Tracking Cardinality Distributions in Network Traffic

Aiyou Chen, Li Li and Jin Cao
Bell Labs, Alcatel-Lucent
600 Mountain Ave, Murray Hill, NJ, USA
Email: [aychen, erranlli, cao]@research.bell-labs.com

***Abstract*—Understanding the aggregate behavior of network host connectivities is important for network monitoring and traffic engineering. One characterization of such an aggregate behavior is the host distributions of distinct communicating peers or flows. For example, during the worm outbreak, the port scanning activities would cause many hosts with increasing number of (one-way) peers (or flows), and hence a change in the host distributions of distinct communicating peers or flows. In this paper, we develop an efficient streaming algorithm for tracking these host distributions of distinct elements, also called cardinality distributions, for a high speed network with a large number of hosts. Our approach utilizes the continuous Flajolet-Martin sketches, which is the minimal order statistics of hashed values, as a compact data summary and develops maximum likelihood estimates of these distributions. By leveraging the aggregation of many hosts, we are able to obtain very accurate estimates of the cardinality distributions by maintaining a compact statistical summary that is as small as one number (at most 32 bits) per host. Extensive experimental studies are carried out to demonstrate their excellent performance.**

***Index Terms*—Network Measurement, Data Streaming, Cardinality Distribution, Mixture model, Expectation-Maximization.**

I. Introduction

Network traffic analysis has become increasingly important for various network management functions such as traffic engineering and anomaly detection and response. Due to the high traffic volume in the high speed networks, it is extremely useful to derive succinct summary information that can characterize the traffic behavior pattern as a whole. One of the most useful characterization of the aggregate behavior pattern is the network feature distributions. Prior work has focused primarily on distributions concerning traffic volume. For example, the flow size distribution (given a flow size s, find the number of flows with size s) is studied in [12], [23], [28], where a flow is a sequence of packets that share the same five tuples of (source IP, port, destination IP, port and protocol), and the inverse distribution of packet contents (given n, find out the number of strings with frequency n) is studied in [21]. Distributional aspects such as entropy (e.g. entropy of the packet distribution over various ports) has also been a subject of current interest [6], [24], [25].

Despite many work on feature distributions concerning the traffic volume, little attention has been paid on traffic feature distributions involving distinct counts, such as the number of destinations or flows for each host. These distributions are very useful for characterizing the communication *connectivity patterns* between hosts inside a network and across the Internet, which are not captured by the volume data. Understanding such a pattern is in no doubt useful for network service providers to manage their network more efficiently.

On the traffic engineering side, if the number of peers for many hosts increases over time, this may indicate that the number of peer-to-peer (P2P) hosts is on the rise, which may further alert the network provider to improve its traffic engineering solution for the P2P traffic [26]. Statistically, the distribution of number of peers per host may have changes in its mode in such scenario: a new mode may appear for the common number of peers that the P2P hosts are communicating with. On the anomaly detection side, if the number of peers for many hosts has a sudden increase, this may indicate attack activities such as port scans. The distribution will have a shift in its mode. Unfortunately, such distributional changes cannot be easily detected using marginal aspects such as entropy, mean or variance. For example, a shift in the mean of a distribution with no shape change will not change the entropy. Good estimates of the distributions in real time will be necessary for capturing all such changes. Besides estimating the cardinality distribution for all hosts going through a high speed provider router, or all hosts inside a stub network, one may also specifically monitor the cardinality distribution for each group of IP addresses. For example, in Botnet detection, once the set of candidate bot controllers are identified [22], there is a need to monitor their behavior. One may want to monitor the cardinality distribution of the peers of each candidate controller. This distribution can identify whether many of them are actively "working for" the bot controller. New attacks will result in changes of the cardinality distribution.

In this paper, we are interested in estimating the aforementioned distinct count, or *cardinality*, distributions in network traffic: given a number n, how many IP addresses communicate with n different destinations or has n number of flows as observed in a network. Unfortunately, methods developed for estimating traffic volume distributions do not directly apply here, simply because the traffic volume and cardinality are intrinsically different quantities: the traffic volume is additive, but the cardinality is not. As a result, it is easy to compute individual volumes, but hard to compute individual cardinalities. For example, to obtain the flow count for a host, one needs to build either a hash table, bitmap or Bloom filter that keeps track of existing flows to avoid duplicate flow records resulting from packets from the same flow. Let T be the total number of flows of all hosts under study, then these straight-forward counting methods would consume $O(T)$ memory space which is too costly.

Instead of the above counting methods for individual hosts, another solution is to derive the distribution from approximate counts using distinct counting algorithms such as the well-known probabilistic counting that employs Flajolet-Martin (FM) sketches [17], or sampling based methods such as [7], [16]. Using the Super-Loglog counting algorithm [2], an improved version of probabilistic counting, it is shown that for a relative error ϵ, one needs a memory space of about $1/\epsilon^2 \log_2 \log_2(n)$ sketches for a host with cardinality n. For example, for a host with cardinality $n = 1,000,000$, to achieve $\epsilon = 0.05$, even with 5 bits per sketch, this would imply about 10,000 bits. Furthermore, notice that the design needs to be uniform over all hosts, since one does not know whether any particular host has a small cardinality or not before the end of the traffic stream. Although $10,000$ bits seems small for an individual host, it can become formidable when we multiply by the number of hosts m, which can be large for a high speed network. For example, when m is 100,000, the total memory would be 1G bits of SDRAM or 100Mb even if only 10% hosts are sampled. Needless to say, using this approach, for a network with many hosts, we cannot afford to count their cardinalities accurately.

