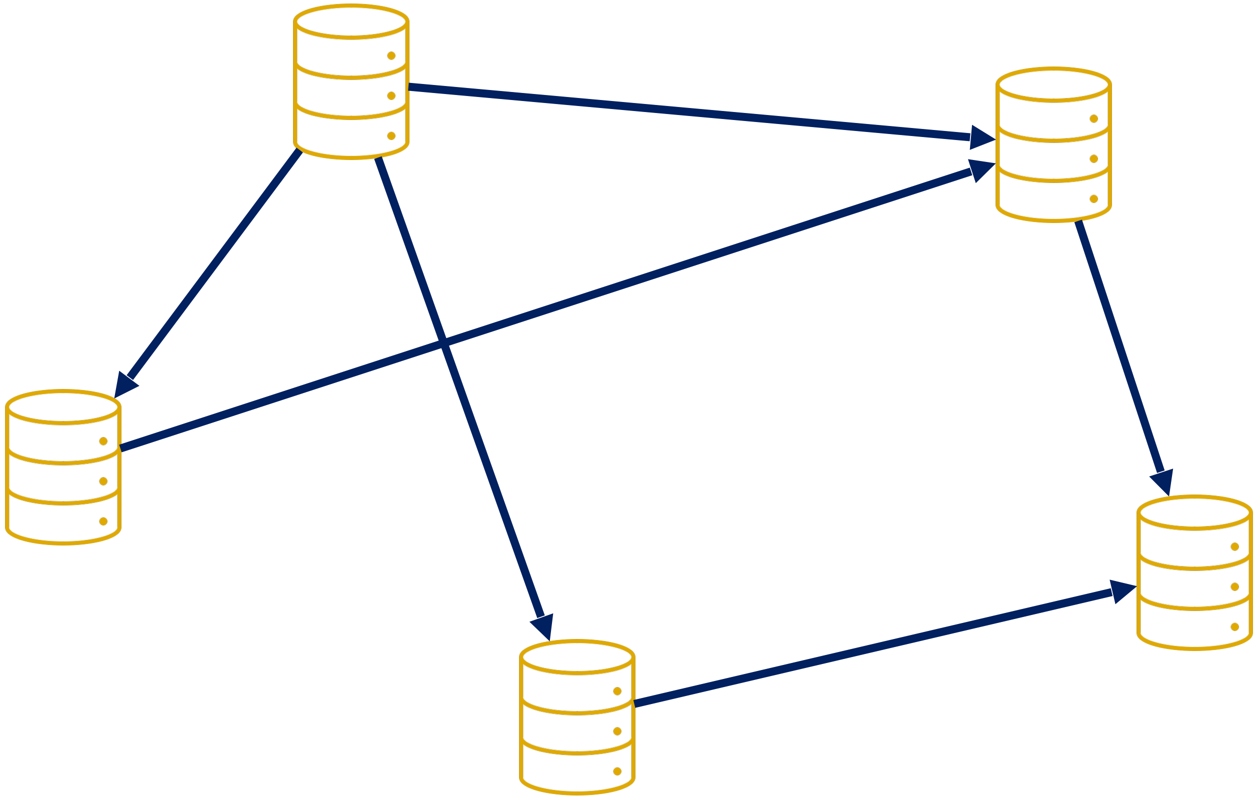
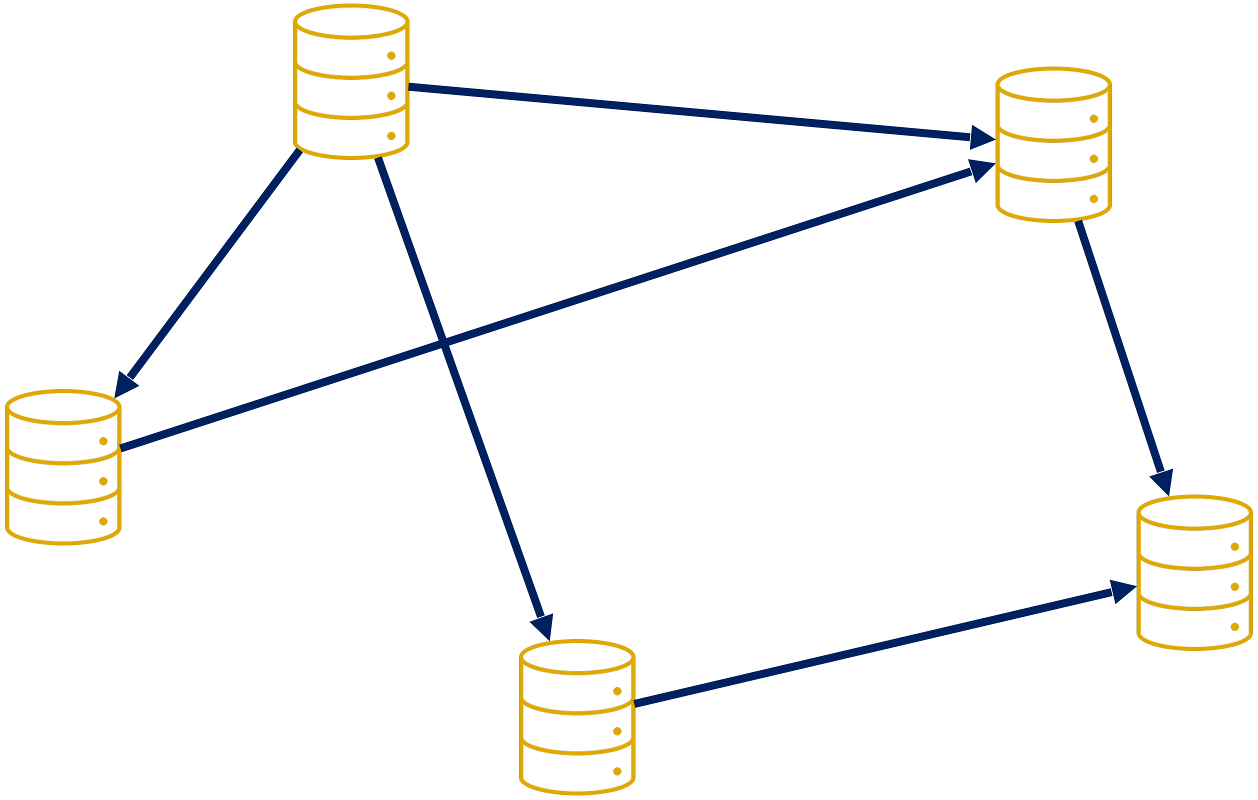
# Extending Optimal Oblivious Reconfigurable Networks to all $N$

*APOCS 2023*

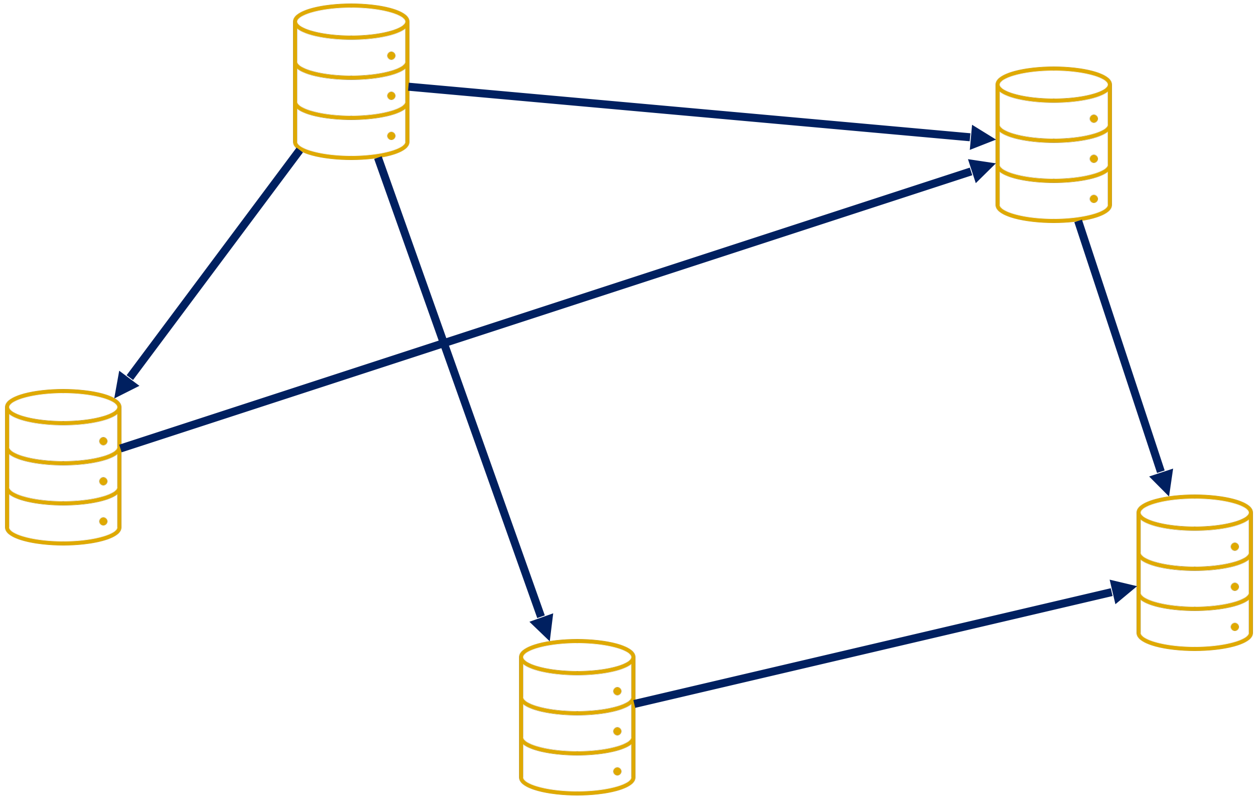
**Tegan Wilson**, Daniel Amir, Vishal Shrivastav, Hakim Weatherspoon, Robert Kleinberg





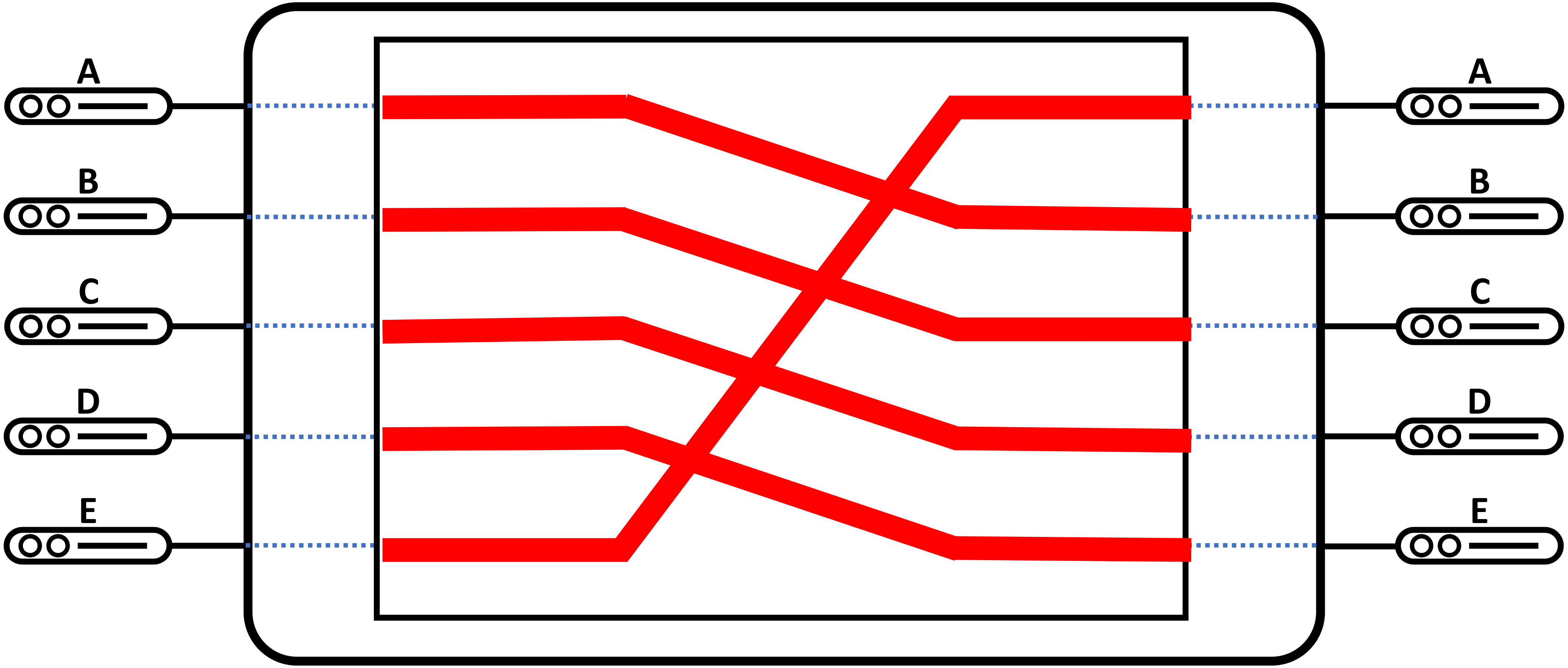
How do we connect servers so they can communicate?

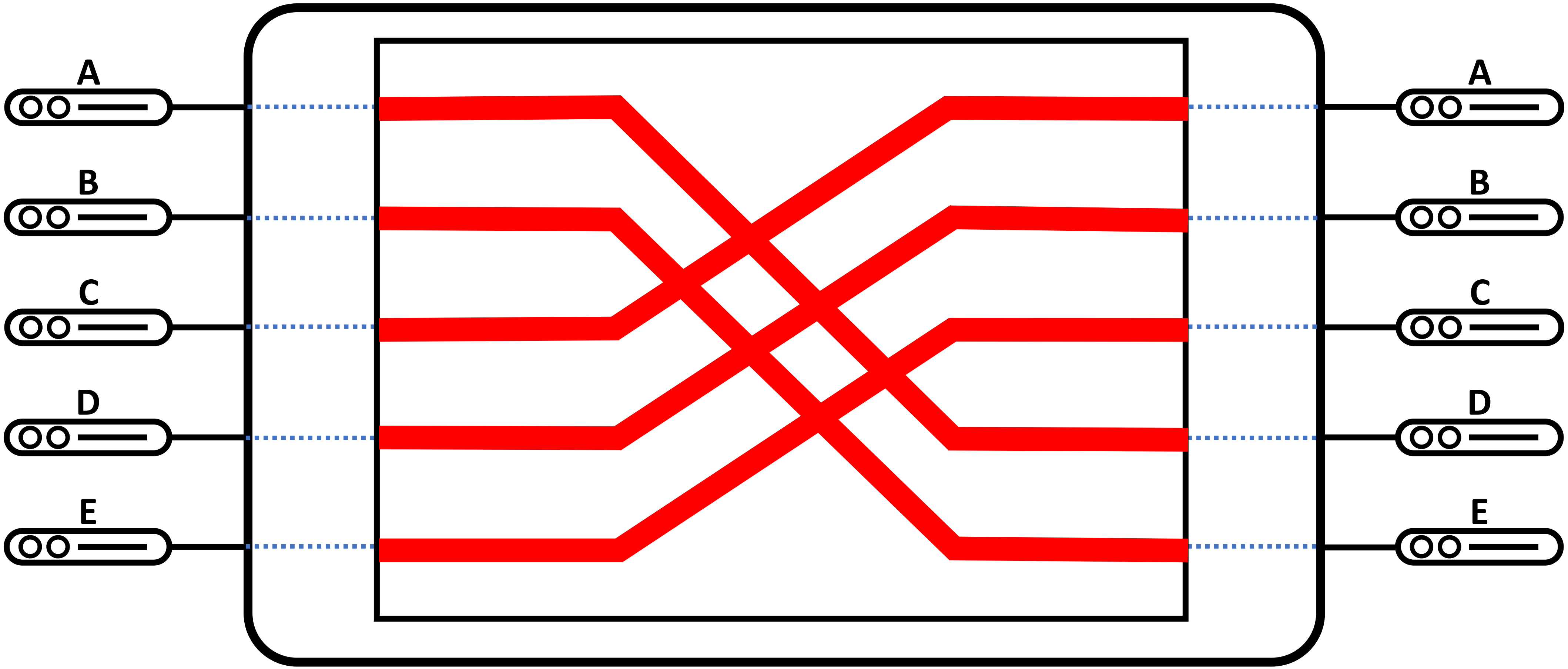


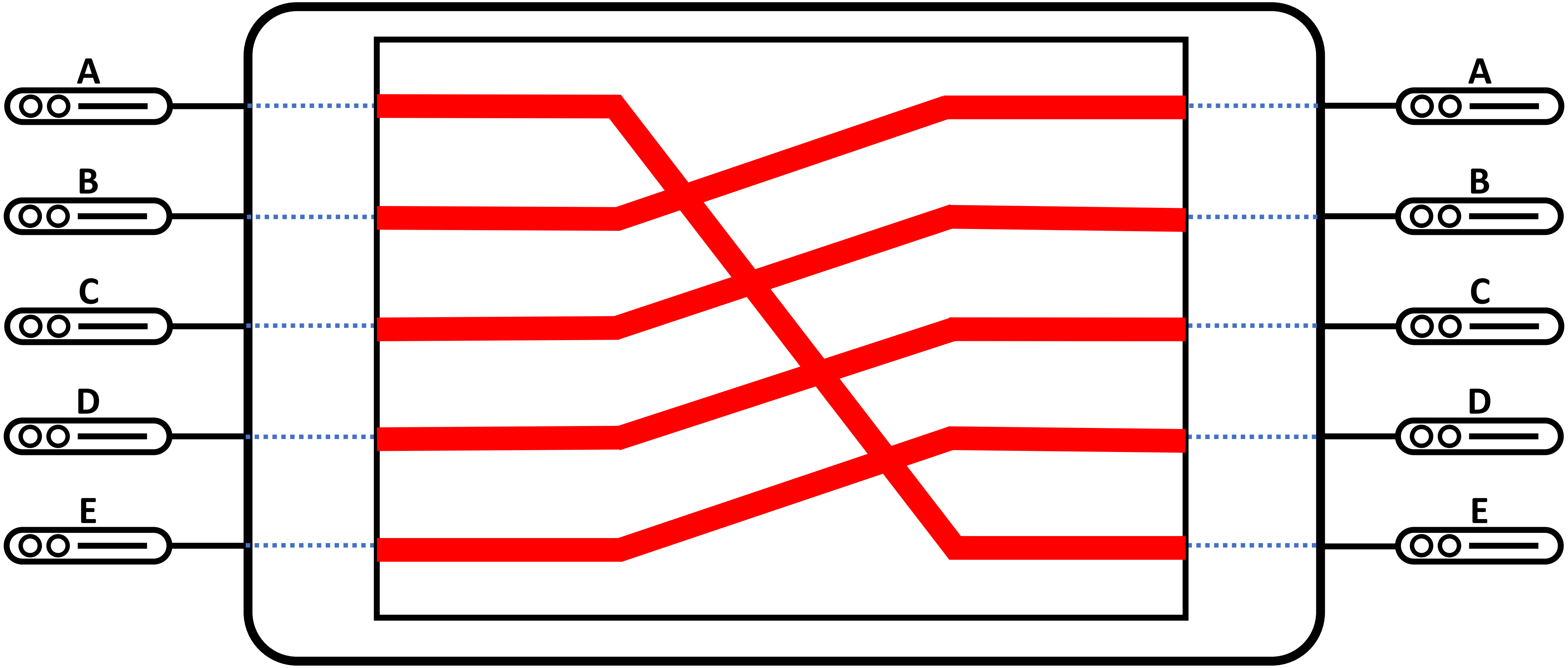
How do we connect servers so they can communicate?

