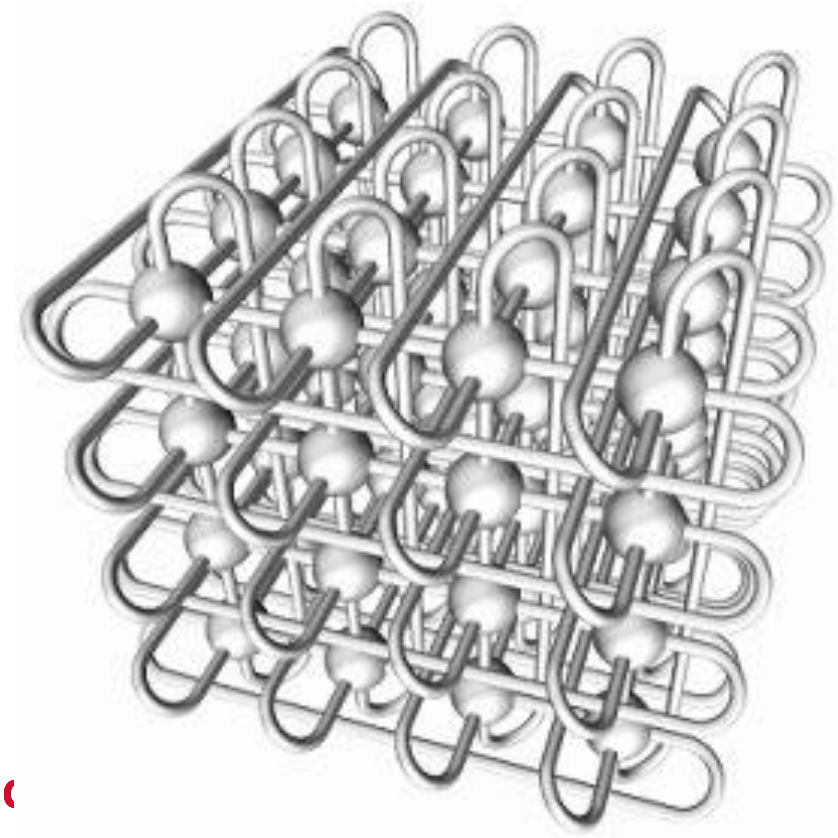
EECS 570

Interconnects: Topology + Routing I

Fall 2025

Prof. Satish Narayanasamy

http://www.eecs.umich.edu/courses/eec



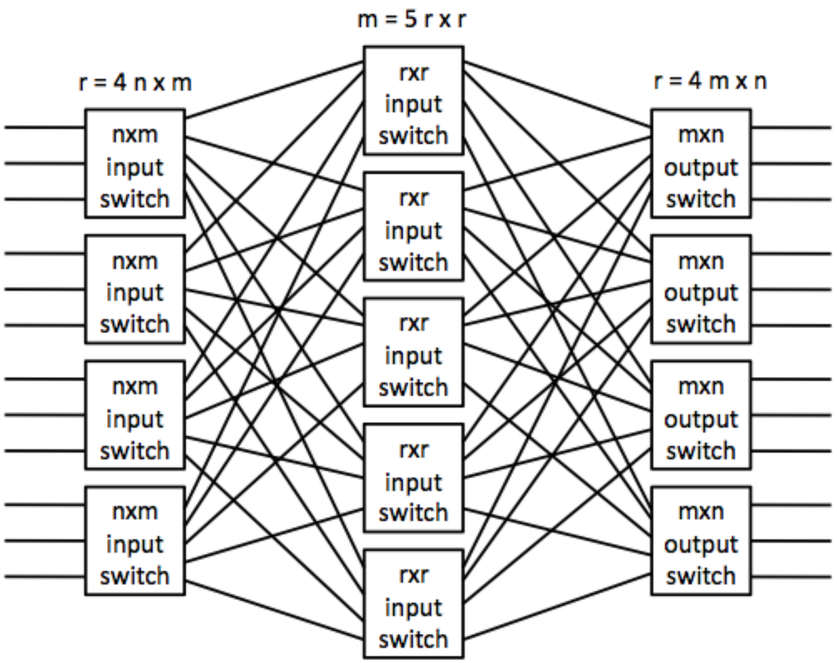
Slides developed in part by Profs. Adve, Falsafi, Hill, Lebeck, Martin, Narayanasamy, Nowatzyk, Reinhardt, Singh, Smith, Torrellas and Wenisch. Special acknowledgement to Prof. Jerger of U. Toronto.

Topics to be covered

- Interfaces
- Topology
- Routing
- Flow Control
- Router Microarchitecture

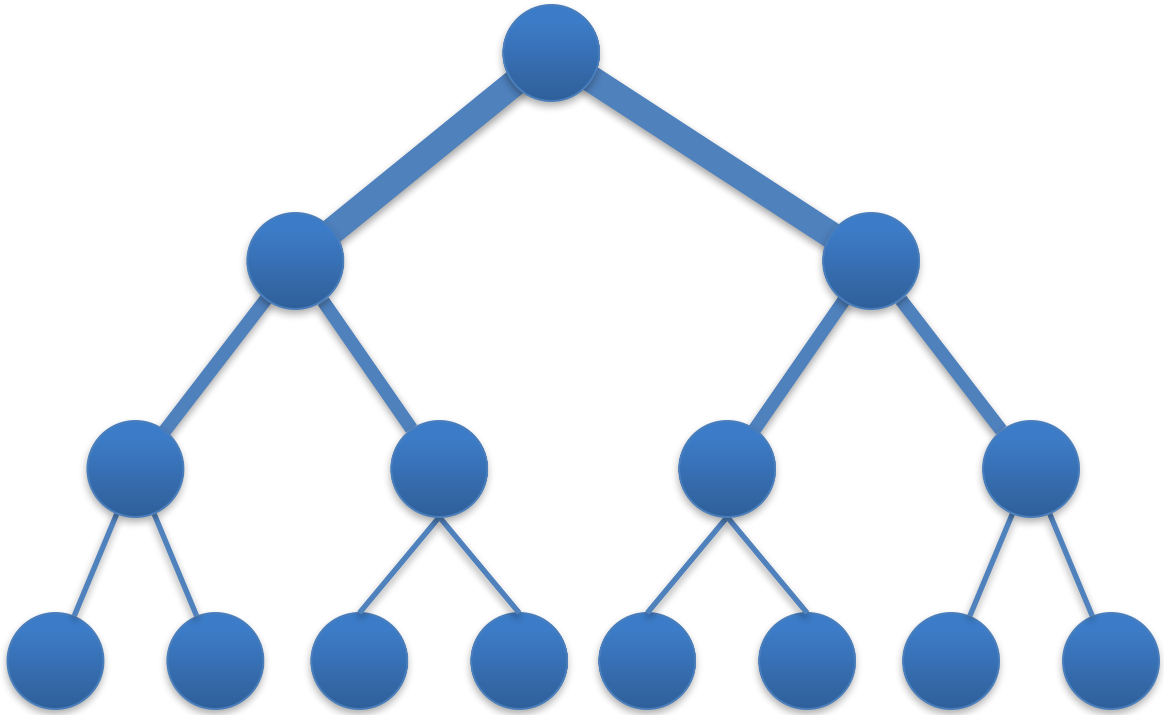
Clos network

- 3-stage networks where all input/output nodes are connected to all middle routers
- Key attribute: path diversity
  - Input node can select any middle router
  - Can enable non-blocking routing algorithms
- (m, n, r)
  - m = Number of middle stage switches
  - n = input/output ports per input/output switch
  - r = number of input/output switches



