FlowRadar:
A Better NetFlow For Data Centers

Yuliang Li

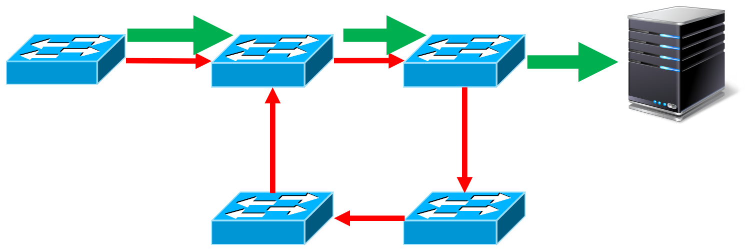
Rui Miao Changhoon Kim Minlan Yu

USC University of Southern California

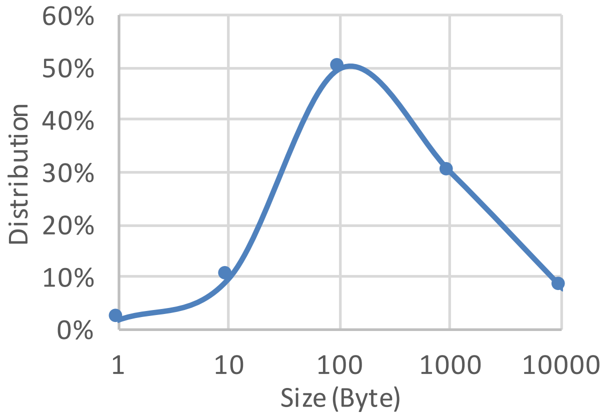
BAREFOOT NETWORKS

Flow coverage in data centers

- Flow coverage
  - – Traffic monitoring needs to cover all the flows



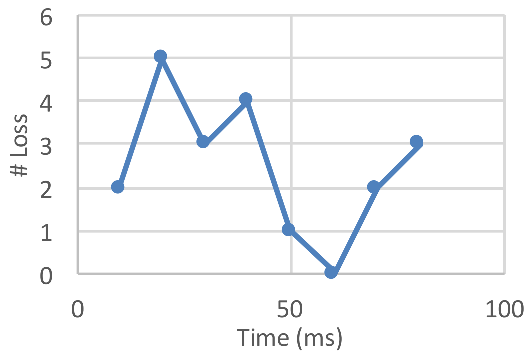
Transient loop/blackhole



Fine-grained traffic analysis

Temporal coverage in data centers

- Temporal coverage
  - Traffic monitoring needs millisecond-level flow information

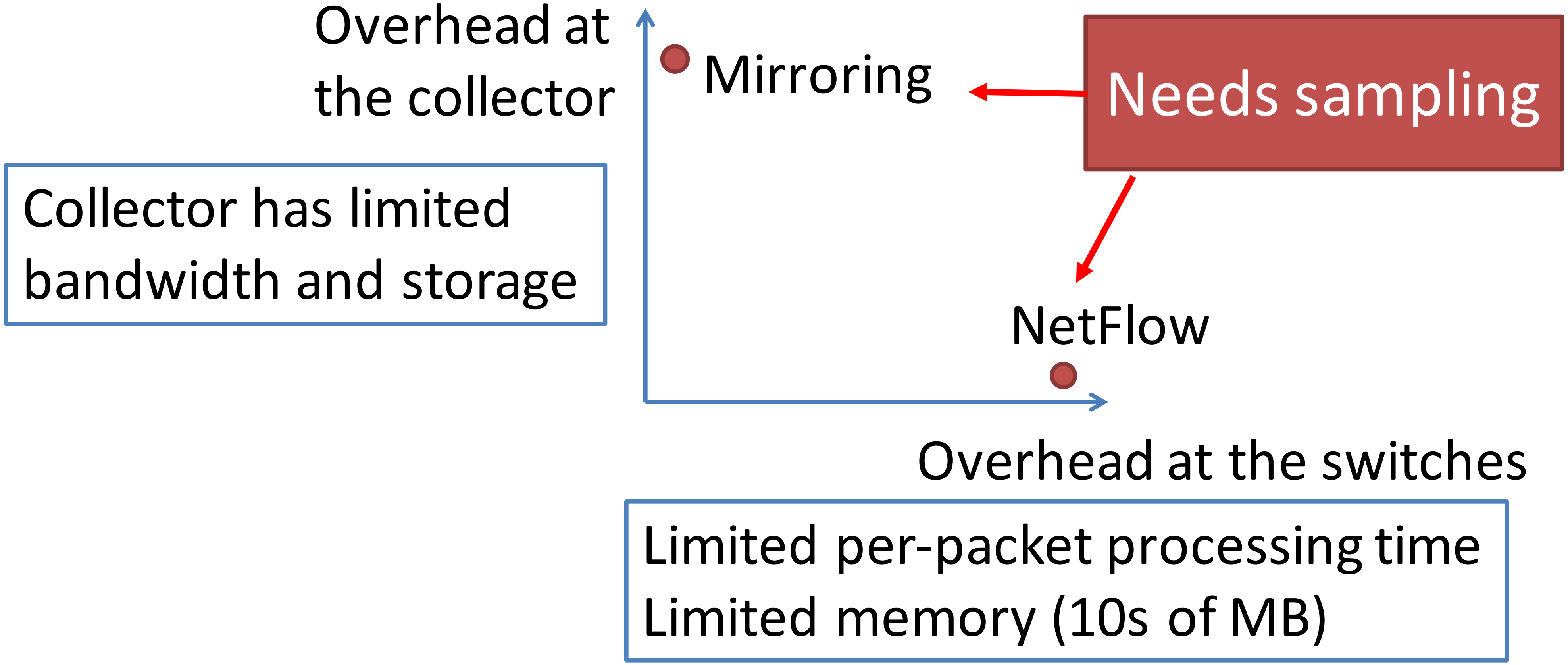


Short-time scale Loss rate

Timely attack detection

Key insight: division of labor

- Goal: report counters for all flows in fine-grained time granularity



Overhead at the collector

Mirroring

Needs sampling

NetFlow

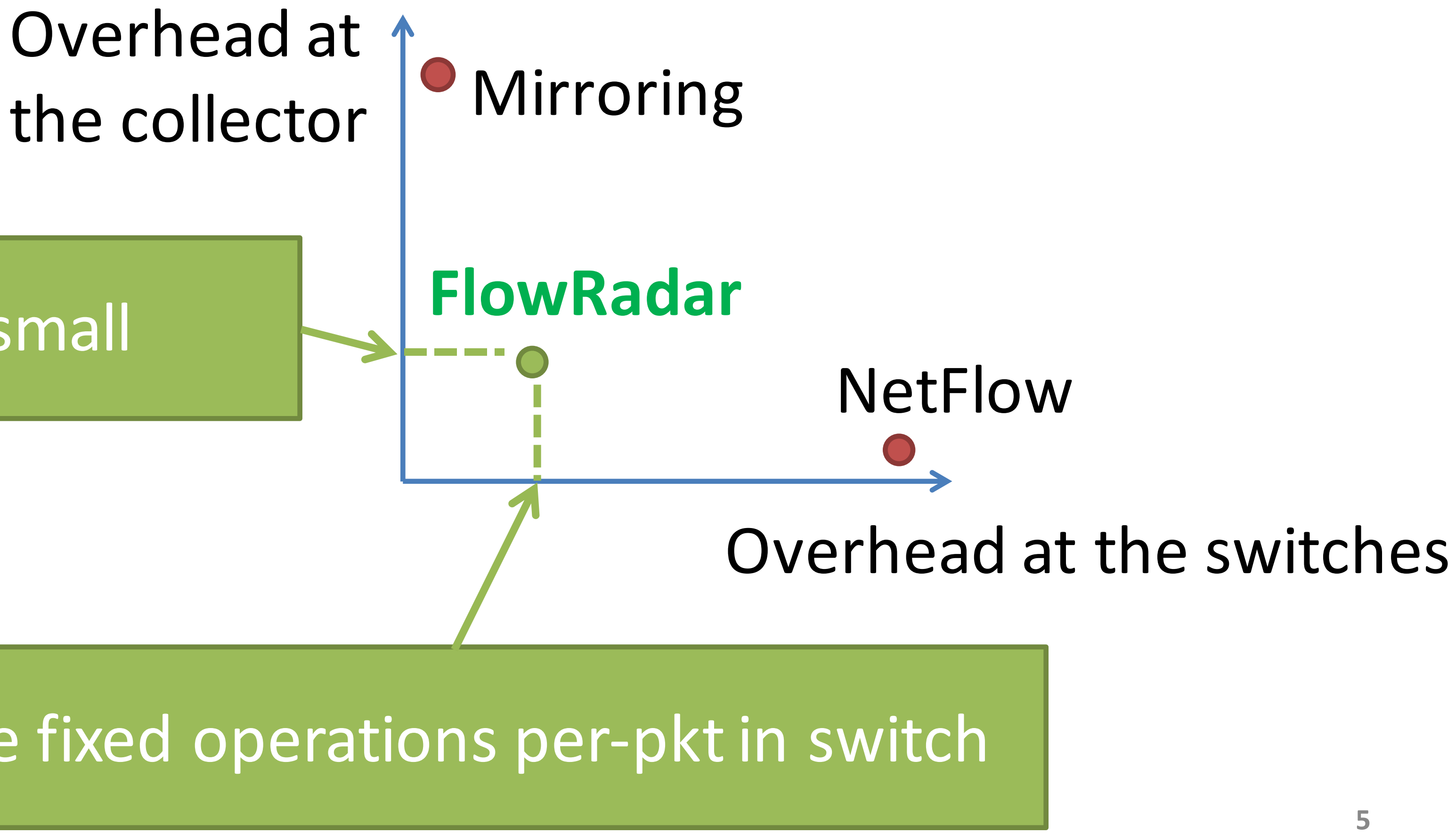
Overhead at the switches

Collector has limited bandwidth and storage

Limited per-packet processing time
Limited memory (10s of MB)

