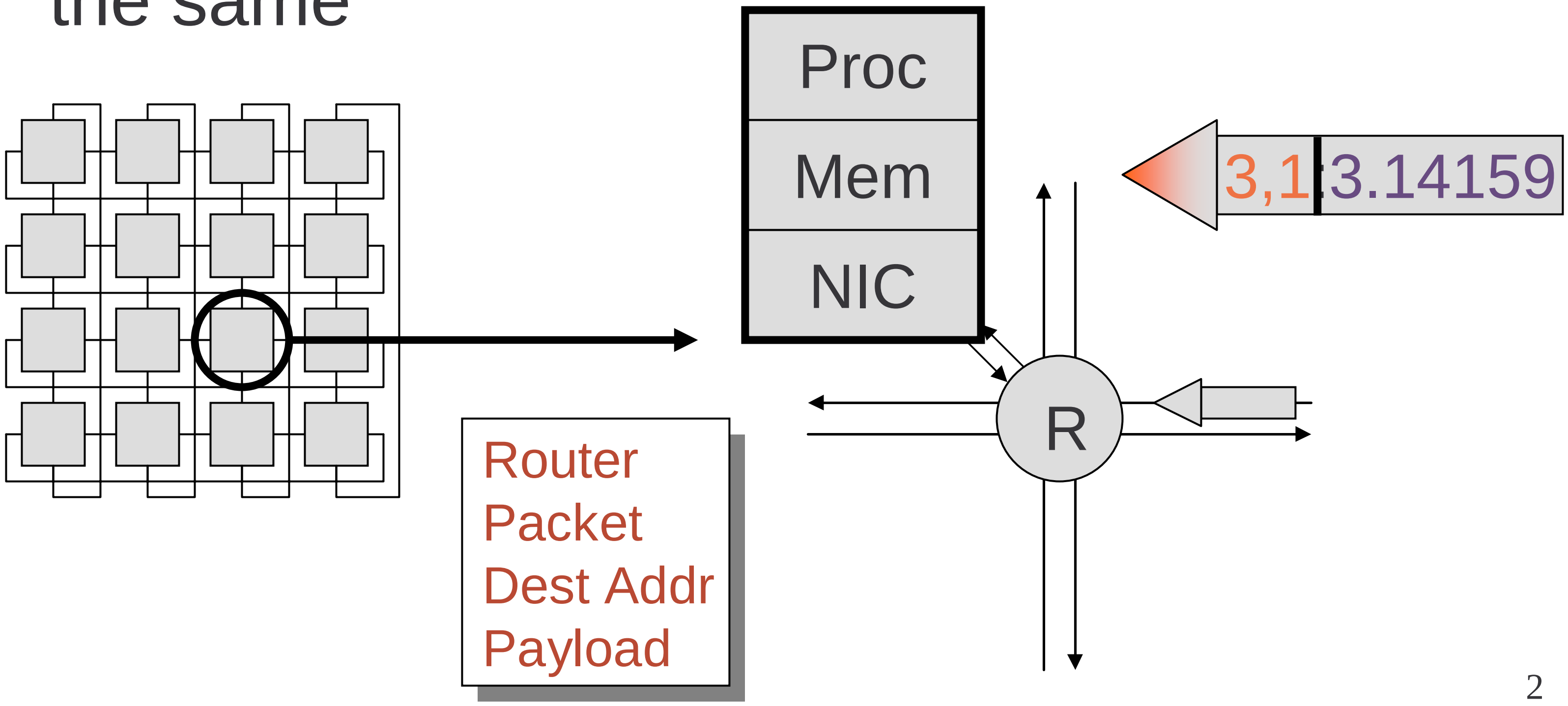
Interprocessor Communication

There are two main differences between sequential computers and parallel computers -- multiple processors and the hardware to connect them together. That hardware is the most important part of the design.

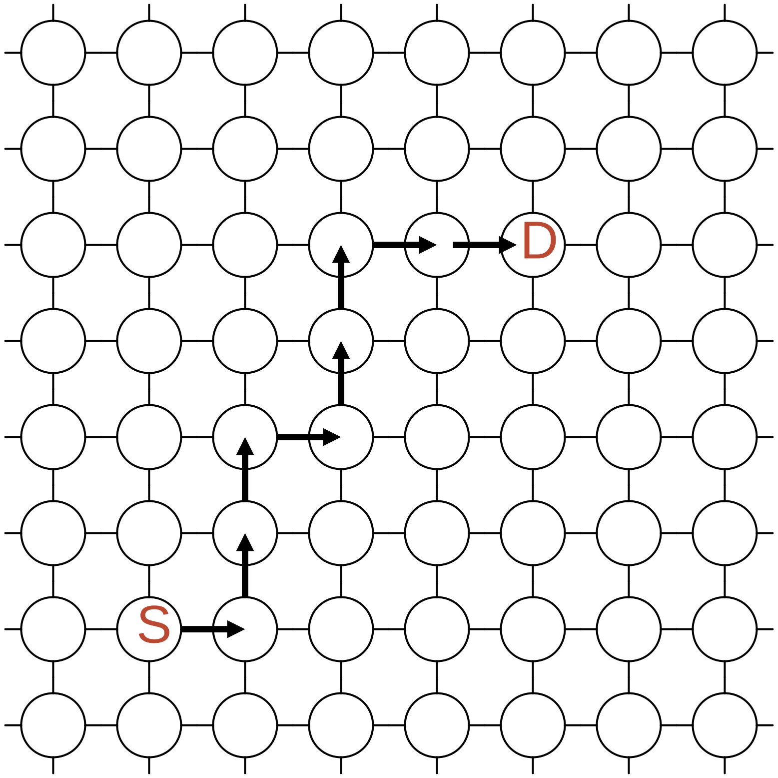
Basics of Network Routing

Routers can be integrated with the processors or they can be a separate interconnection topology -- the two approaches are logically the same



Router
Packet
Dest Addr
Payload

Goals of Network Routing



Must have …

- High Throughput
- Low Latency

Must be …

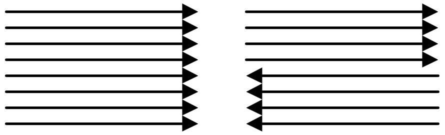
- Deadlock-free
- Livelock-free
- Starvation-free

Should be insensitive to…

- Congestion
- Bursts
- Faults

Physical Connection

- The wires connecting switches can be either unidirectional -- all wires transmit the same way, which alternates -- or bidirectional -- half of the wires are permanently set to transmit in each direction



  - For sustained information flow in both directions, the bandwidth and latency are the same
  - For one packet in the network, the latency is the same, but the bandwidth is doubled with unidirectional

A “flit” is a flow-control unit

A “phit” is a physical-transmission unit

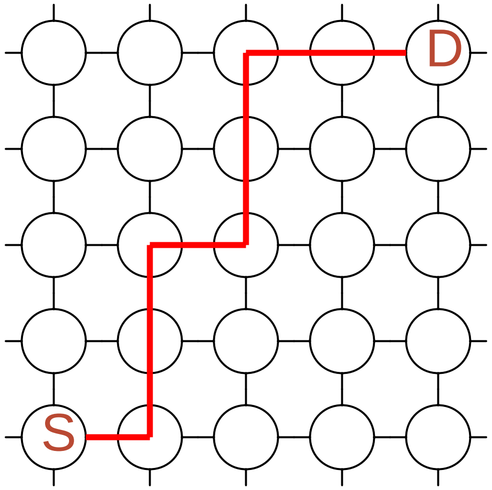
Destination Addressing

- In a regular topology the switches can compute the path to the destination based only on its address
    - Fitting the destination address into the first phit allows the node to begin addressing immediately
- For irregular networks packets are “source routed” -- the path to the destination is computed at the source, and prefixed to the information
    - At each hop its own address is removed from the front

Source routing sets path w/o considering congestion

Transport Approaches -- Circuit Switching

- Circuit Switching
  - A static path is set-up between source and destination
  - Once set up the information is pipelined along the path
  - The path is “torn down” when the transmission is over
- The set-up and tear-down are overhead
- Very effective for large quantities of data
- Concept inherited from telephone switching



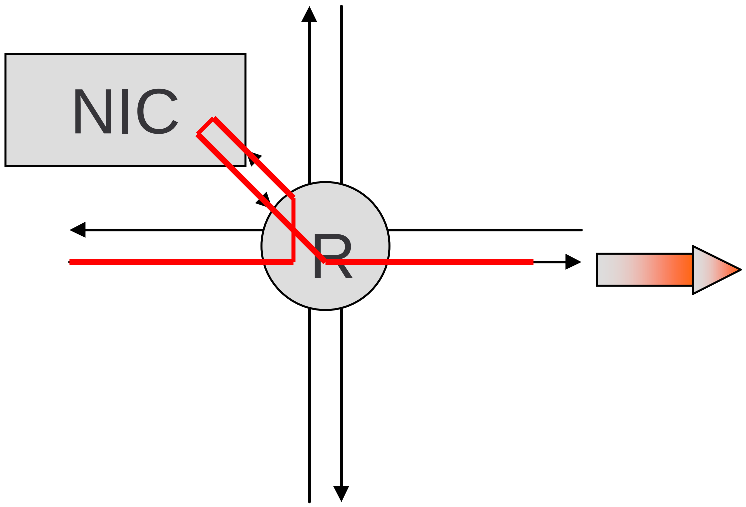
Transport Approaches -- Packet Switching

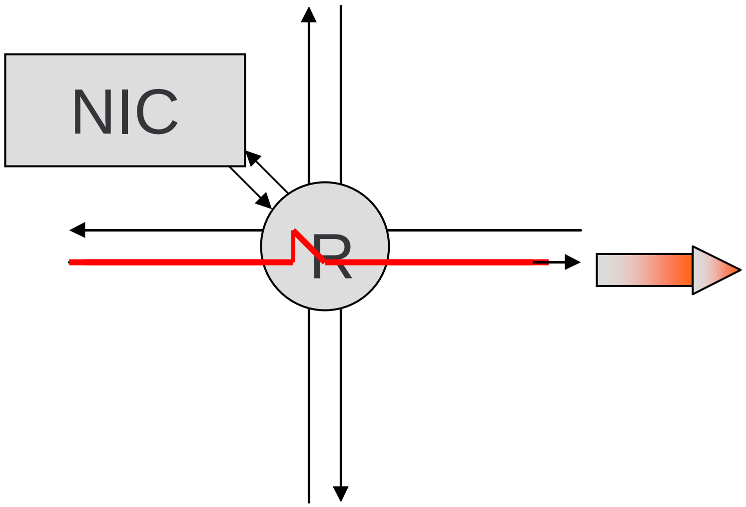
- In packet switching the information is divided up into packets, and the destination address is prefixed to each packet
  - Each packet is treated independently, preventing any message from monopolizing resources
  - Biased to favor short transmissions
  - The header is overhead; pipelining is less effective
  - Allows for adaptivity

- Original approach for store & forward nets
- Virtual Cut-through has replaced S&F

Store and Forward *vs* Virtual Cut-through

- S&F allows for more sophisticated protocol, with higher reliability
- Virtual cut-through allows for greater speed



