SketchVisor

Robust Network Measurement for Software Packet Processing

Qun Huang, Xin Jin, Patrick P. C. Lee, Runhui Li, Lu Tang, Yi-Chao Chen, Gong Zhang
Huawei, Johns Hopkins University, The Chinese University Hong Kong

Presentation Outline

Context

- Network Measurement
- Sketches

Paper

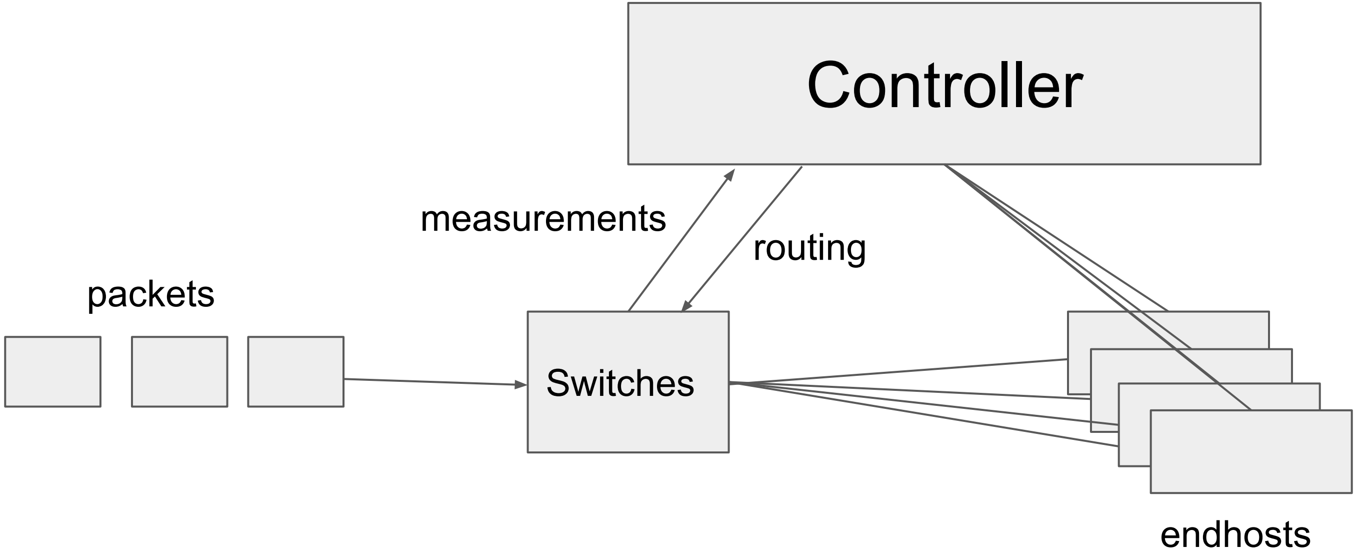
- Design of SketchVisor
    - Data Plane
    - Control Plane
- Evaluation
- Conclusion

Defining Measurement: Traffic Statistics

For each epoch, we want to detect or measure one or several of the following:

- Heavy Hitter
- Heavy Changer
- DDos
- Superspreader
- Cardinality
- Flow size distribution
- Entropy

Defining Measurement: Problem Statement



Measurement Example: Heavy Hitter

Heavy Hitter

A flow whose byte count exceeds a threshold (during an epoch).

Measurement Example: Heavy Hitter

Heavy Hitter

A flow whose byte count exceeds a threshold (during an epoch).

Result: a list of flows (5 tuples)

Need to collect: byte count per flow

Measurement Example: Heavy Hitter

Heavy Hitter

A flow whose byte count exceeds a threshold (during an epoch).

Result: a list of flows (5 tuples)

Need to collect: byte count per flow

Naive algorithm:

A table with all flows and corresponding byte counts. When new packet arrives, add byte count to the associated flow.

