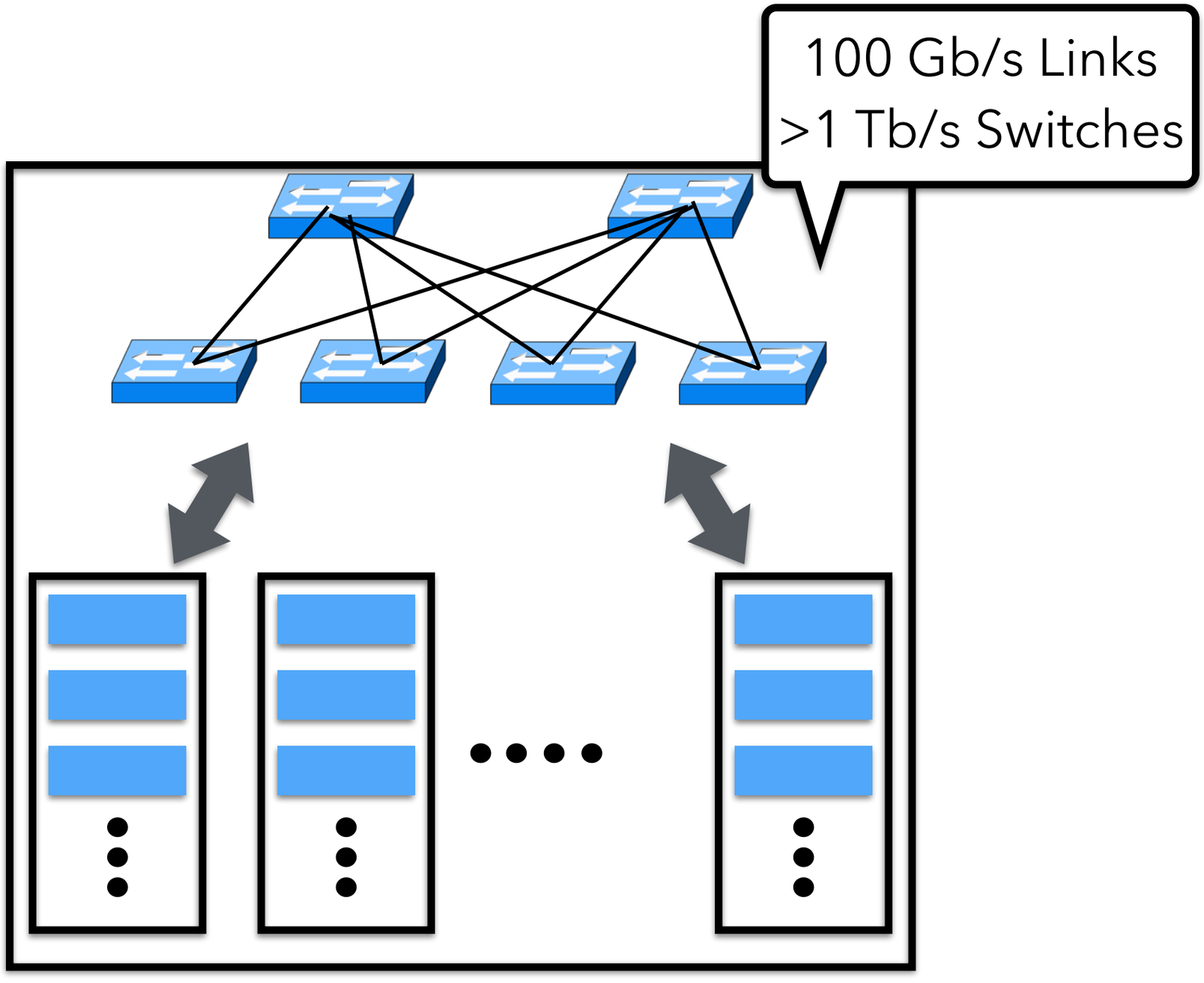
Scaling Hardware Accelerated Network Monitoring to Concurrent and Dynamic Queries with *Flow

John Sonchack, Oliver Michel,
Adam J. Aviv, Eric Keller, Jonathan M. Smith

Measuring High Speed Networks

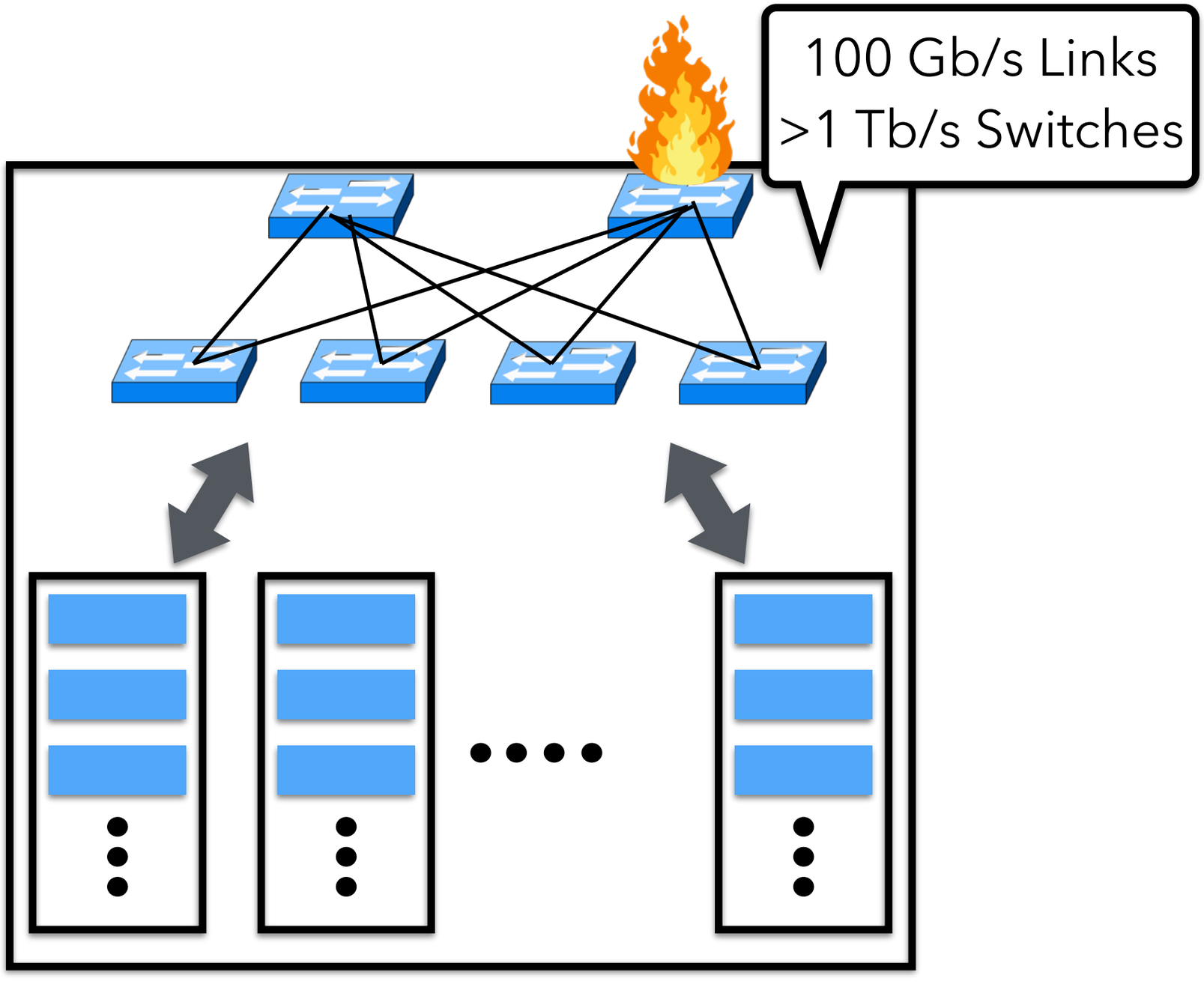


Measuring High Speed Networks

Debugging

Who is causing microbursts?

- *Queue Lengths*
- *Drop Counts*



Measuring High Speed Networks

Debugging

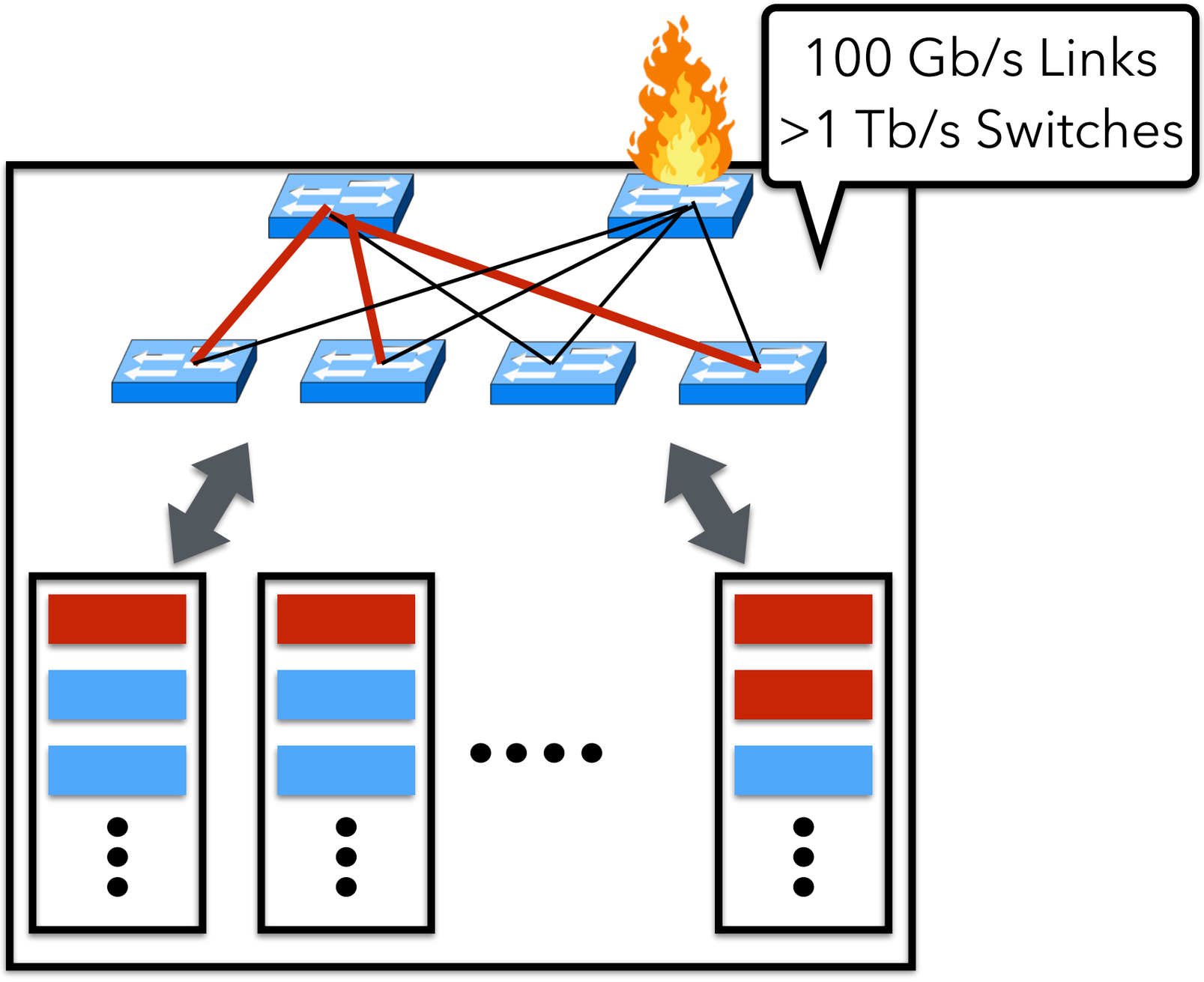
Who is causing microbursts?

- *Queue Lengths*
- *Drop Counts*

Traffic Engineering

Which flows are colliding?

- *Utilization*



Measuring High Speed Networks

Debugging

Who is causing microbursts?

- *Queue Lengths*
- *Drop Counts*

Traffic Engineering

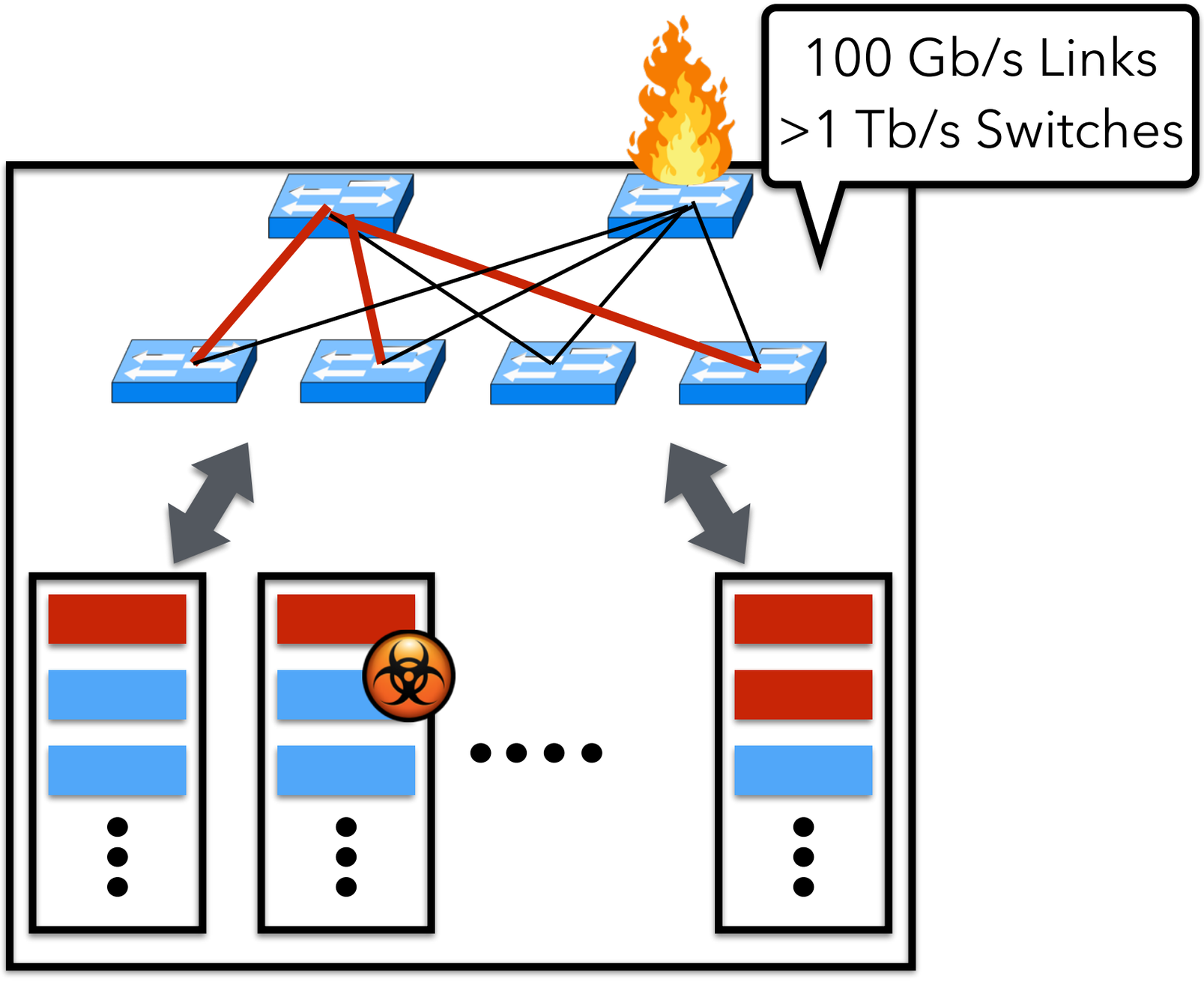
Which flows are colliding?

- *Utilization*

Security

Are any hosts compromised?

- *Packet timing statistics*



Measurement Challenges

Concurrent Applications High Packet Rates

Debugging

Who is causing microbursts?

