Shale: A Practical, Scalable Oblivious Reconfigurable Network

Daniel Amir*, Tegan Wilson*, Nitika Saran*, Robert Kleinberg*, Vishal Shrivastav†, **Hakim Weatherspoon***

*Cornell University †Purdue University

Reconfigurable Networks Workshop

June 2, 2025

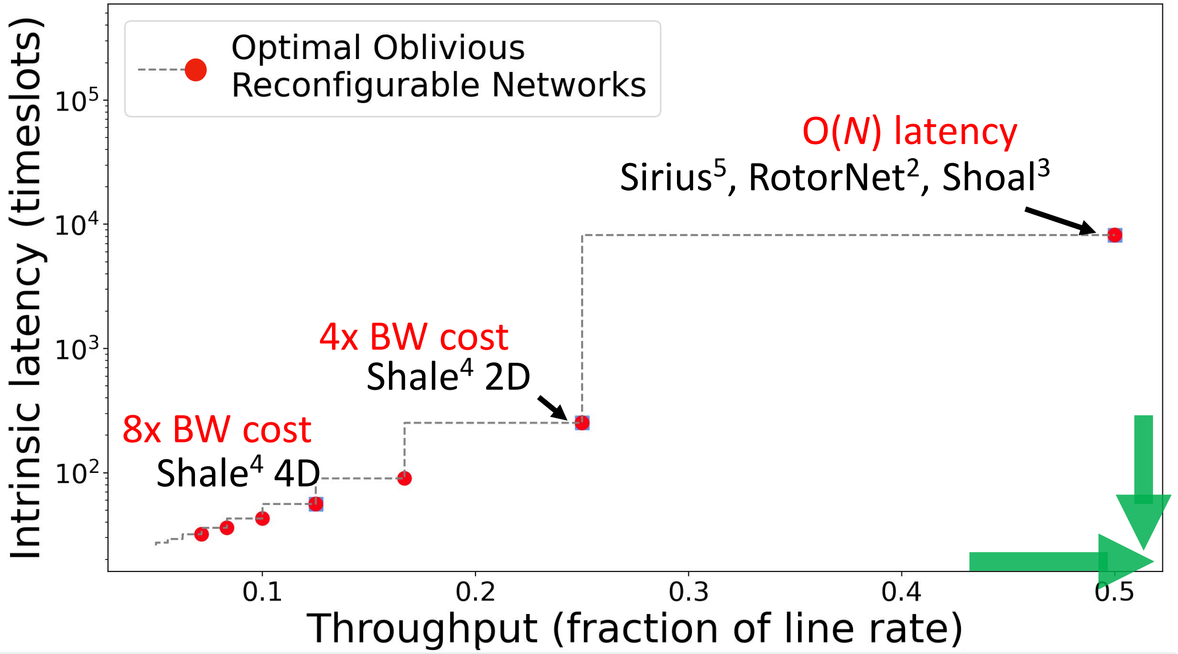
Originally presented at SIGCOMM 2024

Can we design datacenter networks that use only circuit switches?

Oblivious Reconfigurable Networks (ORNs)

A Fundamental tradeoff:

- Pareto-optimal for oblivious designs[1]



[1]Amir et al, STOC 2022

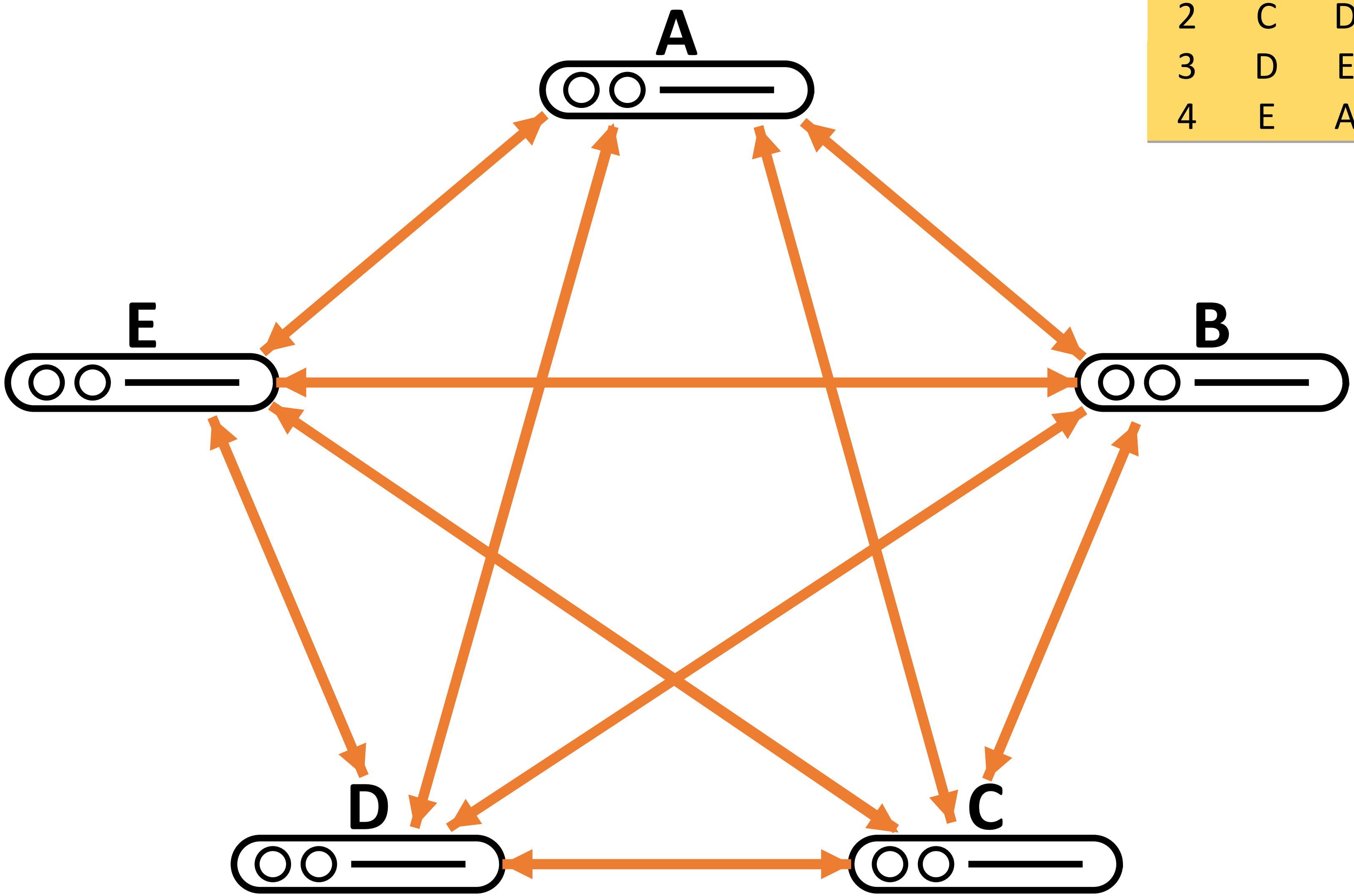
[2]Mellette et al., SIGCOMM 2017 [3]Shrivastav et al., NSDI 2019

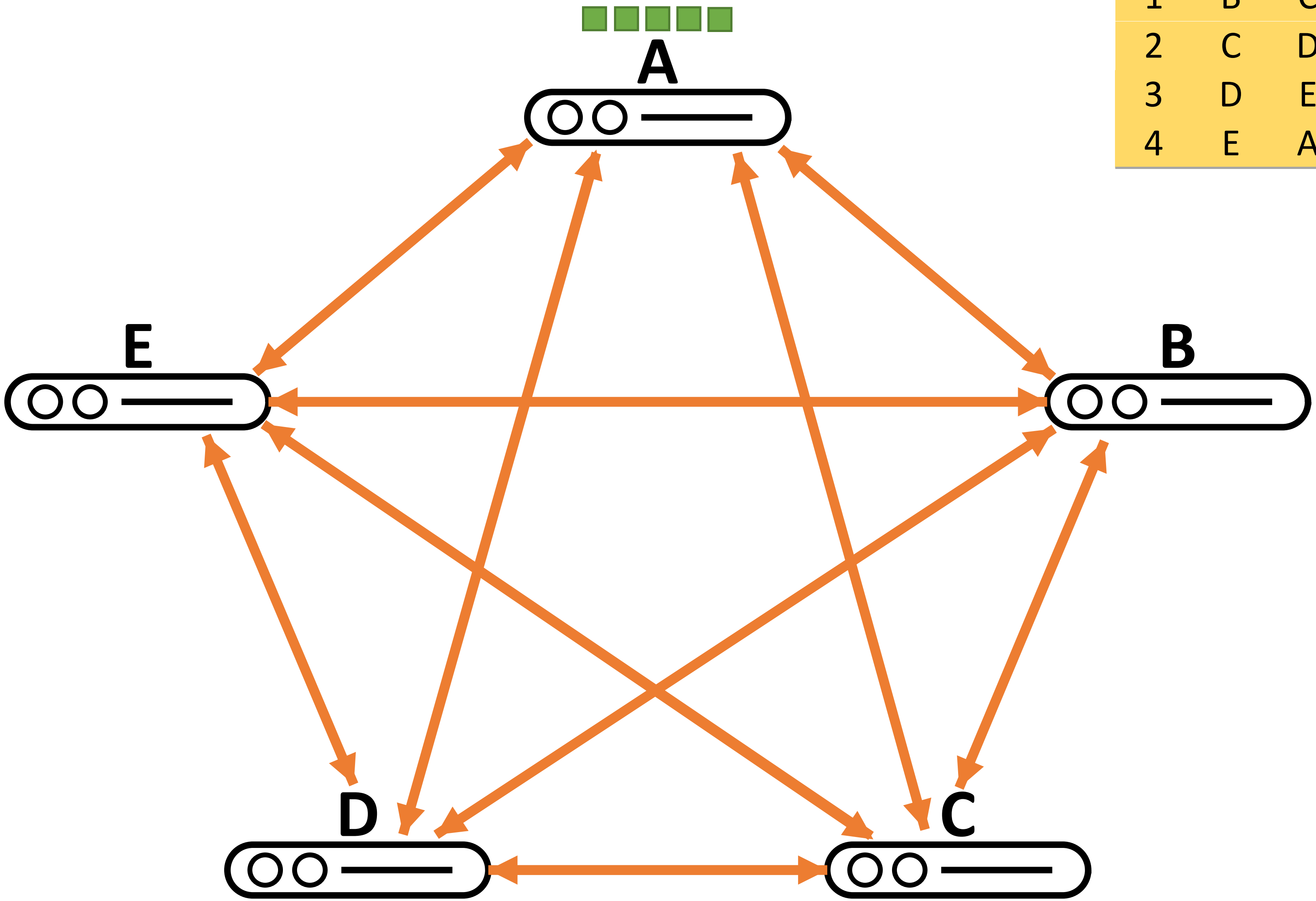
[4]Amir et al, SIGCOMM 2024 [5]Ballani et al, SIGCOMM 2020

Existing ORNs: Rotornet, Shoal, Sirius

- Schedule is a round-robin
- Send via an intermediate node
    - Valiant Load Balancing (VLB)

