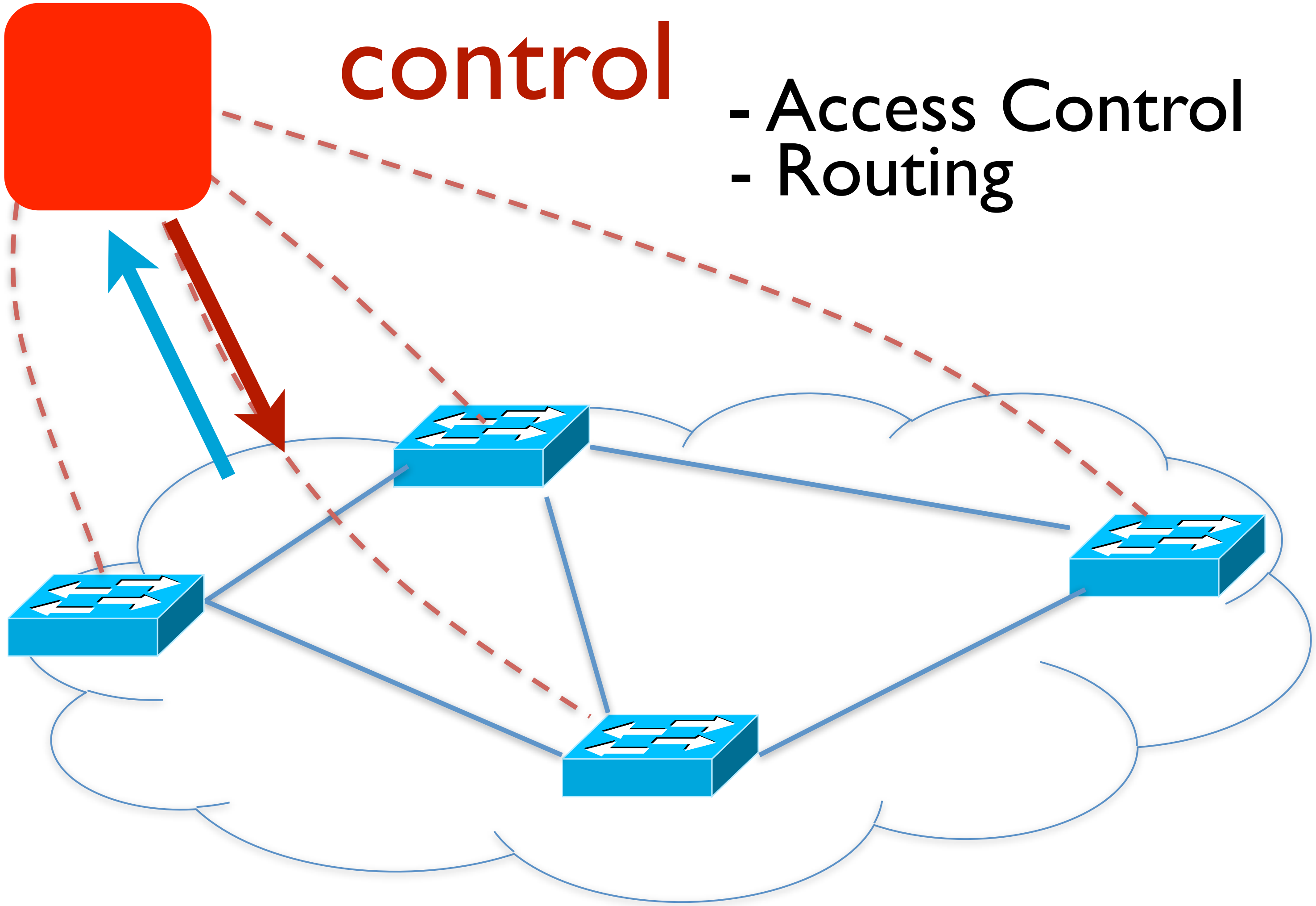
Software_Defined Traffic Measurement with OpenSketch

Lavanya Jose
Stanford University

Joint work with Minlan Yu and Rui Miao at USC

Management is Control + Measurement



control

- Access Control
- Routing

measure

- DDoS
- Flow Size Distribution

Questions we want to ask

1. Who’s sending a lot to 10.0.2.0/16? (Heavy Hitters)
2. How are flow sizes distributed?
3. Is someone doing a port scan?
4. Is someone being DDoS-ed?
5. Who’s getting traffic from blacklisted IPs?
6. How many people downloaded files from 10.0.2.1?

Switches are great at counting per flow bytes and packets

- NetFlow and sFlow sample packets
- NetFlow maintains per flow byte and packet counts
- Can find count of a particular flow, prefix or counts of heavy flows

Problem: NetFlow counts can't answer my questions

Is someone doing a port scan?

NetFlow samples packets from heavy flows. Missed packets from small "port scanners".

- Increase sampling rate --> inefficient

Streaming algorithms

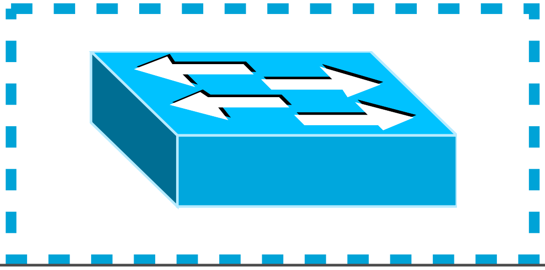
+ Process efficiently at line rate

+ Accurate answers

- But each answers a specific question

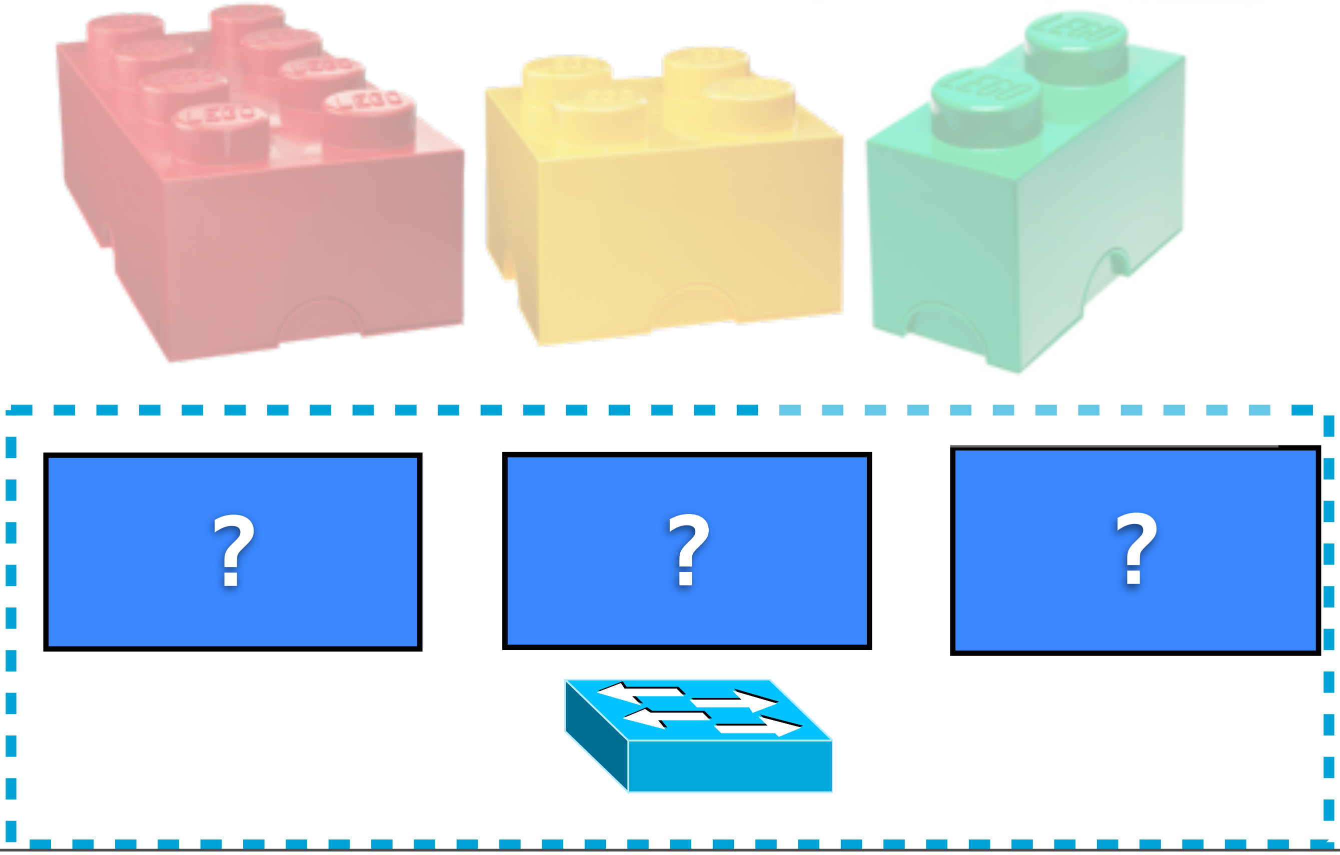
What measurement architecture can answer all my questions?

1. Who’s sending a lot to 10.0.2.0/16? (Heavy Hitters)
2. How are flow sizes distributed?
3. Is someone doing a port scan?
4. Is someone being DDoS-ed?
5. Who’s getting traffic from blacklisted IPs?
6. How many people downloaded files from 10.0.2.1?



SDN Model: Find Building Blocks

1. Who’s sending a lot to 10.0.2.0/16? (Heavy Hitters)
2. How are flow sizes distributed?
3. Is someone doing a port scan? ...

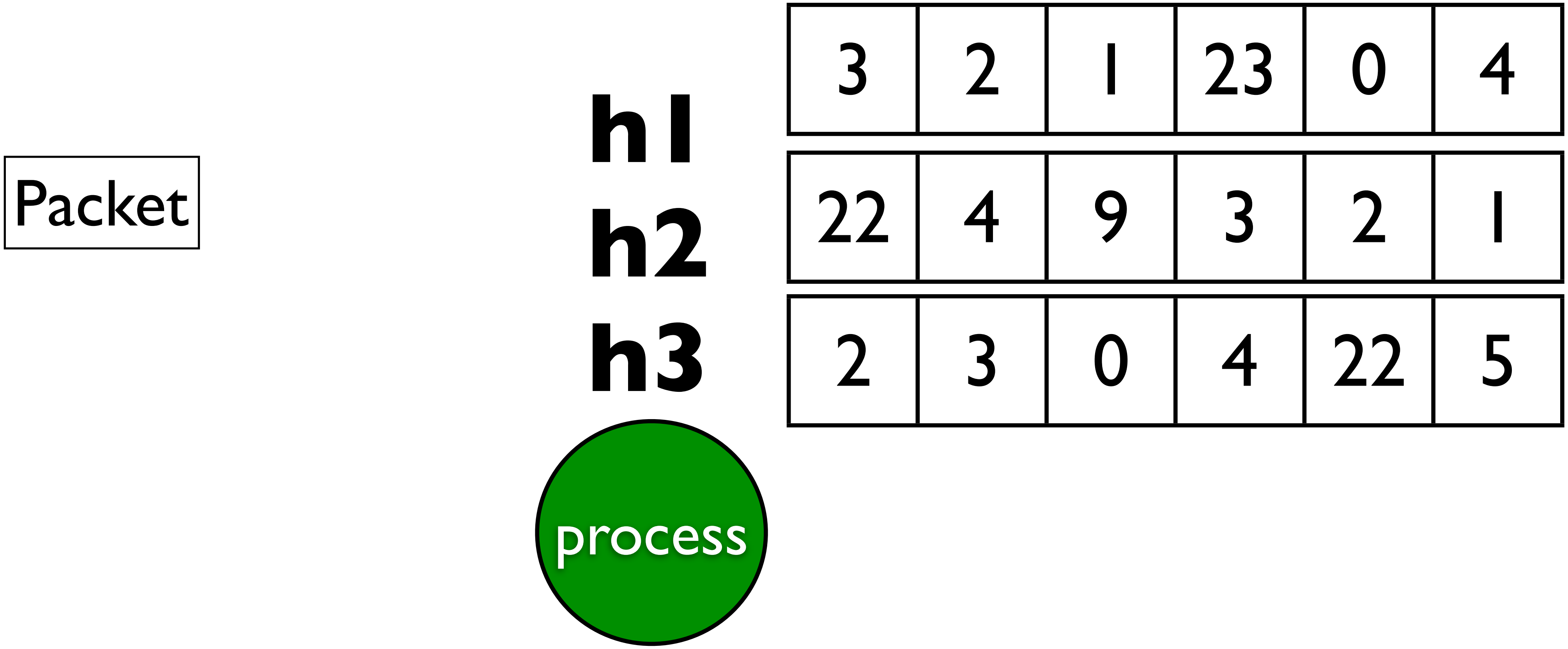


Sketches as building blocks

- Sketch
  - Data structure
  - Support approx. computing some function of data
  - Much smaller than actual data
  - Streaming, small per-item processing cost
  - Provable space-accuracy tradeoffs

