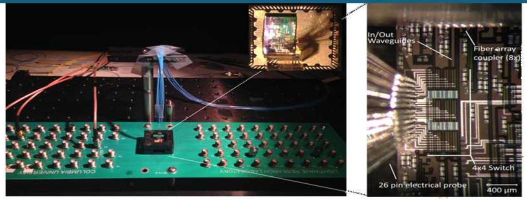
SAND2019-10349PE

Evaluation of novel interconnect technologies for ASC applications



PRESENTED BY

Jeremiah Wilke, Sandia National Labs

Giorgis Georgakoudis, Lawrence Livermore National Lab

Trilab Co-Design L2 Review, September 5, 2019

SAND X-2019

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Milestone Descripton and Completion Criteria

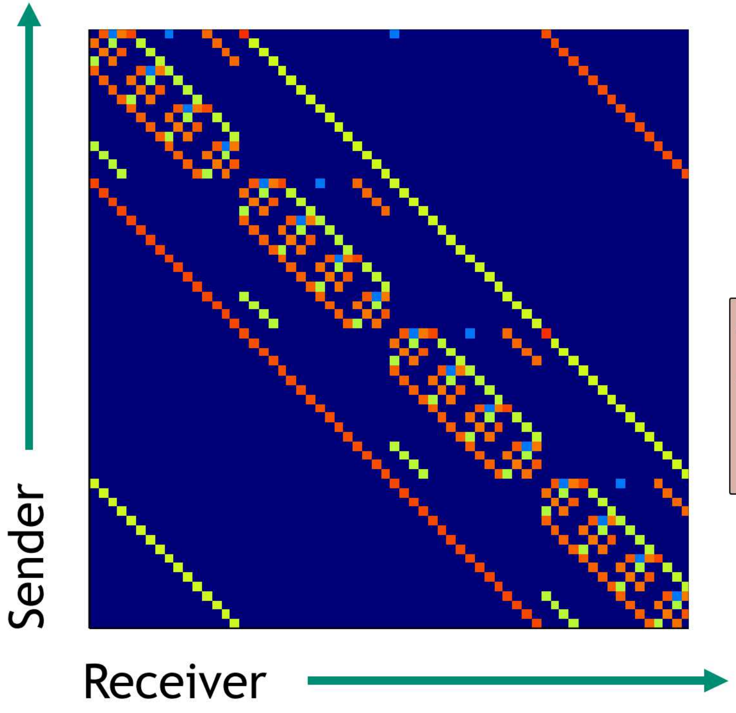
- Description:
  … The FY19 co-design milestone will examine the impact of network interconnects on the performance of ASC applications. The milestone team will work with vendors to analyze DOE workloads and applications to quantify the performance impacts of network options…

- Completion:
  An evaluation of novel interconnect topologies interconnects with performance estimates for ASC applications.

Four Specific Design Issues Addressed Here

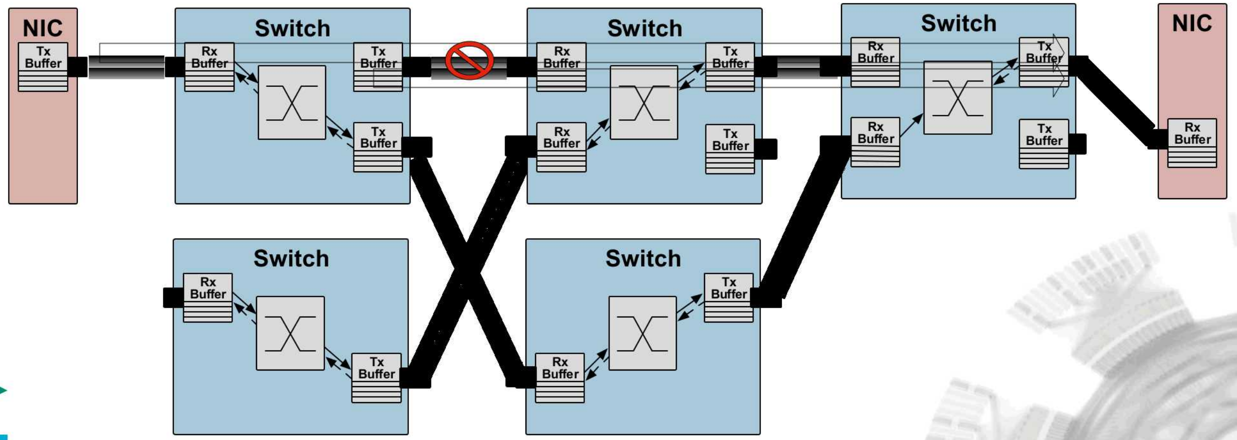
- Dependence on workload
- Dependence on topology/routing
- Scaling question from CTS (1000 nodes) to ATS (16000 nodes)
- *Insight* into performance differences from performance counters