- ***Queue Lengths***
- ***Drop Counts***

Traffic Engineering

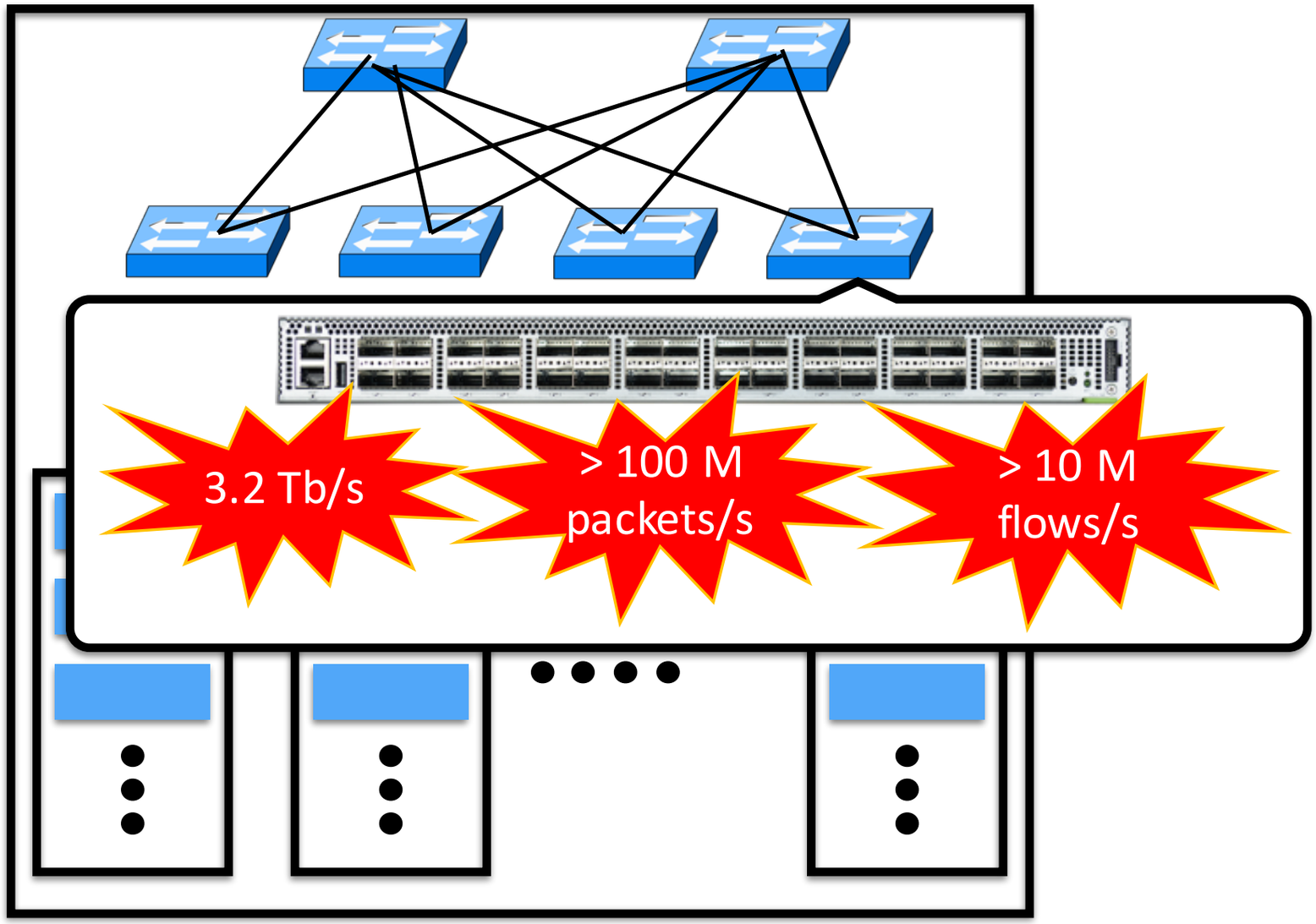
Which flows are colliding?

- ***Utilization***

Security

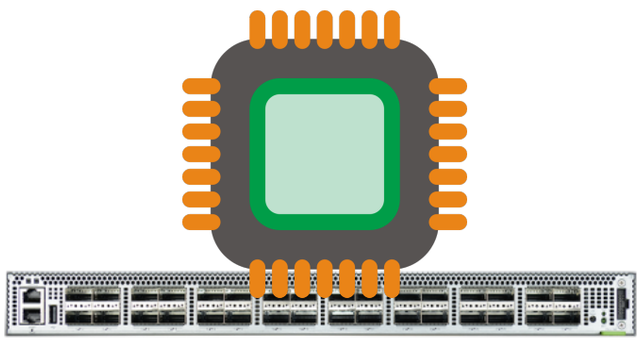
Are any hosts compromised?

- ***Packet timing statistics***

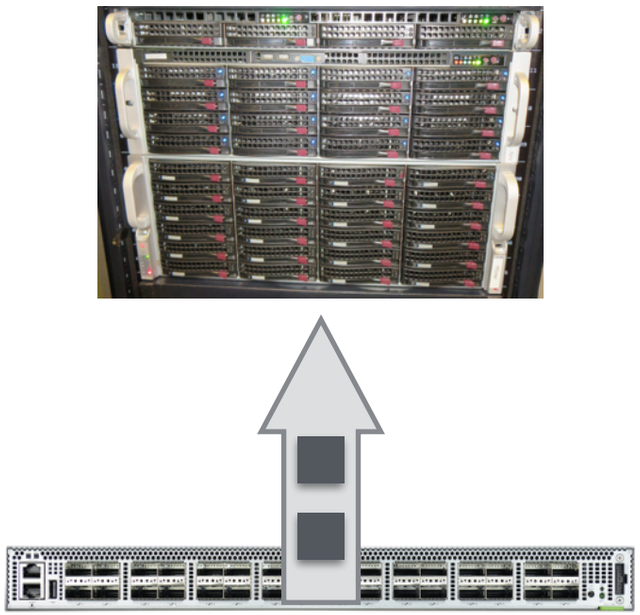


Prior Measurement Systems

Custom Hardware (e.g., NetFlow)



Processing Cluster



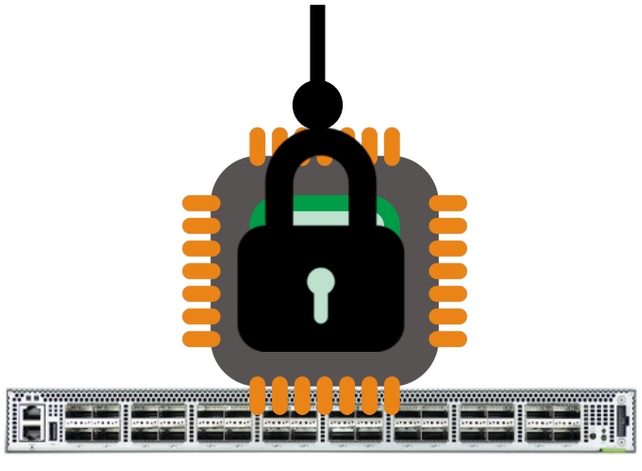
Prior Measurement Systems

Custom Hardware (e.g., NetFlow)

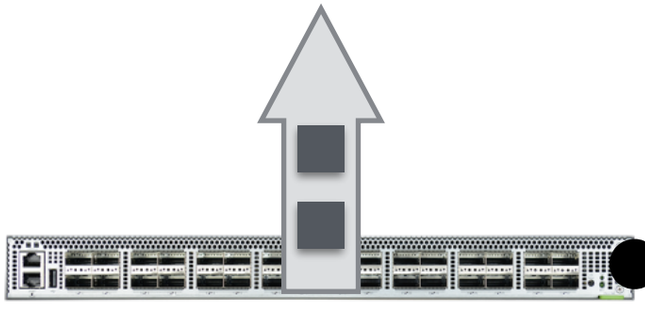
Processing Cluster

Flexibility

Applications can't define custom statistics



Efficiency



Measurement Tput:
< 1 M *packets / second per core*

Network Tput:
> 100 M *packets / second per switch*

Prior Measurement Systems

Custom Hardware (e.g., NetFlow)

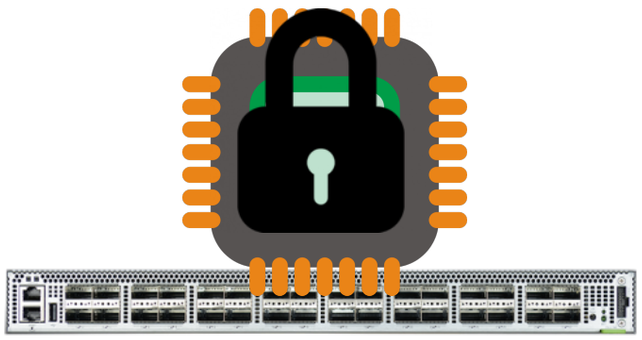
Processing Cluster

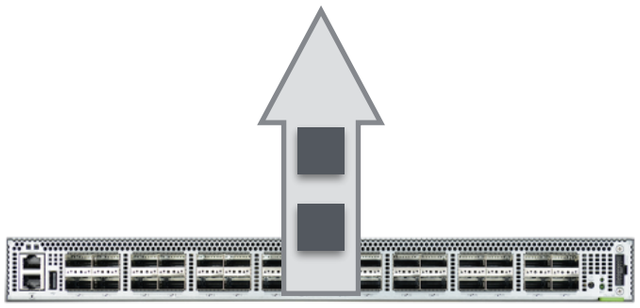
Reconfigurable Switch ASICs

[Marple SIGCOMM 17, TurboFlow EuroSys 18, Sonata SIGCOMM 18]

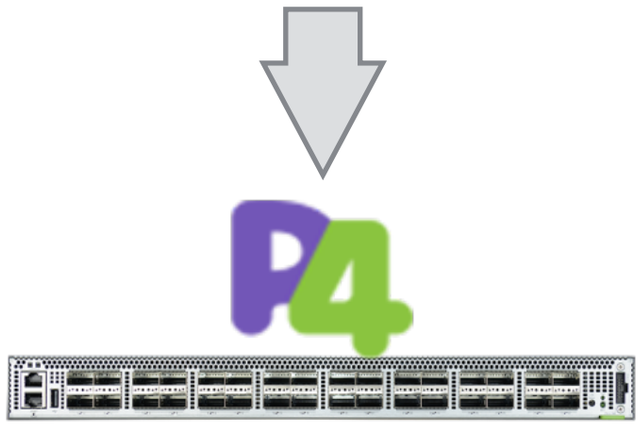
Flexibility

Efficiency





```
SELECT packet.length
GROUP BY tcp flow
AGGREGATE average
```



Prior Measurement Systems

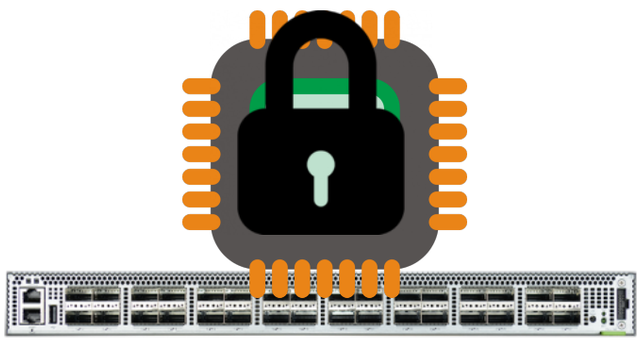
Custom Hardware (e.g., NetFlow)

Processing Cluster

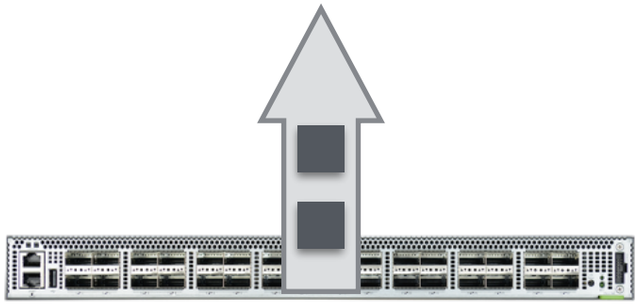
Reconfigurable Switch ASICs

[Marple SIGCOMM 17, TurboFlow EuroSys 18, Sonata SIGCOMM 18]

Flexibility

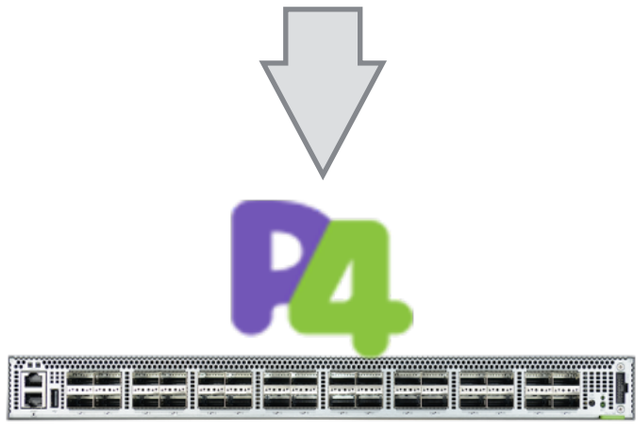


Efficiency



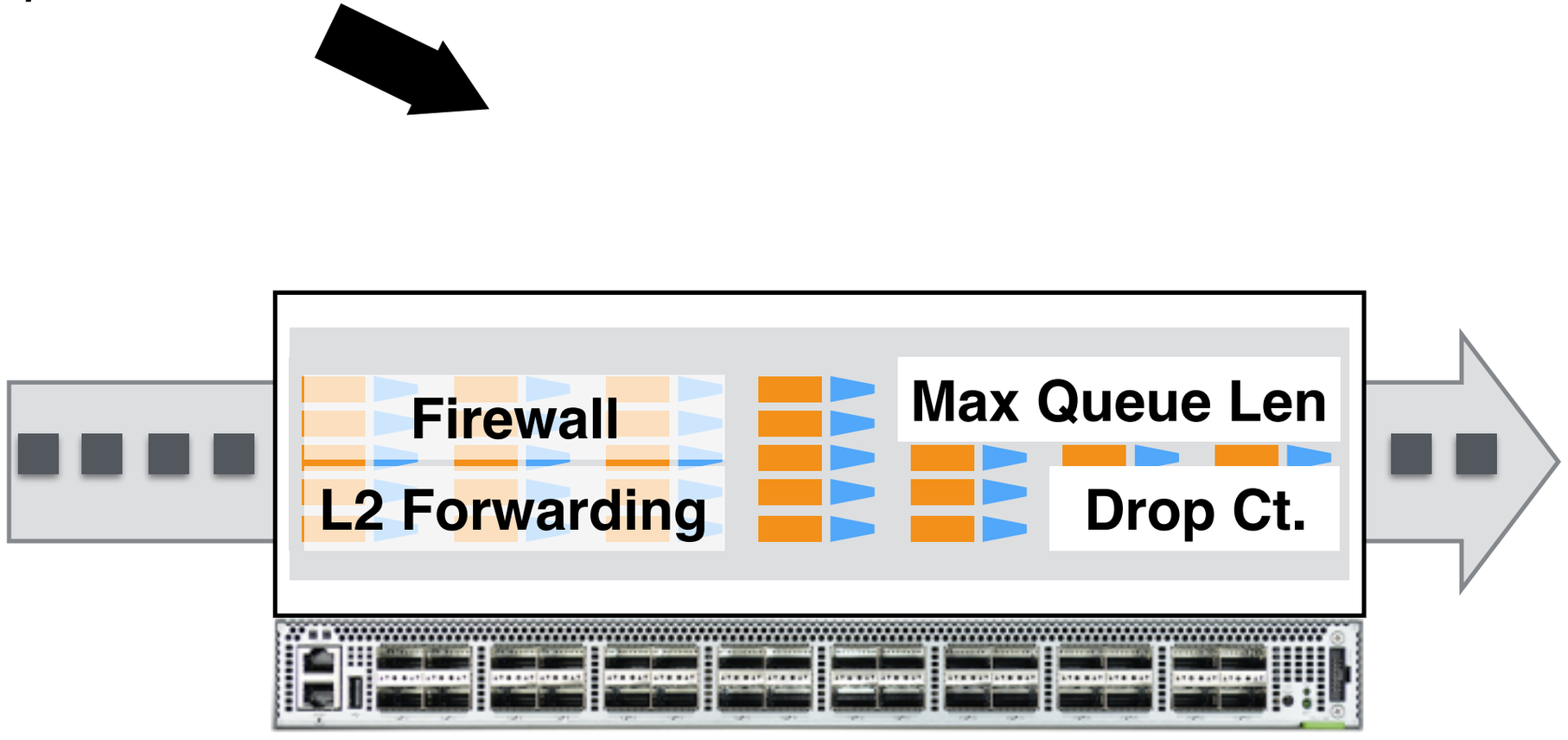
Concurrency

```
SELECT packet.length
GROUP BY tcp flow
AGGREGATE average
```



