北京大學
PEKING UNIVERSITY

Elastic Sketch: Adaptive and Fast Network-wide Measurements

Tong Yang, Jie Jiang, Peng Liu	Peking University, China
Qun Huang,	ICT, CAS, China
Junzhi Gong, Yang Zhou,	Peking University, China
Rui Miao,	Alibaba Group, China
Xiaoming Li,	Peking University, China
Steve Uhlig.	Queen Mary University of London, UK

Tong Yang, Peking University
yangtongemail@gmail.com
http://net.pku.edu.cn/~yangtong

Outline

01

PART ONE

Background

01 / Background

Measurement is important

Network measurement provides indispensable information for network applications.

Best solution: Sketch

1) Memory efficient
2) Constant speed
3) High accuracy

01 / Background

Most of existing solutions focus on:

A good trade-off among
1) memory usage
2) speed
3) accuracy

Recent work: UnivMon [SIGCOMM 16]

the above 3 dimensions plus
4) generality

01 / Background

In addition to the above 4 dimensions , our paper

5) adaptive to traffic characteristics
6) cross platform

Measurements are especially important when network is undergoing problems, such as

1) network congestion
2) scans
3) DDoS attack

In these cases, traffic characteristics vary a lot

01 / Background

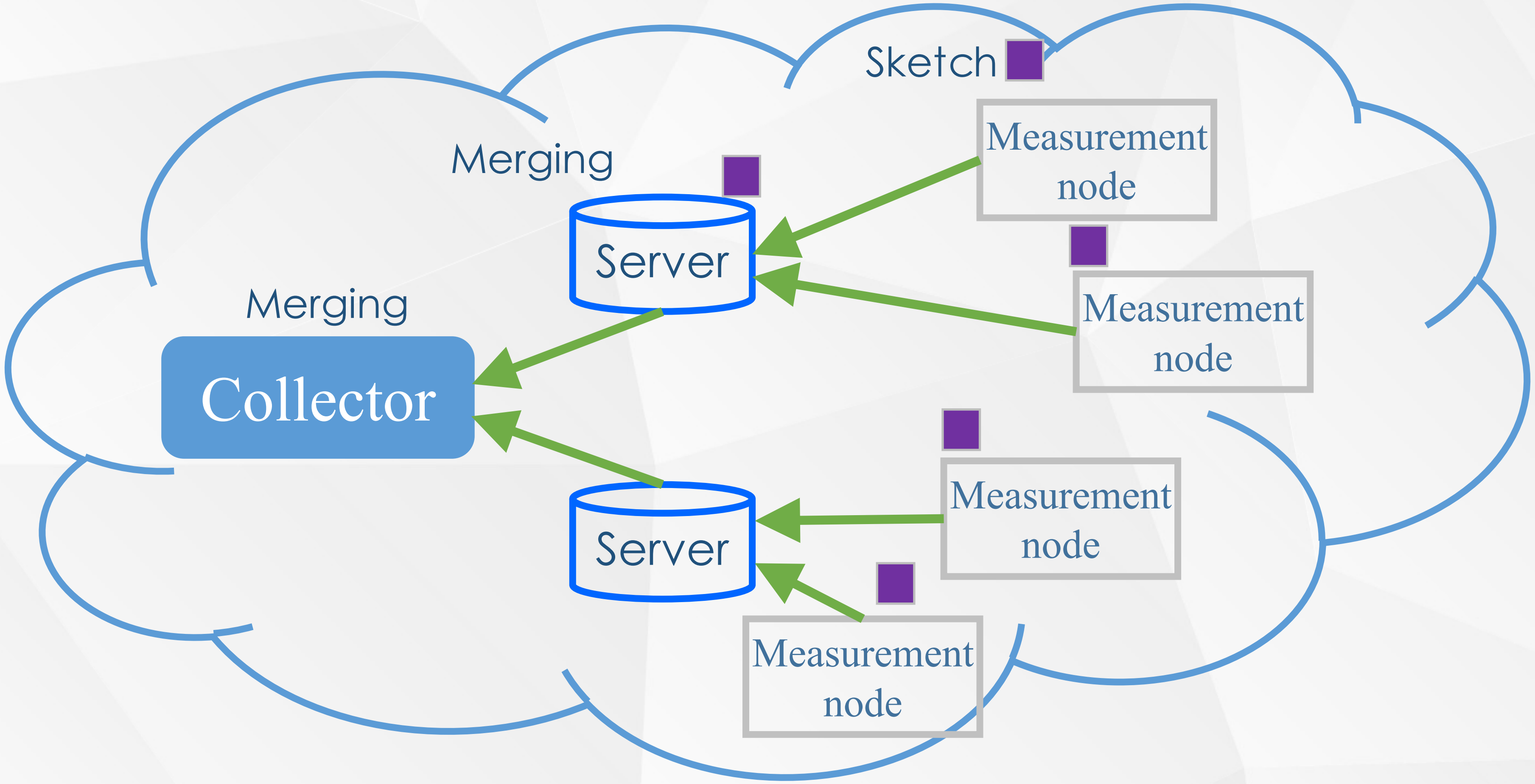
traffic characteristics:

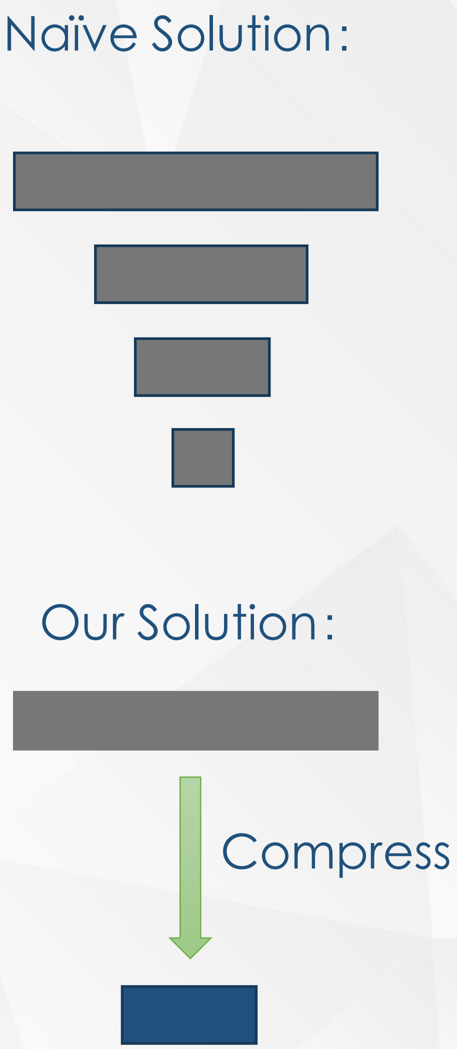
1) the available Bandwidth

2) flow size distribution

3) packet arrival rate

01 / Background---Bandwidth





01 / Background---packet rate

The packet rate is

1) naturally variable

2) could vary drastically.

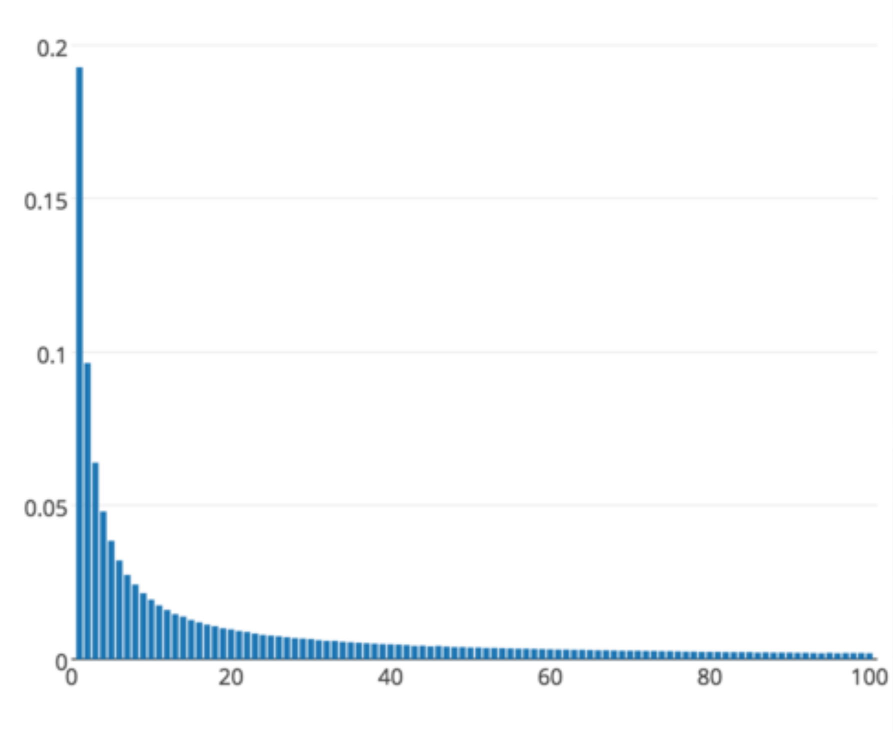
Existing sketches

1) fixed speed
2) drop packets
   when packet rate becomes much higher

Our goal:

1) minimize #memory accesses → 1
2) minimize #hash computations → 1

01 / Background---flow size distribution



Network traffic is skewed

1) Majority: mice flows

2) Minority: elephant flows

Separation is effective

1) use large and small counters

2) use different data structures

Our goal:

1) accurate separation

2) dynamically allocate memory

01 / Background---Cross platform

Existing solutions:

1) for CPU platforms
2) for netFPGA (OpenSketch NSDI13)
3) for P4Switch (UnivMon SIGCOMM16)

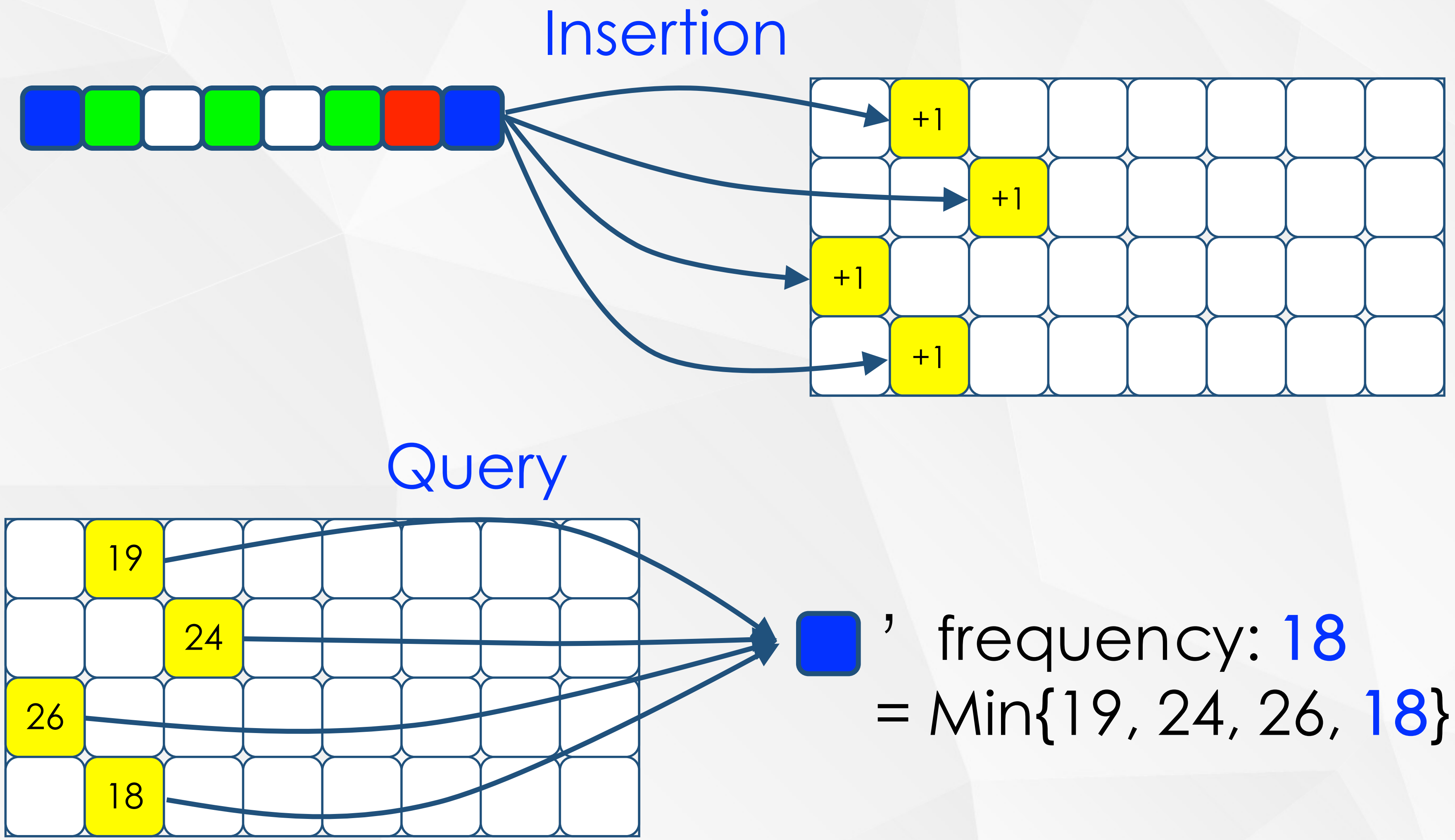
Our goal:

1) P4Switch
2) FPGA
3) GPU
4) OVS
5) CPU
6) multi-core

01 / Background- Tasks and sketches

Tasks	Sketch Algorithms
Flow size estimation	Count-Min, CM-CU, Count, ASketch
heavyhitters	Count-Min, CM-CU, Space-Saving Hashpipe, ASketch, FlowRadar, UnivMon
Heavy changes	RevSketch, FlowRadar, UnivMon
Superspreader /DDoS detection	TwoLevel
Flow size distribution	MRAC, FlowRadar
Cardinality	FM, LC, UnivMon
Entropy	FlowRadar, UnivMon

01 / Background- CM sketches



02

PART TWO

Elastic Sketches

Elastic Sketch

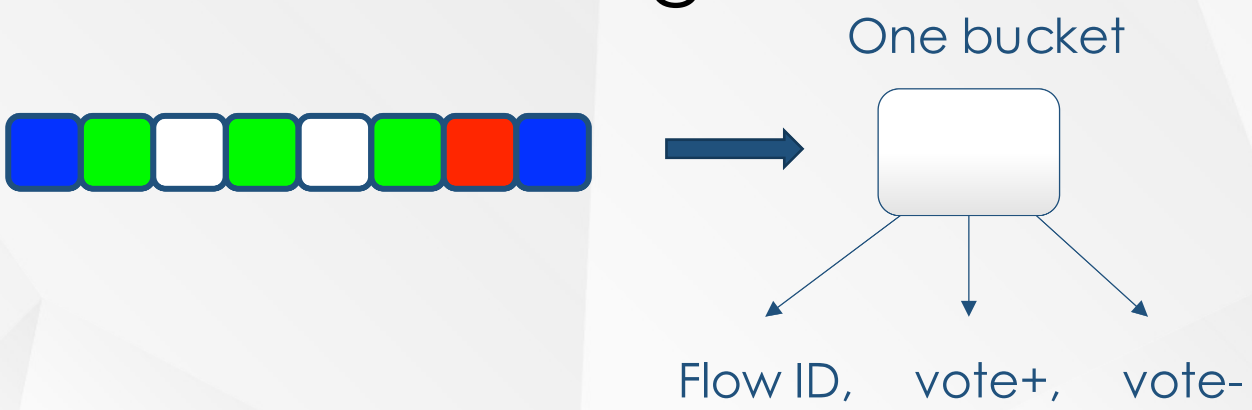
02 / Elastic (Rationale)

