nsdi '24

Empower Programmable Pipeline for Advanced Stateful Packet Processing

Yong Feng[1], Zhikang Chen[1], Haoyu Song[2], Yinchao Zhang[1], Ruoyu Sun[1],
Wenkuo Dong[1], Peng Lu[1], Shuxin Liu[1], Chuwen Zhang[1], Yang Xu[3], and Bin Liu[1]

[1]Tsinghua University; [2]Futurewei Technologies; [3]Fudan University

Stateful Network Functions

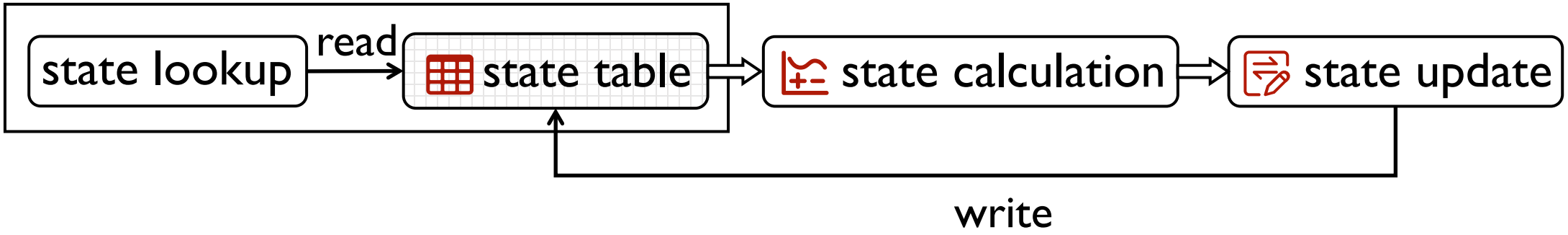
- Stateful Load Balancer
- DDoS Detection and Mitigation
- Traffic Shaping and Policing
- Stateful Firewall with Connection Tracking
- Network Visualization
- ……

Stateful Network Functions

- Stateful Load Balancer
- DDoS Detection and Mitigation
- Traffic Shaping and Policing
- Stateful Firewall with Connection Tracking
- Network Visualization
- ……

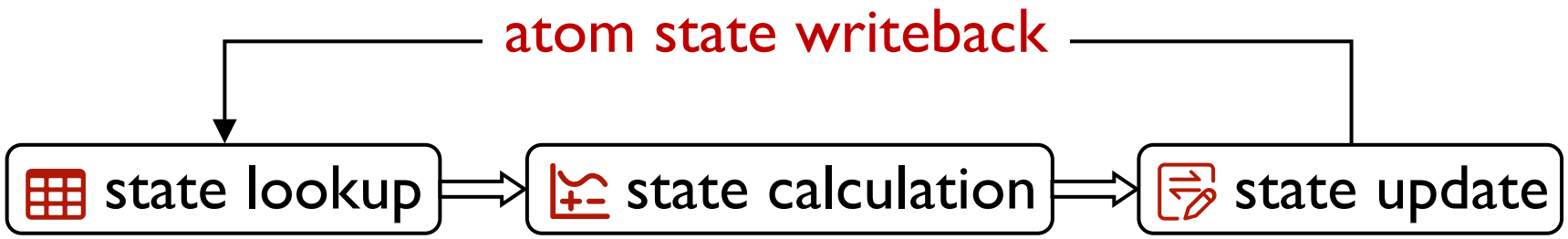
The processing results of preceding packets will affect subsequent packets in a flow.

Abstraction



Abstraction

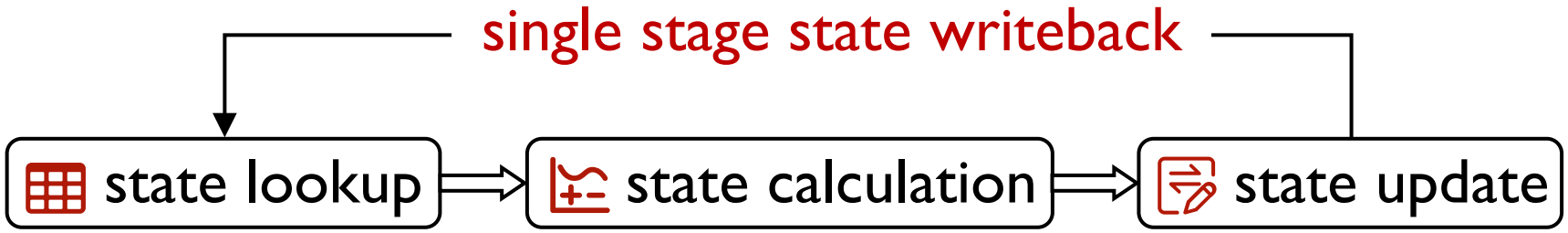
Banzai (SIGCOMM 2017)



e.g., a = a + b

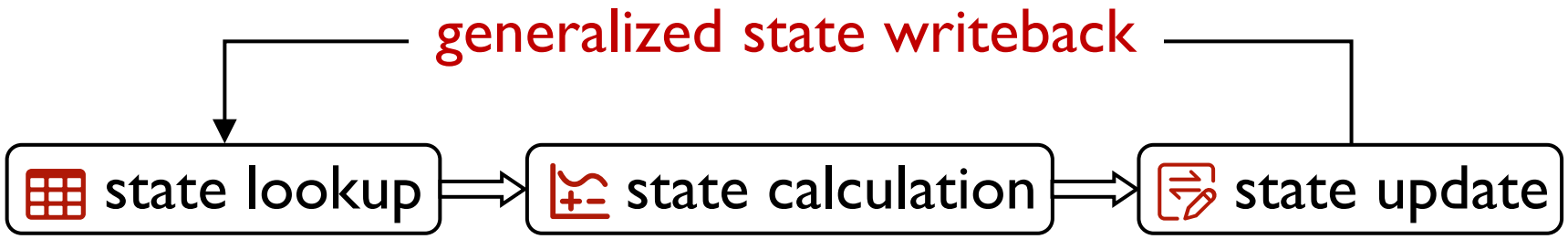
Abstraction

FlowBlaze (NSDI 2019)



e.g., a = tbl(a) + b

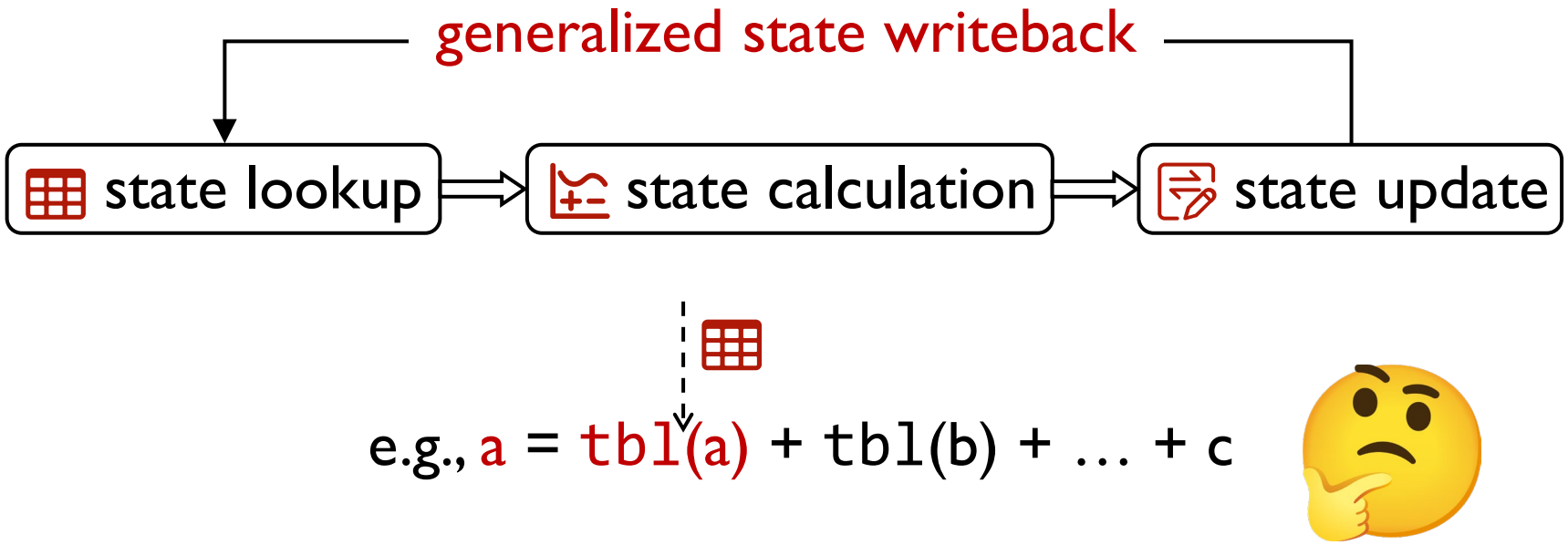
Abstraction



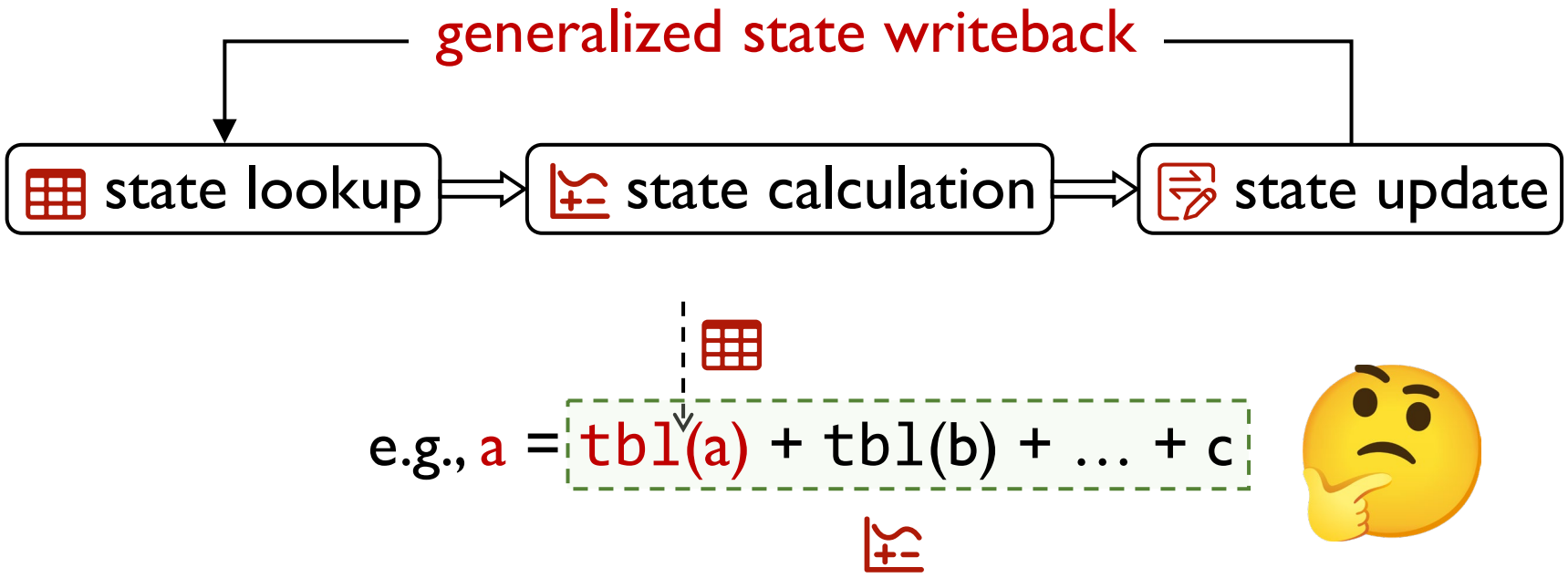
e.g., a = tbl(a) + tbl(b) + … + c

- Stateful Load Balancer
- DDoS Detection and Mitigation
- Traffic Shaping and Policing
- Stateful Firewall with Connection Tracking
- Network Visualization
- ……