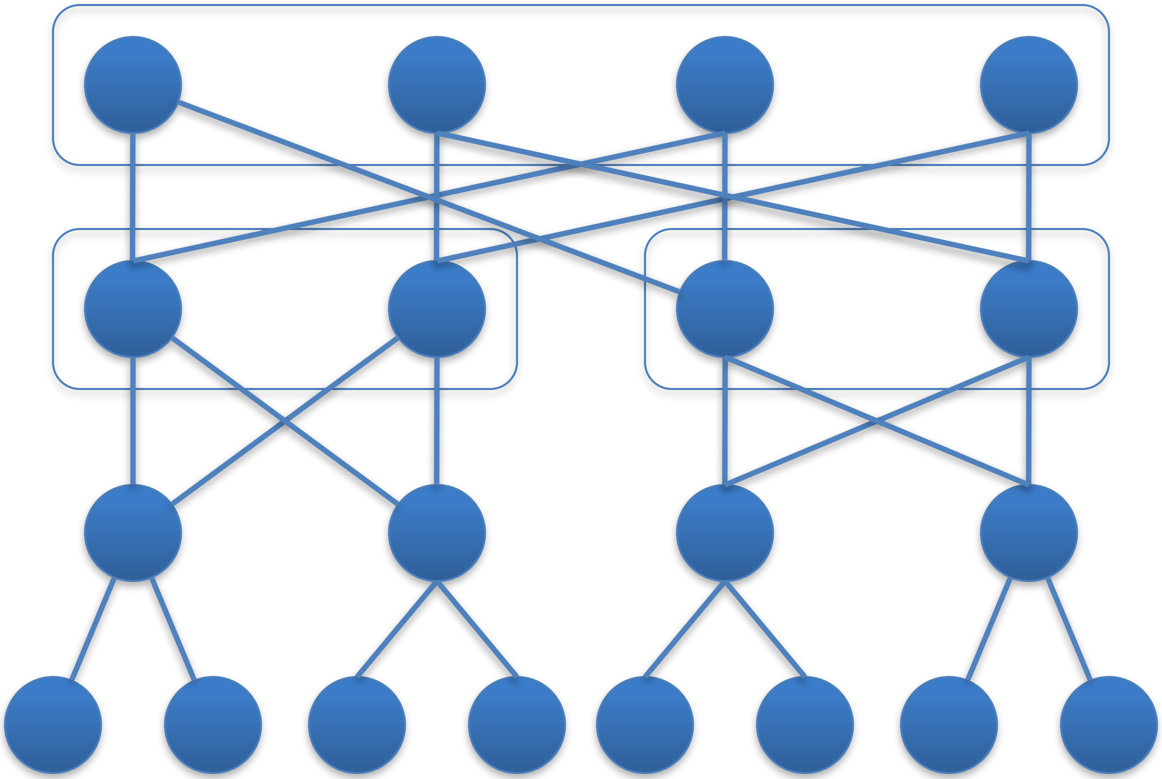
(5,3,4) Clos network

Fat Tree



- Bandwidth remains constant at each level
- Regular Tree: Bandwidth decreases closer to root

Fat Tree (2)



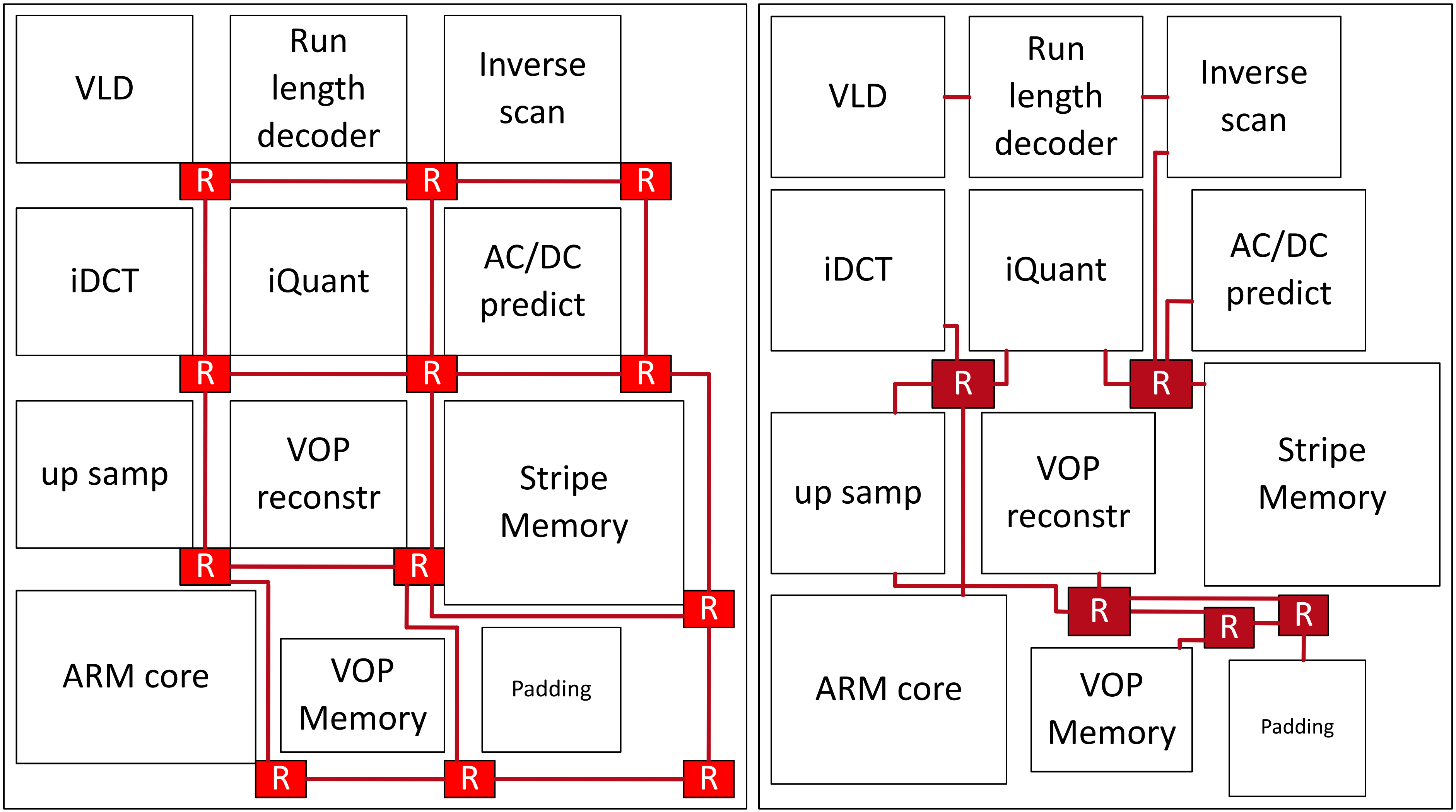
- Can be constructed from folded Clos
- Provides path diversity

Irregular Topologies

Irregular Topologies

- MPSoC design leverages wide variety of IP blocks
  - Regular topologies may not be appropriate given heterogeneity
  - Customized topology
    - Often more power efficient and deliver better performance
- Customize based on traffic characterization

Irregular Topology Example



Topology Customization

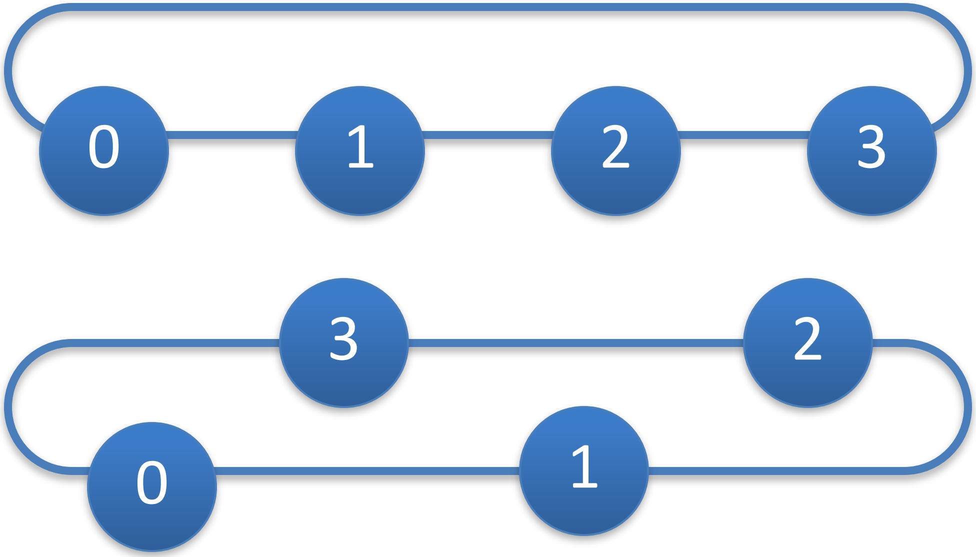
- Merging
  - Start with large number of switches
  - Merge adjacent routers to reduce area and power