Key insight: division of labor

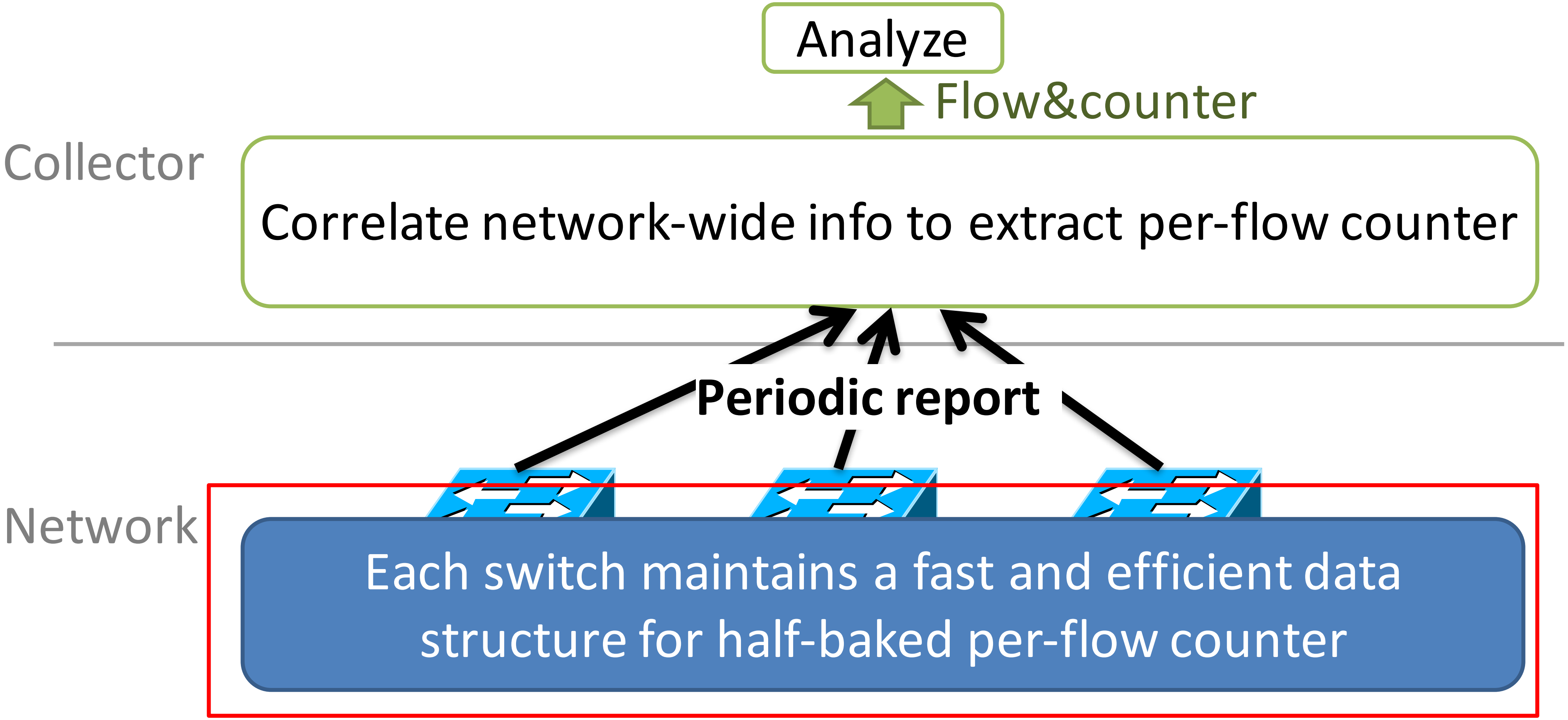
- Goal: report counters for all flows in fine-grained time granularity



Keep the memory usage small

Use fixed operations per-pkt in switch

FlowRadar architecture



Analyze

Flow&counter

Collector

Correlate network-wide info to extract per-flow counter

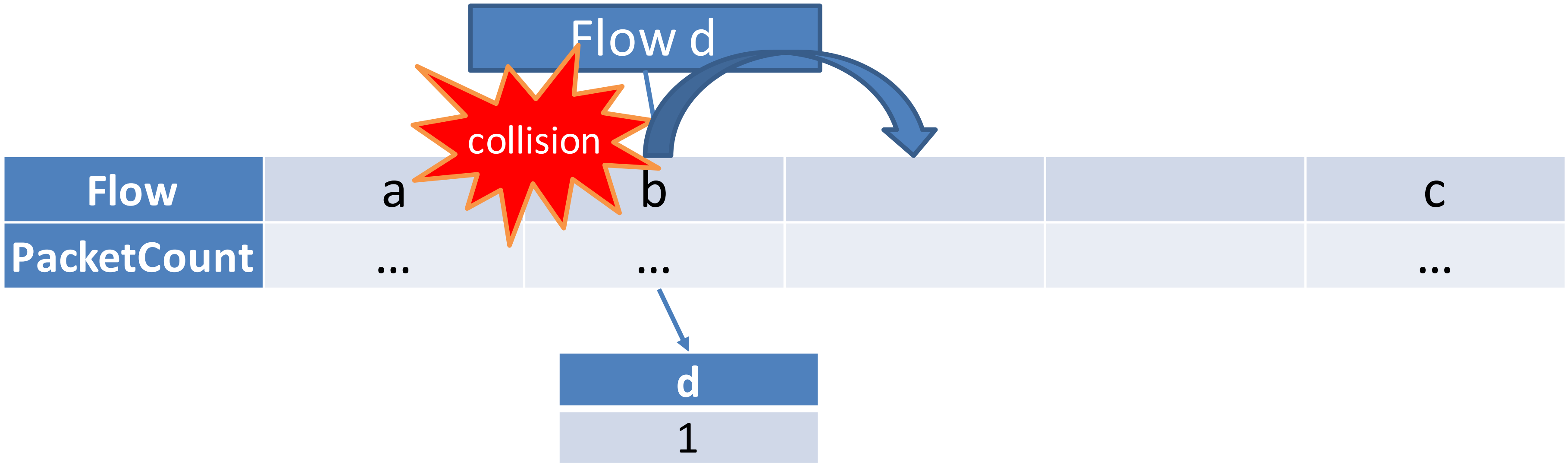
Periodic report

Network

Each switch maintains a fast and efficient data structure for half-baked per-flow counter

Challenge: handling collision?

- Handling hash collision is hard
  - Large hash table → high memory usage
  - Linked list/Cuckoo hashing→ multiple, non-constant memory accesses



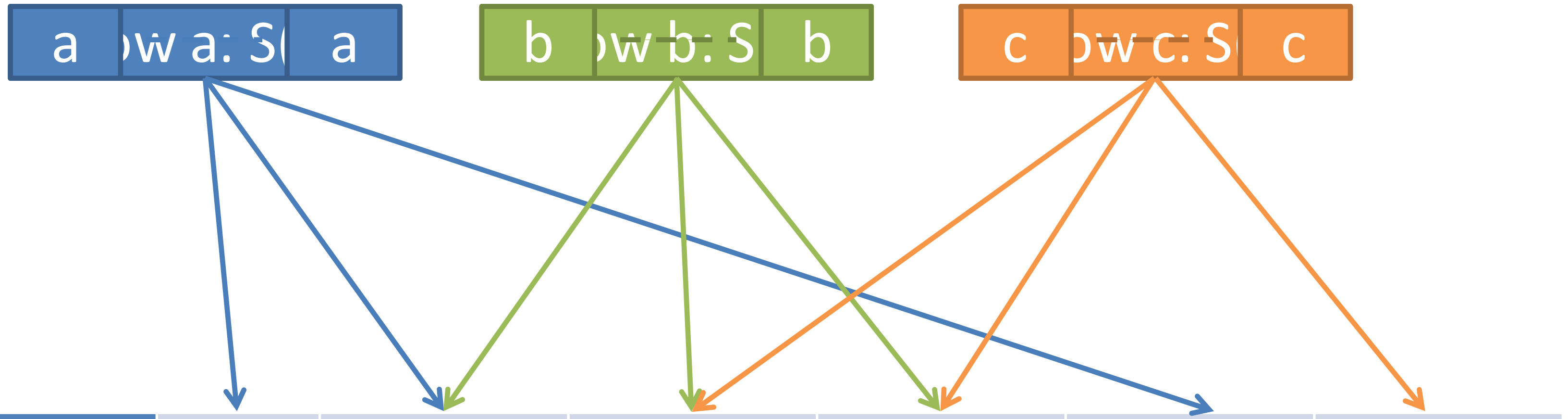
Flow	a	b			c
PacketCount	…	…			…

Switch embraces collisions!

