A New Virtual Indexing Method for Measuring Host Connection Degrees

Pinghui Wang[1], Xiaohong Guan[1,2], Weibo Gong[3], and Don Towsley[4]
[1]SKLMS Lab and MOE KLINNS Lab, Xi'an Jiaotong University, Xi'an, China
[2]Department of Automation and NLIST Lab, Tsinghua University, Beijing, China
[3]Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA
[4]Department of Computer Science, University of Massachusetts, Amherst, MA
Emails: {phwang, xhguan}@sei.xjtu.edu.cn, gong@ecs.umass.edu, towsley@cs.umass.edu

Abstract—**We present a new virtual indexing method for estimating host connection degrees for high speed links. It is based on the virtual connection degree sketch where a compact sketch of network traffic is built by generating the associated virtual bitmaps for each host. Each virtual bitmap consists of a fixed number of bits selected randomly from a shared bit array by a new method for recording the traffic flows of the corresponding host. The shared bit array is efficiently utilized by all hosts since its every bit is shared by the virtual bitmaps of multiple hosts. To reduce the "noise" contaminated in a host's virtual bitmaps due to sharing, we propose a new method to generate the "filtered" bitmap used to estimate host connection degree. Furthermore, it can be easily implemented in parallel and distributed processing environments. The experimental and testing results based on the actual network traffic show that the new method is accurate and efficient.**

Keywords-Bitmap; Data streaming; Host connection degree; Virtual Indexing.

I. INTRODUCTION

The in-degree/out-degree, defined as the number of distinct sources/destinations that a network host connects to during a given time window, is an important statistic metric of network traffic that provides with insights into network measurement and monitoring applications such as host profiling [1], and aids fast detection of security attacks [2], [3], etc. To obtain the total number of flows generated by a host, one needs to build a hash table that keeps track of existing flows to avoid duplicating flow records for packets from the same flow. In this paper, a flow is defined as the set of all packets with the same source and destination addresses in a time window. However, it is not practical to obtain host connection degrees by building per-host hash tables that are resource intensive to maintain for high speed links carrying a huge number of simultaneous active hosts and flows. It may not be possible to accurately measure and monitor massive network traffic simply by upgrading the performance of measuring devices. Hence, it is desirable to develop new methods to meet the challenges of monitoring high speed network traffic online.

Estan and Varghese [4] proposed a family of bitmap algorithms for estimating the total number of distinct flows on high speed links. To estimate each host's connection degree, it need to build a bitmap for each host, which may not be scalable to high speed links carrying flows associated a huge number of hosts. Zhao et al. [5] proposed a data stream method to measure host connection degrees, which is a variant of Bloom filter [6] and consists of $n \times m$ 2-dimentional bit array. Each host is associated with H columns in the bit array randomly selected by H hash functions, and one bit in each of its associated columns is set as one for each of its packets coming. The corresponding H columns of each host can be used to estimate its connection degree, since each column can be viewed as a direct bitmap as proposed in [7]. The direct bitmap method indicates that the number of rows represented in the bit array should be set in the order of thousands to perform the task of estimating connection degrees of sources with thousands of flows. However, most hosts have only several flows and only a very small number of hosts have thousands of flows, which implies that most columns in the bit array are assigned to hosts with a small connection degree, and are mostly zeros. To reduce this space inefficiency, Yoon et al. [8] build a virtual bitmap for each host by taking bits randomly from a shared bit array using a group of hash functions. Each host's virtual bitmap is used to estimate its connection degree similar to what is proposed in [7] using a direct bitmap. The number of distinct bits in a host's virtual bitmap varies due to hash collisions. There is no guarantee on the quality of the estimate of the out-degree of a host whose virtual bitmap is generated with many hash collisions. Furthermore, each bit in the shared bit array may be shared by several hosts and the "noise" contaminated in a host's virtual bitmap exists while estimating its connection degree. In particular, a host with a small number of connections is sensitive to the "noise", since it has more bits in its virtual bitmap that are probably contaminated by other sources.