Concurrency Challenges with Reconfigurable ASICs

- *Max Queue Length*
- *Total Drop Count*

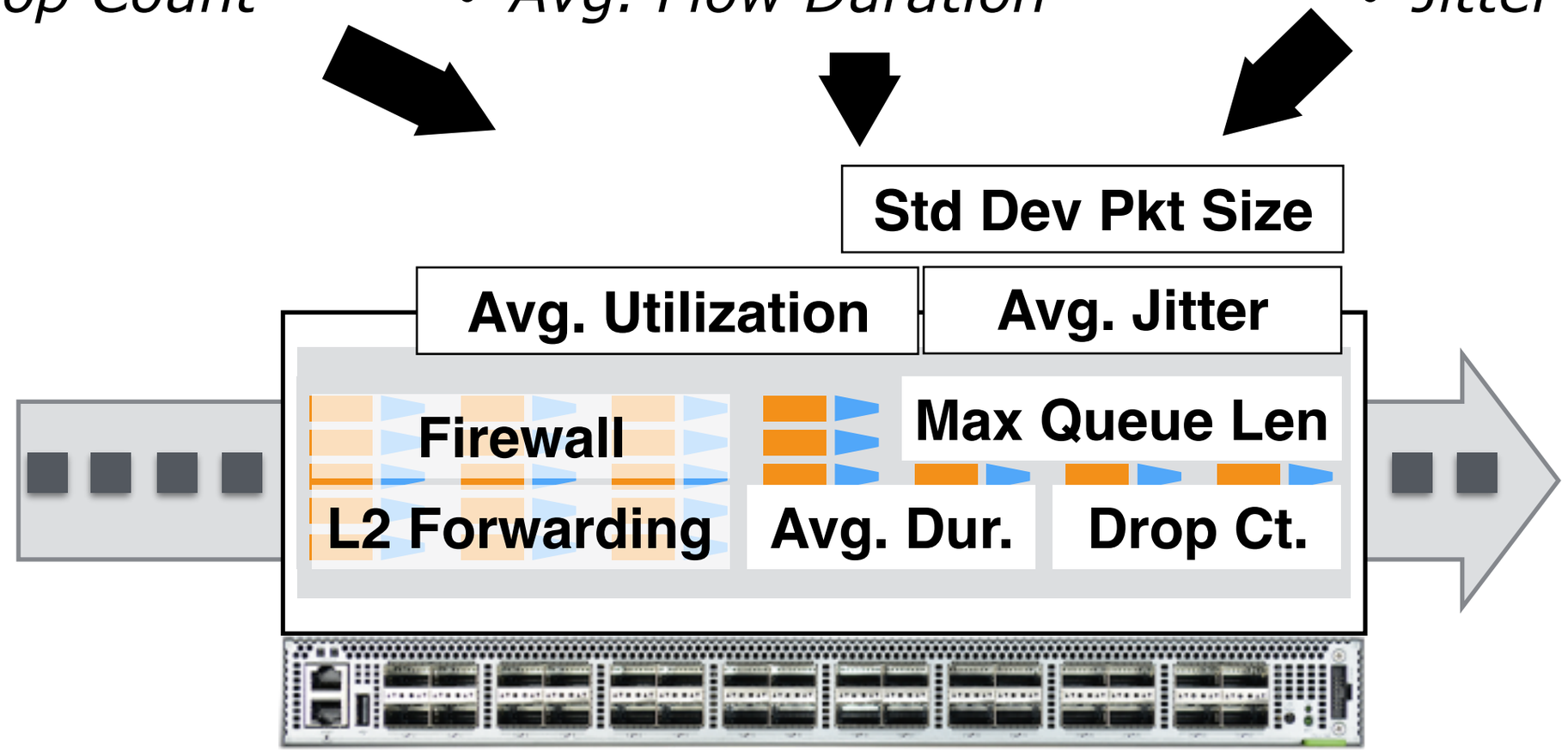


Concurrency Challenges with Reconfigurable ASICs

- *Max Queue Length*
- *Total Drop Count*

- *Std Dev. Packet Size*
- *Avg. Flow Duration*

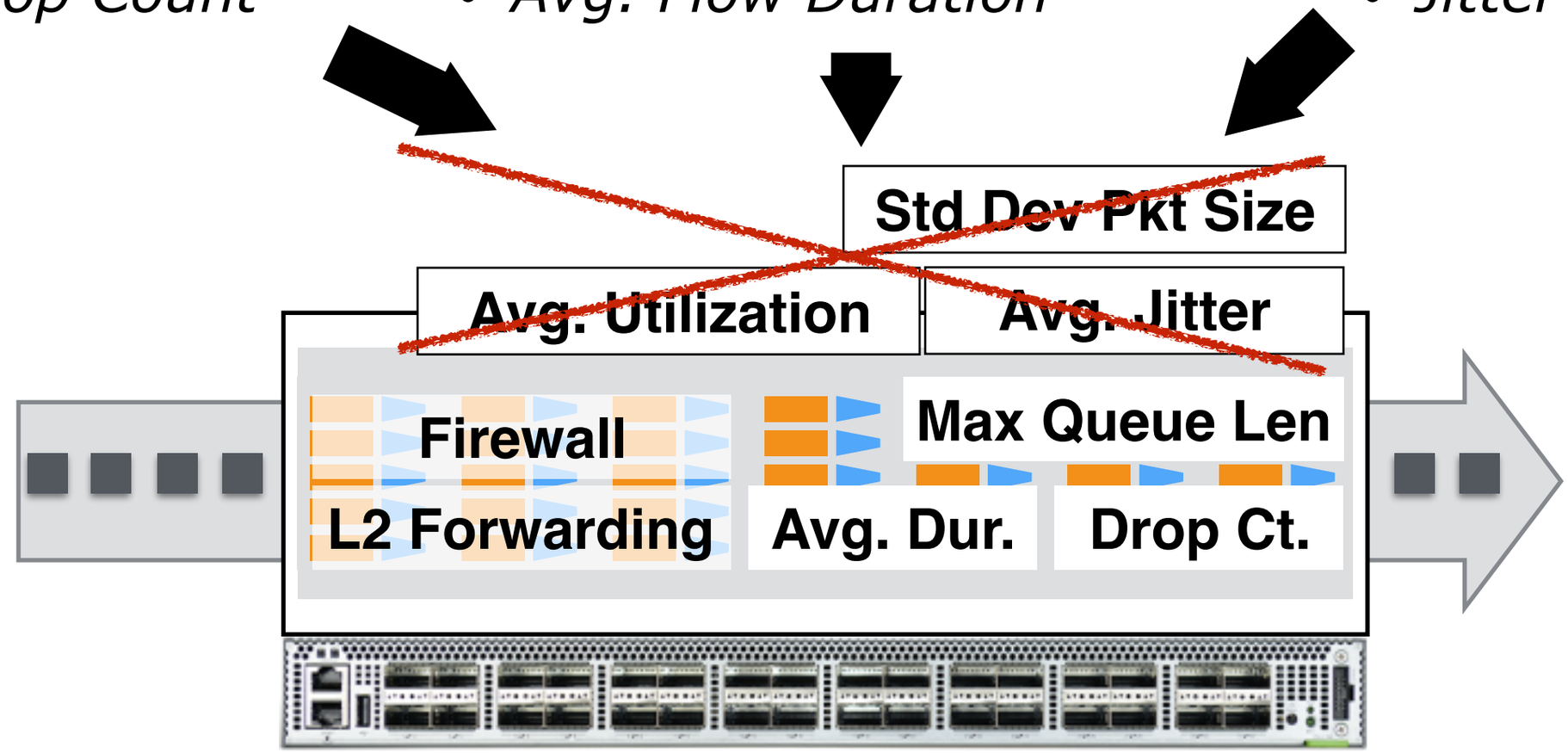
- *Avg. Utilization*
- *Jitter*



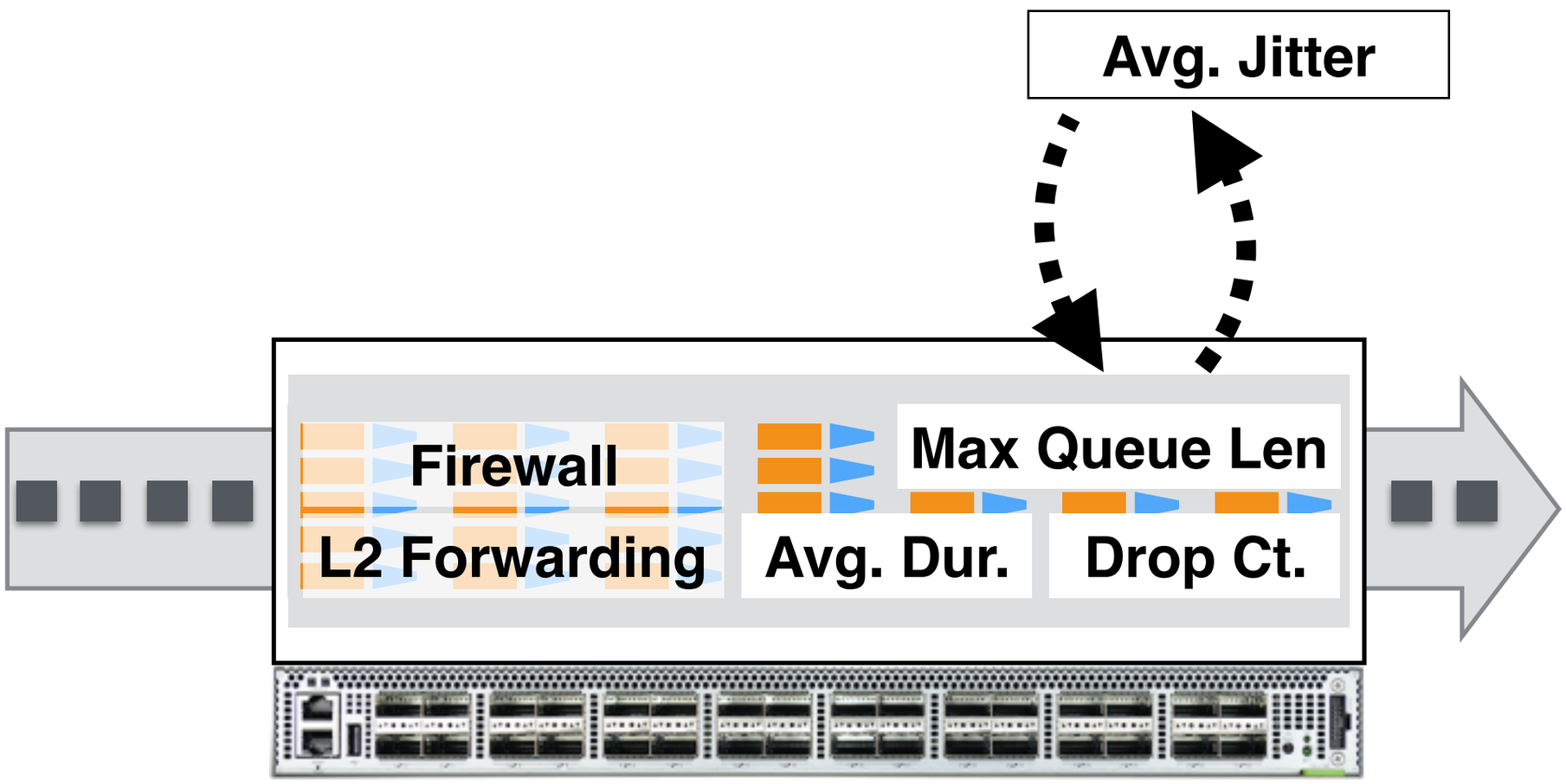
Concurrency Challenges with Reconfigurable ASICs

Not Enough room in the datapath for all the queries.

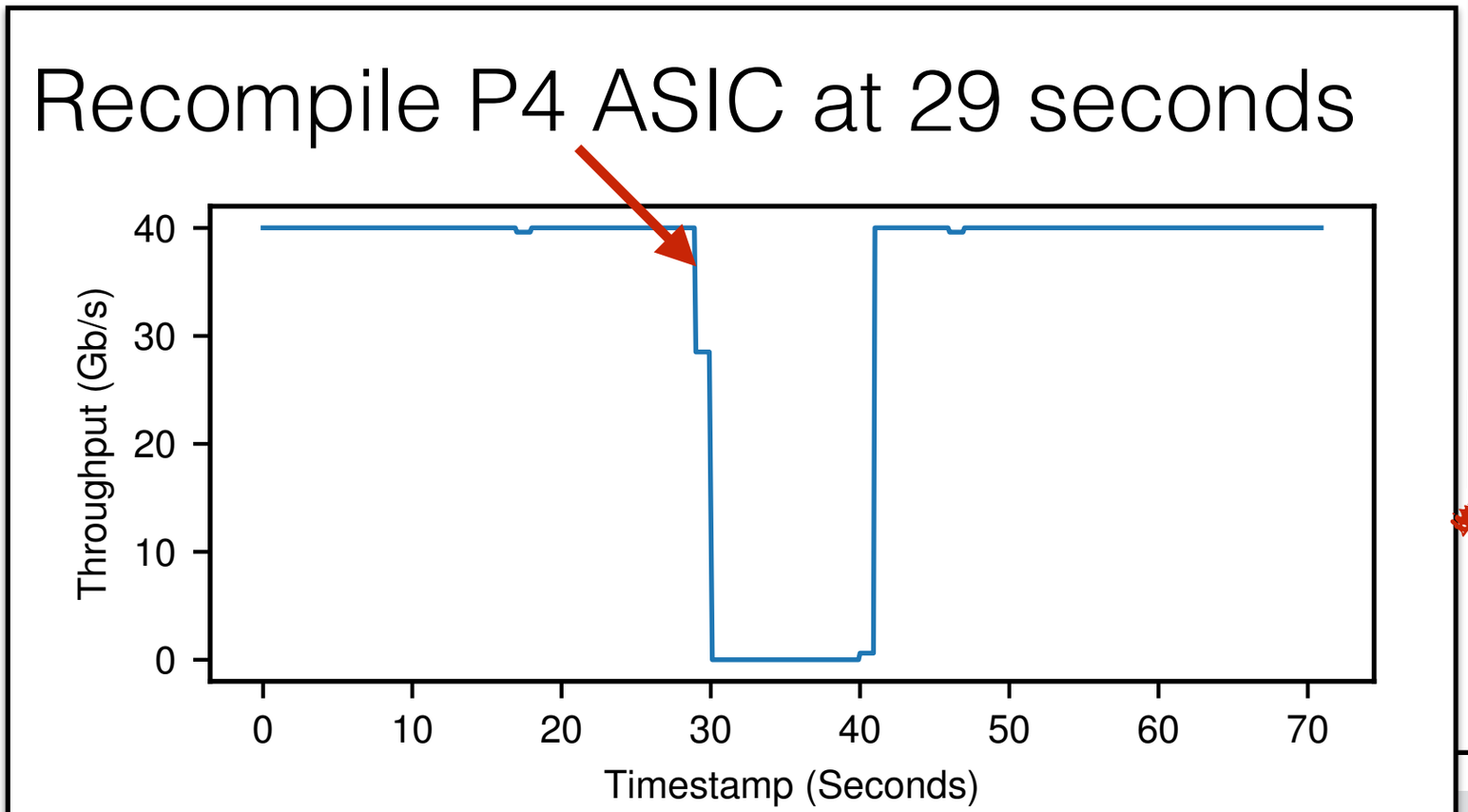
- *Max Queue Length*
- *Total Drop Count*
- *Std Dev. Packet Size*
- *Avg. Flow Duration*
- *Avg. Utilization*
- *Jitter*



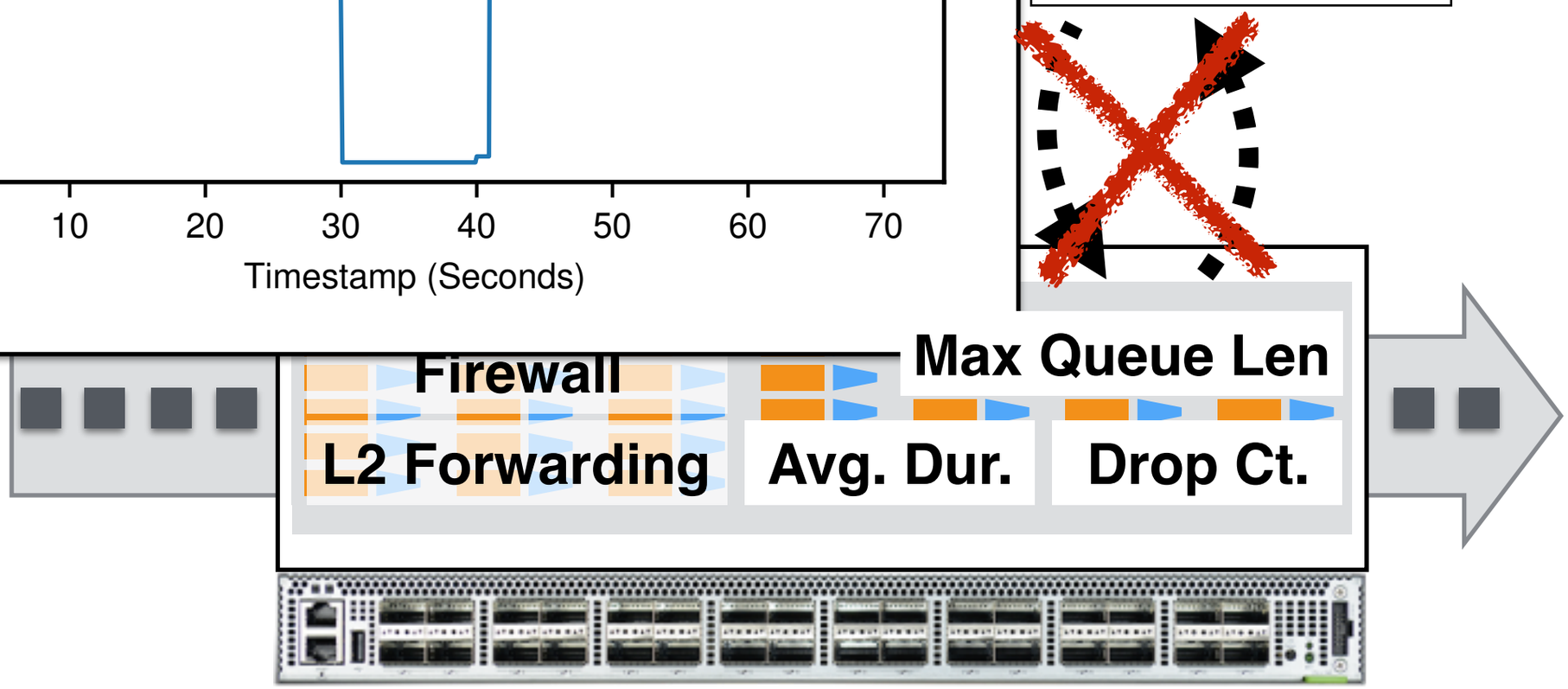
Concurrency Challenges with Reconfigurable ASICs



Concurrency Challenges with Reconfigurable ASICs



Recompiling the datapath disrupts measurement **and forwarding.**



*Flow: Efficiency *and* Concurrency?