- Splitting
  - Large crossbar connecting all nodes
  - Iteratively split into multiple small switches
    - Accommodate design constraints

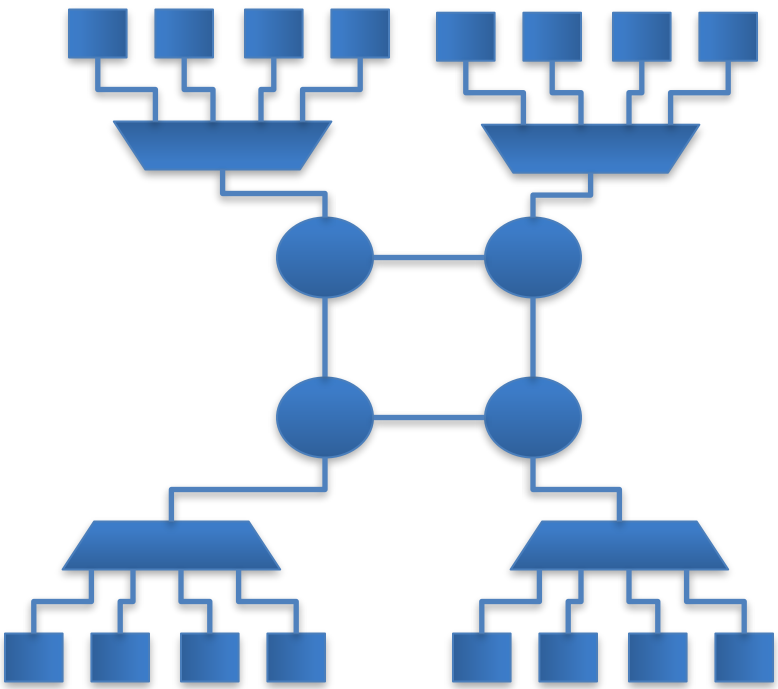
Topology Implementation

Implementation

- Folding
  - Equalize path lengths
    - Reduces max link length
    - Increases length of other links



Concentration



- Don't need 1:1 ratio of routers to cores
  - ❒ Ex: 4 cores concentrated to 1 router
- Can save area and power
- Increases network complexity
  - ❒ Concentrator must implement policy for sharing injection bandwidth
  - ❒ During bursty communication
    - ❍ Can bottleneck

Implication of Abstract Metrics on Implementation

- Degree: useful proxy for router complexity
  - Increasing ports requires additional buffer queues, requestors to allocators, ports to crossbar
  - All contribute to critical path delay, area and power
  - Link complexity does not correlate with degree
    - Link complexity depends on link width
    - Fixed number of wires, link complexity for 2-port vs 3-port is same

Implications (2)

- Hop Count: useful proxy for overall latency and power
  - Does not always correlate with latency
    - Depends heavily on router pipeline and link propagation
  - Exam
    - N k traversal vs
    - N k traversal

Hop Count says A is better than B
But A has 18 cycle latency vs 6 cycle latency for B

Topology Summary

- First network design decision

- Critical impact on network latency and throughput
  - Hop count provides first order approximation of message latency

  - Bottleneck channels determine saturation throughput

Routing

Routing Overview

- Discussion of topologies assumed ideal routing

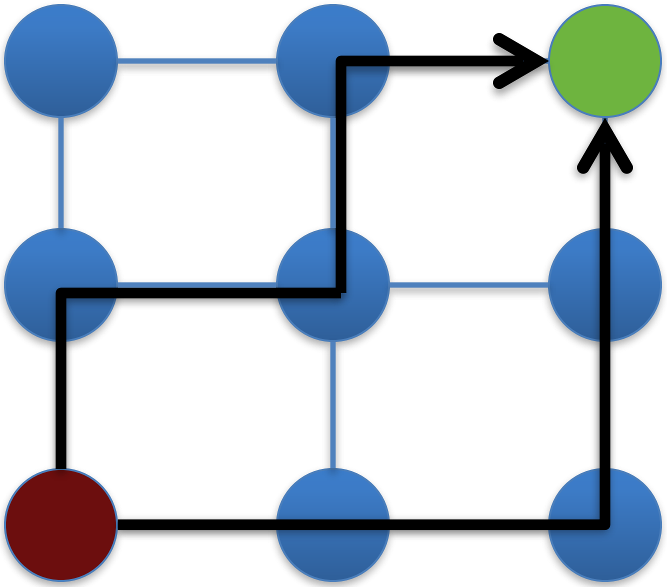
- In practice…
  - Routing algorithms are not ideal

- Goal: distribute traffic **evenly** among paths
  - Avoid hot spots, contention
  - More balanced → closer throughput is to ideal

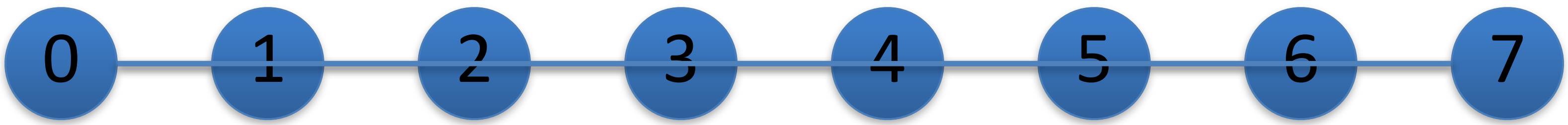
- Keep complexity in mind

Routing Basics

- Once topology is fixed
- Routing algorithm determines path(s) from source to destination



Routing Example

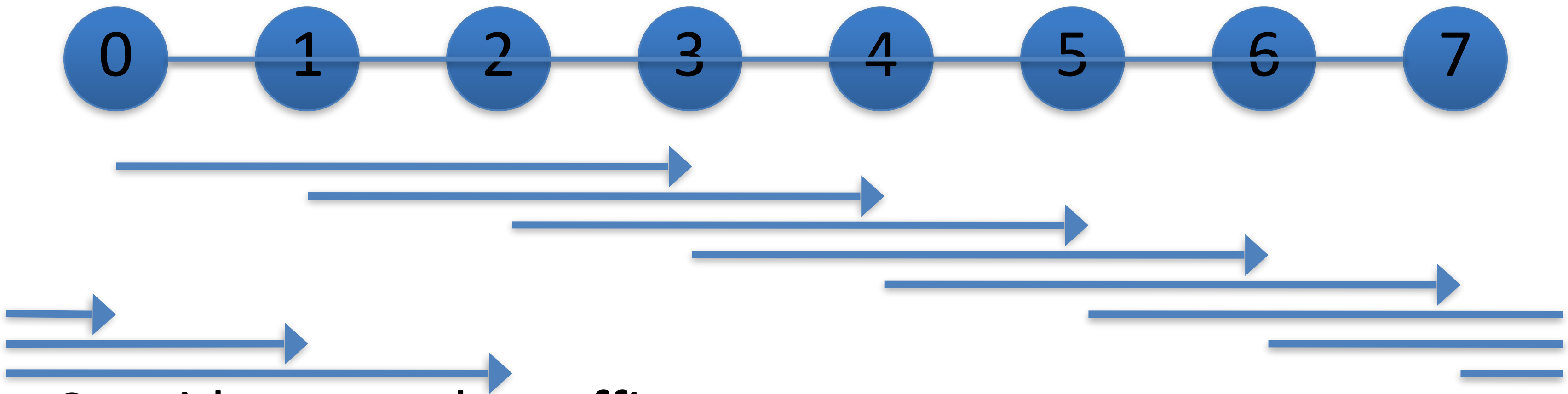


- Some routing options:
  - Greedy: shortest path
  - Uniform random: randomly pick direction
  - Adaptive: send packet in direction with lowest local channel load

- Which gives best worst-case throughput?