	A	B	C	D	E
1	B	C	D	E	A
2	C	D	E	A	B
3	D	E	A	B	C
4	E	A	B	C	D

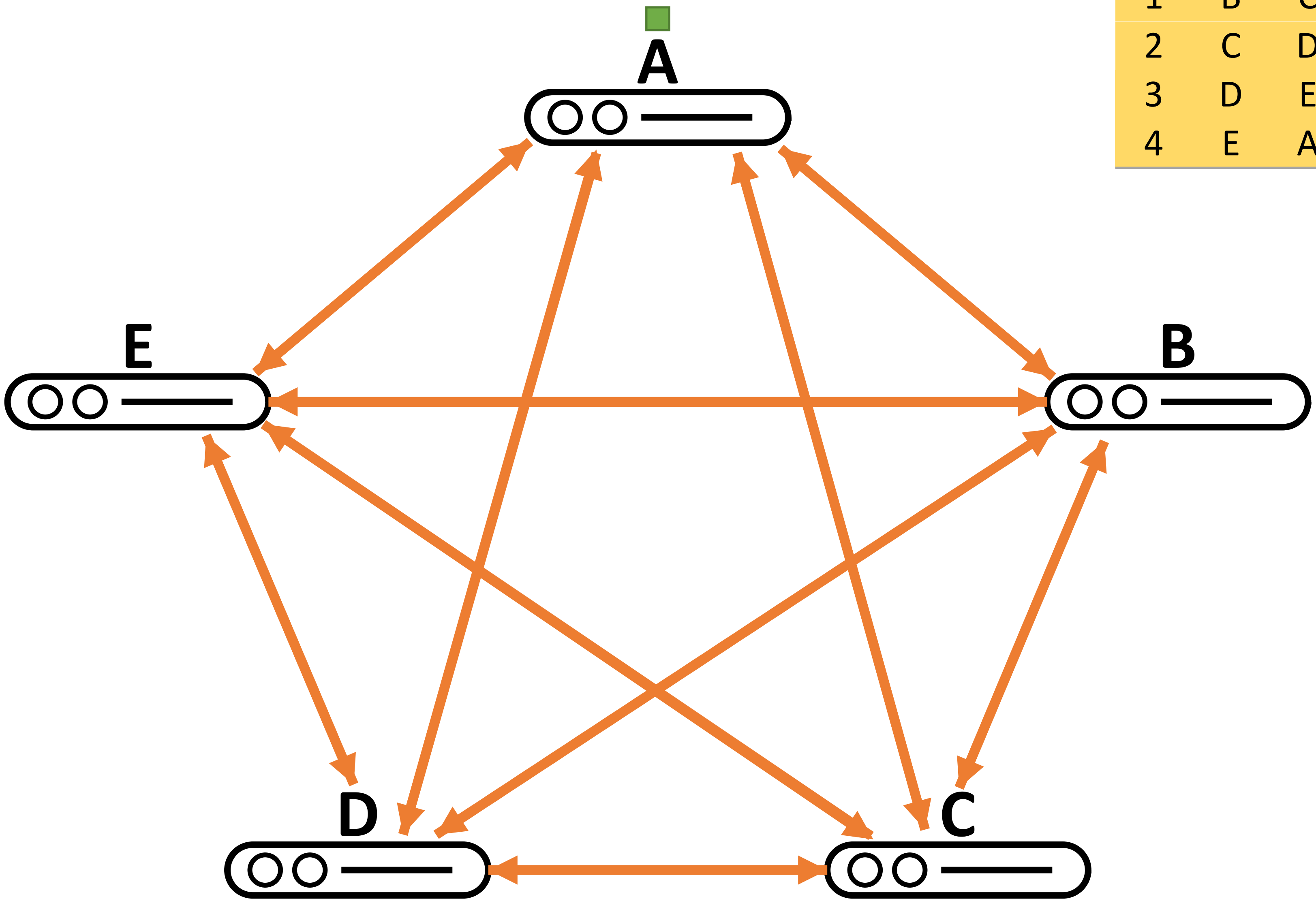




	A	B	C	D	E
1	B	C	D	E	A
2	C	D	E	A	B
3	D	E	A	B	C
4	E	A	B	C	D

Existing ORNs: Rotornet, Shoal, Sirius

- Schedule is a round-robin
- Send via an intermediate node
  - Valiant Load Balancing (VLB)
- Throughput of at least **½ of line rate**, regardless of traffic
- **Latency is O(N)**, so scaling is poor



	A	B	C	D	E
1	B	C	D	E	A
2	C	D	E	A	B
3	D	E	A	B	C
4	E	A	B	C	D

Existing ORNs: Rotornet, Shoal, Sirius

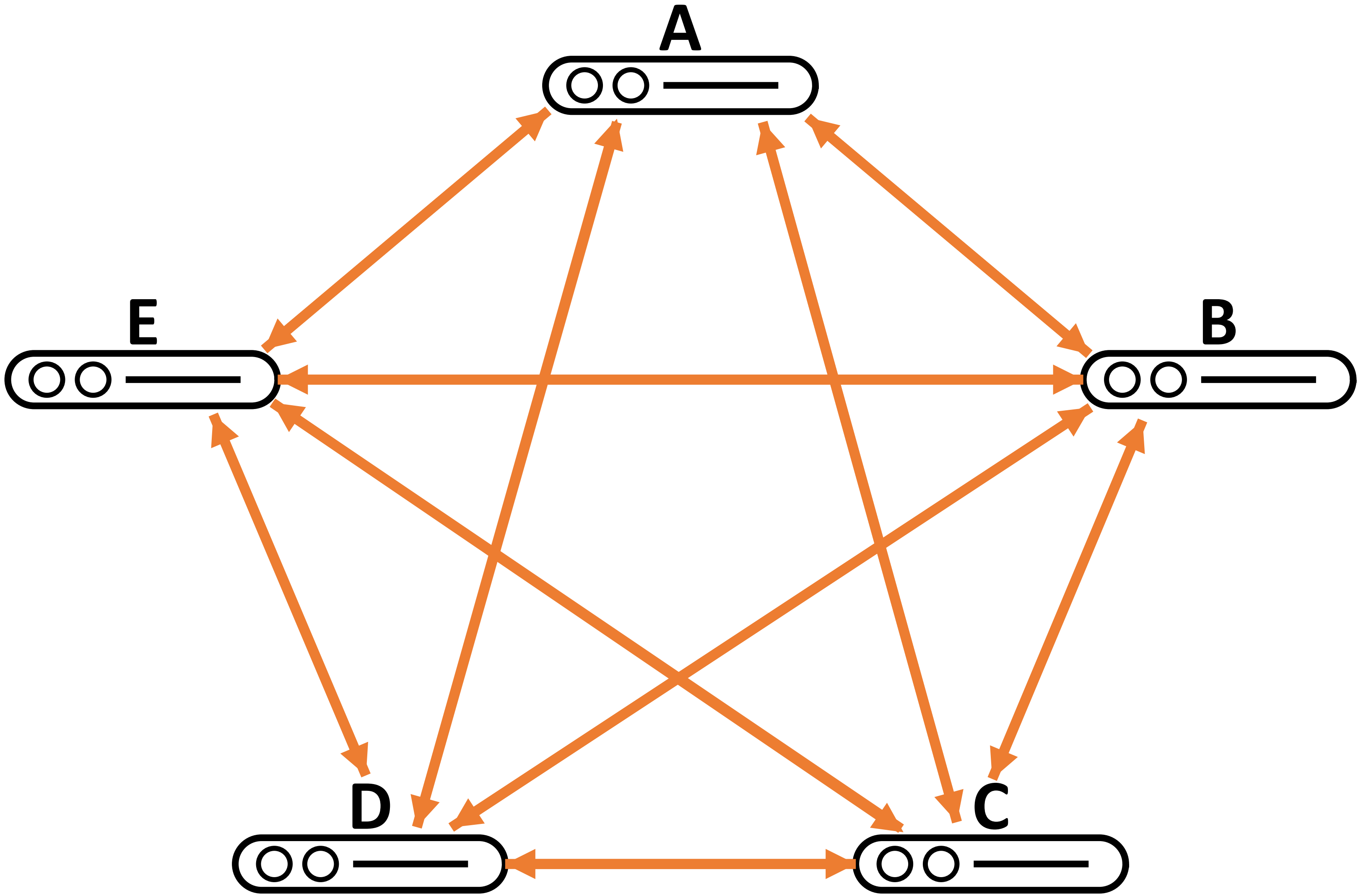
- Schedule is a round-robin
- Send via an intermediate node
  - Valiant Load Balancing (VLB)
- Throughput of at least **½ of line rate**, regardless of traffic
- **Latency is O(N)**, so scaling is poor
  - After first hop, you may need to wait for an entire round-robin.
- For a 100,000-server datacenter and 5ns timeslots, latency is 500 µs
  - Also requires multiple GB of on-chip memory
- **Can we do better?**

