

Adaptive Routing for Data Center Bridges

Cyriel Minkenber¹, Mitchell Gusat¹, German Rodriguez²

¹ *IBM Research - Zurich*

² *Barcelona Supercomputing Center*

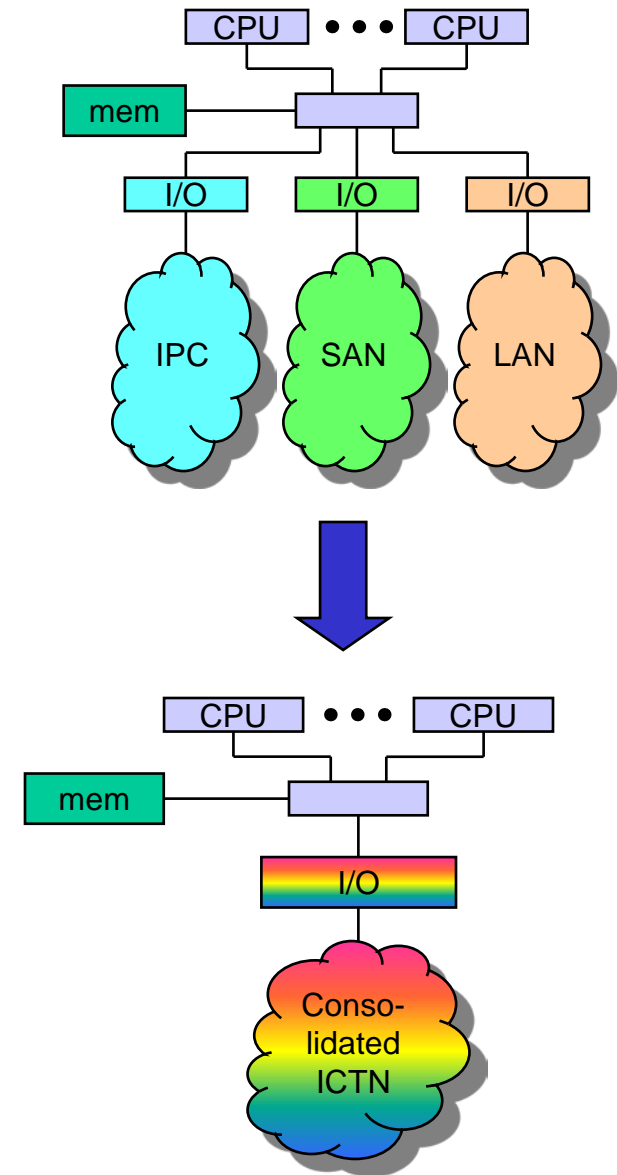


Overview

- Data center network consolidation
- Need for Ethernet congestion management
- Summary of IEEE 802.1Qau standardization effort
- Adaptive routing based on 802.1Qau
- Simulation results
- Conclusions

I/O Consolidation in the Data Center

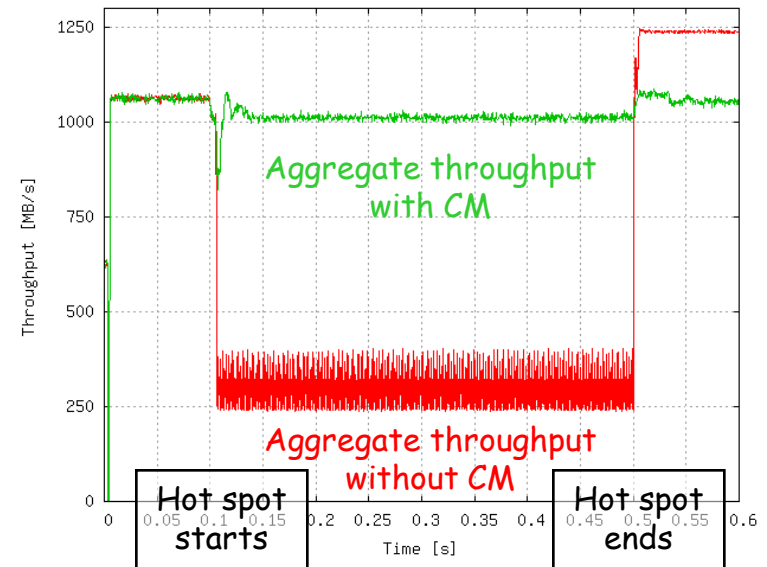
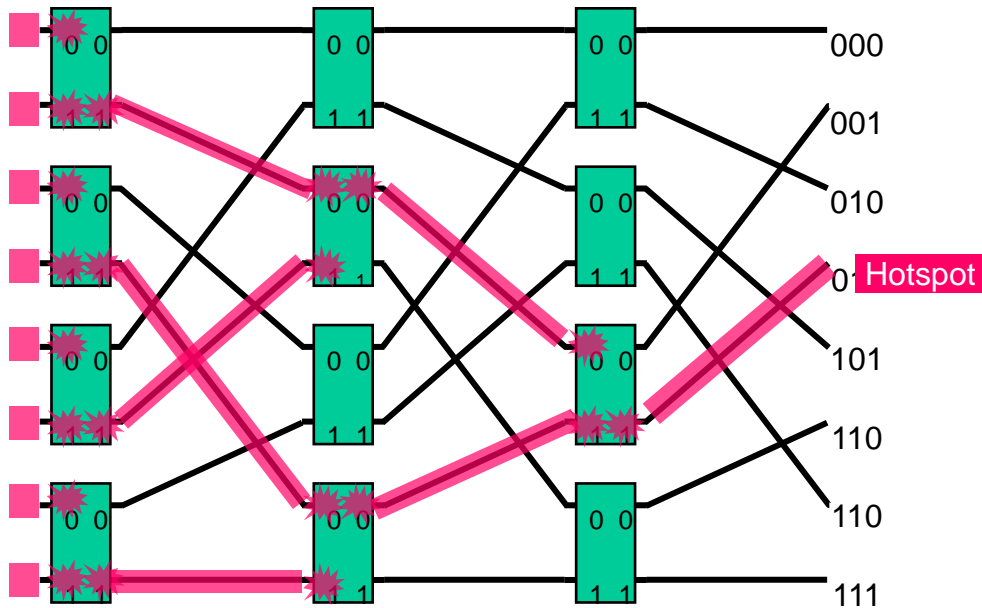
- Today's data centers comprise multiple networks
 - Expensive to build and maintain (TCO)
 - Inefficient use of resources (h/w, space, power)
- Convergence of all DC traffic onto "single wire"
- Conventional Ethernet
 - Philosophy: Keep it simple, low cost, good enough
 - Not optimized for performance: frame drops, throughput, latency, jitter, service-level differentiation
- Specialized networks filled the gap
 - Fibre Channel, InfiniBand, Myrinet, QsNet, PCIe AS
- Convergence Enhance Ethernet (CEE)
a.k.a. Data Center Bridging (DCB)
 - Enhance Ethernet to enable I/O consolidation in the data center
 - Save cost, power, overhead, space



The need for L2 congestion management

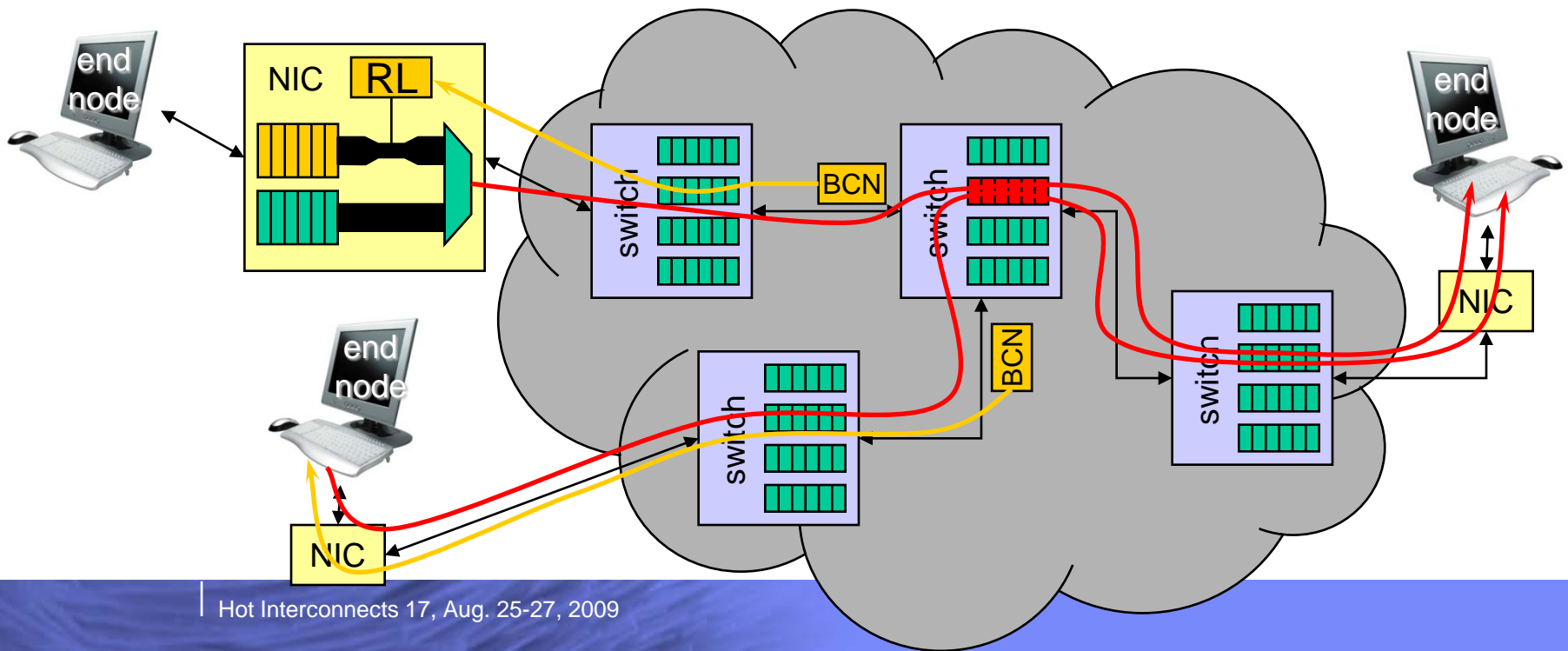
- Network congestion can lead to severe performance degradation
 - Lossy networks suffer from the “avalanche” effect: High load → drops → retransmissions → increased load → even more drops
 - Lossless networks suffer from *saturation tree congestion*: Link-level flow control (LL-FC) can cause congestion to roll back from switch to switch