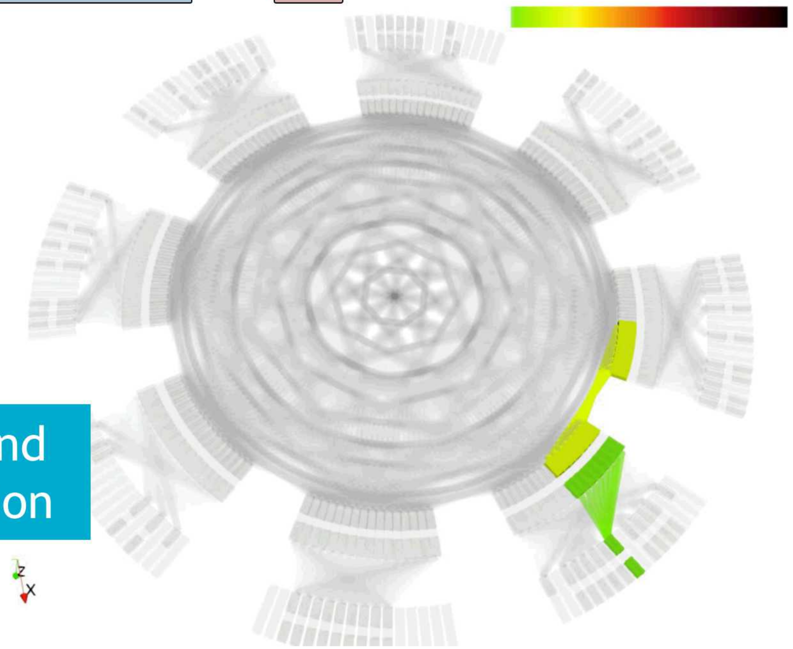
Interconnect design is driven by *geometry*: what topology and routing mechanisms work best for ASC traffic patterns?



ASC applications have characteristic traffic patterns of send/recv pairs

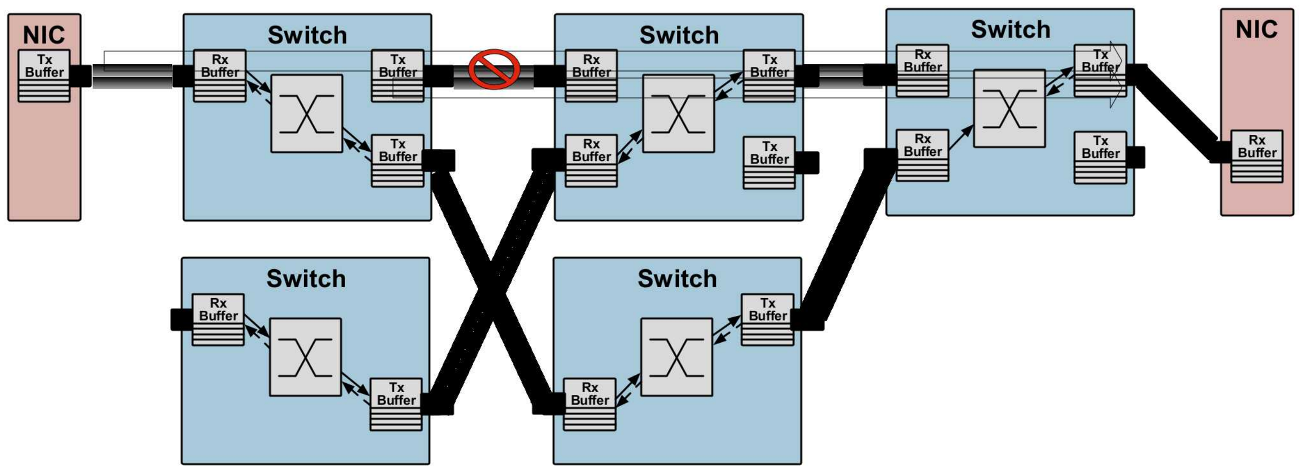
Competing flows (send/recv pairs) will compete for paths in the network