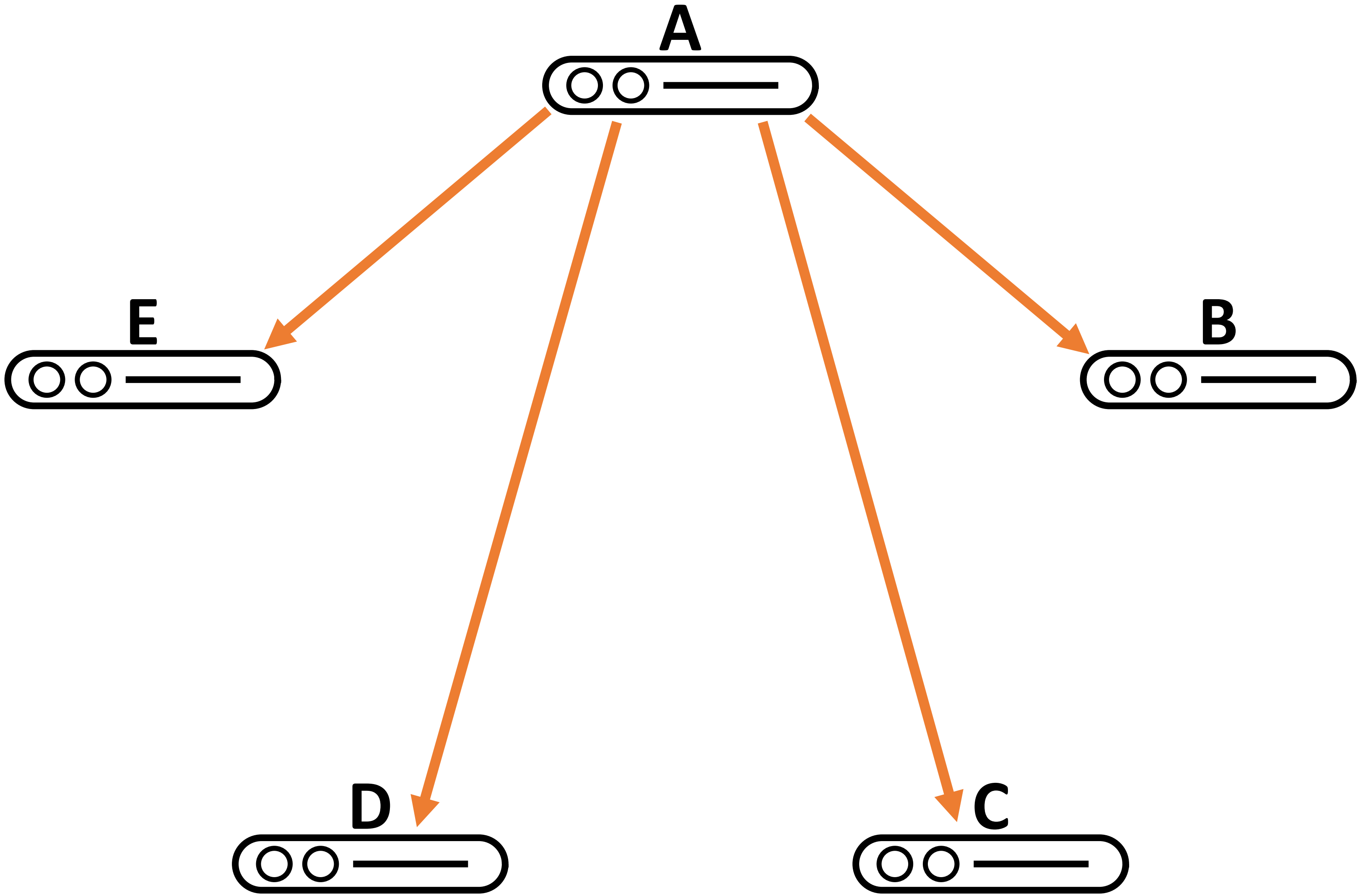
Our Contributions

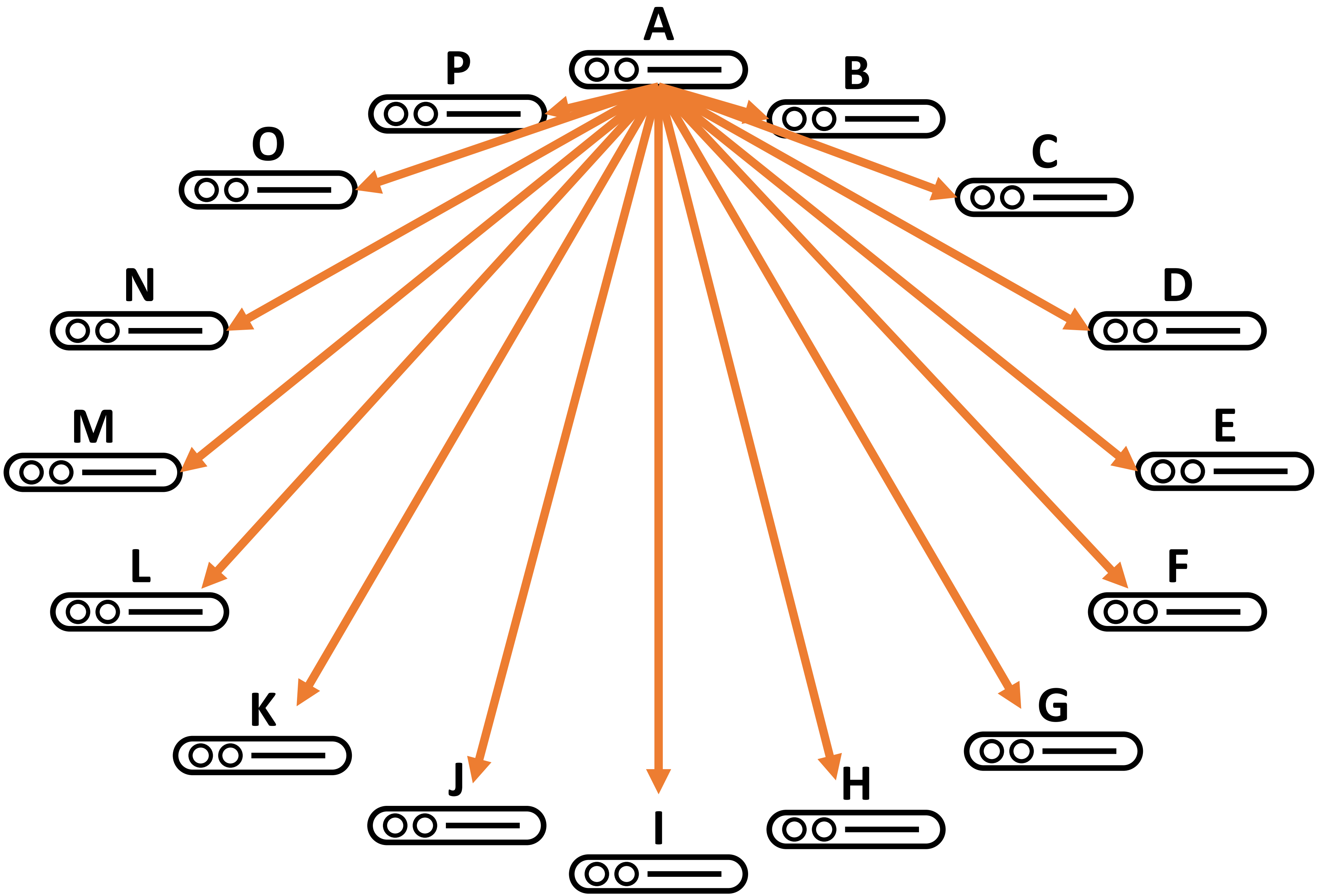
- Shale: a new ORN design that supports tens of thousands of nodes.
- Orders of magnitude better latency and memory requirements than existing designs at these scales
- New schedules bring tunable tradeoff between throughput and latency
  - Each tradeoff is Pareto optimal for ORNs!
- Enabled by a new congestion control
- Optimized FPGA-based hardware implementation (Session #4)
- Competitive for ML workloads (Session #7)
- Semi-oblivious (Session #1)
- Supports interleaving to combine multiple tunings