Question: can we leverage reconfigurable ASICs **for concurrent monitoring**?

*Flow: Efficiency *and* Concurrency?

Question: can we leverage reconfigurable ASICs **for concurrent monitoring**?

Insight: concurrency challenges are caused by trying to do ***too much in the ASIC.***

The Main Idea

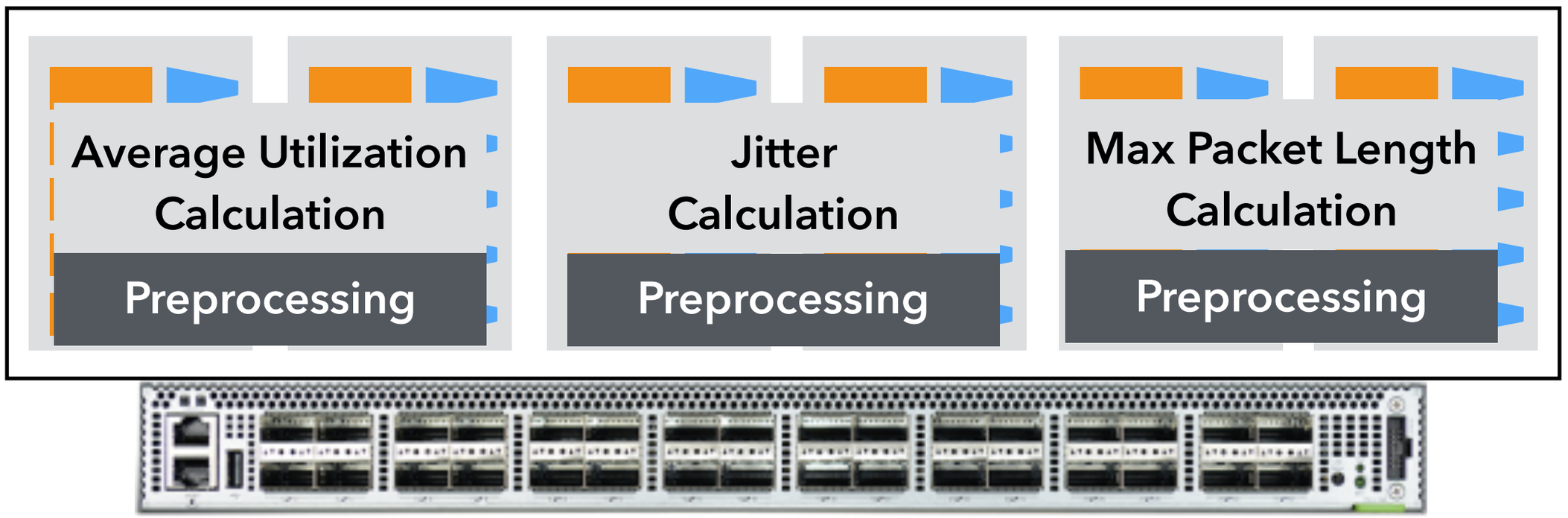
The Main Idea

Preprocessing

```
SELECT packet.length
GROUP BY tcp flow
```

Calculation

```
AGGREGATE max
```

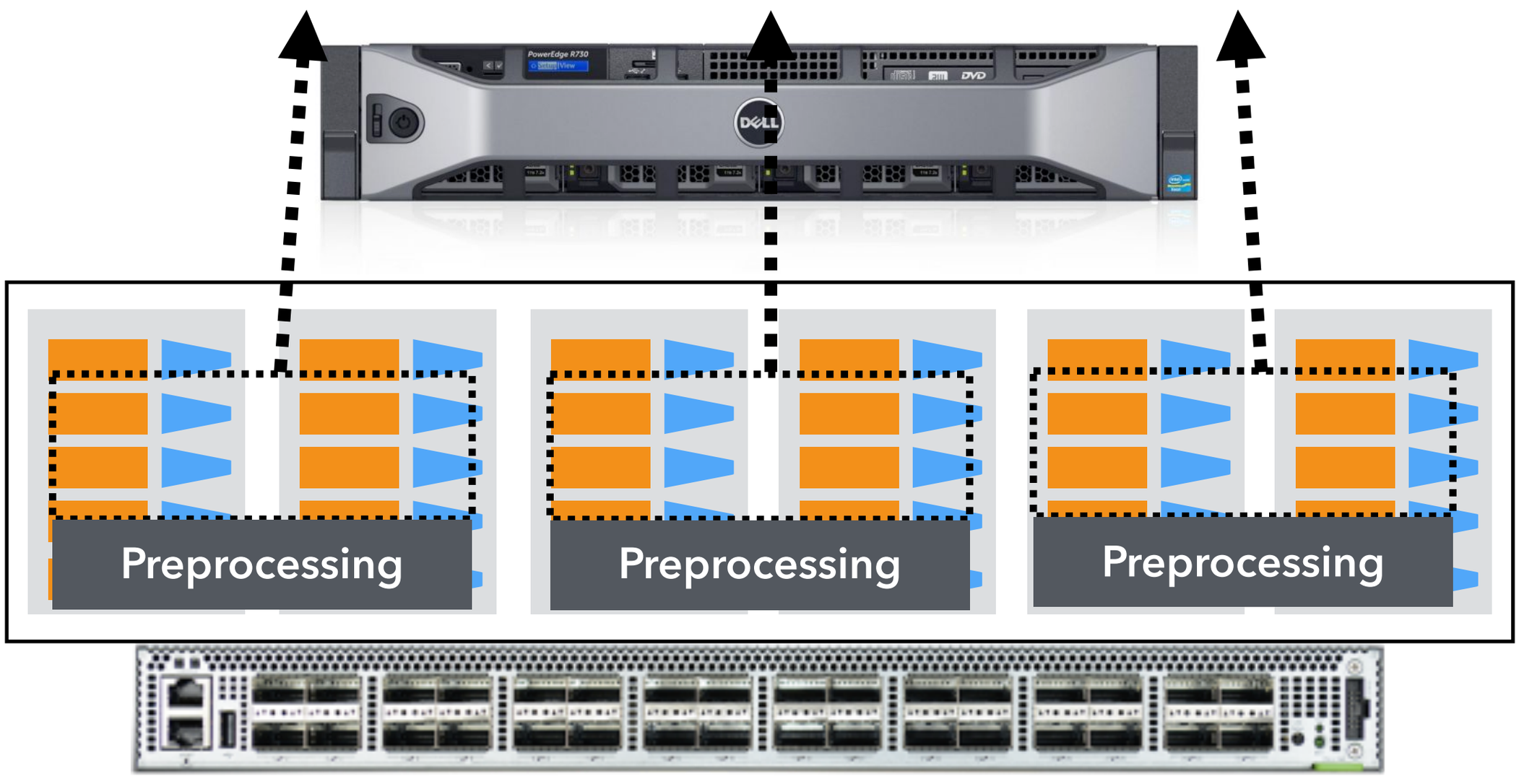


The Main Idea

Average Utilization Calculation

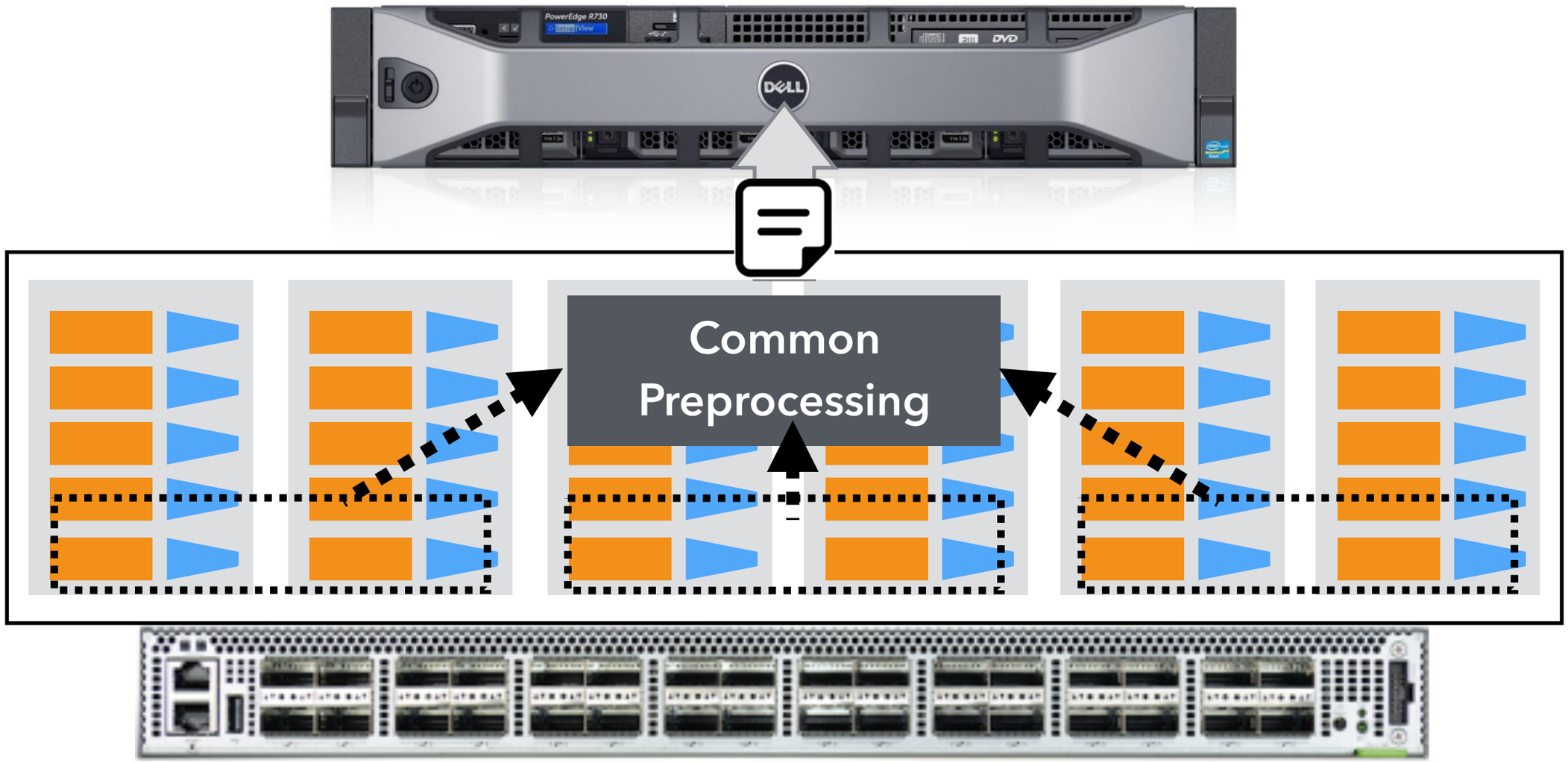
Jitter Calculation

Max Packet Length Calculation



The Main Idea

Average Utilization Calculation	Jitter Calculation	Max Packet Length Calculation



The Main Idea

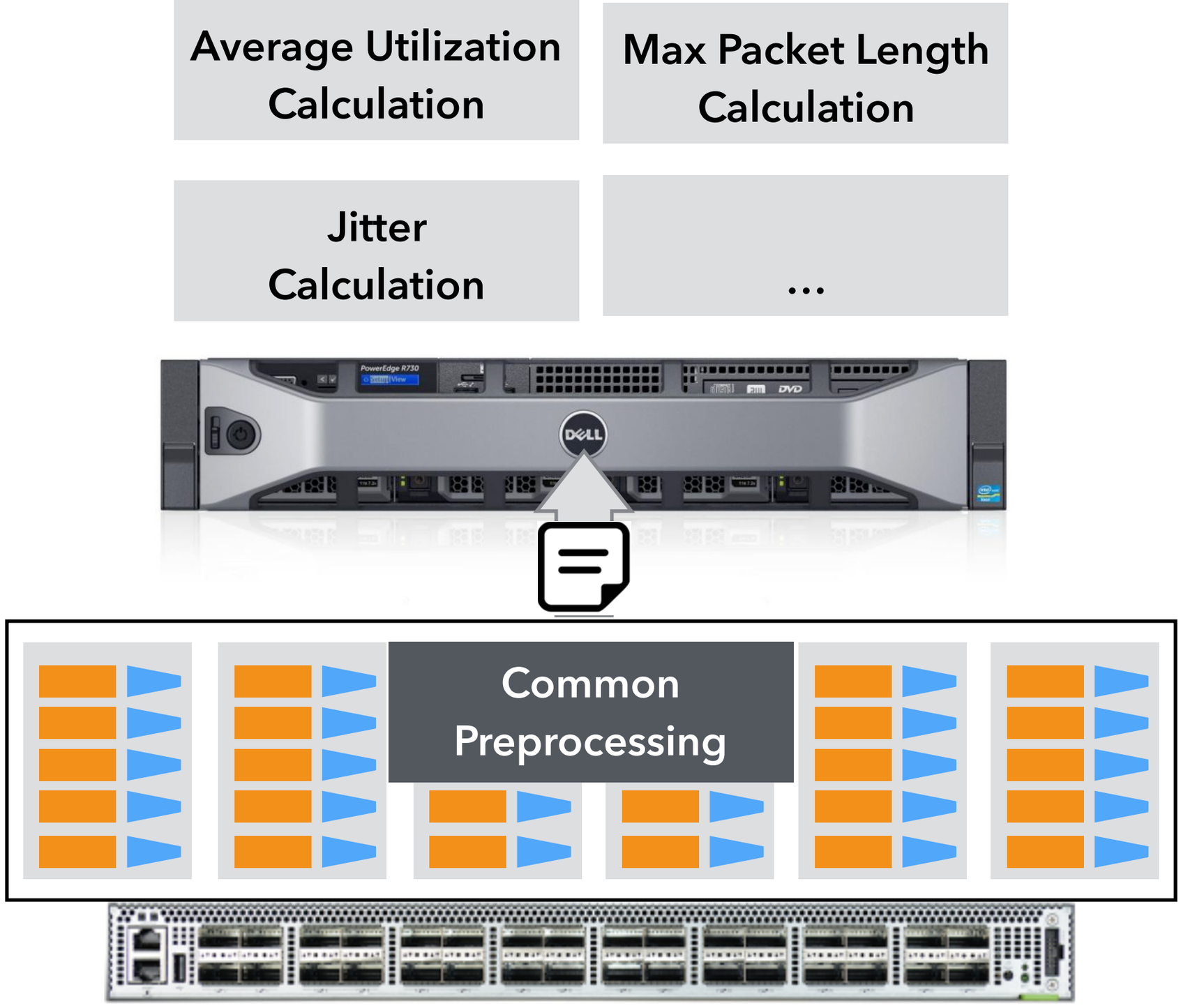
Concurrency

Average Utilization Calculation

Max Packet Length Calculation

Jitter Calculation

…



The Main Idea

✓ **Concurrency**

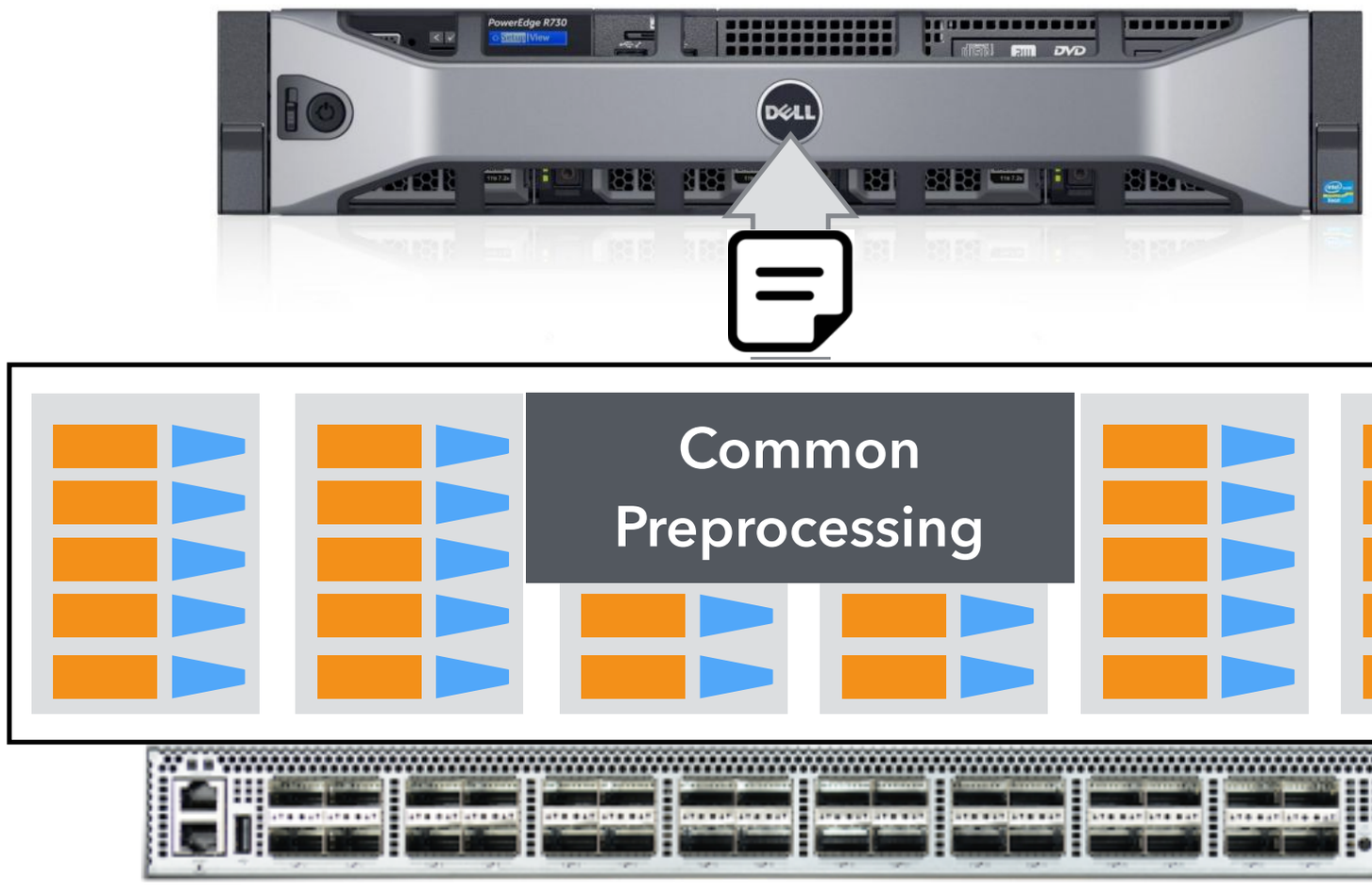
? **Efficiency**

Average Utilization Calculation

Max Packet Length Calculation

Jitter Calculation

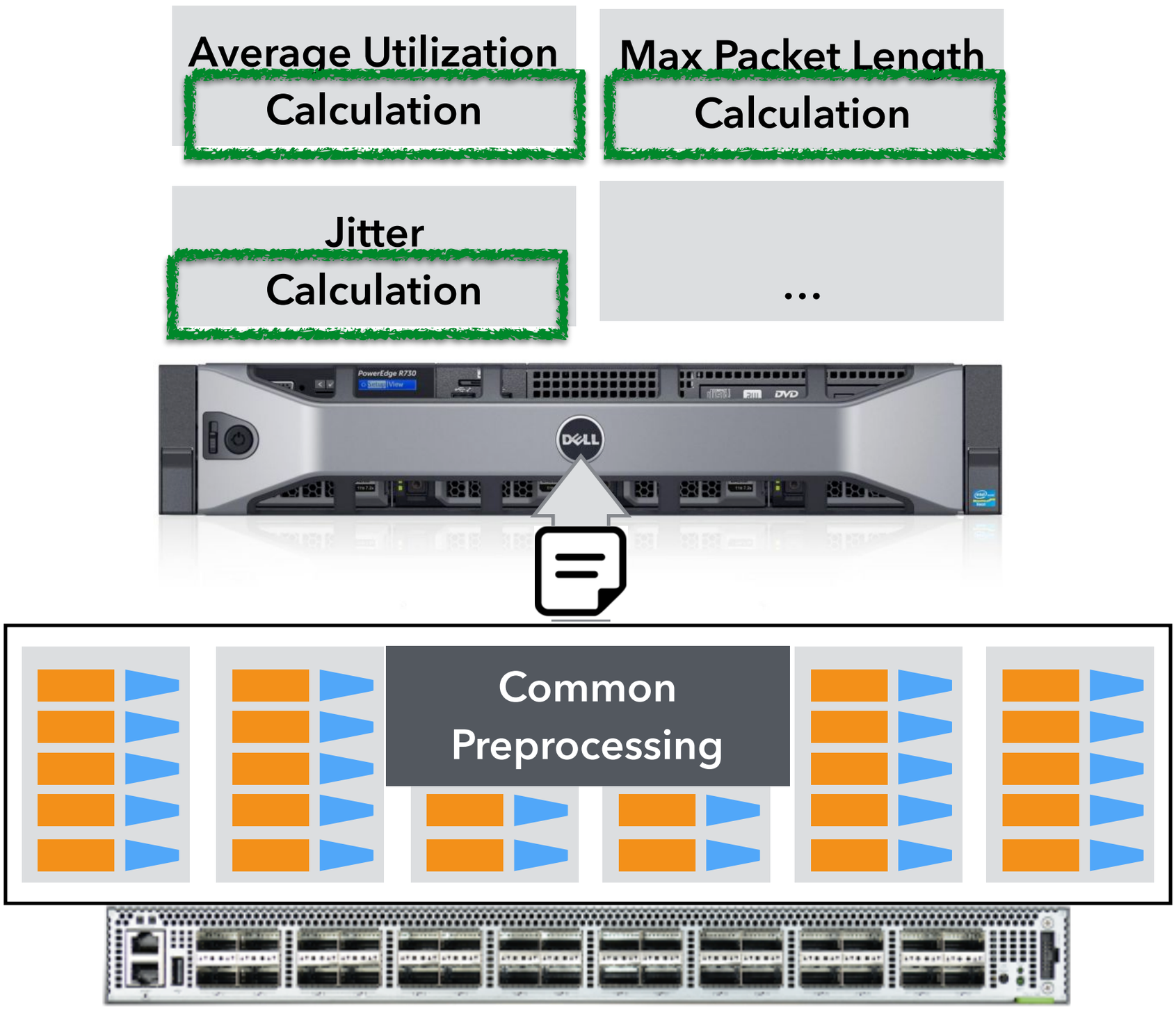
…



The Main Idea

✓ **Concurrency**

? **Efficiency**



The Main Idea

Average Utilization Calculation