1. **Greedy (shortest path)**:
    - Always takes the path with the least number of hops.
    - **Weakness**: Can overload central or commonly used paths, creating bottlenecks.
    - **Worst-case throughput** is not great—too much congestion on shortest paths.
2. **Uniform random**:
    - Routes packets randomly.
    - **Strength**: Avoids overload on a single path.
    - **Weakness**: Inefficient and unpredictable. May take long or inefficient routes.
    - **Worst-case throughput** is unpredictable and often low.
3. **Adaptive (lowest local channel load)**:
    - Chooses paths based on current traffic, trying to avoid congestion.
    - **Strength**: Dynamically balances load and avoids bottlenecks.
    - **Worst-case throughput** is best because it actively avoids congestion.

Routing Example (2)



- Consider tornado traffic
  - node *i* sends to *i+3 mod 8*

Routing Example (3)

- Greedy:
  - All traffic moves counterclockwise
    - Loads counterclockwise with 3 units of traffic
      - Each node gets 1/3 throughput
    - Clockwise channels are idle
- Random:
  - Clockwise channels become bottleneck
    - Load of 5/2
      - Half of traffic traverses 5 links in clockwise direction
      - Gives throughput of 2/5

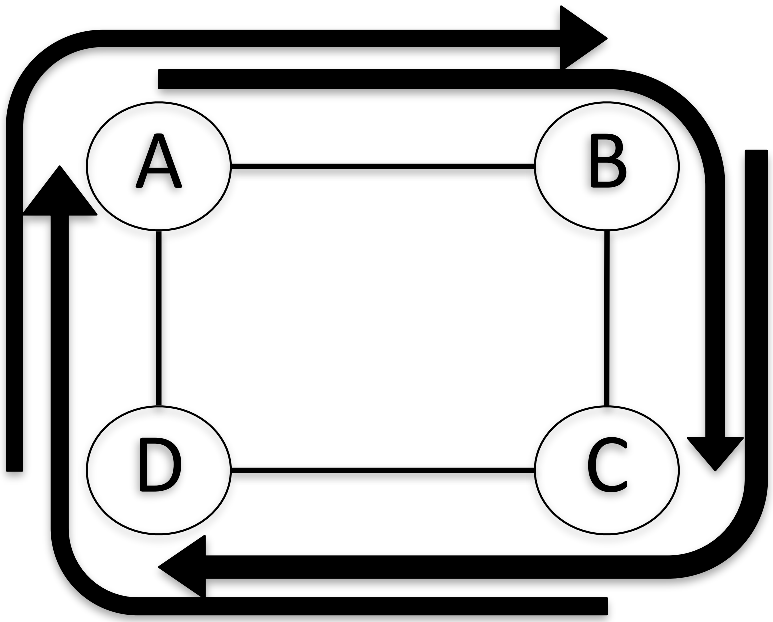
Routing Example (4)

- Adaptive:
  - Perfect load balancing (some assumptions about implementation)
  - Sends 5/8 of traffic over 3 links, sends 3/8 over 5 links
    - Channel load is 15/8, throughput of 8/15
- Note: worst case throughput just 1 metric designer might optimize

Routing Algorithm Attributes

- Types
  - Deterministic, Oblivious, Adaptive
- Number of destinations
  - Unicast, Multicast, Broadcast?
- Adaptivity
  - Oblivious or Adaptive? Local or Global knowledge?
  - Minimal or non-minimal?
- Implementation
  - Source or node routing?
  - Table or circuit?

Routing Deadlock



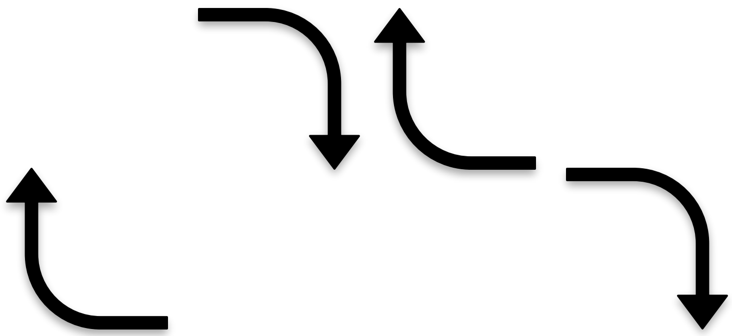
- Each packet is occupying a link and waiting for a link
- Without routing restrictions, a **resource cycle** can occur
  - Leads to deadlock

Types of Routing Algorithms

Deterministic

- All messages from *Source* to *Destination* traverse the same path
- Common example: Dimension Order Routing (DOR)
  - Message traverses network dimension by dimension
  - Aka XY routing
- Cons:
  - Eliminates any path diversity provided by topology
  - **Poor load balancing**
- Pros:
  - **Simple** and inexpensive to implement
  - **Deadlock-free**

Dimension Order Routing



- a.k.a X-Y Routing
  - Traverse network dimension by dimension
  - Can only turn to Y dimension after finished X

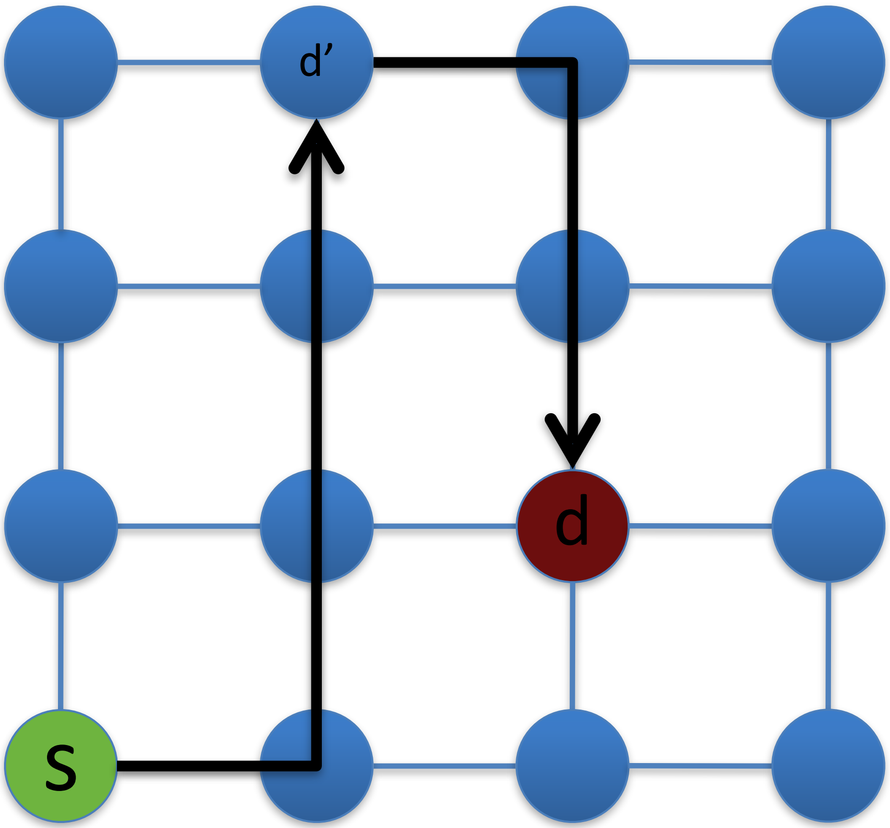
Oblivious

- Routing decisions are made without regard to network state
  - Keeps algorithms simple
  - Unable to adapt