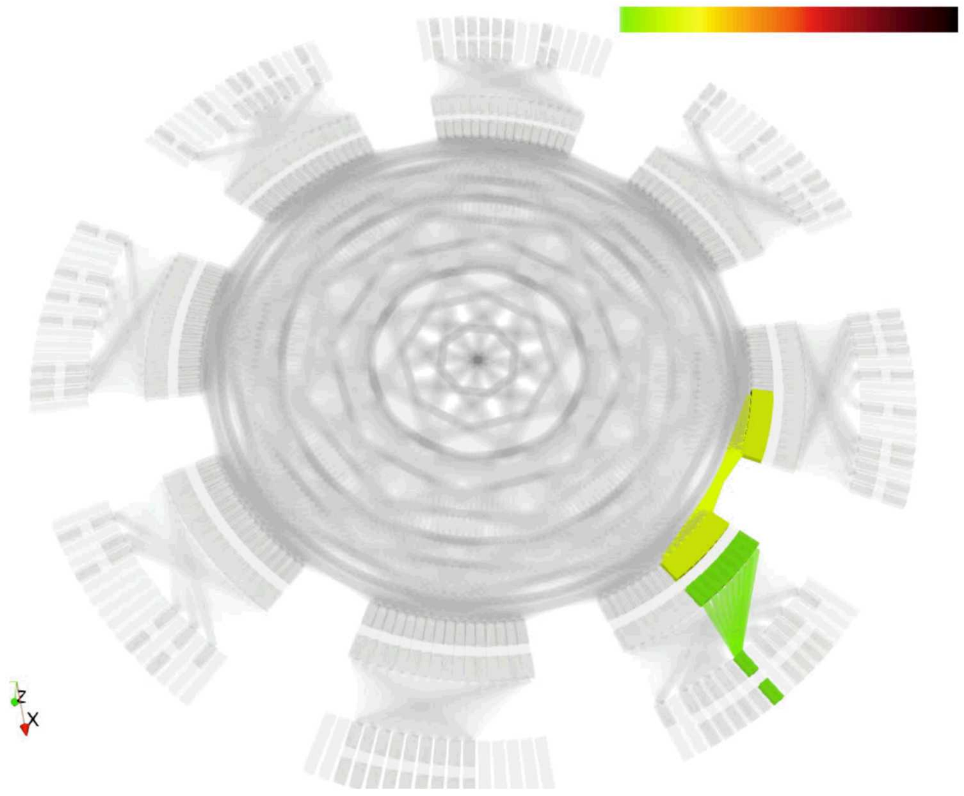


How ports are connected and routed determines contention

Interconnect design is driven by both *local* and *global* geometry

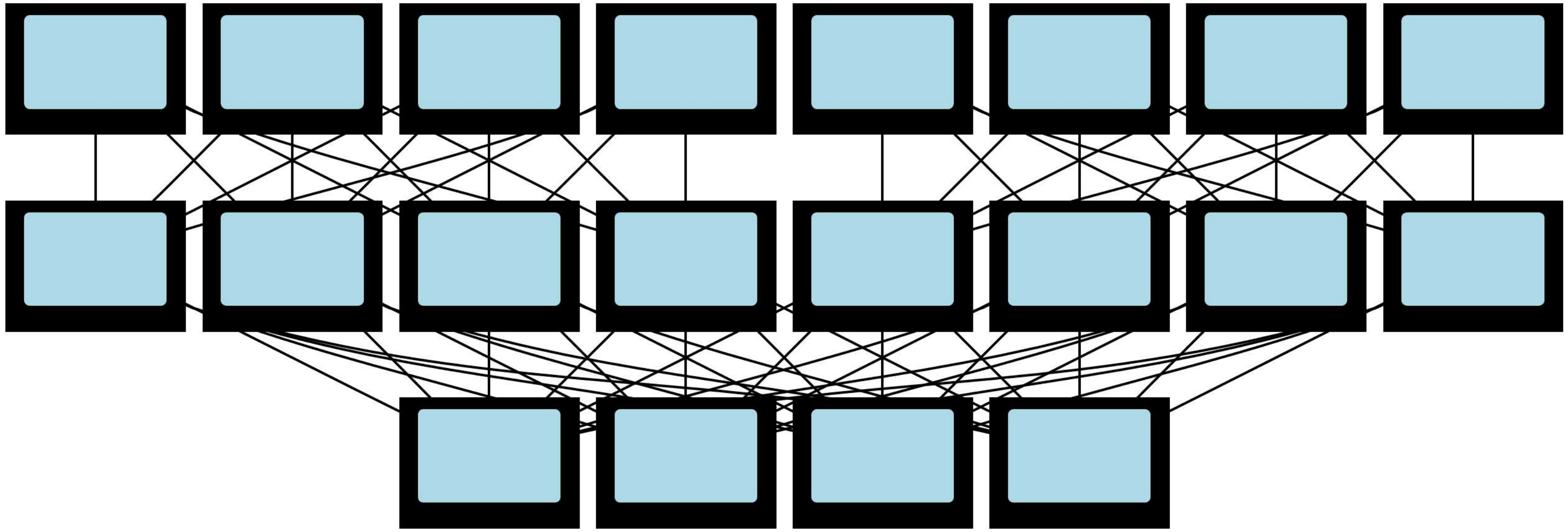


- Each switch has to make routing decisions *locally*
- Each switch has to satisfy *global* constraints
    - Bisection bandwidth limited
    - Virtual channel and deadlock issues
- Routing is better if it can be based on *global* congestion information



- Arrangement of switches and ports determines topology properties
    - Bisection bandwidth
    - Locality
- Choice of *macroscale* global geometry can affect *microscale* routing requirements