Max Packet Length Calculation

...

POPULAR SCIENCE

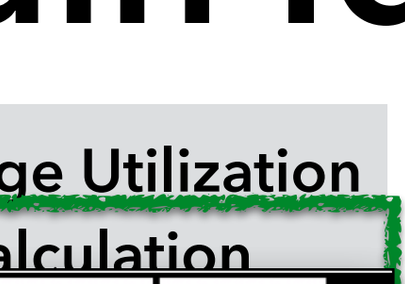
WANT MORE?

TECHNOLOGY

Intel's new chip puts a teraflop in your desktop. Here's what that means

It's as fast as a turn-of-the-century supercomputer.

By Rob Verger *June 1, 2017*



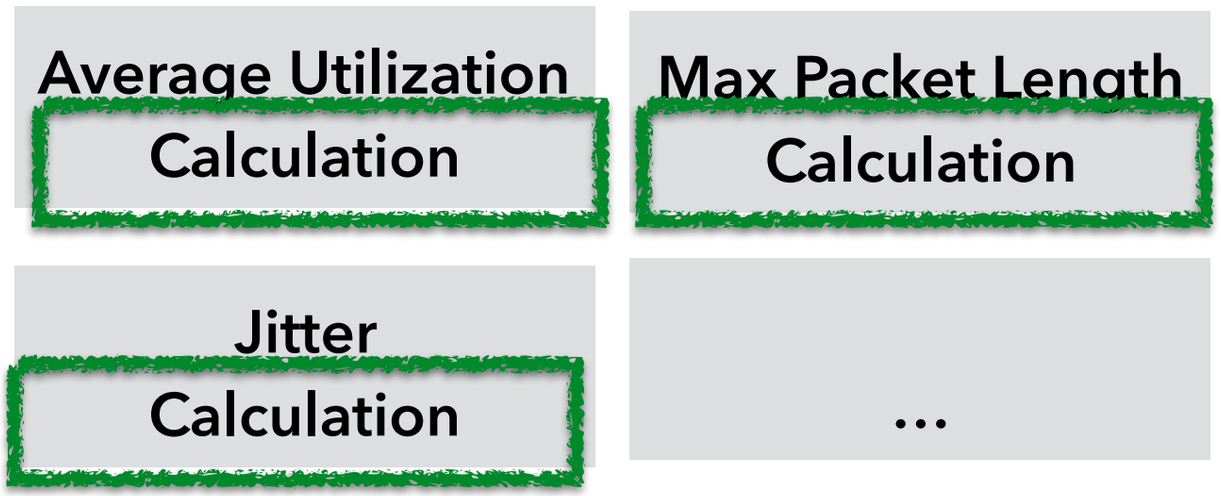
Reconfigure your RTL

www.flex-logix.com

mon
cessing

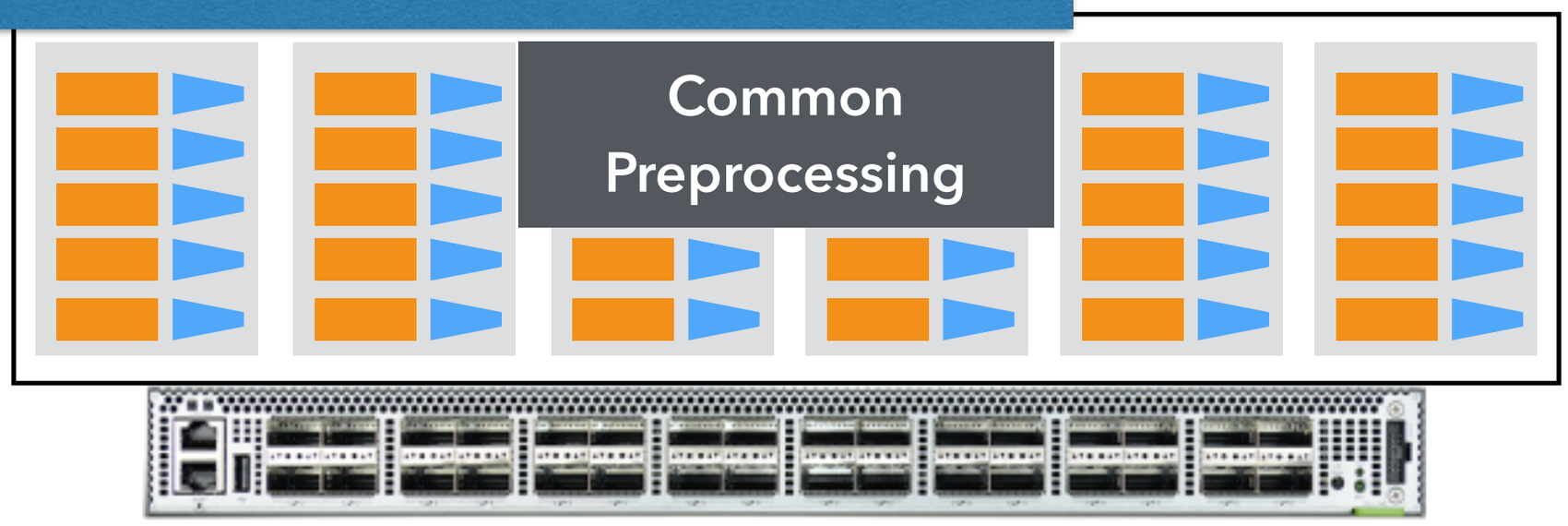
The Main Idea

Concurrency



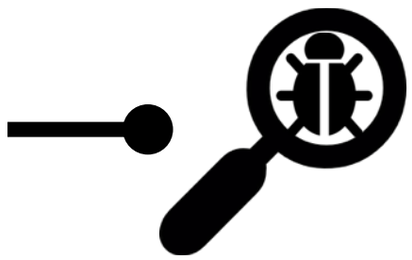
Efficiency

Measure **every packet** in
> 1 Tb/s traffic with 1 server.



The Main Idea

Application specific calculation



Concurrency

*Flow Agent

Preprocessing

Efficiency

*Flow Cache

Outline

- Motivation
- **Design**
- Implementation
- Evaluation

*Flow Design

```
SELECT ip length
GROUP BY tcp flow
AGGREGATE sum
```

1. Decoupling calculation
2. Generalize selection & grouping

Common Preprocessing

App. Specific Calculation

Decoupling Calculation

```
SELECT ip length
GROUP BY tcp flow
AGGREGATE sum
```

Key: A -> B
Length: 200

Flow Key	Length Sum
A -> B	1000
C -> D	250
E -> F	9416
…	…

Preprocessing

App. Specific Calculation

Decoupling Calculation

```
SELECT ip length
GROUP BY tcp flow
AGGREGATE sum
```

Grouped Packet Vector (GPV):