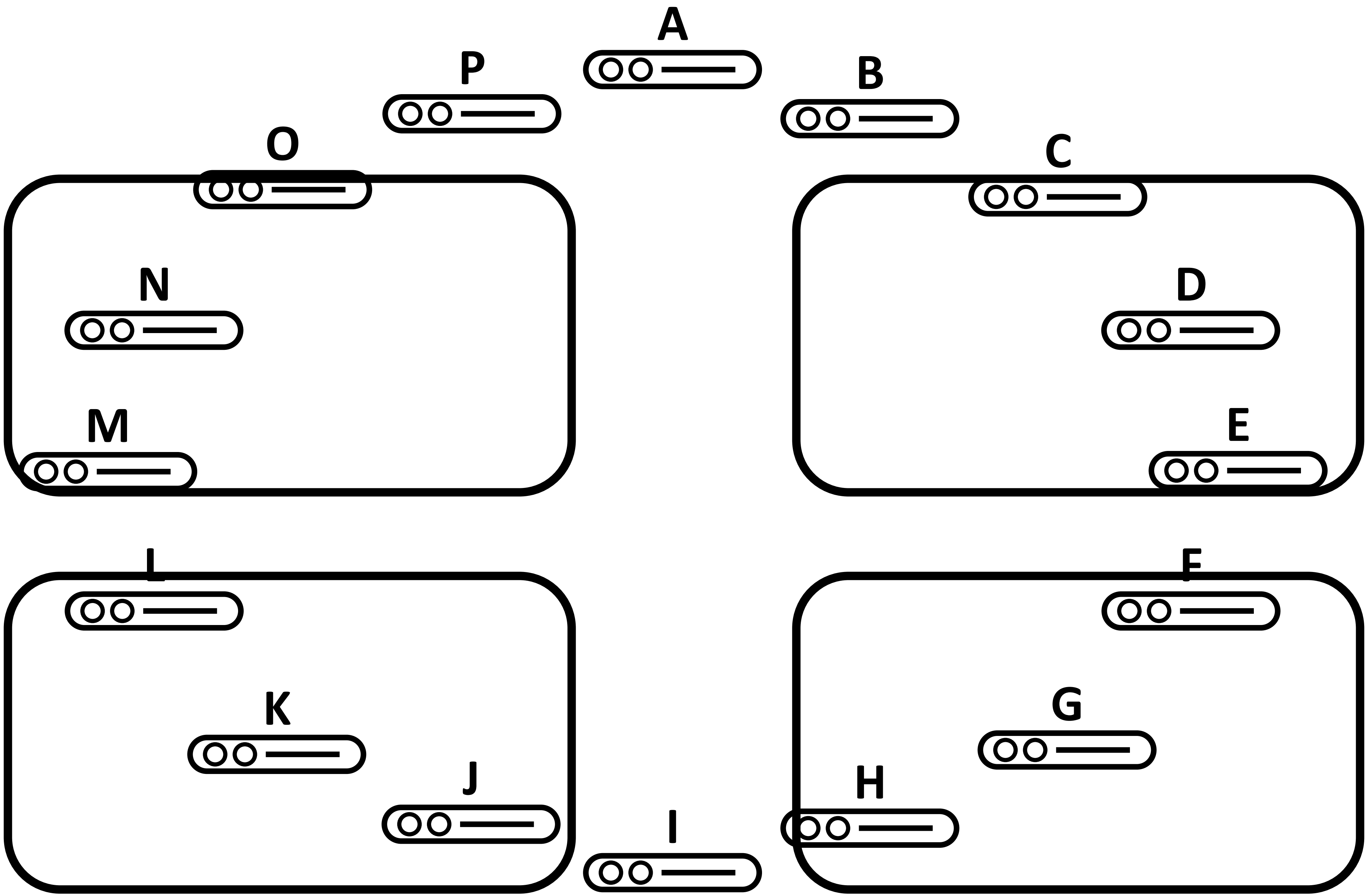
Shale Schedule and Routing

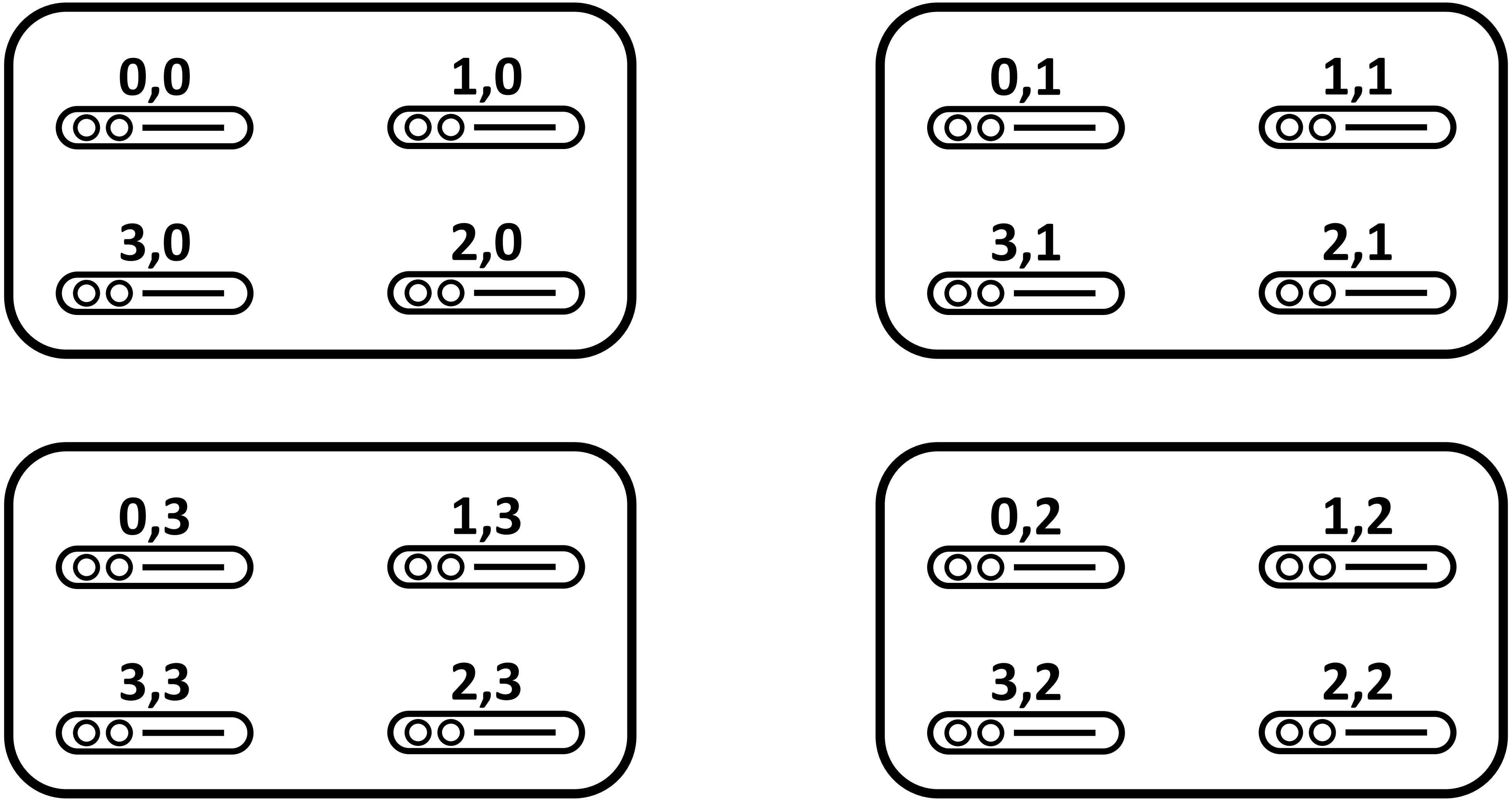
- Generalization of existing round-robin design
- Each node participates in h shorter round robins
  - Each round robin has just $\sqrt[h]{N}$ nodes


A
B
C
D
E

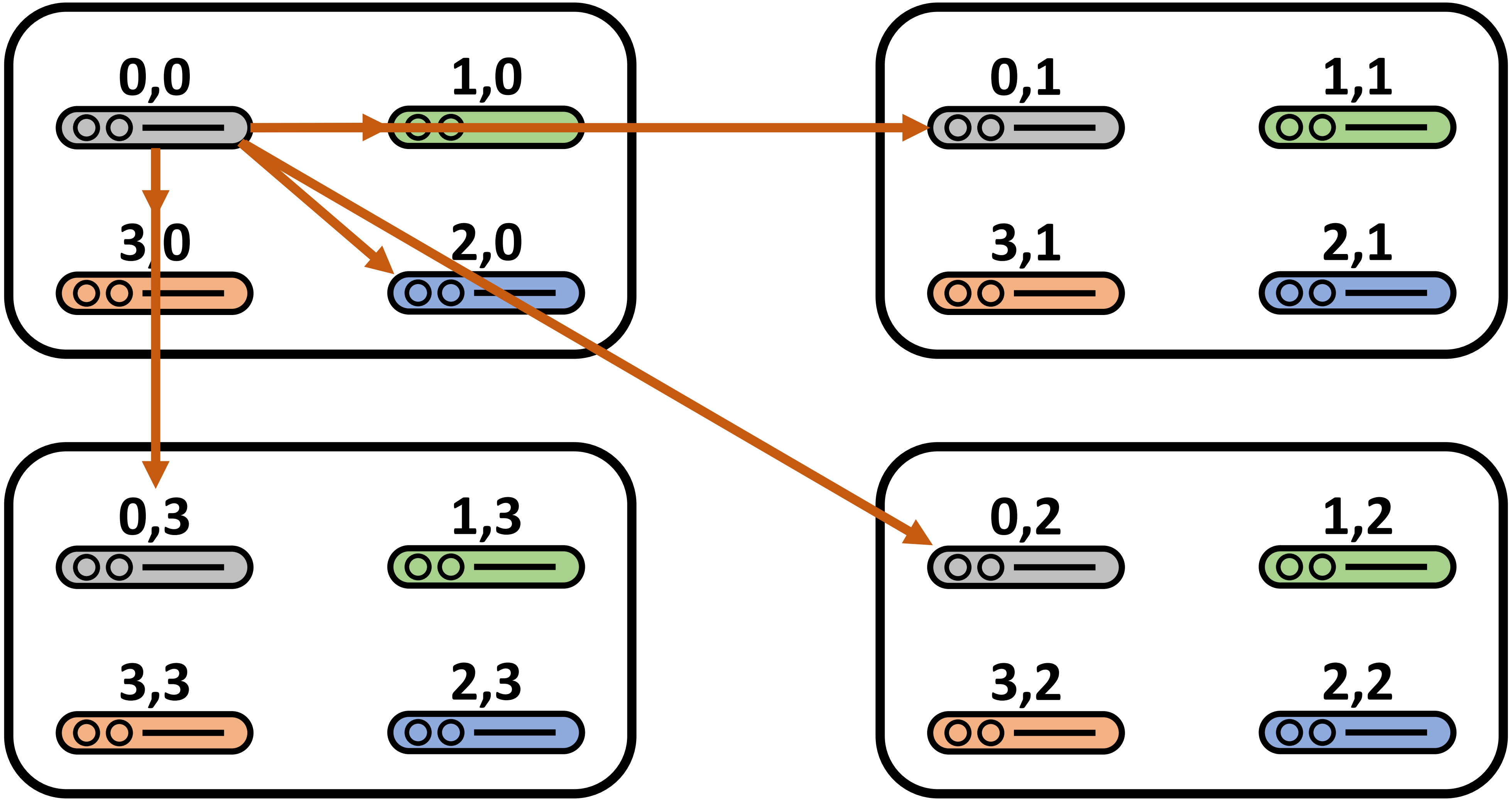

A
E
B
D
C


A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P

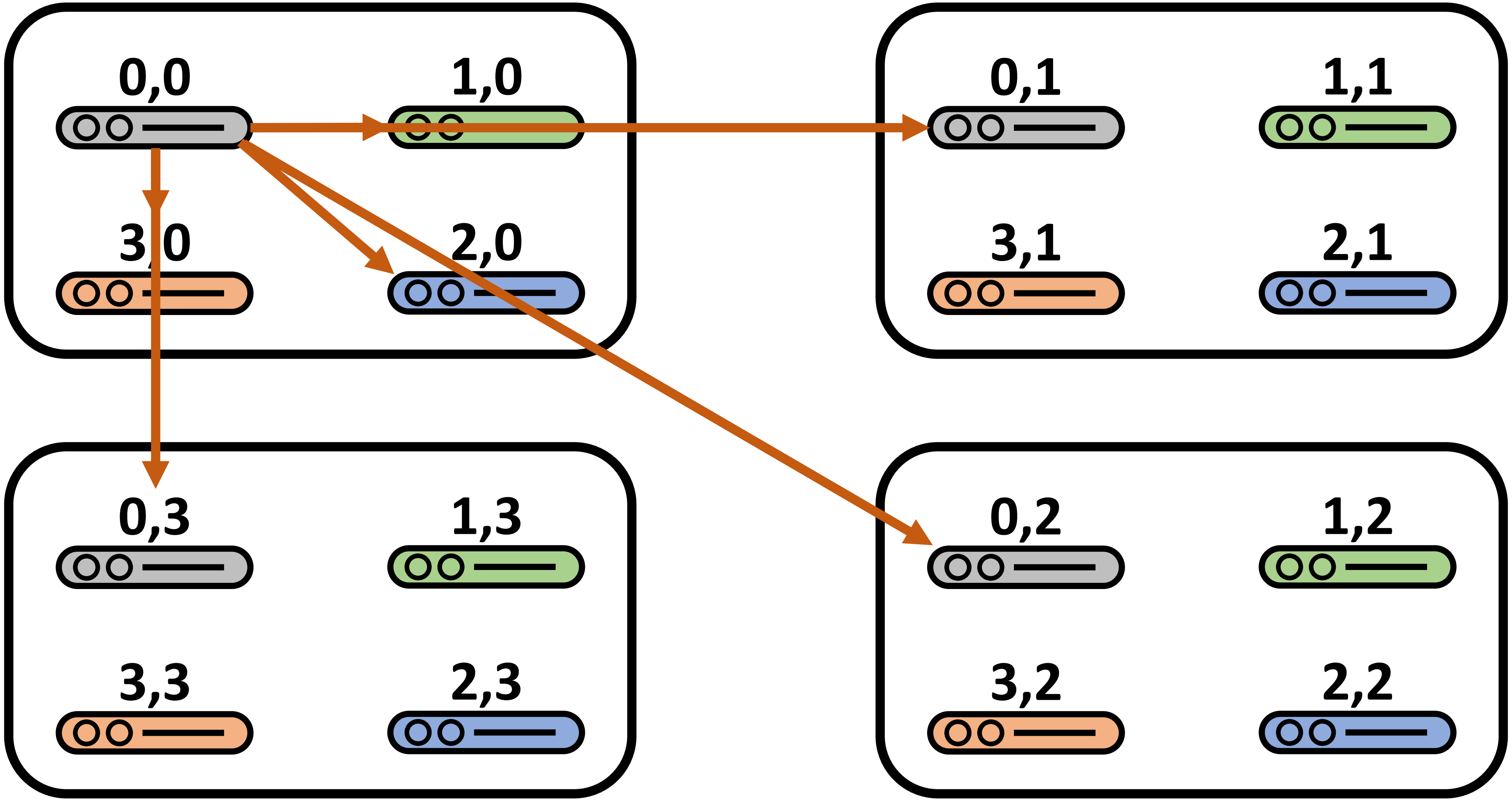

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P


0,0
1,0
3,0
2,0
0,1
1,1
3,1
2,1
0,3
1,3
3,3
2,3
0,2
1,2
3,2
2,2

Shale Schedule (h=2)