- Deterministic algorithms are a subset of oblivious

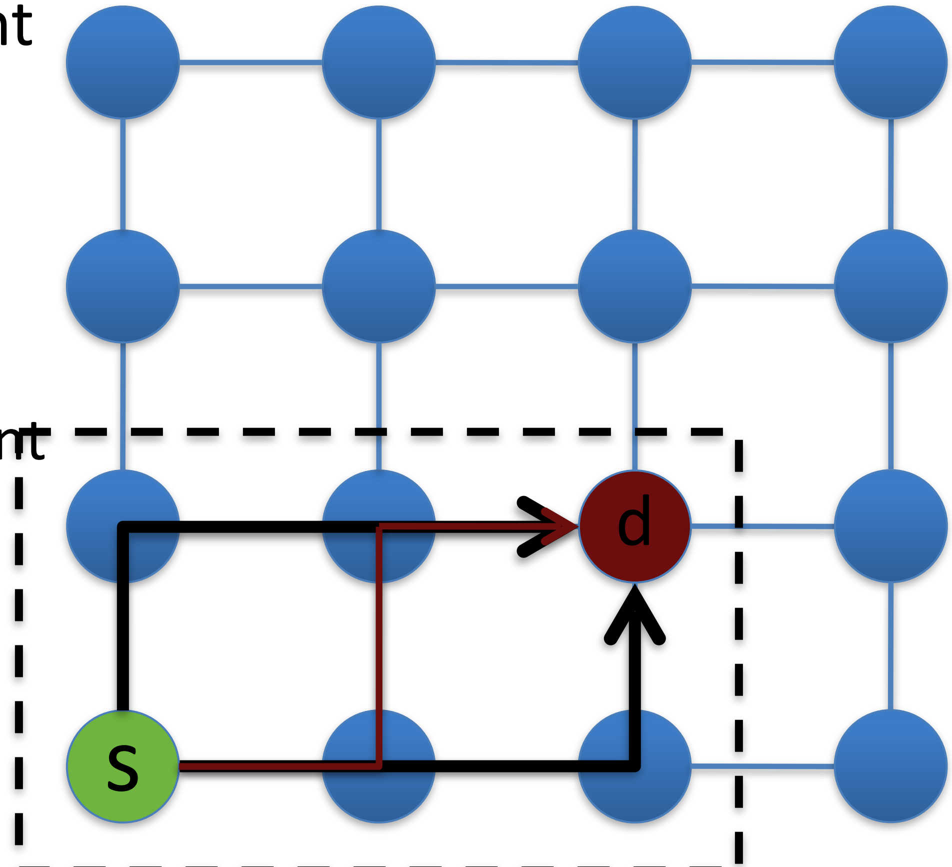
Valiant's Routing Algorithm

- To route from s to d
  - Randomly choose intermediate node d'
  - Route from s to d' and from d' to d.
- Randomizes any traffic pattern
  - All patterns appear uniform random
  - Balances network load
- Non-minimal
- Destroys locality



Minimal Oblivious

- Valiant's: Load balancing but significant increase in hop count
- Minimal Oblivious: some load balancing, but use shortest paths
  - d' must lie within min quadrant
  - 6 options for d'
  - Only 3 different paths



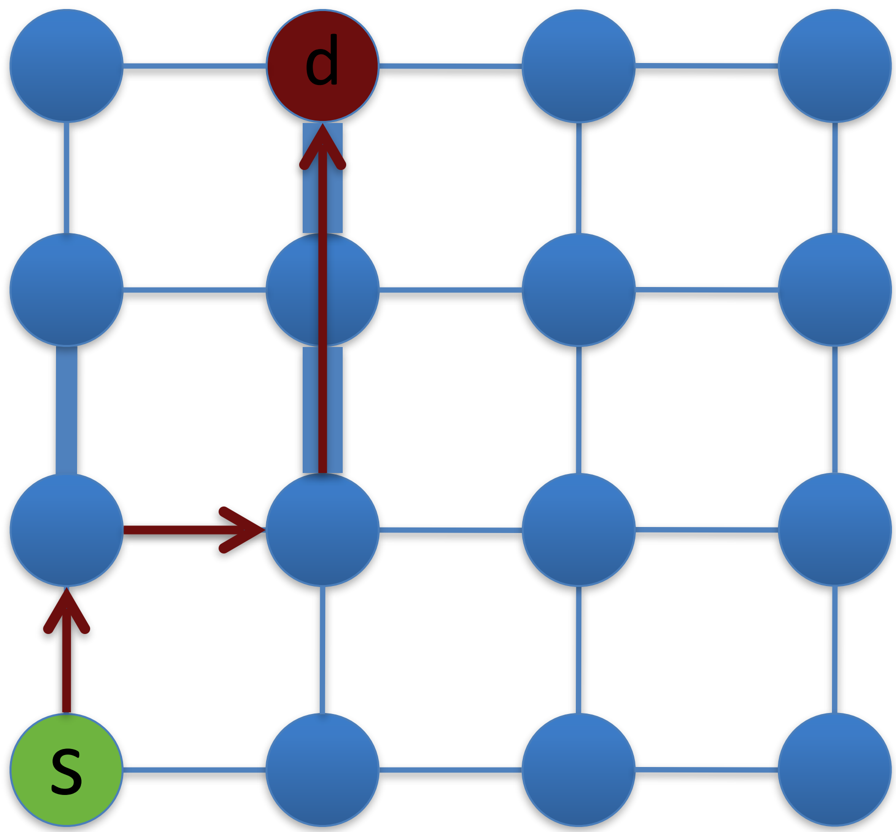
Oblivious Routing

- Valiant's and Minimal Adaptive
  - Deadlock free
    - When used in conjunction with X-Y routing
- Randomly choose between X-Y and Y-X routes
  - Oblivious but not deadlock free!

Adaptive

- Exploits path diversity
- Uses network state to make routing decisions
  - Buffer occupancies often used
  - Coupled with flow control mechanism
- Local information readily available
  - Global information more costly to obtain
  - Network state can change rapidly
  - Use of local information can lead to non-optimal choices
- Can be minimal or non-minimal