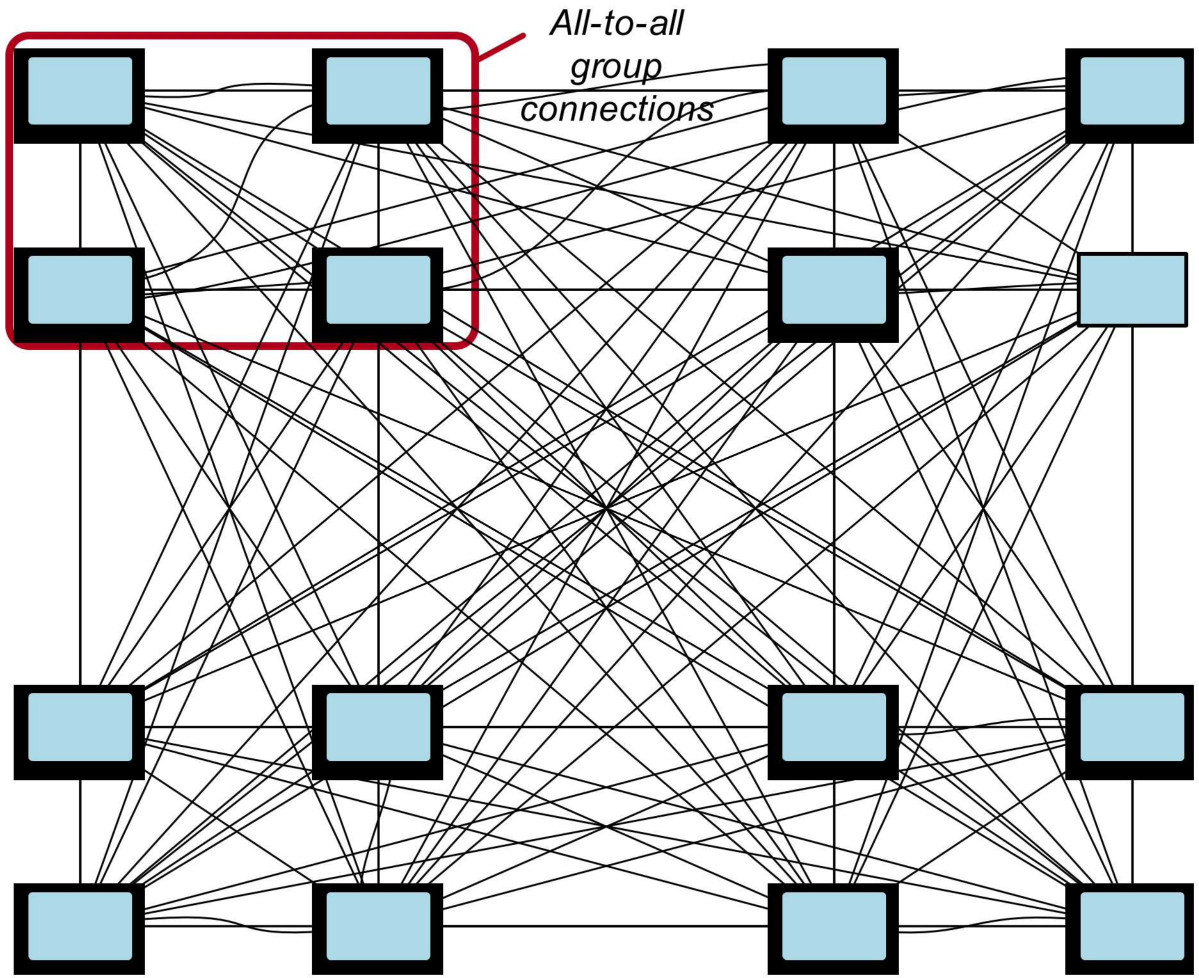
Candidate #1: Fat tree is mature data center topology which with tunable bisection bandwidth and some locality



- Can be full or tapered bisection bandwidth
- Divided into leaf, aggregation, and core switches
- High path diversity for sending from leaf->leaf and agg->agg switches
- Single virtual channel, all traffic flows in “same” direction
- High diameter (max 6 hops)