- Handling hash collision is hard
  - Large hash table → high memory usage
  - Linked list/Cuckoo hashing→ multiple, non-constant memory accesses
- Embrace the collision
  - Less memory and constant #accesses

Switch embraces collisions!

- Embrace the collision: xor up all the flows
  - Less memory and constant #accesses



Counting table

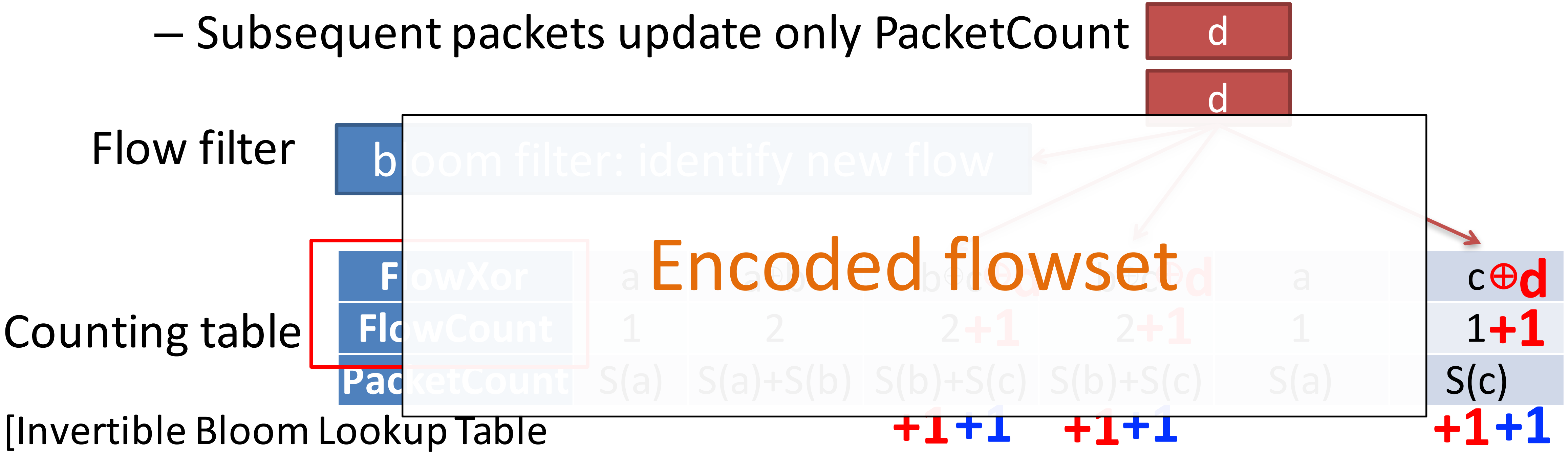
FlowXor	a	a⊕b	b⊕c	b⊕c	a	c
FlowCount	1	2	2	2	1	1
PacketCount	S(a)	S(a)+S(b)	S(b)+S(c)	S(b)+S(c)	S(a)	S(c)

[Invertible Bloom Lookup Table (arXiv 2011)]

S(x): #packets in x

Switch embraces collisions!

- 1. Check and update the flow filter
- 2. Update counting table
  - Packet from a new flow, update all fields
  - Subsequent packets update only PacketCount



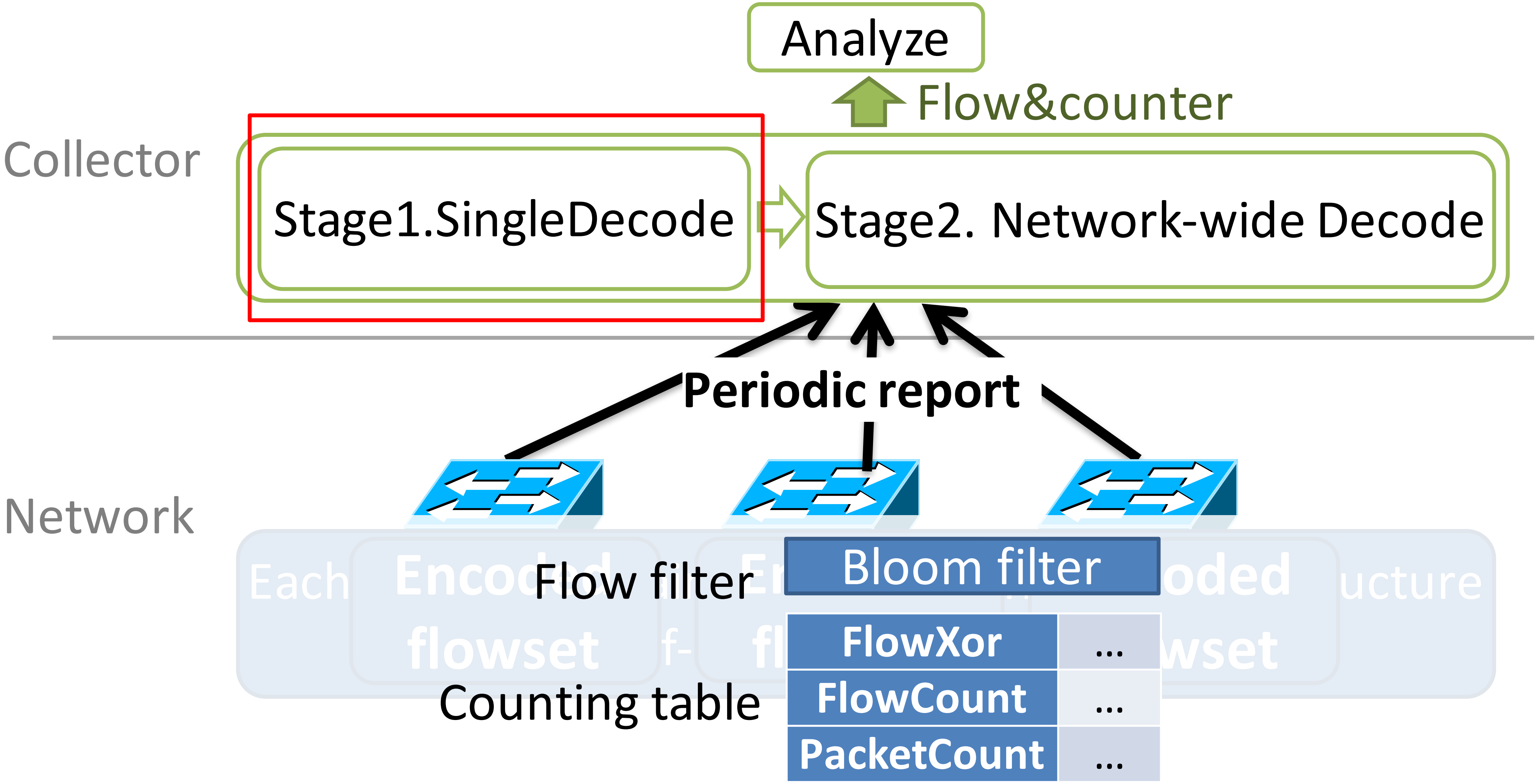
Encoded flowset

[Invertible Bloom Lookup Table (arXiv 2011)]

Easy to implement in merchant silicon

- Switch data plane
  - Fixed operations in hardware
  - Small memory, 2.36MB for 100K flows
- Switch control plane
  - Control plane gets the small flowset every 10ms
- We implemented it using P4 Language.