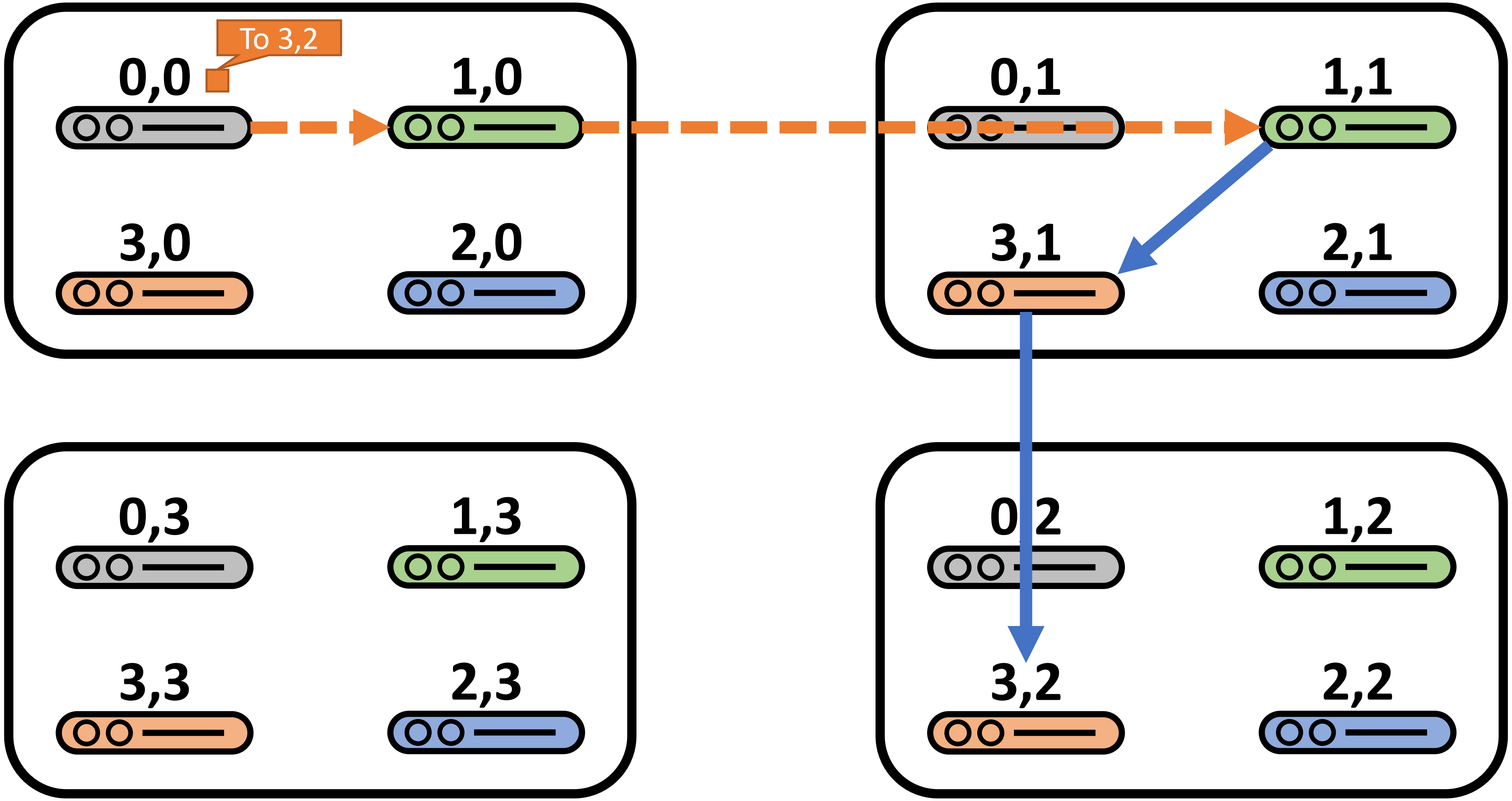
Shale Schedule (h=2)



Shale Schedule and Routing

- Generalization of existing round-robin design
- Each node participates in h shorter round robins
  - Each round robin has just $\sqrt[h]{N}$ nodes
- Direct paths between nodes are now h hops long
- Still use VLB

Shale Schedule (h=2)

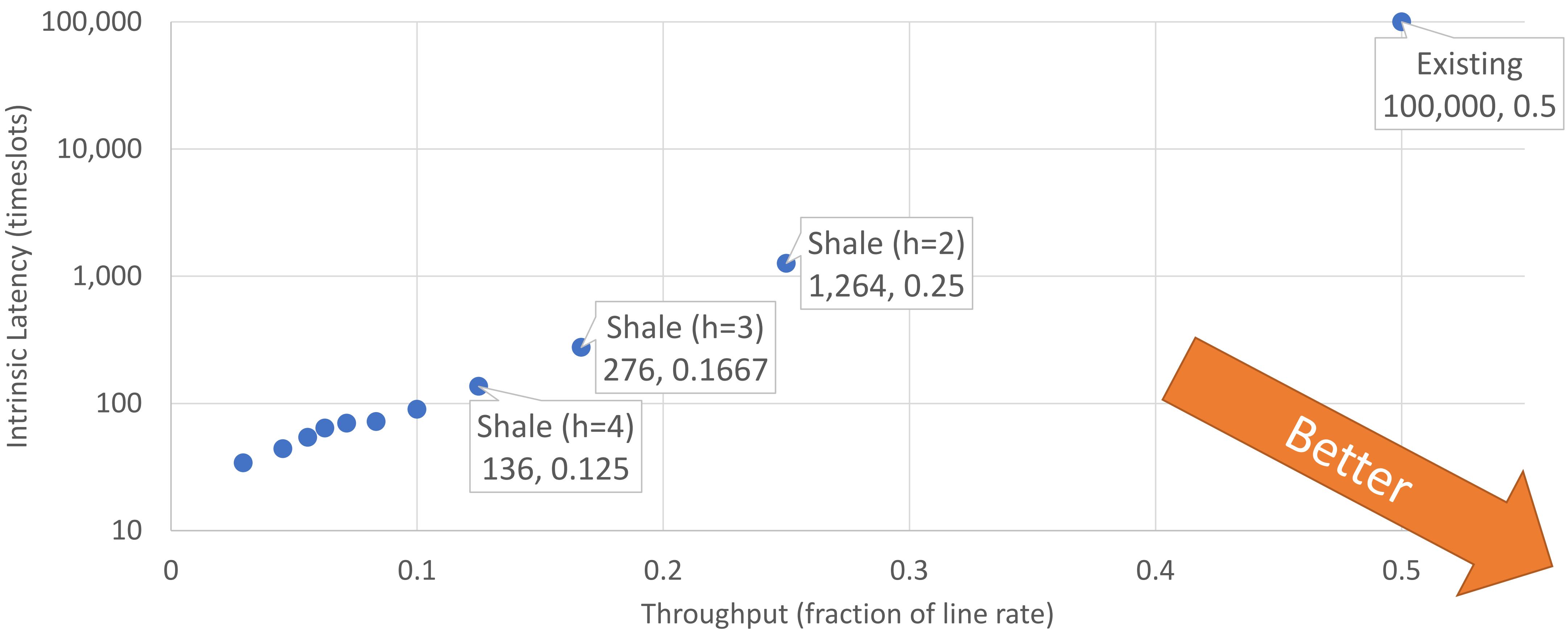


Shale Schedule and Routing

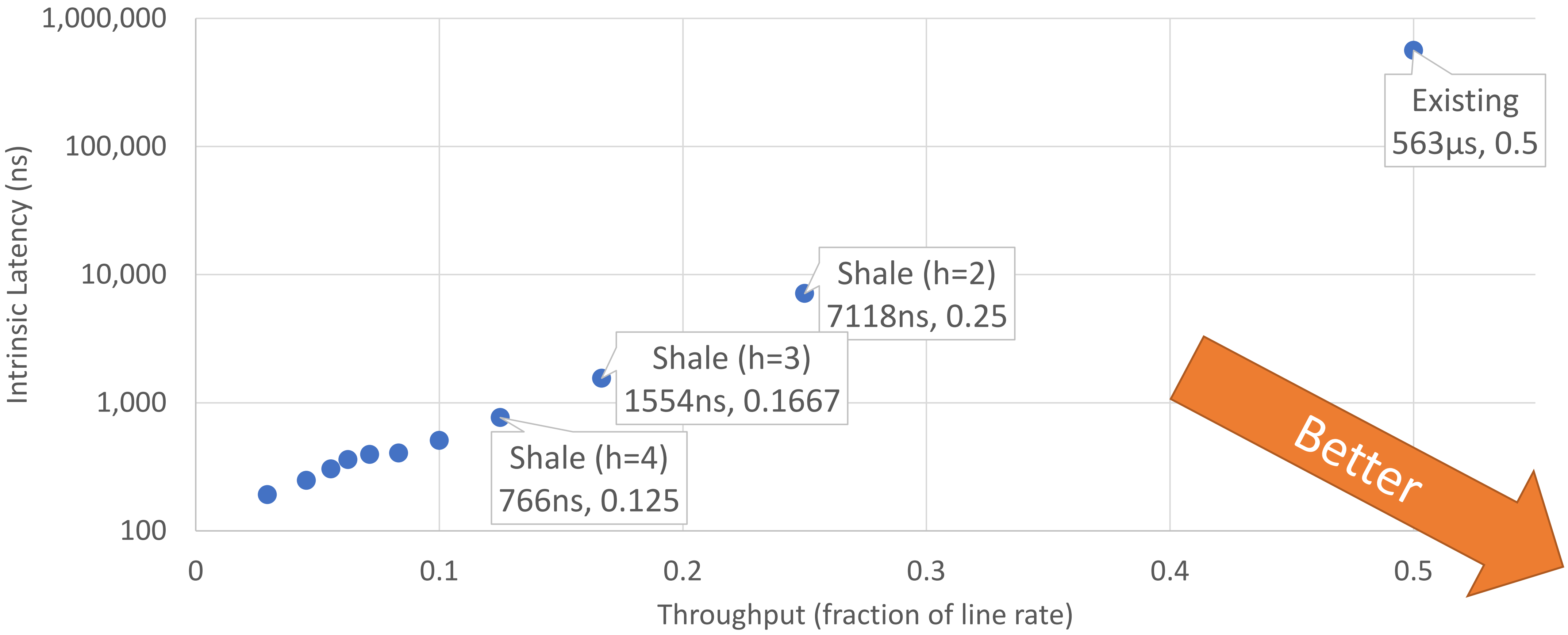
- Generalization of existing round-robin design
- Each node participates in h different round robins
  - Each round robin has just $\sqrt[h]{N}$ nodes
- Direct paths between nodes are now h hops long
- Still use VLB