How do we route messages along those connections?


A
B
C
D
E
A
B
C
D
E


A
B
C
D
E
A
B
C
D
E

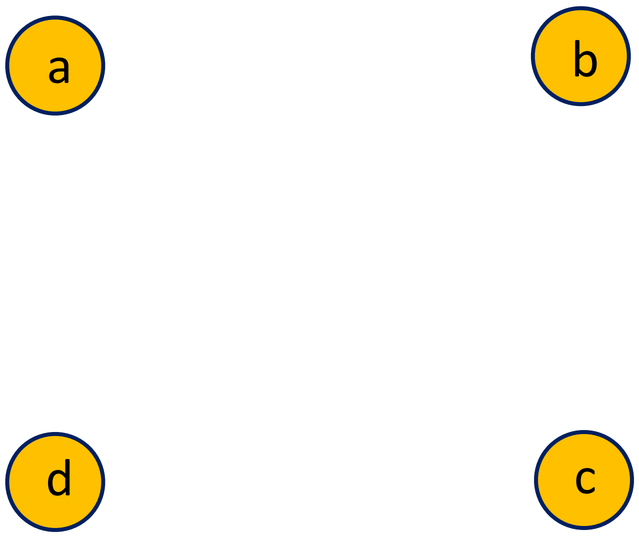

A
B
C
D
E
A
B
C
D
E


A
B
C
D
E
A
B
C
D
E

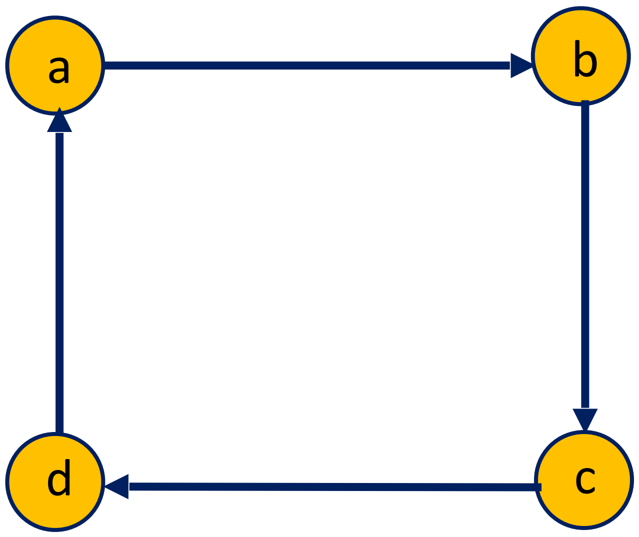
# Oblivious Reconfigurable Networks (ORNs)

- Set of $N$ nodes
- Edges reconfigure between each timestep according to a predefined schedule
- Route messages obliviously
  - Co-designing a connection schedule and routing protocol

