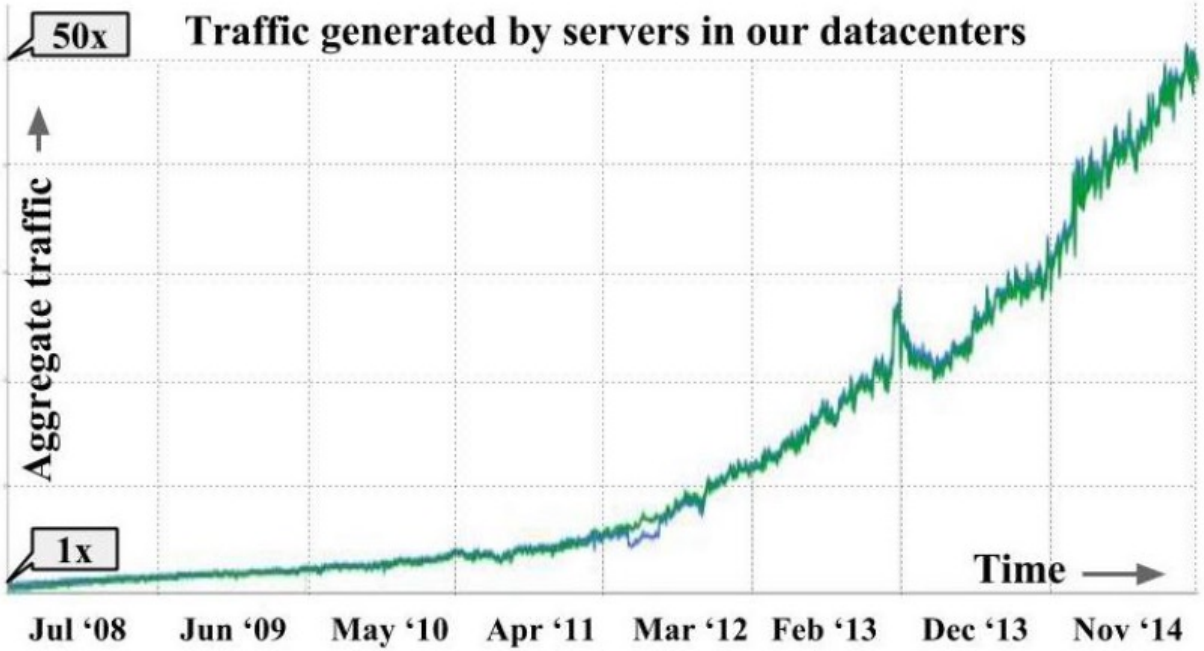

# Shale: A Practical, Scalable Oblivious Reconfigurable Network

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

* Cornell University

† PURDUE UNIVERSITY

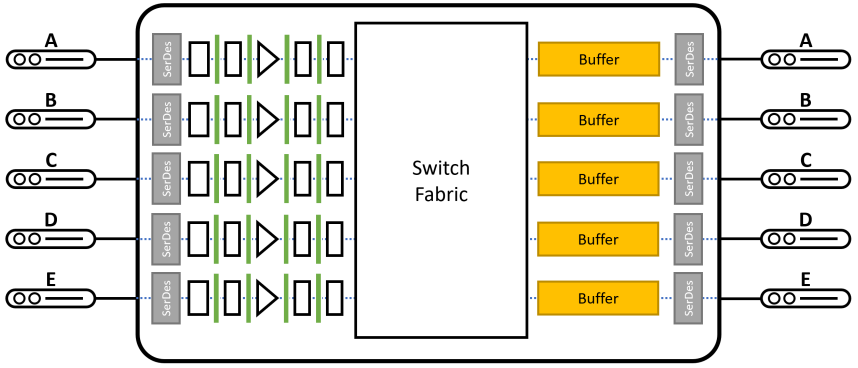
# Growing datacenter network demand




Source: Google

- Google: “Bandwidth demand increasing 2x every 12-15 months”
- Microsoft: “Link rate increasing 400x in 10 years”
- Broadcom: “Switch chip speed increasing 2x / 2 years”
- Moore’s law is ending!

## Packet Switches

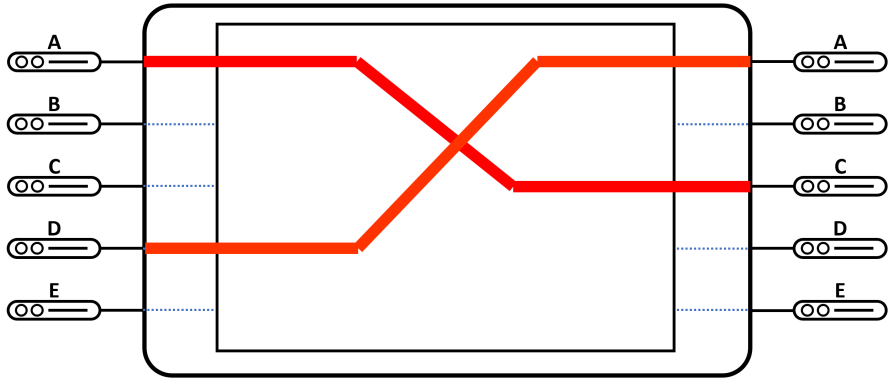



Make their own decisions

- Must convert data to electricity
- Limited by Moore's law

- Easy to use
- Low latency

## Circuit Switches



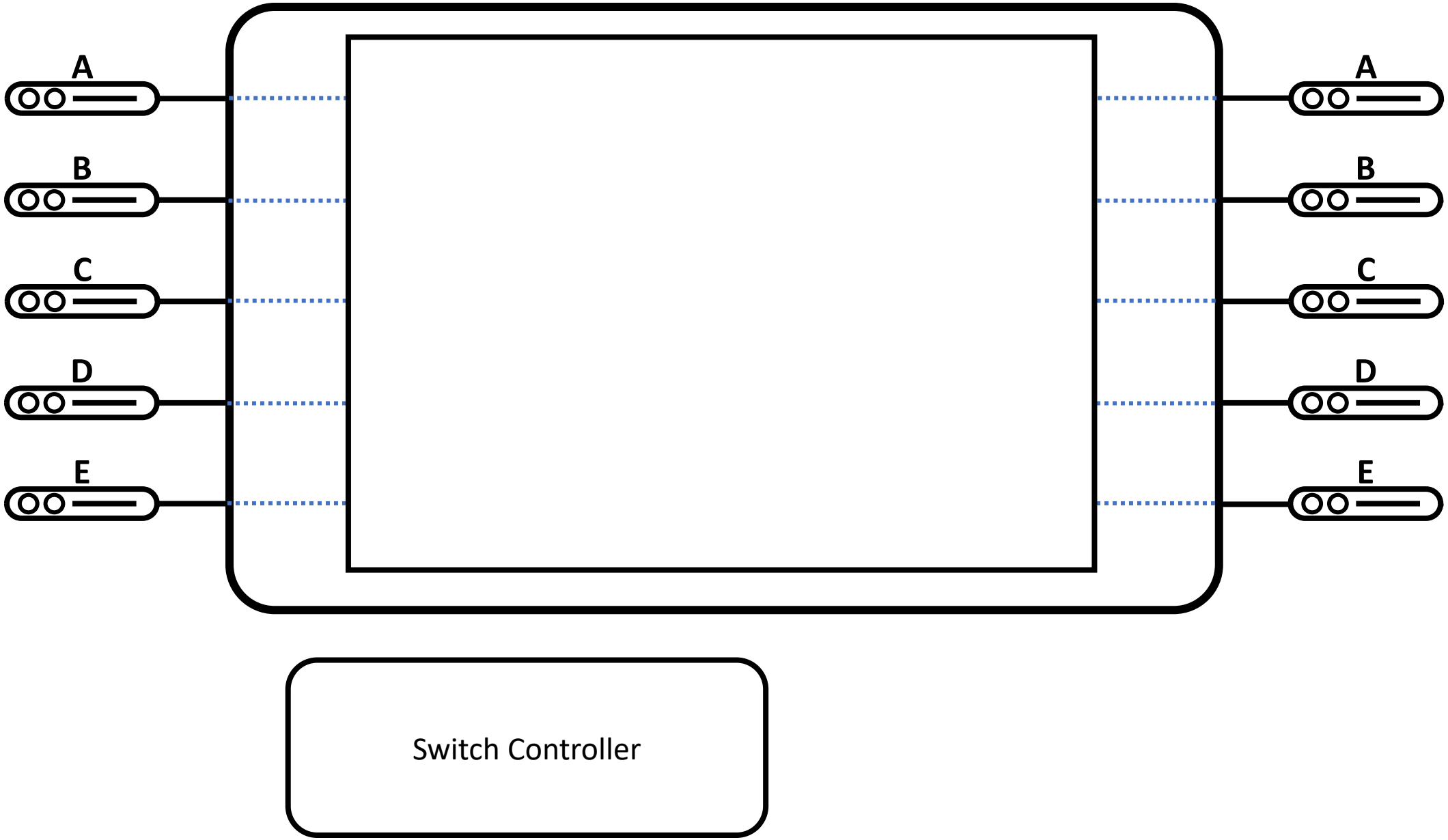

Switch needs instructions

- Data can remain in light form
- Potentially bandwidth-agnostic

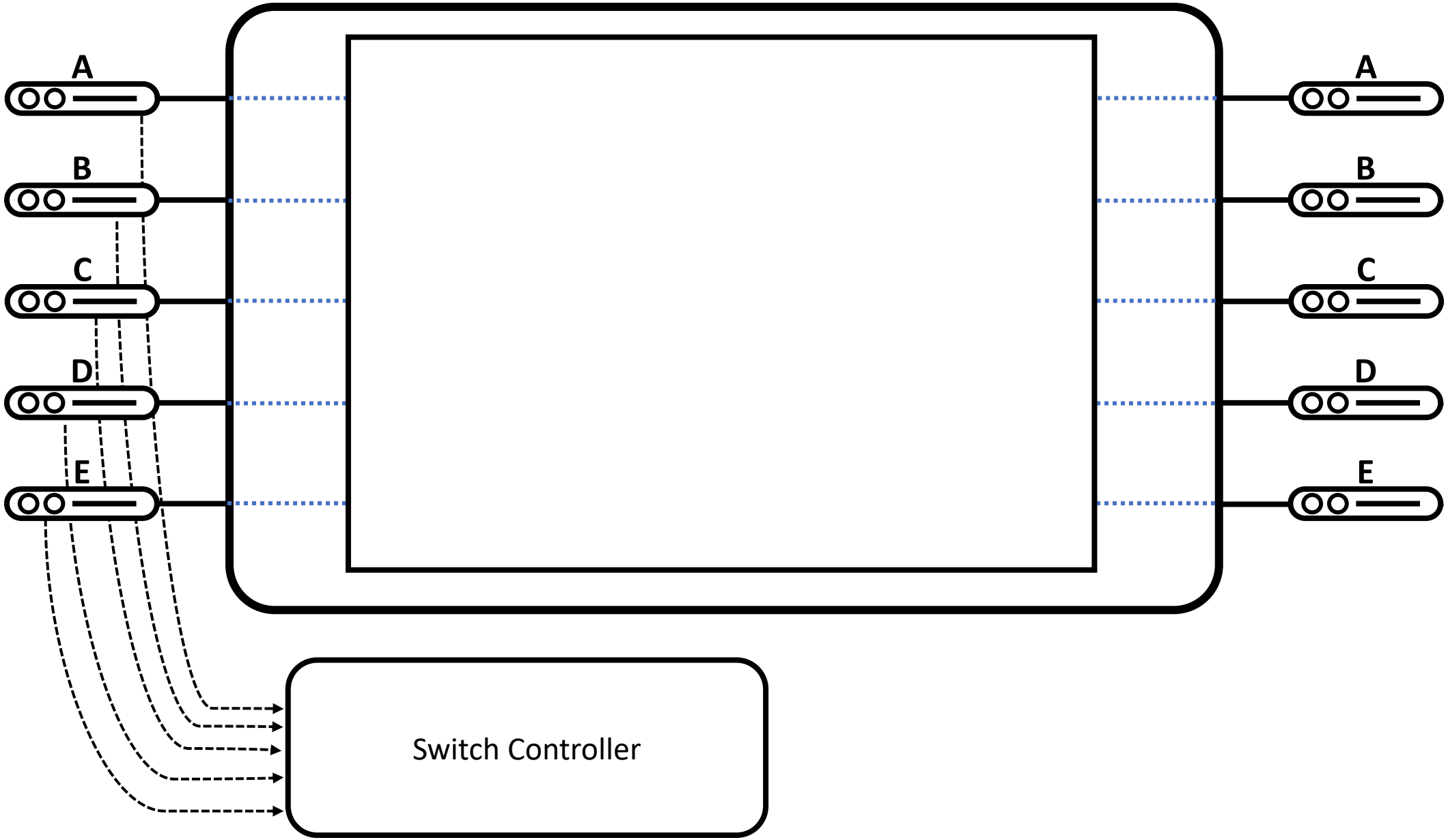
- Network design challenge
- ~~Creating a circuit used to take milliseconds~~ Now just nanoseconds

# Can we design datacenter networks that use only circuit switches?

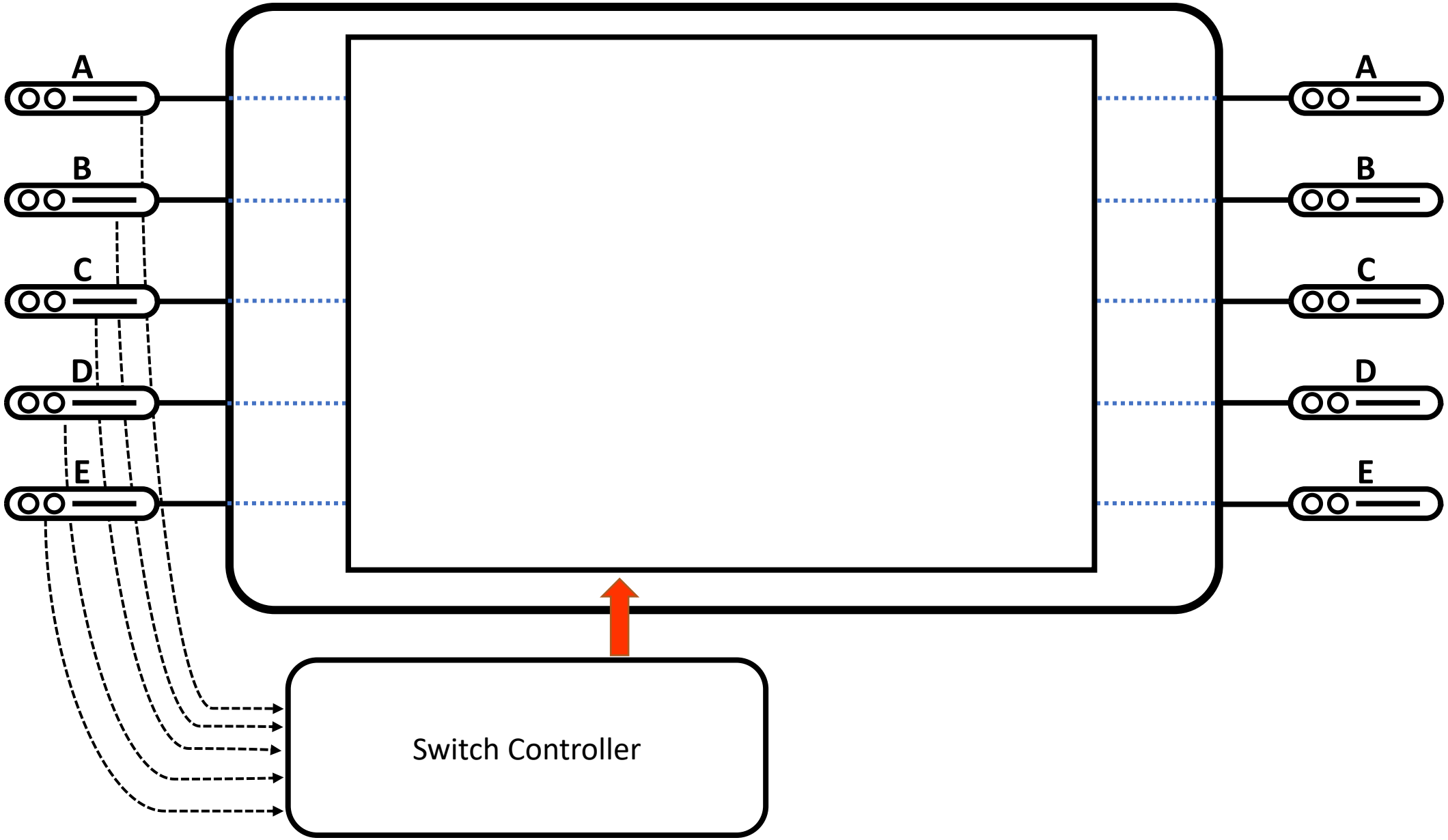
# Traditional Circuit-switched Networks



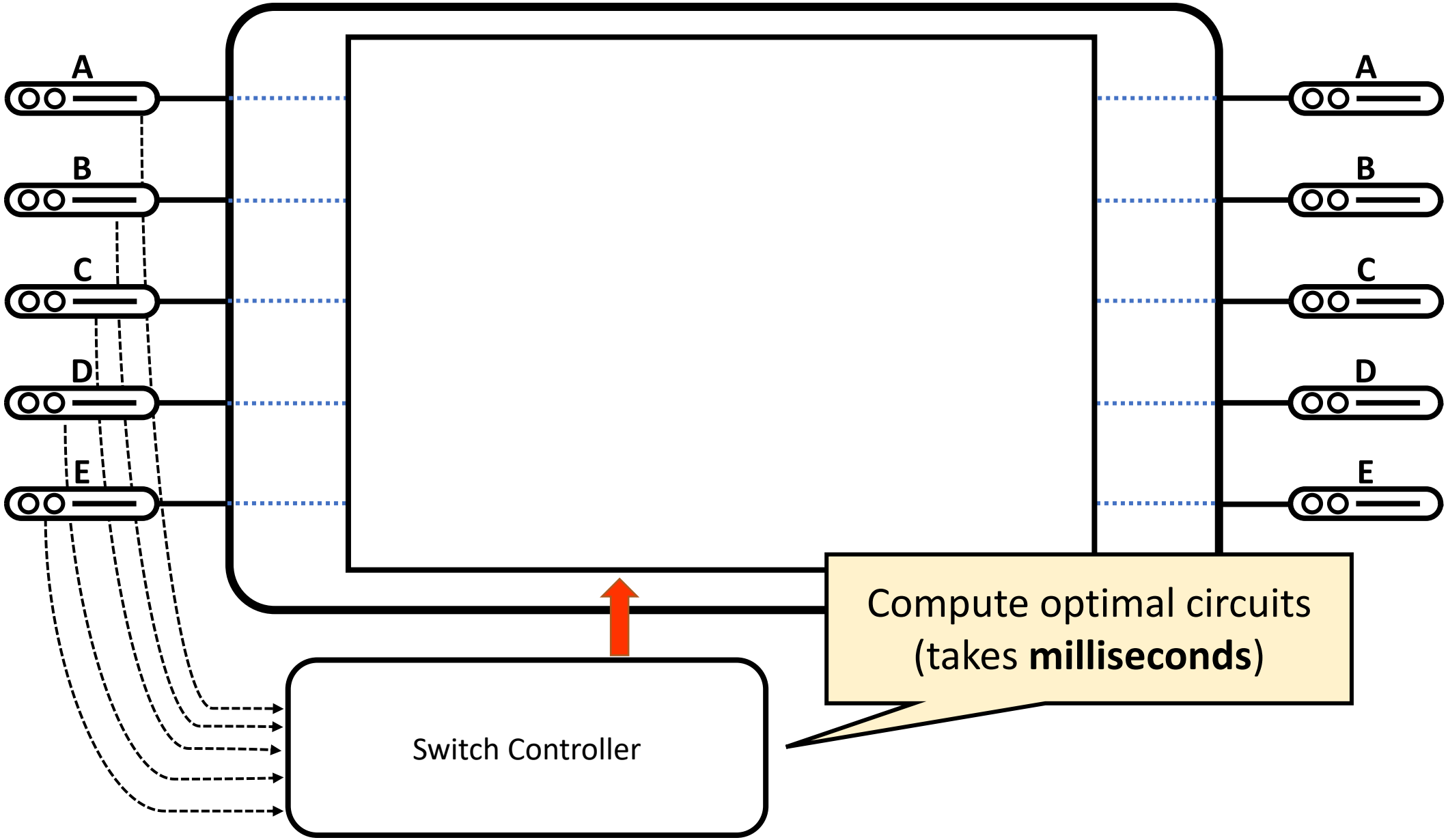
# Traditional Circuit-switched Networks



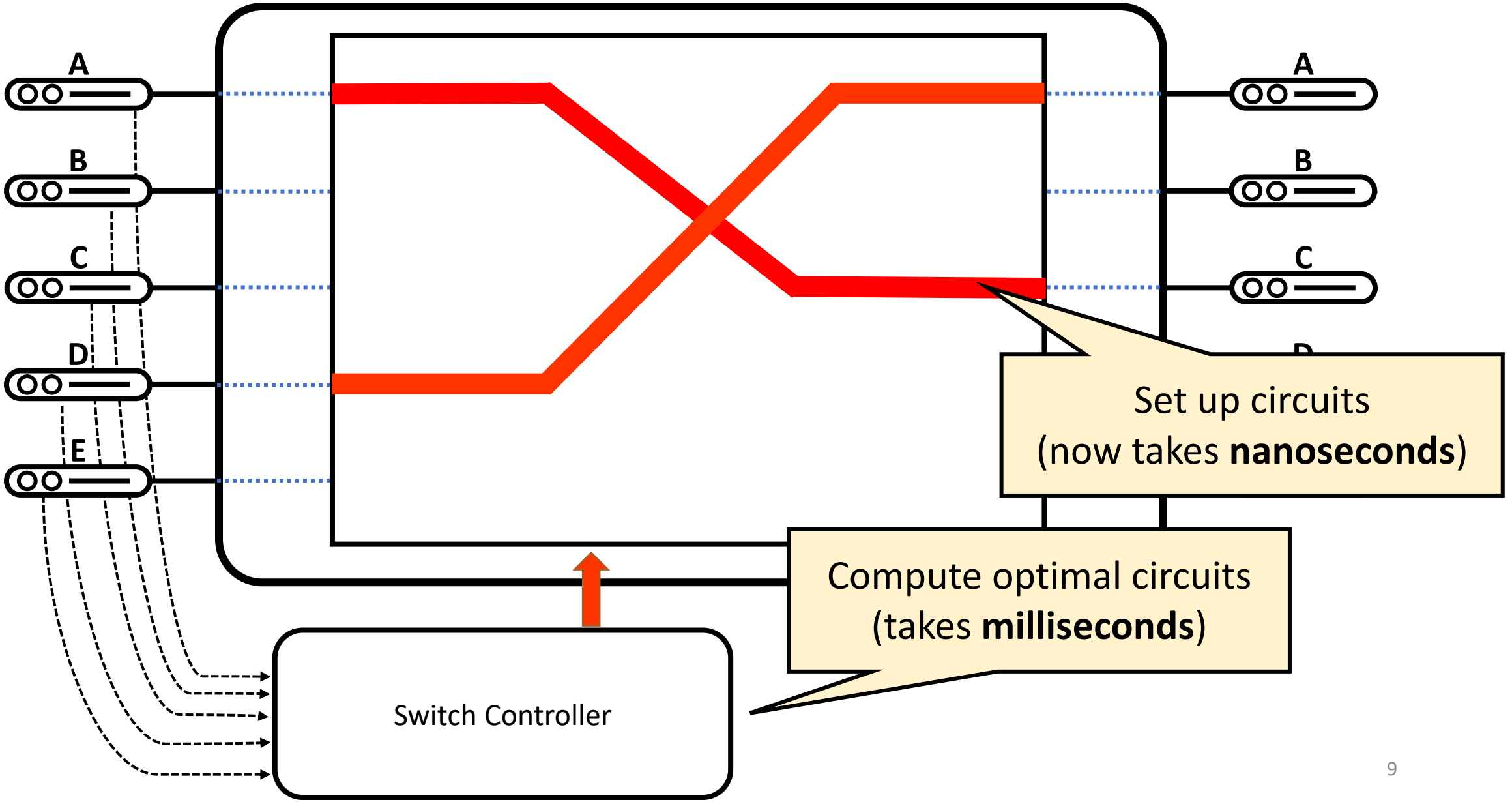
# Traditional Circuit-switched Networks



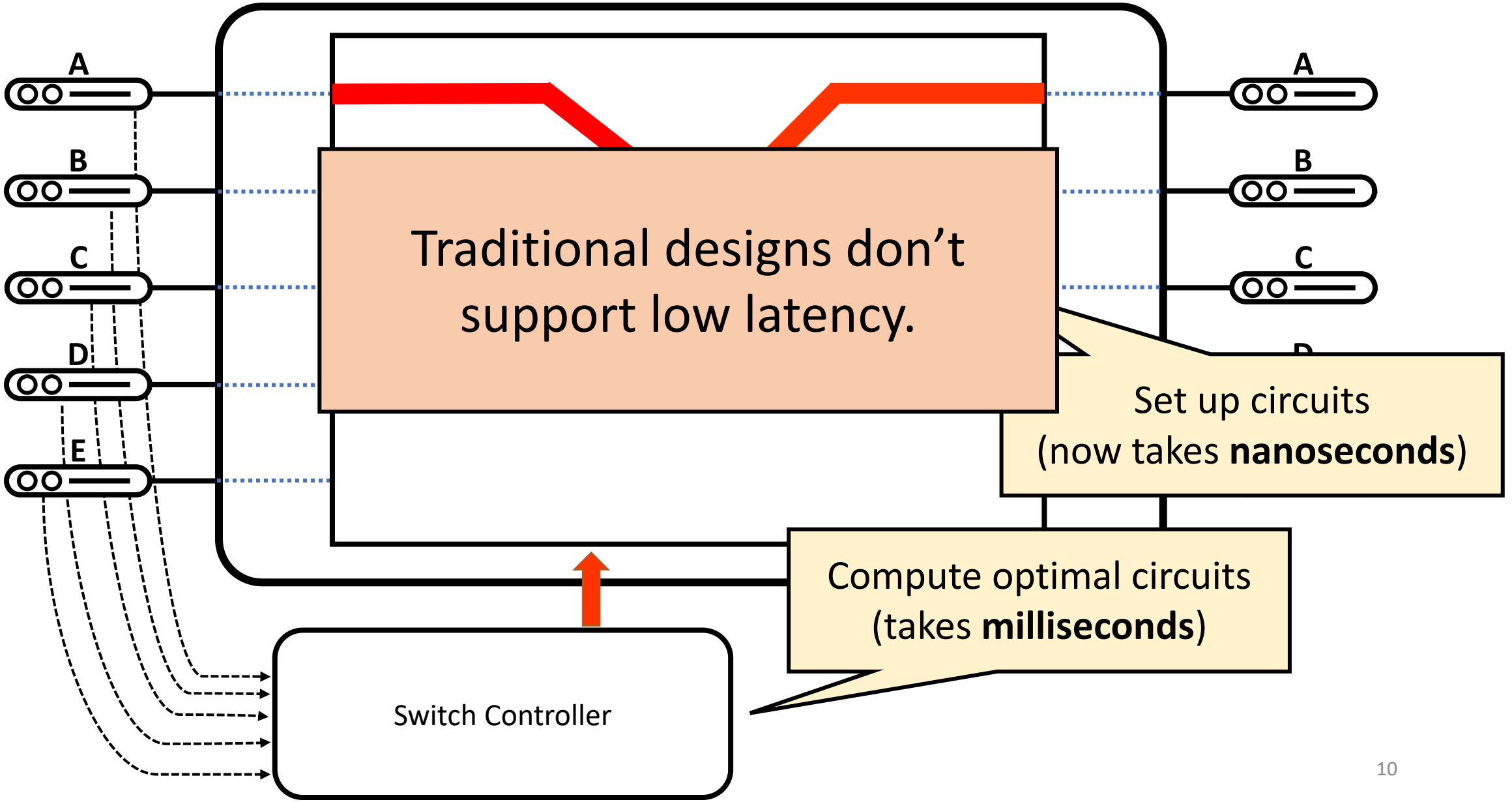
# Traditional Circuit-switched Networks



# Traditional Circuit-switched Networks



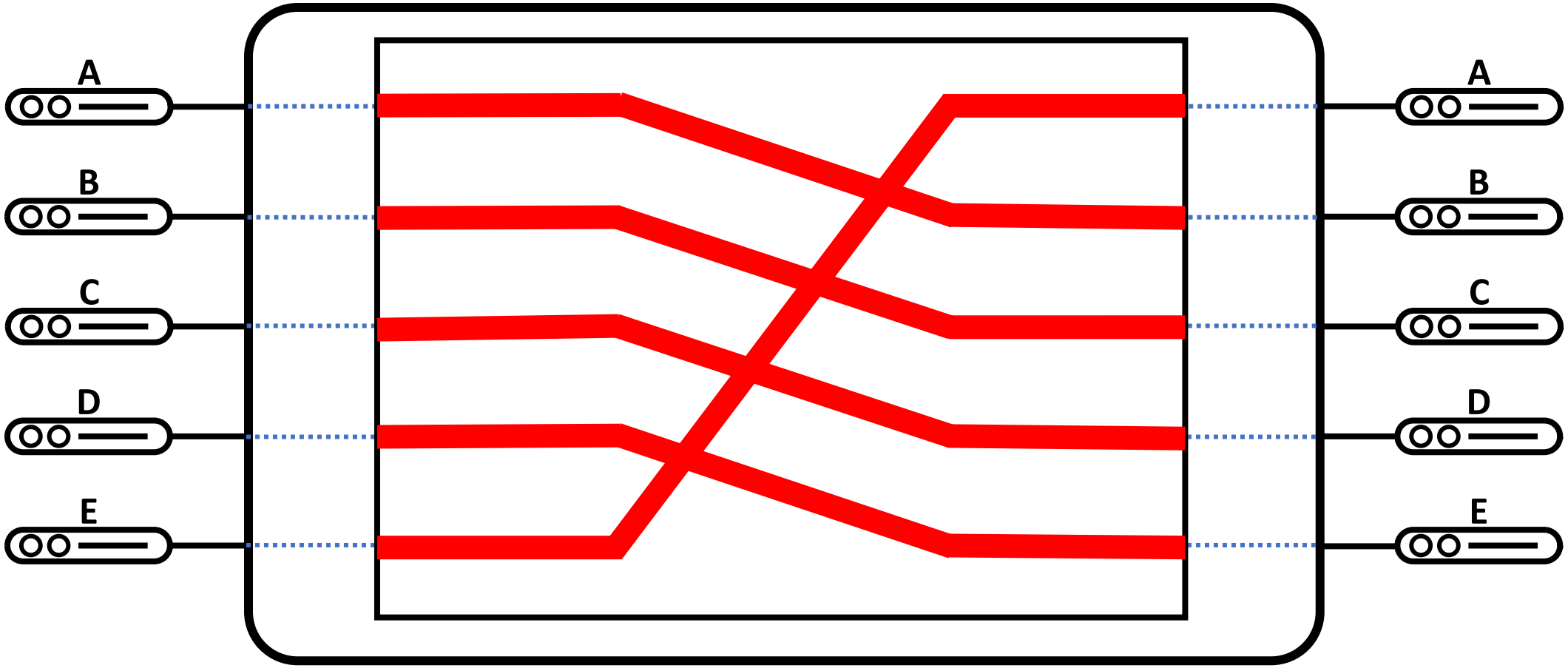
# Traditional Circuit-switched Networks



# Oblivious Reconfigurable Networks (ORNs)

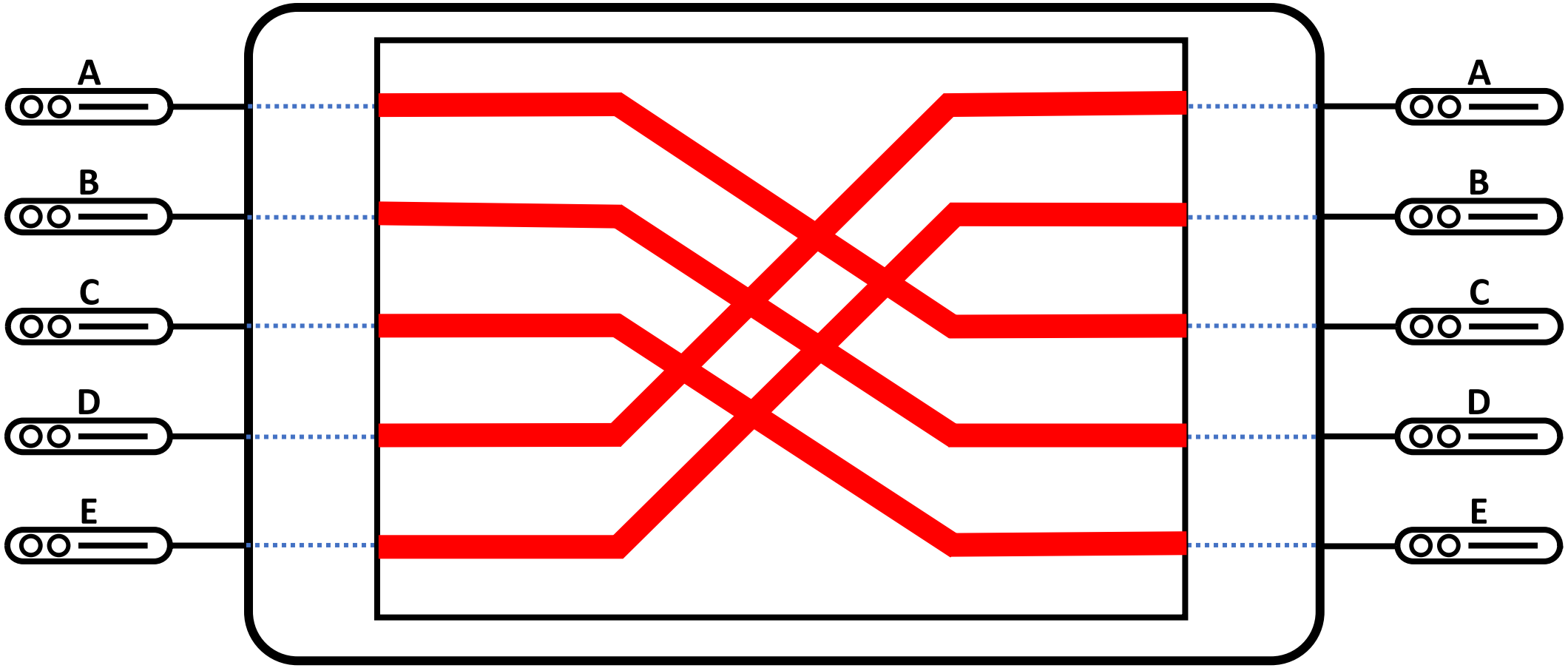

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |

# Oblivious Reconfigurable Networks (ORNs)



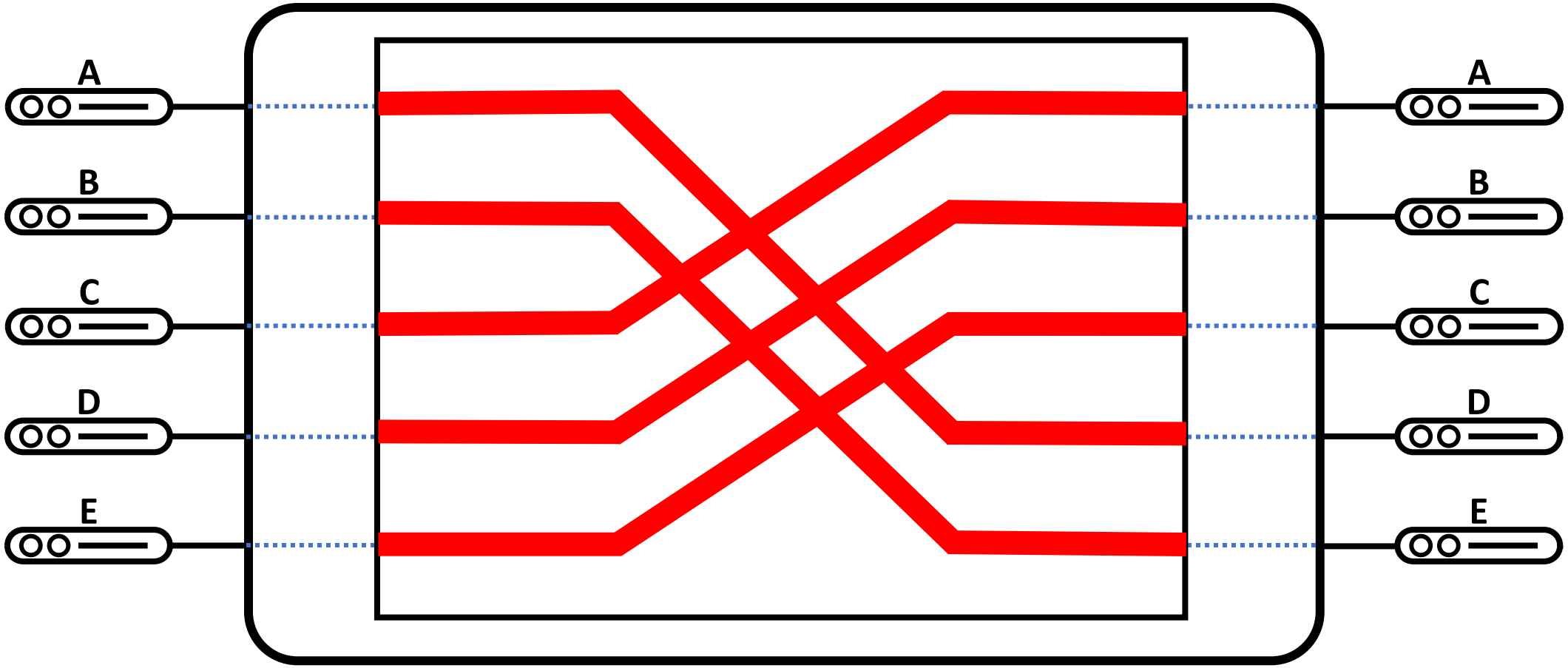

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |

# Oblivious Reconfigurable Networks (ORNs)



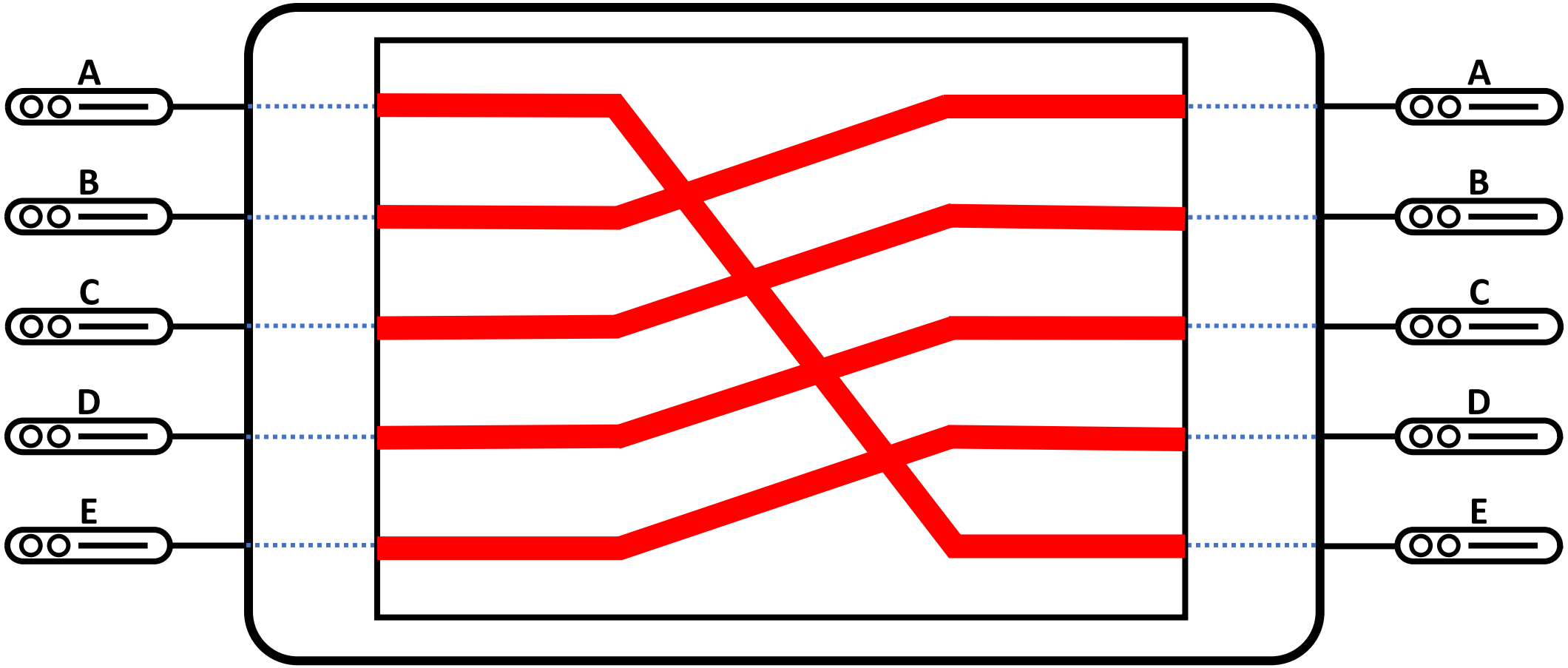

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |

# Oblivious Reconfigurable Networks (ORNs)




| | 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 |

# Oblivious Reconfigurable Networks (ORNs)

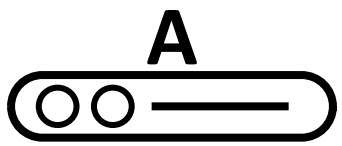



| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |

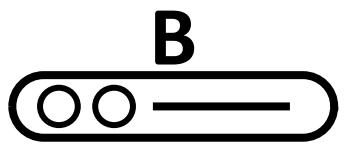
# Existing ORNs: Rotornet, Shoal, Sirius

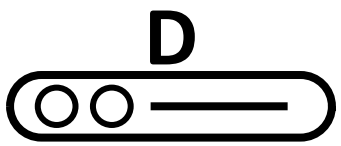
- Schedule is a round-robin

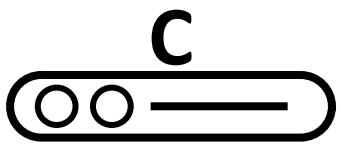
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



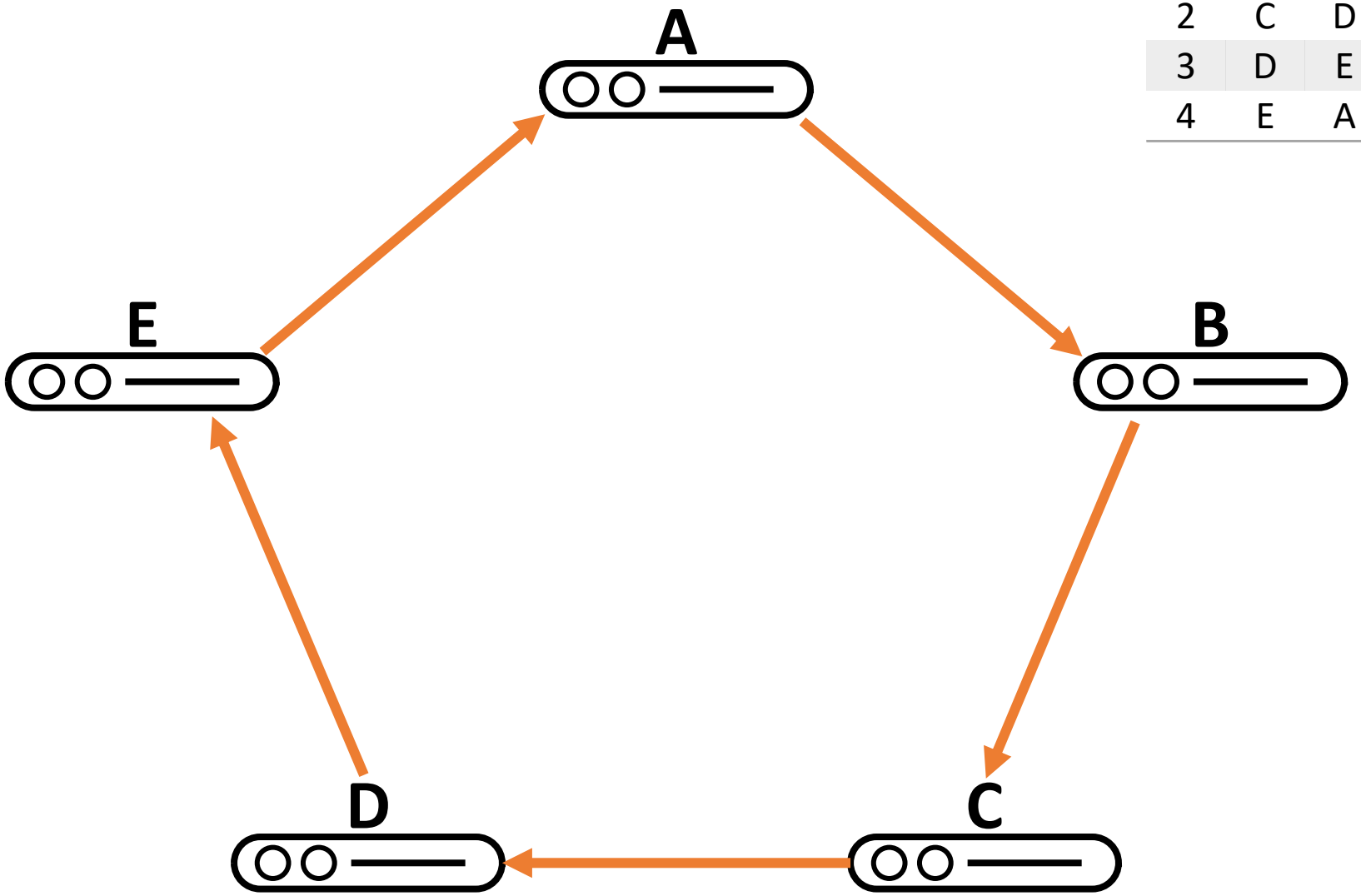




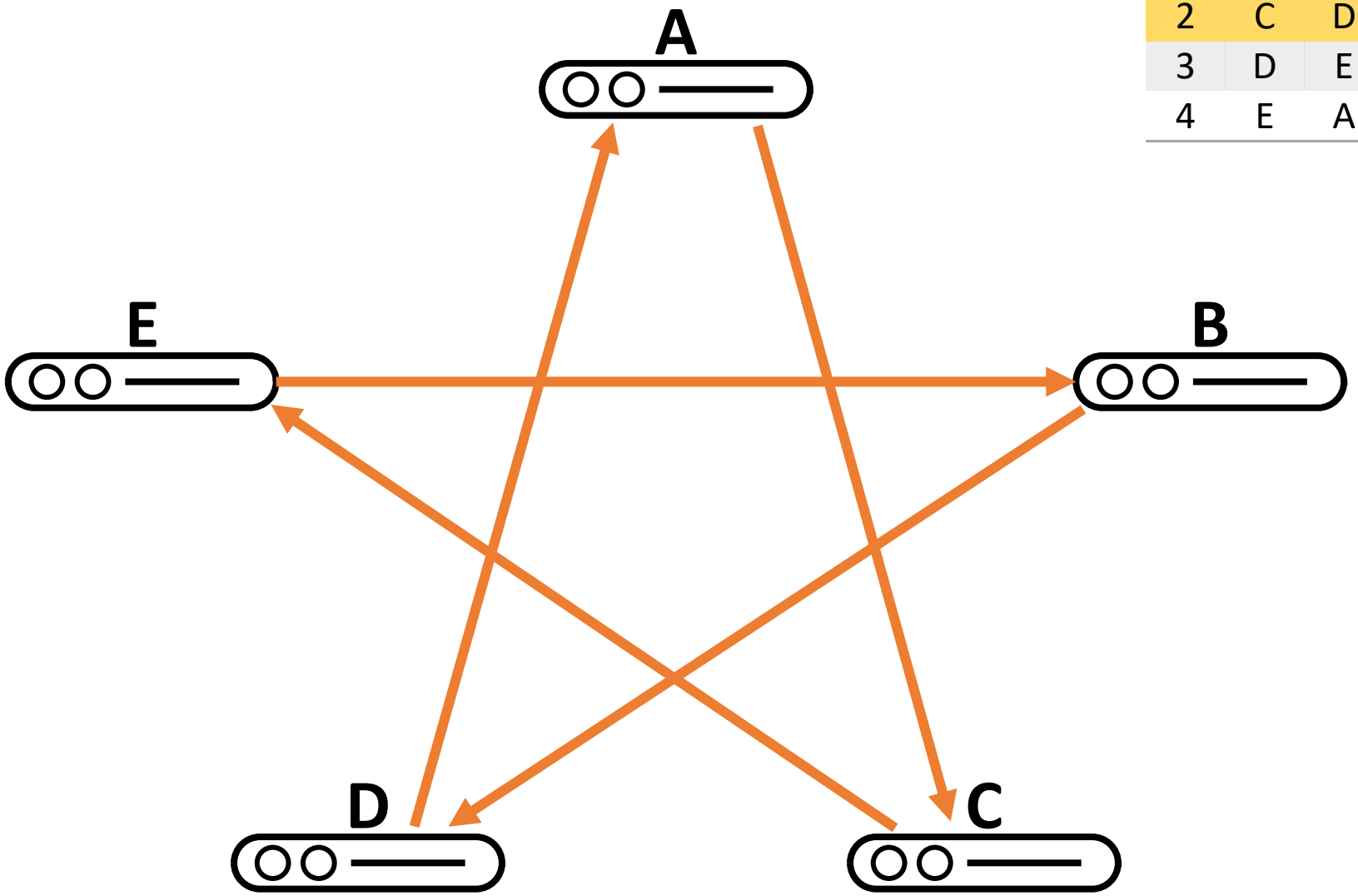



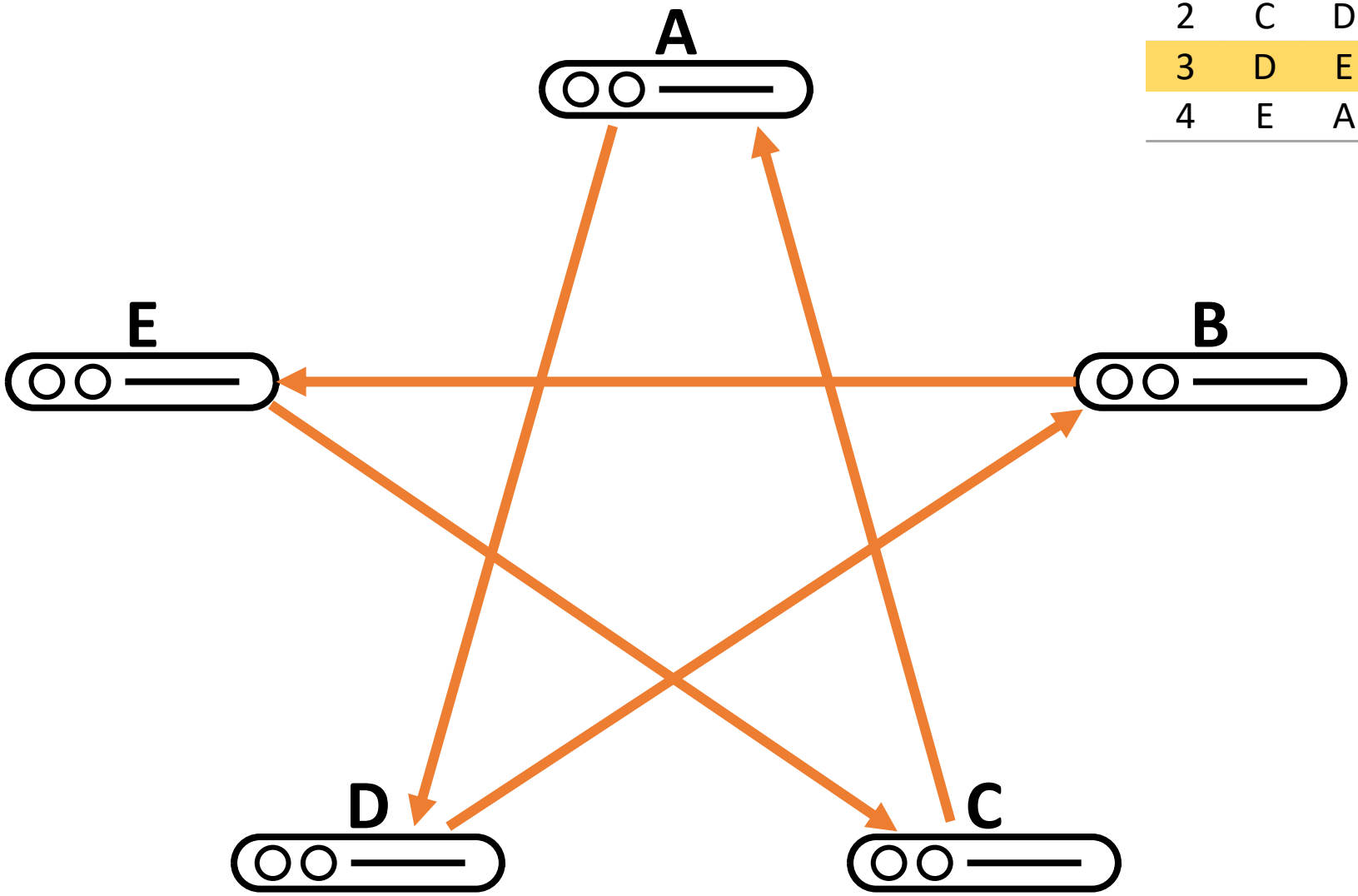
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



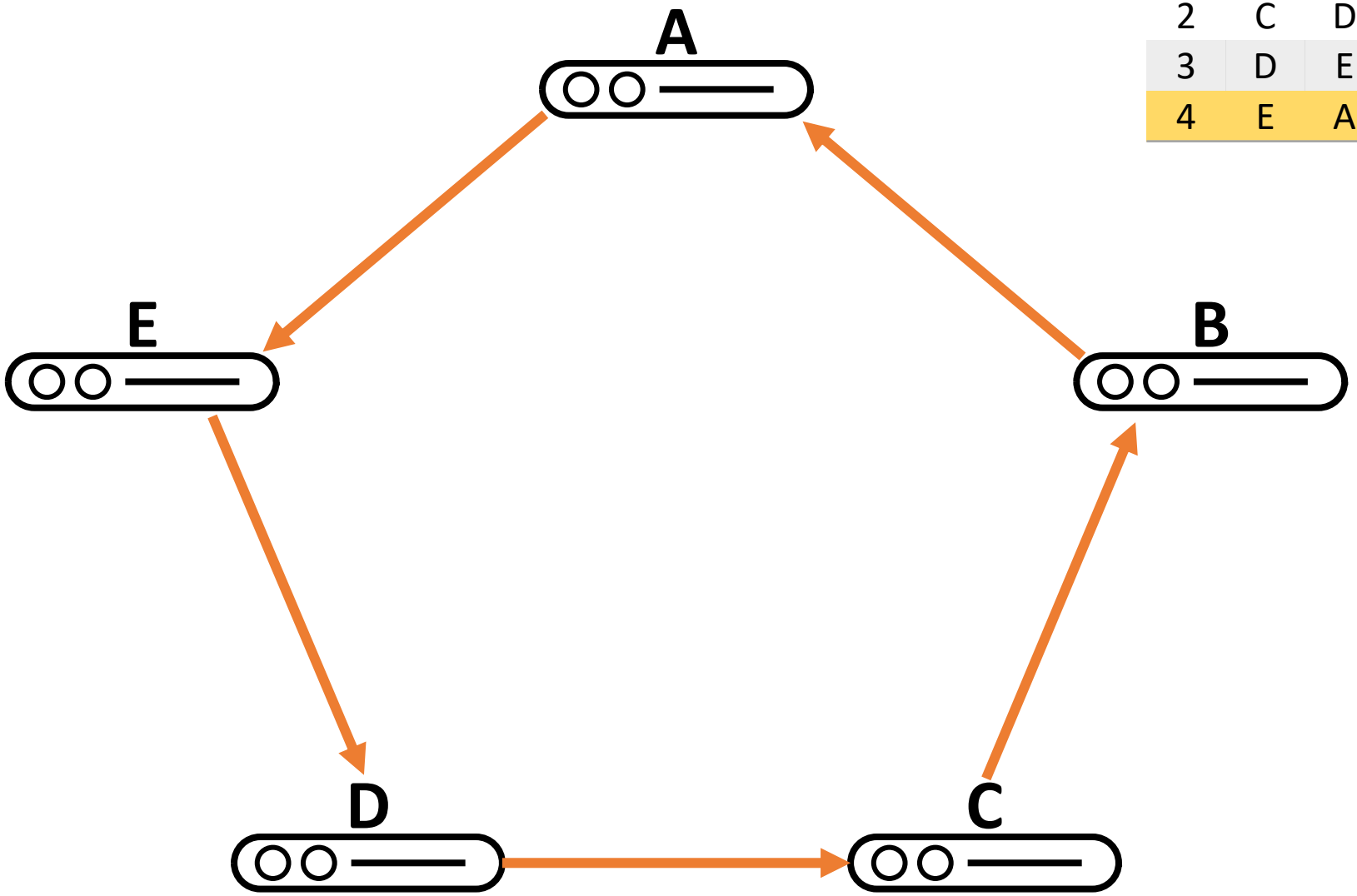
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



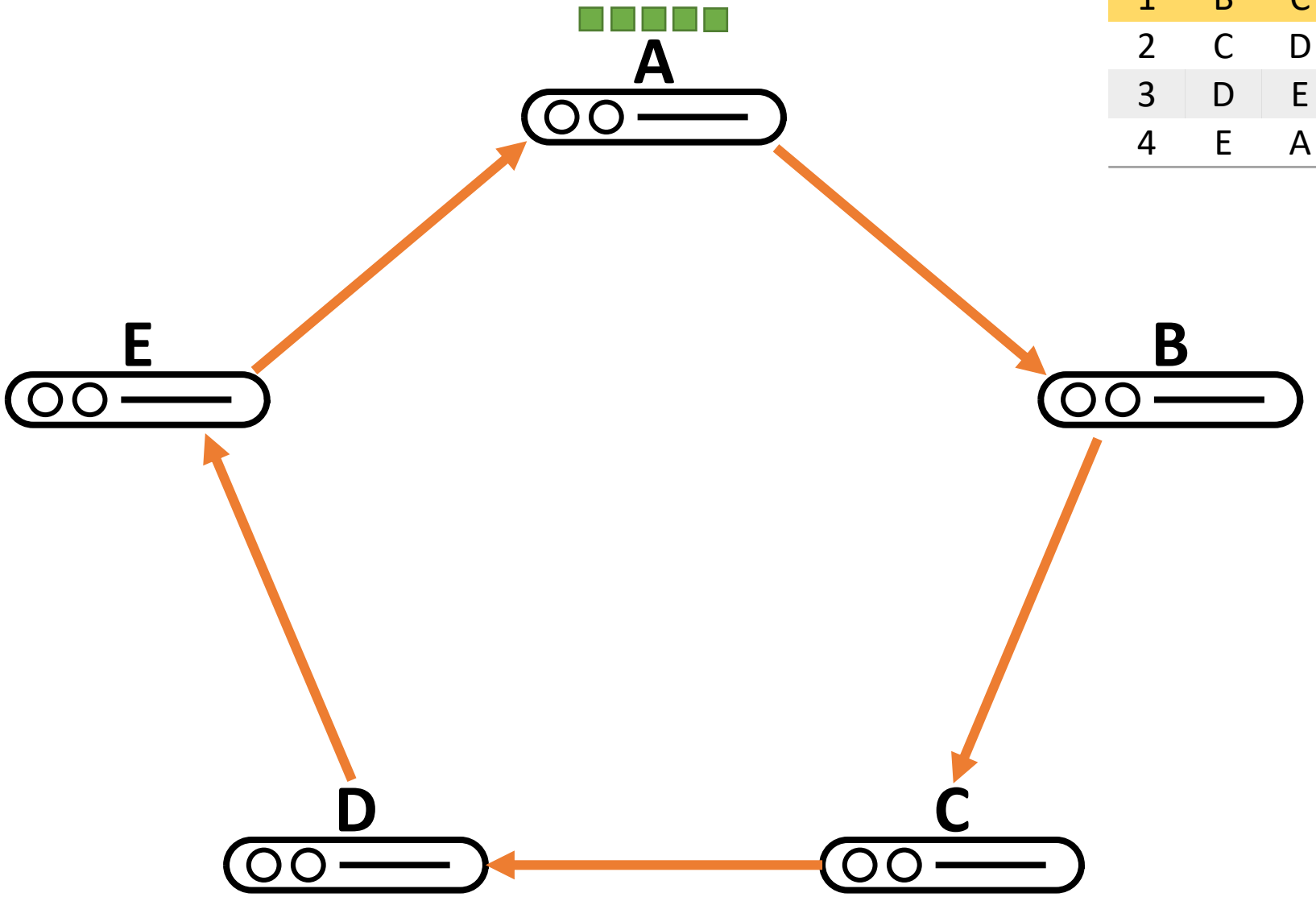
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



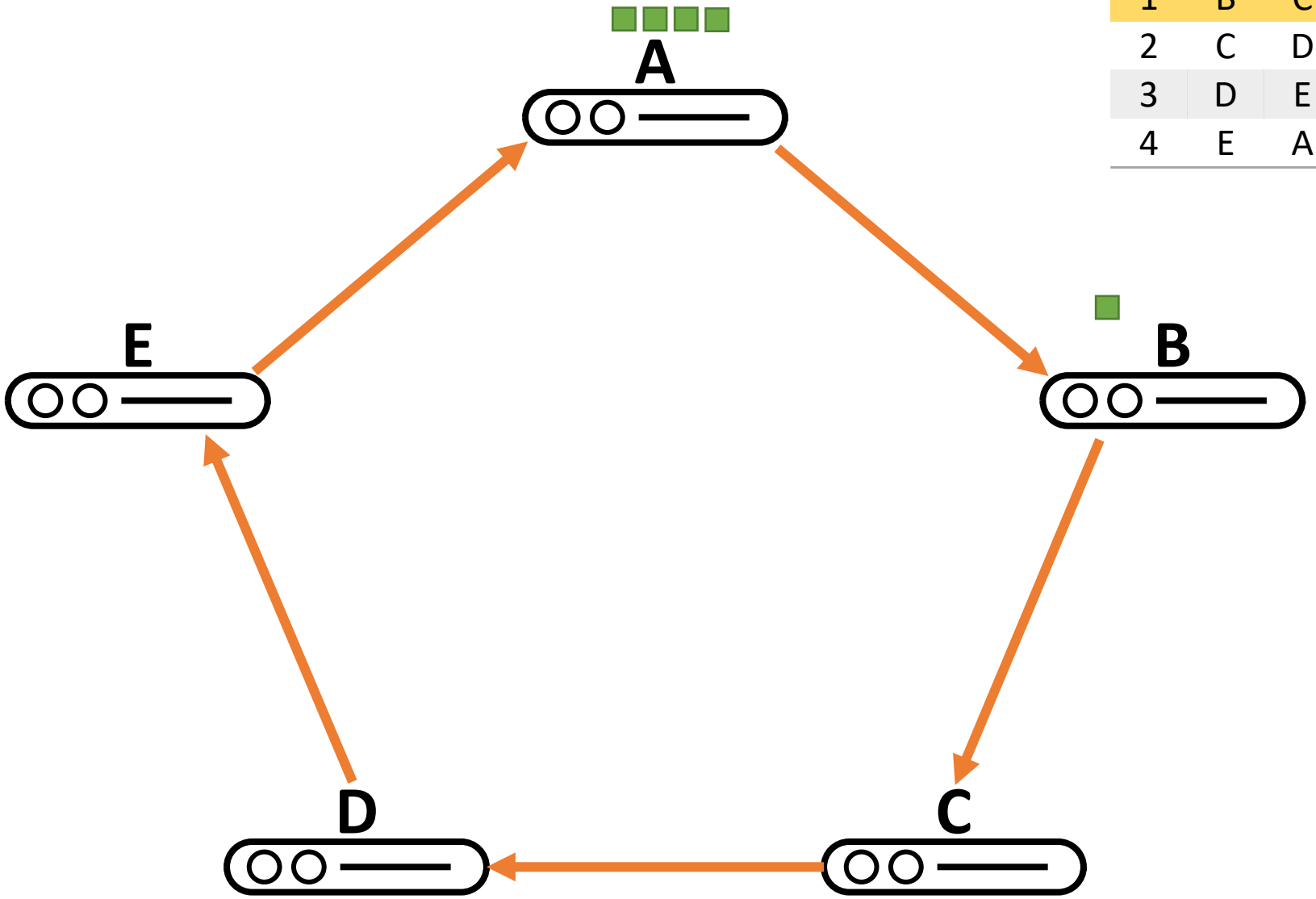
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



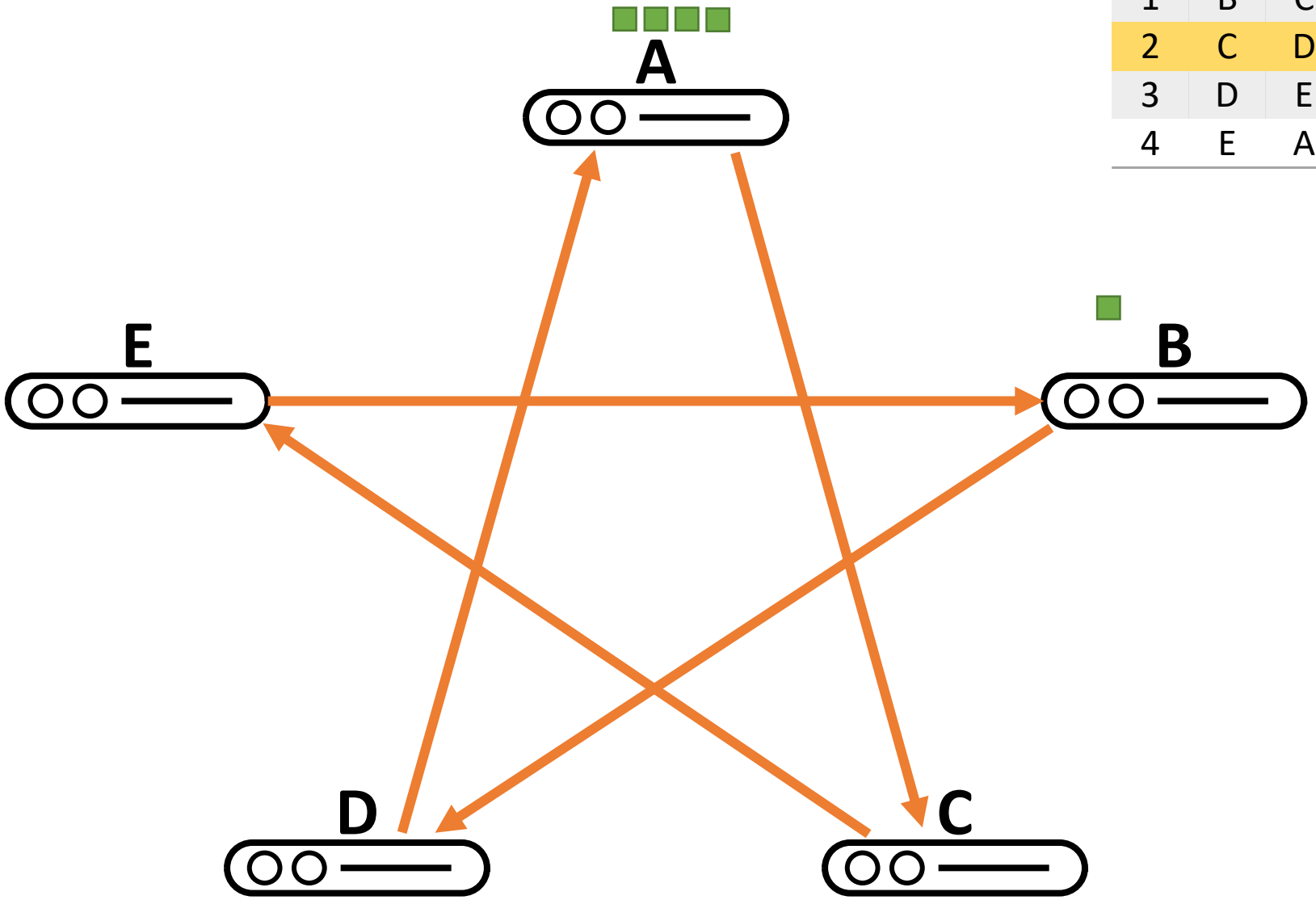
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



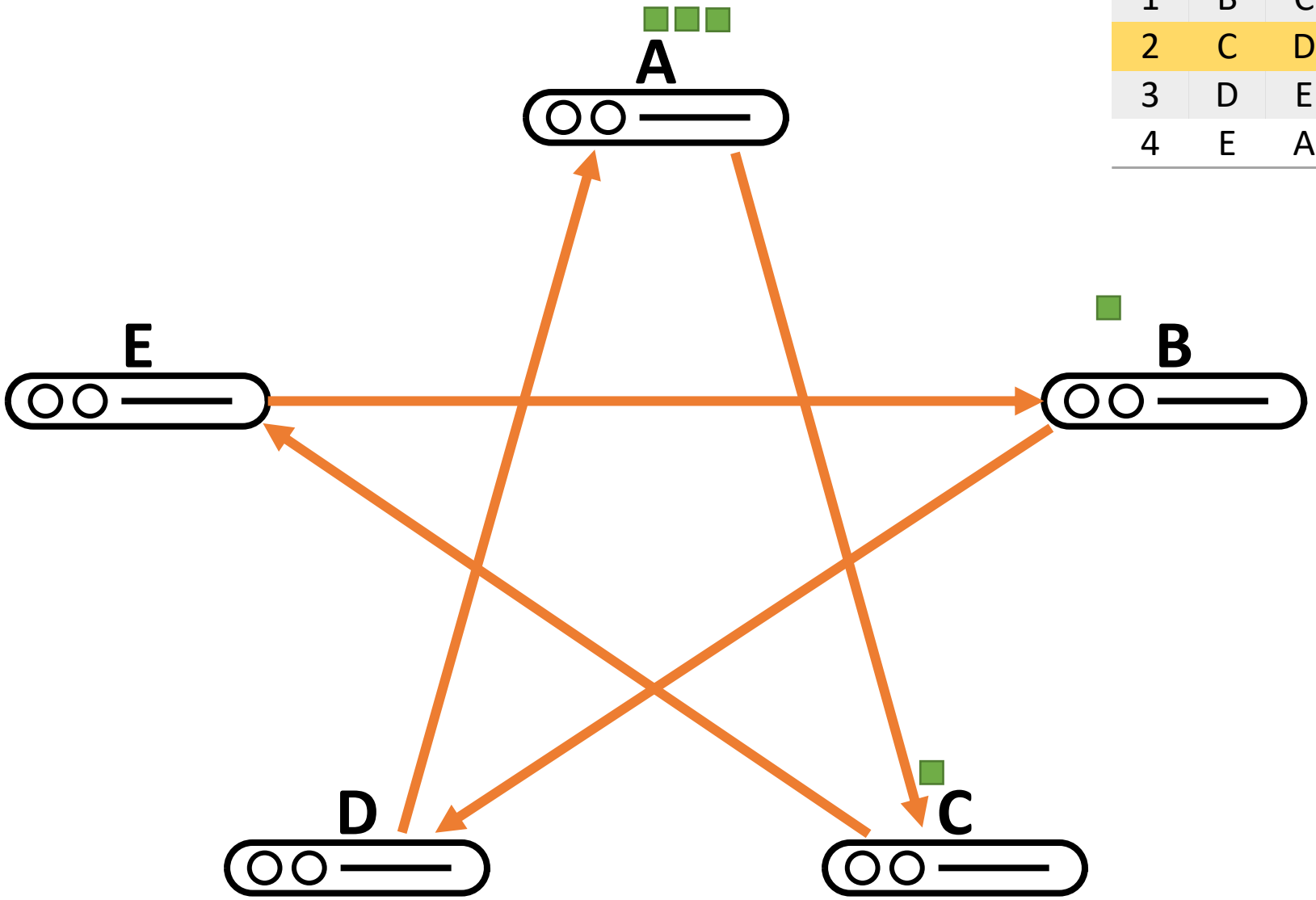
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



