Data Streaming Algorithms for Efficient and Accurate Estimation of Flow Size Distribution

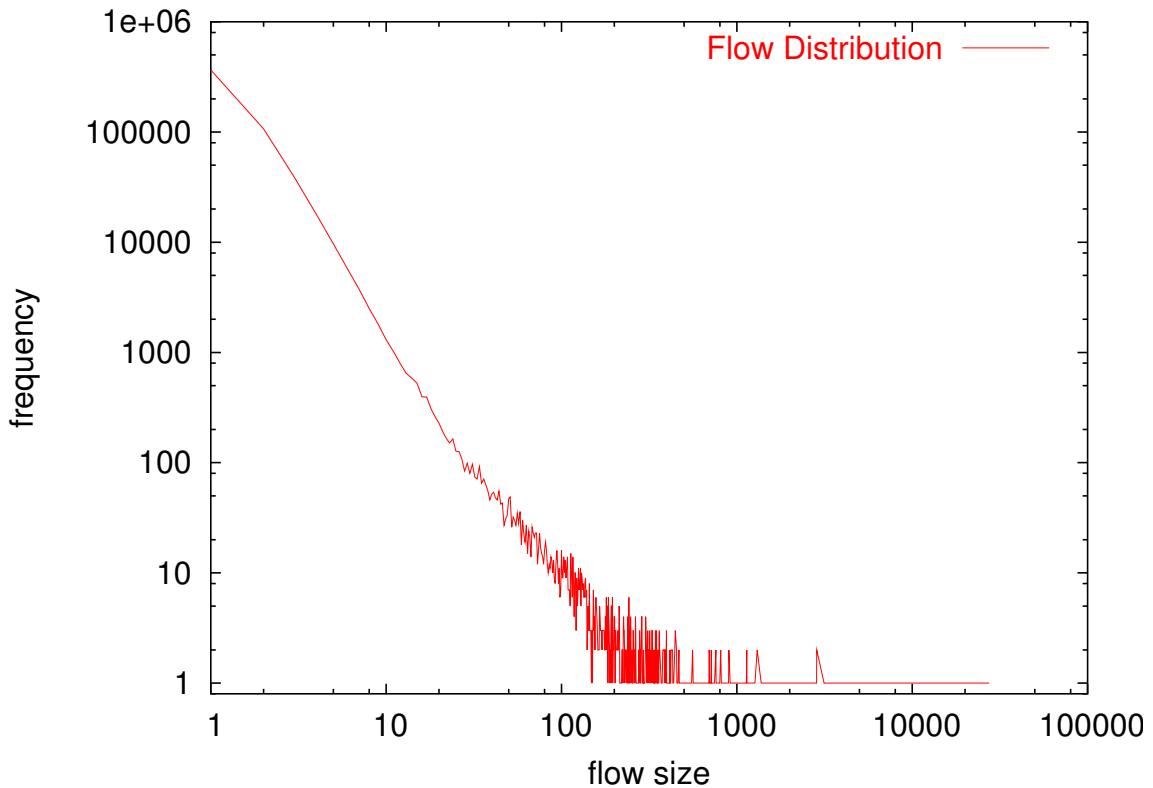
Abhishek Kumar
Minho Sung
Jun (Jim) Xu
Networking and Telecommunications Group
College of Computing
Georgia Institute of Technology
{akumar,mhsung,jx}@cc.gatech.edu

Jia Wang
AT&T Labs - Research
jiawang@research.att.com

Problem Statement

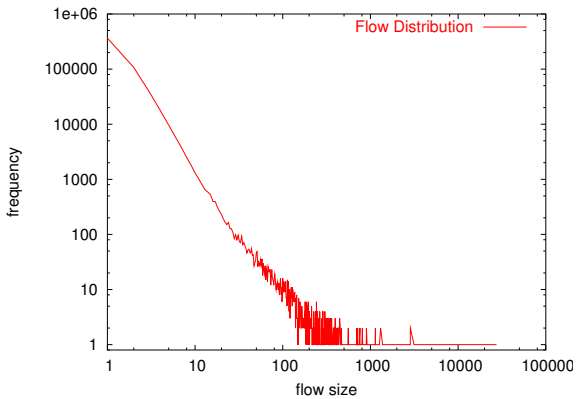
Problem: To estimate the probability distribution of flow sizes. In other words, for each positive integer i, estimate n_i, the number of flows of size i.

Problem Statement



Problem Statement

Problem: To estimate the probability distribution of flow sizes. In other words, for each positive integer i, estimate n_i, the number of flows of size i.



Definition of Flow: All packets with the same flow-label. The flow-label can be defined as any combination of fields from the IP header, e.g <Source IP, source Port, Dest. IP, Dest. Port, Protocol>.

Overview

Motivation

Related work: Inversion of sampled traces

System Model

Estimating total flows and flows of size 1

A holistic approach to estimate the entire distribution

A multiresolution extension of the mechanism

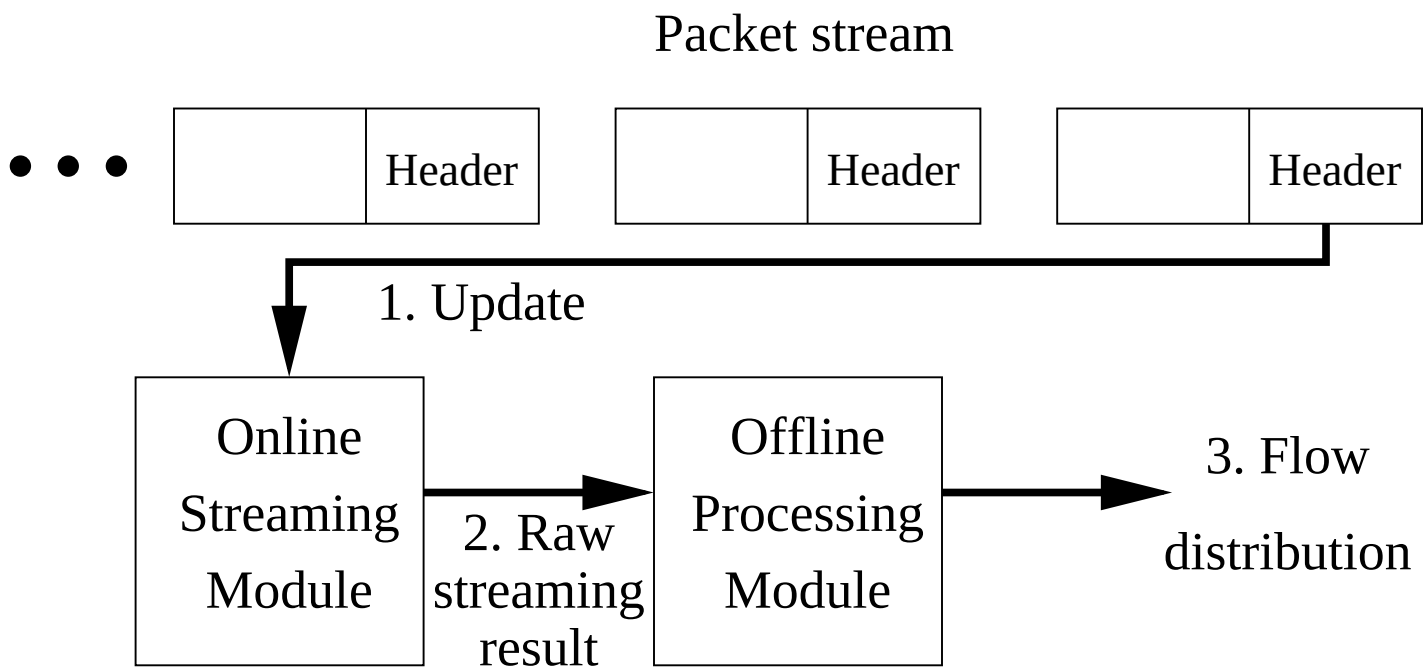
Motivation

- Knowledge of flow-distribution allows us to infer the usage pattern of the network, in terms of:
  - The access bandwidth of the user population.
  - Application types.
- It also helps in detecting anomalous events such as:
  - Incipient worm infections.
  - DDoS attacks.
  - Route flapping.
- Enables other measurement applications, such as traffic matrix estimation.

Related work: Inverting sampled packet traces.

- Current measurement boxes collect traces via packet sampling.
- The approach is to invert the sampled distribution to obtain the actual distribution [Duffield et al., SIGCOMM'03].
- High estimation errors due to low sampling rates.
- Practical limitations to inverting sampled traffic [Hohn & Veitch, IMC'03].