- Congestion management (CM) is needed to deal with long-term (sustained) congestion
 - Dropping and LL-FC are ill suited to this task, dealing only with short-term (transient) congestion
 - **Push congestion from the core towards the edge of the network**



IEEE 802.1Qau congestion management framework

1. Congestion point (CP) = switch output queue
 - Sample output queue length every n bytes received
 - Equilibrium length Q_{eq}
 - Compute feedback: $F_b = Q_{off} - W * Q_{delta}$, where $Q_{off} = Q_{eq} - Q_{now}$ and $Q_{delta} = Q_{old} - Q_{now}$
2. Feedback channel
 - Convey backward congestion notifications (BCN) from CP to sources of "offending" traffic
 - Notifications contain congestion information: source address = switch MAC, destination address = source MAC of sampled frame, feedback: Q_{offset} and Q_{delta} , quantized with respect to $F_{b,max} = (1+2W) * Q_{eq}$ using 6-8 bits
3. Reaction point (RP)
 - Use rate limiters (RL) at the edge to shape flows causing congestion; separately enqueue rate-limited flows
 - Reduce rate limit multiplicatively when $F_b < 0$
 - Autonomously increase rate limit based on byte counting or timer (QCN; similar to BIC-TCP)
 - Release rate limiter when limit returns to full link capacity



Congestion management vs. adaptive routing

- CM solves congestion by reducing injection rate
 - Useful for saturation tree congestion, where many “innocent” flows suffer because of backlog of some hot flows
 - Does not exploit path diversity
 - Typical data center topologies offer high path diversity
 - Fat tree, mesh, torus

- Adaptive routing (AR) approach
 - Allow multi-path routing
 - By default route on shortest path (latency)
 - Detect downstream congestion by means of BCN
 - In case of congestion
 - First try to reroute hot flows on alternative paths
 - Only if no uncongested alternative exists, reduce send rate

Adaptive routing based on 802.1Qau CM

- Concept
 - Upstream switches **snoop** congestion notifications,
 - **annotate** routing tables with congestion information, and
 - **modify routing decisions** to route to the **least congested** port among those enabled for a given destination

- Routing table
 - Maps a destination MAC to one or more switch port numbers, listed in order of preference, e.g., shortest path first

- Congestion table
 - Maps a key <destination MAC, switch port number> to a congestion entry comprising the following information:

- Receiver checks frame order and performs resequencing if needed

field	type	meaning
congested	boolean	Flag indicating whether port is congested
local	boolean	Flag indicating whether congestion is local or remote
fbCount	integer	Number of notifications received
feedback	integer	Feedback severity value

Updating congestion table entries

- Upon reception (remote) or generation (local) of a congestion notification corresponding to a flow with destination MAC d on local switch port p with feedback value fb :

- if (`entry[d,p].local || !entry[d,p].congested`)
- `entry[d,p].local = local`
- `entry[d,p].congested = true`
- `entry[d,p].fbCount++`
- `entry[d,p].feedback += (fbCount*fb)`
- if (`! entry[d,p].local`) restart timer

- local congestion can be overridden by remote, but not vice versa

- ensures that newer notifications receive larger weight
- reduces effect of stale information and false positives
- frequent notifications lead to high congestion level

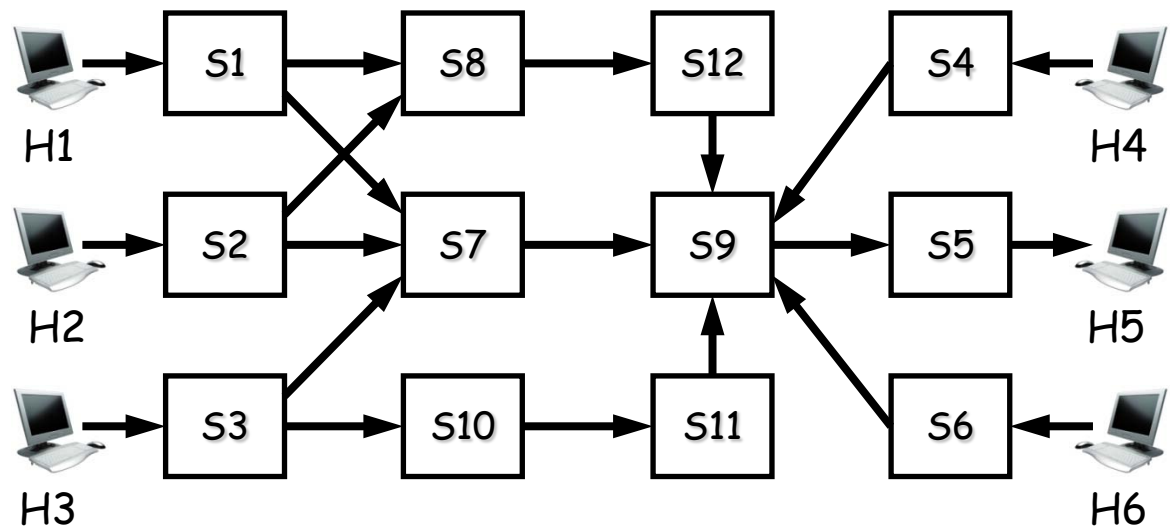
- When timer expires (remote) or queue level $\leq Q_{eq}/2$ (local):

- `entry[d,p].congested = false`
- `entry[d,p].local = false`
- `entry[d,p].fbCount = 0`
- `entry[d,p].feedback = 0`
- stop timer

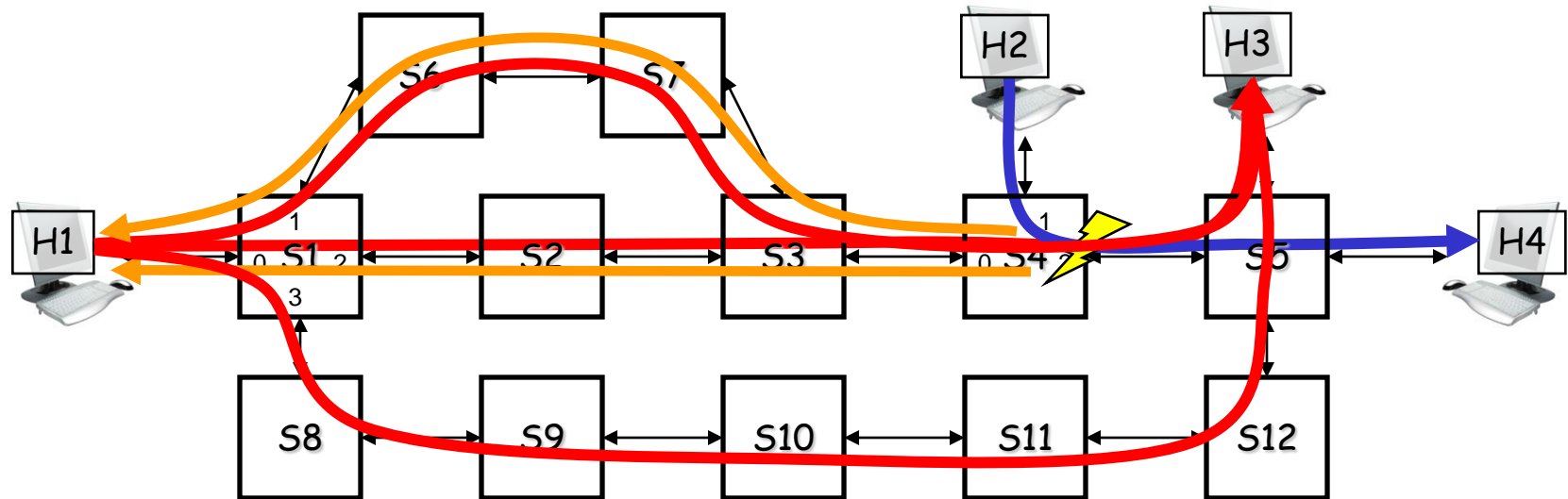
- QCN does not provide positive feedback; hence remote entries must be expired using a timer
- local entries are expired using a threshold on the local queue ($Q_{eq}/2$)

Routing decisions & configuration

- For a frame destined to MAC address d
 - try eligible ports in order of preference
 - select first port not flagged as congested
 - if all ports flagged as congested, route to port with minimum *feedback* value
- To ensure *productive* and *loop-free* routing without deadlocks
 - For each destination node n , construct a directed acyclic graph connecting all nodes $\neq n$ to node n