Sketches as building blocks

e.g., Count Min sketch
to store counts of frequent source IP addresses



Packet

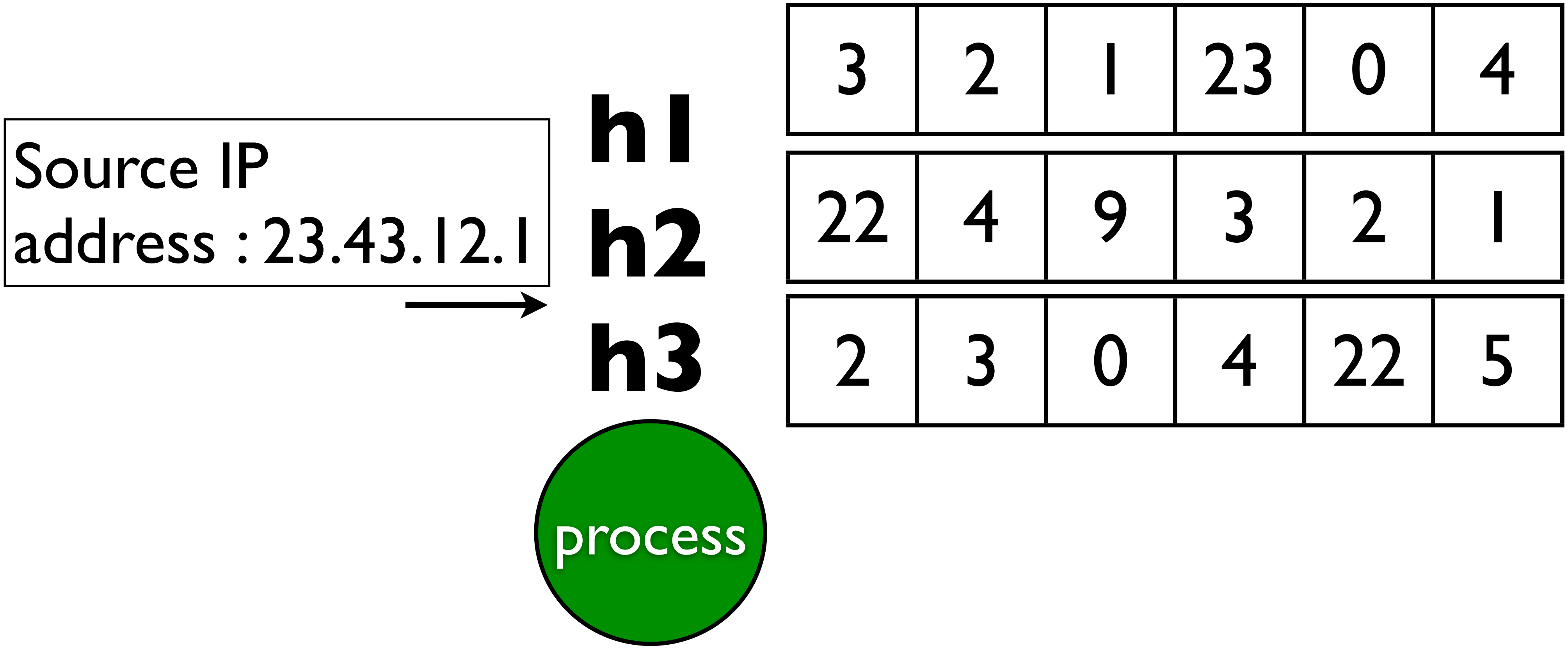
h1	3	2	1	23	0	4
h2	22	4	9	3	2	1
h3	2	3	0	4	22	5

process

(Cormode 2005)

Sketches as building blocks

e.g., Count Min sketch
to store counts of frequent source IP addresses



Source IP address : 23.43.12.1

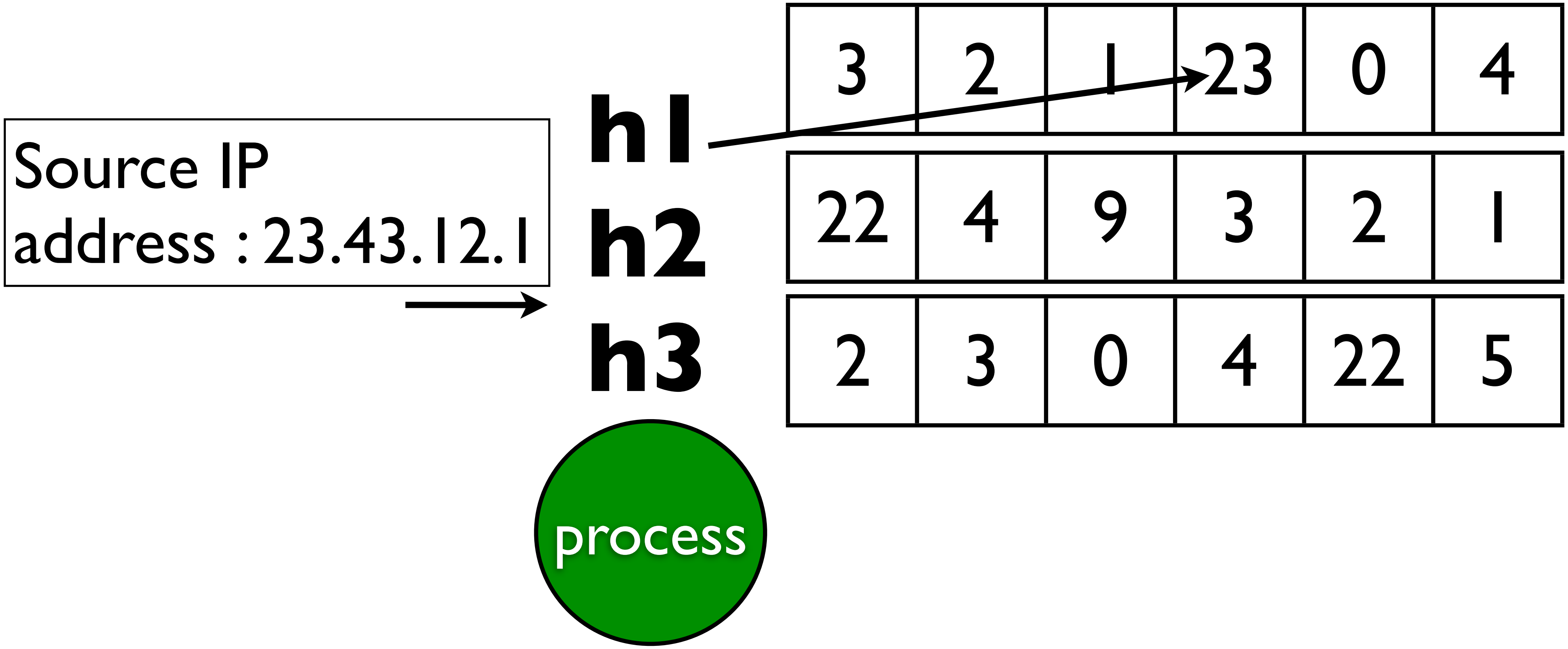
h1	3	2	1	23	0	4
h2	22	4	9	3	2	1
h3	2	3	0	4	22	5

(Cormode 2005)

Sketches as building blocks

e.g., Count Min sketch

to store counts of frequent source IP addresses



Source IP address : 23.43.12.1

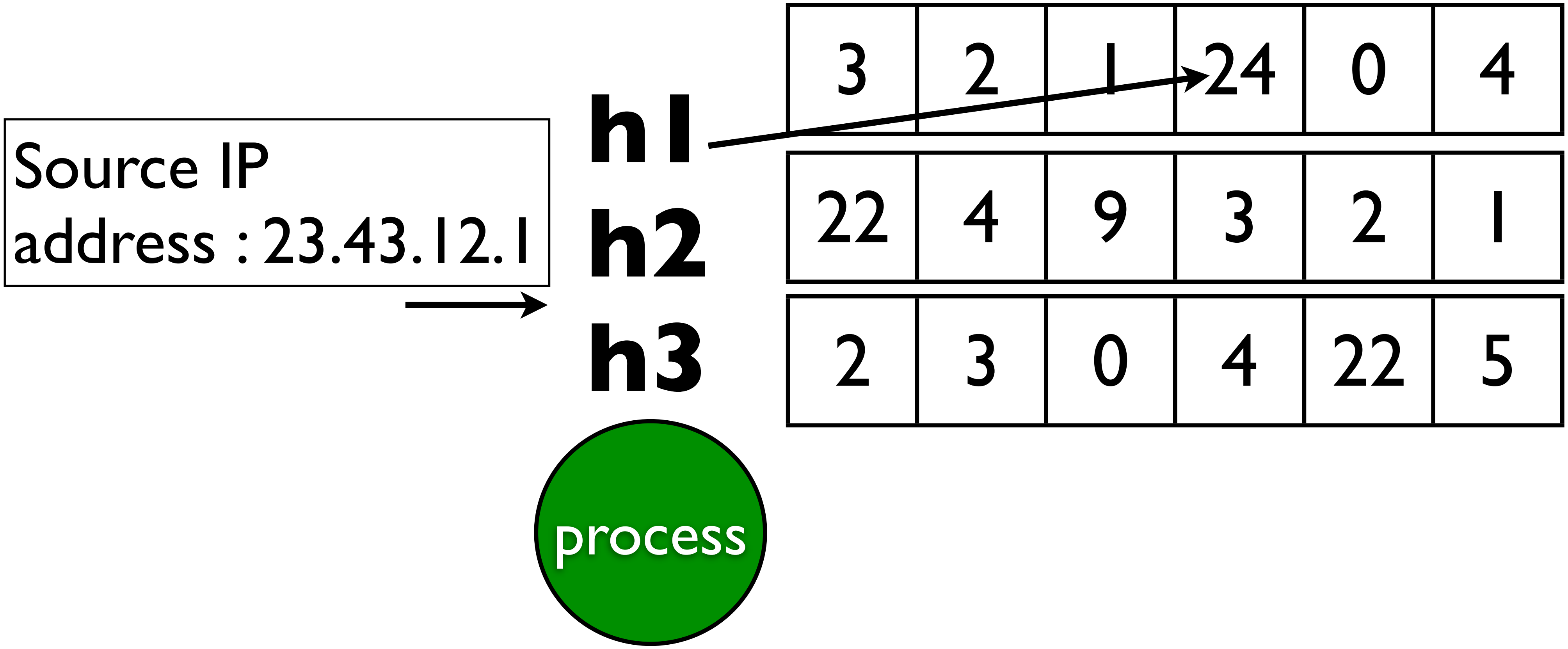
h1	3	2	1	23	0	4
h2	22	4	9	3	2	1
h3	2	3	0	4	22	5