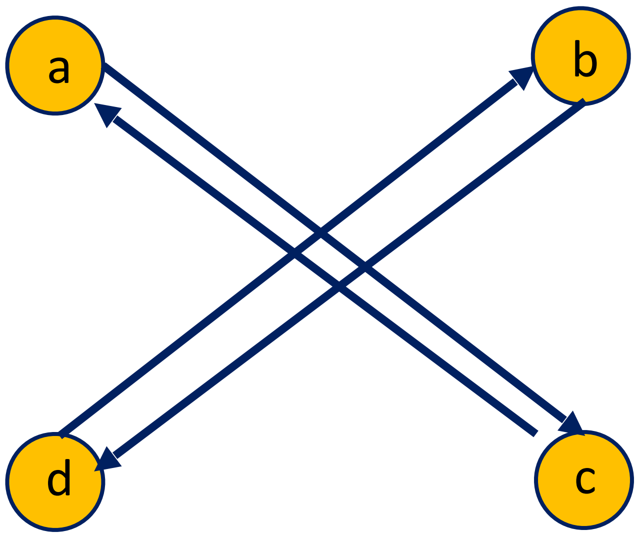
# Oblivious Reconfigurable Networks



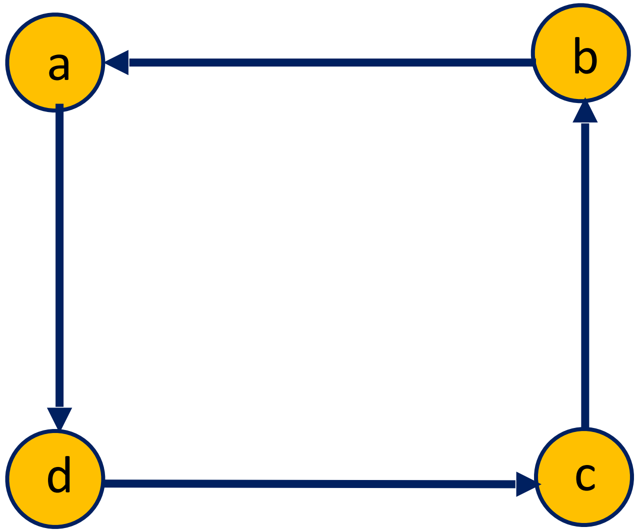
# Oblivious Reconfigurable Networks



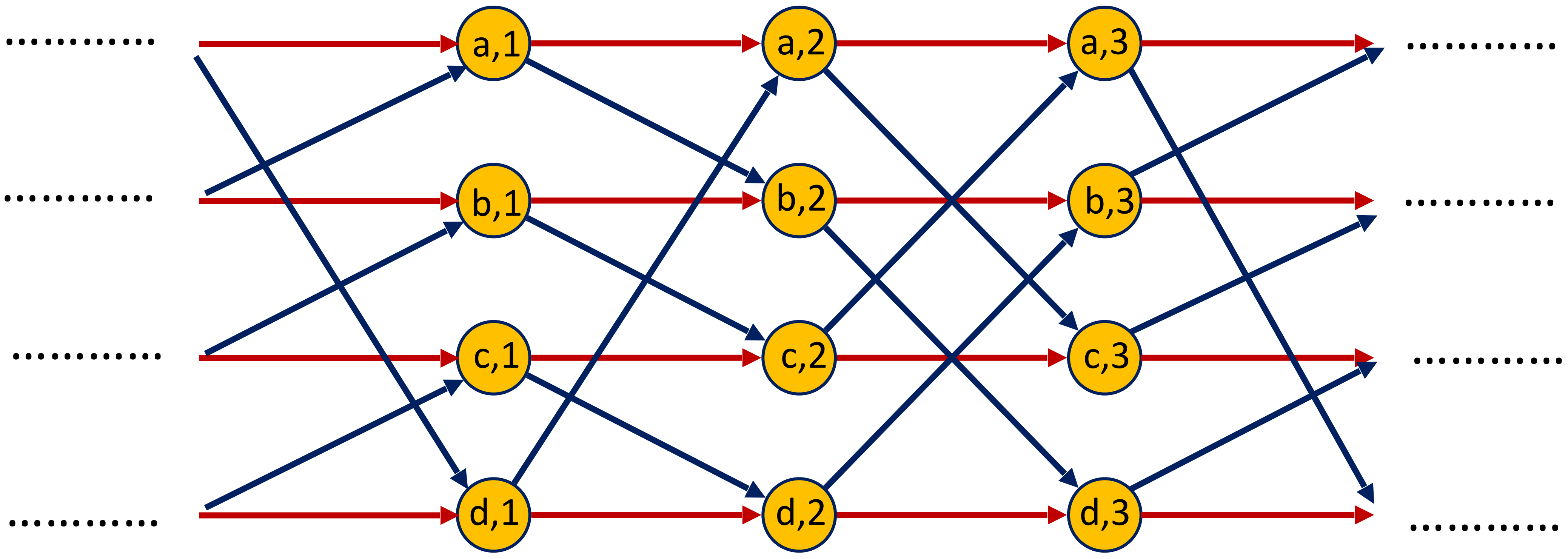
# Oblivious Reconfigurable Networks



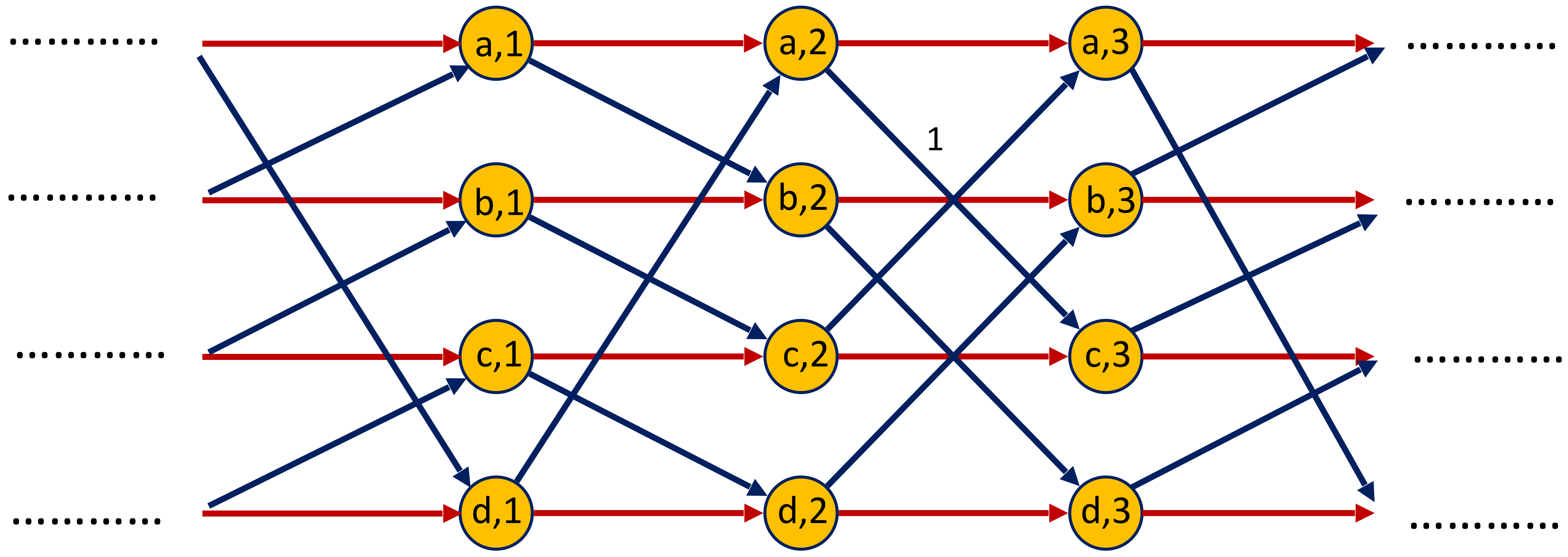
# Oblivious Reconfigurable Networks



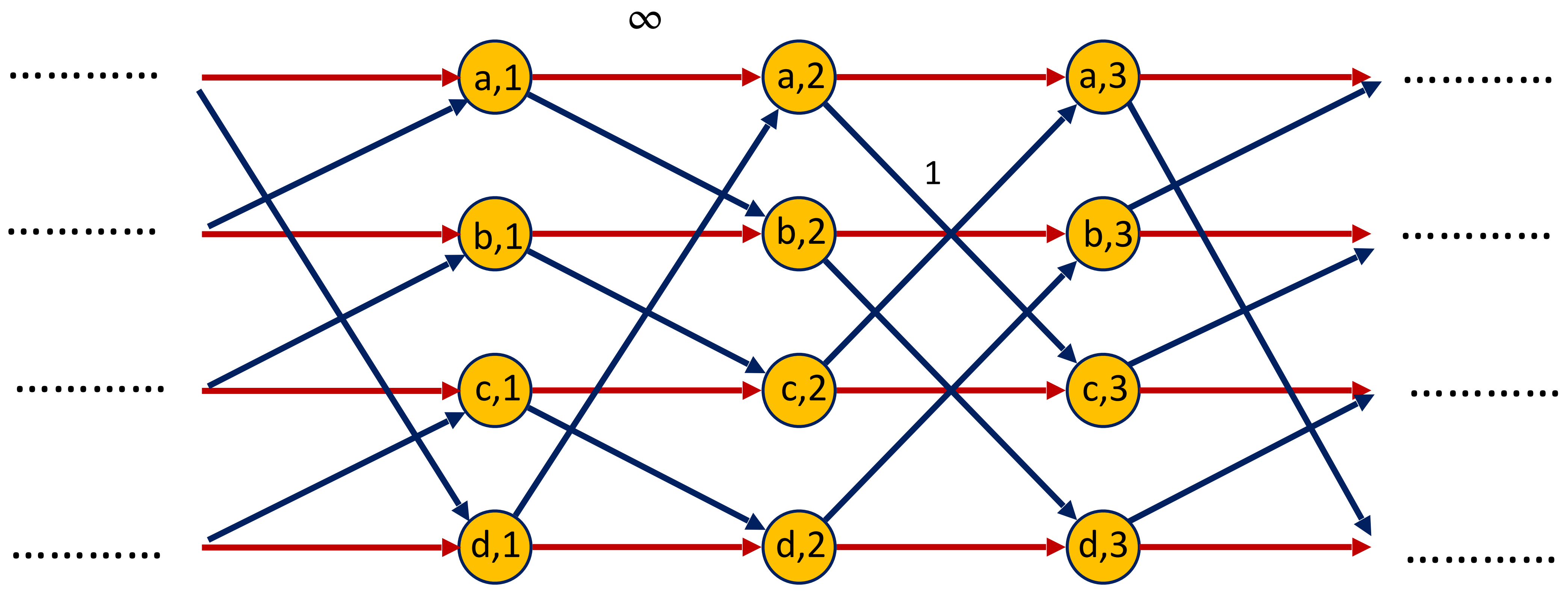
# Oblivious Reconfigurable Networks



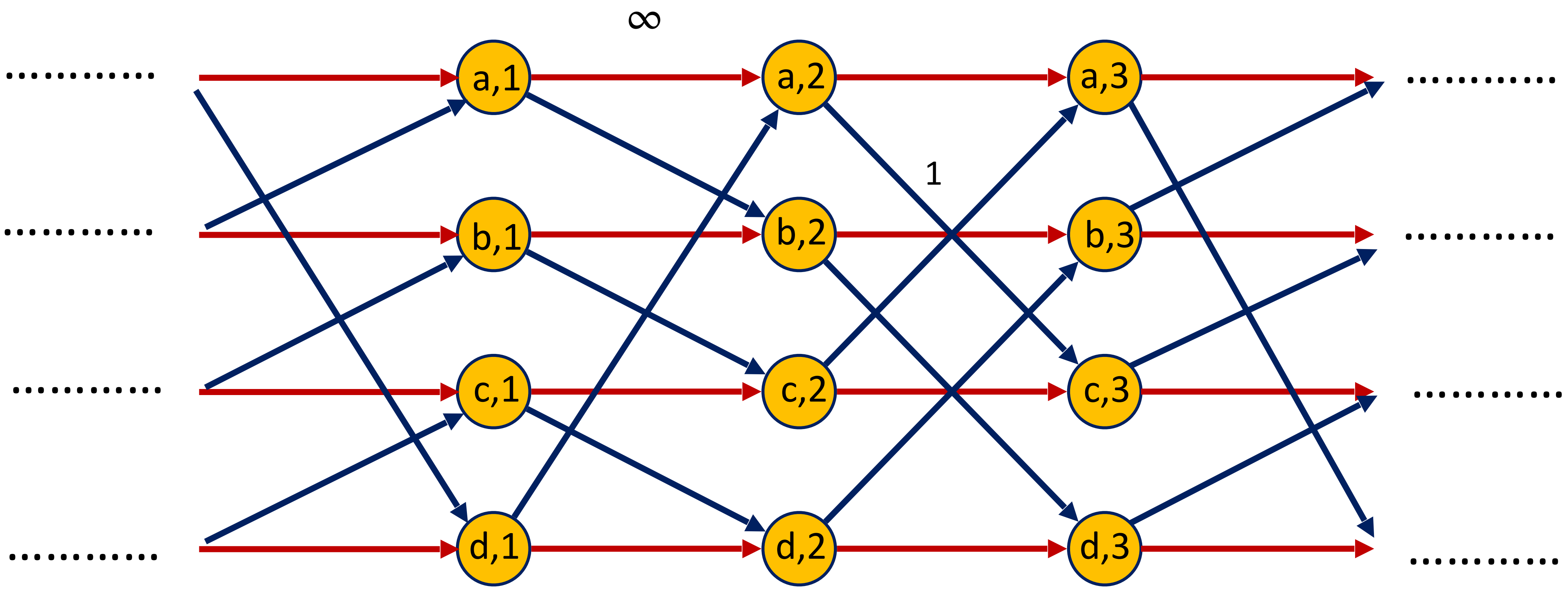
# Oblivious Reconfigurable Networks



# Oblivious Reconfigurable Networks

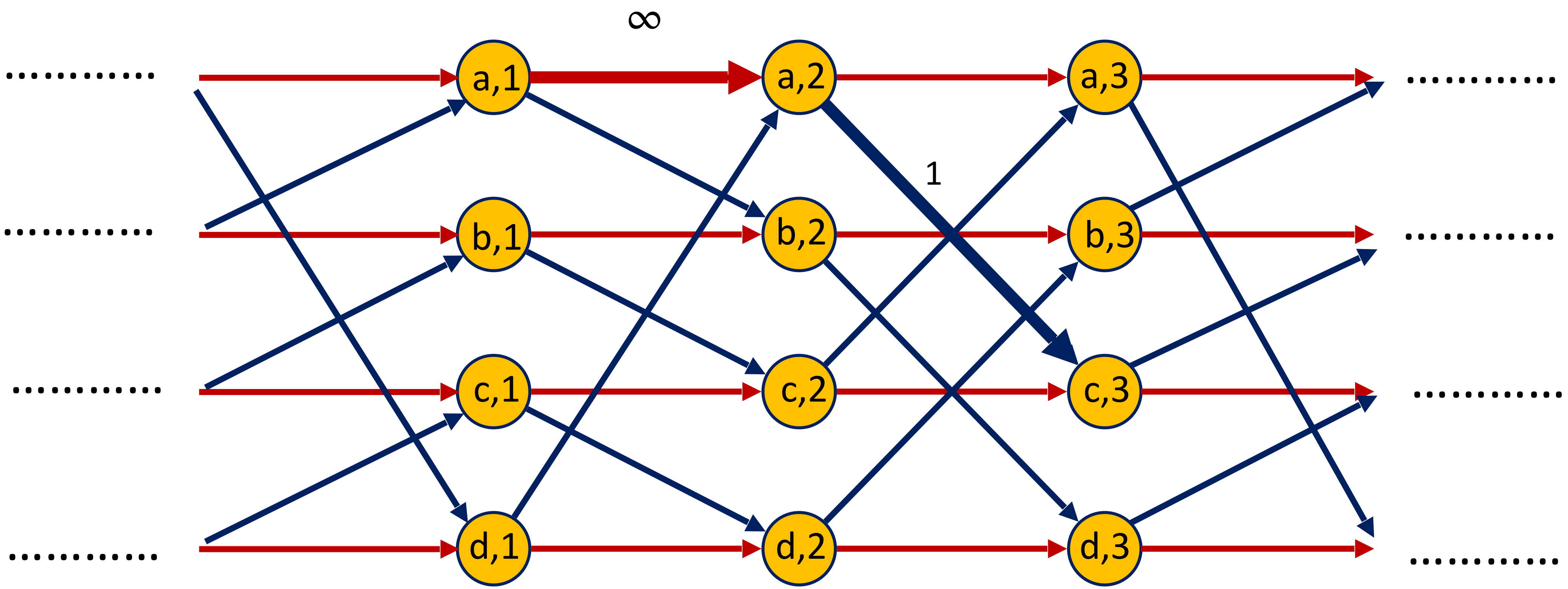


# Oblivious Reconfigurable Networks



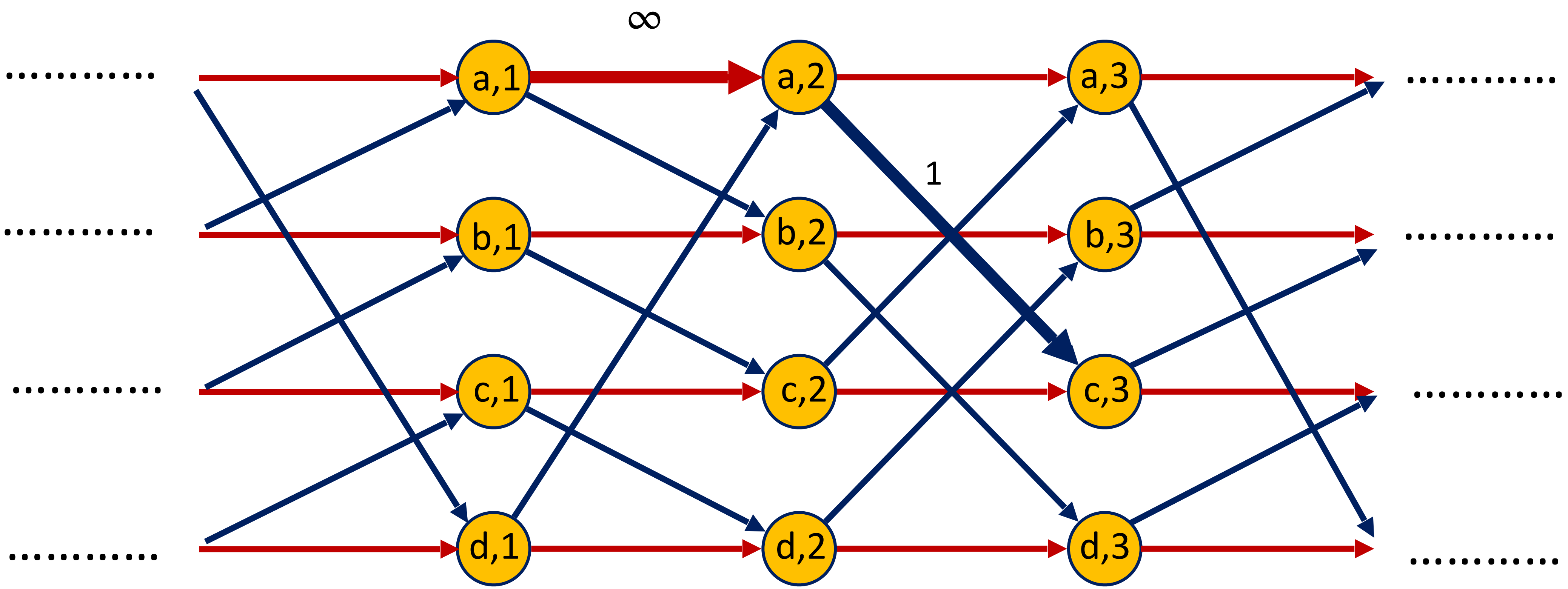

Route $a \to c$ starting at $t = 1$

# Oblivious Reconfigurable Networks




Route $a \to c$ starting at $t = 1$

# Oblivious Reconfigurable Networks



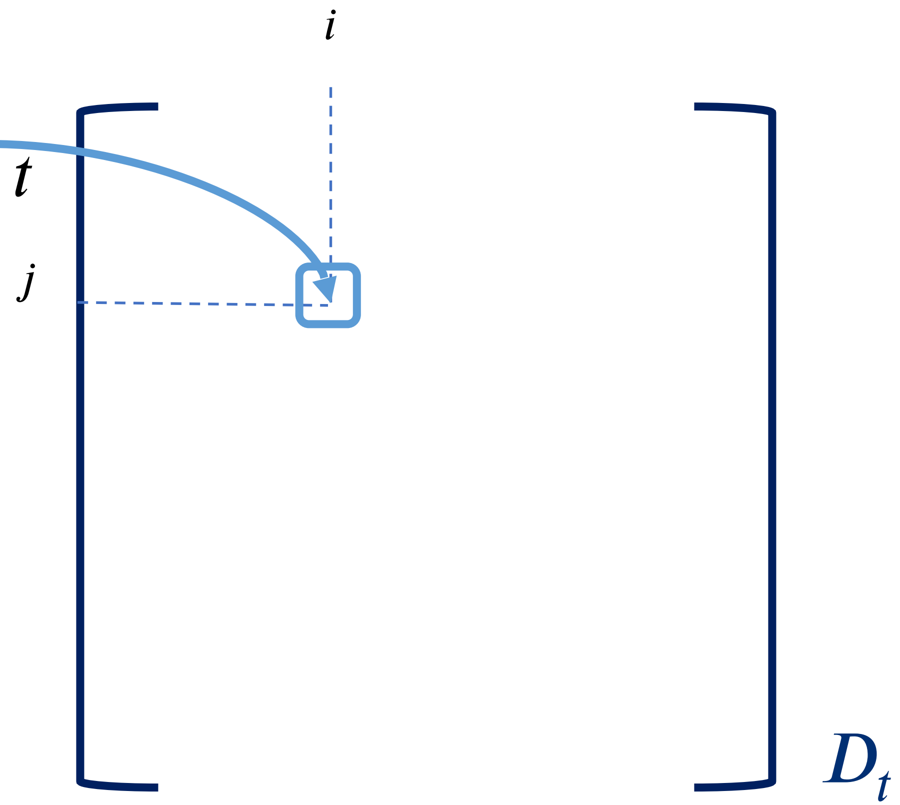

Route $a \rightarrow c$ starting at $t = 1$
Path has latency $L = 2$

# Throughput

$$\left[ \qquad\qquad \right]_{D_t}$$

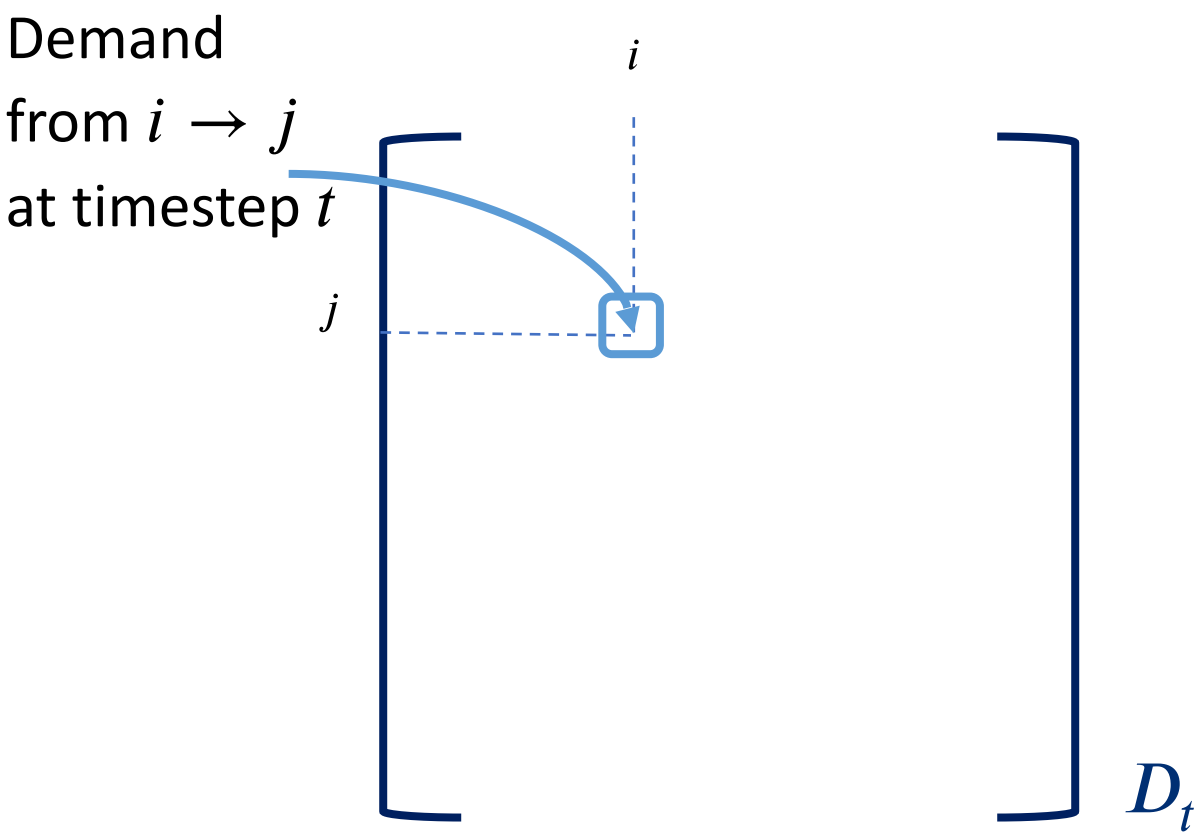
# Throughput

Demand
from $i \rightarrow j$
at timestep $t$



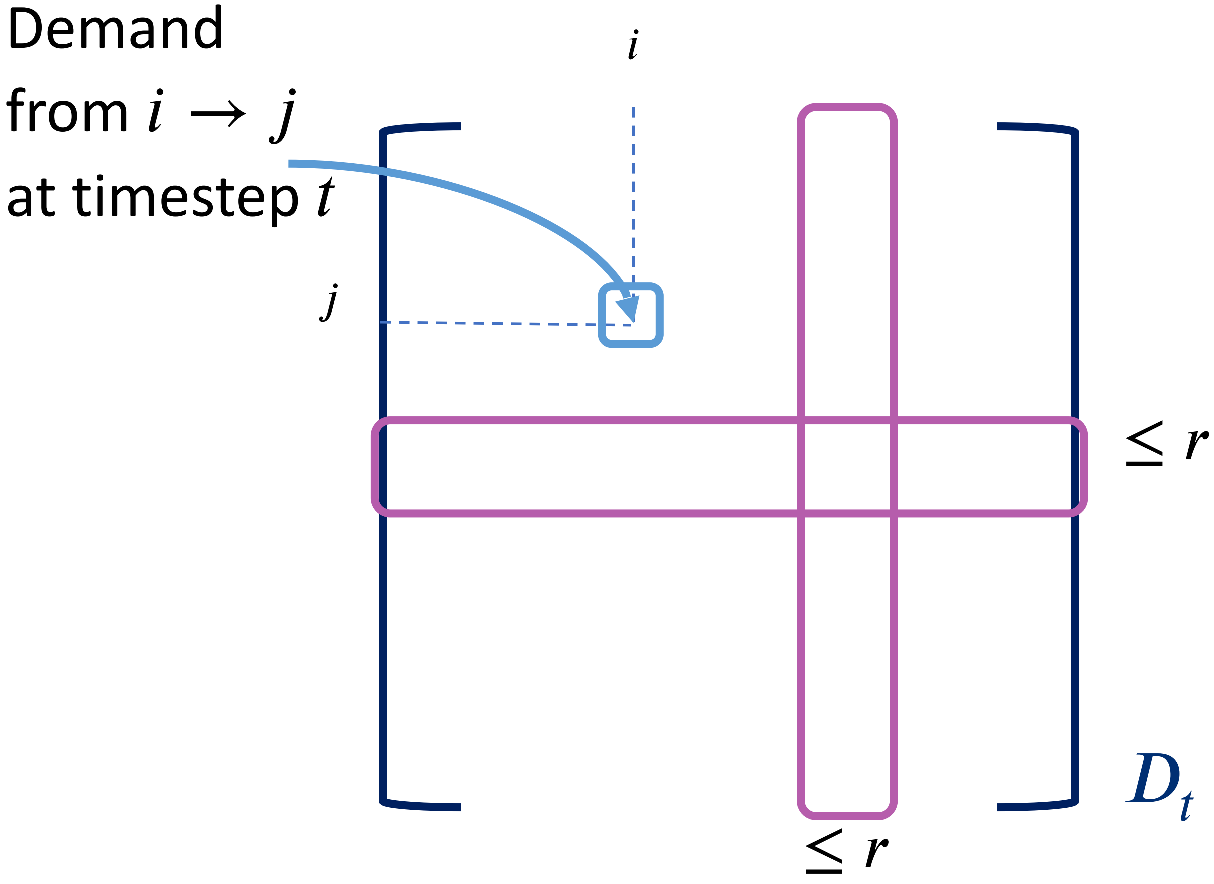
# Throughput

- A matrix requests throughput $r$



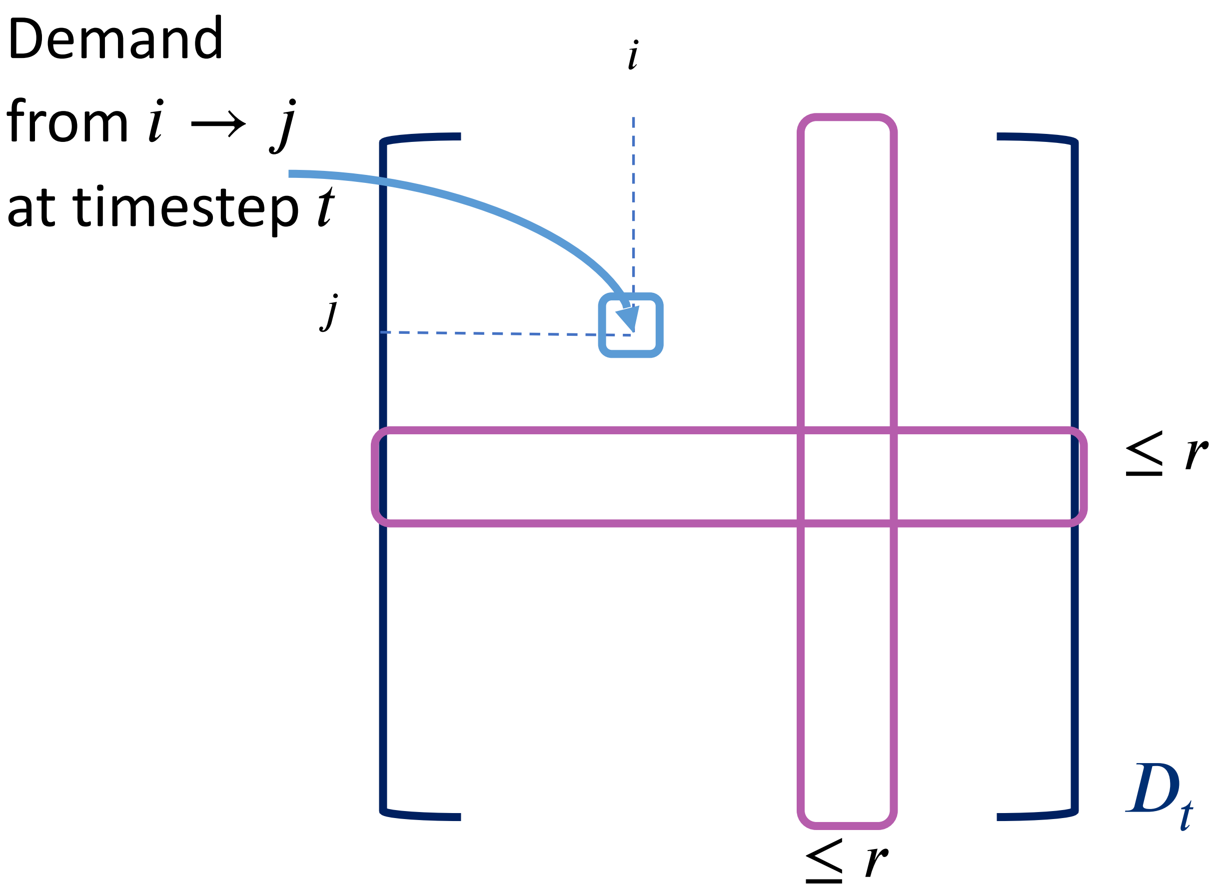
# Throughput

- A matrix requests throughput $r$



# Throughput

- A matrix requests throughput $r$
- An ORN design guarantees throughput $r$ if it can route all matrices requesting throughput $r$ without overloading edges



# The Problem

- Build an ORN design with:
  - High guaranteed throughput $r$
  - Low max latency $L$
- These objectives are in conflict with each other!
  - So looking for a tradeoff

*Theorem*[1]*:* Let $0 < r \le \frac{1}{2}$ be a constant, and $h = \left\lfloor \frac{1}{2r} \right\rfloor$, and $\varepsilon = h + 1 - \frac{1}{2r} \in (0,1]$, and let $L^*(r, N)$ be the function

$$L^*(r, N) = h(N^{1/(h+1)} + (\varepsilon N)^{1/h})$$

[1]Amir, Wilson, Shrivastav, Weatherspoon, Kleinberg, Agarwal. "*Optimal Oblivious Reconfigurable Networks.*" STOC'22.

*Theorem*[1]*:* Let $0 < r \leq \frac{1}{2}$ be a constant, and $h = \left\lfloor \frac{1}{2r} \right\rfloor$, and $\varepsilon = h + 1 - \frac{1}{2r} \in (0,1]$, and let $L^*(r, N)$ be the function

$$L^*(r, N) = h(N^{1/(h+1)} + (\varepsilon N)^{1/h})$$

$$\left.\begin{array}{l}\text{Then } \textit{for every} \text{ ORN design on } N \text{ nodes that guarantees} \\ \text{throughput } r\text{, the maximum latency is at least} \\ \Omega(L^*(r, N)).\end{array}\right\} \text{Lower bound}$$

[1]Amir, Wilson, Shrivastav, Weatherspoon, Kleinberg, Agarwal. "*Optimal Oblivious Reconfigurable Networks.*" STOC'22.

*Theorem*[1]: Let $0 < r \leq \frac{1}{2}$ be a constant, and $h = \left\lfloor \frac{1}{2r} \right\rfloor$, and $\varepsilon = h + 1 - \frac{1}{2r} \in (0,1]$, and let $L^*(r, N)$ be the function

$$L^*(r, N) = h(N^{1/(h+1)} + (\varepsilon N)^{1/h})$$

Then *for every* ORN design on $N$ nodes that guarantees throughput $r$, the maximum latency is at least $\Omega(L^*(r, N))$.

Lower bound