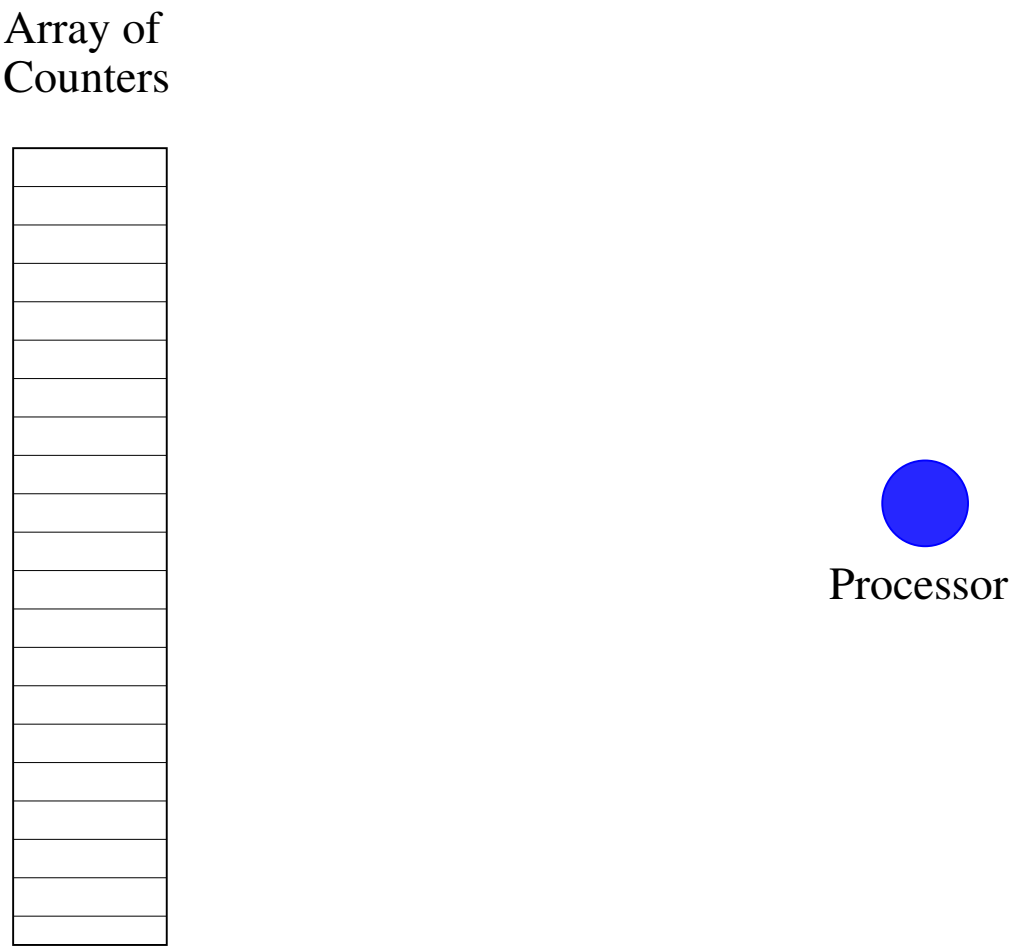
Solution Architecture – System Model



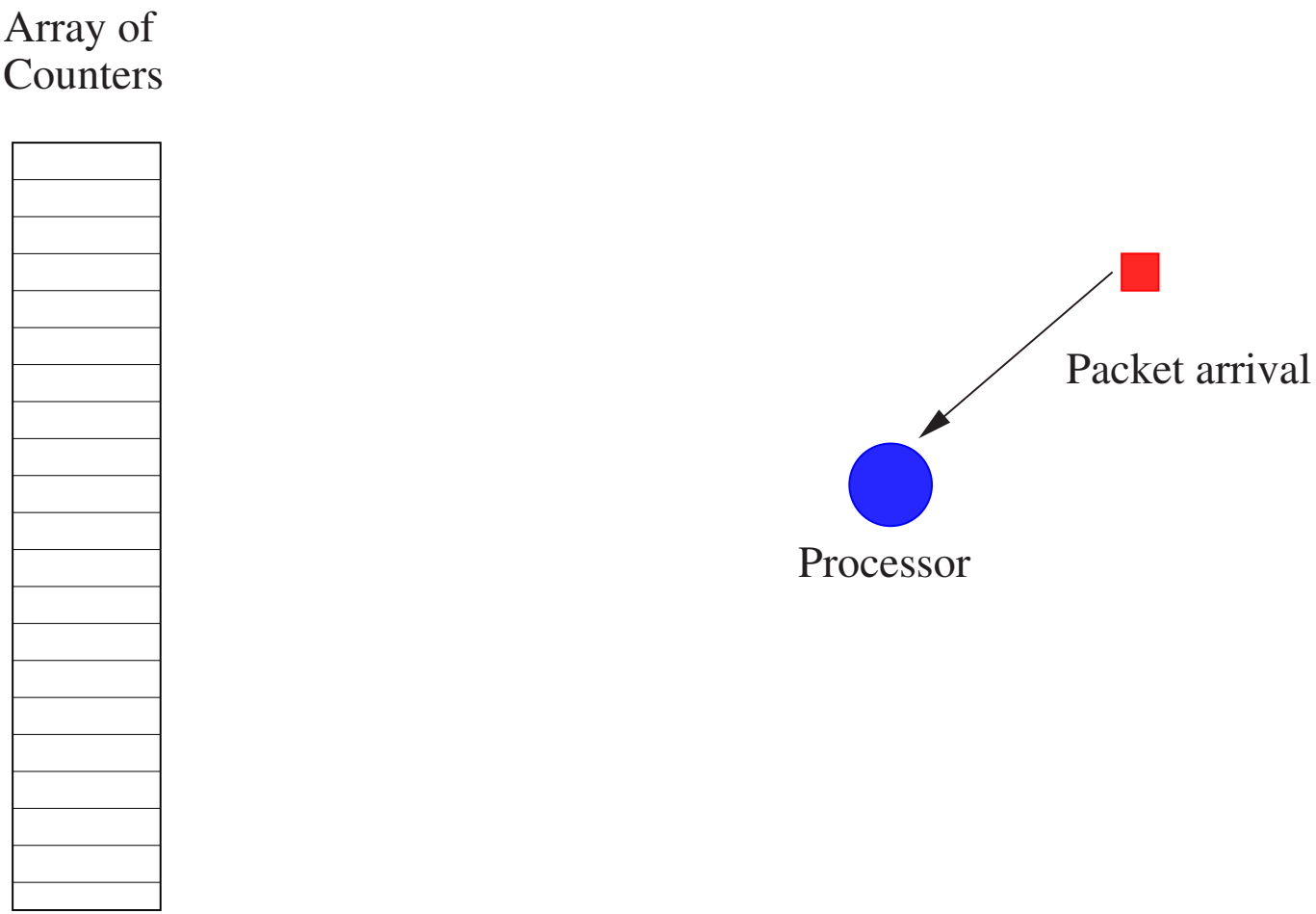
Solution Architecture — Insertion Module

- Measurement proceeds in epochs (e.g. 100 seconds).
- Maintain an array of counters in fast memory (SRAM).
- For each packet, a counter is chosen via hashing, and incremented.
- No attempt to detect or resolve collisions.
- Data collection is lossy (erroneous), but very fast.
- At the end of the epoch, the counter array is paged to disk.