(Cormode 2005)

Sketches as building blocks

e.g., Count Min sketch

to store counts of frequent source IP addresses



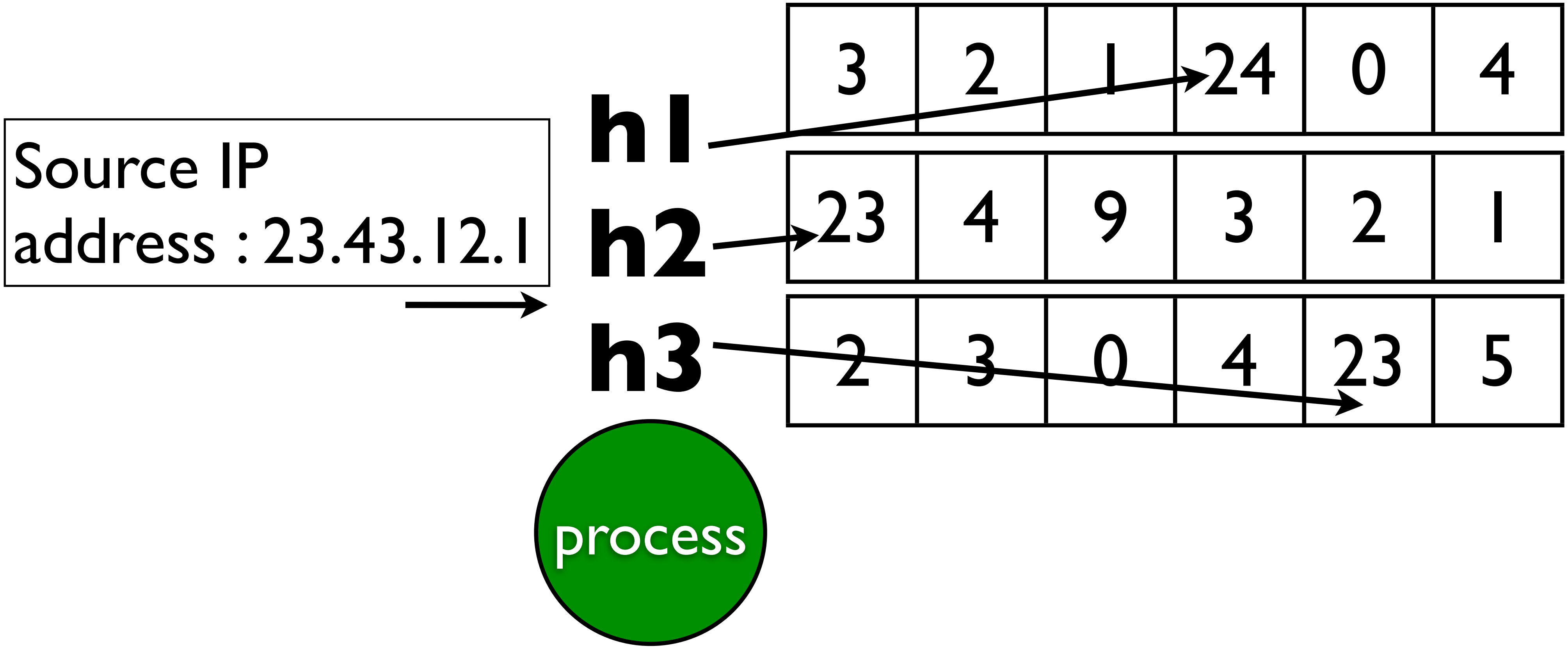
Source IP address : 23.43.12.1

h1	3	2	1	24	0	4
h2	22	4	9	3	2	1
h3	2	3	0	4	22	5

(Cormode 2005)

Sketches as building blocks

e.g., Count Min sketch
to store counts of frequent source IP addresses



Source IP address : 23.43.12.1

3	2	1	24	0	4
23	4	9	3	2	1
2	3	0	4	23	5

(Cormode 2005)

Sketches as building blocks

e.g., Count Min sketch
to store counts of frequent source IP addresses

h1	3	2	1	24	0	4
h2	23	4	9	3	2	1
h3	2	3	0	4	23	5

query

(Cormode 2005)

Sketches as building blocks

e.g., Count Min sketch
to store counts of frequent source IP addresses

# packets from 23.43.12.1?

query

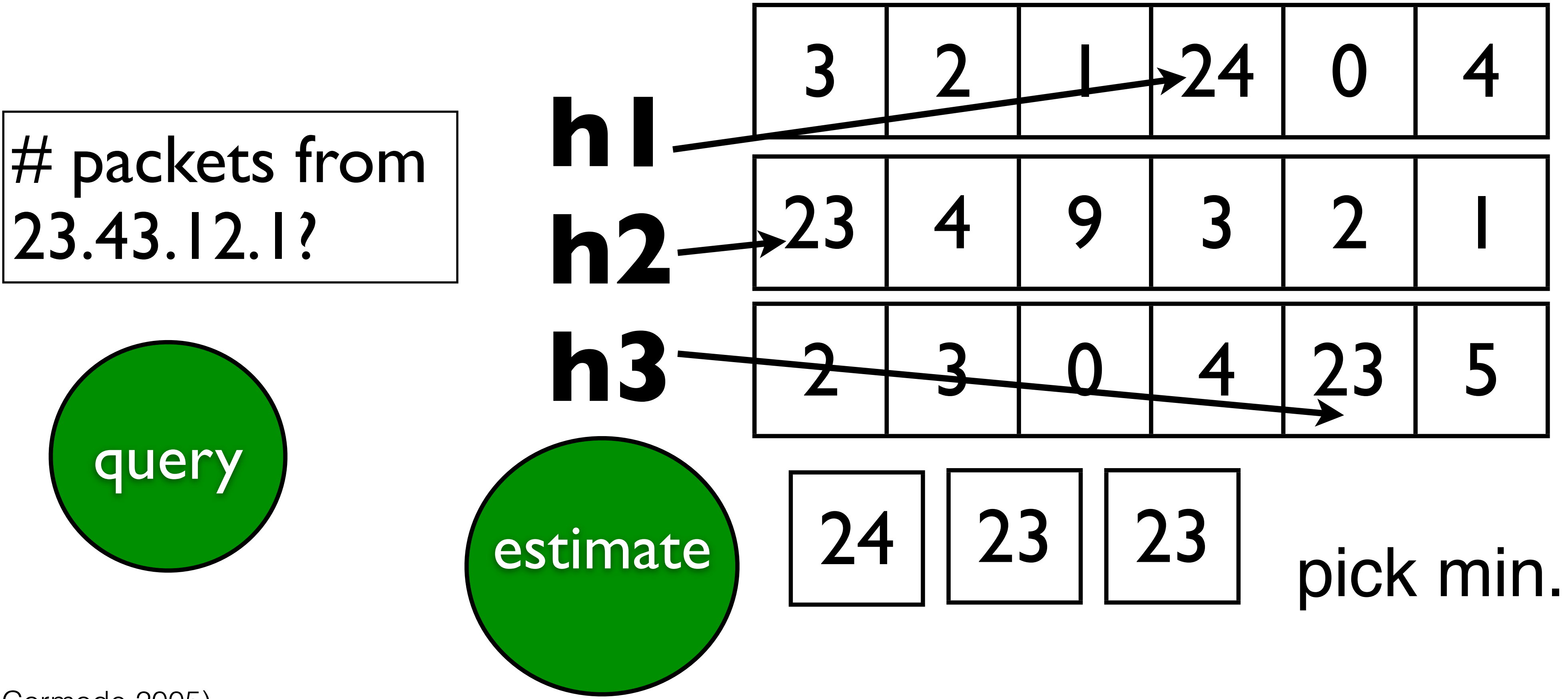
h1	3	2	1	24	0	4
h2	23	4	9	3	2	1
h3	2	3	0	4	23	5

(Cormode 2005)

Sketches as building blocks

e.g., Count Min sketch

to store counts of frequent source IP addresses



# packets from 23.43.12.1?

h1	3	2	1	24	0	4
h2	23	4	9	3	2	1
h3	2	3	0	4	23	5

24	23	23

pick min.

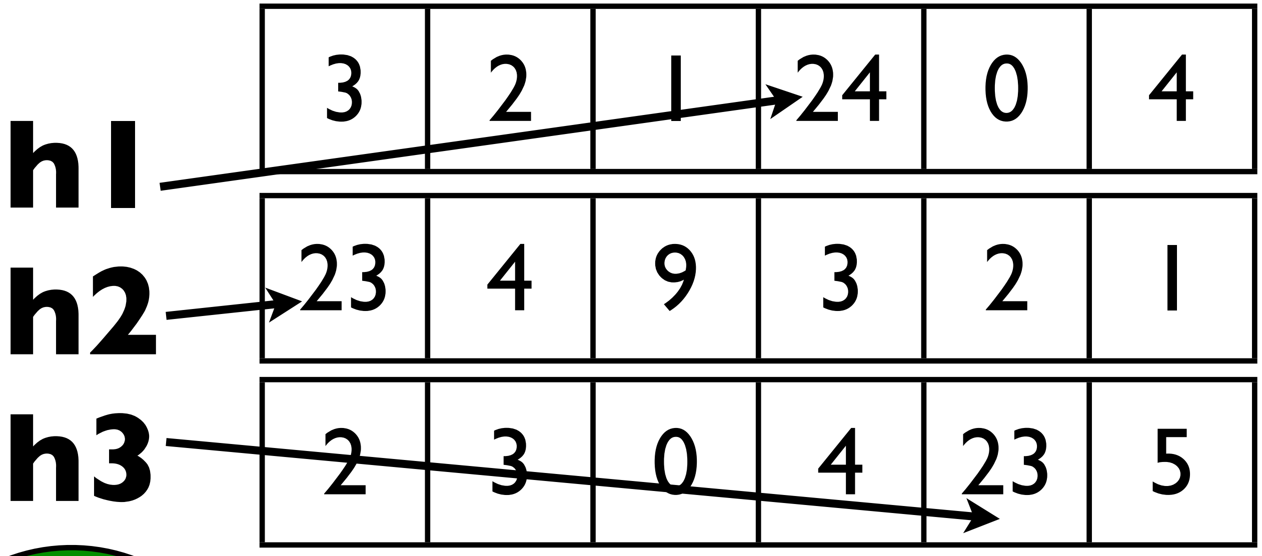
(Cormode 2005)

Sketches as building blocks

e.g., Count Min sketch

to store counts of frequent source IP addresses

# packets from 23.43.12.1?



h1	3	2	1	24	0	4
h2	23	4	9	3	2	1
h3	2	3	0	4	23	5

query

estimate

23

pick min.