In this paper, we present a new virtual indexing method to accurately estimate host connection degrees over high speed links. A virtual connection degree sketch data structure is designed to build a very compact sketch of network traffic, which can be used to estimate the connection degree of each host based on the associated virtual bitmaps. Each virtual bitmap consists of a fixed number of bits selected randomly from a shared bit array by a new virtual bitmap generation method. The shared bit array is small, and is efficiently utilized by all hosts since its every bit is shared by the virtual bitmaps of multiple hosts. The new method is computationally efficient since it only needs to set several bits for each incoming packet. To reduce the "noise" contaminated in each

The research presented in this paper is supported in part by National Natural Science Foundation (60574087), 863 High Tech Development Plan (2007AA01Z475, 2007AA01Z480, 2007AA01Z464, 2008AA01Z415)

host's virtual bitmaps because of sharing, we present a new method for generating the "filtered" bitmap used to estimate the host connection degree. Furthermore, it can be easily implemented in parallel and distributed processing environments. Experiments based on the actual network traffic show that the new method is truly accurate and efficient. It should be noted that the algorithm for measuring host out-degrees can also be applied for host in-degrees.

This paper is organized as follows. In Section II the new virtual indexing method is described in details. The performance evaluation is presented in Section III. Concluding remarks then follow.

II. Virtual Connection Degree Sketch

A. Data Structure

A virtual connection degree sketch (*VCDS*) consists of a bit array $A[k]$ ($0 \le k \le m-1$) associated with H independent groups of hash functions $\{f_{1,0}, f_{1,1}, \ldots, f_{1,L-1}\}$, $\{f_{2,0}, f_{2,1}, \ldots, f_{2,L-1}\}$, …, and $\{f_{H,0}, f_{H,1}, \ldots, f_{H,L-1}\}$. Each $f_{i,j}$ ($1 \le i \le H$, $0 \le j \le L-1$) is a hash function: $\{0, 1, \ldots, N\text{-}1\} \to \{0, 1, \ldots, m\text{-}1\}$, where N is the size of source space $\boldsymbol{S}$.

Each source s has H corresponding virtual bitmaps $B_i(s)$ ($1 \le i \le H$) where $B_i(s)$ is defined as a bit array consisting of L bits selected randomly from A by the group of hash functions $\{f_{i,0}, f_{i,1}, \ldots, f_{i,L-1}\}$, that is

$$B_i(s) = (A[f_{i,0}(s)], A[f_{i,1}(s)], \ldots, A[f_{i,L-1}(s)]), \quad 1 \le i \le H .$$

$B_i(s)$ can be viewed as a direct bitmap [7] occupied only by s. The length of the bit array in a direct bitmap is constant and is a vital parameter to estimate the out-degree of s. However the number of distinct bits in $B_i(s)$ is smaller than L, when $A[f_{i,0}(s)]$, $A[f_{i,1}(s)]$, …, $A[f_{i,L-1}(s)]$ are selected from A with hash collisions. There is no guarantee on the quality of the estimate of the out-degree of a host whose virtual bitmap is generated with many collisions. For example, when m=10^6 and L=10^4, the number of distinct bits selected from A by the group of hash functions $\{f_{i,0}, f_{i,1}, \ldots, f_{i,L-1}\}$ is smaller than L with a probability of $1\text{-}10^{-22}$ and its expectation is 9950. This problem also exists but is not noticed in [8]. To address this issue, we propose a virtual bitmap generating method by designing $f_{i,j}$ ($1 \le i \le H$, $0 \le j \le L-1$) based on the *double hashing* scheme [9]

$$f_{i,j}(s) = \psi_{i,1}(s) + j\psi_{i,2}(s) \bmod m$$

where $\psi_{i,1}$ is a hash function that maps the source space uniformly to the range $\{0, 1, \ldots, m\text{-}1\}$, $\psi_{i,2}$ is a hash function that maps the source space uniformly to the range $\{1, 2, \ldots, m\text{-}1\}$, and m is a prime. It is easily validated that each $f_{i,j}$ also maps the source space uniformly to the range $\{0, 1, \ldots, m\text{-}1\}$. The following theorem shows that each virtual bitmap is hashed into L different bits in A.