FlowRadar architecture

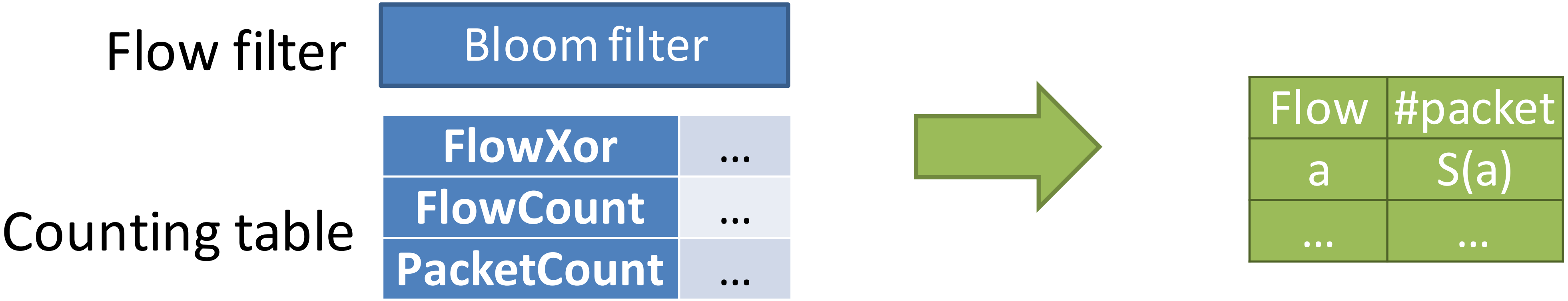


FlowXor	…
FlowCount	…
PacketCount	…

Stage1. SingleDecode

Input: a single encoded flowset

Output: per-flow counters



Flow filter

Bloom filter

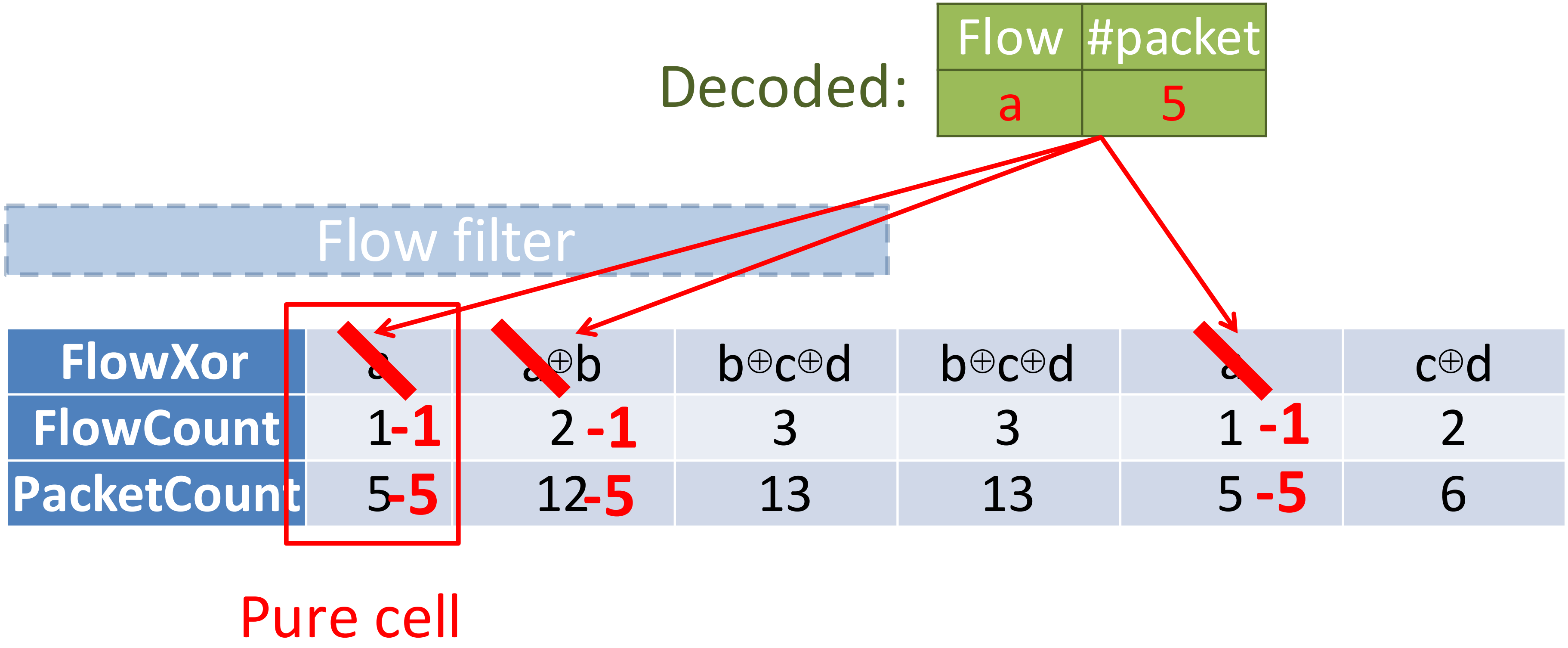
Counting table

FlowXor	…
FlowCount	…
PacketCount	…

Flow	#packet
a	S(a)
…	…

Stage1. SingleDecode

- Find a **pure cell**: a cell with one flow
- Remove the flow from all cells



Decoded:

Flow	#packet
a	5

Flow filter

FlowXor	~~a~~	~~a~~⊕b	b⊕c⊕d	b⊕c⊕d	~~a~~	c⊕d
FlowCount	1-1	2 -1	3	3	1 -1	2
PacketCount	5-5	12-5	13	13	5 -5	6

Pure cell

Stage1. SingleDecode

- Find a cell with one flow (pure cell)
- Remove the flow from all cells
  - Create more pure cells
- Iterate until no pure cells

Decoded:

Flow	#packet
a	5

Flow filter

FlowXor	0	b	b⊕c⊕d	b⊕c⊕d	0	c⊕d
FlowCount	0	1	3	3	0	2
PacketCount	0	7	13	13	0	6

Stage1. SingleDecode

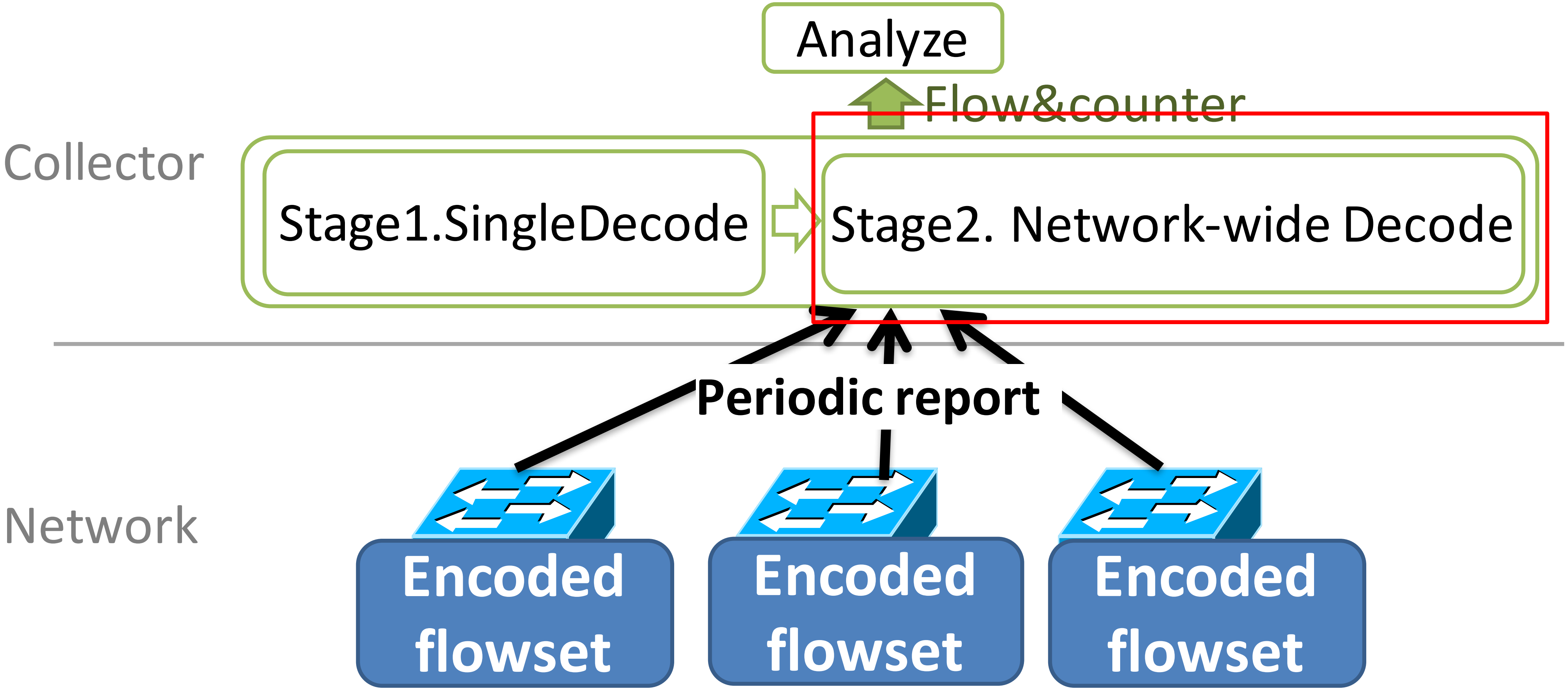
Decoded:

Flow	#packet
a	5
b	7

We want to leverage the network-wide info to decode more flows

FlowXor	0	0	c⊕d	c⊕d	0	c⊕d
FlowCount	0	0	2	2	0	2
PacketCount	0	0	6	6	0	6

FlowRadar architecture



Key insight: overlapping sets of flows

- The sets of flows overlap across hops
  - We can use the redundancy to decode more flows
- Use d… h… f… …

Provision memory based on avg(#flows), not max(#flows)

- SingleDecode for normal case
- Network-wide decoding for bursts of flows

FlowXor	a	a⊕c⊕d	b⊕c⊕d	a⊕b⊕c	b⊕d
FlowCount	1	3	3	3	2
PktCount					

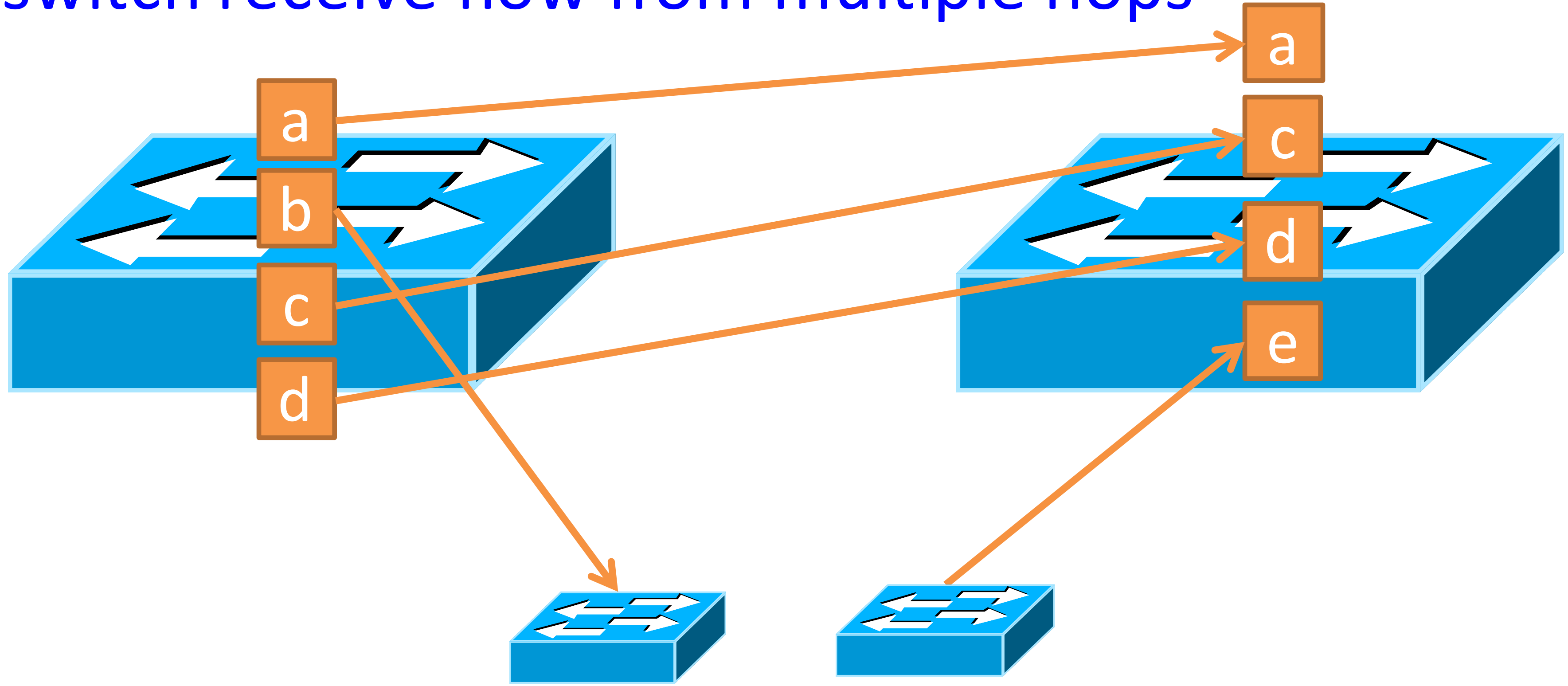
FlowXor	a⊕d	a⊕c	b⊕c⊕d	a⊕b⊕c	b⊕d
FlowCount	2	2	3	3	2
PktCount					

Use ~~5~~ 10 cells to decode 4 flows

Collector can leverage flowsets from all switches to decode more

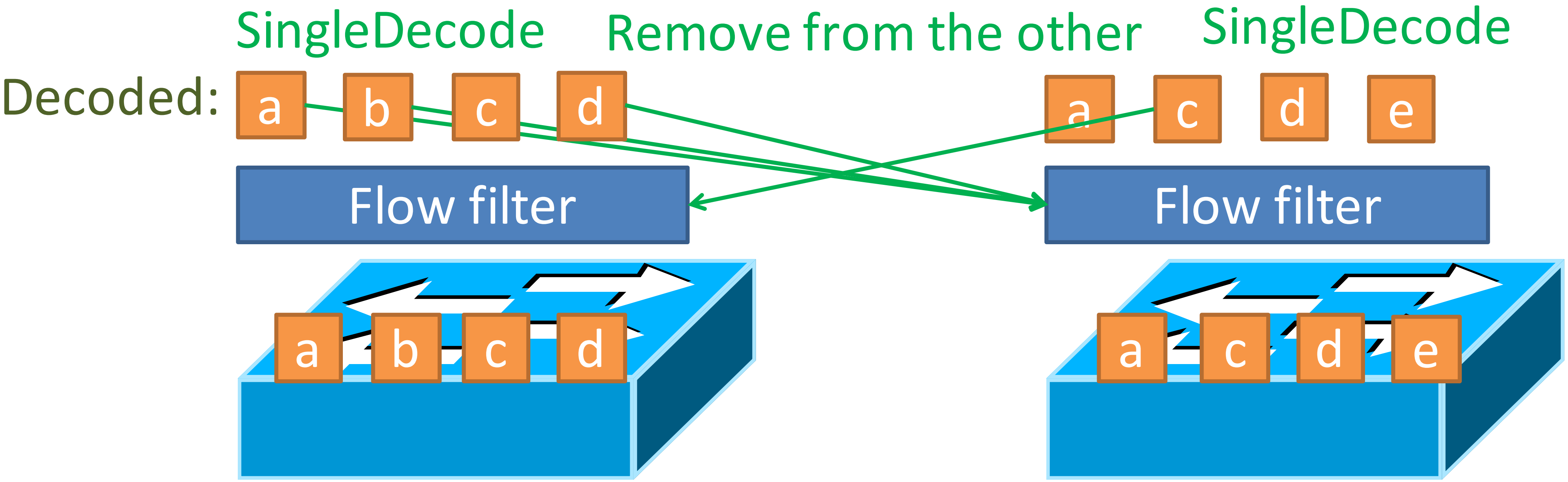
Challenge 1: sets of flows not fully overlapped

- Flows from one switch may go to different next hops
- One switch receive flow from multiple hops



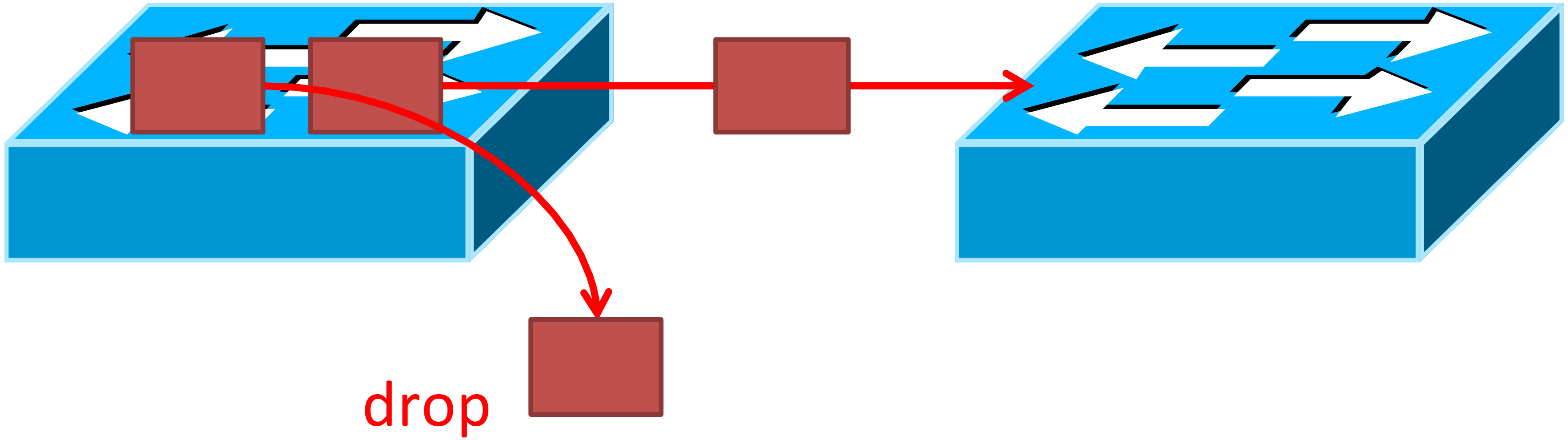
Challenge 1 solution: use flow filter to check

