Modeling Residual-Geometric Flow Sampling

Xiaoming Wang, *Student Member, IEEE, ACM*, Xiaoyong Li, and Dmitri Loguinov, *Senior Member, IEEE*

***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 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.**

***Index Terms*—Flow-size estimation, traffic sampling.**

I. Introduction

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 accuracy objectives while satisfying real-time resource constraints of high-speed Internet routers (e.g., fixed memory consumption and per-packet processing delay). This is commonly accomplished (e.g., [4], [7]–[13], [16], [17], [19]–[24], and [33]) 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 individual flows [13], [17], [19], [22]–[24]; and 2) building the distribution of flow sizes seen by the router in some time window [9], [20], [33]—coupled in a single measurement technique. The former problem arises in usage-based accounting and traffic engineering [8], [13]–[15], [27], while the latter has many security applications such as anomaly and intrusion detection [1], [18], [25].

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 fixed size (in packets) of a particular flow, sampled residual R_l is a random variable $l-A$. Stochastically larger A results in fewer flows being sampled and leads to lower CPU/RAM overhead, but also increases the amount of estimation error. Besides omitting 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.

Residual sampling has been modeled in the setting of peer-to-peer (P2P) networks [31], where the initial point A was uniformly distributed within each user's lifetime. Flow-based estimation [13], [19], however, 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 only limited analytical treatment in [13], [19], which did not shed light on the possibility of unbiased estimation of individual flow sizes, asymptotically accurate recovery of the flow-size distribution, or space-CPU requirements in steady state. We overcome these limitations in what follows.

A. Problem Statement

In this paper, we seek to develop a flow-measurement mechanism that can achieve the following objectives. First, the algorithm should provide unbiased estimation of single-flow usage [13], [17], [19], [22]–[24] and flow-size distribution [9], [20], [33]. Second, the method should be able to reliably identify elephant flows as their size $l \to \infty$ [8], [13]–[15], [27]. Third, the algorithm must scale in terms of memory consumption and processing speed. Specifically, the number of flows maintained in RAM and the time spent on each packet should not exceed certain constraints of the system. Finally, the algorithm must be real-time and cannot assume information only available offline. As we discuss in Section II, none of the existing methods simultaneously satisfy these criteria.

B. 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 the proper notation and desired properties for the estimation algorithm.

Assume that for a flow of size L the sampling process produces residual R_L, where both L and R_L are now random variables. In this setting, it is common to call estimator $e(R_L)$ *unbiased* if its expectation produces the correct flow size, i.e., $E[e(R_L)|L=l] = E[e(R_l)] = l$. We further call an estimator *elephant-accurate* if ratio $e(R_l)/l \to 1$ in mean-square as $l \to \infty$. If an estimator is unbiased, this condition is equivalent to $\mathrm{Var}[e(R_l)/l] \to 0$ as $l \to \infty$, which implies that random variable $e(R_l)/l$ converges in probability to 1.

Manuscript received June 02, 2011; revised June 21, 2012; accepted August 27, 2012; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor M. Kodialam. Date of publication January 01, 2013; date of current version August 14, 2013. This work was supported by the NSF under Grant CNS-1017766. A shorter version of this paper appeared in the IEEE International Conference on Computer Communications (INFOCOM), Shanghai, China, April 10–15, 2011.

X. Wang is with Amazon.com, Seattle, WA 98101 USA (e-mail: xmwang@gmail.com).

X. Li and D. Loguinov are with Texas A&M University, College Station, TX 77843 USA (e-mail: xiaoyong@cse.tamu.edu; dmitri@cse.tamu.edu).

Digital Object Identifier 10.1109/TNET.2012.2231435

Prior work on RGS [13], [19] 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), our first contribution is to build in Section III 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 show in Section IV that (1) is generally biased and tends to overestimate the original flow size by a factor of up to $1/p$. Motivated by this finding, our second contribution is to derive in Section V a novel estimator $\hat{e}(.)$ that is not only provably unbiased and elephant-accurate, but also unique. We obtain its mean-square error $\hat{\delta}_l$ for small l, showing that it is orders of magnitude smaller than the corresponding value for estimator (1). We also discuss how to use our model to determine when the considered estimators approximate the true flow size with sufficient accuracy.

C. Flow-Size Distribution

A seemingly related, but largely orthogonal, problem is to use RGS to estimate the original flow-size probability mass function (PMF), which we assume is given by $f_i = P(L = i)$, for $i = 1, 2, \ldots$. We call PMF estimator $\{q_i\}_{i=1}^{\infty}$ *asymptotically unbiased* if each q_i converges in probability to f_i as the number of sampled flows $M \to \infty$.

As this problem has not been addressed in the context of RGS, one may be at first tempted to compute the distribution of L based on the values produced by (1) or its unbiased replacement $\hat{e}(.)$ proposed in this paper. However, we show in Section IV that such $\{q_i\}_{i=1}^{\infty}$ almost always differ from the original distribution $\{f_i\}_{i=1}^{\infty}$, and the bias persists as sample size $M \to \infty$. The reason for this discrepancy is that single-flow models can only capture 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 toward the tail.

Using the general model of RGS derived in the paper, our third contribution is to obtain in Section V an entirely different PMF estimator $\{\tilde{q}_i\}_{i=1}^{\infty}$ and show that it tends to $\{f_i\}_{i=1}^{\infty}$ in probability as $M \to \infty$. We go a step further and provide asymptotically unbiased estimators for other metrics of interest (e.g., the total number of flows and the number of them with exactly i packets). We call the resulting combination of algorithms *Unbiased Residual-Geometric Estimators* (URGE).

D. Properties and Evaluation

Since prior work has not discussed how residual sampling should be implemented nor examined its overhead, our fourth contribution is to provide in Section VI a detailed exposition on these topics, including deriving the expected number of active flows and the number of them sampled by the router (i.e., kept in RAM). We also address the issue of optimizing accuracy while simultaneously satisfying certain constraints on RAM utilization and processing speed. Our results show that residual-geometric sampling admits a unique optimal pair of sampling probability p and hash-table size K, which we derive in closed form based on our model.

Our final contribution is to apply in Section VII the proposed technique to real Internet traces. We discover that URGE is not only very accurate for all studied metrics of interest, but also quite robust against noise when sample size becomes very small, which makes it suitable for extremely high-speed operation (due to the small hash-table requirement) and/or monitoring of individual customer networks and specific protocols that transmit only a limited amount of packets.

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) [4] 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, [16] shows that it is impossible to accurately recover the original flow-size distribution from sampled SNF data. Estan *et al.* [12] 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 [4] and [12], 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) [22] and *Adaptive Non-Linear Sampling* (ANLS) [17]. 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* [16], each flow is sampled independently with probability p, and then all packets in sampled flows are counted. Hohn *et al.* [16] 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 [8], [13]–[15], [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.* [13] introduce an algorithm called *Sample-and-Hold* (S&H), which captures flows of size l with probability $1-(1-p)^l$. While S&H identifies large flows with probability $1-o(1)$ as $l \to \infty$, it does not provide an estimate of the flow-size distribution. Another direction of size-dependent flow sampling has been explored by Duffield *et al.* in [7], [8], and [10], which present a 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 offline.

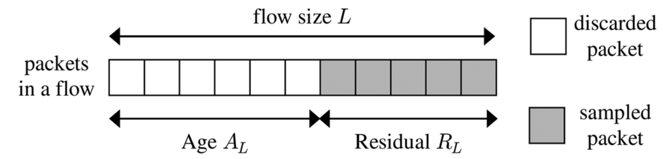


Fig. 1. Residual-geometric of a flow with size L.

Kompella *et al.* [19] examine a technique called Flow Slicing (FS), which combines SNF and S&H with a variant of smart sampling. Other nonsampling methods include *exact counting* [3], [28], [30], [34] and *lossy counting* [20], [24], which are orthogonal to our work.