- Latency is better! Now $O(h\sqrt[h]{N})$
- Throughput is worse. Now $^1/_{2h}$

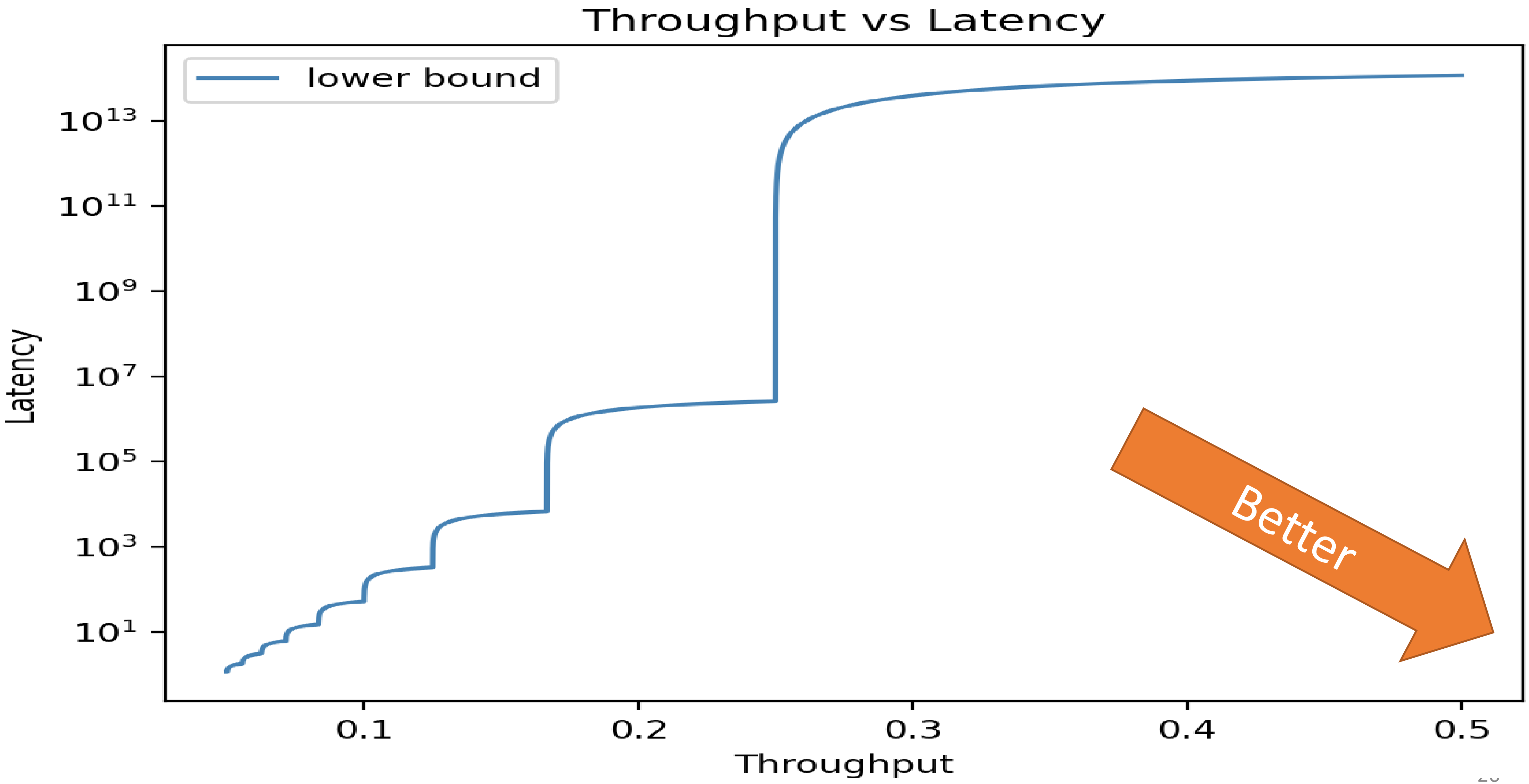
Comparison for 100,000 nodes



Comparison for 100,000 nodes



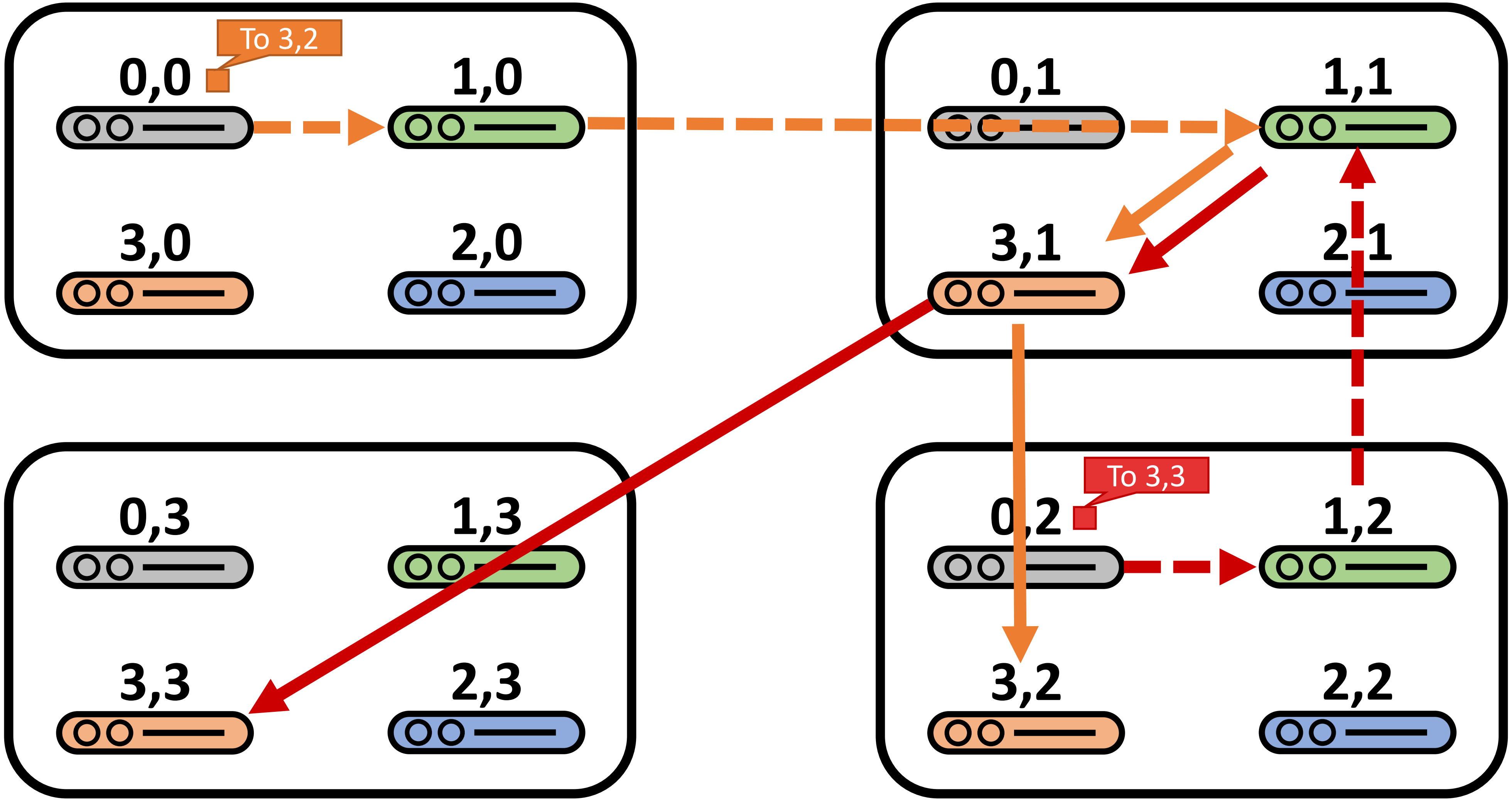
Each tradeoff is Pareto optimal for ORNs!



Optimal Oblivious Reconfigurable Networks: Summary

- **What are the best tradeoffs possible for ORNs?**
  - For an ORN that can sustain a given throughput for all traffic patterns, what is the lowest possible worst-case latency?
- **We have found:**
  - **A lower bound** (through theoretical analysis of ORNs)
  - **An upper bound** (by creating and analyzing ORN designs)
- **These bounds are tight!**
  - **STOC 2022**
- Our upper bound is based in part on a formalization of Shale's schedule, proving that **Shale is a Pareto optimal ORN**
  - SIGCOMM 2024, STOC 2022

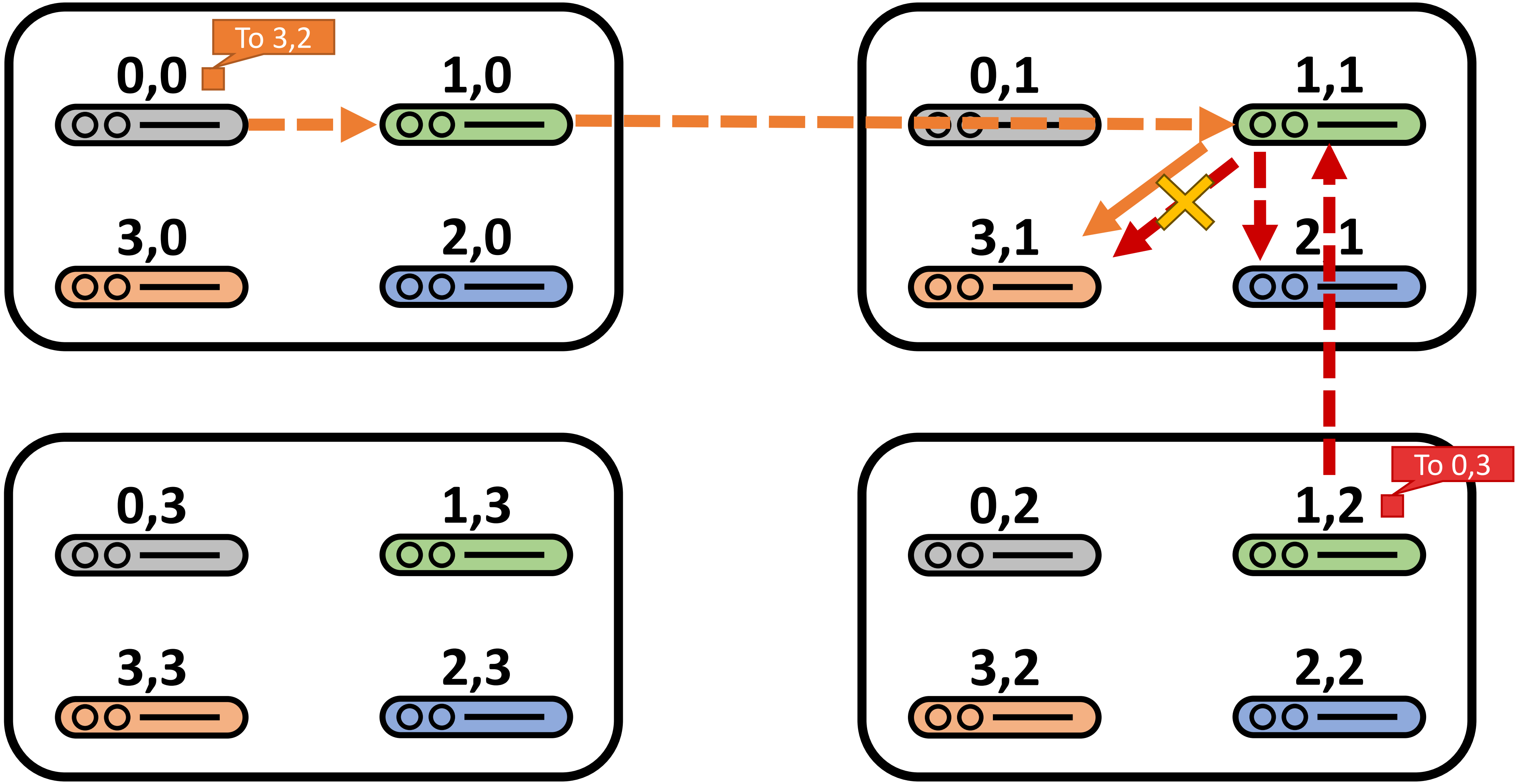
Queuing in Shale



Queuing in Shale

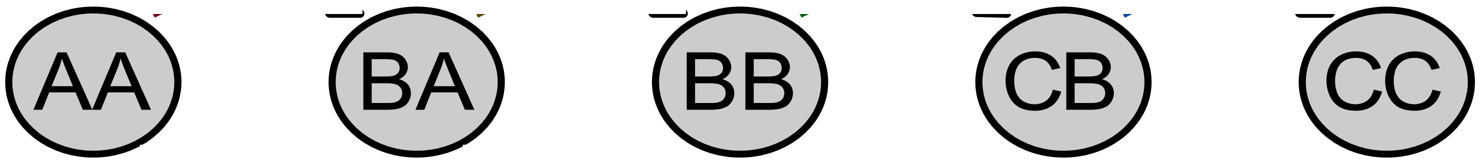
- ORNs pose unique challenges
  - Queuing has a large impact – queues empty slower than line rate
  - Each flow uses a huge number of paths (O(N) due to VLB)
- Existing ORN designs use an elegant hop-by-hop approach
- More difficult for scalable ORNs with path lengths >2
  - Due to multi-hop paths, congestion can occur far from both source and dest.
- Two types of congestion:
  - Path collision congestion
  - Egress congestion