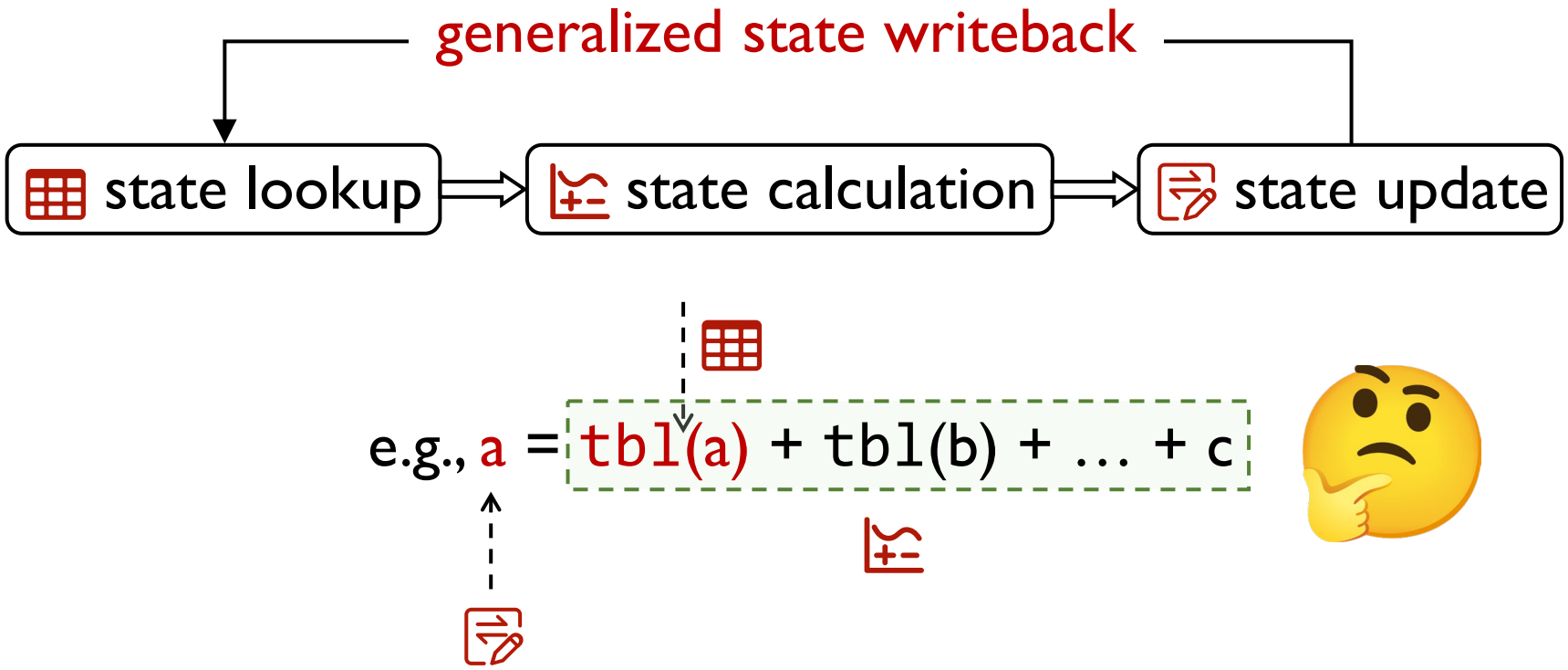
Abstraction



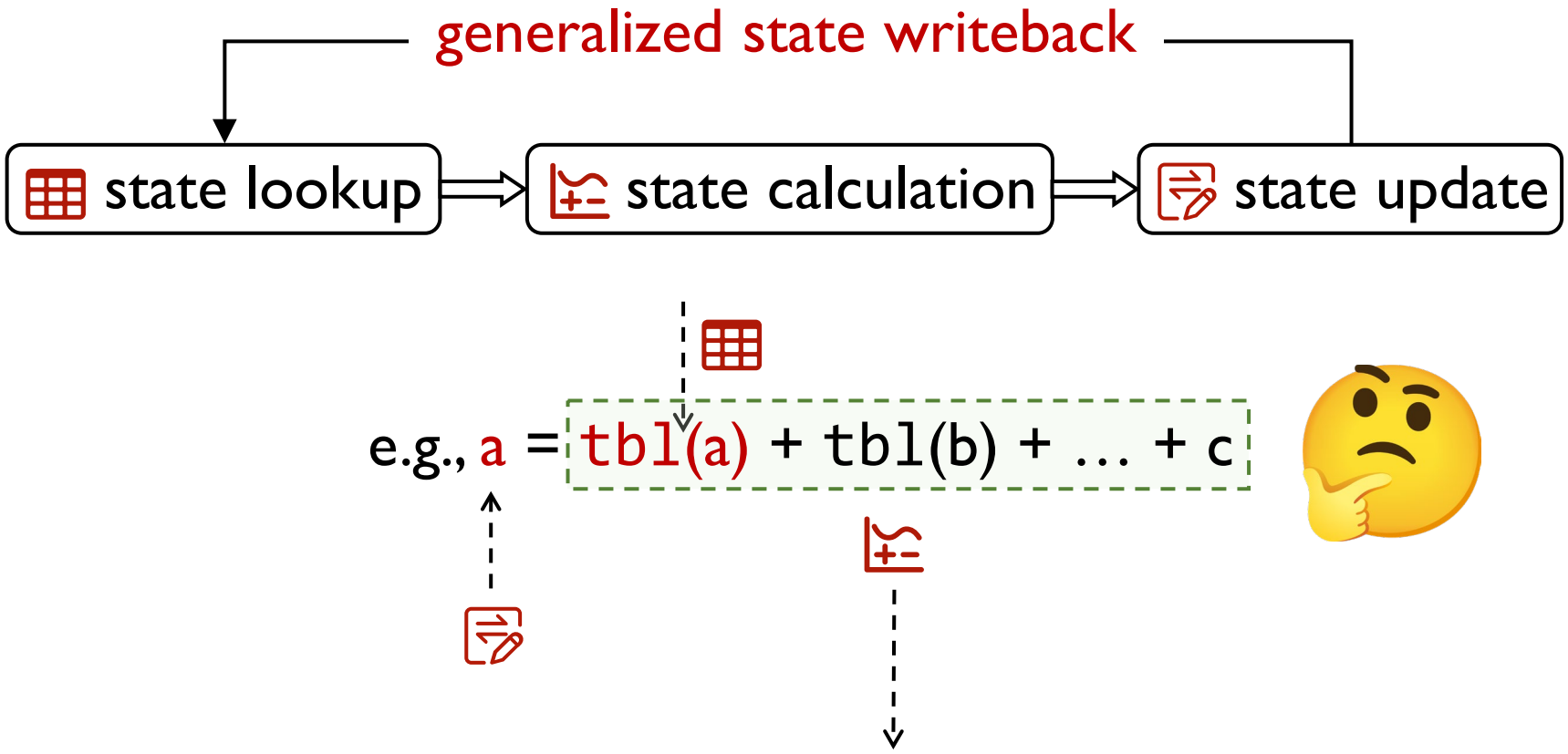
Abstraction



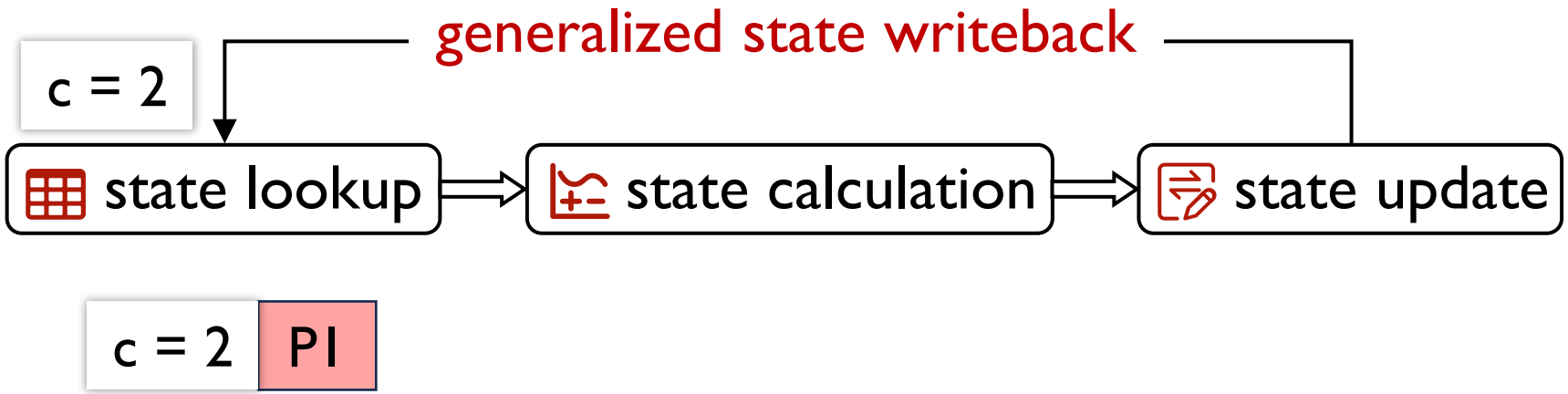
Abstraction



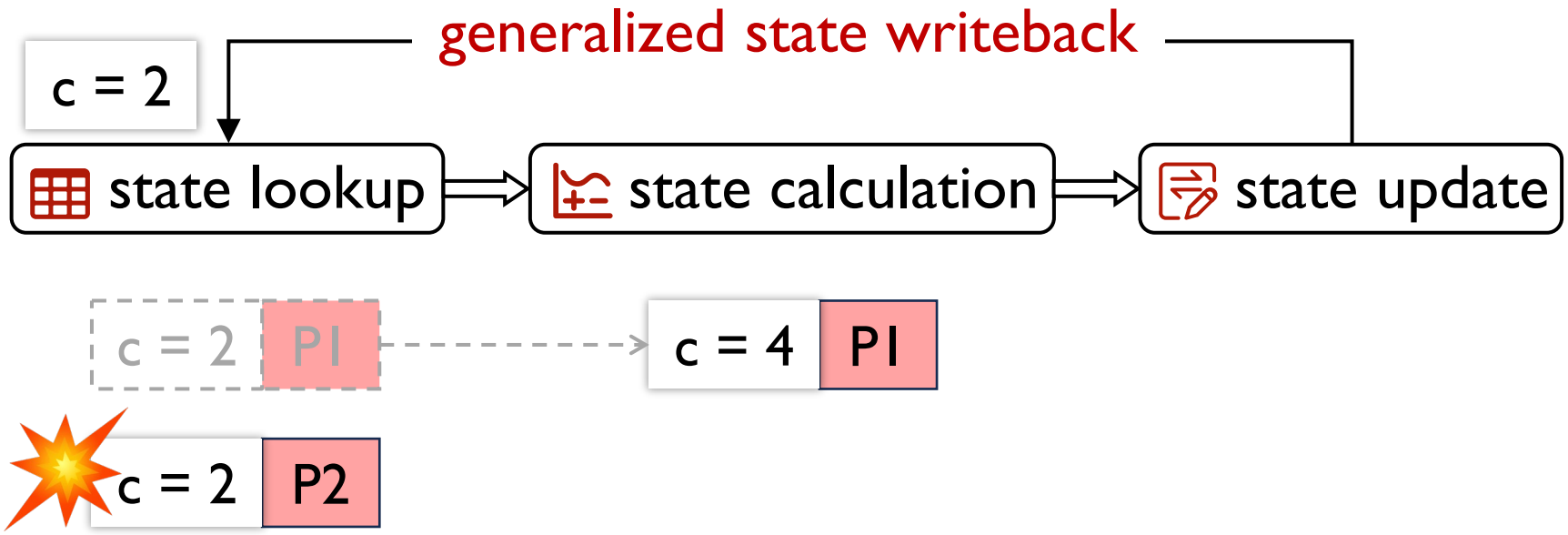
Abstraction



Architecture Issue

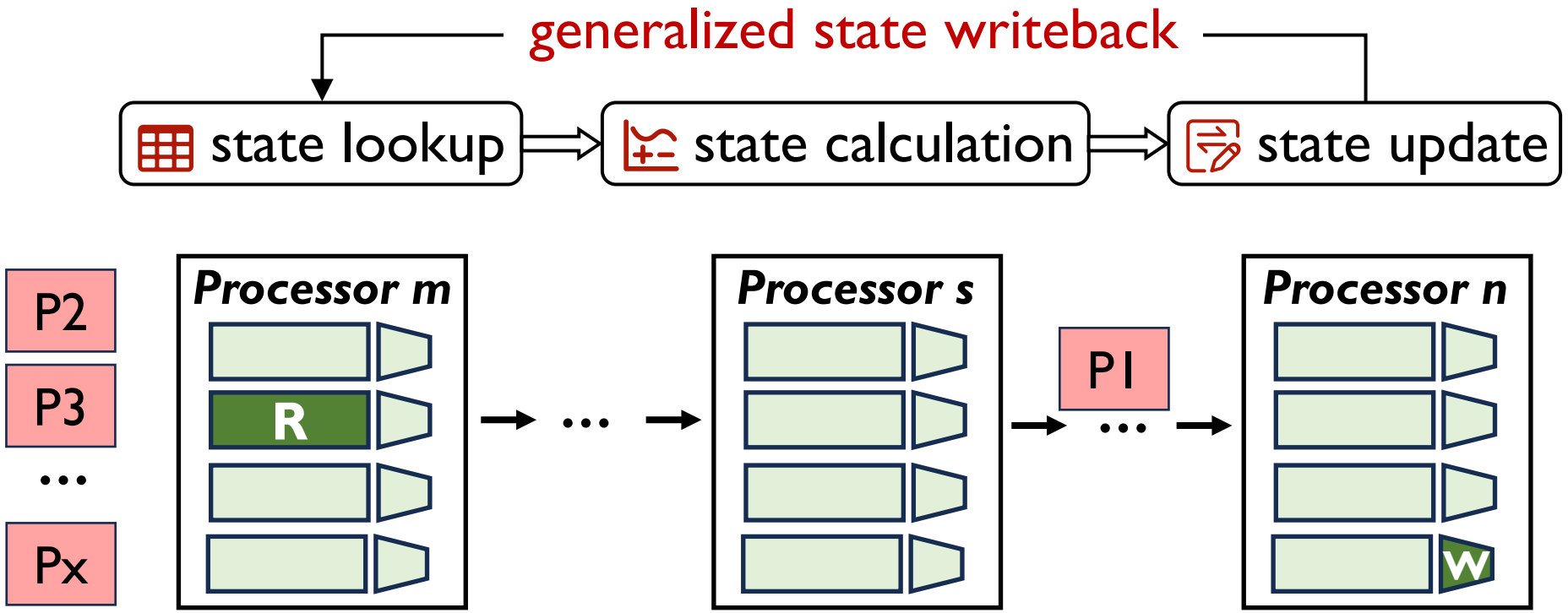


Architecture Issue