Theorem 1. For a source s, L different bits are selected from A by each group of hash functions $\{f_{i,0}, f_{i,1}, \ldots, f_{i,L-1}\}$ ($1 \le i \le H$), that is,

$$f_{i,j_1}(s) \ne f_{i,j_2}(s) , \qquad 0 \le j_1 < j_2 \le L-1 .$$

Proof. (Proof by contradiction) Assume to the contrary that there is $f_{i,j_1}(s) = f_{i,j_2}(s)$, for $j_1 \ne j_2$ then

$$(j_2 - j_1)\psi_{i,2}(s) \equiv 0 \bmod m .$$

Since $1 \le j_2 - j_1 \le L-1 \ll m$ and m is prime, we have $j_2 - j_1 \ne 0 \bmod m$ and $\psi_{i,2}(s) \equiv 0 \bmod m$. Note $\psi_{i,2}(s) \le m-1$, so we have $\psi_{i,2}(s) = 0$. This contradicts with the definition of $\psi_{i,2}(s) \in \{1, 2, \ldots, m-1\}$. □

Theorem 2. For each $B_i(s)$ ($1 \le i \le H$ and $s \in \boldsymbol{S}$), denote set $SB_i(s)$=$\{f_{i,j}(s) | \ 0 \le j \le L-1\}$. Then the probability that any bit $A[k]$ ($0 \le k \le m-1$) is included in $B_i(s)$ is

$$P\{k \in SB_i(s)\} = \frac{L}{m} . \qquad (1)$$

Proof. If $k \in SB_i(s)$ and k is the the j-th ($0 \le j \le L-1$) bit in $B_i(s)$=$(A[f_{i,0}(s)], A[f_{i,1}(s)], \ldots, A[f_{i,L-1}(s)])$, we have

$$k = \psi_{i,1}(s) + j\psi_{i,2}(s) \bmod m.$$

For each $\psi_{i,2}(s)$ in $\{1, 2, \ldots, m\text{-}1\}$ and each j in $\{0, 1, \ldots, L\text{-}1\}$, there is one and only one virtual bitmap contained $A[k]$, since $\psi_{i,1}(s)$ is determined as follows:

$$\psi_{i,1}(s) = k - j\psi_{i,2}(s) \bmod m .$$

Therefore the total number of distinct virtual bitmaps contained $A[k]$ is $L(m\text{-}1)$. Since each virtual bitmap $B_i(s)$ is selected uniformly from the total $m(m\text{-}1)$ distinct virtual bitmaps, we have (1). □

B. Update Procedure

Each bit in A is initially set to zero. When a packet $p = (s, d)$ arrives, we set the $g(s||d)$-th bit in each virtual bitmap $B_i(s)$ to one. Here g is a uniform hash function with the range $\{0, \ldots, L\text{-}1\}$, and the flow label $s||d$ is the concatenation of source s and destination d. As the $g(s||d)$-th position in $B_i(s)$, corresponds to the $f_{i,g(s||d)}(s)$-th position in A, we only need to set H bits for each incoming packet as follows:

$$A[f_{i,g(s||d)}(s)] = 1, \quad 1 \le i \le H .$$

C. Connection Degree Estimator

The bits in $B_i(s)$ ($1 \le i \le H$) that the flows of source s hash into using hash functions $\{f_{i,0}, f_{i,1}, \ldots, f_{i,L-1}\}$ are denoted as the bits used by s in the following part. They are set to one to store the flow information of s, so each $B_i(s)$ can be used to estimate the out-degree of s similar to the direct bitmap proposed in [7]. Since each bit in $B_i(s)$ is selected randomly from A and also shared by other sources, the other bits in $B_i(s)$ not used by s might also be set to one by flows belonging to other sources. This introduces "noise" into the estimation of the out-degree of s. Therefore the more bits in $B_i(s)$ are not used by s, the more "noise" is generated. The size of the virtual bitmap, L, is usually set to thousands to guarantee the high accuracy of estimating the out-degree of a source associated with a huge number of flows, which will generate a large number of bits containing "noise" especially for the source associated with a small number of flows. In what follows, we introduce a new "filtered" bitmap generation method to reduce the "noise" generated by other sources. The "filtered" bitmap B_s defined as a bit vector is computed from $B_1(s)$, $B_2(s)$, …, and $B_H(s)$ as follows:

$$B_s = B_1(s) \otimes B_2(s) \otimes \cdots \otimes B_H(s)$$

where $\otimes$ is the bit-AND operation. For any flow (s, d) of s, the $g(s\|d)$-th bit in each $B_i(s)$ ($1 \le i \le H$) is set to one, so the $g(s\|d)$-th bit in B_s is still one. Define $\varphi(j)$ as follows:

$$\varphi(j) = \frac{m-L}{m} + \frac{L}{m}\left(1 - \frac{1}{L}\right)^{OD_j} \tag{2}$$

where OD_j is the out-degree of source j. Then for any given bit in each $B_i(s)$ not used by s, the probability that it is set to zero by any other source j is $\varphi^H(j)$ based on Theorem 2. Therefore it is a noise bit when all corresponding bits in $B_i(s)$ ($1 \le i \le H$) are ones with probability

$$q_1^{(s)} = \left(1 - \frac{\Phi^H}{\varphi(s)}\right)^H$$

where $\Phi = \prod_j \varphi(j)$. Suppose m and L are given, we want to optimize H to minimize the "noise" contaminated in the "filtered" bitmap B_s. There are two competing forces: for each bit in B_s not used by s, using larger H gives us a greater chance of finding a zero bit in its H corresponding bits in $B_i(s)$; but using more bitmaps results in more bits in A being set to one, which increases the probability that a host's virtual bitmap is contaminated with "noise". When m>>L, we have $q_1^{(s)} \approx \tau(H) = \left(1-\Phi^H\right)^H$. The impact of H on the "noise" is described in the following lemma.

Lemma 1. $\tau(H) = \left(1-\Phi^H\right)^H$ decreases with $H \in \left(0, -\frac{1}{\ln\Phi}\right)$, increases with $H \in \left(-\frac{1}{\ln\Phi}, +\infty\right)$, and obtains the minimum at $H = -\frac{1}{\ln\Phi}$.

Proof. Define $y = \ln \tau(H)$, then its first derivative and second derivative are

$$\frac{dy}{dH} = \ln\left(1-\Phi^H\right) - \frac{H\Phi^H \ln\Phi}{1-\Phi^H}$$

$$\frac{d^2y}{dH^2} = \frac{\Phi^H\left(2-2\Phi^H - H\ln\Phi\right)\ln\Phi}{\left(1-\Phi^H\right)^2}.$$

It can be easily shown that $\frac{dy}{dH} < 0$ when $H \in \left(0, -\frac{1}{\ln\Phi}\right)$ and $\frac{dy}{dH} > 0$ when $H \in \left(-\frac{1}{\ln\Phi}, +\infty\right)$, therefore function $\tau(H)$ decreases with $H \in \left(0, -\frac{1}{\ln\Phi}\right)$ and increases with $H \in \left(-\frac{1}{\ln\Phi}, +\infty\right)$. Meanwhile we have $\left.\frac{dy}{dH}\right|_{H=-\frac{1}{\ln\Phi}} = 0$ and $\left.\frac{d^2y}{dH^2}\right|_{H=-\frac{1}{\ln\Phi}} > 0$, so the optimal value of $\tau(H)$ is obtained at $H = -\frac{1}{\ln\Phi}$. □

In what follows a new method is proposed to estimate the out-degree of source s. For any given bit in B_s not used by s, the probability that no other source uses it either is $p_s = 1 - q_1^{(s)}$. Denote the total number of bits in B_s not used by s as U_s, the expectation of U_{B_s} defined as the total number of zero bits in B_s is

$$E\left(U_{B_s}\right) = E\left(U_s p_s\right) = p_s E\left(U_s\right). \tag{3}$$

The probability that a given bit in B_s is not used by s is $\left(1-\frac{1}{L}\right)^{OD_s}$, so the expectation of U_s is computed as follows:

$$E\left(U_s\right) = L\left(1-\frac{1}{L}\right)^{OD_s} \approx Le^{-\frac{OD_s}{L}}. \tag{4}$$

p_s is computed with inputs of each hosts' out-degrees which are unknown. We find that p_s can be estimated from A. Denote $AB_i(s)$ as the bits in $A[k]$ ($0 \le k \le m-1$) except $A[f_{i,0}(s)]$, $A[f_{i,1}(s)]$, …, $A[f_{i,L-1}(s)]$. For each bit in $AB_i(s)$, the probability of being set to zero is $\alpha_s = \frac{\Phi^H}{\varphi(s)}$. Let $U'_{B_i(s)}$ be the total number of zero bits in $AB_i(s)$. Then

$$E\left(U'_{B_i(s)}\right) = (m-L)\alpha_s. \tag{5}$$

Since $p_s = 1 - q_1^{(s)} = 1 - \left(1-\alpha_s\right)^H$, we have the following equation from (5):

$$p_s = 1 - \left(1 - \frac{E\left(U'_{B_i(s)}\right)}{m-L}\right)^H. \tag{6}$$

From (3), (4) and (6), we have

$$OD_s \approx -L\ln\frac{E\left(U_{B_s}\right)}{L} + L\ln\left(1-\left(1-\frac{E\left(U'_{B_i(s)}\right)}{m-L}\right)^H\right). \tag{7}$$