Diameter = 6 with adaptive routing

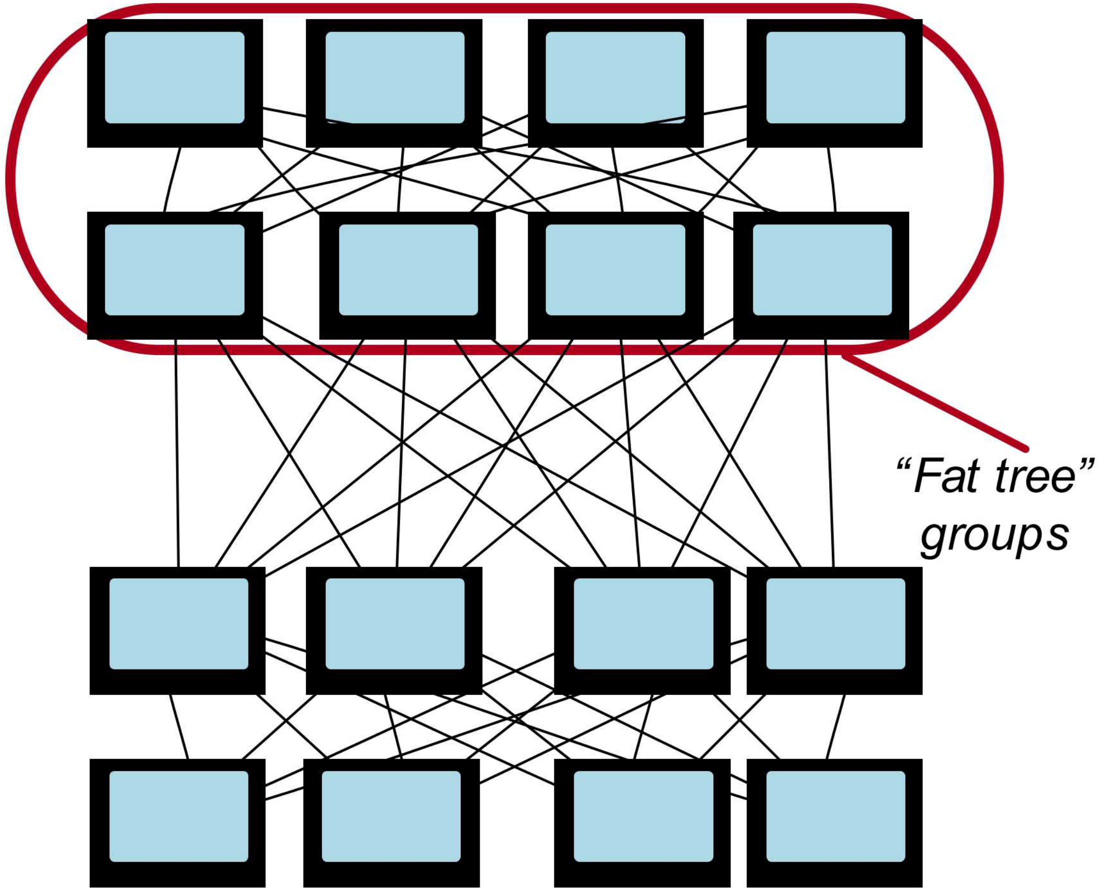
Candidate #2: Dragonfly topology has high path diversity, strong locality, and tunable bisection bandwidth



- Full bisection bandwidth
- Low diameter (max 3 hops)
- All-to-all groups connected with long-reach global links between groups