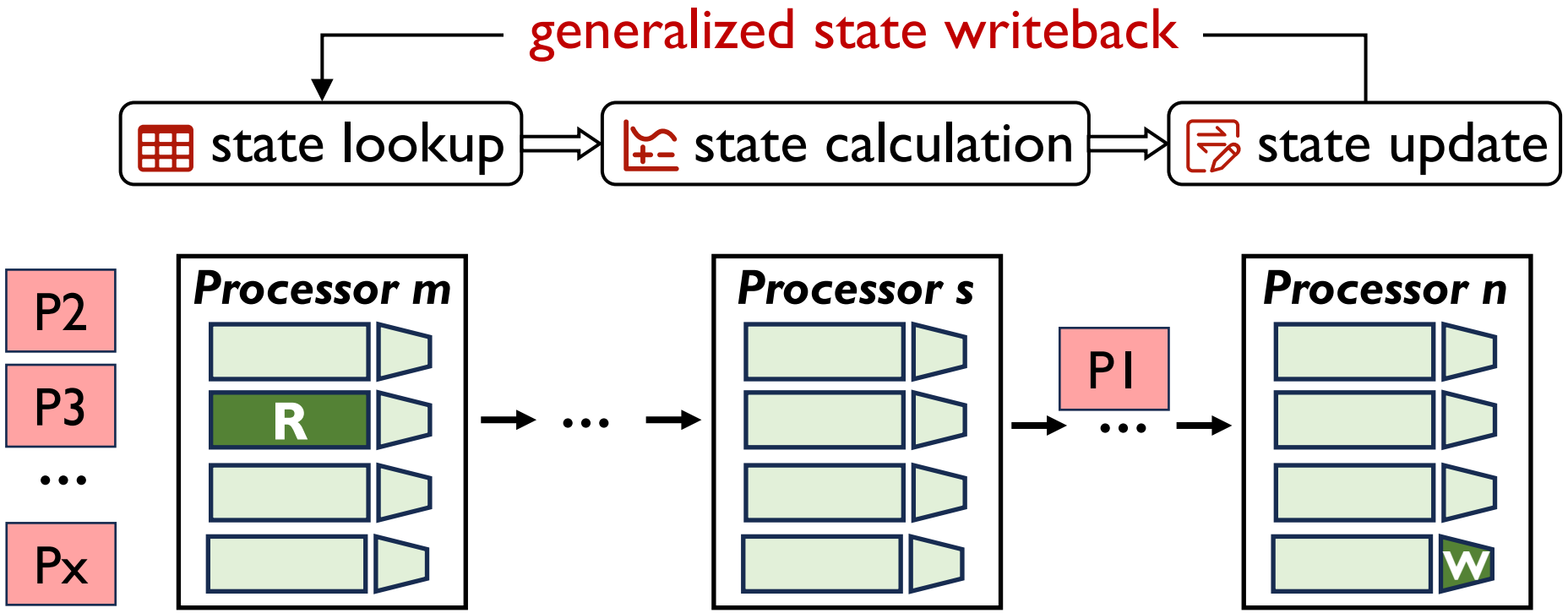


State Inconsistency due to RAW hazard!

Architecture Issue (pipeline-based)

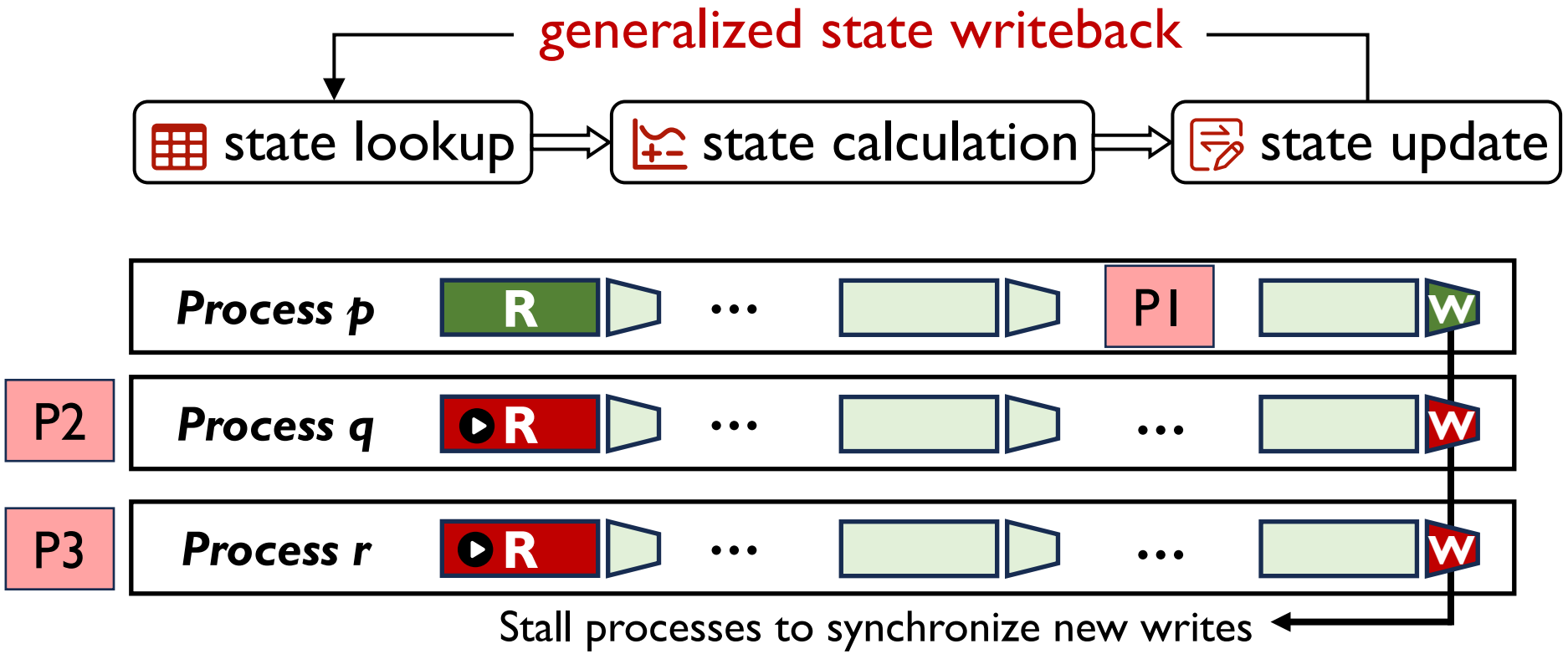


Architecture Issue (pipeline-based)

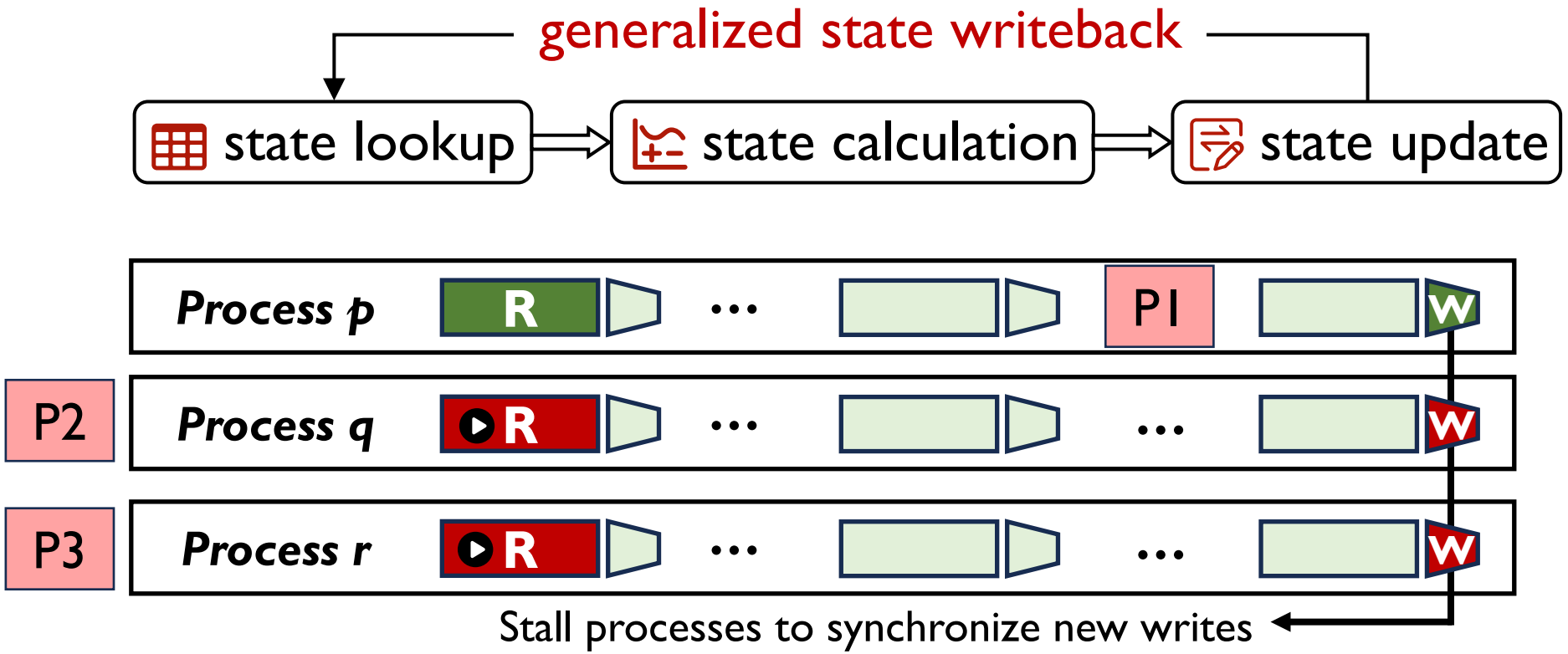


Pipeline: degrade from Pipeline to Run-To-Completion

Architecture Issue (multicore-based)