(Cormode 2005)

Sketches as building blocks

e.g., Count Min sketch
to store counts of frequent source IP addresses

within ϵ total packets with high probability

$$\epsilon = \frac{e}{\text{no. of counters}}$$

$$\Pr\{\text{error} > \epsilon\,\text{total packets}\} < e^{-\text{ no. of hash functions}}$$

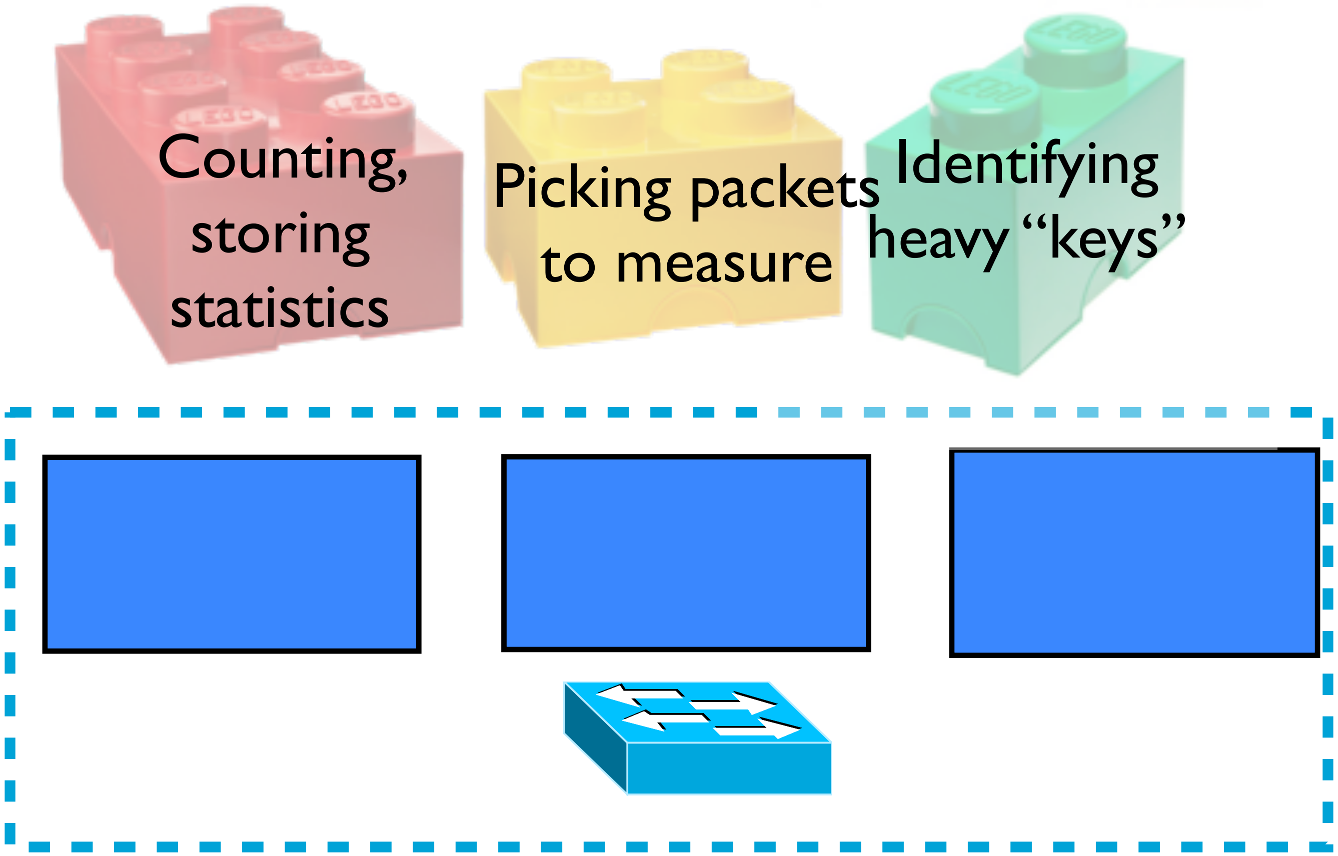
+ Provable space-accuracy tradeoffs

$\boxed{23}$

estimate
pick min.

(Cormode 2005)

Sketches as building blocks



(Reversible Sketch:
Schweller 2004)

…answer many questions

1. Who's sending a lot to 10.0.2.0/16? (Heavy Hitters)
2. How are flow sizes distributed?
3. Is someone doing a port scan?
4. Is someone being DDoS-ed?
5. Who's getting traffic from blacklisted IPs?
6. How many people downloaded files from 10.0.2.1?

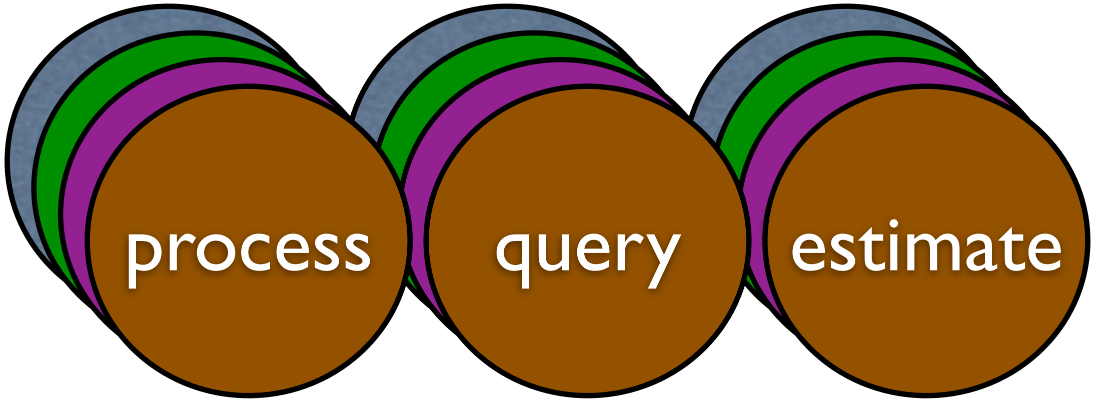
(Reversible Sketch: Schweller 2004)

Counting, storing statistics

Identifying heavy "keys"

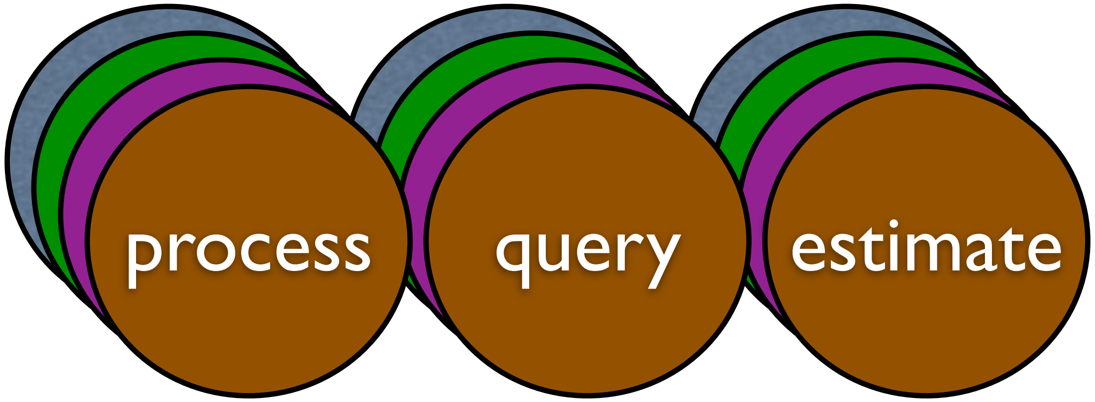
But each sketch estimates only one function

- frequency count
- cardinality
- set membership
- heavy "keys"



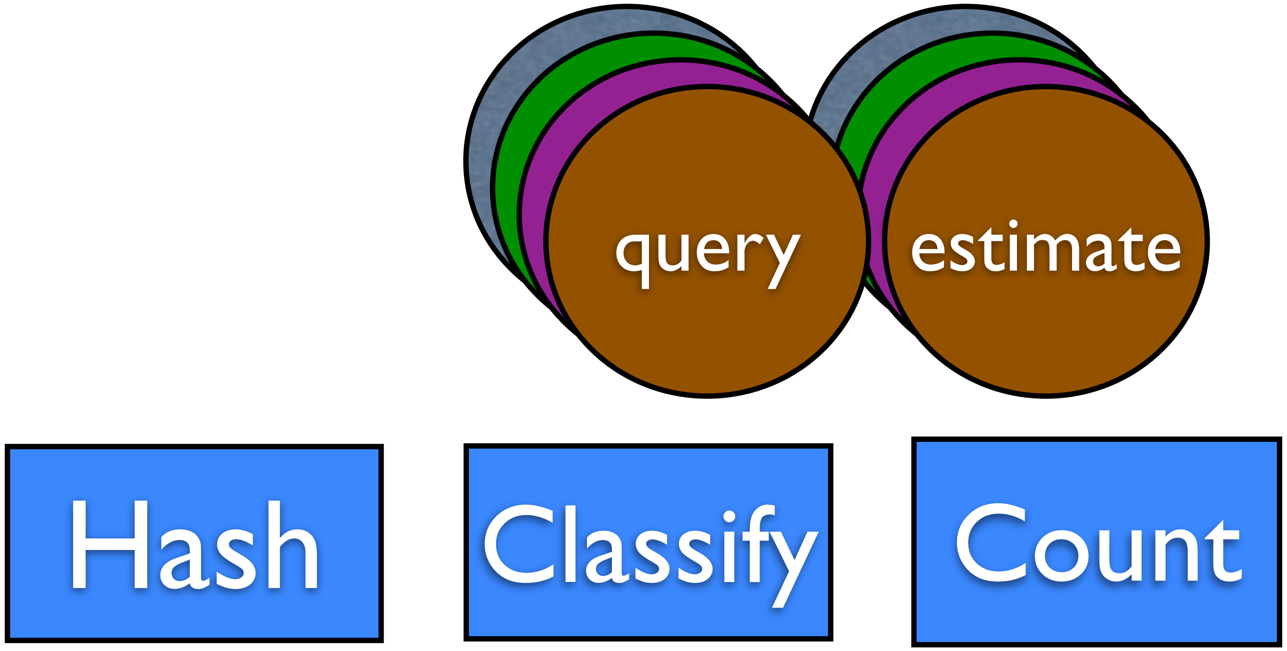
3_stage pipeline

- frequency count
- cardinality
- set membership
- heavy “keys”



3_stage pipeline

- frequency count
- cardinality
- set membership
- heavy “keys”