III. Underlying Model

In this section, we build a general probabilistic model of Sample-and-Hold [13] and establish the necessary analytical foundation for the results that follow.

A. Formalizing 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.

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-by-one geometric random variable with success probability p, i.e., $P(G=j)=(1-p)^j p$ for $j=0,1,\ldots$ It thus follows that A_L is simply

$$A_L = \min(G, L). \tag{2}$$

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., $A_L < L$)

$$R_L = L - A_L \tag{3}$$

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*.

Assume that L is distributed according to some PMF $f_i = P(L=i)$, where $i=1,2,\ldots$ Then, we have the following result.

Theorem 1: RGS samples flows with probability

$$p_s = P(A_L < L) = E\left[1-(1-p)^L\right] = 1 - \sum_{i=1}^{\infty} f_i (1-p)^i. \tag{4}$$

Proof: Observe that for a fixed flow size $L=l$, we have $P(A_l < l) = 1-(1-p)^l$. Unconditioning L, we get (4). ■

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

Theorem 2: The PMF of geometric residual R_L is

$$h_i = \frac{p\sum_{j=i}^{\infty} f_j (1-p)^{j-i}}{p_s}. \tag{5}$$

Proof: Using (3), we have

$$\begin{aligned} h_i &= P(R_L = i) = P(L - A_L = i | A_L < L) \\ &= \frac{P(L-A_L = i \cap A_L < L)}{p_s} \end{aligned} \tag{6}$$

where $p_s = P(A_L < L)$. Substituting (2) into (6) and combining the fact that $L - G = i \geq 1$, we establish

$$h_i = \frac{P(L-G=i)}{p_s} = \frac{\sum_{j=1}^{\infty} P(G=j-i) f_j}{p_s} \tag{7}$$

which gives the desired result in (5) after substituting the PMF of G into (7). ■

The result of Theorem 2 is fundamental as many of the results in this paper are conveniently derived from (5).

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. The properties of R_l are given next.

Theorem 3: Given flow size $L=l$, the PMF of R_l is

$$P(R_l = i) = \frac{(1-p)^{l-i} p}{1-(1-p)^l} \tag{8}$$

and its expectation is

$$E[R_l] = \frac{l}{1-(1-p)^l} + 1 - 1/p. \tag{9}$$

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 (5)

$$P(R_l = i) = \frac{\sum_{j=i}^{\infty} f_j (1-p)^{j-i} p}{1-(1-p)^l} = \frac{(1-p)^{l-i} p}{1-(1-p)^l} \tag{10}$$

which is exactly (8).

We next derive $E[R_l]$ by expanding it as

$$E[R_l] = E[l - A_l | A_l < l] = l - E[G | G < l]. \tag{11}$$

Recall that for any nonnegative 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 (11) reduces to

$$\begin{aligned} E[R_l] &= l - \sum_{j=0}^{l-1} P(G > j | G < l) \\ &= \sum_{j=0}^{l-1} P(G \leq j | G < l) = \frac{\sum_{j=0}^{l-1} P(G \leq j)}{P(G < l)}. \end{aligned} \tag{12}$$

Substituting $P(G \leq j) = 1 - (1-p)^{j+1}$ into (12), we have

$$E[R_l] = \frac{\sum_{j=0}^{l-1}\left[1-(1-p)^{j+1}\right]}{1-(1-p)^l} = \frac{l-(1-p)\left(1-(1-p)^l\right)/p}{1-(1-p)^l} \tag{13}$$

which can be simplified to (9). ■

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 [13], [19] to estimating single-flow usage and whether their results can be generalized to recover the PMF of L.

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 $l \geq 1$.

One commonly faces an additional requirement to not only capture all large flows, but also to accurately estimate their size *with just a single value* $e(R_l)$, which is formalized next.

Definition 2: Estimator $e(R_l)$ is called *elephant-accurate* if $e(R_l)/l \to 1$ in mean-square as $l \to \infty$, i.e., $E[(e(R_l)/l - 1)^2] \to 0$. For unbiased estimators, this is equivalent $\mathrm{Var}[e(R_l)/l] \to 0$.

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 [13] and [19] suggest the following estimator:

$$e(R_l) = R_l - 1 + 1/p \tag{14}$$

where p is the probability of \samplingname. The next result directly follows from (9).

Corollary 1: The average value produced by (14) is

$$E[e(R_l)] = \frac{l}{1-(1-p)^l}. \tag{15}$$

Note that (15) indicates that (14) 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 to zero, which is consistent with the conclusion in [19].

To see the extent of bias in (14) and verify (15), we apply residual-geometric sampling to flows of size l ranging from 1 to 10^6 packets, feed the measured sizes to (14), and average the result after 1000 iterations for each l. Fig. 2 plots the obtained $E[e(R_l)]$ along with model (15). The figure indicates that (15) indeed captures the bias and that (14) tends to over-estimate the size of short flows *even in expectation*, where smaller sampling probability p leads to more error.

To quantify the error of individual values $e(R_l)$ in estimating flow size l and to understand elephant-accuracy, denote

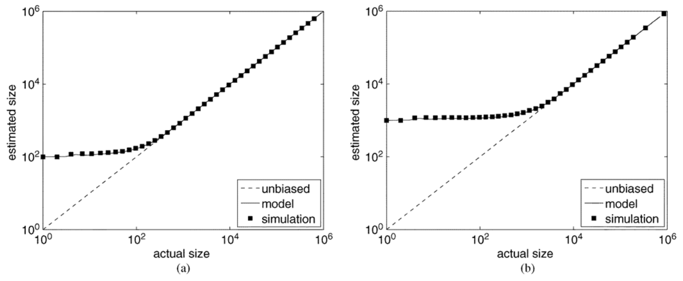


Fig. 2. Expectation of estimator (14) in simulations and its model (15). (a) $p = 0.01$. (b) $p = 0.001$

by $Y_l = e(R_l)/l$ and define the *relative root mean square error* (RRMSE) to be

$$\delta_l = \sqrt{E\left[(Y_l - 1)^2\right]}. \tag{16}$$

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.

Theorem 4: The RRMSE of (14) is given by

$$\delta_l = \frac{1}{lp}\sqrt{\frac{(1-p) - l^2p^2(1-p)^l - (1-p)^{l+1}}{1-(1-p)^l}}. \tag{17}$$

Proof: Using $e(R_l) = R_l - 1 + 1/p$ and $Y_l = e(R_l)/l$ in (16), we have

$$\delta_l = \frac{1}{l}\sqrt{E\left[(R_l - 1 + 1/p - l)^2\right]}. \tag{18}$$

Since geometric age $A_l = l - R_l$, (18) becomes

$$\delta_l = \frac{1}{l}\sqrt{E\left[A_l^2\right] + 2(1-1/p)E[A_l] + (1-1/p)^2}. \tag{19}$$

Defining $C = 1-(1-p)^l$ and applying (8) and (9), we get

$$E\left[A_l^2\right] = \sum_{i=0}^{l-1} i^2 P(A_l = i) = \frac{p}{C}\sum_{i=0}^{l-1} i^2(1-p)^i \tag{20}$$

and

$$E[A_l] = l - E[R_l] = l - 1 + 1/p - \frac{l}{C}. \tag{21}$$

As the summation term in (20) simplifies to [32]

$$\frac{p^2 - 3p + 2 - (l^2p^2 - 2lp^2 + p^2 + 2lp - 3p + 2)(1-p)^l}{p^3} \tag{22}$$

the statement of the theorem follows after several arithmetic manipulations. ■

Fig. 3 plots (17) against simulations, indicating a close match. The relative error starts at $(1-p)/p$ for $l = 1$ and decays approximately as $1/(lp)$ as $l \to \infty$. It generally does not drop below ϵ percent until l reaches approximately $1/(p\epsilon)$ packets (e.g., 100% error requires flows with size at least $1/p$, and 1% error at least $100/p$). As $l \to \infty$, $\delta_l \to 0$ and the estimator is elephant-accurate.