- High path diversity for adaptive routing
- Many virtual channels required for most sophisticated routing
    - Progressive adaptive (PAR)

Diameter = 5 with adaptive routing

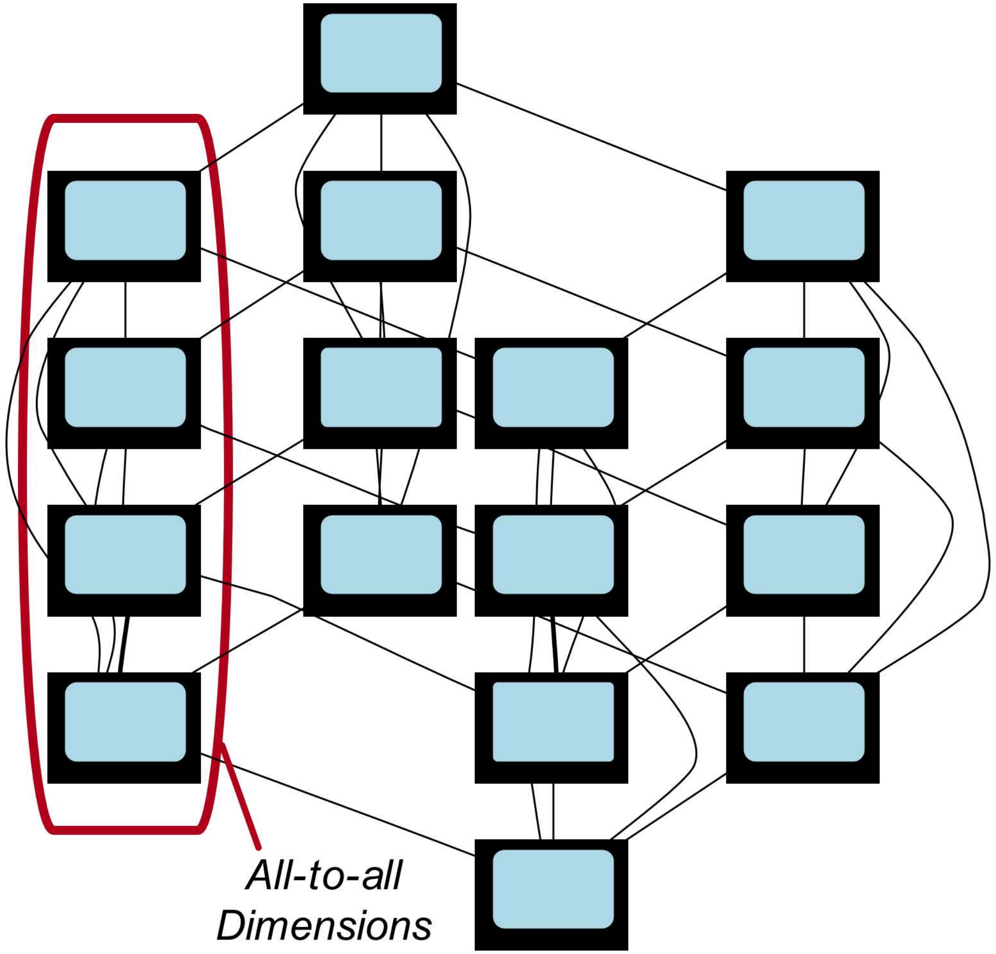
Candidate #3: Dragonfly+ topology sacrifices some locality for simpler routing and larger groups