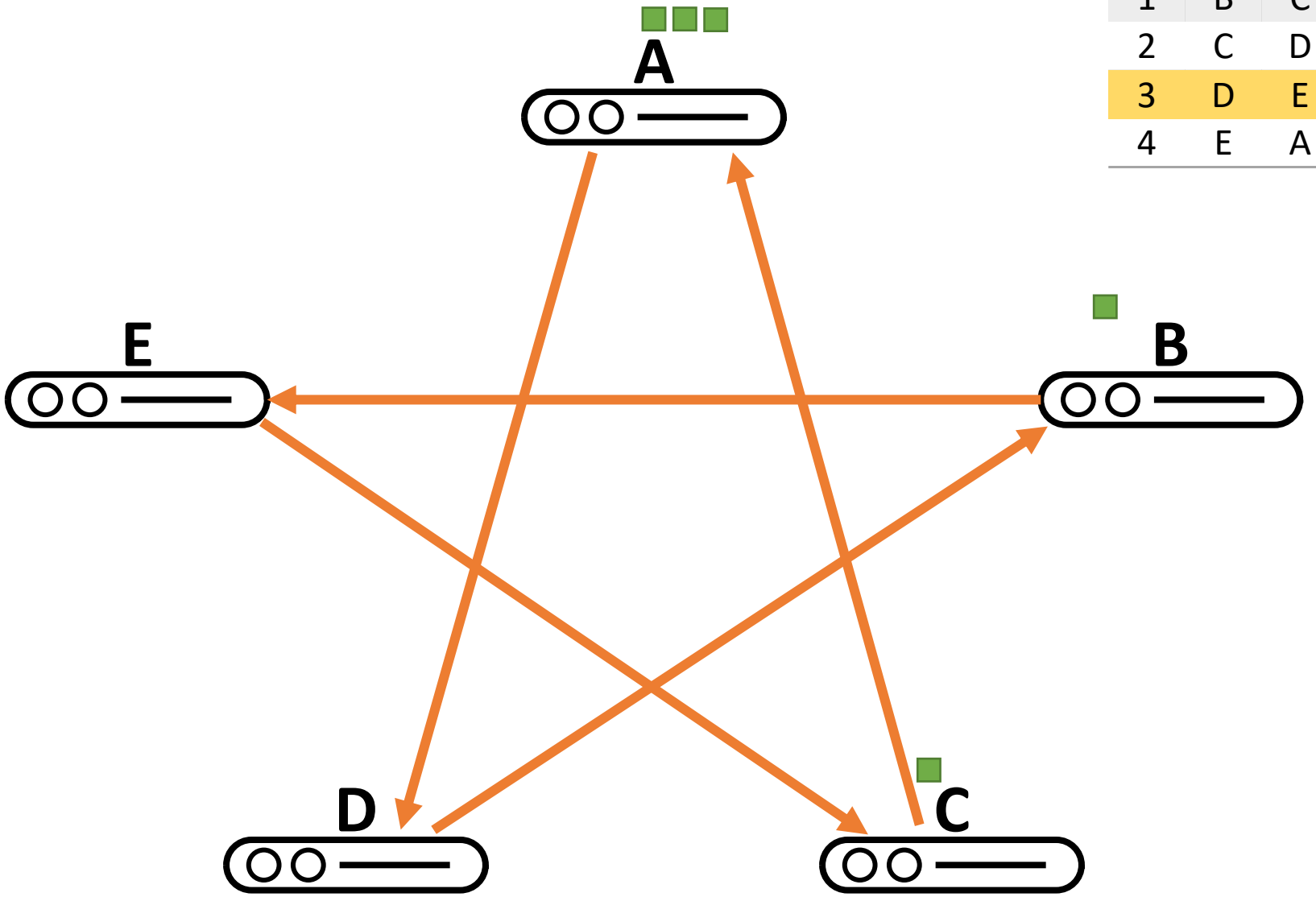
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



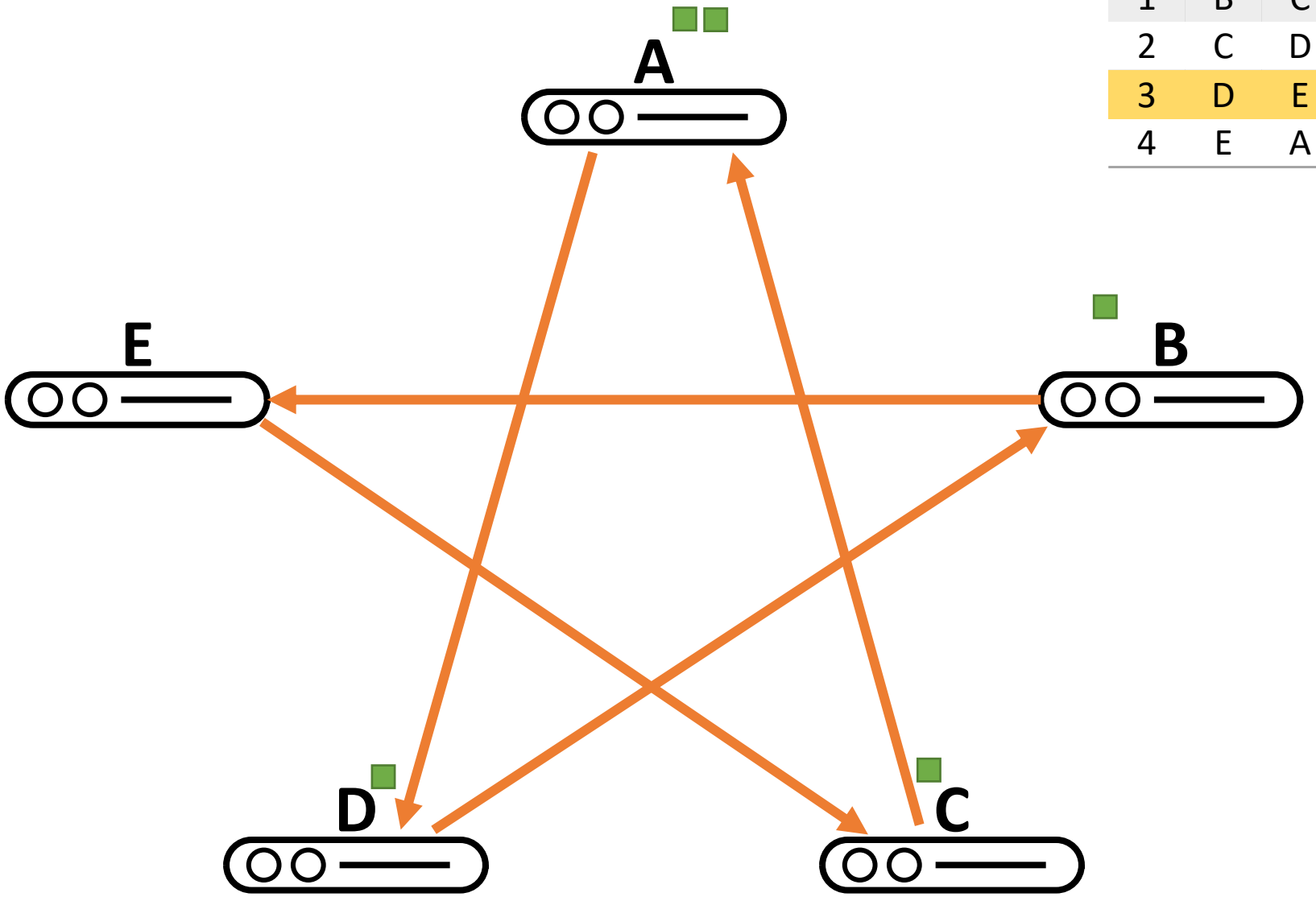
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



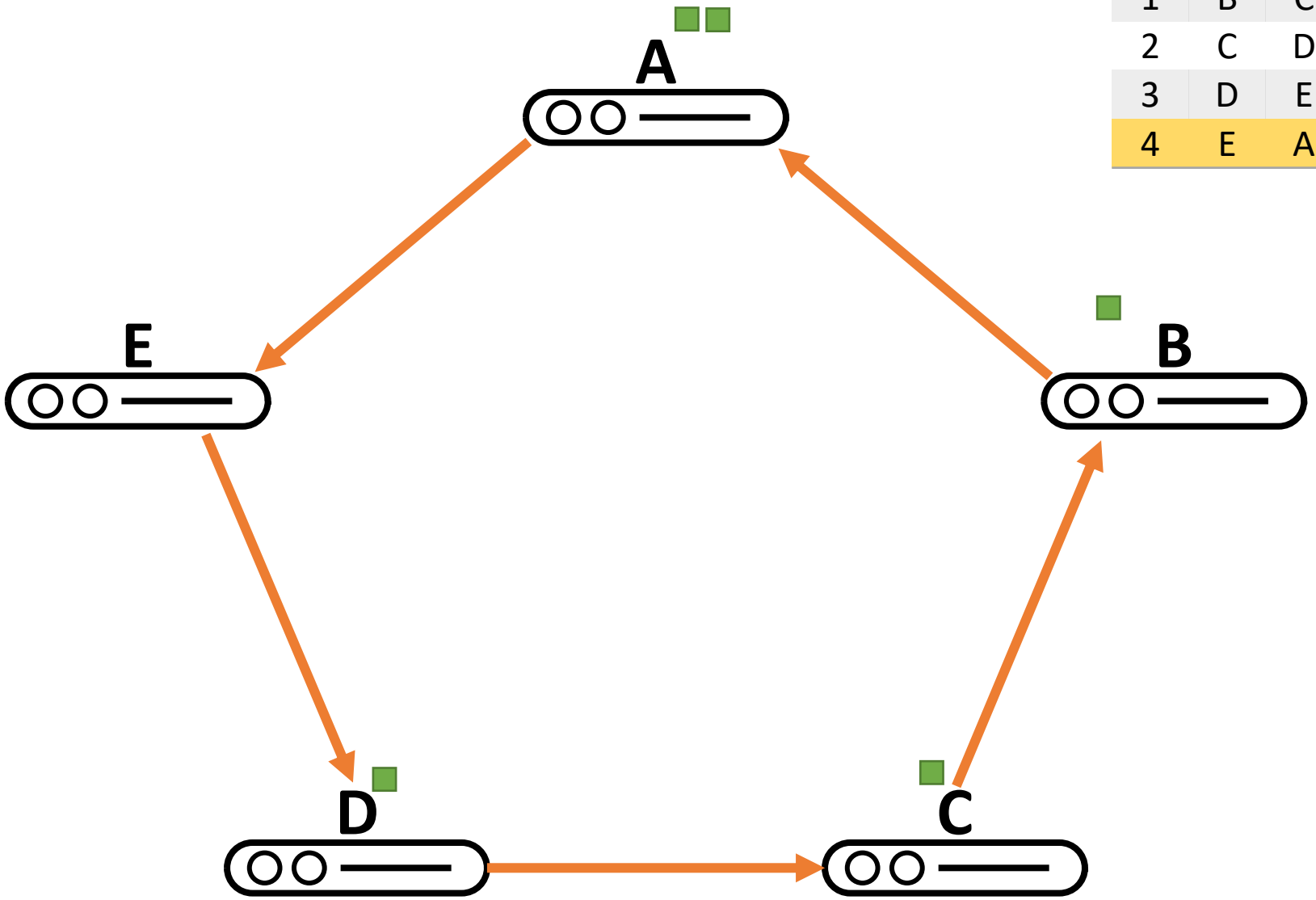
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



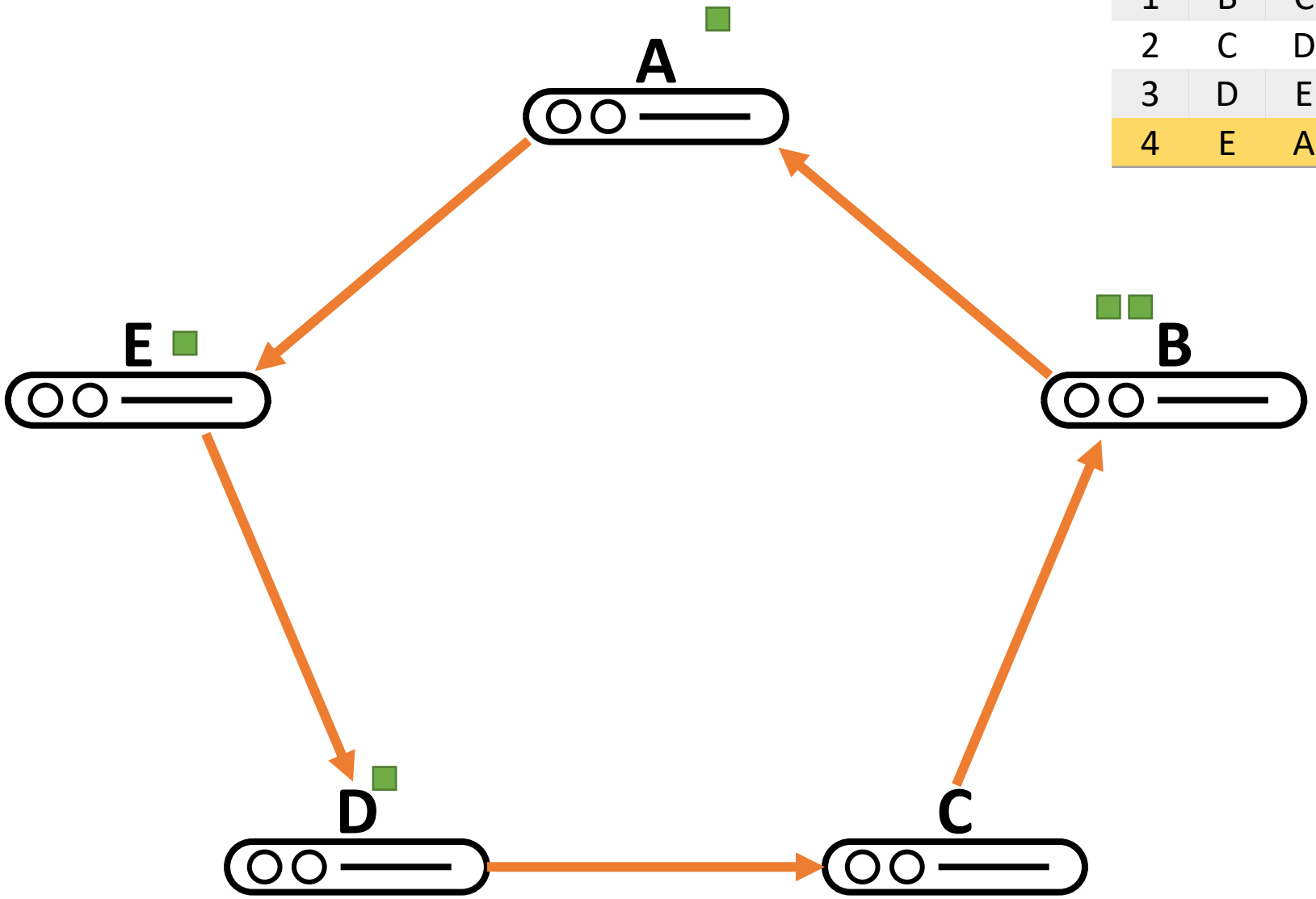
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



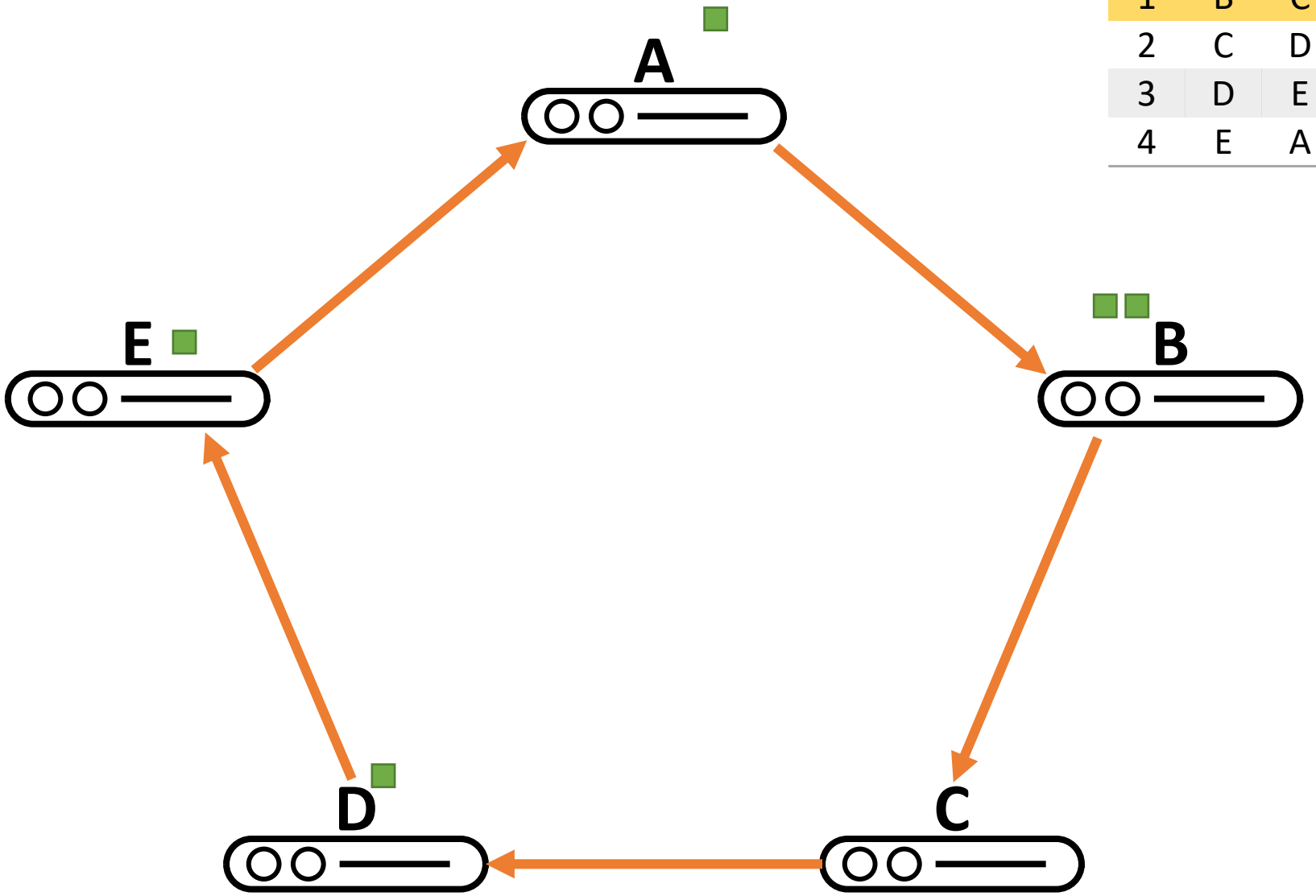
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



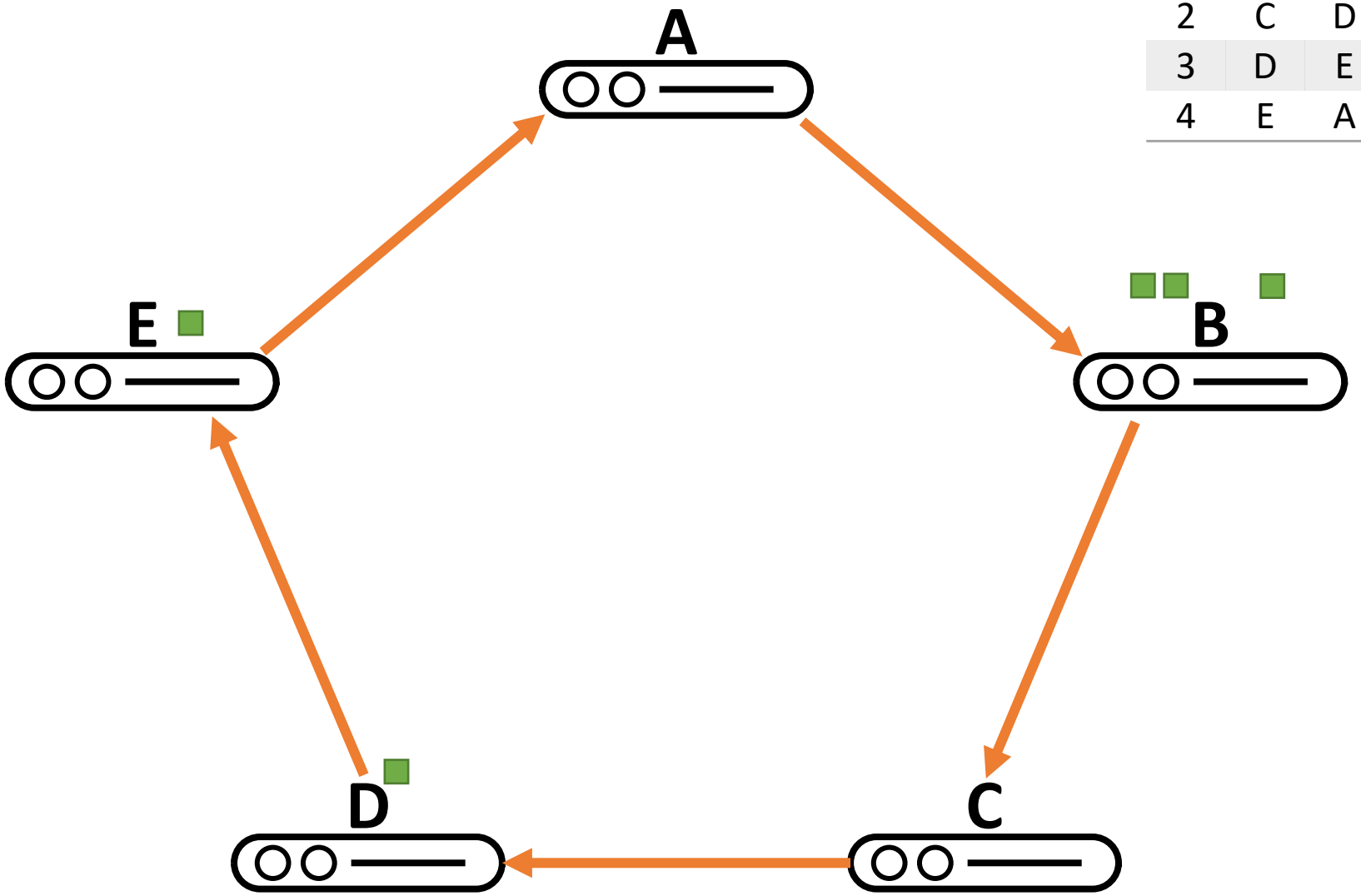
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



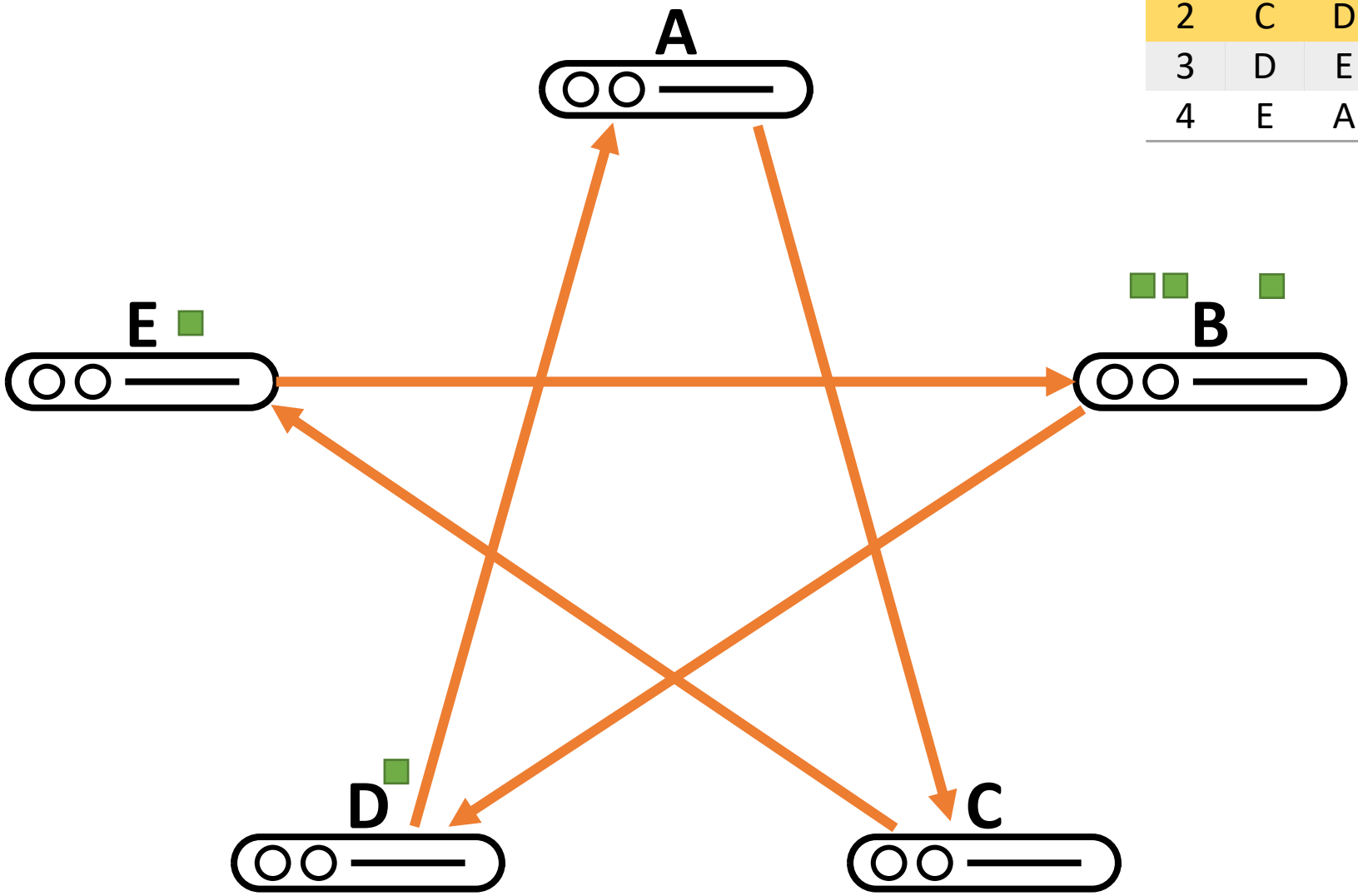
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



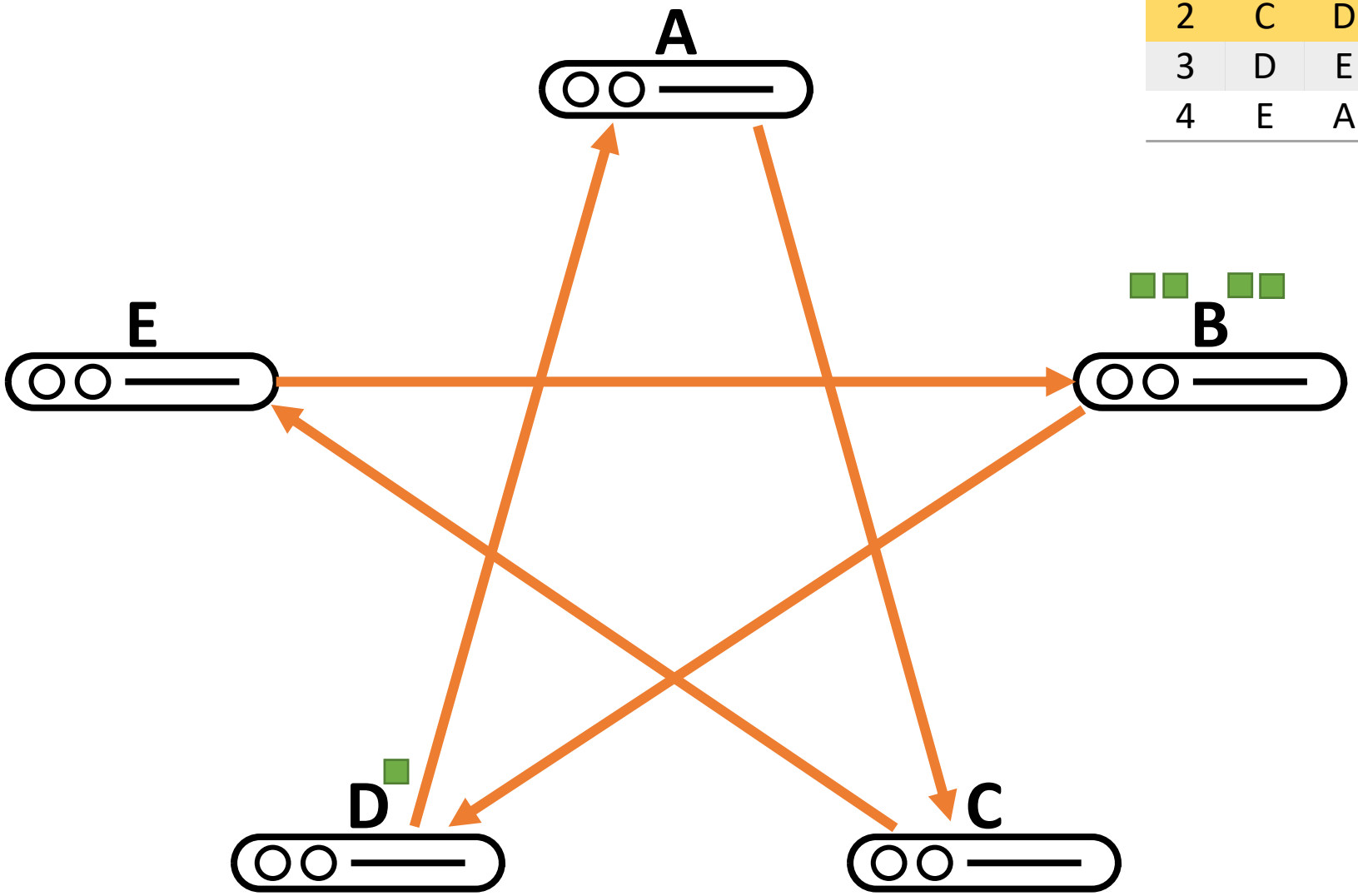
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



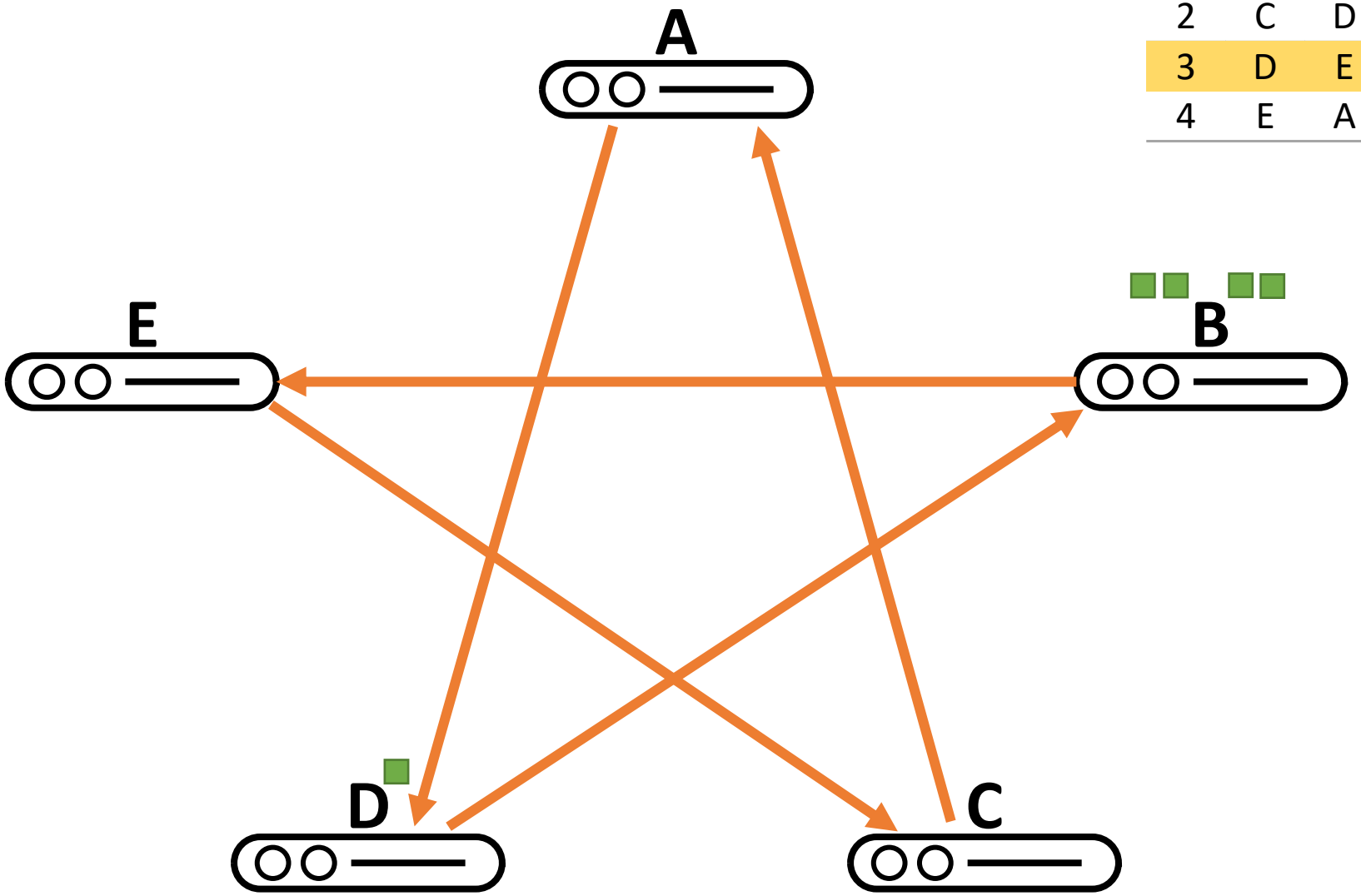
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



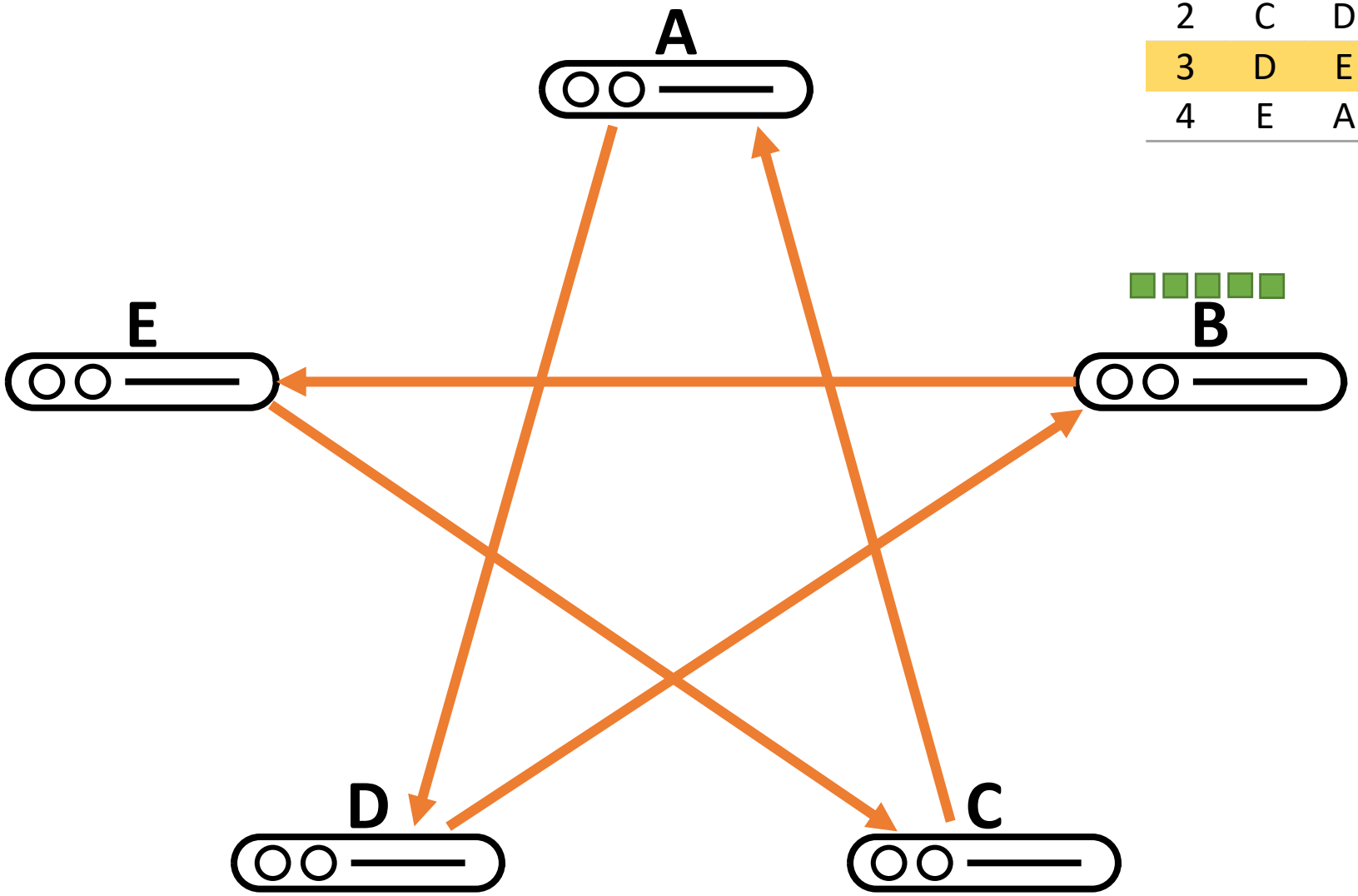
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



| | 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 |



| | 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)

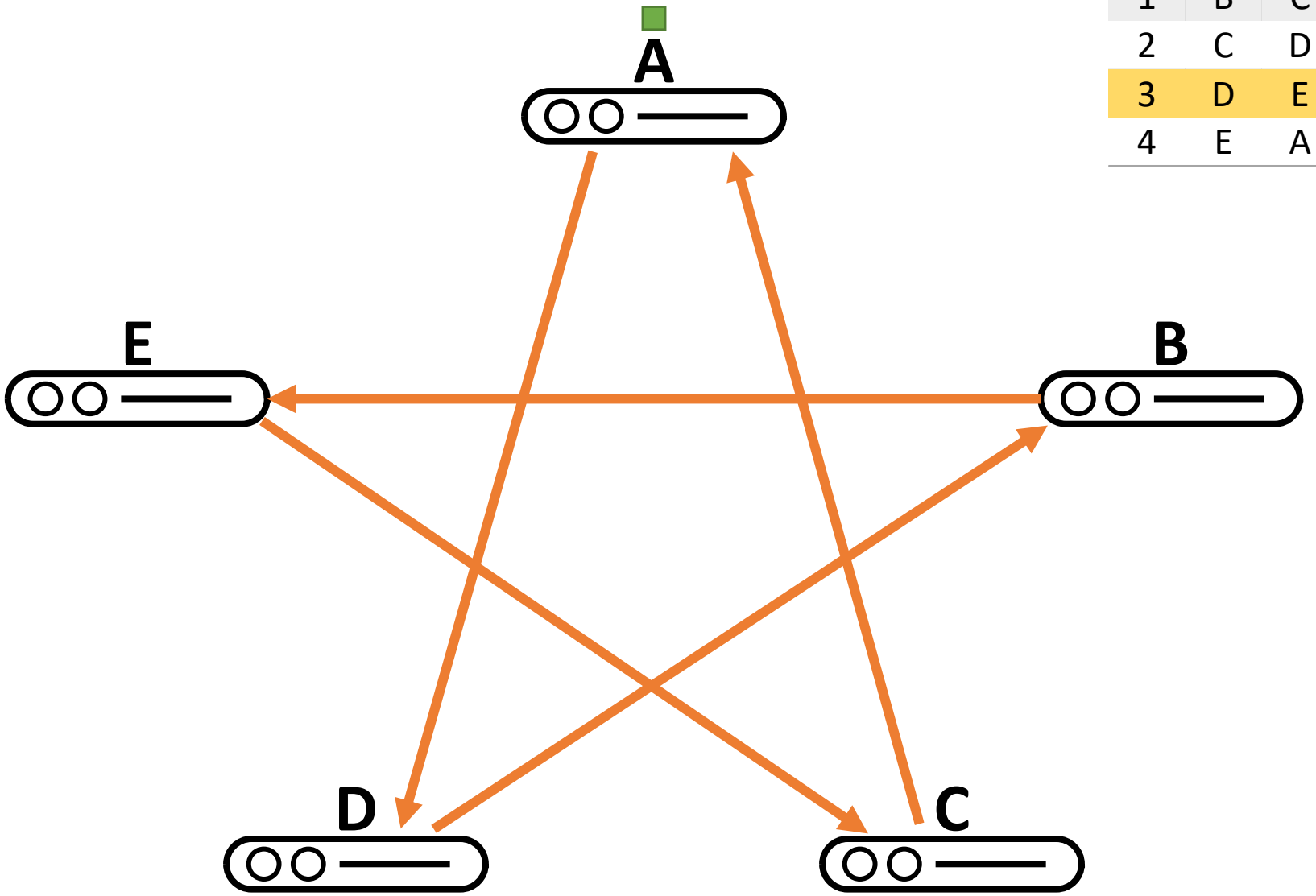
# Existing ORNs: Rotornet, Shoal, Sirius

- 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

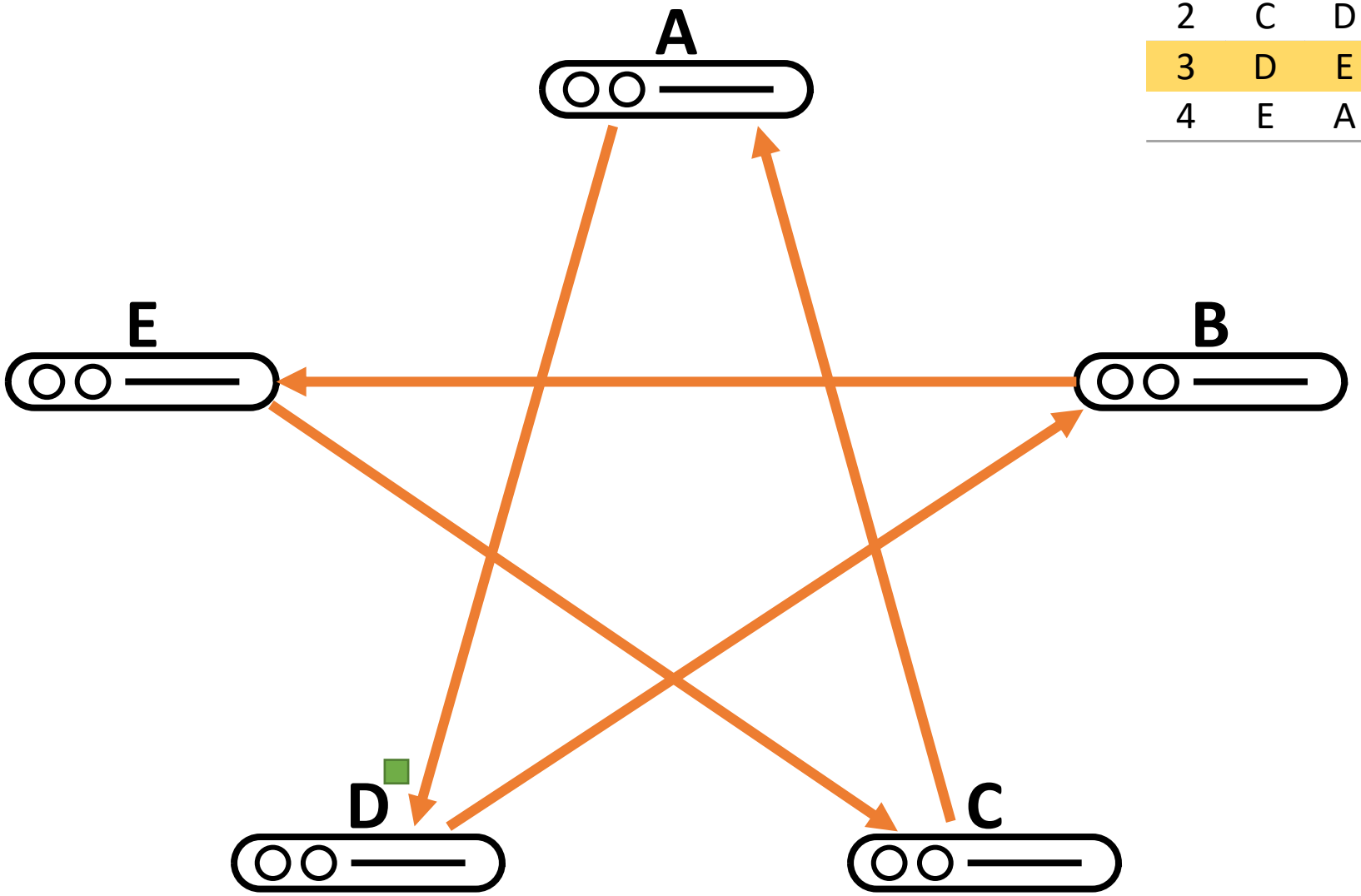
# Existing ORNs: Rotornet, Shoal, Sirius

- 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

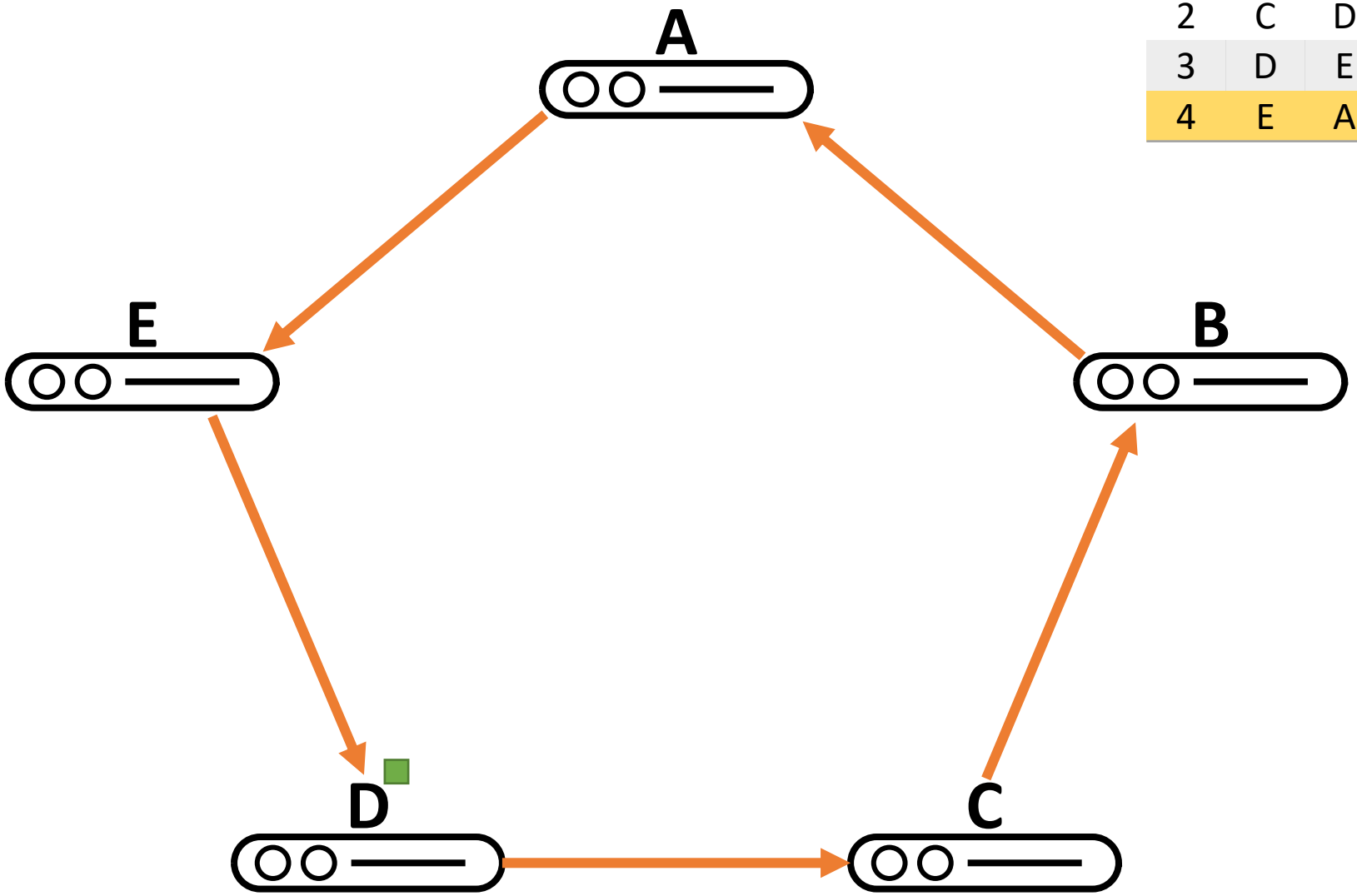
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



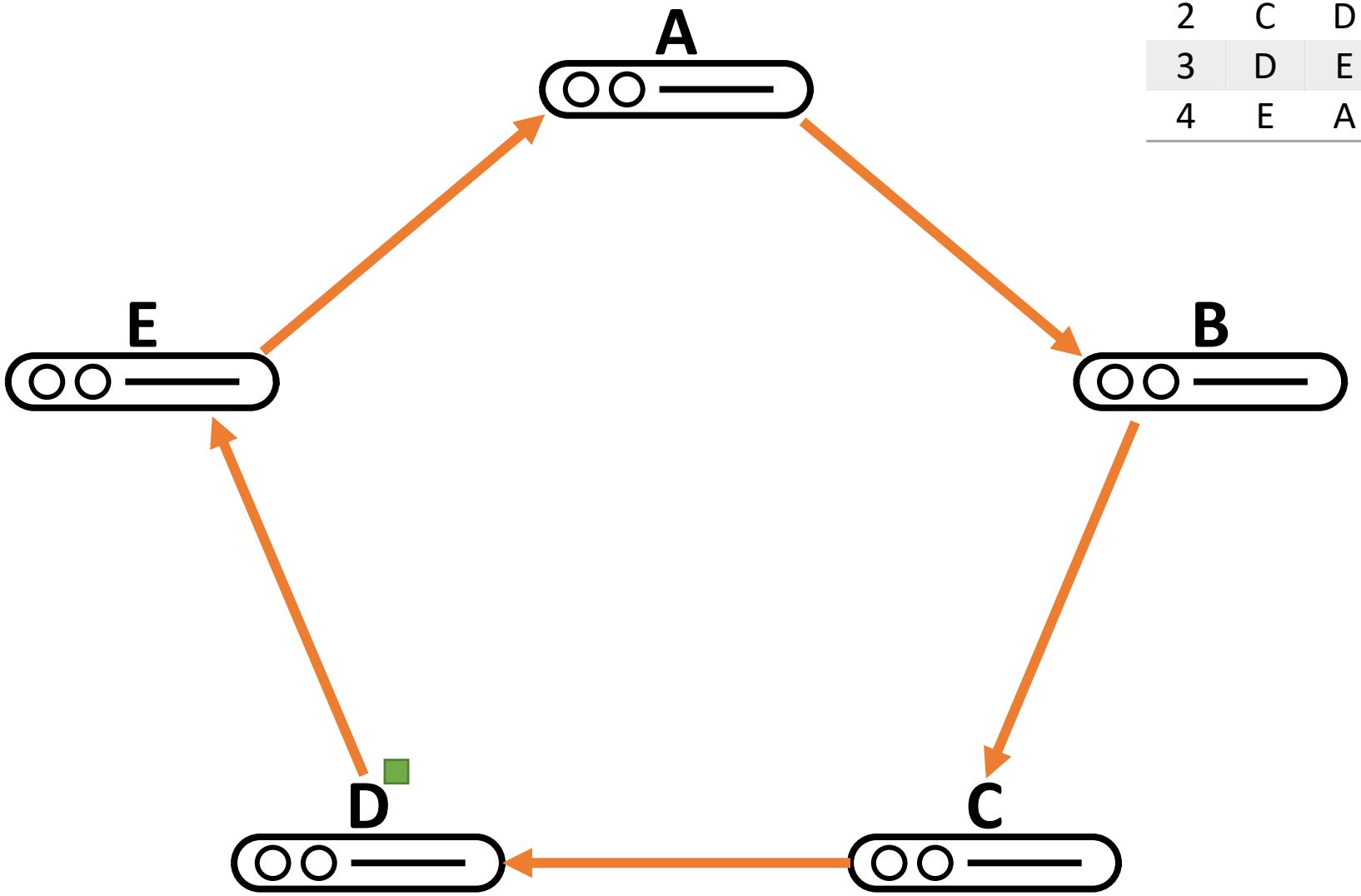
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



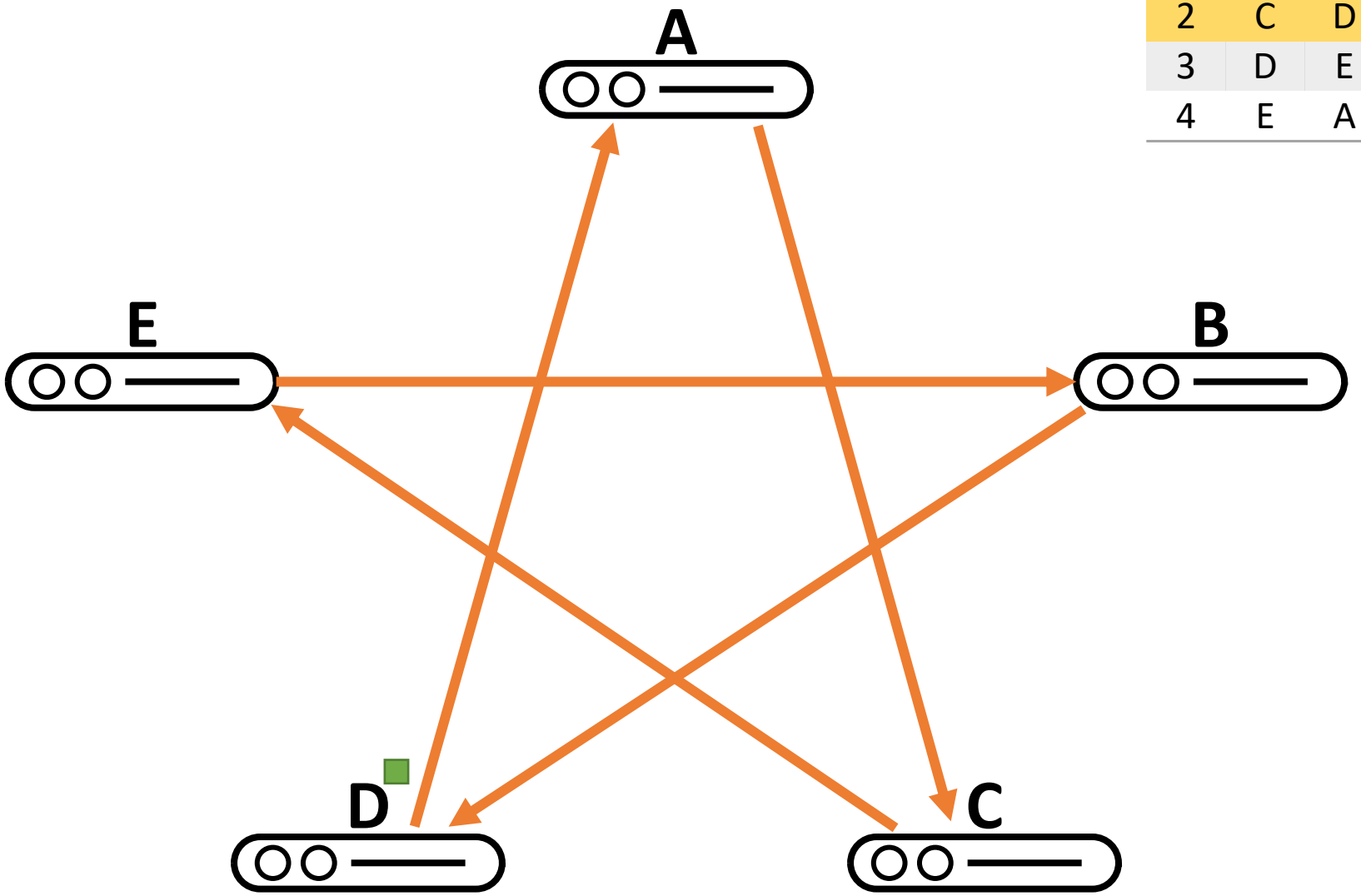
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



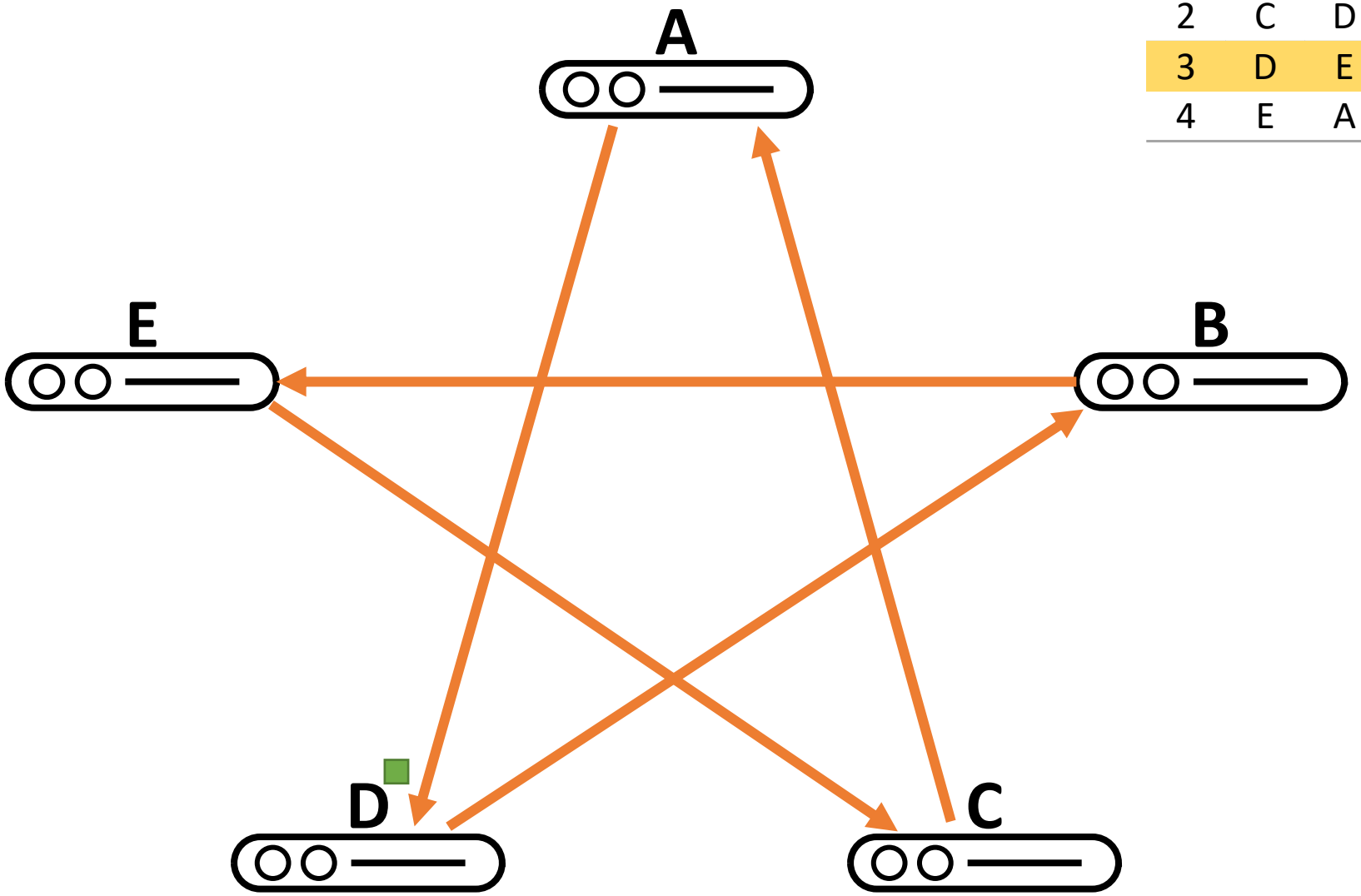
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



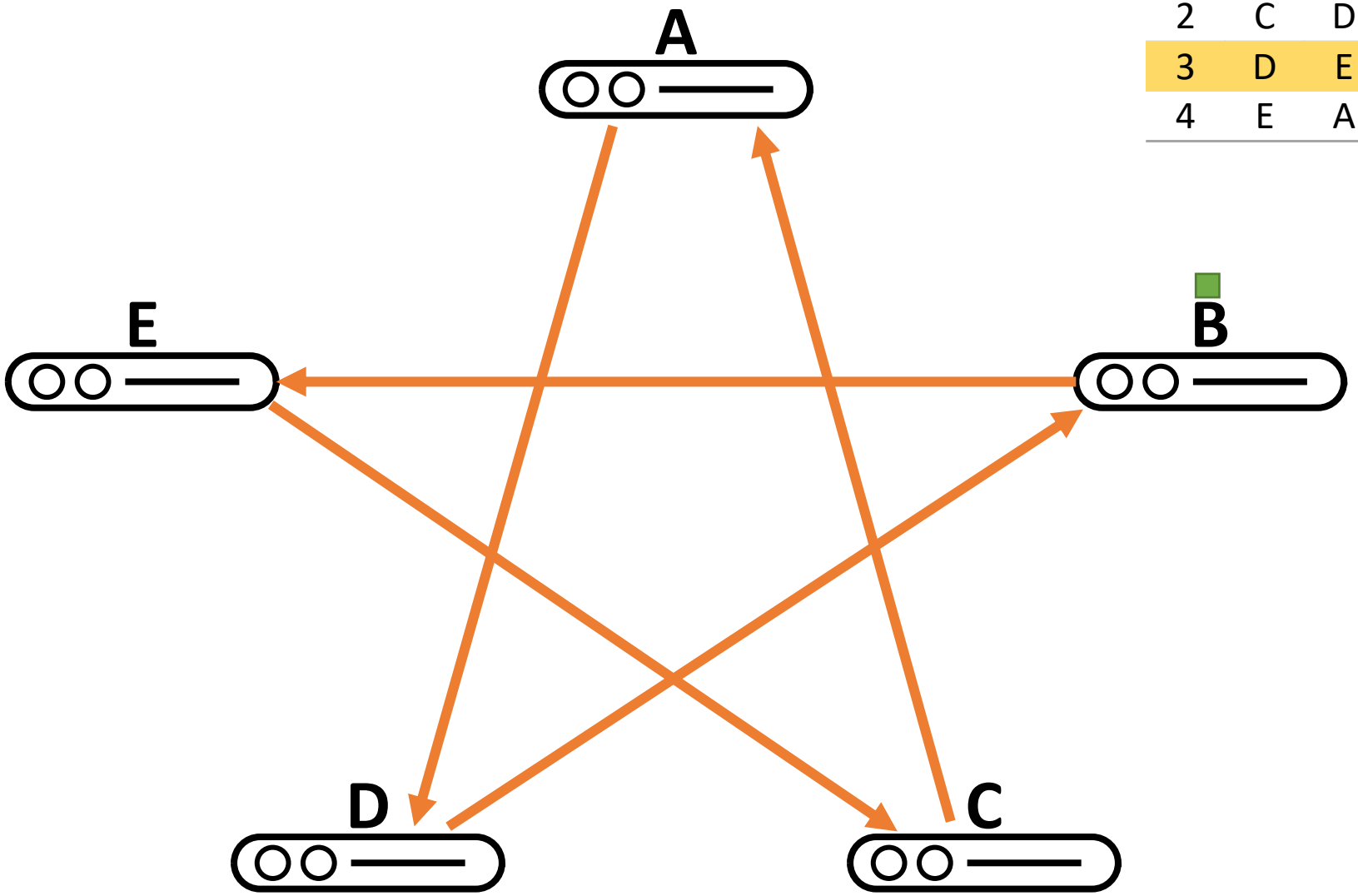
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | B | C | D | E | A |
| 2 | C | D | E | A | B |
| 3 | D | E | A | B | C |
| 4 | E | A | B | C | D |



| | 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
  - After first hop, you may need to wait for an entire round-robin.