In this paper, we develop an efficient solution for estimating the cardinality distributions. Our solution does not rely on counting the cardinalities of individual hosts accurately, but rather, we aggregate the information of individual cardinalities in an optimal way using statistical inference. Note that tracking individual host cardinalities accurately is sufficient but not *minimal sufficient* (see [3]) for obtaining accurate distribution estimation. The key insight of our solution lies in the fact that for a high speed network with many hosts, the minimal sufficient statistics for distribution estimation (or histogram) are groups of hosts whose cardinalities are close to each other (i.e. within bins). Our solution uses the FM sketches as summary statistics, but we only need to maintain one random number (at most 32 bits) per host when the number of hosts is large. The closeness of cardinalities is of course not observable, but is probabilistically equivalent to the closeness of the corresponding hosts' FM sketches which can be 'grouped' from statistical inference. We develop a maximum likelihood estimate of the cardinality distribution of network traffic, and analyze its accuracy. Most importantly, our algorithm is a computationally efficient *online streaming* algorithm, thus relieving the burden of storing large volume of traffic data. For the prior example, we only need one sketch per host and thus the total memory is at most 3.2Mb as opposed to 1Gb, or 0.32Mb as opposed to 100Mb if only 10% hosts are sampled. In fact, our algorithm achieves very good accuracy in both simulation and trace-based evaluation. Even with 10 sketches per host, the straightforward application of sketches can still have an error more than our algorithm. Therefore, the space requirement of our algorithm is at least an order of magnitude smaller than the straightforward approach.

Our major contribution can be summarized as follows. We propose a simple summary statistics (minimum order statistics) and the first algorithm for estimating the cardinality distributions of a large number of hosts in a high speed network. Maximum likelihood estimate of the distribution is obtained with discretized bins. The algorithm only requires updating one number (at most 32 bits) for each host and thus is both computationally and memory efficient. Simulation and trace studies are carried out to demonstrate its effectiveness in network monitoring under a variety of conditions such as anomaly detection.

A 2-page extended abstract of our preliminary results has appeared in ACM SIGMETRICS'2008 [9].

A. Outline of the Paper

The structure of the paper is as follows. In Section II, we propose to use the continuous FM sketches as the summary statistics which can be updated in real time with cheap operation and describe the log scale histogram model for the cardinality distribution. In Section III, we develop the algorithm for estimating the distribution with discretized bins and describe how to compute confidence intervals. Simulation studies and comparison with a naive approach are carried out in Section IV. Trace studies are carried out in Section V to demonstrate the effectiveness of our approach in anomaly detection and identifying connectivity patterns in a large corporate network and a university network.

II. Data streaming using sketches

In this section, we first describe our system model. We then describe the continuous version of FM sketches [8], [17], [19] that our statistical model is based on. Finally, we derive statistical models of cardinality distribution from the sketches.

A. System Model

Our overall architecture consists of two modules: the on-line streaming module and the statistical estimation module. The on-line streaming module is updated upon each packet arrival. The estimation module proceeds in epochs. At the end of each measurement epoch, the sketch values for all hosts collected will be passed to the estimation module. The estimation module will produce an accurate estimate of the cardinality distribution using maximum likelihood.

B. Continuous FM Sketch

Below we describe the continuous Flajolet-Martin sketches for estimating the distribution of N, where N is the number of distinct peers (as a working example) that a randomly chosen host communicates with. For simplicity, we assume that a medium size network (say 10,000 hosts) is under monitoring, and that we can only afford recording one number, say Y, for each of its hosts. For a very large size network, one can maintain the sketches for uniformly sampled hosts. We are only interested in hosts that meet certain criteria (denoted as t) from the data stream $\mathcal{T}$ (e.g. sampled hosts). Let g be a universal hash function that maps an IP pair to a uniform random number in [0,1]. Given the criteria filter function t, universal hash functions g, and the number of hosts m, the on line streaming module is summarized in Algorithm 1. We assume that there are a set of at most m hosts of interest that can pass the filter function t. Let Y_i, initialized with 1, be the number associated with host i. For each packet arrival, let e denote the (host,peer) IP pair, and update the record of the host, say i, by

$$Y_i \quad = \quad \min(Y_i, g(e)), \tag{1}$$

where g is a uniform random number generator using seed e. At the end of a measurement epoch, Y_i is the minimum of N_i uniform random numbers associated with host i, which has N_i distinct peers. In high probability, the larger N_i is, the smaller Y_i is. Thus the magnitude of Y_i provides information for N_i.

Algorithm 1 Algorithm for updating sketches

```
1: Initialize a hash array $Y$ of size $m$ with values 1.
2: for each incoming packet with IP pair $(s,d)$ of $\mathcal{T}$ do
3:     If $t(s) = 1$, update $Y[s]$ by $\min(Y[s], g(s,d))$
4:     If $t(d) = 1$, update $Y[d]$ by $\min(Y[d], g(d,s))$
5: Return $Y$ at the end of a measurement epoch.
```

We call (1) the continuous Flajolet-Martin (FM) sketch, because the classical Flajolet-Martin sketch uses a geometrical distribution which is discrete. The purpose of using 'continuous' is to keep technical complexity to minimal. In actual implementation, discrete random numbers generated by universal hash are used. The error introduced by discretization has been studied and found to be ignorable according to [8].

C. A discrete model

To estimate the cardinality distribution for a network, one can of course estimate the cardinality of each host individually, using existing probabilistic counting algorithms, e.g. [13] or [1], [2], [18]. As an example of them, we call the following one as the naive FM approach for estimating the cardinality distribution: 1) use a few FM sketches for each host that record the minimal statistics independently, 2) estimate the cardinality of each host using the corresponding FM sketches, 3) compute the histogram as an estimate of the cardinality distribution. We will show later that this approach is not efficient to derive a good estimate. We next describe a log scale histogram model for the cardinality distribution, denoted as F. The estimation approach and analysis follow in next sections.

Since the goal is to estimate the cardinality distribution, say $N_i \sim F$, the hosts are considered independent and thus the cardinality of each host can be treated as a random sample from F.

Notice that the cardinality takes values of positive integers, and its distribution usually has heavy tails (see extensive distribution examples from trace studies in later sections). For simplicity, we model F using histogram in the $\log_2$ scale, that is, we assign bins to

$$1, \{2,3\}, \quad \cdots, \quad \{2^K, 2^K+1, \cdots, 2^{K+1}-1\}$$

where 2^{K+1} is assumed to be the upper bound of the cardinalities. One can truncate the right tail into one bin if it is more than 2^{K+1}. Then assign weight p_k to bin $\{2^k, \cdots, 2^{k+1}-1\}$, for $k = 0, \cdots, K$, where $\sum_{k=0}^{K} p_k = 1$. In other words, the log scale histogram model is:

$$P(N \in \{2^k, \cdots, 2^{k+1}-1\}) \quad = \quad p_k \tag{2}$$

with a total of $K+1$ bins. Notice that it is not possible to track the probabilities of all values within each bin accurately. As an approximation, we do not differentiate the probabilities at integers within each bin, and they are modeled with equal probabilities, i.e. for $j \in \{2^k, \cdots, 2^{k+1}-1\}$,

$$P(N = j) \quad = \quad 2^{-k} p_k.$$

Thus the cardinality distribution F can be characterized by parameters $p_0, \cdots, p_K$. Let $\mathbf{p} = (p_0, \cdots, p_K)^T$, then $\mathbf{p}$ falls onto the simplex space $p_k \geq 0$ and $\sum p_k = 1$. Since this simple model is only an approximation to F, estimates of $\mathbf{p}$ may be biased under this model. However, this simple model is a reasonable approximation and allows us to estimate the probability of each bin quickly and accurately as we will show later.