Addressing path collisions: spray-short



Addressing egress congestion: hop-by-hop

- When a node sends a cell to an intermediate node, it stops sending future cells along that path until it receives a corresponding token
- Once it forwards the cell, the intermediate node also sends back a token

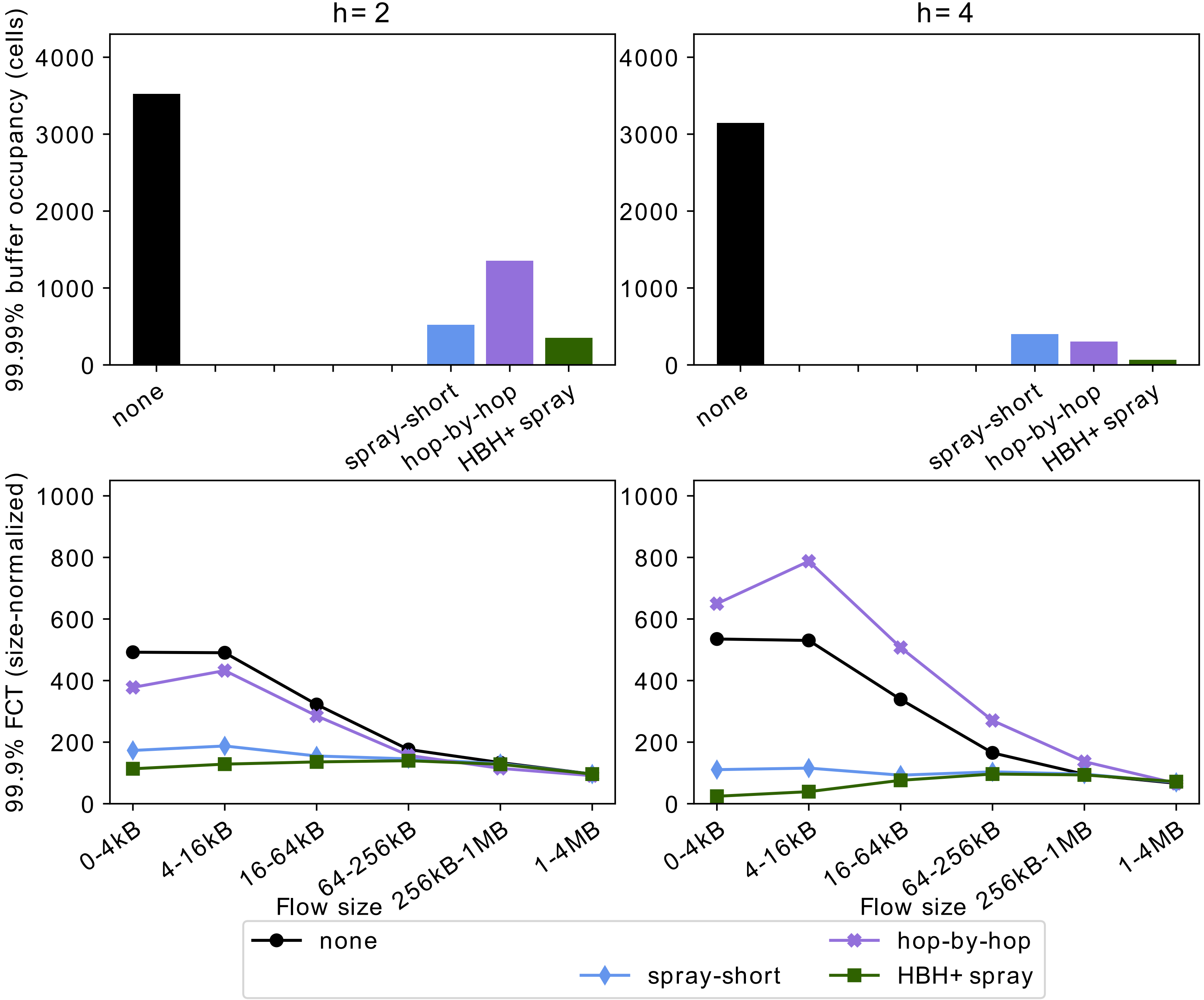


- Limits the number of cells queuing for the same destination at any node
- Deadlock-free!

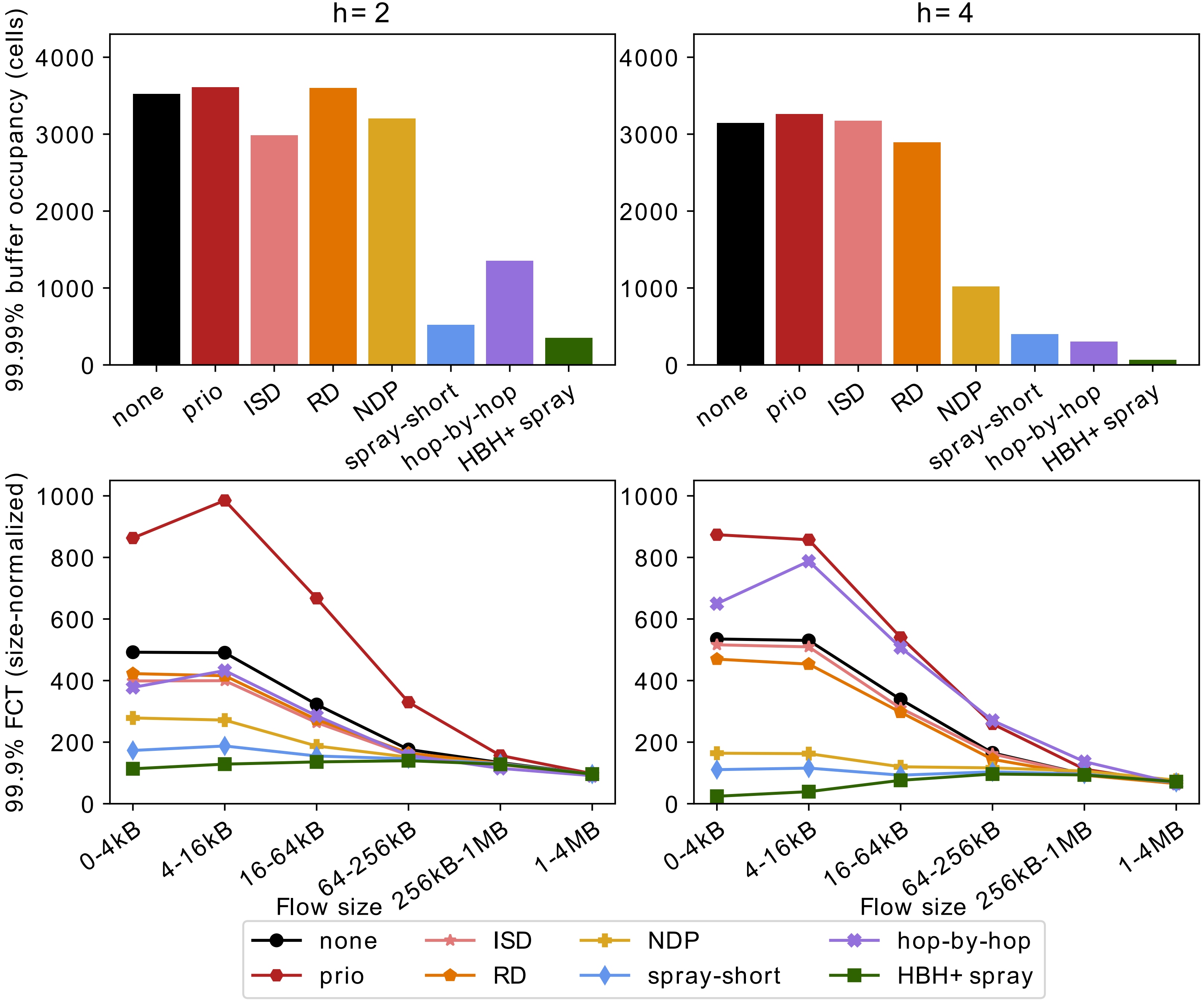
Testing our congestion control mechanisms

- Packet-level simulations of Shale with 10,000 nodes
- Simulation parameters:
  - Cell payload: 244 B
  - New timeslot every 5.632 ns
  - Propagation delay: 500 ns
- Tested various congestion control mechanisms
  - Our congestion control (spray-short, hop-by-hop, and HBH-spray)
  - In-network prioritization of short flows (prio)
  - Receiver-driven, both alone and with packet trimming (RD and NDP respectively)
  - Idealized clairvoyant sender-driven congestion control (ISD)