# 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

# 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
- 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

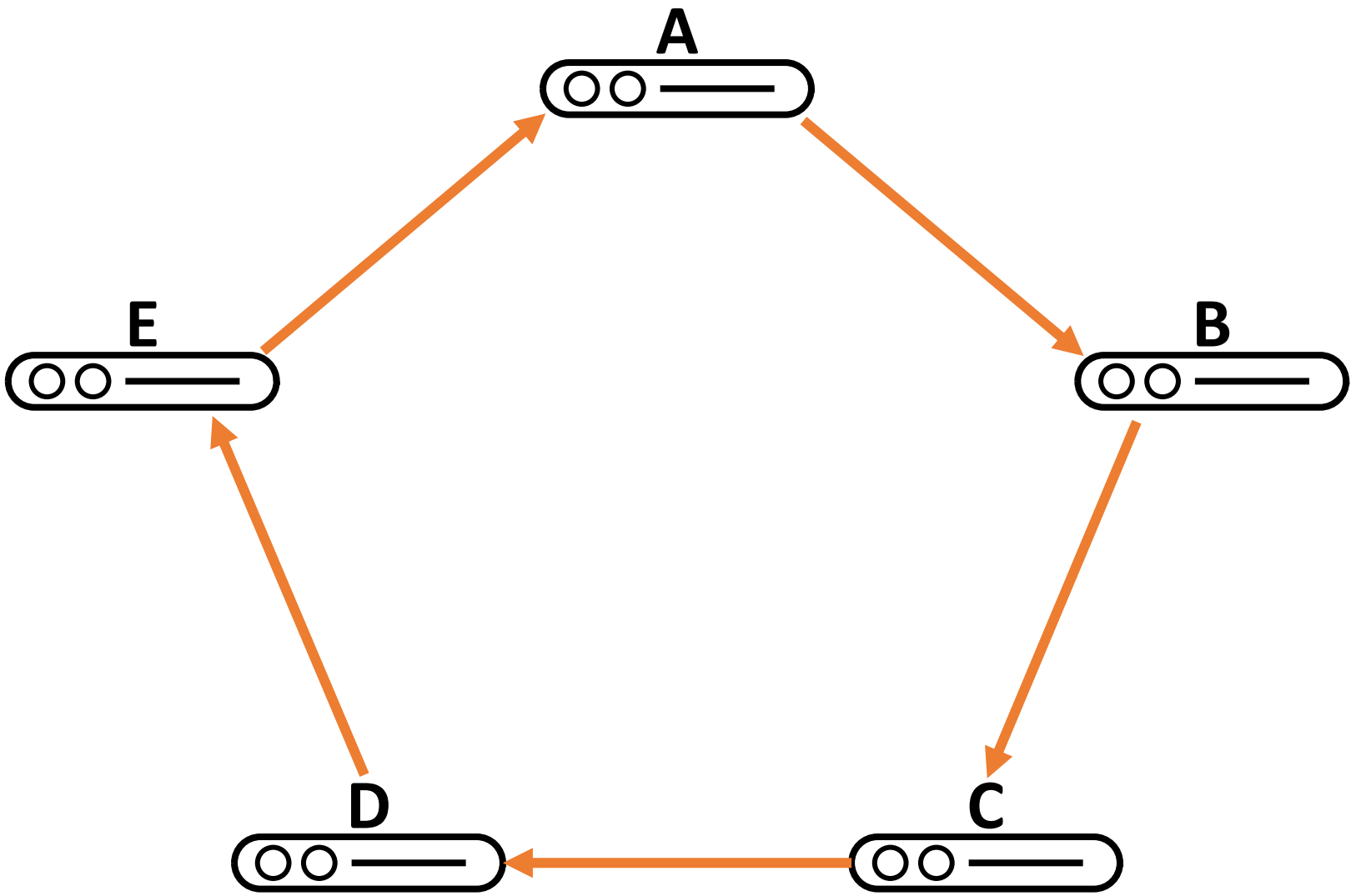
A

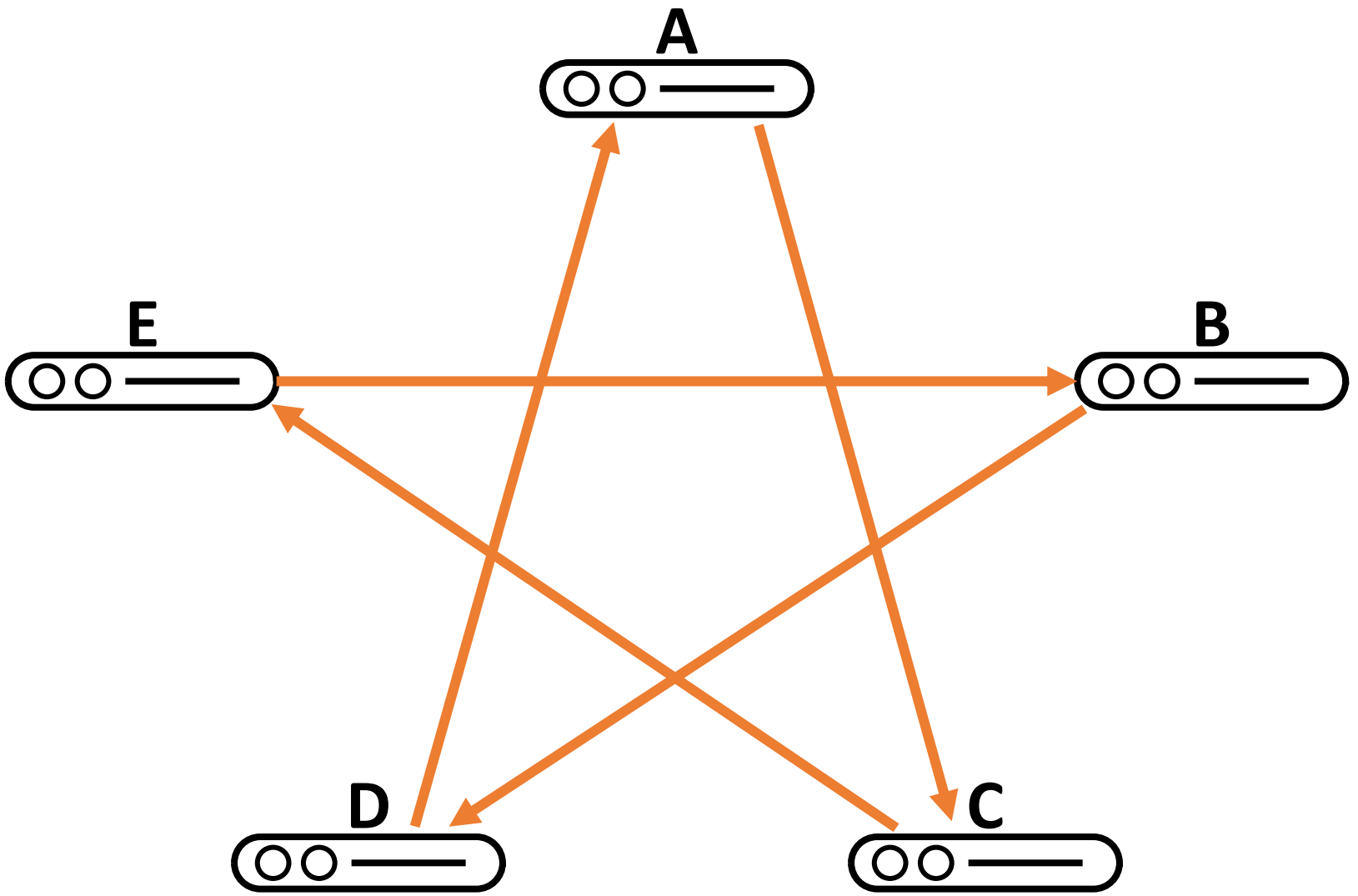
E

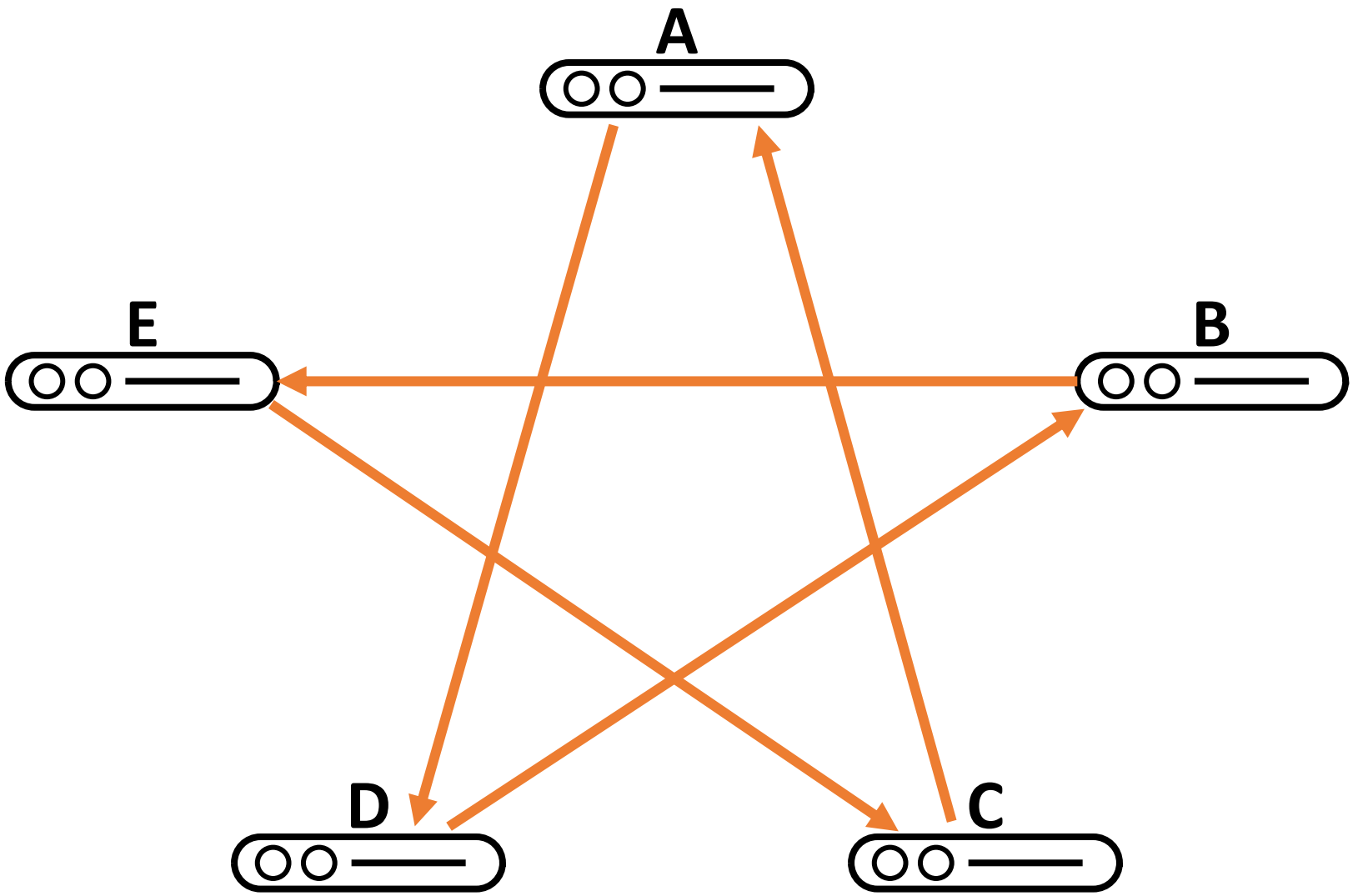
B

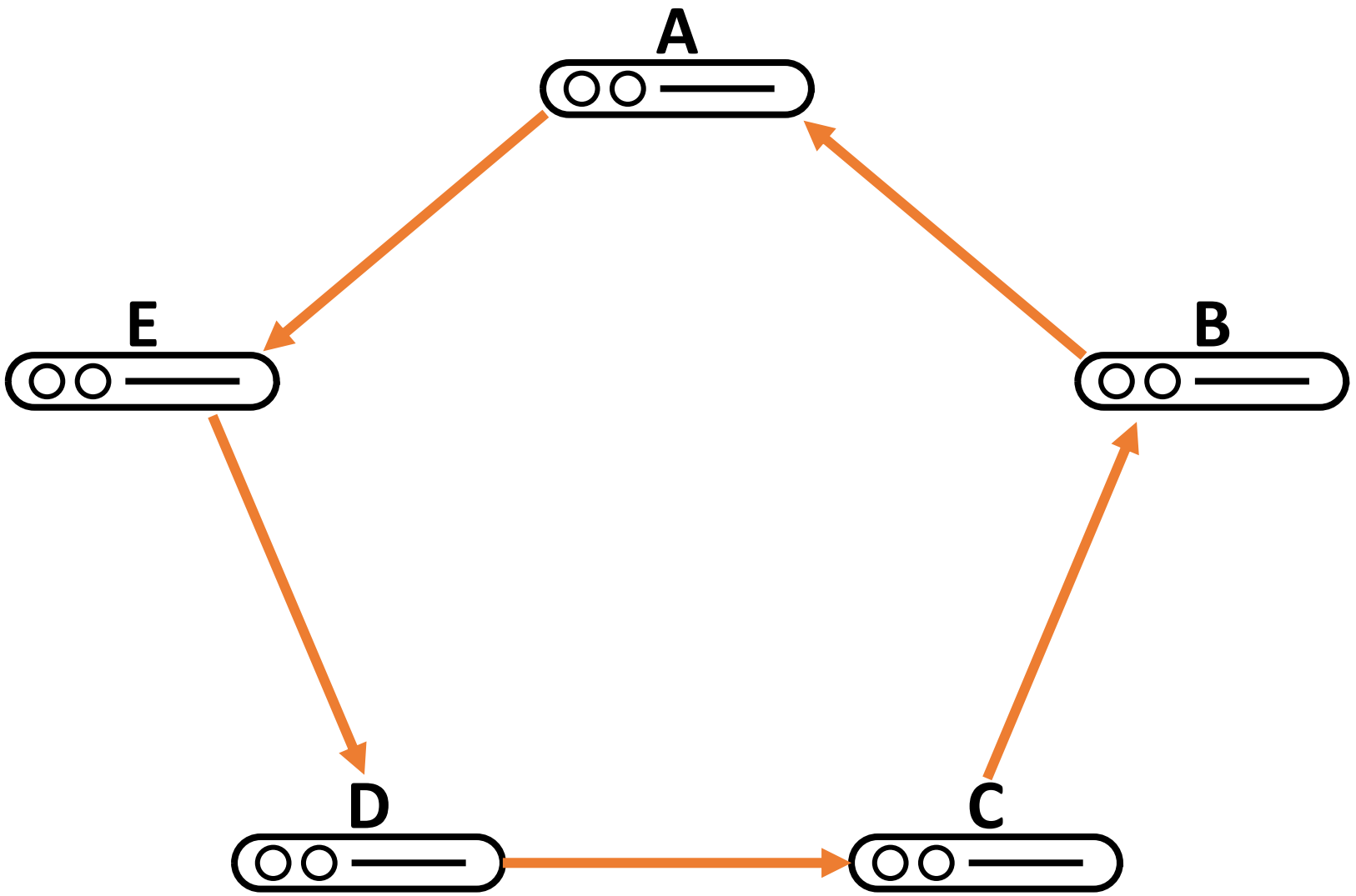
D

C


A
B
C
D
E


A
E
B
D
C


A
E
B
D
C


A
B
C
D
E

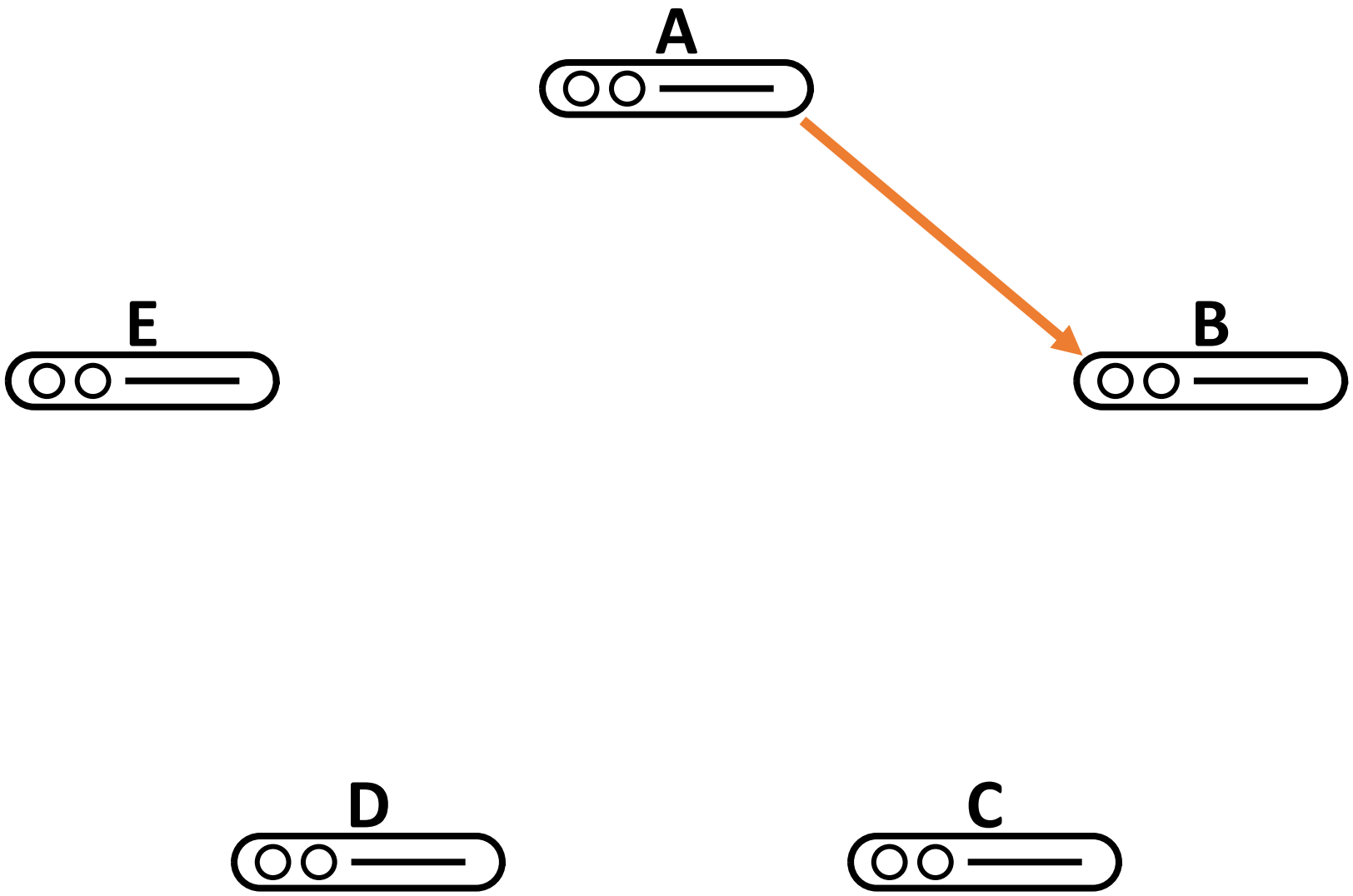
A

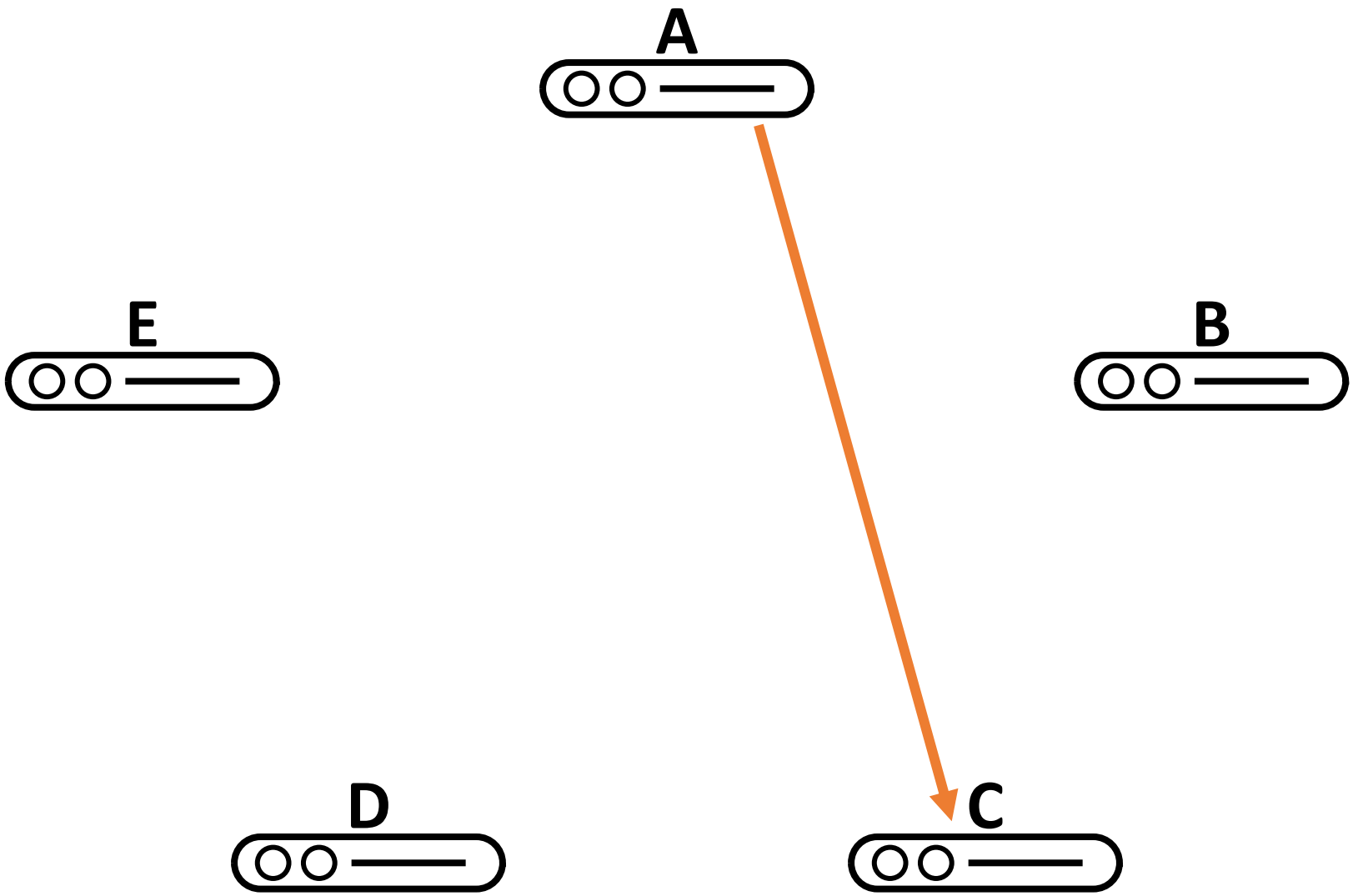
E

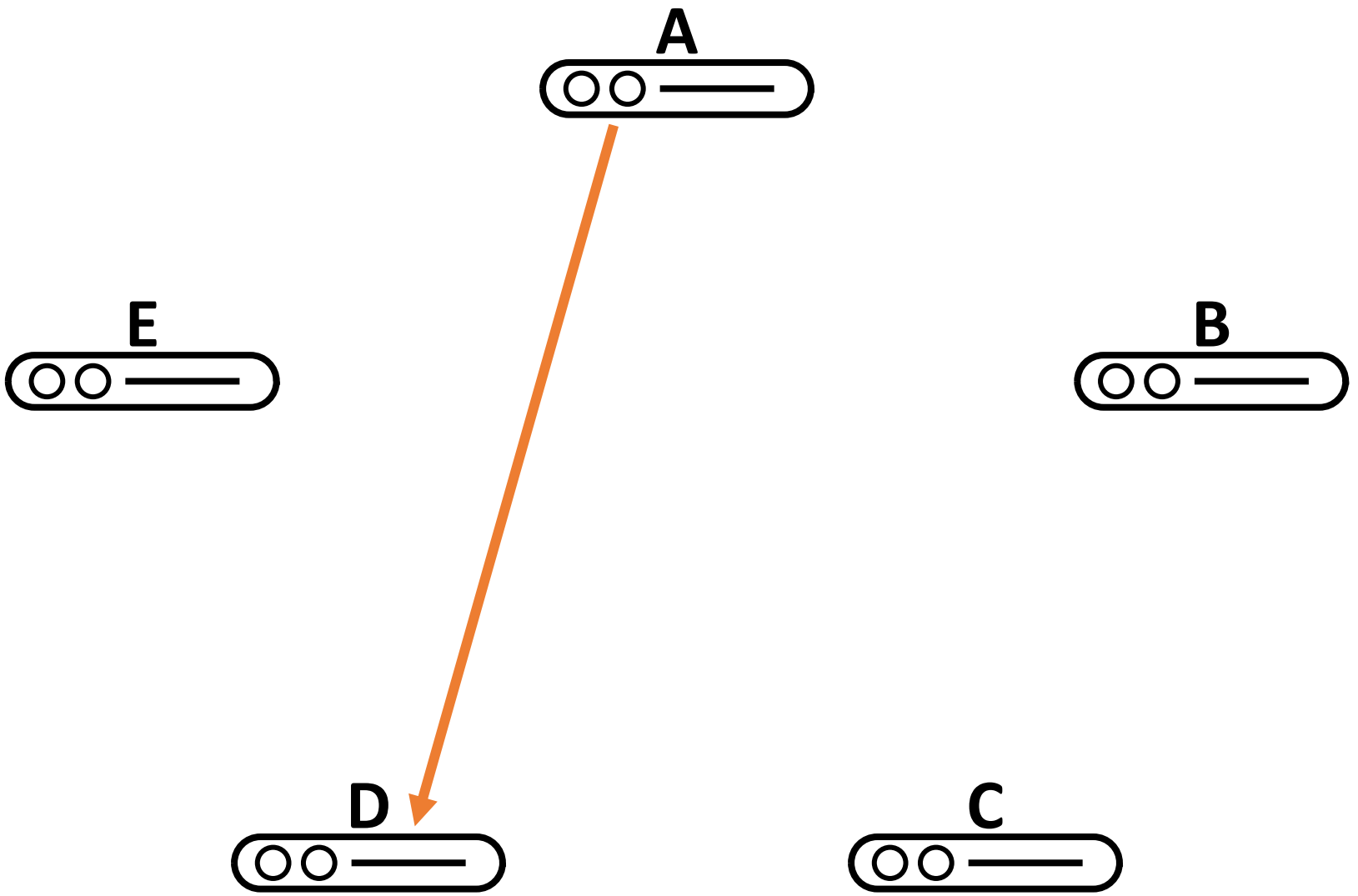
B

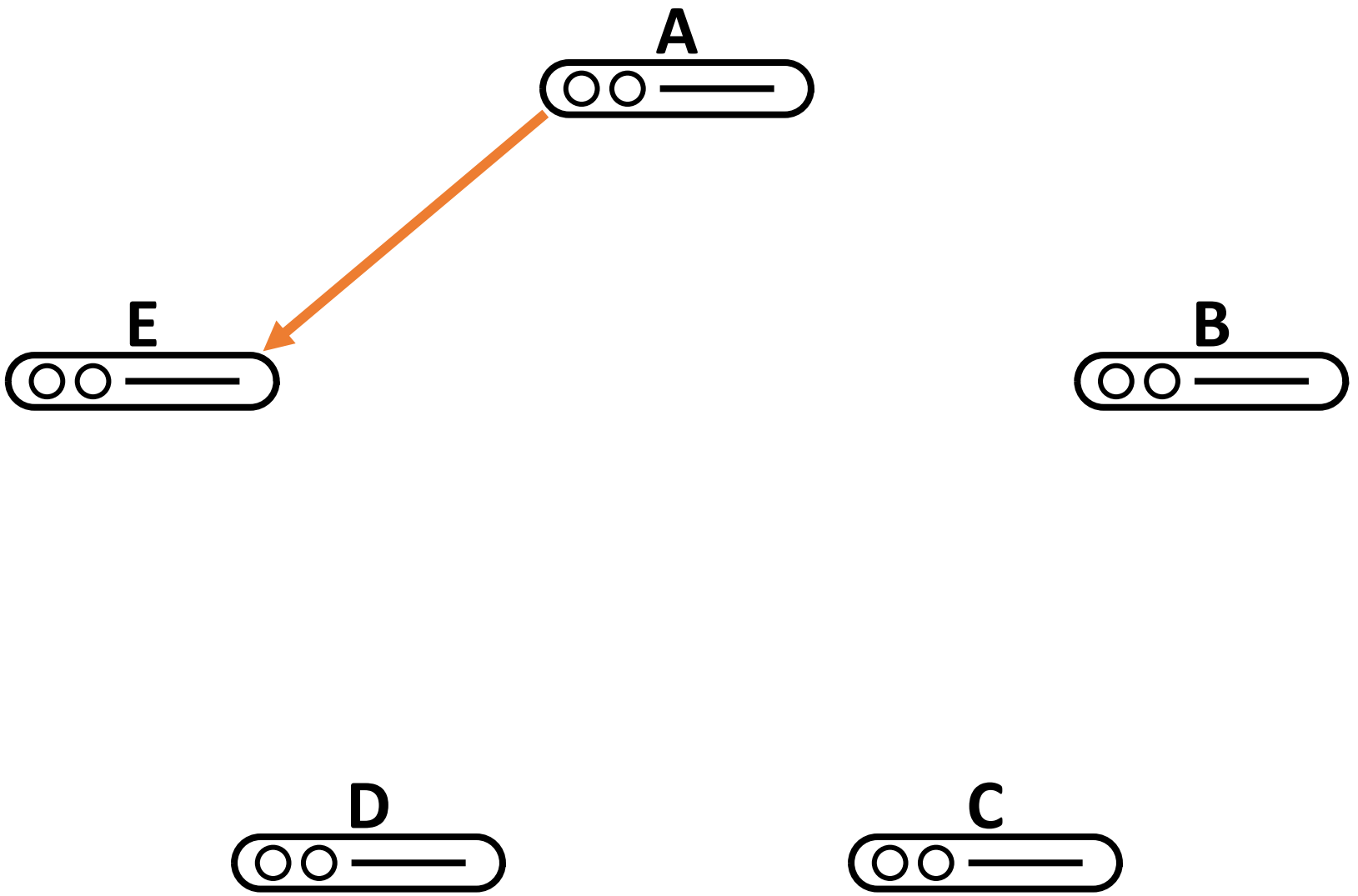
D

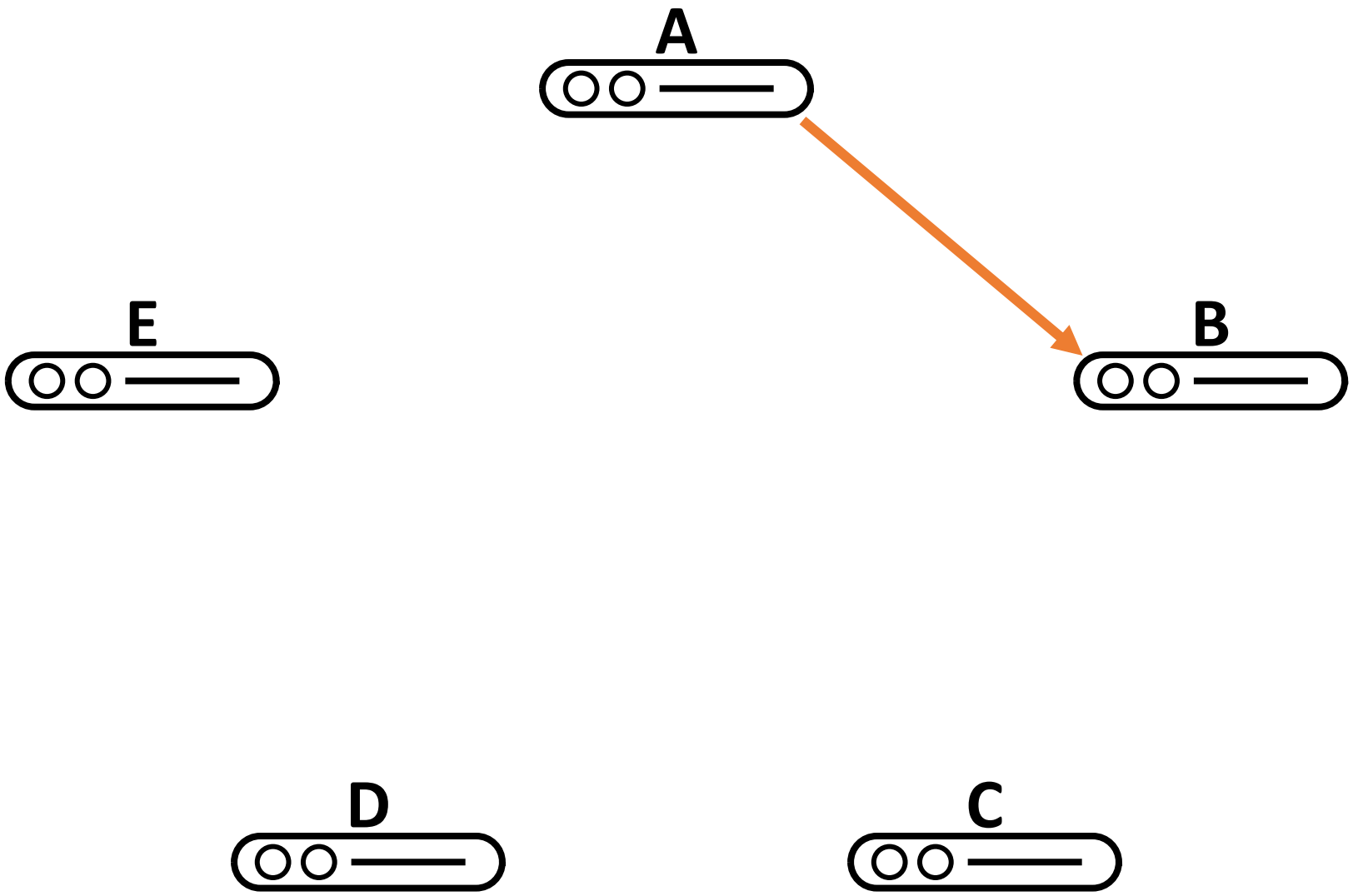
C

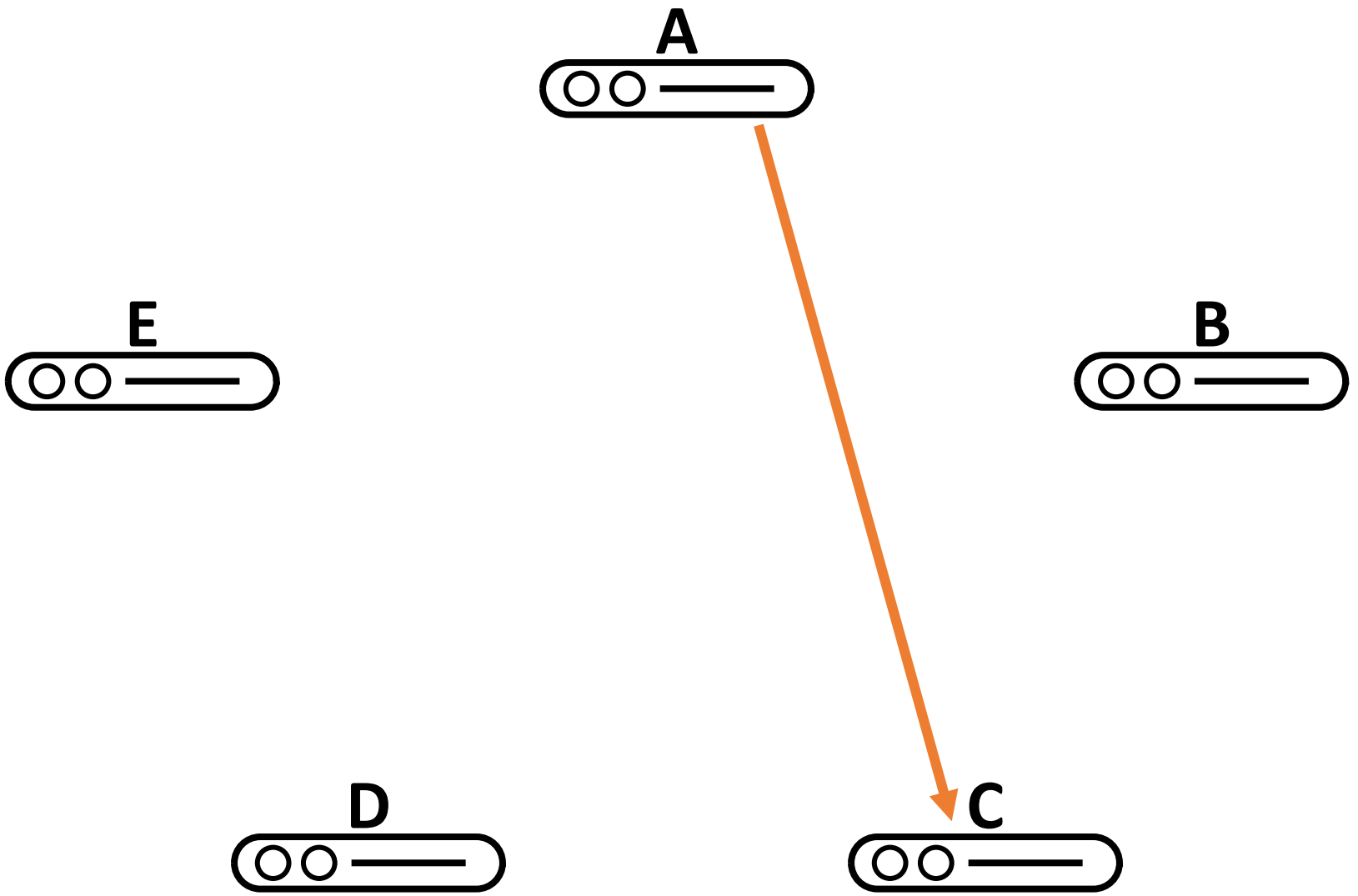

A
E
B
D
C

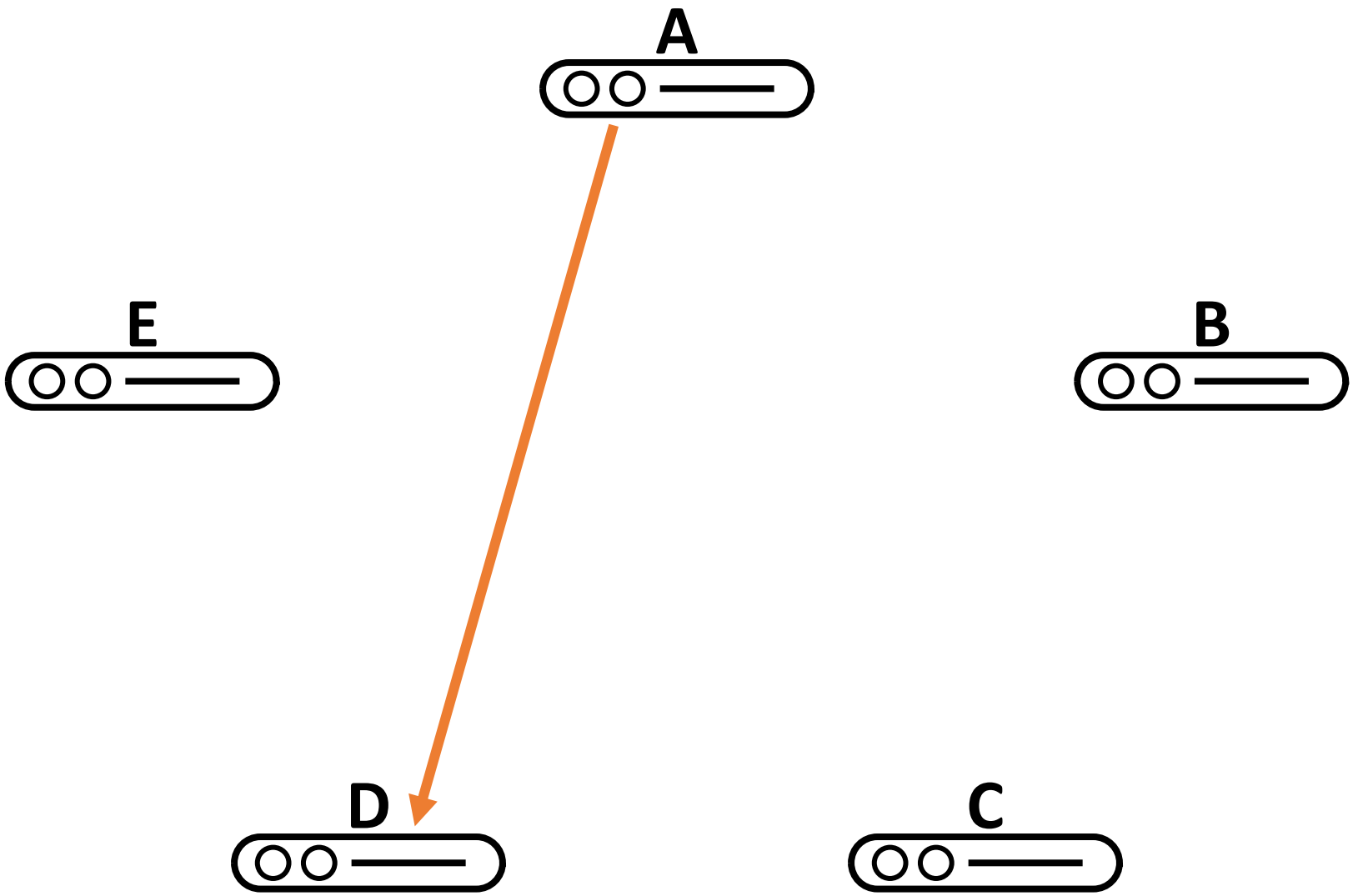

A
E
B
D
C

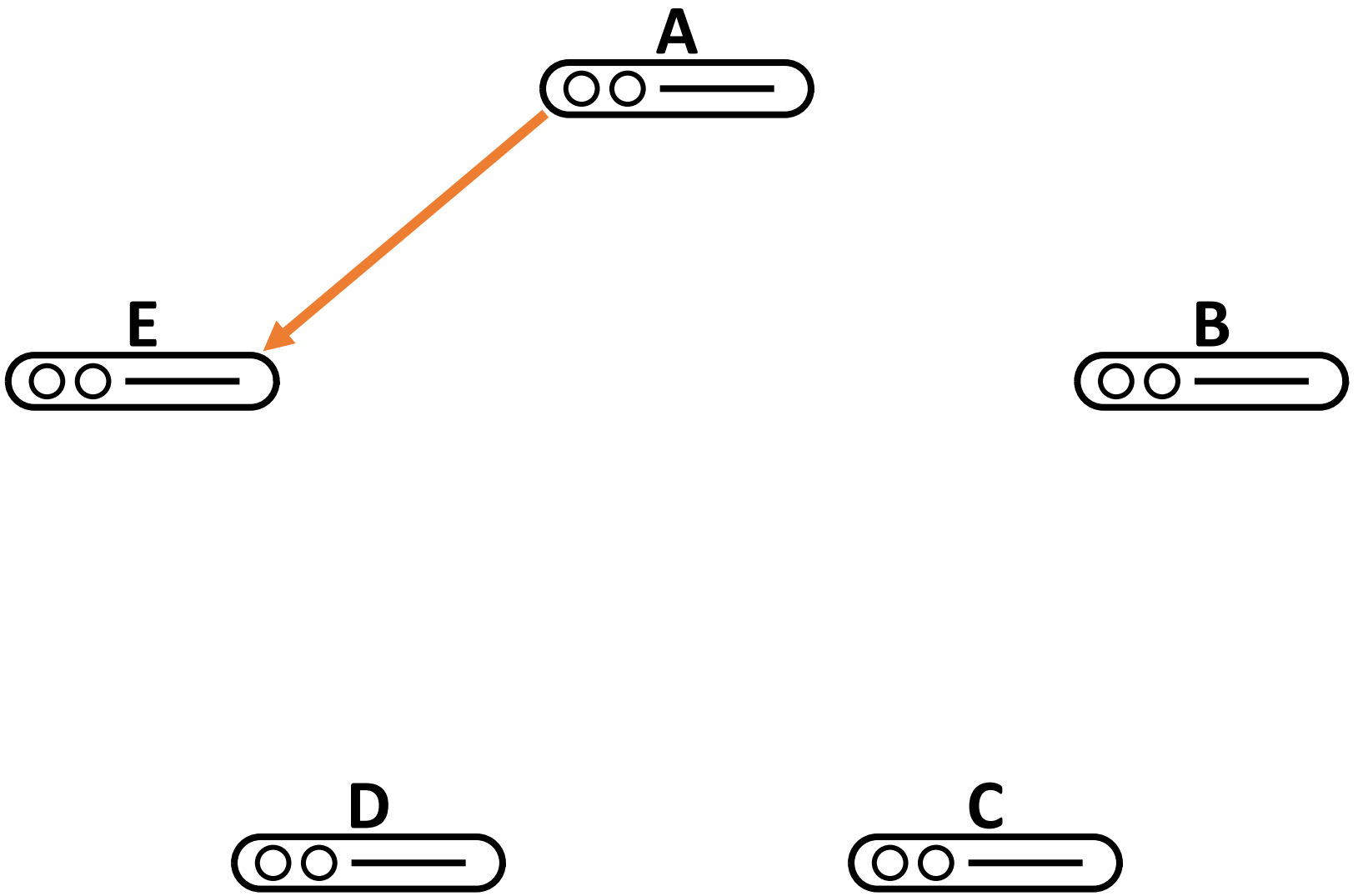

A
E
B
D
C


A
E
B
D
C


A
E
B
D
C


A
E
B
D
C


A
E
B
D
C


A
E
B
D
C

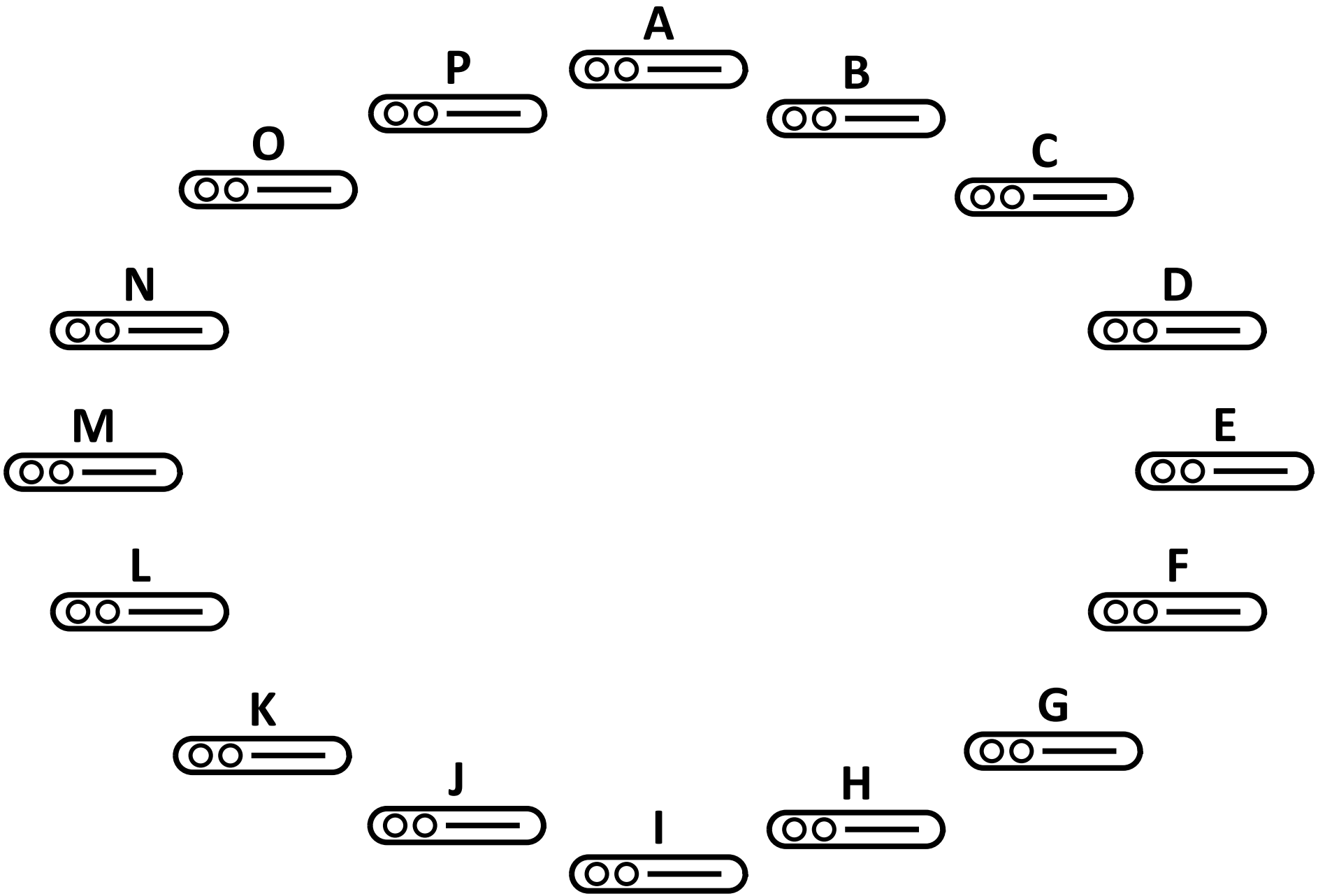
A

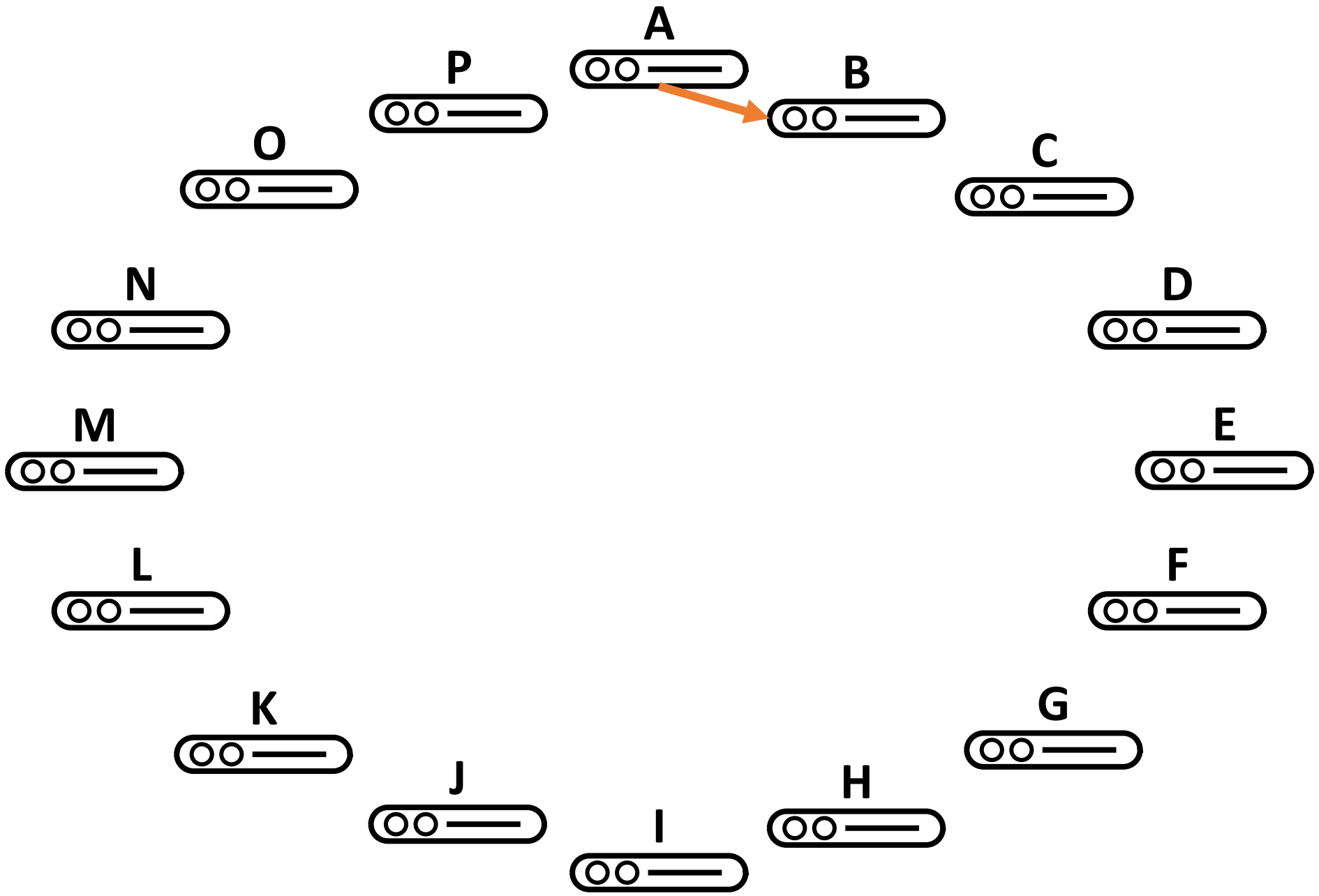
E

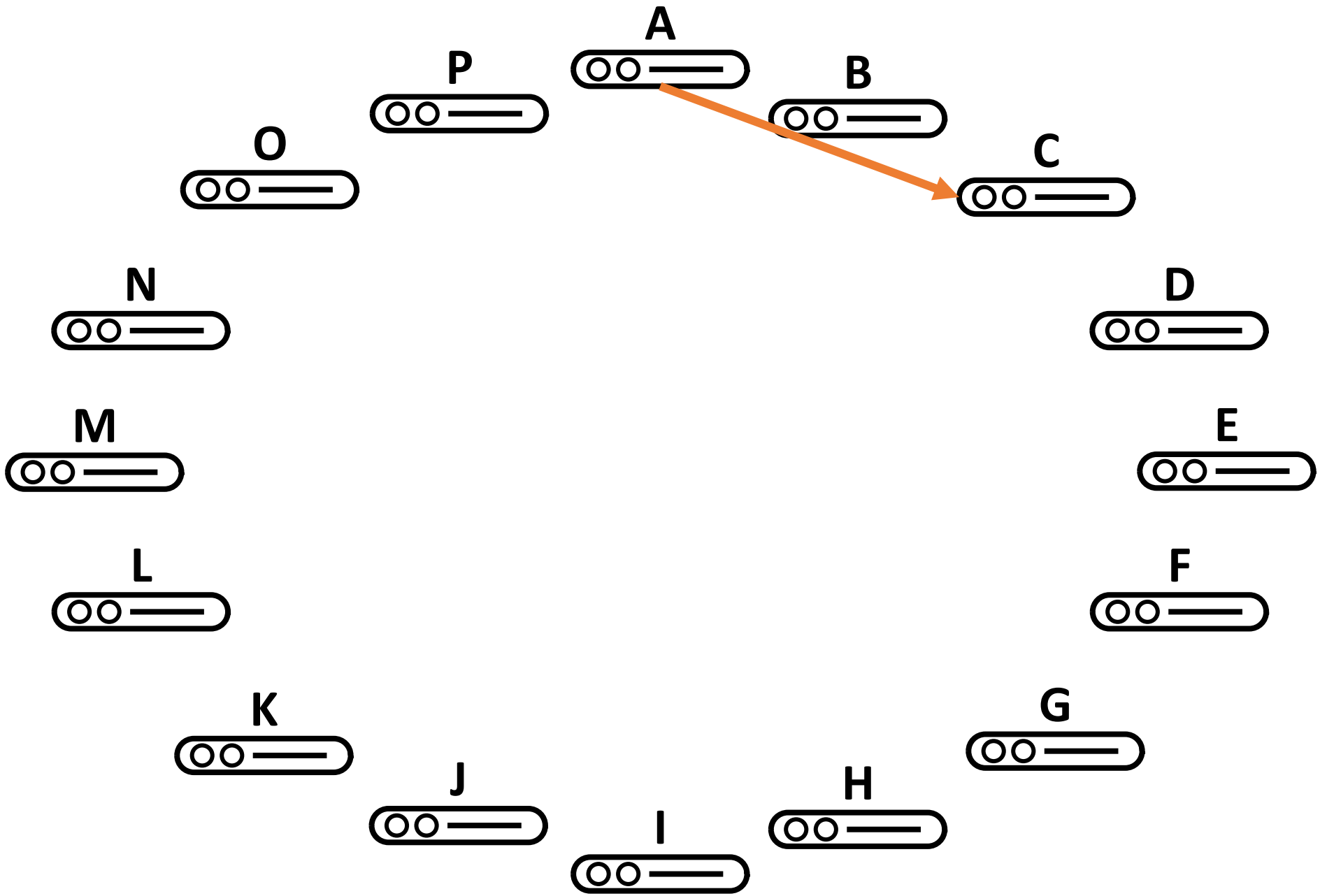
B

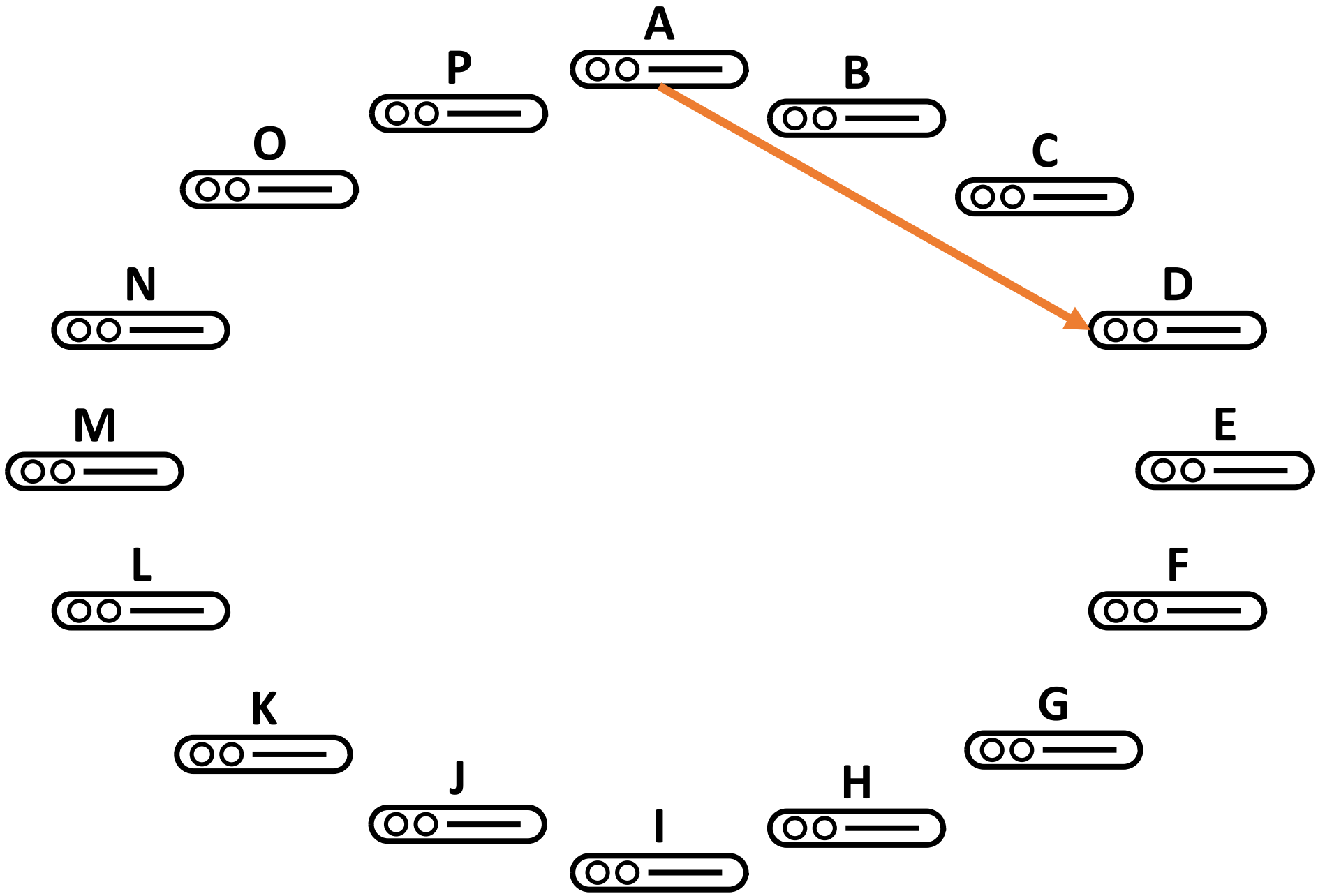
D

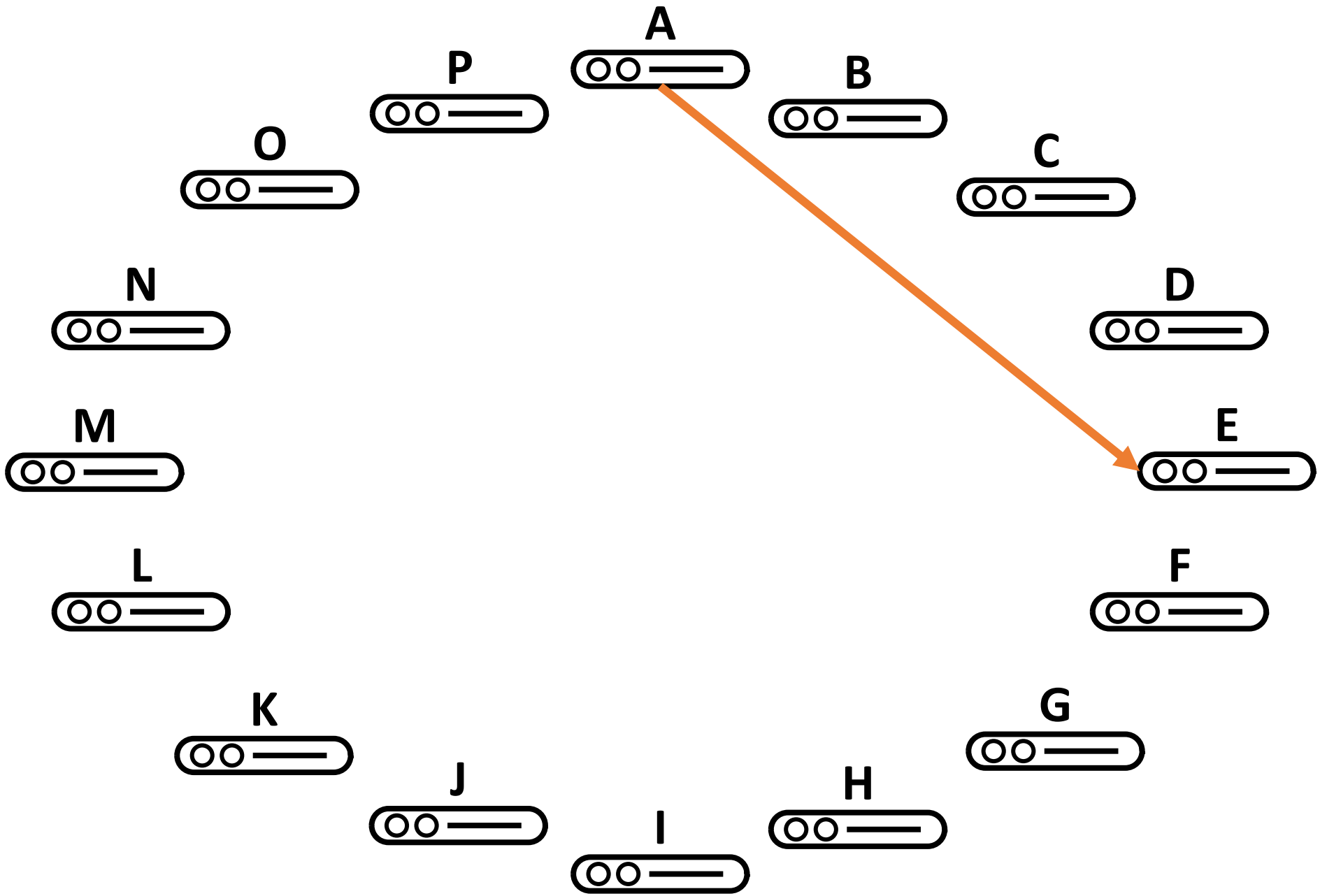
C

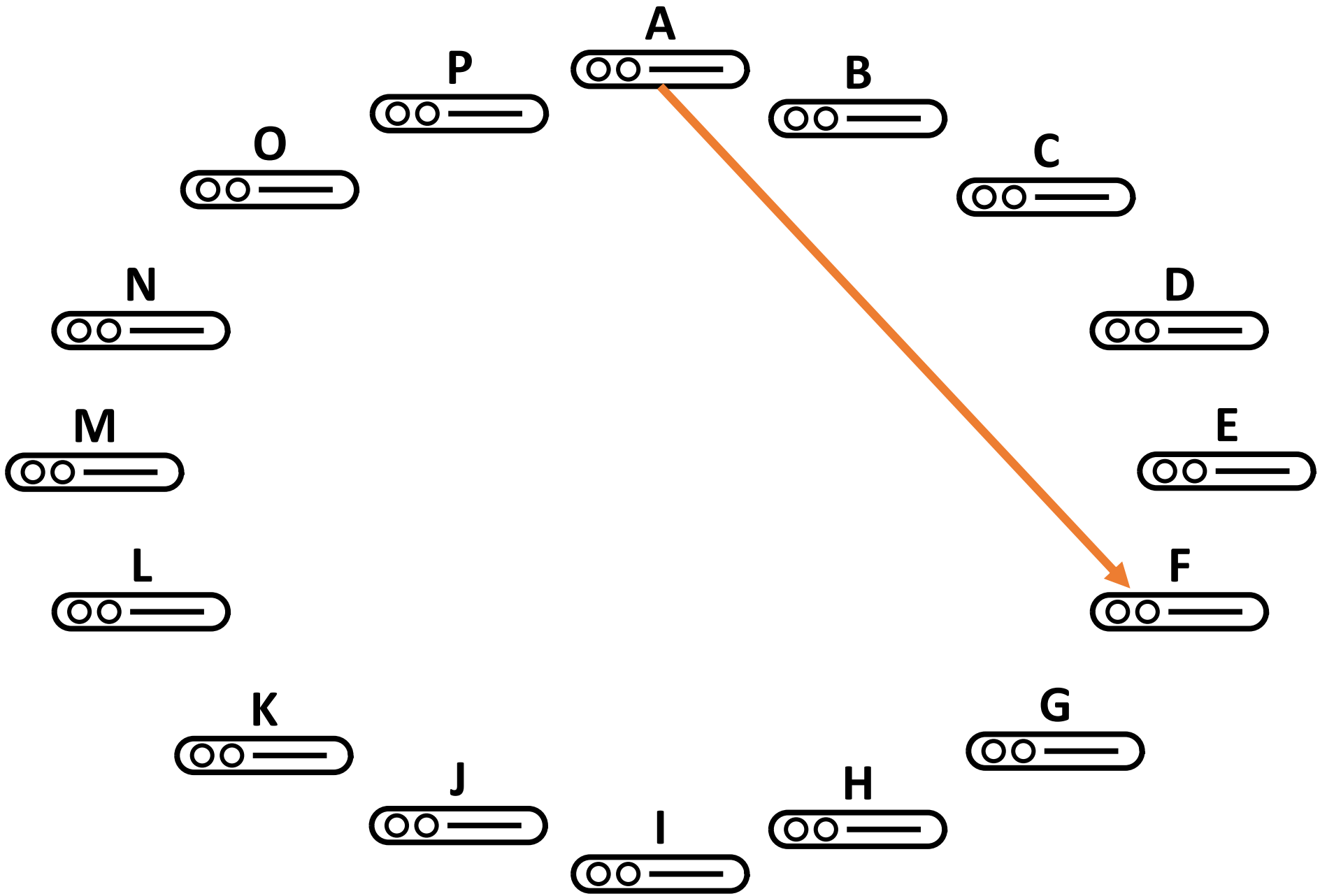

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P

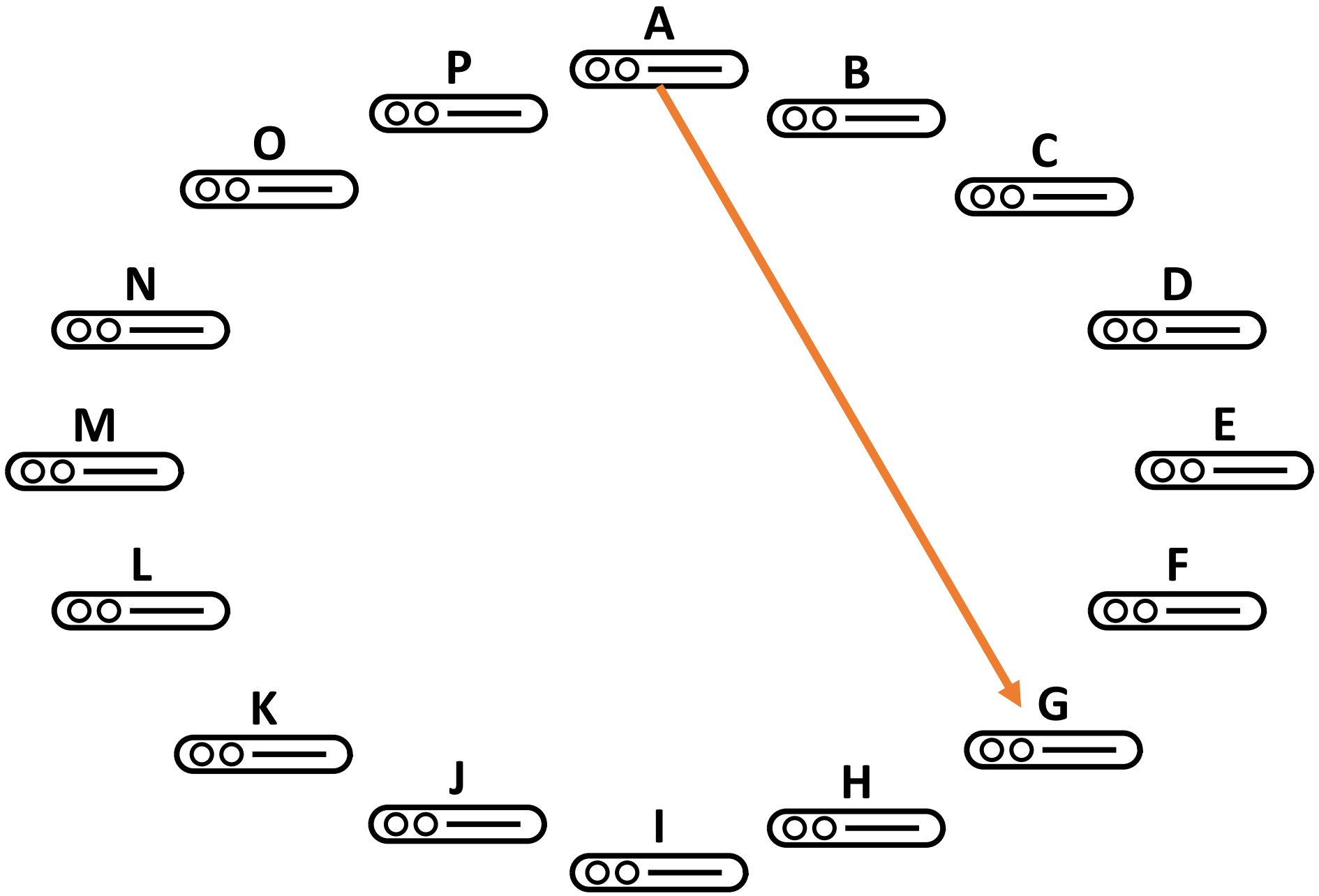

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P

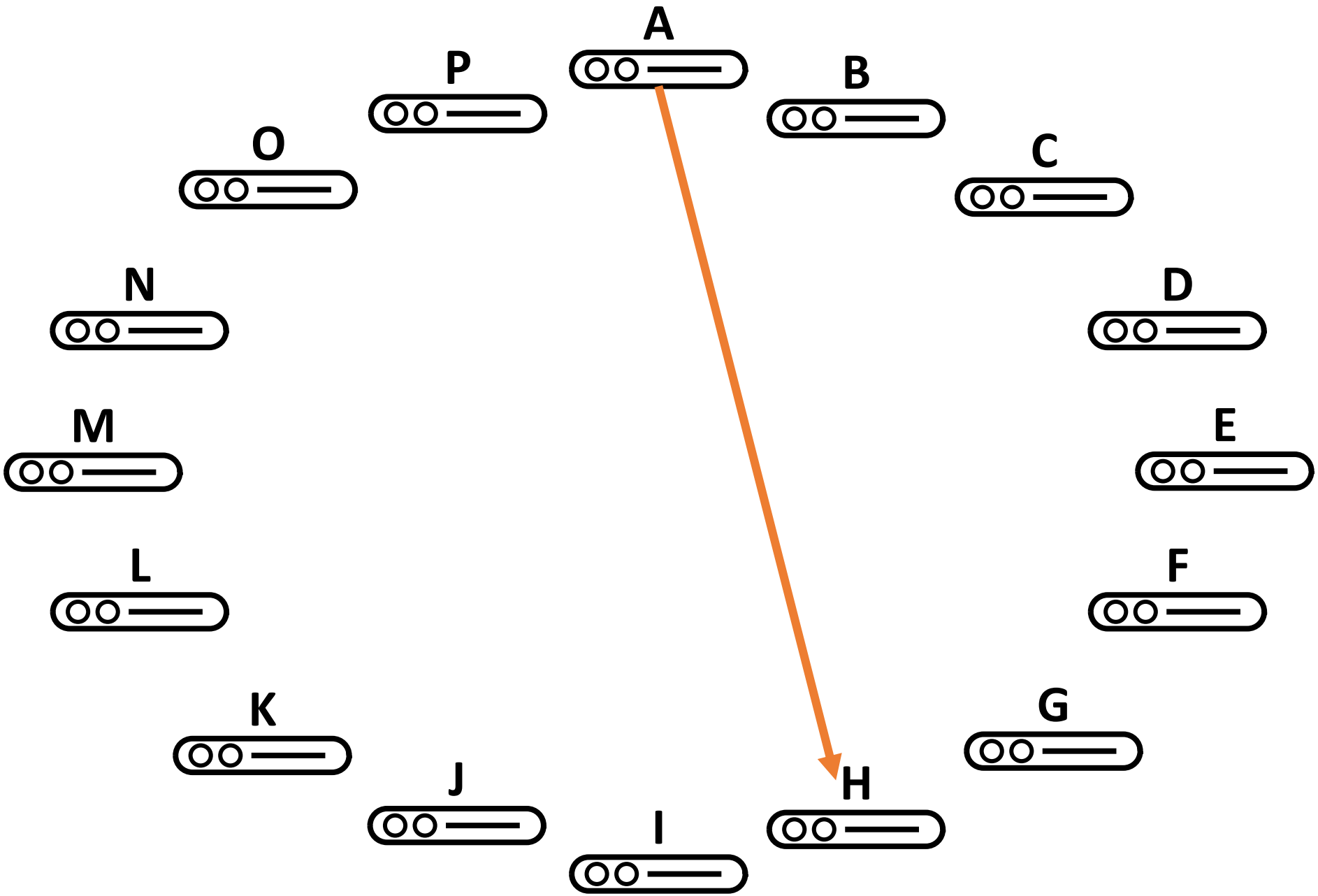

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P

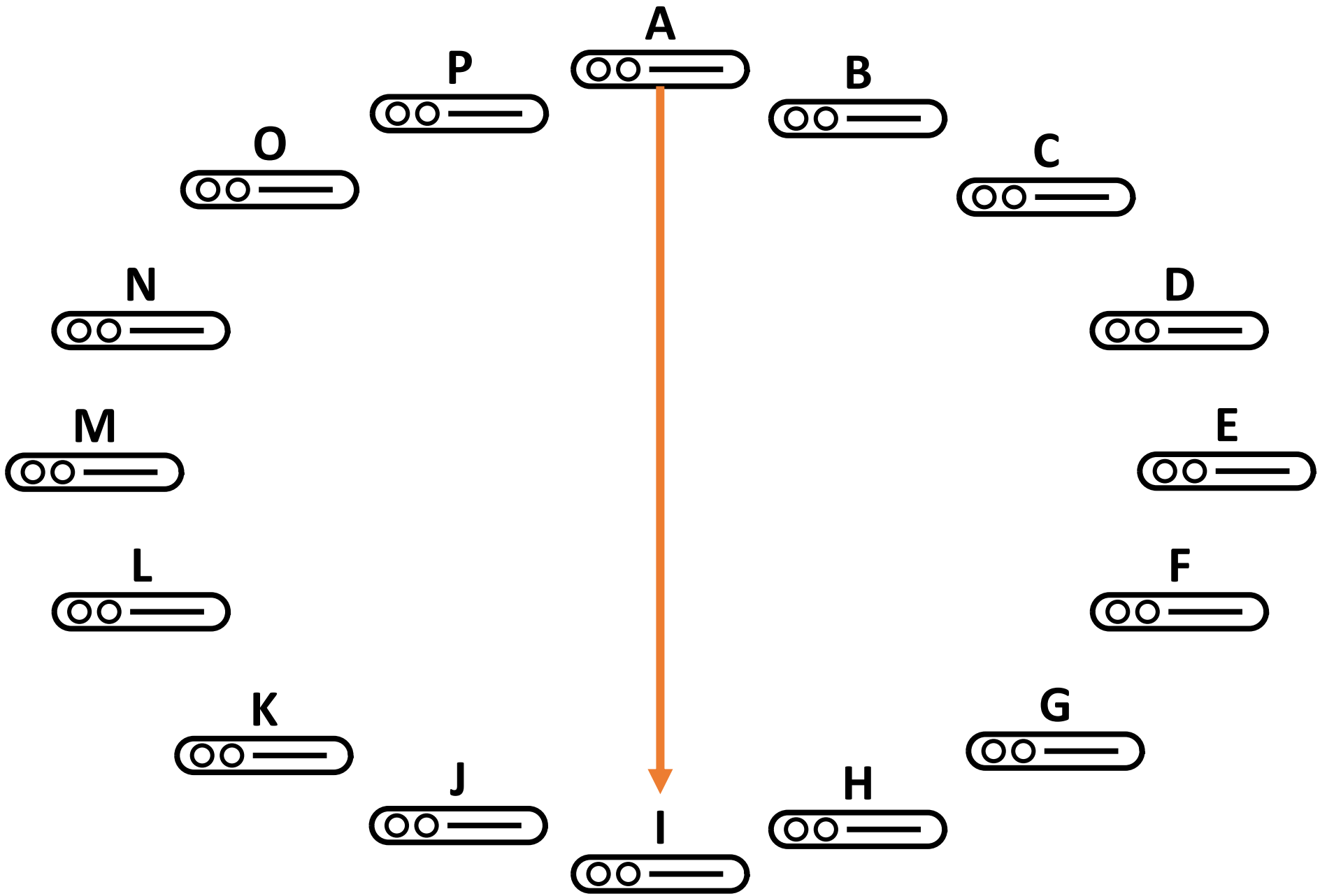

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P

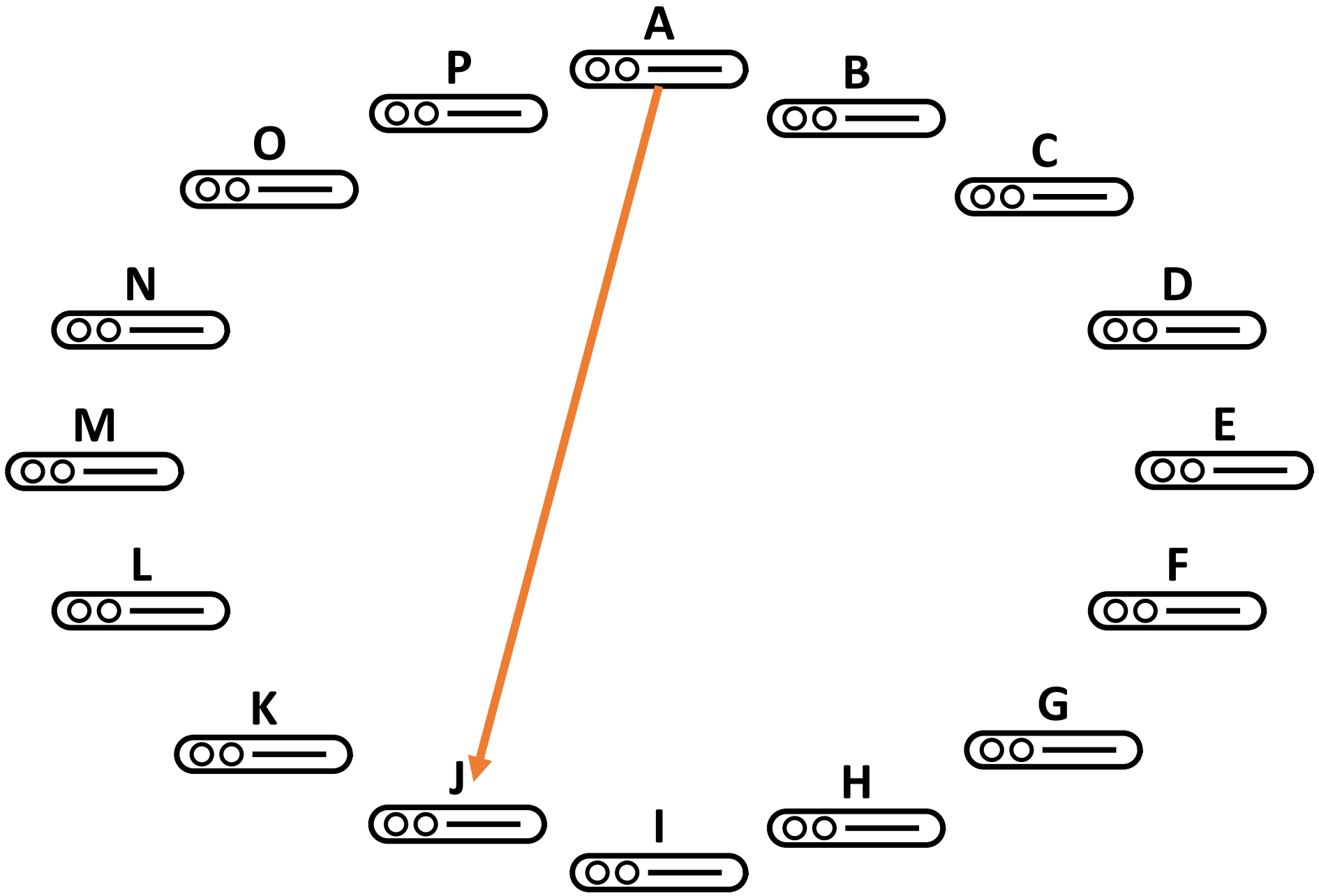

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P

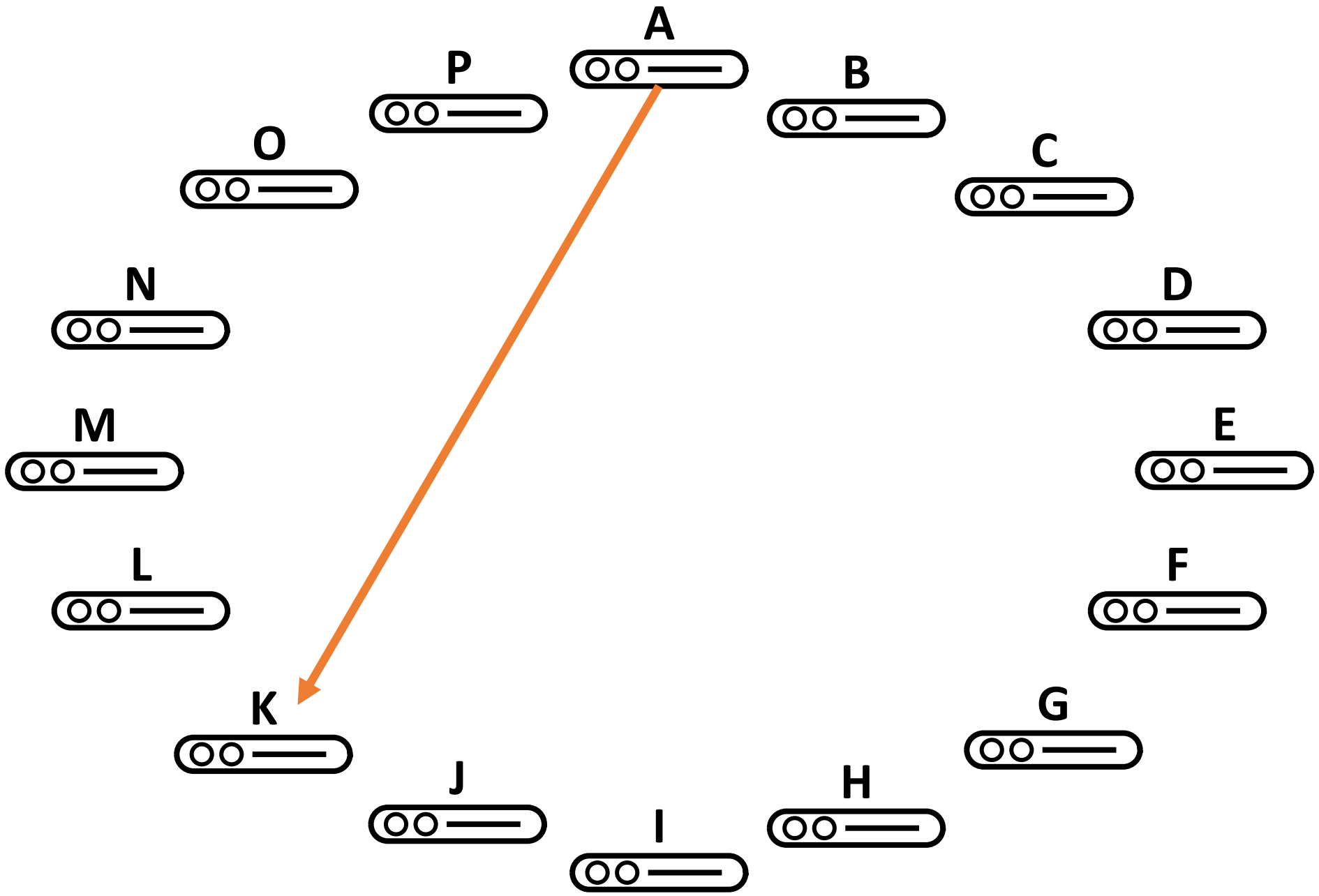

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P

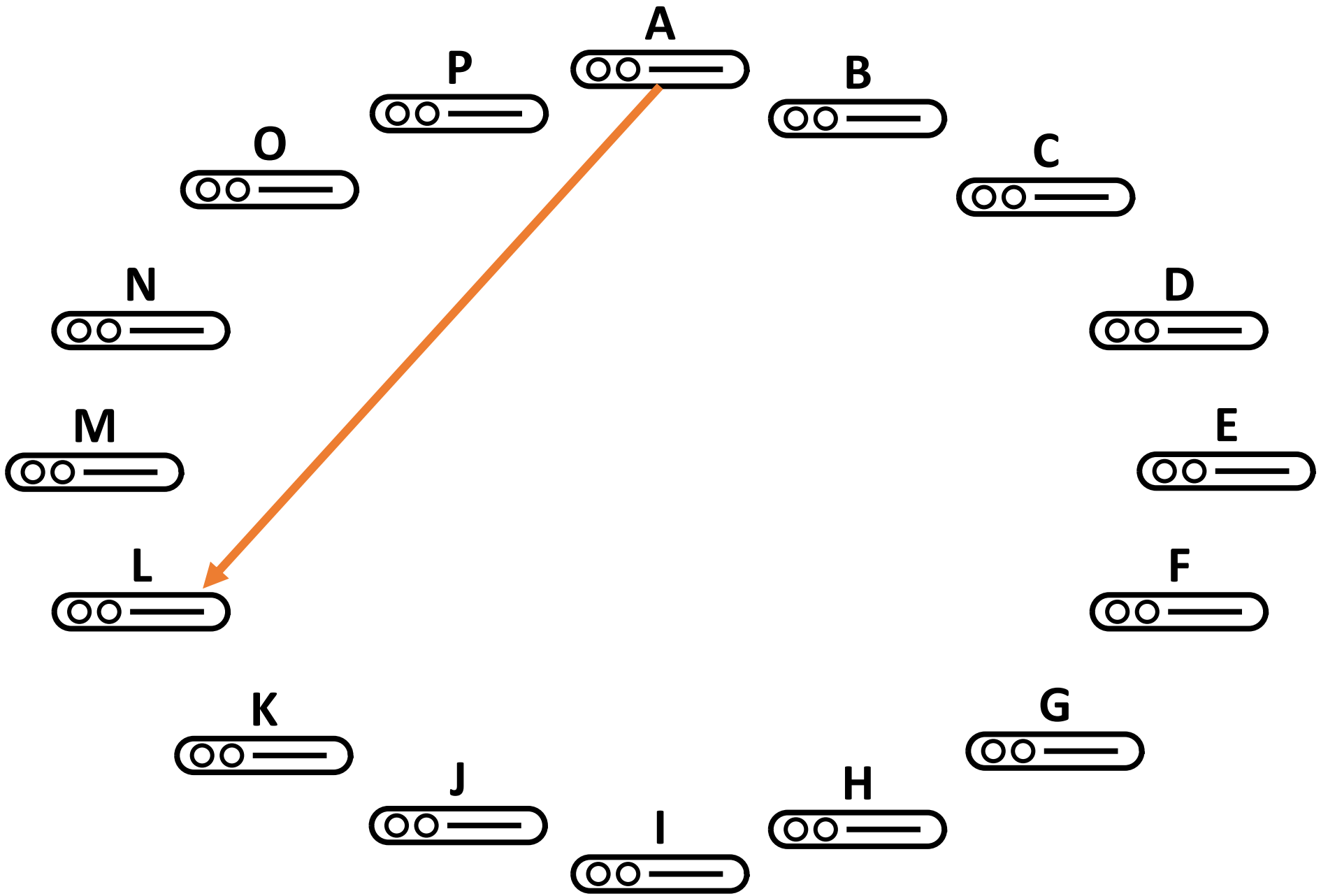

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P

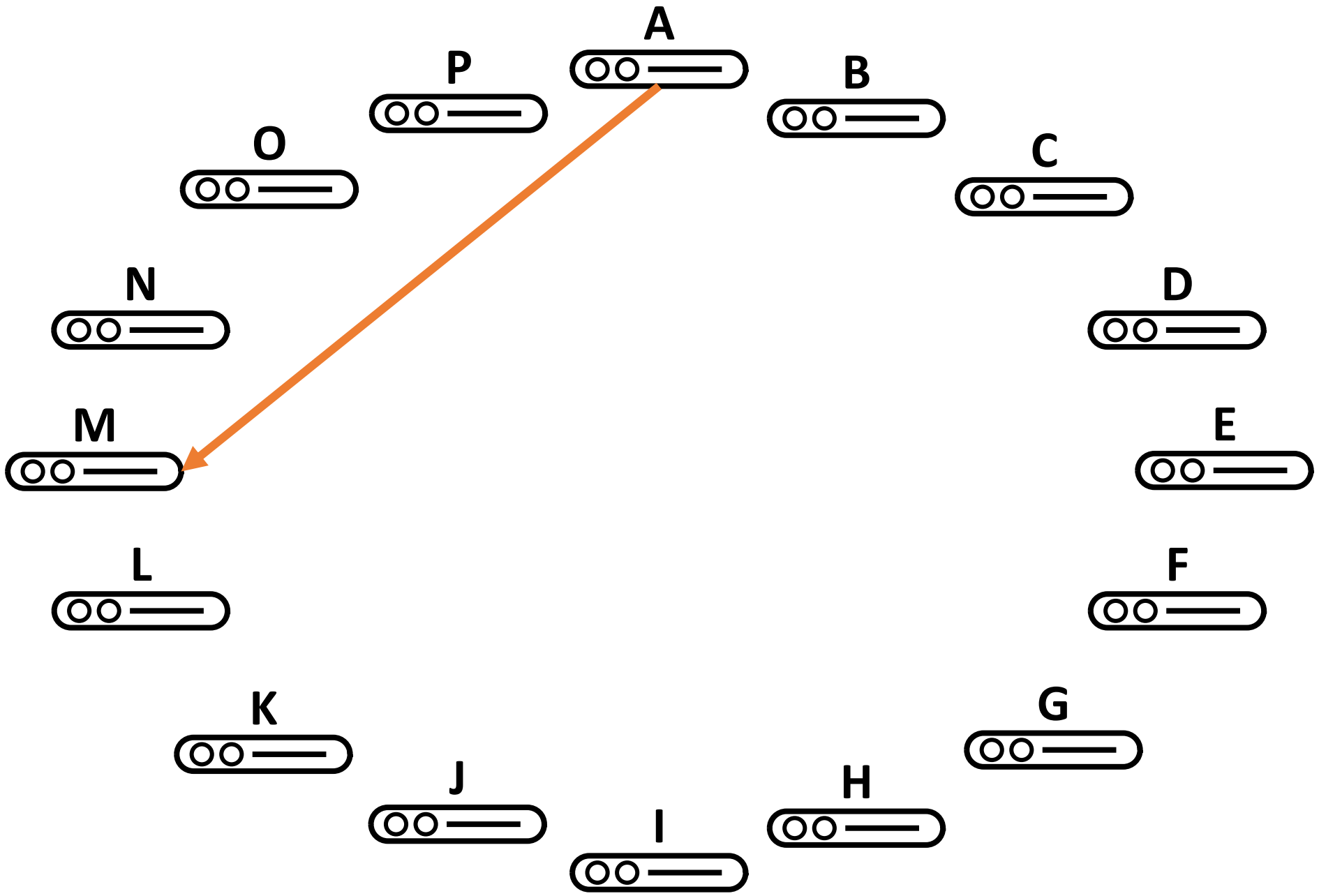

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P

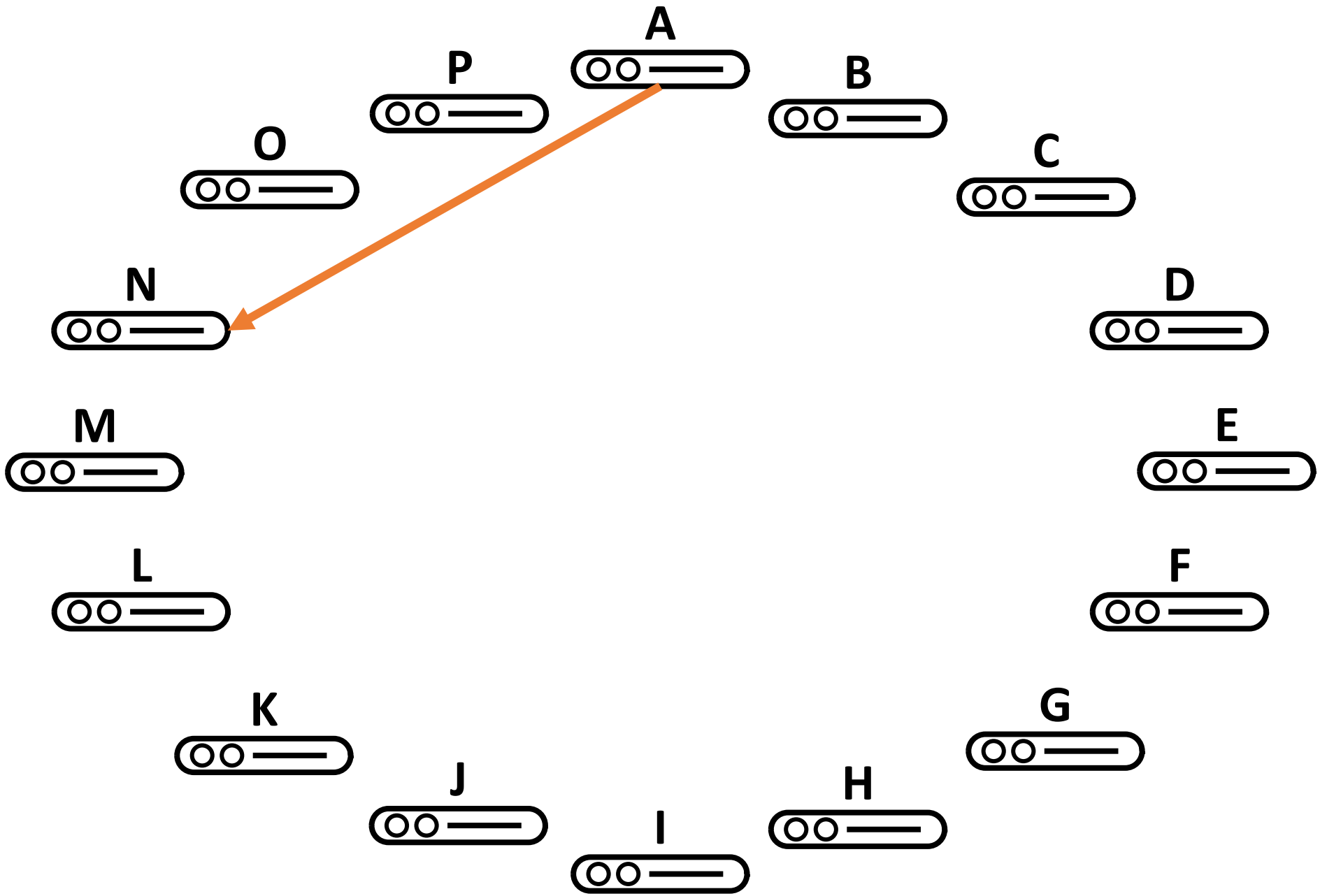