query

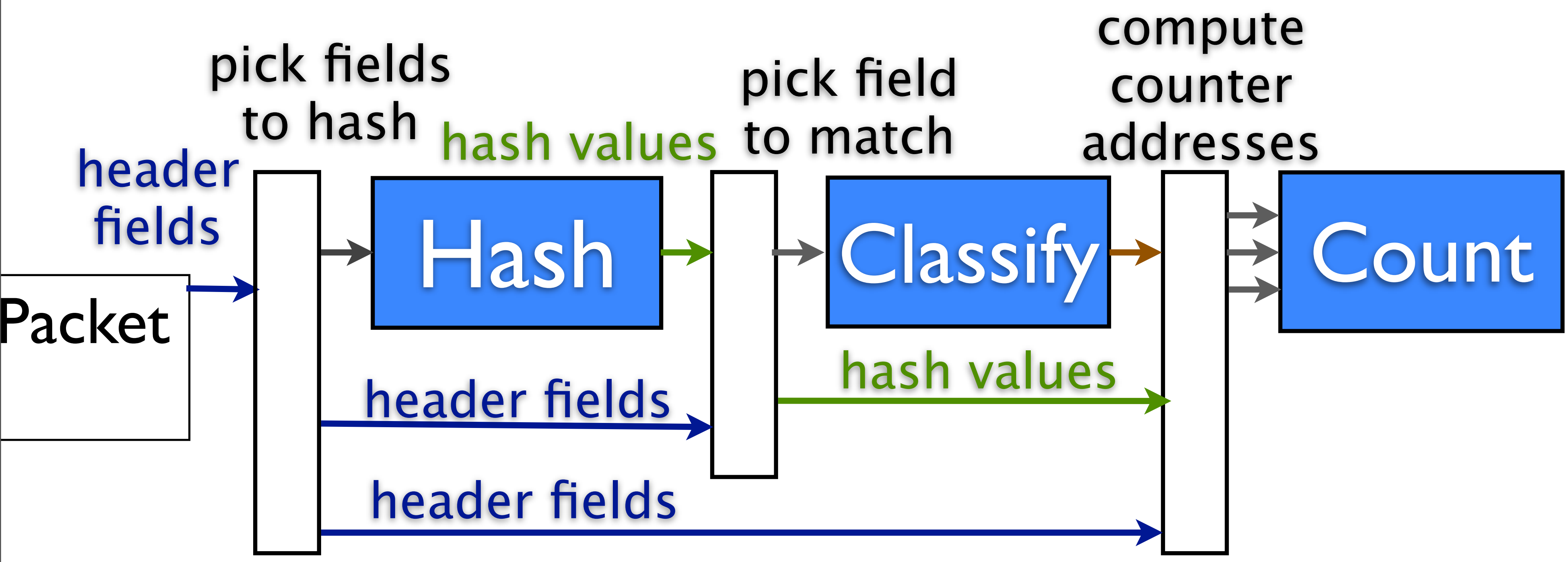
estimate

Hash

Classify

Count

3_stage pipeline



pick fields to hash

header fields

Packet

Hash

hash values

header fields

header fields

pick field to match

Classify

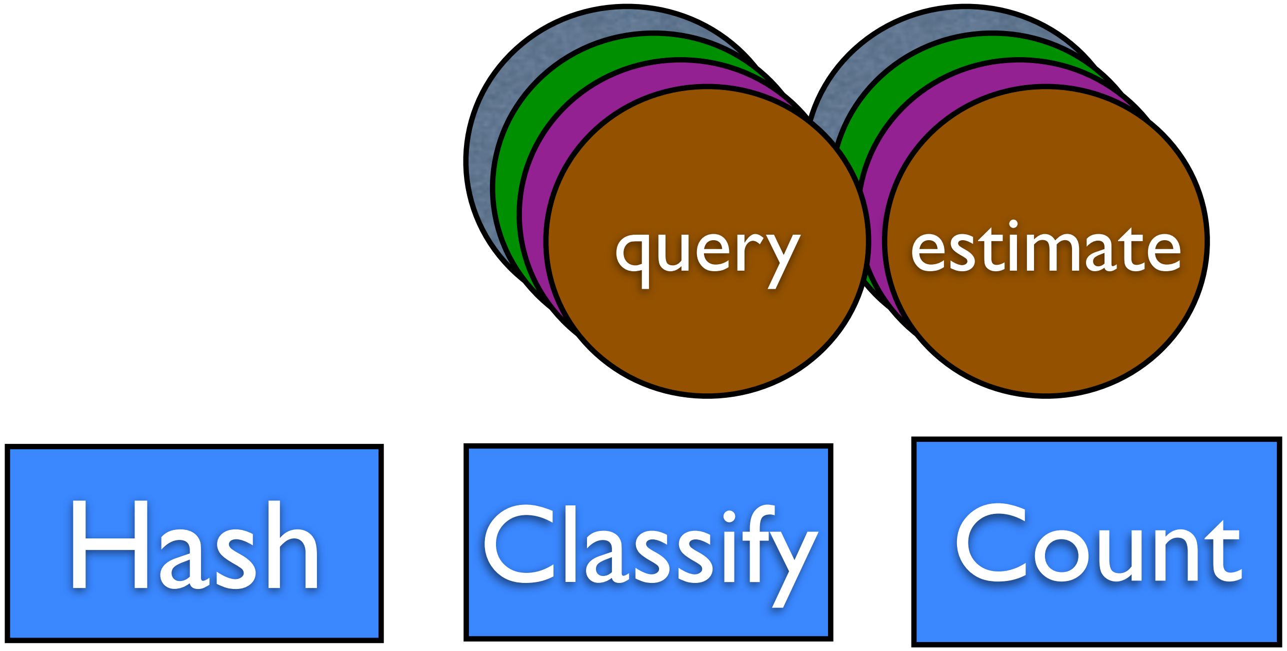
hash values

compute counter addresses

Count

3_stage pipeline

- frequency count
- cardinality
- set membership
- heavy “keys”


query

estimate

Hash

Classify

Count

3_stage pipeline

Identifying heavy “keys”

Counting, storing statistics

Picking packets to measure

Hash

Classify

Count

3–stage pipeline

1. Who’s sending a lot to 10.0.2.0/16? (Heavy Hitters)
2. How are flow sizes distributed?
3. Is someone doing a port scan?

Identifying heavy “keys”