III. Statistical Inference

In this section, we develop the Maximum Likelihood Estimate (MLE) of the cardinality distribution under the statistical models discussed in Section II-C. We first derive the log-likelihood function of the unknown parameter $\mathbf{p}$, and a computationally elegant EM algorithm for obtaining the MLE. We then construct the confidence intervals of the estimate.

A. Maximum likelihood Estimation via EM Algorithm

By the Fisher information theory, for a model with finite parameters, an estimate that maximizes the (logarithmic) likelihood function (MLE) with given data is most efficient under regular conditions [3]. Below we derive the likelihood function of the histogram parameters $\mathbf{p}$ given the continuous FM sketches $Y_1, \cdots, Y_m$ associated with m hosts. We list the following simple fact for technical convenience.

Lemma 1: Let $Y = \min(U_1, \cdots, U_n)$, where $U_1, \cdots, U_n$ are n independent uniform random numbers. Then $-\log(1 - Y)$ follows an exponential distribution with mean $\frac{1}{n}$.

Following [8], we will use the transformed values $-\log(1 - Y_i)$, for simplicity still denoted as Y_i, $i = 1, 2, \cdots, m$, to estimate the distribution F. Based on Lemma 1, we have that for the ith host, there exists a unit exponential random number ϵ_i such that

$$Y_i \quad = \quad \frac{\epsilon_i}{N_i},$$

where N_i is the cardinality of the i-th host. To be convenient, we omit the subscript i of Y_i, ϵ_i and N_i whenever there is no confusion.

From the above, the tail probability function of Y can be written as:

$$P(Y \geq y) \quad = \quad E\left[e^{-yN}\right] \tag{3}$$

Using the log scale histogram model, after some calculus, we have,

$$\begin{aligned} E\left[e^{-yN}\right] &= p_0 e^{-y} + \sum_{k=1}^{K} \frac{p_k}{2^k} \sum_{i=2^k}^{2^{k+1}-1} e^{-yi} \\ &= p_0 q + \sum_{k=1}^{K} \frac{p_k}{2^k} \times \frac{q^{2^k}(1-q^{2^k})}{1-q}, \end{aligned}$$

where $q = e^{-y}$. Define for $k = 0, 1, \cdots, K$, and $y > 0$,

$$f_k(y) \quad = \quad \frac{q^{2^k}(1-q^{2^k})}{2^k(1-q)}$$

and let $\mathbf{f}(y) = (f_0(y), \cdots, f_K(y))^T$ be the column function vector. Then,

$$E\left[e^{-yN}\right] \quad = \quad \mathbf{p}^T\mathbf{f}(y).$$

Now the probability density function (PDF) of Y can be obtained from the derivative of the tail probability function, $1 - P(Y \geq y)$. This is summarized as follows.

Theorem 1: Given the log scale histogram model (2) with parameters $\mathbf{p}$ for the distribution of N, the probability density function of Y can be written as

$$p_Y(y, \mathbf{p}) \quad = \quad \mathbf{p}^T\overline{\mathbf{f}}'(y), \tag{4}$$

where $\overline{\mathbf{f}}'(y) = -(f_0'(y), \cdots, f_K'(y))^T$, and

$$f_k'(y) \quad = \quad -\left\{\frac{q^{2^k}(1-2q^{2^k})}{1-q} + \frac{2^{-k}q^{2^k+1}(1-q^{2^k})}{(1-q)^2}\right\} \tag{5}$$

is the derivative of $f_k(y)$. That is, Y follows a mixture distribution with component density functions $\overline{\mathbf{f}}'(y)$.

The proof follows from basic calculus and is omitted. It can be verified that each component of $\overline{\mathbf{f}}'(y)$ is nonnegative and the integral of each component function for $y \geq 0$ is 1, and thus each component of $\overline{\mathbf{f}}'(y)$ is in fact a probability density function, which validates the result of the above theorem. Note that (4) is a linear function of these density functions, which implies that Y follows a mixture distribution.

Given observations $(Y_1, \cdots, Y_m)$ for m hosts respectively, the MLE of $\mathbf{p}$ can be defined as below:

$$\hat{\mathbf{p}} \quad = \quad \arg\max_{\mathbf{p}} \frac{1}{m} \sum_{i=1}^{m} \log(\mathbf{p}^T\overline{\mathbf{f}}'(Y_i)). \tag{6}$$

There is no closed form solution to the above optimization. It can be seen that this is a convex optimization problem with constraints. Standard Primal-Dual interior-point type algorithms can be used to solve the optimization when $K+1$ is small. However, when $K+1$ is large, the convergence becomes unstable. In our experiences of the optimization with constraints routine (fmincon) in MATLAB, the convergence becomes very unstable even for $K+1$ as small as 13, which of course also depends on the input data of Y_i. Below we develop an efficient algorithm where each step is a closed-form iteration. This is motivated by the fact that the Y distribution belongs to the parametric family of mixture models, the MLE of whose parameters can be obtained conveniently using the Expectation-Maximization algorithm [11]. We summarize the result in the folling Theorem, whose proof is included in the Appendix for completeness.