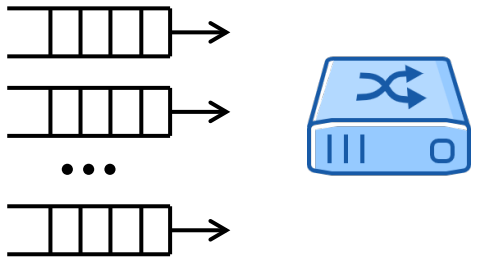
Architecture Issue (multicore-based)



Multi-core: degrade from Parallel to Sequential

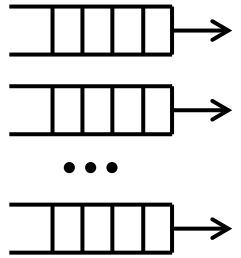
Evidence

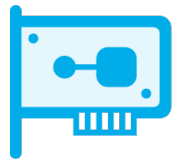
State Consistency solution (multi-queue) in FlowBlaze is used.



960Gbps/1.6Tbps

Different number of queues



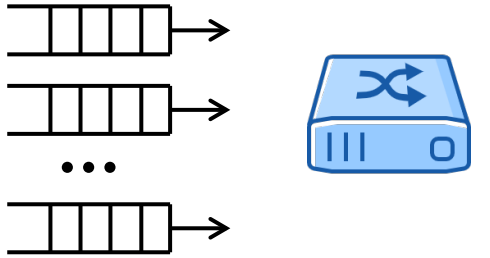


28Gbps/100Gbps

Different number of queues

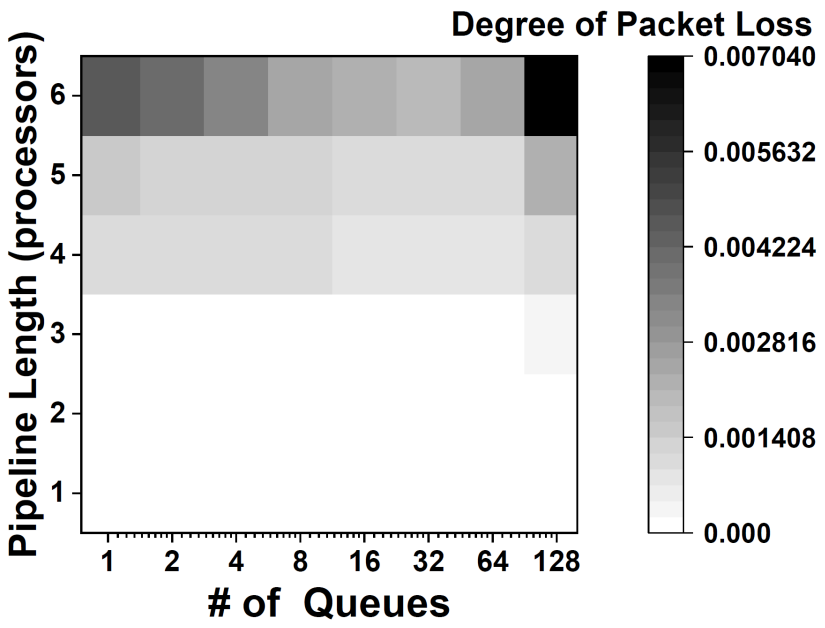
Evidence

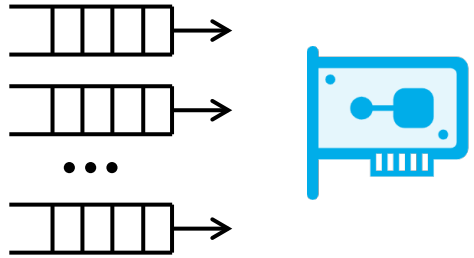
State Consistency solution (multi-queue) in FlowBlaze is used.