Goal: separate elephant flows from mice flows

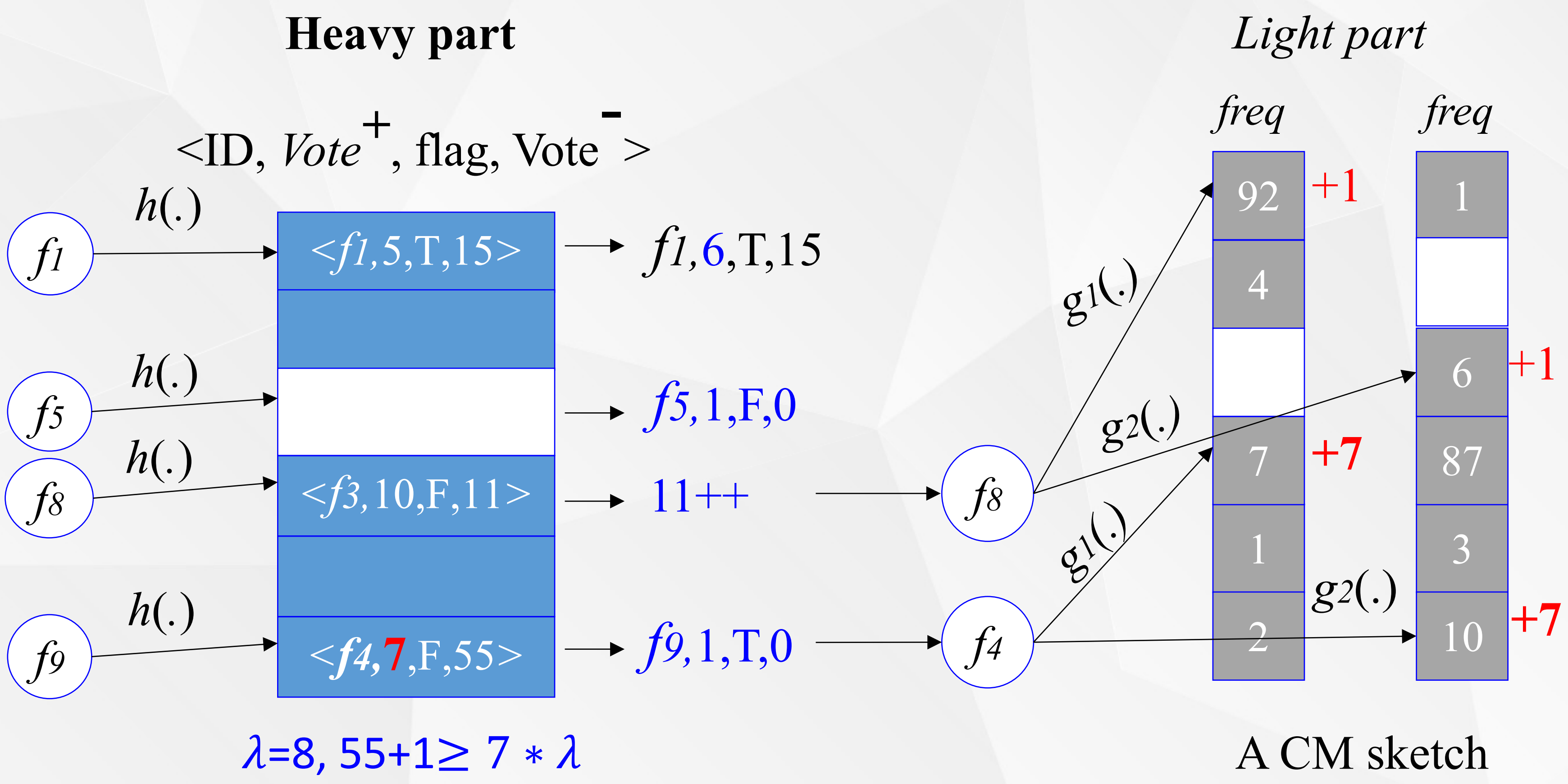
Ostracism: vote for elephant flows

Ostracism was a procedure in ancient Athens, any citizen could be voted to be expelled.

Problem: one bucket to elect the largest one



02 / Elastic (Basic version)



For elephant flows:

For mice flows:

For a bucket:

02 / Elastic (Basic version)

To query a flow, heavy part → light part

1. To query it in the heavy part

   check the flag of the mapped bucket

   1) flag = false: report the $vote^+$ with no error

   2) flag = true: $vote^+$ + *f_light*

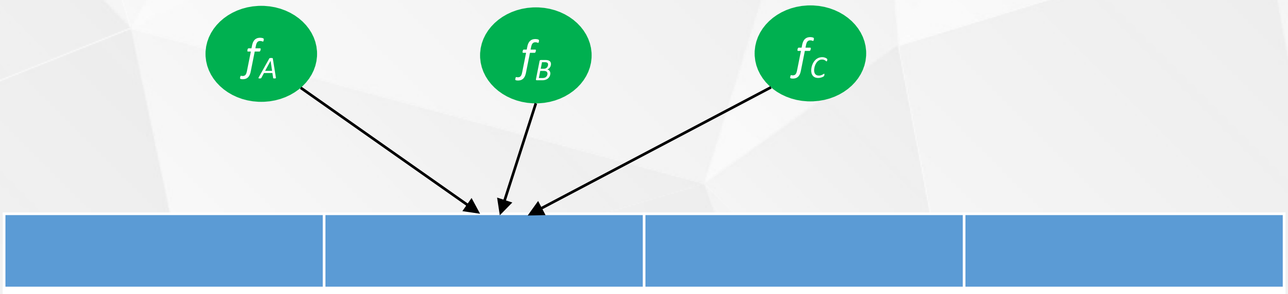
2. To query it in the light part

   report its frequency *f_light* as how CM sketch does

02 / Elastic (Basic version)

Error Bound:

$$\hat{f}_i \leqslant f_i + \epsilon\|\boldsymbol{f}_l\|_1 < f_i + \epsilon\|\boldsymbol{f}\|_1$$



Elephant collision: 1+ elephant flows are mapped to the same bucket. Elephant collision rate

$$P_{hc} = 1 - \left(\frac{H}{w} + 1\right) e^{-\frac{H}{w}}$$

02 / Elastic (Adaptivity)

1) Adaptive to Available Bandwidth

2) Adaptive to Packet Rate

3) Adaptive to Flow Size Distribution

02 / Elastic (Adaptive to Bandwidth)

To adapt to available bandwidth

1) the light part is large
2) compress the light part before sending

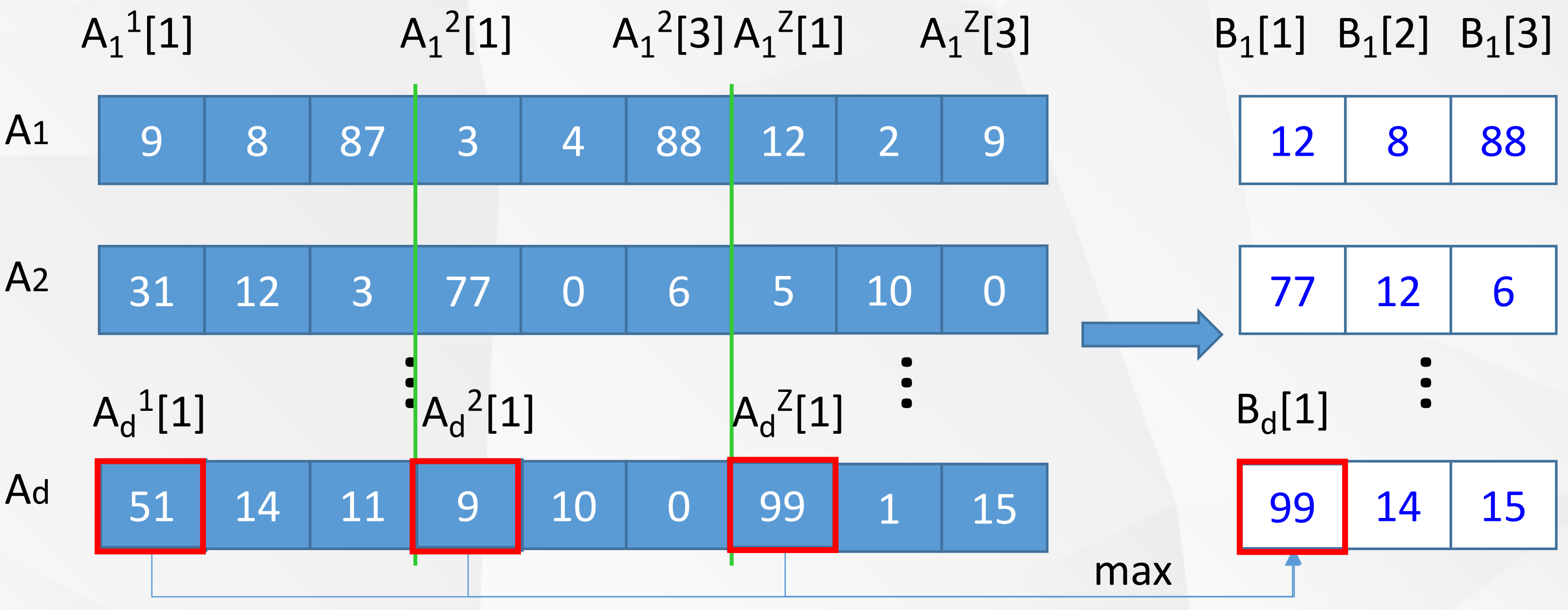
3 key operations to compress the sketch

1) how to group counters?
2) how to merge counters in a group?
3) how to change hash functions?

