracy in an IP Flow Measurement Collection Infrastructure," in *Proc. ACM IMC*, Oct. 2003, pp. 179–191.

[6] N. Duffield, C. Lund, and M. Thorup, "Charging from Sampled Network Usage," in *Proc. ACM IMW*, Nov. 2001, pp. 245–256.

[7] N. Duffield, C. Lund, and M. Thorup, "Estimating Flow Distributions from Sampled Flow Statistics," in *Proc. ACM SIGCOMM*, Aug. 2003, pp. 325–336.

[8] N. Duffield, C. Lund, and M. Thorup, "Flow Sampling under Hard Resource Constraints," in *Proc. ACM SIGMETRICS*, Jun. 2004, pp. 85–96.

[9] N. Duffield, C. Lund, and M. Thorup, "Learn More, Sample Less: Control of Volume and Variance in Network Measurement," *IEEE Trans. Inform. Theory*, vol. 51, no. 5, pp. 1756–1775, May 2005.

[10] C. Estan, K. Keys, D. Moore, and G. Varghese, "Building a Better Netflow," in *Proc. ACM SIGCOMM*, Aug. 2004, pp. 245–256.

[11] C. Estan and G. Varghese, "New Directions in Traffic Measurement and Accounting," in *Proc. ACM SIGCOMM*, Aug. 2002, pp. 323–336.

[12] W. Fang and L. Peterson, "Inter-AS Traffic Patterns and their Implications," in *Proc. IEEE GLOBECOM*, Dec. 1999, pp. 1859–1868.

[13] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, and F. True, "Deriving Traffic Demands for Operational IP Networks: Methodology and Experience," *IEEE/ACM Trans. Netw.*, vol. 9, no. 3, pp. 265–280, Jun. 2001.

[14] N. Hohn and D. Veitch, "Inverting Sampled Traffic," in *Proc. ACM IMC*, Oct. 2003, pp. 222–233.

[15] C. Hu, S. Wang, J. Tian, B. Liu, Y. Cheng, and Y. Chen, "Accurate and Efficient Traffic Monitoring Using Adaptive Non-linear Sampling Method," in *Proc. IEEE INFOCOM*, Apr. 2008, pp. 421–429.

[16] K. Ishibashi, R. Kawahara, T. Mori, T. Kondoh, and S. Asano, "Effect of Sampling Rate and Monitoring Granularity on Anomaly Detectability," in *Proc. IEEE Global Internet Symposium*, May 2007, pp. 25–30.

[17] R. R. Kompella and C. Estan, "The Power of Slicing in Internet Flow Measurement," in *Proc. USENIX/ACM IMC*, Oct. 2005, pp. 105–118.

[18] A. Kumar, M. Sung, J. Xu, and J. Wang, "Data Streaming Algorithms for Efficient and Accurate Estimation of Flow Size Distribution," in *Proc. ACM SIGMETRICS*, Jun. 2004, pp. 177–188.

[19] A. Kumar, M. Sung, J. Xu, and E. Zegura, "A Data Streaming Algorithm for Estimating Subpopulation Flow Size Distribution," in *Proc. ACM SIGMETRICS*, Jun. 2005, pp. 61–72.

[20] A. Kumar and J. Xu, "Sketch Guided Sampling – Using On-Line Estimates of Flow Size for Adaptive Data Collection," in *Proc. IEEE INFOCOM*, Apr. 2006, pp. 1–11.

[21] A. Kumar, J. Xu, J. Wang, O. Spatschek, and L. Li, "Space-Code Bloom Filter for Efficient Per-Flow Traffic Measurement," in *Proc. IEEE INFOCOM*, Mar. 2004, pp. 1762–1773.

[22] Y. Lu, A. Montanari, B.Prabhakar, S. Dharmapurikar, and A. Kabbani, "Counter Braids: A Novel Counter Architecture for Per-Flow Measurement," in *Proc. ACM SIGMETRICS*, Jun. 2008, pp. 121–132.

[23] J. Mai, C.-N. Chuah, A. Sridharan, T. Ye, and H. Zang, "Is Sampled Data Sufficient for Anomaly Detection?" in *Proc. ACM IMC*, Oct. 2006, pp. 165–176.

[24] National Laboratory for Applied Network Research (NLANR). [Online]. Available: http://moat.nlanr.net/.

[25] Cisco NetFlow. [Online]. Available: http://www.cisco.com/en/US/products/ps6601/prod_white_papers_list.html.

[26] Cisco Sampled NetFlow. [Online]. Available: http://www.cisco.com/en/US/docs/ios/12_0s/feature/guide/12s_sanf.html.

[27] R. Pan, L. Breslau, B. Prabhakar, and S. Shenker, "Approximate Fairness through Differential Dropping," *ACM SIGCOMM Comp. Comm. Rev.*, vol. 33, no. 2, pp. 23–39, Apr. 2003.

[28] S. Ramabhadran and G. Varghese, "Efficient Implementation of a Statistics Counter Architecture," in *Proc. ACM SIGMETRICS*, Jun. 2003, pp. 261–271.

[29] M. Ramakrishna, E. Fu, and E. Bahcekapili, "Efficient Hardware Hashing Functions for High Performance Computers," *IEEE Trans. Comput.*, vol. 46, no. 12, pp. 1378–1381, 1997.

[30] D. Shah, S. Iyer, B. Prabhakar, and N. McKeown, "Analysis of a Statistics Counter Architecture," in *Proc. IEEE Hot Interconnects*, Aug. 2001, pp. 107–111.

[31] X. Wang, Z. Yao, and D. Loguinov, "Residual-Based Measurement of Peer and Link Lifetimes in Gnutella Networks," in *Proc. IEEE INFOCOM*, May 2007, pp. 391–399.

[32] L. Yang and M. Michailidis, "Sampled Based Estimation of Network Traffic Flow Characteristics," in *Proc. IEEE INFOCOM*, May 2007, pp. 1775–1783.

[33] Q. Zhao, J. Xu, and Z. Liu, "Design of a Novel Statistics Counter Architecture with Optimal Space and Time Efficiency," in *Proc. ACM SIGMETRICS*, Jun. 2006, pp. 323–334.