960Gbps/1.6Tbps

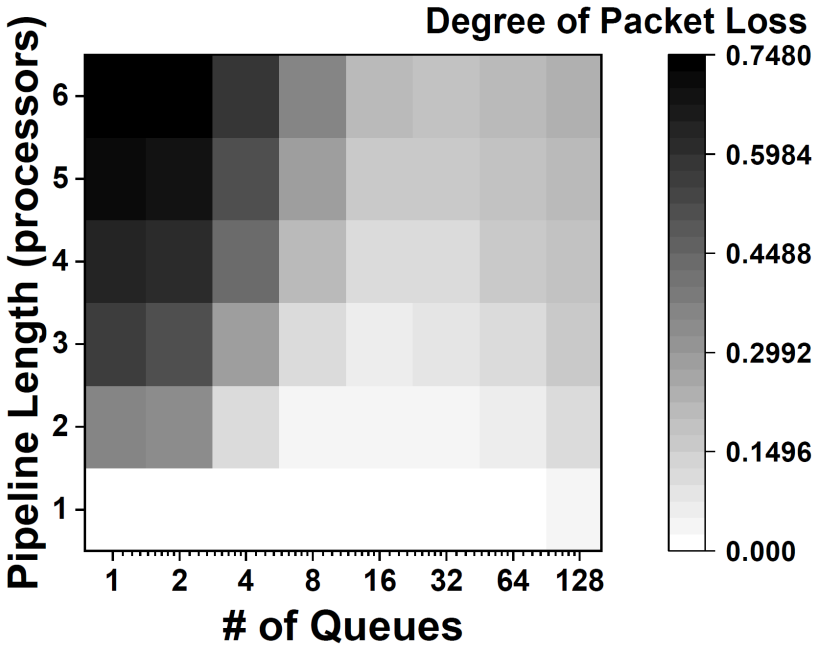
Different number of queues



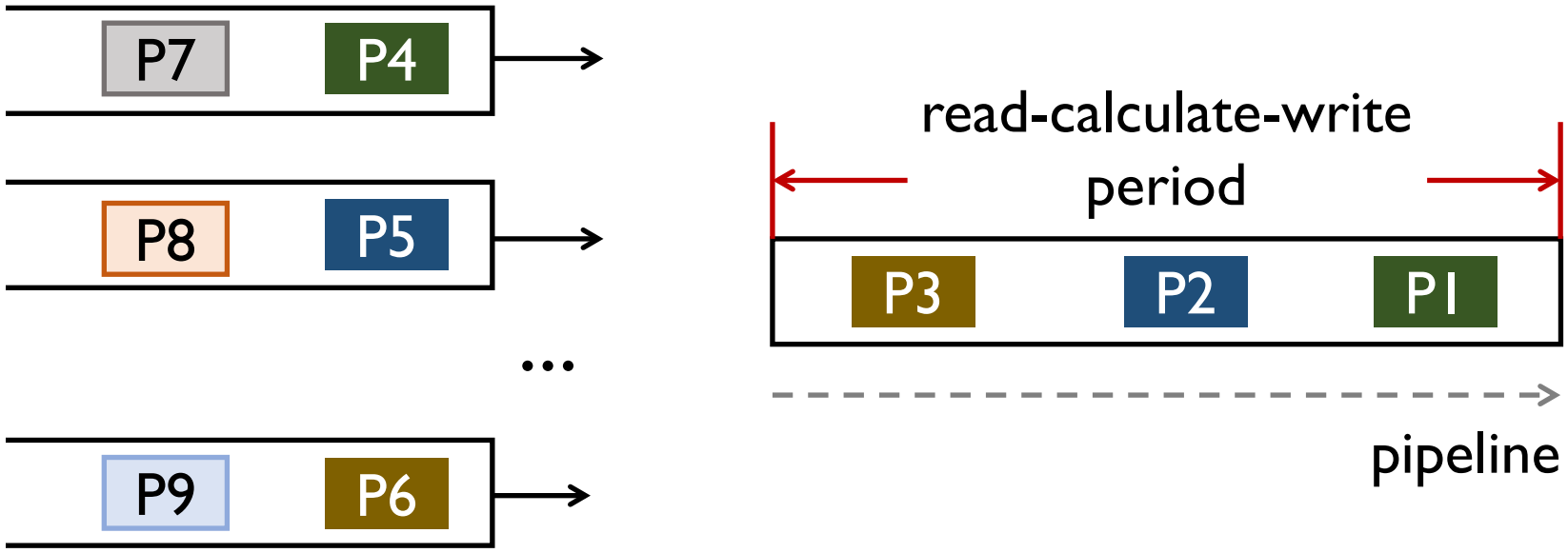


28Gbps/100Gbps

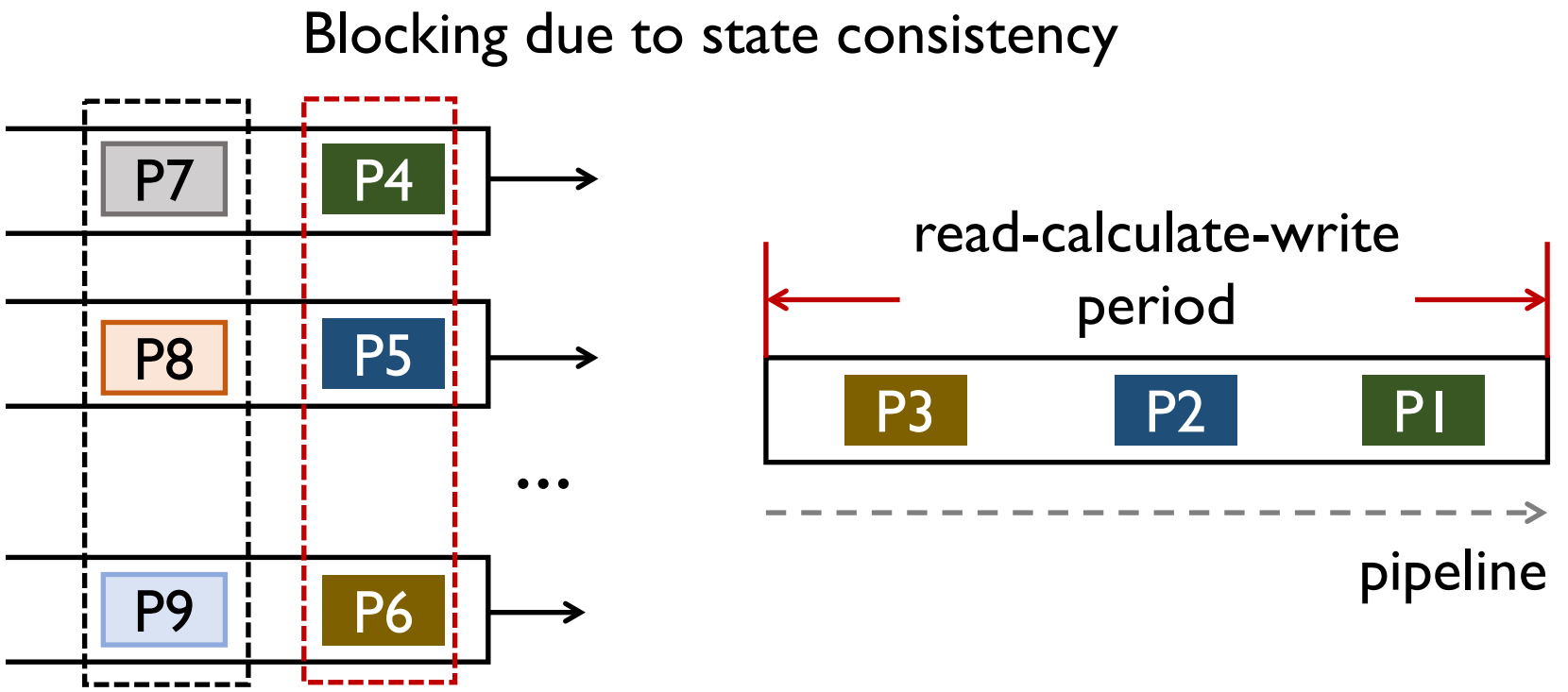
Different number of queues



Evidence

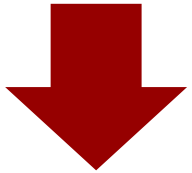

P7
P4
P8
P5
...
P9
P6
read-calculate-write
period
P3
P2
P1
pipeline

Evidence



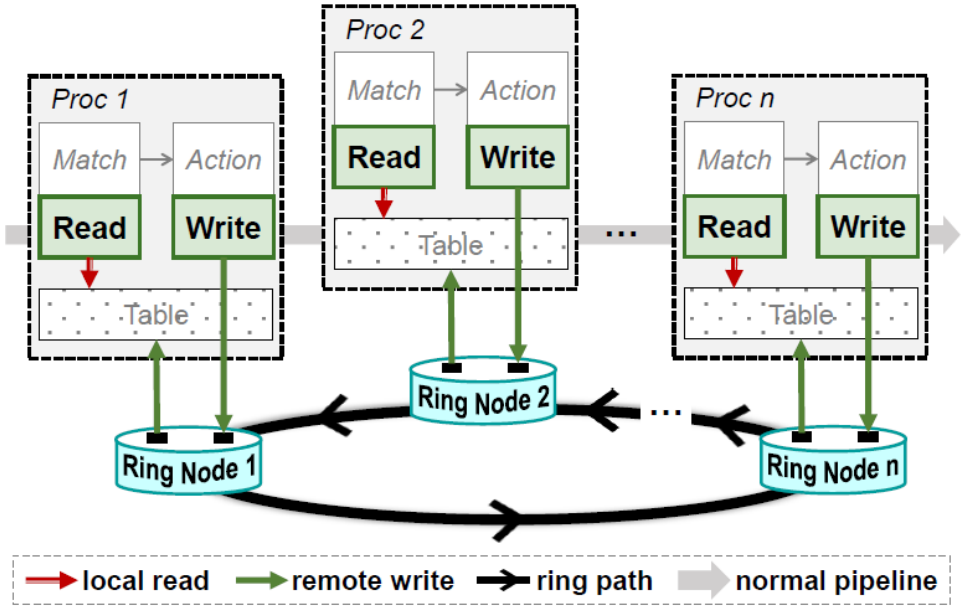
Head-of-Line Blocking

More Generic, More Efficient, Higher Performance



New Abstraction, New Execution Mode, New Architecture

New Abstraction – Dataplane Writable Table



- Local Read - Remote Write
  - From exclusive to shared memory
  - Saves resources than global reading and writing
- Enlarge range of state tables accessible at each stage

New Execution Mode

Stateful Network Function	State Update Ratio
Heavy hitter	$1/n$
Load balancer	$1/n$
NAT	$1/n$
Stateful firewall	a/n
Port knocking	a/n
SYN flood detection	$1/n$
TCP connection tracking	a/n

* $\#/n$ means the first # packets of a flow

New Execution Mode

Stateful Network Function	State Update Ratio
Heavy hitter	$1/n$
Load balancer	$1/n$
NAT	$1/n$
Stateful firewall	a/n
Port knocking	a/n
SYN flood detection	$1/n$
TCP connection tracking	a/n

* $\#/n$ means the first # packets of a flow

Not every packet updates flow state!

New Execution Mode

Stateful Network Function	State Update Ratio
Heavy hitter	$1/n$
Load balancer	$1/n$
NAT	$1/n$
Stateful firewall	a/n
Port knocking	a/n
SYN flood detection	$1/n$
TCP connection tracking	a/n

* $\#/n$ means the first # packets of a flow

Not every packet updates flow state!

100% blocking for low state update ratio!

New Execution Mode

Stateful Network Function	State Update Ratio
Heavy hitter	$1/n$
Load balancer	$1/n$
NAT	$1/n$
Stateful firewall	a/n
Port knocking	a/n
SYN flood detection	$1/n$
TCP connection tracking	a/n

* $\#/n$ means the first # packets of a flow

Not every packet updates flow state!

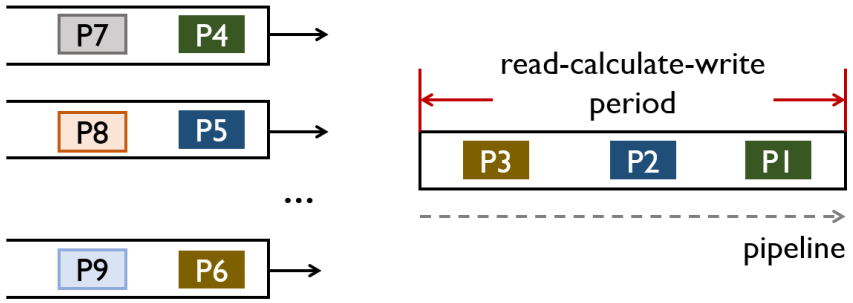
100% blocking for low state update ratio!

What happens if packets aren't blocked?

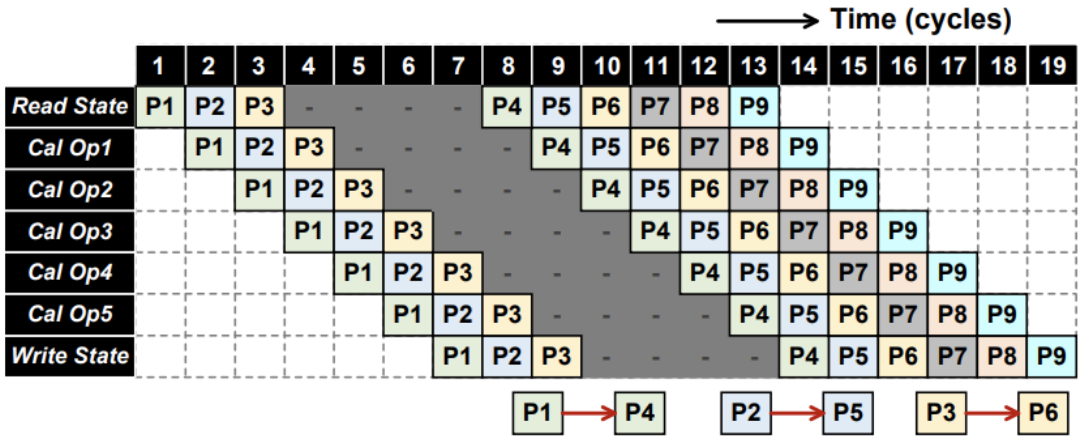
New Execution Mode – Speculative Execution

Predict packets won't modify states;
if states change, take measures to recover.

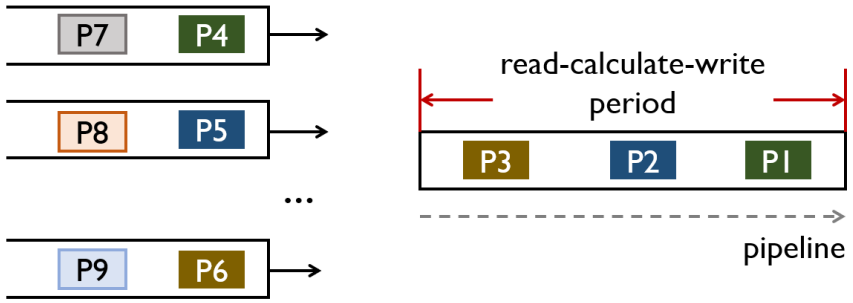
New Execution Mode – Speculative Execution



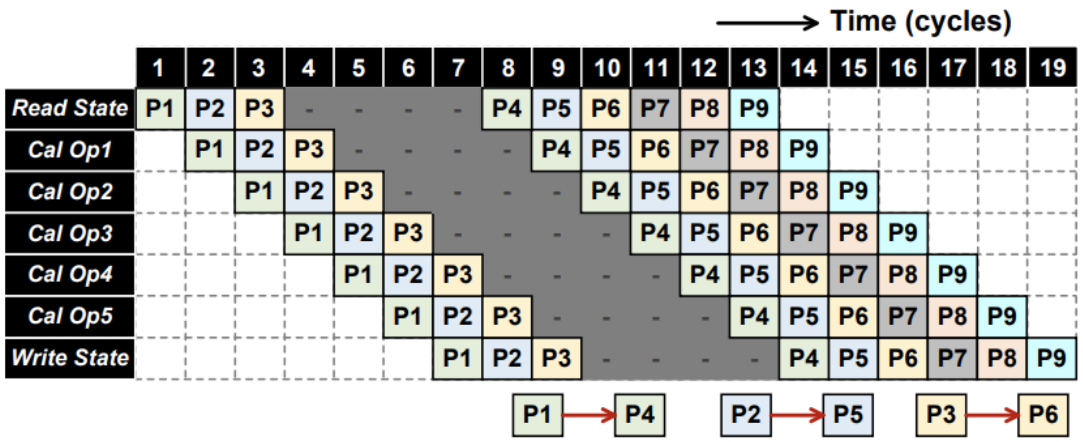
Predict packets won't modify states;
if states change, take measures to recover.