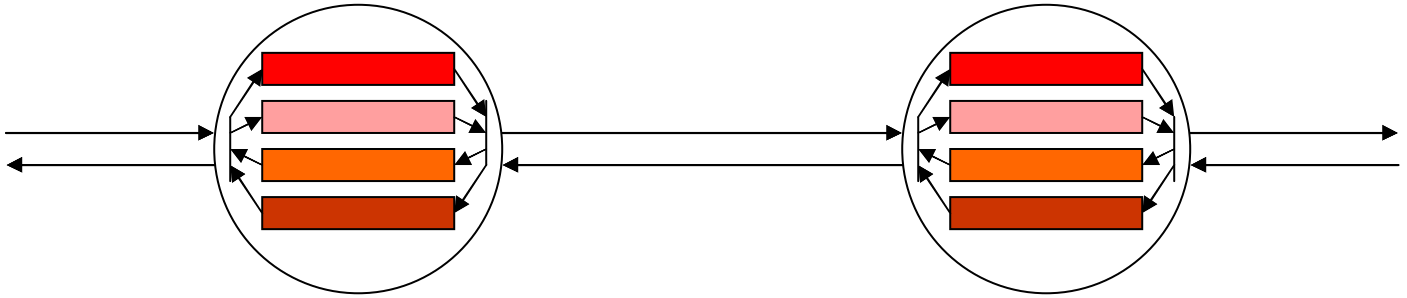


Xport Approaches -- Wormhole Switching

- Worm-hole routers send entire message as a single packet -- dynamically circuit switched
    - Eliminates the overhead of set-up and tear-down
    - Fully exploits pipelining, minimizes overhead of headers
    - Monopolizes resources and penalizes short messages
    - Messages delivered in order of transmission
- WH is the most popular transmission method of communication networks -- simple
- Compromise schemes
    - Large, e.g. page, variable length packets
    - Allow small messages to “play through”

Virtual Channels

- A single physical network can transmit information for multiple logical networks
- Keep separate buffers for each network
- Virtual channels are often used to safeguard against deadlock in a single network design



Router Design

- Router design is an intensively studied topic
- Inventing a routing algorithm is the easy part … demonstrating that it is low latency, high throughput, deadlock free, livelock free, starvation free, reliable, etc. is tougher
- Generally ...
    - Low latency is the most significant part
    - Throughput -- delivered bits -- is next most significant
    - The only interesting case is “performance under load,” so the challenge is handling contention

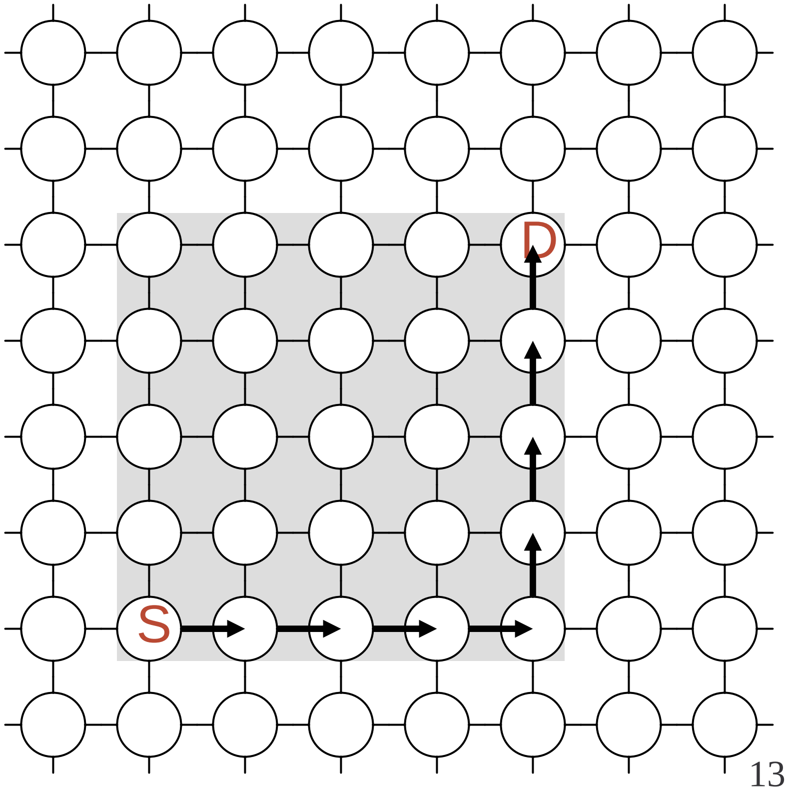
Topologies

- Many regular network topologies have been considered … there is no best topology
- A common family of useful topologies are the *k-ary d-cubes*, which have k nodes in each of d dimensions
    - 2-ary d-cude is the d-dimensional binary hypercube
    - n-ary 2-cube is the nxn mesh or torus
- All routing algorithms considered here will at least apply to the k-ary, d-cubes

Oblivious Routing

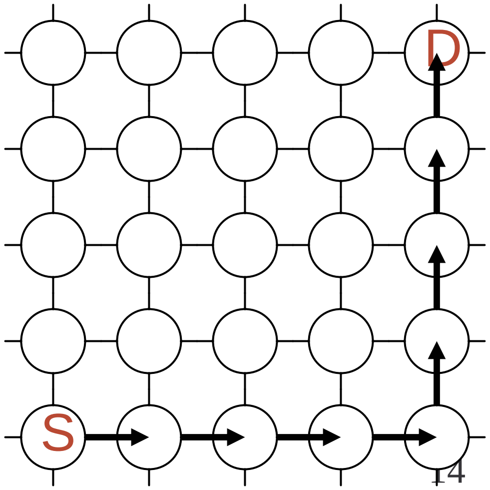
- Oblivious Routers use a single path between any [source, destination] pair
  - Dimension order
  - Simple logic, fast
  - Virtual cut-through
  - State-of-the-art for MIMD computers

Though any path in the gray area is possible, oblivious uses only one



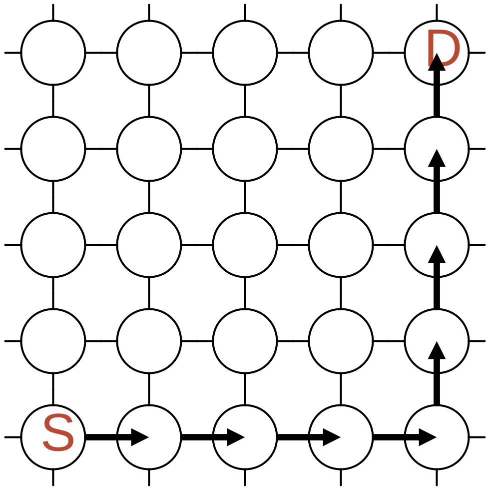
Oblivious Router

- Oblivious router's speed comes from very simple decision logic
- Row-first and Column-first are alternatives
- When a packet arrives, a node must decide
    - Stop? The destination has been reached
    - Turn? The column has been reached
    - Otherwise, continue



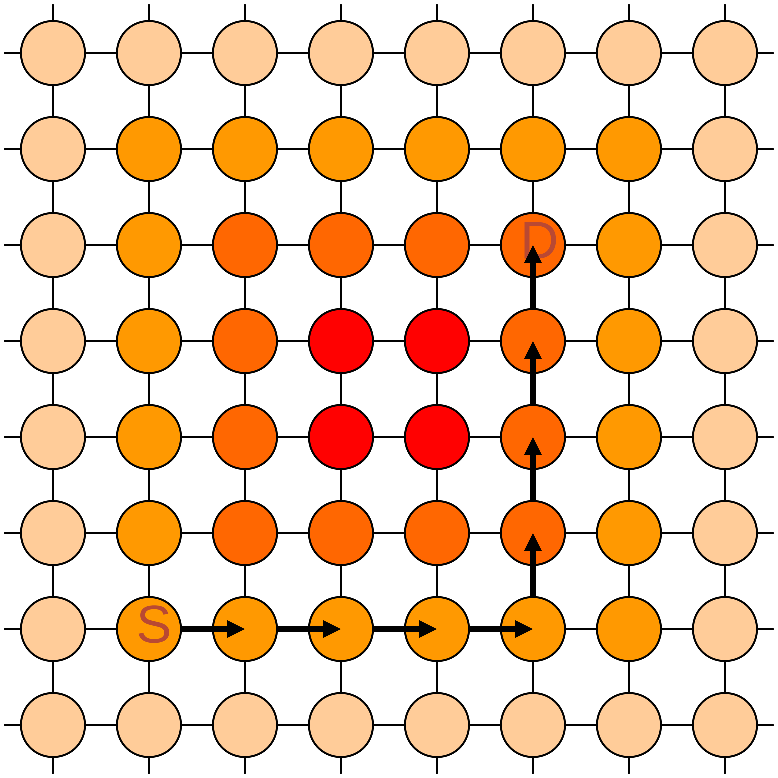
Deadlock Free

- Is the oblivious mesh router deadlock free?
  - Can packets get in a state where they block each other?
  - Give separate L/R, U/D wires and 1 turn



Row, Then Column Gets Hot

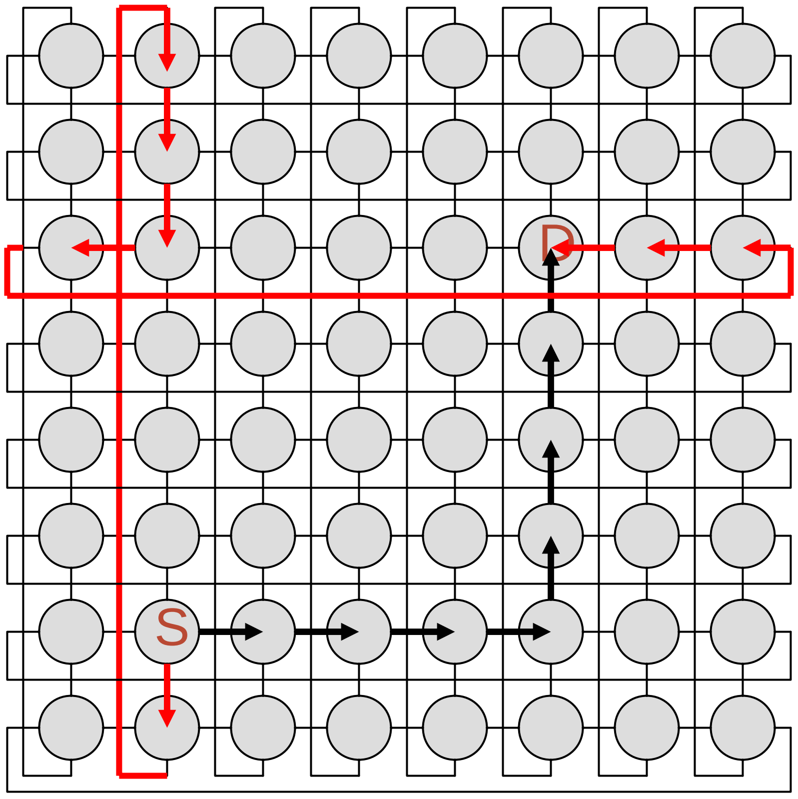
- For a mesh most traffic crosses the center