Routing of congestion notifications



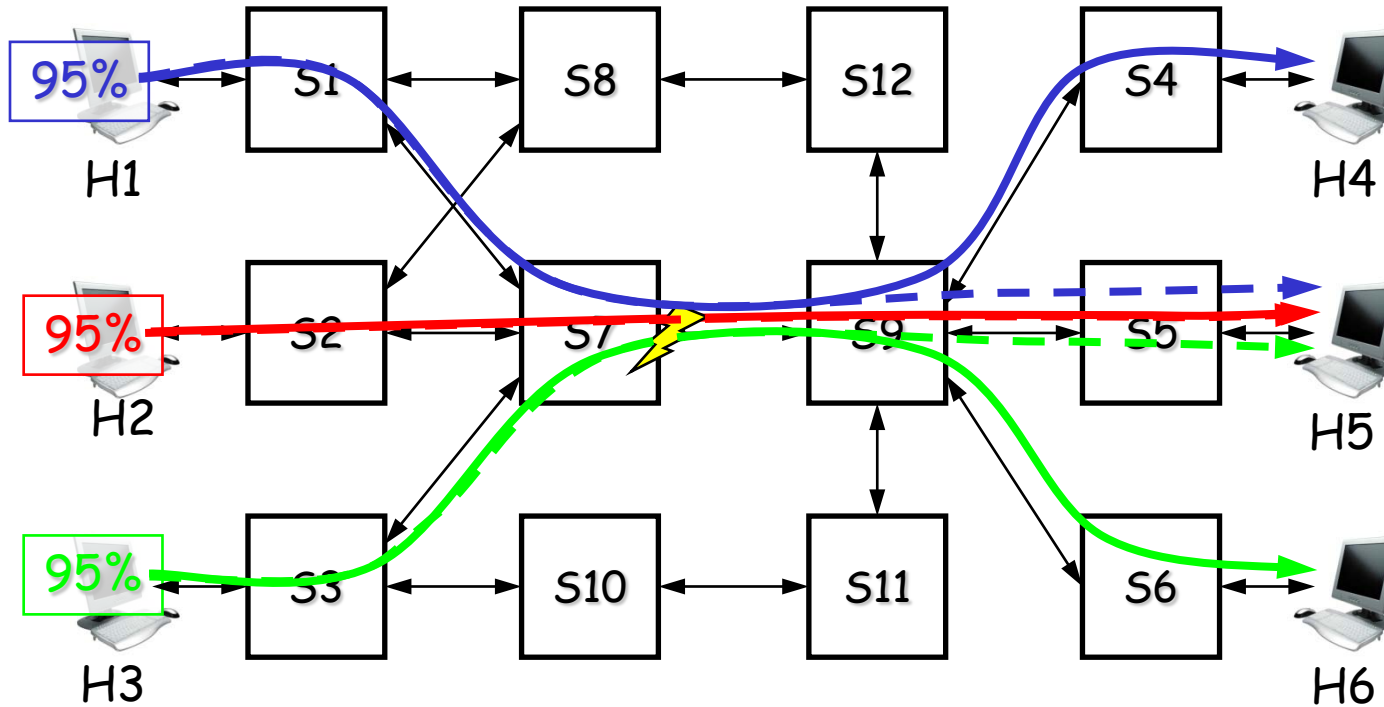
- Congestion notifications need to be routed on all alternative paths leading to the CP
- When generated at CP, notification is routed to port on which sampled frame arrived
- In upstream switches, notification is routed on a random port leading to the sampled flow's source

Simulation experiments

- “Artificial” topologies
 - Highlight problems of CM without AR
 - Test basic functionality of proposed AR scheme

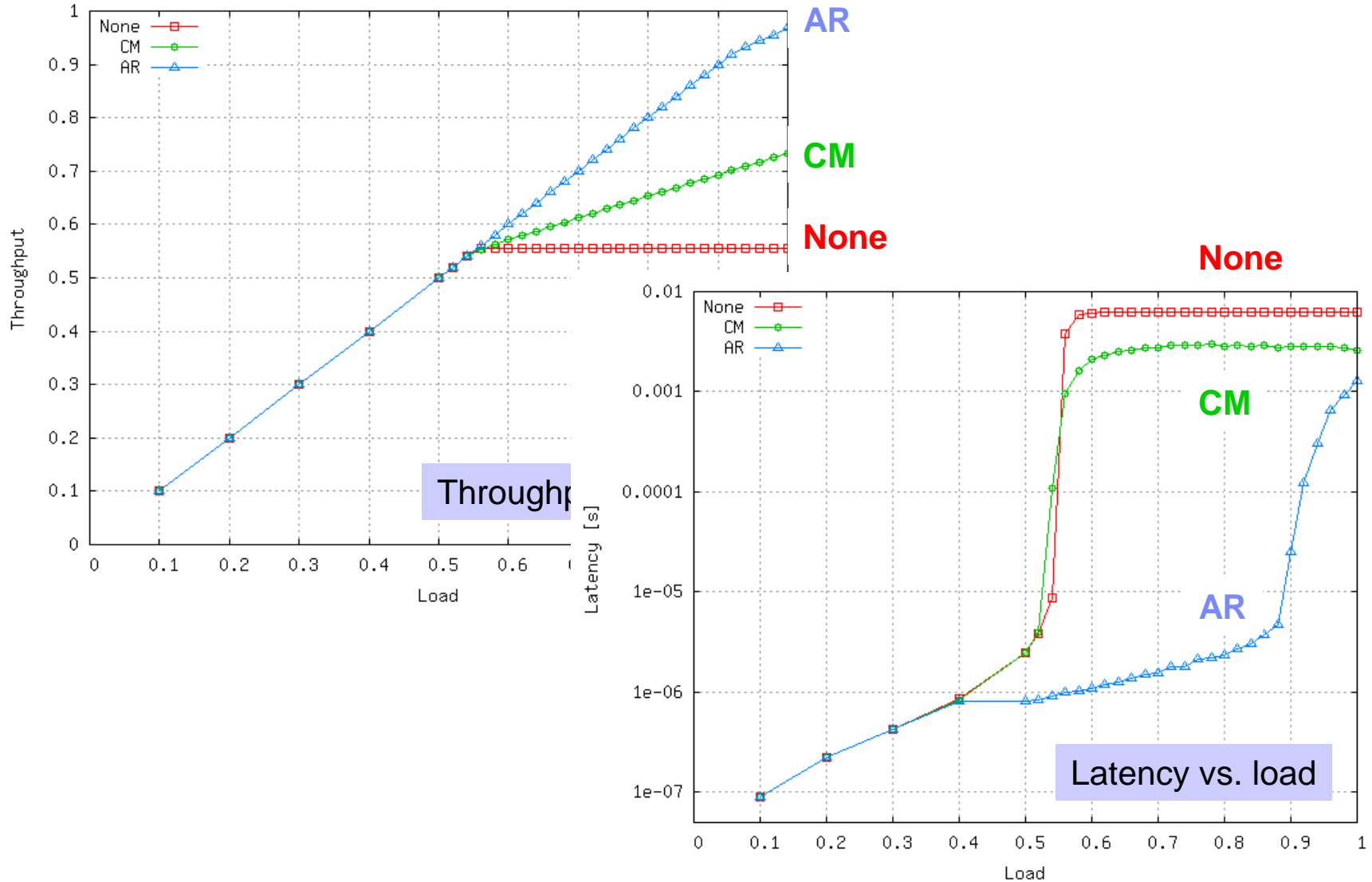
- Traffic patterns
 - Uniform Bernoulli traffic
 - Test saturation throughput
 - Latency-throughput characteristics
 - Scenarios with congestion solvable through re-routing
 - No end-point contention
 - Demonstrate AR functionality
 - Scenarios with congestion not solvable through re-routing
 - End-point contention
 - Demonstrate fallback CM functionality

Test topology



- Three flows (1 → 4, 2 → 5, 3 → 6) of 10 Gb/s each
- Shortest-path routing leads to contention in S7
- Adaptive routing should be able to provide 10 Gb/s to each flow
- Dashed lines = end-point contention; solid = no end-point contention

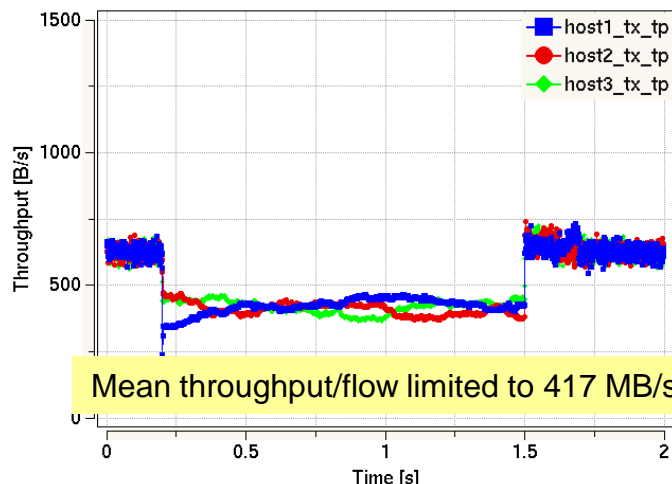
Results for uniform Bernoulli traffic



Results with 3-flow congestion scenarios

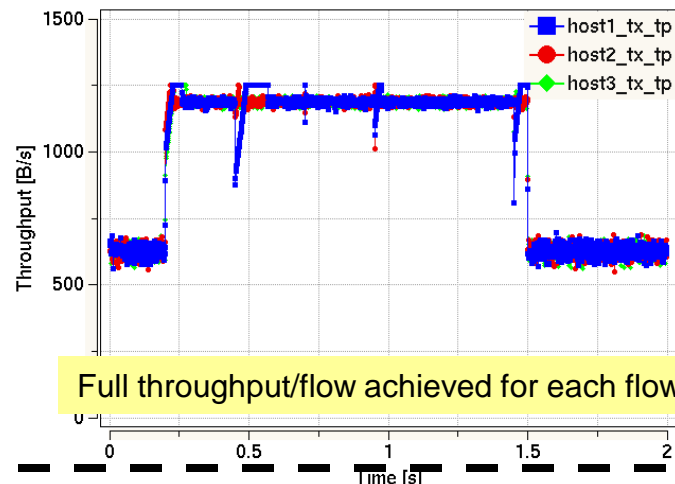
Without end-point contention

Flow throughput without adaptive routing



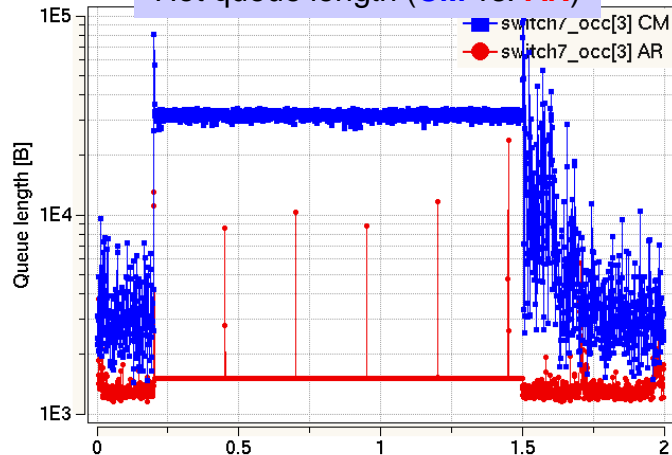
Mean throughput/flow limited to 417 MB/s