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P

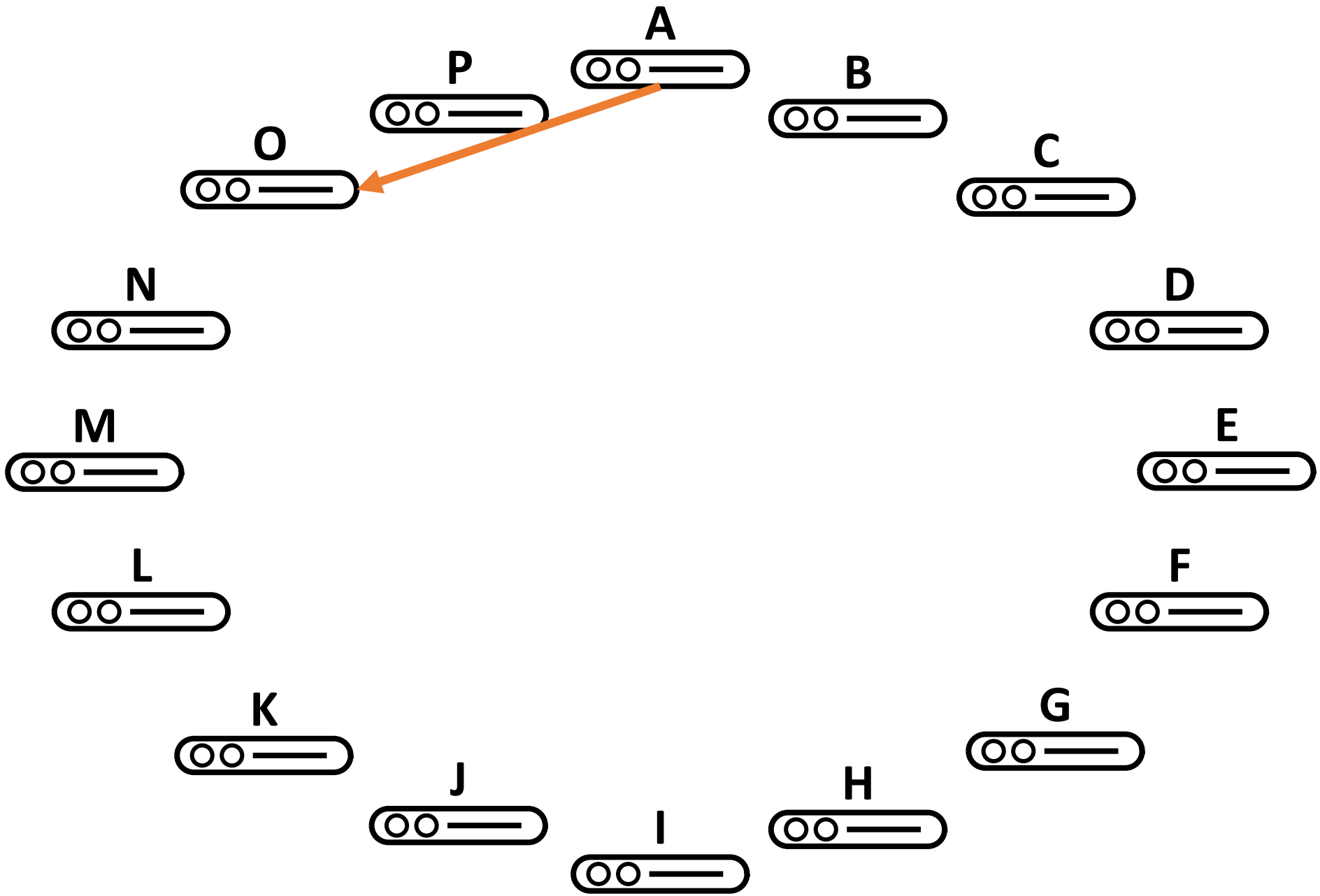

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P

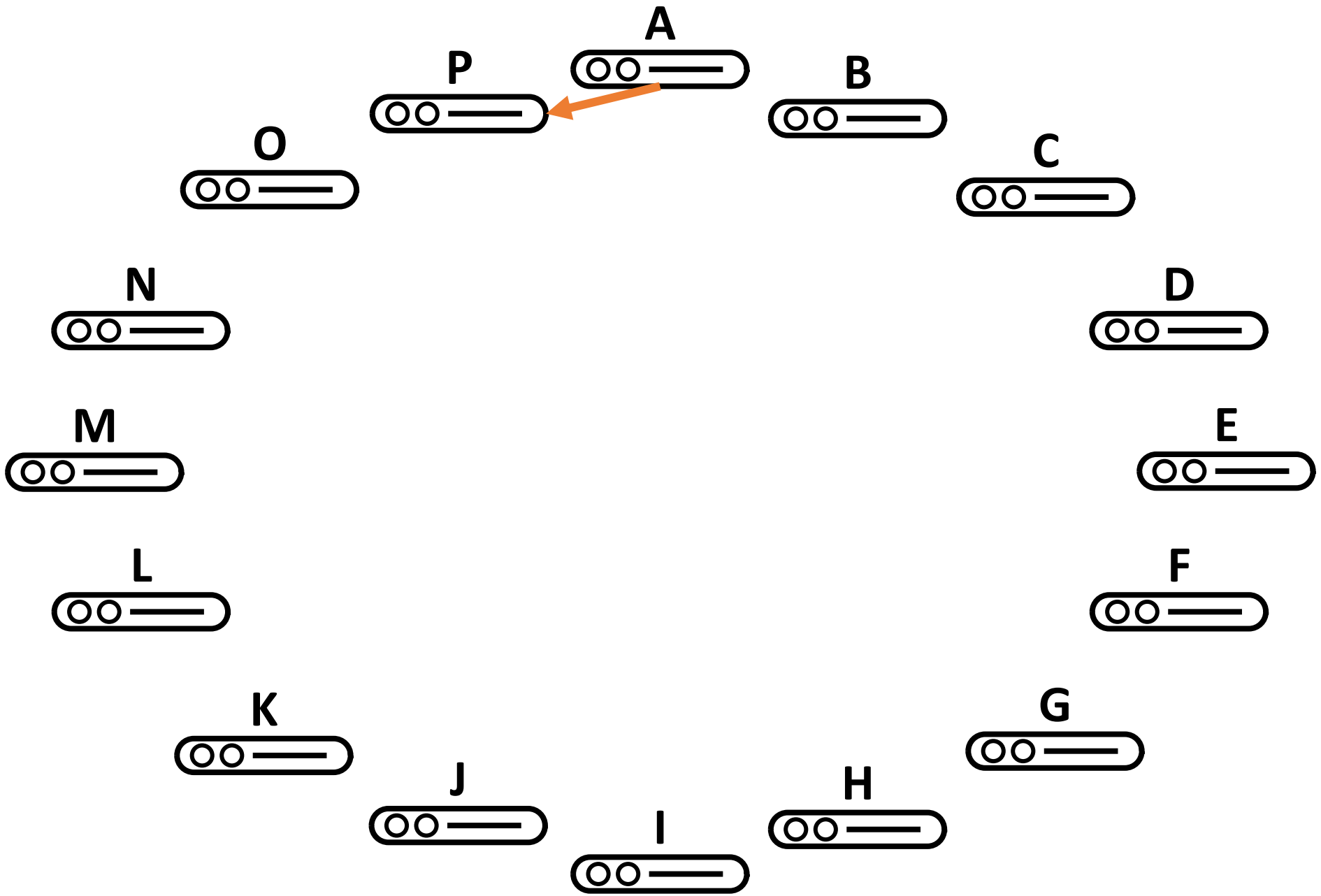

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P

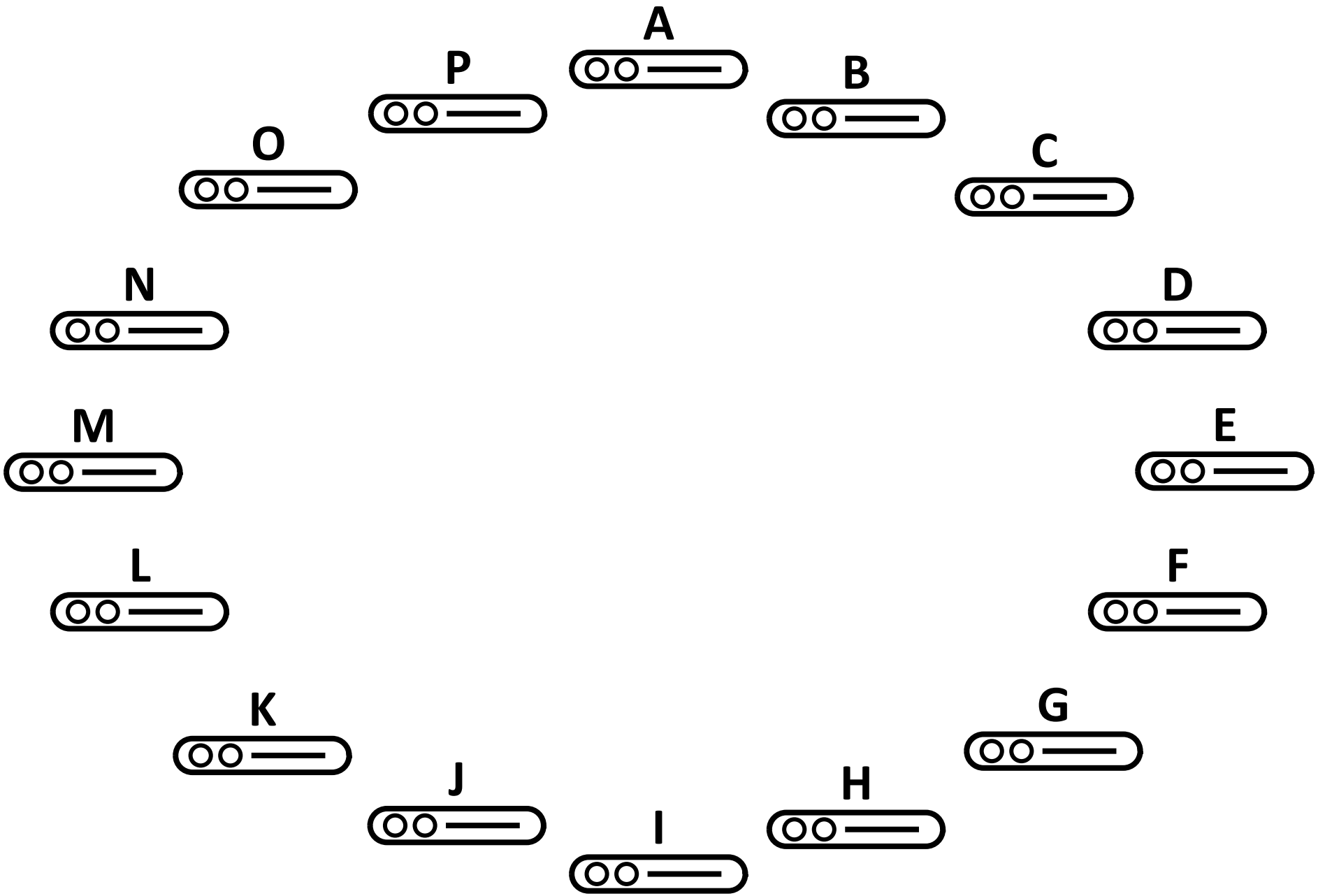

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P


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


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


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

N

P

A

C

M

O

B

D

L

J

G

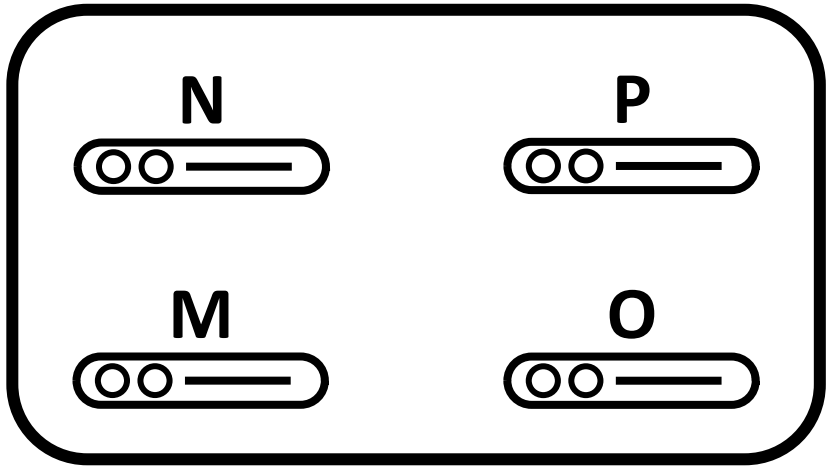
E

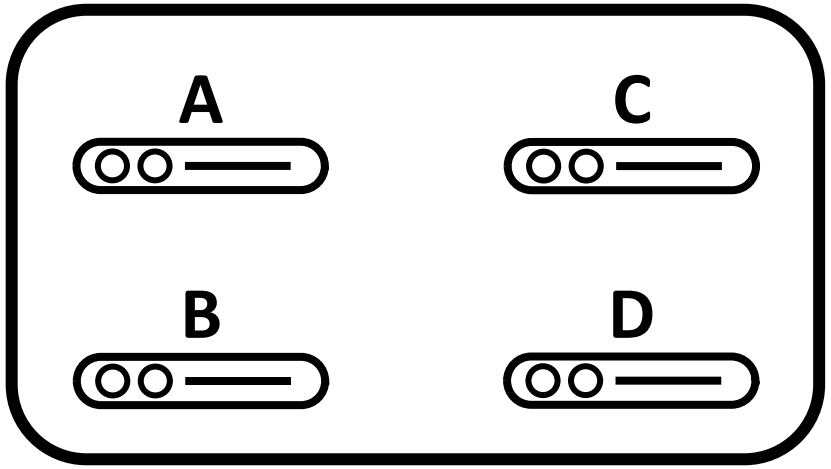
K

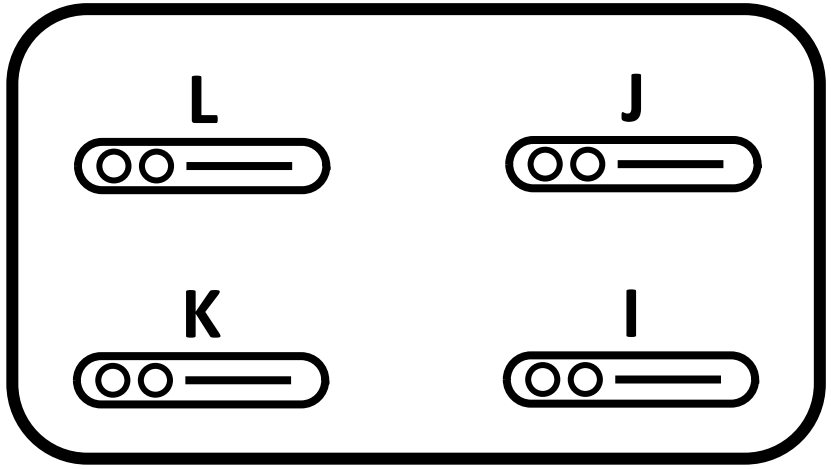
I

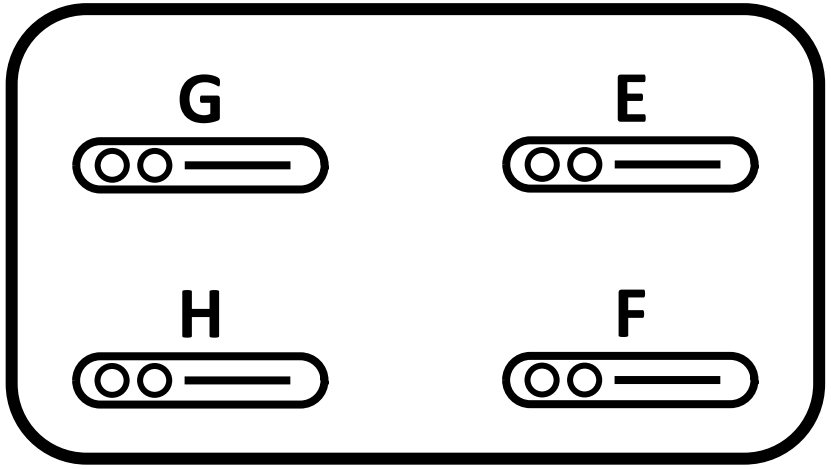
H

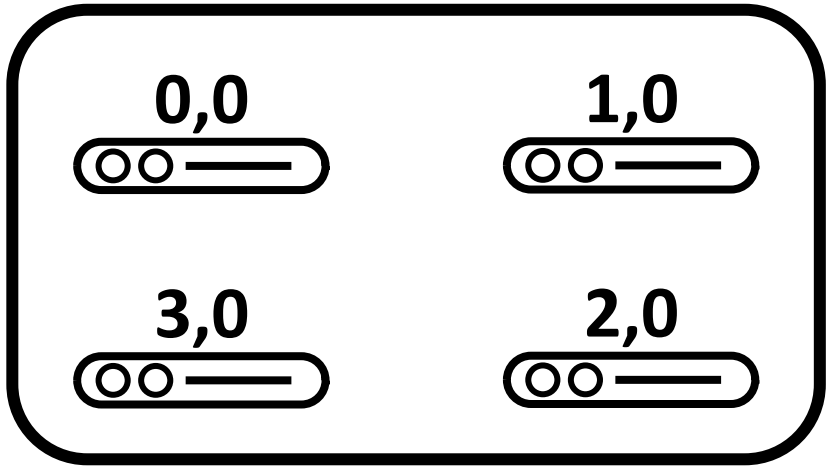
F

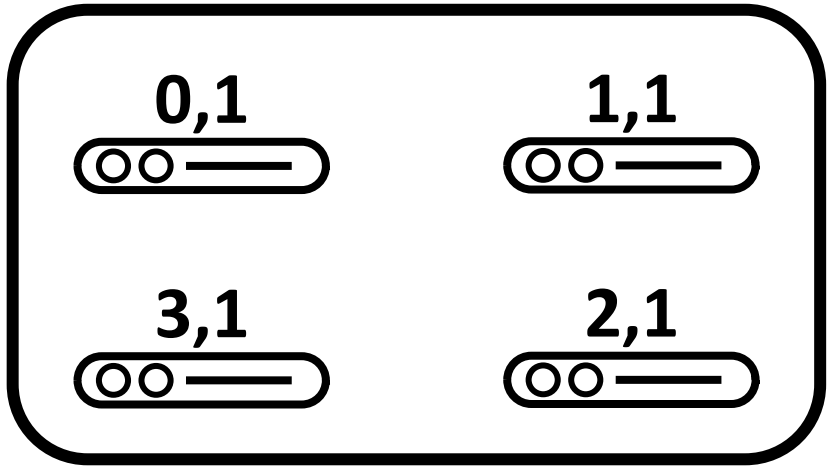

N
P
M
O

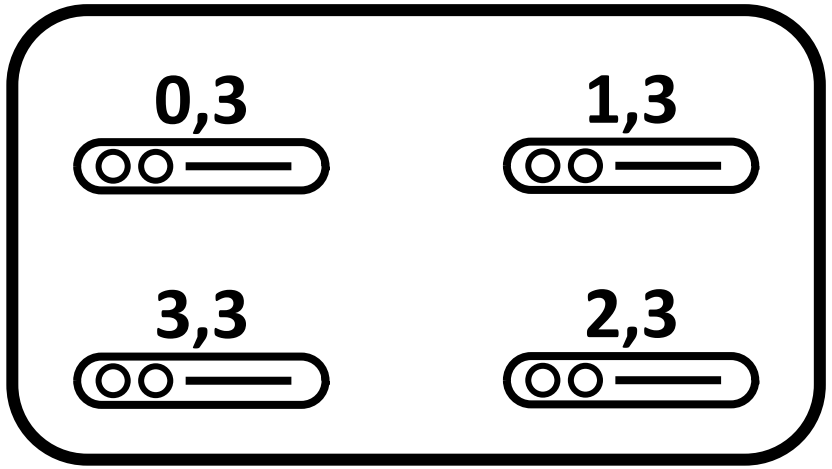

A
C
B
D

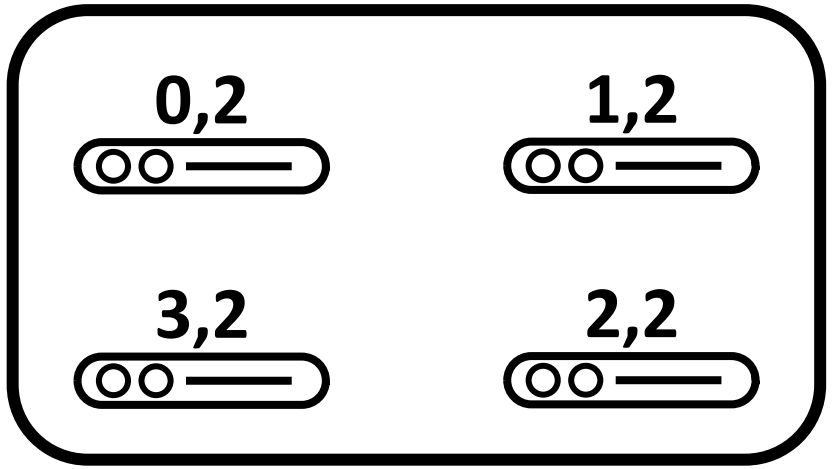

L
J
K
I

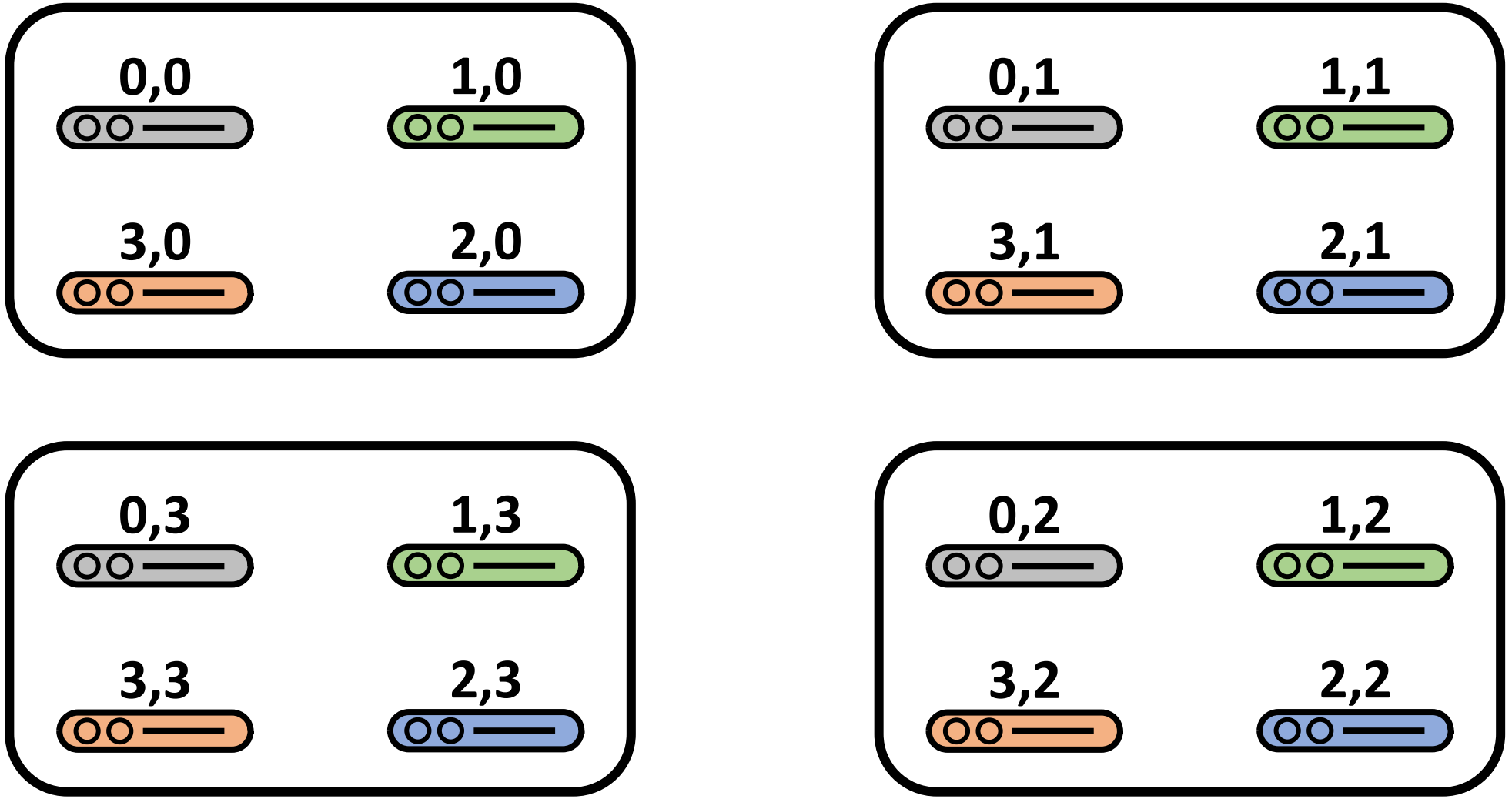

G
E
H
F


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

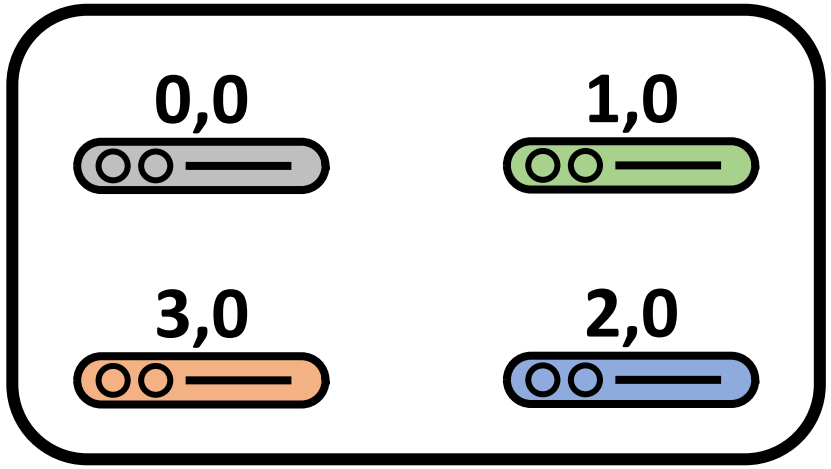

0,1
1,1
3,1
2,1

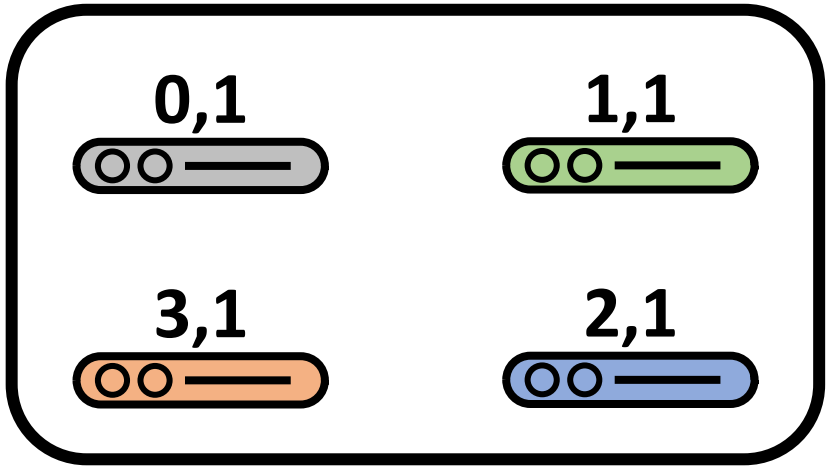

0,3
1,3
3,3
2,3

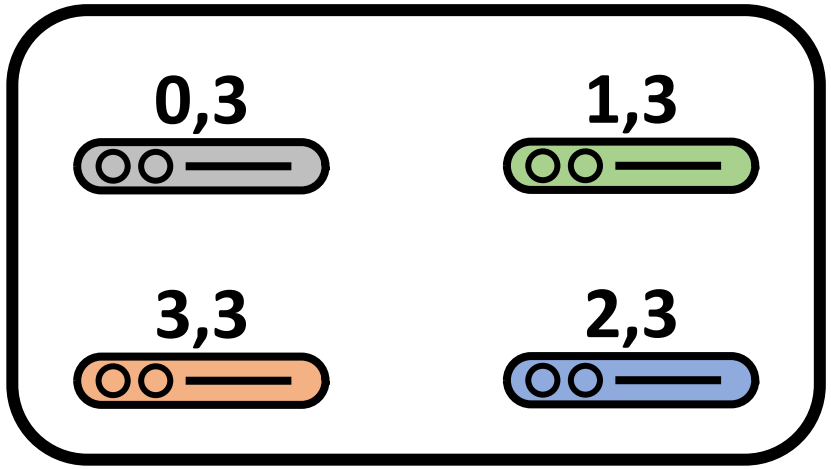

0,2
1,2
3,2
2,2

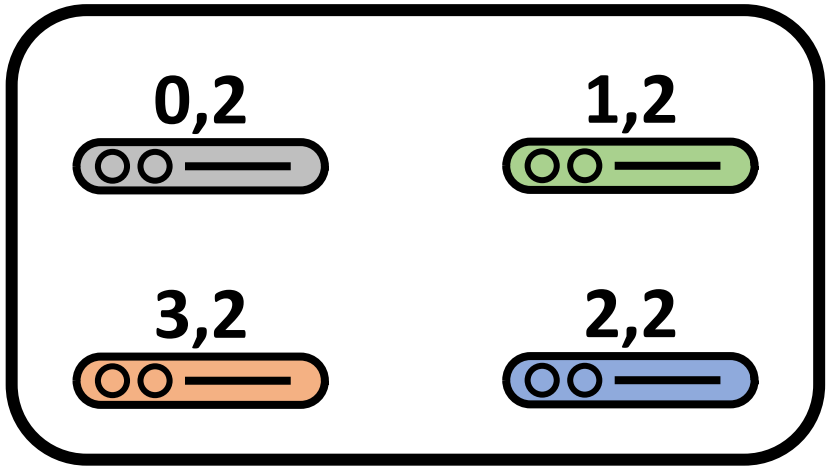

0,0
1,0
3,0
2,0
0,1
1,1
3,1
2,1
0,3
1,3
3,3
2,3
0,2
1,2
3,2
2,2

# Shale Schedule ($h$=2)

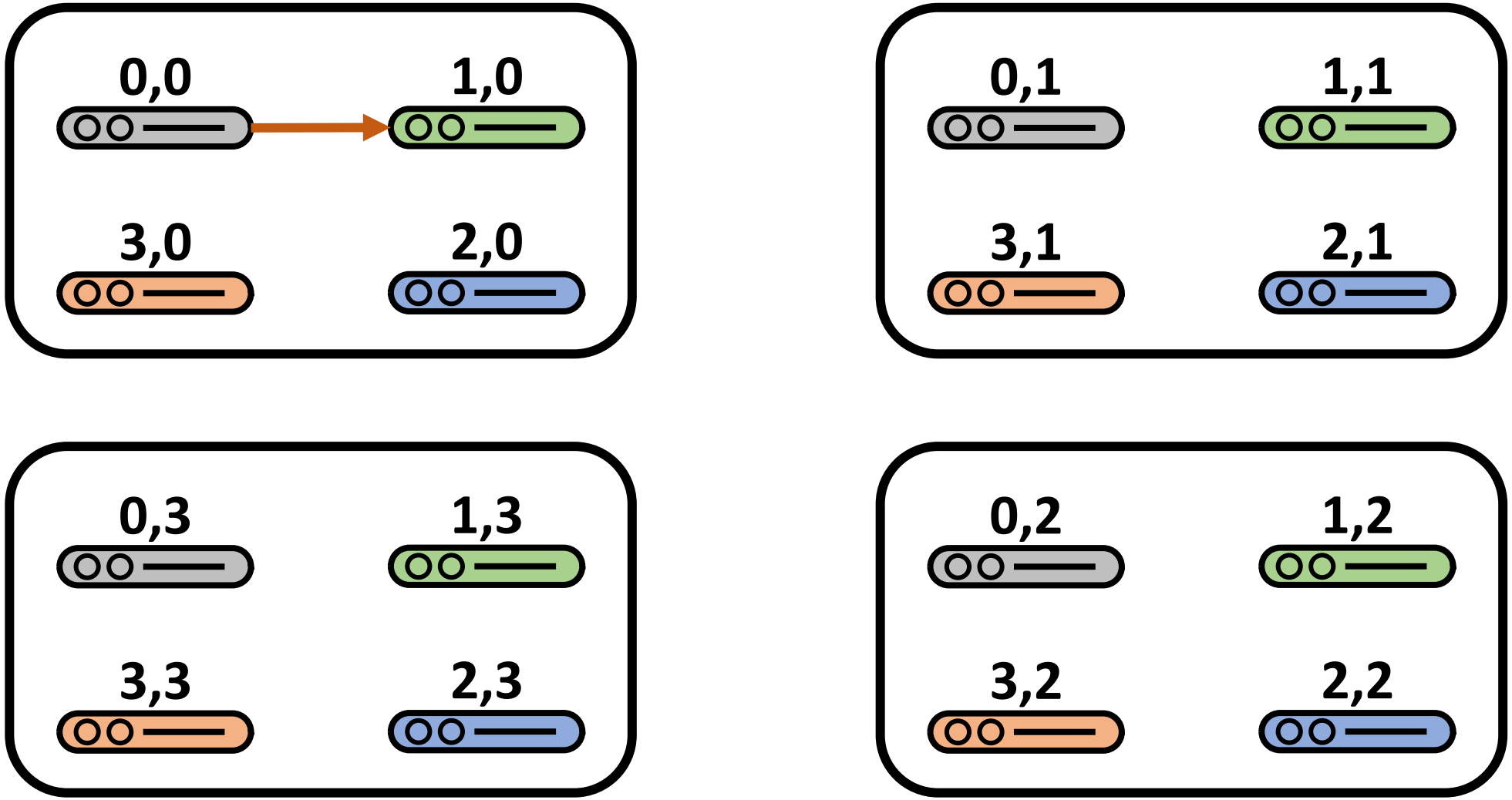


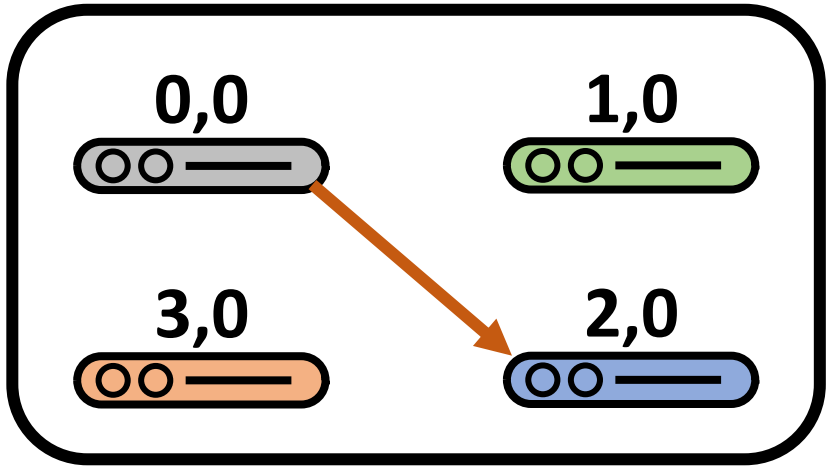



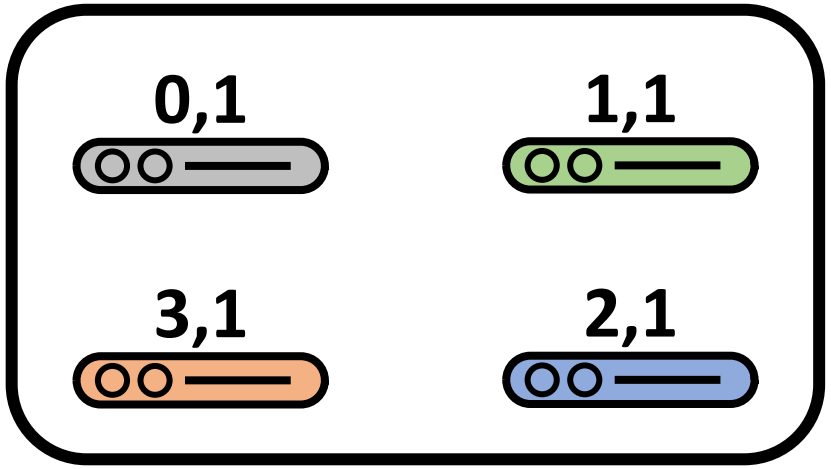



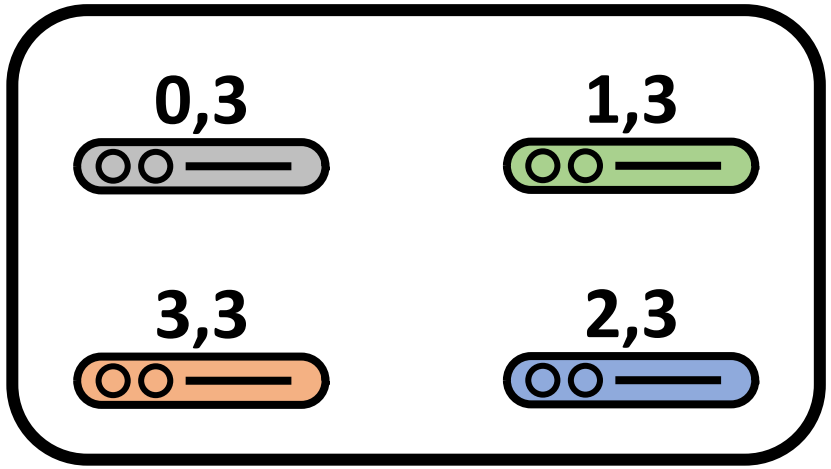
# Shale Schedule ($h$=2)

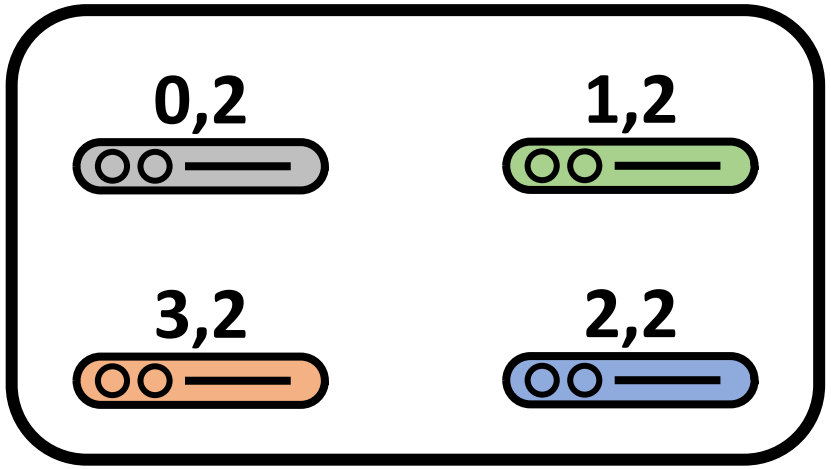


# Shale Schedule ($h$=2)

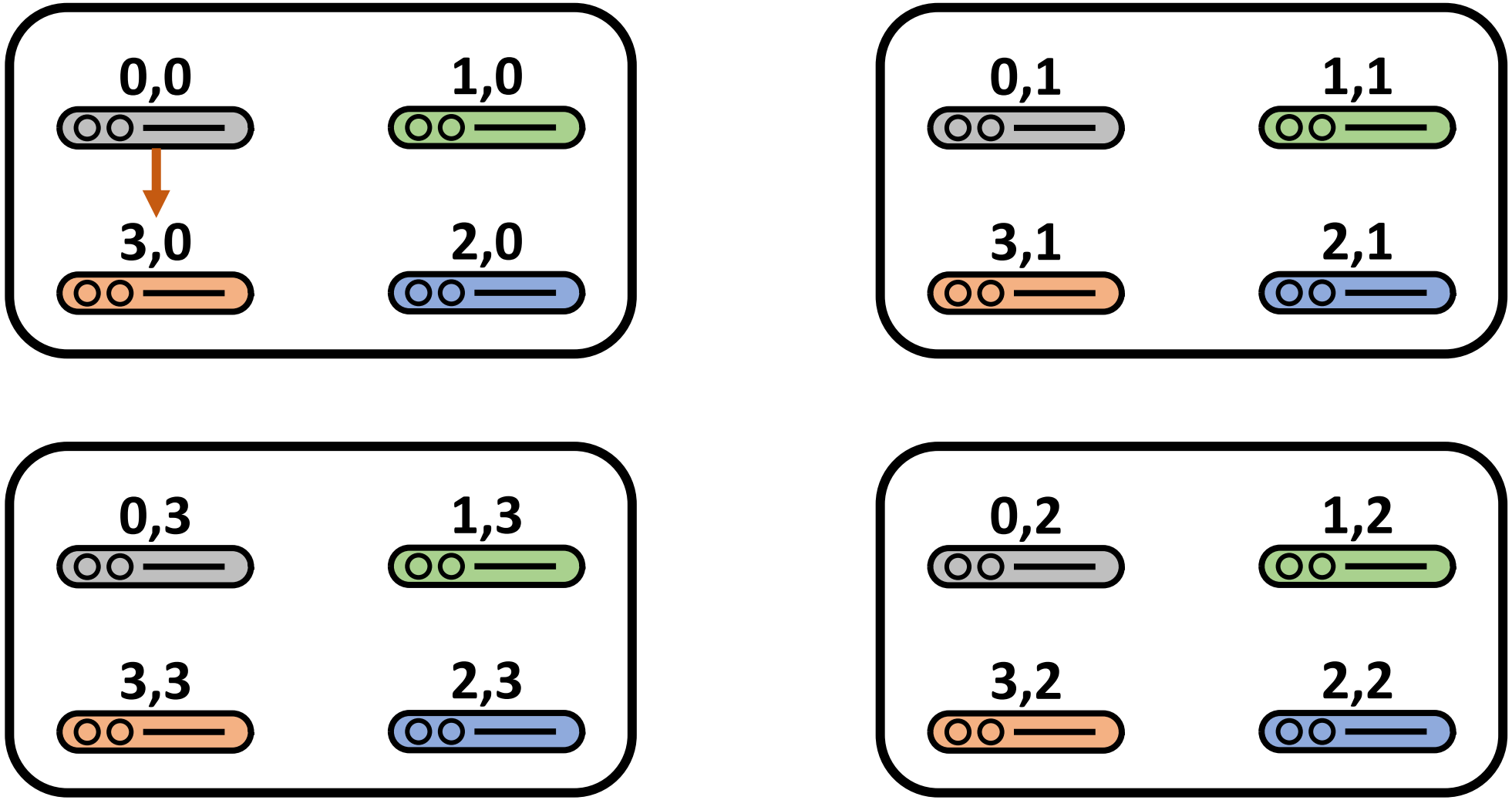


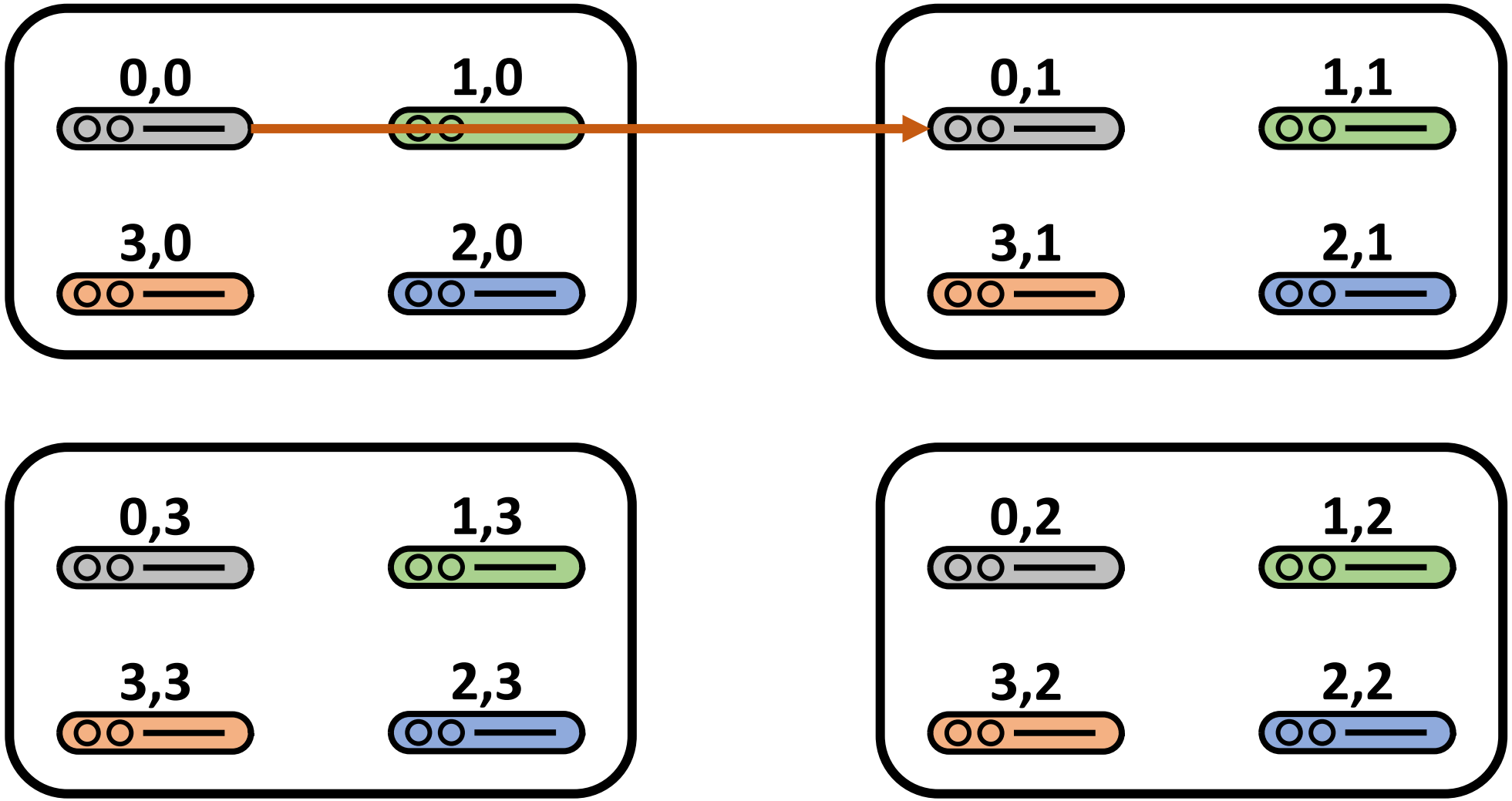



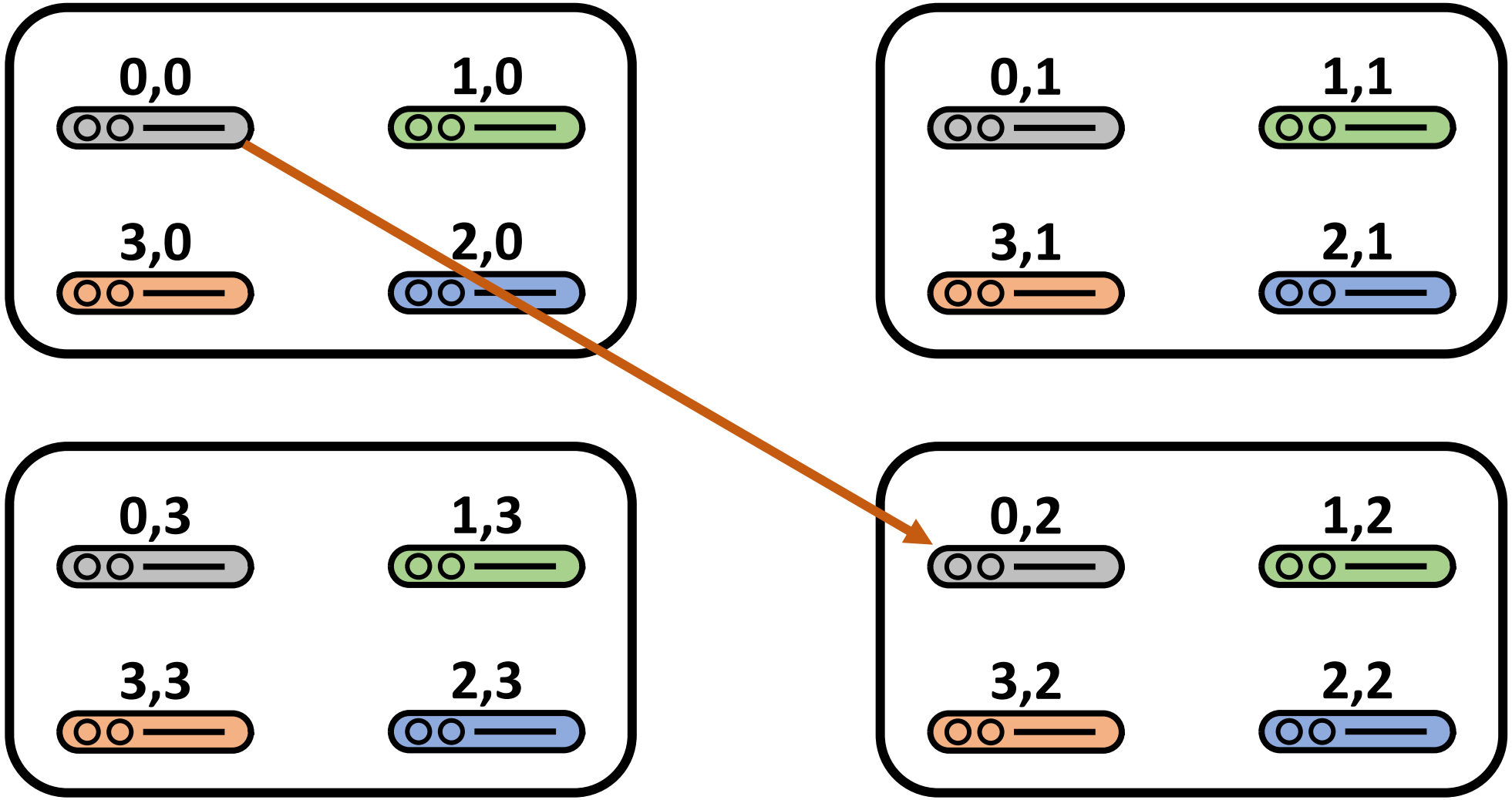
# Shale Schedule ($h$=2)



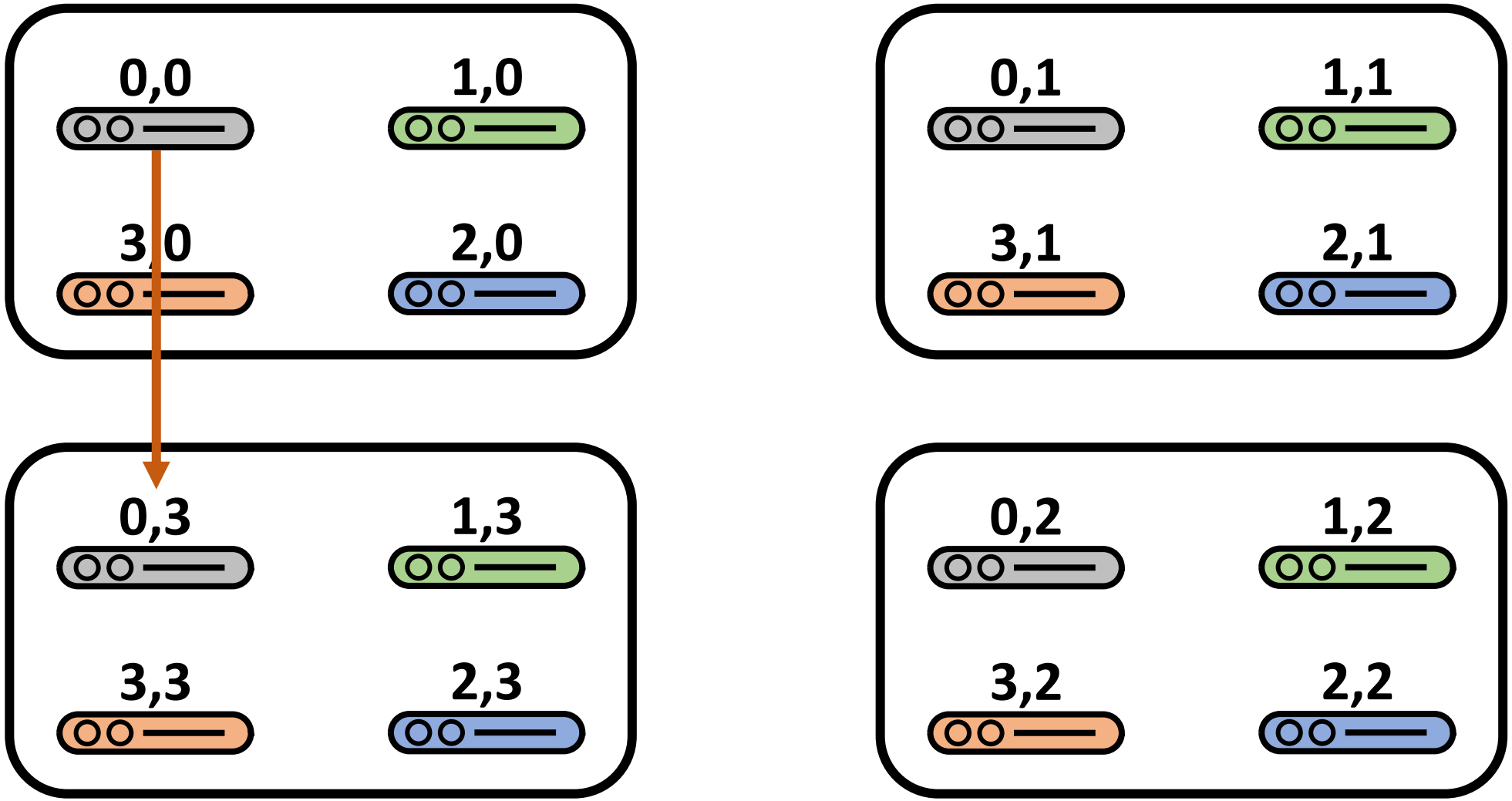
# Shale Schedule ($h$=2)



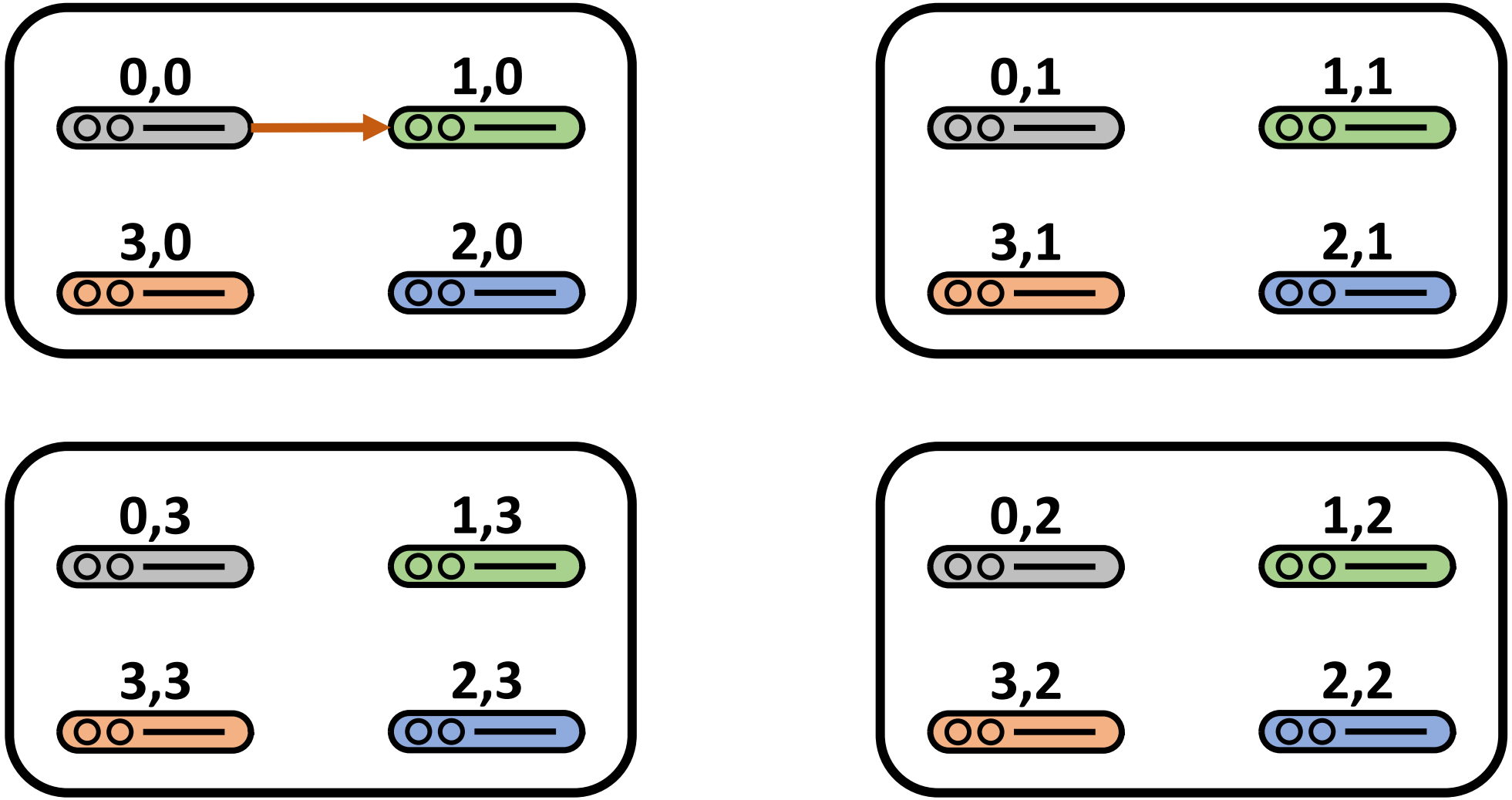
# Shale Schedule ($h$=2)