- Generalize to network
  - No need for routing info
  - Incremental deployment



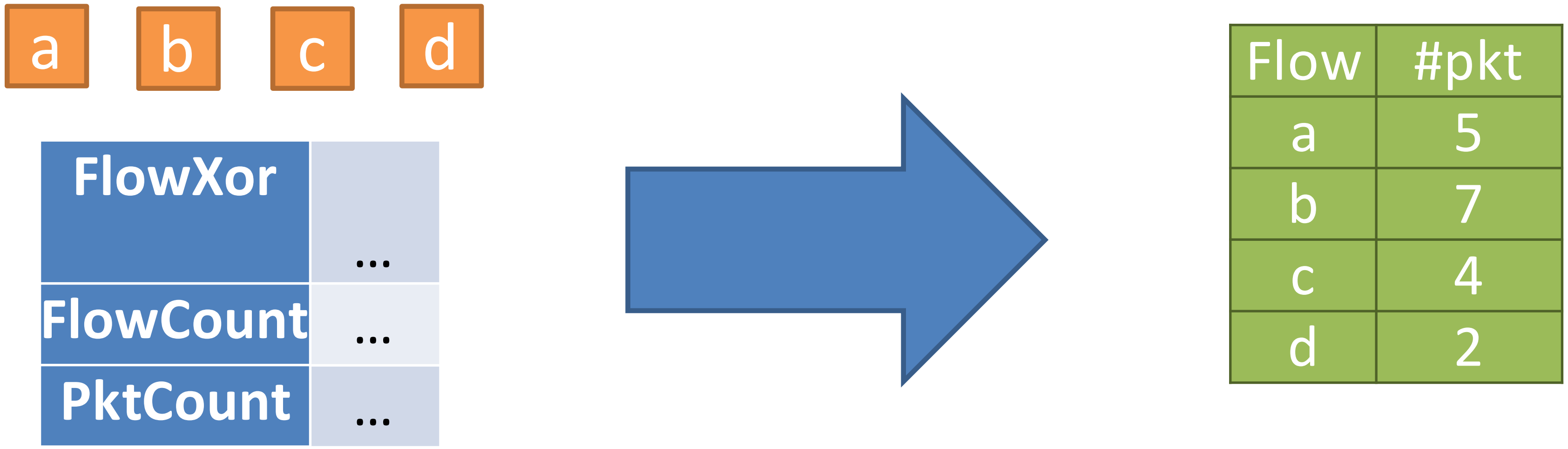
Challenge 2: counters are different across hops

- The counter of a flow may be different across hops
  - Some packets may get lost
  - On-the-fly packets



Challenge 2 solution: solve linear equations

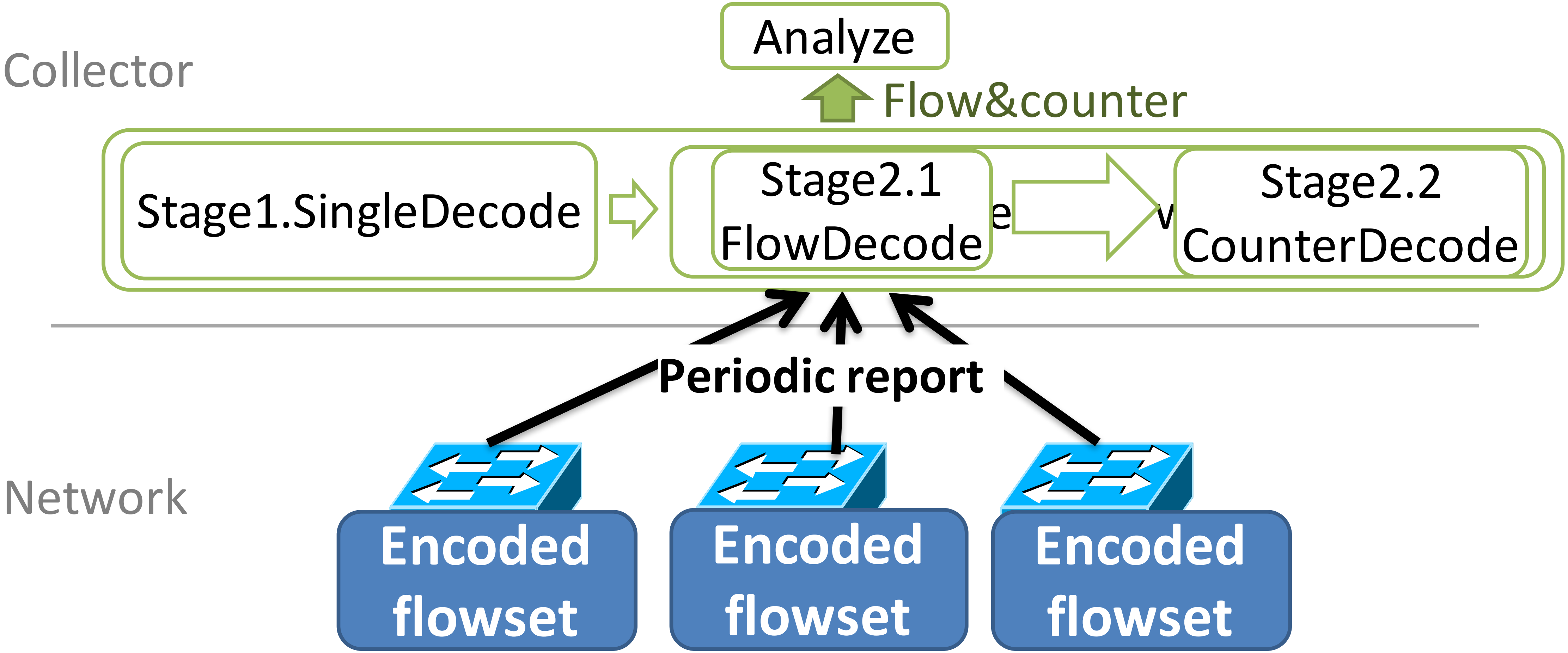
- We got full list of flows
- Combine with counting table
- Construct and solve a linear equation system for each switch
- Speed up by using counter's properties to stop solver earlier