Replacing $E\left(U_{B_s}\right)$ and $E\left(U'_{B_i(s)}\right)$ in (7) by the instantaneous values, U_{B_s} and $U'_{B_i(s)}$ obtained from B_s and $AB_i(s)$ respectively, we have the following estimate of $OD_s^{(i)}$ ($1 \le i \le H$):

$$OD_s^{(i)} = -L\ln\frac{U_{B_s}}{L} + L\ln\left(1-\left(1-\frac{U'_{B_i(s)}}{m-L}\right)^H\right). \tag{8}$$

The first term of the right hand side of (8) corresponds to the estimator used in the direct bitmap [7]. The second term of the right hand side of (8) is used to compensate for the error caused by the "noise" generated by the other sources.

Finally the following equation gives a more accurate estimator of the out-degree of *s*:

$$OD_s^{est} = \underset{1\le i\le H}{median}\, OD_s^{(i)} .$$

D. Combination Operation

A *VCDS A* can be distributed across *G* routers as follows. Each router *i* ($1 \le i \le G$) can maintain a *VCDS* A_i, based on the traffic that it observes. Based on all network traffic at all *G* routers, *VCDS A* can be calculated by the following equation

$$A = A_1 \oplus A_2 \oplus \cdots \oplus A_G$$

where $\oplus$ is the bit-OR operation. Each distributed node *i* ($1 \le i \le G$) only needs to send A_i to the control center, which greatly reduces the amount of communications between the distributed nodes and the control center.

E. Parameter Configuration

Since the total complexity for updating each packet is $O(H)$, *H* should be small. If the out-degree of source *s* is much larger than $L\ln L + O(L)$, we will obtain all 1's in its corresponding virtual bitmaps with high probability due to the result of the "coupon collector's problem" [10]. In this case, the out-degree of *s* cannot be estimated accurately, and we only know that it is not smaller than $L\ln L$. To address this issue, we can directly use a larger *L* or apply the pre-sampling method proposed in [8].

III. Testing and Performance Evaluation.

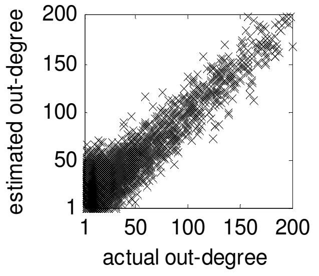
A. Data Collection

Experimental data used in this paper is based on the network traffic of the backbones of CERNET (China Education and Research Network) Northwest Regional Center and the campus network of Xi'an Jiaotong University with more than 300000 hosts. The actual traffic data was collected at an egress router with a bandwidth of 1.5Gbps from a B-class network by using TCPDUMP. The total number of collected packets is about 1.2×10^8. There are 1.4×10^5 distinct sources and 4.7×10^5 distinct flows. We further compare *VCDS* with the state-of-the-art method compact spread estimator (*CSE*) [8].

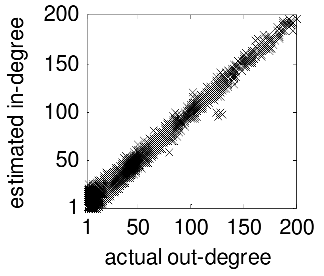
B. Host Out-Degree Estimation

The following experiments are used to evaluate the performance of estimating host out-degrees provided by *VCDS*. The relative error of estimated out-degree of host *s* is defined as $|OD_s^{est} - OD_s|/OD_s$, where OD_s^{est} is its estimated out-degree and OD_s is its actual out-degree. Fig. 1 and 2 show the relative errors of host out-degree estimates for different *H*, where *m*=4 M and *L*=2048. The accuracy of *VCDS* first increases with *H* and then decreases with *H*, which is consistent with Lemma 1.