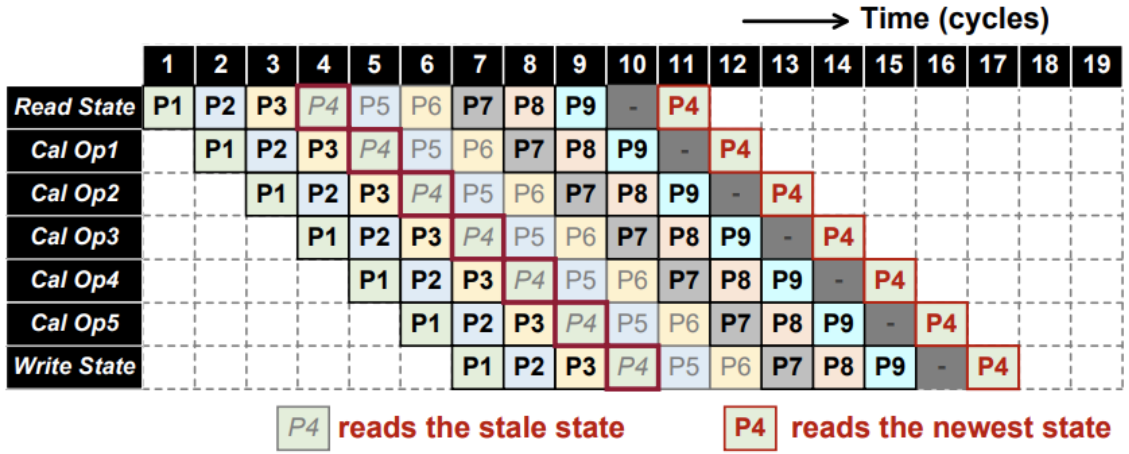


New Execution Mode – Speculative Execution

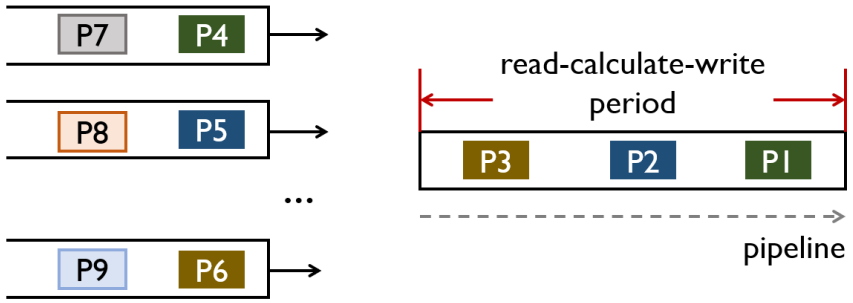


Predict packets won't modify states;
if states change, take measures to recover.

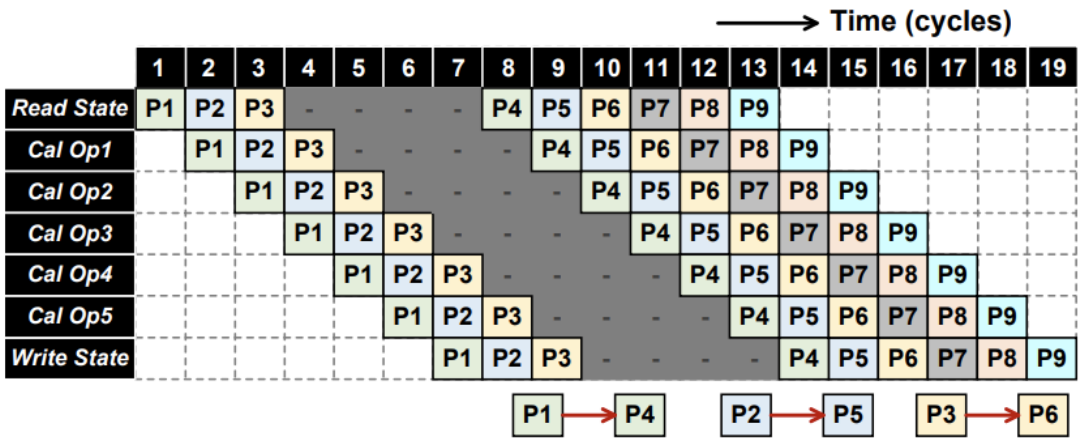


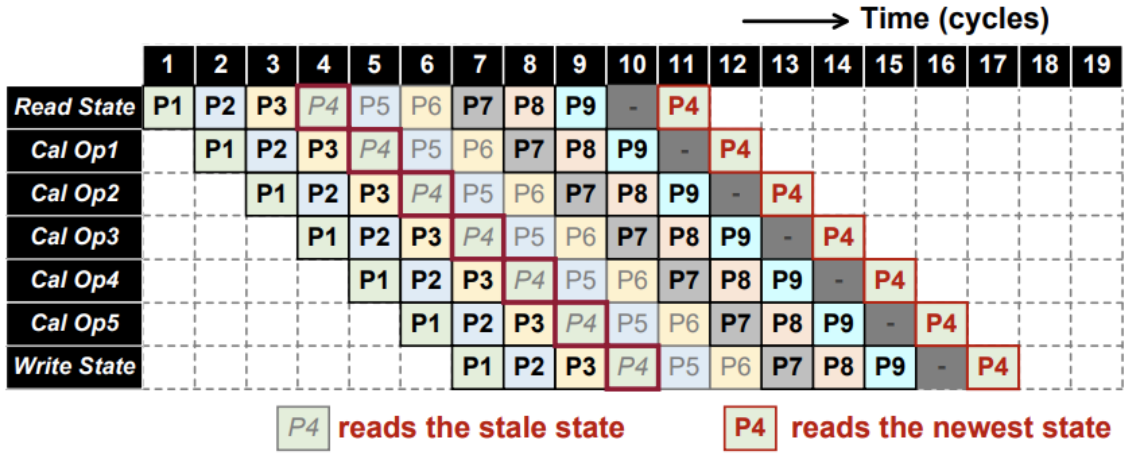


New Execution Mode – Speculative Execution



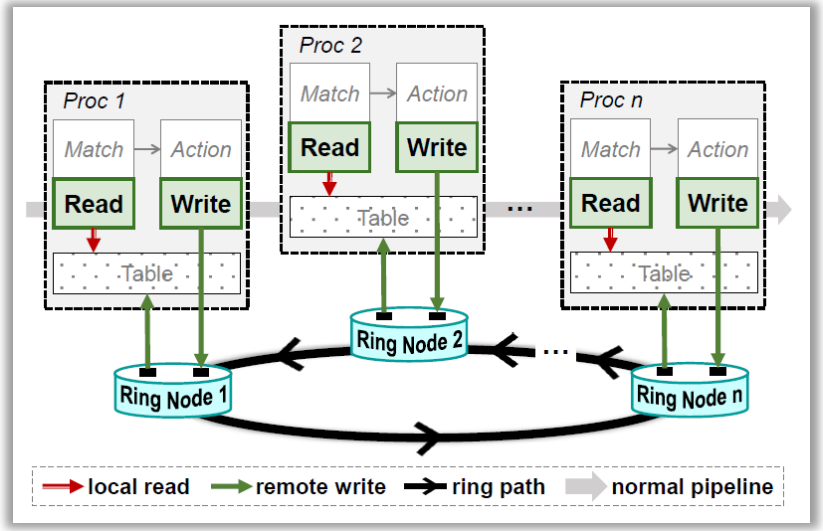
Predict packets won't modify states;
if states change, take measures to recover.

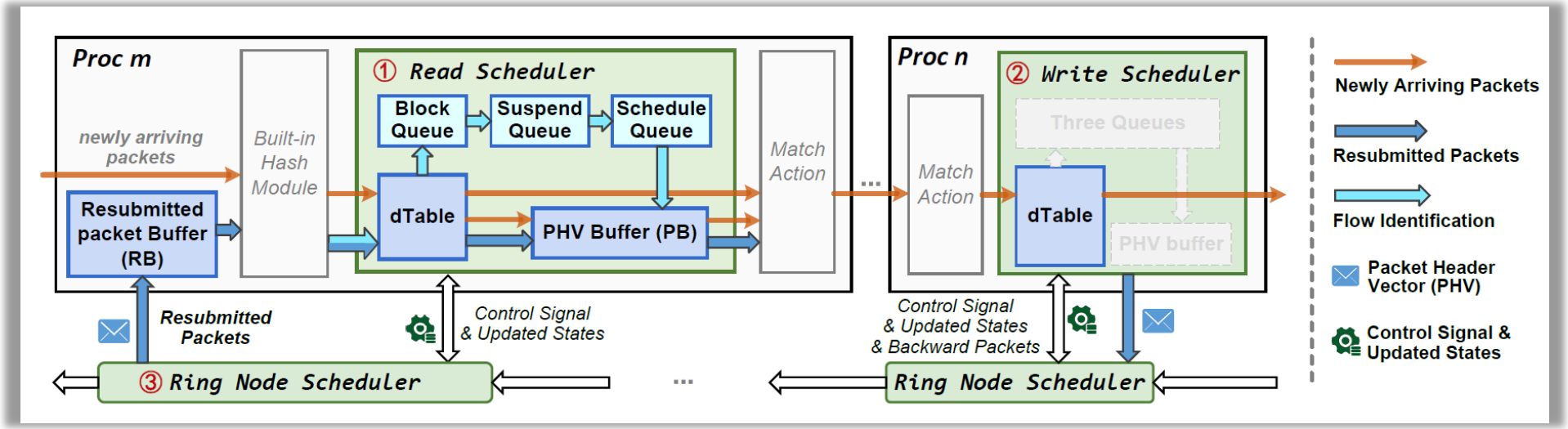




Processing packets in non-blocking mode saves hardware processing cycles!

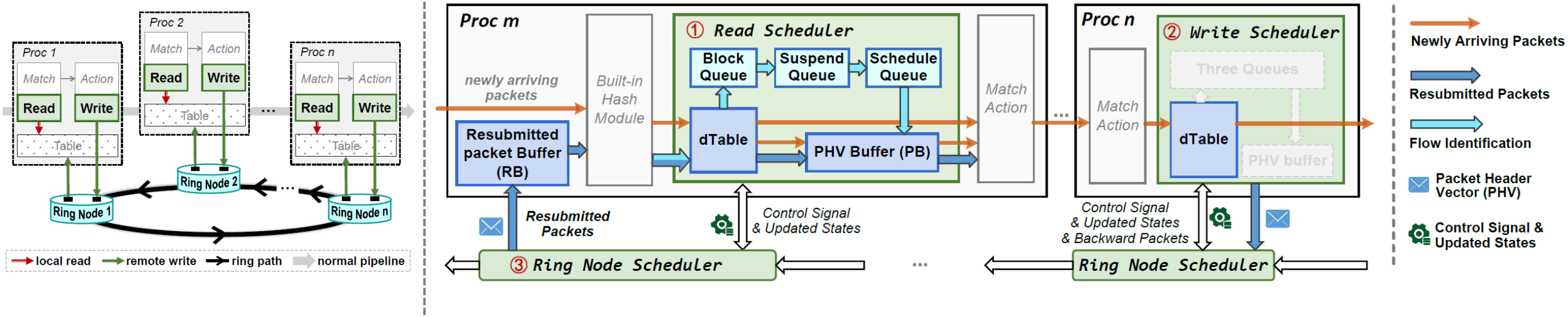
New Architecture – RAPID (Ring-Augmented PIpeline Dataplane)





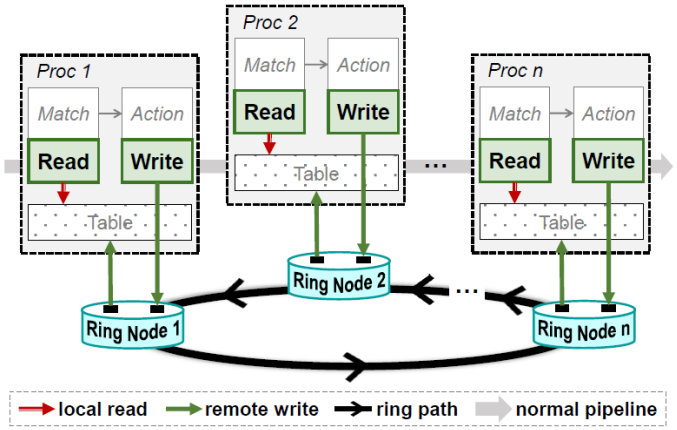
New Architecture – RAPID (Ring-Augmented PIpeline Dataplane)