Measurement Qualities Tradeoff

- Naive Counting Table
  - very accurate
  - bad performance
  - high memory usage
- Packet Sampling
  - better performance
  - much less accuracy

Measurement Qualities Tradeoff

- Naive Counting Table
  - very accurate
  - bad performance
  - high memory usage
- Packet Sampling
  - better performance
  - much less accuracy
- Sketches
  - less memory usage
  - slightly less accurate
  - **until now: bad performance under stress**

Count-Min Sketch for HH: Hashmap

hash range **w** →

hashfunctions **d** ↓

h0	0	22	12	14	22
h1	32	32	12	22	12
h2	22	12	28	2	24
h3	14	18	3	12	36

Count-Min Sketch for HH: Adding a Packet

id: 5-tuple

size: 8

hash range **w** →

hashfunctions **d** ↓

h0	0	22 +8	12	14	22
h1	32	32	12 +8	22	12
h2	22	12	28	2 +8	24
h3	14	18	3 +8	12	36

Count-Min Sketch for HH: Querying

*What is the bytecount of 5-tuple **x**?*

hash range **w**

hashfunctions **d**

h0	0	30	12	14	22
h1	32	32	20	22	12
h2	22	12	28	10	24
h3	14	18	11	12	36

Count-Min Sketch for HH: Querying

What is the bytecount of 5-tuple ***x****?*

hash range **w**

hashfunctions **d**

h0	0	30	12	14	22
h1	32	32	20	22	12
h2	22	12	28	10	24
h3	14	18	11	12	36

Conclusion on Sketches

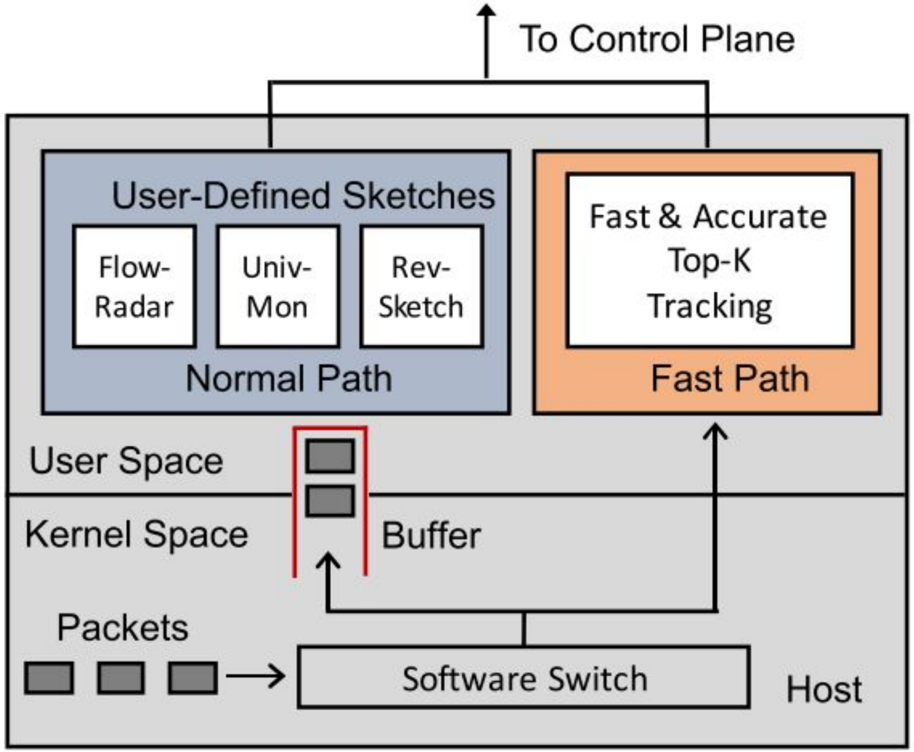
- Sketches use a small amount of memory
- Query approximates the solution by identifying boundaries
- Relatively good performance due to few operations
- Variety of different Sketches can be used simultaneously

BUT

- Under high traffic: Sampling often still necessary, which reduces accuracy.