B. Flow-Size Distribution

We now investigate whether $e(R_L)$ defined in (14) can be used to estimate the actual flow-size distribution $\{f_i\}_{i=1}^{\infty}$.

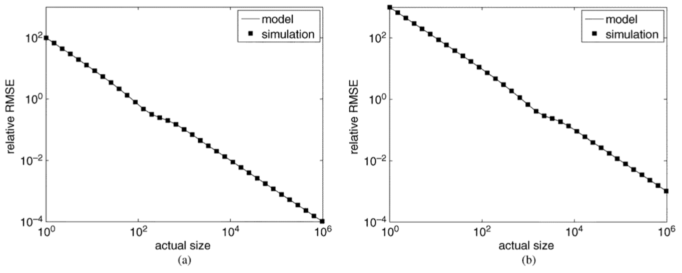


Fig. 3. RRMSE of (14) in simulations and its model (17). (a) $p = 0.01$. (b) $p = 0.001$.

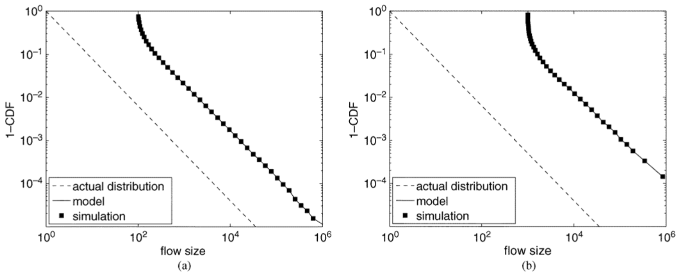


Fig. 4. Distribution $\{q_i\}$ in simulations and its model (23). (a) $p = 0.01$. (b) $p = 0.001$.

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, another 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 (5).

Theorem 5: The PMF of flow sizes estimated from (14) is

$$q_i = \frac{p \sum_{j=y(i)}^{\infty} f_j (1-p)^{j-y(i)}}{p_s} \tag{23}$$

where $y(i) = \lceil i + 1 - 1/p \rceil$ and p_s is in (4).

The result in (23) 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 (23) 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 complementary cumulative distribution function (ccdf) of random variable $e(R_L)$ obtained from simulations as well as model (23), both in comparison to the tail of the actual distribution. The figure shows that (23) 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 generally biased in single-flow usage and unable to recover the flow-size distribution from residuals R_L. This motivates us to seek alternative estimation approaches, which we perform next.

V. URGE

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

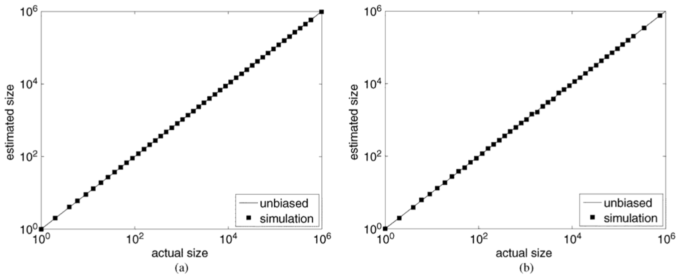


Fig. 5. Expectation of estimator (25) in simulations. (a) $p = 0.01$. (b) $p = 0.001$.

A. Single-Flow Usage

For estimating individual flow sizes, we first consider an estimator directly implied by the result in (9). Notice that solving (9) for l and expressing it in terms of $E[R_l]$, we get

$$l = u - \frac{1}{\log(1-p)} W\left(u(1-p)^u \log(1-p)\right) \tag{24}$$

where $u = E[R_l] + 1/p - 1$ and $W(z)$ is Lambert's function (i.e., a multivalued solution to $We^W = z$) [6]. Thus, a possible estimator can be computed from (24) with $E[R_l]$ replaced by the measured value of geometric residual R_l.

However, there are two reasons that (24) 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, (24) is still biased due to the crude approximation of $E[R_l]$ with a single instance of random variable R_l. Abandoning this direction, our next approach is to directly solve $E[e(R_l)] = l$ for function $e(.)$.

Theorem 6: The following is a unique unbiased estimator:

$$\hat{e}(R_l) = R_l - 1 + 1/p - \frac{(1-p)^{R_l}}{p}. \tag{25}$$

Proof: Our goal is to derive such function $\psi(.)$ that satisfies $E[\psi(R_l)] = l$. First, it follows from (8) that

$$E[\psi(R_l)] = \sum_{j=1}^{l} \psi(j) P(R_l = j) = \frac{\sum_{j=1}^{l} \psi(j)(1-p)^{l-j} p}{1-(1-p)^l}.$$

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

$$\sum_{j=1}^{l} \psi(j)(1-p)^{-j} = \frac{l\left(1-(1-p)^l\right)}{p(1-p)^l}. \tag{26}$$

Writing (26) twice for l and $l-1$ and subtracting the two equations from each other, we get

$$\psi(l)(1-p)^{-l} = \frac{1 + p(l-1) - (1-p)^l}{p(1-p)^l}. \tag{27}$$

Simplifying (27), we obtain (25) as the unique solution. ■

We plot in Fig. 5 simulation results obtained from (25). The figure indicates that $\hat{e}(R_l)$ accurately estimates actual sizes for all l. Next, we derive the corresponding error.

Theorem 7: The RRMSE of (25) is given by

$$\hat{\delta}_l = \frac{1}{lp} \sqrt{\frac{(1-p) - lp(2-p)(1-p)^l - (1-p)^{2l+1}}{1-(1-p)^l}}. \tag{28}$$

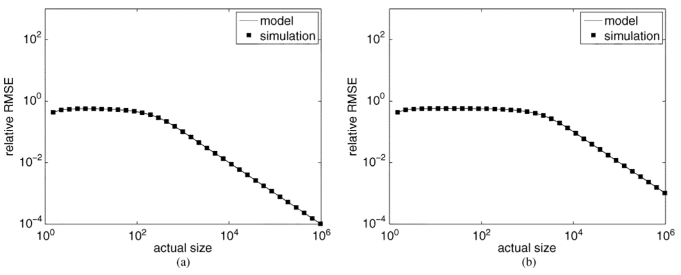


Fig. 6. RRMSE of (25) in simulations and model (28). (a) $p = 0.01$. (b) $p = 0.001$.

Proof: Define $Z = A_l + (1 - 1/p) + (1-p)^{l-A_l}/p$ and observe that

$$\hat{\delta}_l = \frac{1}{l}\sqrt{E\left[(\hat{e}(R_l) - l)^2\right]} = \frac{1}{l}\sqrt{E[Z^2]}. \quad (29)$$

Setting $C = 1-(1-p)^l$ and leveraging (8), we get

$$E[Z^2] = \frac{p}{C}\sum_{i=0}^{l-1}(i + A + B_i)^2(1-p)^i \quad (30)$$

where $A = 1 - 1/p$ and $B_i = (1-p)^{l-i}/p$. We next split

$$\sum_{i=0}^{l-1}\left[i^2 + A^2 + B_i^2 + 2(iA + iB_i + AB_i)\right](1-p)^i \quad (31)$$

into six summations $S_1, \ldots, S_6$, respectively. Note that S_1 is available from (22) and

$$S_2 = \frac{(1-p)^2}{p^2}\sum_{x=0}^{l-1}(1-p)^x = \frac{(1-p)^2 C}{p^3} \quad (32)$$

$$S_3 = \frac{(1-p)^{2l}}{p^2}\sum_{x=0}^{l-1}(1-p)^{-x} = \frac{(1-p)^{l+1}C}{p^3} \quad (33)$$

$$S_5 = \frac{2(1-p)^l}{p}\sum_{x=0}^{l-1}x = \frac{(1-p)^l(l-1)l}{p} \quad (34)$$

$$S_6 = \frac{2(1-1/p)(1-p)^l}{p}\sum_{x=0}^{l-1}1 = -\frac{2(1-p)^{l+1}l}{p^2}. \quad (35)$$

For S_4, one requires the following result [32]:

$$\sum_{i=0}^{l-1} i(1-p)^i = \frac{1 - p - (lp - p + 1)(1-p)^l}{p^2} \quad (36)$$

which produces

$$S_4 = 2\frac{(lp-p+1)(1-p)^{l+1} - (1-p)^2}{p^3}. \quad (37)$$

Finally, combining the various summations and canceling redundant terms, we get (28). ■

It is easy to verify from (28) that URGE has zero RRMSE not just for $l = \infty$, which confirms its elephant-accuracy, but also for $l = 1$. We plot $\hat{\delta}_l$ obtained from simulations along with the model in Fig. 6, which shows that (28) accurately tracks the actual relative error. From Figs. 5 and 6, it is clear that $\hat{e}(R_l)$ significantly improves the accuracy of estimating small flow sizes compared to $e(R_l)$, with $\hat{\delta}_l$ always staying below 100%.