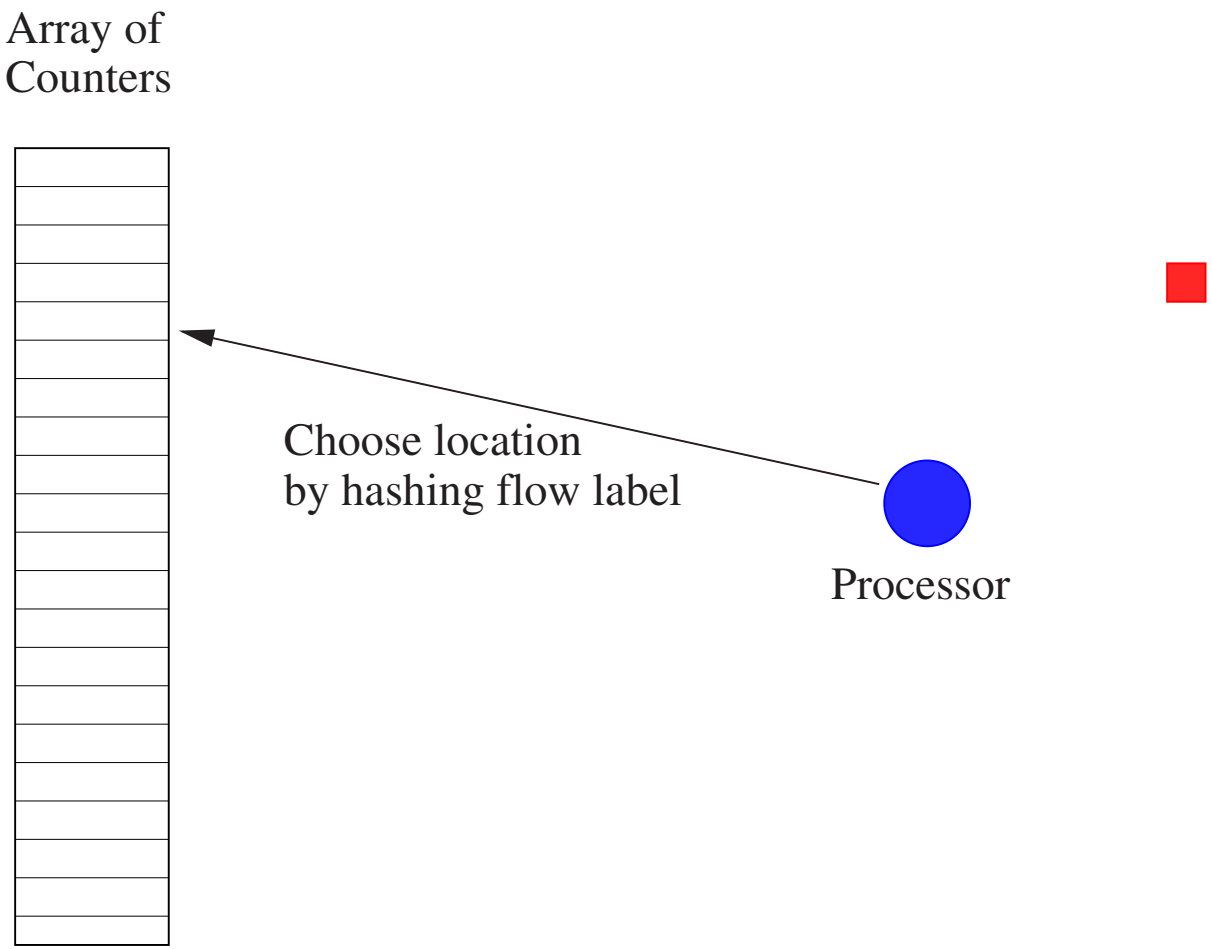
Array of counters



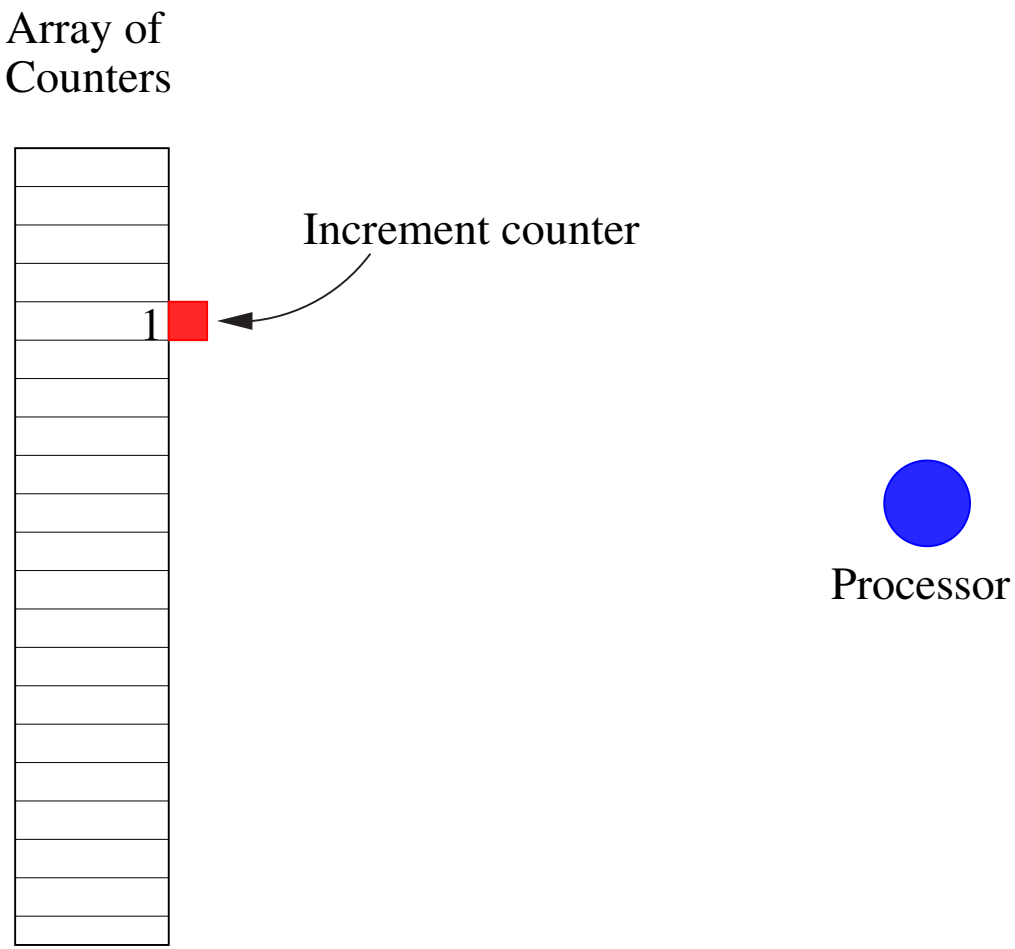
Array of counters



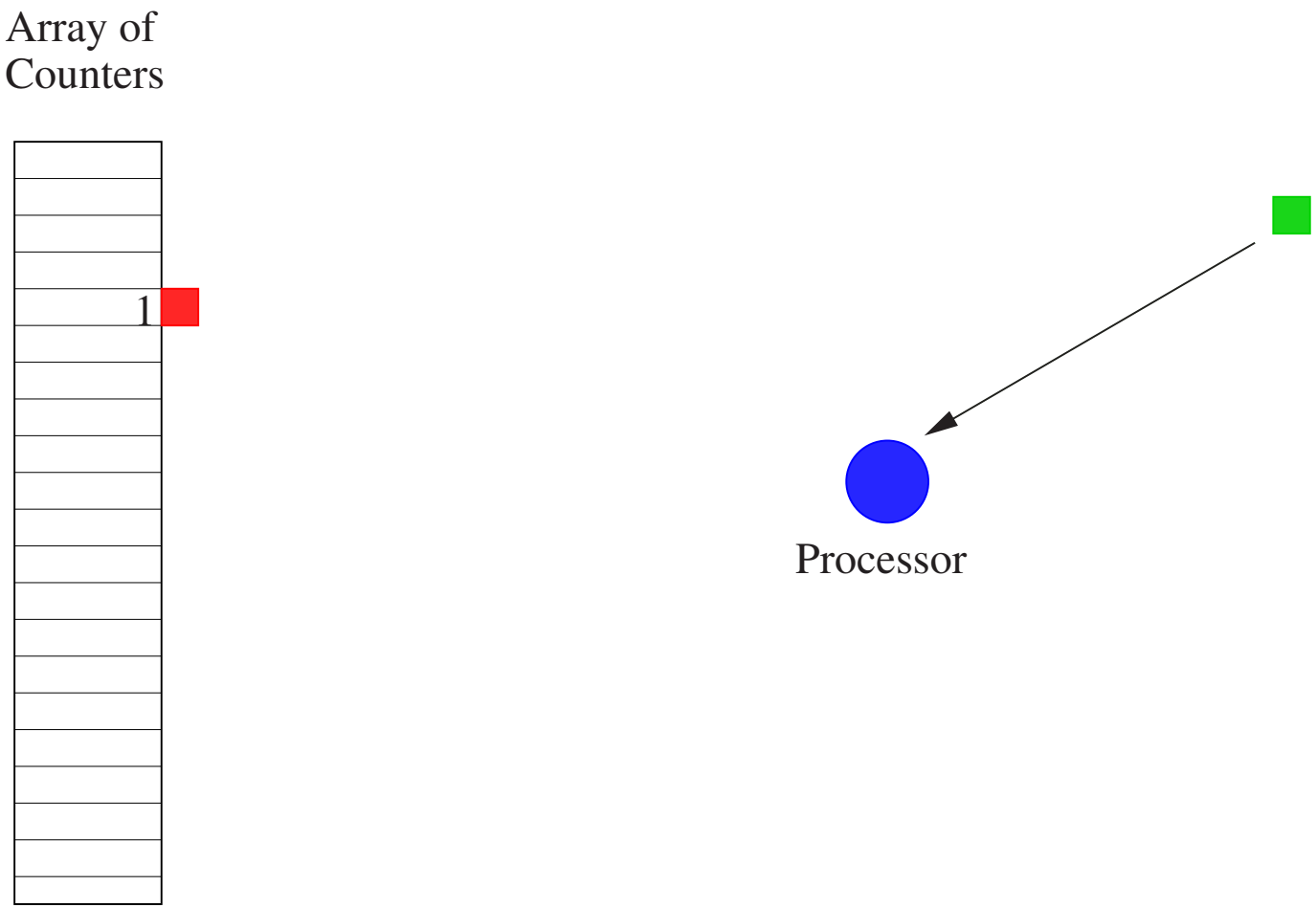
Array of counters



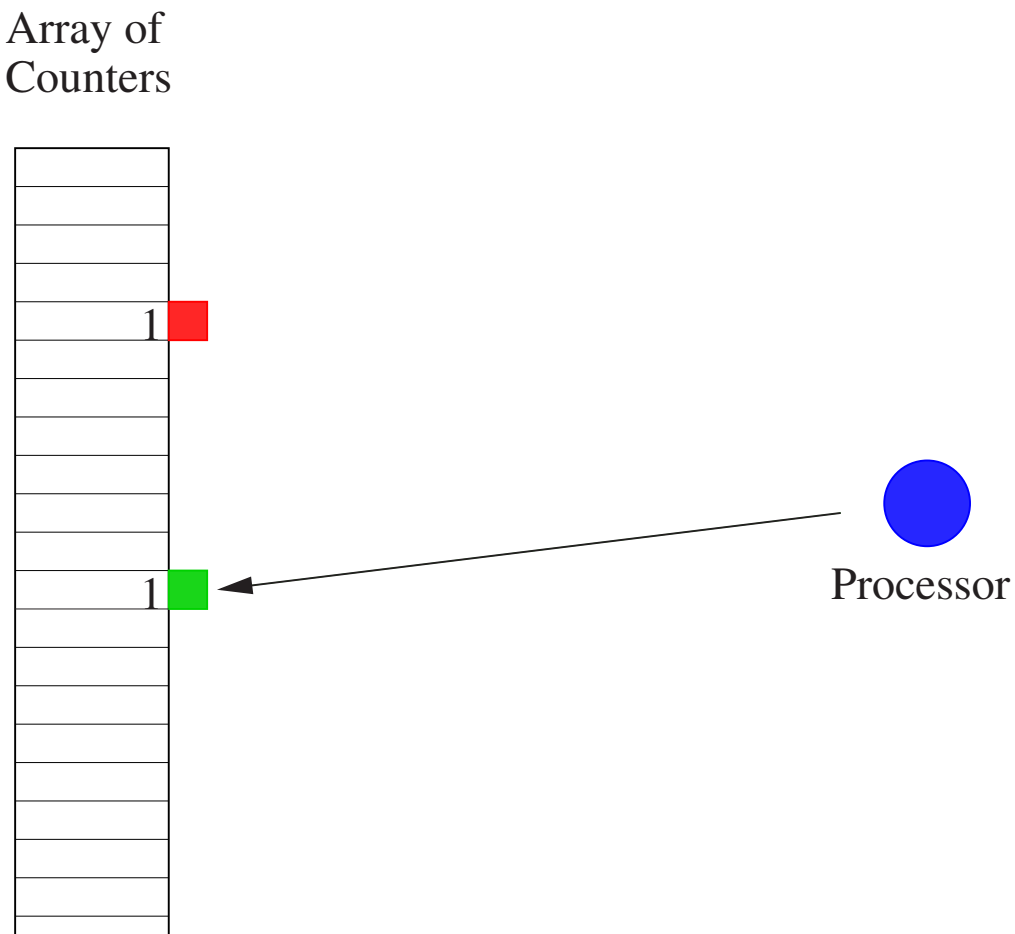
Array of counters



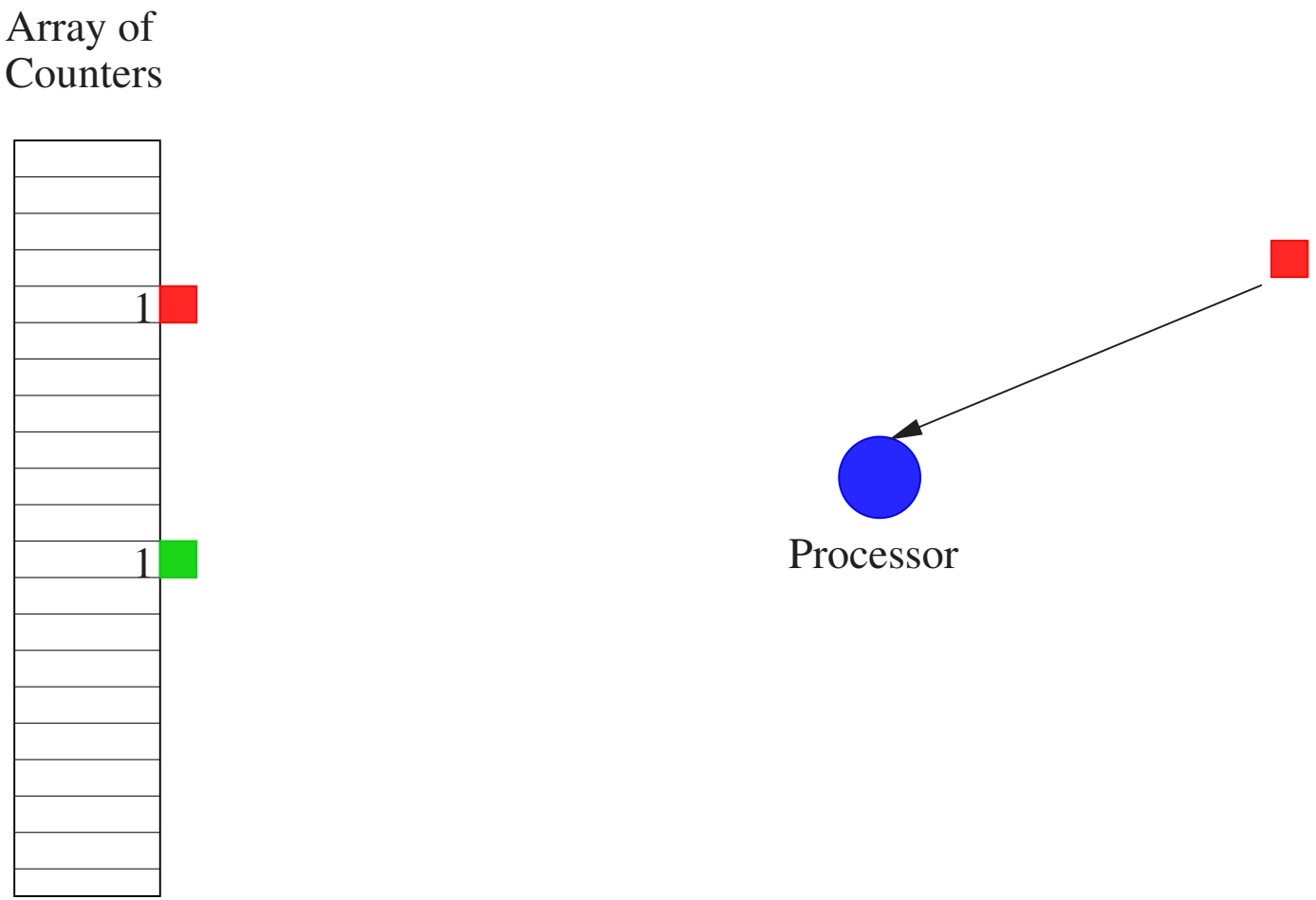