Counting, storing statistics

Picking packets to measure

Hash

Classify

Count

OpenSketch Measurement Framework

Controller

1. Who’s sending a lot to 10.0.2.0/16? (Heavy Hitters)
2. How are flow sizes distributed?
3. Is ..

Data Plane

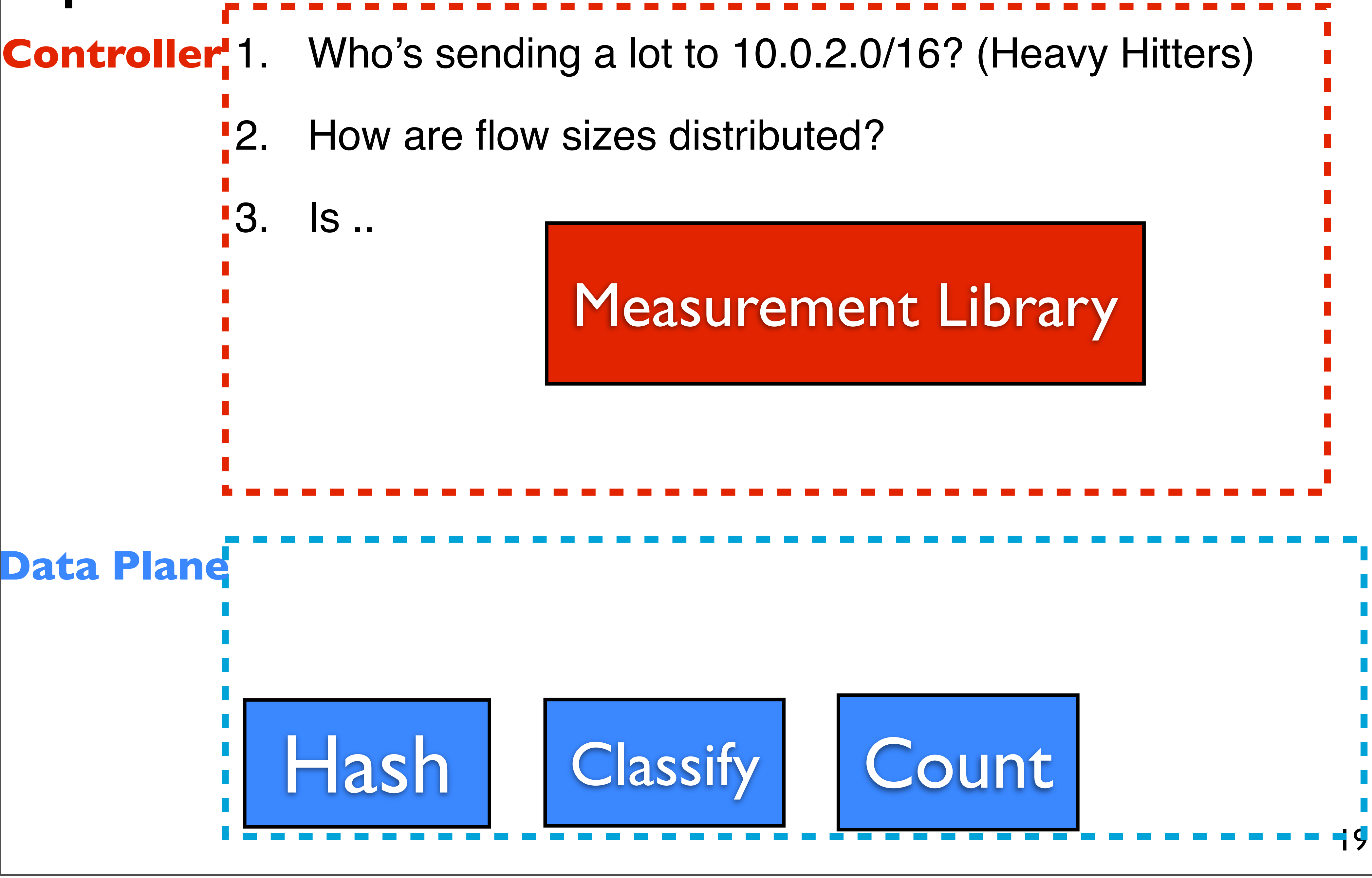
Hash

Classify

Count

19

OpenSketch Measurement Framework



Controller

1. Who’s sending a lot to 10.0.2.0/16? (Heavy Hitters)
2. How are flow sizes distributed?
3. Is ..

Measurement Library

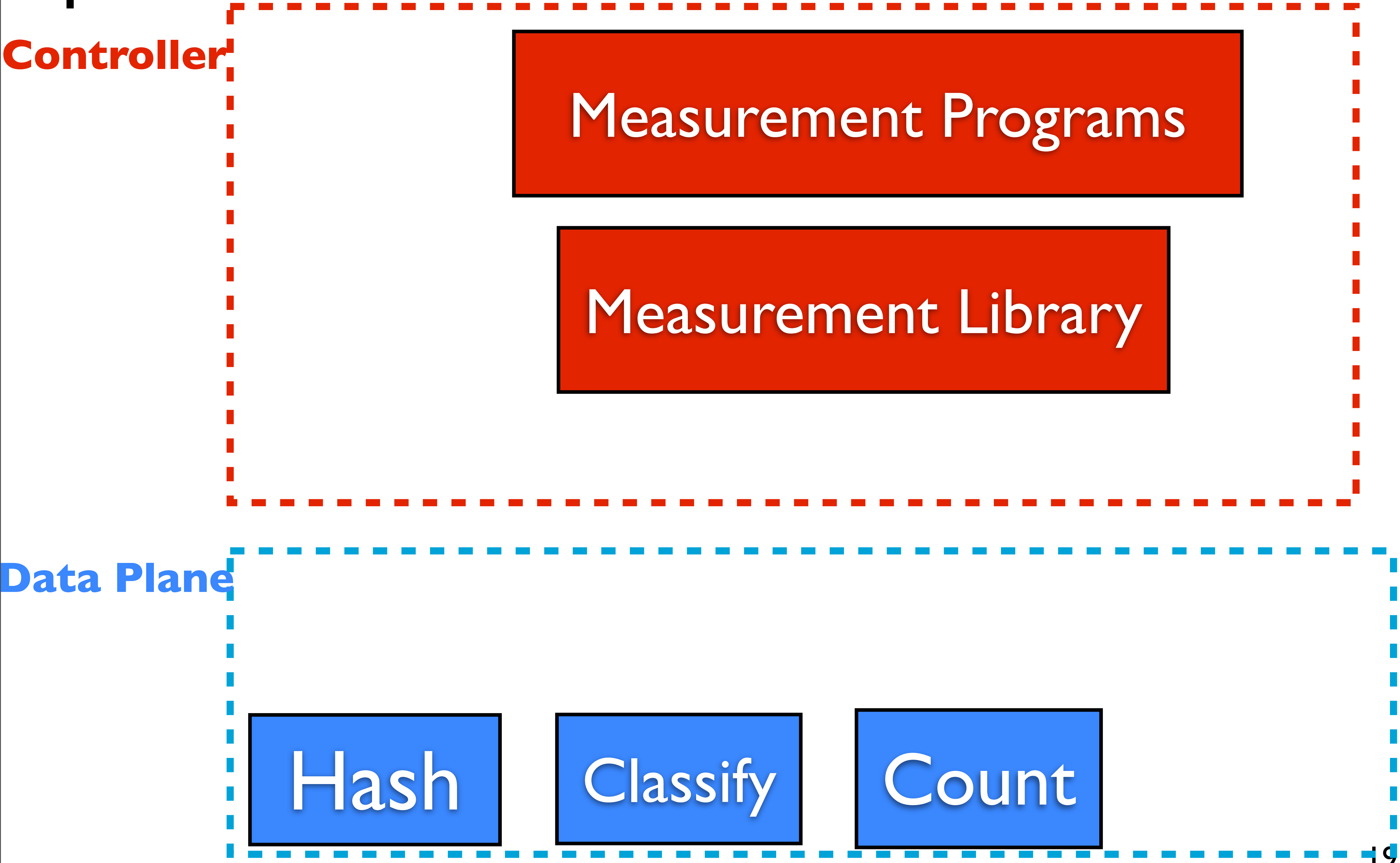
Data Plane

Hash

Classify

Count

OpenSketch Measurement Framework



Controller

Measurement Programs

Measurement Library

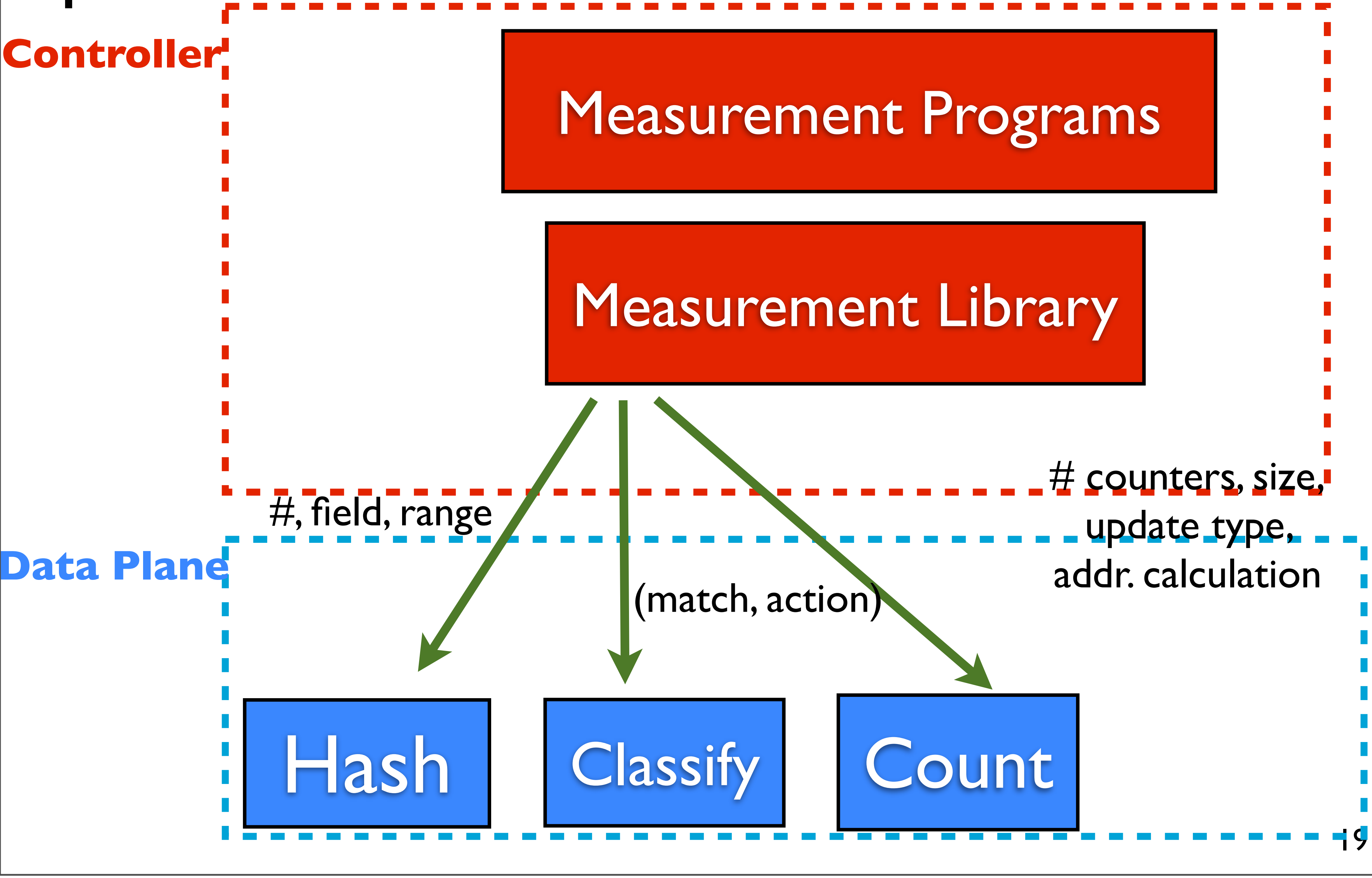
Data Plane

Hash

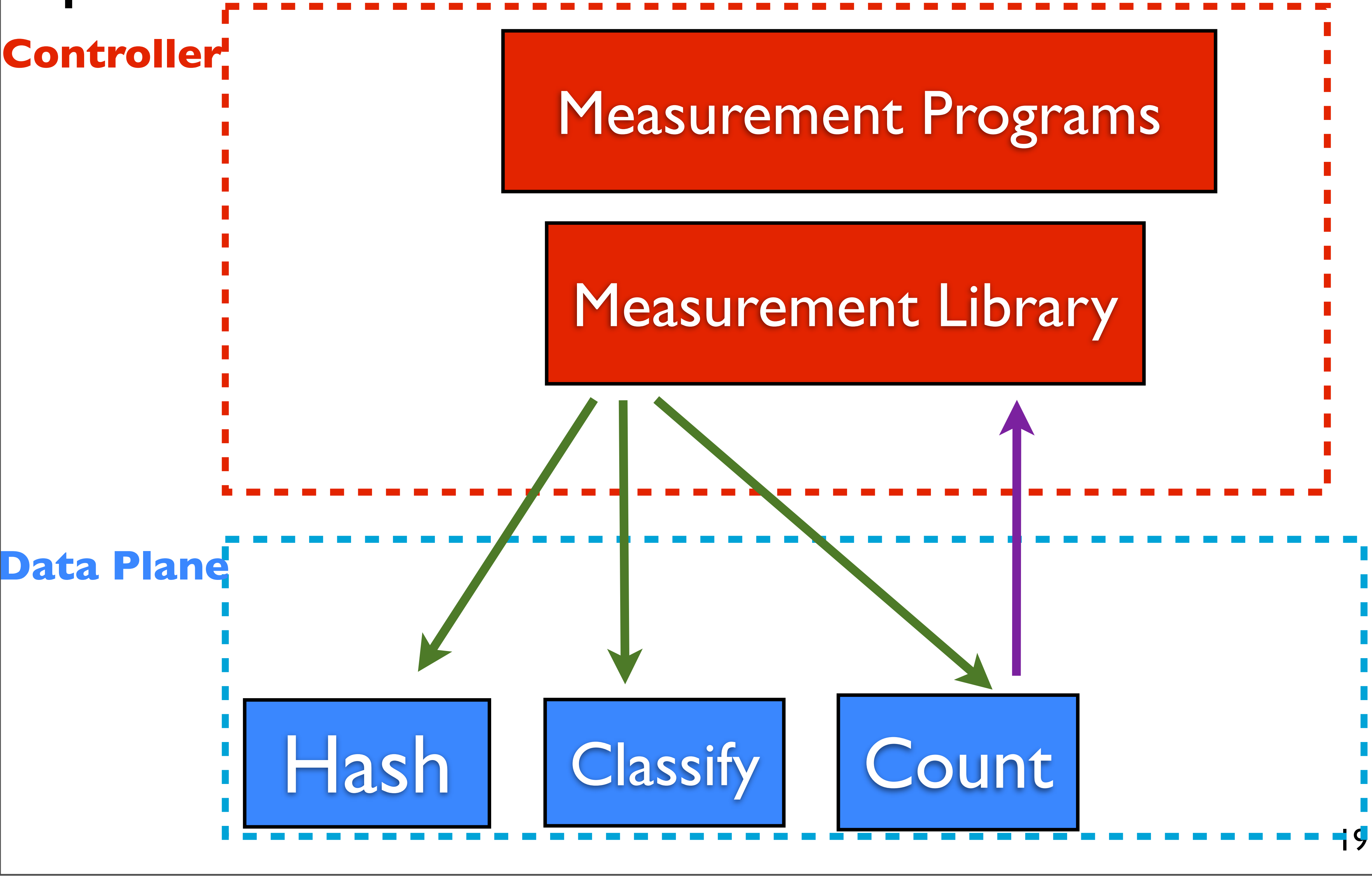
Classify

Count

OpenSketch Measurement Framework



OpenSketch Measurement Framework



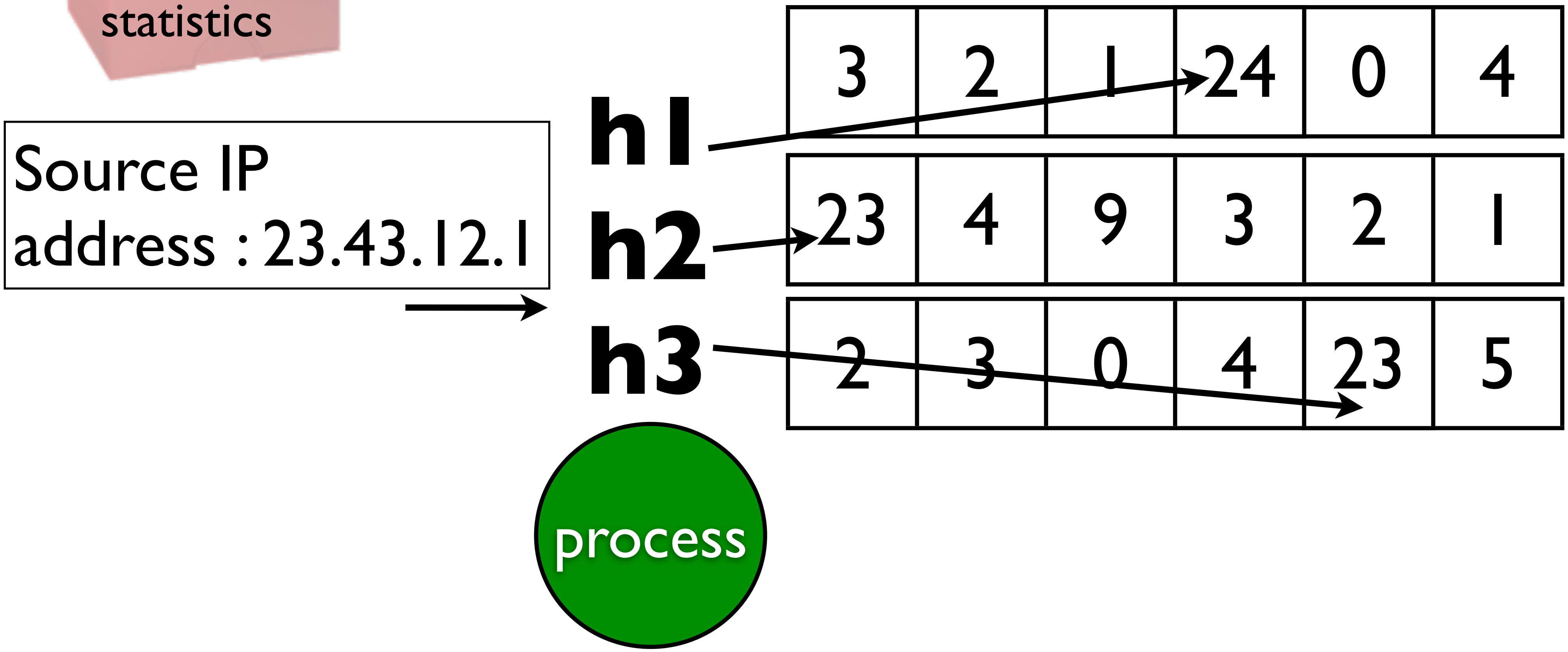
Details

- Implementing sketches with the Pipeline
- Configuring the Pipeline
- Evaluation and NetFPGA prototype