Furthermore for infinitely many $N$, *there exists* an ORN design on $N$ nodes that guarantees throughput $r$ and whose maximum latency is $\mathrm{O}(L^*(r, N))$.

Upper bound

[1] Amir, Wilson, Shrivastav, Weatherspoon, Kleinberg, Agarwal. "*Optimal Oblivious Reconfigurable Networks.*" STOC'22.

*Theorem*[1]: Let $0 < r \leq \frac{1}{2}$ be a constant, and $h = \left\lfloor \frac{1}{2r} \right\rfloor$, and $\varepsilon = h + 1 - \frac{1}{2r} \in (0,1]$, and let $L^*(r, N)$ be the function

$$L^*(r, N) = h(N^{1/(h+1)} + (\varepsilon N)^{1/h})$$

Then *for every* ORN design on $N$ nodes that guarantees throughput $r$, the maximum latency is at least $\Omega(L^*(r, N))$.

Lower bound

Furthermore **for infinitely many $N$**, *there exists* an ORN design on $N$ nodes that guarantees throughput $r$ and whose maximum latency is $\mathrm{O}(L^*(r, N))$.

Upper bound

[1]Amir, Wilson, Shrivastav, Weatherspoon, Kleinberg, Agarwal. "*Optimal Oblivious Reconfigurable Networks.*" STOC'22.

*Theorem*[1]: Let $0 < r \leq \frac{1}{2}$ be a constant, and $h = \left\lfloor \frac{1}{2r} \right\rfloor$, and $\varepsilon = h + 1 - \frac{1}{2r} \in (0,1]$, and let $L^*(r, N)$ be the function

$$L^*(r, N) = h(N^{1/(h+1)} + (\varepsilon N)^{1/h})$$

Then **for all sufficiently large $N$**, *there exists* an ORN design on $N$ nodes that guarantees throughput $r$ and whose maximum latency is $\mathrm{O}(L^*(r, N))$.

[1]Amir, Wilson, Shrivastav, Weatherspoon, Kleinberg, Agarwal. "*Optimal Oblivious Reconfigurable Networks.*" STOC'22.

*Theorem*[1]: Let $0 < r \leq \frac{1}{2}$ be a constant, and $h = \left\lfloor \frac{1}{2r} \right\rfloor$, and $\varepsilon = h + 1 - \frac{1}{2r} \in (0,1]$, and let $L^*(r, N)$ be the function

$$L^*(r, N) = h(N^{1/(h+1)} + (\varepsilon N)^{1/h})$$

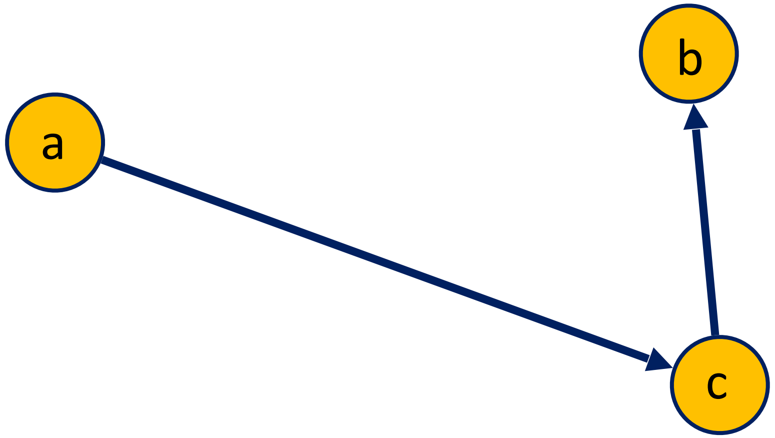
Then **for all sufficiently large $N$**, *there exists* an ORN design on $N$ nodes that guarantees throughput $r$ and whose maximum latency is $\mathrm{O}(L^*(r, N))$.

…Whenever the throughput $r$ is not $\frac{1}{even\ integer}$

[1]Amir, Wilson, Shrivastav, Weatherspoon, Kleinberg, Agarwal. "*Optimal Oblivious Reconfigurable Networks.*" STOC'22.

# Valiant Load Balancing[2]

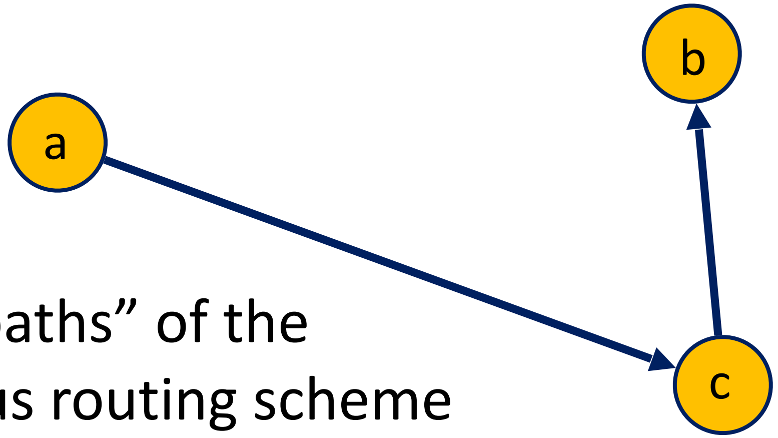
- Given routing protocol $R$ for the uniform demand matrix $D_{unif}(2r)$
- To route throughput $r$ obliviously from $a \rightarrow b$, choose a random intermediate node $c$ and use $R$ to route from $a \rightarrow c$ then $c \rightarrow b$




[2]Leslie G. Valiant. A scheme for fast parallel communication. *SIAM J Comput. '82*

# Valiant Load Balancing[2]

- Given routing protocol $R$ for the uniform demand matrix $D_{unif}(2r)$
- To route throughput $r$ obliviously from $a \to b$, choose a random intermediate node $c$ and use $R$ to route from $a \to c$ then $c \to b$

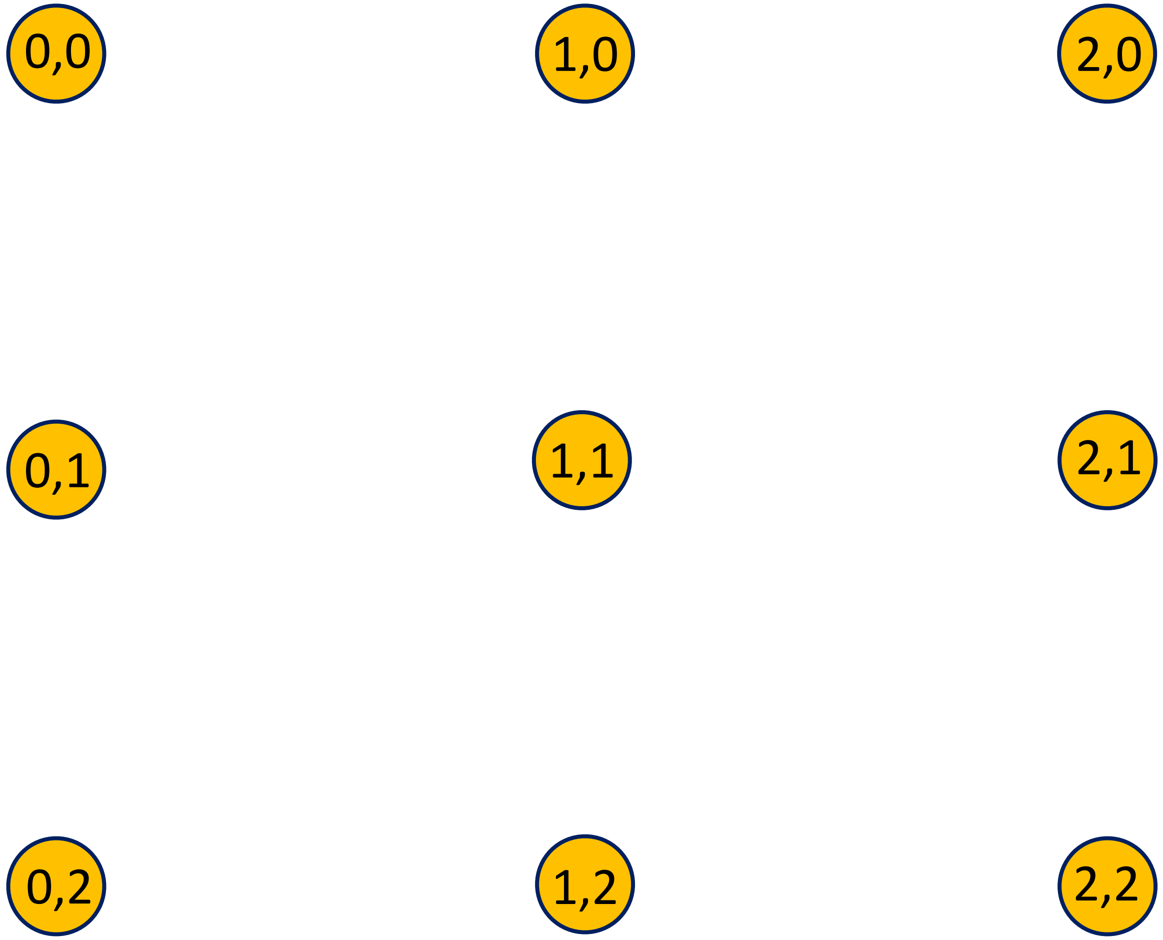



"semi-paths" of the oblivious routing scheme

[2]Leslie G. Valiant. A scheme for fast parallel communication. *SIAM J Comput. '82*

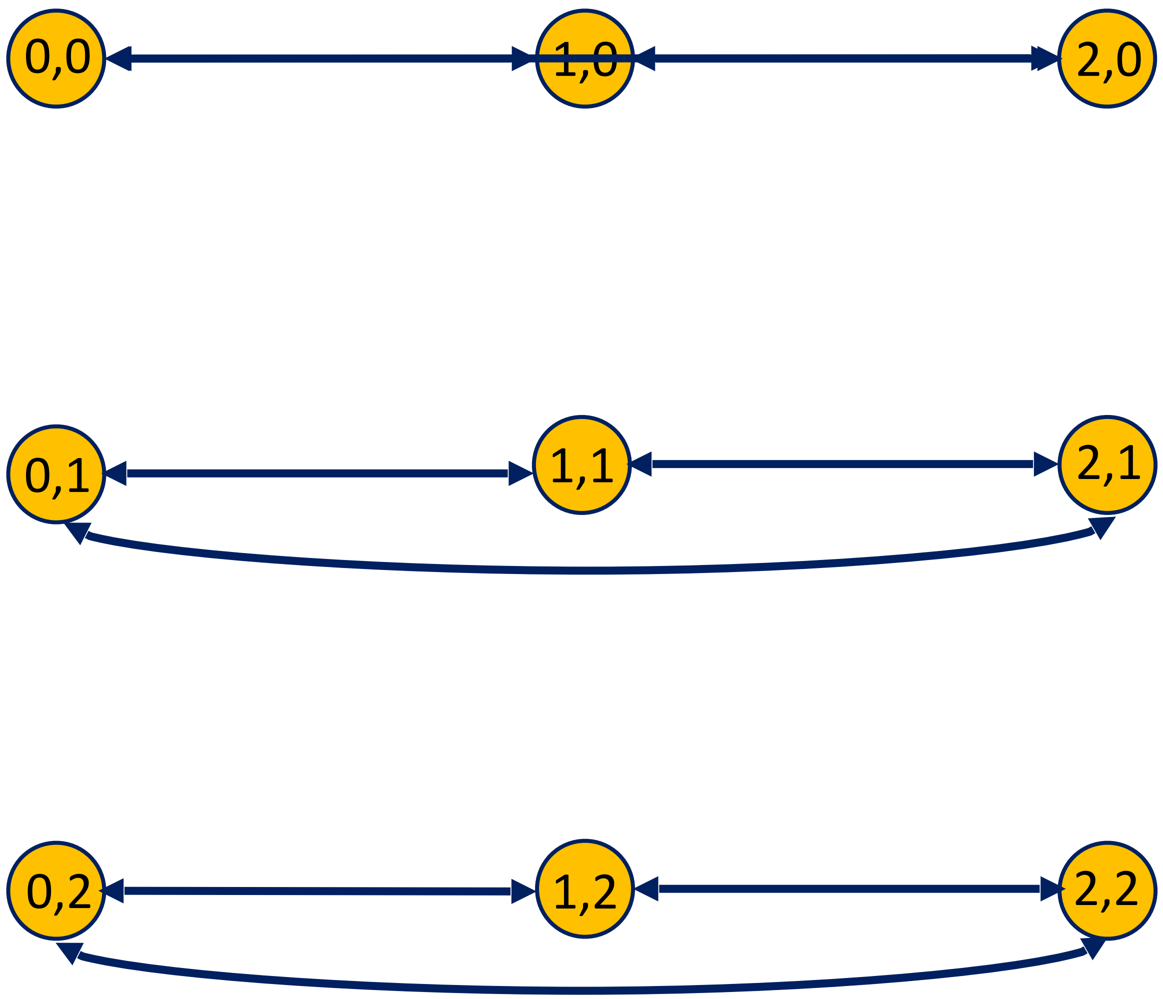
# The Elementary Basis Scheme (EBS)