SketchVisor: **Robust** Network Measurement for Software Packet Processing

Data Plane: Overview



- use several sketches to accurately measure statistics
- if the buffer for the normal path is full the fast path is used instead

Data Plane: Fast Path

Idea: Only track the most influential flows (highest bytecounts).
Drop all other flows.

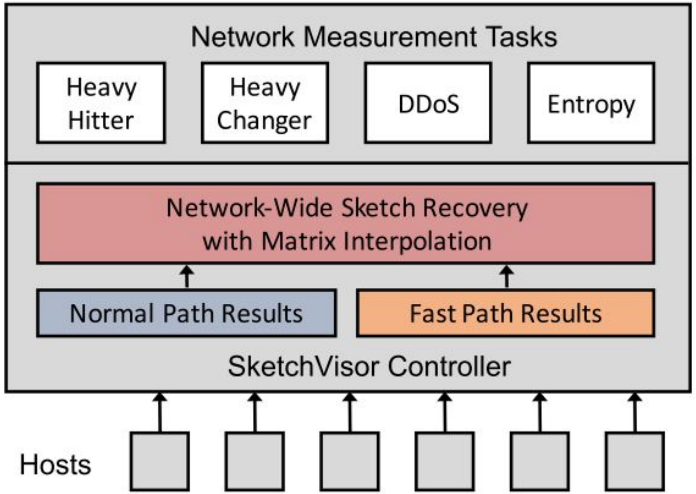
Fast path algorithm (section 4.2), key points:

- estimates byte counts for the top-k flows
- stores additional data for defining bounds
- estimation error upper bound for any flow:
  O(V / k) V: sum of the byte counts of all packets in fast path

Control Plane: Network-Wide Recovery

- Control Plane collects the measurements from all switches

The goal is to recover a total sketch **T,** as if the fast path was tracked by a sketch.



(b) SketchVisor control plane.

Control Plane: Compressive Sensing

Approximations for compressive sensing:

- T is a low rank matrix
- x and sk(x) are sparse
- y and sk(y) are of small noise

$$T = N + sk(x + y)$$

LENS objective function:

$$\text{minimize:} \quad \alpha \|T\|_* + \beta \|\mathbf{x}\|_1 + \frac{1}{2\gamma} \|\mathbf{y}\|_F^2$$

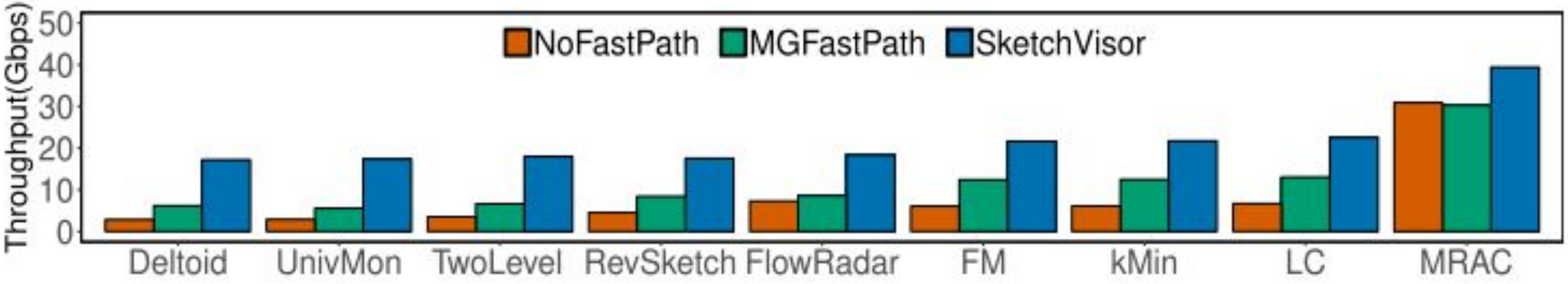
Solving it with the Alternating Direction Method