Theorem 2: Let $\mathbf{a}_i = \overline{\mathbf{f}}'(Y_i)$. The objective function $\sum_{i=1}^{m} \log(\mathbf{p}^T\mathbf{a}_i)$ is convex, and it is strictly convex and thus has a unique maximizer on the simplex space unless that the number of distinct values of Y_is is less than $K+1$. The maximizer can be obtained by using the following iteration:

$$\mathbf{p} \quad \leftarrow \quad \frac{1}{m} \sum_{i=1}^{m} \frac{\mathbf{p} \cdot \mathbf{a}_i}{\mathbf{p}^T\mathbf{a}_i}, \tag{7}$$

where $\mathbf{p} \cdot \mathbf{a}$ is a column vector defined by component-wise products of $\mathbf{p}$ and $\mathbf{a}_i$, no matter what starting point on the simplex is used.

The MLE of $\mathbf{p}$, i.e. $\hat{\mathbf{p}}$, is thus obtained by the iteration algorithm described in the above Theorem. Though EM converges slowly with a linear speed, based on our extensive numerical studies, 100 iterations usually lead to satisfactory results. We use the above algorithm to compute the MLE of the distribution parameters $\mathbf{p}$. We note that there have been some recent work on EM algorithms that can significantly accelerate its convergence, see [20] and references therein for details.

In the next subsection, we describe how to obtain the confidence interval for the estimate that calibrates the performance of the method.

B. Confidence Intervals of the Cardinality Distribution

There are two ways to characterize the performance of the estimate. One is through asymptotic theory [3] and the other through simulation methods using bootstrap [14]. The following theorem characterizes the asymptotic error distribution. This provides a way to compute the confidence interval of the parameter values for large sample size m. Note that there are only K free parameters since $p_0 = 1 - \sum_{k=1}^{K} p_k$. Let $\theta = (p_1, \cdots, p_K)^T$.

Theorem 3: Assume that the MLE estimate of θ, say $\hat{\theta}$, is asymptotically consistent. Then it has an error of order $O(m^{-1/2})$ and is asymptotically normal, i.e. as $m \to \infty$,

$$\sqrt{m}\left(\hat{\theta} - \theta\right) \quad \to \quad \mathcal{N}\left(0, \mathbf{I}^{-1}\right),$$

where $\mathcal{N}\left(0, \mathbf{I}^{-1}\right)$ is the multi-variate Gaussian distribution with zero means and covariance matrix $\mathbf{I}^{-1}$. Here $\mathbf{I}$ is the Fisher information matrix, equal to $E\left[\frac{\partial L}{\partial \theta}\frac{\partial L}{\partial \theta^T}\right]$, and can be estimated consistently by

$$\hat{\mathbf{I}} \quad = \quad \frac{1}{m} \sum_{i=1}^{m} \frac{\mathbf{b}(Y_i)\mathbf{b}^T(Y_i)}{(\mathbf{p}^T\overline{\mathbf{f}}'(Y_i))^2},$$

where $\mathbf{b}(y) = (b_1(y), \cdots, b_K(y))^T$ with $b_k(y) = f_0'(y) - f_k'(y)$. Thus for large m, the 95% confidence interval for p_k is given by $\hat{p}_k \pm \frac{1.96}{\sqrt{m(\hat{\mathbf{I}}^{-1})_{kk}}}$, for $k = 1, \cdots, K$.

We note that if m is not large enough relative to the number of parameters K, the confidence interval obtained in the above theorem may be inaccurate, because the inverse of an approximate $K \times K$ Fisher information matrix can be very unstable especially when the number of bins, i.e. $(K+1)$, is large. In such cases, we can use the bootstrap method to estimate the standard deviation and then obtain the confidence interval. The parametric bootstrap method works as follows (see details in [14]).

First, repeat the following many times, say 1000 times: 1) generate the number of peers from the estimated cardinality distribution with $\hat{\mathbf{p}}$, 2) compute the continuous FM sketches, and 3) estimate the weight parameters of the log scale histogram model for the cardinality distribution. These 1000 replicates give us a sampling distribution of the parameter estimates. Second, compute the 2.5% and 97.5% quantiles from the sampling distribution of each parameter which give the 95% confidence interval.

Statistical theory [3], [14] tells that the bootstrap can estimate standard errors for complex models in general more accurately than traditional methods based on Fisher information matrix.

IV. SIMULATION

In this section, we use simulations to evaluate our algorithm and demonstrate its excellent performance by reporting confidence intervals. In order to investigate the generality of the results, we have tested our algorithms on a variety of cardinality distributions that have heavy tails and one or more modes, and we have compared it with the naive FM approach mentioned earlier. The simulation results validate our claim in Section II-C. Due to space limitation, we only report the simulation case with a Pareto (power law) distribution [1] since it is most popularly used for network traffic modeling.

A. Algorithm Evaluation

We simulate $m = 10,000$ hosts and the number of peers for each host is generated from the Pareto distribution with lower bound 1 and α-index 1, rounded into integers. Roughly speaking, 50% of the hosts have only one communication peer, 25% of the hosts have 2 to 3 peers, 12.5% of the hosts have 4 to 7 peers, and so on. The black solid line in Figure 1 shows that the Pareto distribution after binning follows nearly a straight line in the $\log_2$ vs $\log_2$ scale. Note that the probability is also converted by $\log_2$ scale in order to see the linear property.

The continuous FM sketches are obtained according to (1) described in Section II. That is. for each host, each (host,peer) IP pair is used as the seed to generate a uniform random number, and the minimal statistic obtained by all of its peers (transformed by $-\log(1-u)$ where u is the minimal statistic) is recorded for the host. Then we obtain the MLE of $\mathbf{p}$ that characterizes the cardinality distribution modeled by the histogram in the $\log_2$ scale (2). The number of bins is chosen to be an upper bound for the maximum number of peers of all hosts. In the simulation, we fix $K = 12$ and thus there are 13 bins.

We note that to obtain confidence intervals, the Fisher information matrix approach is not good enough, because the empirical information matrix with a sample size 10,000 does not provide a sufficiently accurate estimate for the inverse of the 12×12 information matrix. Instead, we use the bootstrap method to estimate the standard deviation and then obtain the confidence interval, which is described earlier.

In Figure 1, the dashed line is the estimated value for the log scale histogram which is very close to the solid line for the ground true values. The dotted lines are the confidence interval estimated using the bootstrap method with 1000 replications. Note that the lower confidence interval after bin 7 is not shown since it becomes negative while the probability is plotted in $\log_2$ scale. Here we have 13 bins, and the standard error for the weight estimates for these bins is no more than .002. This is very small compared with the probability of most bins in the body distribution.