- Hybrid of fat-tree and dragonfly
- Full bisection bandwidth
- Medium diameter (max 4 hops)
- Fat-tree groups connected with long-reach global links between groups
- High path diversity for adaptive routing
- Fewer virtual channels required for most sophisticated routing (2 needed)
  - Progressive adaptive (PAR)

Diameter = 5 with adaptive routing

Candidate #4: HyperX provides path diversity, locality reduces bisection "pressure"

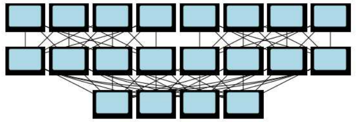


- Full bisection bandwidth
- Low diameter (max 3 hops)
- High path diversity for adaptive routing
- Many virtual channels required for most sophisticated routing (6 needed)
  - Variable dimension progressive adaptive (PAR)
- Locality can reduce bisection pressure with uniform random traffic

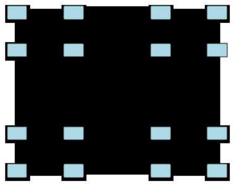
Diameter = 6 with adaptive routing

Broad survey over the interconnect design space covers different workloads and range of scales

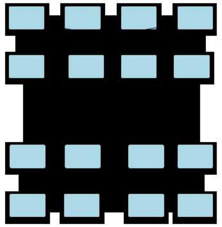
4 Topologies



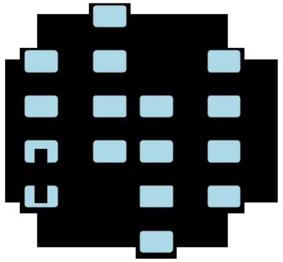
Fat tree



Dragonfly



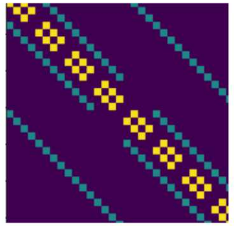
Dragonfly+



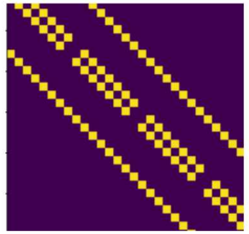
HyperX

3 Workloads x 2 Placements

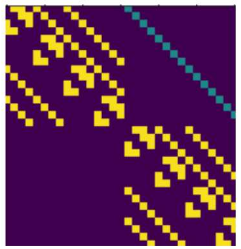
Random + Linear Placement



Halo



Sweep



Subcomm (FFT)