Congestion in the middle means communication is often delayed there

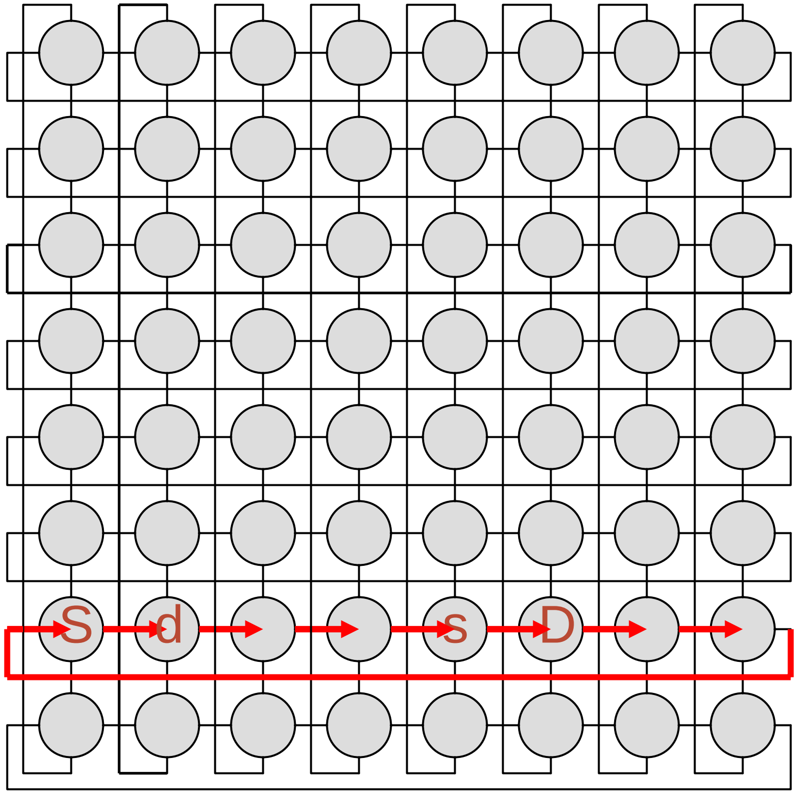
Topological Solution

- The mesh gets hot because not all edges are “equally useful” … change to a torus



The Torus Is Tough for Oblivious

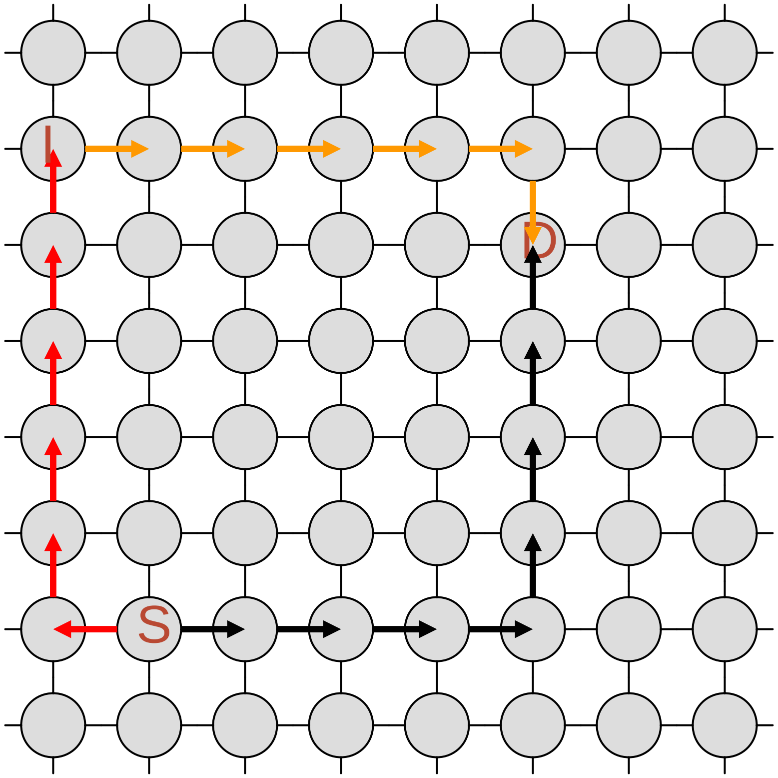
- Oblivious routers are fast, but not clever … it is easy to deadlock a torus oblivious router



Dally and Seitz showed that it is possible to route worm-hole packets on a torus without deadlock … used virtual channels

Randomized Oblivious Routing

- Pick at random an intermediate destination

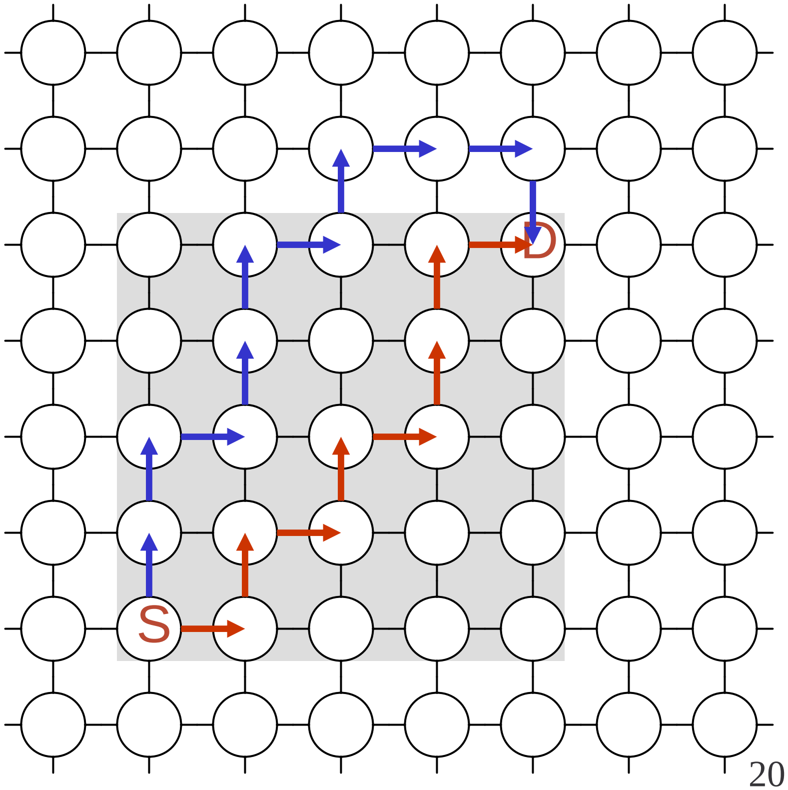


The expected path-length is doubled

- Better for better-connected topologies, e.g. hypercube

Adaptive Routing

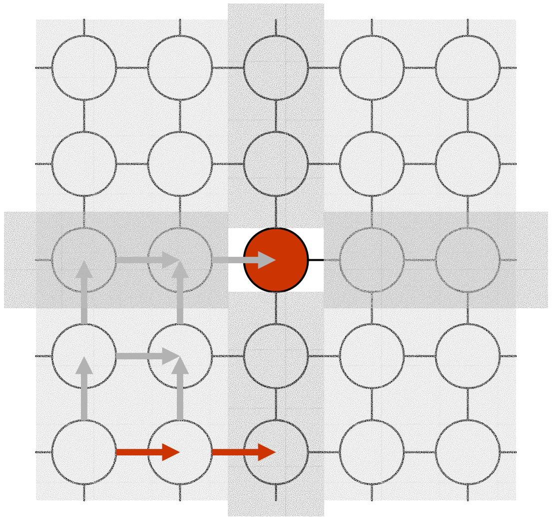
- Adaptive routers try to make routing decisions adaptively to by-pass congestion
- Two types …
  - Minimal adaptive ... adaptive path is a shortest path; always go forward
  - Non-minimal adaptive … any path is allowed



Minimal Adaptive -- It Doesn’t Work

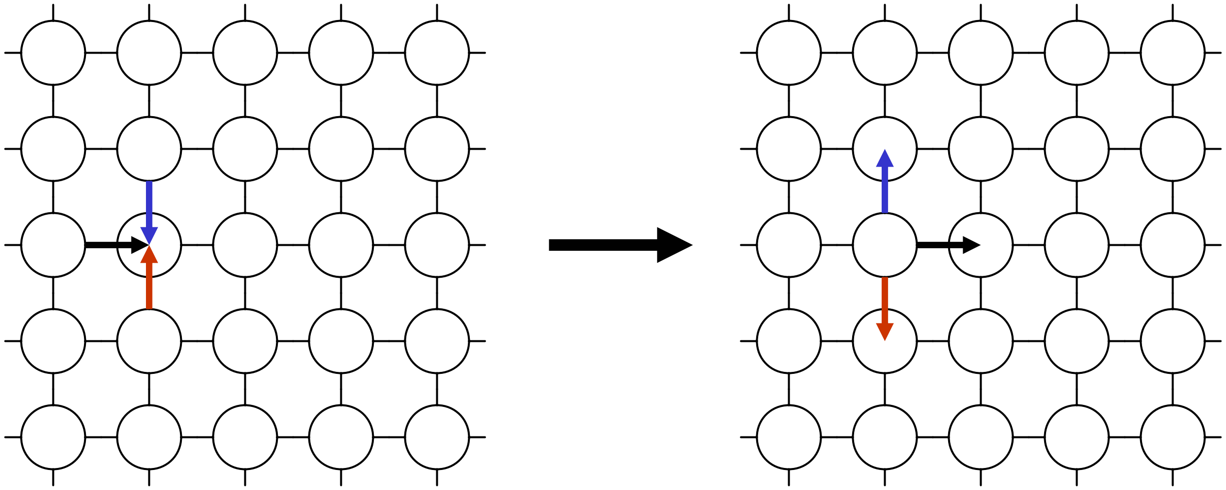
Choosing among shortest paths seems like the best way to be adaptive, but it doesn’t work

- Once hot spot develops, rows, cols build up because there is no more flexibility left
- Congested rows and cols. act as barriers
- The congestion spreads



Deflection Routing

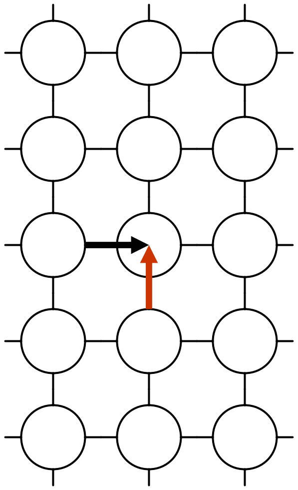
- “Hot Potato” routers try to keep going
    - An adaptive synchronous approach
    - Incoming packets are matched to outgoing channels
    - Losers are assigned arbitrarily
    - All packets leave on the next step



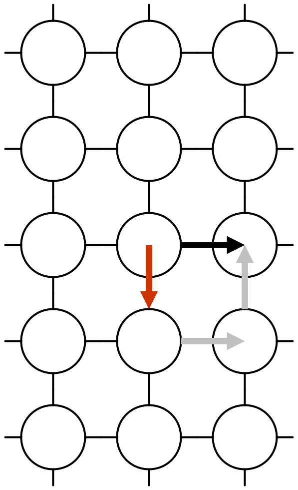
Livelock is a potential problem

Hot Potato Routers

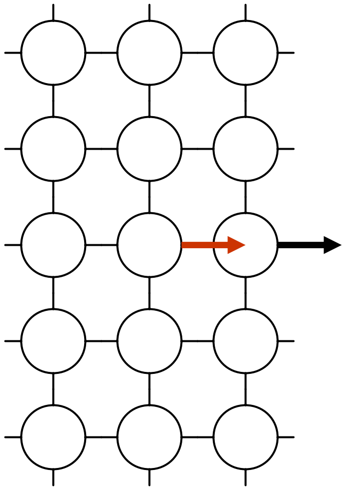
- Deflection routing used in Tera Computer
- Perhaps it causes too much turbulence in the network … waiting one step might save one



Collision



Protocol



Alternative

Chaos Router

- Chaos Router is a randomizing, non-minimal adaptive packet router (It is not related to Chaos theory of physics)
- Chaos Router sends packets along minimal paths in almost all cases, but when blocked by severe congestion, it sends packets along any path
    - Avoids the hot potato router’s “too eager policy” to send packets the wrong direction
    - Avoids the minimal adaptive router’s “catch-22” of discovering congestion after all of the flexibility is gone