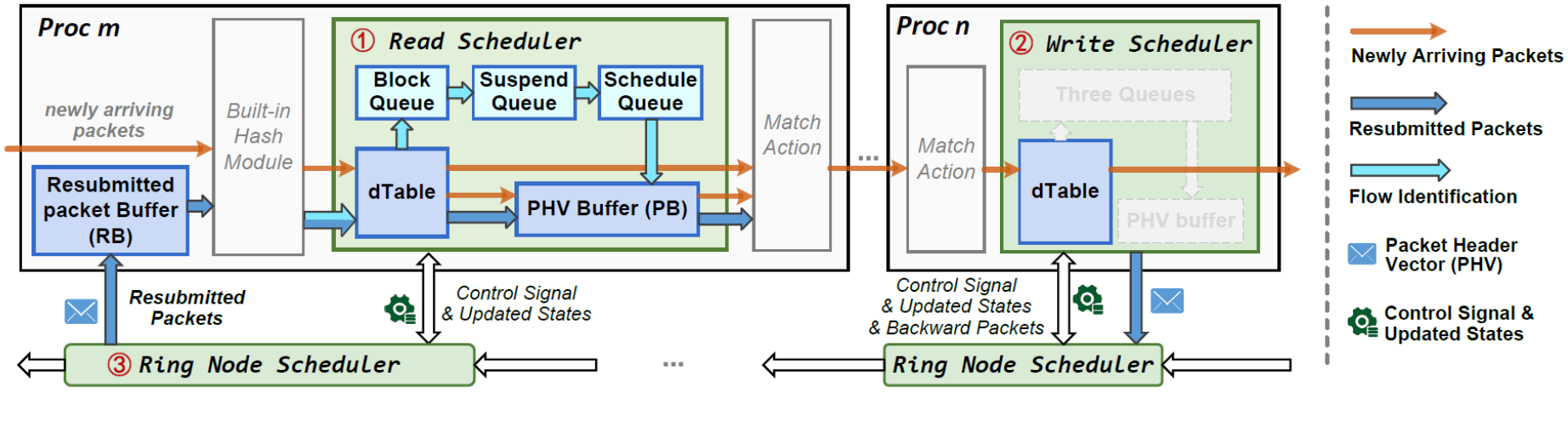
✓ Write back updated states; **Side Ring**



New Architecture – RAPID (Ring-Augmented PIpeline Dataplane)

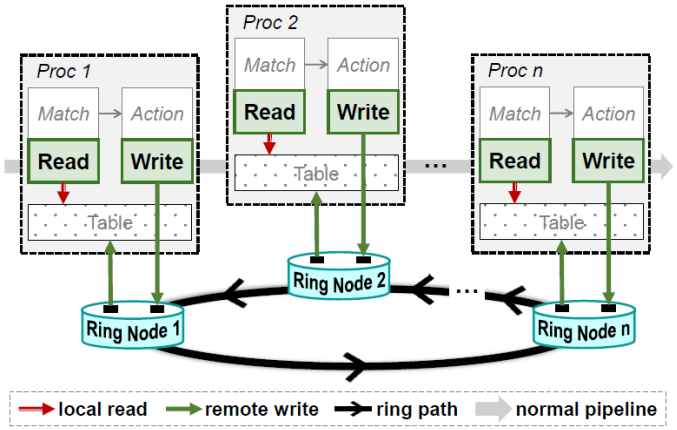
- ✓ Write back updated states; **Side Ring**
- ✓ Handle packets with stale states; **Resubmitted with Side Ring**

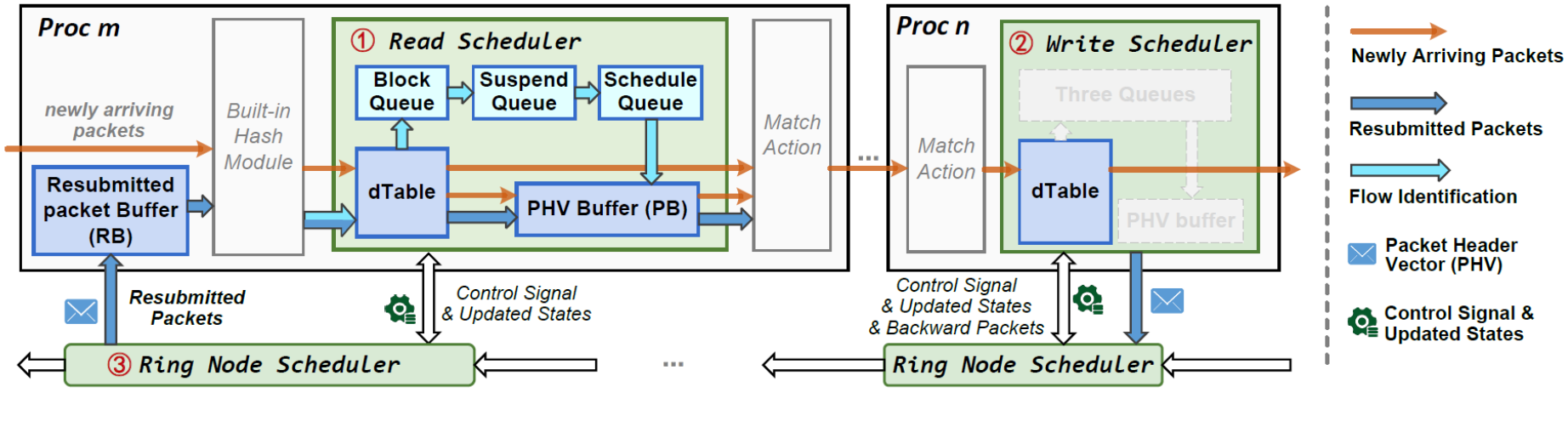




New Architecture – RAPID (Ring-Augmented PIpeline Dataplane)

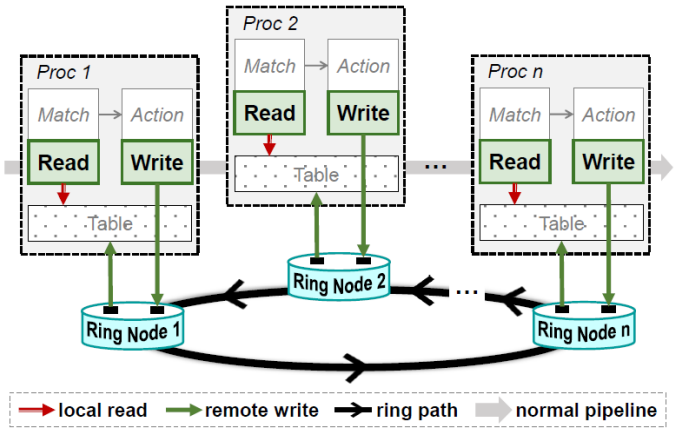
- ✓ Write back updated states; **Side Ring**
- ✓ Handle packets with stale states; **Resubmitted with Side Ring**
- ✓ Maintain state consistency and in-order processing; **Scheduling Mechanism**

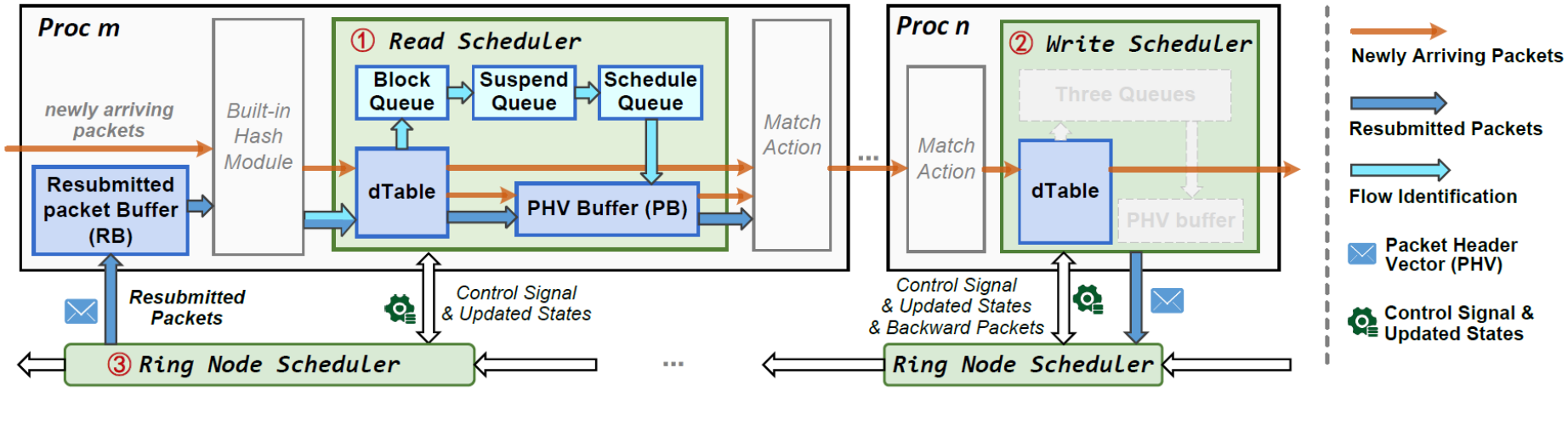




New Architecture – RAPID (Ring-Augmented PIpeline Dataplane)

- ✓ Write back updated states; **Side Ring**
- ✓ Handle packets with stale states; **Resubmitted with Side Ring**
- ✓ Maintain state consistency and in-order processing; **Scheduling Mechanism**
- ✓ Support different levels of consistency. (Swish [NSDI 2022]) **Finite State Machine**

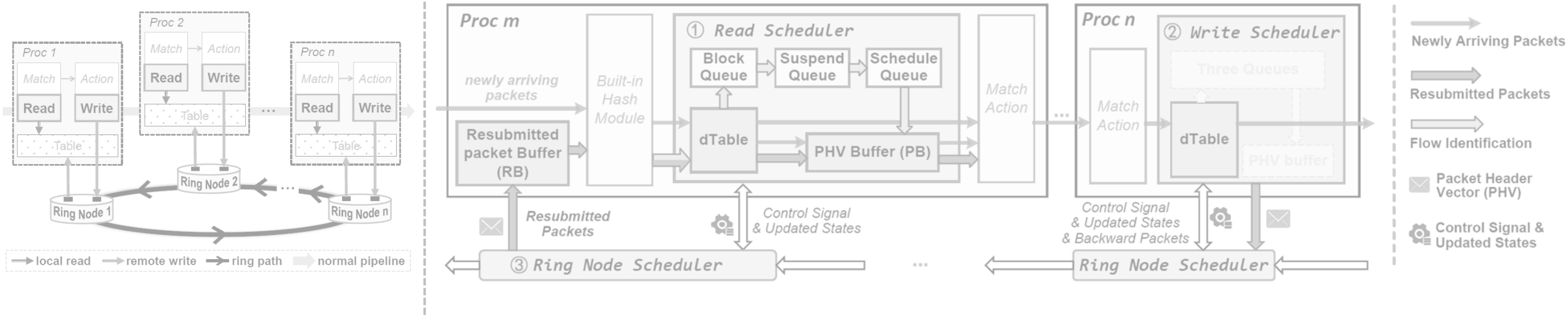




New Architecture – RAPID (Ring-Augmented PIpeline Dataplane)

- ✓ Write back updated states; **Side Ring**
- ✓ Handle packets with stale states; **Resubmitted with Side Ring**
- ✓ Maintain state consistency and in-order processing; **Scheduling Mechanism**
- ✓ Support differen **State Machine**

Refer to the paper for specific implementation details.