Minimal Adaptive Routing



- Local info can result in sub-optimal choices

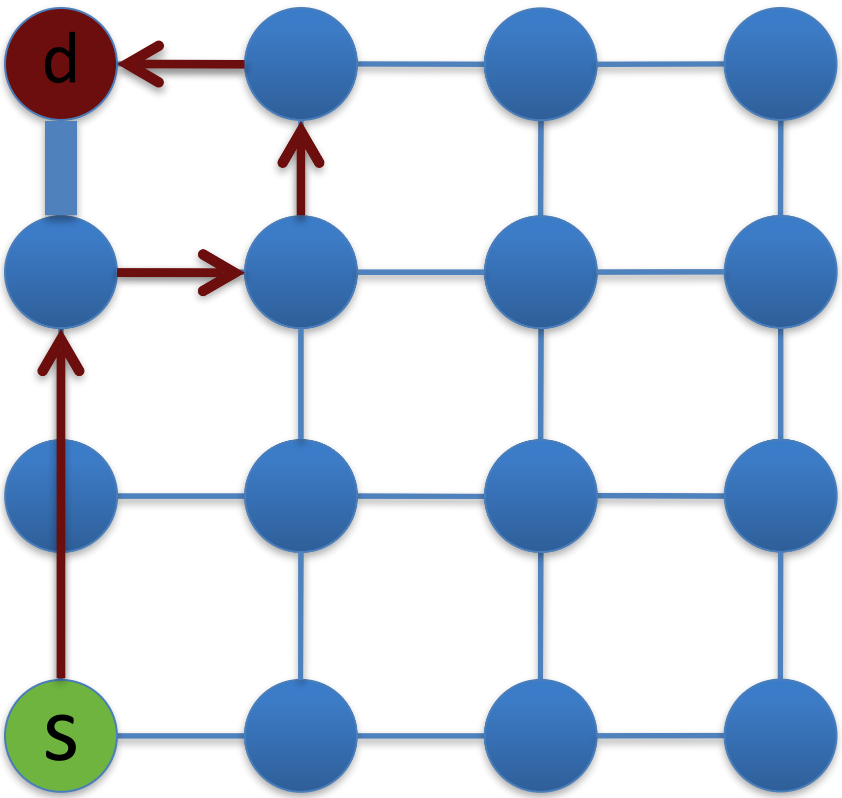
Non-minimal adaptive

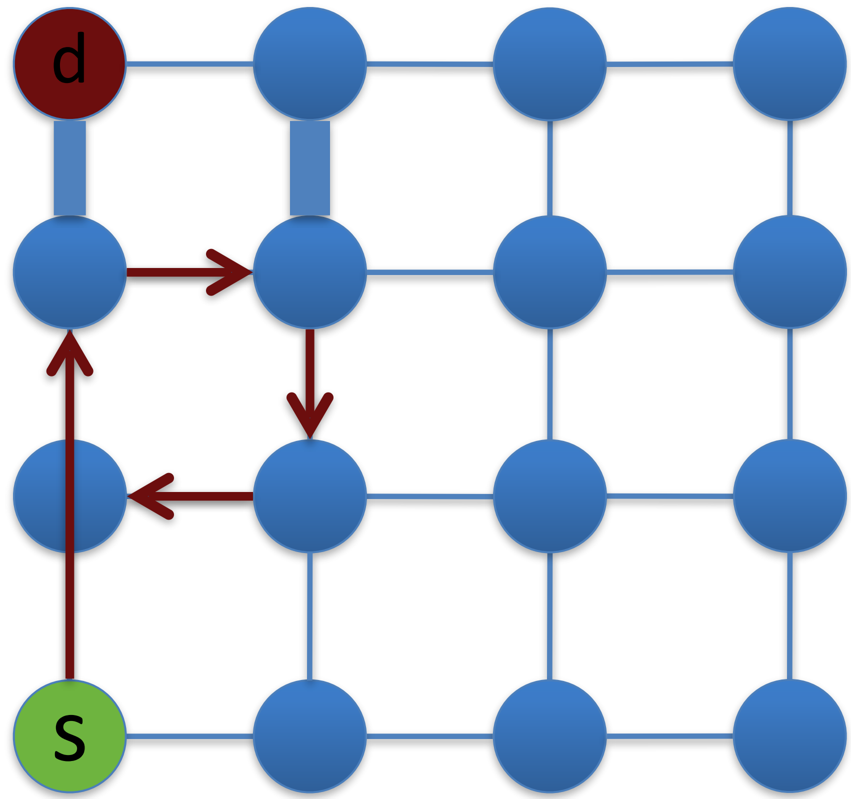
- Fully adaptive
- Not restricted to take shortest path

- Misrouting: directing packet along non-productive channel
  - Priority given to productive output
  - Some algorithms forbid U-turns

- Livelock potential: traversing network without ever reaching destination
  - Mechanism to guarantee forward progress
    - Limit number of misroutings

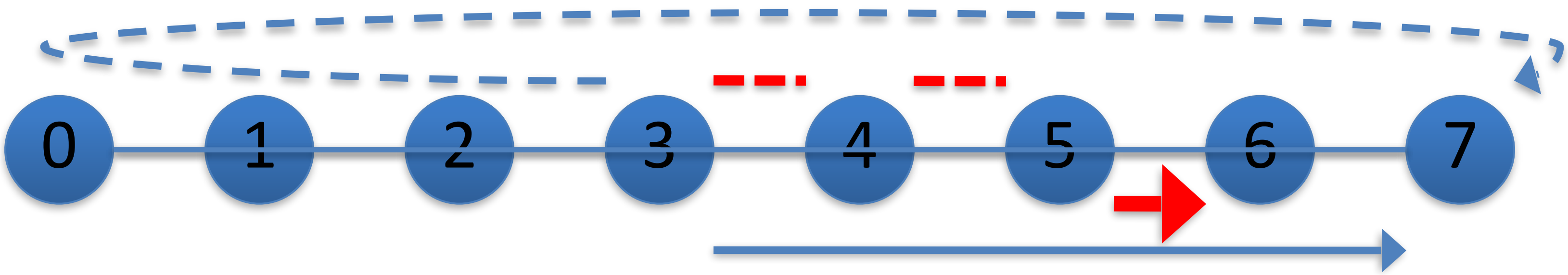
Non-minimal routing example





- Longer path with potentially lower latency
- Livelock: continue routing in cycle

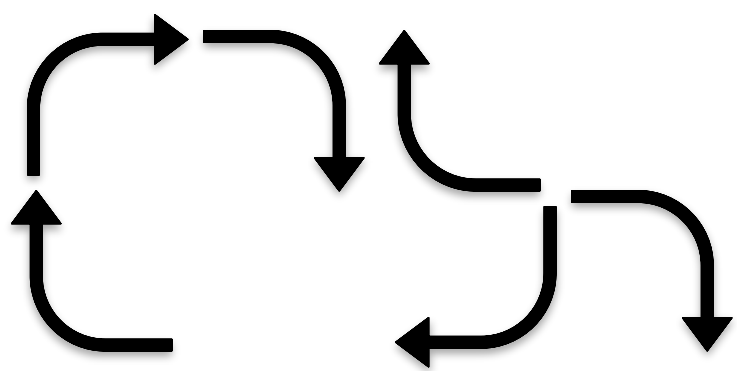
Adaptive Routing Example



- Should 3 route clockwise or counterclockwise to 7?
  - If 5 is using all the capacity of link 5 → 6…
  - …queue at node 5 will sense contention but not at node 3
- Backpressure: allows nodes to indirectly sense congestion
  - Queue in one node fills up, it will stop receiving flits
  - Previous queue will fill up
- If each queue holds 4 packets
  - 3 will send 8 packets before sensing congestion

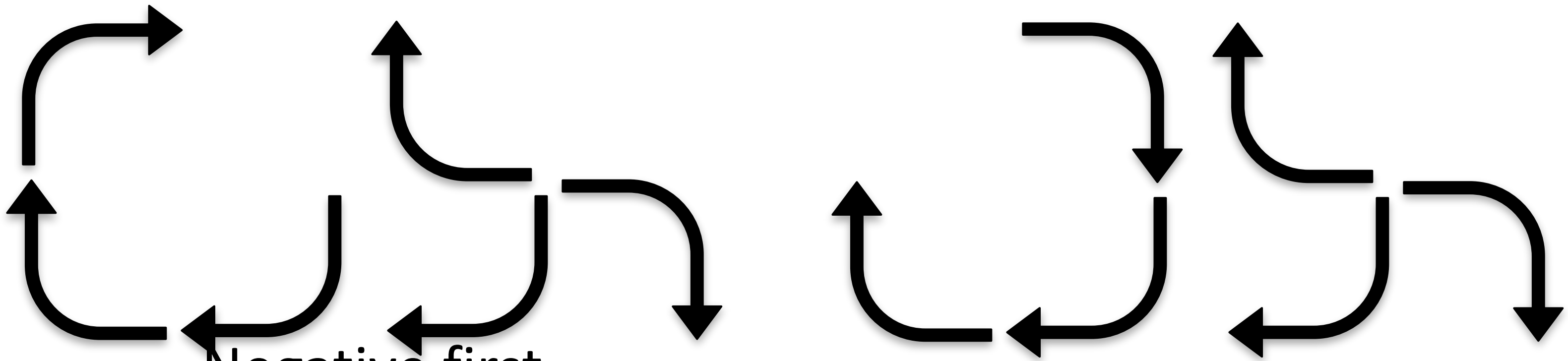
Adaptive Routing: Turn Model

- DOR eliminates 4 turns
  - N to E, N to W, S to E, S to W
  - No adaptivity
- Some adaptivity by removing 2 of 8 turns
  - Remains deadlock free (like DOR)
- West first
  - Eliminates S to W and N to W