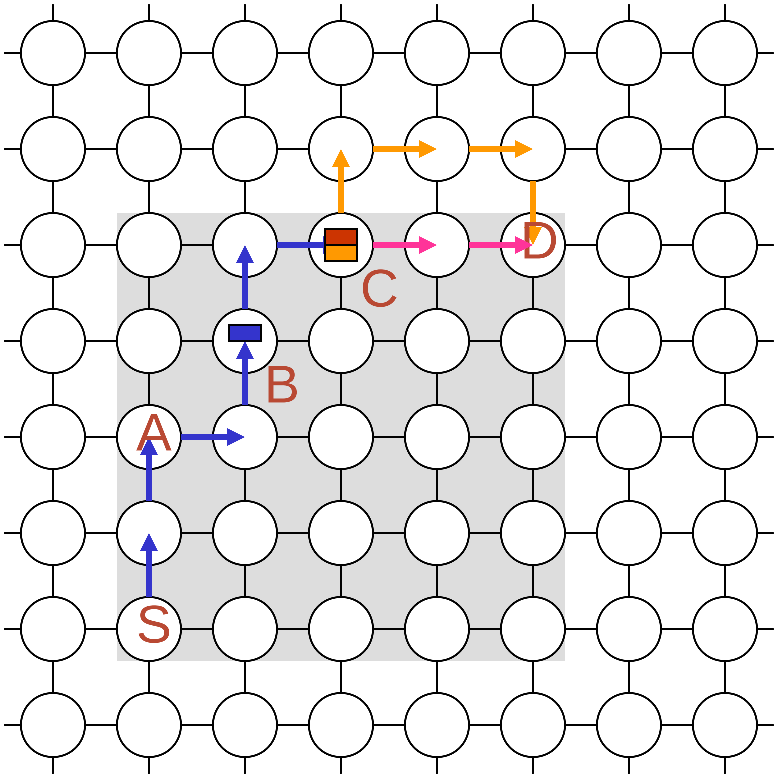
The Chaos Algorithm

- The Chaos router directs packets according to the following scheme
  0) Transmit packets with virtual cut-through
  1) Send packets along a random shortest-path whenever possible, i.e. when traffic is light
  2) Wait briefly inside the switch (no store-and-forward) when traffic is moderate
  3) In heavy traffic when there is no more space in the switch to wait, “deroute” some (randomly chosen) waiting packet

Derouting (probably) takes a packet further from its destination

Chaos Example

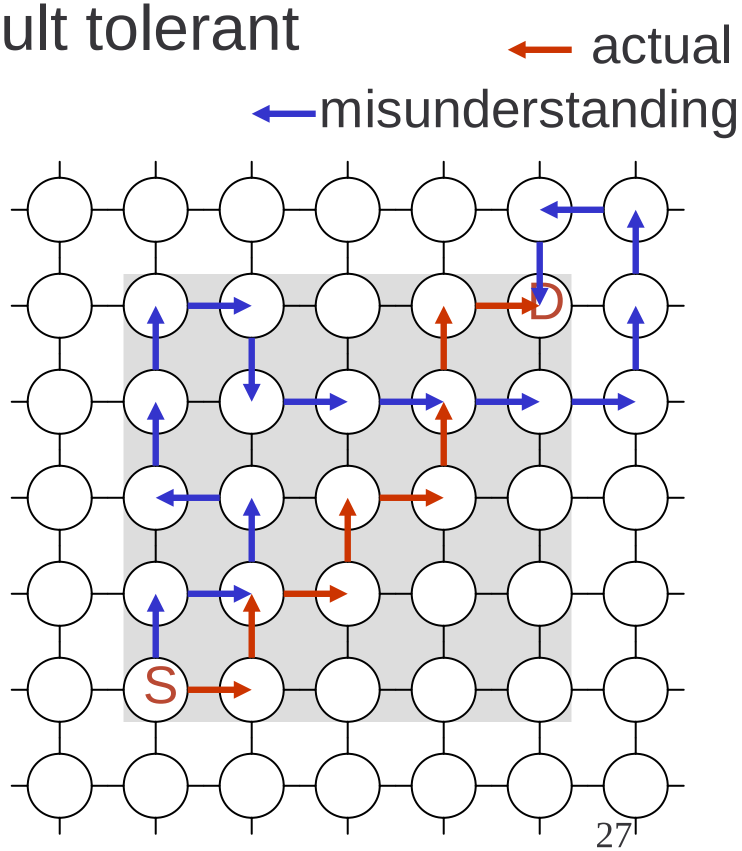
- Steps of Chaos between a source (S) and a destination (D)
  1) Take random shortest path (A)
  2) Wait briefly inside of a switch (B)
  3) When no waiting room is available deroute a random packet (C)



Chaos Router Properties

- Packets take random minimal path except in cases of extreme congestion
- Chaos Routers are inherently fault tolerant
- Adaptivity reduces latency and increases throughput because packets select paths incrementally based on congestion …they take productive paths if available

Packets can be delivered out of order and must be reassembled

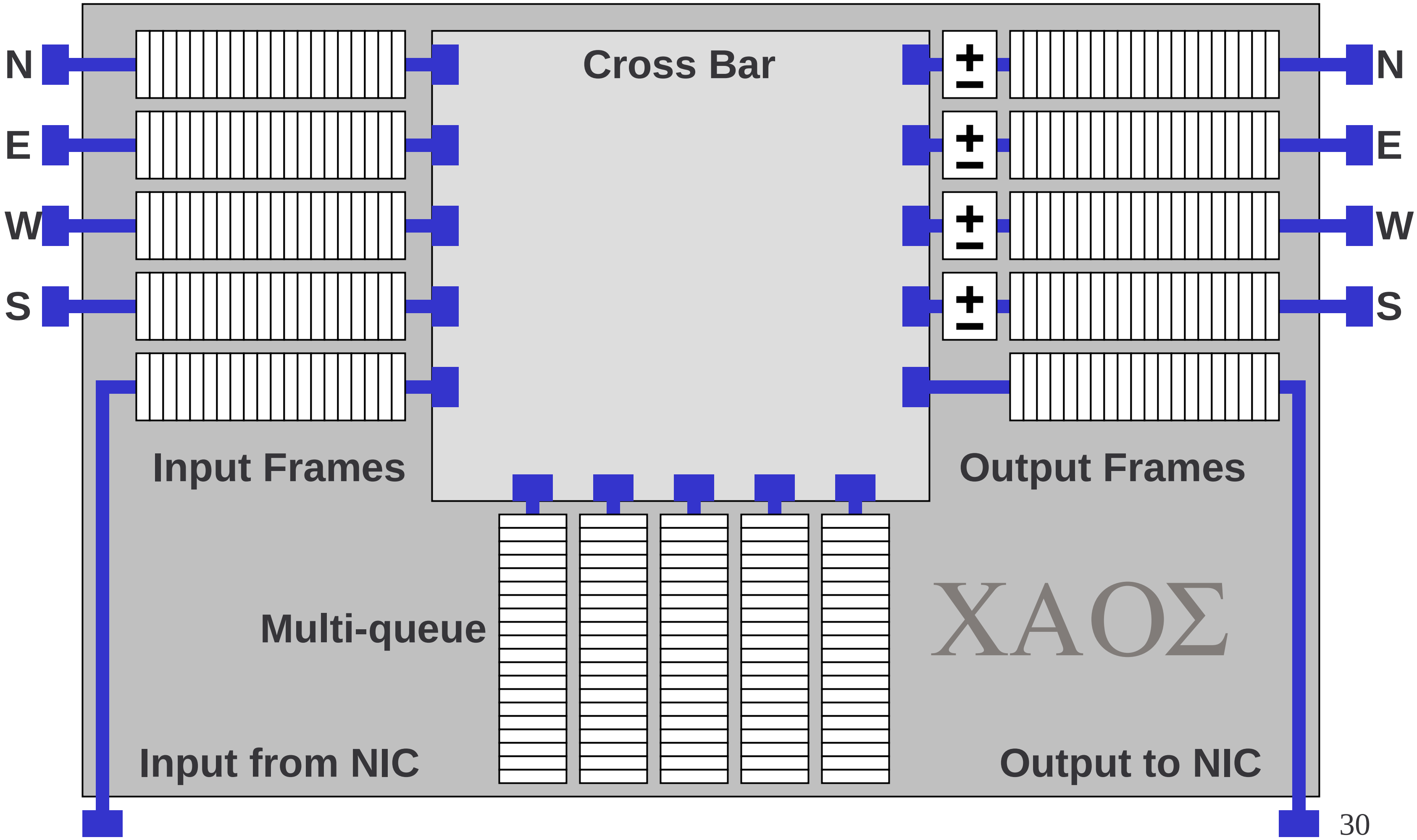


Break

A Closer Look: A Chaos Implementation

- How does Chaos compare with the oblivious routing?
- We review results of a fair comparison from mid 90's
  - Elko Router constructed at Caltech
  - Chaos Router constructed at UW
  - Identical technologies; same topological domain; comparable design and engineering investment

Chaos Router Design (2-ary, n-cube)