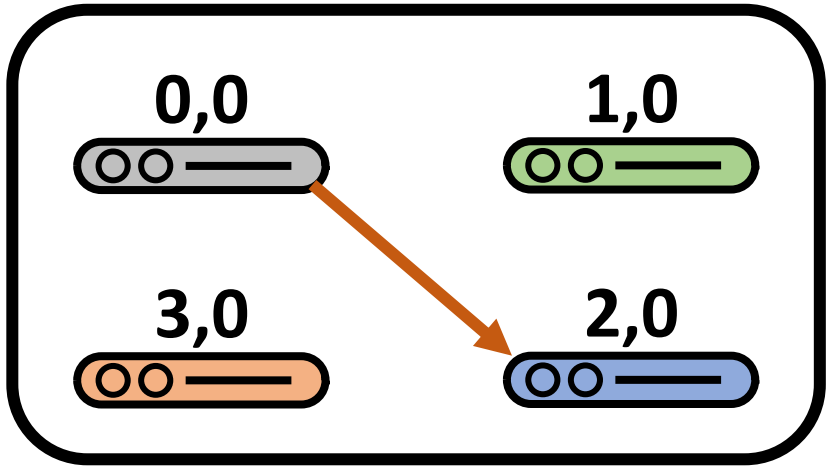
# Shale Schedule ($h$=2)

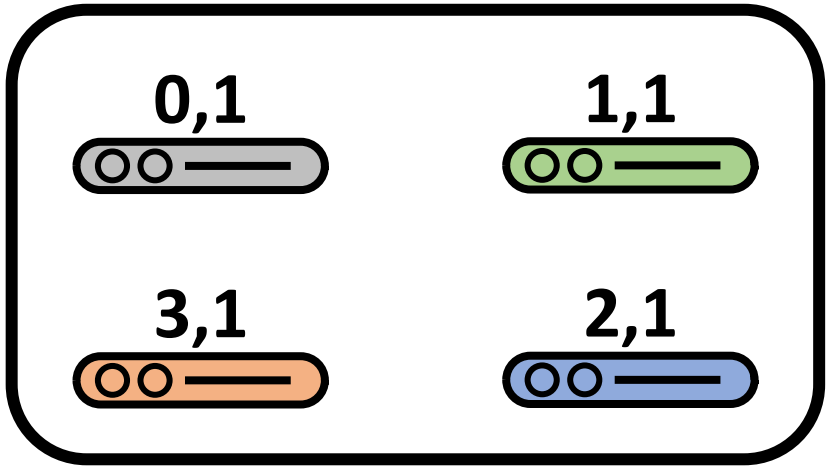


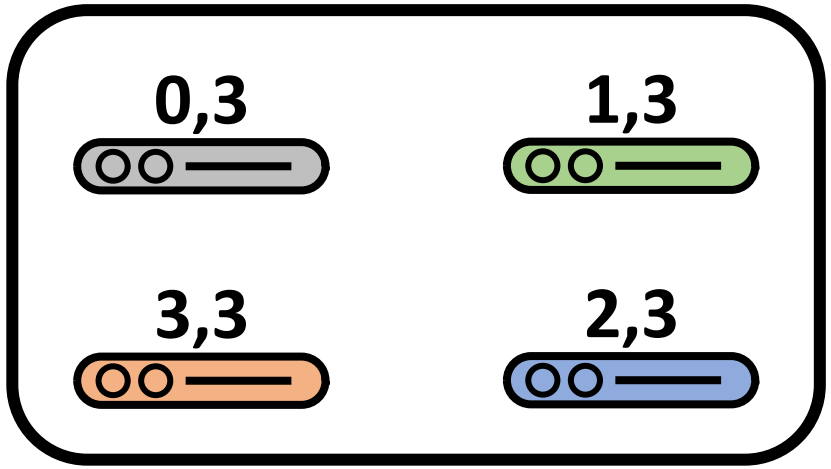
# Shale Schedule ($h$=2)

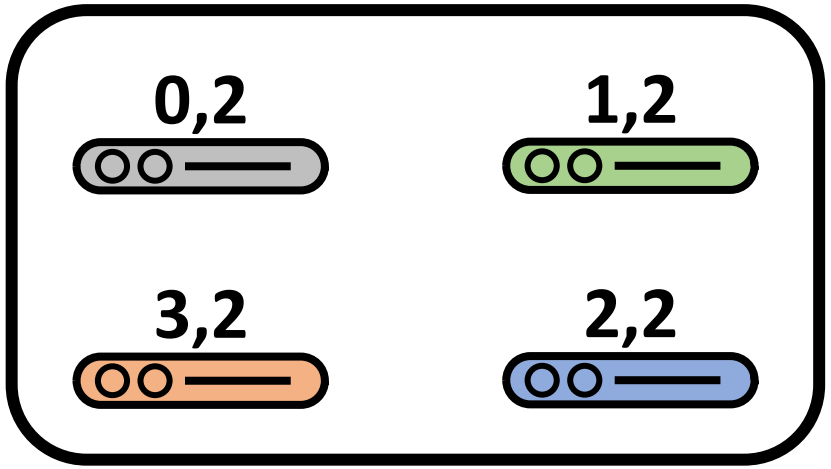


# Shale Schedule ($h$=2)

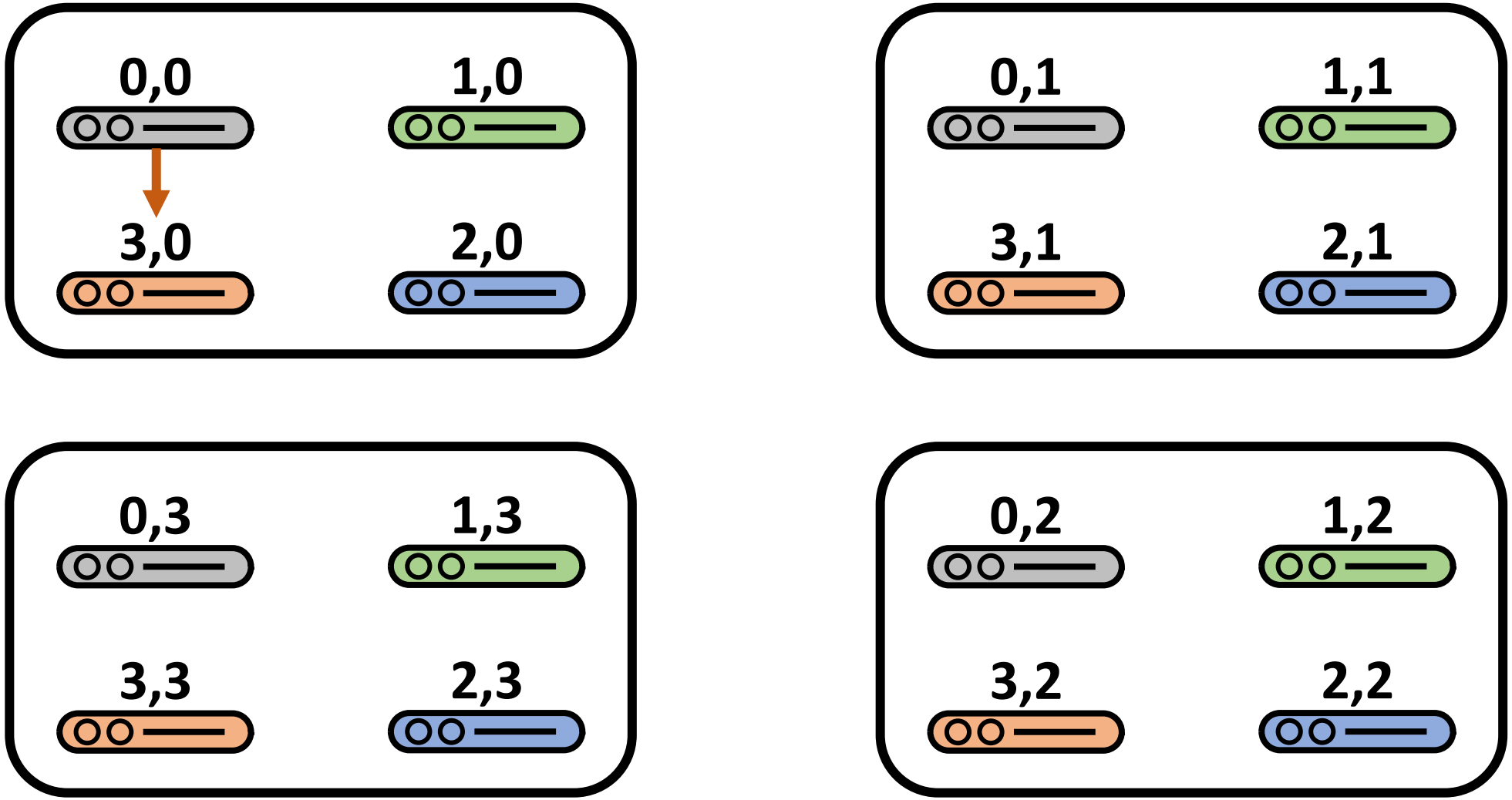


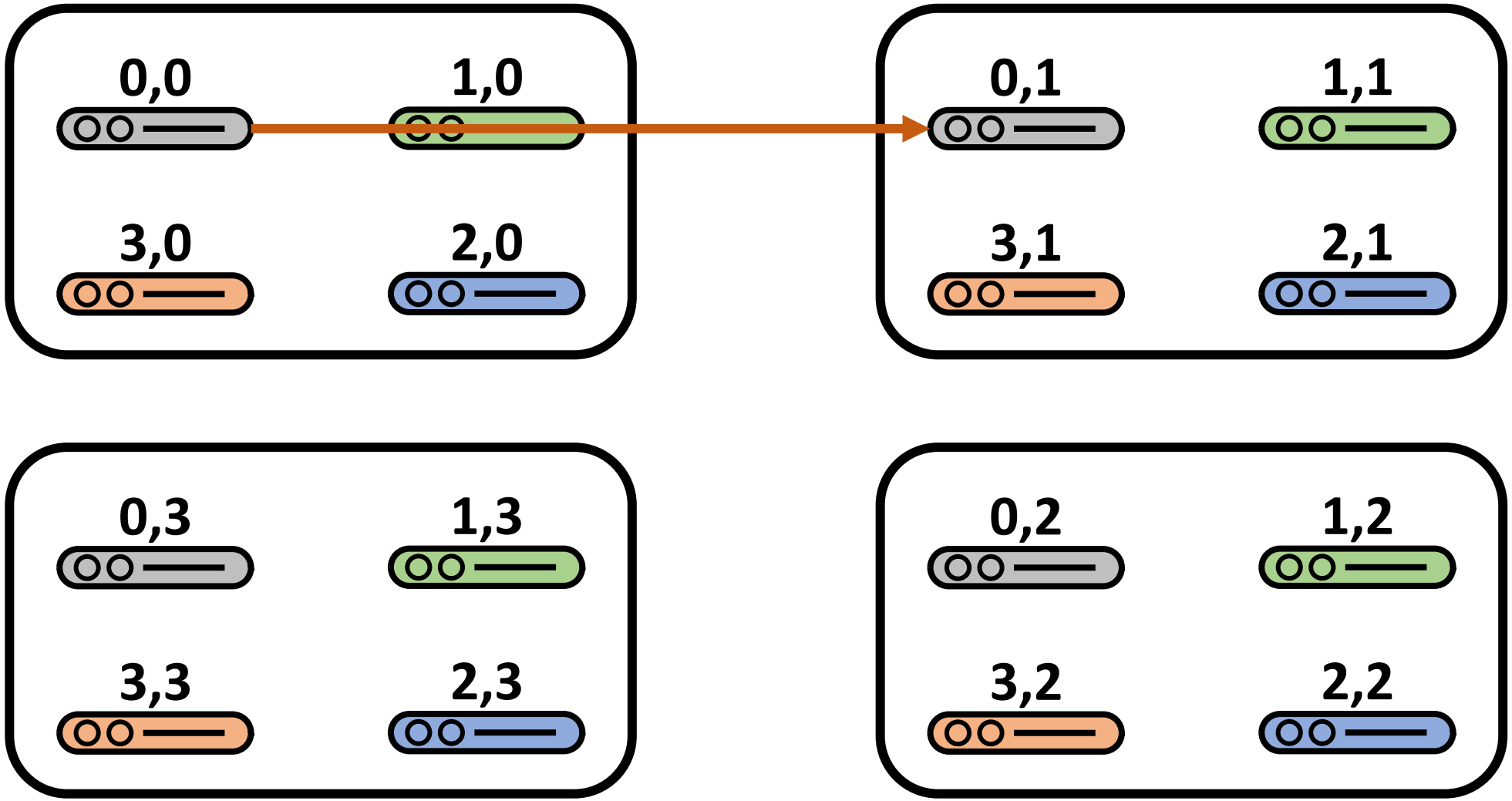



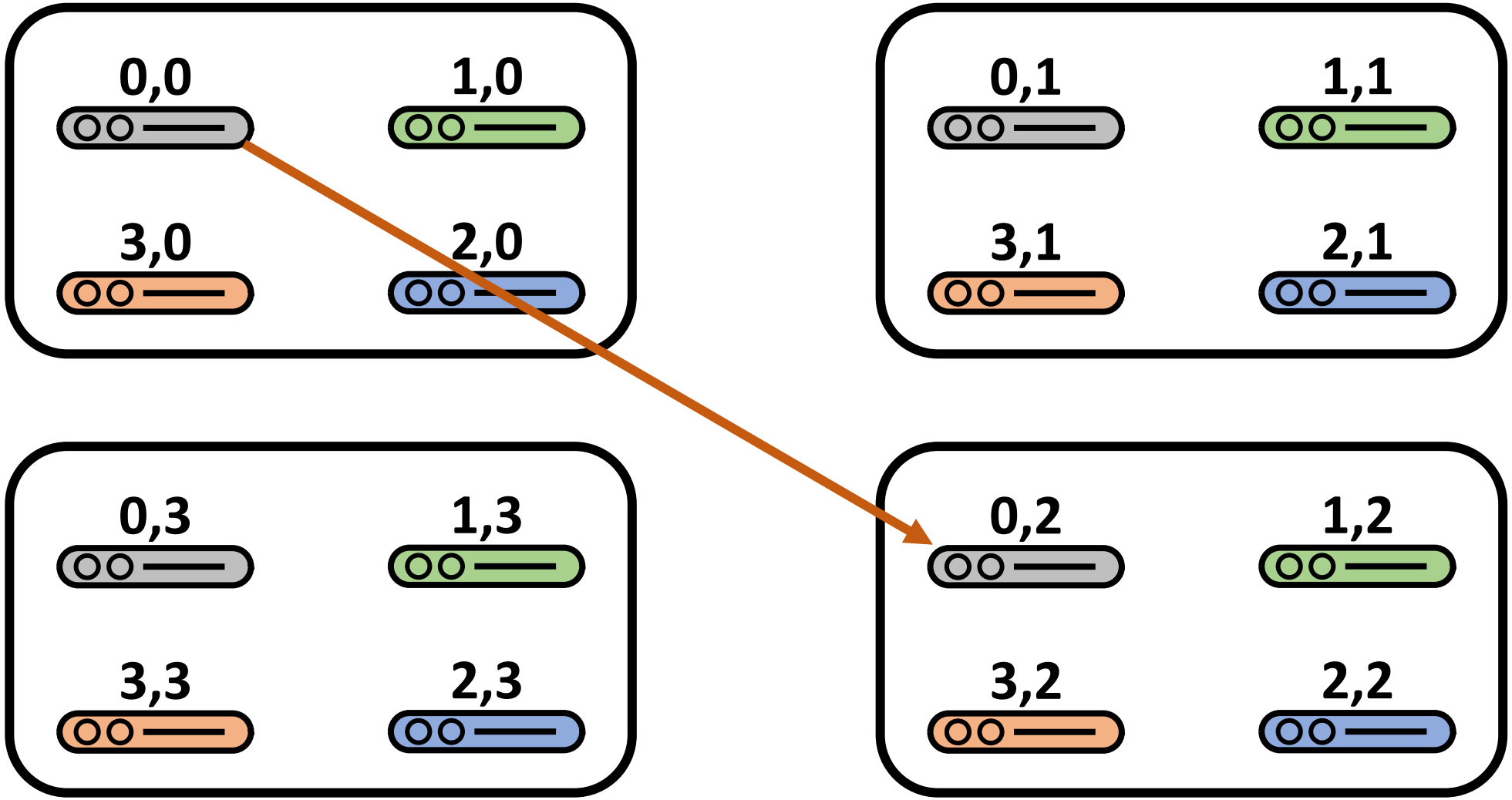
# Shale Schedule ($h$=2)



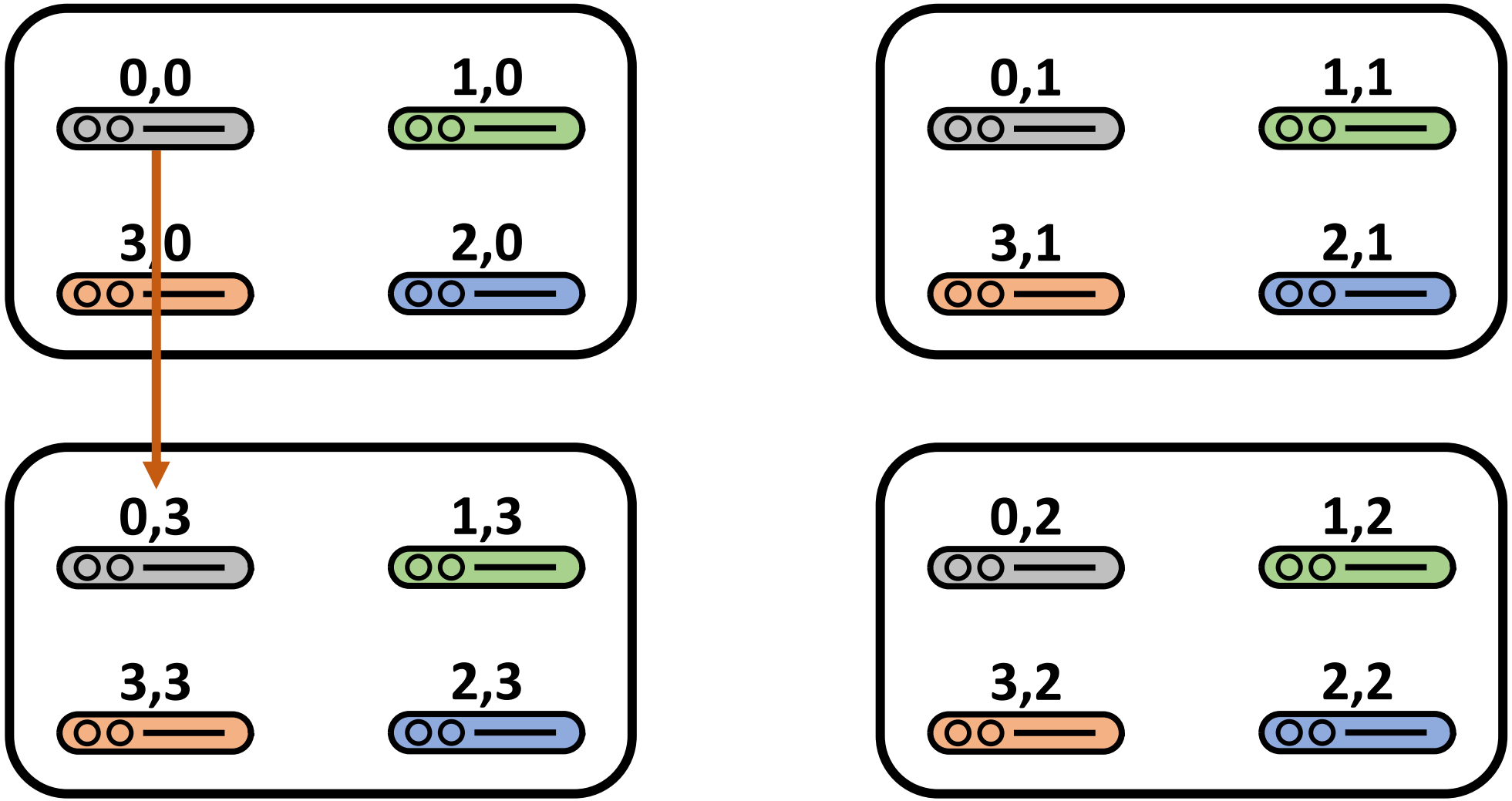
# Shale Schedule ($h$=2)



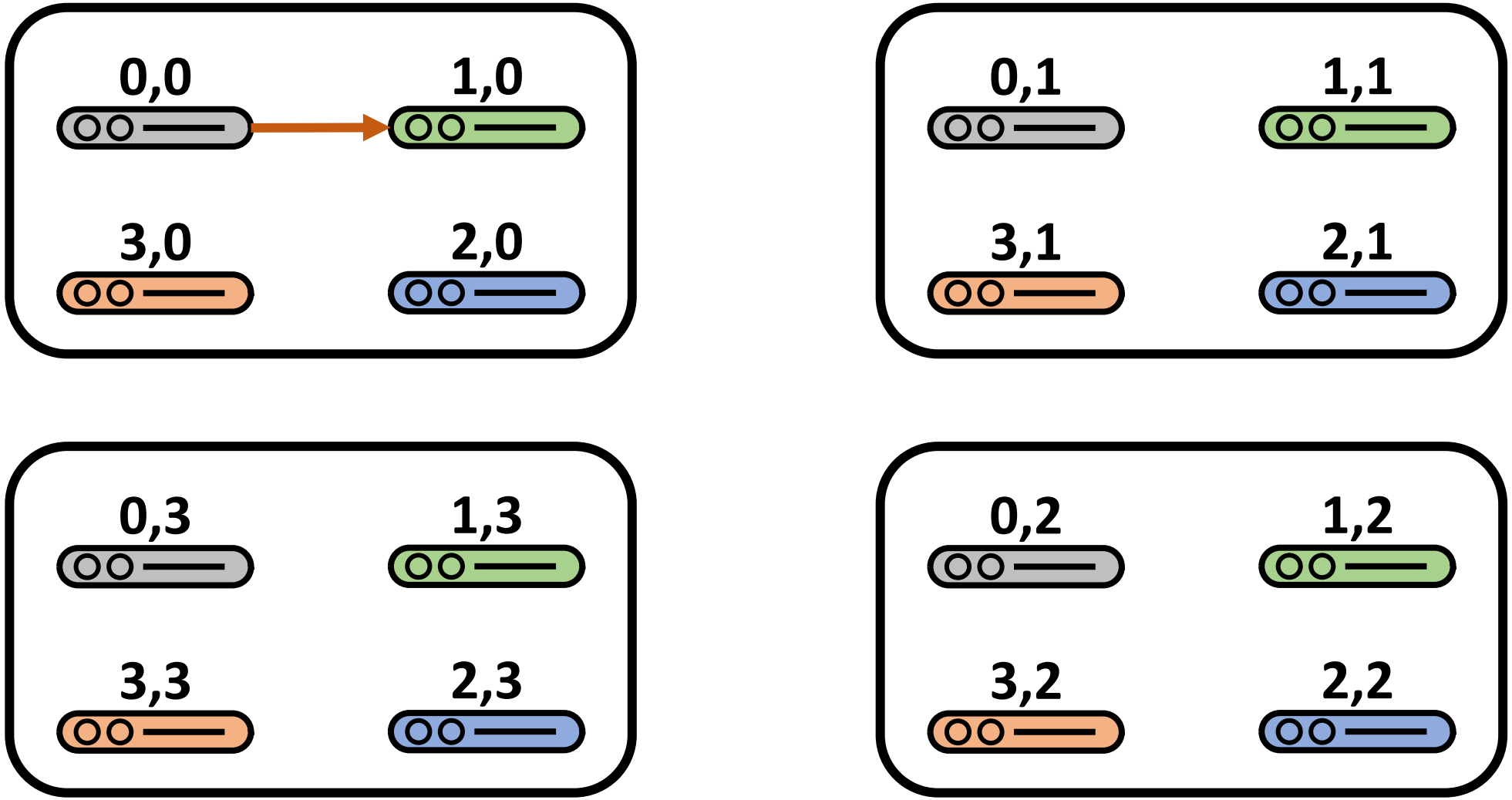
# Shale Schedule ($h$=2)



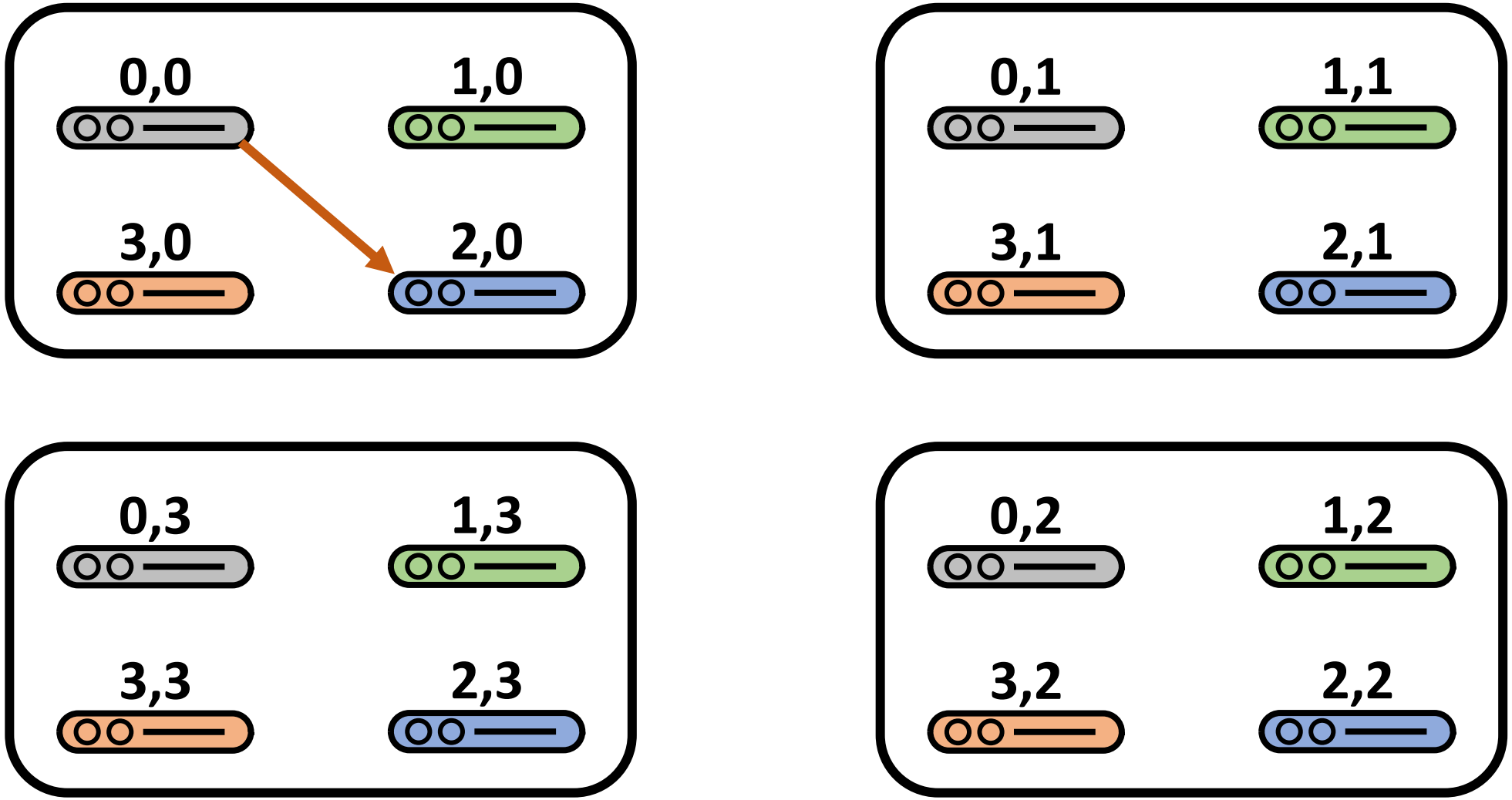
# Shale Schedule ($h$=2)



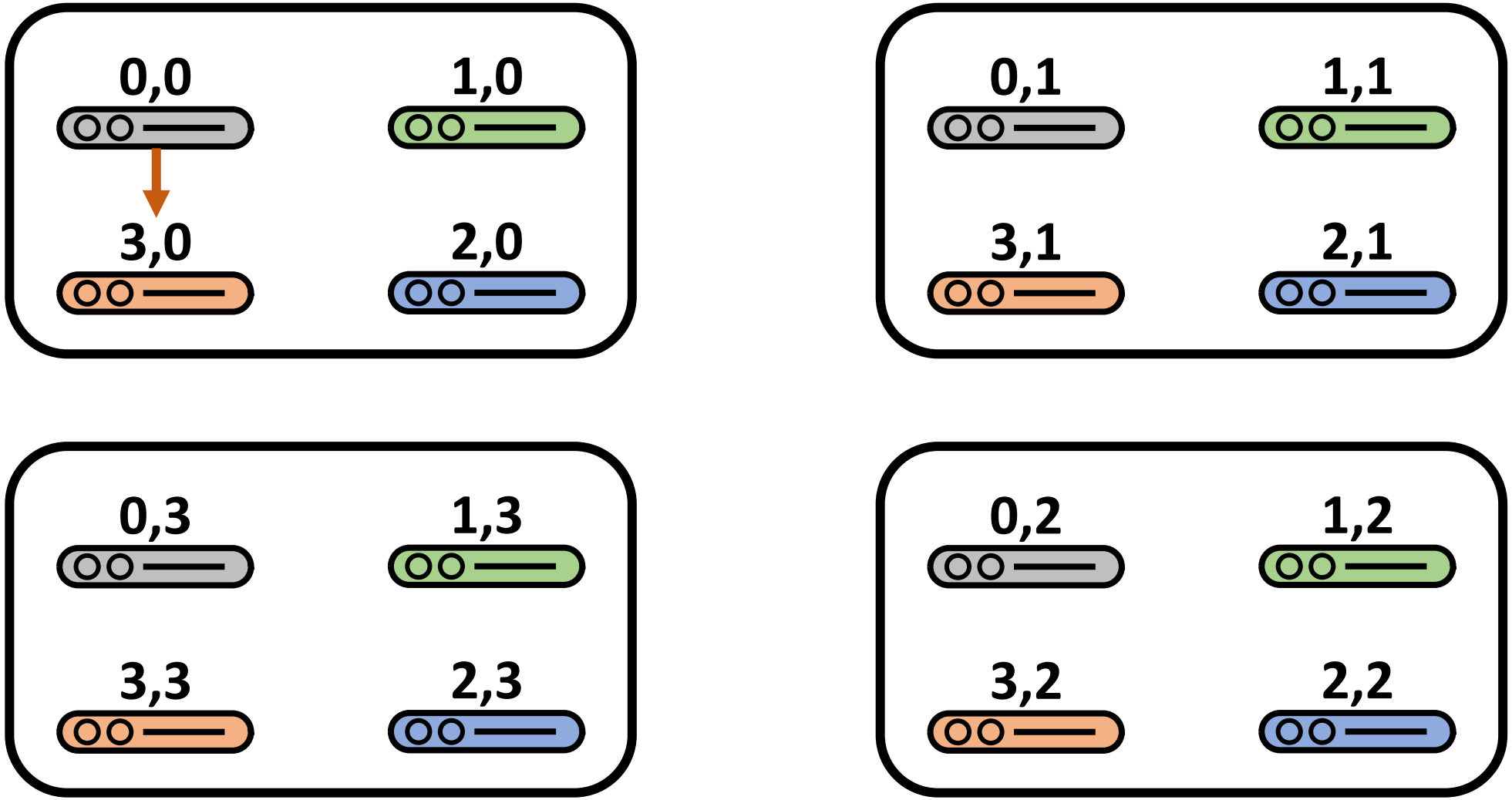
# Shale Schedule ($h$=2)



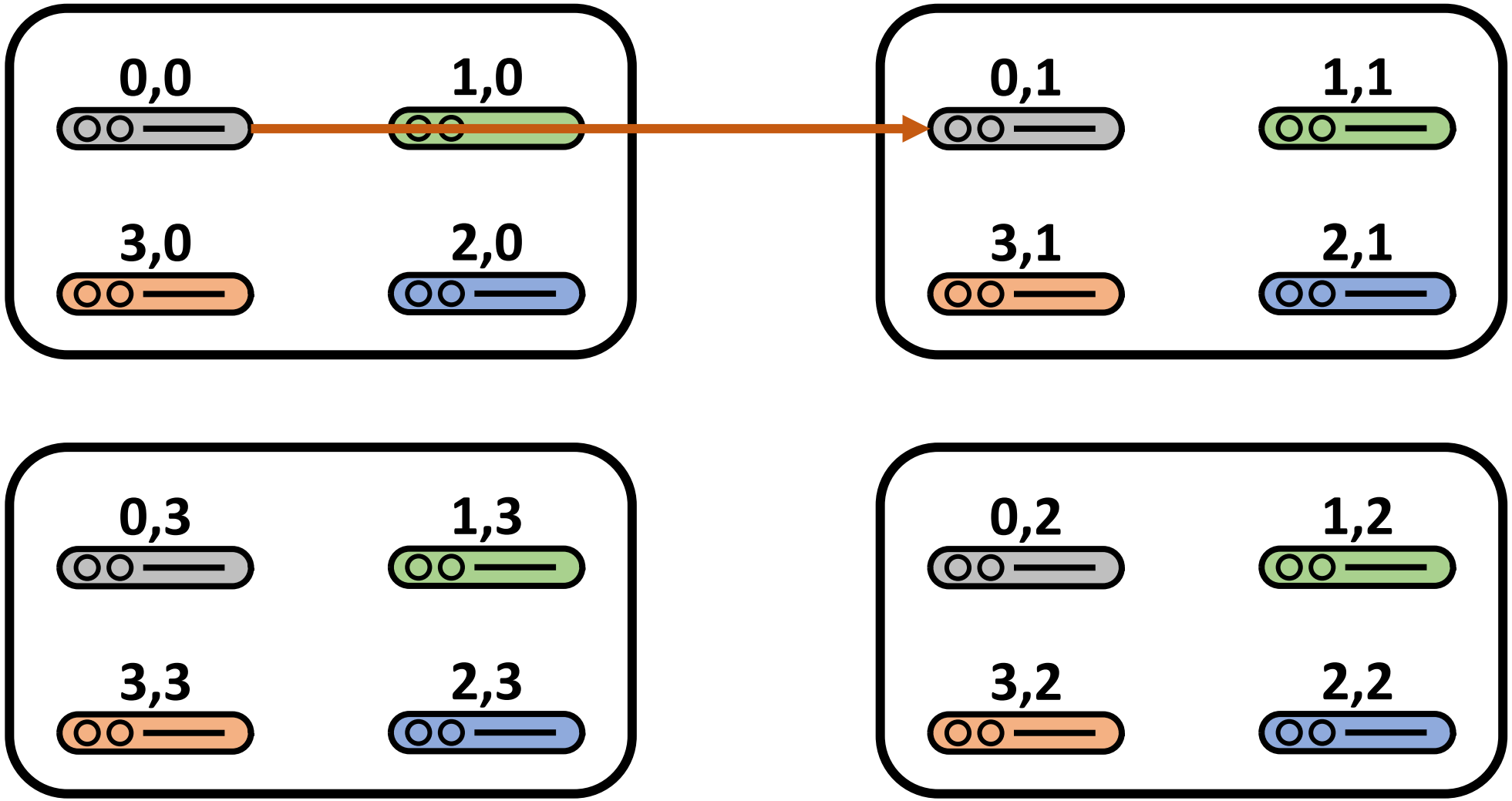
# Shale Schedule ($h$=2)



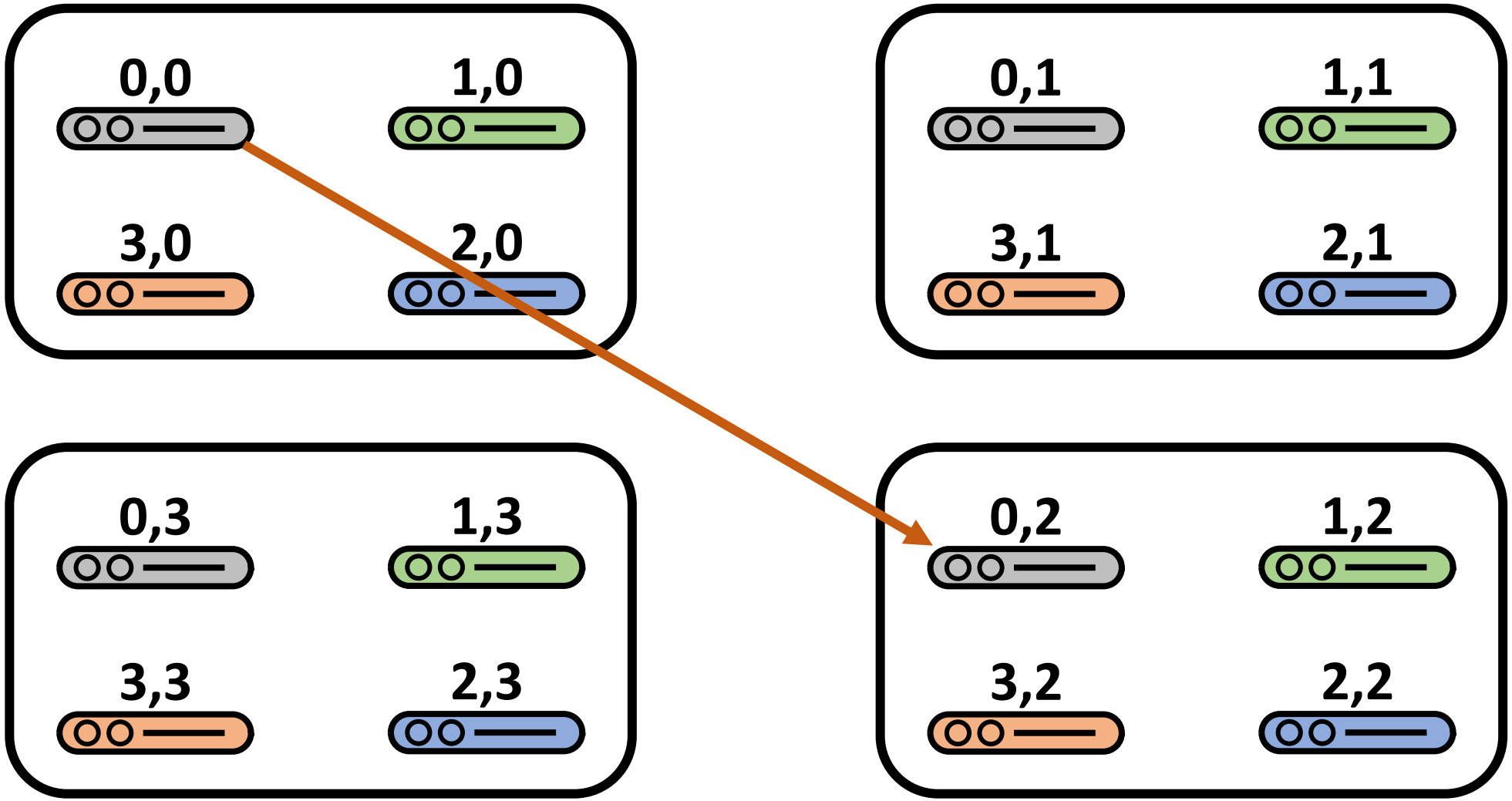
# Shale Schedule ($h$=2)



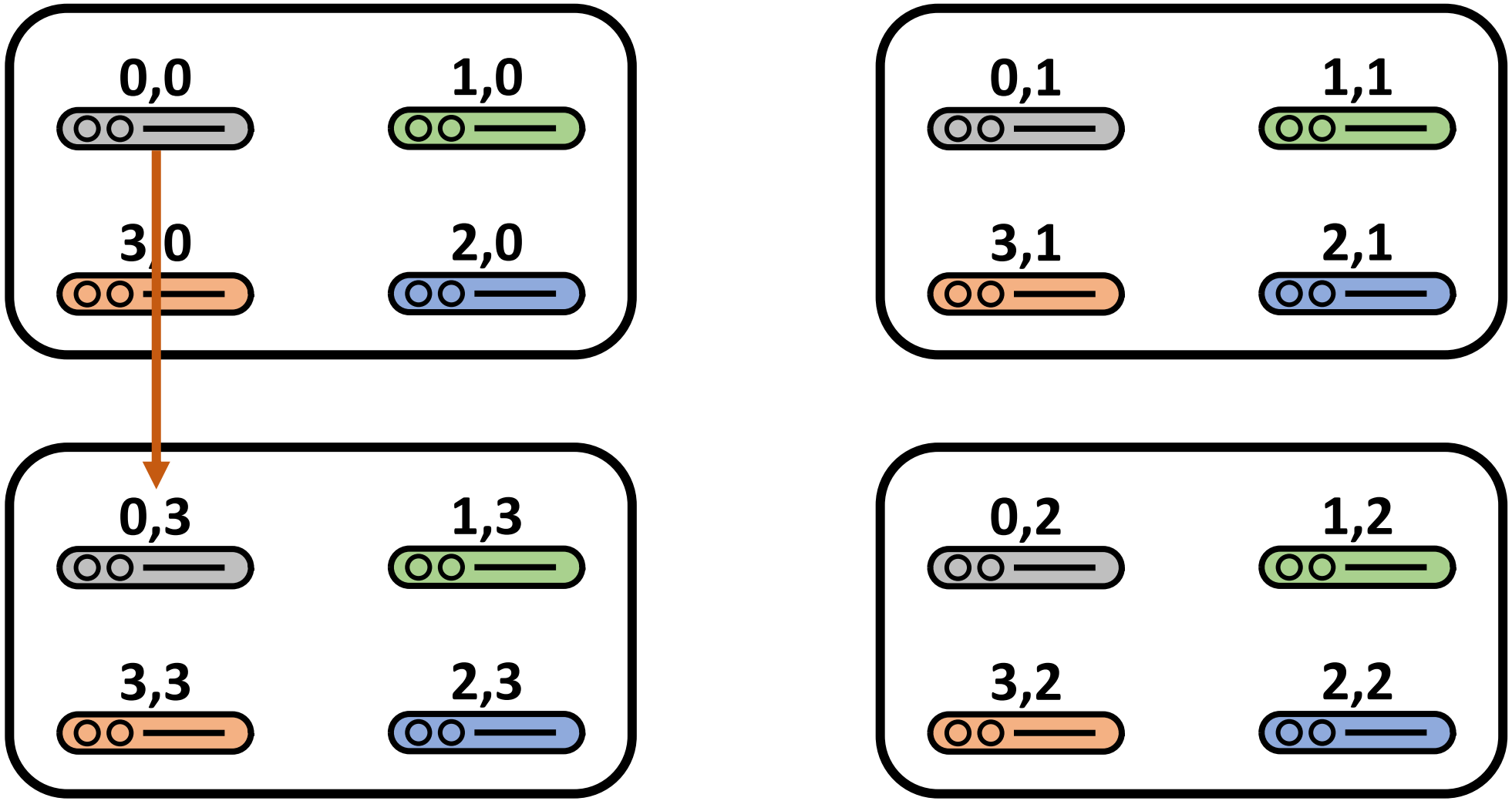
# Shale Schedule ($h$=2)



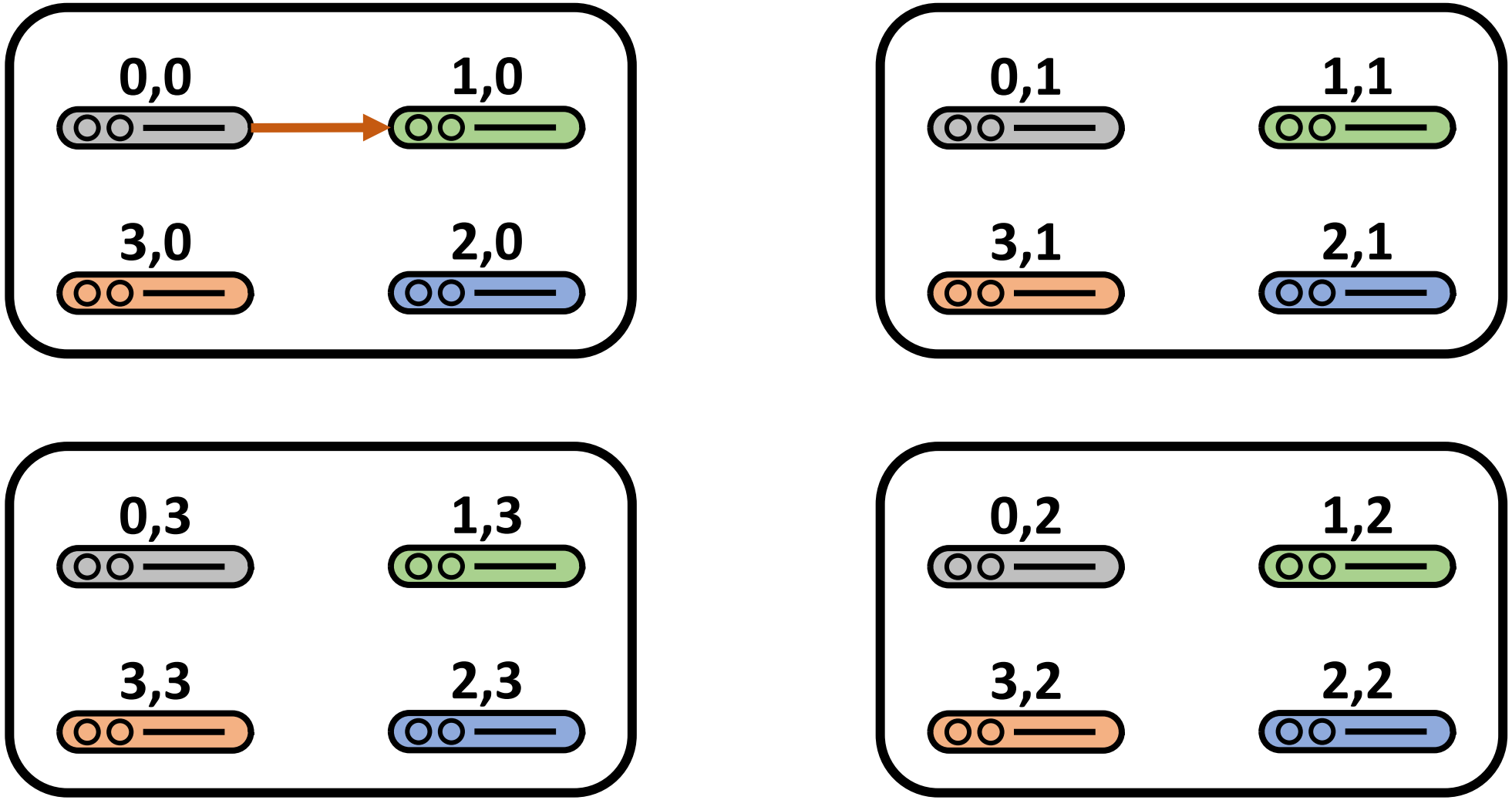
# Shale Schedule ($h$=2)



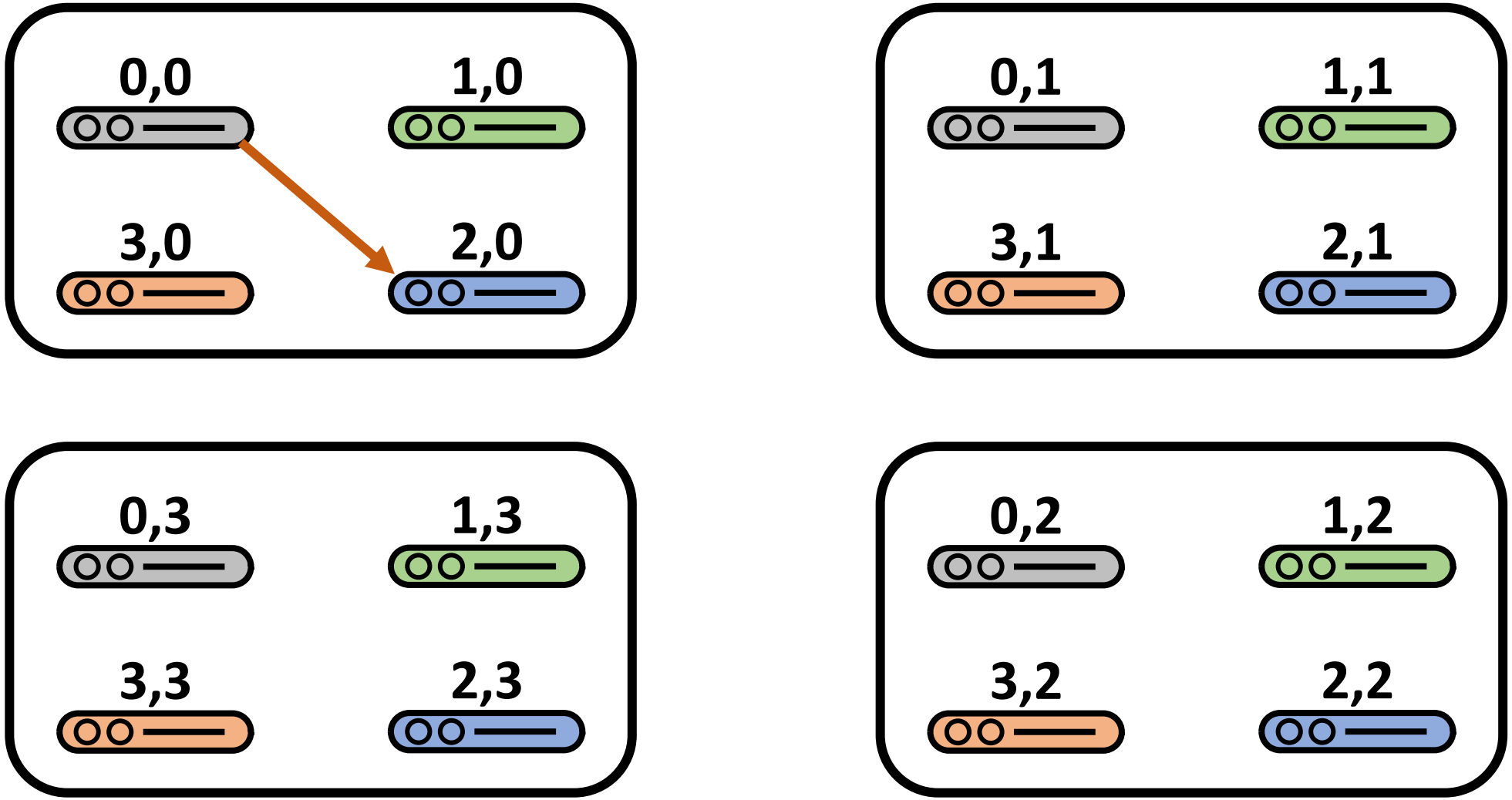
# Shale Schedule ($h$=2)



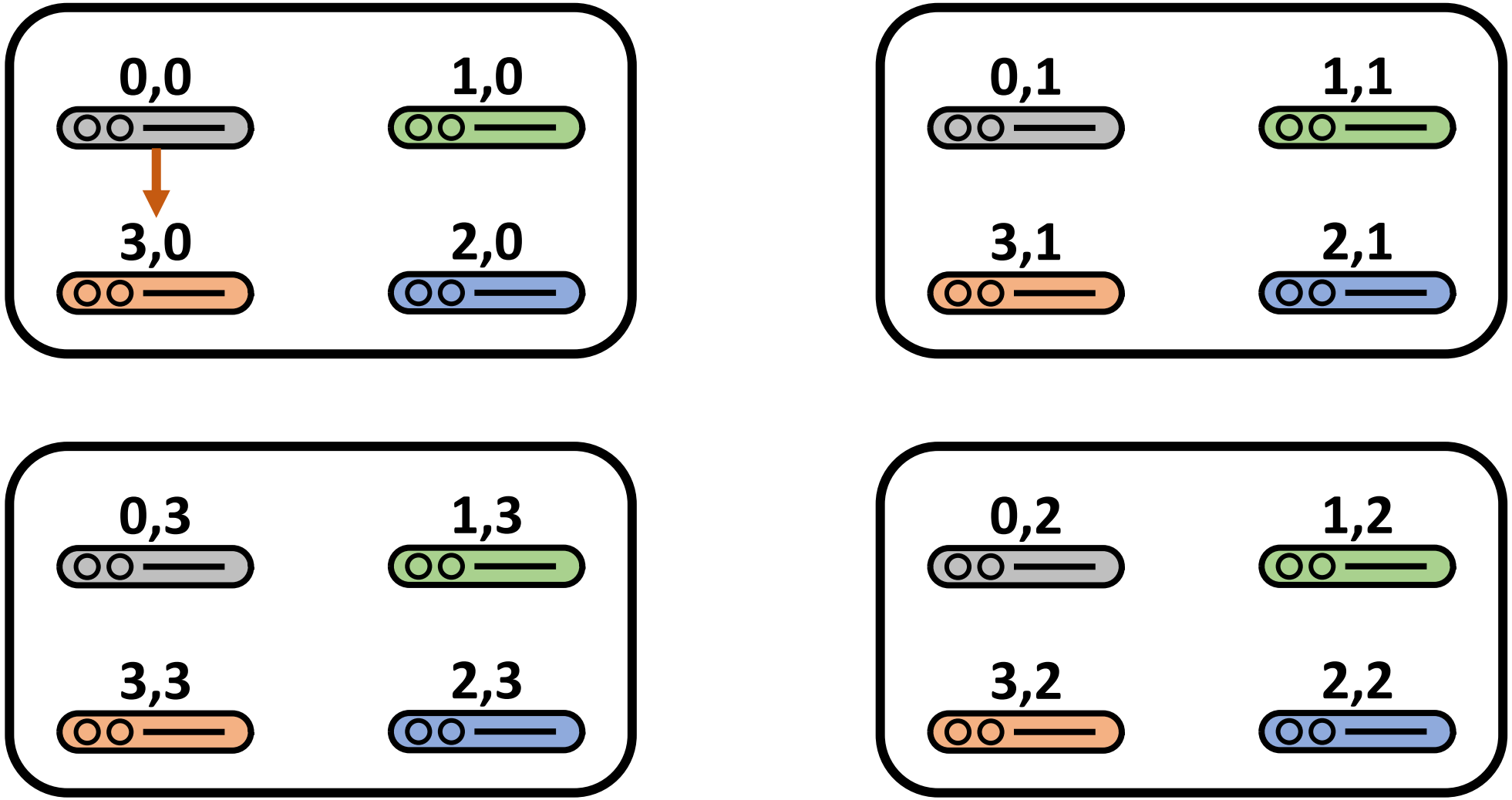
# Shale Schedule ($h$=2)



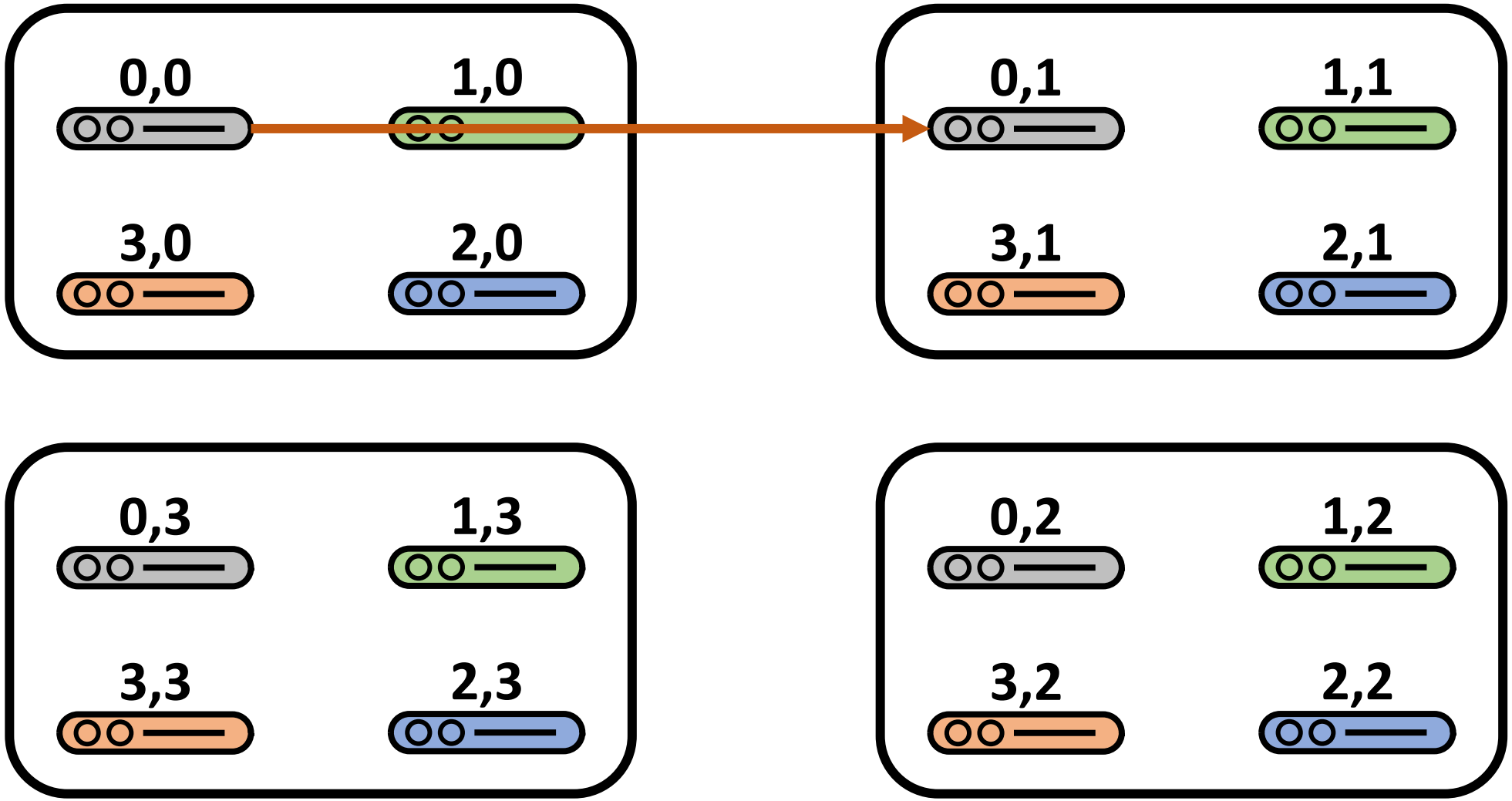
# Shale Schedule ($h$=2)



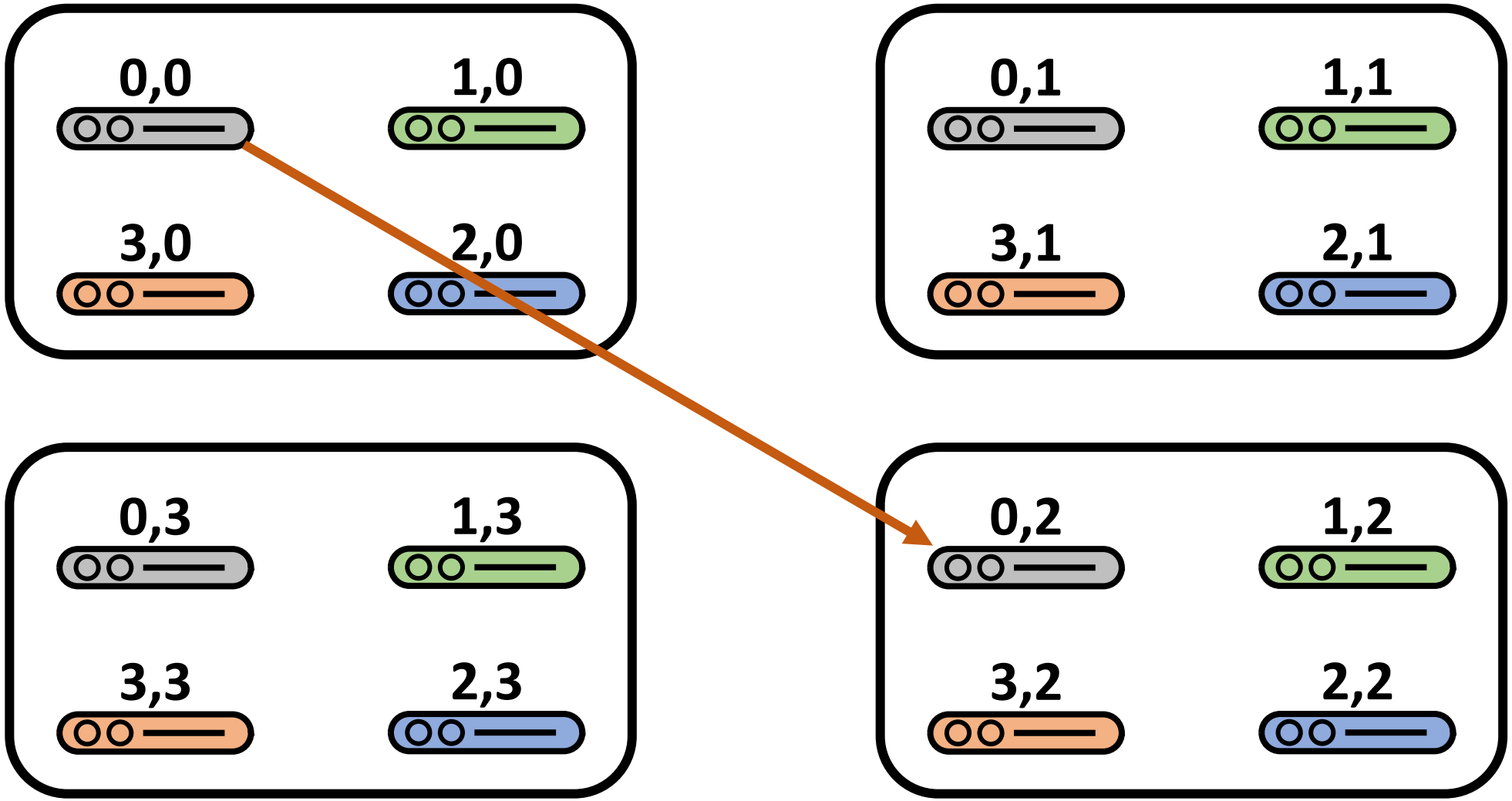
# Shale Schedule ($h$=2)



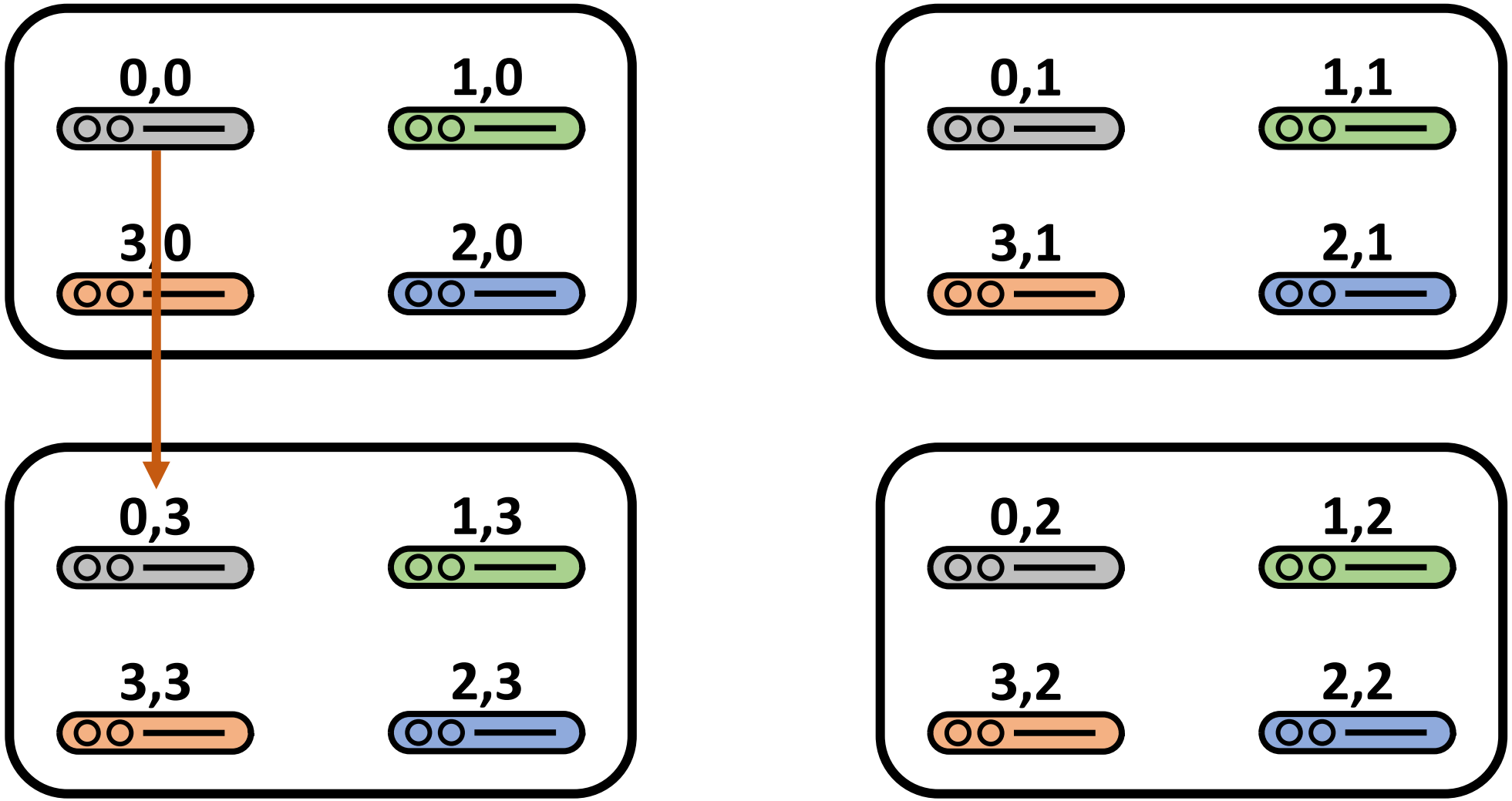
# Shale Schedule ($h$=2)



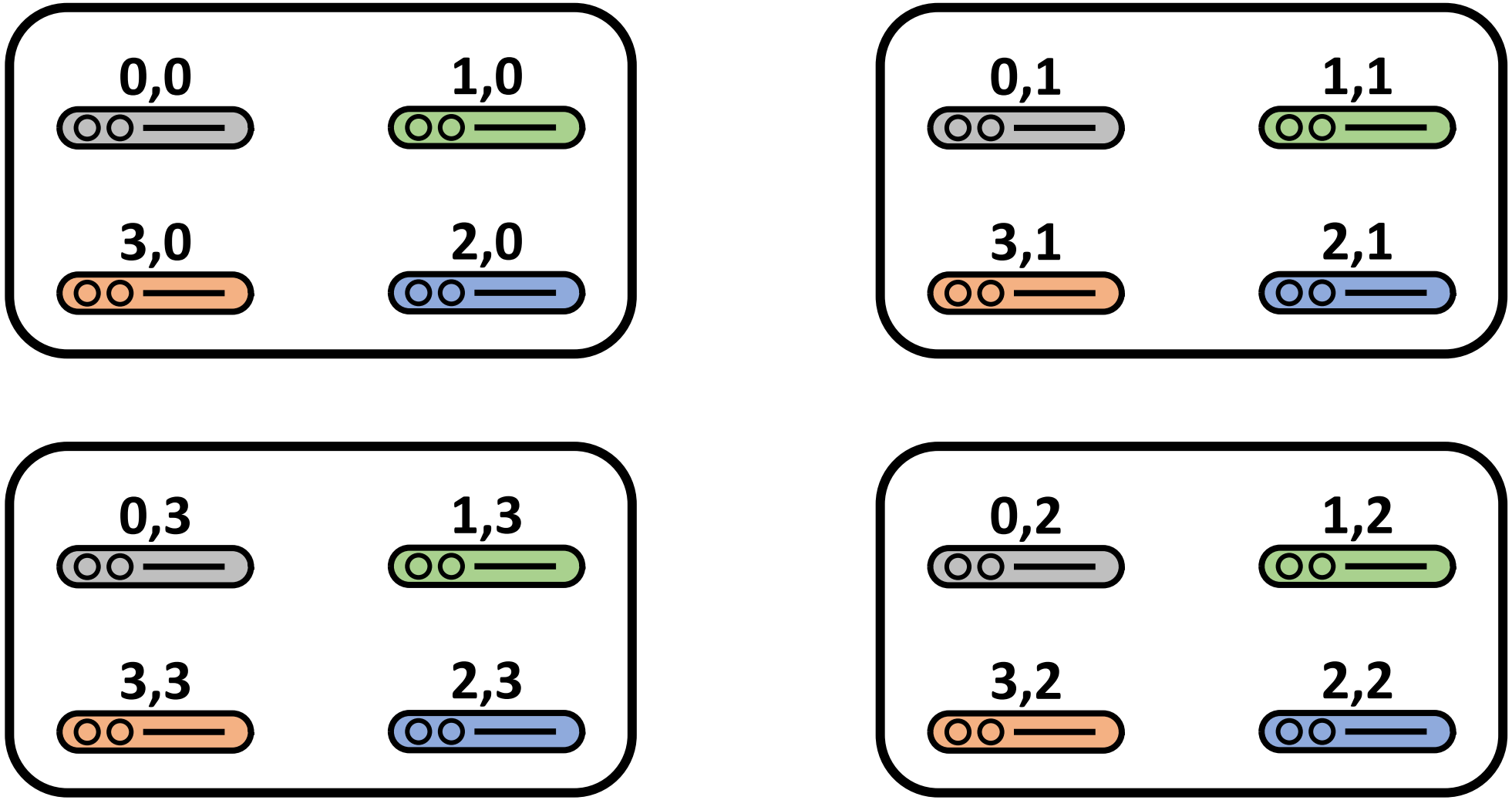
# Shale Schedule ($h$=2)