In terms of practical applications for (28), observe that it can be used by ISPs to decide what pairs of (l, p) lead to desired accuracy when relying on a single sample of random variable $e(R_l)$ to estimate l.

B. Flow-Size Distribution

It is worth mentioning that while (25) produces unbiased estimation of individual flow sizes, $\hat{e}(R_L)$ is not suitable for computing the flow-size distribution, as we show in the following. Denote by $\hat{q}_i = P(\hat{e}(R_L) = i)$ the PMF of $\hat{e}(R_L)$. Then, we have the following result.

Theorem 8: PMF of $\hat{e}(R_L)$ is given by

$$\hat{q}_i = \frac{1}{p_s}\sum_{j=y(i)}^{\infty}(1-p)^{j-y(i)}f_j p \quad (38)$$

where p_s is in (4), function $y(i)$ is

$$y(i) = \lceil i + 1 - 1/p - \omega \rceil \quad (39)$$

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

Proof: We first solve

$$R_L + 1/p - 1 - (1-p)^{R_L}/p = i \quad (40)$$

for R_L and express it in terms of i, i.e., $R_L = y(i)$, where $y(i)$ is given by (39), ignoring approximate roundoffs to the nearest integer. Combining with (5), we have

$$\hat{q}_i = P\left(R_L = y(i)\right) = h_{y(i)} \quad (41)$$

where h_i is in (5). This directly leads to (38). ■

Notice from (38) and (39) that distribution $\hat{q}_i$ does not even remotely approximate the original PMF f_i. This problem is fundamental since residual sampling exhibits bias toward 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 result.

Theorem 9: The flow-size distribution f_i can be expressed using the PMF of geometric residuals $\{h_i\}$ in (5) as

$$f_i = \frac{h_i - (1-p)h_{i+1}}{p + (1-p)h_1}. \quad (42)$$

Proof: From (5), we obtain that

$$h_i - (1-p)h_{i+1} = \frac{p}{p_s}f_i. \quad (43)$$

It then immediately follows that f_i is given by

$$f_i = \frac{p_s\left(h_i - (1-p)h_{i+1}\right)}{p}. \quad (44)$$

Notice that p_s in (4) 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

$$\sum_{i=1}^{\infty} h_i = 1 \quad \text{and} \quad \sum_{i=1}^{\infty} h_{i+1} = 1 - h_1. \quad (45)$$

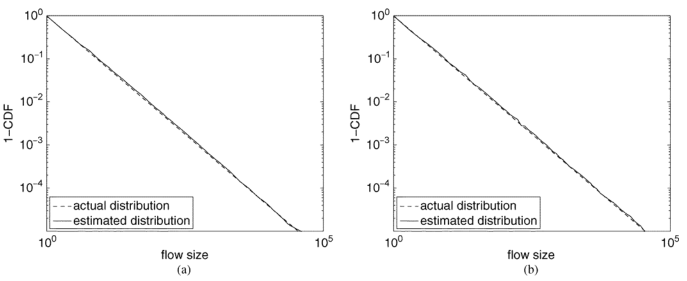


Fig. 7. Estimator (47) in simulations. (a) $p = 0.01$, $M = 194\,208$. (b) $p = 0.001$, $M = 26\,233$.

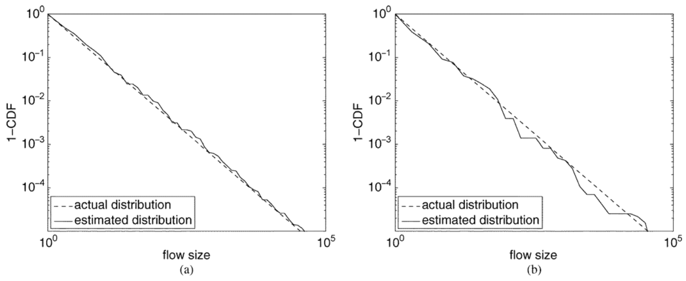


Fig. 8. Estimator (47) in simulations with very small p. (a) $p = 10^{-4}$, $M = 3090$. (b) $p = 10^{-5}$, $M = 337$.

Then, summing up both sides of (44) for i from 1 to infinity gives us

$$p_s = \frac{p}{\sum_{i=1}^{\infty}(h_i - (1-p)h_{i+1})} = \frac{p}{p + (1-p)h_1} \tag{46}$$

which together with (44) establishes (42). ■

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

$$\tilde{q}_i = \frac{M_i - (1-p)M_{i+1}}{Mp + (1-p)M_1} \tag{47}$$

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: Estimator (47) 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 (47) along with the actual distribution. The figure shows that $\tilde{q}_i$ accurately follows the true distribution for both cases of p.

C. Convergence Speed

We next examine the effect of sample size M, whose expectation depends on both p and the flow-size distribution

$$E[M] = np_s = nE\left[1 - (1-p)^L\right] \tag{48}$$

on the convergence of PMF estimator $\tilde{q}_i$. To illustrate the problems arising from small M, we study (47) with $p = 10^{-4}$ and 10^{-5} in simulations with the same 5M flows. The estimator obtained $M = 3090$ 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 increasing levels of noise as p reduces.

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 result follows from Theorem 9 and Corollary 2, indicating that the accuracy of $\tilde{q}_i$ directly depends on whether $\tilde{h}_i$ approximates h_i accurately.

Theorem 10: Suppose that $|\tilde{h}_j - h_j| \le \eta h_j$ holds with probability $1 - \xi$ for $j \in [1, i+1]$, where both η and ξ are in [0,1]. Then, there exists a constant

$$\zeta_i = \frac{\eta\left(p\left(h_i + (1-p)h_{i+1}\right) + 2(1-p)h_1 h_i\right)}{\left(h_i - (1-p)h_{i+1}\right)\left(p + (1-p)(1-\eta)h_1\right)} \tag{49}$$

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

Proof: Our goal is to find such ζ_i that guarantees $|\tilde{q}_i - f_i| \le \zeta_i f_i$. Applying (47) and expanding

$$(1 - \zeta_i)f_i \le \tilde{q}_i = \frac{\tilde{h}_i - (1-p)\tilde{h}_{i+1}}{p + (1-p)\tilde{h}_1} \le (1 + \zeta_i)f_i. \tag{50}$$

First, consider the right inequality and notice

$$\tilde{q}_i \le \frac{(1+\eta)h_i - (1-p)(1-\eta)h_{i+1}}{p + (1-p)(1-\eta)h_1} = \frac{A + \eta D}{B - \eta C} \tag{51}$$

where $A = h_i - (1-p)h_{i+1}, B = p + (1-p)h_1, C = (1-p)h_1$, and $D = h_i + (1-p)h_{i+1}$. We now need to solve

$$\frac{A + \eta D}{B - \eta C} \le (1 + \zeta_i)f_i = (1 + \zeta_i)\frac{A}{B}. \tag{52}$$

