Joint Data Streaming and Sampling Techniques for Detection of Super Sources and Destinations

Qi Zhao  Abhishek Kumar  Jun (Jim) Xu
College of Computing
Georgia Institute of Technology
Definitions and Problem

- Fan-out/Fan-in: the number of distinct destinations/sources a source/destination communicates during a small time interval.

- Super Source and Destination: the sources/destinations having a large fan-out/fan-in.

- Problem: How to detect super sources and destinations in near real-time at a high speed link (e.g., 10 Gbps or 40 Gbps).
Example Motivating Applications

- Detection of port-scans: a port-scan launcher is a super source.

- Detecting DDoS attacks: the victim is a super destination.

- Internet worms: an infected host is a super source.

- "hot spots" in P2P or CDN networks: a busy server/peer is a super destination.
Previous Approaches

- Maintaining per-flow state (Snort and FlowScan): prohibitive fast memory consumption

- Triggered Bitmap scheme [Estan et al. IMC’03]: not complete

- Hash-based flow sampling [Venkataraman et al. NDSS’05]: not accurate
Our Tools to attack the problem

- Network data streaming: process each and every packet passing through to glean the most important information for answering a class of query using a small yet well-organized data structure.

- Sampling: classical technique to reduce the processing load

- Close collaboration between them
Our Schemes

● The simple scheme: filtering after sampling

● The advanced scheme: separating identity gathering and counting
Our Schemes

- The simple scheme: filtering after sampling
- The advanced scheme: separating identity gathering and counting
Limitation of Traditional Hash-based Flow Sampling

Packet pkt

\( h \)

\(<p\)

Hash table

not sampled

Yes

No
Limitation of Traditional Hash-based Flow Sampling

- If a flow is sampled (i.e., hash value $< p$), all packets belonging to it need to be processed by the hash table.

- $p$ has to be much smaller than the ratio between the operating speed of the hash table and the arrival rate of the traffic.

- Small sampling rate leads to large estimation error.
How to Overcome the Limitation?

**Traditional flow sampling:**

Packet Stream → flow sampling → Hash Table → Estimation Result

**Filtering after sampling:**

Packet Stream → flow sampling → Data streaming module → Hash Table → Estimation Result
The Design of the Data Streaming Module

- The data structure is very simple, i.e., a bit array.

```
0 0 0 0 0 0 ・ ・ ・ 0 0 0 0 0
0 1 2                     w−1
```

- The operation is also very simple.
How to Operate?

sampled packet

<table>
<thead>
<tr>
<th>dst</th>
<th>src</th>
</tr>
</thead>
</table>

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

Hash Table
How to Operate?

sampled packet

\[ \text{dst} \quad \text{src} \rightarrow h \]

Hash Table

\[
\begin{array}{cccccccc}
0 & 0 & 0 & 0 & 0 & 0 & \cdot & \cdot & \cdot & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 2 & \quad & \quad & \quad & \quad & w-1
\end{array}
\]
How to Operate?

sampled packet

\[ h \]

\[
\begin{array}{ccccccccc}
0 & 0 & 0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & 0 & 0 & 0 & 0 & 0
\end{array}
\]

Hash Table
How to Operate?

sampled packet

0 0 0 0 0 1 • • • 0 0 0 0 0

0 1 2 w−1

src dst src

h

src src.fan−out += w/u

• • • •
Accuracy

- unbiased estimator

- its approximate variance is given by

\[ \text{Var}[\widehat{F}_s] \approx \frac{\sum_{j=1}^{pF_s} \frac{w-u_j}{u_j}}{p^2} + \frac{F_s(1-p)}{p} \]
Our Schemes

- The simple scheme: filtering after sampling

- The advanced scheme: separating identity gathering and counting
System Model

Packet stream

1. Update
2. Sample

Online Streaming Module

Identity Sampling Module

3. Data digest
4. Query

Estimation Algorithm

5. Estimation Result
Online streaming module

2D bit array A
Identity sampling module

- The purpose is to capture the identities of potential super sources that will be used to look up the previous 2D bit array to obtain their fan-out estimations.

- Use aforementioned filtering after sampling technique

- Use different recording strategy: only record the sampled source identities sequentially in DRAM instead of constructing a hash table.
Estimation module

- Given a source $s$, we compute $h_i(s)$, $i = 1, 2, 3$, to obtain 3 column vectors $A_1, A_2$ and $A_3$ (viewed as bit arrays).

- Let the set of $<\text{src, dst}>$ pairs hashed into $A_i$ be $T_i$. $F_s$, the fan-out of $s$, is approximately equal to $|T_1 \cap T_2 \cap T_3|$.

- If there are more than one sources hashed into the same 3 columns $A_1, A_2, A_3$, $|T_1 \cap T_2 \cap T_3|$ actually is the sum of the fan-outs of all these sources.
Estimation module (Con’t)

- Whang et al. proposed a fairly accurate estimator of $|T_i|$ based on $A_i$ in 1990, say,

\[
\hat{\mid T_i \mid} = m \ln \frac{m}{U_{T_i}} \tag{1}
\]

where $m$ is the size of $A_i$ and $U_{T_i}$ denotes the number of “0”s in $A_i$.

- According to the inclusion and exclusion principle, we have

\[
\hat{F}_s \approx |T_1 \cap T_2 \cap T_3| \\
\approx |T_1| + |T_2| + |T_3| - |T_1 \cup T_2| - |T_1 \cup T_3| - |T_2 \cup T_3| + |T_1 \cup T_2 \cup T_3|
\]
Accuracy

- almost unbiased

its approximate variance is given by

\[
\text{Var}[\hat{F}_s] \approx -m \sum_{i=1}^{3} f(t_{T_i}) - m \sum_{1 \leq i_1 < i_2 \leq 3} f(t_{T_{i_1} \cup T_{i_2}}) \\
+ 2m(f(t_{T_1 \cup (T_2 \cap T_3)}) + f(t_{T_2 \cup (T_1 \cap T_3)}) + f(t_{T_3 \cup (T_2 \cap T_1)})) \\
+ 2m \sum_{1 \leq i_1 < i_2 \leq 3} f(t_{T_{i_1} \cap T_{i_2}}) \\
- 2m(f(t_{T_1 \cap (T_2 \cup T_3)}) + f(t_{T_2 \cap (T_1 \cup T_3)}) + f(t_{T_3 \cap (T_2 \cup T_1)})) \\
+ mf(t_{T_1 \cup T_2 \cup T_3})
\]

where \( f(t) = e^t - t - 1 \).
Evaluation

- Traces from USC, UNC and NLANR

- Simulate our schemes running on a fully utilized OC-192 (10 Gbps) using 128KB SRAM.

- For the simple scheme we set a flow sampling rate of 25% which is required to fit in 128KB SRAM.

- Both online streaming model and identity sampling module of the advanced scheme have enough speed to process the incoming traffic with 100% sampling.
The simple scheme v.s. The advanced scheme
The advanced scheme
Conclusion

- filtering after sampling: the simple scheme

- separating identity gathering and counting: the advanced scheme
Thank you!

- We thank to IMC and GTISC for supporting our travel.

- Questions?