# Shale Schedule ($h$=2)



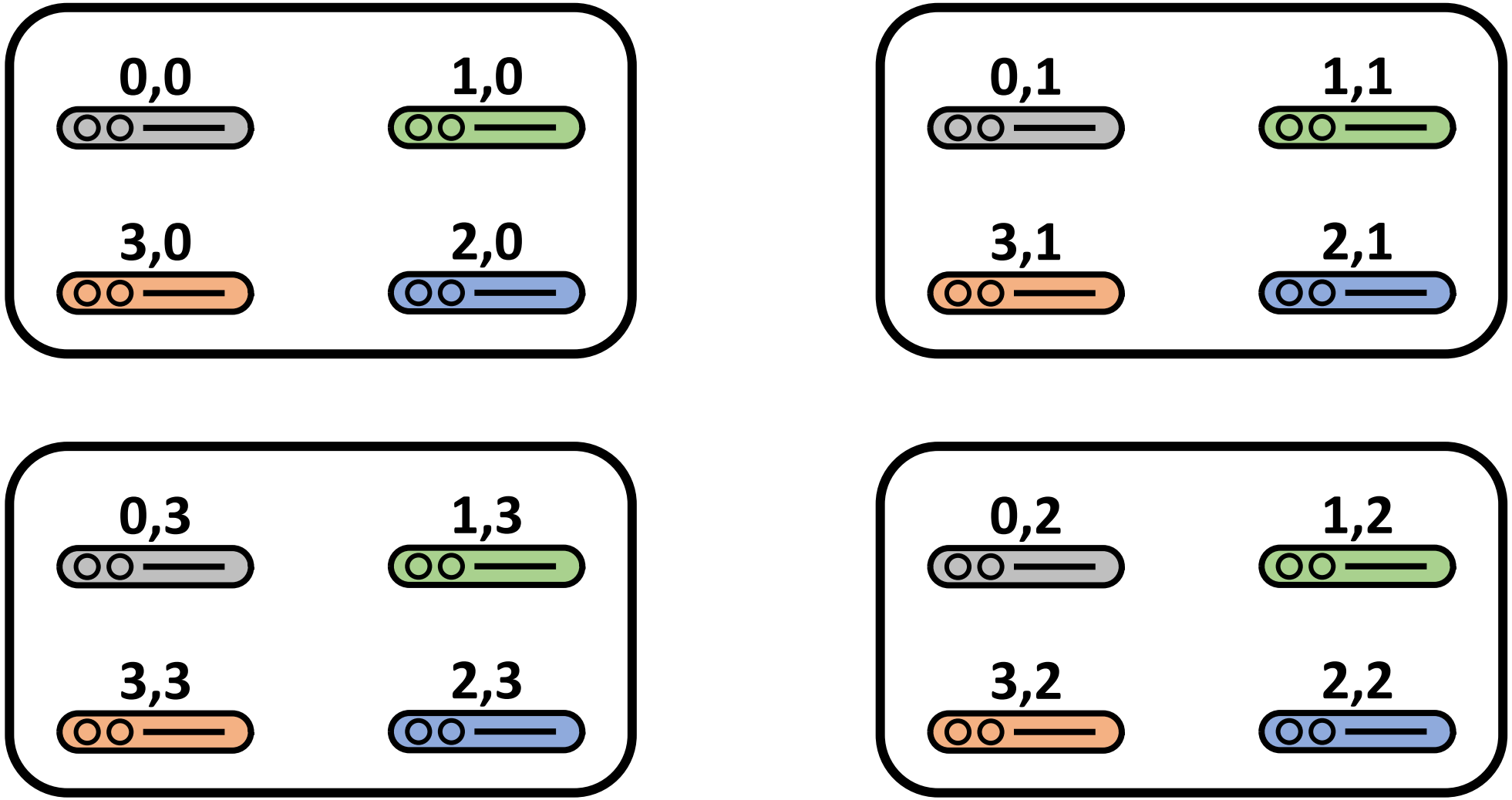
# Shale Schedule ($h$=2)



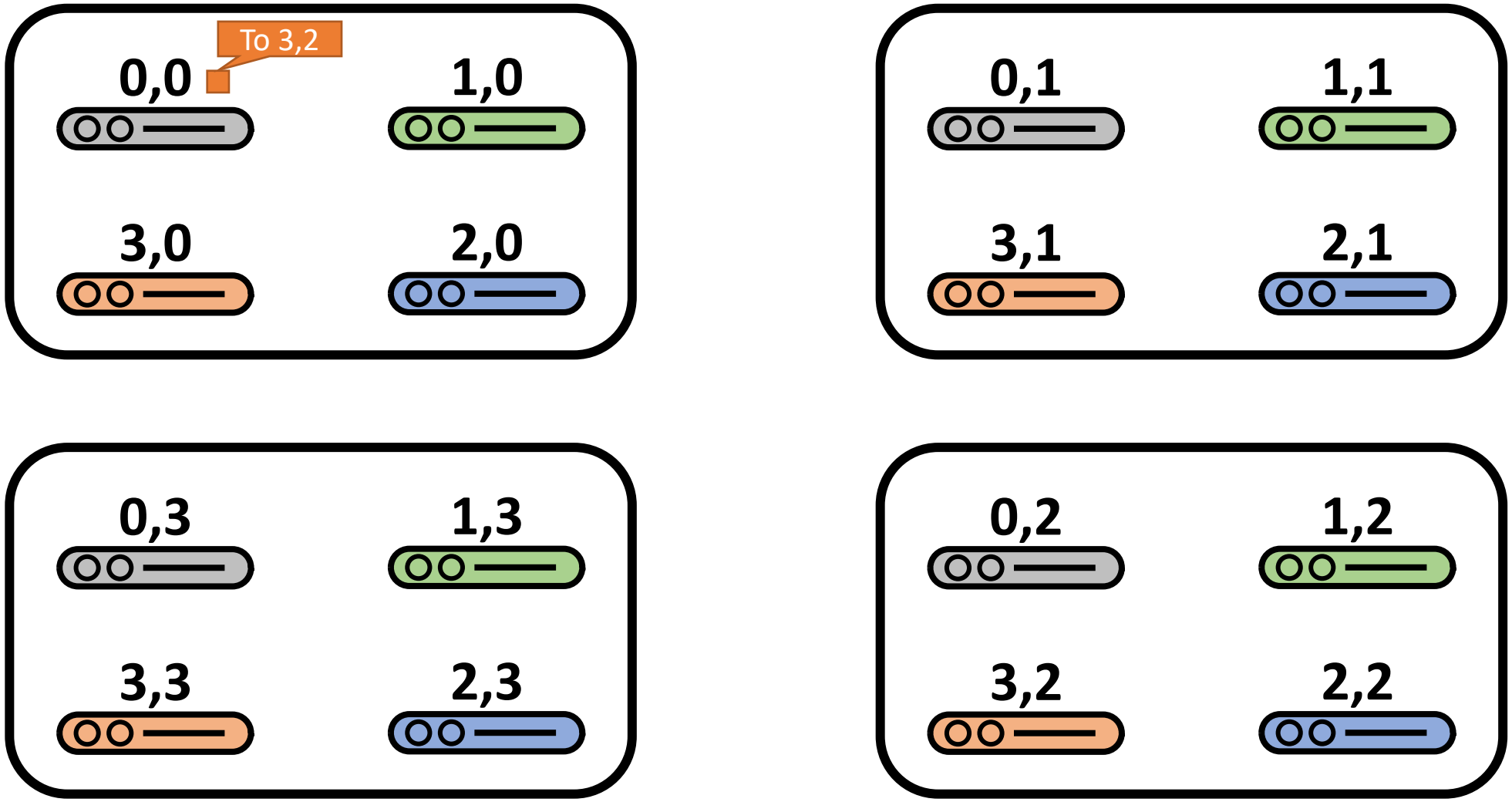
# Shale Schedule and Routing

- 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

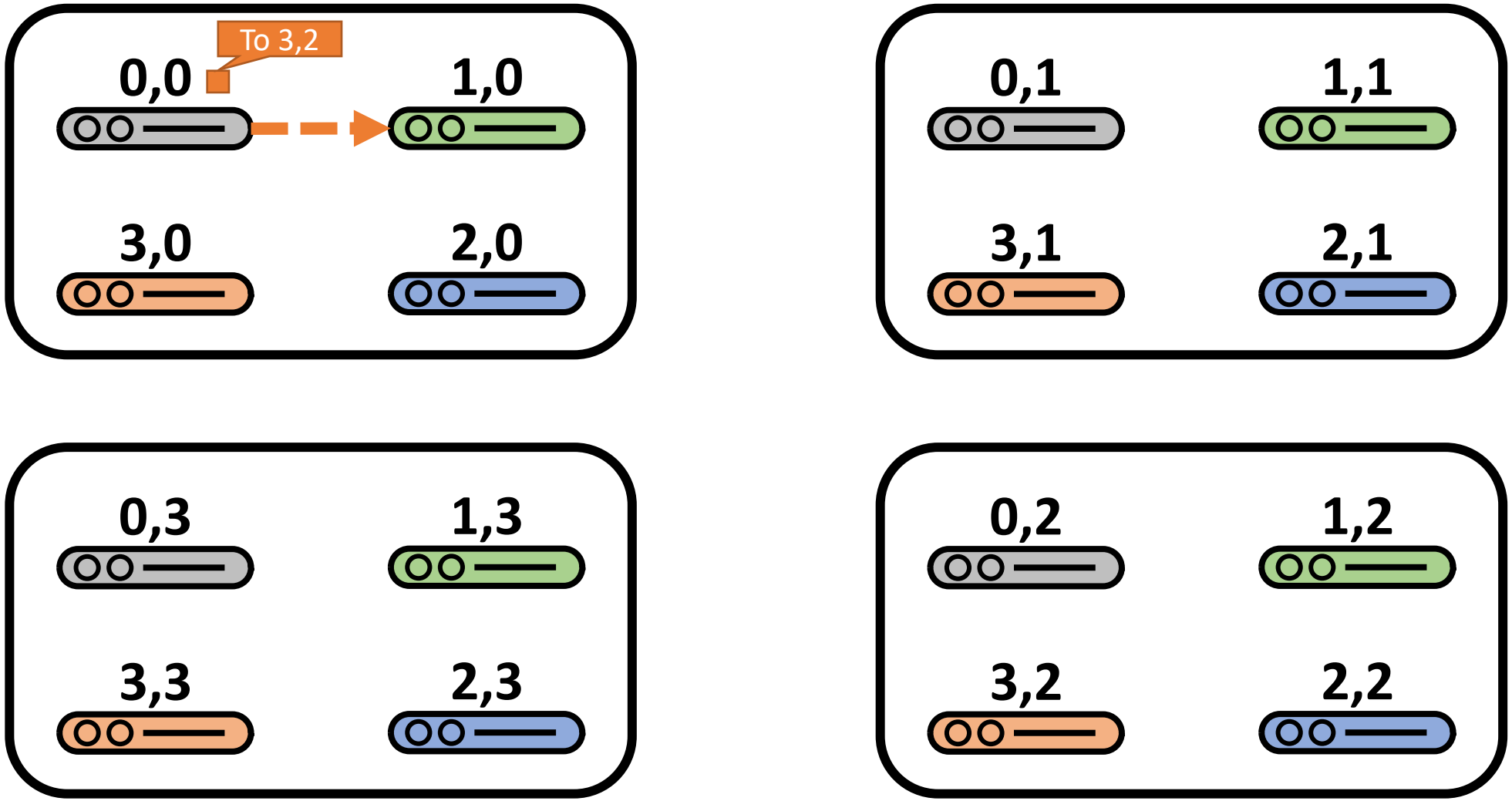
# Shale Schedule ($h$=2)



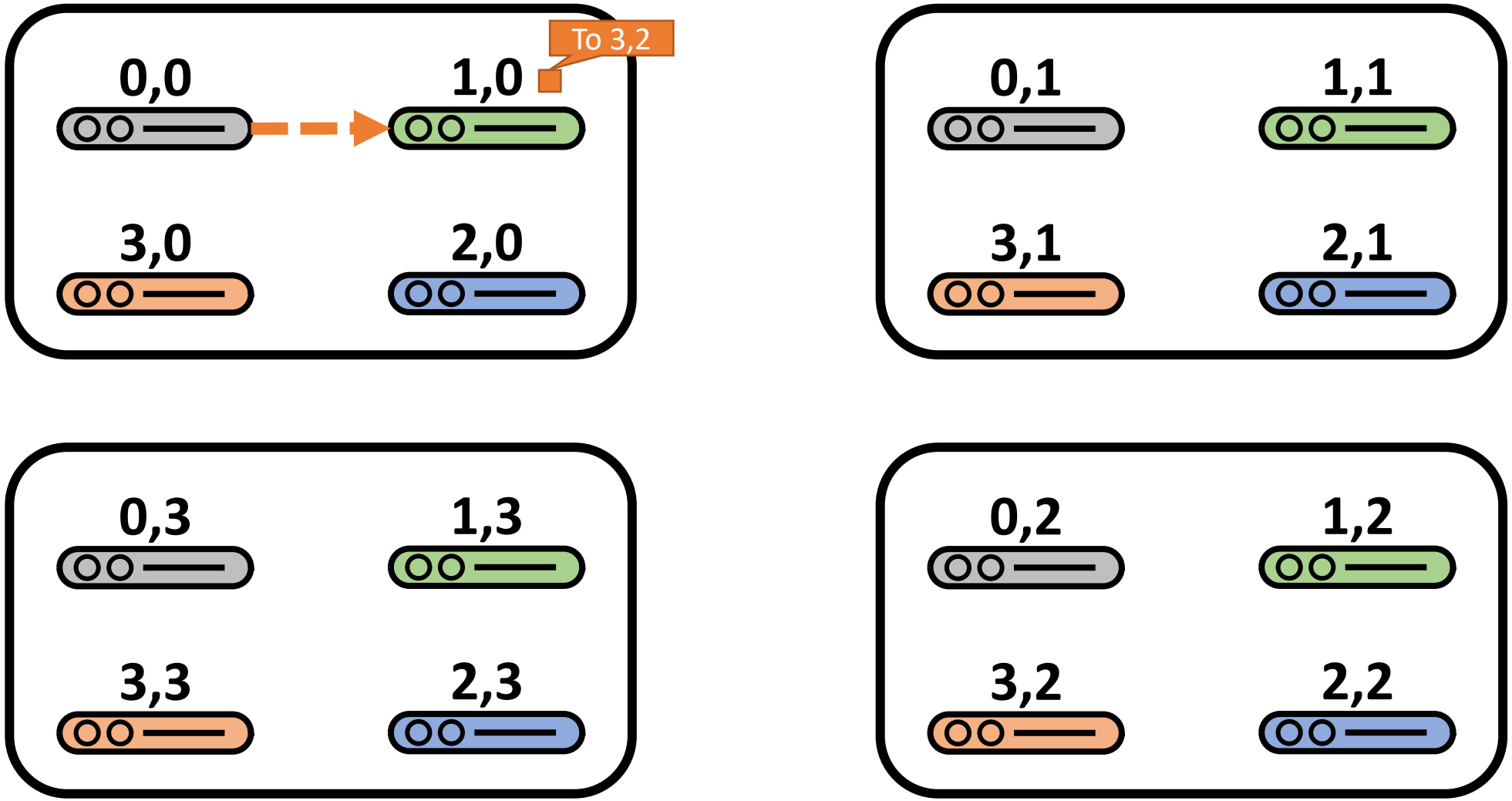
# Shale Schedule ($h$=2)



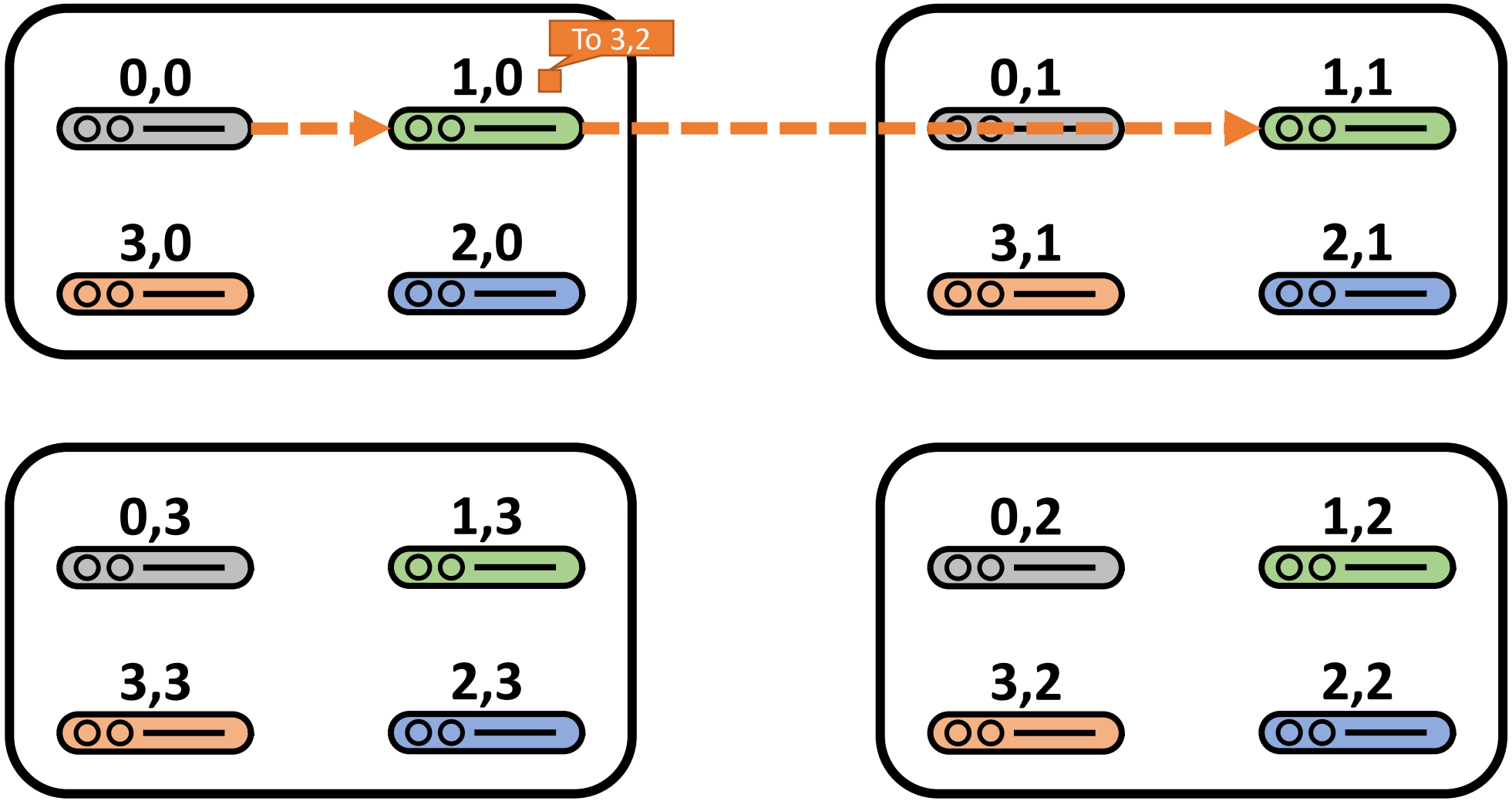
# Shale Schedule ($h$=2)



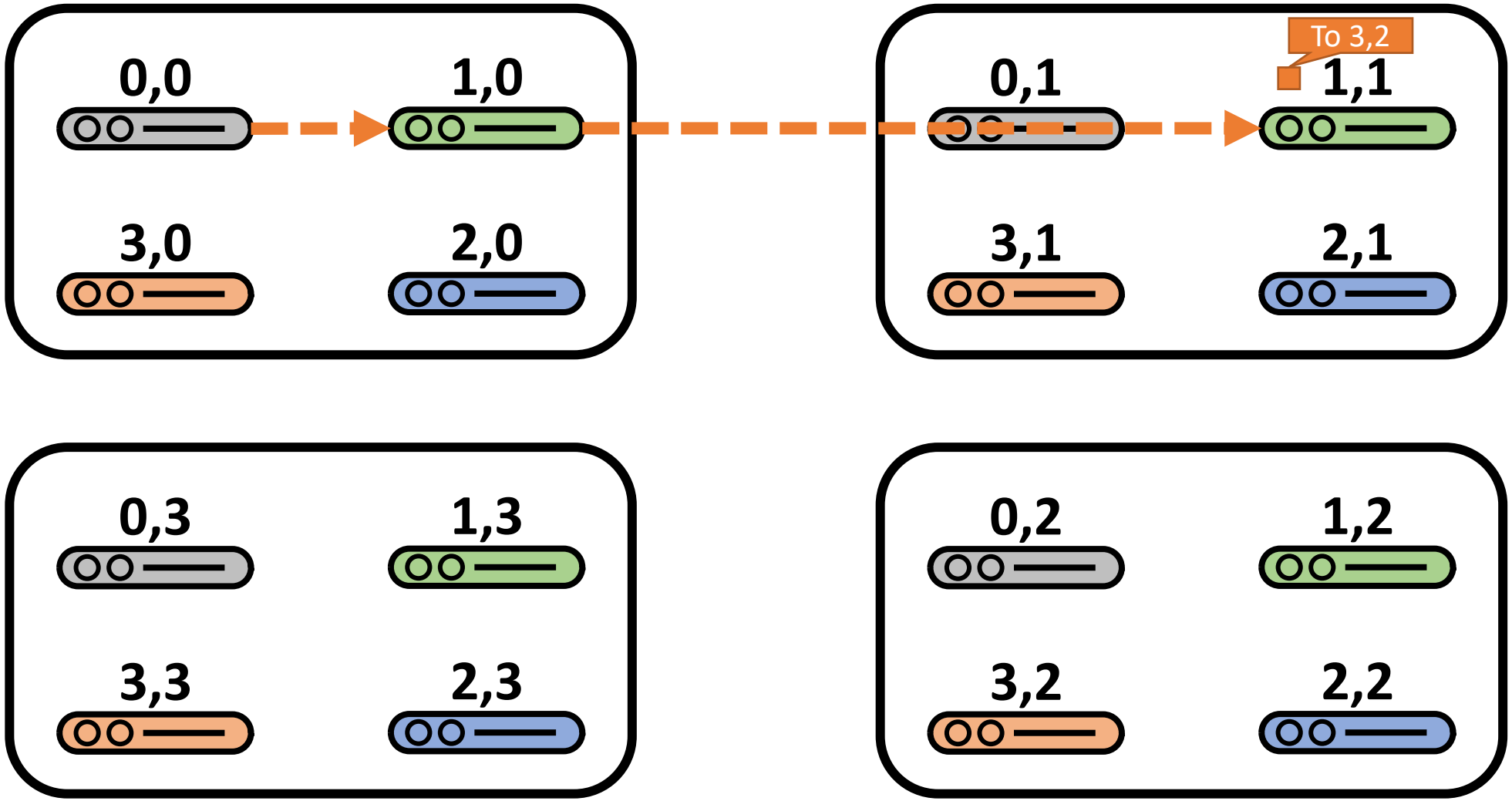
# Shale Schedule ($h$=2)



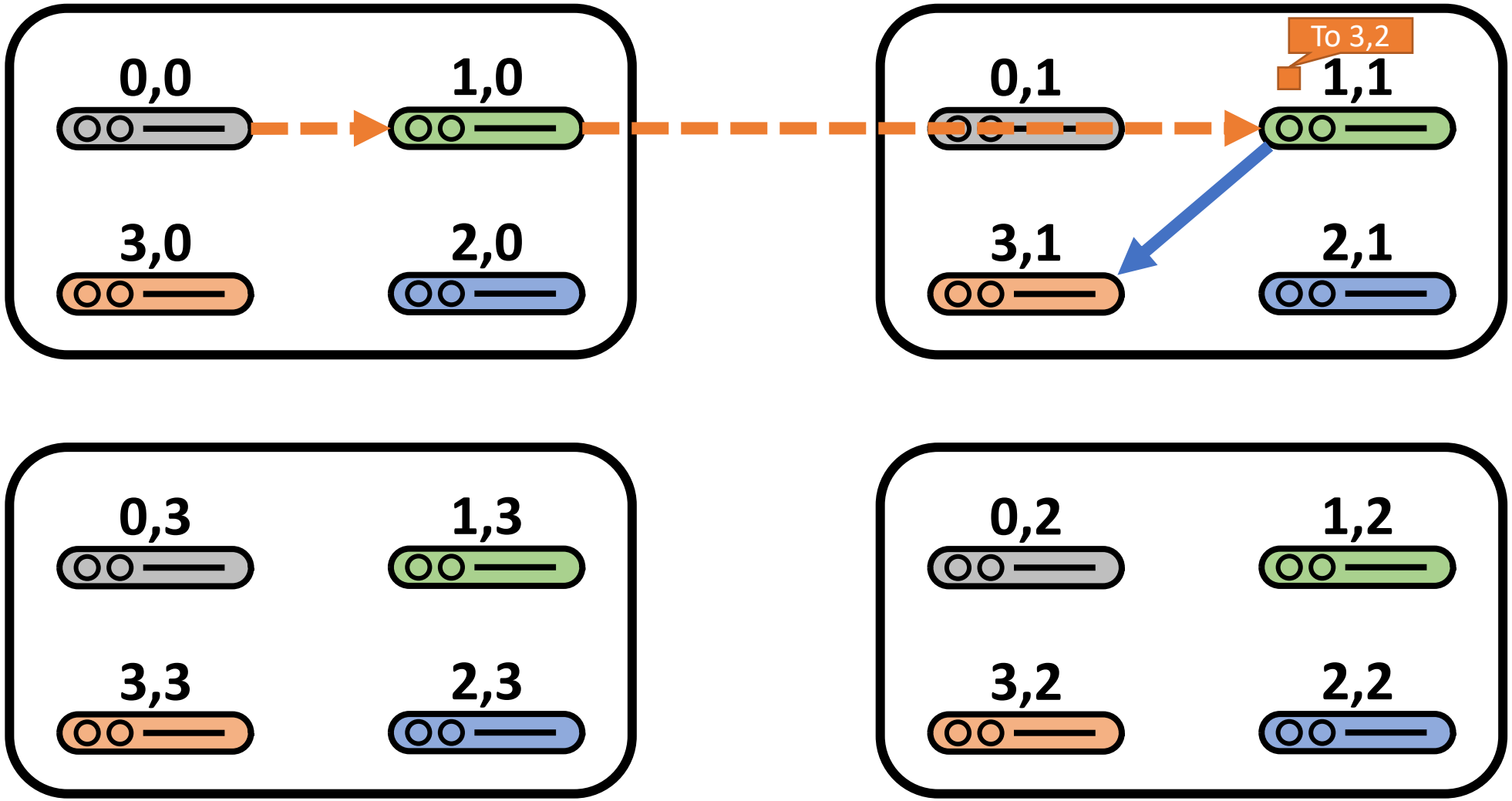
# Shale Schedule ($h$=2)



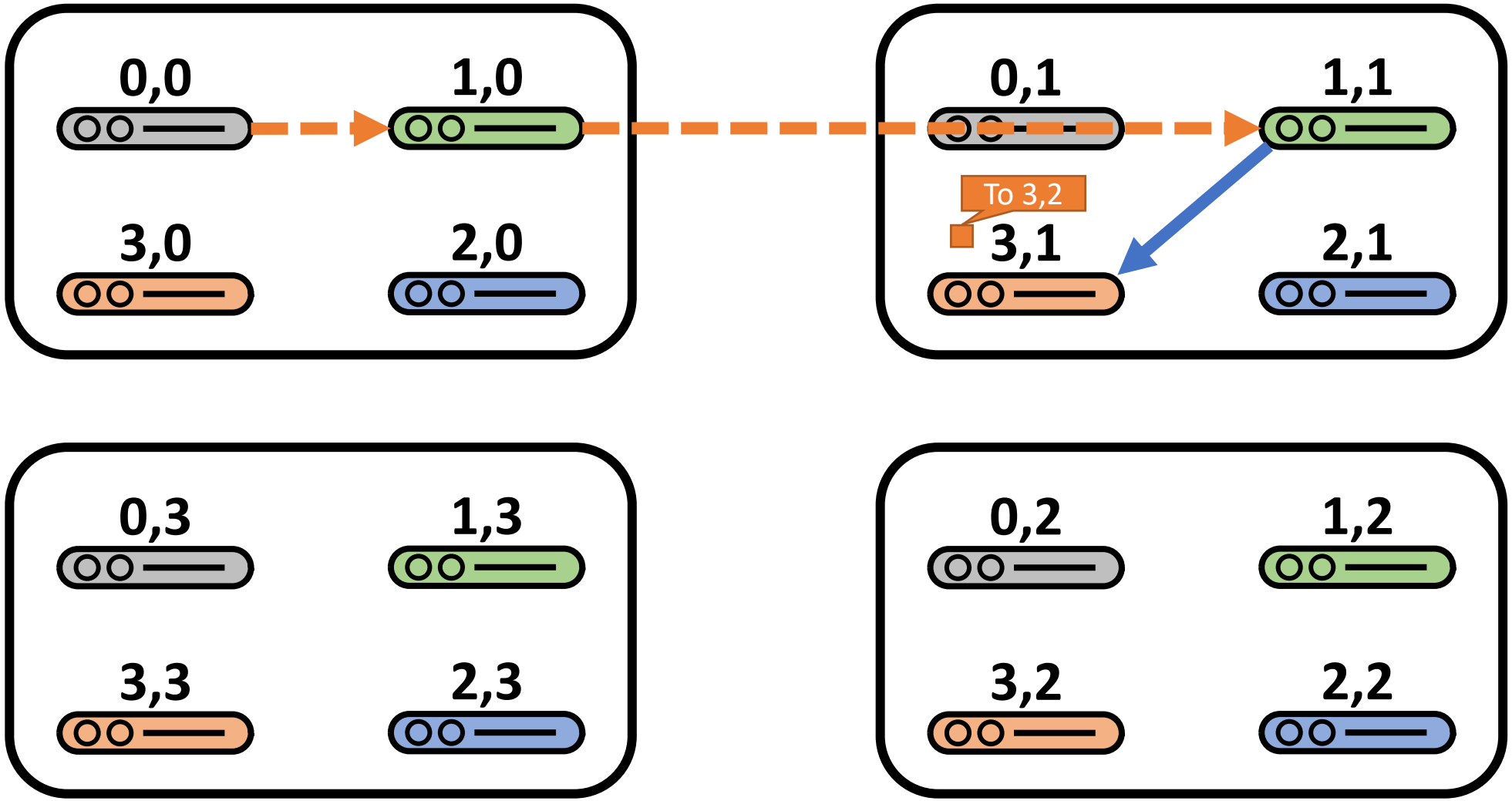
# Shale Schedule ($h$=2)



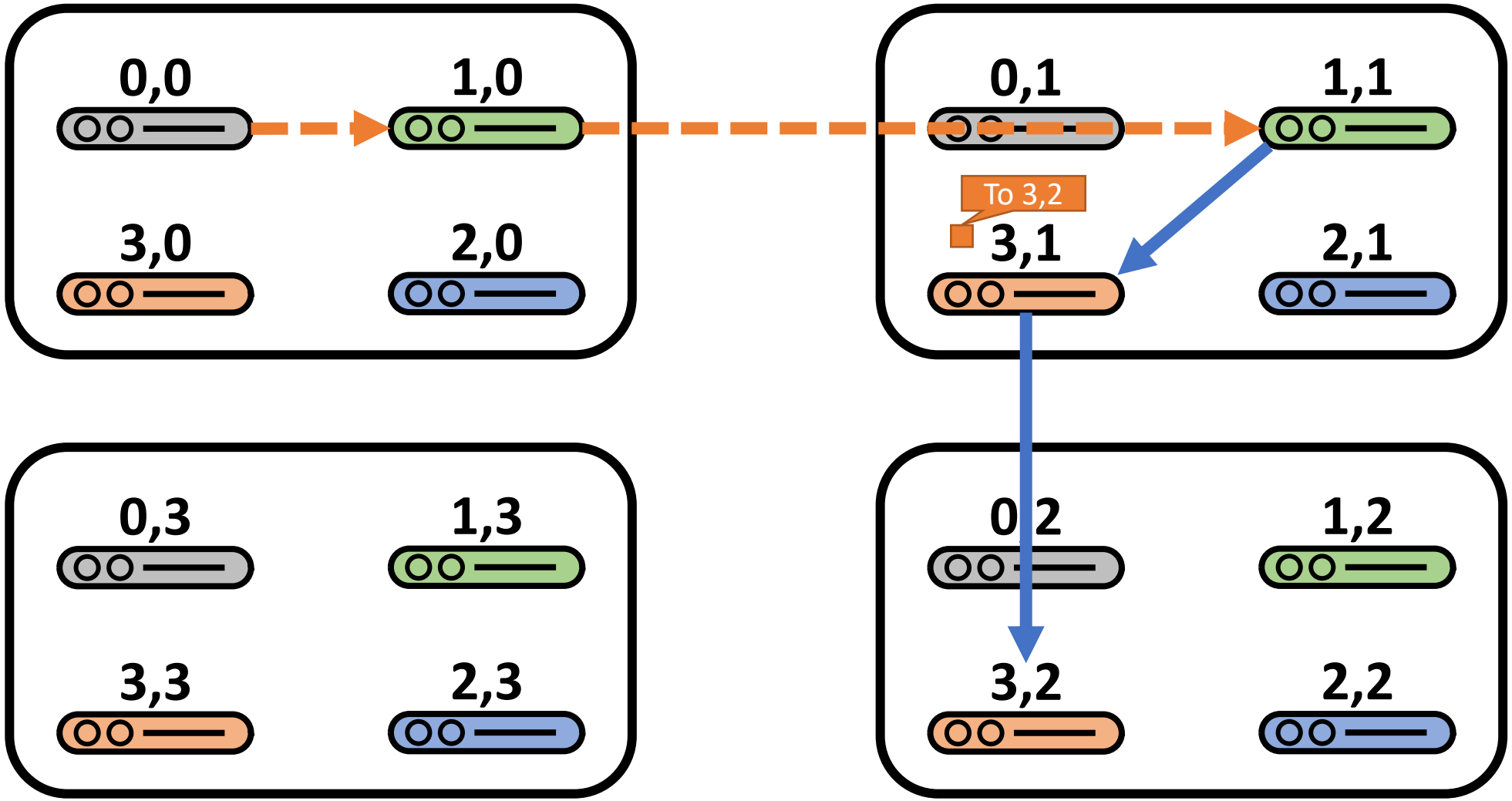
# Shale Schedule ($h$=2)



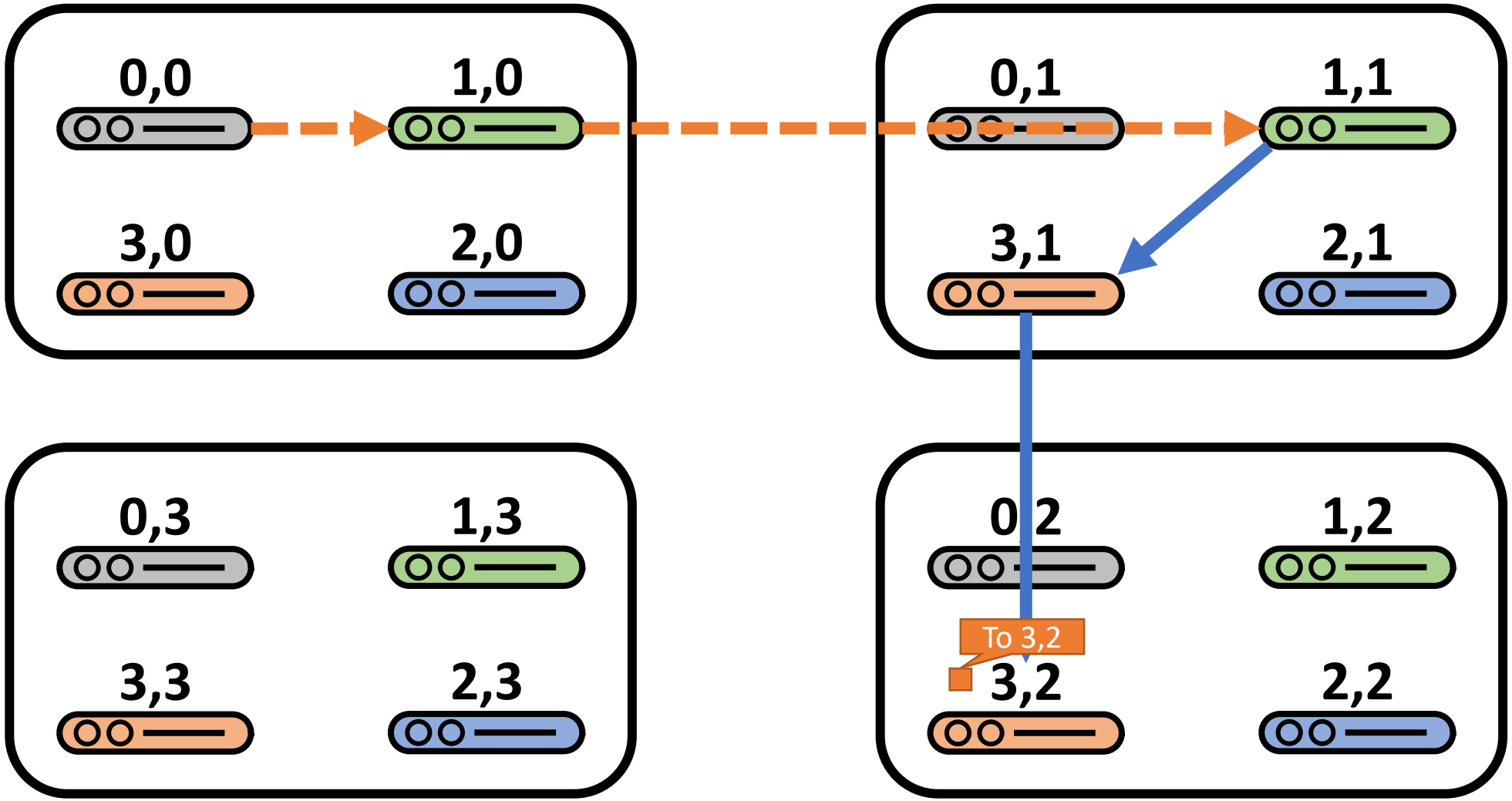
# Shale Schedule ($h$=2)



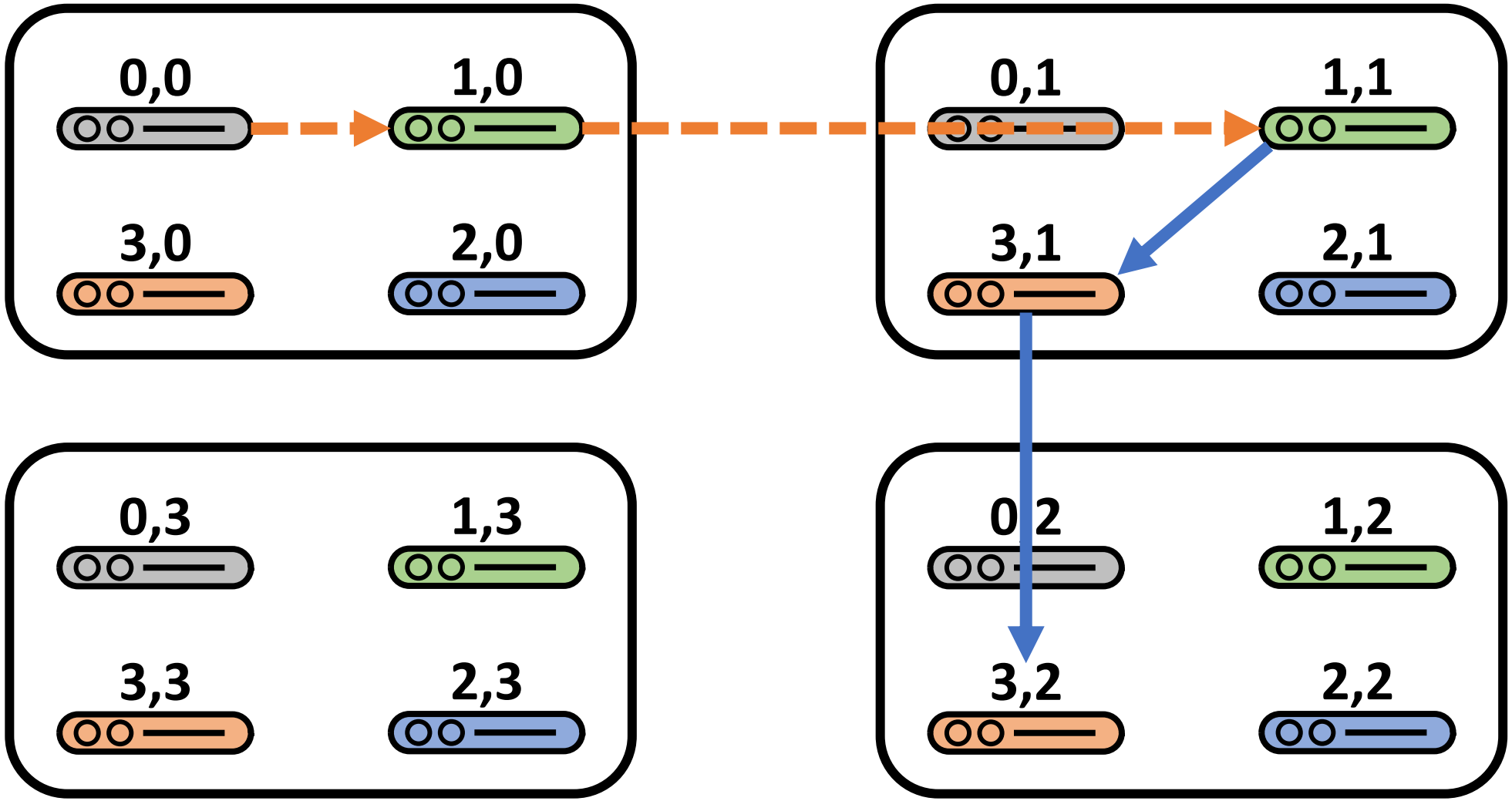
# Shale Schedule ($h$=2)



# Shale Schedule ($h$=2)



# Shale Schedule ($h$=2)

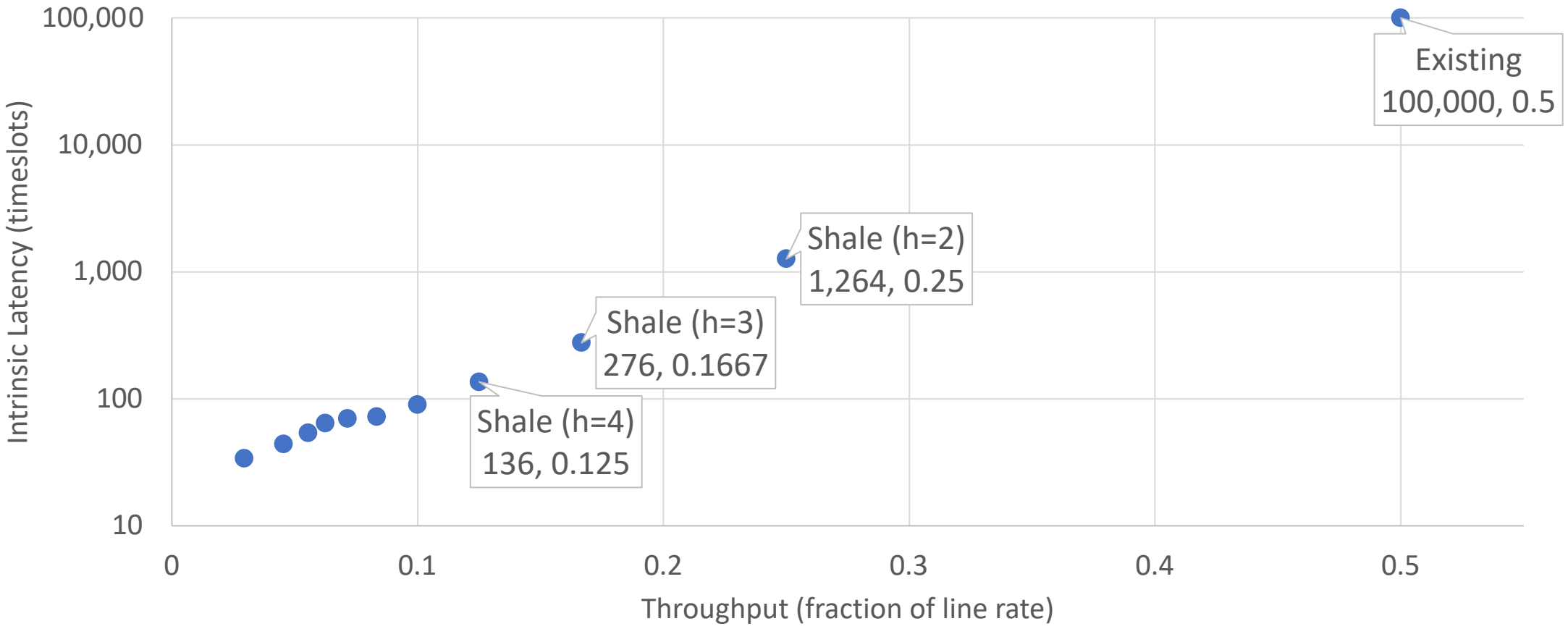


# Shale Schedule and Routing

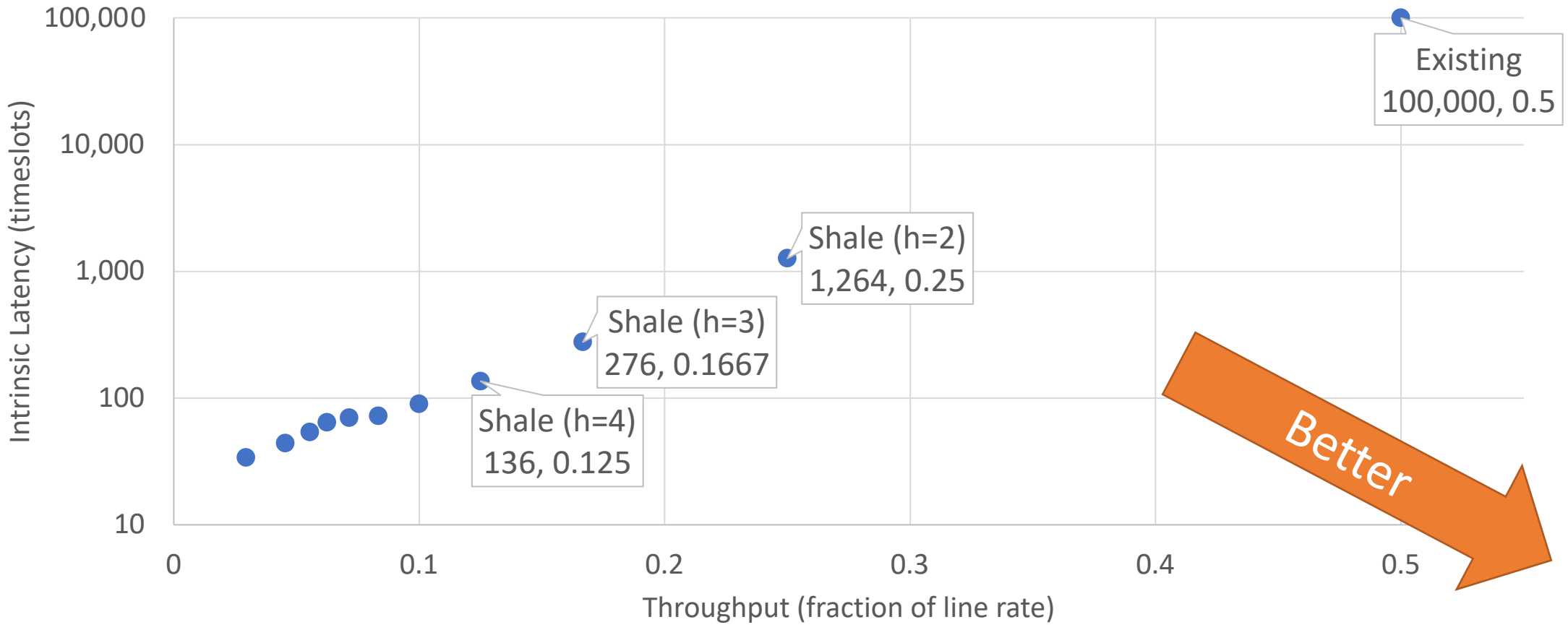
- Generalization of existing round-robin design
- Each node participates in $h$ different round robins
  - Each round robin has just $\sqrt[h]{N}$ nodes
- Direct paths between nodes are now $h$ hops long
- Still use VLB