Using the fact that $B - \eta C$ is positive, this reduces to

$$\zeta_i \ge \frac{\eta(BD + AC)}{A(B - \eta C)}. \tag{53}$$

Now, returning to the left inequality in (50), we get

$$\tilde{q}_i \ge \frac{(1-\eta)h_i - (1-p)(1+\eta)h_{i+1}}{p + (1-p)(1+\eta)h_1} = \frac{A - \eta D}{B + \eta C}. \tag{54}$$

Similar to the previous case, we obtain

$$\zeta_i \ge \frac{\eta(BD + AC)}{A(B + \eta C)}. \tag{55}$$

Of the two lower bounds (53) and (54), the former is clearly larger. Expanding it, we obtain (49). ■

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

Theorem 11: 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

$$M \ge \frac{1 - h_i}{h_i \eta^2}\left(\Phi^{-1}(1 - \xi/2)\right)^2 \tag{56}$$

where $\Phi(x)$ is the cdf of the standard Gaussian distribution $\mathcal{N}(0,1)$ and $\Phi^{-1}(x)$ is its inverse.

Proof: Notice that M_i is a random variable whose distribution is given by Binomial(M, h_i) and that $\tilde{h}_i = M/M_i$ 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

$$Z = \frac{\tilde{h}_i - \mu_i}{\sigma_i} \tag{57}$$

and observe that it is a Gaussian random variable with mean 0 and variance 1. It then follows that

$$P(|Z| \le z) = 2\Phi(z) - 1 \tag{58}$$

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

$$P\left(|\tilde{h}_i - h_i| \le z\sigma_i\right) = 2\Phi(z) - 1. \tag{59}$$

We can guarantee the desired accuracy by setting $z\sigma_i = \eta h_i$ and $2\Phi(z) - 1 = 1 - \xi$, which simplifies to

$$\frac{\eta h_i}{\sigma_i} = \Phi^{-1}(1 - \xi/2). \tag{60}$$

Substituting $\sigma_i = \sqrt{h_i(1-h_i)/M}$ into the above equation and solving for M, we obtain (56). ■

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

$$M \ge \frac{(1 - 10^{-2}) \times 1.96^2}{10^{-2} \times 0.1^2} \approx 3.8 \times 10^4 \tag{61}$$

which indicates that $M = 38$K flows must be sampled to achieve this accuracy. If we reduce η to 1%, increase $1 - \xi$ to 99%, and require the approximation to hold for all $h_i \ge 10^{-3}$, then M must be at least 66M flows. Converting η into ζ_i using (49), 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 result.

Theorem 12: The expected number of flows with sampled residuals $R_L = i$ is

$$E[M_i] = E[M]h_i = nh_i p_{\mathrm{s}} \tag{62}$$

where h_i is the PMF of geometric residuals and p_{s} is in (4).

Proof: Writing

$$\begin{aligned} E[M_i] &= nP(A_L < L \cap R_L = i) \\ &= nP(R_L = i | A_L < L)P(A_L < L) \end{aligned} \tag{63}$$

notice that (62) follows from the fact that $P(R_L = i | A_L < L) = h_i$ and $P(A_L < L) = p_{\mathrm{s}}$. ■

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

$$\tilde{n} = M + \frac{1-p}{p} M_1 \tag{64}$$

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

$$\tilde{n}_i = \frac{M_i - (1-p)M_{i+1}}{p}. \tag{65}$$

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

Theorem 13: 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 $\mathrm{Var}[\tilde{n}/n] \to 0$ as $n \to \infty$. From (64), we have

$$E[\tilde{n}] = E[M] + \frac{1-p}{p} E[M_1]. \tag{66}$$

Applying (48) and (62), we get

$$E[\tilde{n}] = np_{\mathrm{s}} \left(1 + \frac{(1-p)h_1}{p}\right) \tag{67}$$

which simplifies to $E[\tilde{n}] = n$ using (46).

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

$$\mathrm{Var}\left[\frac{M}{n}\right] = \frac{1}{n^2} \sum_{j=1}^{n} \mathrm{Var}[A_j] = \frac{p_{\mathrm{s}}(1-p_{\mathrm{s}})}{n} \tag{68}$$

where the last term is bounded by $1/n$. Applying similar reasoning to M_1, we obtain that $\mathrm{Var}[\tilde{n}/n] \le 1/(np)$. Since we assumed that the number of sampled flows $M \to \infty$, this implies that $np_{\mathrm{s}} \to \infty$ and thus from (46) that $np \to \infty$, which establishes that $\mathrm{Var}[\tilde{n}/n] \to 0$. 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 (62), (46), and finally (42), we have

$$\begin{aligned} E[X_n] &= \frac{E[M_i] - (1-p)E[M_{i+1}]}{np} = \frac{p_{\mathrm{s}}(h_i - (1-p)h_{i+1})}{p} \\ &= \frac{h_i - (1-p)h_{i+1}}{p + (1-p)h_1} = f_i. \end{aligned} \tag{69}$$

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 $\mathrm{Var}[X_n] \to 0$ and $\mathrm{Var}[Y_n] \to 0$, 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 [19] provided a similar estimator as (64) 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 (64) and (65) are omitted for brevity.

VI. Steady-State Properties

We are now ready to discuss the various implementation details and run-time efficiency of URGE.

A. General Structure

For each incoming packet, URGE extracts the packet's flow ID x (e.g., the IPv4 five-tuple consisting of the protocol field, source/destination IP addresses, and both ports) and checks if the flow table has an existing entry for x. If so, the corresponding counter is incremented by 1; if not, a new entry is created for x with probability p, and its counter is initialized to 1. At the end of the measurement process, the collected counter values from the flow table are passed through URGE estimators to obtain the desired flow statistics.

The flow-counter table keeps a mapping between flow IDs and associated counters. This is a chained hash table of size K with three operations: 1) $\mathrm{lookup}(x)$ to retrieve the record of flow x; 2) $\mathrm{add}(x)$ to insert a new entry for flow x in the table with the initial counter value 1; and 3) $\mathrm{increment}(x)$ to add 1 to the counter of flow x. We assume that hash function $\mathrm{hash}(x)$ produces a uniformly random integer in $[0, 1, \ldots, K-1]$ and is fast enough to keep up (e.g., 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 linked 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 continuous removal of dead flows (i.e., either using FIN/RST packets or upon expiration of some idle timeout τ). Upon purging, flow information is saved to disk (single-flow usage) or aggregated into a RAM-based PMF table (flow-distribution usage). Operations $\mathrm{add}(x)$ and $\mathrm{increment}(x)$ automatically modify the timestamps associated with each flow and allow timeout-based expulsion of dead connections.

Notice that the flow table is accessed by residual-geometric sampling upon each packet arrival. Therefore, both scalability and overhead depend on the size of this data structure, which we investigate next.

B. Active Flows

Assume stationary flow arrivals and a measurement window $[1, T]$, where T is given in packets seen by the router. For each flow j with size $L_j \geq 2$, let interpacket 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. Note that Δ_j, which we assume is independent of L_j, is a measure of intraflow spareseness at the router (i.e., an inverse of its arrival rate). Denoting by $\Delta = E[\Delta_j]$ across all flows j, we have the following result.

Theorem 14: The expected number of active flows $N(t)$ seen by a packet arriving at time t is given by

$$E[N(t)] = (\Delta+1)\left(1 - \frac{1}{E[L]}\right). \tag{70}$$

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

$$E[N(t)] = nE[A_j(t)] = nP(A_j(t) = 1) \tag{71}$$

and notice that each flow j "exists" at the router for

$$Z_j = \sum_{k=1}^{L-1} (\Delta_j^k + 1) \tag{72}$$

packet units, where $\Delta_j^1, \Delta_j^2, \ldots$ are i.i.d. instances of variable Δ_j and 1 is added to each gap to account for the corresponding packet from flow j. The very first packet of each flow is excluded from this computation as the flow does not belong to the hash table yet and does not contribute to the processing cost of its own front packet.