B. Comparison with the Naive Approach

Here we compare our approach with the naive FM approach, where the naive FM approach is defined as follows: 1) use a few FM sketches for each host that recording the minimal statistics independently, 2) estimate the cardinality of each host using the FM sketches, 3) compute the histogram in the $\log_2$ scale. Given L independent sketches for a host with cardinality n, say $y_1, \cdots, y_l$, then the maximum likelihood estimate of n is $\frac{l}{\sum_{j=1}^{l} y_j}$. Note that for n, the standard method (e.g. [17]) uses stochastic averaging to avoid multiple hashing for generating FM sketches, but its performance is not as good as independent hashing as a compromise.

We generate the cardinality random number for $m = 10,000$ hosts from the Pareto distribution the same as in Figure 1. To apply the naive FM approach, we have tried a variety number of l. In Figure 2, the solid line shows the empirical cardinality distribution based on the cardinality random numbers, the dashed line marked with '0' is the estimate using the MLE method, the dashed line marked with '1', '2', and so on, are the naive approach with $l = 1$, 2, and so on, respectively. The results show that the estimates using the naive FM approach converge to the true values as l increases to infinity. We note that it converges quickly for the right tail distribution part, for example, for bins greater than 64, the naive FM approach with $l = 3$ gives similar accuracy to our approach. However, it converges very slowly in the body of the distribution, especially for smaller values of cardinalities, while the memory requirement grows linearly with l. Even with $l = 9$, the L_1 norm of the estimation errors is still a little larger than that of the MLE method. Since we only use one sketch, with the same accuracy, our MLE method

[1]If X follows a Pareto distribution, then for $x \geq x_m$ $P(X \geq x) = (x/x_m)^{-\alpha}$, where x_m is the lower bound and α is the index. When $alpha = 1$, EX does not exist.

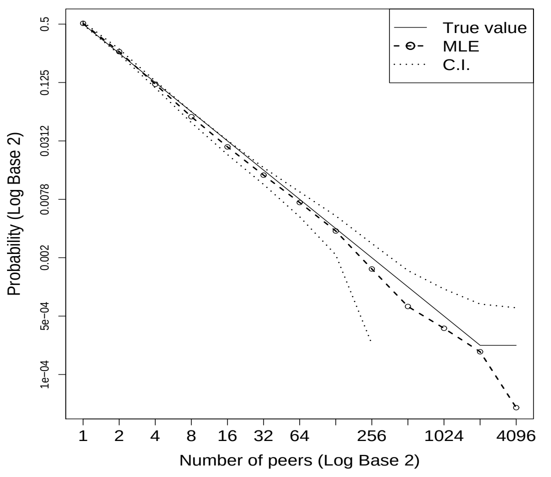

Fig. 1. Estimating a cardinality distribution and its confidence curves with simulated $m = 10,000$ hosts, where the number of distinct peers for each host follows a Pareto distribution with lower bound 1 and alpha index 1.

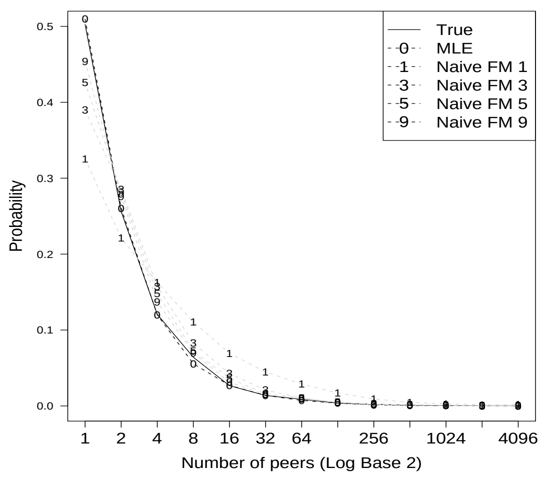

Fig. 2. Comparison with the naive FM approach with simulated $m = 10,000$ hosts, where the number of distinct peers for each host follows a Pareto distribution with lower bound 1 and alpha index 1.

reduces space requirement by at least an order of magnitude when compared with the naive approach, and computationally is also much simpler.

V. Experimental studies

We now evaluate our algorithms using two real network traces. The first one is a trace collected at a large corporation's gateway router in March 19 and 20 of 2004 (*trace1*), during which the network was hit by the Witty worm. The corporation's IP address space consists of two /16 address blocks, and most of the addresses are not used. The other one is one-hour trace collected at a large university's gateway router in the daytime of a weekday in Feb 2006 (*trace2*). Its IP addresses consist of one /16 address block, about half of which is used. We use the first one to illustrate the distributional change before and after the worm outbreak, and use the second trace to highlight differences in cardinality distributions between different organizations. The first one has very little peer-to-peer traffic where the latter has a significant amount of peer-to-peer traffic. We also report the accuracy of our estimation algorithms. For all estimation cases below, we randomly sample 10,000 hosts from each corresponding network or subnet and the continuous FM sketches are obtained for the sampled hosts.

A. Changes in Cardinality Distribution During Unusual Events

Figure 3 shows the flow cardinality distribution (histogram) of internal hosts in four consecutive time periods, each around 6 hours. The x-axis shows the number of flows; the y-axis is the fraction of internal hosts in a given bin (a range of flow cardinalities). Before the worm outbreak, we see that the distribution has two modes centered at cardinality 8 and 16. For example, if we look at the bottom histogram for the period from 6AM to 12PM of March 19, 2004, there are 50% internal hosts, each of which has between 8 and 15 flows, and there are 40% internal hosts, each of which has between 16 to 31 flows. A closer look at the trace shows that this is due to background scanning activities. For example, for the first period, we have 10 external hosts which scan at least half of the address range. Notice the distribution also has a long tail, which means there are a few internal hosts with a large number of flows. These hosts are the servers of this organization such as VPN servers, DNS servers, etc.