West first

Turn Model Routing

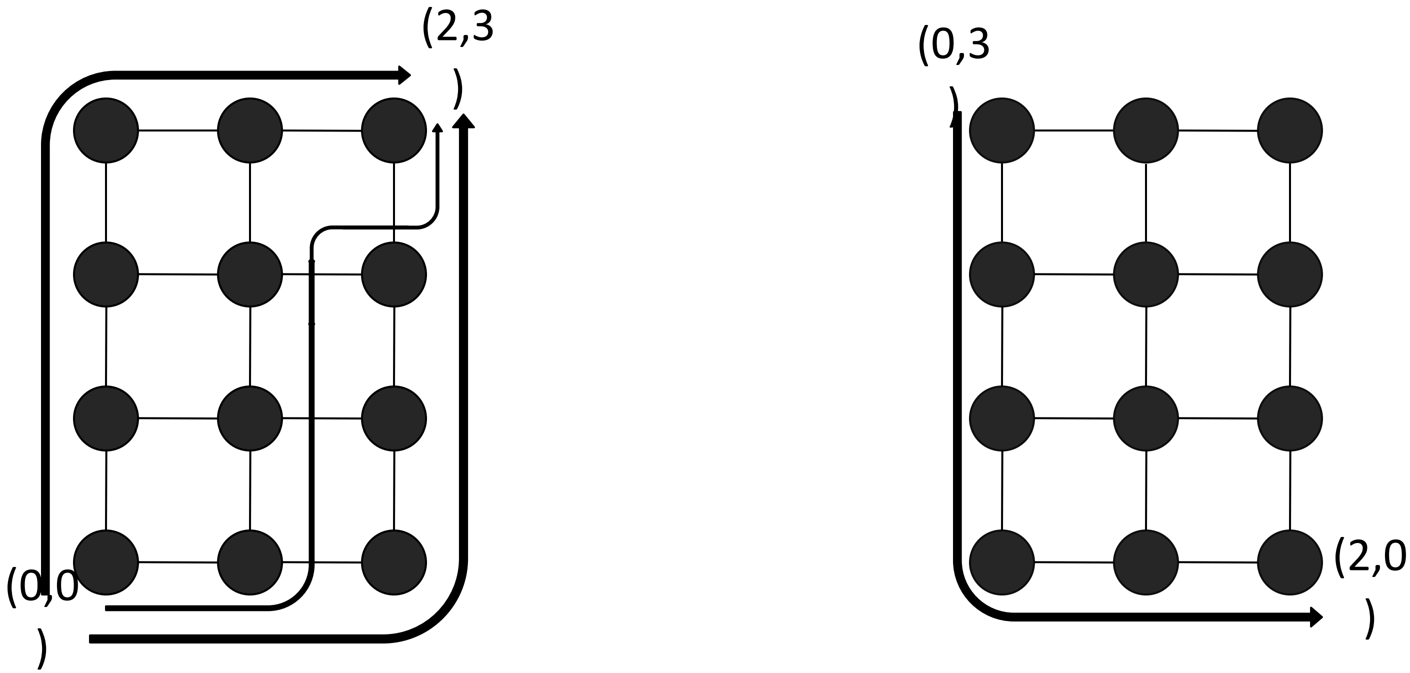


Negative first

North last

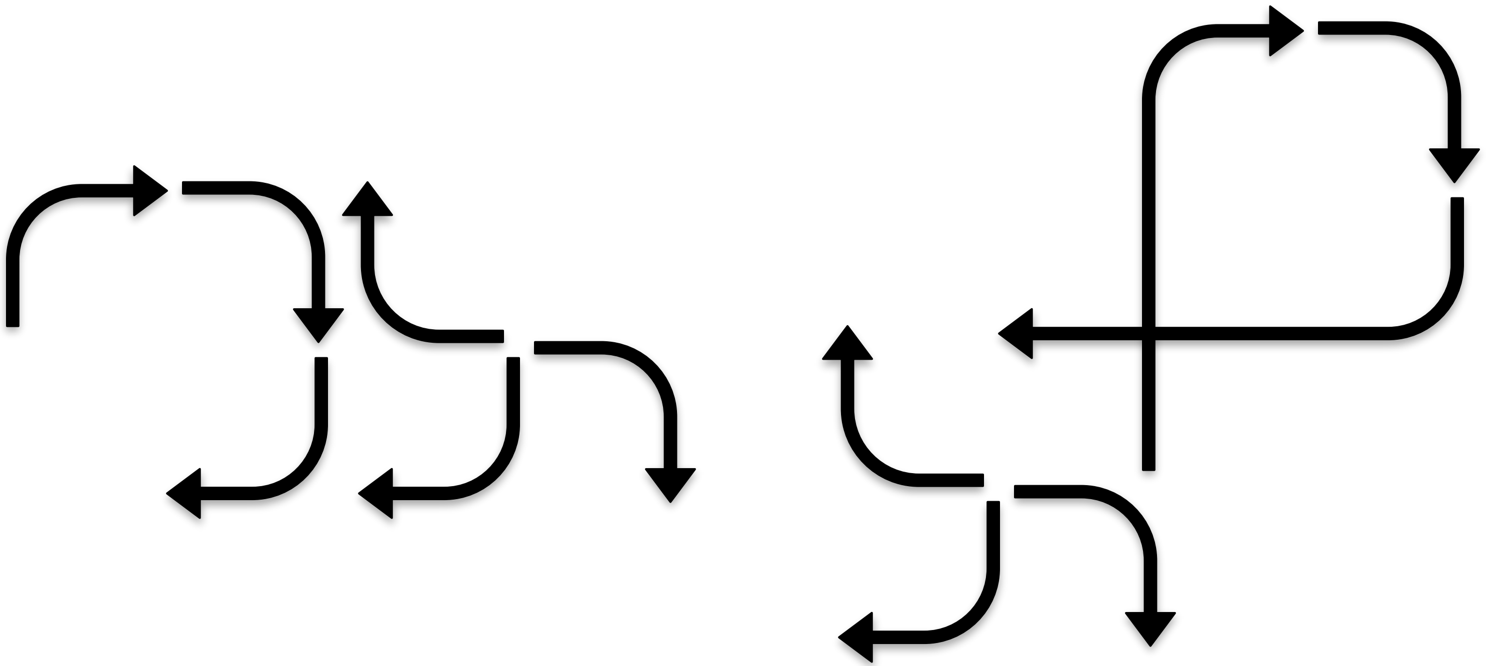
- Negative first
  - Eliminates E to S and N to W
- North last
  - Eliminates N to E and N to W
- Odd-Even
  - Eliminates 2 turns depending on if current node is in odd or even col.
    - Even column: E to N and N to W
    - Odd column: E to S and S to W
  - Deadlock free if 180 turns are disallowed
  - Better adaptivity

Negative-First Routing Example



- Limited or no adaptivity for certain source-destination pairs

Turn Model Routing Deadlock



- What about eliminating turns NW and WN?
- Not a valid turn elimination
  - Resource cycle results

Adaptive Routing and Deadlock

- Option 1: Eliminate turns that lead to deadlock
  - Limits flexibility

- Option 2: Allow all turns
  - Give more flexibility
  - Must use other mechanism to prevent deadlock
  - Rely on flow control (later)
    - Escape virtual channels