Now, the probability that $t \in [1, T]$ lands within a given flow is simply the flow's expected footprint (in packets seen by the router) normalized by the window size

$$E[A_j(t)] = \frac{E[Z_j]}{T} = \frac{(\Delta+1)(E[L]-1)}{T}. \tag{73}$$

Since $T = nE[L]$, we get (70) using (71). ■

Our baseline reduction in flow volume comes from geometric sampling, which reduces the number of flows by a factor of $r_1 = n/E[M]$. Now, additionally define ratio

$$r_2 = \frac{n}{E[N(t)]} = \frac{T}{(\Delta+1)(E[L]-1)} \tag{74}$$

and notice that longer observation windows (i.e., larger T), smaller flow sizes (i.e., smaller $E[L]$), and more bursty 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. Denote by $M(t)$ the number of sampled flows that are currently alive at t and consider the next result.

Theorem 15: The expected number of flows seen in the hash table by a packet arriving at time t is given by

$$E[M(t)] = (\Delta+1)\left(1 - \frac{p_s}{pE[L]}\right) \tag{75}$$

where p_s in (4) is the probability to sample a flow.

Proof: Following Theorem 14, packet footprint Z_j is the number of arriving packets that see flow j in the hash table within window $[1, T]$. Define P_l as the number of interpacket gaps during which a flow of size l stays in the table (as before, the footprint does not include the first packet that triggers the router to sample the flow and add it to the table)

$$P_l = \begin{cases} R_l - 1, & \text{flow sampled} \\ 0, & \text{otherwise.} \end{cases} \tag{76}$$

Then, the flow's footprint Z_j, conditioned on $L_j = l$, is

$$\sum_{k=1}^{P_l} (\Delta_j^k + 1) \tag{77}$$

where as before Δ_j^k are i.i.d. interpacket delays induced by cross-traffic. Next

$$\begin{aligned} E[Z_j | L_j = l] &= E[P_l](\Delta+1) \\ &= E[R_l - 1 | \text{sampled}]P(\text{sampled})(\Delta+1). \end{aligned}$$

Using (9) and $P(\text{sampled}) = 1 - (1-p)^l$, we have

$$E[Z_j | L_j = l] = (\Delta+1)\left[l - \frac{1-(1-p)^l}{p}\right]. \tag{78}$$

Unconditioning L_j, we have the expected footprint as

$$E[Z_j] = (\Delta+1)\left[E[L] - \frac{p_s}{p}\right] \tag{79}$$

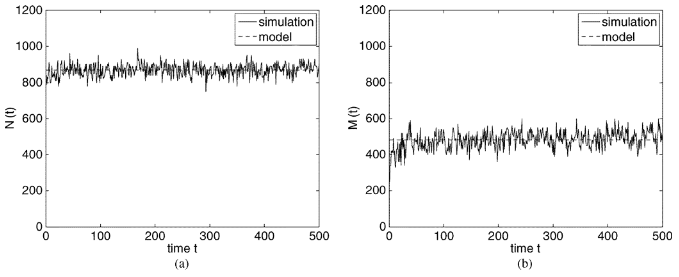


Fig. 9. Verifying models (70) and (75). (a) Live flows $N(t)$. (b) Sampled live flows $M(t)$.

TABLE I
COMPARISON OF (70) AND (75) TO SIMULATION RESULTS

time t	$E[N(t)]$		$E[M(t)]$	
	simulation	model (70)	simulation	model (75)
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

where $p_s = E[1-(1-p)^L]$ is the probability that a flow is sampled by the router. Multiplying $E[Z_j]$ by n and dividing by T, we get (75). ■

Performing a self-check, observe that $p_s \geq p$, and therefore $E[M(t)] \leq E[N(t)]$, which means that the former indeed always results in more reduction in table size than the latter. Also, notice that (75) reduces to (70) when $p = 1$. Define $r_3 = n/E[M(t)]$ as the expected reduction of space when tracking only active flows compared to all seen flows at the router, and notice that this ratio now also grows as p decreases. In fact, for small $p \to 0$, Taylor expansion of p_s shows that $E[M(t)] \approx (\Delta+1)pE[L^2]/(2E[L])$ is approximately a linear function of p, assuming of course that $E[L^2] < \infty$.

We evaluate models (70) and (75) in simulations with 1000 iterations through window $[1, T]$ 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. 9 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 1000 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 average number of bytes required by URGE in steady state is

$$E[W_R(t)] = Kw_p + E[M(t)](w_p + w_f). \tag{80}$$

From (80), observe that memory consumption can be reduced by lowering either $E[M(t)]$ or K. However, as discussed in the previous section, the number of sampled flows M cannot be arbitrarily small as it leads to lower estimation 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 $\text{lookup}(x)$ 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 $k = \text{hash}(x)$; 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 new entry for x is created with probability p and inserted to the location returned by $\text{lookup}(x)$. Notice that creating of the record in the fourth step is executed only when a new flow arrives and is sampled, which is much less frequent than continuous per-packet table lookups. Thus, we consider its contribution to the overall overhead negligible and omit it from the 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. It is not difficult to notice that $T_R(t)$ is directly determined by the expected depth of lookups in a hash table with $M(t)$ keys. Define $X_i(t)$ to be the number of items in bin i at time t, and recall that each chain is sorted by the key. Assuming the next lookup hits each bin with an equal probability, the expected lookup depth is $E[X_i(t)]/2$, as both successful and unsuccessful searches terminate on average after traversing half the chain. Since $X_i(t)$ is binomial with parameters $M(t)$ and $1/K$, we immediately obtain that the expected per-packet processing time is

$$E[T_R(t)] = t_h + t_p + (t_c + t_p)\frac{E[M(t)]}{2K}. \tag{81}$$

This result indicates that larger hash tables and smaller populations of sampled flows reduce the CPU overhead. Since larger K reduces (81), but increases (80), 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[1] and fixed. Define two constants

$$K_l = \frac{(t_c + t_p)E[M(t)]}{2(T_0 - (t_h + t_p))} \tag{82}$$

and

$$K_u = \frac{W_0 - E[M(t)](w_p + w_f)}{w_p}. \tag{83}$$

[1]In actual routers, $E[M(t)]$ can be predicted based on short-term and/or long-term history of the link.

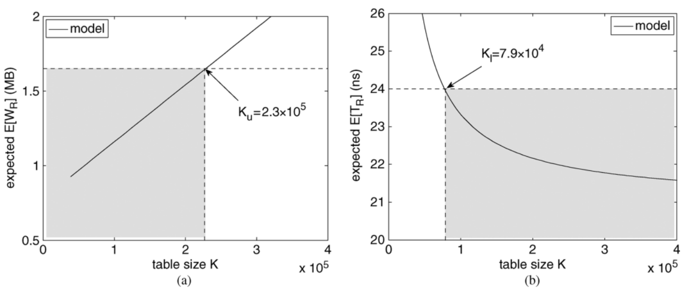


Fig. 10. Tradeoff: (a) memory consumption and (b) processing time with $E[M(t)] = 3.9 \times 10^4$. Gray areas display the acceptable ranges of K. (a) $E[W_R(t)]$. (b) $E[T_R(t)]$.

TABLE II
CONSTANTS USED IN (80) AND (81)

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

Assuming $K_l \leq K_u$, it then follows from (80) and (81) that one can choose any value $K \in [K_l, K_u]$ to satisfy the two 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 (80) and (81). We also impose the following constraints on memory and delay: $W_0 = 1.65$ MB and $T_0 = 24$ ns.[2] Fig. 10 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. 11 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 $[1, T]$ 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 $K_0 = 1.2 \times 10^5$.