Array of counters



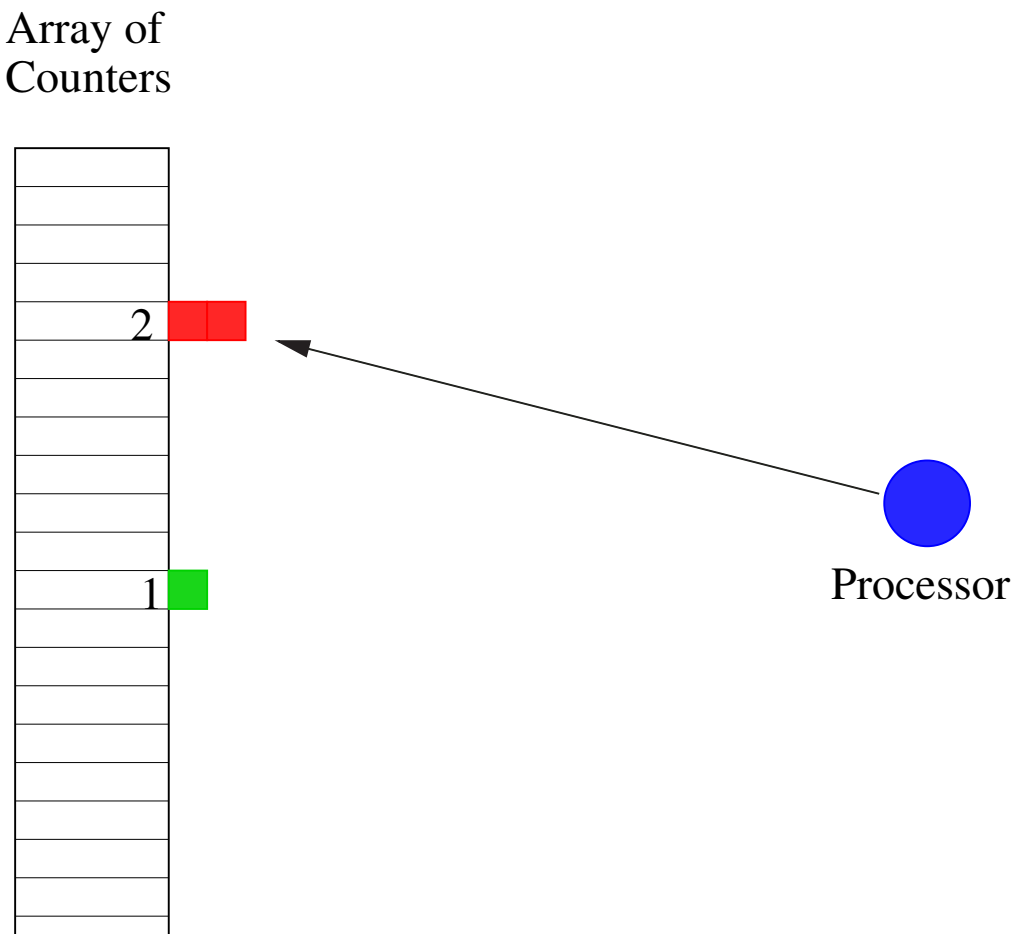
Array of counters



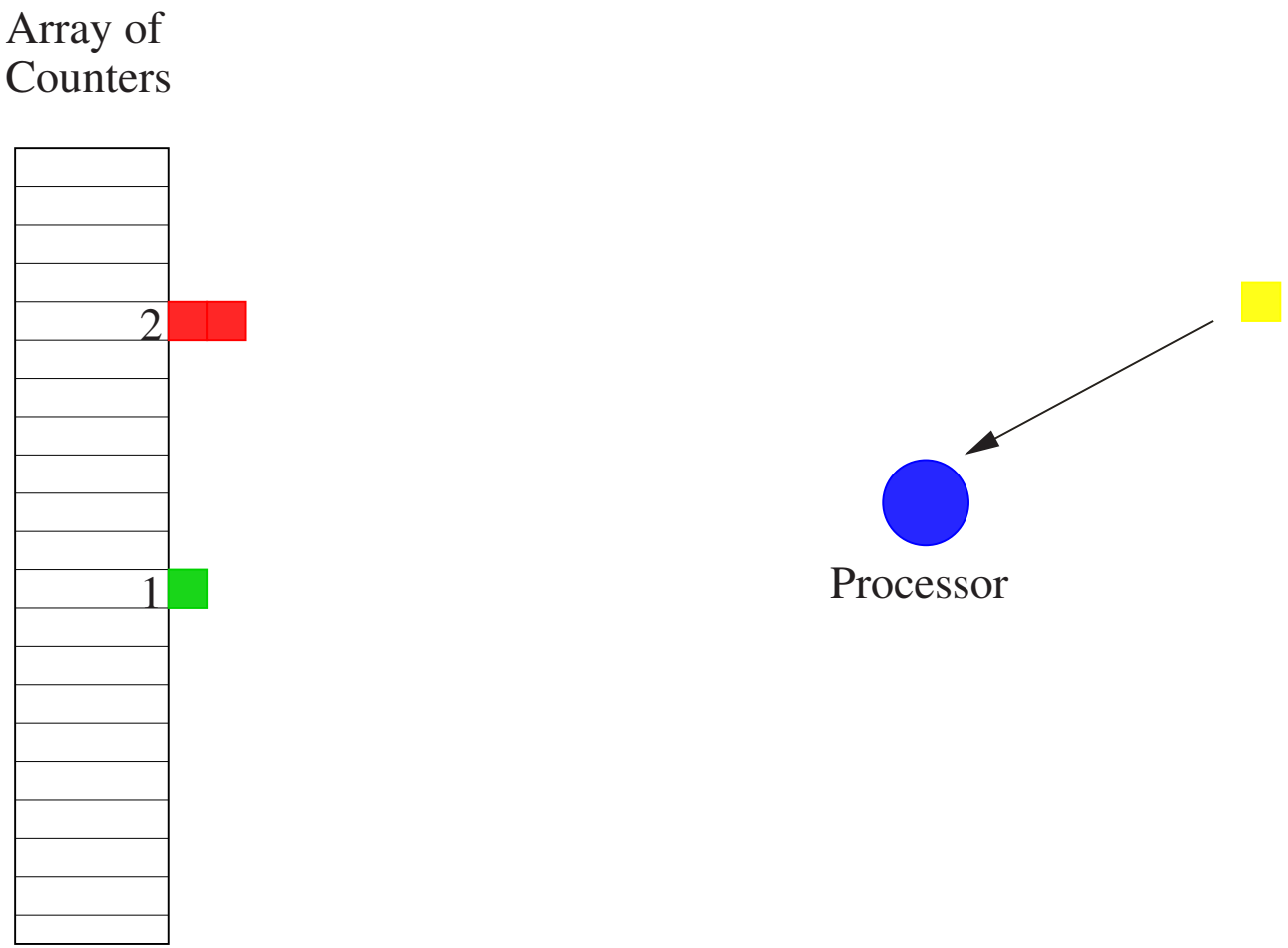
Array of counters



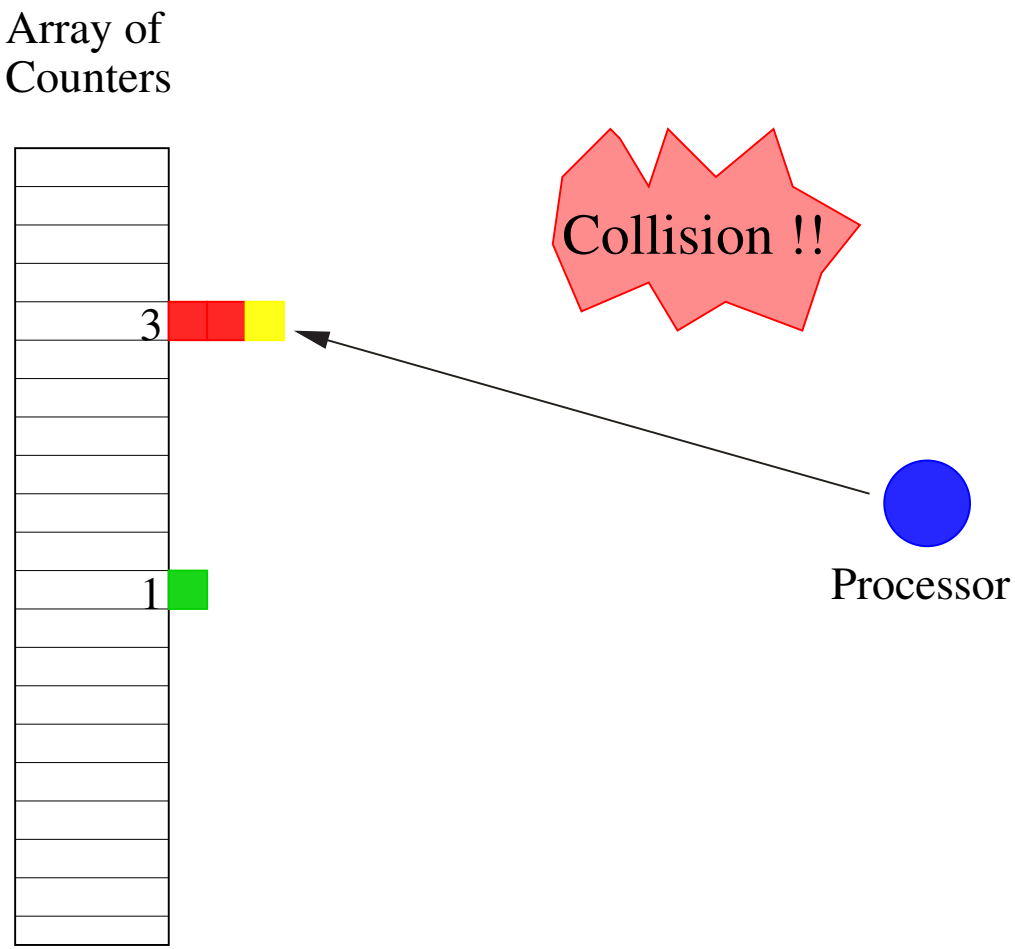
Array of counters



Array of counters



Array of counters



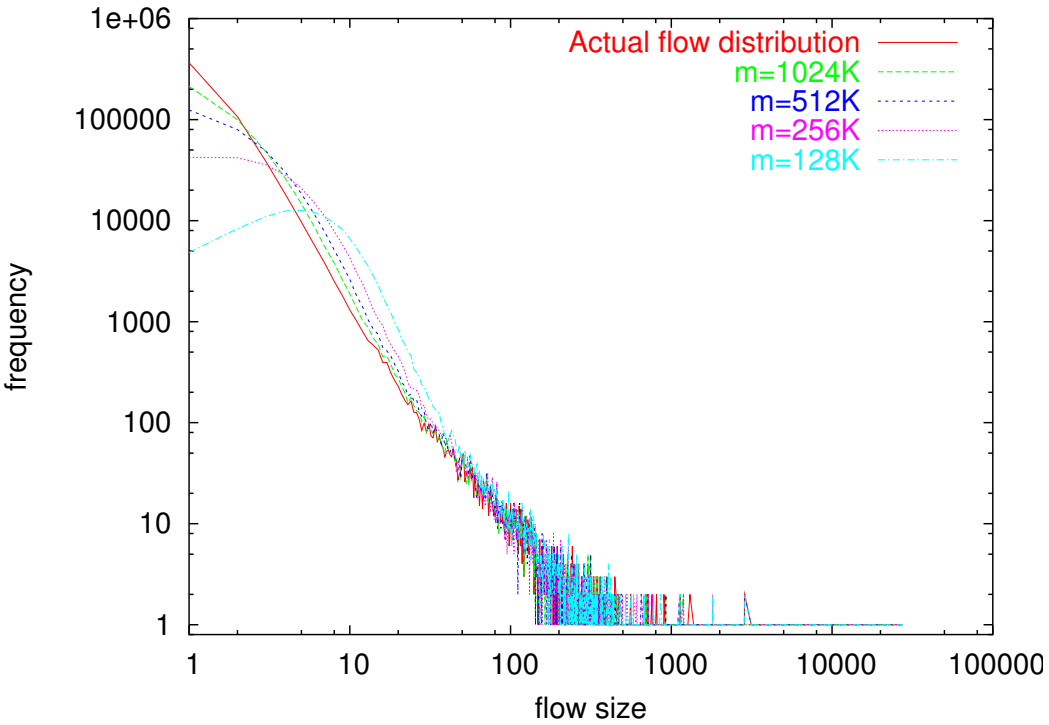
Array of counters — Implementation

- *Efficient Implementation of a Statistics Counter Architecture.* [Ramabhadran & Varghese, SIGMETRICS'03]
- Small (7-bit) counter in fast memory.
- Large (32 or 64 bit) counter in slow memory.
- Perfectly fits our requirements.