$N =$ a perfect square



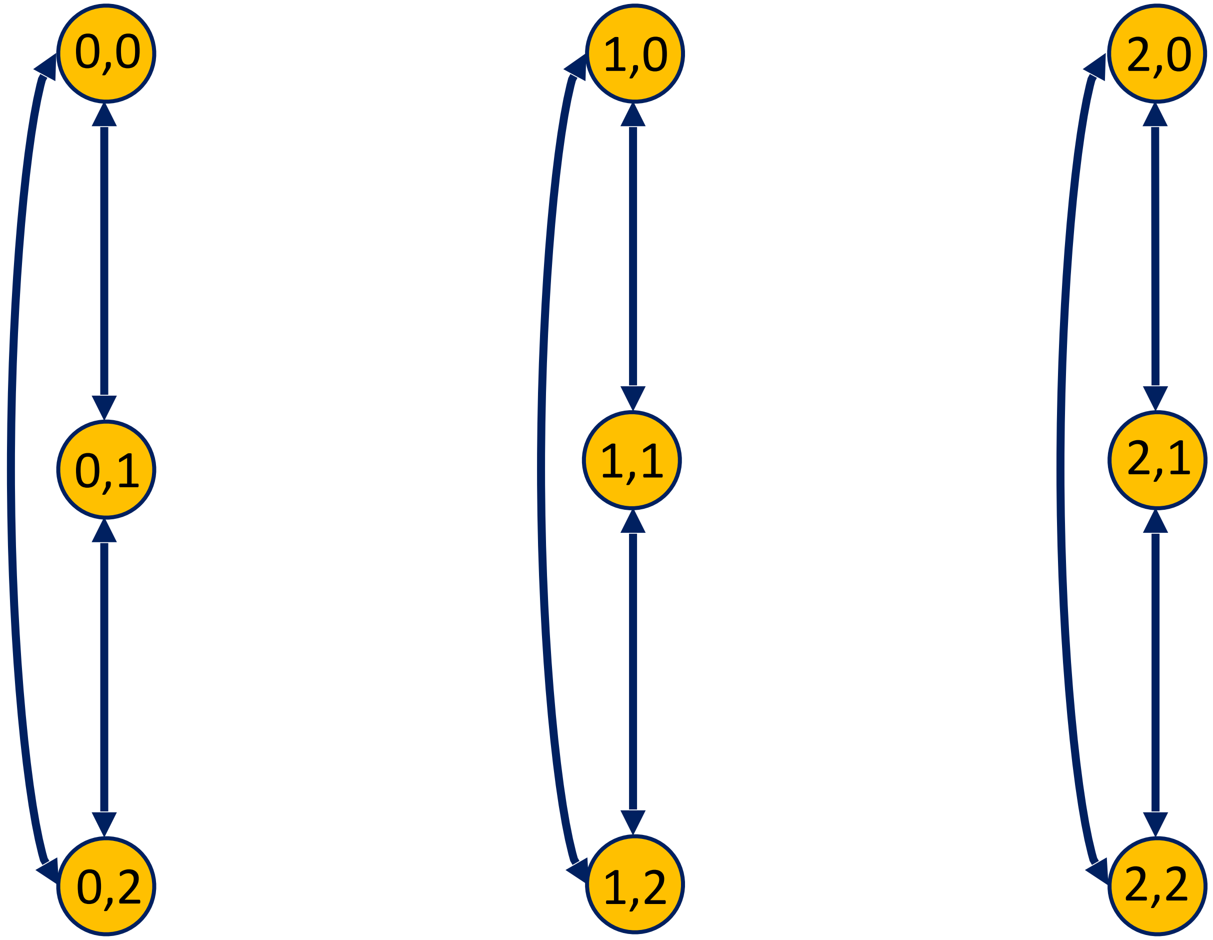
# The Elementary Basis Scheme (EBS)

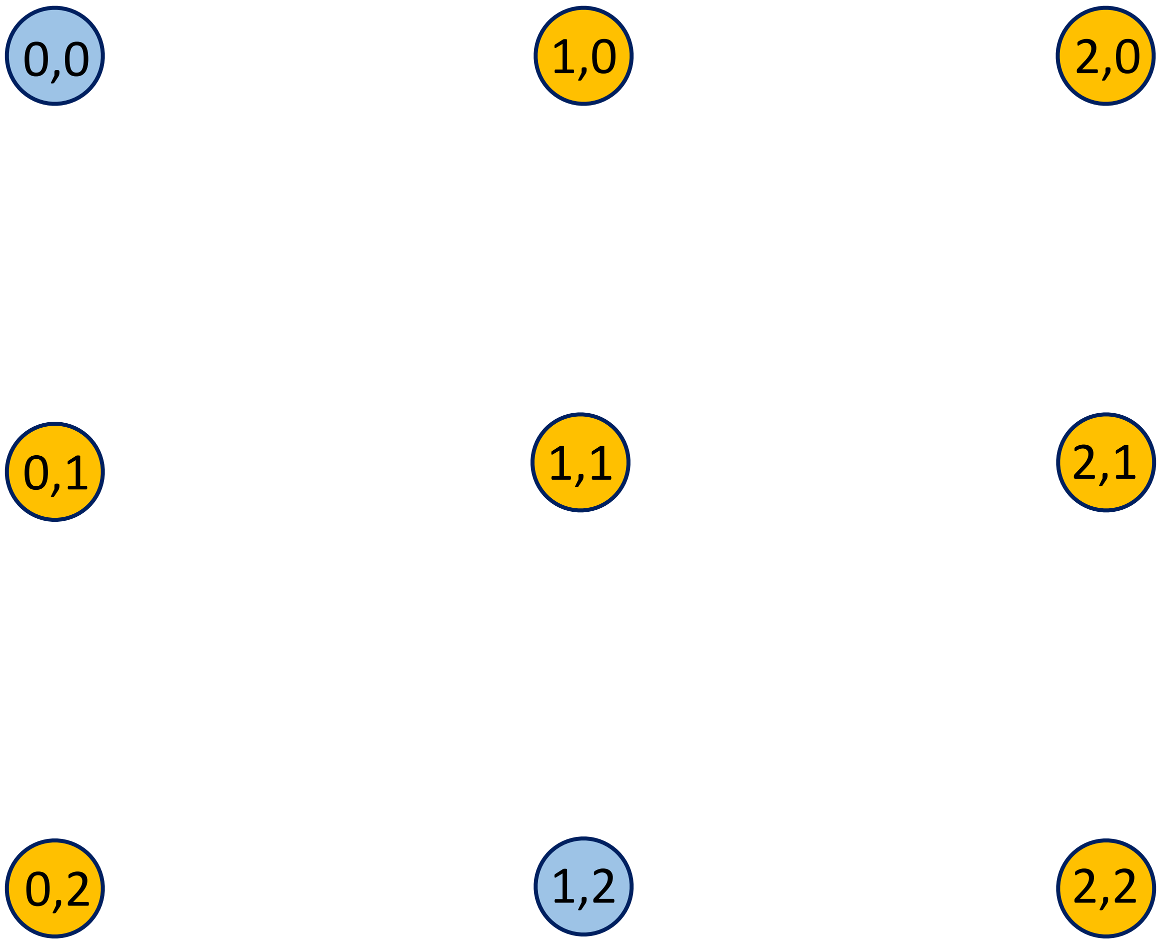
Phase 1 groups



# The Elementary Basis Scheme (EBS)

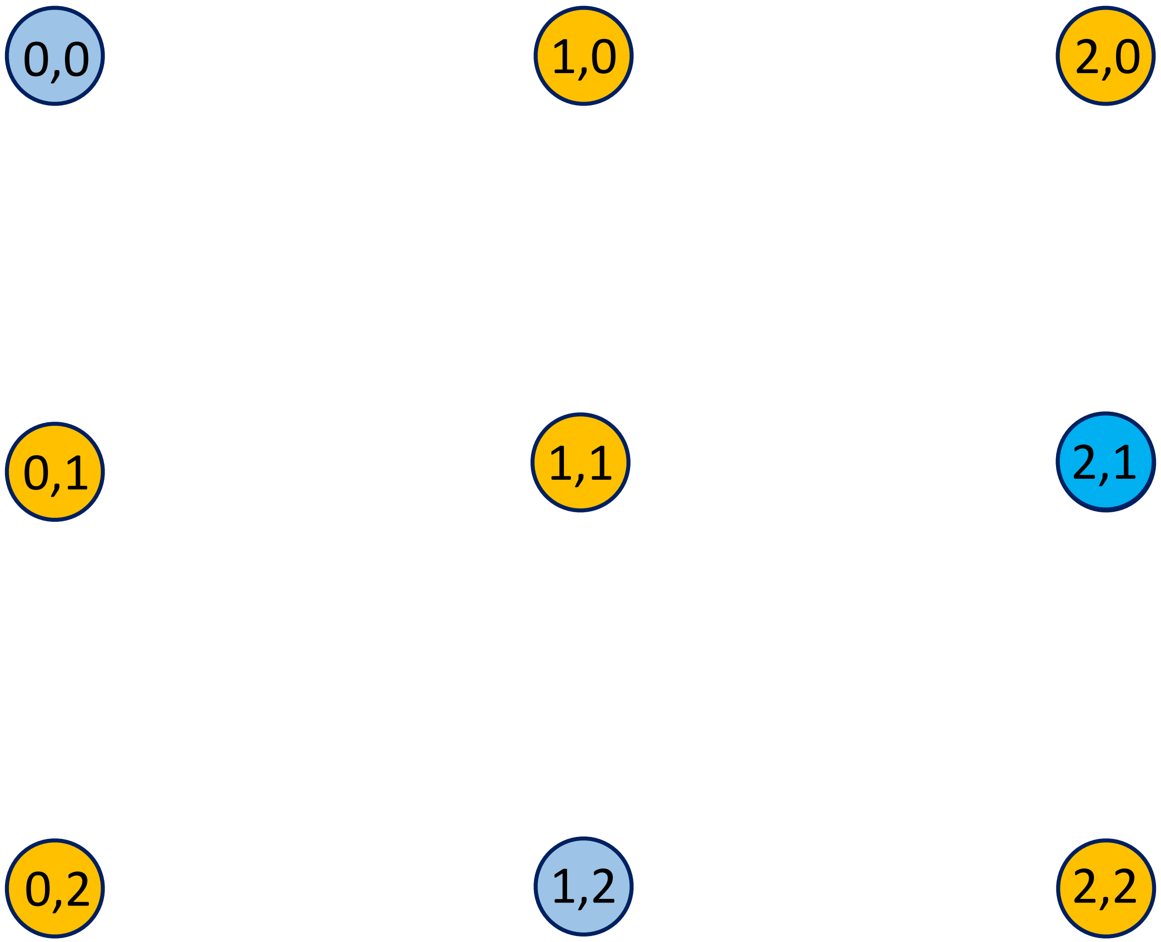
Phase 2 groups




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

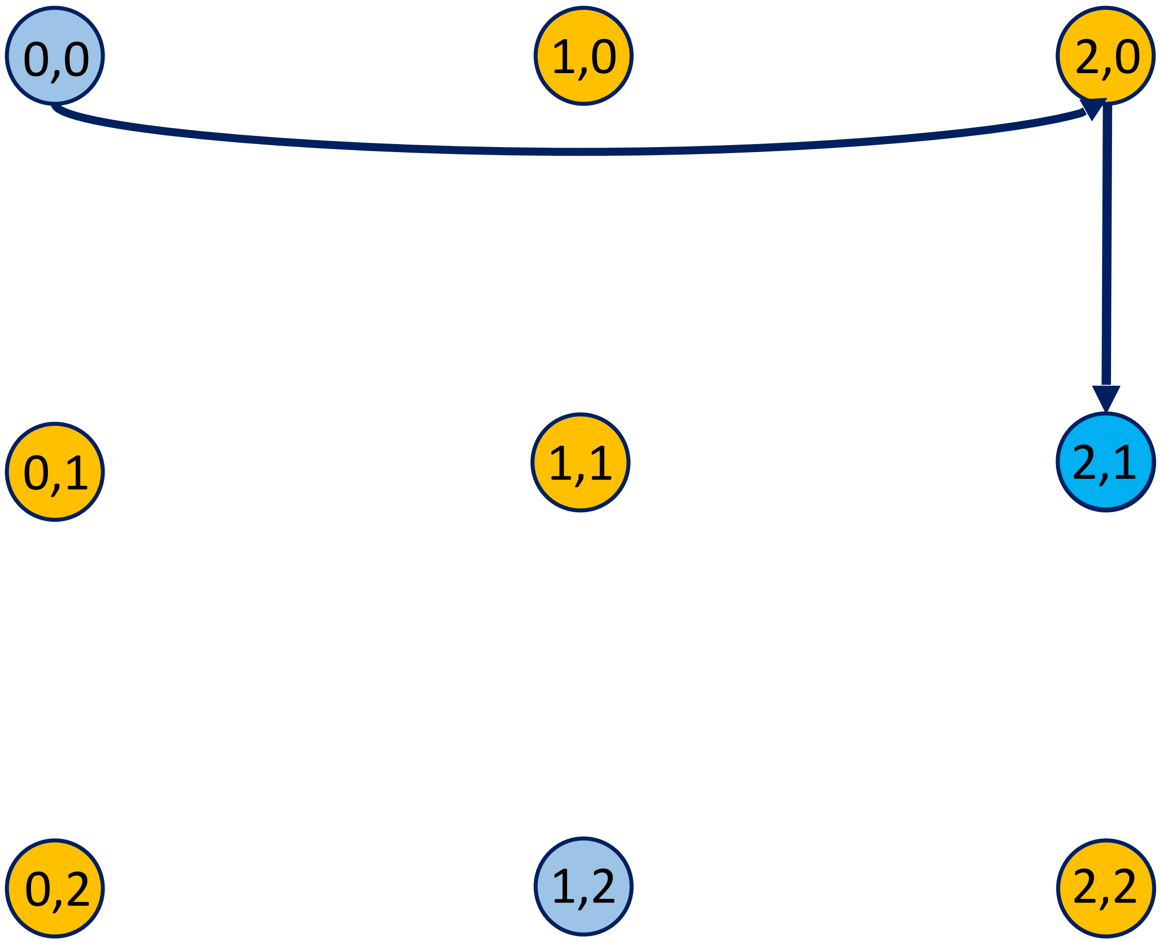
(0,0)→(1,2)

- Choose intermediate (2,1)



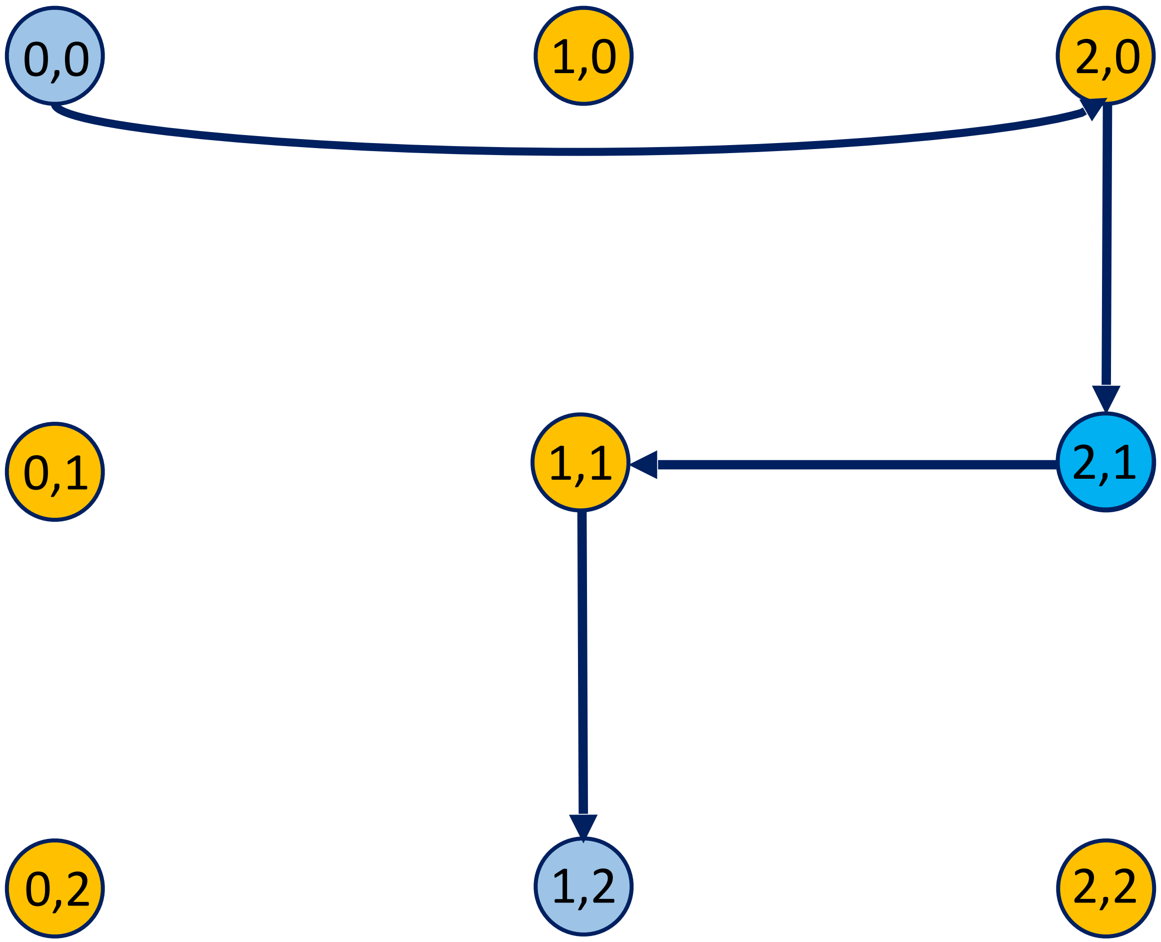
(0,0)→(1,2)