02 / Elastic (Adaptive to Bandwidth)

1) split the sketch A into 3 divisions
2) build a sketch B
3) counters with the same index as one group

10%6%3=10%3
10%8%4=10%4



h(.)%9 →

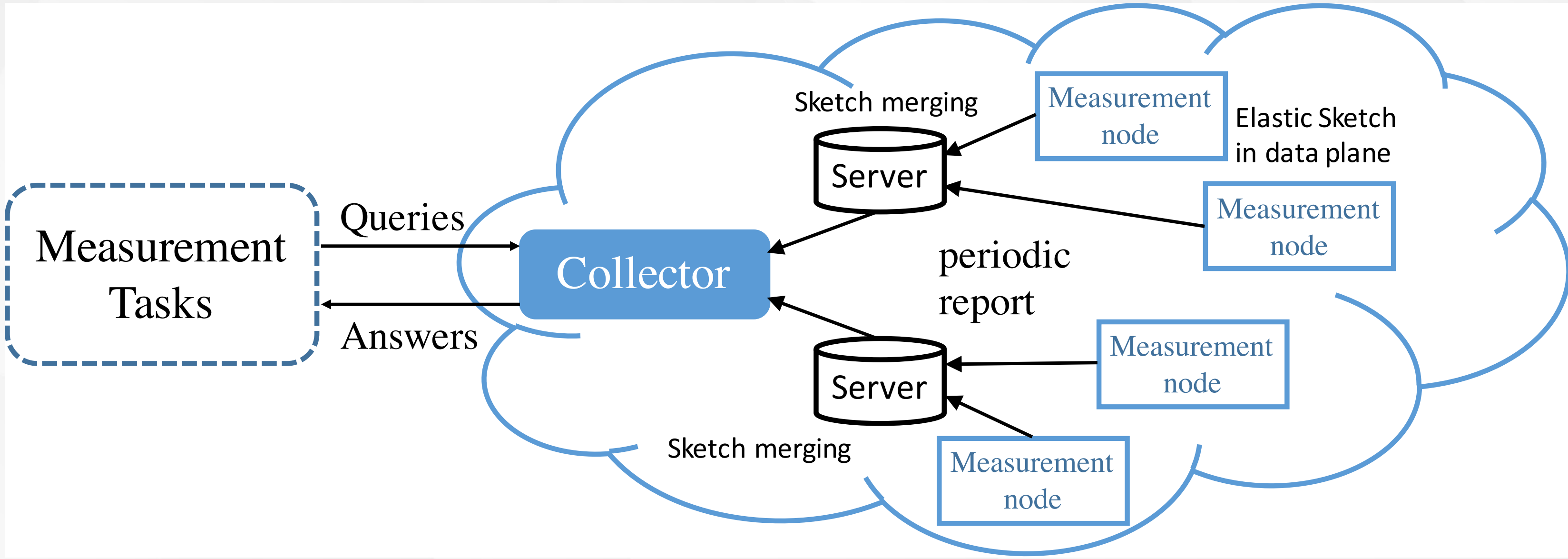
h(.)%9%3

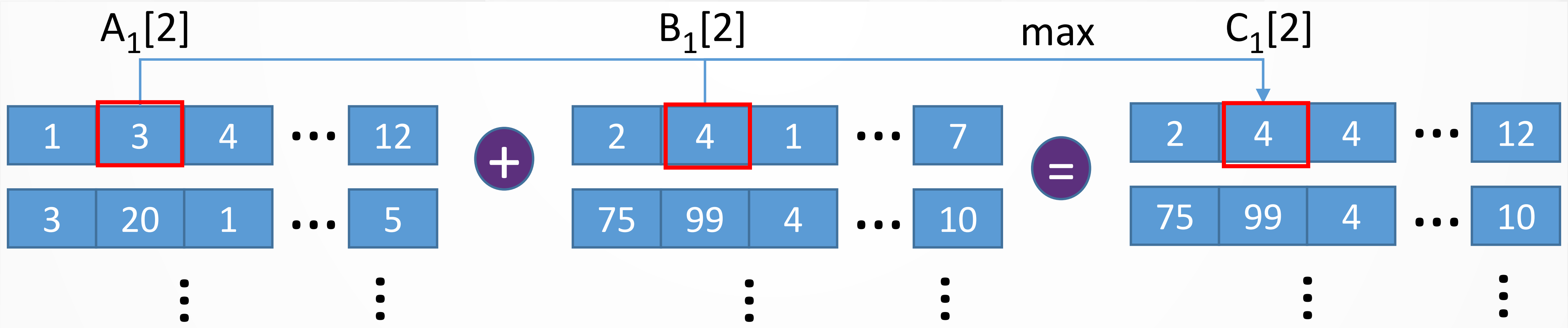
=h(.)%3

02 / Elastic (Adaptive to Bandwidth)

Merging sketches

1) sum merging
2) max merging





02 / Elastic (Adaptive to packet rate)

Second, we show how to adapt to high packet rate?

1) buffer the incoming packets in an input queue

2) when # packets in the input queue > Threshold

(1) access only the heavy part

(2) the insertion operation is modified:

if f1 is replaced by f2, then sizeof(f2)= sizeof(f1).

02 / Elastic (Adaptive to flow size distribution)

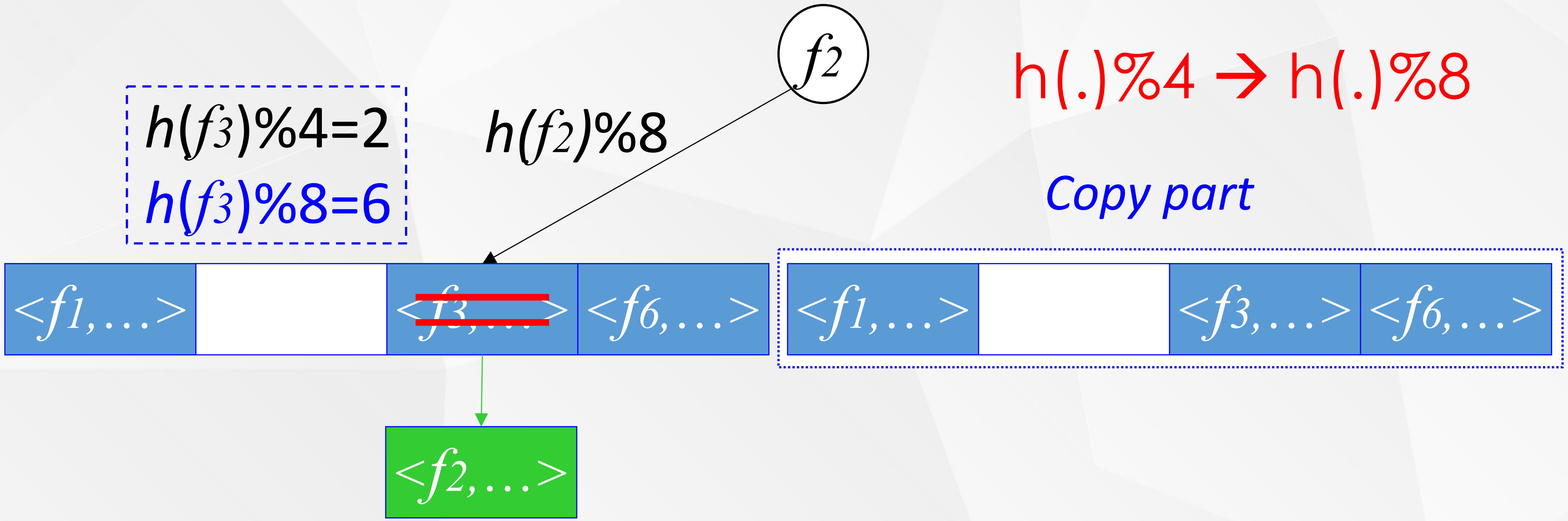
Third, #elephant flows is unknown and can vary a lot

1) # elephant flows in the heavy part is increasing

2) heavy part should be adaptive to changes in traffic distribution

Solution: copy the heavy part when #elephant flows exceeds a threshold

02 / Elastic (Adaptive to flow size distribution)