Flow throughput with adaptive routing



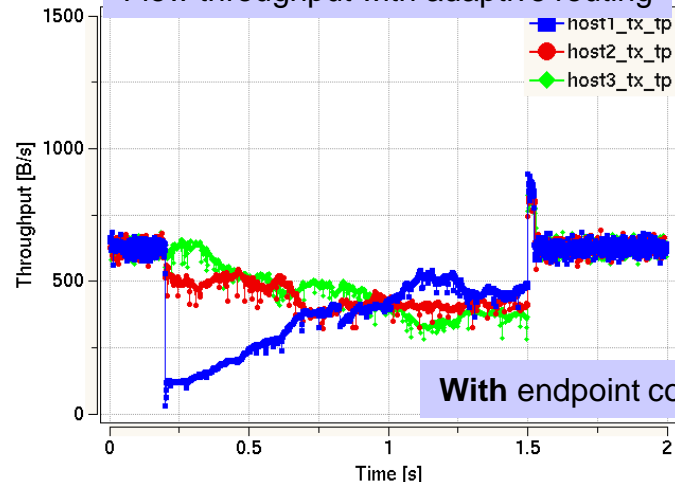
Full throughput/flow achieved for each flow

Hot queue length (CM vs. AR)



With CM, hot queue is controlled around equilibrium
 With AR, hot spot disappears owing to re-routing
 Reset spikes every 250 ms clearly visible

Flow throughput with adaptive routing



With endpoint contention

Hot queue lengths are controlled, indicating that CM still works well in conjunction with AR

Applications

- Simulated two applications from the NAS Parallel Benchmarks
 - Conjugate Gradient (CG), 128 nodes
 - Fast Fourier Transform (FT), 128 nodes
 - Characterized by heavy network loading during communication phases
 - Large messages (CG: 750 KB, FT: 130 KB)
 - Multiple sub-phases (CG: 5, FT: 127)
 - Symmetric permutation patterns in each sub-phase (each node sends to a distinct peer node in a pairwise fashion)

- Simulation methodology
 - Combination of Dimemas MPI trace-replay engine and Venus detailed network simulator
 - Parallel distributed event simulation (PDES) via client-server socket interface
 - Collect trace of application run on real machine (MareNostrum)
 - Co-simulation
 - Dimemas replays application trace
 - Dimemas forwards remote communication message to Venus
 - Venus performs flit-/frame-level network simulation of message
 - Venus returns message to Dimemas once completely received at destination NIC
 - Visualization using Paraver