Experiments

- testbed
  9 hosts, 10Gb NIC
  does not show scalability

- in-memory
  simulating the whole network from memory

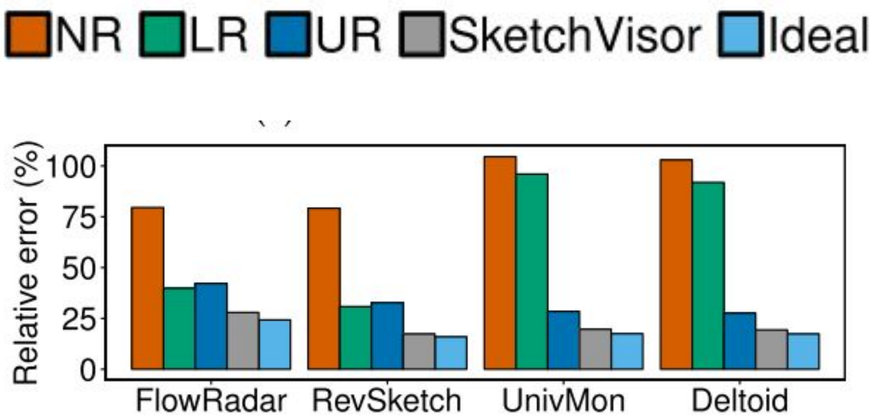
Results - Throughput



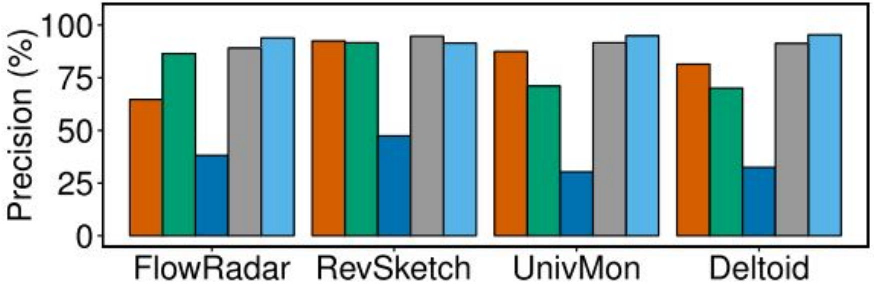
(b) In-memory tester result

and when using parallelization: 2 cores are sufficient to achieve above 40Gbps throughput (in-memory test)

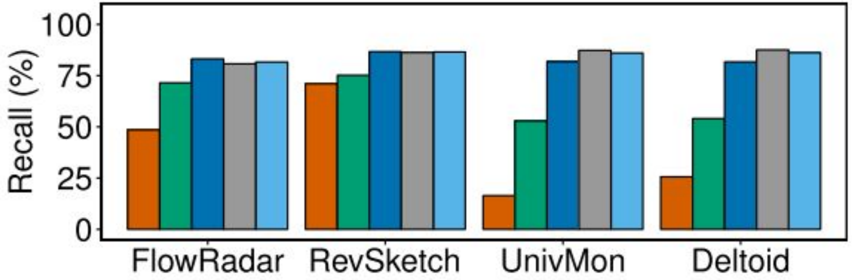
Results - Accuracy



(f) HC Relative error



(e) HC Precision



(d) HC Recall

Results - further

- accuracy increases with the number of hosts
- accuracy increases with fast path size (should increase linearly with number of flows)
- kick out in fast path algorithm very computationally expensive, but rather rare

Conclusion

- Sketches still are a very effective and efficient way of measuring network statistics.
- In comparison to similar methods, SketchVisor exceeds in every aspect (at least in the paper)
- Is easily applicable on existing sketch-solutions without needing specialized hardware.

- Potential problem: Reversibility: which flow contributed to which flow by how much? (not adressed in paper)

Conclusion on the paper

- Well structured: identify problem, pose solution, elaborate and demonstrate
- Detailed description of theory
- Very detailed description of experiments and results. This probably makes replication easy.

- No suggestions of further work

Thank You!

and now: discussion