- Choose intermediate (2,1)



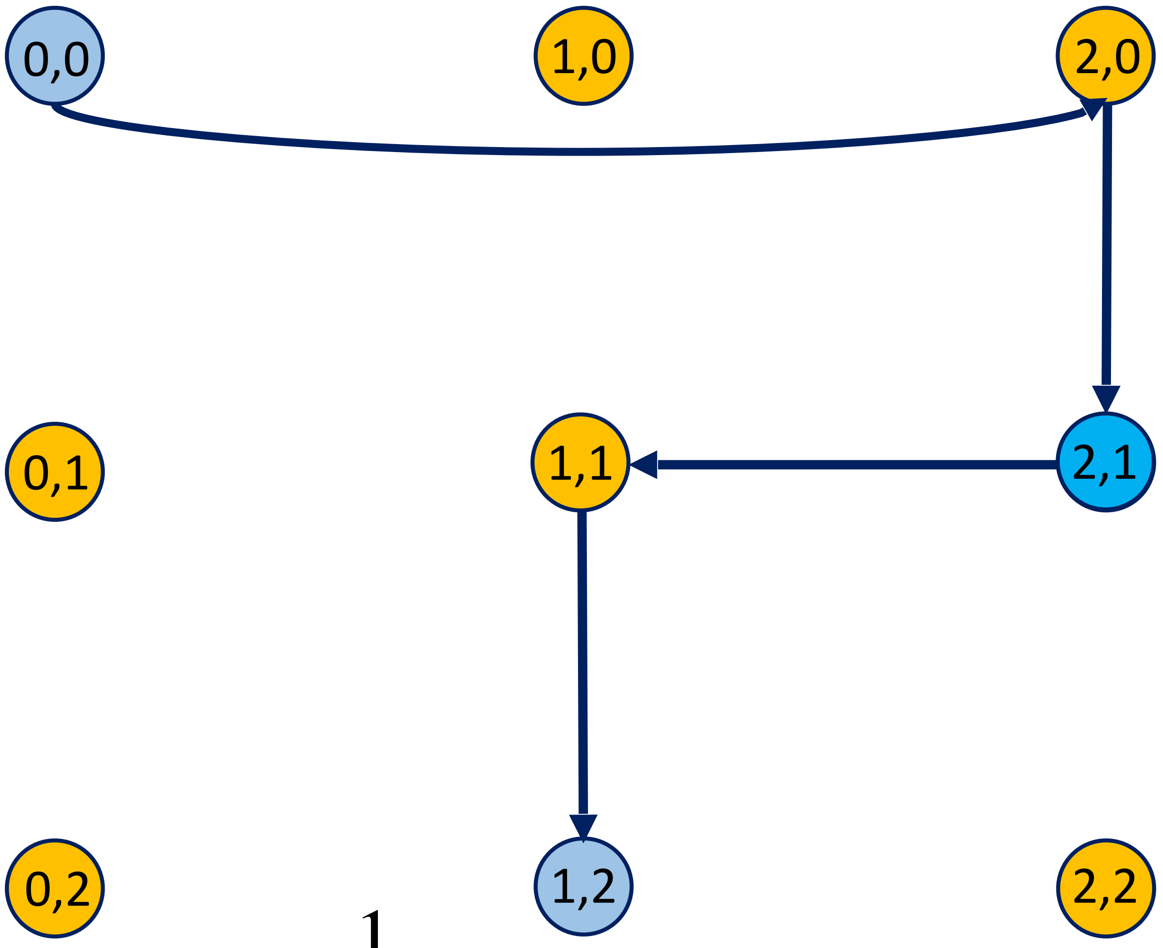
$(0,0) \rightarrow (1,2)$

- Choose intermediate $(2,1)$



(0,0)→(1,2)

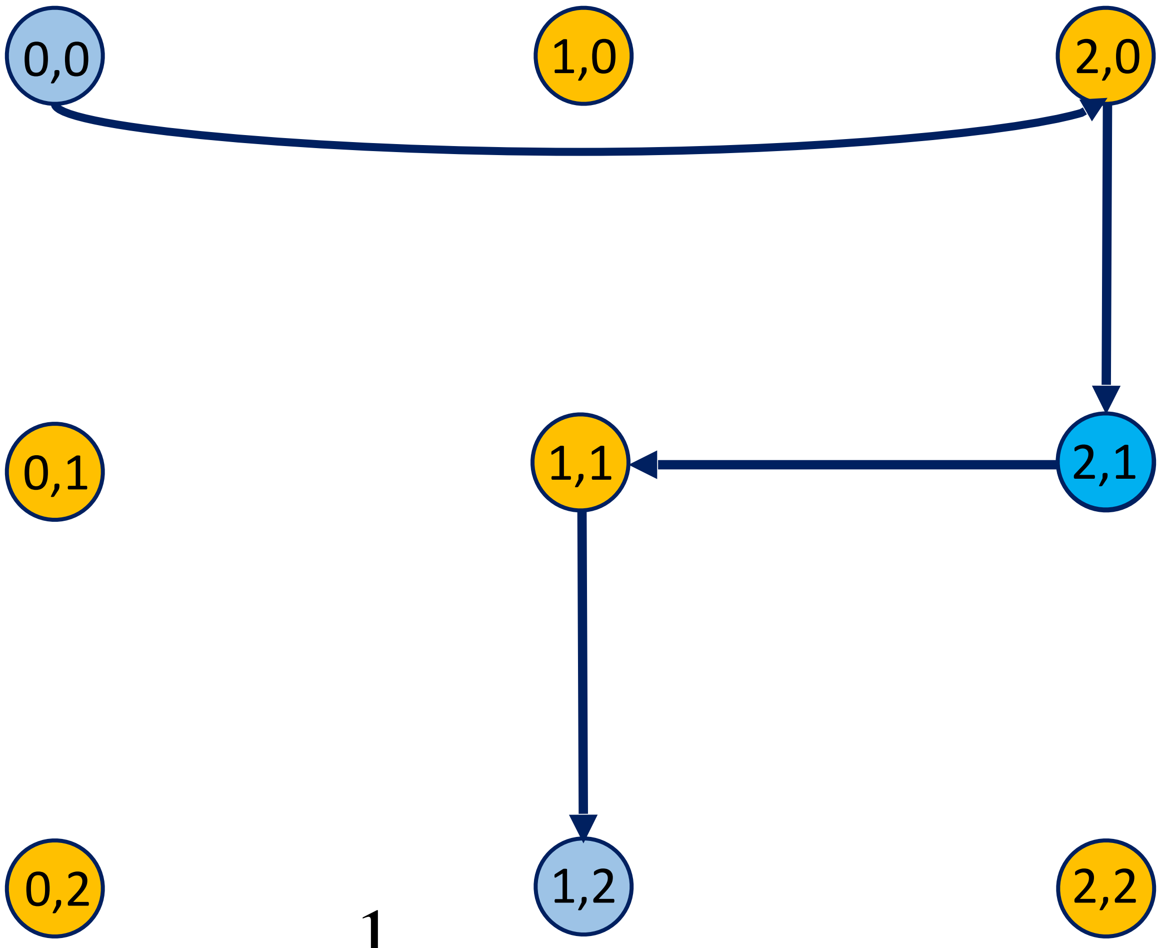
- Choose intermediate (2,1)




Guarantees throughput $r = \frac{1}{4}$

(0,0)→(1,2)

- Choose intermediate (2,1)

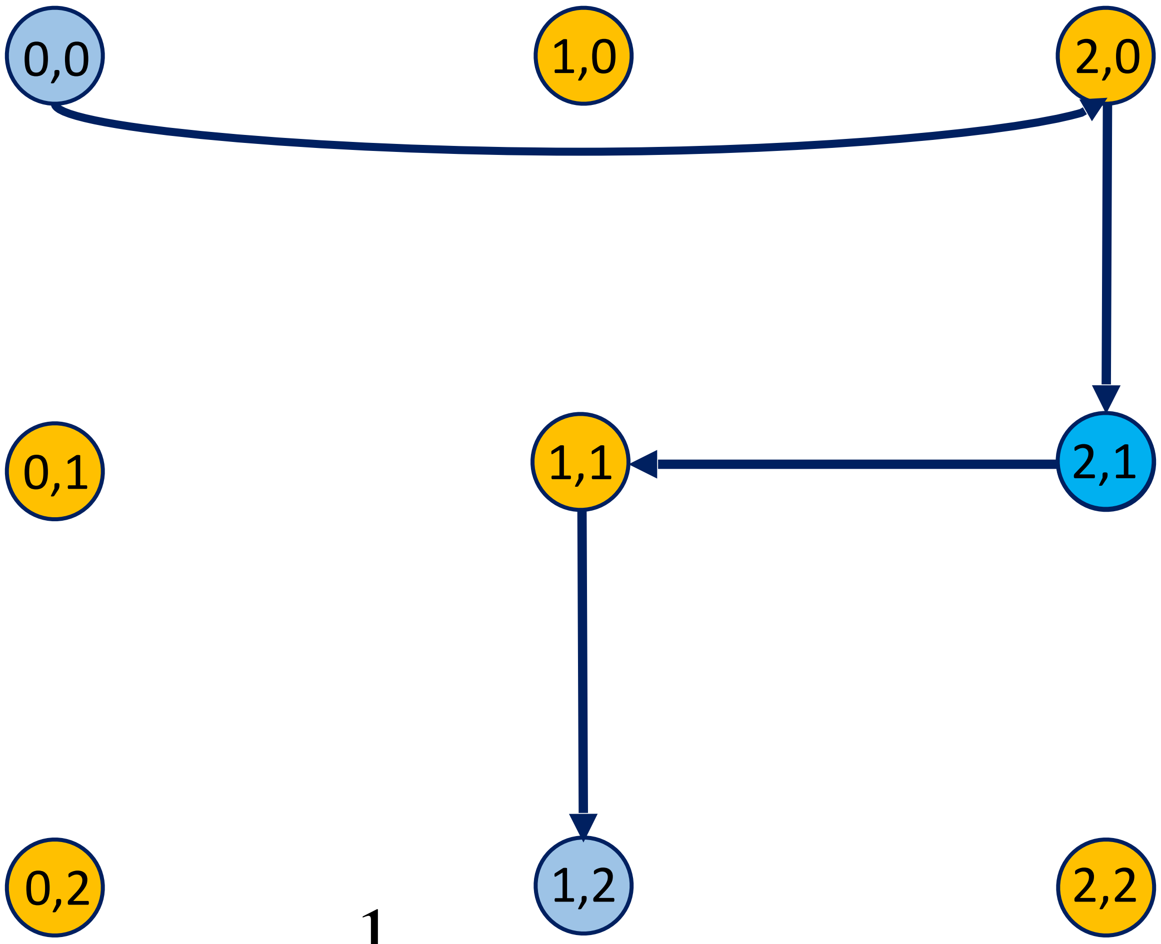



Guarantees throughput $r = \frac{1}{4}$

Max latency $L = 4\sqrt{N}$

(0,0)→(1,2)

- Choose intermediate (2,1)