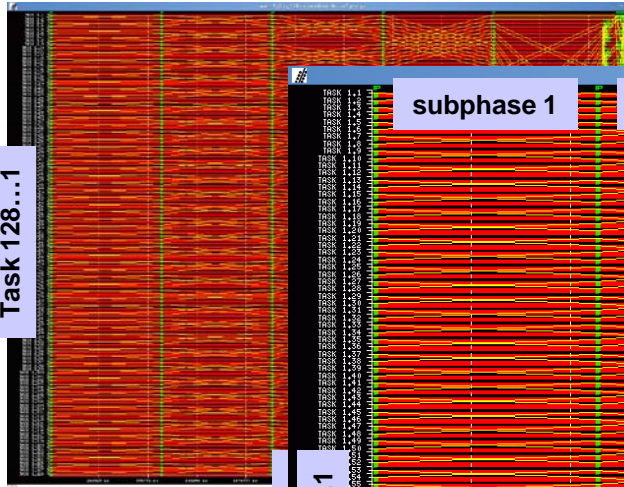
CG and FT communication patterns

Communication pattern

Traffic volume per node pair

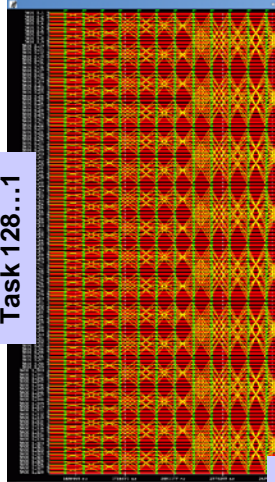
Conjugate Gradient

Task 128...1

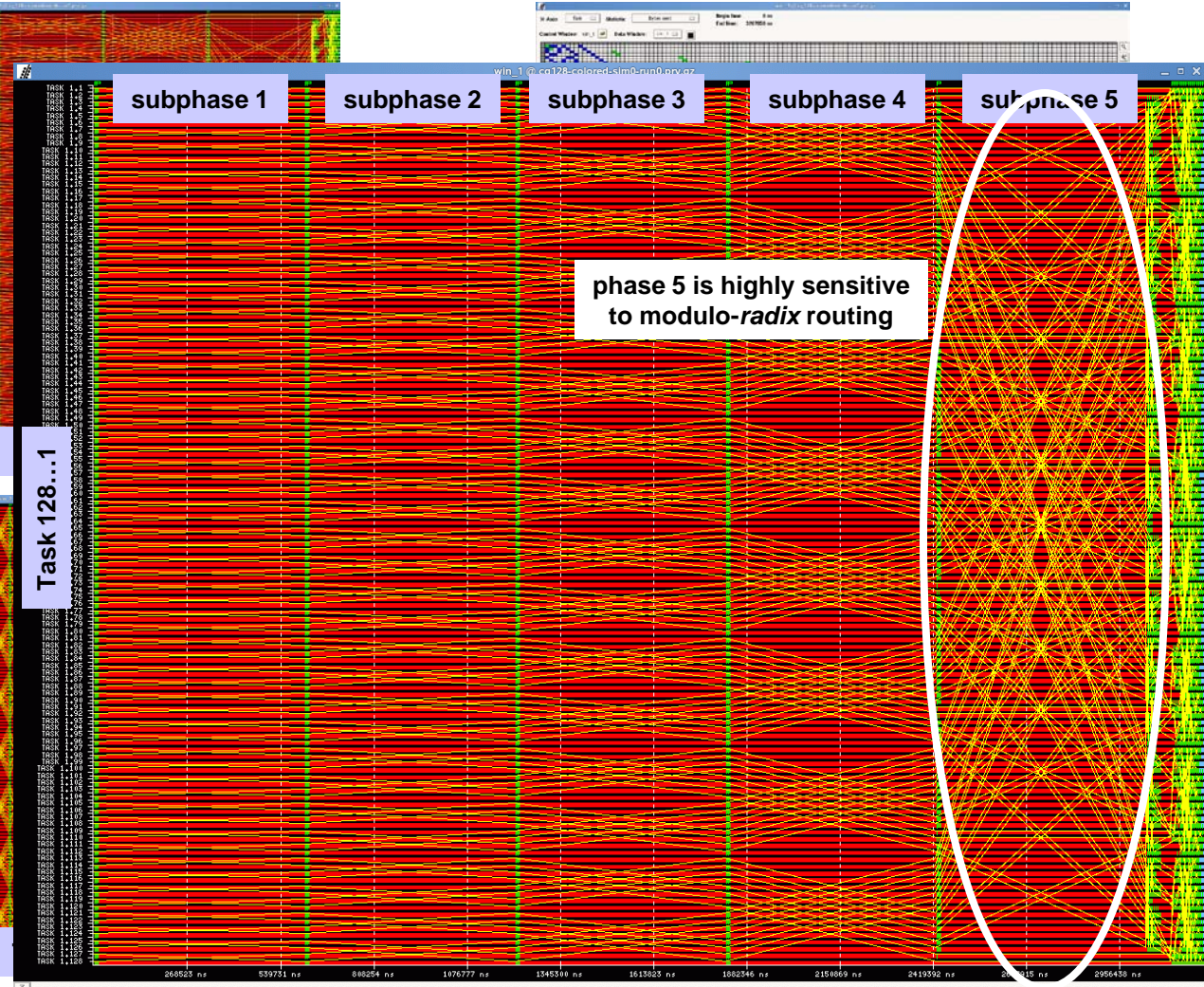


Fast Fourier Transform

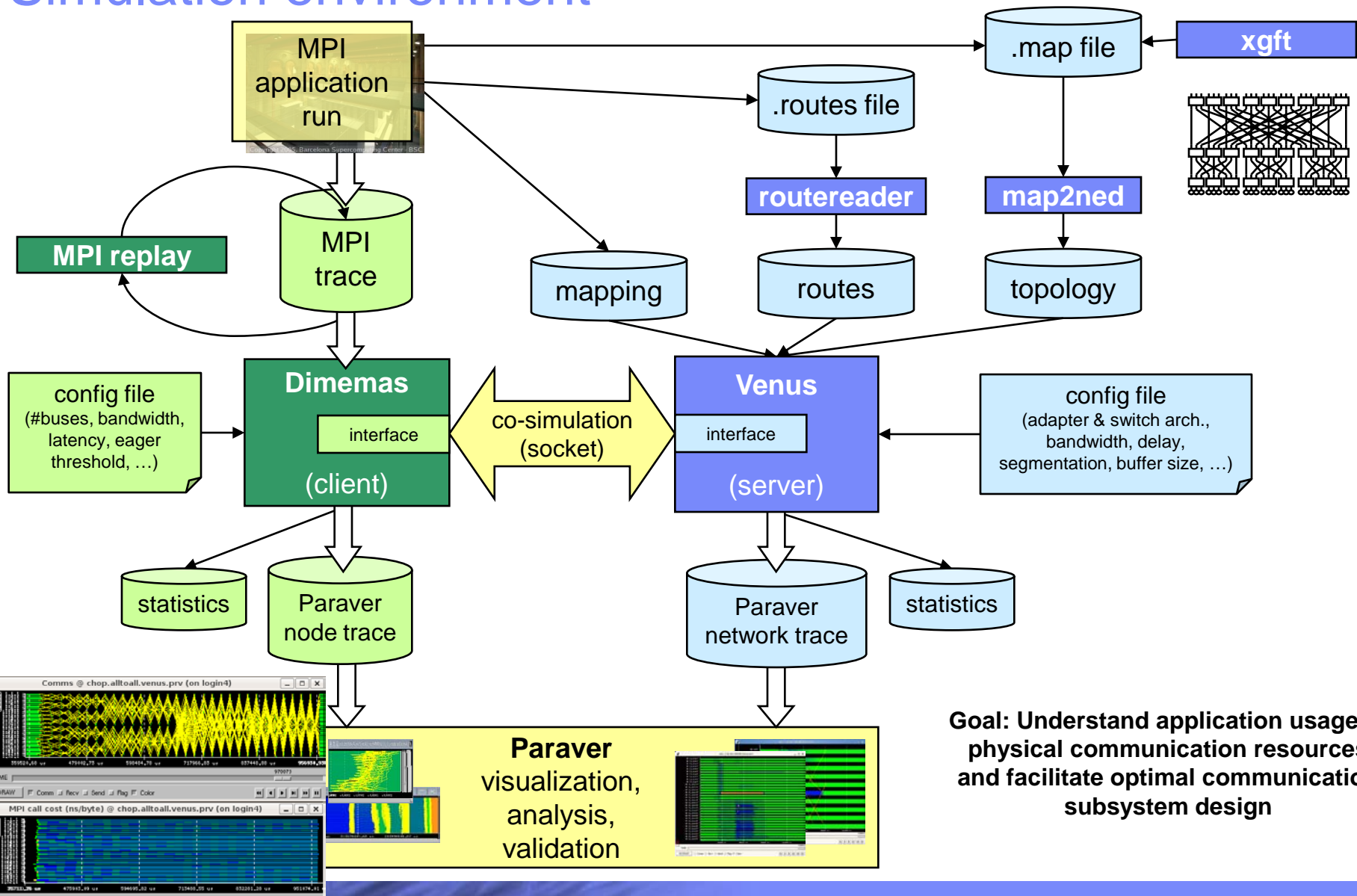
Task 128...1



Task 128...1



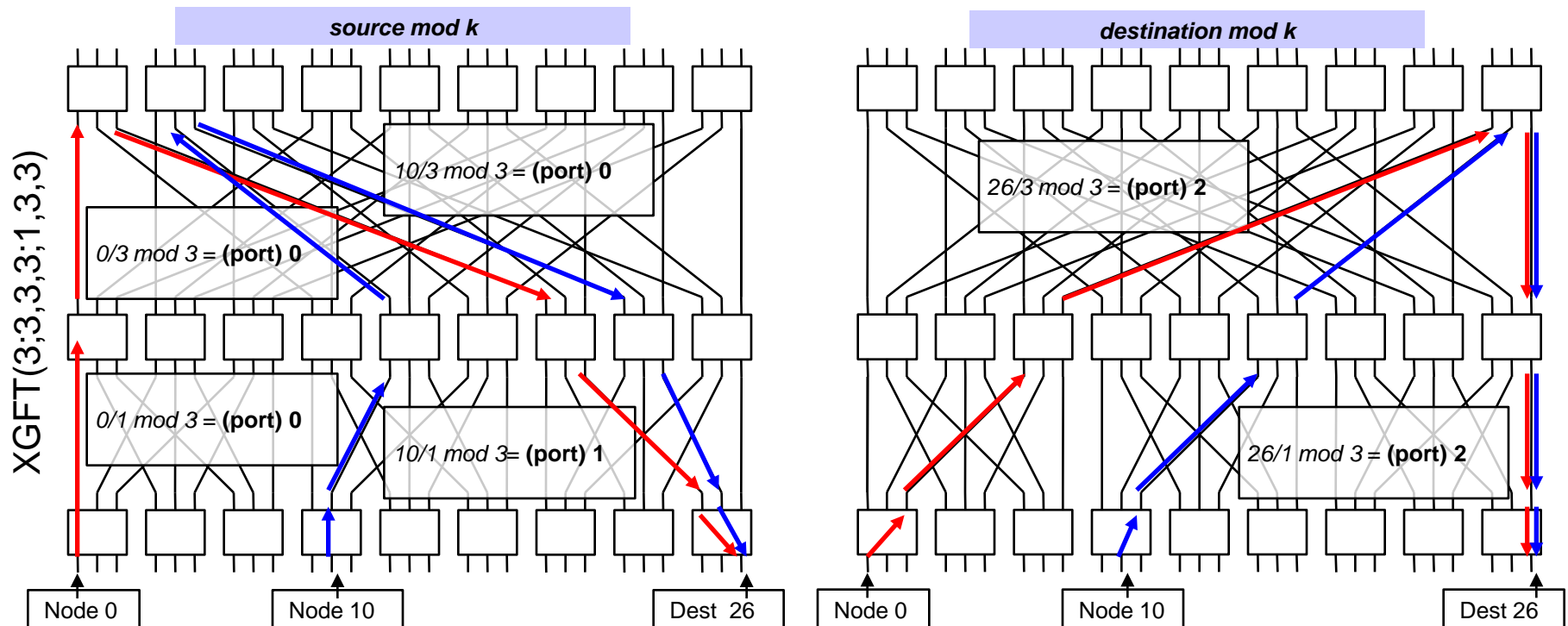
Simulation environment



Goal: Understand application usage of physical communication resources and facilitate optimal communication subsystem design

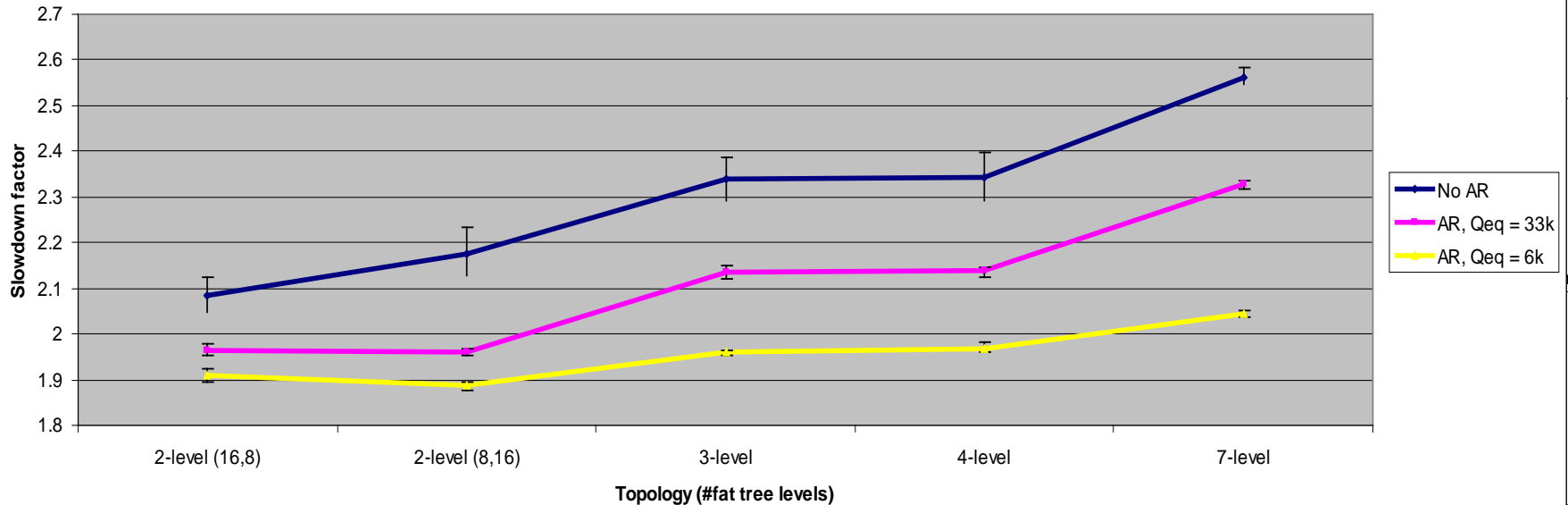
Extended generalized fat tree (XGFT) topology

- Multi-path: one path via each top-level switch
- Self-routing
- Usual static, oblivious routing method based on label of source or destination node to select path; can lead to significant contention
- Problem of assigning paths to connections with min. number of conflicts
 - Non-oblivious offline route optimization taking into account traffic pattern



Results

FT with random placement



- CG with linear task placement
 - Random, d-mod-, s-mod-m all perform about equally well
 - Colored is almost ideal
 - CM by itself is terrible
 - AR performs significantly better than oblivious algorithm (25 to 45% better than d-mod-m; within 25% of ideal)
- CG with random task placement
 - AR performs best of all schemes
- FT with random task placement
 - Smaller Qeq performs better; about 10 to 25% better than without AR

Conclusions

- IEEE 802.1Qau framework
 - defines **congestion management** for Ethernet data center bridging
 - addresses **saturation tree** congestion
 - provides **temporal** reaction: selectively rate-limiting sources
 - can improve, but also harm performance
 - provides sufficient **hooks** to implement fully **adaptive routing**

- Snoop-based adaptive routing (our proposal)
 - snoops notifications to annotate routing tables with congestion information
 - takes advantage of **multi-path** capabilities present in many HPC/DC topologies
 - significantly **reduces latency** and **increases saturation throughput** under uniform traffic
 - efficiently reroutes specific flows in case of congestion
 - interoperates seamlessly with underlying CM scheme
 - improves usefulness for selected parallel benchmarks on realistic topology

Questions?