Solution Architecture — Estimation Module

- The counter array is processed to obtain the *Counter Value Distribution*. $\{m_0 =$# counters with value '0', $y_1 =$# counters with value '1', $\cdots$, $y_z =$# counters with value 'z'.$\}$
- Use Bayesian statistics to derive the following quantities from the counter value distribution:
  - The total no. of flows, n.
  - The total no. of flows with exactly one packet, n_1.
  - The flow distribution ϕ.

The shape of the “Counter Value Distribution”



The distribution of flow sizes and raw counter values (both x and y axes are in log-scale). m = *number of counters*.

Estimating the no. of total flows, n, and flows of size one, n_1

- Let total number of counters be m.
- The number of flows hashing to any counter c is modeled by the Poisson random variable with parameter $\lambda = n/m$.
- There is a simple estimator for the total number of flows: $n = m \ln \frac{m}{m_0}$.
- This is a standard result, first used for **Probabilistic Counting** by Whang et al.[ACM Trans. Database Sys. 1990].
- We extend this result to derive an estimator for the total number of flows of size 1:

$$n_1 = y_1 e^{\frac{n}{m}} \tag{1}$$

- **Extremely accurate estimates (within $\pm 1\%$).**
- It is difficult to extend this approach for arbitrary i.

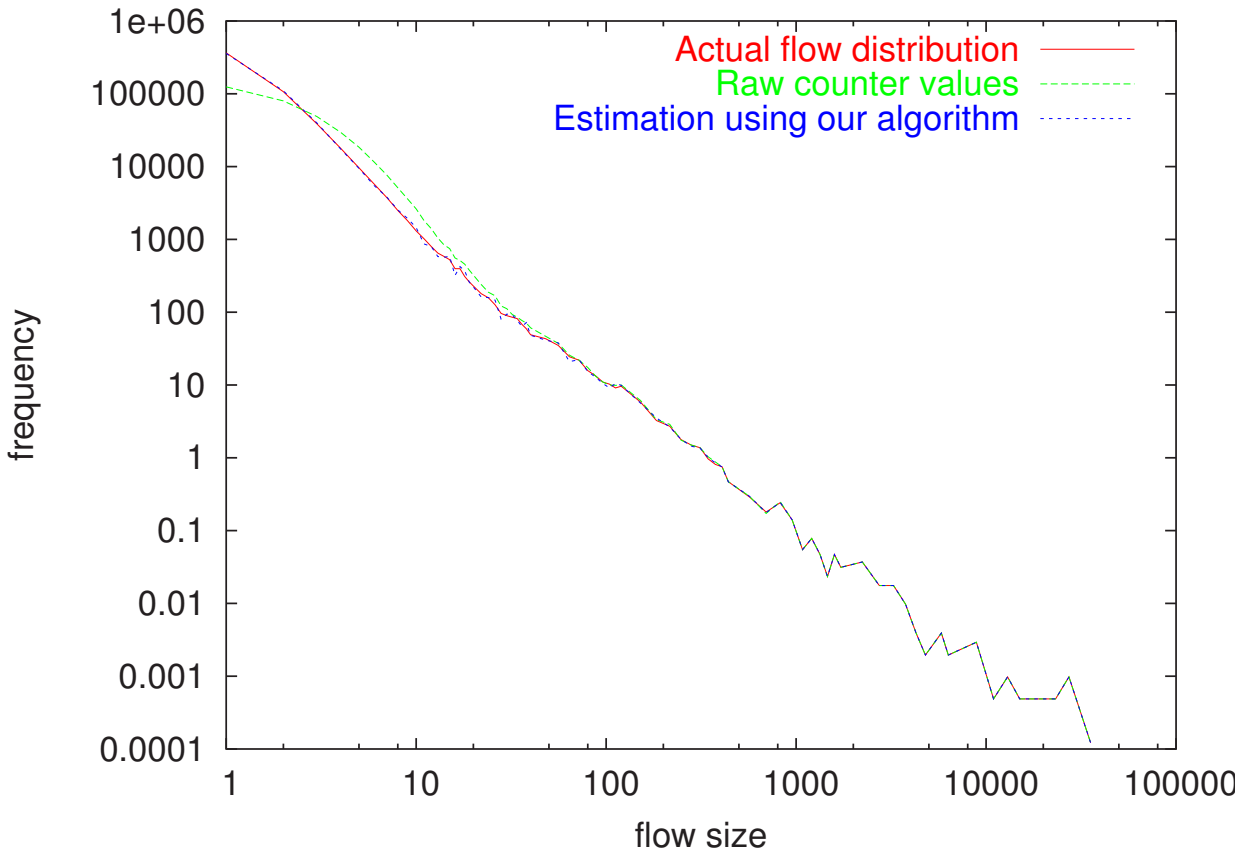
Estimating the entire flow distribution, ϕ

- Begin with a guess of the flow distribution, ϕ^{old}.
- Based on this ϕ^{old}, compute the various possible ways of "splitting" a particular counter value and the respective probabilities of such events.
- This allows us to compute a refined estimate of the flow distribution ϕ^{new}.
- Use $\phi^{old} \leftarrow \phi^{new}$ for the next iteration.
- Repeating this multiple times allows the estimate to converge to a *local maximum*.
- This is an instance of *Expectation maximization*.

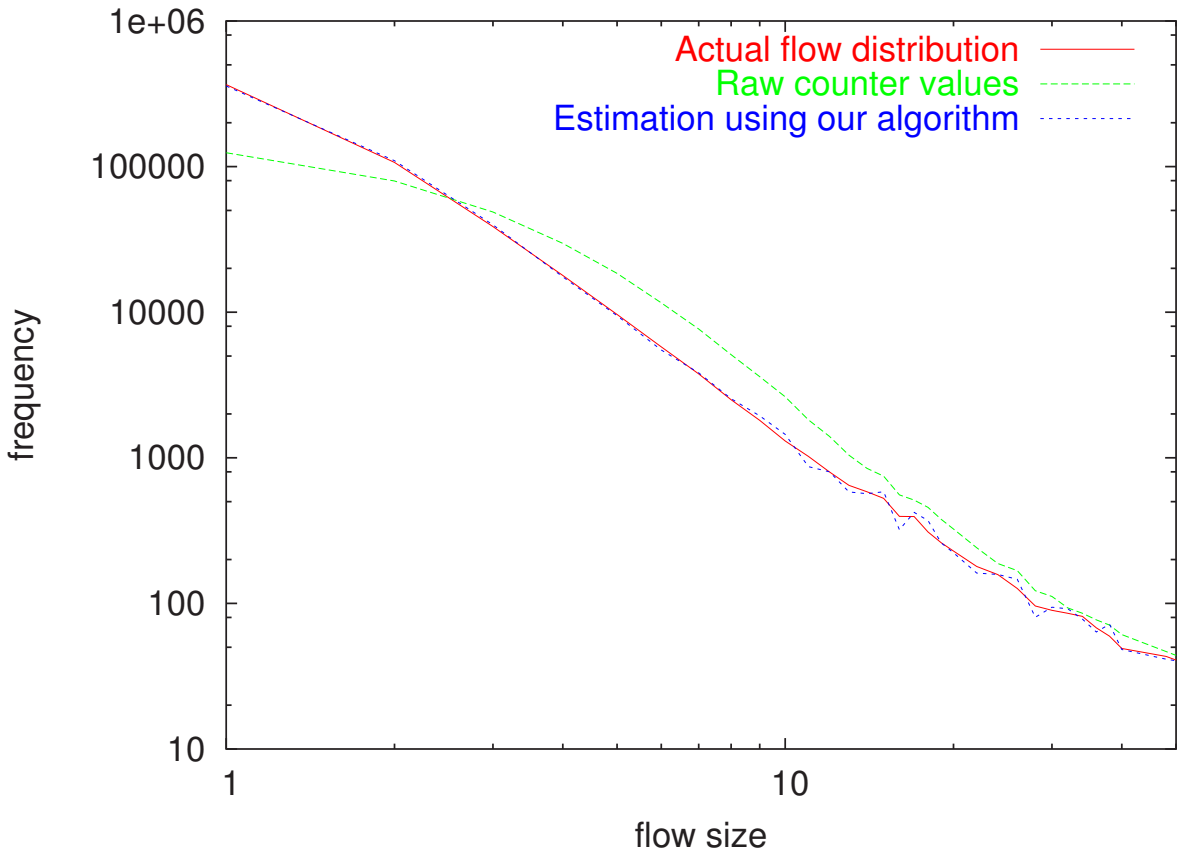
Estimating the entire flow distribution — an example

- For example, a counter value of 3 could be caused by three events:
  - 3 = 3 (no hash collision);
  - 3 = 1 + 2 (a flow of size 1 colliding with a flow of size 2);
  - 3 = 1 + 1 + 1 (three flows of size 1 hashed to the same location)
- Suppose the respective probabilities of these three events are 0.5, 0.3, and 0.2 respectively, and there are 1000 counters with value 3.
- Then we estimate that 500, 300, and 200 counters split in the three above ways, respectively.
- So we credit 300 * 1 + 200 * 3 = 900 to n_1, the count of size 1 flows, and credit 300 and 500 to n_2 and n_3, respectively.