03

PART THREE

Optimizations

03 / Optimizations

To minimize the elephant collision rate,

1) Software Version

2) Hardware Version

3) P4Switch Version

4) Multi-Core Version

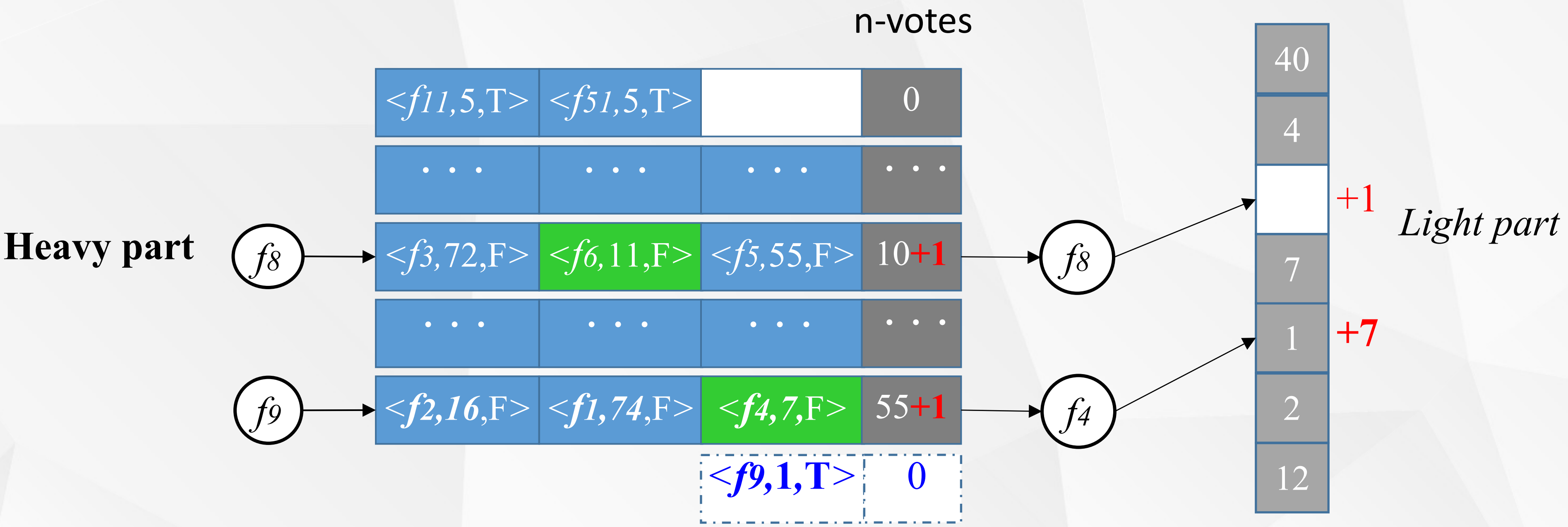
03 / Optimizations (Software Version)

1) use one bucket to store multiple flows

2) all the flows in each bucket share one vote⁻ field

3) try to evict the smallest flow in the mapped bucket

4) use one array in the light part

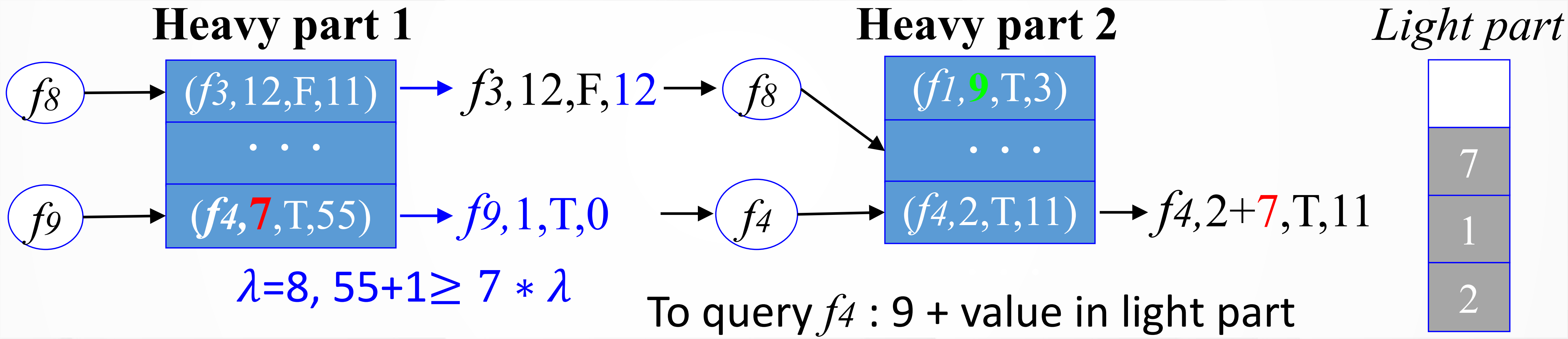


03 / Optimizations (Hardware Version)

1) using several sub-tables in the heavy part

2) each flow have several candidate buckets, and thus the elephant collision rate drops significantly.

3) the sub-tables have the same operation but different hash functions, thus suitable for hardware.



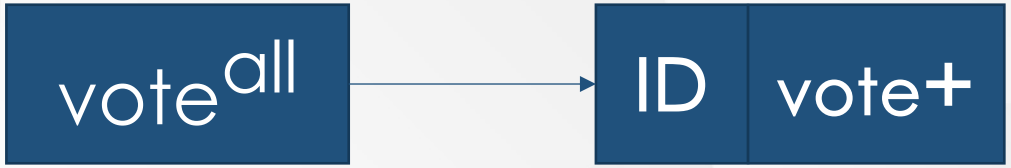
03 / Optimizations (P4Switch Version)

1) each stage in two physical stages: $vote^{all}$, and (key, vote+)

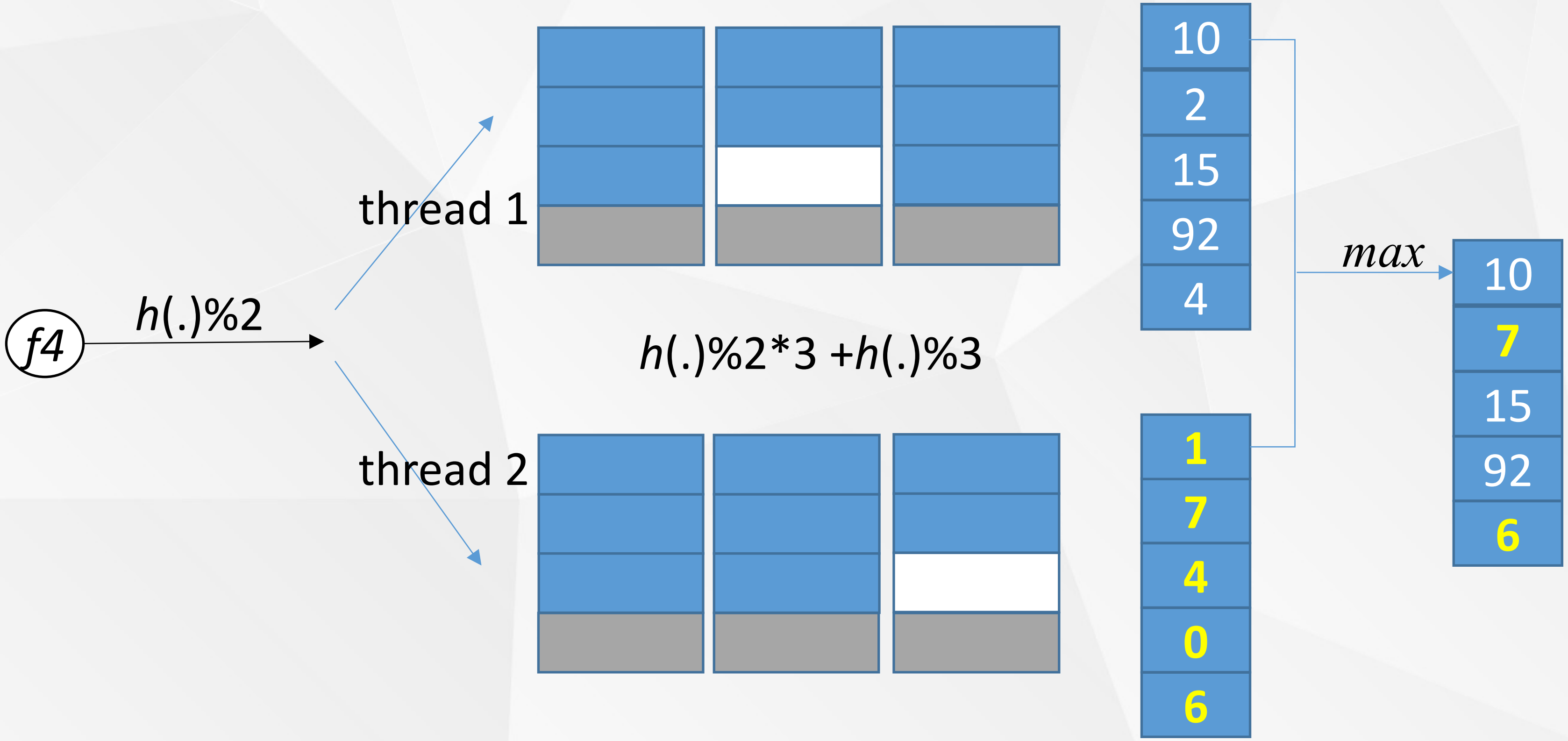
2) When $vote^{all}$ /vote+ $\geqslant \lambda'$, we perform an eviction operation. We recommend $\lambda' = 32$.