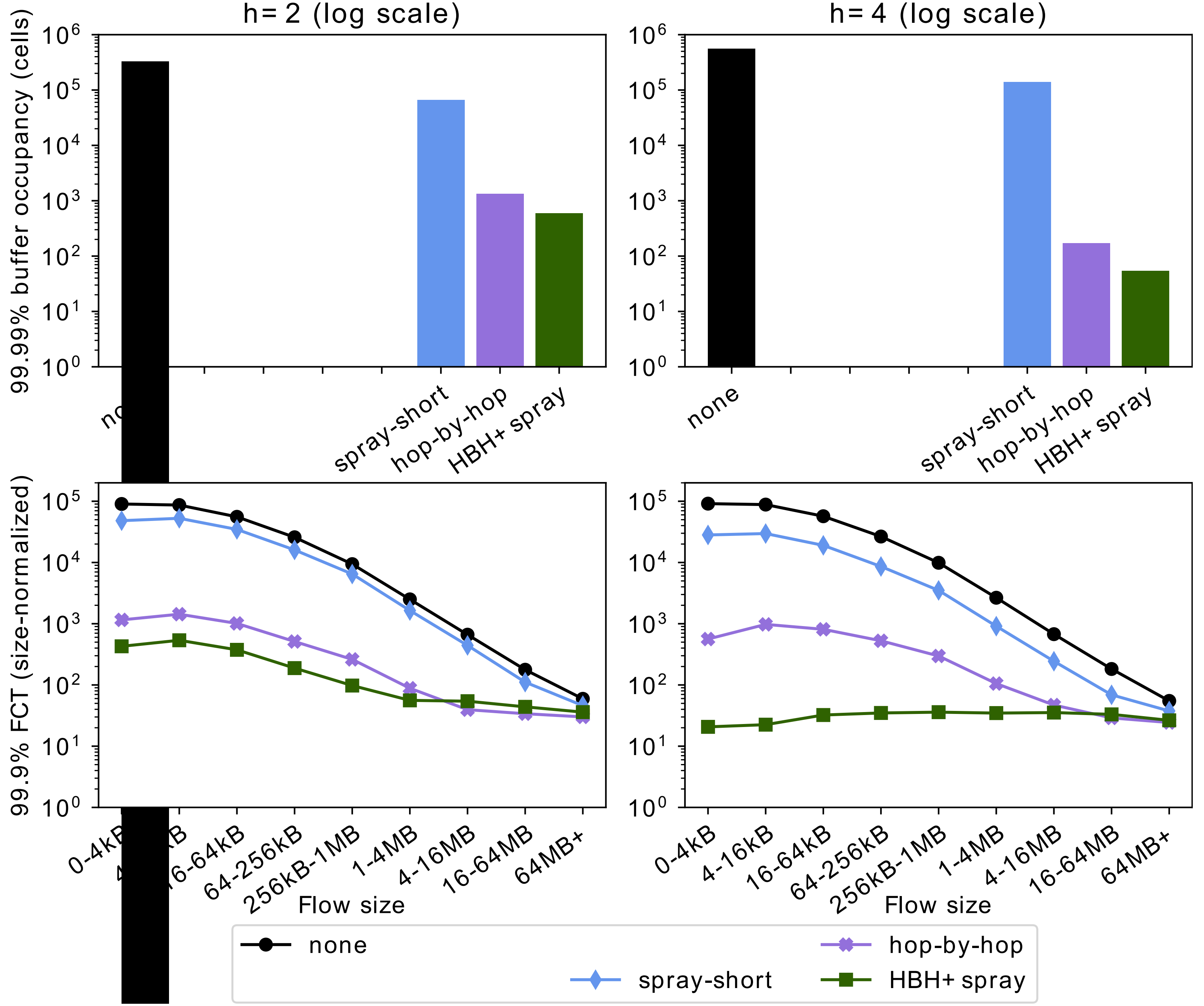
Short flow workload — N=10,000



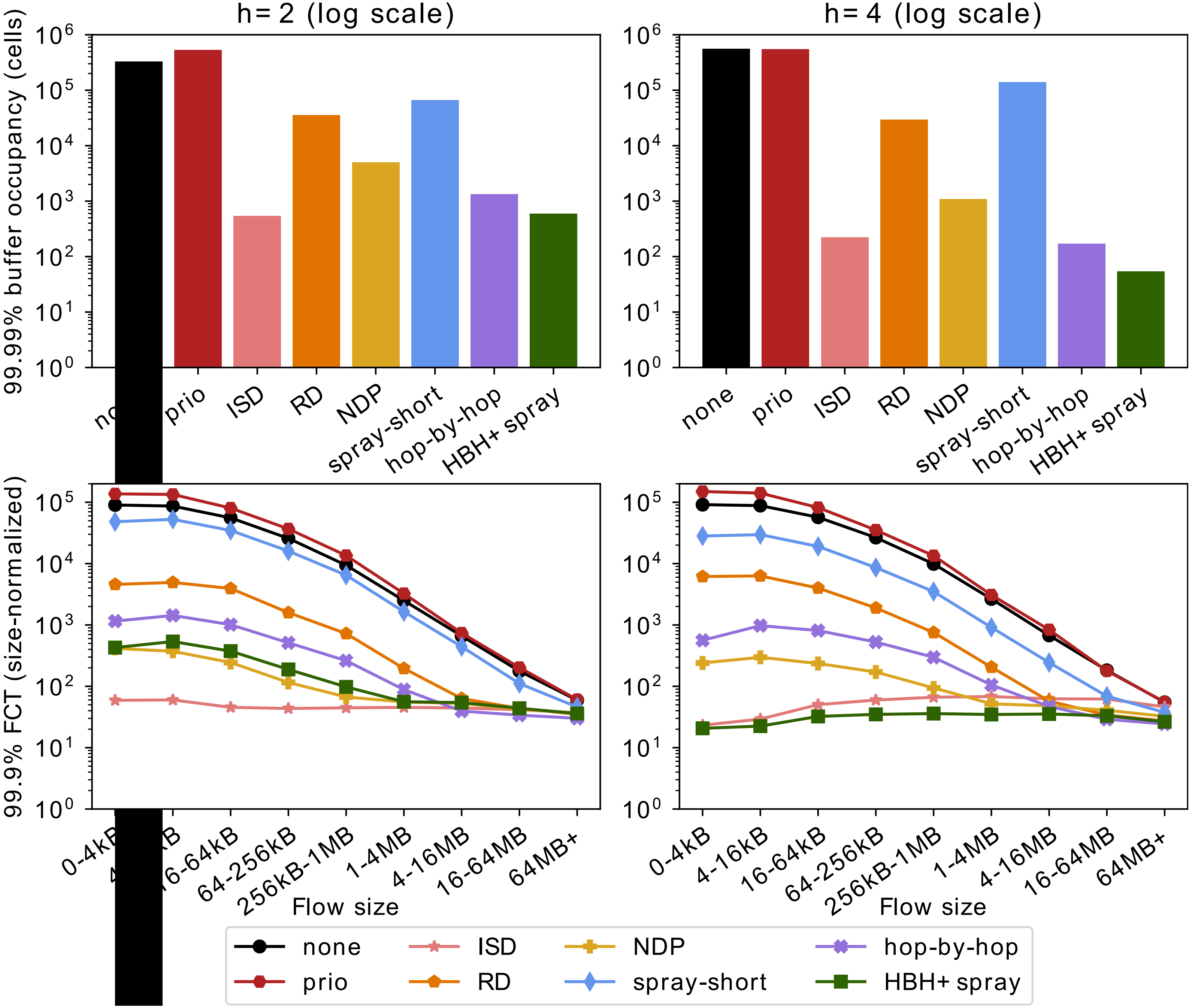
Short flow workload — N=10,000



Heavy tailed workload — N=10,000

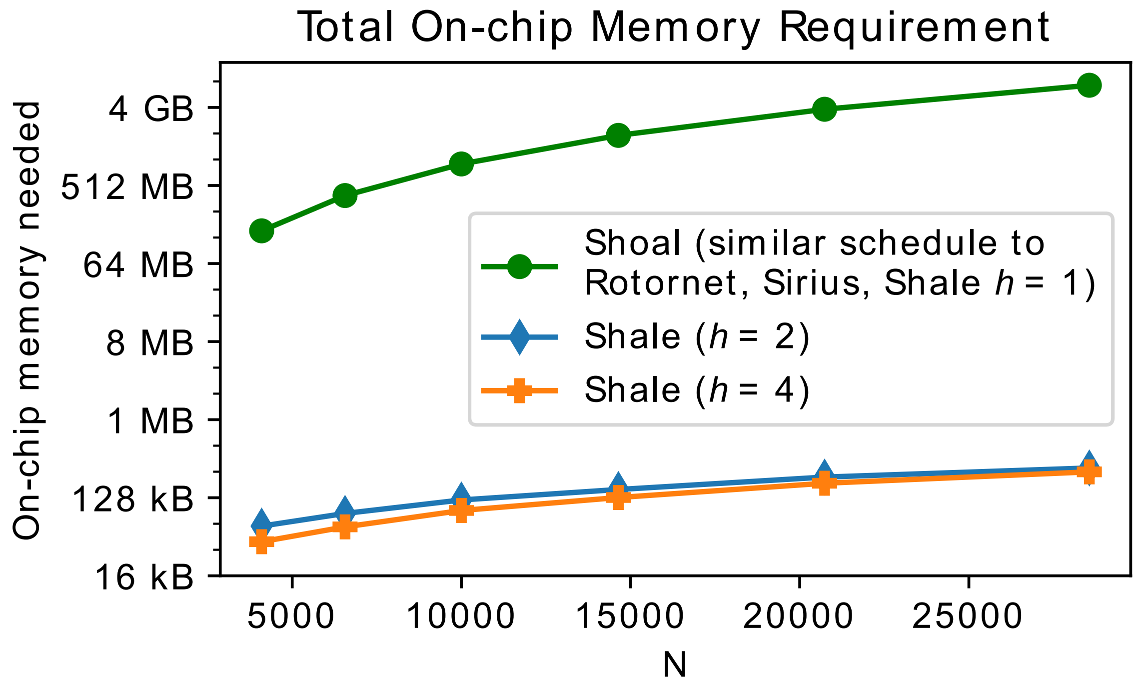


Heavy tailed workload — N=10,000



Implementing Shale

- We implemented a prototype based on an FPGA NIC
- Added several optimizations to reduce memory requirements



Summary

- Shale is a new ORN design that supports tens of thousands of nodes.
- Orders of magnitude better latency and memory requirements than existing designs at these scales
- New schedules bring tunable tradeoff between throughput and latency
  - Each tradeoff is Pareto optimal for ORNs!
- Enabled by a new congestion control
- Optimized FPGA-based hardware implementation (Session #4)
- Competitive for ML workloads (Session #7)
- Semi-oblivious (Session #1)
- Supports interleaving to combine multiple tunings

Future work

- Optimized ORNs for highly predictable workloads (e.g. machine learning model training)
- Semi-oblivious ORNs which adjust their schedule periodically to optimize for time-stable patterns in datacenters
- Finding a solution for in-network computing in ORNs

Summary of Publications

- Optimal Oblivious Reconfigurable Networks
  Daniel Amir, Tegan Wilson, Vishal Shrivastav, Hakim Weatherspoon, Robert Kleinberg, and Rachit Agarwal. **STOC 2022**.
- Extending Optimal Oblivious Reconfigurable Networks to all N
  Tegan Wilson, Daniel Amir, Vishal Shrivastav, Hakim Weatherspoon, and Robert Kleinberg. **APOCS 2023**.
- Breaking the VLB Barrier: Improving Oblivious Reconfigurable Networks with High Probability
  Tegan Wilson, Daniel Amir, Nitika Saran, Vishal Shrivastav, Robert Kleinberg, and Hakim Weatherspoon. **STOC 2024**.
- Shale: A Practical, Scalable Oblivious Reconfigurable Network
  Daniel Amir, Tegan Wilson, Nitika Saran, Robert Kleinberg, Vishal Shrivastav, and Hakim Weatherspoon. **SIGCOMM 2024**.
- Semi-Oblivious Reconfigurable Datacenter Networks
  Nitika Saran, Daniel Amir, Tegan Wilson, Robert Kleinberg, Vishal Shrivastav, and Hakim Weatherspoon. **HotNets 2024**.

Thank you!

Nitika Saran

Tegan Wilson

Robert Kleinberg

Vishal Shrivastav

Hakim Weatherspoon