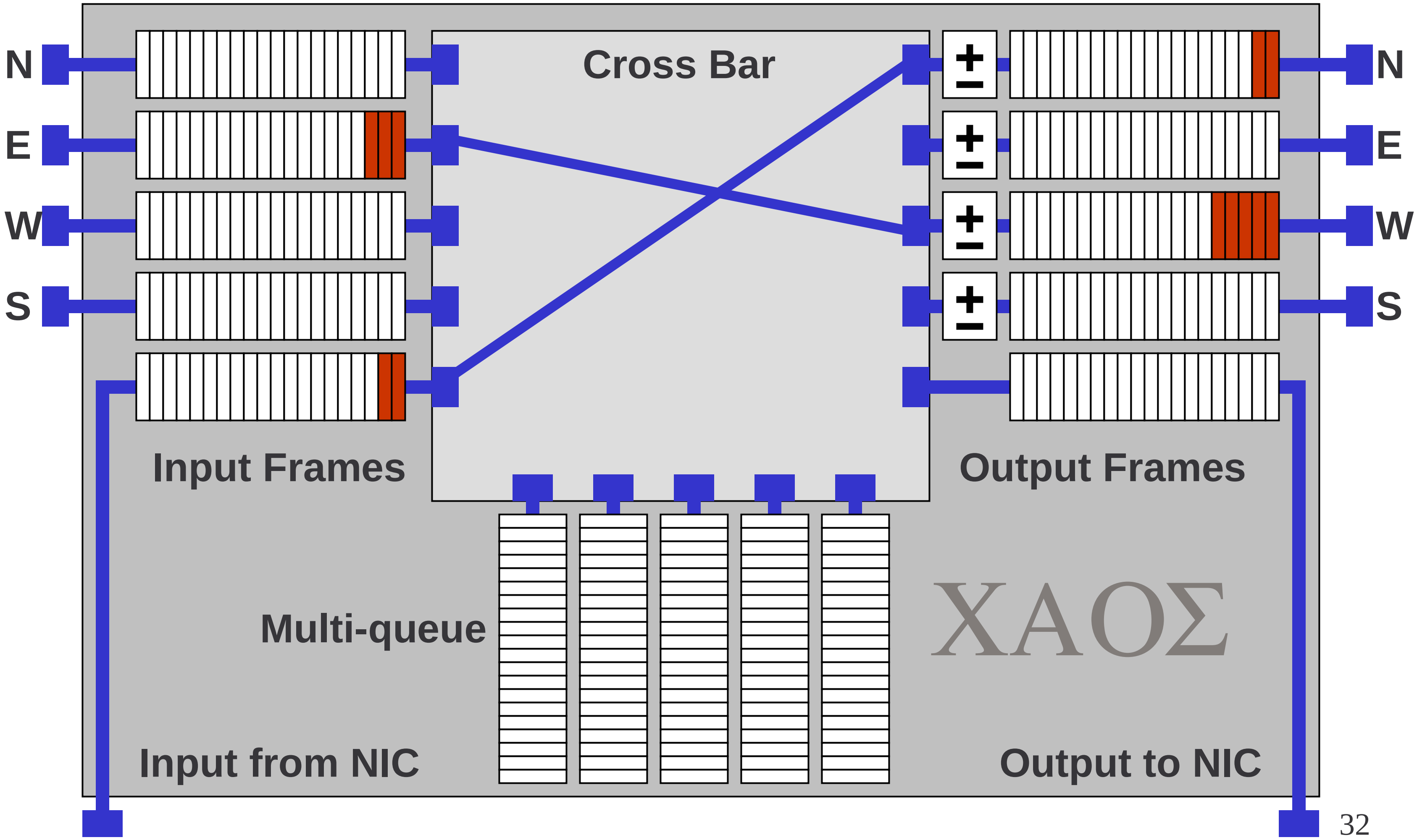


Management of Multi-queue

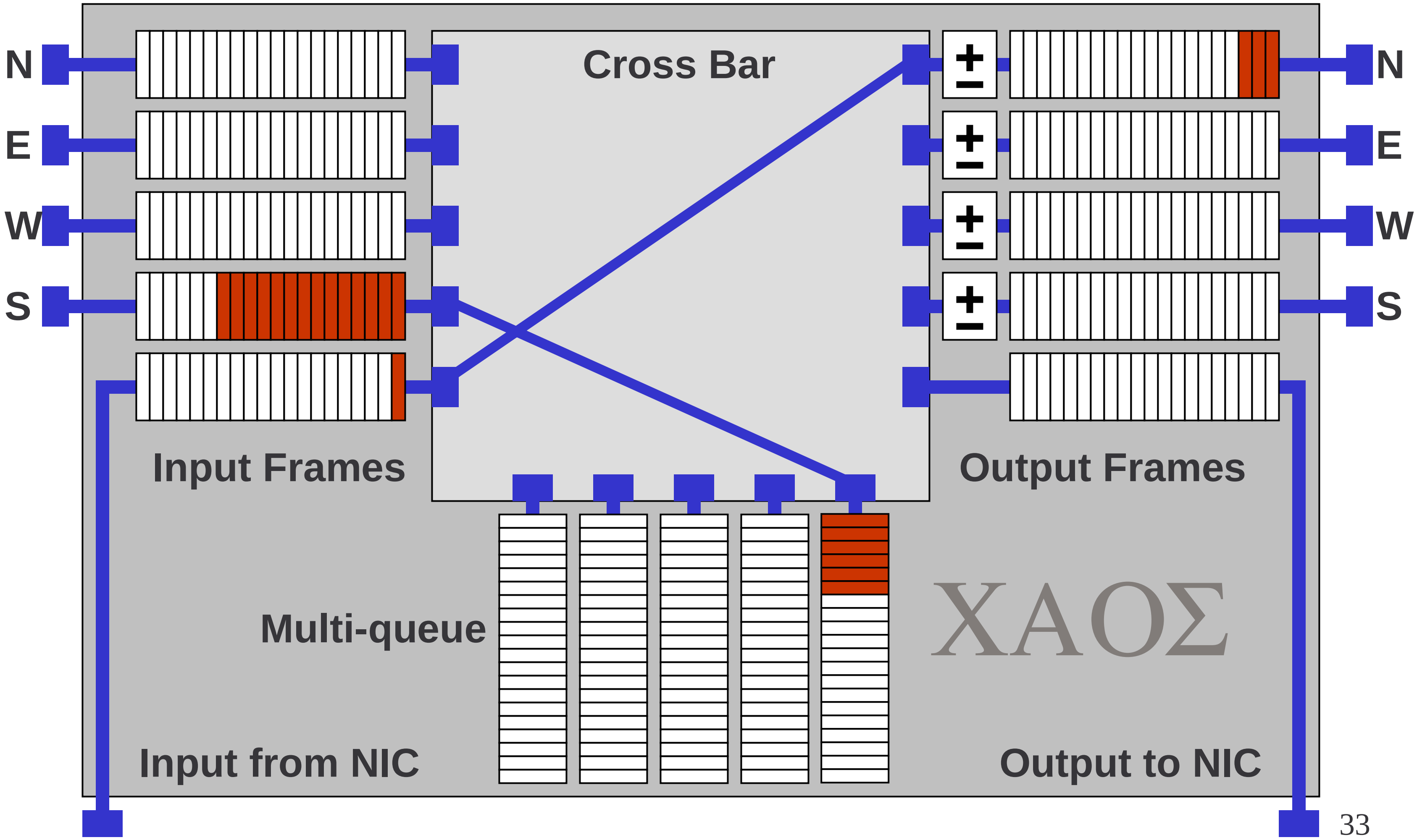
Multi-queue implements logical queue discipline

- Physical position in list of frames is unimportant
- 2 packets with same destination serviced in arrival order
- When multiple output channels are possible, packet takes the first free
- Multi-queue packets take precedence over newly arriving packets
- No packet moves to MQ until it is fully arrived -- implying there is no chance of cutting through
- One multi-queue frame must always be empty, ready to receive a packet -- for livelock protection
- Derouting victims are chosen fairly at random from MQ
- Packets can go from input frame directly into multi-queue

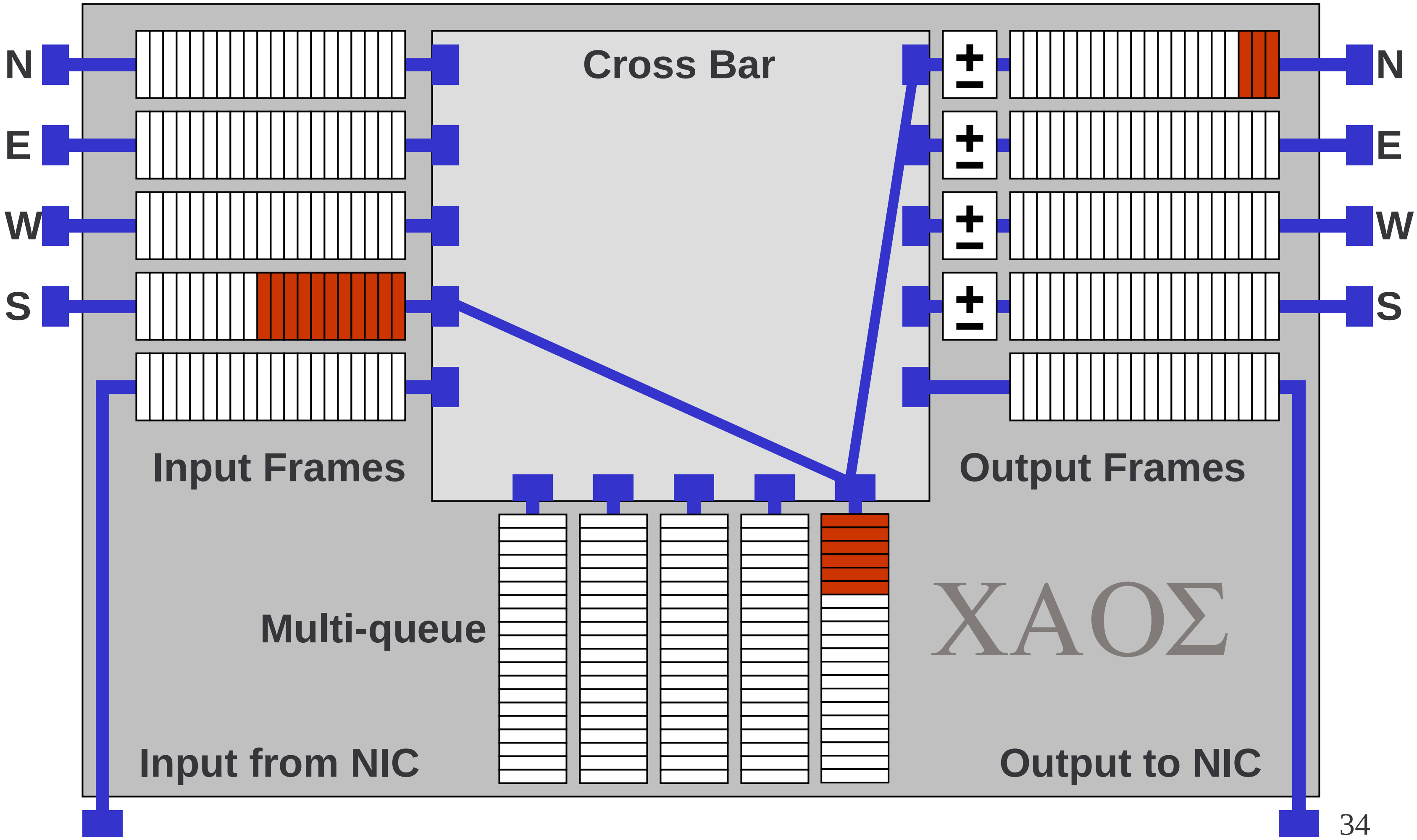
Connecting Channels with Virtual C-Thru



Waiting In The Multi-queue



Cutting Through Multi-queue



A Difficult Lesson

- Input packets can move directly into the multi-queue
- Requiring them to make their first move productively is an error
    - Causes starvation
    - Causes asymmetry in the network … and the asymmetry caused the router to slow down

Lesson: Randomizing, adaptive routers (probably all routers) want to be as regular and consistent and symmetric as possible … otherwise traffic will get “caught” and become hot

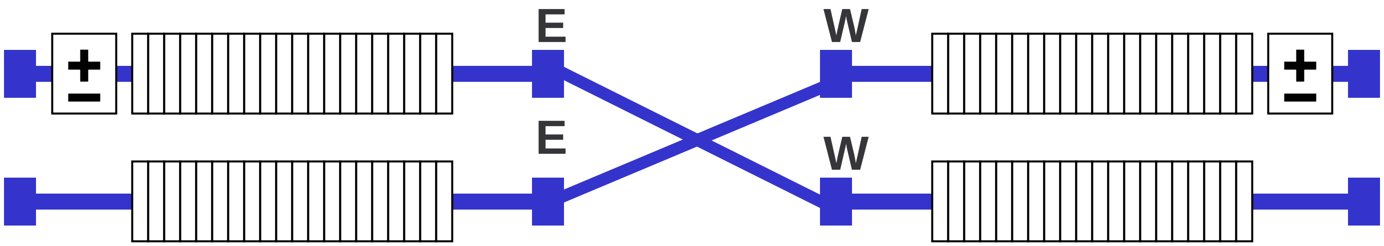
Deadlock

- Deadlock is a condition in which packets are permanently blocked
- Logically impossible for non-minimal routers because a packet can always deroute
- Physically, deadlock is a concern because packets monopolize resources
- Chaos is deadlock-free by using packet-exchange protocol: A router wanting to send a packet must be willing to receive a packet

Invariant -- 1 of the 4 frames is always available

Packet Exchange Protocol

- Outputs Connect to Inputs



Logically, a router should be able to accept a packet by moving it to the multi-queue … unless it is full … but it can't be full since one frame is always available … when it is used, some packet must deroute to make a frame available, but it *can* go because accepting gave the router the right to send!