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

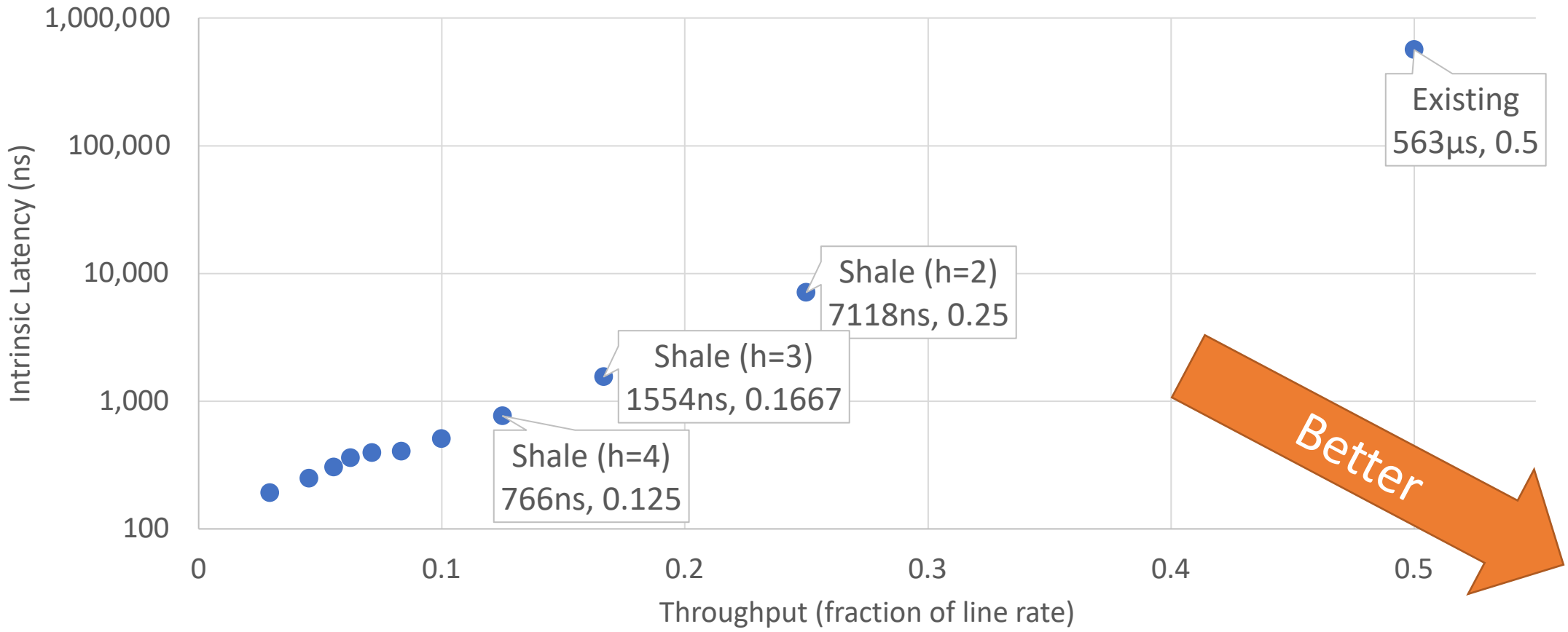
# Comparison for 100,000 nodes



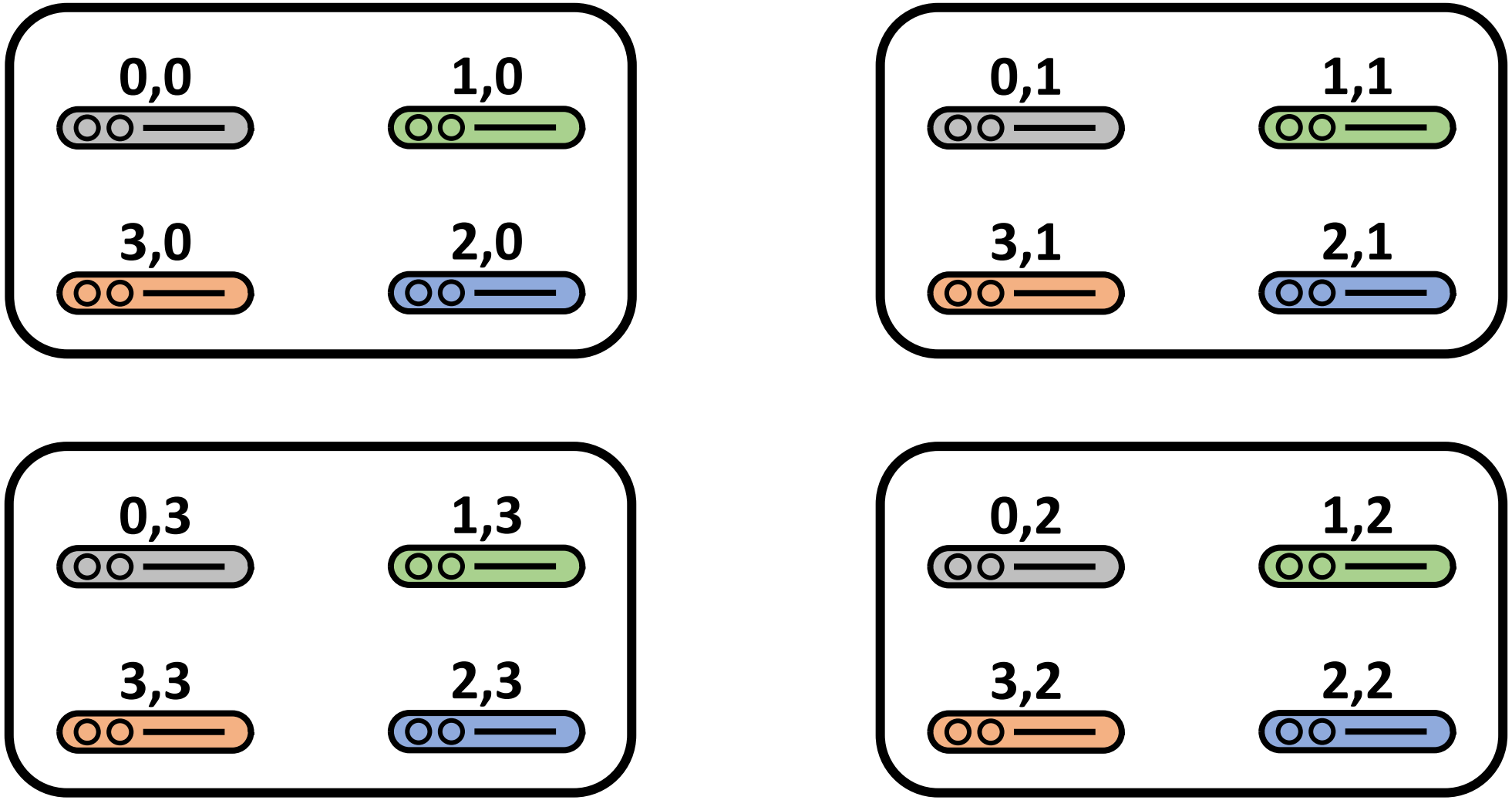
# Comparison for 100,000 nodes



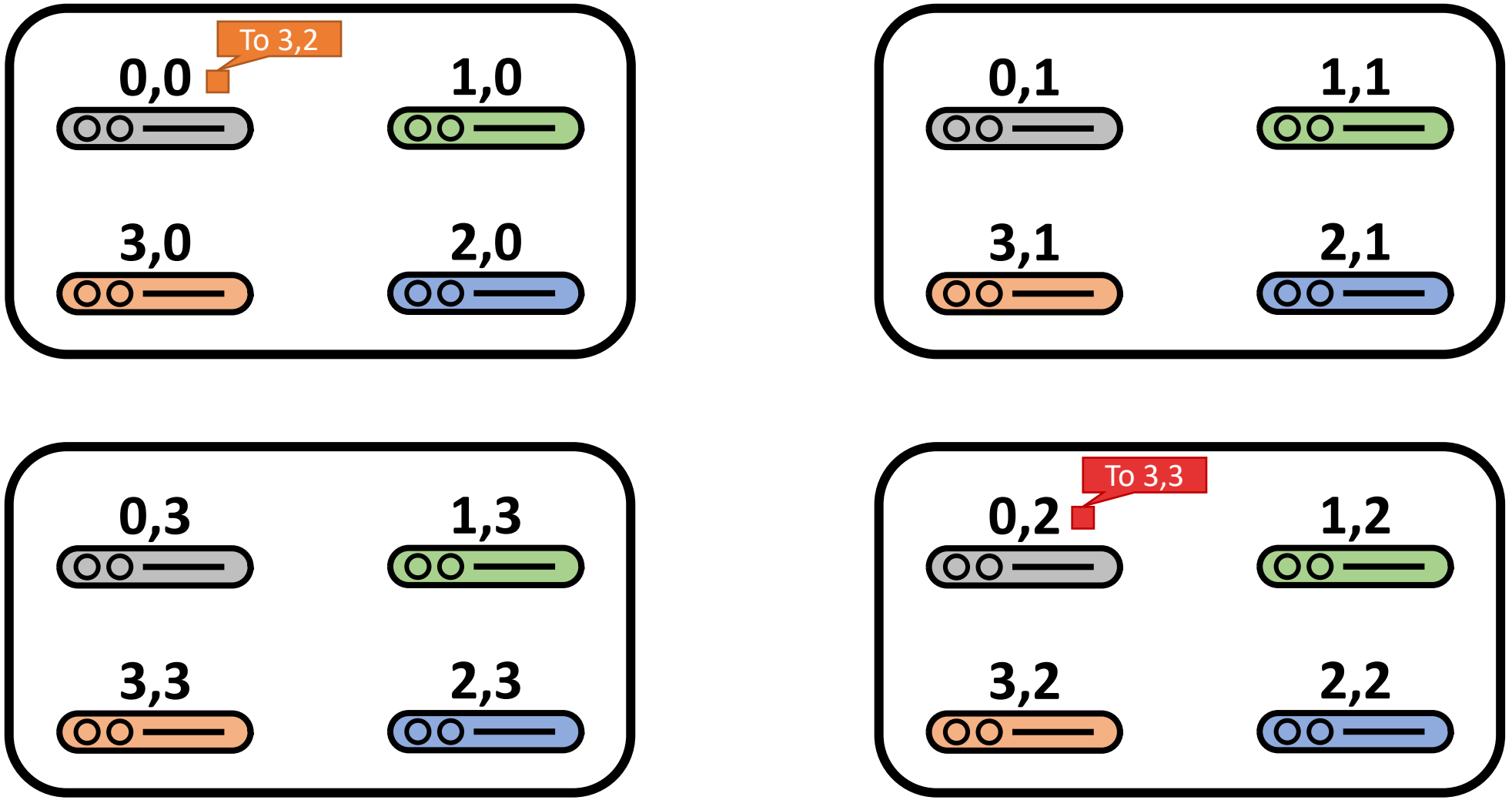
# Comparison for 100,000 nodes



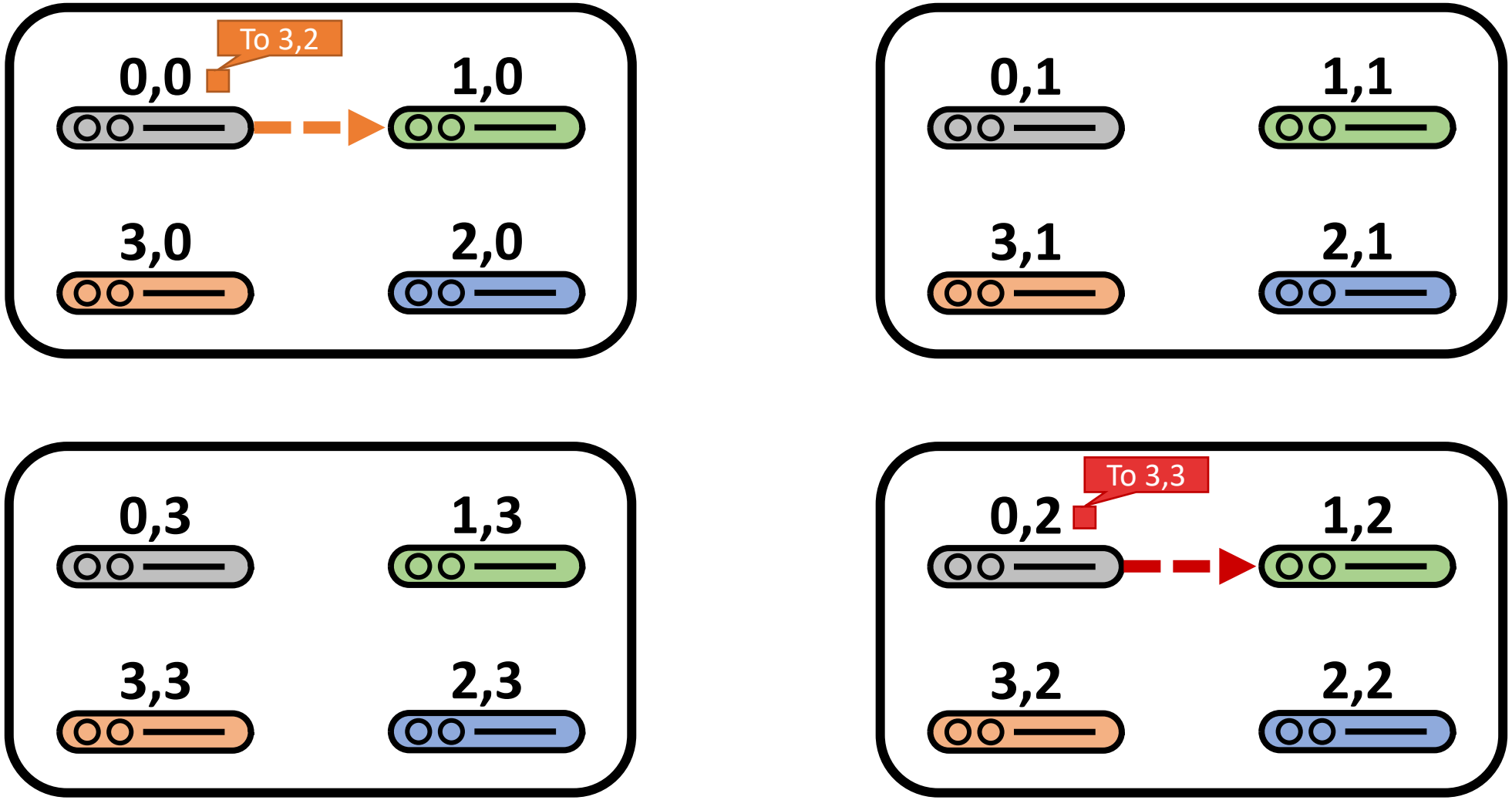
# Queuing in Shale



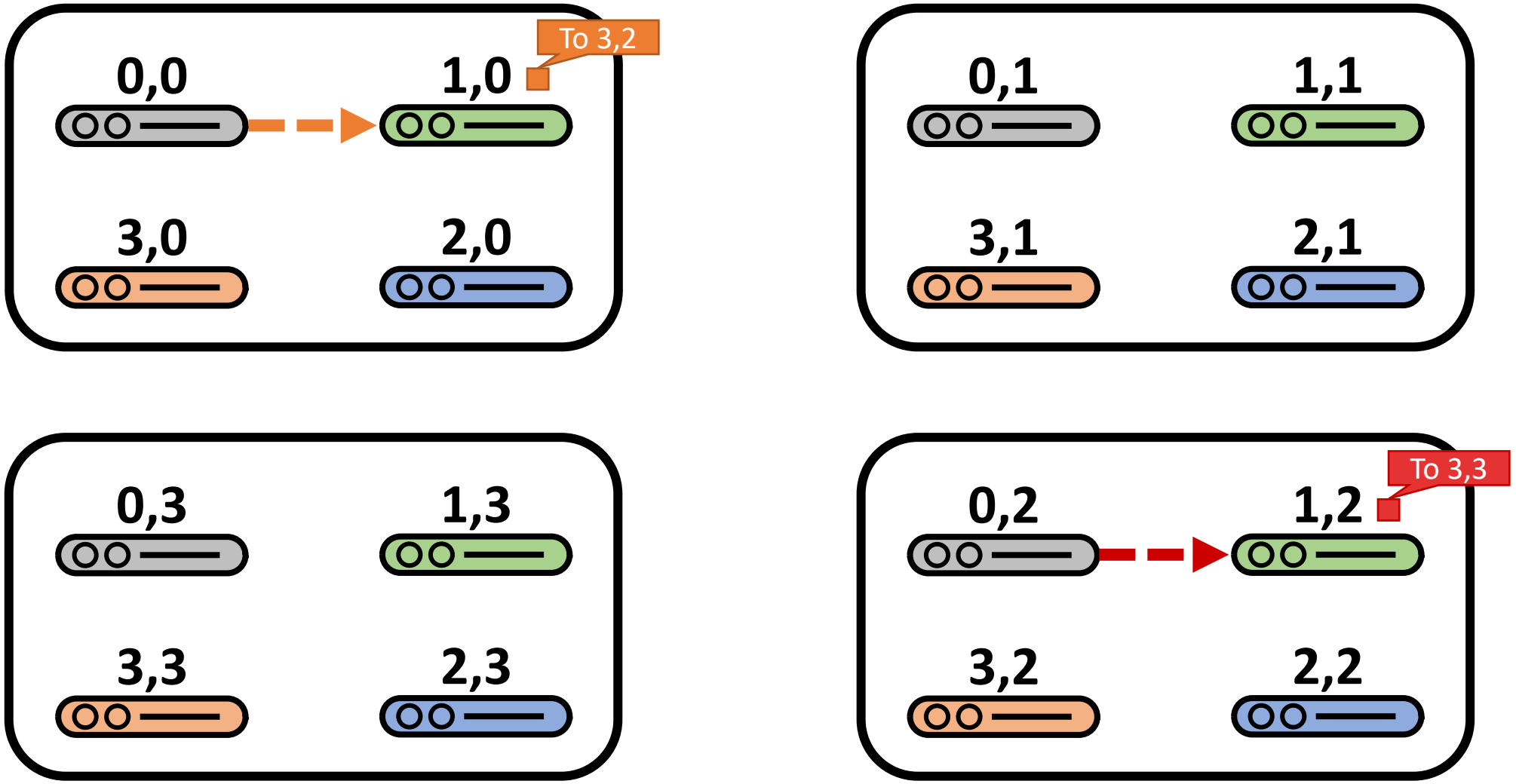
# Queuing in Shale



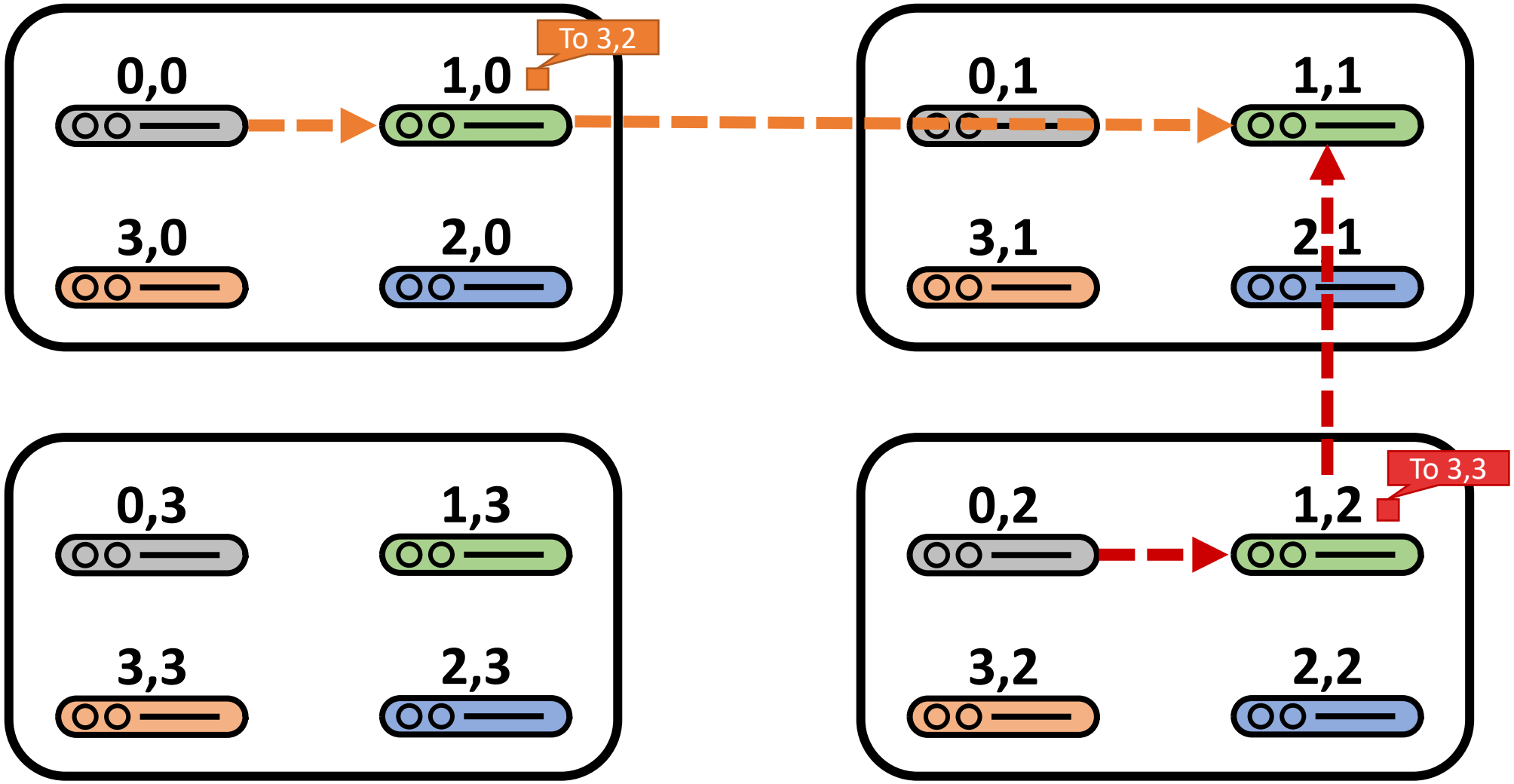
# Queuing in Shale



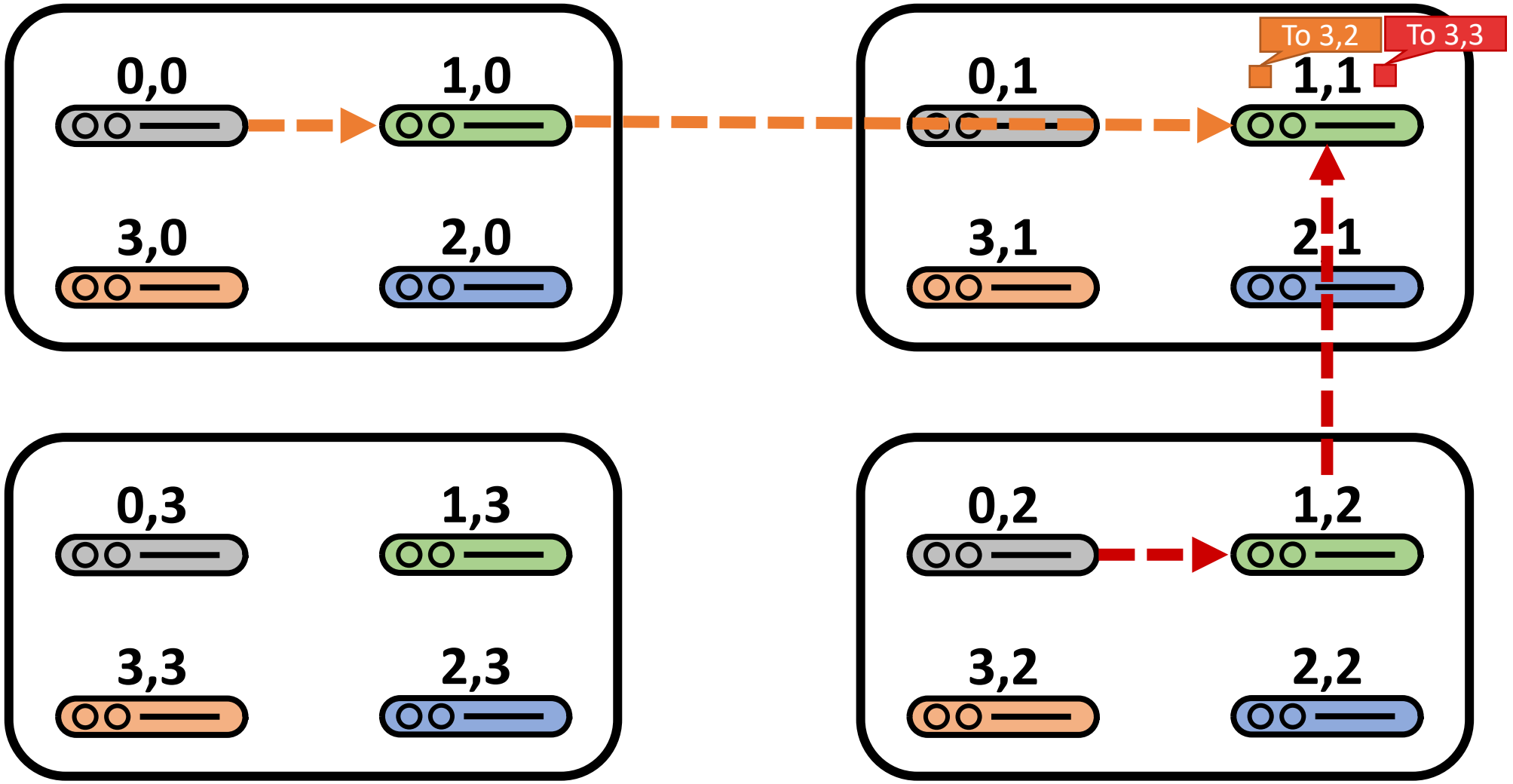
# Queuing in Shale



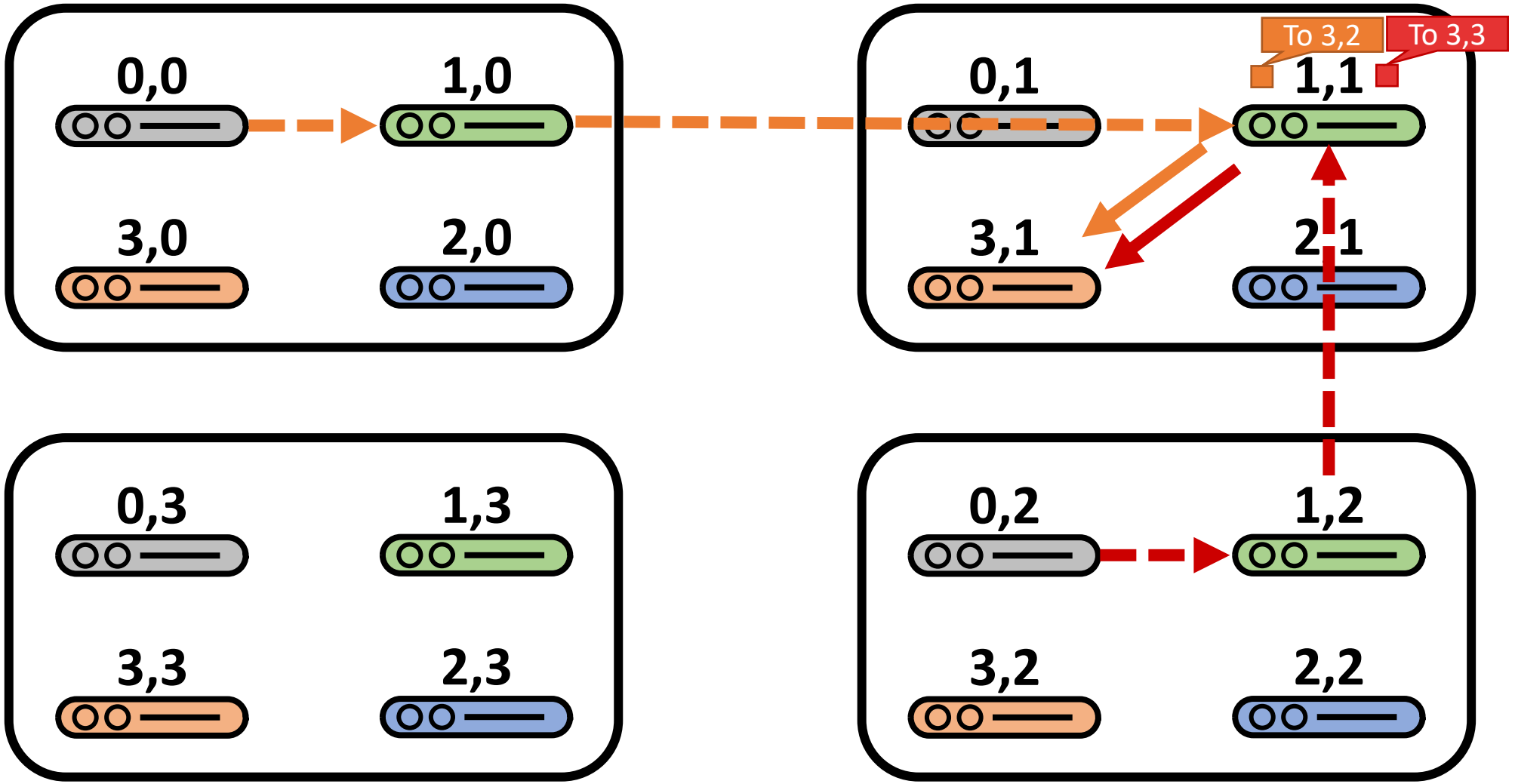
# Queuing in Shale



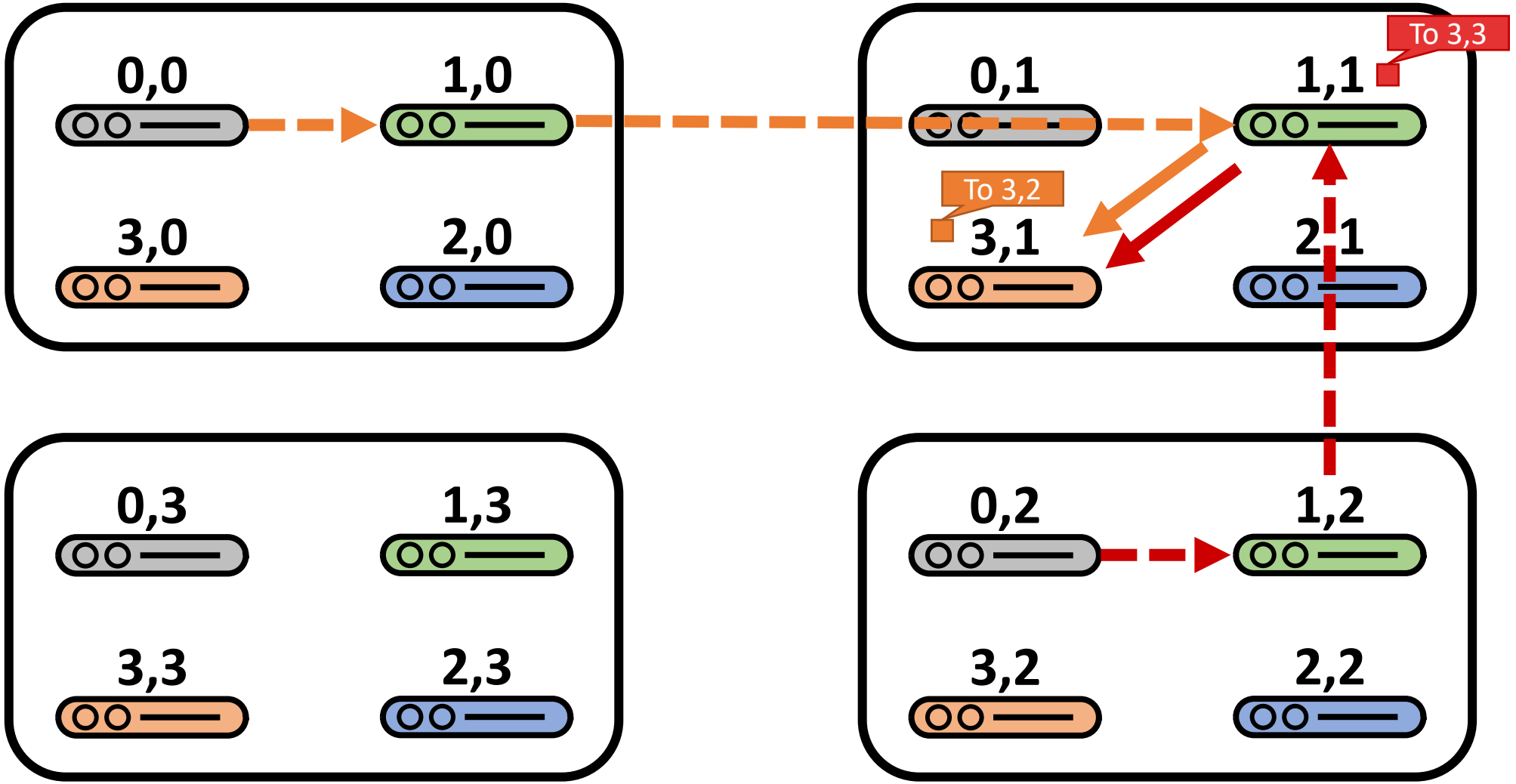
# Queuing in Shale



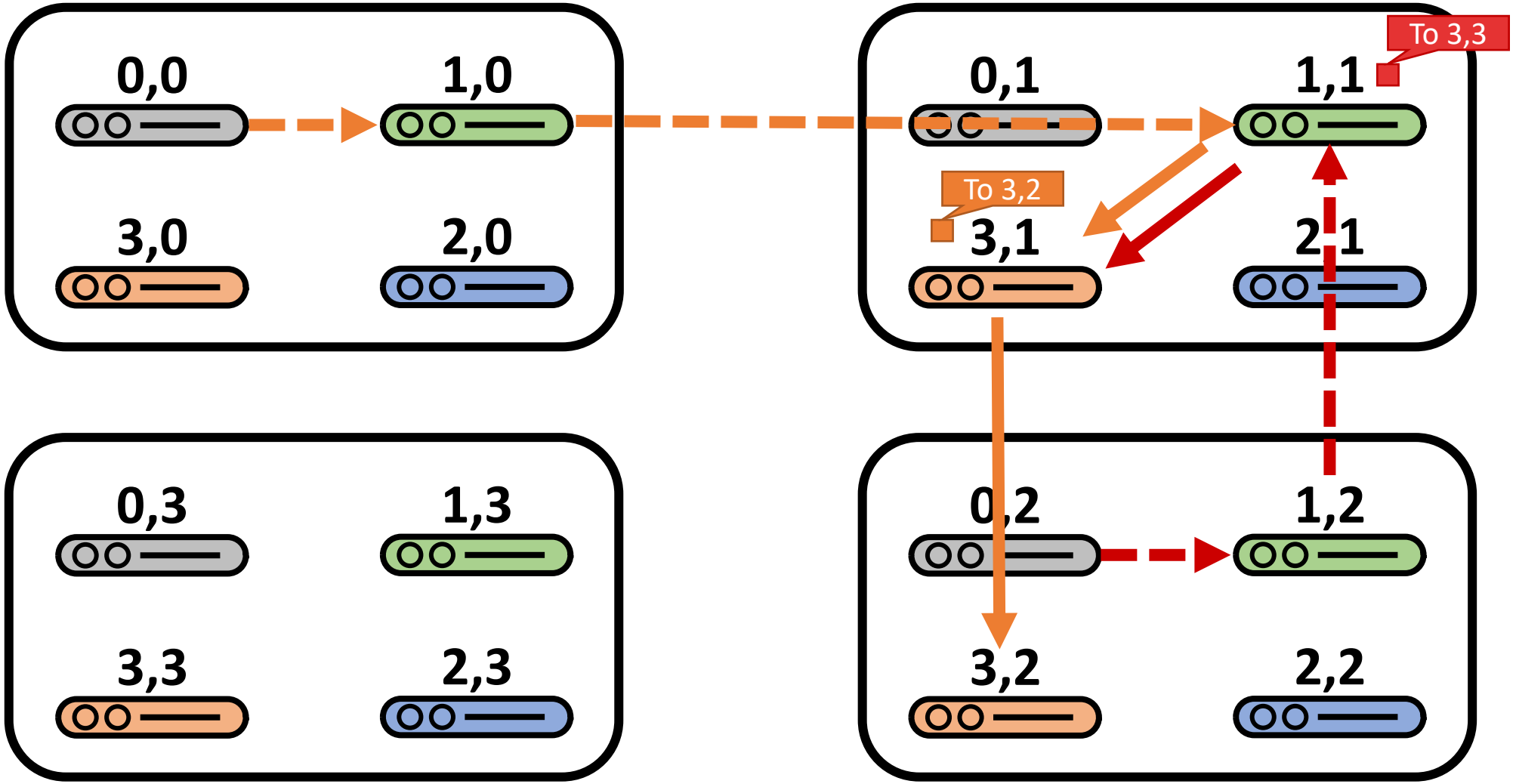
# Queuing in Shale



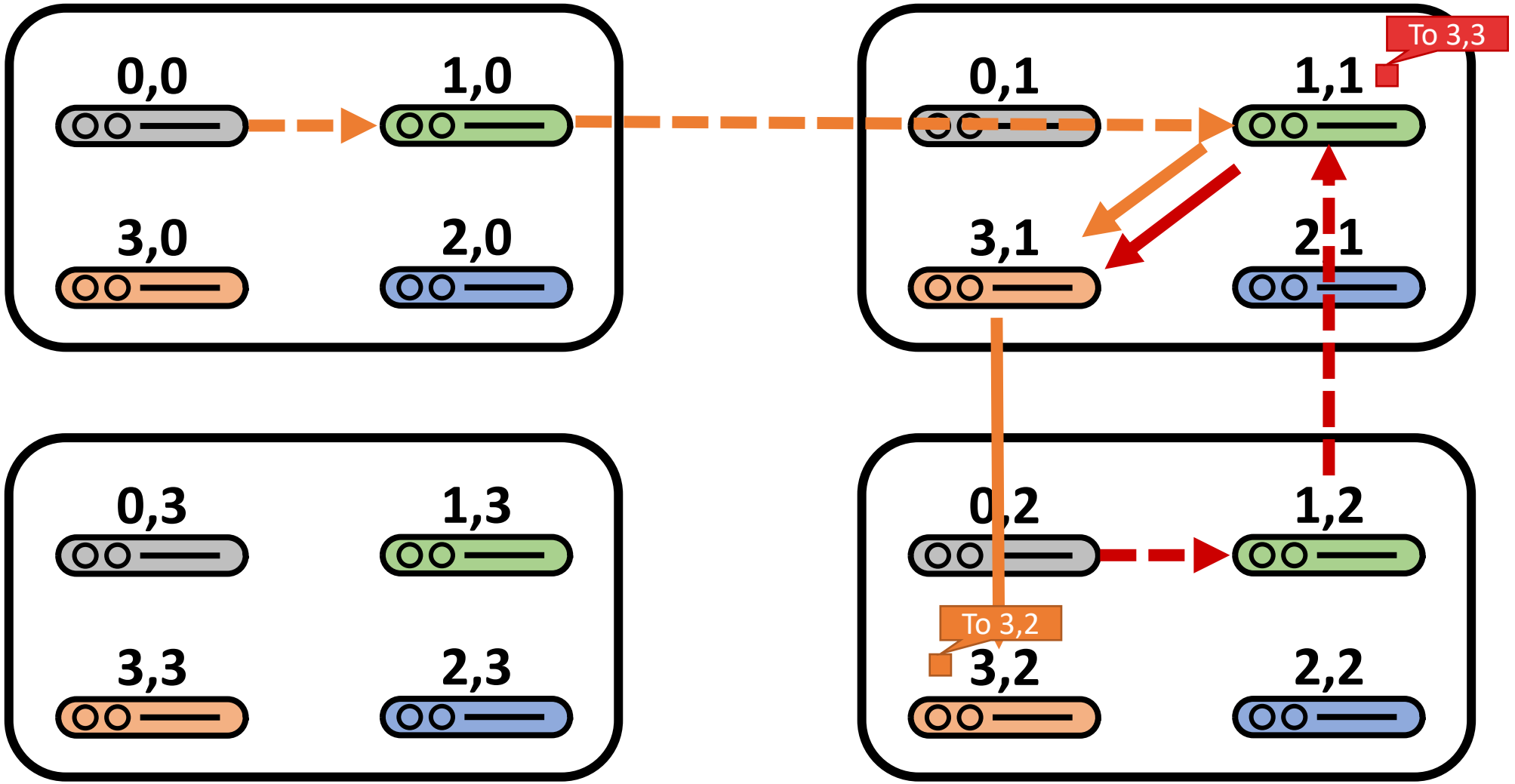
# Queuing in Shale



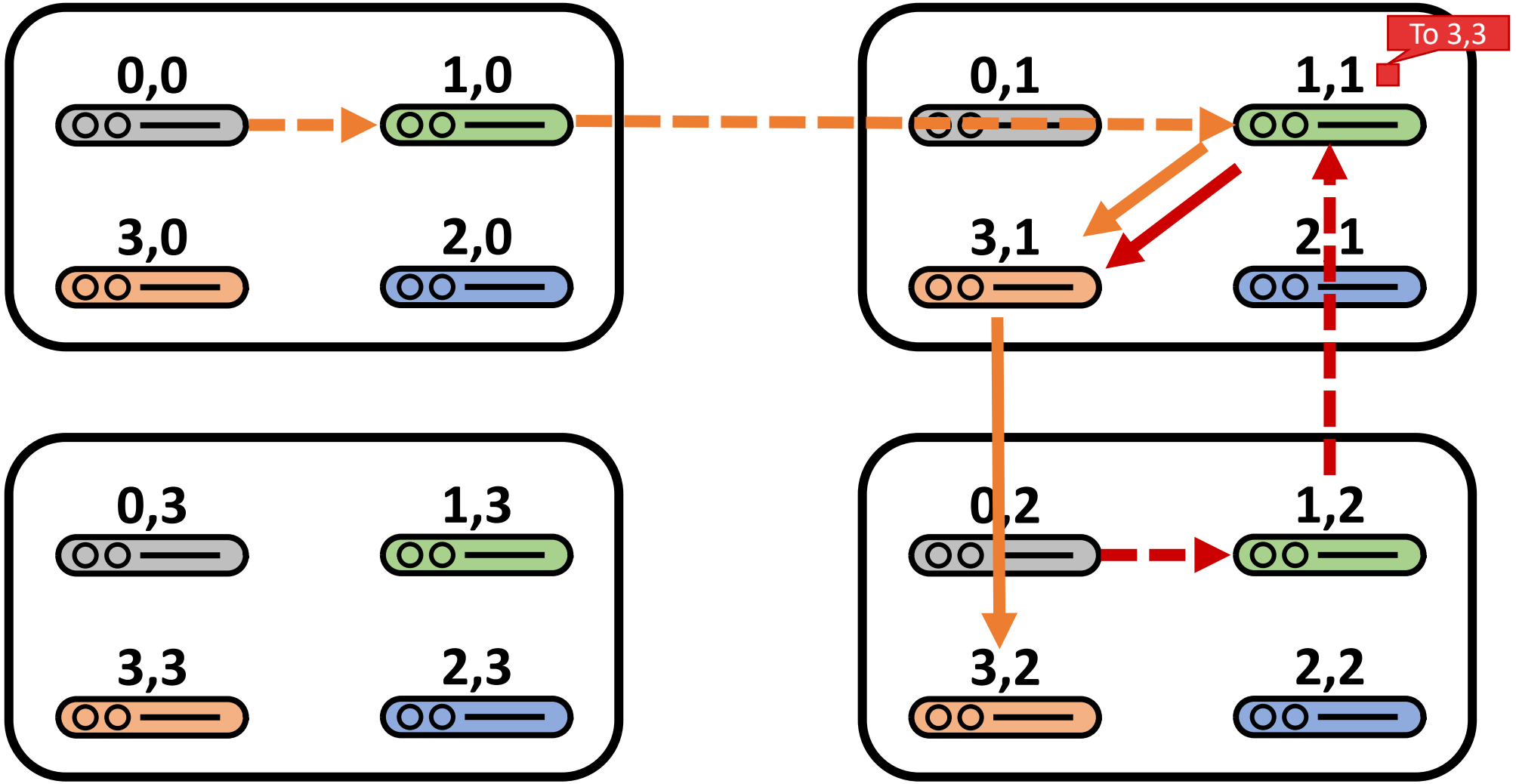
# Queuing in Shale



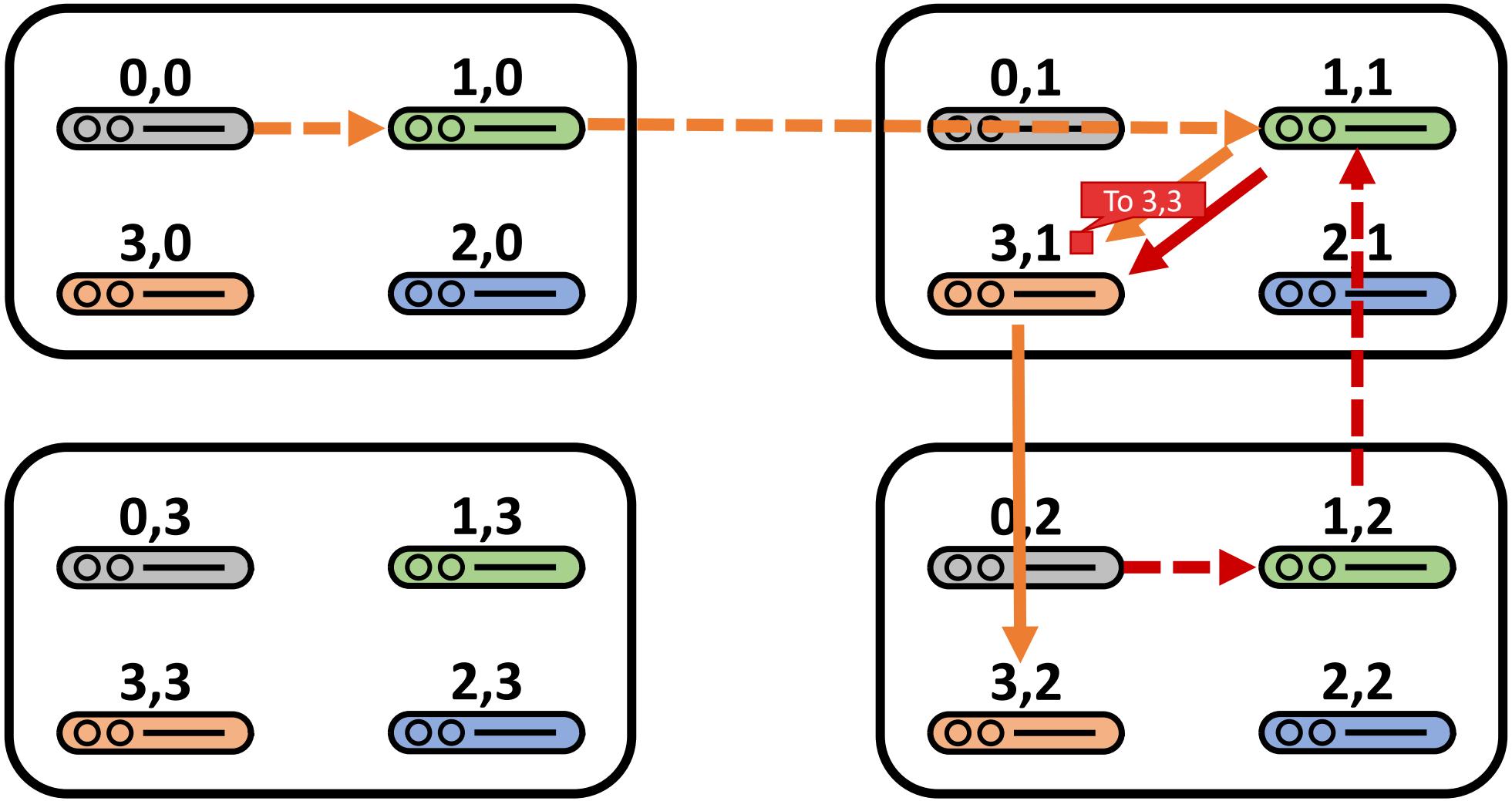
# Queuing in Shale



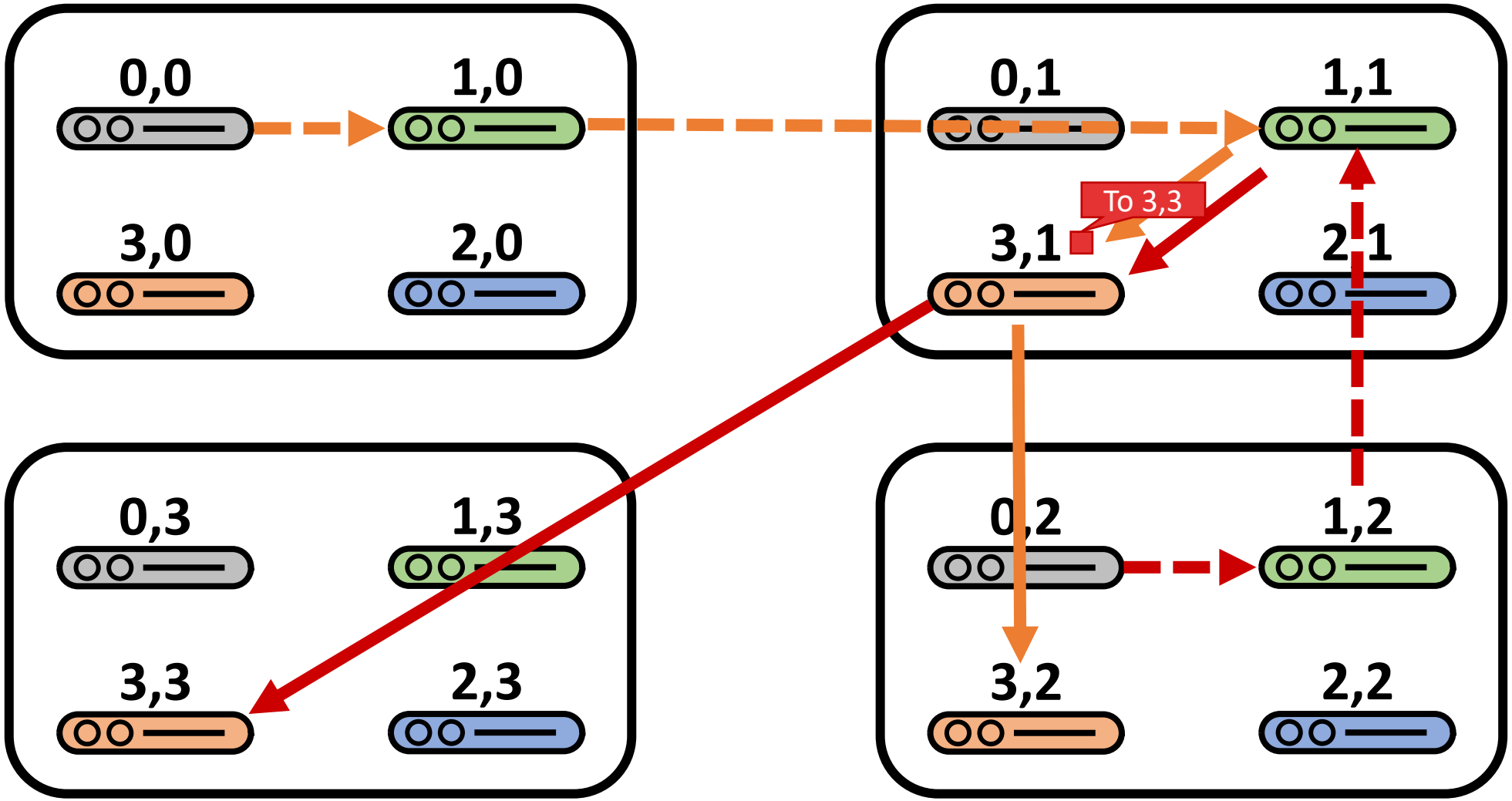
# Queuing in Shale



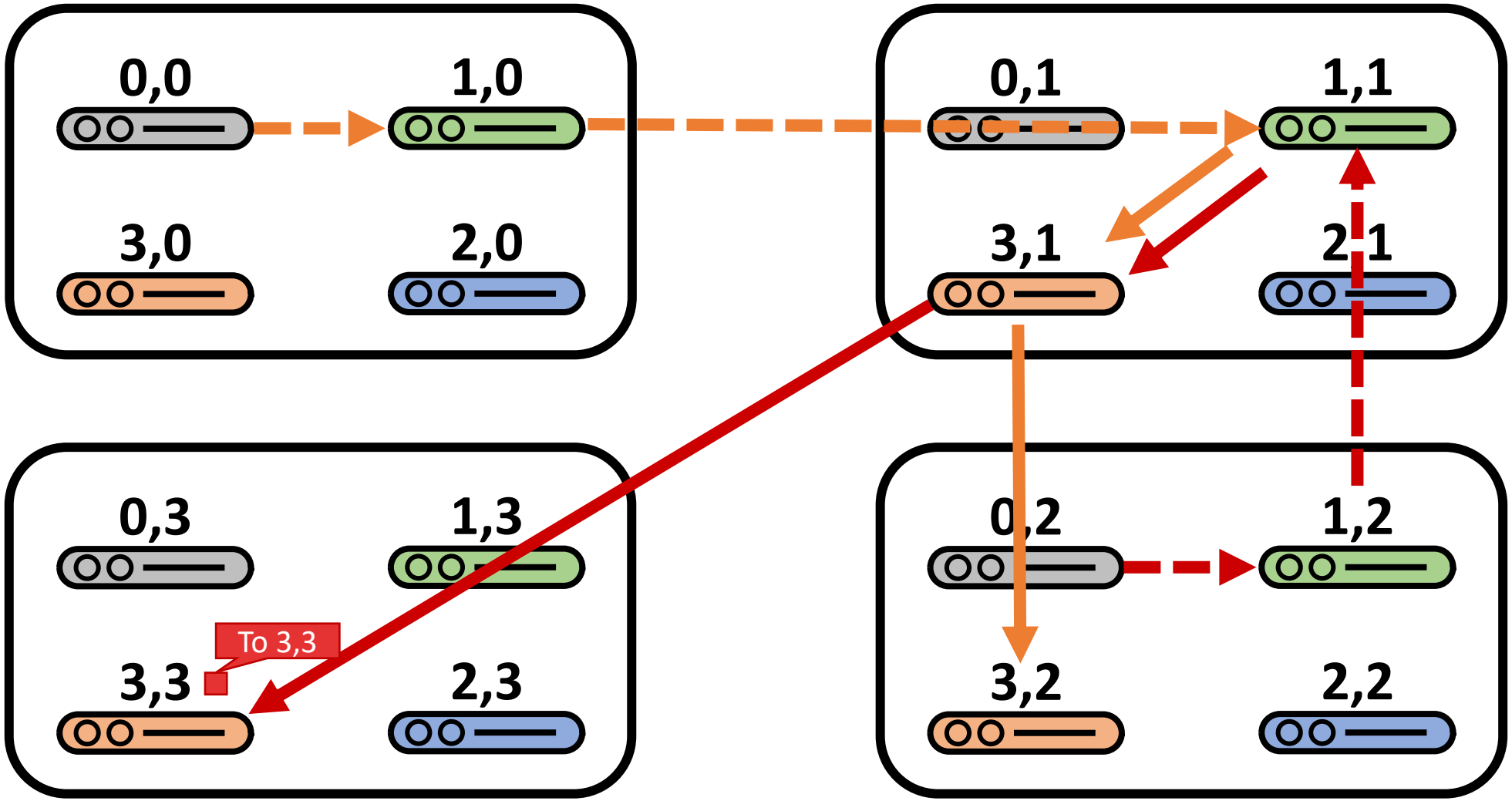
# Queuing in Shale



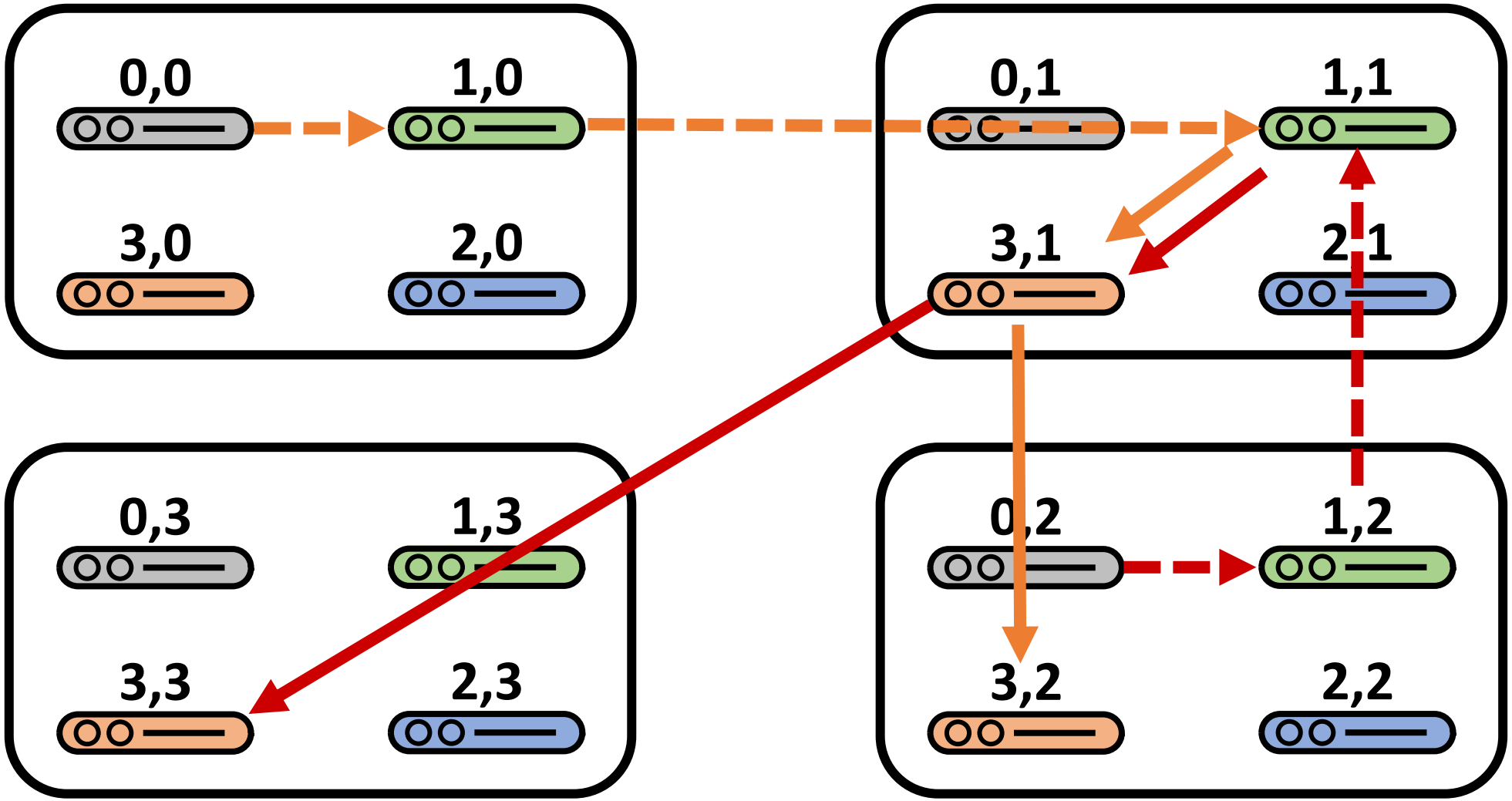
# Queuing in Shale



# Queuing in Shale



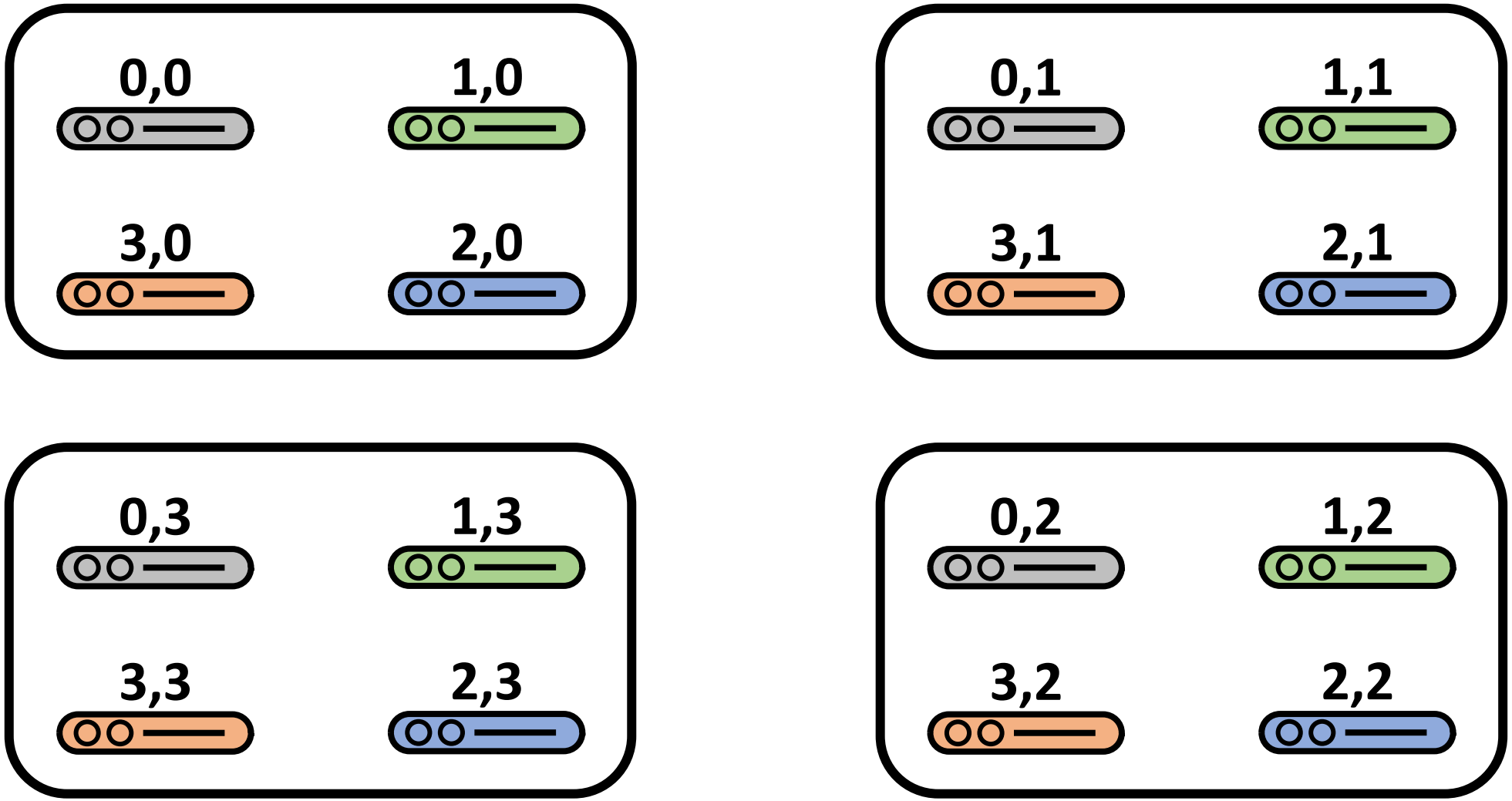
# Queuing in Shale



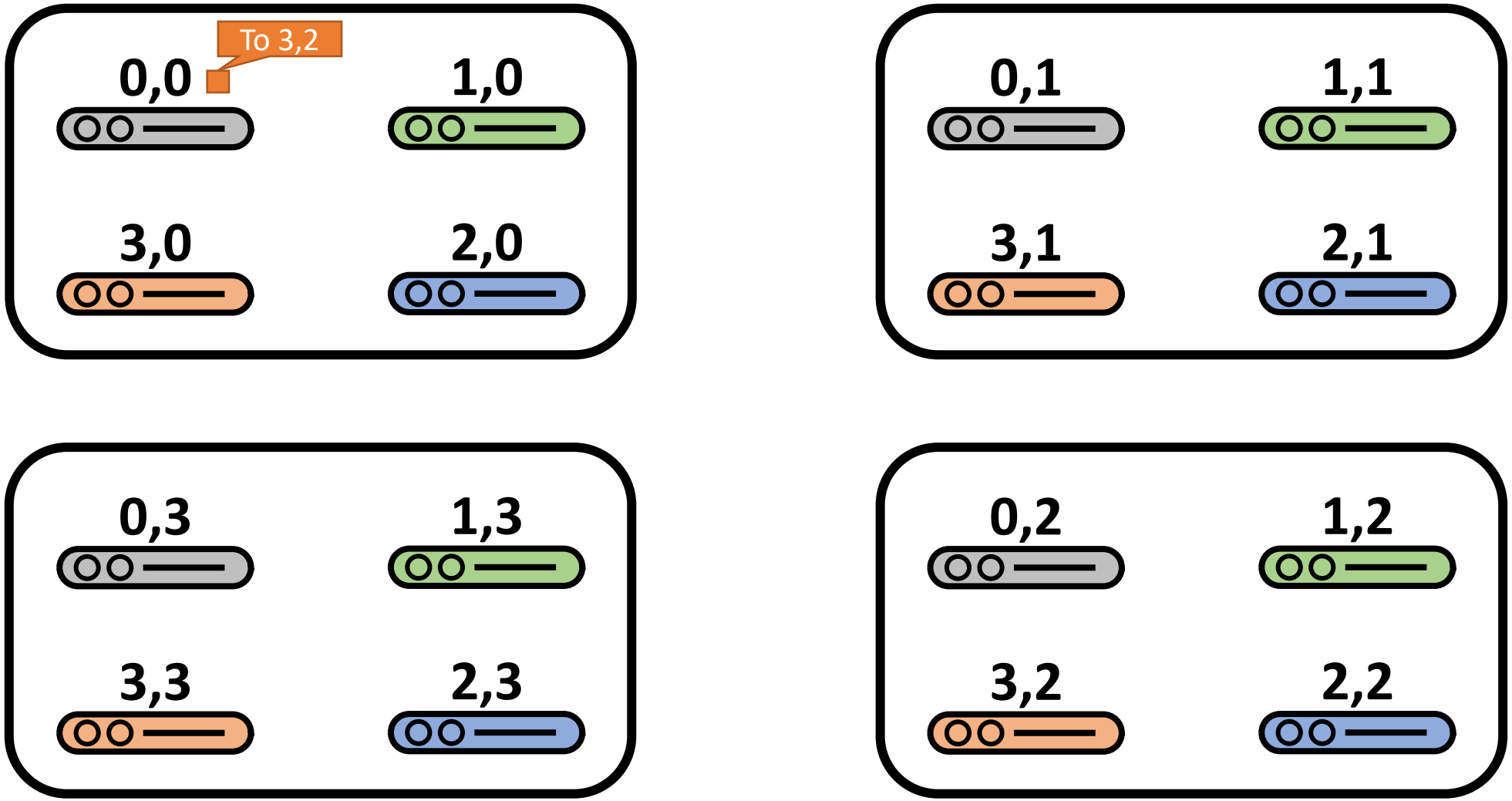
# Queuing in Shale

- 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

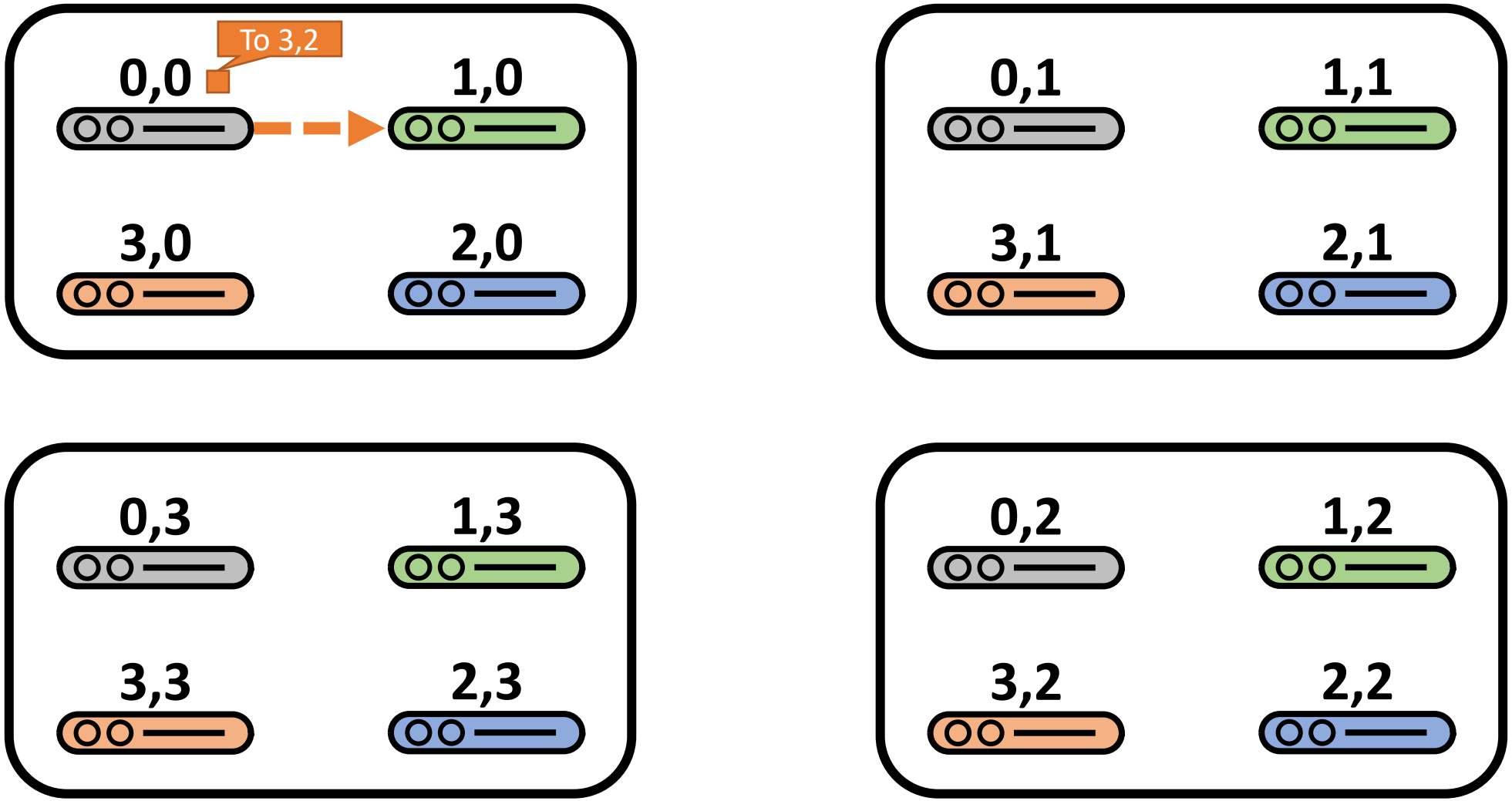
# Addressing path collisions: spray-short



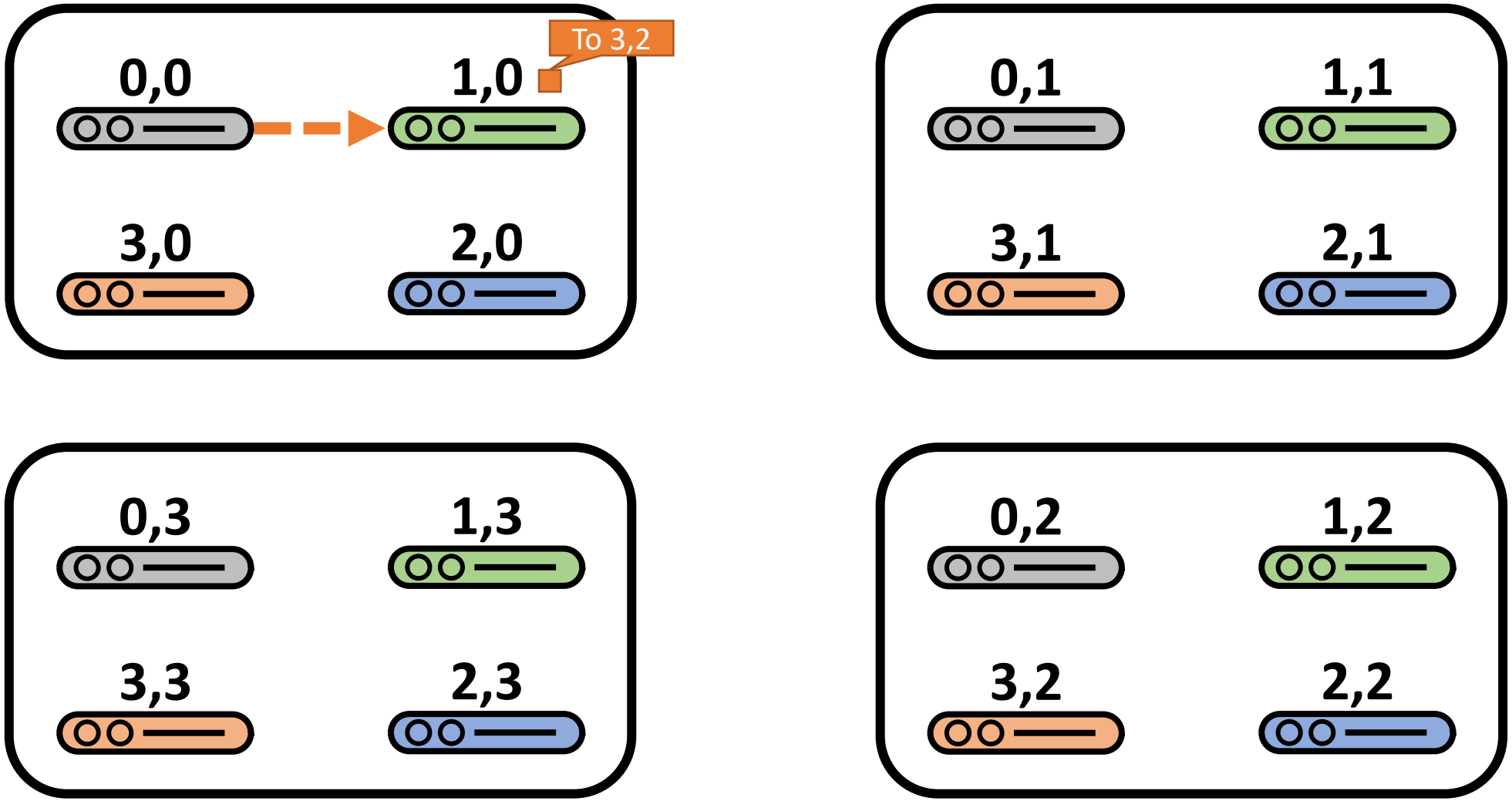
# Addressing path collisions: spray-short



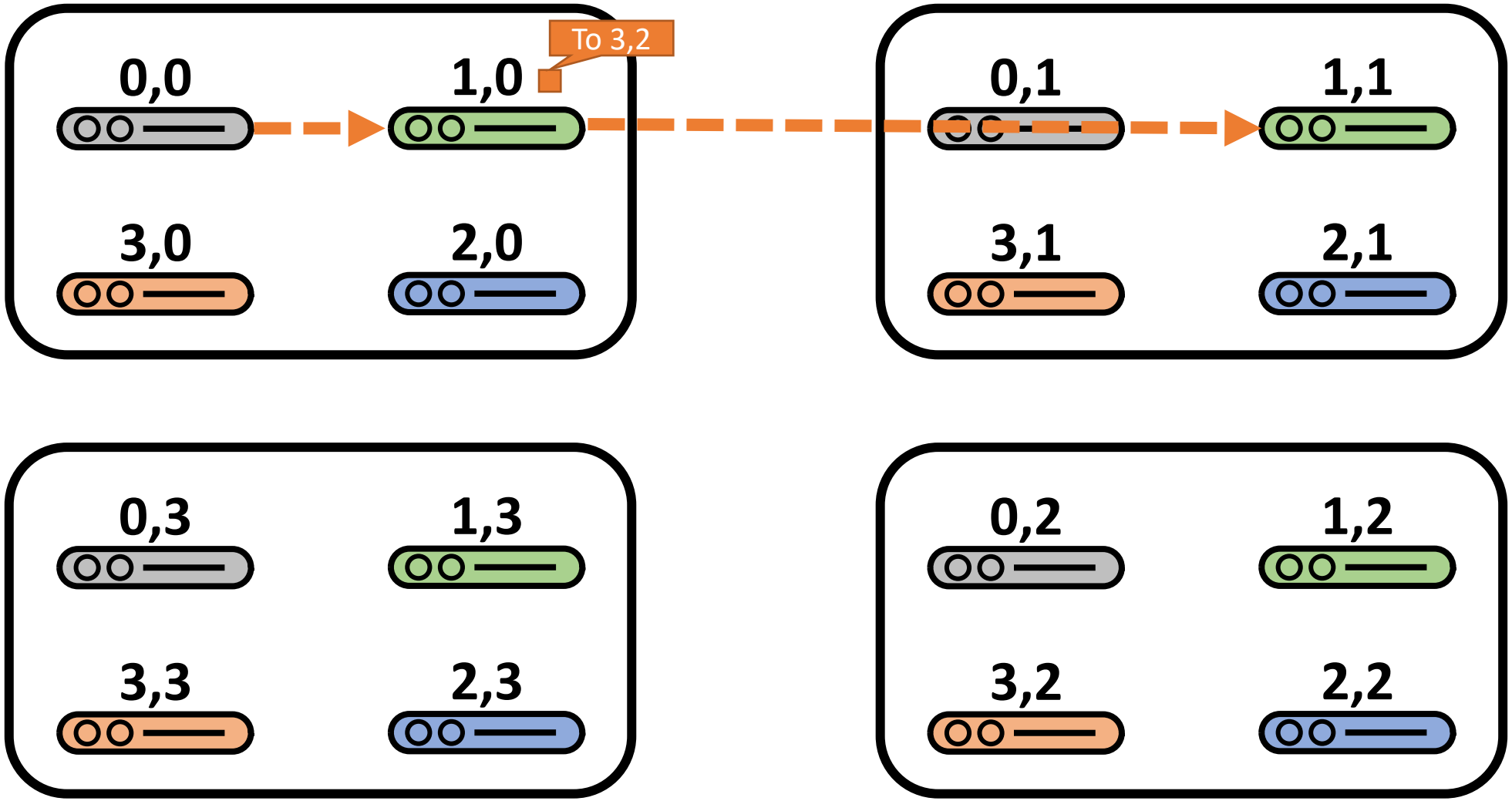
# Addressing path collisions: spray-short



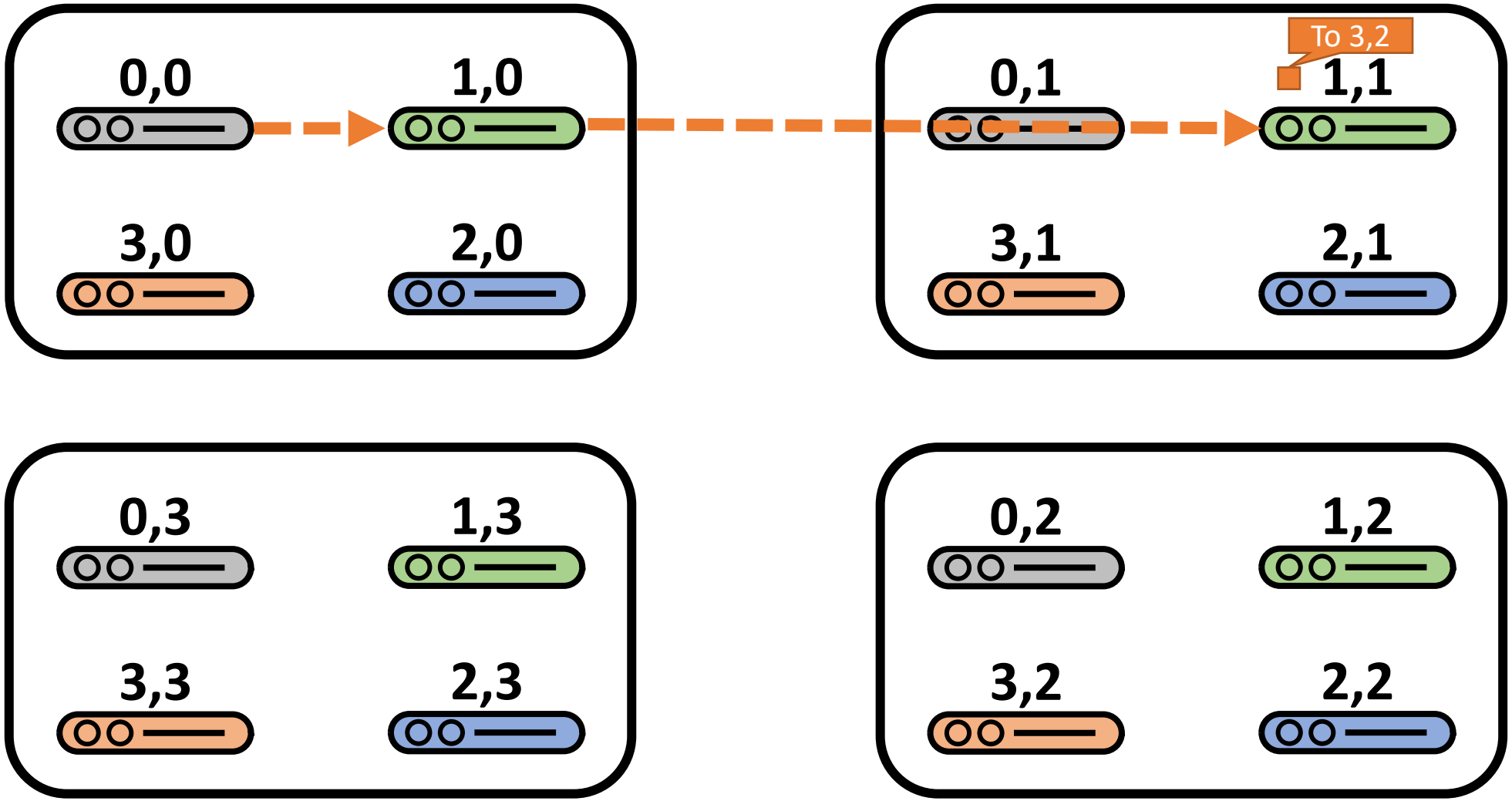
# Addressing path collisions: spray-short



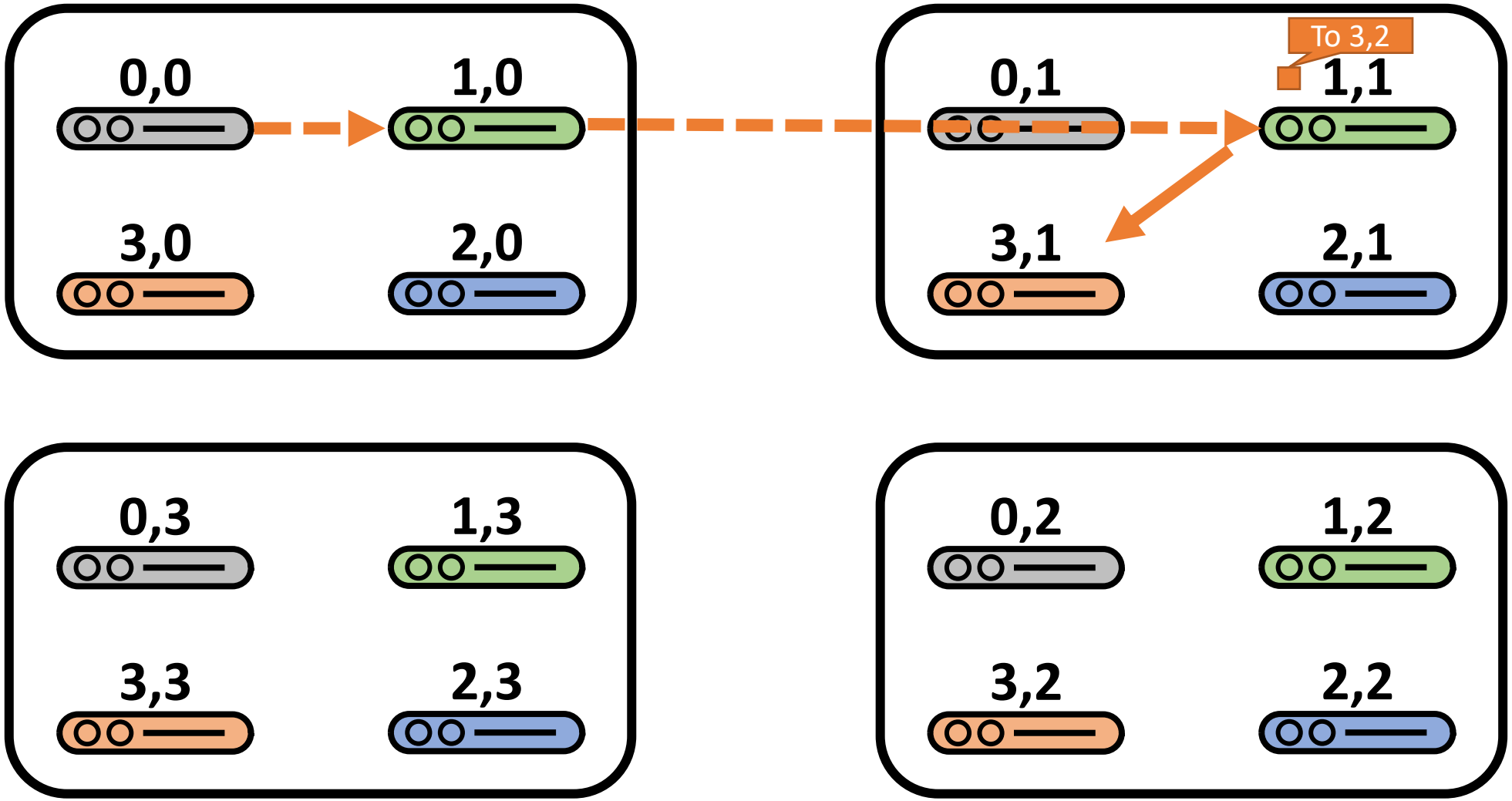
# Addressing path collisions: spray-short



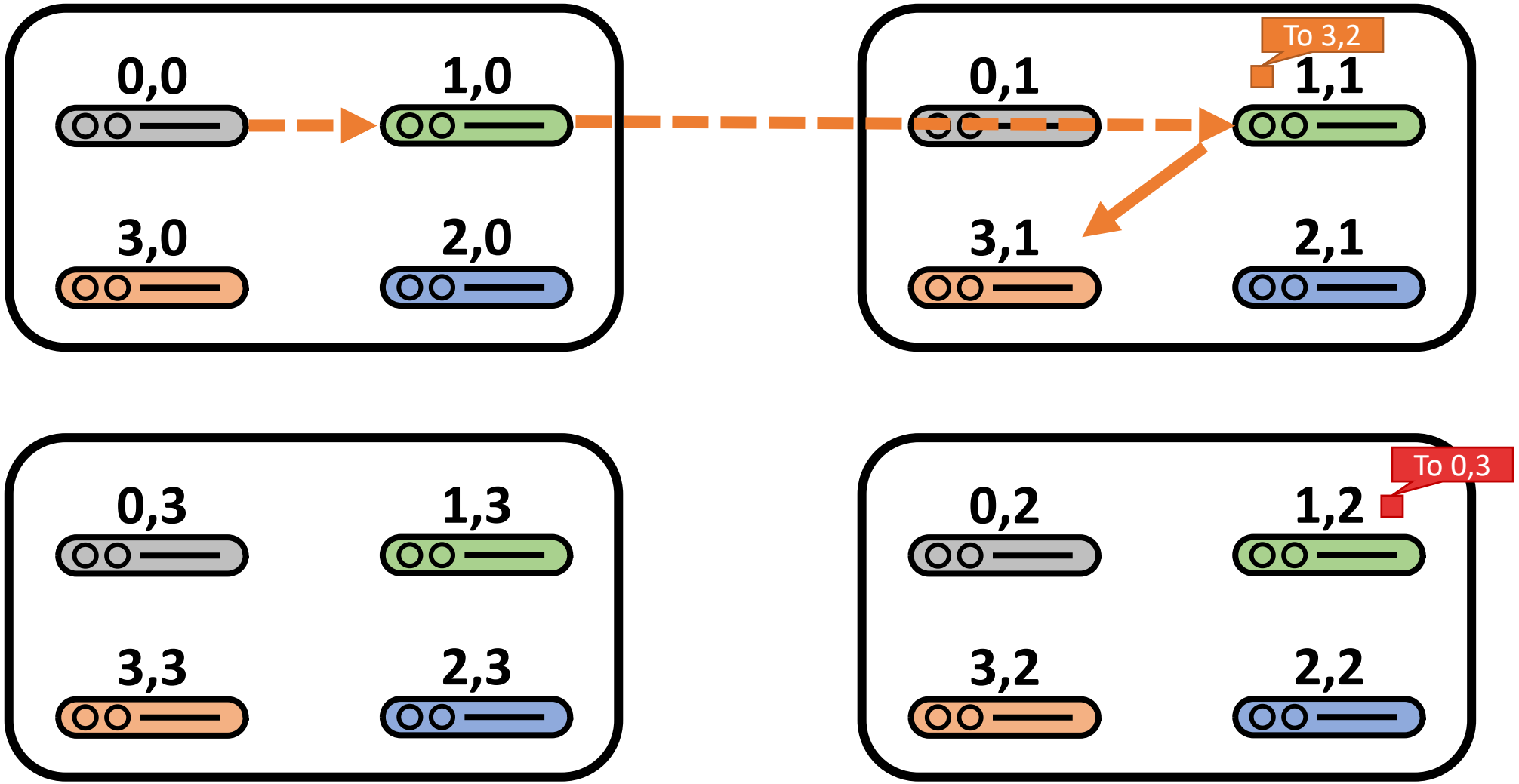
# Addressing path collisions: spray-short



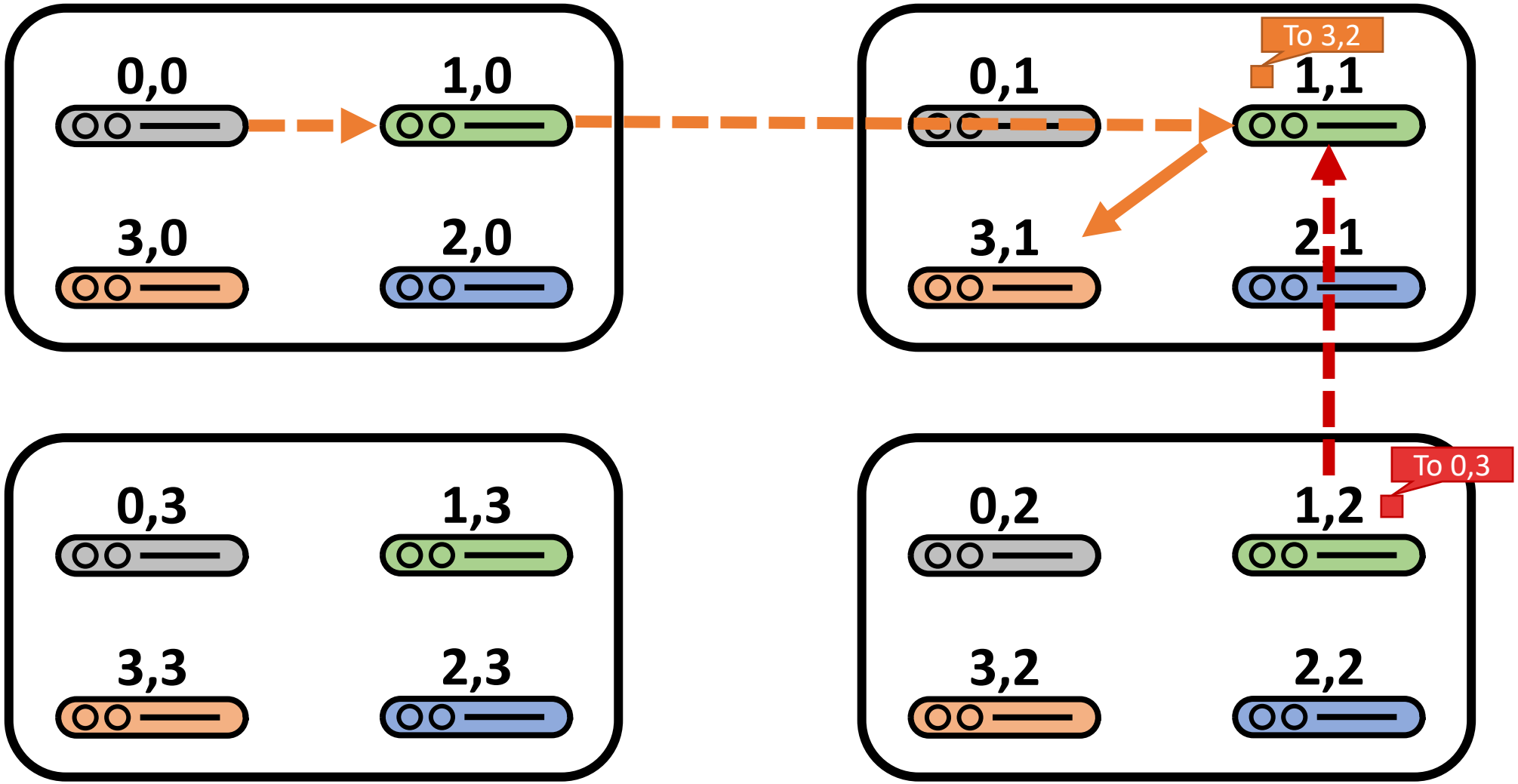
# Addressing path collisions: spray-short



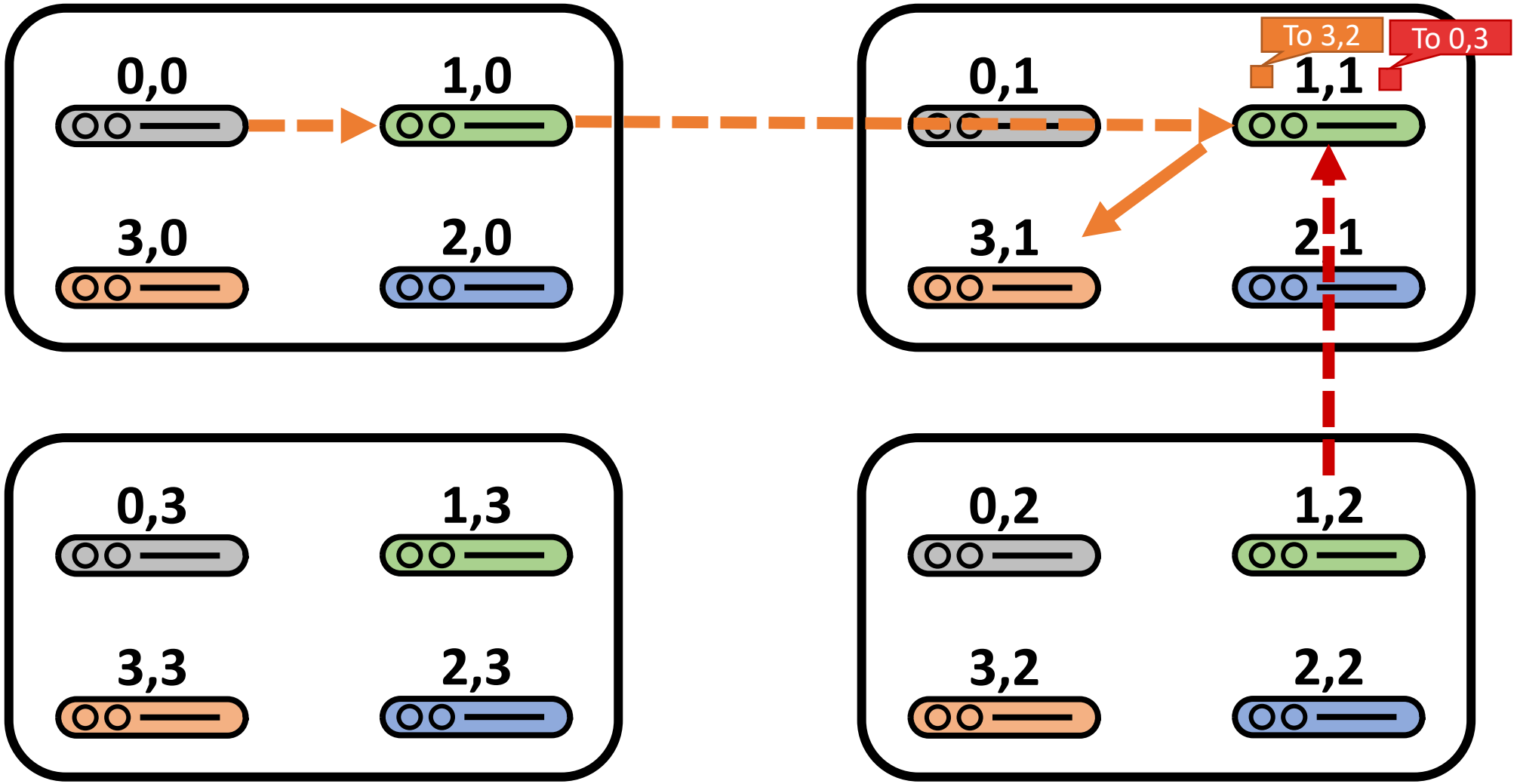
# Addressing path collisions: spray-short