3) When an item in a bucket is evicted to the next stage, we consider its frequency as 1.

4) When (f , vote+) is evicted by (f1, 1), we set the bucket to (f1, vote+ + 1).



03 / Optimizations (Multi-Core Version)



04

PART FOUR

Applications

04 / Applications

1) Flow size estimation

2) Heavy Hitter detection

3) Heavy Change detection

4) Flow Size Distribution

5) Entropy

6) Cardinality

05

PART Five

Implementations

05 / Applications

1) P4Switch

2) FPGA

3) GPU

4) CPU

5) multi-core

6) OVS

06

PART SIX

Experimental results

06 / Experiments (Setup)

Traces: CAIDA

Trace	Date	#packets	#flows (SrcIP)
CAIDA1	2015/02/19	1164.9M	2.6M
CAIDA2	2015/05/21	1081.0M	3.9M
CAIDA3	2016/01/21	1835.1M	8.9M
CAIDA4	2016/02/18	1799.7M	8.4M

Metrics:

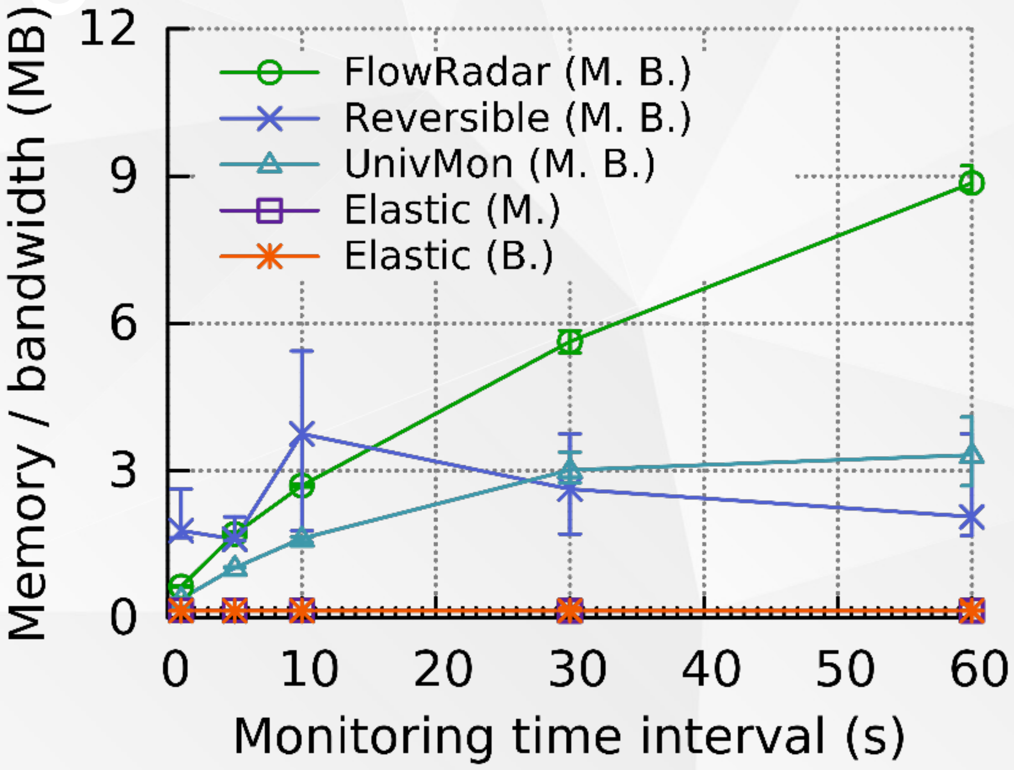
ARE, AAE, WMRE, RE, F1Score, Throughput

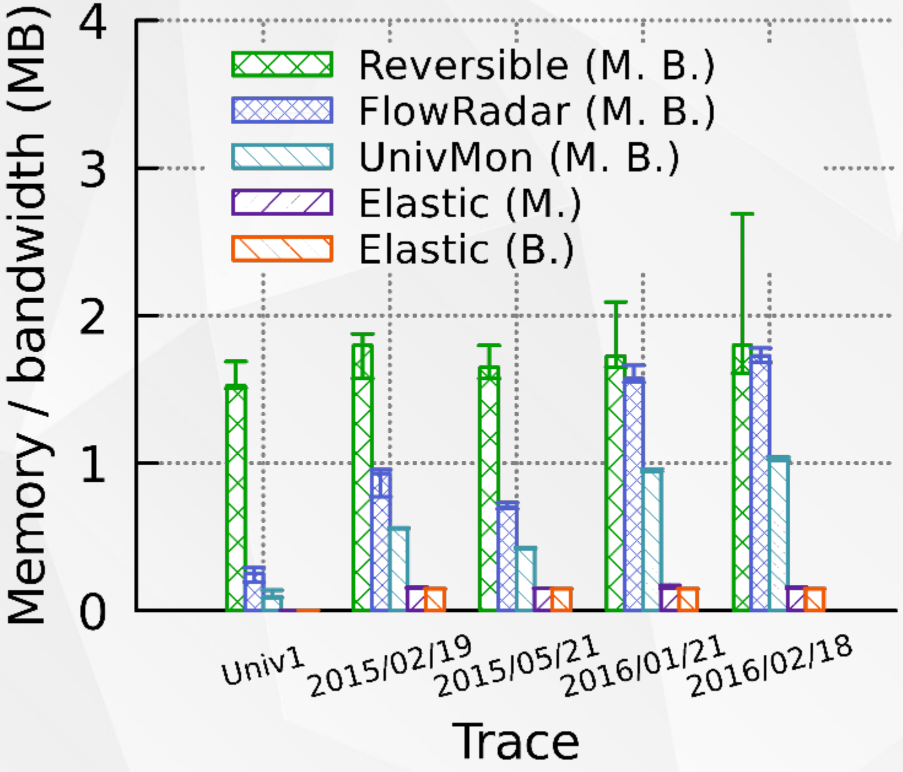
06 Experiments (Setup)

Comparisons:

1) Flow size:	CM, CU, Count
2) Heavy Hitter:	**UnivMon**, SS, CM/C+heap,, Hash pipe
3) Heavy Change:	**UnivMon**, Reversible sketch, FlowRadar
4) Distribution:	MRAC
5) Entropy:	**UnivMon**, Sieving
6) Cardinality:	**UnivMon**, linear counting (LC)

06 / Experiments (Memory/Bandwidth)