The IEEE 802.1Qau Approach

■ Detection

- Congestion point = output queue
- Sample output queue length every n bytes received
- Equilibrium length Q_{eq}
- Compute offset: $Q_{off} = Q_{eq} - Q_{now}$
- Compute delta: $Q_{delta} = Q_{old} - Q_{now}$
- Compute feedback: $F_b = Q_{off} - W * Q_{delta}$
 $F_{b,max} = (1+2W) * Q_{eq}$

■ Signaling

- CP sends notification frame directly to source of sampled frame (BCN)
 - Source address = switch MAC
 - Destination address = source MAC of sampled frame
 - Feedback: Q_{offset} and Q_{delta} , quantized with respect to $F_{b,max}$ using 6-8 bits
 - Congestion point ID

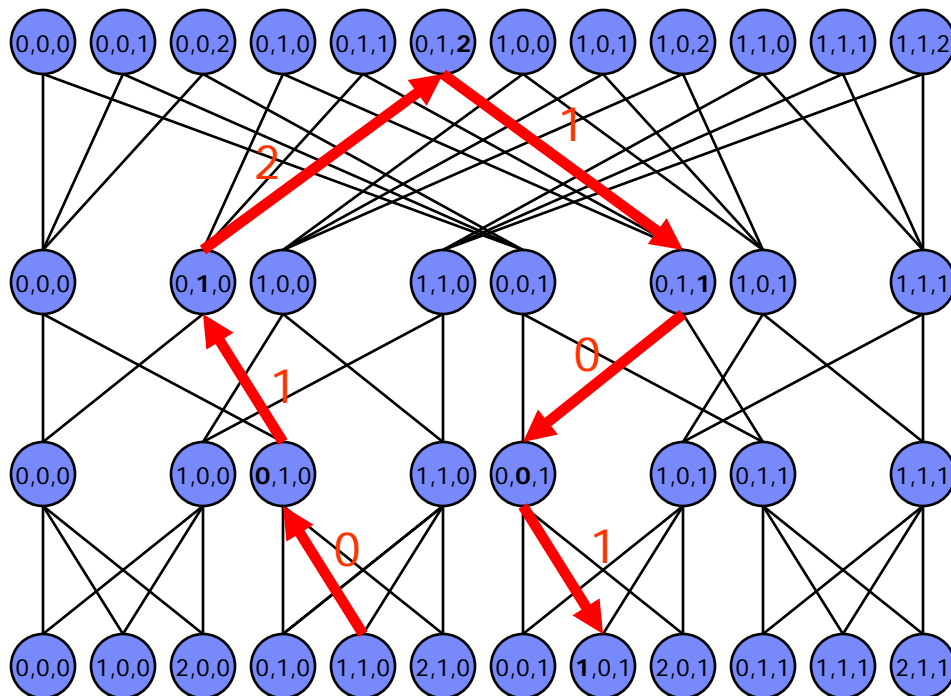
■ Reaction

- Instantiate rate limiter; separately enqueue rate-limited flows
- Reduce rate limit multiplicatively when $F_b < 0$
- Autonomously increase rate limit based on byte counting or timer (QCN; similar to BIC-TCP)
- Release rate limiter when limit returns to full link capacity

Extended Generalized Fat Trees (XGFTs)

- XGFT ($h ; m_1, \dots, m_h ; w_1, \dots, w_h$)
- $h =$ height
 - number of levels-1
 - levels are numbered 0 through h
 - level 0 : compute nodes
 - levels 1 ... h : switch nodes
- $m_i =$ number of children per node at level $i, 0 < i \leq h$
- $w_i =$ number of parents per node at level $i-1, 0 < i \leq h$
- number of level 0 nodes = $\prod_i m_i$
- number of level h nodes = $\prod_i w_i$

XGFT (3 ; 3, 2, 2 ; 2, 2, 3)



Conclusions

- Layer-2 DCB congestion management scheme being defined by IEEE 802.1Qau: QCN
 - Congestion notifications generated by switches, based on queue length offset and delta
 - Rate-limiting at the sources; multiplicative decrease, binary increase
 - Addresses **saturation tree** congestion

- We proposed an **adaptive routing** scheme based on QCN
 - Snoop notifications; annotate routing table; prefer least-congested path
 - Takes advantage of **multi-path** capabilities present in many HPC/DC topologies
 - Significantly **reduces latency** and **increases saturation throughput** under uniform traffic
 - Efficiently reroutes specific flows in case of congestion
 - Interoperates seamlessly with underlying CM scheme
 - Demonstrated usefulness for selected parallel benchmarks on realistic topology

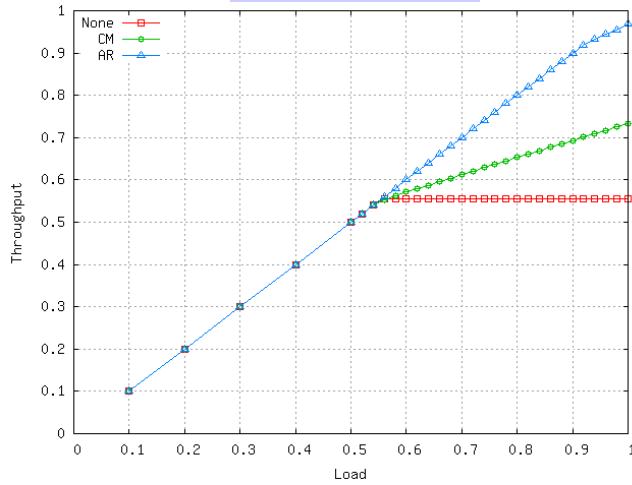
DC requirements not met by conventional Ethernet

- Fine-granularity link-level flow control (802.1Qbb)
 - PAUSE (802.3x) too coarse – stops/starts entire link at once
 - New draft PAR: “Priority-based Flow Control”
 - <http://www.ieee802.org/1/files/public/docs2007/new-cn-pelissier-draft-pfc-par-5c-070913.pdf>
- Bandwidth management (802.1Qaz)
 - Support for bandwidth management, virtual channels/lanes, deadlock-free routing, QoS
 - 802.1p not flexible enough
 - New PAR: “Enhanced Transmission Selection for Bandwidth Sharing Between Traffic Classes”
 - <http://www.ieee802.org/1/files/public/docs2007/new-cm-thaler-trans-select-par-0709-v2.pdf>
- Multi-pathing
 - Spanning tree provides *single* path from any source to any sink: Not suitable for typical multi-path ICTN topologies
- Congestion management
 - IEEE 802.1Qau

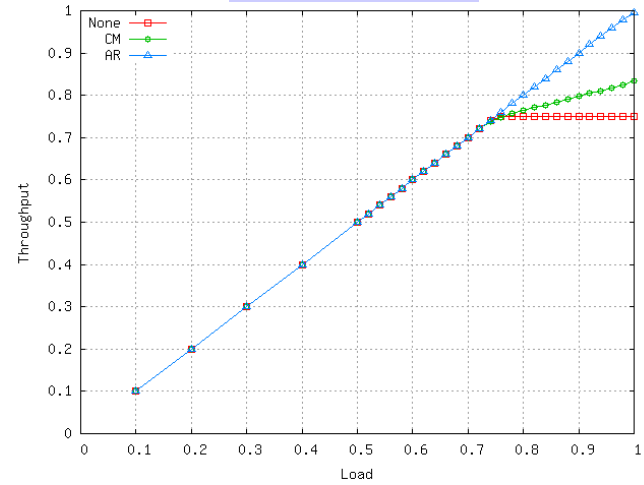
Uniform Bernoulli traffic

Topology 1

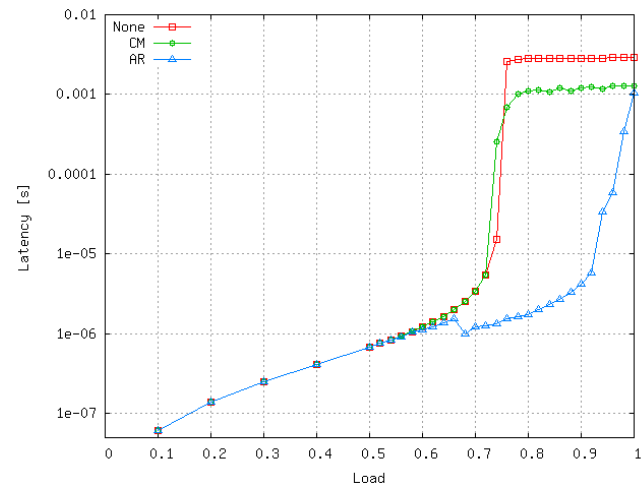
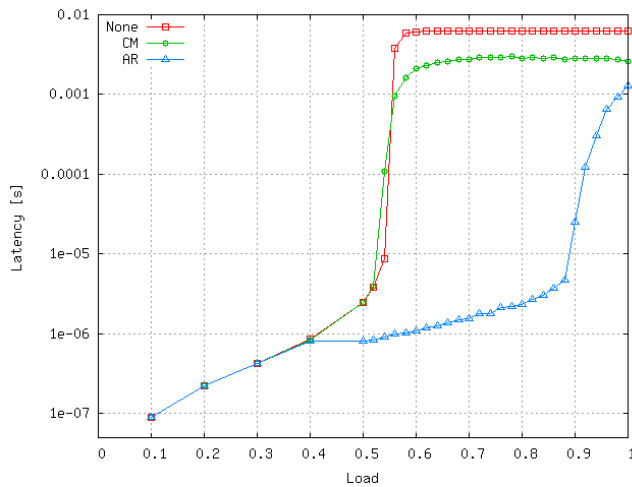
Throughput vs. load