Language Enhancement

Stateful table definition

```
/* define stateful table */
muTable port_knocking {
    keys = {
        hdr.ipv4.src_addr;
        hdr.ipv4.dst_addr;
    }

    values = {
        bit<8> state;
    }

    type = exact;
    consistency = STRICT;
    size = 4096;
}

table port_FSM {
    keys = {
        meta.cur_state;
        hdr.ipv4.dst_port;
    };

    actions = {
        get_new_state; // modify meta.new_state
    };
}
```

Stateful operations

```
/* read out the cur_state */
meta.cur_state = port_knocking.read(hdr);
/* get new_state to look up table */
portFSM.apply();
/* write back the new_state */
port_knocking.write(hdr, meta.new_state);
```

Evaluation - Resources

- Four Prototypes with Scala/Chisel
  - PISA (SIGCOMM 2013) --2,584 LOCs
  - Banzai (SIGCOMM 2017) --2,292 LOCs
  - FlowBlaze (NSDI 2019) --3,627 LOCs
  - **RAPID** --4,676 LOCs
- Synthesized with FPGA and an open-source technology library

Evaluation - Resources

- Four Prototypes with Scala/Chisel
  - PISA (SIGCOMM 2013) --2,584 LOCs
  - Banzai (SIGCOMM 2017) --2,292 LOCs
  - FlowBlaze (NSDI 2019) --3,627 LOCs
  - **RAPID** --4,676 LOCs
- Synthesized with FPGA and an open-source technology library

Architecture	LUT	FF	BRAM
PISA	13.91%	1.71%	14.08%
Banzai	15.77%	1.73%	14.08%
FlowBlaze	27.32%	2.48%	25.99%
RAPID	20.22%	2.35%	19.55%

Functionality stronger than FlowBlaze.
FPGA overhead is smaller.

Evaluation - Resources

- Four Prototypes with Scala/Chisel
  - PISA (SIGCOMM 2013) --2,584 LOCs
  - Banzai (SIGCOMM 2017) --2,292 LOCs
  - FlowBlaze (NSDI 2019) --3,627 LOCs
  - **RAPID** --4,676 LOCs
- Synthesized with FPGA and an open-source technology library

Architecture	**LUT**	**FF**	**BRAM**
PISA	13.91%	1.71%	14.08%
Banzai	15.77%	1.73%	14.08%
FlowBlaze	27.32%	2.48%	25.99%
RAPID	20.22%	2.35%	19.55%

Functionality stronger than FlowBlaze.
FPGA overhead is smaller.

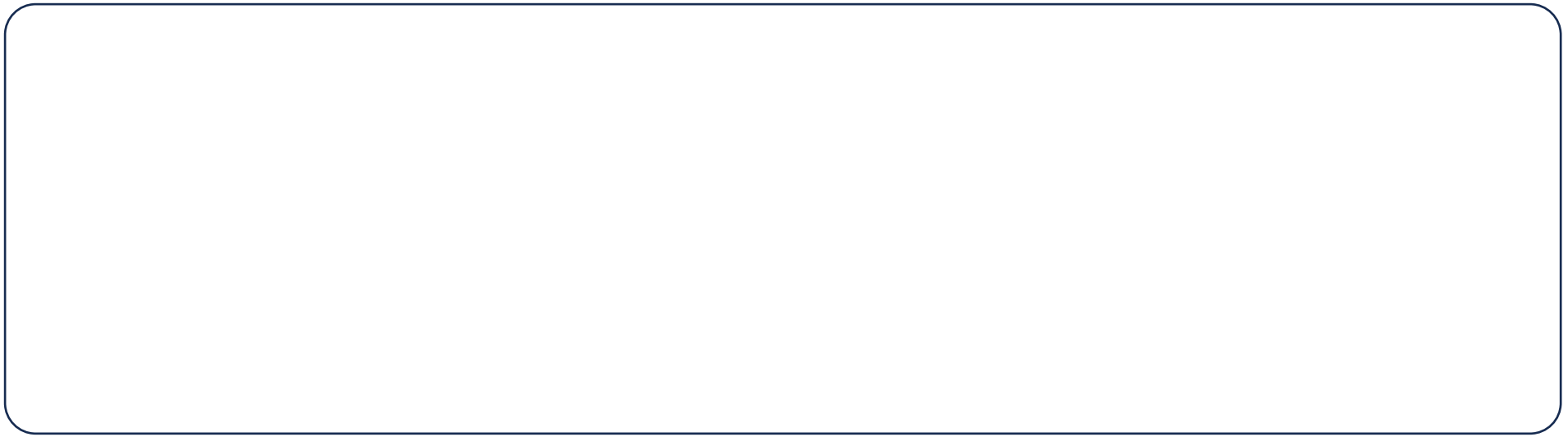
	Area (mm^2)	**Power** (mW)
PISA	94.33	65000
Banzai	95.17	66100
FlowBlaze	176.08	86900
RAPID	99.45	67500

Lower ASIC area and power overhead vs. baseline, better than FlowBlaze.

Evaluation – Performance (with ASIC simulator)

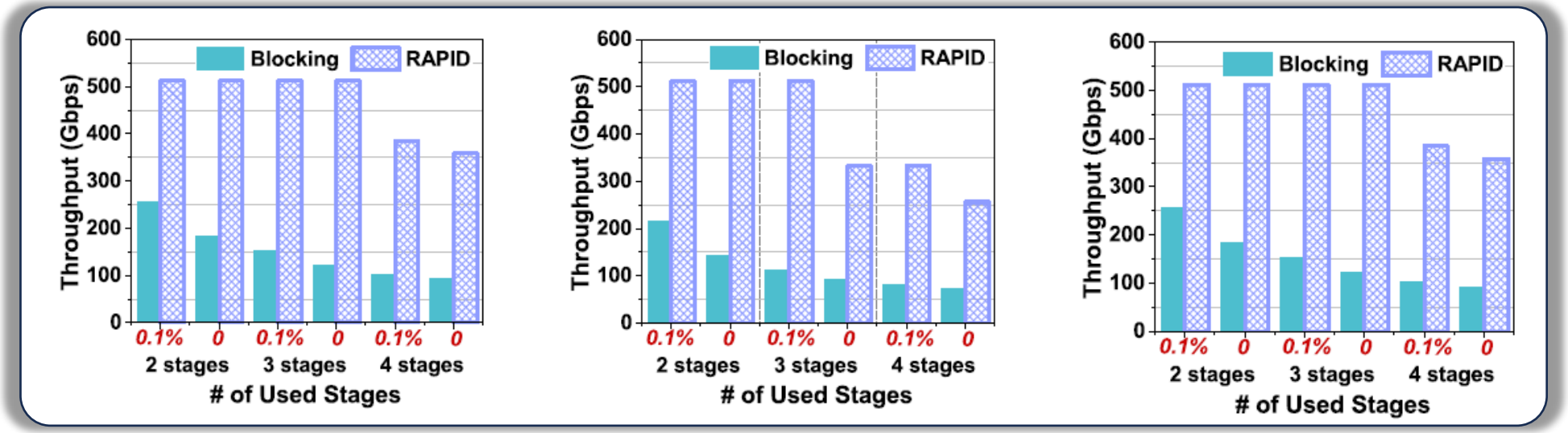
Throughput

Trace 1 Trace 2 Trace 3



Latency

Evaluation – Performance (with ASIC simulator)

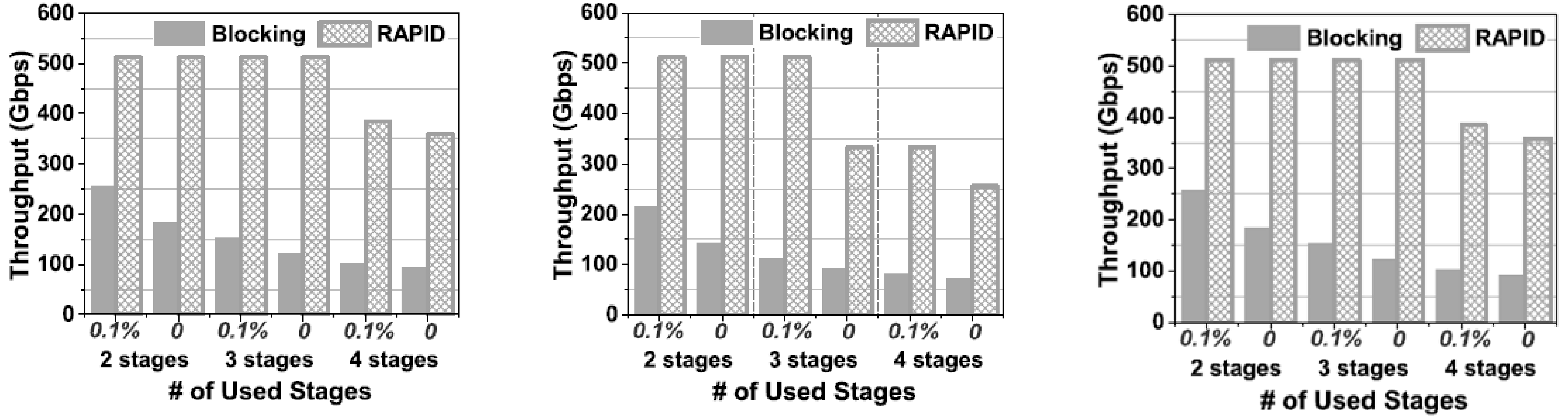


Trace 1 Trace 2 Trace 3

Throughput increased by 2 to 3 times!

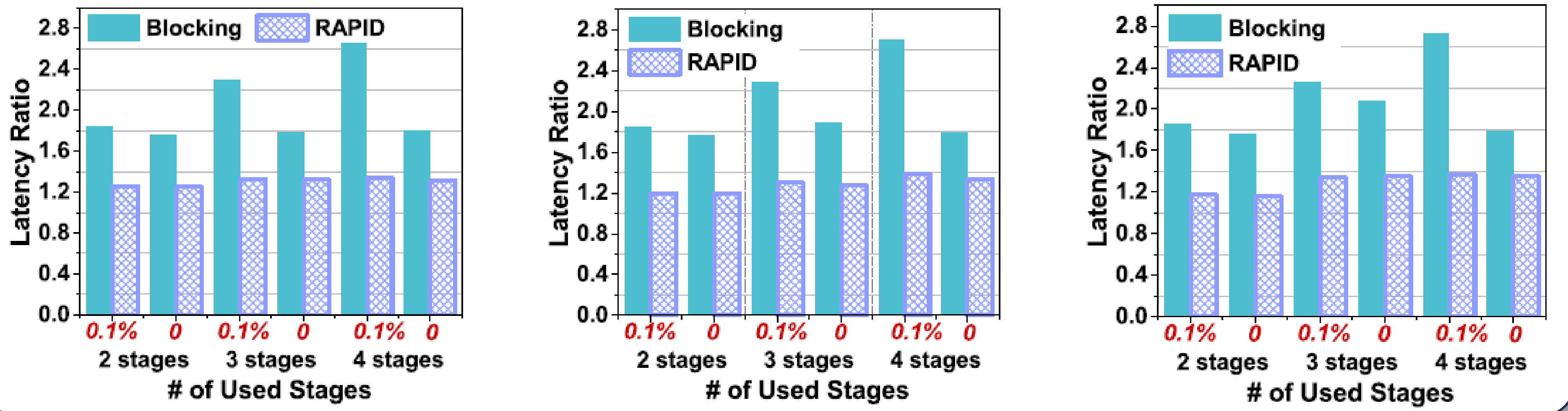
Latency

Evaluation – Performance (with ASIC simulator)



Trace 1 | Trace 2 | Trace 3

Throughput increased by 2 to 3 times!



Latency reduced by 1 to 2 times!

Conclusion

- Proposed an abstraction of **dataplane writable table**;
- Introduced the **speculative execution** mode with stateful network functions;
- Implemented the abstraction and execution mode using the **RAPID architecture**;
- Enhanced the P4 language to support complex stateful functions.

Conclusion

- Proposed an abstraction of **dataplane writable table**;
- Introduced the **speculative execution** mode with stateful network functions;
- Implemented the abstraction and execution mode using the **RAPID architecture**;
- Enhanced the P4 language to support complex stateful functions.

Thank you!