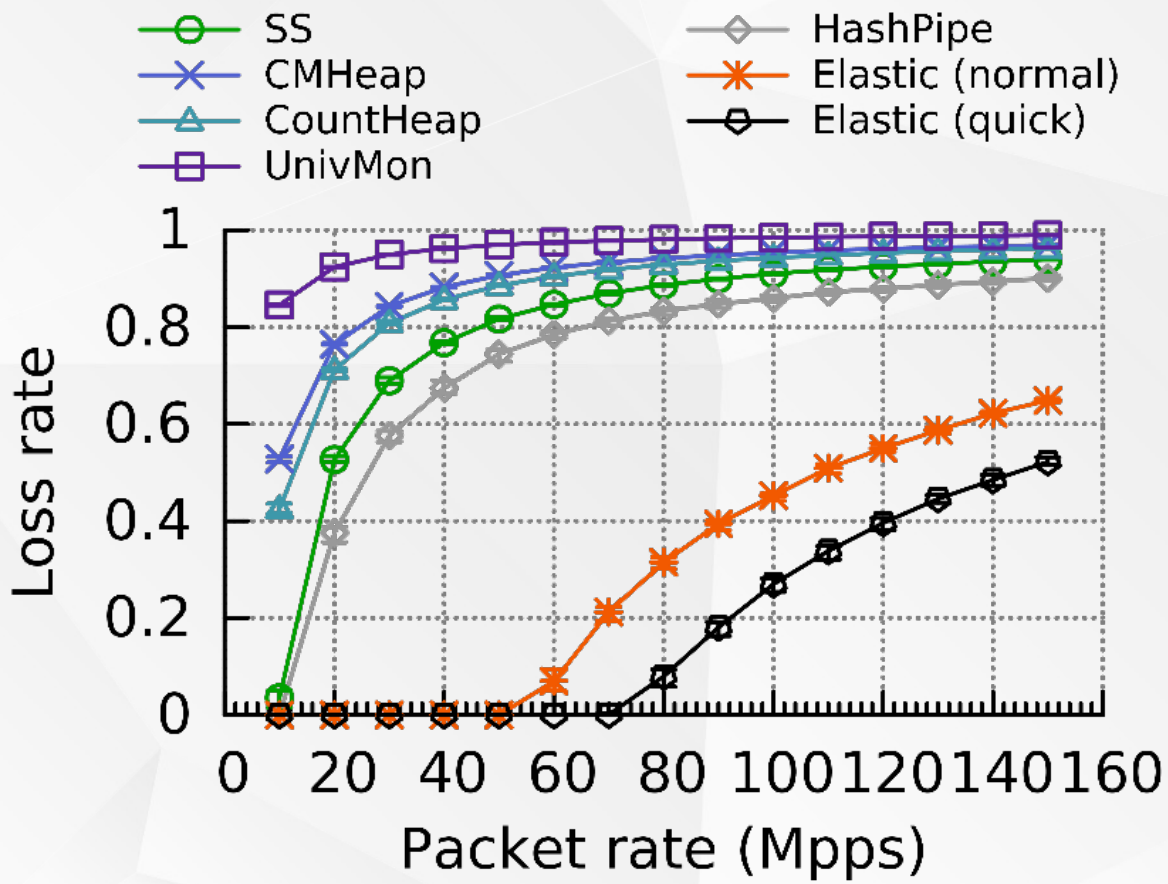


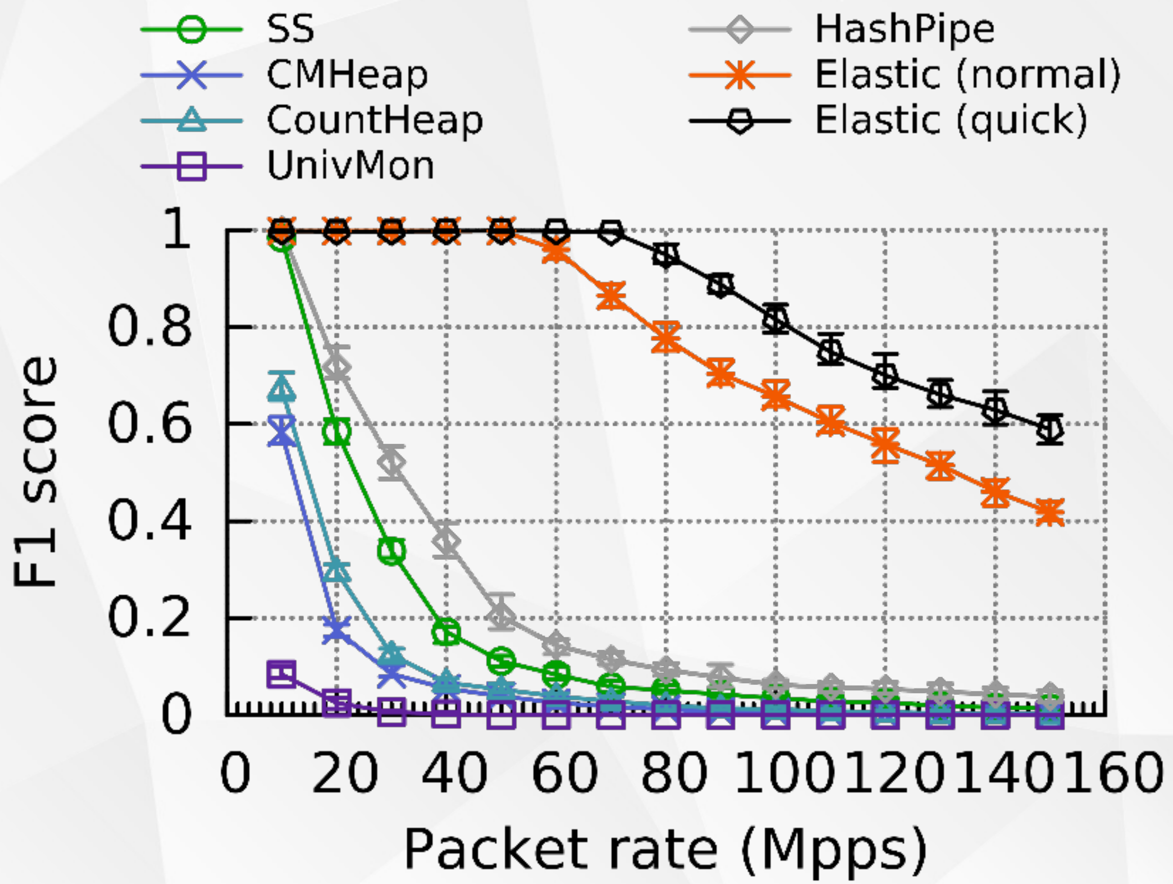


Memory or bandwidth needed to achieve 99% precision and recall in heavy change detection

06 / Experiments (Adaptivity)