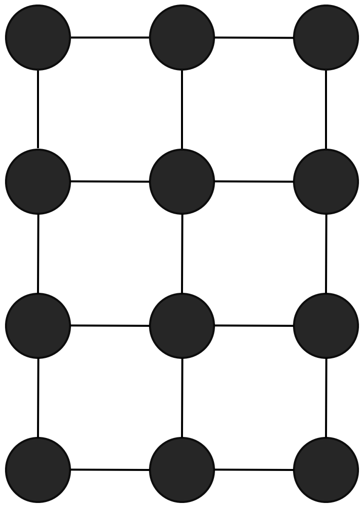
Routing Algorithm Implementation

Routing Implementation

- Source tables
  - Entire route specified at source
  - Avoids per-hop routing latency
  - Unable to adapt dynamically to network conditions
  - Can specify multiple routes per destination
    - Give fault tolerance and load balance
  - Support reconfiguration

Source Table Routing

Destination	Route 1	Route 2
00	X	X
10	EX	EX
20	EEX	EEX
01	NX	NX
11	NEX	ENX
21	NEEX	ENEX
02	NNX	NNX
12	ENNX	NNEX
22	EENNX	NNEEX
03	NNNX	NNNX
13	NENNX	ENNNX
23	EENNNX	NNNEEX



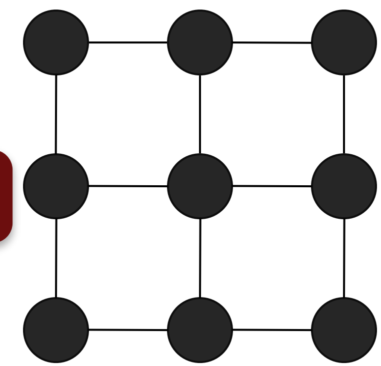
- Arbitrary length paths: storage overhead and packet overhead

Node Tables

- Store only next direction at each node
- Smaller tables than source routing
- Adds per-hop routing latency
- Can adapt to network conditions
  - Specify multiple possible outputs per destination
  - Select randomly to improve load balancing

Node Table Routing

	To								
From	00	01	02	10	11	12	20	21	22
00	X |-	N | -	N | -	E | -	E | N	E | N	E | -	E | N	E | N
01	S | -	X | -	N | -	E | S	E | -	E | N	E | S	E | -	E | N
02	S | -	S| -	X | -	E | S	E | S	E | -	E | S	E | S	E | -
10	W|-	W|-	W|-	X | -	N | -	N | -	E | -	E | N	E | N
11	W|-	W|-	W|-	S | -	X | -	N | -	E | S	E | -	E | N
12	W|-	W|-	W|-	S | -	S | -	X | -	E | S	E | S	E | -
20	W|-	W|-	W|-	W|-	W|-	W|-	X | -	N | -	N | -
21	W|-	W|-	W|-	W|-	W|-	W|-	S | -	X | -	N | -
22	W|-	W|-	W|-	W|-	W|-	W|-	S | -	S | -	X | -



(1,0)

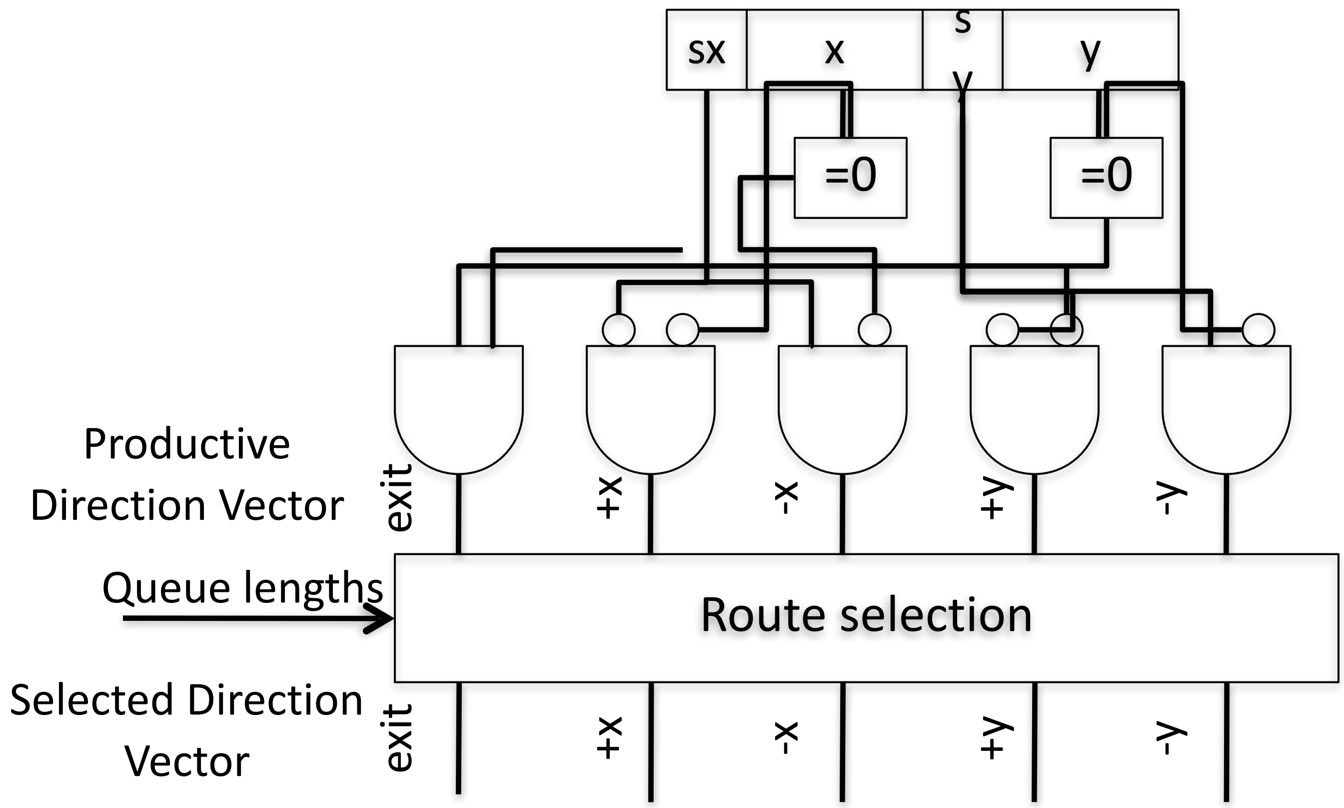
- Implements West-First Routing
- Each node would have 1 row of table
  - ❒ Max two possible output ports

Implementation

- Combinational circuits can be used
  - Simple (e.g. DOR): low router overhead
  - Specific to one topology and one routing algorithm
    - Limits fault tolerance

- OTOH, tables can be updated to reflect new configuration, network faults, etc

Circuit Based



- Next hop based on buffer occupancies in a 2D mesh
- Or could implement simple DOR
- Fixed w.r.t. topology

Routing Algorithms: Implementation

Routing Algorithm	Source Routing	Combinational	Node Table
Deterministic			
DOR	Yes	Yes	Yes
Oblivious			
Valiant's	Yes	Yes	Yes
Minimal	Yes	Yes	Yes
Adaptive	No	Yes	Yes

Routing: Irregular Topologies

- MPSoCs
  - Power and performance benefits from irregular/custom topologies
- Common routing implementations
  - Rely on source or node table routing
- Maintain deadlock freedom
  - Turn model may not be feasible
    - Limited connectivity

Routing Summary

- Latency paramount concern
  - Minimal routing most common for NoC
  - Non-minimal can avoid congestion and deliver low latency
- To date: NoC research favors DOR for simplicity and deadlock freedom
  - On-chip networks often lightly loaded
- Only covered unicast routing
  - Recent work on extending on-chip routing to support multicast