An optimal strategy in practice would be to dynamically adjust $p(t)$ [e.g., using our model (75), or a closed-loop feedback controller for nonstationary arrival processes] so that the average table size $E[M(t)]$ stays around some constant, which is

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

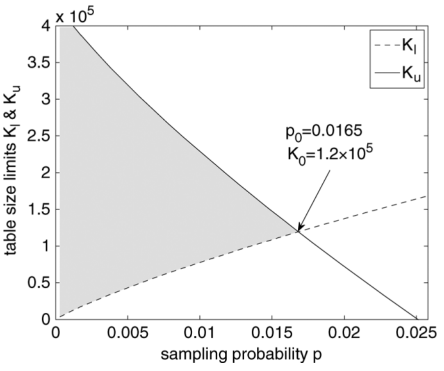


Fig. 11. Lower and upper bounds on table size K with varying probability p. Gray areas display the acceptable range of K and p.

large enough for $\tilde{q}_i$ to track any nonstationary effects in the flow population and overcome sampling noise, but not too large for URGE to become unreasonable in terms of consumed memory and CPU overhead. In fact, for stationary systems, this method allows $E[M(t)] \to 0$ as the observation window $T \to \infty$, which can be accomplished with any slowly decaying $p(t)$ that maintains $p_s(t) \to 0$ and $M = Tp_s/E[L] \to \infty$. In any case, the main problem becomes how to recover flow statistics from the data collected by RGS using time-varying $p(t)$. We leave this adaptive sampling for future work and proceed to performance analysis.

VII. EVALUATION

In this section, we evaluate our models using several Internet traces in Table III from NLANR [26] and CAIDA [5]. Trace FRG was collected over a period of 92.5 h from a gigabit link between UCSD and Abilene in 2006. We extracted from it auxiliary traces with only Web, DNS, and NTP flows (also seen in the table). Additionally, we use three datasets from CAIDA: LARGE, a 1-h trace from an OC48 link; MEDIUM, a 1-min trace from a OC192 link; and SMALL, a 7-min trace from a gigabit link.

As the table shows, URGE typically sees a reasonably large number of flows M over the entire interval $[1, T]$. However, the number of active flows $N(t)$ and those constantly kept in memory $M(t)$ is much smaller. For example, in the FRG trace, $E[M]$ is 15 times smaller than n, while $E[N(t)]$ is 81 and $E[M(t)]$ is 658 times smaller.

It should be noted that FRG benefits more from removal of dead flows than CAIDA, partially because the former was collected over four consecutive days and thus had a significantly larger observation window T. The same reasoning can help explain the fact that the LARGE trace exhibits much higher r_2 than the MEDIUM or SMALL traces. However, determination of the exact underlying reasons for these phenomena requires more intricate parameters of arriving traffic (e.g., interpacket gap Δ_j, its correlation with flow size L_j, nonstationary properties of $M(t)$), which are beyond our scope.

A. Memory and Speed

We use the settings of Table II to compute the amount of memory consumed by URGE according to (80). As shown in the third column of Table IV for $p = 0.001$ and $K = E[M(t)]$, the average required memory size is small and rarely exceeds 40 kB.

TABLE III
RESIDUAL SAMPLING WITH $p = 0.01$

source	trace	total flows n	total pkts $nE[L]$	sampling only		removal only		both	
				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%

Even for the LARGE trace that has the most flows in this comparison, the expected RAM utilization of URGE is only 132 kB, much smaller than roughly 120 MB required for all flow counters. Note that while our model provides the mean RAM usage, it may be wise in practice to equip routers with enough memory to hold several times $E[M(t)]$ to account for random fluctuation in the flow population. The specific overprovisioning factor can be determined by administrators using the history of the link and/or other models that describe the flow-arrival process of the network.

We also compute per-packet processing time from (81) 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. If $M(t)$ suffers from prolonged periods of highly variable fluctuation (e.g., in nonstationary systems), then it is also advisable to design the processing capabilities of the router to handle the worst-case $M(t)$ based on the knowledge of its distribution and diurnal fluctuation patterns.

B. Estimation Accuracy

First, we examine the problem of estimating the total number of flows n in $[1, T]$ and size-one flows n_1 in this interval. The seventh and tenth columns of Table IV list the absolute error of models (64) and (65), 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. 12 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. 13, 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. 14 indicates that estimation for all three values of

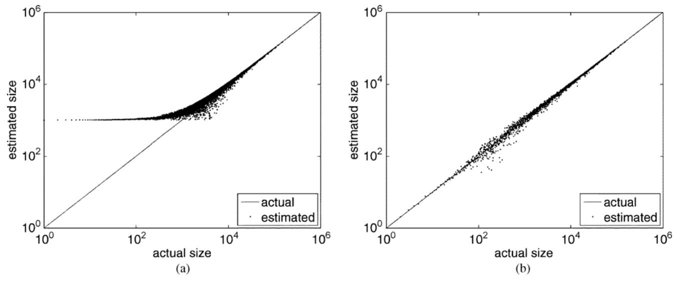


Fig. 12. Estimating single-flow usage in the FRG trace with $p = 0.001$. (a) $E[e(R_l)]$. (b) $E[\hat{e}(R_l)]$.

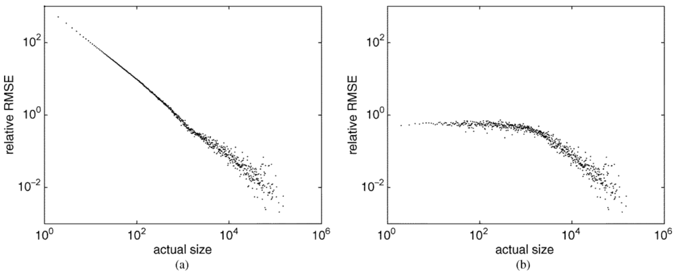


Fig. 13. RRMSE of single-flow usage in the FRG trace with $p = 0.001$. (a) δ_l. (b) $\hat{\delta}_l$.

p are very consistent and follow the actual distribution very well. In our experiments with $p = 0.0001$, URGE recovered the original PMF $\{f_i\}$ using only $M = 7616$ total flows out of $n = 1.75$M.

We next apply URGE with $p = 0.001$ to NLANR traces of different traffic types and plot in Fig. 15 the estimated distributions along with the actual ones. As the figure shows, URGE estimates flow statistics of different applications/protocols very accurately as well. A similar match is observed in our experiments with the three CAIDA traces as shown in Fig. 16.

It should be finally noted that residual sampling and URGE have applications beyond traffic monitoring, e.g., detection of most frequently referenced objects (i.e., "elephants") in streaming databases, where L_i can be viewed as the number

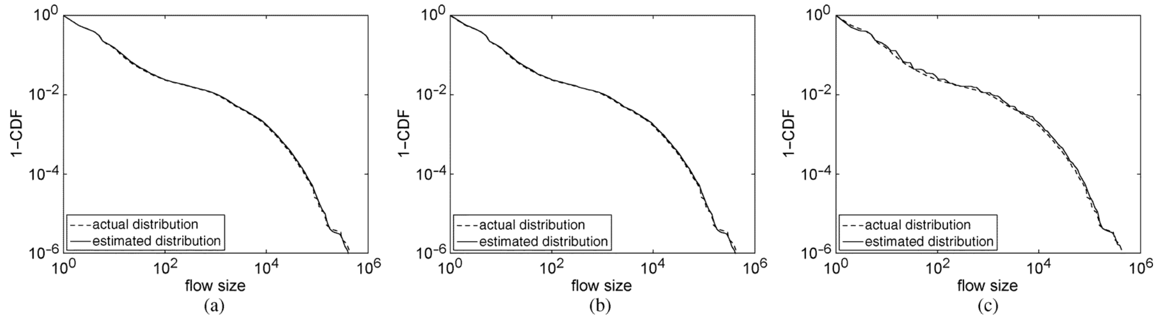


Fig. 14. Estimating the flow-size distribution using URGE in the FRG trace. (a) $p = 0.01$. (b) $p = 0.001$. (c) $p = 0.0001$.