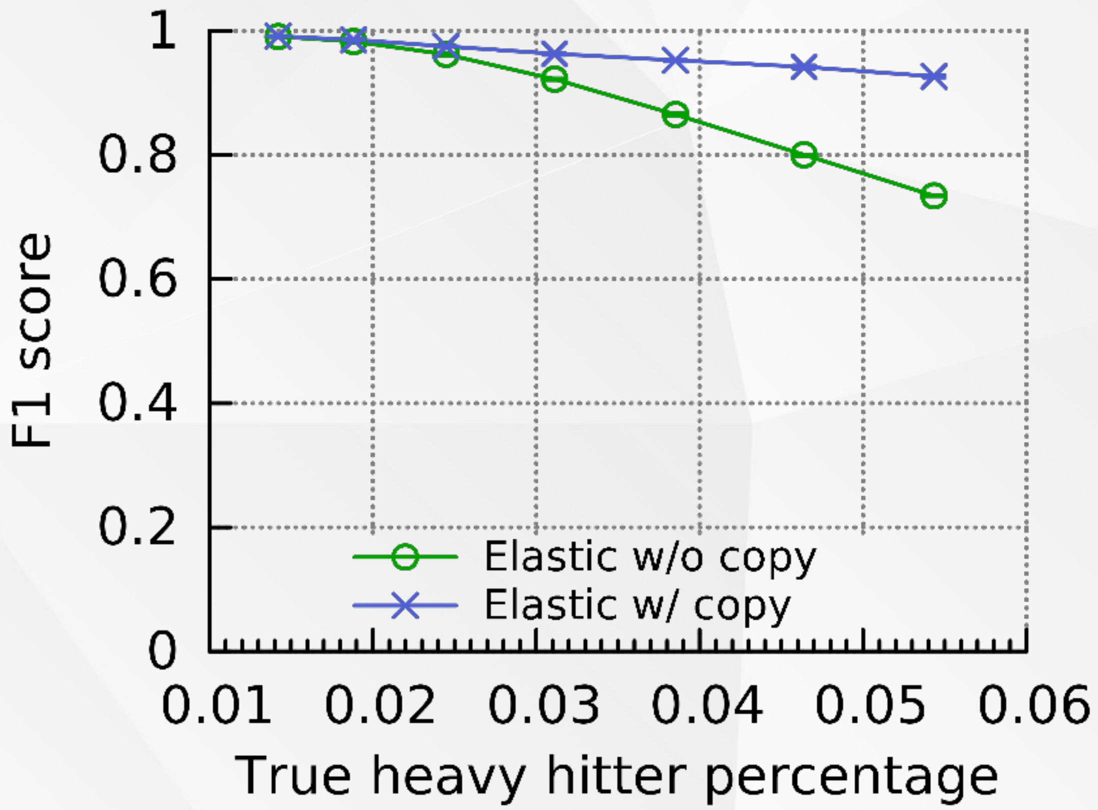
Adaptivity to packet rate

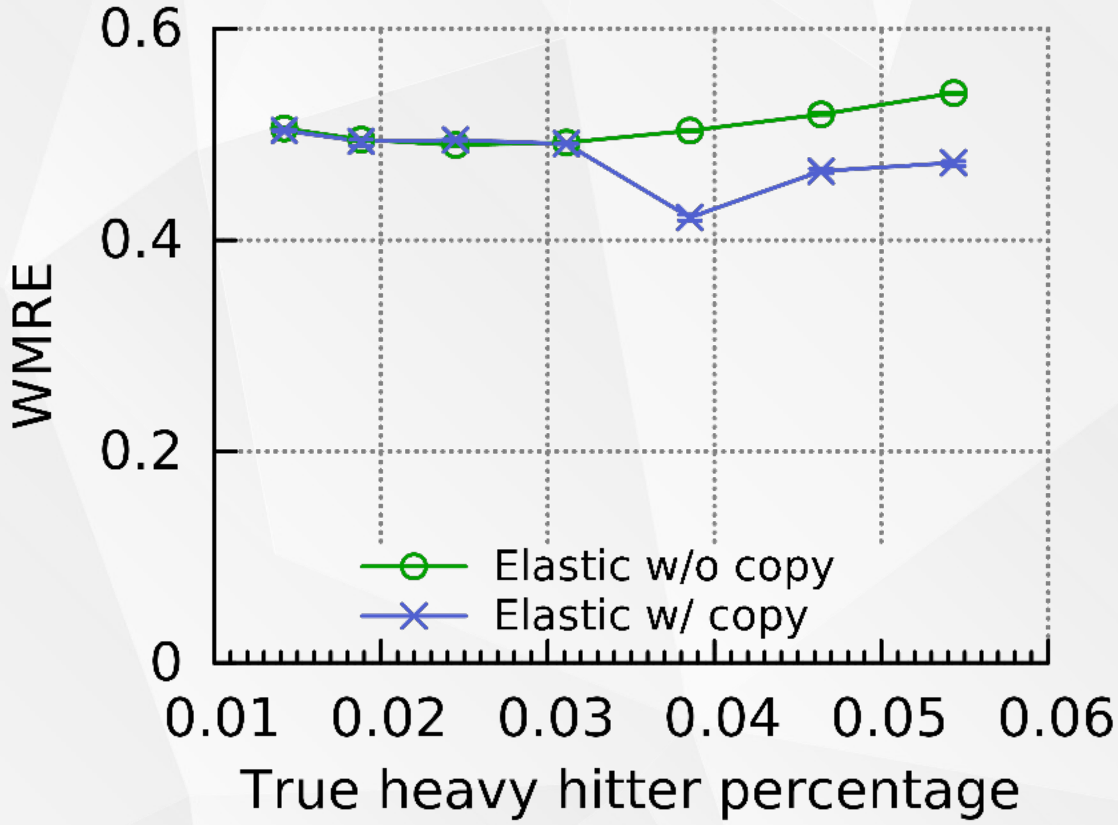




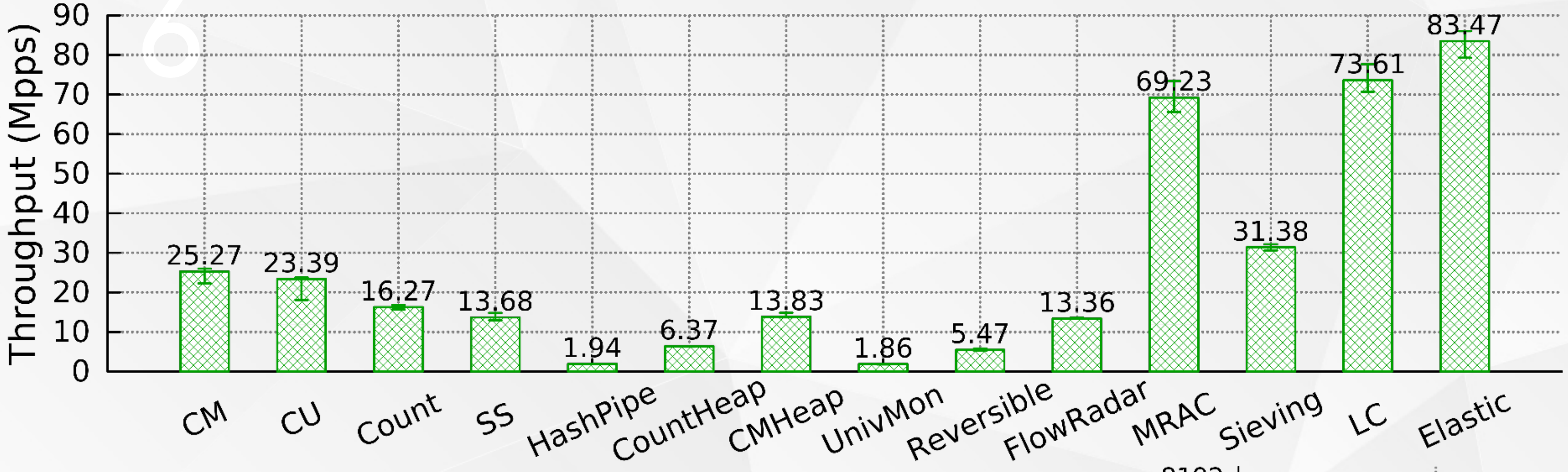
06 / Experiments (Setup)

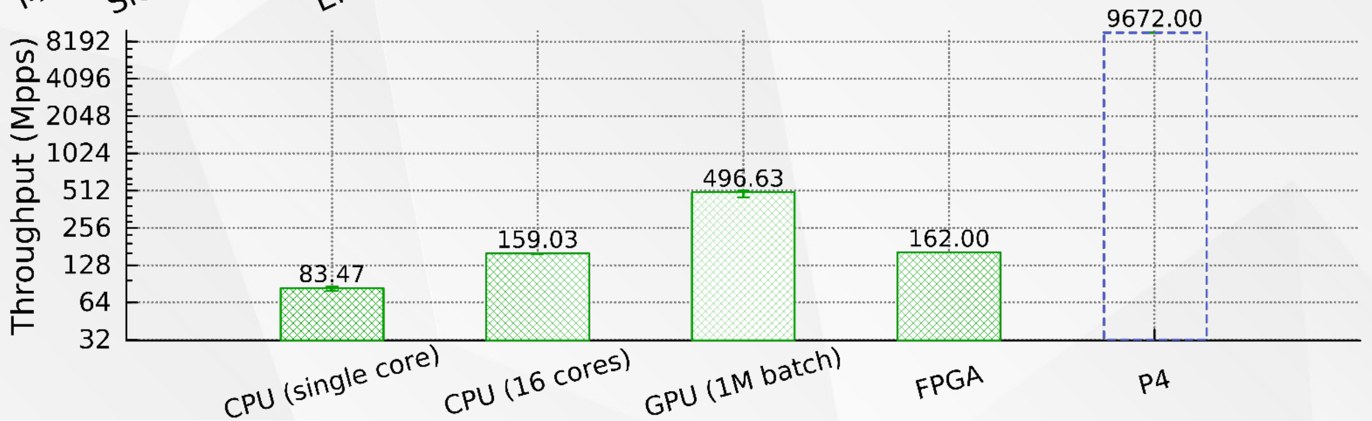
Adaptivity to flow size distribution

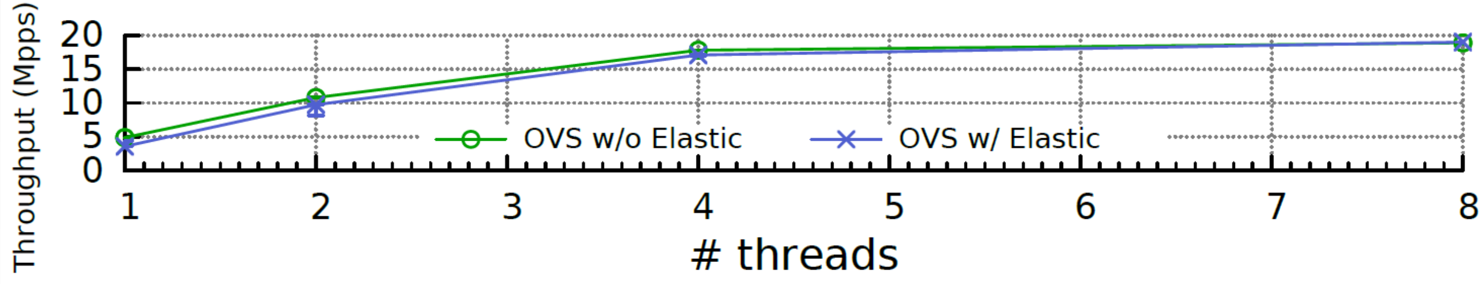




06 / Experiments (Processing Speed)







06 / Experimental Results Summary

Compared to the state-of-the-art,

1) speed improvement:	44.6 ~ 45.2 times
2) accuracy improvement:	2.0 ~ 273.7 times

Applications for more tasks in the future work.

07

PART SEVEN

Conclusion

07 / Conclusion

1. Elastic sketch:

   1) elastic, generic, fast, and accurate

   2) adaptive to traffic characteristics

   3) one sketch for 6 tasks

2. Key techniques: ostracism and compression

3. implemented on 6 platforms

   P4Switch, FPGA, GPU, CPU, multi-core CPU and OVS

THANKS

Source code: https://github.com/ElasticSketch/ElasticSketch

Tong Yang
Peking University, China
Email: yangtongemail@gmail.com
Homepage: http://net.pku.edu.cn/~yangtong/