$$\{flowKey : \overrightarrow{packetLength}\}$$

Flow Key	Pkt Len		
A -> B	500	500	200
C -> D	250		
E -> F	20	100	…
…	…		

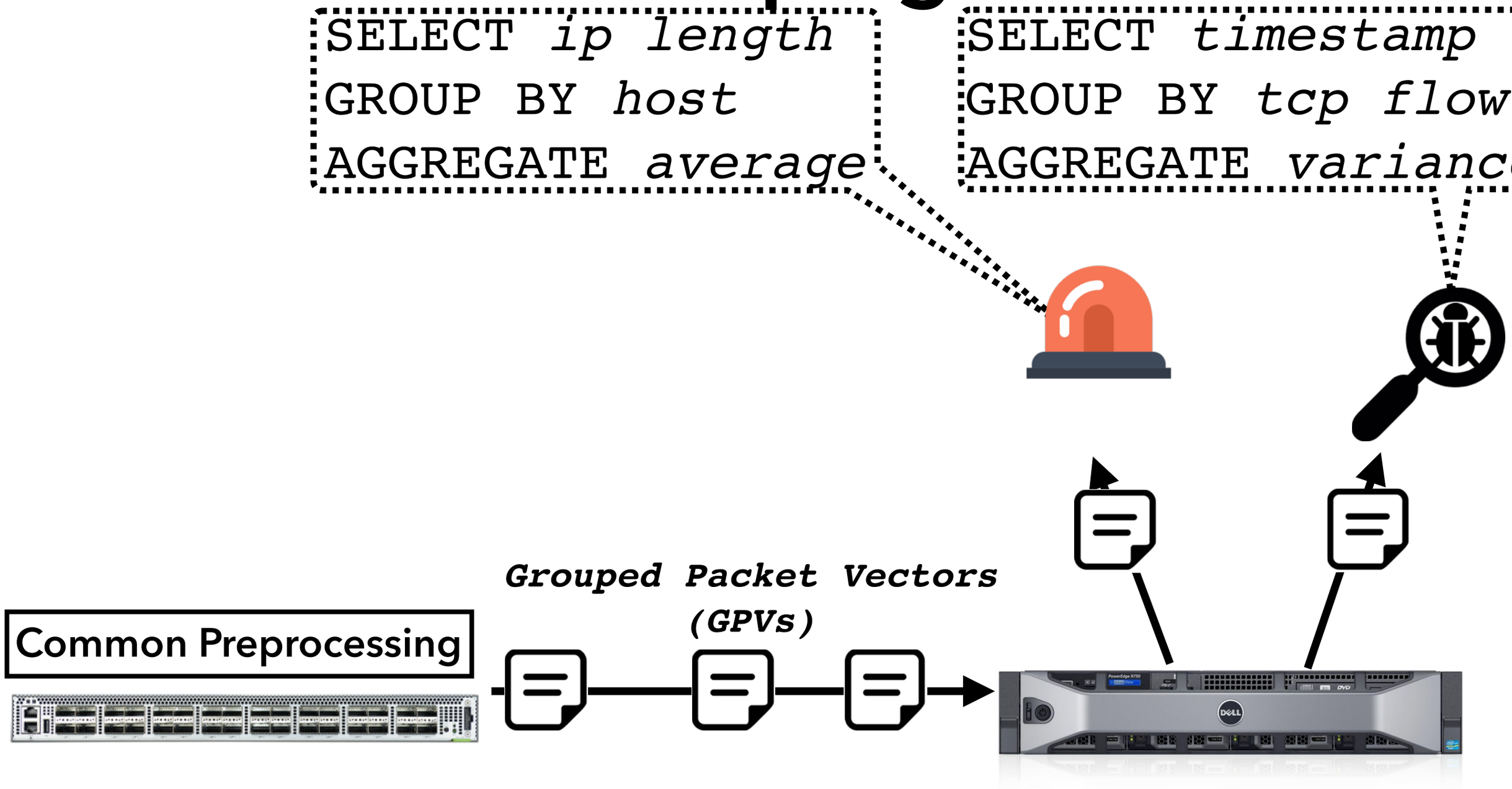
Length Sum
1200
250
9416
…

Key: A -> B
Length: 200

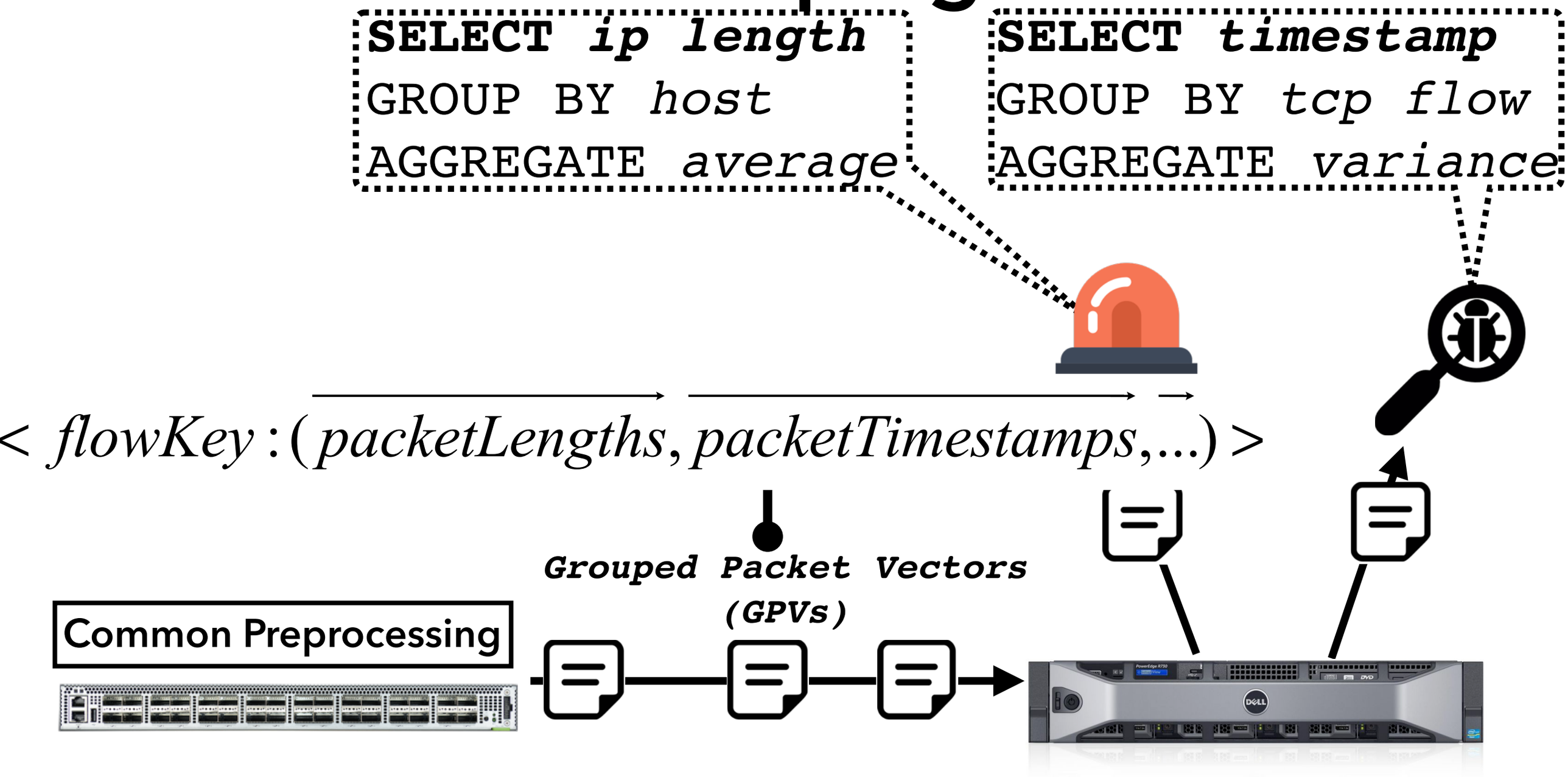
Preprocessing

App. Specific Calculation

Generalizing Selection and Grouping



Generalizing Selection and Grouping



Generalizing Selection and Grouping

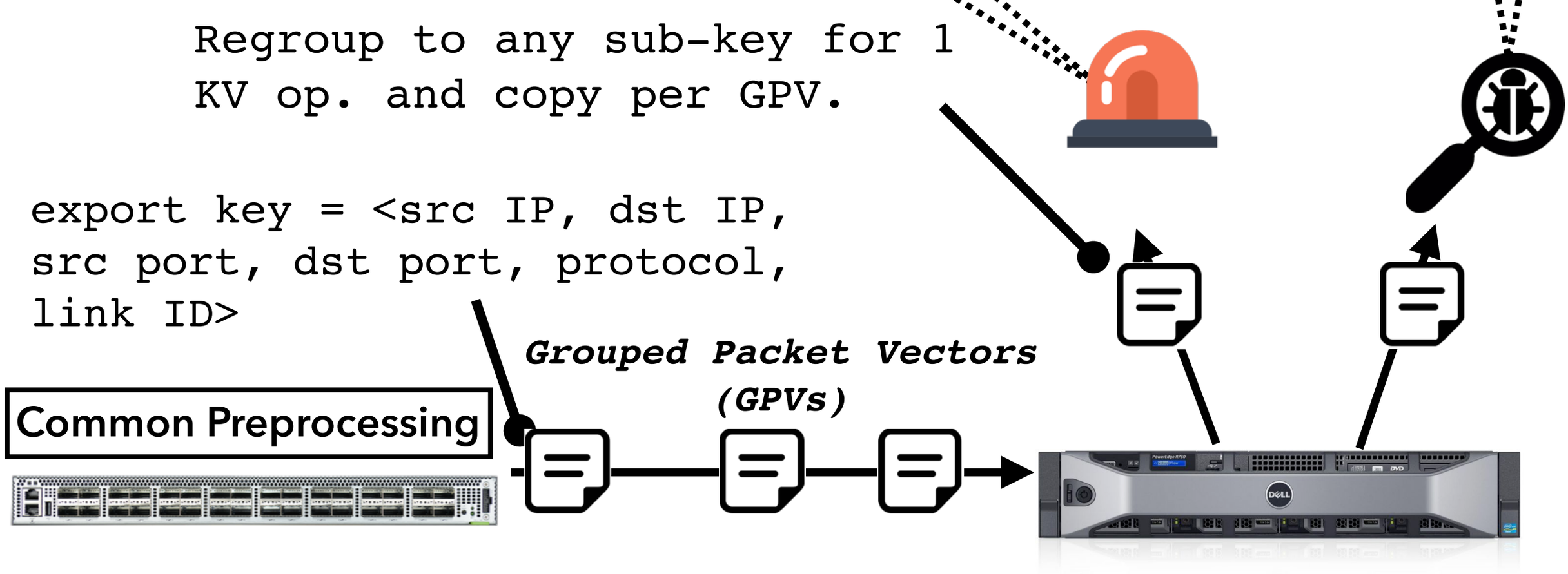
```
SELECT ip length
GROUP BY host
AGGREGATE average
```

```
SELECT timestamp
GROUP BY tcp flow
AGGREGATE variance
```

Regroup to any sub-key for 1 KV op. and copy per GPV.

```
export key = <src IP, dst IP,
src port, dst port, protocol,
link ID>
```

Grouped Packet Vectors (GPVs)



*Flow Architecture

Regrouping and statistics calculation.

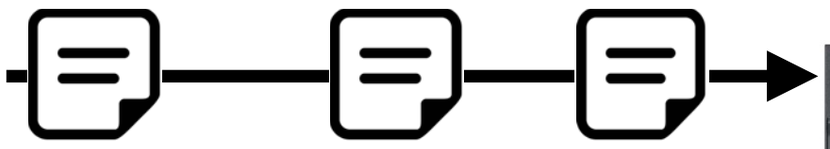
Monitoring Applications

Packet feature selection and initial grouping.

***Flow Cache**

Thin interface to grouped packet vectors.

Grouped Packet Vectors



***Flow Agent**

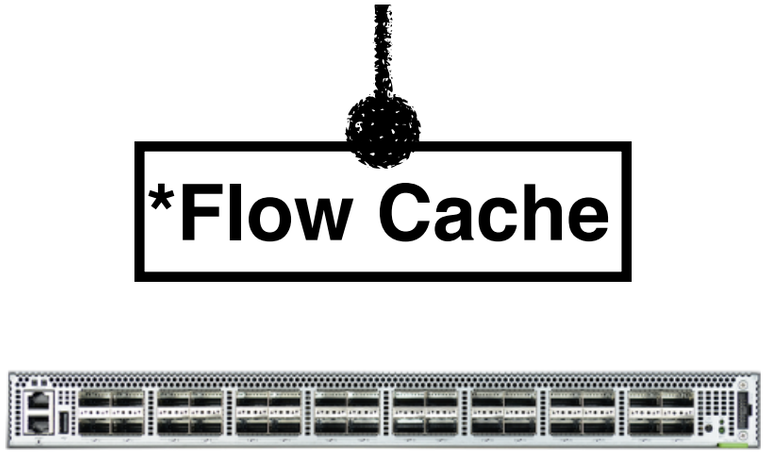
Outline

- Introduction
- Design
- **Implementation**
- Evaluation

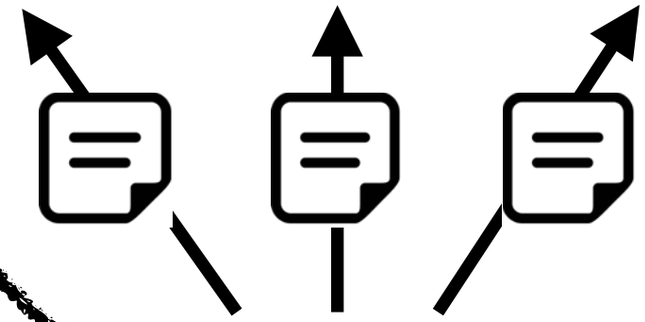
*Flow Implementation

Proof-of-concept implementation.

P4 implementation for 3.2 Tb/s Barefoot Tofino.



Grouped Packet Vectors

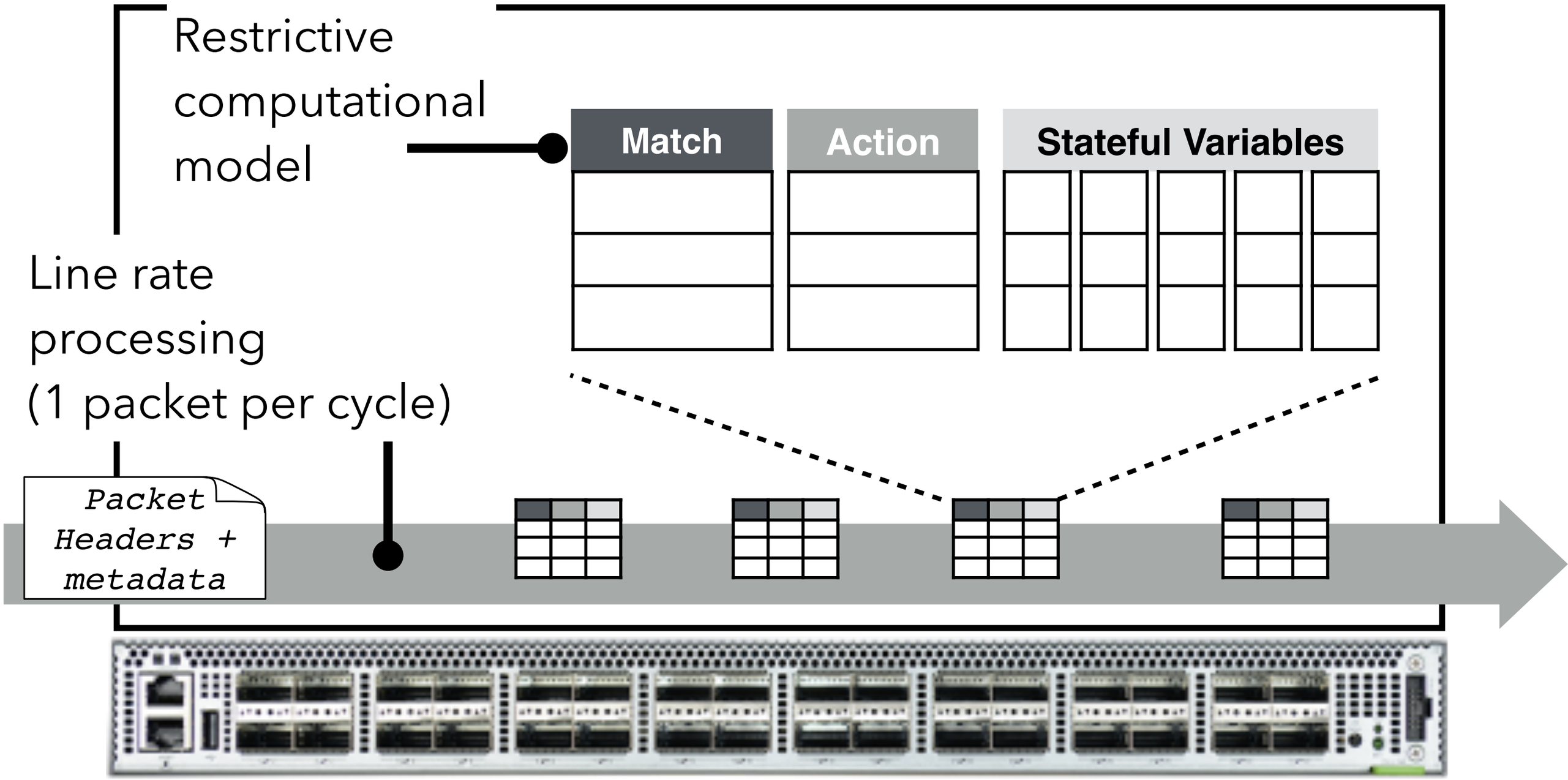


Background:

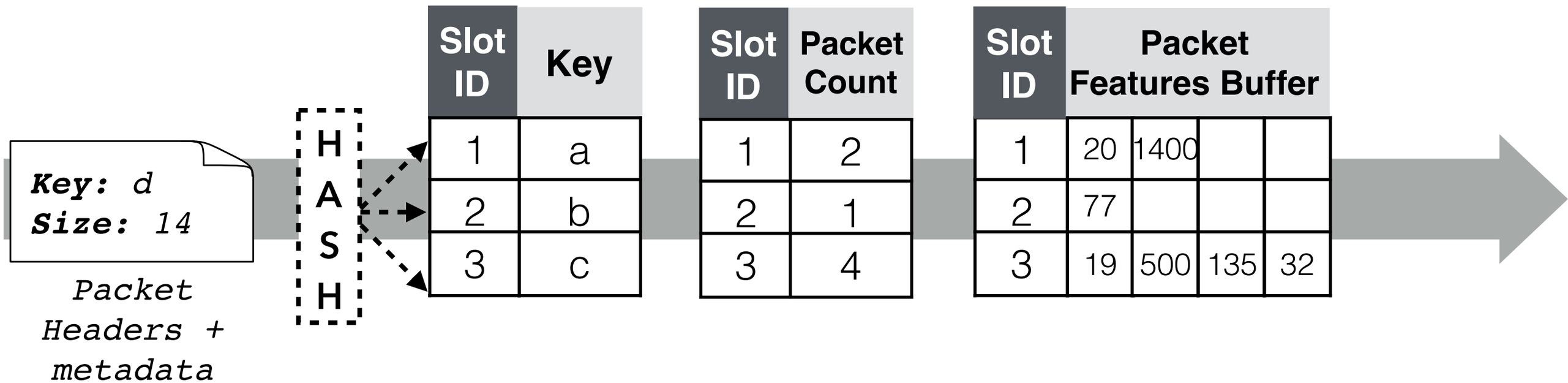
Stateful Match Action ASICs

Restrictive computational model

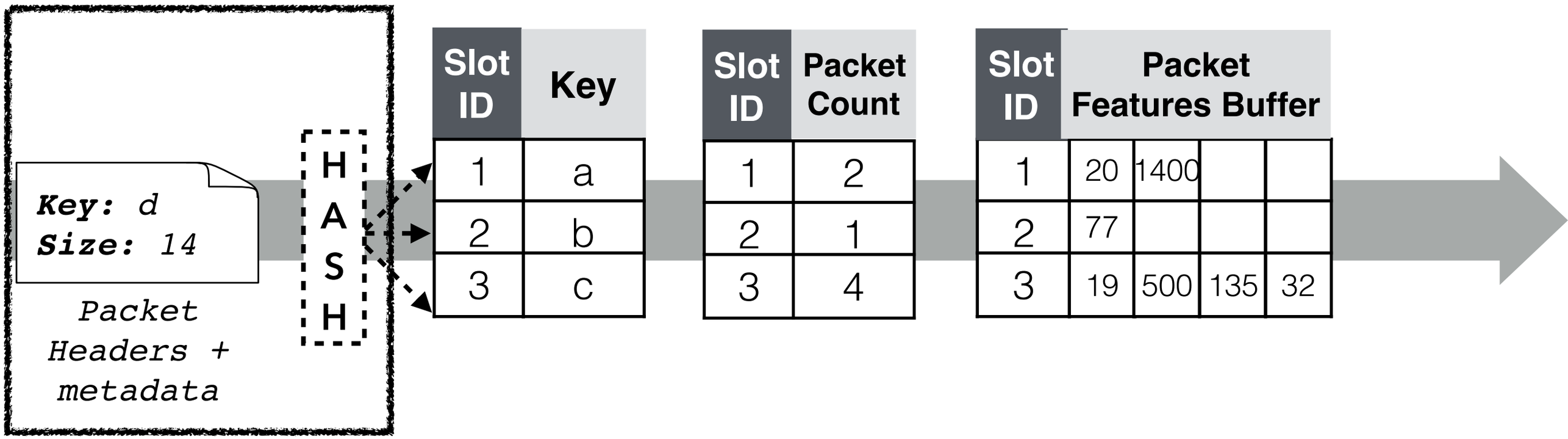
Line rate processing (1 packet per cycle)



*Flow as a Stateful Match Action Pipeline

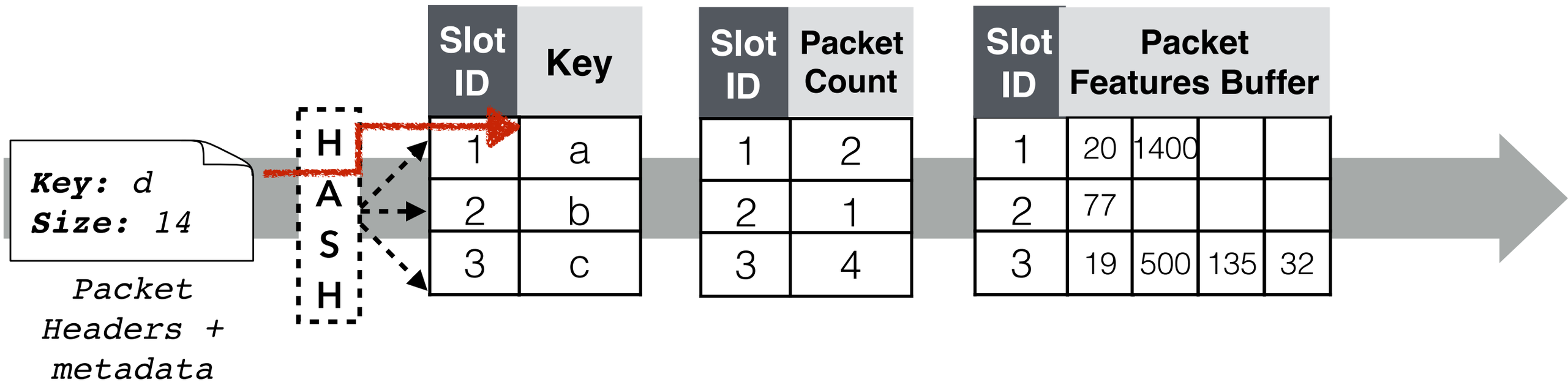


*Flow as a Stateful Match Action Pipeline



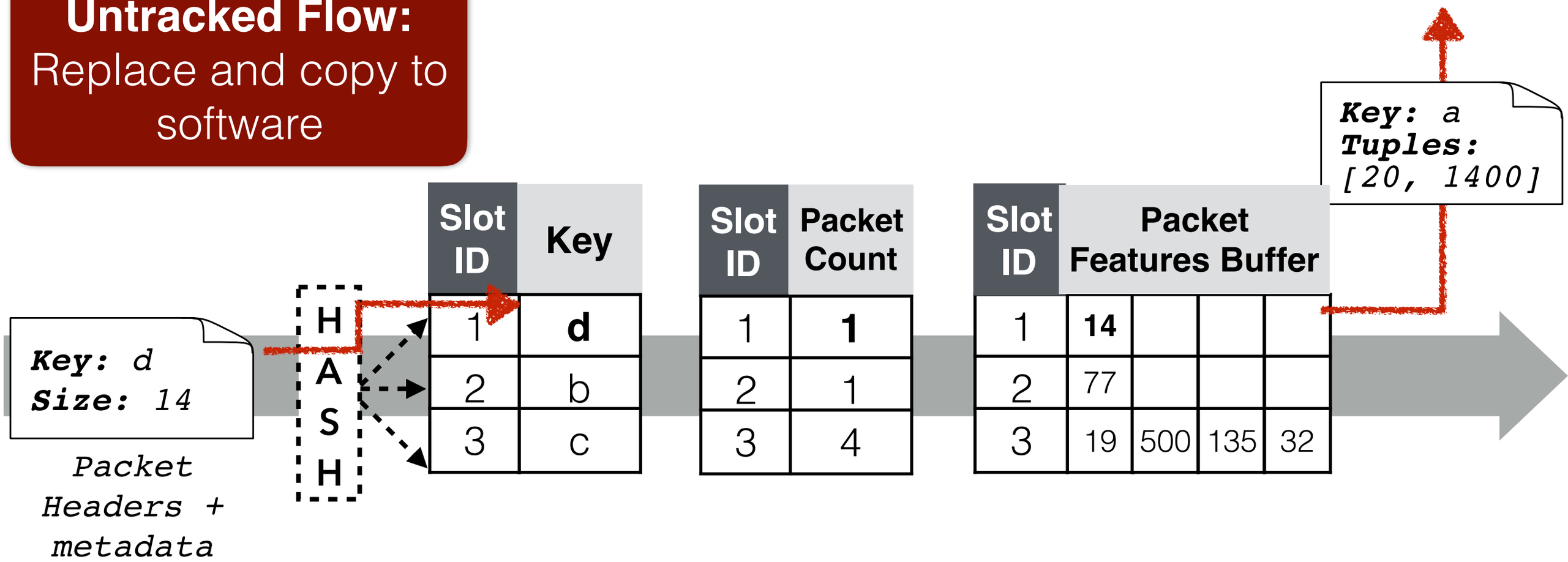
*Flow as a Stateful Match Action Pipeline

Untracked Flow:
Replace and copy to software



*Flow as a Stateful Match Action Pipeline

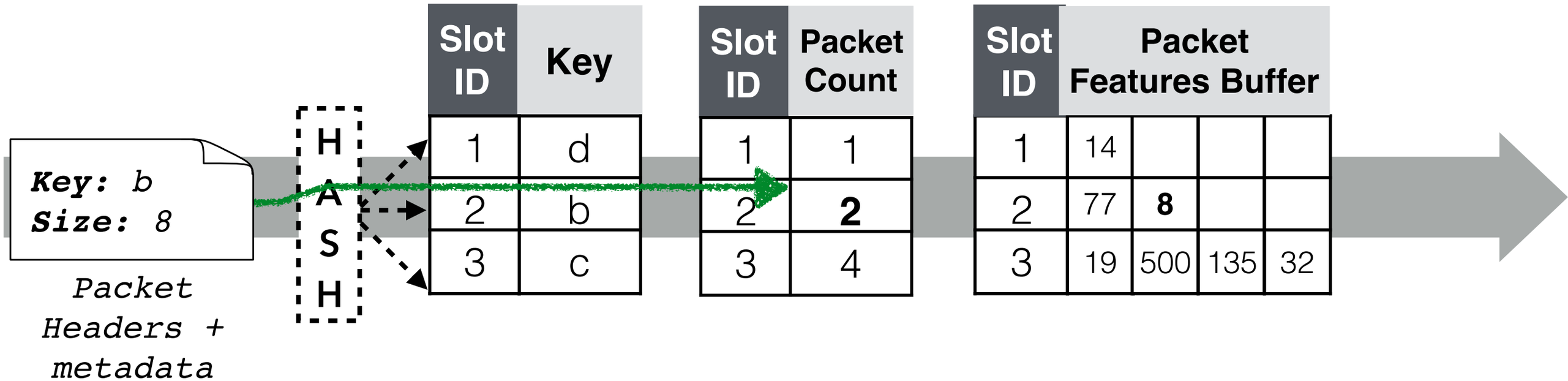
Untracked Flow:
Replace and copy to software



*Flow as a Stateful Match Action Pipeline

Untracked Flow:
Replace and copy to software

Tracked Flow:
Append tuple

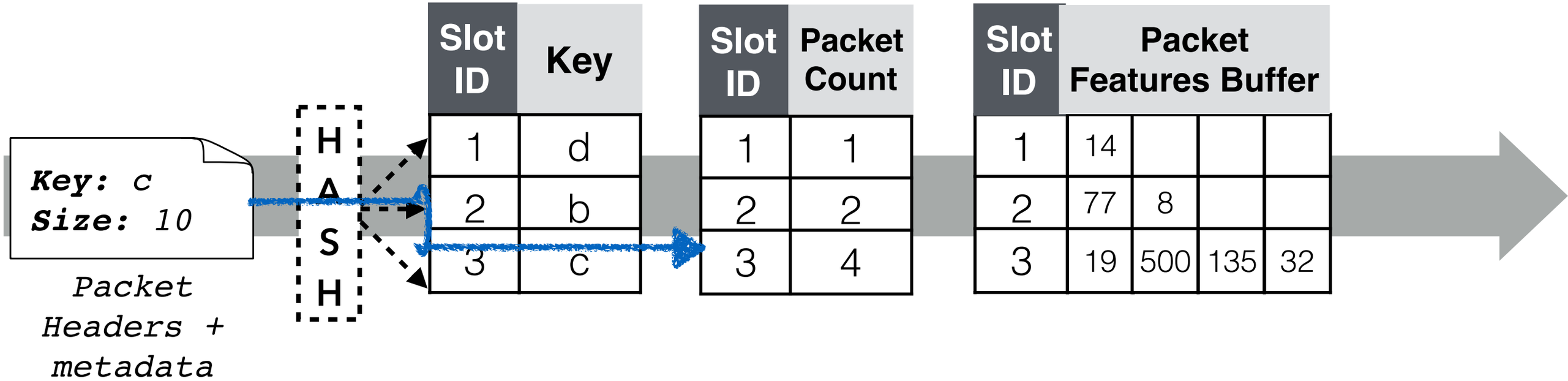


*Flow as a Stateful Match Action Pipeline

Untracked Flow: Replace and copy to software

Tracked Flow: Append tuple

Tracked Flow, Buffer Full: rollover buffer

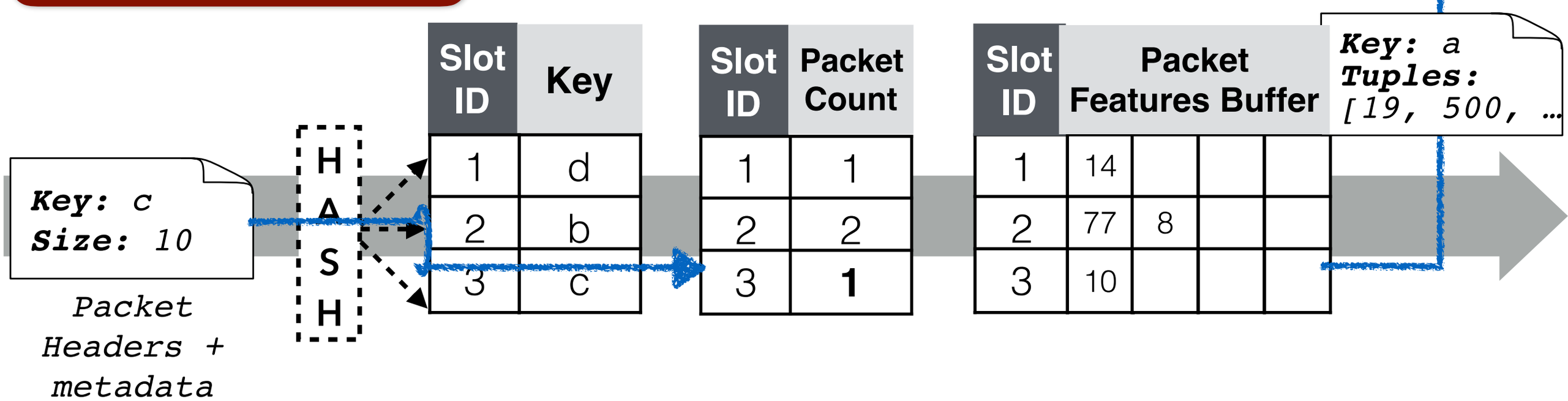


*Flow as a Stateful Match Action Pipeline

Untracked Flow: Replace and copy to software

Tracked Flow: Append tuple

Tracked Flow, Buffer Full: rollover buffer

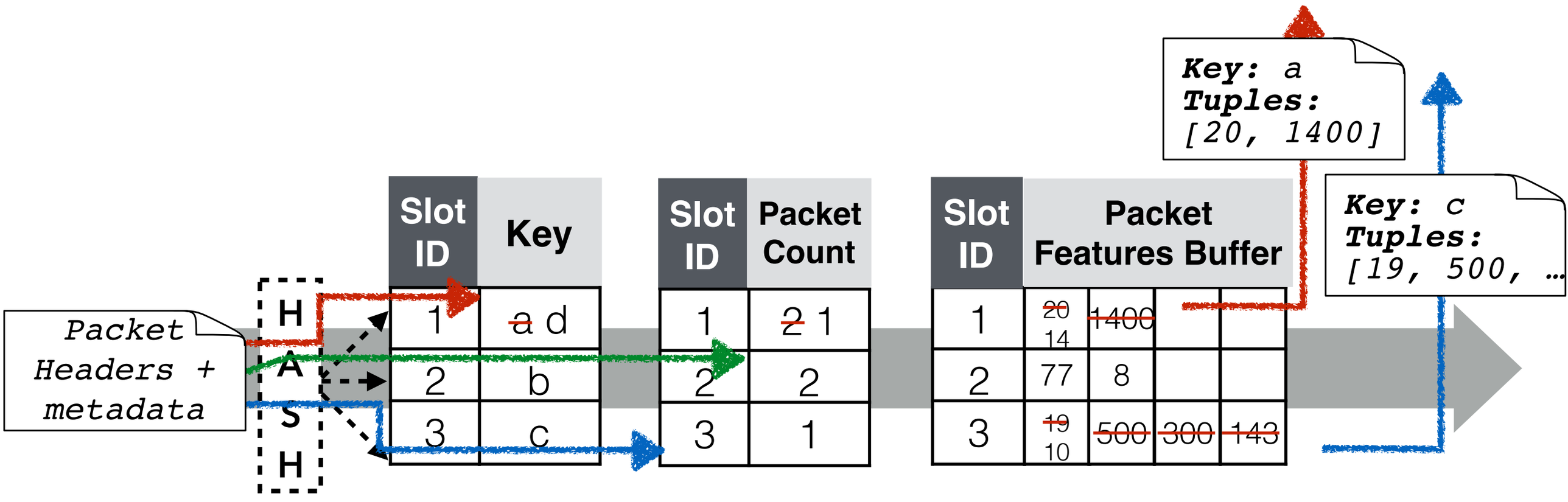


*Flow as a Stateful Match Action Pipeline

Untracked Flow: Replace and copy to software

Tracked Flow: Append features

Tracked Flow, Buffer Full: rollover buffer



Outline

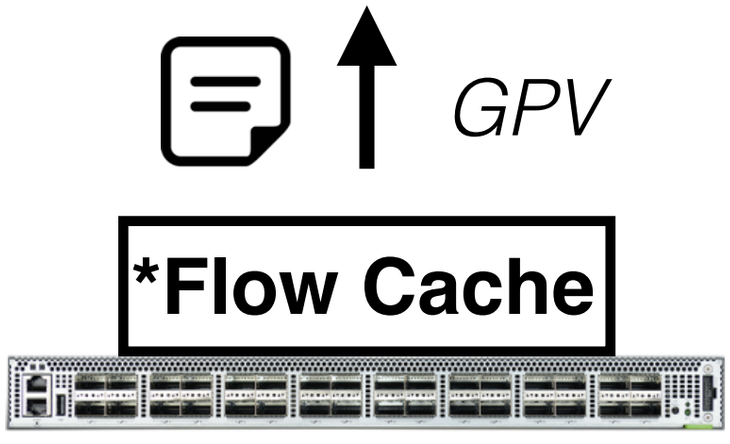
- Motivation
- Design
- Implementation
- **Evaluation**