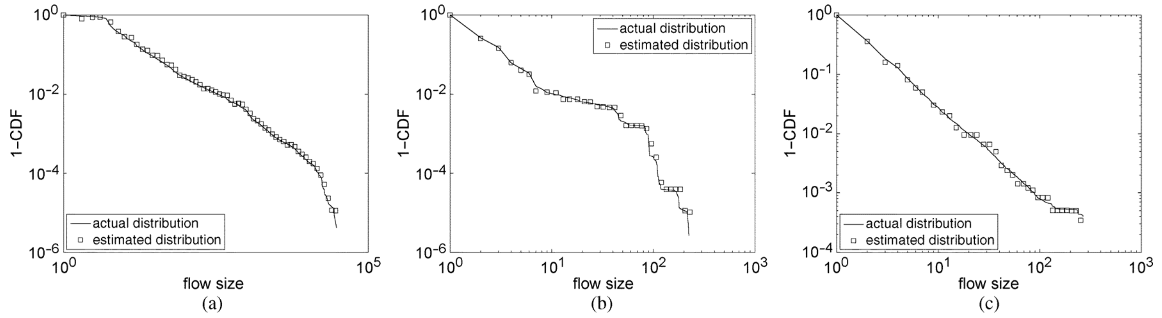


Fig. 15. Estimating the flow-size distribution using URGE in NLANR traces with $p = 0.001$. (a) Web. (b) NTP. (c) DNS.

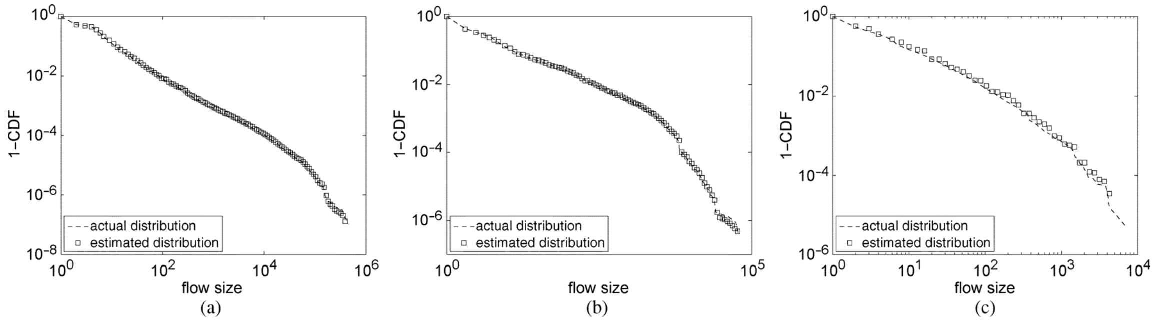


Fig. 16. Estimating the flow-size distribution using URGE in CAIDA traces with $p = 0.001$. (a) LARGE. (b) MEDIUM. (c) SMALL.

of times item i appears in the stream, and simultaneously constructing the distribution of object popularity $\{f_i\}_{i=1}^{\infty}$.

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.

Future work includes developing mechanisms that adjust sampling probability $p(t)$ based on varying traffic conditions and designing algorithms to recover flow statistics from such adaptive sampling.

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. Pacific Grove, CA: Duxbury/Thomson Learning, 2002.

[3] Cisco, San Jose, CA, "Cisco NetFlow," accessed 2008 [Online]. Available: http://www.cisco.com/en/US/products/ps6601/prod_white_papers_list.html

[4] Cisco, San Jose, CA, "Cisco sampled NetFlow," 2003 [Online]. Available: http://www.cisco.com/en/US/docs/ios/12_0s/feature/guide/12s_sanf.html

[5] CAIDA, La Jolla, CA, "Cooperative Association for Internet Data Analysis (CAIDA)," [Online]. Available: http://www.caida.org/

[6] R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth, "On the lambert W function," *Adv. Comput. Math.*, vol. 5, pp. 329–359, 1996.

[7] 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.

[8] N. Duffield, C. Lund, and M. Thorup, "Charging from sampled network usage," in *Proc. ACM IMW*, Nov. 2001, pp. 245–256.

[9] N. Duffield, C. Lund, and M. Thorup, "Estimating flow distributions from sampled flow statistics," in *Proc. ACM SIGCOMM*, Aug. 2003, pp. 325–336.

[10] N. Duffield, C. Lund, and M. Thorup, "Flow sampling under hard resource constraints," in *Proc. ACM SIGMETRICS*, Jun. 2004, pp. 85–96.

[11] N. Duffield, C. Lund, and M. Thorup, "Learn more, sample less: Control of volume and variance in network measurement," *IEEE Trans. Inf. Theory*, vol. 51, no. 5, pp. 1756–1775, May 2005.

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

[13] C. Estan and G. Varghese, "New directions in traffic measurement and accounting," in *Proc. ACM SIGCOMM*, Aug. 2002, pp. 323–336.

[14] W. Fang and L. Peterson, "Inter-AS traffic patterns and their implications," in *Proc. IEEE GLOBECOM*, Dec. 1999, pp. 1859–1868.

[15] 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.

[16] N. Hohn and D. Veitch, "Inverting sampled traffic," in *Proc. ACM IMC*, Oct. 2003, pp. 222–233.

[17] 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.

[18] 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 Symp.*, May 2007, pp. 25–30.

[19] R. R. Kompella and C. Estan, "The power of slicing in internet flow measurement," in *Proc. USENIX/ACM IMC*, Oct. 2005, pp. 105–118.

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] Y. Lu, A. Montanari, B. Prabhakar, S. Dharmapurikar, and A. Kabbani, "Counter braids: A novel counter architecture for per-flow measurement," in *Proc. ACM SIGMETRIC*, Jun. 2008, pp. 121–132.

[25] 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.

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

[27] R. Pan, L. Breslau, B. Prabhakar, and S. Shenker, "Approximate fairness through differential dropping," *Comput. Commun. 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, Dec. 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] Wolfram-Alpha, "Computational knowledge engine," [Online]. Available: http://www.wolframalpha.com/

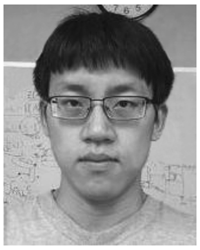
[33] L. Yang and M. Michailidis, "Sampled based estimation of network traffic flow characteristics," in *Proc. IEEE INFOCOM*, May 2007, pp. 1775–1783.

[34] 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.

Xiaoming Wang (S'04) received the B.S. degree in computer science and the M.S. degree in electronic engineering from Beijing University of Posts and Telecommunications, Beijing, China, in 1999 and 2002, respectively, and the Ph.D. degree in computer science from Texas A&M University, College Station, in 2009.

He currently works for Amazon.com, Seattle, WA. His research interests include peer-to-peer systems, probabilistic analysis of computer networks, and topology modeling.

Dr. Wang has been a student member of the Association for Computing Machinery (ACM) since 2007.



Xiaoyong Li received the B.S. degree in computer engineering and M.S. degree in electrical engineering from Beijing University of Posts and Telecommunications, Beijing, China, in 2005 and 2008, respectively, and is currently pursuing the Ph.D. degree in computer science at Texas A&M University, College Station.

His research interests include flow sampling, information retrieval, and Web crawling.

Dmitri Loguinov (S'99–M'03–SM'08) received the B.S. degree (with honors) from Moscow State University, Moscow, Russia, in 1995, and the Ph.D. degree from the City University of New York, New York, in 2002, both in computer science.

He is currently a Professor with the Department of Computer Science and Engineering, Texas A&M University, College Station. His research interests include P2P networks, information retrieval, congestion control, Internet measurement, and modeling.