FlowXor	…
FlowCount	…
PktCount	…

Flow	#pkt
a	5
b	7
c	4
d	2

FlowRadar architecture

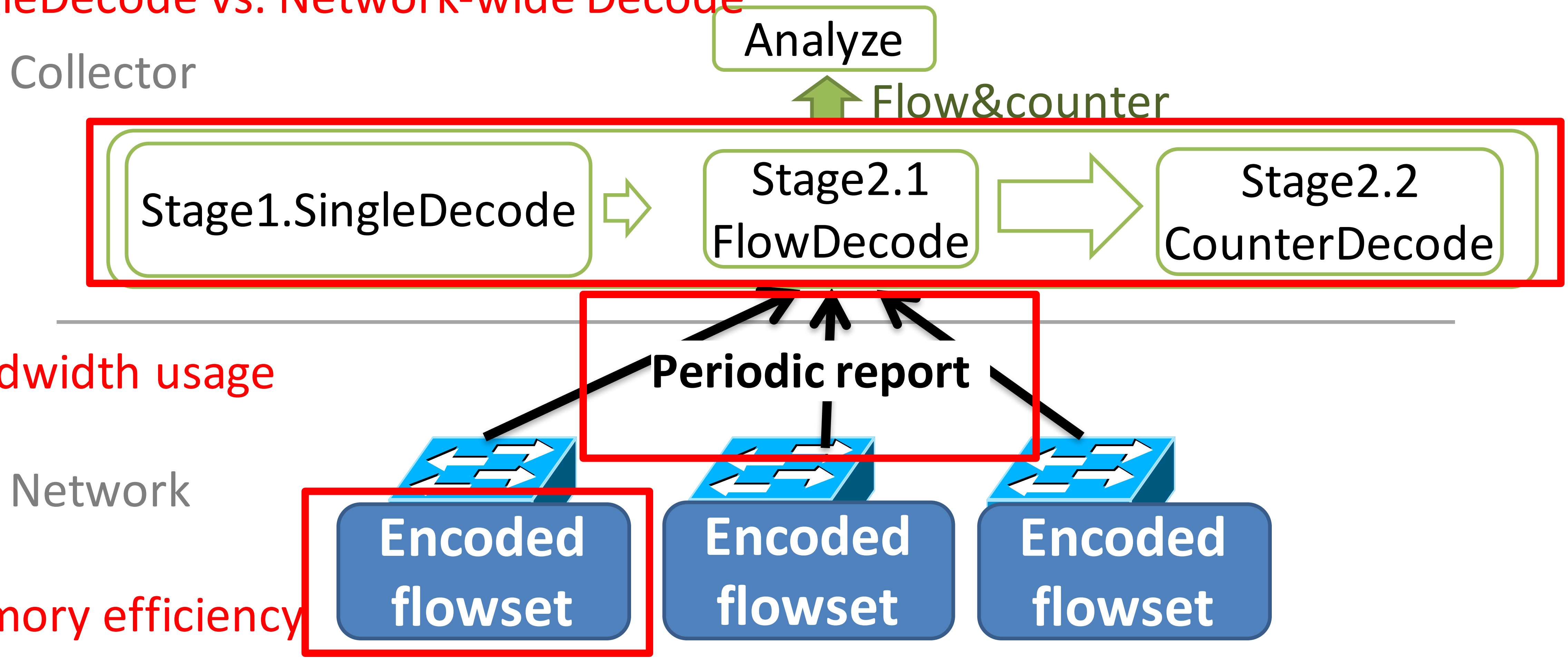


Evaluations

SingleDecode vs. Network-wide Decode

Bandwidth usage

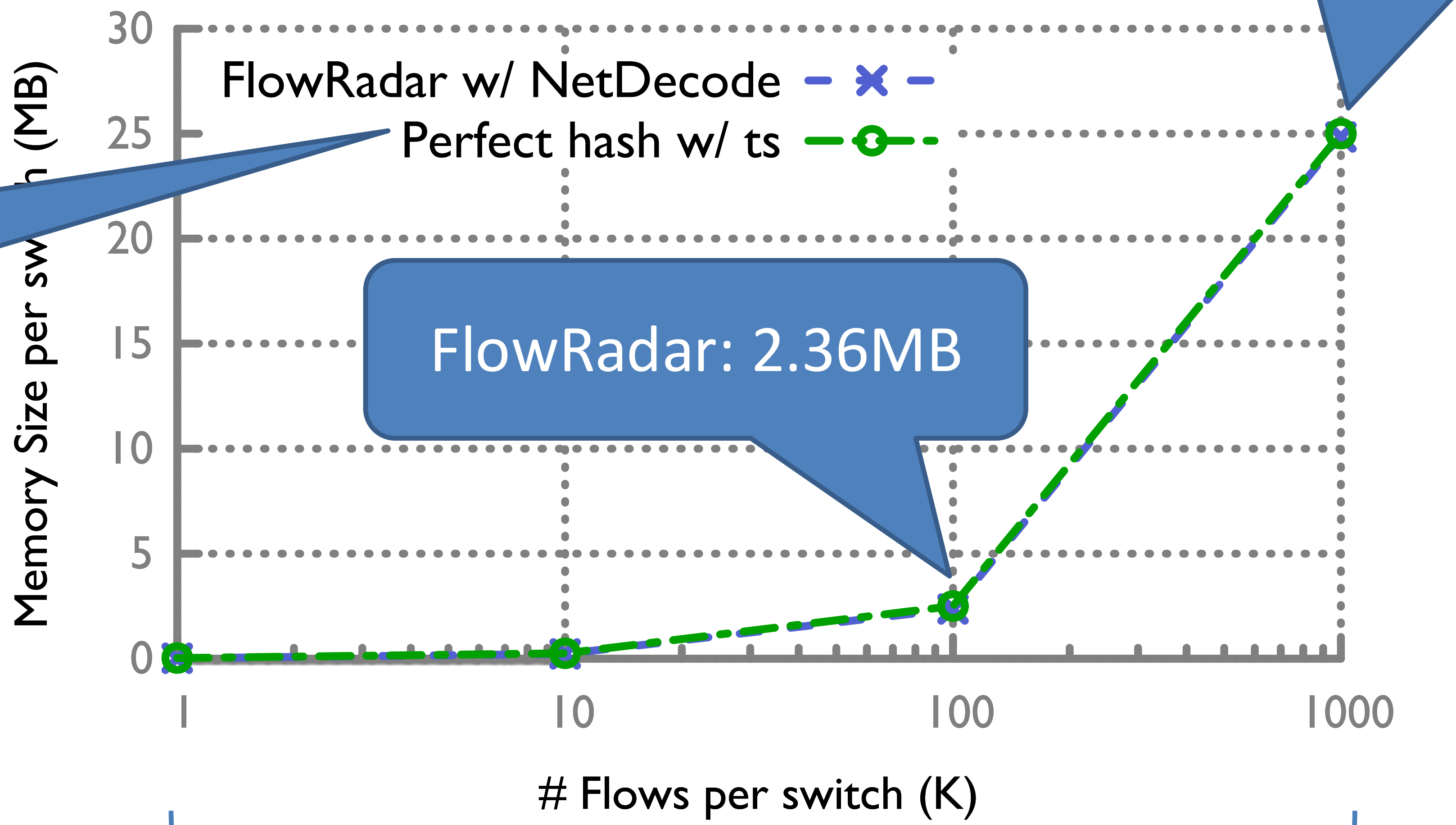
Memory efficiency



Evaluation

- Simulation of k=8 FatTree (80 switches, 128 hosts) in ns3
- Config the memory base on avg(#flow),
  - when burst of flows happens, use network-wide decode
- The worst case is all switches are pushed to max(#flow)
  - Traffic: each switch has same number of flows, and thus same memory
- Each switch reports the flowset every 10 ms.

Memory efficiency



FlowRadar: 24.8MB

#cell=#flow (Impractical)