Count Min Sketch with the Pipeline

Counting, storing statistics

to store counts of frequent source IP addresses



Source IP address : 23.43.12.1

h1	3	2	1	24	0	4
h2	23	4	9	3	2	1
h3	2	3	0	4	23	5

process

Hash

Count

Bitmap Sketch with the Pipeline

Counting, storing statistics

to store number of different destination port numbers

Packet

h

0	0	1	0	0	1	0	0	1	0

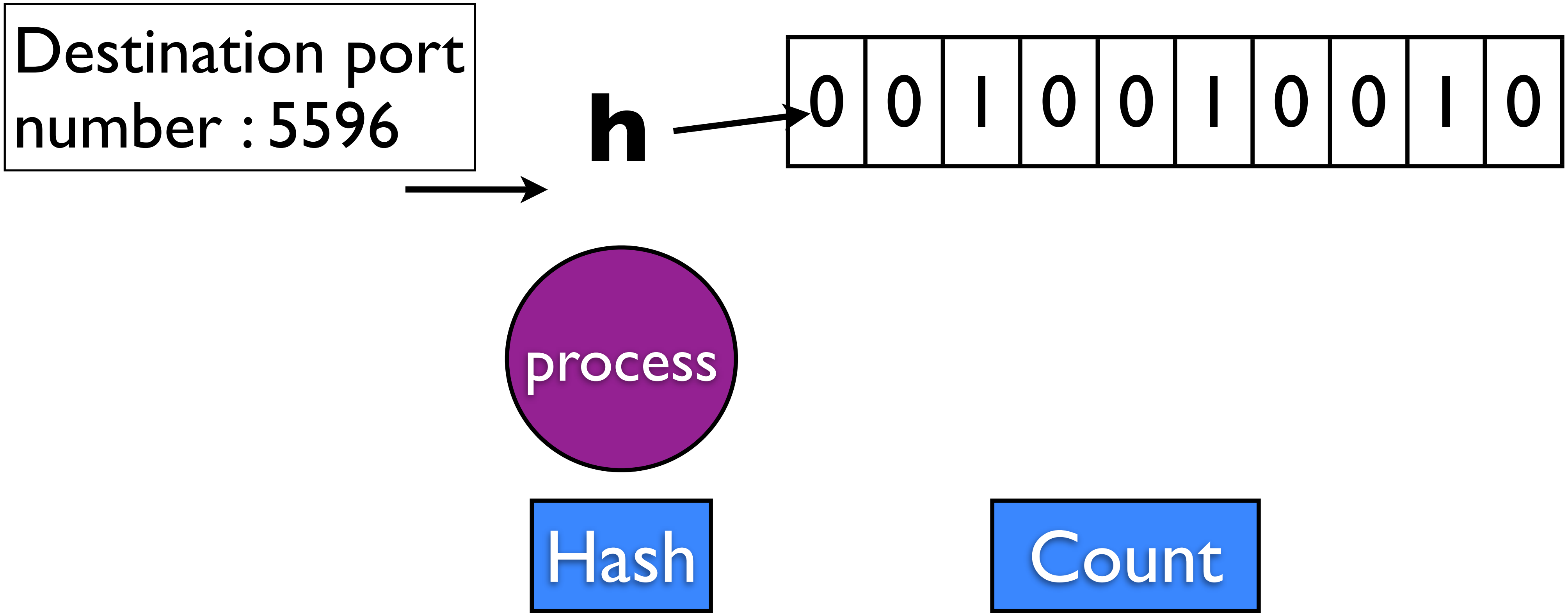
process

Hash

Count

Bitmap Sketch with the Pipeline

to store number of different destination port numbers



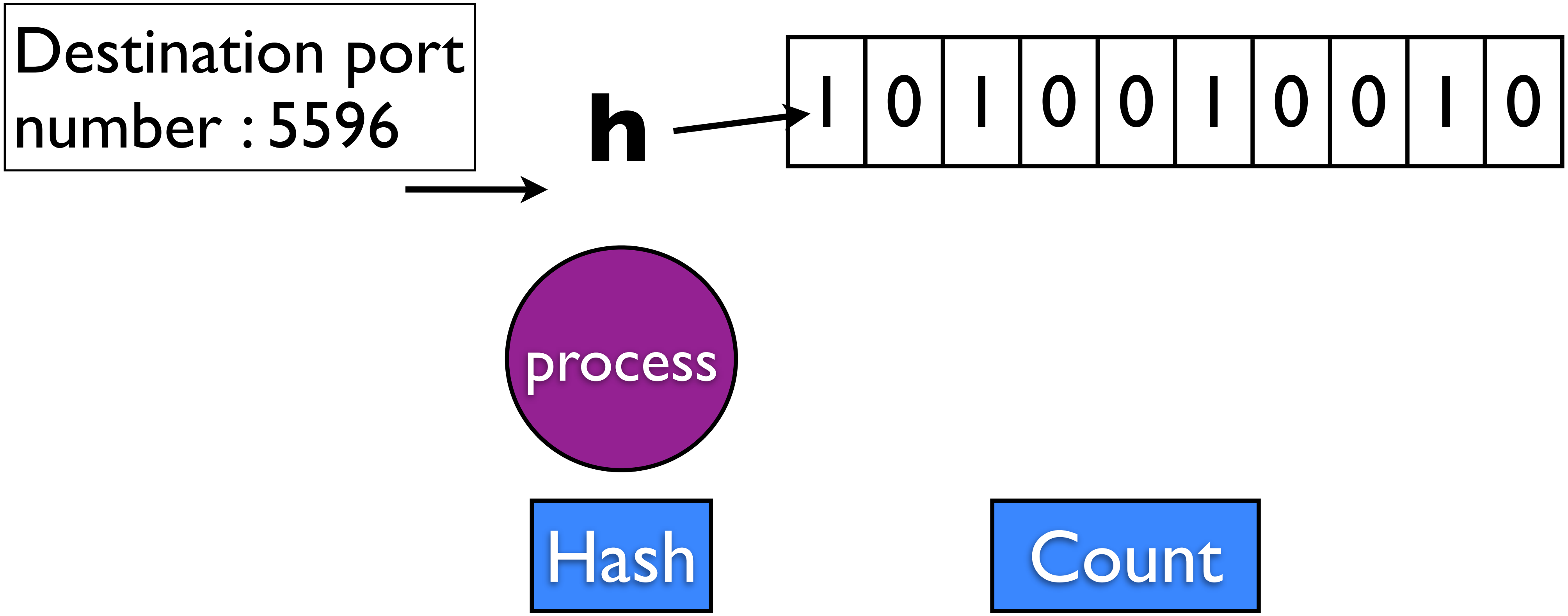
Destination port number : 5596

0	0	1	0	0	1	0	0	1	0

Bitmap Sketch with the Pipeline

Counting, storing statistics

to store number of different destination port numbers



Destination port number : 5596

1	0	1	0	0	1	0	0	1	0

Bitmap Sketch with the Pipeline

to store number of different destination port numbers

# different destination port numbers?

1	0	1	0	0	1	0	0	1	0

query

(Whang 1990)

Bitmap Sketch with the Pipeline

Counting, storing statistics

to store number of different destination port numbers

# different destination port numbers?

1	0	1	0	0	1	0	0	1	0

query

estimate

6/10

estimate
N = -10 ln(6/10)= 5

Six counters out of ten are 0.

(Whang 1990)

Other Sketches

- K-ary Sketch for heavy changes
- Bloom Filter Sketch to check set membership
- PCSA sketch to count distinct values

(Schweller 2004; Goel 2010; Flajolet 1985)

Efficient implementation of 3_stage pipeline

Hash

Classify

Count

hash
in parallel

TCAM rules

cheap fast memory
MBs of SRAM

Similar functions, diverse configurations

Hash

?? hash functions

Classify

?? TCAM entries for classify rules

Count

?? MBs of SRAM for tables of counters

Similar functions, diverse configurations

Hash

4-8 simple hash functions per question

- Count Min: 3
- Bloom Filters: 7-8
- Fixed size reversible sketch: 5
- Can share hash functions

Similar functions, diverse configurations

Classify

30-40 TCAM entries per question maximum

- Match a prefix/ value: 1 rule

- Match a set of values: Bloom Filters

Similar functions, diverse configurations

Count

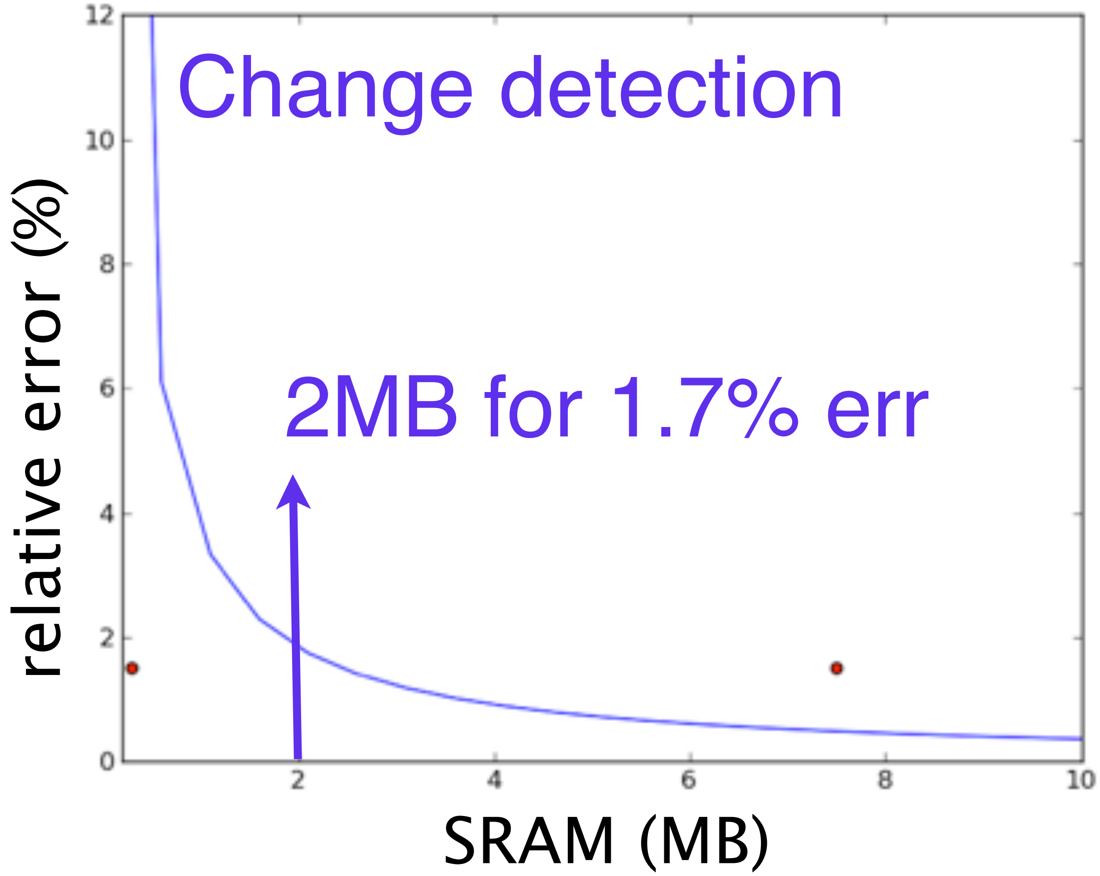
From simulation and worst case bounds for different tasks