Now let us look up the distributional change around the time of Witty outbreak. Before the Witty outbreak, the distributional modes jitter between the bin 3 (cardinality range 8-15) and bin 4 (cardinality range 16-31) and the top two modes are the same. As shown in the Figure 3, however, the mode of the distribution shifts much more significantly when the Witty worm outbreaks, where the top two modes shift from the bins 3 and 4 (cardinality range 8-31) to bins 4 and 5 (cardinality range 16-63). The figure also tells that our estimate is reasonably accurate and that the distribution body shift is captured by the estimate. Errors on all bins are within 4% except for the distribution after Witty worm outbreak (the top panel). We also observe that the estimation errors are dominated by the bias. As a result, the confidence intervals described earlier are not good measure of the estimation errors. Further investigation shows that the bias is introduced by the equal weight assumption within each bin since the bias is reduced when finer bins are used and then probabilities are merged according to the logarithmic bins.

B. Cardinality Distributions of Different Organizations

We also obtain estimates of the distribution of the number of distinct communicating peers for each internal host in *trace2*, where the total number of internal hosts is about 44,600. Figure 5 shows the estimated distribution along with empirical observations, where the x-axis is the number of peers in log base 2 scale and the y-axis is the fraction of internal hosts.

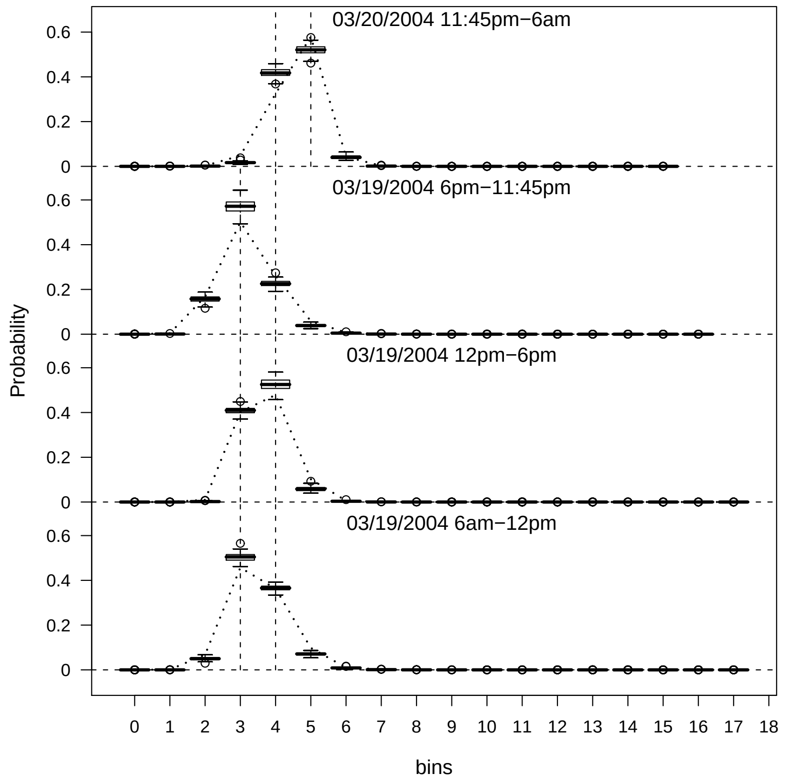

Fig. 3. Change of the cardinality distributions (flow counts) for about 131,000 internal IP addresses before and after the Witty worm outbreak. The panels bottom-up show the cardinality distributions and their estimates for about six-hour time interval from 6AM, March 19, 2004 to 6AM, March 20, 2004. The dotted lines show the true distributions and the boxplots show the estimates in 100 simulations. The two thick vertical lines in each panel are the top 2 modes.

We observe that the largest error is less than 2%, which is contributed mostly by the bias caused by the equal weight assumption within each bin as above.

We now compare the cardinality distributions of *trace1* and *trace2*. For *trace2*, we see that there are around 30% internal hosts with communicating peers ranging from 64 to 1024. A closer look at the traces reveals a large amount of bidirectional traffic are exchanged between these hosts, which indicates that these hosts are very likely running peer-to-peer applications. In contrast, for *trace1*, from the cardinality distribution shown in Figure 4, we see that more than 98% of internal hosts have from 4 to 64 communicating peers. This demonstrates that the communication patterns in terms of peer cardinality distributions can vary dramatically among different organizations, depending on the applications.

C. Cardinality Distributions using traffic subsets

We have presented cardinality distributions where we make use of both incoming traffic and outgoing traffic. We now show that, by looking at distribution from incoming traffic and outgoing traffic separately, interesting observations can be made. Figure 6 and 7 shows the peer cardinality distribution of university hosts calculated from incoming traffic and outgoing traffic respectively. We see that there are around 55% hosts with number of peers between 1 to 15 for incoming traffic. The number is 40% for outgoing. This suggests a significant number of hosts in the first distribution have fewer hosts than the second one. This is caused by unused IP addresses which are scanned from external hosts. A closer look at the trace shows that we have 44,600 hosts in the distribution of incoming traffic, the number is 29,600 in the distribution of outgoing traffic. So a significant fraction of unused IP addresses get scanned. Since the cardinality is small (< 15) for an hour long trace, the amount of scanning activity is not very severe.

VI. Related work

Our work falls in the broader scope of data streaming algorithms for network traffic analysis. For real time analysis, streaming algorithms must process the data in one pass as storing the data would entail a large delay. Such algorithms have been developed for tracking heavy hitters (top ranked hosts by traffic volume) [4], [15] and superspreaders [5], [27], estimating frequency moments [1] and entropy [6], [25], flow-size distribution [12], [23], and inverse distribution [10], [21].

Most of these algorithms are traffic volume-based summary information. Even the inverse distribution is volume-based: given a volume x, how many hosts n_x have such x volume of traffic in a given period of time? Conceptually, one can maintain a counter per volume denote $f(x)$, and a counter per IP address i denote $g(i)$. When a new packet with IP address i and size s arrives, $f(g(i))$ will be subtracted by 1, $f(g(i)+s)$ will be incremented by 1 and $g(i)$ will be incremented by s.