FlowRadar: 2.36MB

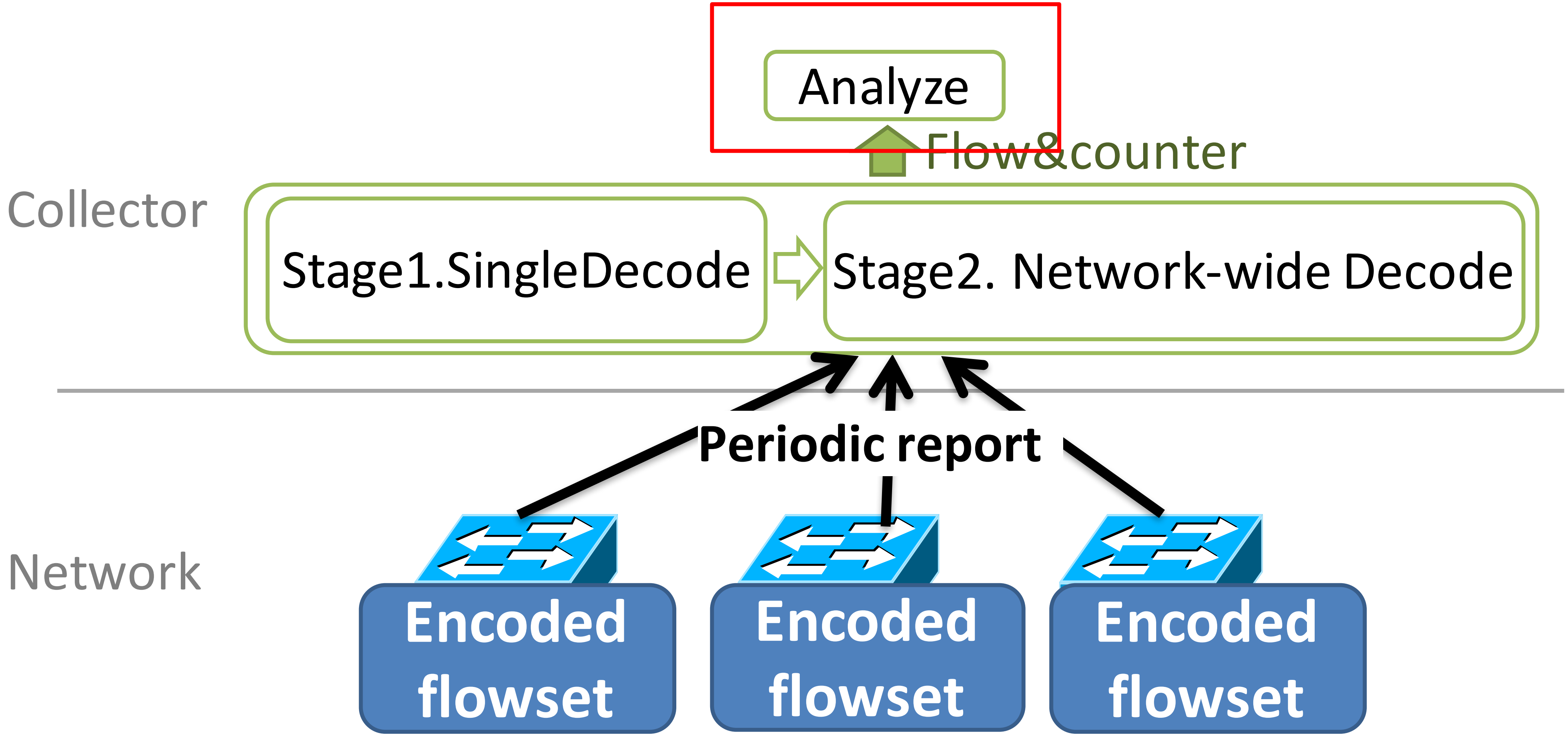
Log scale

Other results

- Bandwidth usage
  - Only 0.52% based on topology and traffic of Facebook data centers (sigcomm’15)
- NetDecode improvement over SingleDecode
  - SingleDecode 100K flow, which takes 10ms
  - NetDecode 26.8% more flows with the same memory,

    which takes around 3 sec

FlowRadar analyzer



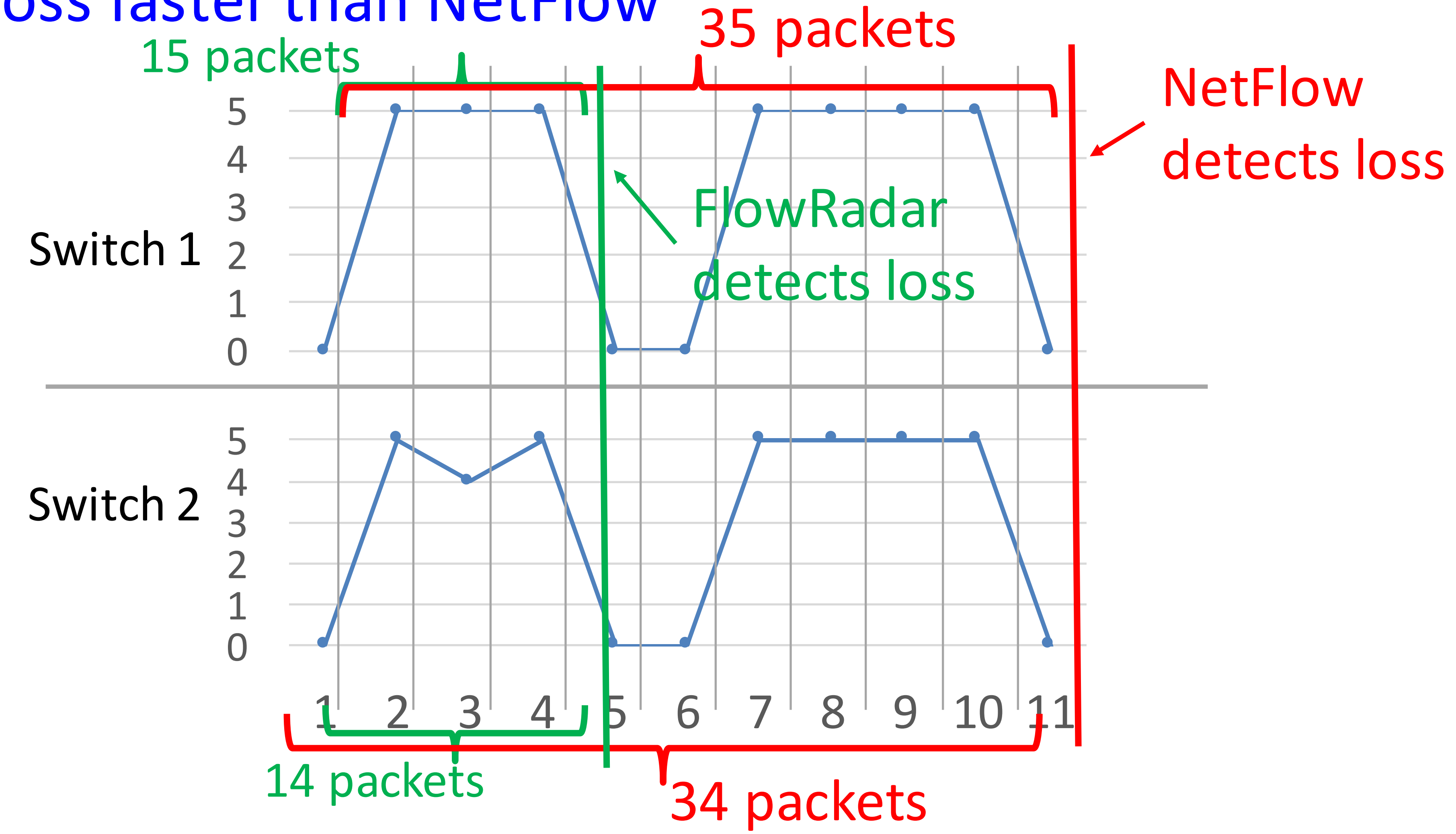
Analysis applications

- Flow coverage
  - Transient loop/blackhole
  - Error in match-action table
  - Fine-grained traffic analysis

- Temporal coverage
  - **Short time-scale per-flow loss rate**
  - ECMP load imbalance
  - Timely attack detection

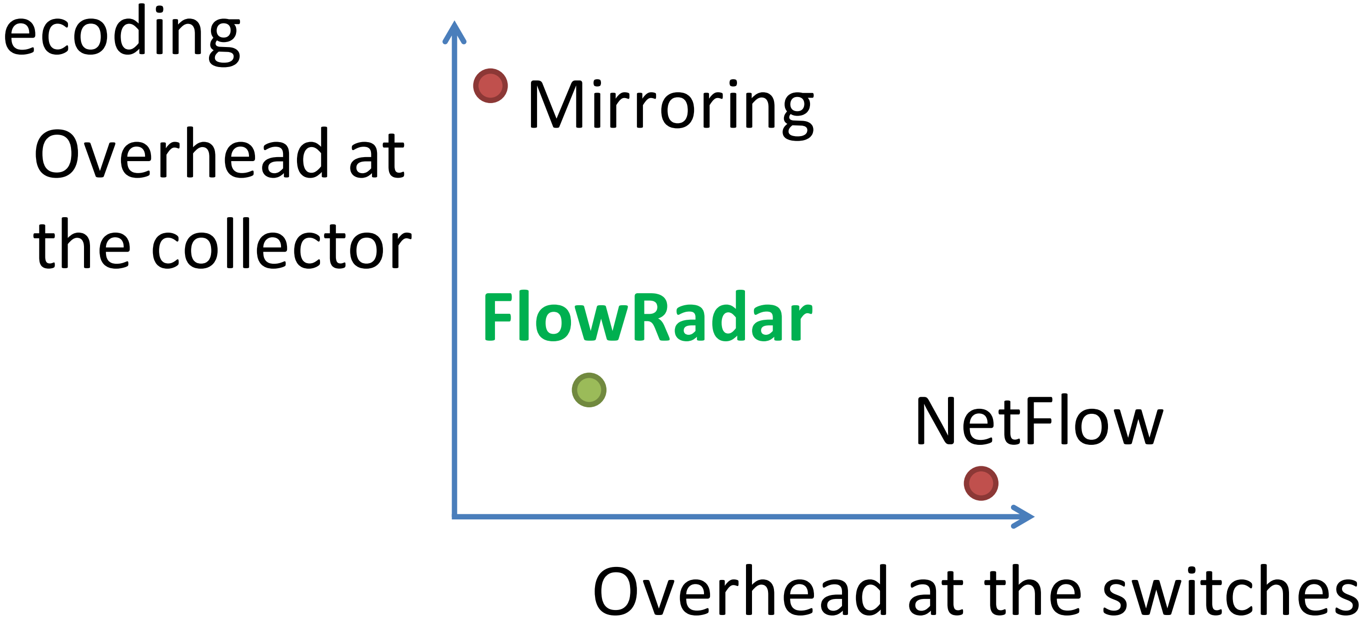
Per-flow loss map: better temporal coverage

- Detect loss faster than NetFlow



Conclusion

- Report counters for all flows in fine-grained time granularity
- Fully leverage the capability of both the switches and the collector
  - Switch: fixed per-packet processing time, memory-efficient
  - Collector: Network-wide decoding


Overhead at the collector
Mirroring
FlowRadar
NetFlow
Overhead at the switches