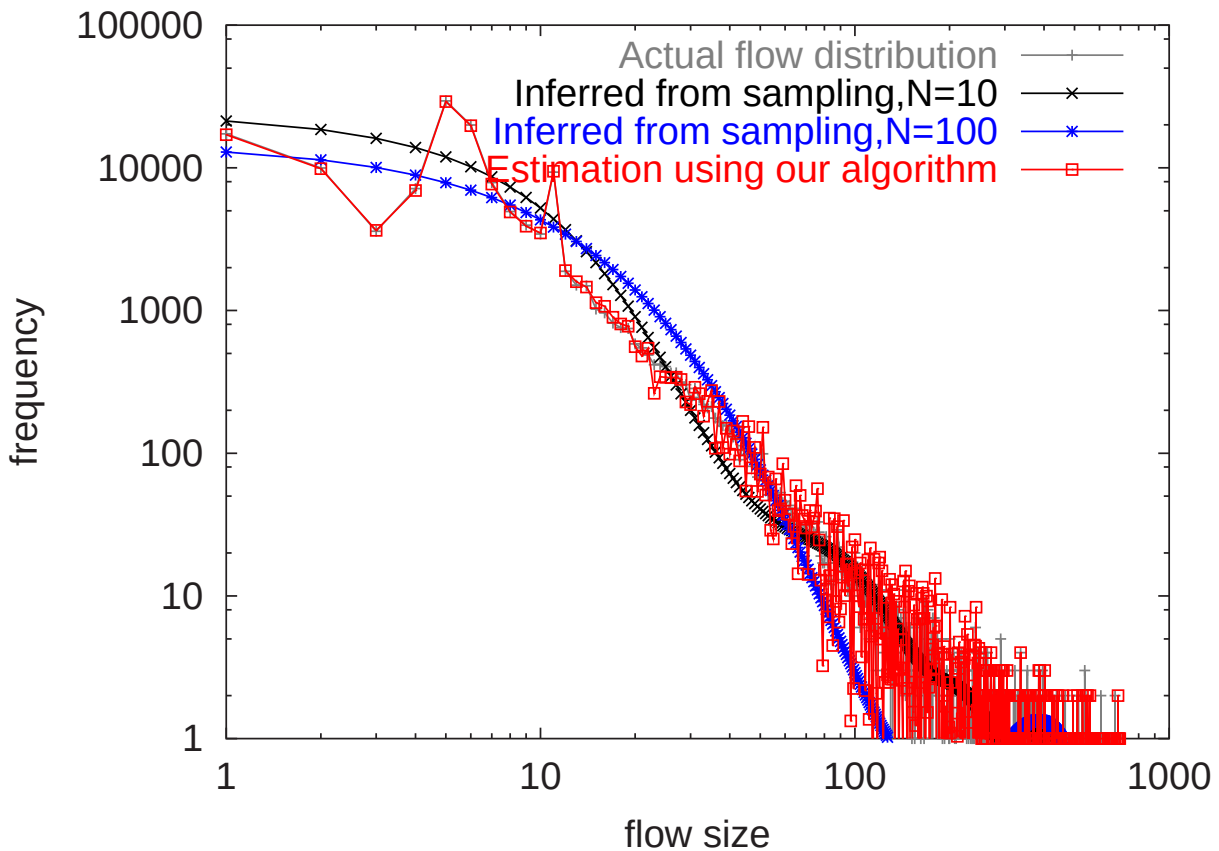
Evaluation — Before and after running the Estimation algorithm



