Classic Network Measurements meets Software Defined Networking

Ran Ben Basat, Technion, Israel

Joint work with Gil Einziger and Erez Waisbard (Nokia Bell Labs)

Roy Friedman (Technion) and Marcello Luzieli (UFGRS)
Network Measurements

220.6.67.9

188.13.7.2

181.7.20.6

181.7.20.6

188.13.7.2

188.13.7.2

181.7.20.6

181.7.20.6
Network Measurements

- Elephant Flows Detection
- Load Balancing
- Traffic Engineering
- Customer Satisfaction
- Counting Distinct Elements
- Link-based SEO

Estimating the fraction of rare flows
- DDoS Detection
- Link Utilization
- Trend Detection

Averaging over Sliding Windows

Computing Quantiles
- Data Log Analysis
- Network Health Monitoring

DDoS Identification
Worm Propagation
frequency estimation

How many packets has sent?

Classic Setting:
Can’t allocate a counter for each flow!

In software:
Memory is cheap (to an extent)
Speed is the new bottleneck
Agenda

- Classical Frequency Estimation Algorithms.
- Accelerating space-oriented algorithms.
- Hierarchical Heavy Hitters.
Count Min Sketch (Cormode and Muthukrishnan)

\[ \frac{2}{\epsilon} \text{ counters} \]

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ E[\text{Noise}] \leq N\epsilon/2 \]

\[ \Pr[\text{Noise} > N\epsilon] \leq \frac{1}{2} \]

Query(\( \epsilon \)) = 2

Overall \[ \Pr[\text{Noise} > N\epsilon] \leq \delta \]
Space Saving (Metwally et al.)

- For a N-sized stream, when allocated with $m$ counters, Space Saving guarantees:
  - $\text{Query}(x)$ returns $\min(f_x, \frac{N}{m})$
  - $\text{Query}(x)$ returns 2

- For achieving an $N\varepsilon$ approximation, $\lceil 1/\varepsilon \rceil$ counters are needed.
Agenda

- Classical Frequency Estimation Algorithms.

- Accelerating space-oriented algorithms.

- Hierarchical Heavy Hitters.
Start with a $N$ sized stream and a $(\epsilon, \delta)$ measurement algorithm.

Get an $O(\log N \cdot \epsilon^{-2} \log \delta)$ sized stream and a $(2\epsilon, 3\delta)$ measurement algorithm.

We can iteratively accelerate ourselves!
Empirical Evaluation

- Graph 1: Speed vs. Stream Length (Packets)
  - CMS
  - CS
  - SSH
  - SSL
  - ACC(CMS)
  - ACC(CS)
  - ACC(SSH)

- Graph 2: Speed vs. Algorithm's Accuracy ($\epsilon_a$)
Empirical Evaluation

On-Arrival RMSE [Packets]

Stream Length [Packets]

On-Arrival RMSE [Packets]

Algorithm's Accuracy ($\epsilon_n$)
Distributed SDN Implementation

(a) VM implementation

(b) Dataplane implementation
Agenda

- Classical Frequency Estimation Algorithms.
- Accelerating space-oriented algorithms.
- Hierarchical Heavy Hitters.
Hierarchical Heavy Hitters

181.7.20.1
181.7.20.2
...
181.7.21.1
181.7.21.2
...

Can we block only the attacking devices?
Multi-Dimensional Hierarchies

181.7.20.*

181.7.*.*

220.7.16.*

220.7.16.9
Hierarchical Heavy Hitters using Space Saving (Mitzenmacher et al.)

Compute all prefixes
Constant Time Hierarchical Heavy Hitters (SIGCOMM2017)

Compute a random prefix

181.7.20.*
Constant Time Hierarchical Heavy Hitters (SIGCOMM2017)

With probability 9/10

Compute a random prefix

Ignore packet

Level0 Measurement

Level1 Measurement

Level2 Measurement

Level3 Measurement

Level4 Measurement
• How to detect the minimal prefix networks?

• How long it takes to converge?
Constant Time Hierarchical Heavy Hitters (SIGCOMM2017)

![Graph showing Accuracy Errors and Coverage Errors vs. Number of packets for RHHH and 10-RHHH]
Constant Time Hierarchical Heavy Hitters (SIGCOMM2017)

![Graph showing false positives and updates per second versus stream length and accuracy guarantee.]

- **False Positives (%):** Plotted against stream length on a logarithmic scale.
- **Updates/second [Millions]:** Plotted against accuracy guarantee on a logarithmic scale.

Legend:
- Partial Ancestry
- Full Ancestry
- MST
- 10-RHHH
- RHHH
Constant Time Hierarchical Heavy Hitters (SIGCOMM2017)

HHH Method

Packets per Second ( Millions )

<table>
<thead>
<tr>
<th>Method</th>
<th>OVS Baseline</th>
<th>10-RHHH</th>
<th>RHHH</th>
<th>Partial Ancestry</th>
<th>MST</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>14.88 Mpps</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Packets per Second ( Millions )

<table>
<thead>
<tr>
<th>Method</th>
<th>OVS Baseline</th>
<th>V=10H</th>
<th>V=5H</th>
<th>V=3.3H</th>
<th>V=2.5H</th>
<th>V=2H</th>
<th>V=H</th>
<th>VNF (V=10H)</th>
<th>VNF (V=H)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>14.88 Mpps</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Any Questions