Livelock

- Recall that livelock is the situation where a packet keeps moving, but is never delivered
- The standard technique is to use counting or timestamps to measure the “age” of a packet, and never let a packet get too old … everything is eventually delivered

  - Timestamps/counts take up valuable payload space
  - The number must be tested before a route is committed
  - Testing the number for old age is tougher than routing
  - Protecting against livelock was a showstopper for adaptive routers before Chaos

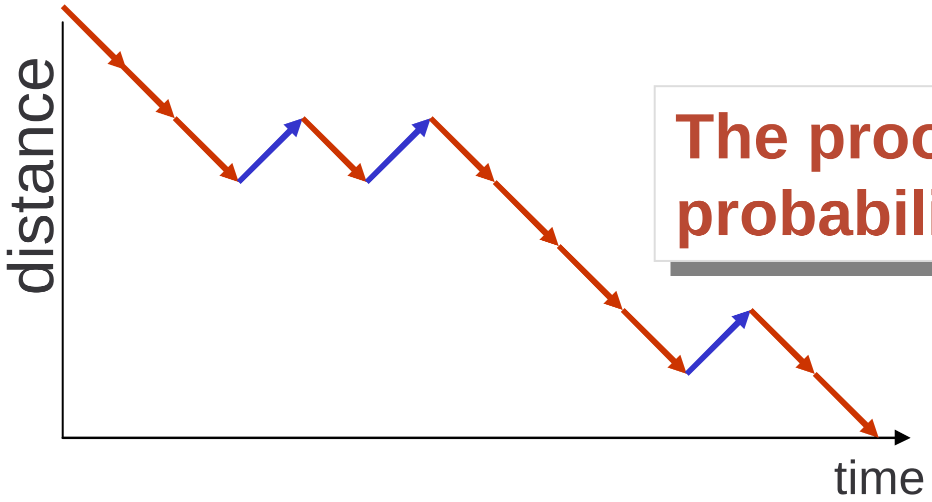
Livelock for Chaos

- Fact 1: Livelock is an extremely rare situation
- Fact 2: Chaos routers randomize routes, and randomizes the victim when picking a deroute
- So the strategy is, ignore livelock and gamble
- Problem: For any packet age, *t*, it is possible that a Chaos packet can be so unlucky that it is not delivered in *t* seconds
- Conclusion: Chaos Router is not livelock free

Probabilistic Livelock Freedom

A router is probabilistically livelock free if the probability that a packet remains in the network after *t* seconds goes to 0 as *t* increases

- Probabilistically livelock free ≈ deterministic in practice



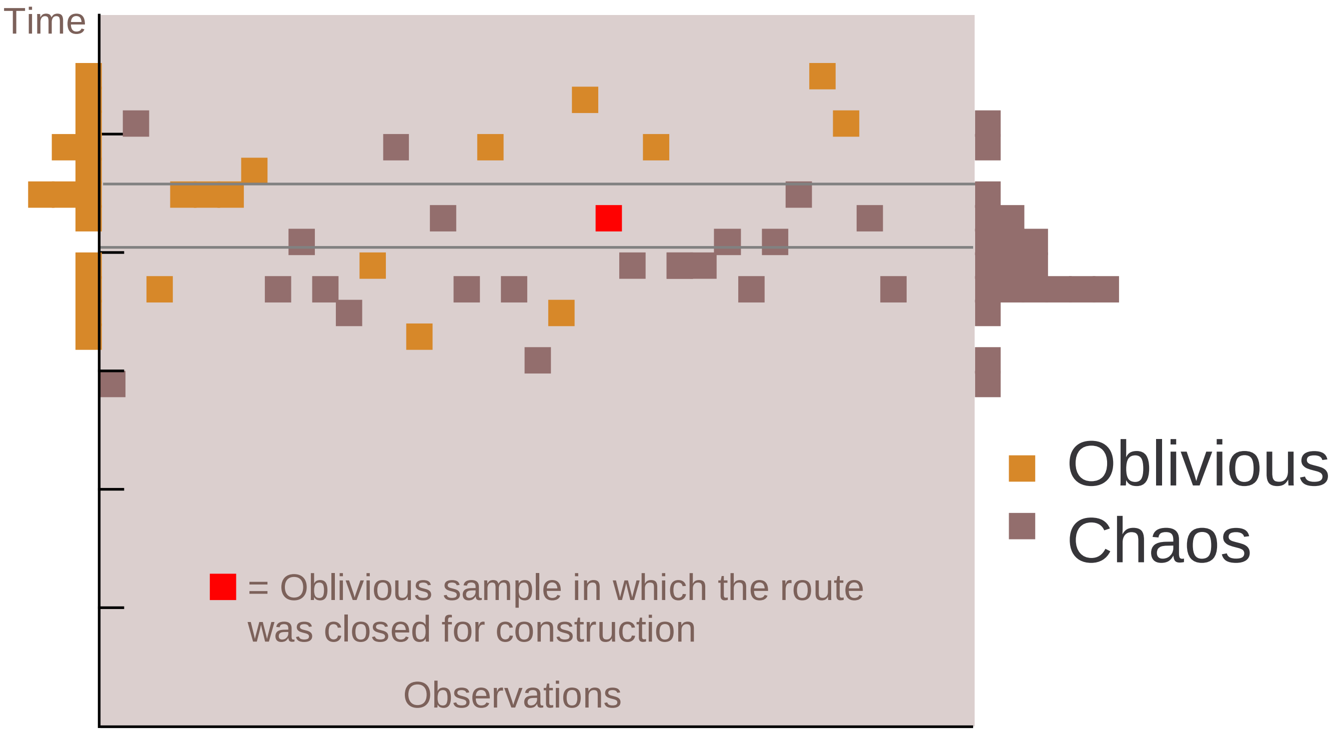
The proof requires a probabilistic argument

Experimental Commuting

Methodology

Adopt fixed shortest path oblivious routes between home & UW

When the clock parity was odd, I used an oblivious algorithm; otherwise, I used a Chaotic algorithm

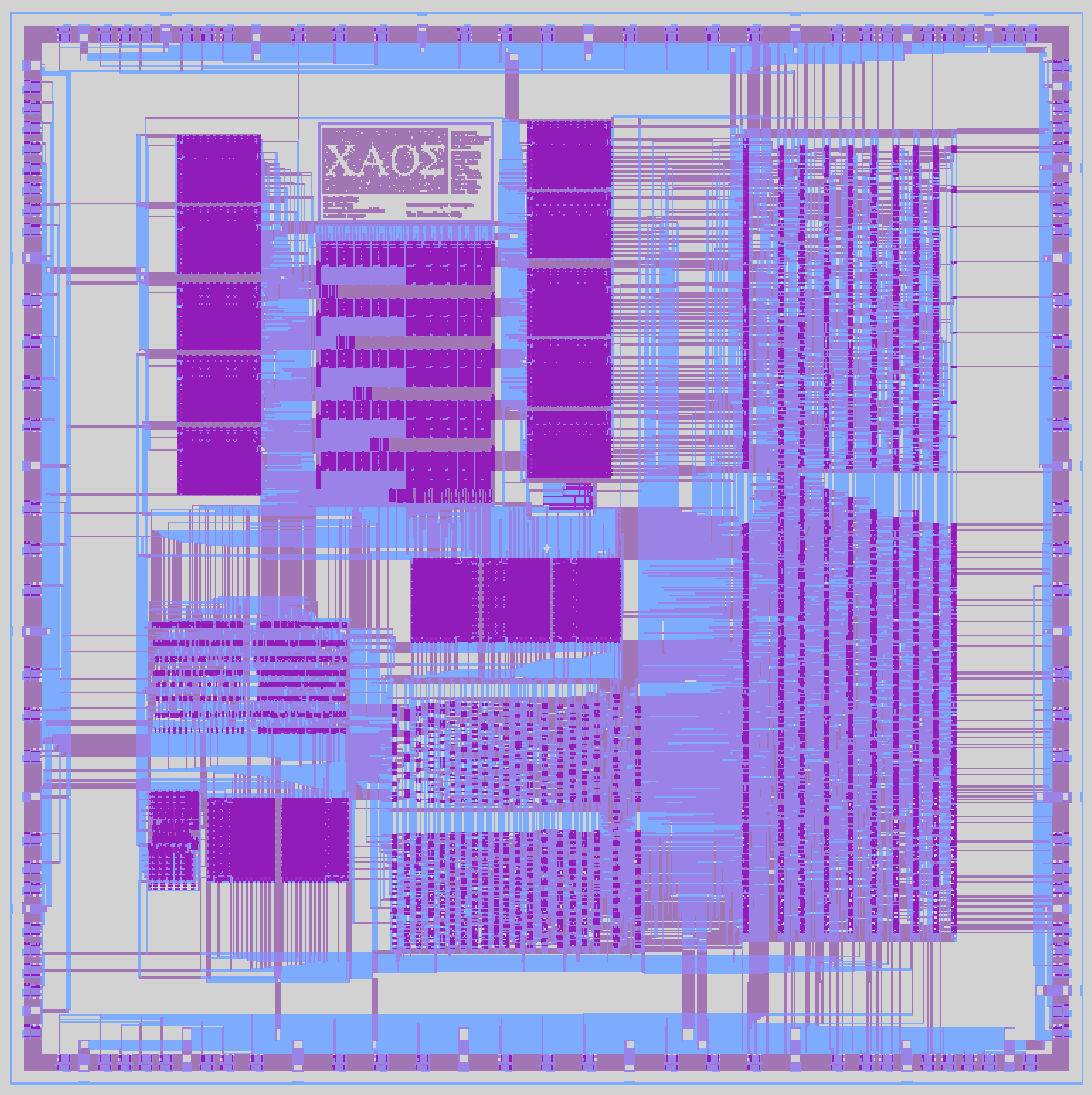


Implementation

- Designed by Kevin Bolding
    - Degree 4, suitable for mesh and torus routing
    - 20 phit frames, 16 bit phits, 5 frame multi-queue
    - Linear-feedback shift register for random numbers
    - Unidirectional channels with alternating opportunity at the end of each packet
    - Separate injection and delivery channels
    - Node latency is 4 ticks with 15 ns clock
    - Technology is 1.2 μ CMOS, with scalable design rules
    - Comparable to the Elko router, an oblivious router designed at Caltech using the same technology