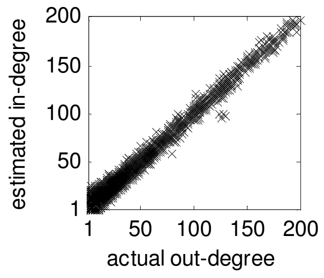
The impact of *L* is shown in Fig. 3, where *H*=5 and *m*=4 M. For a host with a small out-degree, the error of its estimated out-degree increases with *L*, since a larger value of *L* introduce more unused bits in the host's virtual bitmaps, which might be contaminated. However, a smaller value of *L* generates a larger estimation error when the out-degree is large similar to the direct bitmap algorithm [7].



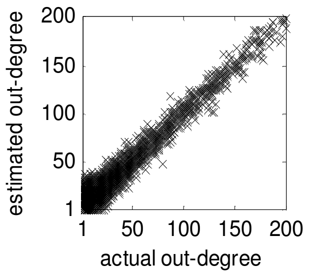
(a). $H = 1$



(b). $H = 3$

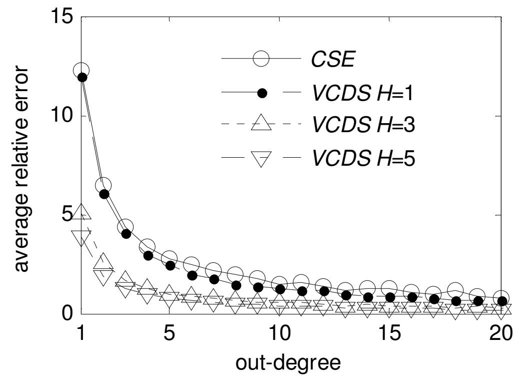


(c). $H = 5$

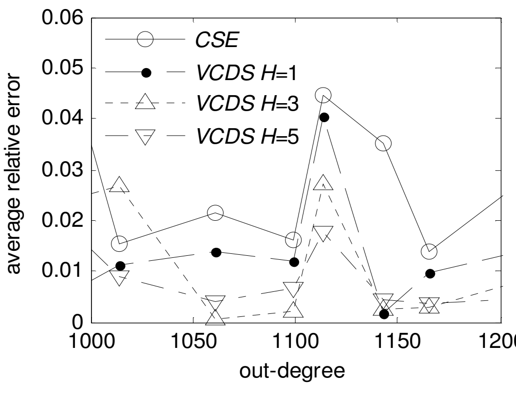


(d). $H = 15$

Figure 1. Actual vs estimated out-degree for different *H*

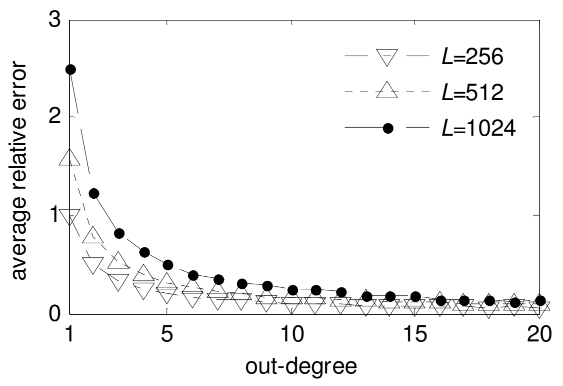


(a). Average relative error for small out-degrees

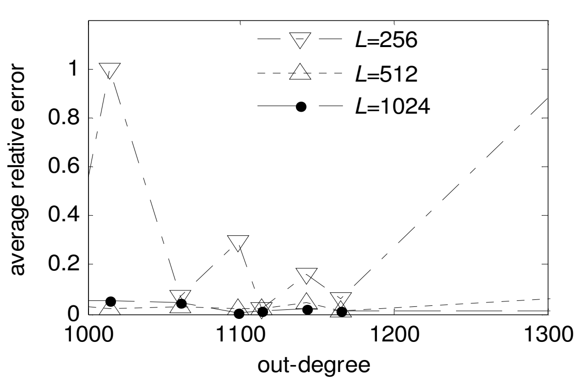


(b). Average relative error for large out-degrees

Figure 2. Average relative error for different *H*

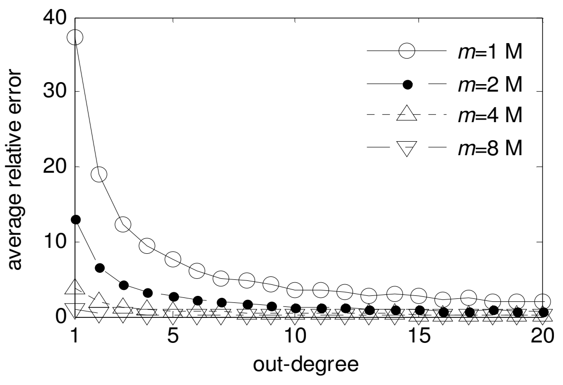


(a). Average relative error for hosts with small out-degrees

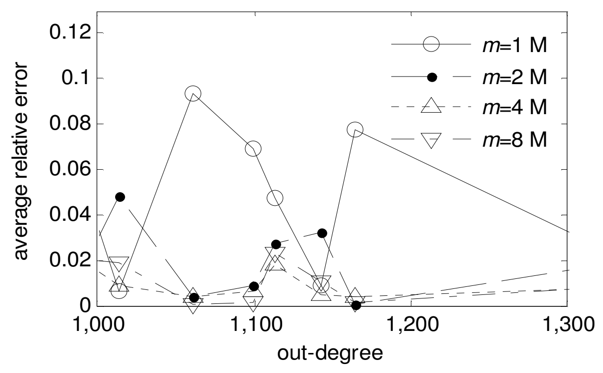


(b). Average relative error for hosts with large out-degrees

Figure 3. Average relative error for different *L*



(a). Average relative error for hosts with small out-degrees



(b). Average relative error for hosts with large out-degrees

Figure 4. Average relative error for different *m*

Fig. 4 shows the average relative errors of host out-degree estimates for different *m*, where *H*=5 and *L*=2048. We observe that the accuracy of estimated out-degrees is improved by increasing *m*, which reduces the "noise", especially for hosts with small out-degrees. For a host with a small out-degree, the number of unused bits in its virtual bitmap is larger than that of a host with a large out-degree. Therefore the estimated out-degree of a host with a small out-degree is more sensitive to the "noise".

IV. CONCLUSIONS