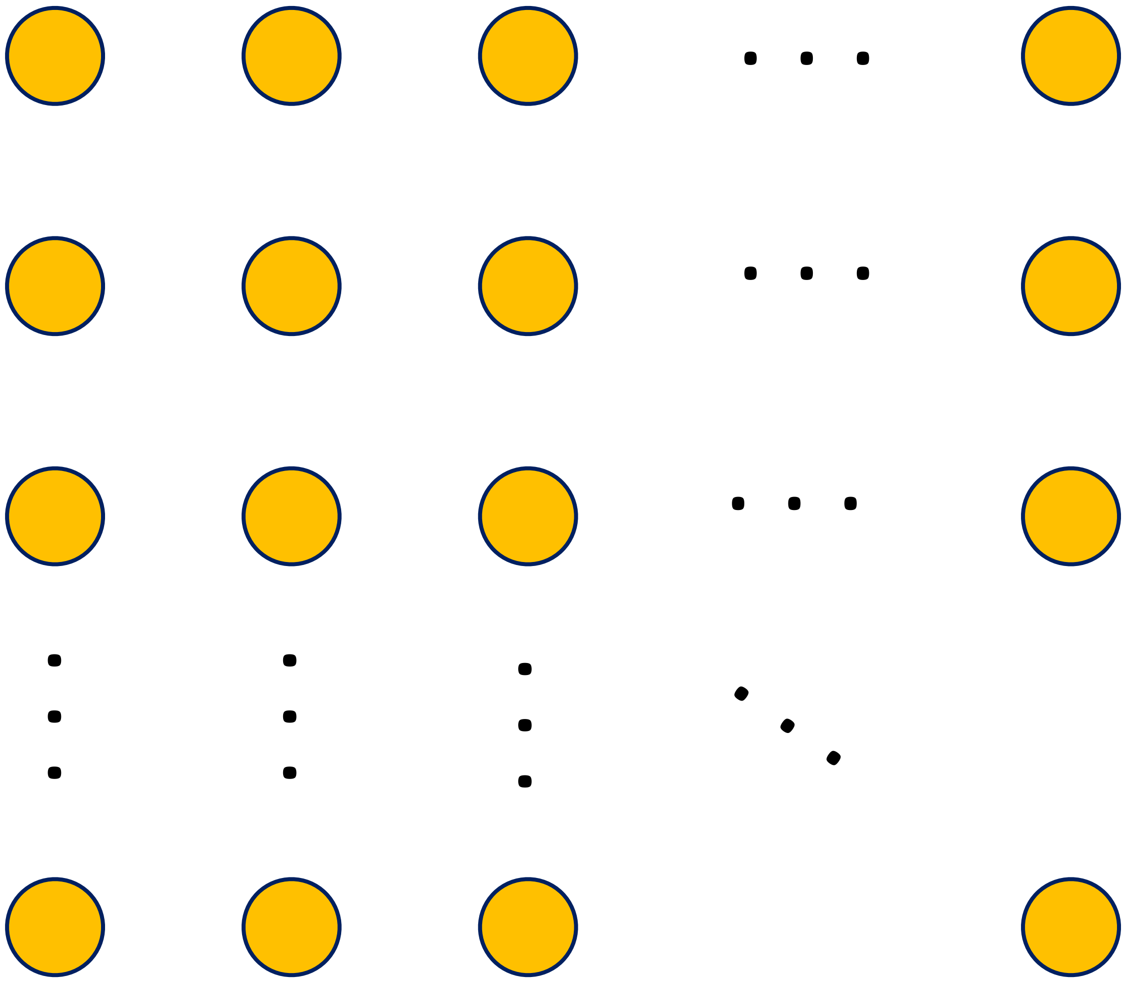


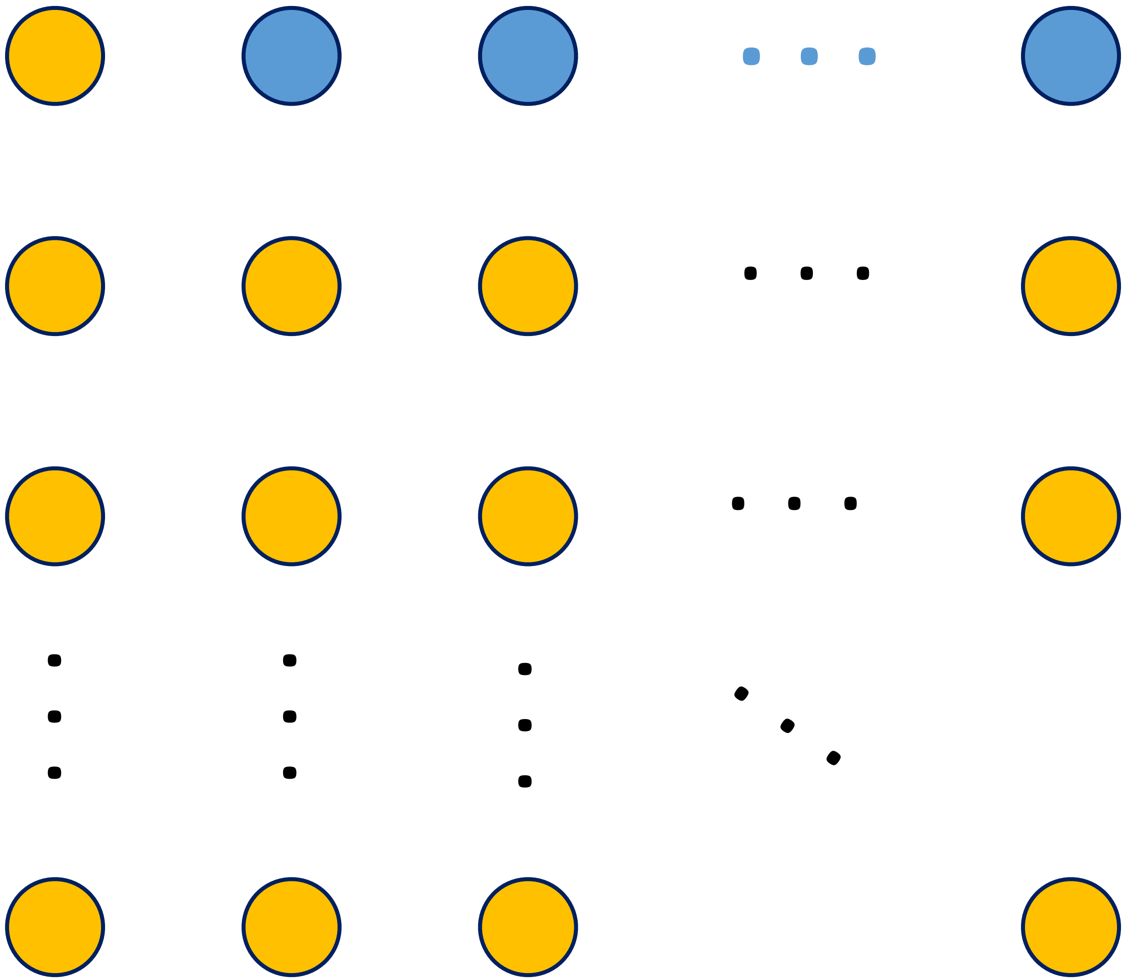

Guarantees throughput $r = \frac{1}{4}$

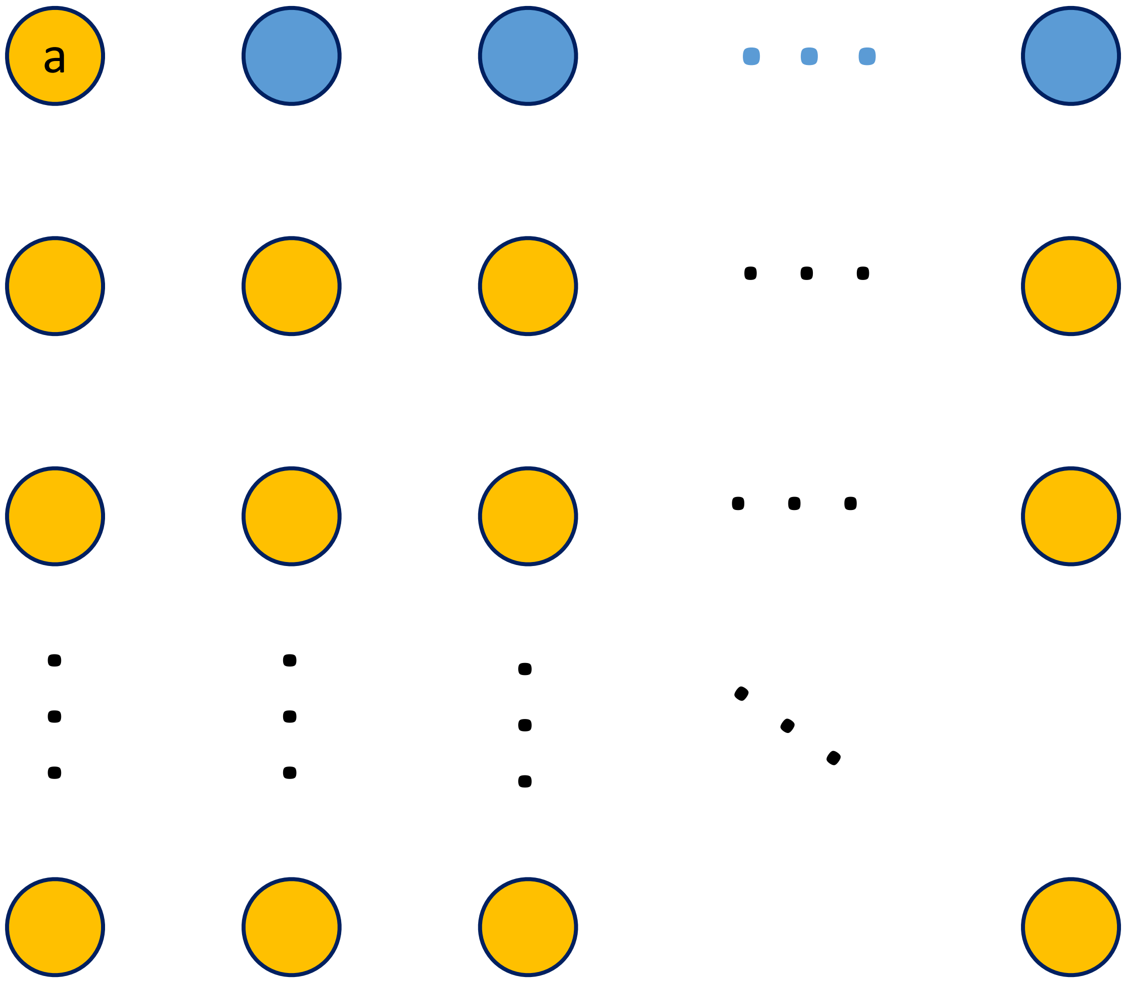
Max latency $L = 4\sqrt{N} \leq \mathrm{O}(L^*(r, N))$

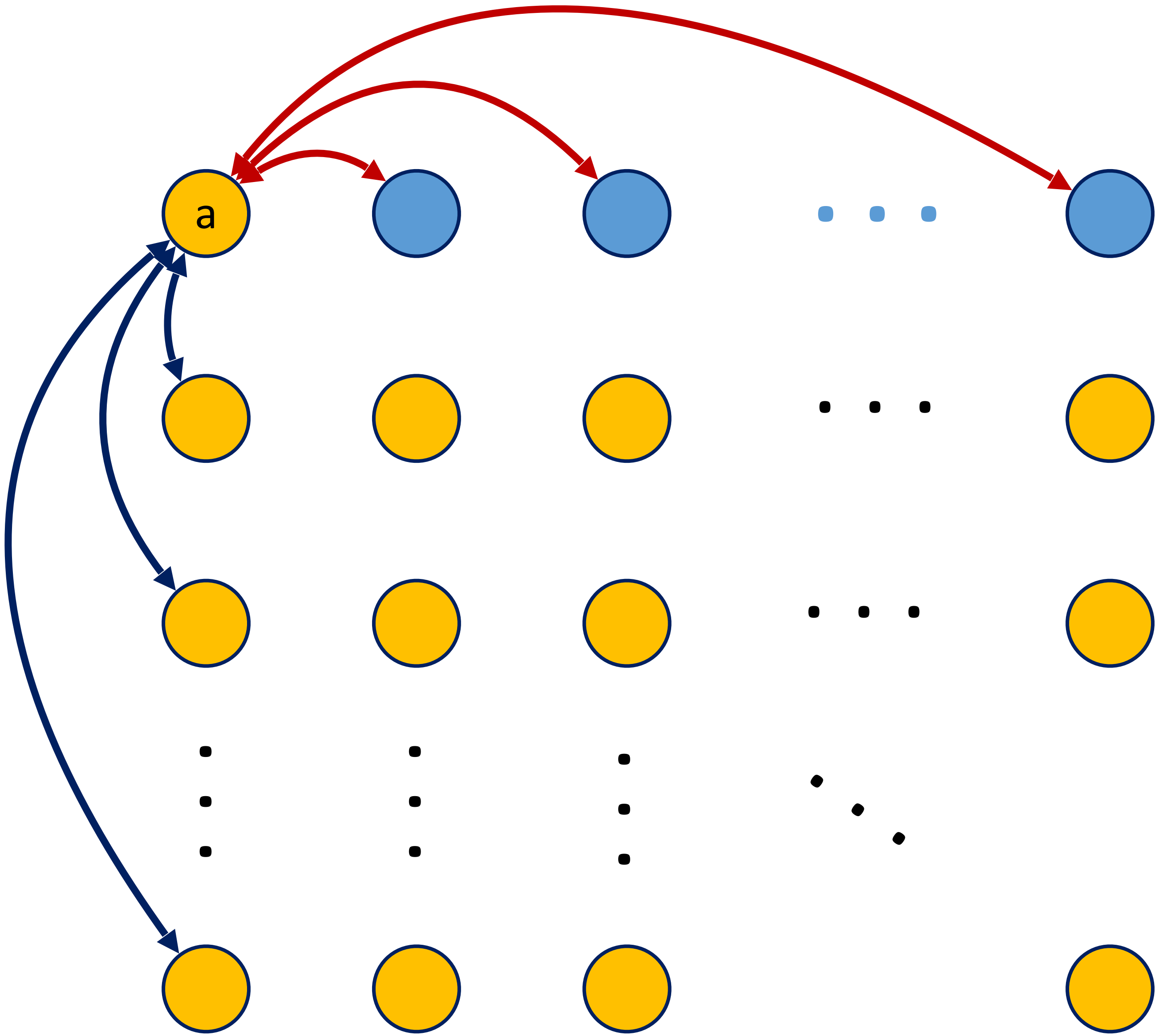
# When $N$ is Not a Perfect Square

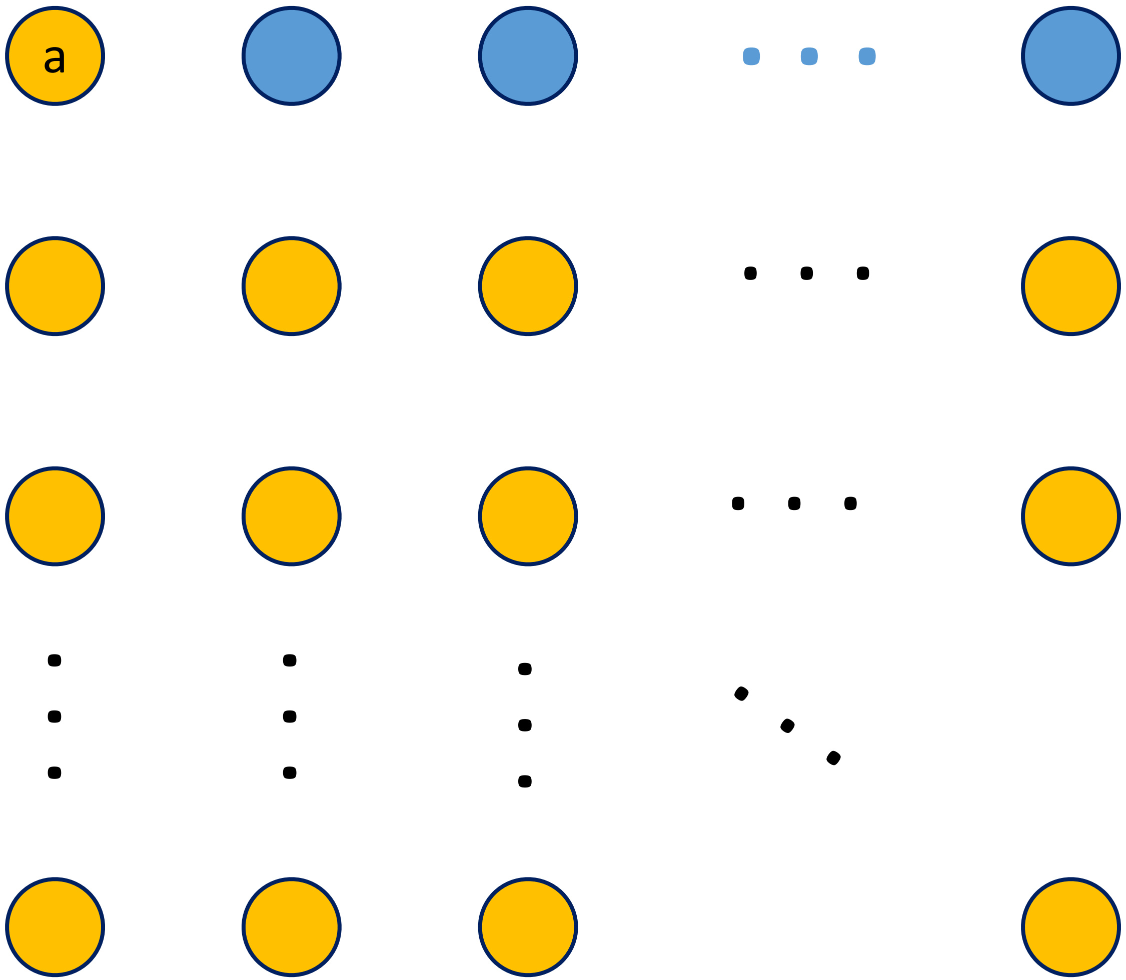
- Inflate $N$ to the next largest perfect square $M$
- Denote $(M - N)$ nodes as “dummy nodes”
- Ignore flow on routing paths that would go through dummy nodes
- Show this doesn’t decrease throughput too much