Chaos Chip

The frames, scoreboard for the MQ logic, and cross bar were custom design



Timing

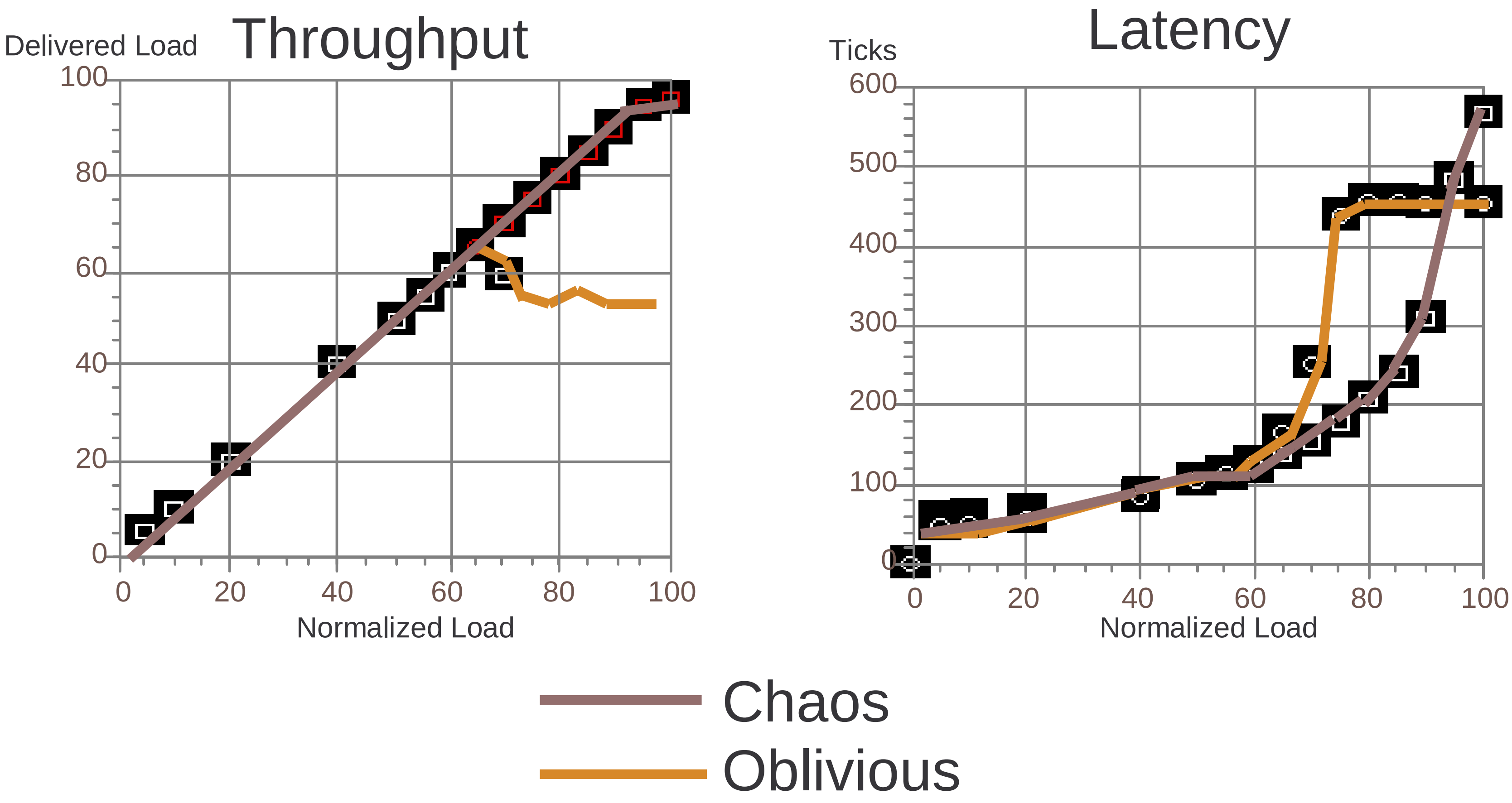
- Chaos Router has a 4 tick node latency
  - One tick to read the header and determine the productive channels
  - One tick to set up the cross bar switch
  - One tick to increment/decrement the address
  - One tick to travel across the wires
- The Elko Router has a 3 tick node latency
  - One tick to read the header and decide to stop/turn/go
  - One tick to increment/decrement the address
  - One tick to travel across the wires

Performance Assessment

- Performance Analysis by Melanie Fulgham
  - Chaos and Elko routers simulated at the flit level
  - Batched means computing 95% confidence intervals
  - Expected throughput -- proportion of the network bisection bandwidth that was used
  - Expected latency -- a packet’s injection to delivery time exclusive of source queuing
  - Learmonth-Lewis prime modulus multiplicative congruential pseudo random number generator
  - Traffic Patterns: random (all destinations equally likely, including self), transpose, bit-reversal, complement, perfect-shuffle, hot spots (10 destinations 4x more likely than random)

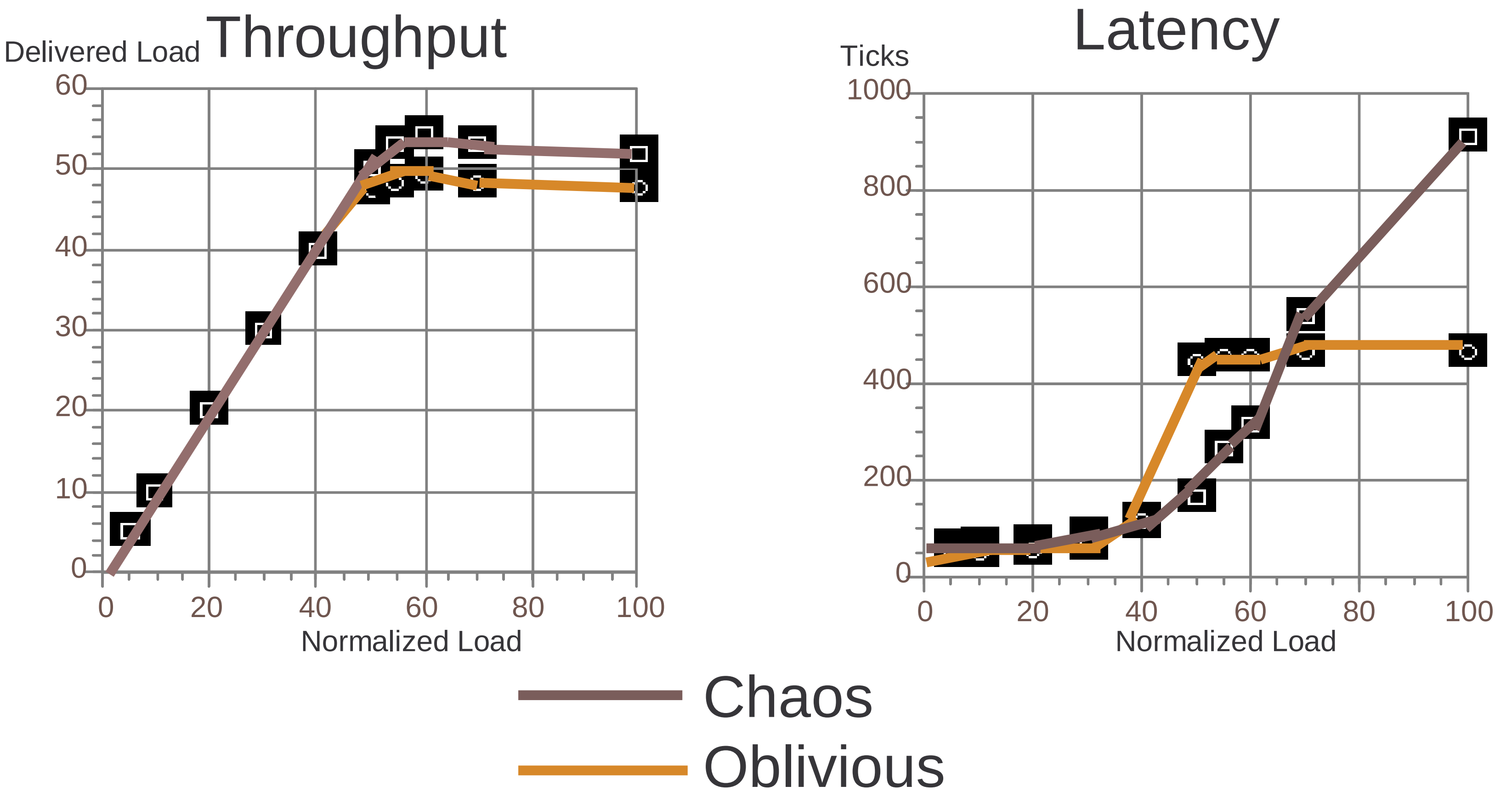
Elko has 3 tick node latency

Throughput and Latency



16x16 2-D Torus, Random Traffic

Throughput and Latency



16x16 2-D Torus, Transpose Traffic

Performance Description

- All performance numbers are given in Fulgham’s thesis + IEEE Computer Paper
- The main conclusions for 2D torus, 256 nodes
  - When 1 packet in network Elko is slightly faster gaining one tick per hop
  - Elko and Chaos are comparable to about 40% of BBW
  - Throughput peak: Elko is 60% of BBW, Chaos is 96% BBW on random traffic
  - Between 40%-90% of BBW, Chaos has significantly better latency
  - After 90% of BBW, Chaos has worse latency, but both routers are fully saturated and congested

Performance Description

- On other traffic patterns Chaos is usually better than Elko for both throughput and latency, though sometimes not by a lot
- Some patterns are inherently difficult
    - Transpose saturates at about 50% BBW
- Measuring the point at which saturation occurs -- Chaos is significantly better (often carries 2x the load) except for “bit reversal”

Bottom line: Chaos carries more traffic faster

Bursts

- Chaos router handles bursts well because the wires “fill up” carrying traffic quickly, and the multi-queue can hold “over capacity”
- In general Chaos works best at about 90% of saturation, when it is carrying great capacity and latency is still small
- A burst that floods the network beyond saturation, however, creates excessive amounts of derouting, wasting channel capacity

Chaos does not transmit “back pressure” well

Fault Tolerance

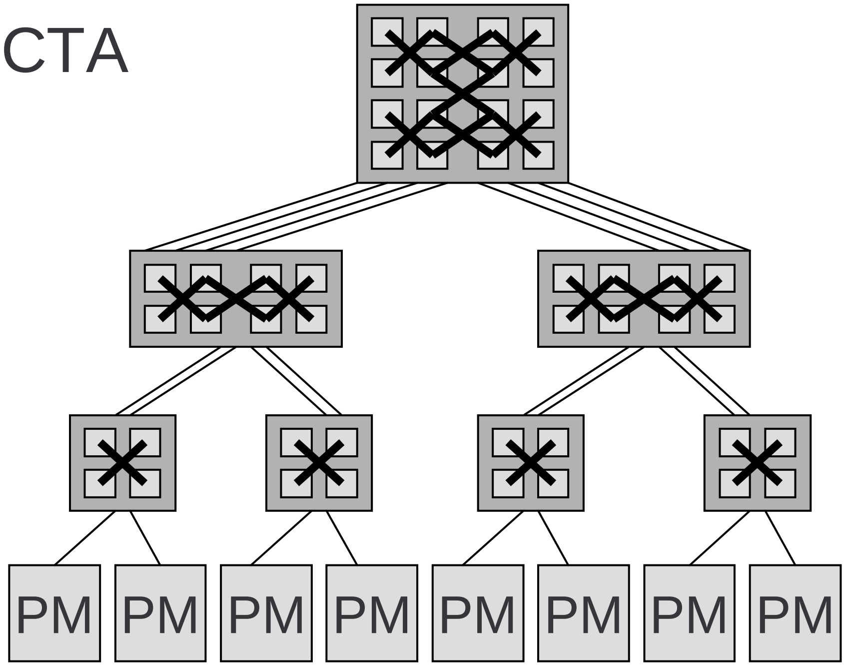
- Non-minimal adaptive routers are inherently fault-tolerant because a faulty neighbor is indistinguishable from a busy neighbor
- Chaos router requires special design considerations to be fault tolerant …
    - Back out of an output frame
    - Packet reassembly is a good place to recognize lost packets … but they may just be slow
    - Time out on reassembly, force a halt, deliver everything, apply diagnostics, restart if everything is OK