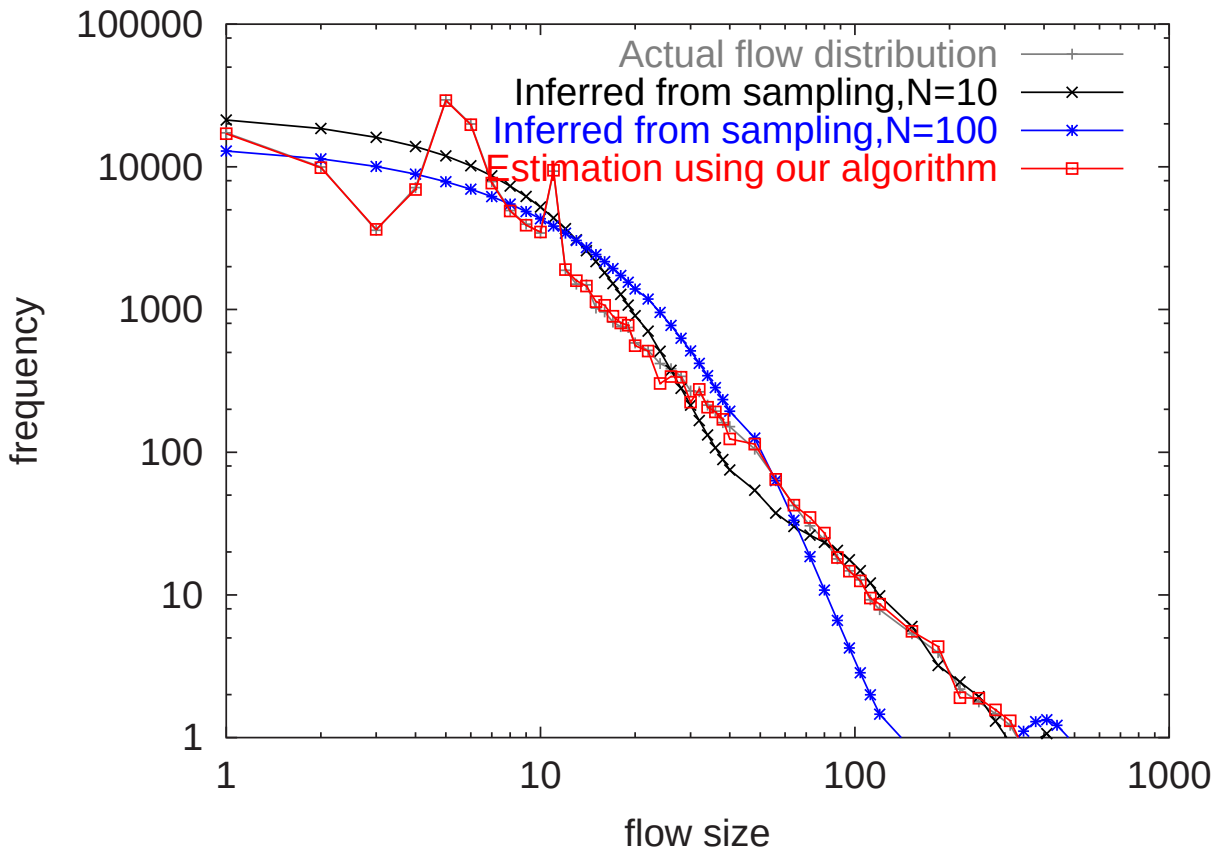
Estimation for small flow sizes



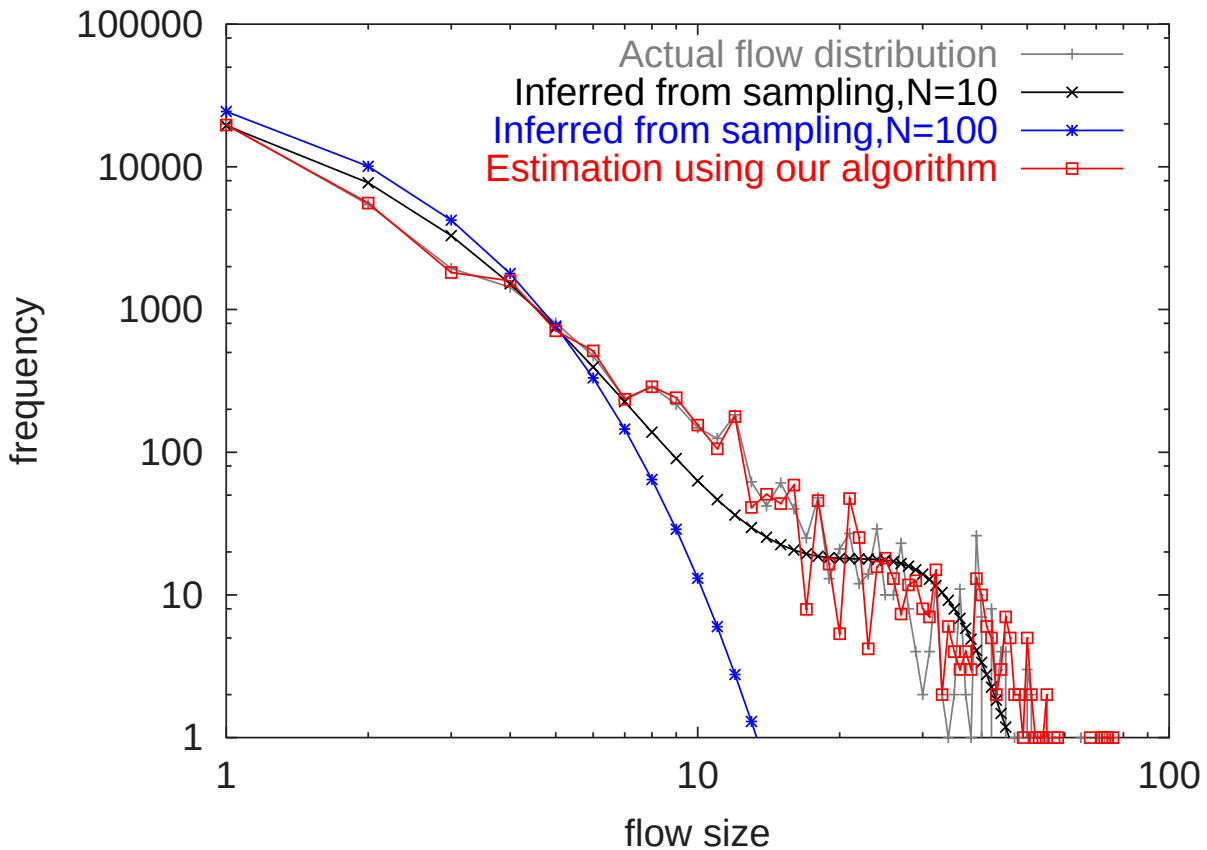
Sampling vs. array of counters – Web traffic.



Replot with bucket-based smoothing – Web traffic.

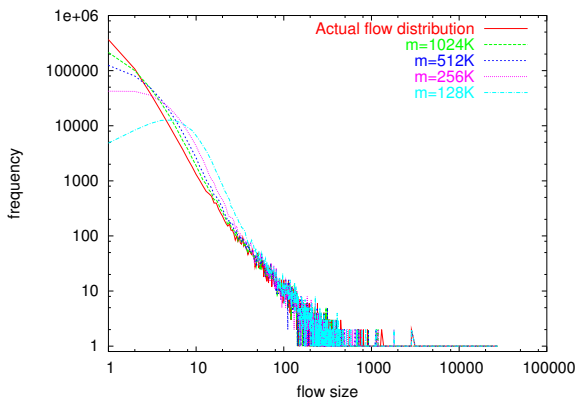


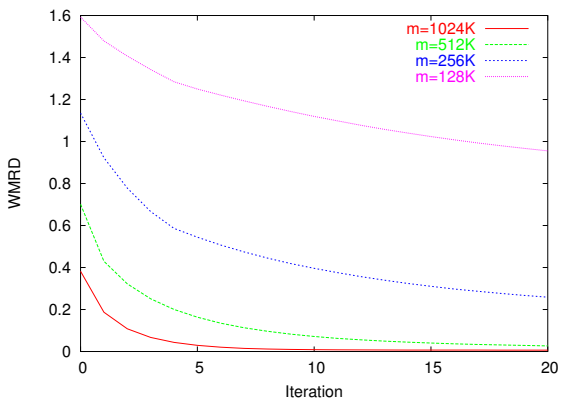
Sampling vs. array of counters – DNS traffic.



Variability in the number of flows

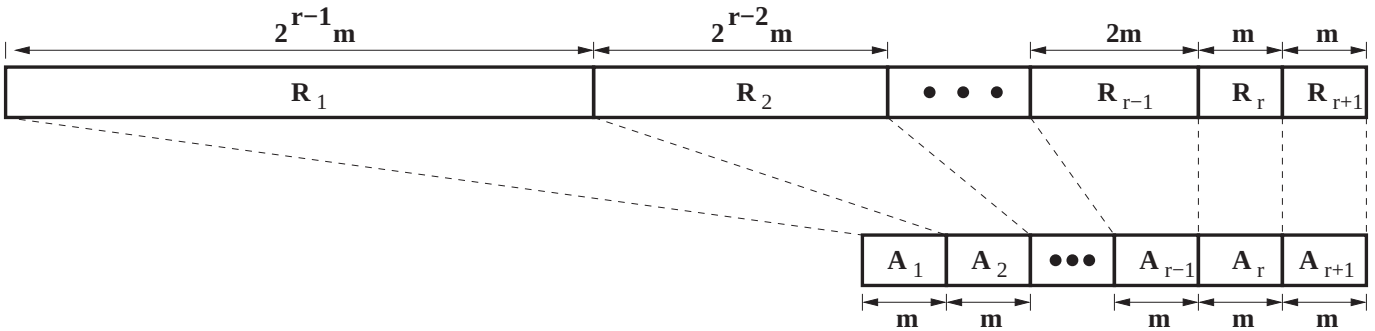
- The total number of flows can change by orders of magnitude.
- Our mechanism is sensitive to the "load factor" n/m.





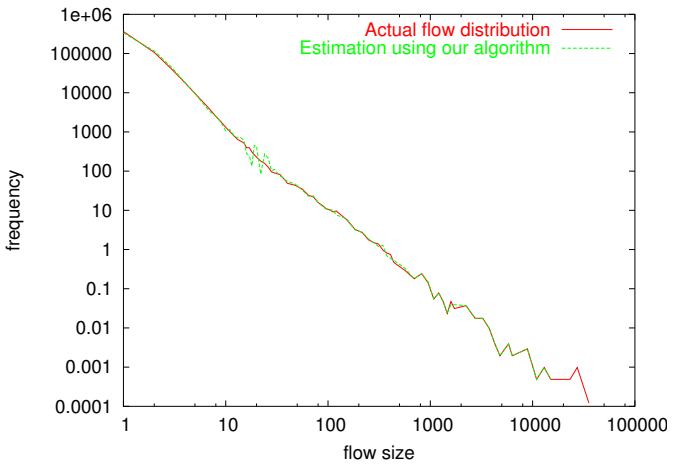
- To accommodate all possible values of n, we extend our work to a multi-resolution version.

The multi-resolution array of counters

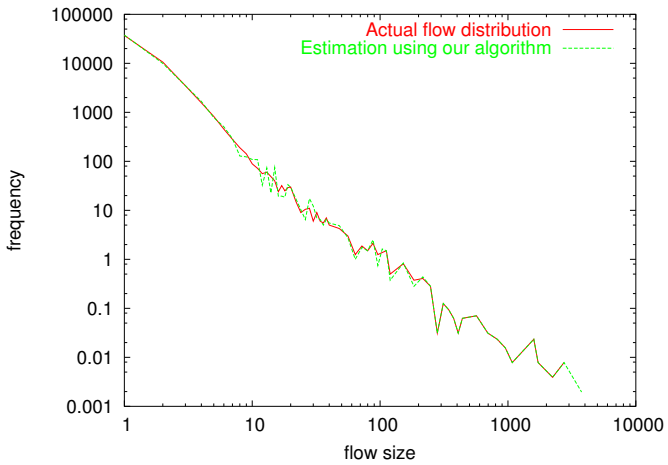


- The multi-resolution array of counters allows our scheme to operate for any value of n, with graceful degradation in accuracy for large number of flows.

Original and estimated distributions using MRAC.



(c) Trace “Long” (563,080 flows).



(d) Trace “Short” (55,515 flows).

Conclusions

- Data-Streaming based solution for estimatong flow-distribution.
- *Lossy data structure + Bayesian statistics = Accurate streaming*
- Fast but “lossy” data-collection.
- Estimation using EM algorithm.
- An order of magnitude of improvement in estimation accuracy over sampling based solutions.
- Multiresolution version allows a tradeoff between storage cost and accuracy.

Future Work

- Reusing EM results from preceeding epochs to speed up the computation.
- A “sliding-window” version that removes the restriction of fixed “epochs”.
- Evaluating the suitability of the mechanism for other “uncommon” distributions.

Acknowledgments

- Dr. Nick Duffield — for generously sharing the data from his work on inverting sampled traces.
- Dr. Oliver Spatschek — for providing us the Internet packet header traces collected by Gigascope servers.
- NLANR — For the publicly available traces.
- The anonymous reviewers, for their valuble inputs.

Acknowledgments

- Dr. Nick Duffield — For generously sharing the data from his work on inverting sampled traces.
- Dr. Oliver Spatschek — for providing us the Internet packet header traces collected by Gigascope servers.
- NLANR — For the publicly available traces.
- The anonymous reviewers, for their valuble inputs.
- And, the Awards Committee for giving us the Best Student Paper Award !!

Thank You !