up to 8MB SRAM

Similar functions, diverse configurations

Count

up to 8MB SRAM



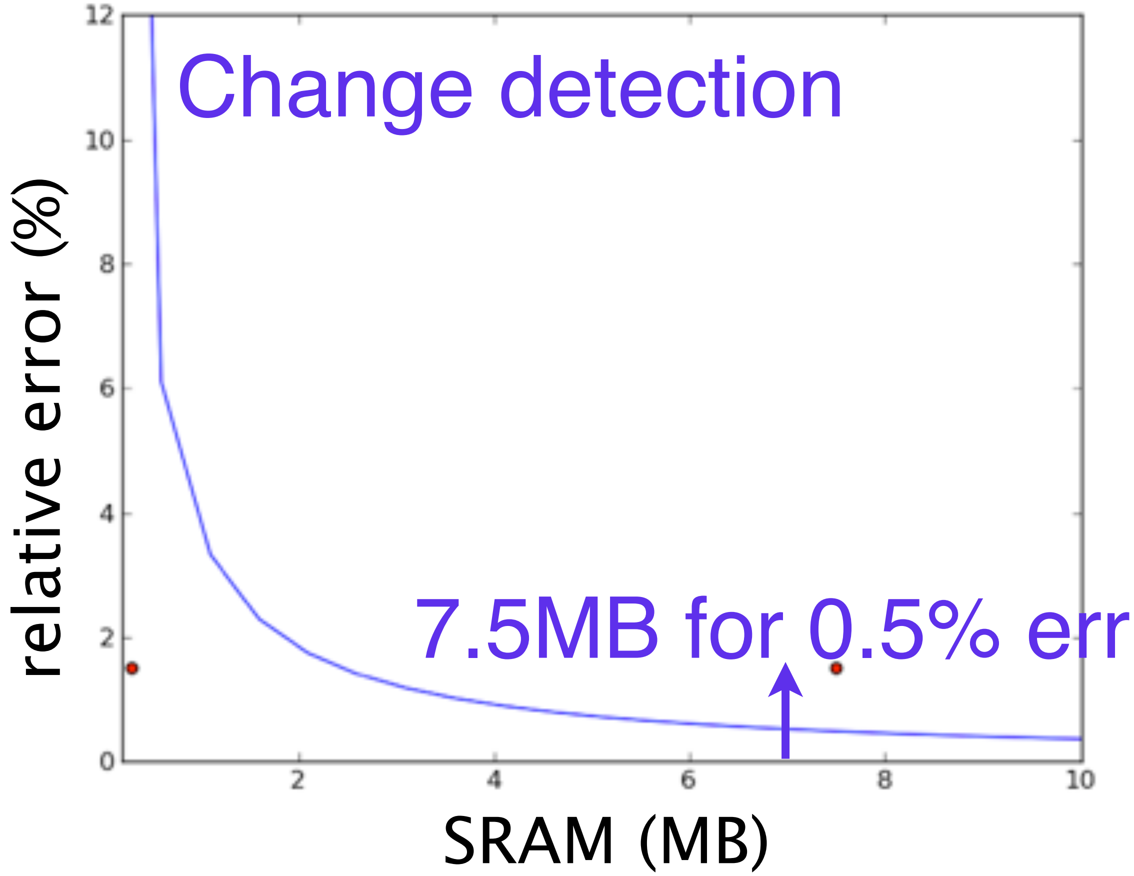
Change detection

2MB for 1.7% err

Similar functions, diverse configurations

Count

up to 8MB SRAM



Change detection

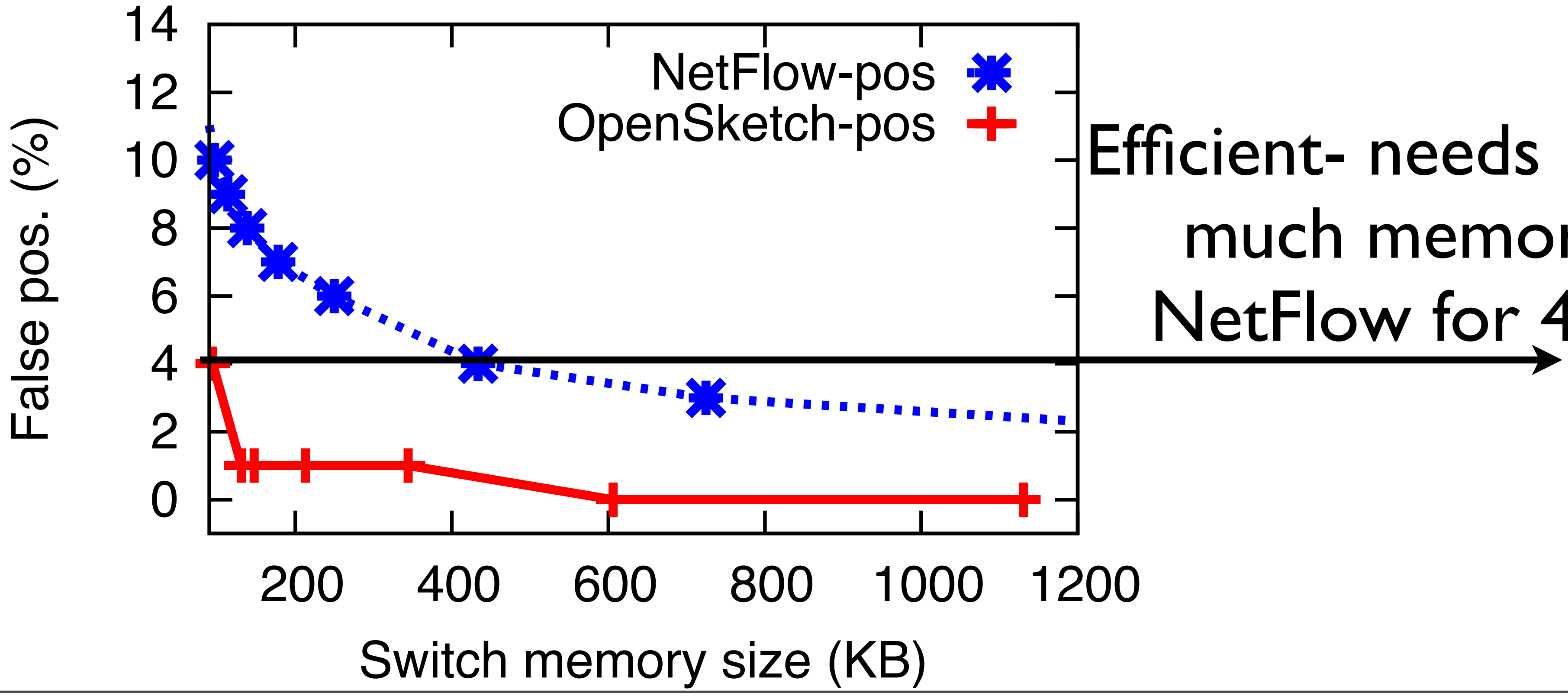
7.5MB for 0.5% err

Measurement tasks

1. Who’s sending a lot to 10.0.2.0/16? (Heavy Hitters)
2. How are flow sizes distributed?
3. Is someone doing a port scan?
4. Is someone being DDoS-ed?
5. Who’s getting traffic from blacklisted IPs?
6. How many people downloaded files from 10.0.2.1?

(Heavy Hitters: Cormode 2005; Flow Size Distribution: Kumar 2004;
Change detection: Schweller 2004; DDoS detection: Venkataraman 2005)

More efficient than NetFlow (Heavy Hitters)

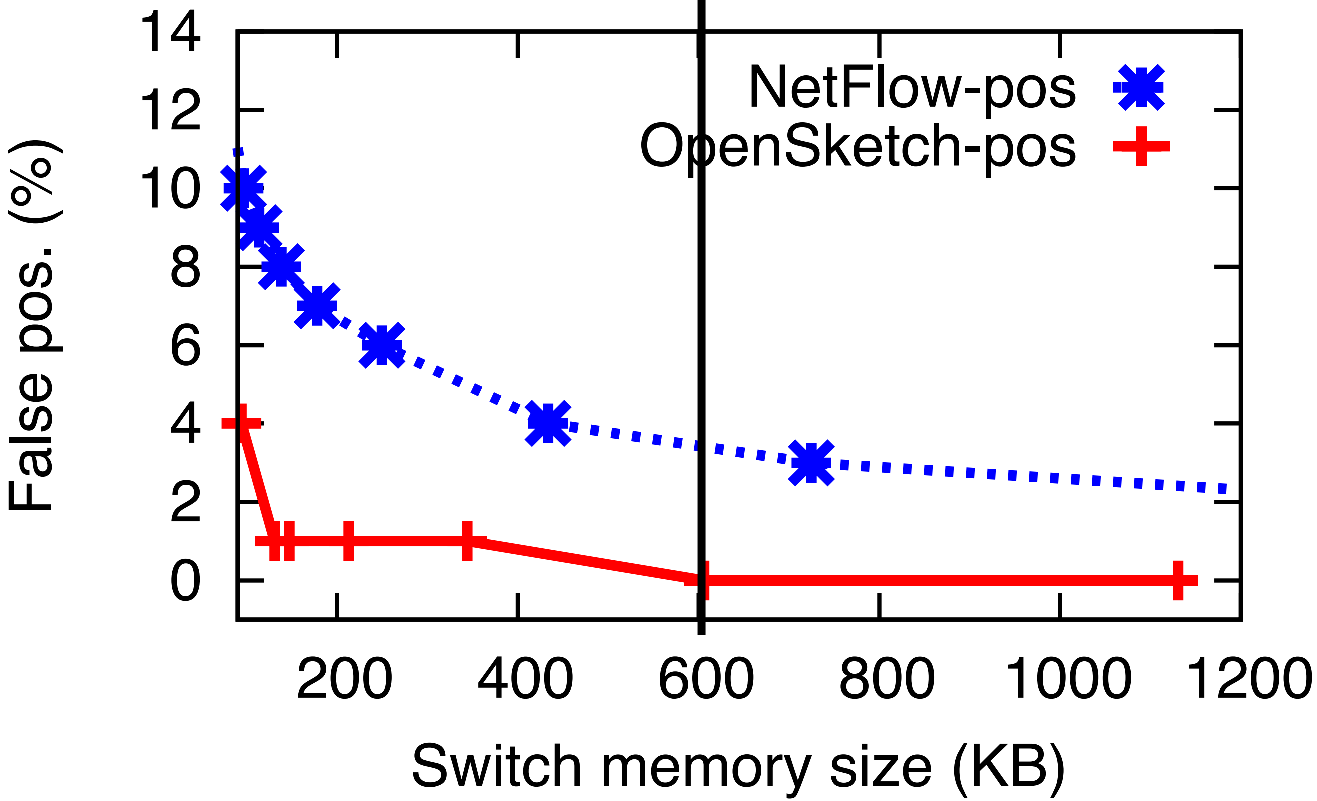


Efficient- needs 1/4th as much memory as NetFlow for 4% f.p.

More efficient than NetFlow

(Heavy Hitters)

Accurate- with 600KB, OpenSketch has less than 0.05% f.p. NetFlow has around 3%



OpenSketch NetFPGA Prototype

- 3-stage meas. pipeline parallel to forwarding
- Full throughput 1Gbps @ 4 ports
- Measurement pipeline takes fewer cycles than forwarding

Conclusion

- Current switches good for flow statistics
- But they don’t answer basic measurement questions
- Like identify heavy hitters, detect DDoS attacks, port scans, traffic from blacklisted IP address etc.

Takeaway

- Hash, classify and count pipeline in the Data Plane
- And sketch based building blocks in the Control Plane
- Make measurement in switches efficient and easy