Do It Yourself Routing

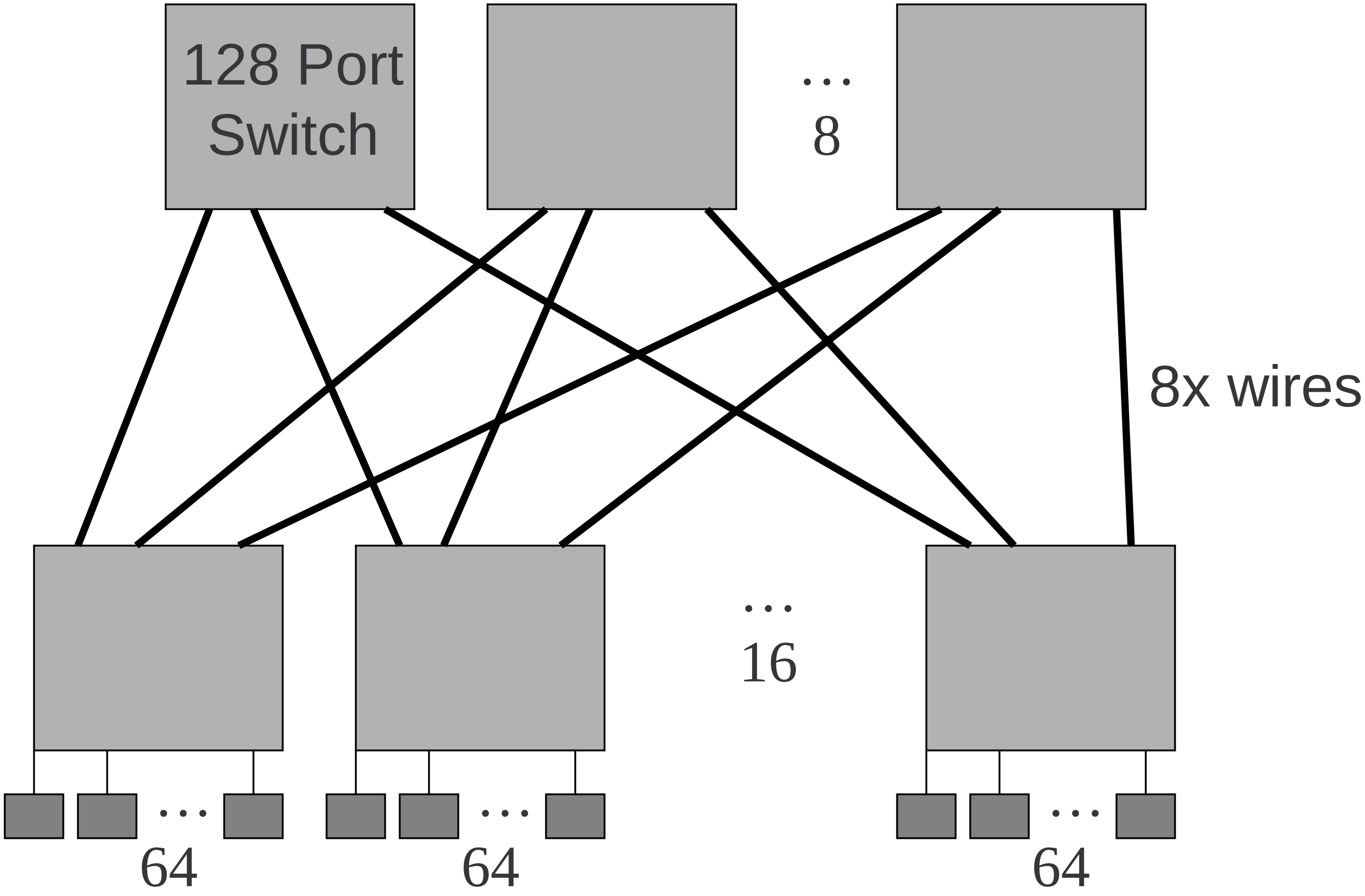
- The popularity of clusters has motivated "assemble-yourself" networking systems
  - Used as alternatives to etherNets, which are subject to bus (1-at-a-time) limitations
  - Modeled by the CTA
  - Technologies:
    - Myrinet
    - InfiniBand
    - Quadrics
  - Fat Tree is Key

Do It Yourself Routing

- The popularity of clusters has motivated “assemble-yourself” networking systems
  - – Used as alternatives to etherNets, which are subject to bus (1-at-a-time) limitations
  - – Modeled by the CTA
  - – Technologies:
    - Myrinet
    - InfiniBand
    - Quadrics
  - – Fat Tree is Key

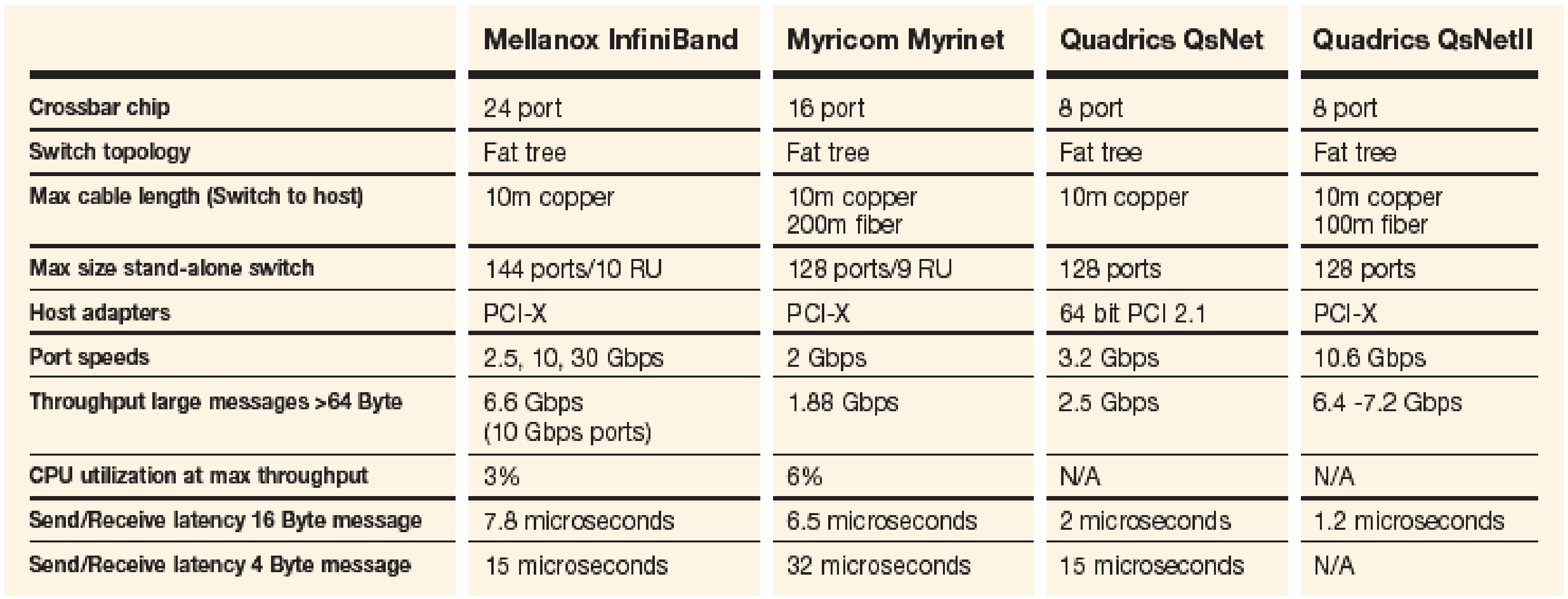


Example 1024 Processor Cluster



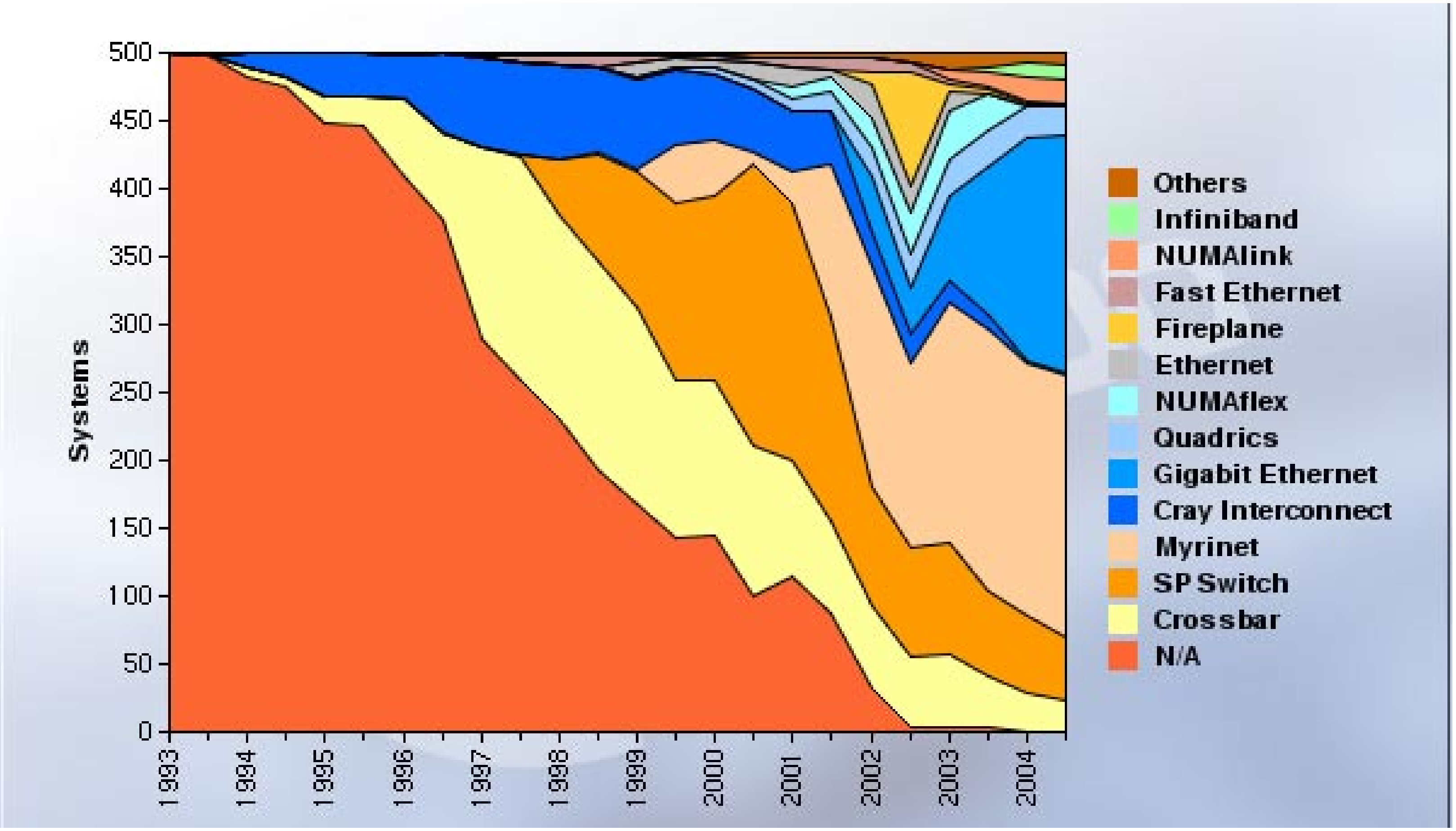
Comparison

- Latency is critical



	Mellanox InfiniBand	Myricom Myrinet	Quadrics QsNet	Quadrics QsNetII
Crossbar chip	24 port	16 port	8 port	8 port
Switch topology	Fat tree	Fat tree	Fat tree	Fat tree
Max cable length (Switch to host)	10m copper	10m copper 200m fiber	10m copper	10m copper 100m fiber
Max size stand-alone switch	144 ports/10 RU	128 ports/9 RU	128 ports	128 ports
Host adapters	PCI-X	PCI-X	64 bit PCI 2.1	PCI-X
Port speeds	2.5, 10, 30 Gbps	2 Gbps	3.2 Gbps	10.6 Gbps
Throughput large messages >64 Byte	6.6 Gbps (10 Gbps ports)	1.88 Gbps	2.5 Gbps	6.4 -7.2 Gbps
CPU utilization at max throughput	3%	6%	N/A	N/A
Send/Receive latency 16 Byte message	7.8 microseconds	6.5 microseconds	2 microseconds	1.2 microseconds
Send/Receive latency 4 Byte message	15 microseconds	32 microseconds	15 microseconds	N/A

Trends for Top 500 List



Summary

- Introduced network communication and routing
    - Circuit switched, packet switched, worm-hole switched
- Discussed the physical set-up
    - Unidirectional or bidirectional wires
- Routing Algorithms
    - Oblivious on a mesh
        - Hot spots and handling that problem with a torus
    - Randomized techniques
    - Adaptive techniques
        - Minimal adaptive -- busted
        - Hot Potato -- too eager
        - Chaos -- randomizing

> Chaos is a friend of mine
> -- *Bob Dylan*