In this paper we present a data streaming method *VCDS* for estimating host connection degrees over high speed links. *VCDS* can generate a very compact sketch of network traffic, and the connection degree of each host is estimated by the associated virtual bitmaps consisting of a fixed number of bits randomly selected from a shared bit array. The experimental and test results based on the actual network traffic show that the new method is accurate and computationally efficient.

REFERENCES

[1] K. Xu, Z. L. Zhang, and S. Bhattacharyya, "Profiling Internet backbone traffic: behavior models and applications," in *Proceedings of ACM SIGCOMM 2005*, Philadelphia, PA, 2005, pp. 169–180.

[2] M. Roesch, "Snort–lightweight intrusion detection for networks," in *Proceedings of the USENIX LISA Conference on System Administration 1999*, Seattle, WA, 1999, pp. 229–238.

[3] D. Plonka, "Flowscan: A network traffic flow reporting and visualization tool," in *Proceedings of USENIX LISA 2000*, New Orleans, LA, 2000, pp. 305–317.

[4] C. Estan and G. Varghese, "Bitmap algorithms for counting active flows on high speed links," in *Proceedings of ACM SIGCOMM IMC 2003*, Miami, FL, 2003, pp. 182–209.

[5] Q. Zhao, A. Kumar, and J. Xu, "Joint data streaming and sampling techniques for detection of super sources and destinations," in *Proceedings of ACM SIGCOMM IMC 2005*, Berkeley, CA, 2005, pp.77–90.

[6] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, July 1970.

[7] K. Y. Whang, B. T. Vander-zanden, and H. M. Taylor, "A linear-time probabilistic counting algorithm for database applications," *IEEE Transaction of Database Systems*, vol. 15, no. 2, pp. 208–229, June 1990.

[8] M. Yoon, T. Li, S. G. Chen, J. Peir, "Fit a spread estimator in small memory," in *Proceedings of IEEE INFOCOM 2009*, Rio de Janeiro, Brazil, 2009, pp. 504–512. Journal version is to be appeared in *IEEE Transaction on Newtworking*.

[9] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, "Introduction to algorithms (2nd ed.)," MIT Press, Cambridge, MA, 2001.

[10] R. Motwani and P. Raghavan, "Randomized Algorithms," *Cambridge University Press*, 1995.