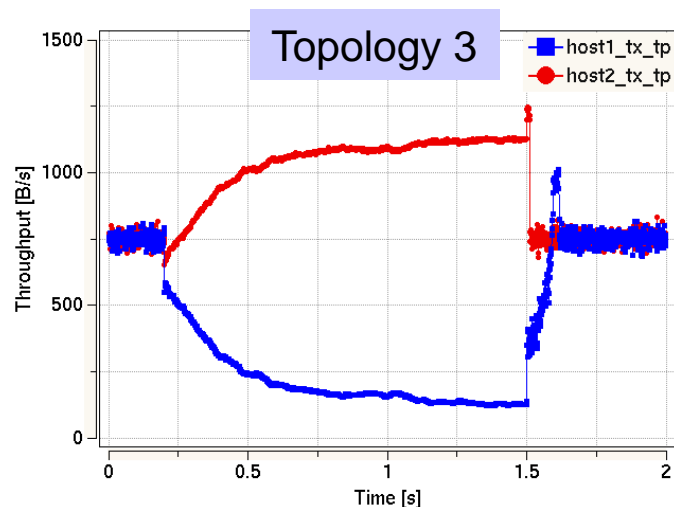
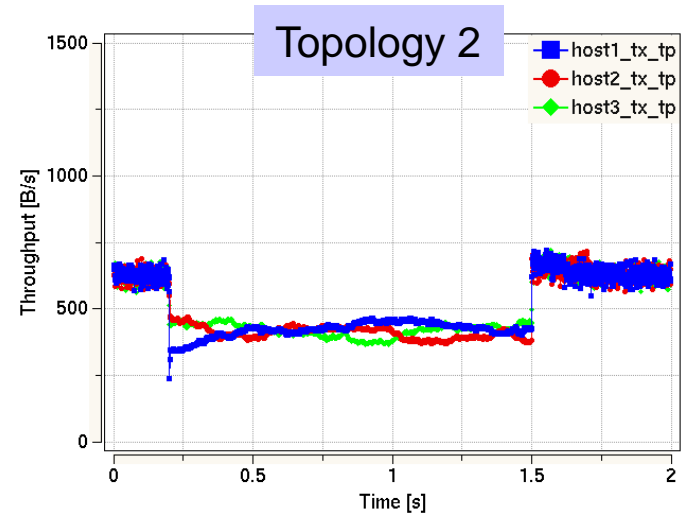
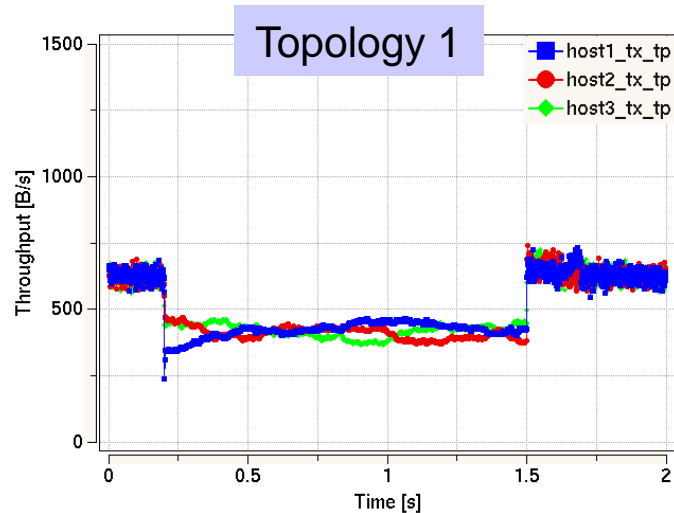
Topology 3



Latency vs. load

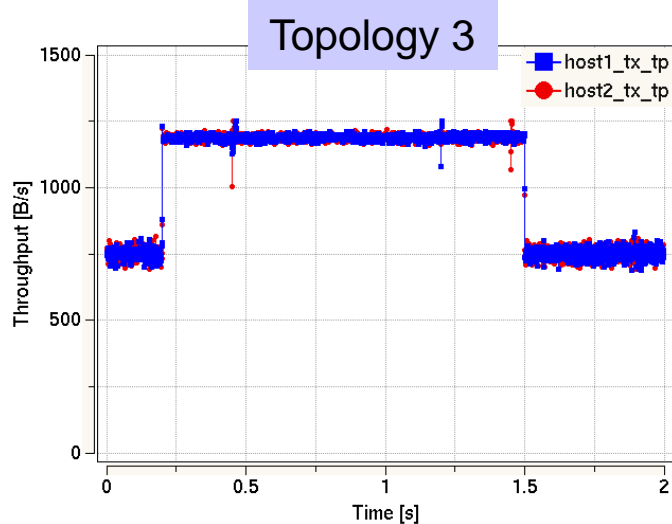
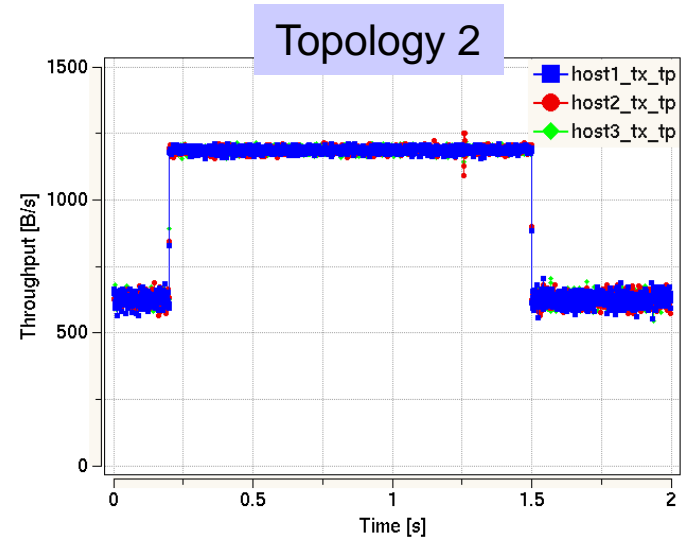
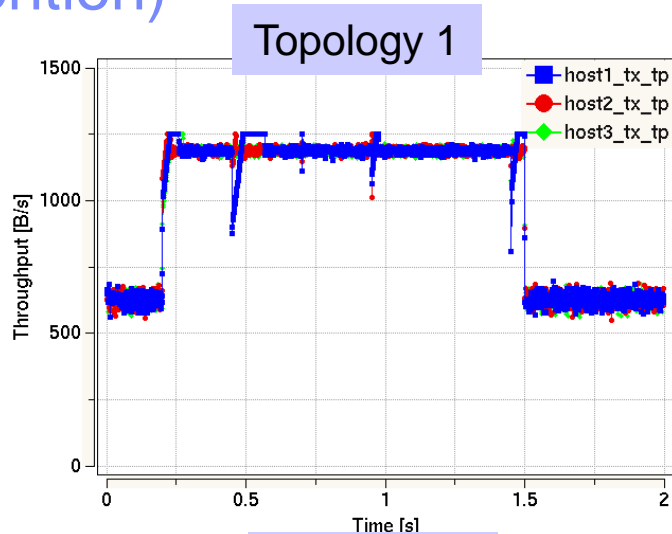


Flow throughput **without** adaptive routing (no end-point contention, CM only)



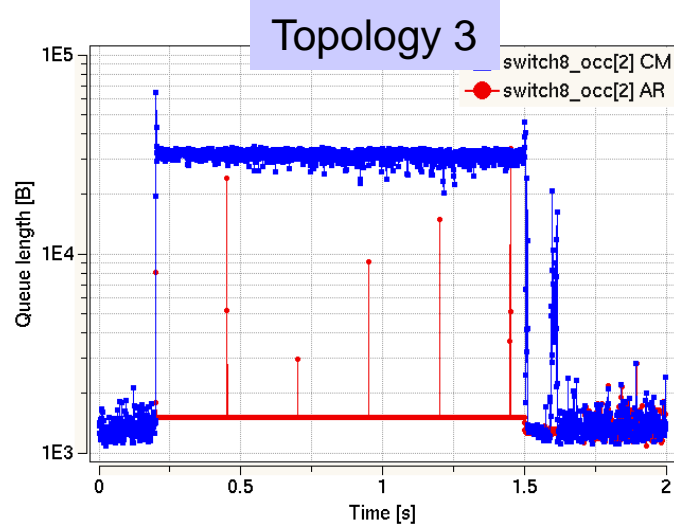
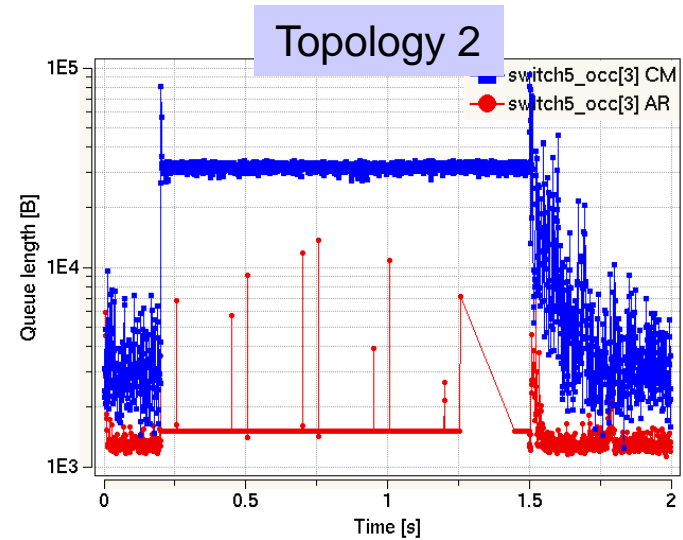
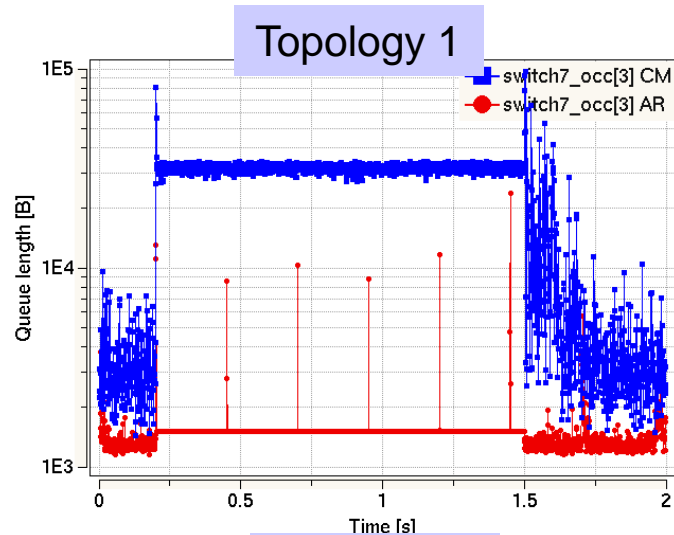
- Mean throughput/flow limited to 417 and 625 MB/s, respectively
- Significant unfairness in Topo 3
 - QCN has no inherent fairness criterium