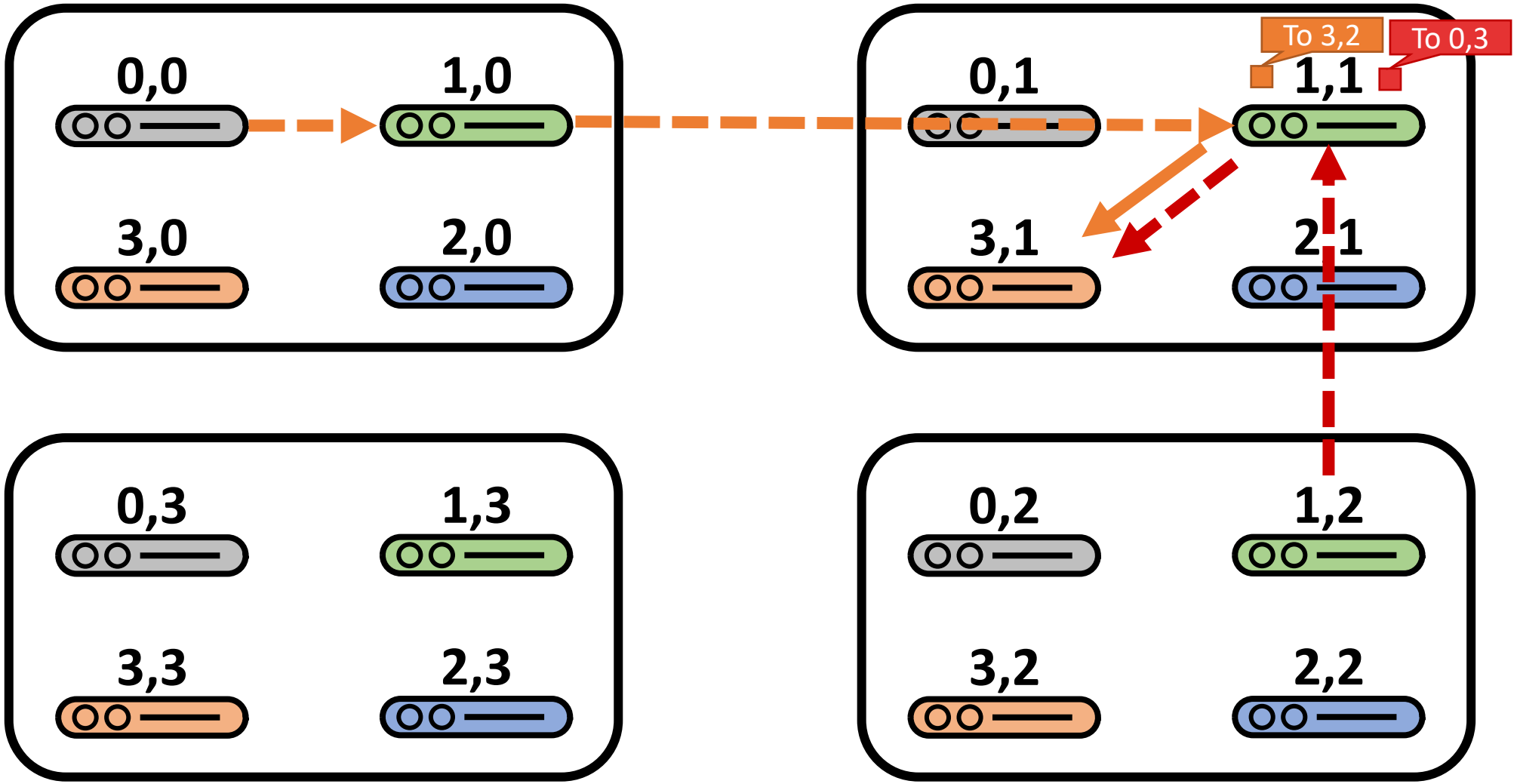
# Addressing path collisions: spray-short



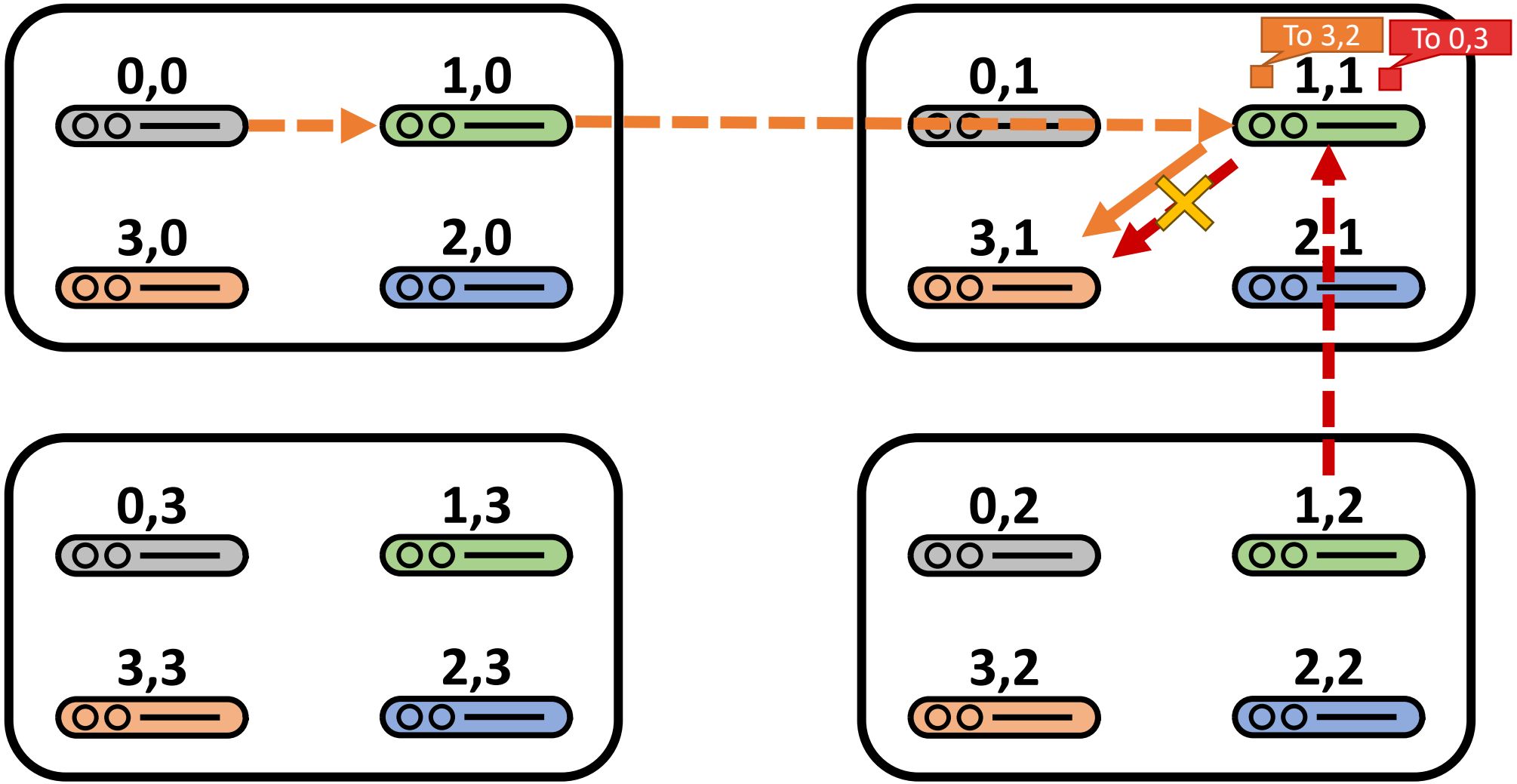
# Addressing path collisions: spray-short



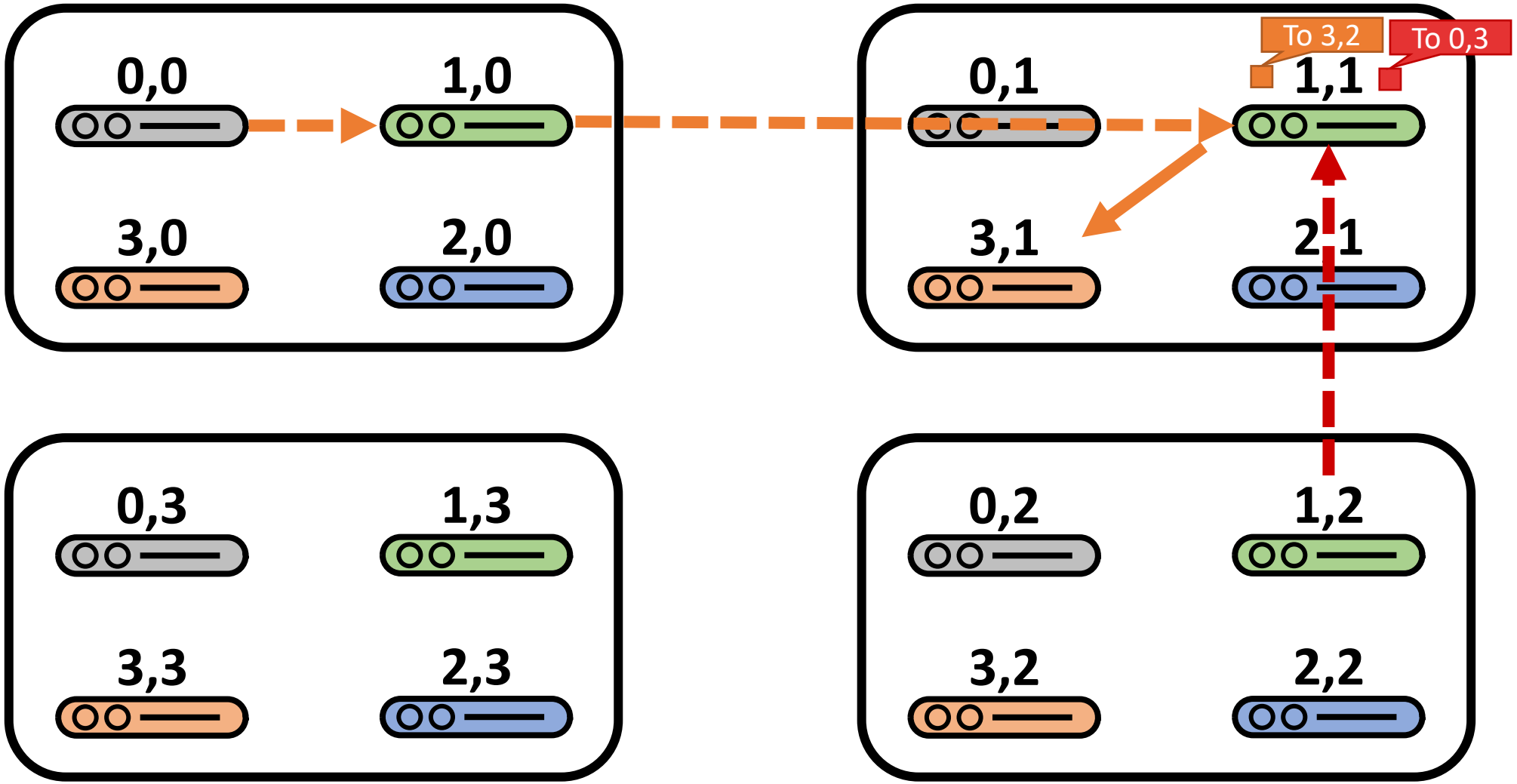
# Addressing path collisions: spray-short



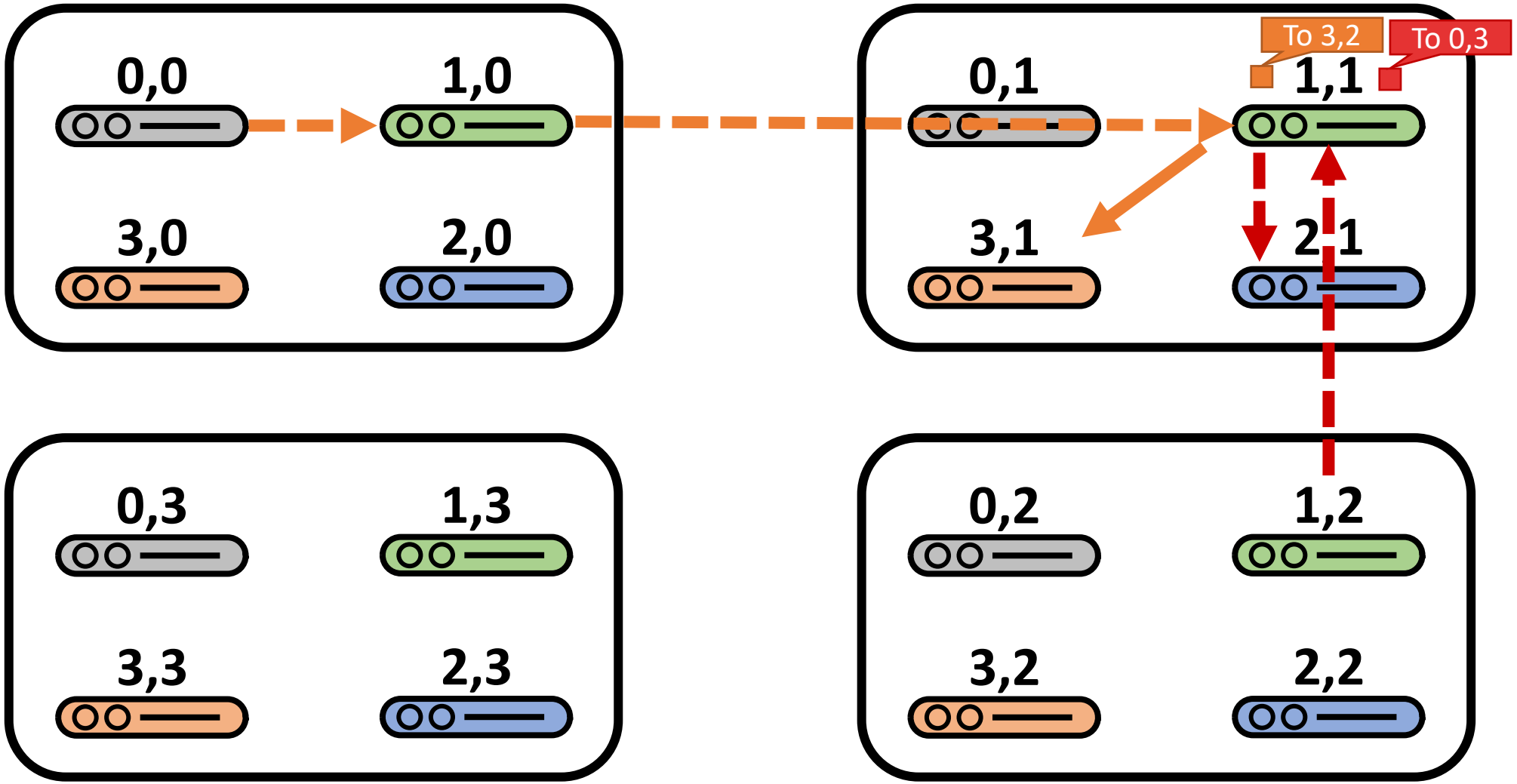
# Addressing path collisions: spray-short



# Addressing path collisions: spray-short



# Addressing path collisions: spray-short

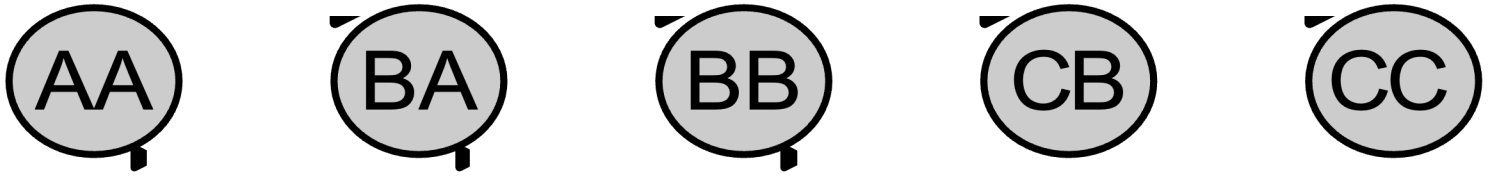


# Addressing egress congestion: hop-by-hop

- 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

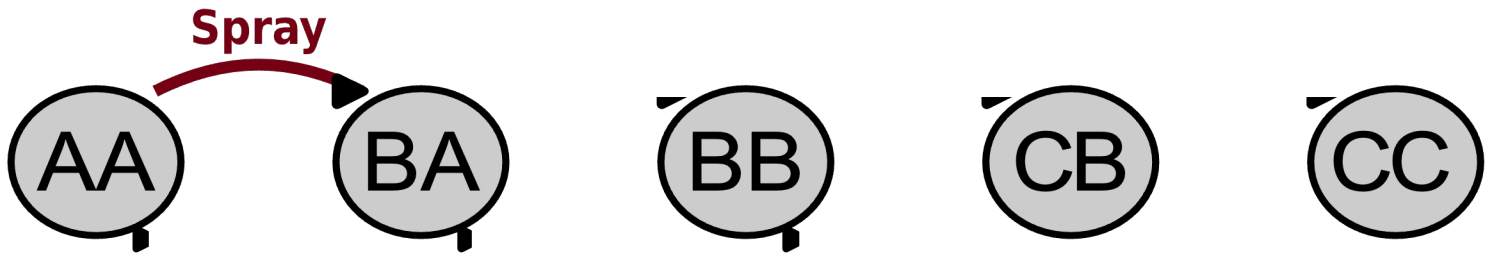
# Addressing egress congestion: hop-by-hop

- 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



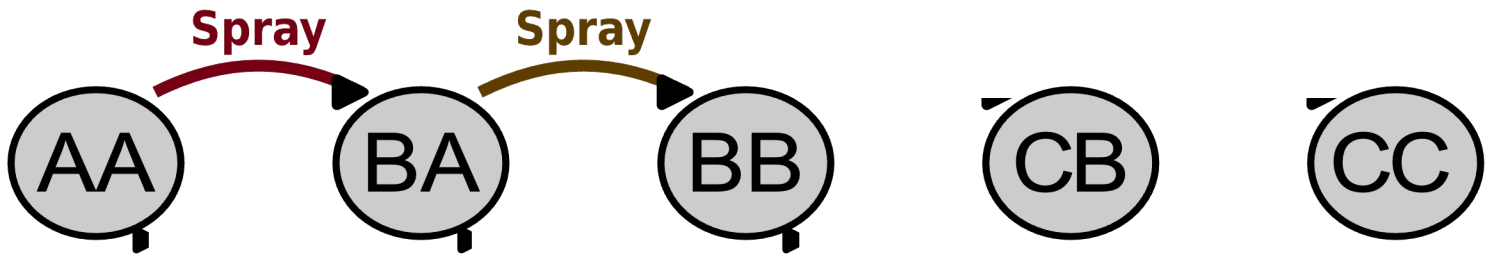
# Addressing egress congestion: hop-by-hop

- 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



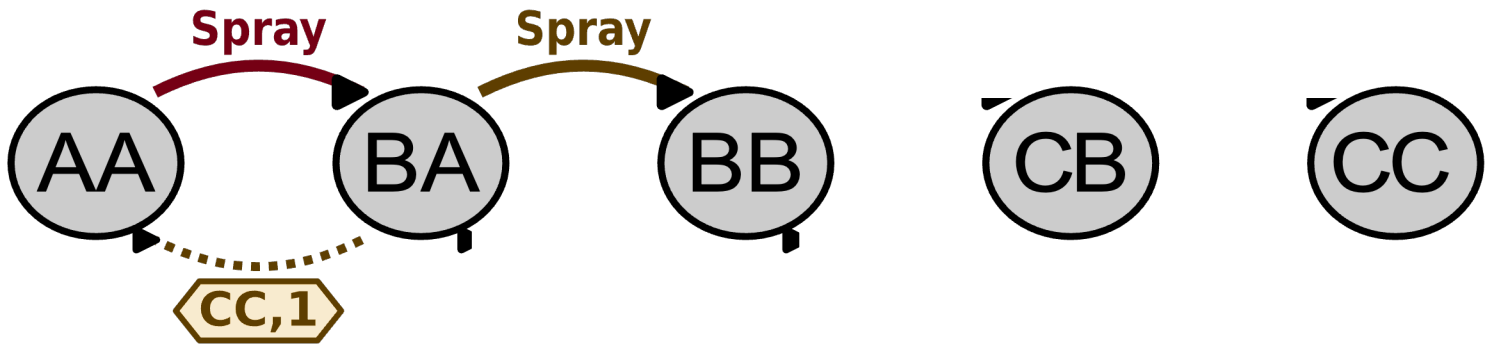
# Addressing egress congestion: hop-by-hop

- 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



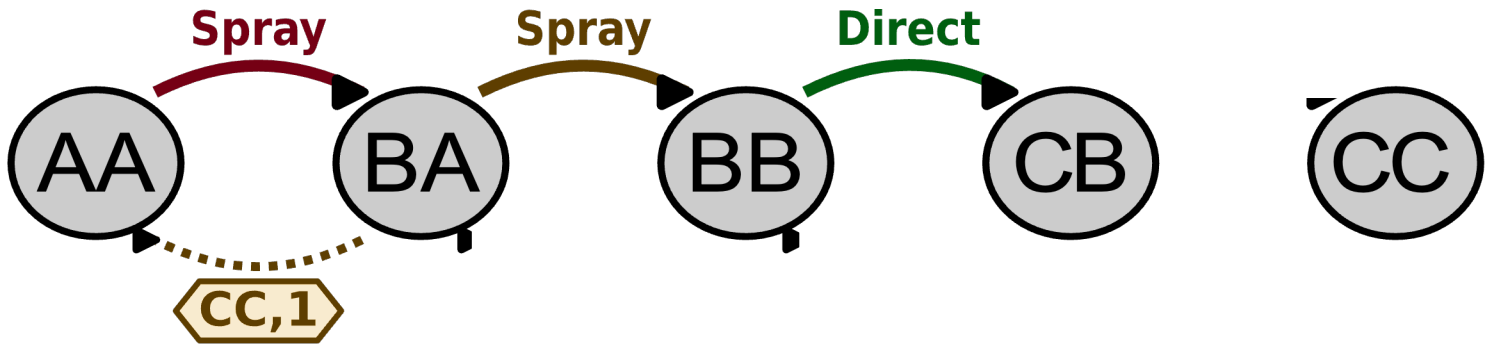
# Addressing egress congestion: hop-by-hop

- 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



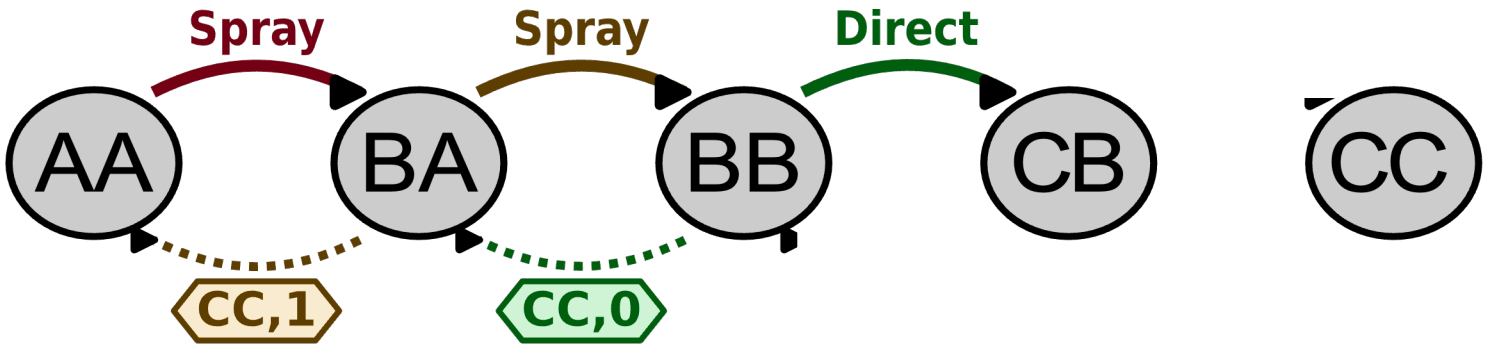
# Addressing egress congestion: hop-by-hop

- 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



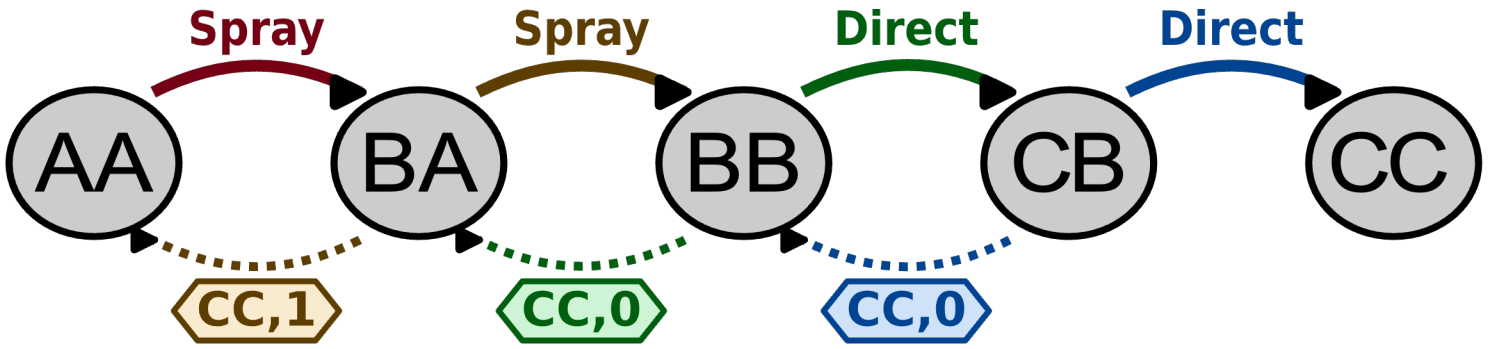
# Addressing egress congestion: hop-by-hop

- 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



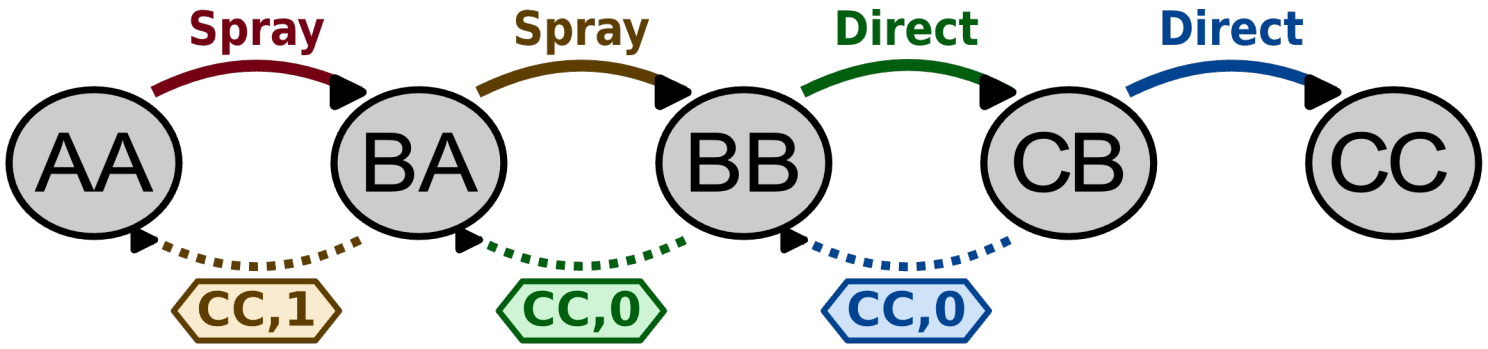
# Addressing egress congestion: hop-by-hop

- 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



# Addressing egress congestion: hop-by-hop

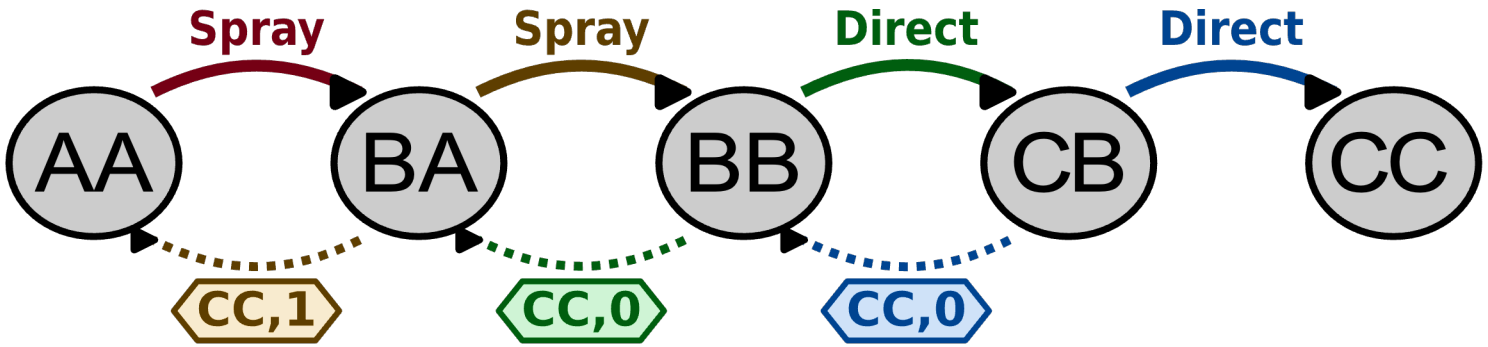
- When a node sends a cell to an intermediate node, it stops sending future cells along that path until it receives a corresponding token
- Once it forwards the cell, the intermediate node also sends back a token




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

# Addressing egress congestion: hop-by-hop

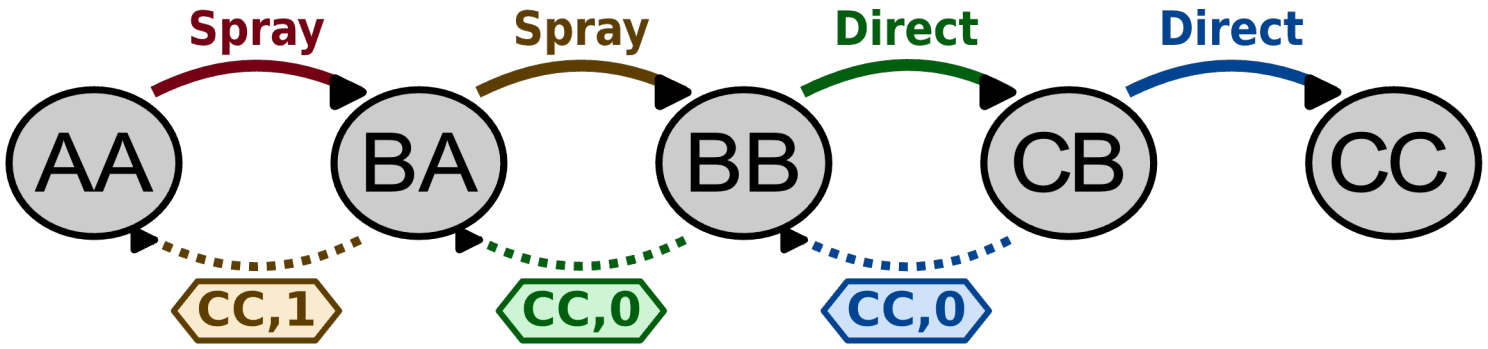
- When a node sends a cell to an intermediate node, it stops sending future cells along that path until it receives a corresponding token
- Once it forwards the cell, the intermediate node also sends back a token




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

# Addressing egress congestion: hop-by-hop

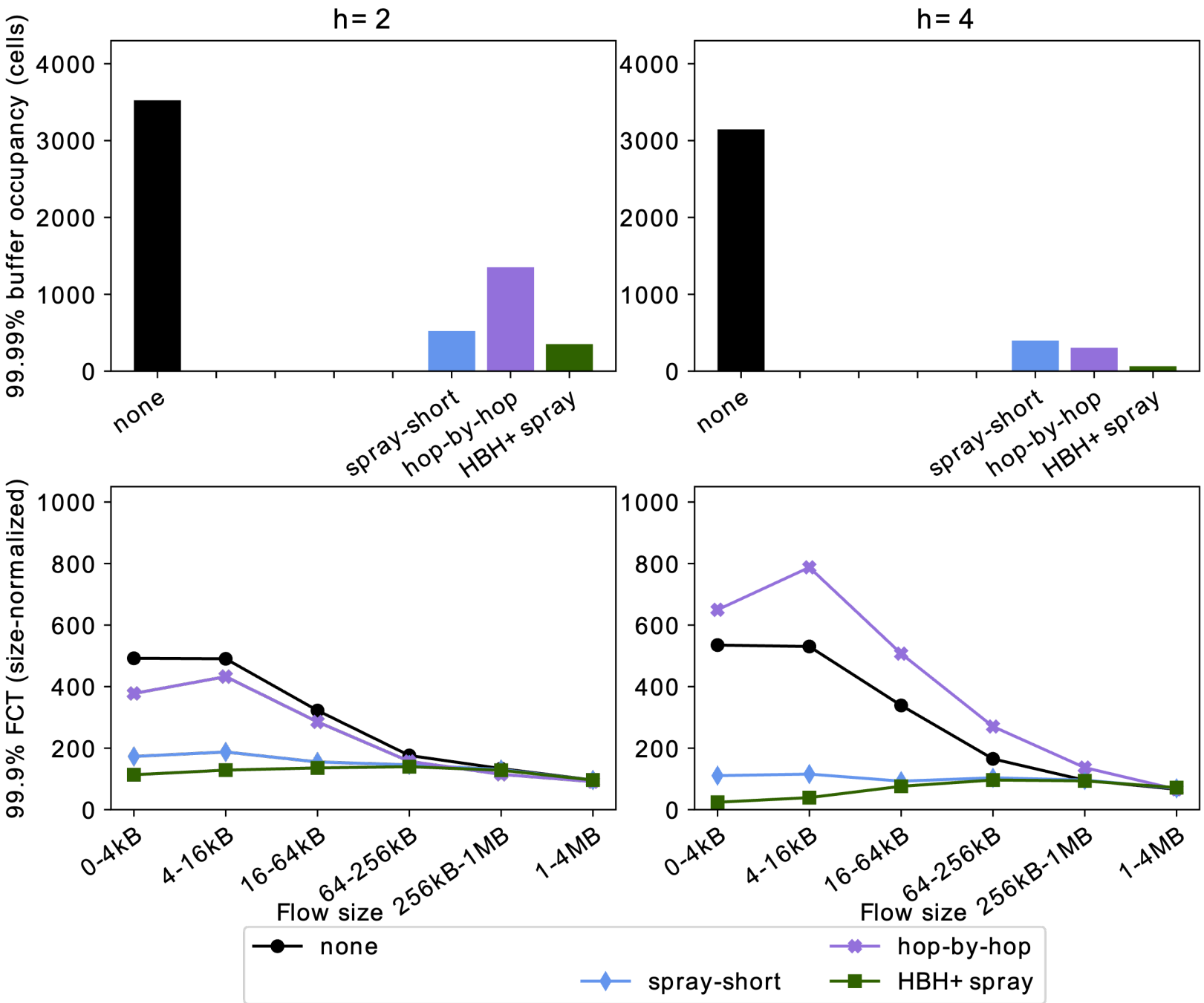
- When a node sends a cell to an intermediate node, it stops sending future cells along that path until it receives a corresponding token
- Once it forwards the cell, the intermediate node also sends back a token

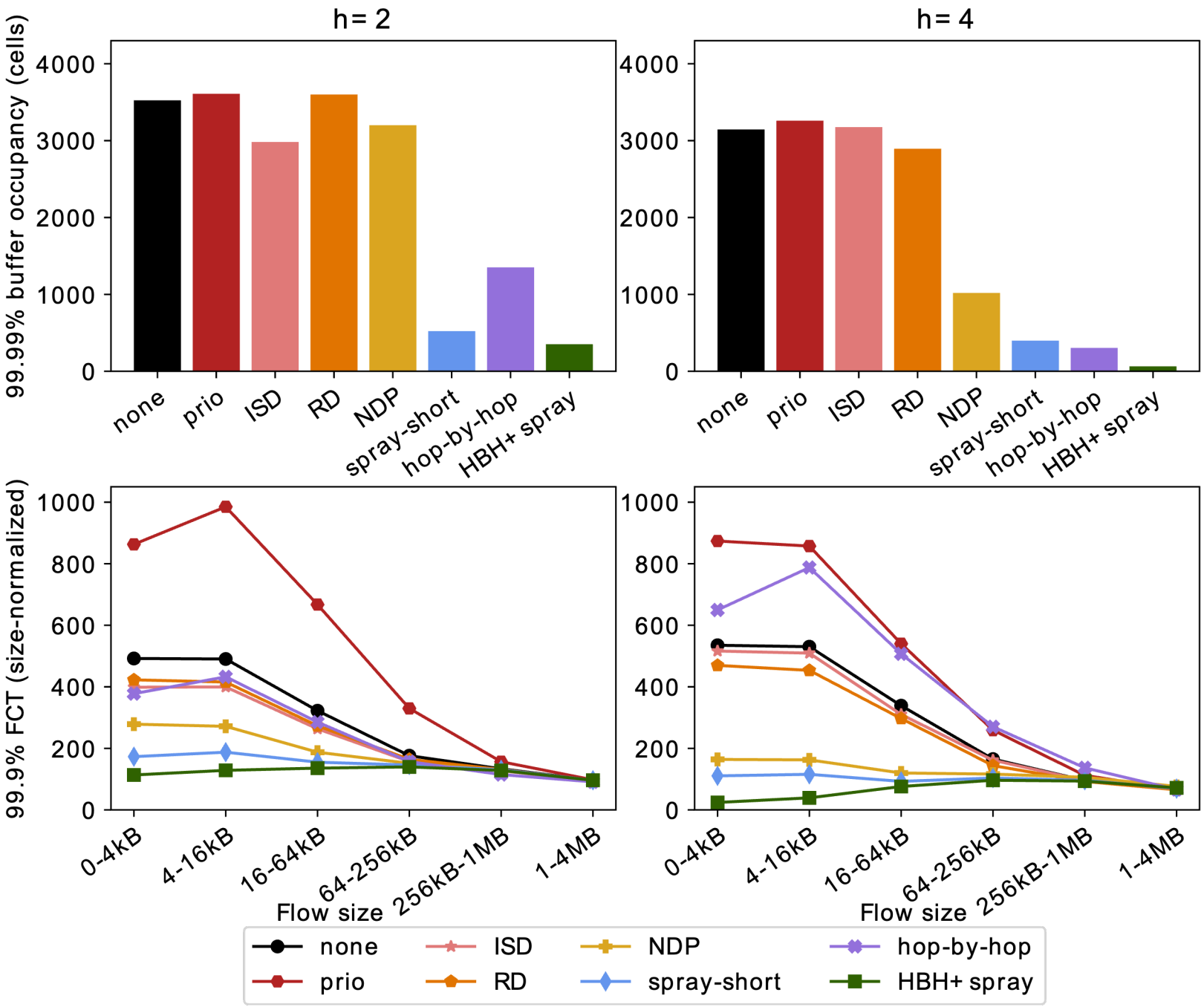


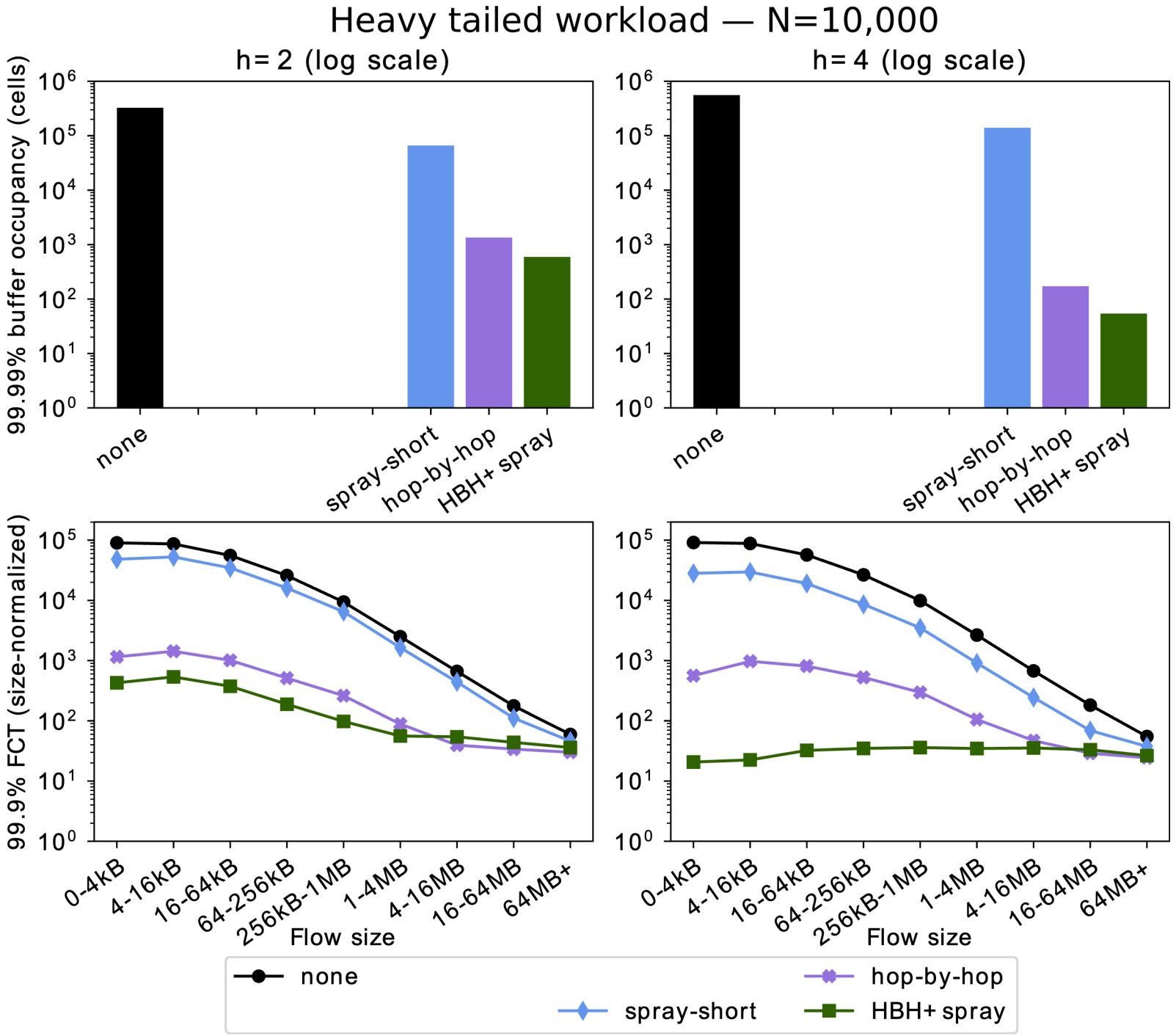

- Limits the number of cells queuing for the same destination at any node
- Deadlock-free!  No head-of-line blocking!

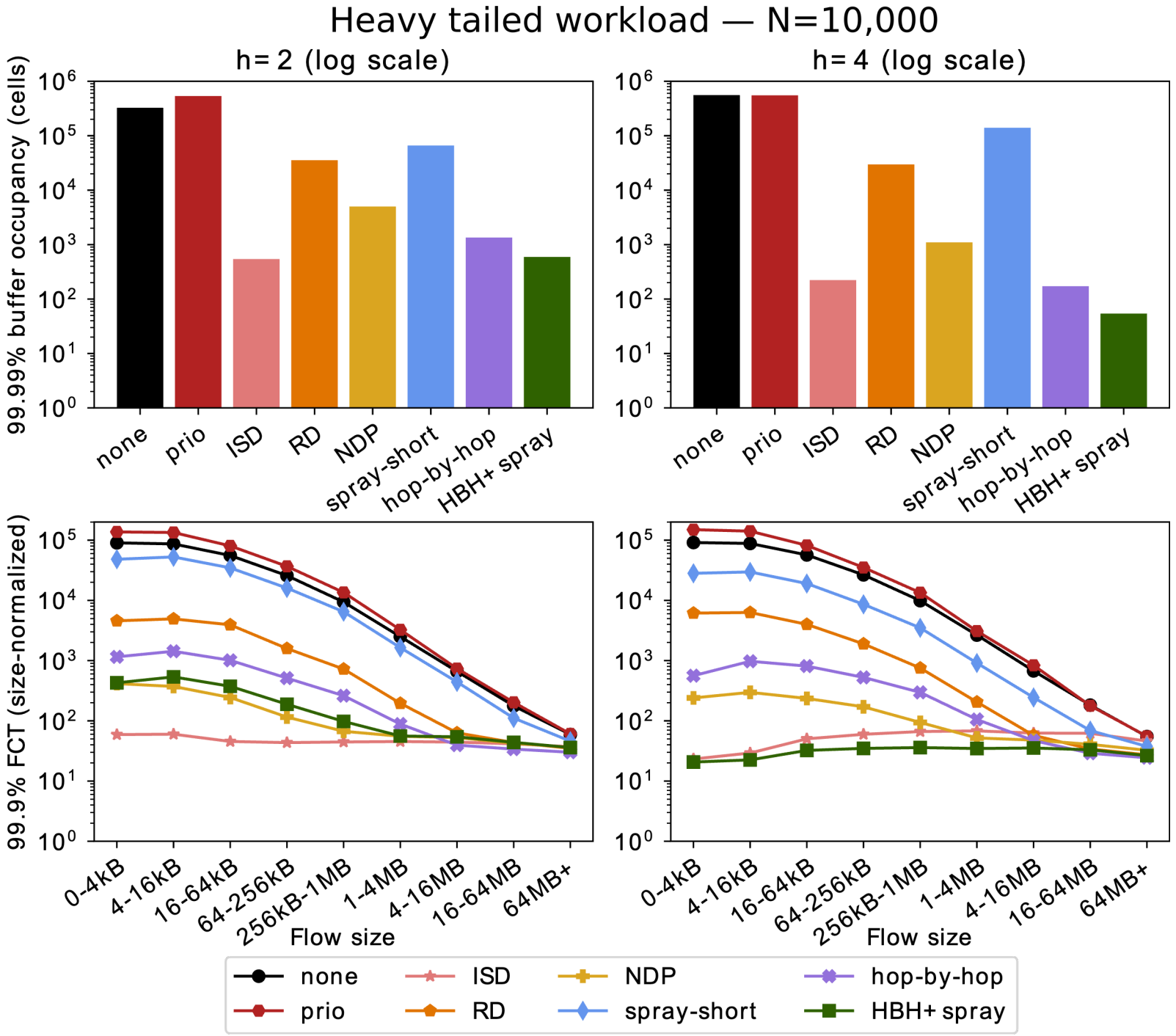
# Testing our congestion control mechanisms

- 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
h= 2
h= 4
99.99% buffer occupancy (cells)
0
1000
2000
3000
4000
none
spray-short
hop-by-hop
HBH+ spray
99.9% FCT (size-normalized)
0
200
400
600
800
1000
0-4kB
4-16kB
16-64kB
64-256kB
256kB-1MB
1-4MB
Flow size
none
hop-by-hop
spray-short
HBH+ spray

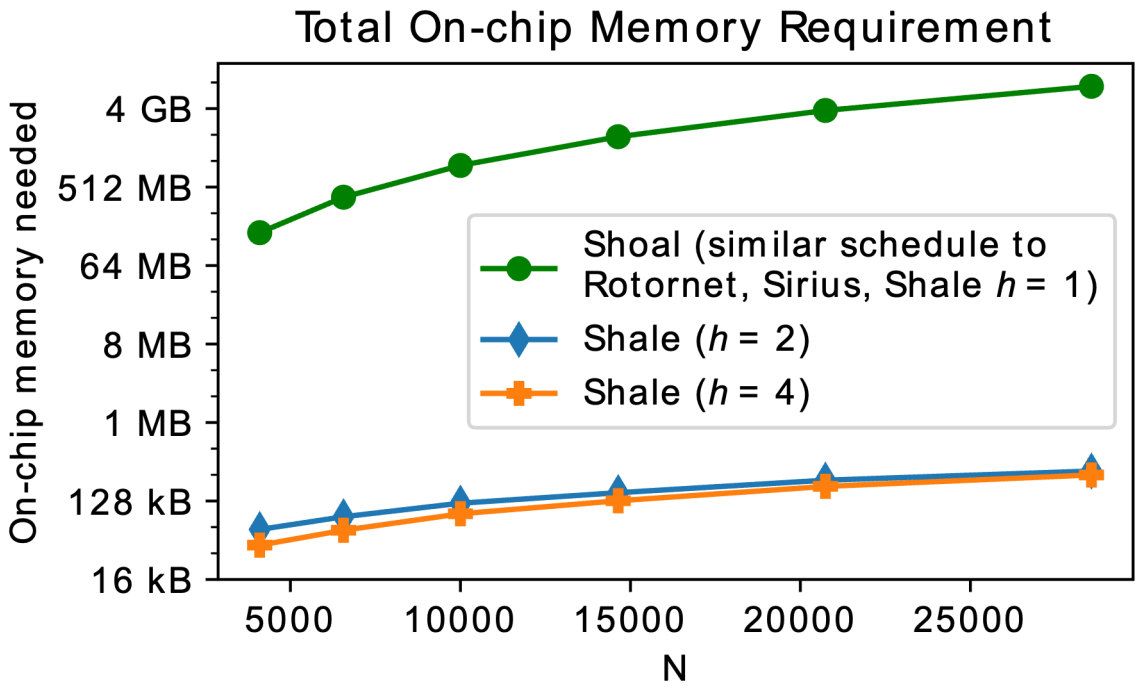

Short flow workload — N=10,000
h= 2
h= 4
99.99% buffer occupancy (cells)
0
1000
2000
3000
4000
none
prio
ISD
RD
NDP
spray-short
hop-by-hop
HBH+ spray
99.9% FCT (size-normalized)
0
200
400
600
800
1000
0-4kB
4-16kB
16-64kB
64-256kB
256kB-1MB
1-4MB
Flow size
none
prio
ISD
RD
NDP
spray-short
hop-by-hop
HBH+ spray


Heavy tailed workload — N=10,000
h= 2 (log scale)
h= 4 (log scale)
99.99% buffer occupancy (cells)
$10^0$
$10^1$
$10^2$
$10^3$
$10^4$
$10^5$
$10^6$
none
spray-short
hop-by-hop
HBH+ spray
99.9% FCT (size-normalized)
$10^0$
$10^1$
$10^2$
$10^3$
$10^4$
$10^5$
0-4kB
4-16kB
16-64kB
64-256kB
256kB-1MB
1-4MB
4-16MB
16-64MB
64MB+
Flow size
none
hop-by-hop
spray-short
HBH+ spray

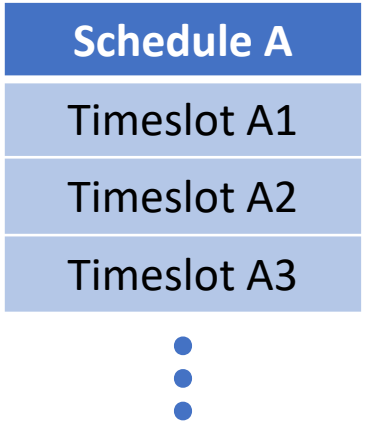

Heavy tailed workload — N=10,000
h= 2 (log scale)
h= 4 (log scale)
99.99% buffer occupancy (cells)
99.9% FCT (size-normalized)
none
prio
ISD
RD
NDP
spray-short
hop-by-hop
HBH+ spray
0-4kB
4-16kB
16-64kB
64-256kB
256kB-1MB
1-4MB
4-16MB
16-64MB
64MB+
Flow size

# Implementing Shale

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



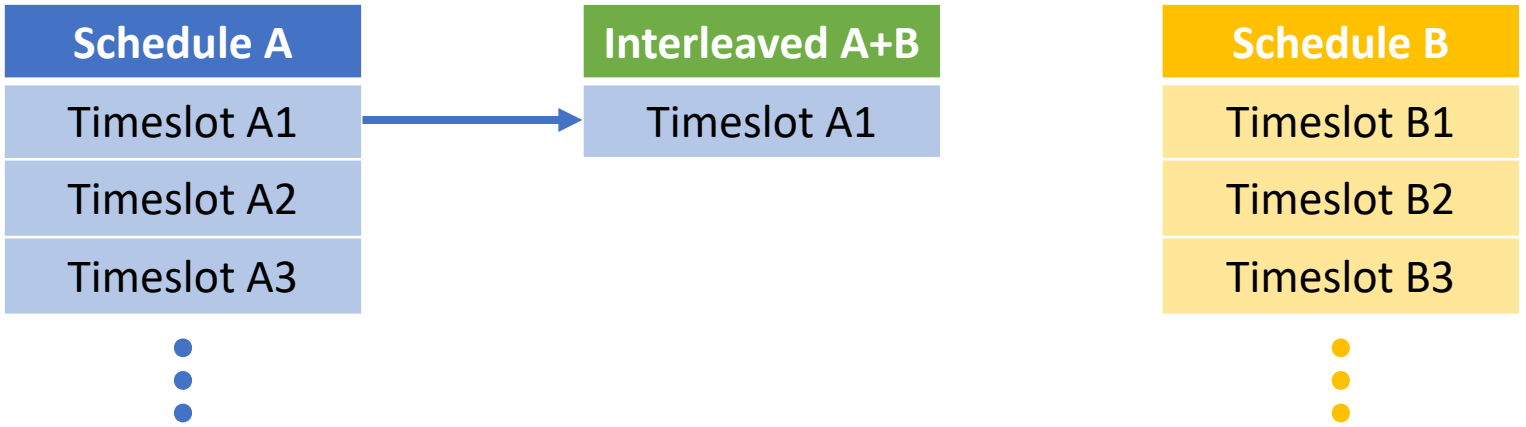
# Interleaving ORN Schedules



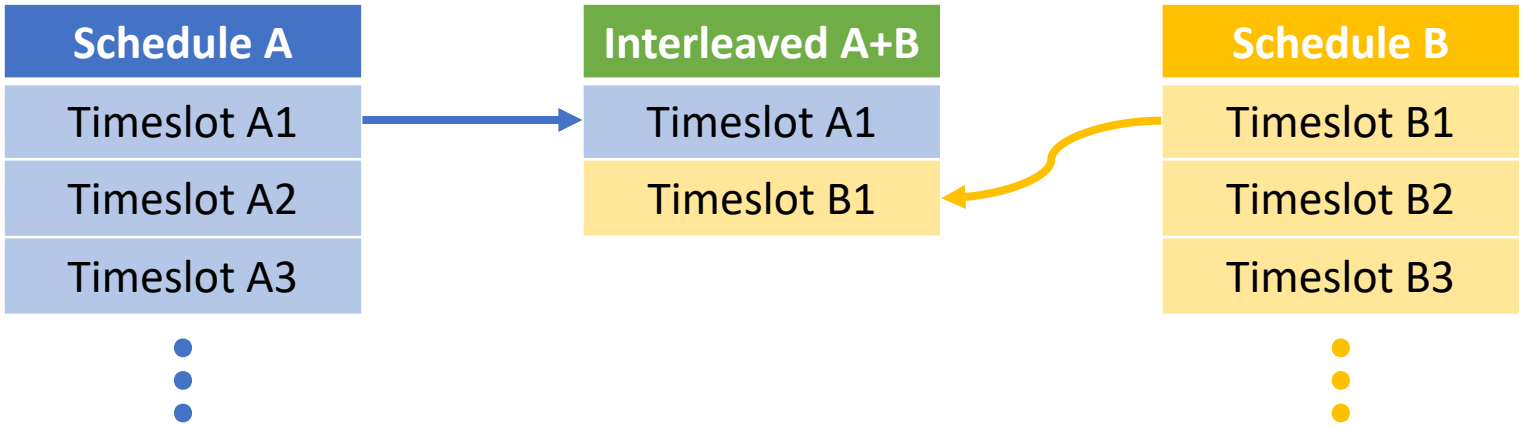
# Interleaving ORN Schedules

# Interleaving ORN Schedules

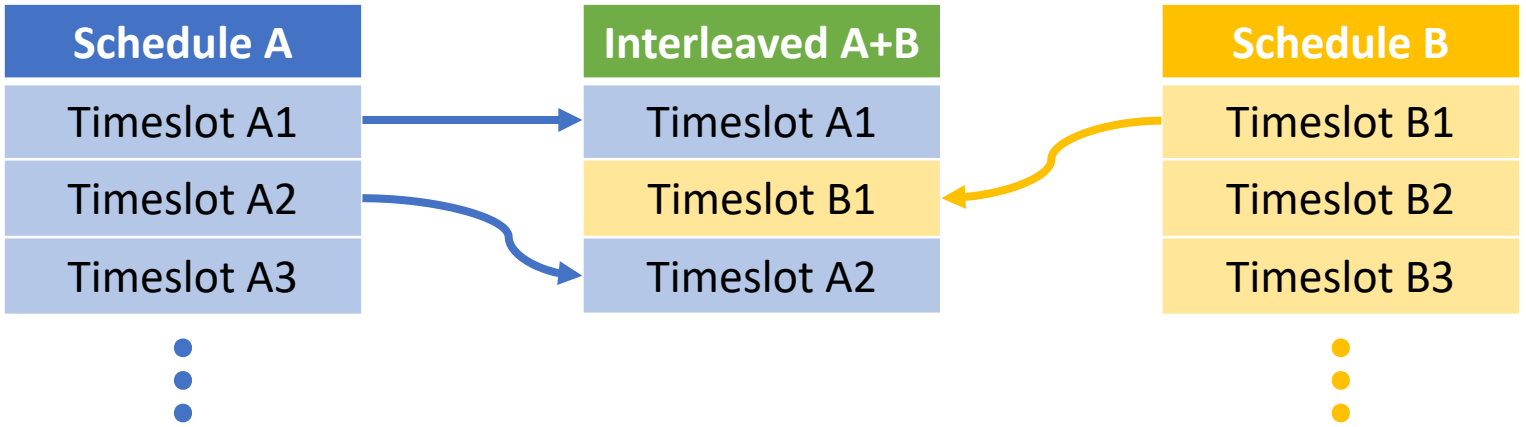
# Interleaving ORN Schedules



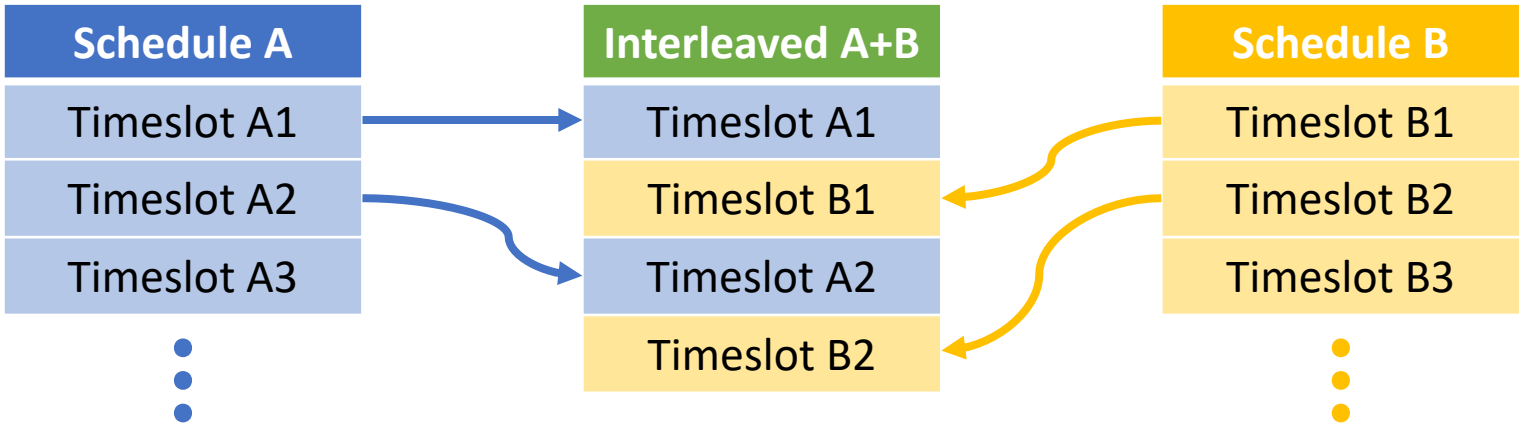
# Interleaving ORN Schedules



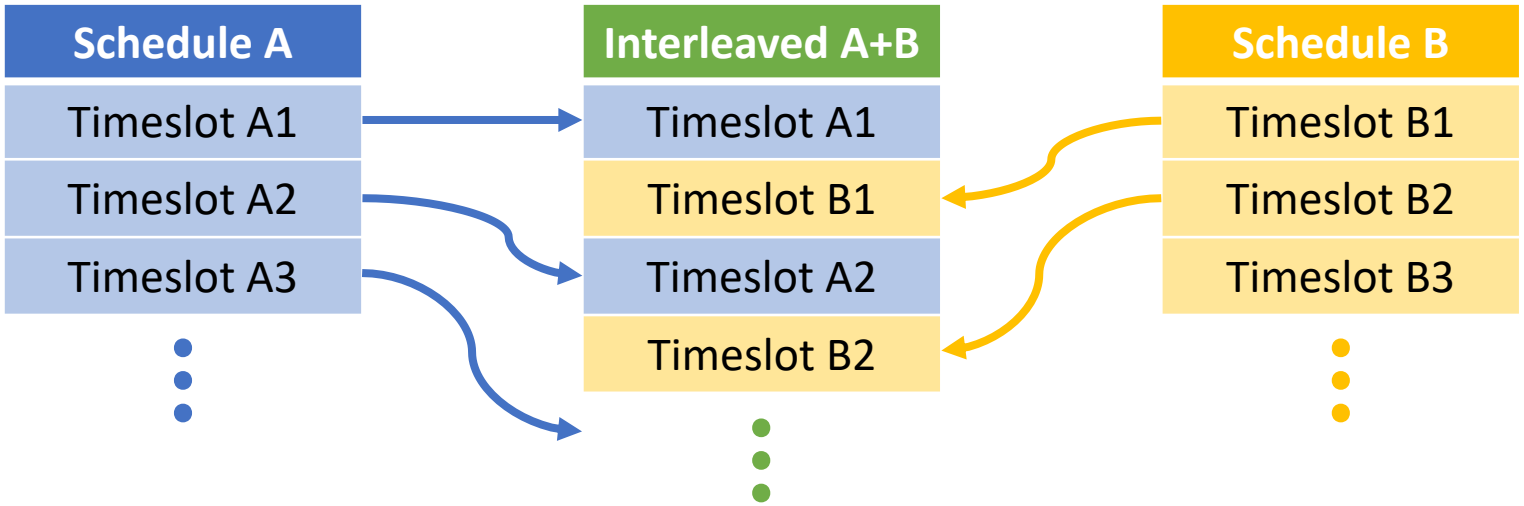
# Interleaving ORN Schedules



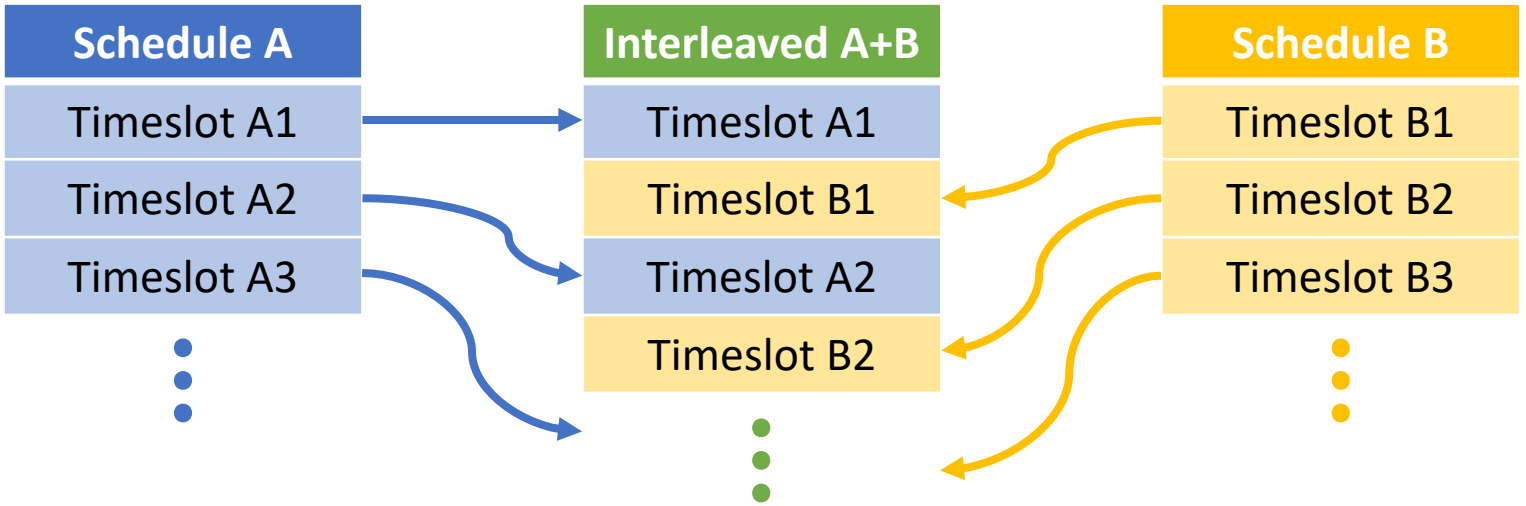
# Interleaving ORN Schedules



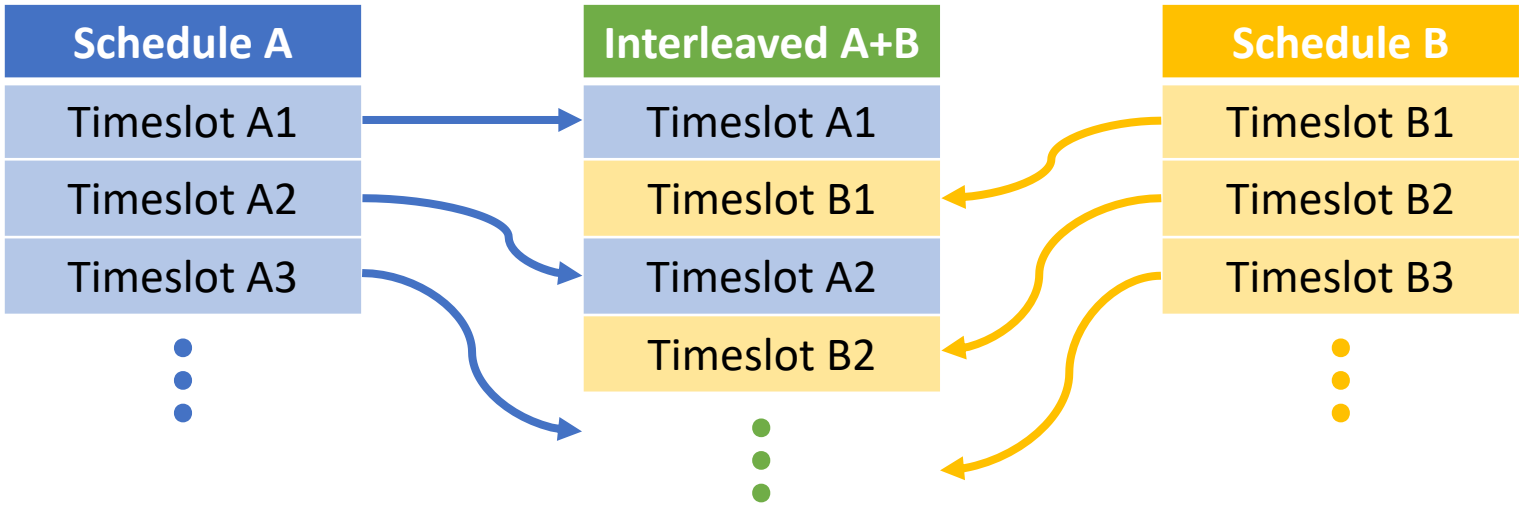
# Interleaving ORN Schedules



# Interleaving ORN Schedules

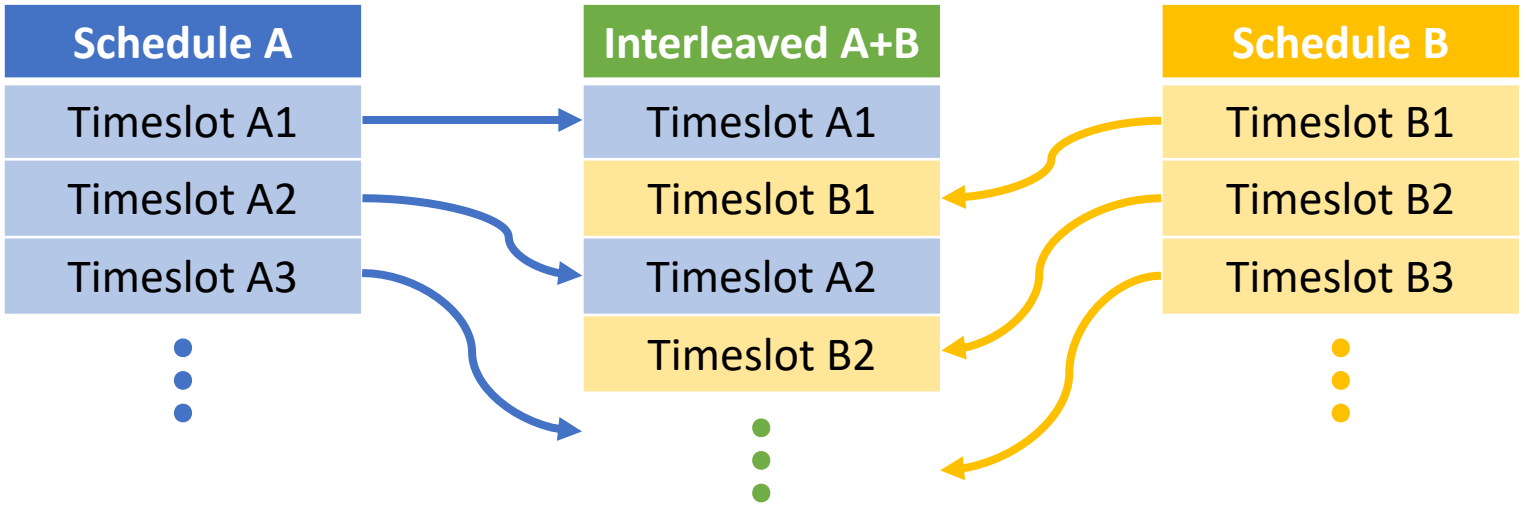


# Interleaving ORN Schedules



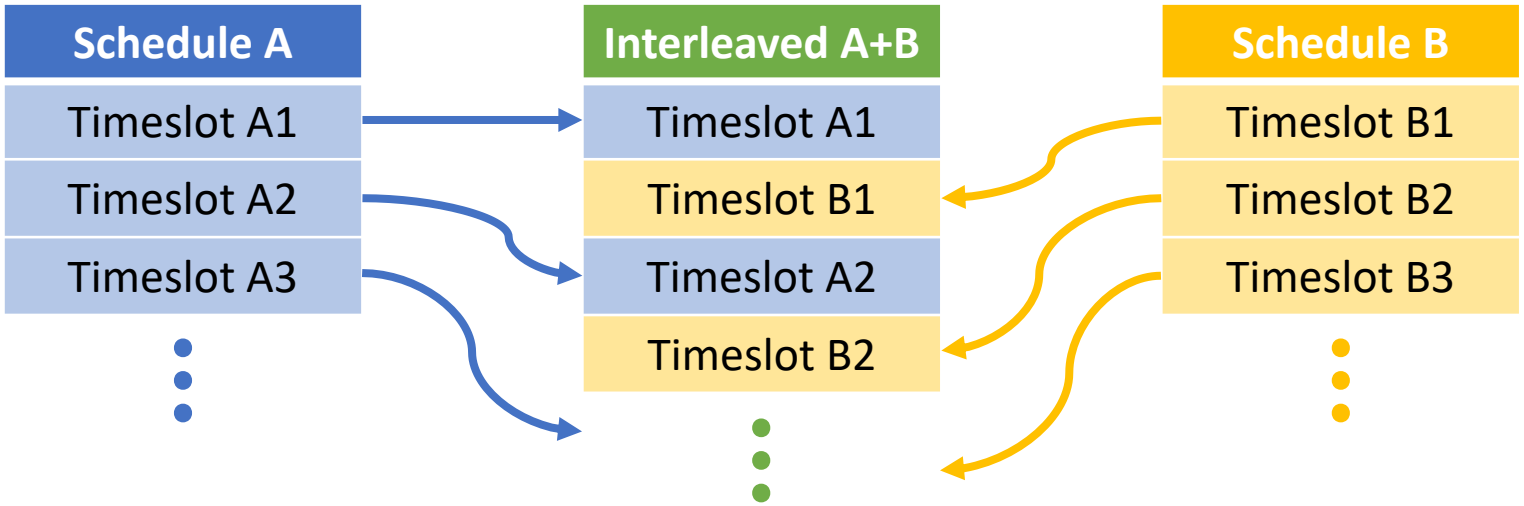

- Total throughput is a weighted average of each of the sub-schedules

# Interleaving ORN Schedules




- Total throughput is a weighted average of each of the sub-schedules
- Intrinsic latency of each sub-schedule is increased

# Interleaving ORN Schedules




- Total throughput is a weighted average of each of the sub-schedules
- Intrinsic latency of each sub-schedule is increased
  - Propagation delay remains unchanged!

# 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
- Supports interleaving to combine multiple tunings

# Thank you!

Nitika Saran

Tegan Wilson

Robert Kleinberg

Vishal Shrivastav

Hakim Weatherspoon