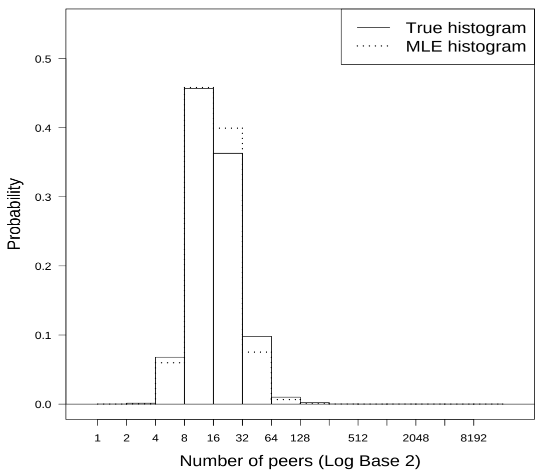

Fig. 4. The peer cardinality distributions of internal hosts of a large corporate network (in *trace1*), where the solid and dashed lines are the true and MLE estimated distributions, respectively.

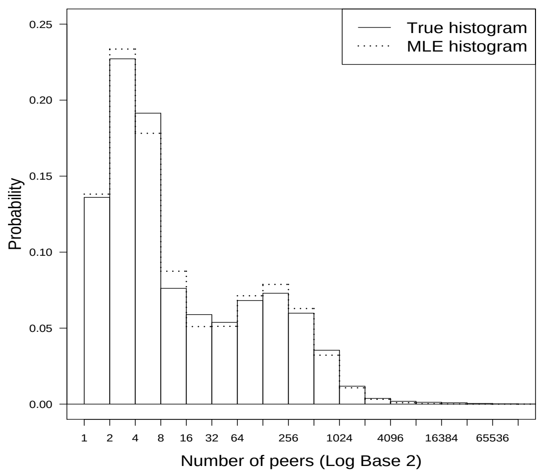

Fig. 5. The peer cardinality distributions of internal hosts of a large university network (in *trace2*), where the solid and dashed lines are the true and MLE estimated distributions, respectively.

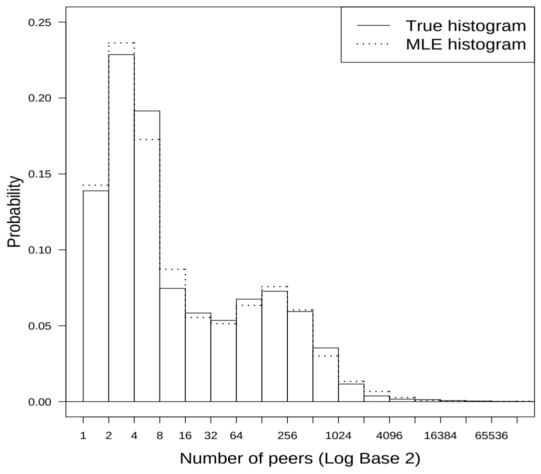

Fig. 6. The peer cardinality distributions of internal hosts of a large university network from incoming traffic (in *trace2*), where the solid/dashed lines are the true/MLE estimated distributions.

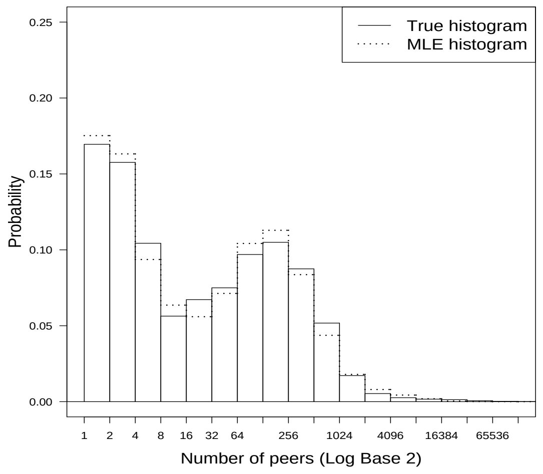

Fig. 7. The peer cardinality distributions of internal hosts of a large university network from outgoing traffic (in *trace2*), where the solid/dashed lines are the true/MLE estimated distributions.

These counters do not count distinct elements. Therefore, their problems are very different from ours.

One notable exception is the work on tracking individual superspreaders [27]. A superspreader is defined as the host that communicates with a large number of peers (at least k). This consists of the right tail part only of the distribution we are estimating in this paper. Their solution requires two hash tables, each of size c_1N/k where c_1 is constant, N is the number of unique source destination pairs. In contrast, we maintain a sketch (< 32 bits) per source. In addition, they only consider supspreaders in terms of communicating peers, not in terms of flows. Our algorithms deal with flows gracefully without increasing the space requirement.

VII. Conclusion and future work

Real time tracking of summary information in network traffic is crucial for many network functions such as network monitoring and traffic engineering. If these summary information changes, network operators may want to understand the causes and respond accordingly. Previous work has focused on flow size distribution, entropy of various packet feature (IP addresses and port numbers) distributions. In this paper, we investigate the cardinality distributions of communicating peers or flows of hosts. Cardinality distributions can capture events in the network traffic that may not register in volume-based distributions. We propose a simple summary statistics and an efficient algorithm for estimating cardinality distributions. It is efficient in both space and computational time. In terms of space, we only require one (at most 32 bit) random number per host. For each packet arrival, we only needs one random number generation and one comparison for updating one of the sketches. Our estimation approach is based on the maximum likelihood method and thus is optimal in terms of statistical efficiency. We have also developed a simple and fast algorithm derived from Expectation-Maximization which can be computed quickly. Notice that our approach is scalable due to the nice property of continuous FM sketches even when some hosts have a huge number of cardinalities. This is advantageous over purely sampling based methods, which requires a good but hard to design sampling rate (see [7]).

For future work, we would like to classify cardinality distributions in terms of the traffic direction. The motivation is that two-way traffic are usually legitimate, which consists exchanges between clients and servers, or between P2P hosts. One-way traffic is typically illegitimate, which consists of mostly failed connections due to port scanning. Therefore, it is more useful to separate the traffic into different classes and study their cardinality distributions separately. Inside a provider network, routing may not be symmetric, therefore, it is challenging to obtain these directional distributions. Another technical difficulty is that separating traffic into different directions cannot be easily done without a peer (or flow) lookup table which may consume a large amount of memory.

VIII. Appendix

Proof of Theorem 2.