Evaluation: Concurrency and Efficiency

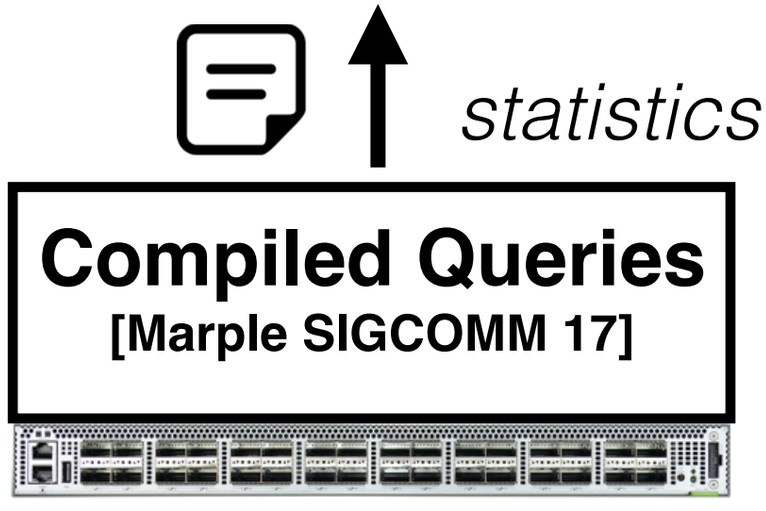
Evaluation: Concurrency and Efficiency

Query	Stages	ALUs
Connection Count		
Avg. Latency		
Flowlet Sizes		
Packet Counts		
TCP Non-Monotonic		
TCP Out of Sequence		

*Flow Cache	Stages	ALUs



Vs

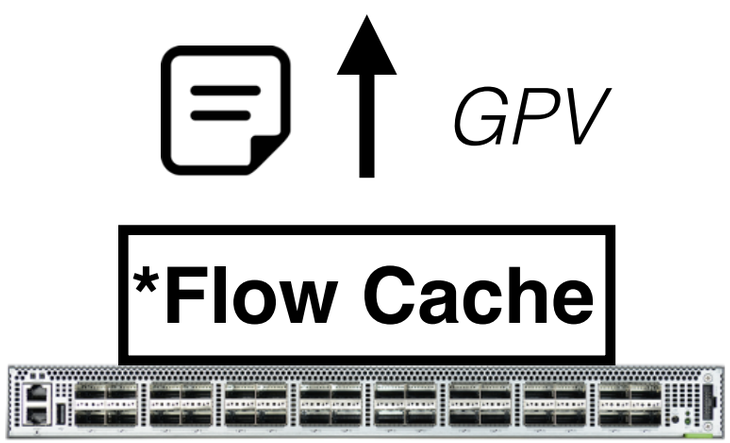


Evaluation: Concurrency and Efficiency

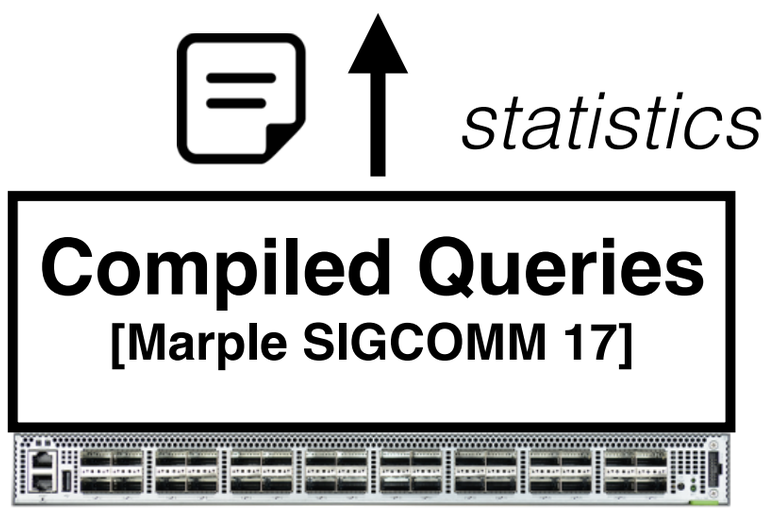
The *Flow Cache can service **any** query but only requires as many hardware resources as **~1** compiled query.

	Stages	ALUs
***Flow Cache**	11	33

Query	Stages	ALUs
Connection Count	4	10
Avg. Latency	6	11
Flowlet Sizes	11	31
Packet Counts	5	7
TCP Non-Monotonic	5	6
TCP Out of Sequence	7	14



Vs

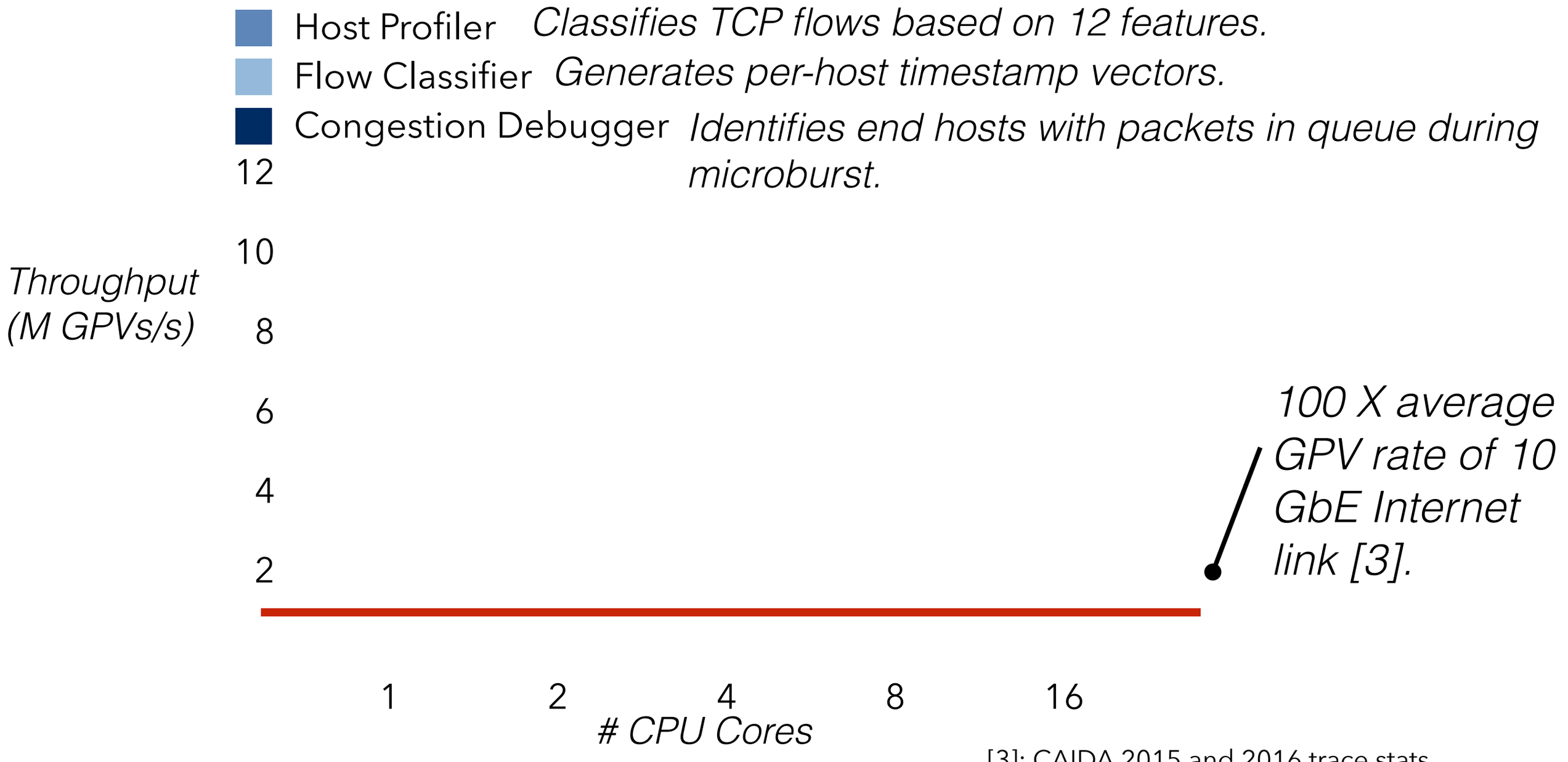


Evaluation: Concurrency and Efficiency

Host Profiler — *Classifies TCP flows based on 12 features.*

Flow Classifier — *Generates per-host timestamp vectors.*

Congestion Debugger — *Identifies end hosts with packets in queue during microburst.*



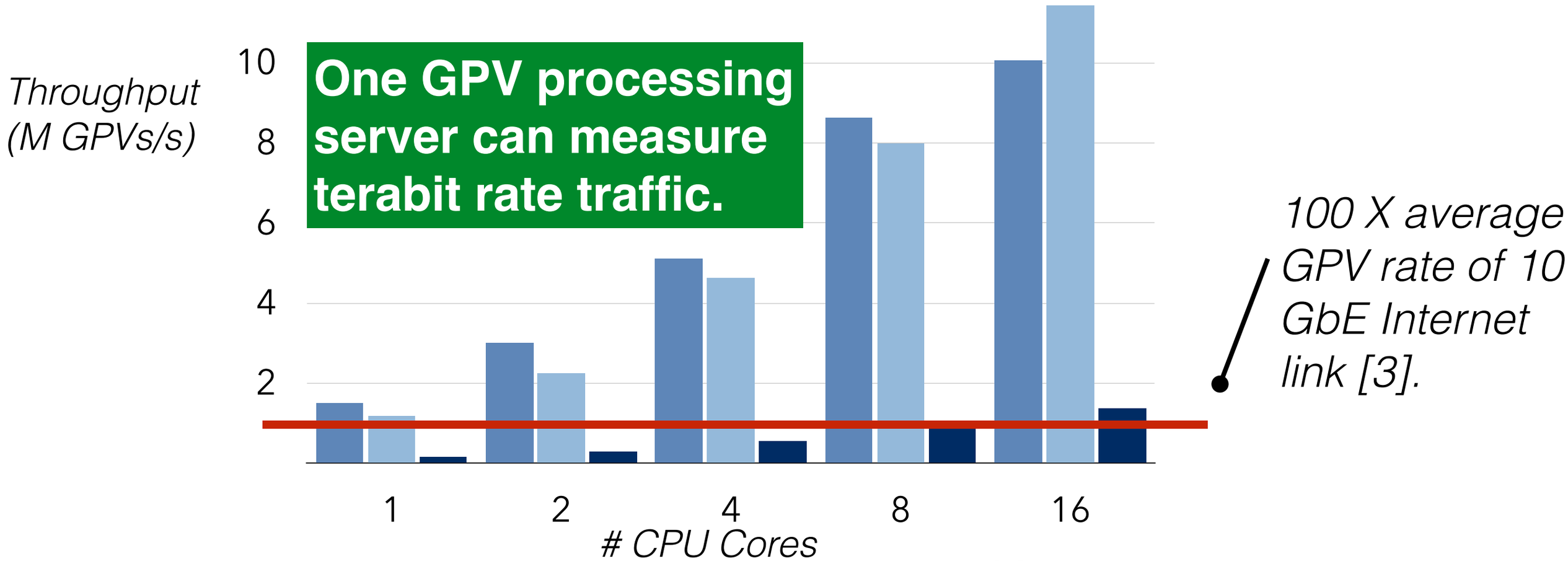
[3]: CAIDA 2015 and 2016 trace stats https://www.caida.org/data/passive/trace_stats/

Evaluation: Concurrency and Efficiency

Host Profiler *Classifies TCP flows based on 12 features.*

Flow Classifier *Generates per-host timing profiles.*

Congestion Debugger *Identifies end hosts with packets in queue during microburst.*

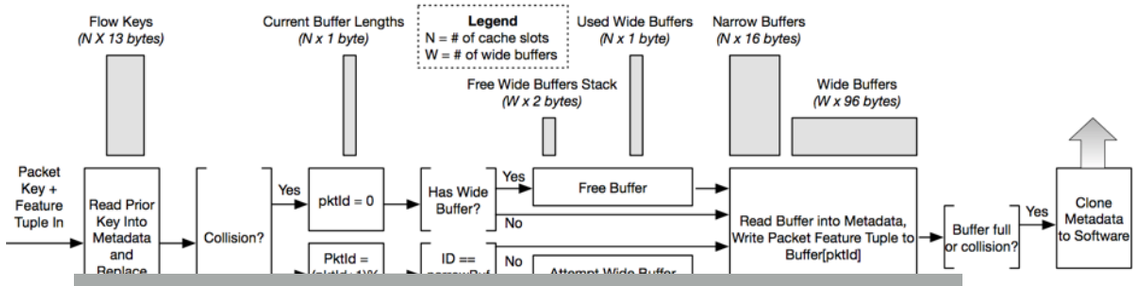


[3]: CAIDA 2015 and 2016 trace stats
https://www.caida.org/data/passive/trace_stats/

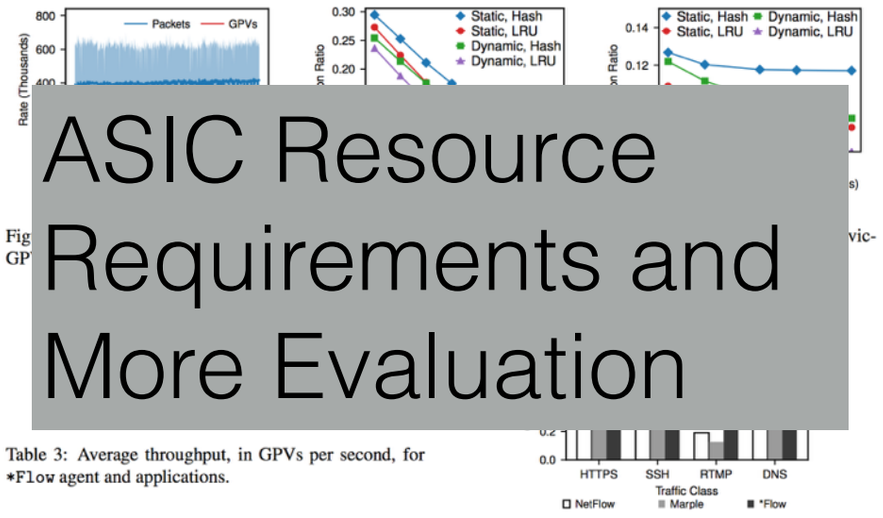
Outline

- Motivation
- Design
- Implementation
- Evaluation

In The Paper



Match+Action Layout



ASIC Resource Requirements and More Evaluation

Table 3: Average throughput, in GPVs per second, for *Flow agent and applications.

	Efficient	Flexible	Concurrent	Dynamic
Netflow	✓	✗	✓	✓
Software	✗	✓	✓	✓
PFE Queries	✓	✓	✗	✗
***Flo**				✓

Background

On Github

www.github.com/jsonch/starflow

```
// Tables.
table UpdateKey { default_action :UpdateKeyAction(); }
table UpdateFeatures { default_action
     :UpdateFeaturesAction(); }
table ResetFeatures { default_action
     :ResetFeaturesAction(); }

// Actions.
// Update key for every packet.
   register_write(evictBufArr, 0, tempMfr.evictBufPos+1);
   register_write(evictBufKey, tempMfr.evictBufPos,
        tempMfr.key);
   register_write(evictBufPktCt, tempMfr.evictBufPos,
        tempMfr.pktCt);
}
```

3.2 Tb/s *Flow Cache Prototype for Barefoot Tofino

Conclusion
(and Thank You for Listening!)

*What are **the right** measurement subtasks to do in hardware?*

✓ **Concurrency**

✓ **Efficiency**

www.github.com/jsonch/starflow