Flow throughput with adaptive routing (no end-point contention)



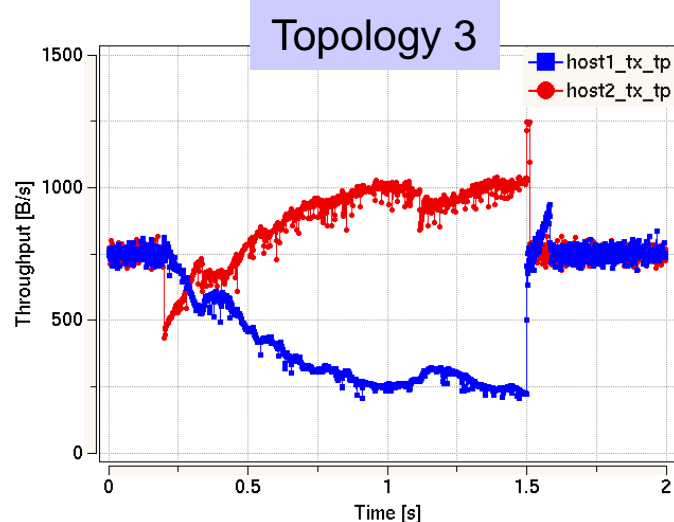
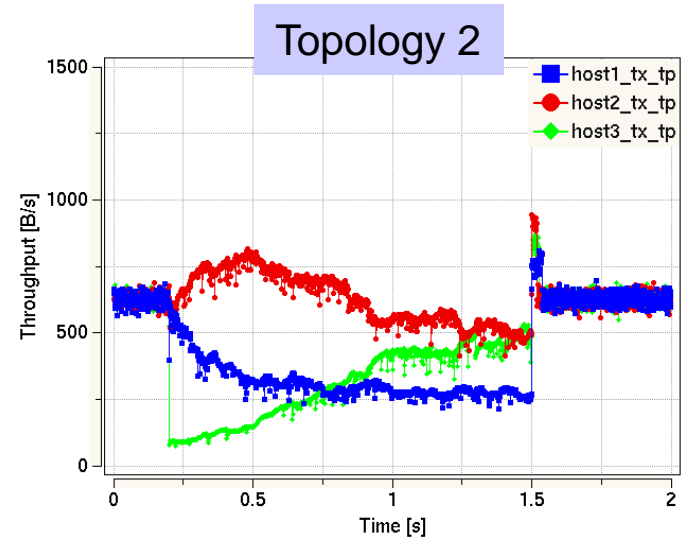
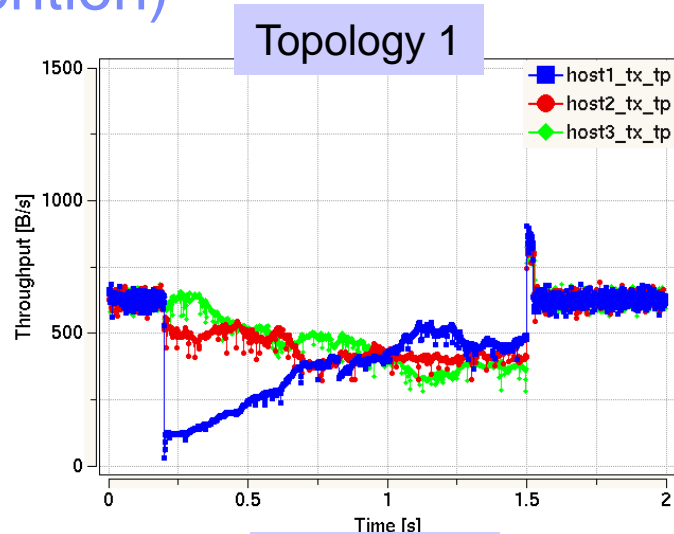
- Full throughput/flow achieved for each flow (1187 MB/s) in each topology
- Spikes occur when AR timer expires, i.e., reset of congestion entries every 250 ms

Hot queue length (CM vs. AR, no end-point contention)



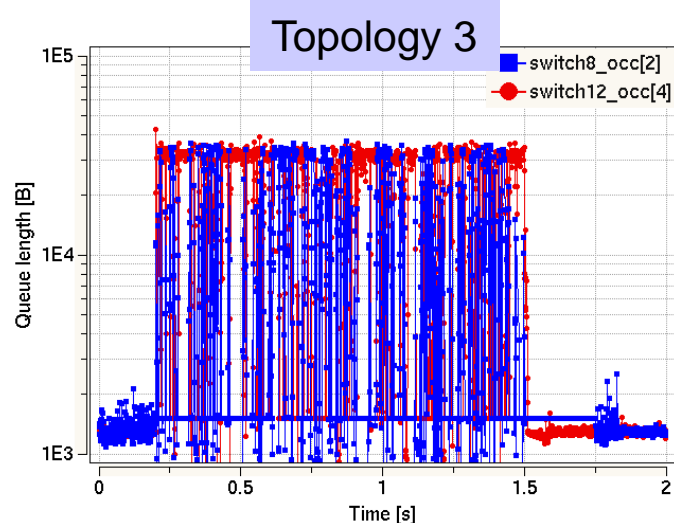
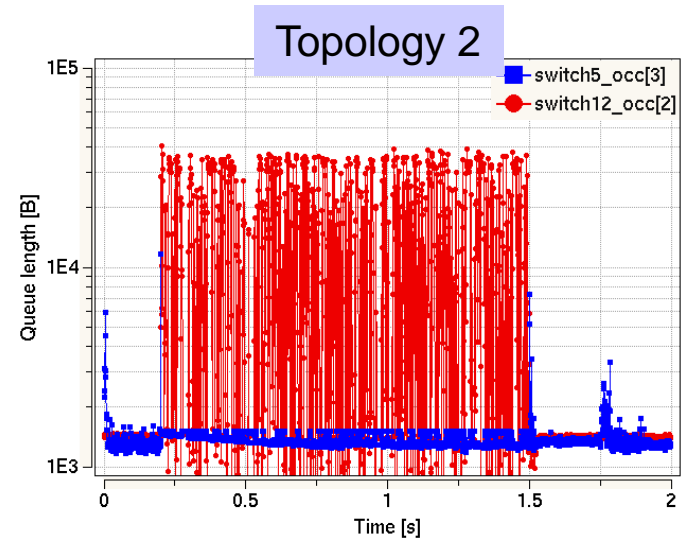
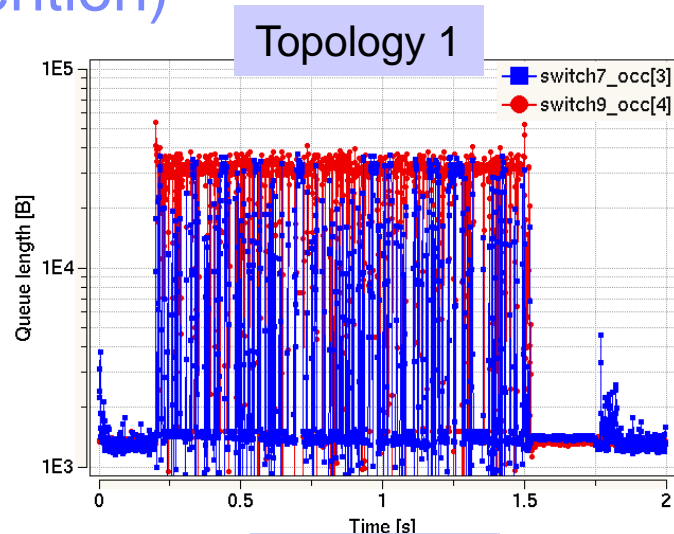
- With CM, hot queue is controlled around equilibrium
- With AR, hot spot disappears owing to re-routing
- Reset spikes every 250 ms clearly visible

Flow throughput with adaptive routing (with end-point contention)



- End-point contention is the limiting factor; AR does not help here
- However, CM still works properly
- Mean throughput/flow limited to 417 and 625 MB/s, respectively, as in CM case

Hot queue length with adaptive routing (with end-point contention)



- Hot queue lengths are controlled, indicating that CM still works well in conjunction with AR