Proof: We only need to derive (7). Let $C \in \{0, \cdots, K$ be a random number with probability $P(C = k) = p_k$, $k = 0, \cdots, K$. Then Y can be obtained as follows: generate a random number from the C distribution, if it is k, then generate Y from the distribution corresponding to density function $\overline{f}'_k$. The likelihood function of (Y, C) can be written as

$$p(Y, C) = \prod_{k=0}^{K} p_k^{I(C=k)} \overline{f}'_k(Y).$$

Thus given complete data $\{(Y_i, C_i) : i = 1, \cdots, m\}$, the logarithmic likelihood function is

$$L = \sum_{i=1}^{m} \sum_{k=0}^{K} I(C_i = k) \log p_k \overline{f}'_k(Y_i).$$

Since C_is are not observed, the EM algorithm first estimates the expectation of L given Y_is and $\mathbf{p}^{(old)}$, for which we need to compute $P(C_i = k|Y_i, \mathbf{p}^{(old)})$ (E-step). and then maximize the expected L with respect to $\mathbf{p}$ to obtain $\mathbf{p}^{(new)}$(M-step). By the Bayes rule, the E-step is equivalent to computing

$$\pi_{ik} = P\left(I_i = k|Y_i, \mathbf{p}^{(old)}\right) = \frac{p_k^{(old)} \overline{f}'_k(Y_i)}{\sum_{k=0}^{K} p_k^{(old)} \overline{f}'_k(Y_i)}.$$

Then the expected L becomes $\sum_{i=1}^{m} \sum_{k=0}^{K} \pi_{ik} \log\left(p_k \overline{f}'_k(Y_i)\right)$, for which the optimization has closed form. That is, the M-step becomes

$$p_k^{(new)} = \frac{\sum_{i=1}^{m} \pi_{ik}}{m}.$$

■

References

[1] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing (STOC)*, pages 20–29, 1996.

[2] Z. Bar-Yossef, T. S. Jayram, R. Kumar, D. Sivakumar, and L. Trevisan. Counting distinct elements in a data stream. In *RANDOM*, pages 1–10, 2002.

[3] P. Bickel and K. Doksum. *Mathematical Statistics: Basic Ideas and Selected Topics, Vol 1, (2nd Edition)*. Prentice Hall, 2000.

[4] T. Bu, J. Cao, A. Chen, and P. P. C. Lee. A fast and compact method for unveiling significant patterns in high speed networks. In *INFOCOM*, pages 1893–1901, 2007.

[5] J. Cao, Y. Jin, A. Chen, T. Bu, and Z. Zhang. Identifying high cardinality internet hosts. In *Proceedings of IEEE INFOCOM (to appear)*, 2009.

[6] A. Chakrabarti, G. Cormode, and A. McGregor. A near-optimal algorithm for computing the entropy of a stream. In *SODA*, pages 328–335, 2007.

[7] A. Chen and J. Cao. Distinct counting with a self-learning bitmap. In *Proceedings of the 25th International Conference on Data Engineering (ICDE)*, 2009.

[8] A. Chen, J. Cao, and T. Bu. A simple and efficient estimation method for stream expression cardinalities. In *VLDB*, pages 171–182, 2007.

[9] A. Chen, L. Li, and J. Cao. Estimating cardinality distributions in network traffic: extended abstract. In *SIGMETRICS*, pages 459–460, 2008.

[10] G. Cormode, S. Muthukrishnan, and I. Rozenbaum. Summarizing and mining inverse distributions on data streams via dynamic inverse sampling. In *VLDB*, pages 25–36, 2005.

[11] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.

[12] N. Duffield, C. Lund, and M. Thorup. Estimating flow distributions from sampled flow statistics. In *SIGCOMM*, pages 325–336, 2003.

[13] M. Durand and P. Flajolet. Loglog counting of large cardinalities. In *Proceedings of European Symposium on Algorithms*, pages 605–617, 2003.

[14] B. Efron and R. Tibshirani. *An Introduction to the Bootstrap*. Chapman Hall, 1994.

[15] C. Estan and G. Varghese. New directions in traffic measurement and accounting. In *Proc. ACM SIGCOMM Internet Measurement Workshop*, Nov. 2001.

[16] C. Estan, G. Varghese, and M. Fisk. Bitmap algorithms for counting active flows on high speed links. *IEEE/ACM Trans. on Networking*, 14(5):925–937, 2006.

[17] P. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. *Journal of Computer and System Sciences*, 31(2):182–209, 1985.

[18] S. Ganguly. Counting distinct items over update streams. In *ISAAC*, pages 505–514, 2005.

[19] F. Giroire. Order statistics and estimating cardinalities of massive data sets. In C. Martínez, editor, *International Conference on Analysis of Algorithms*, pages 157–166, 2005.

[20] Y. He and C. Liu. The dynamic ECME algorithm. Technical report, 2009.

[21] V. Karamcheti, D. Geiger, Z. Kedem, and S. Muthukrishnan. Detecting malicious network traffic using inverse distributions of packet contents. In *MineNet: Proceeding of the ACM SIGCOMM workshop on Mining network data*, pages 165–170, 2005.

[22] A. Karasaridis, B. Rexroad, and D. Hoeflin. Wide-scale botnet detection and characterization. In *First Workshop on Hot Topics in Understanding Botnets (HotBots)*, 2007.

[23] A. Kumar, M. Sung, J. J. Xu, and J. Wang. Data streaming algorithms for efficient and accurate estimation of flow size distribution. In *SIGMETRICS/Performance*, pages 177–188, 2004.

[24] A. Lakhina, M. Crovella, and C. Diot. Mining anomalies using traffic feature distributions. *SIGCOMM Comput. Commun. Rev.*, 35(4):217–228, 2005.

[25] A. Lall, V. Sekar, M. Ogihara, J. Xu, and H. Zhang. Data streaming algorithms for estimating entropy of network traffic. *SIGMETRICS Perform. Eval. Rev.*, 34(1):145–156, 2006.

[26] S. Sen and J. Wang. Analyzing peer-to-peer traffic across large networks. *IEEE/ACM Trans. Netw.*, 12(2):219–232, 2004.

[27] S. Venkataraman, D. Song, P. Gibbons, and A. Blum. New streaming algorithms for superspreader detection. In *Proc. of Network and Distributed Systems Security Symposium*, Feb. 2005.

[28] L. Yang and G. Michailidis. Sampled based estimation of network traffic flow characteristics. In *INFOCOM*, pages 1775–1783, 2007.