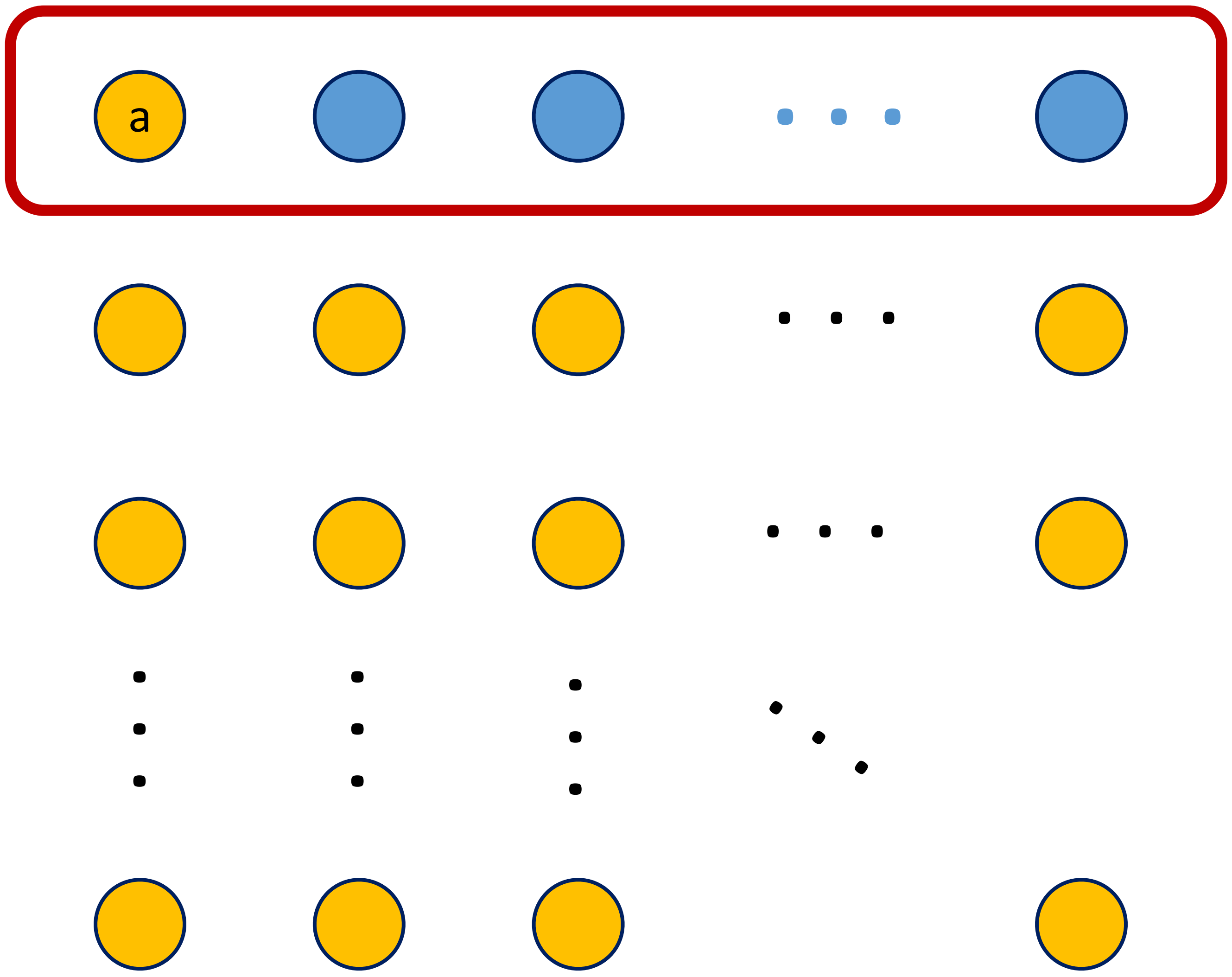


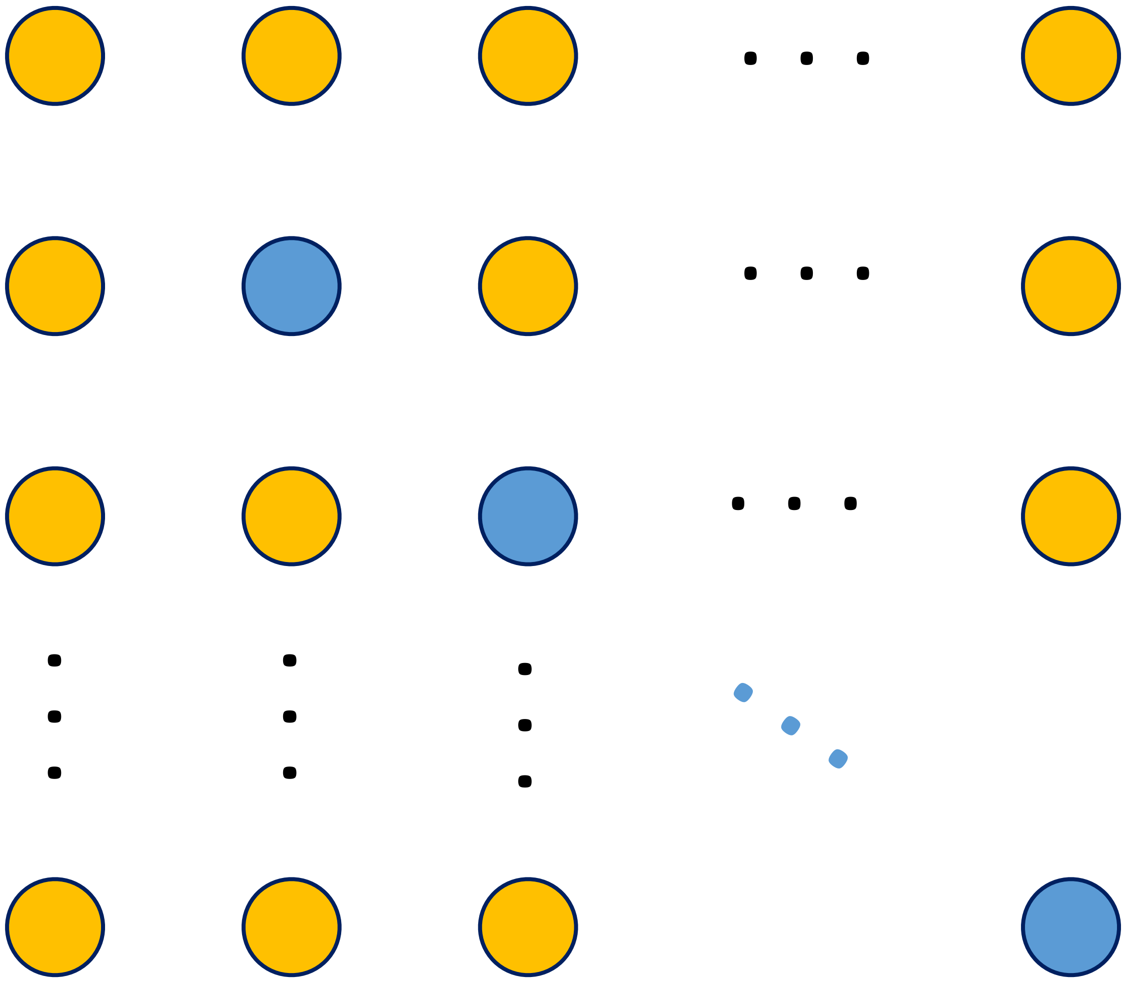


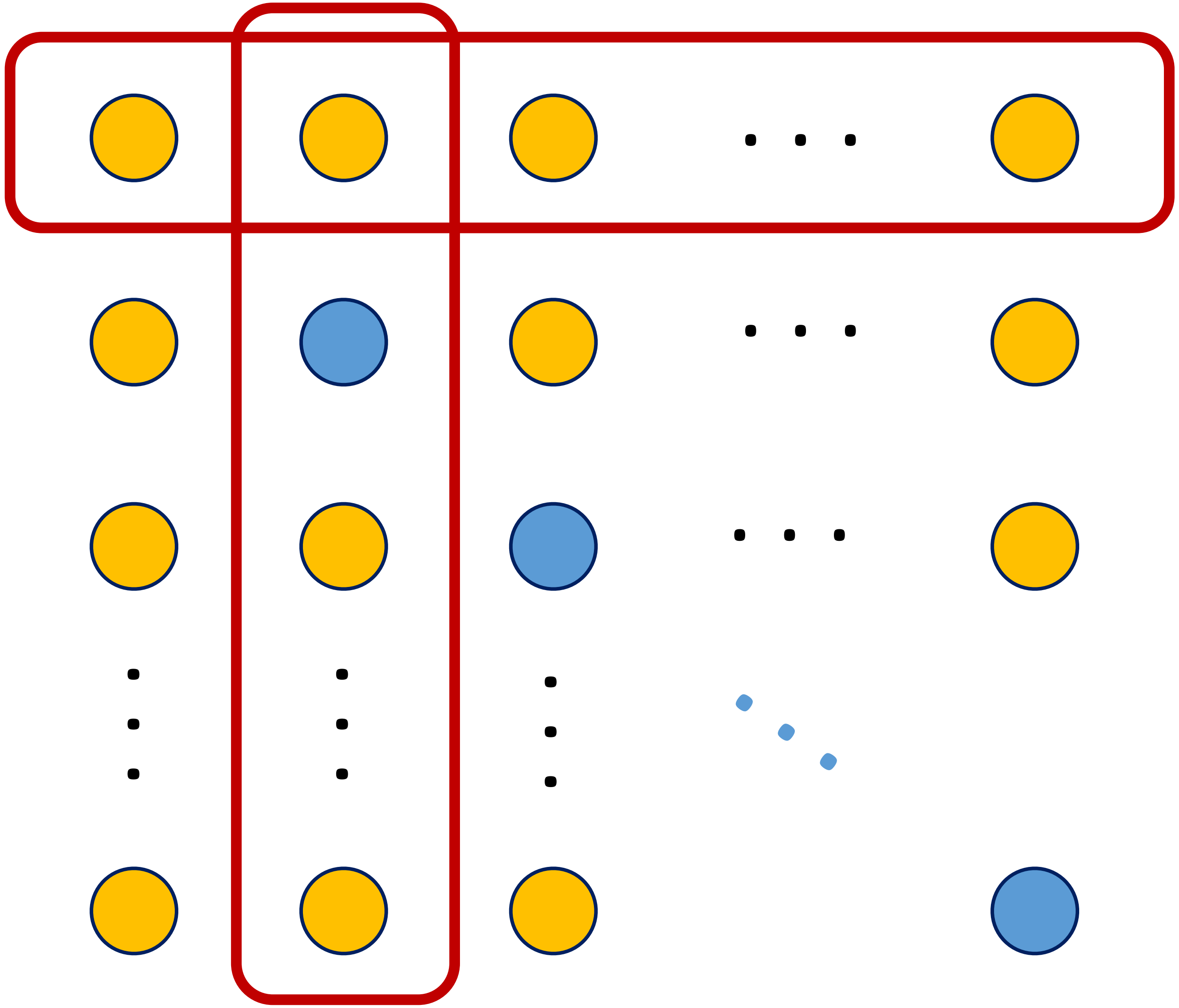

a

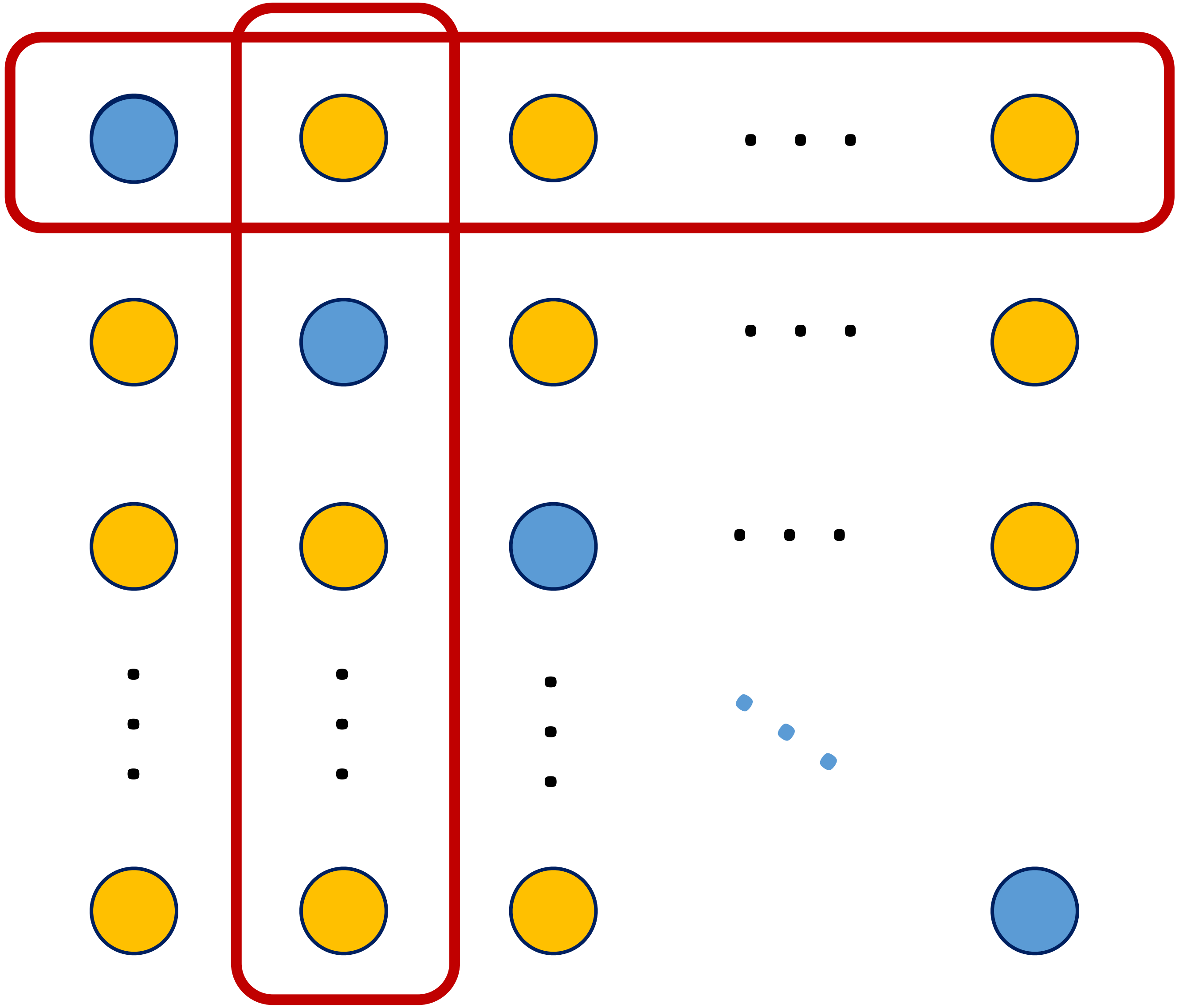

a

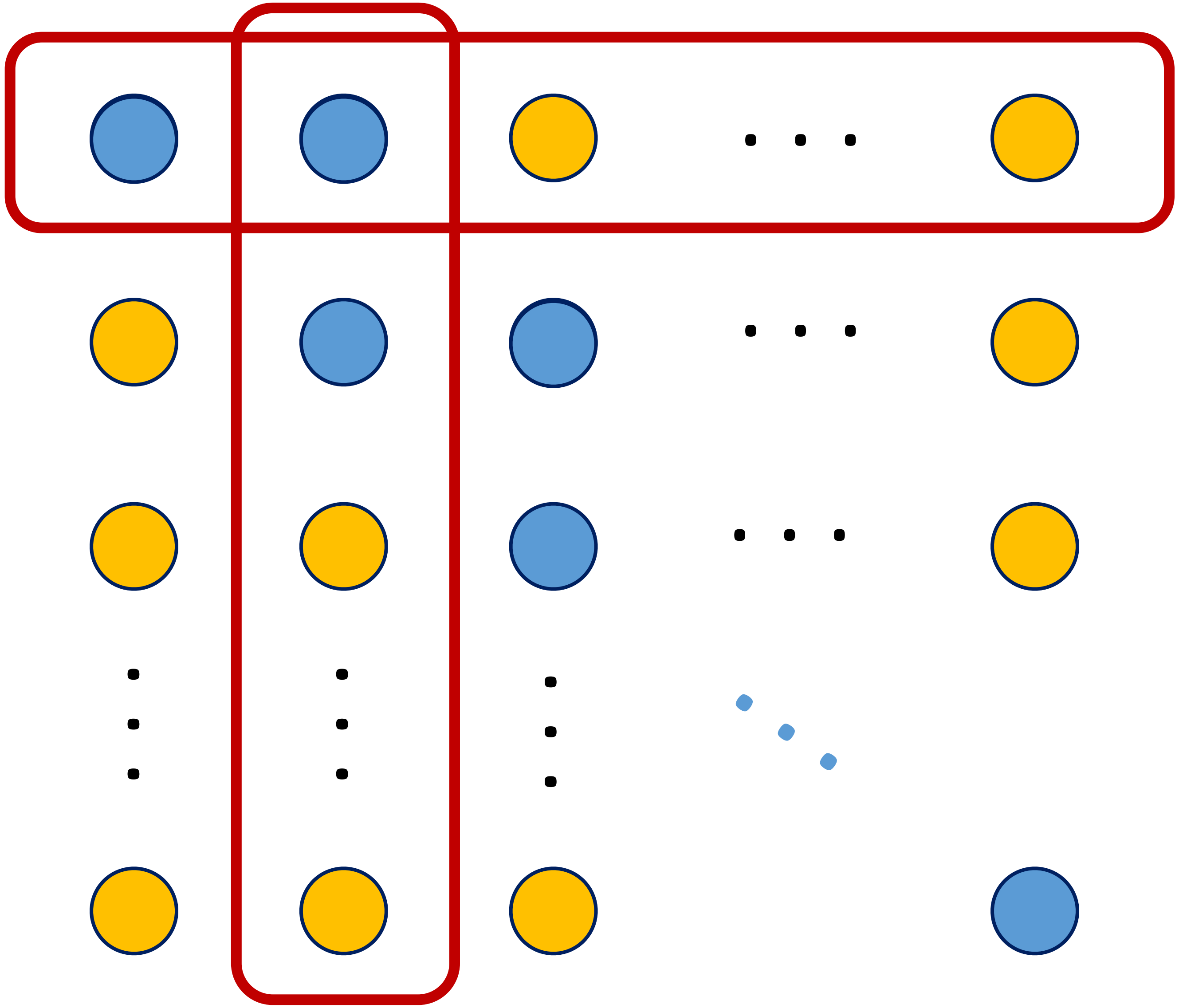

a


a









# General EBS

- $a \in [N] \rightarrow h$-tuples $\in \left[N^{1/h}\right]^h$
- Split period into $h$ phases, one for each index of the tuples
- Semi paths use $\leq 1$ hop per phase over next $h$ phases
  - Apply VLB to the semi-paths
- Guarantees throughput $\dfrac{1}{2h}$
- Max latency $2hN^{1/h} \leq \mathrm{O}(L^*\left(\dfrac{1}{2h}, N\right))$
- Achieves most optimal throughput-latency tradeoff points

# Choosing a Dummy Node Set

$$\left\{\left(i_1, i_2, \dots, i_{h-1}, \sum_{j=1}^{h-1} i_j\right) : i_1, \dots, i_{h-1} \in \left[M^{1/h}\right], \right\}$$

Exactly 1 node per phase group

# Choosing a Dummy Node Set

$$\mathcal{D} = \left\{ \left( i_1, i_2, \dots, i_{h-1}, \ell + \sum_{j=1}^{h-1} i_j \right) : i_1, \dots, i_{h-1} \in \left[ M^{1/h} \right], \ell \in [h] \right\}$$

Exactly 1 node per phase group

$h$ different diagonals

# The Vandermonde Basis Scheme (VBS)

- Defines phase connections using Vandermonde vectors
  - Allows greater flexibility in semi-path choice, allowing fine-tuning when EBS fails
- Treat nodes as vectors in an $(h+1)$-dimensional vector space over $\mathbb{F}_q$ for $q = N^{1/(h+1)}$
  - So $N$ must be a prime $(h+1)$-power
- Define "diagonal" set carefully to keep it well distributed across the Vandermonde phase groups
- Use a prime gap theorem[3] to bound number of "diagonals" we need

[3]Baker, Harman, Pintz. *The Difference Between Consecutive Primes.* Proceedings of the London Mathematical Society '01

# Putting Everything Together

- Want: guarantee throughput $r$ for arbitrary number of nodes
- Then need to build a design which can guarantee $r' > r$ throughput without dummy nodes
- This design will achieve max latency $\mathrm{O}\Big(L^*(r', M)\Big)$
- Then show that $\mathrm{O}\Big(L^*(r', M)\Big) \leq \mathrm{O}\Big(L^*(r, N)\Big)$
  - This is possible when $r$ is not $\dfrac{1}{even\ integer}$
  - Right derivative of $L^*$ is too steep at this point
  - Open Q: can we fix this?

# Future Directions & Open Problems

- Address problems that arise when you remove theoretical assumptions
  - No fractional flow $\longrightarrow$ queueing and congestion control
  - Propagation delay
- Node failures
- If we know the workload when routing, can we do better?

# Thank You!

Questions?