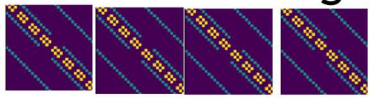
3 Scales

- CTS: 1K nodes e.g. Serrano
- "Small" ATS: 4K nodes e.g. Sierra
- "Big" ATS: 16K nodes e.g. Trinity

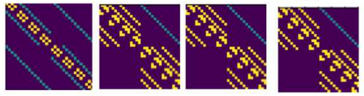
4 "Environments"

¼, No Background

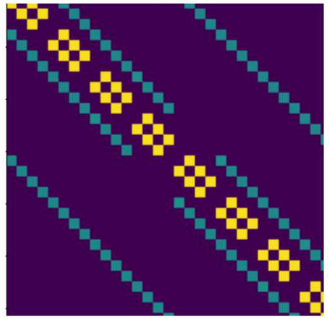
¼ + Halo Background



¼ + FFT Background



Full System

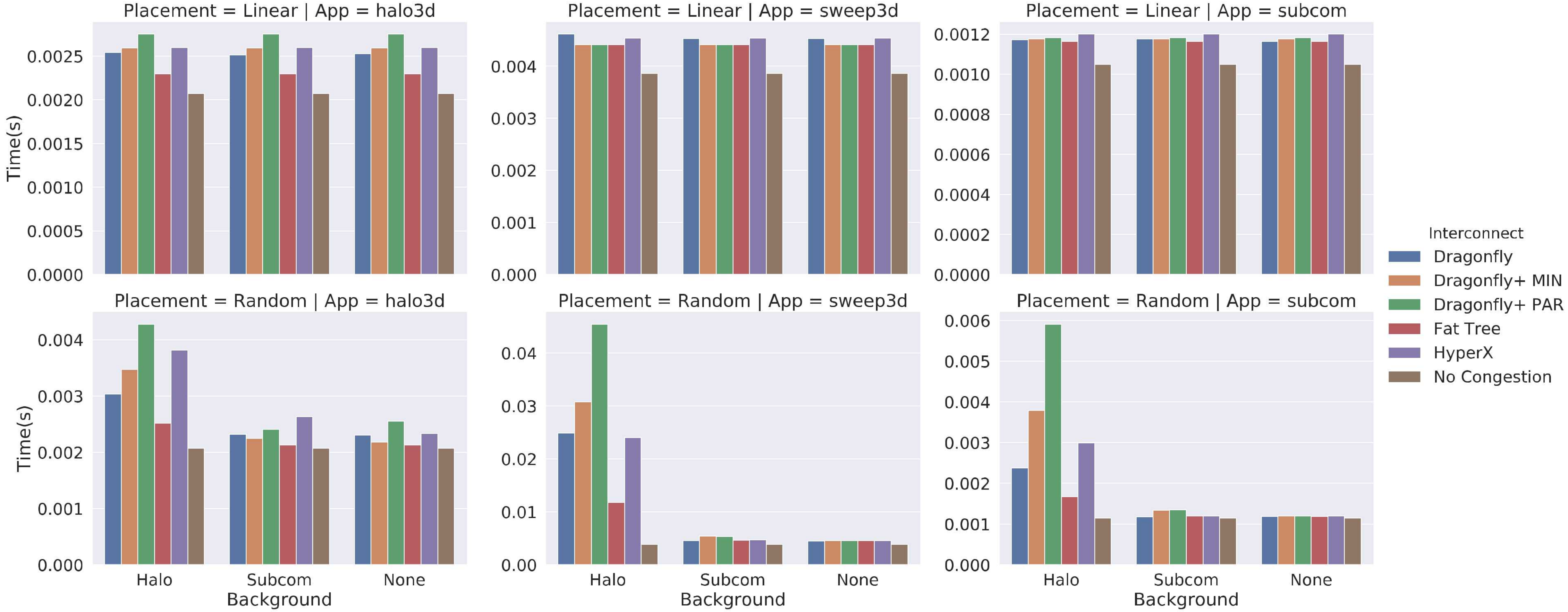


4 "MPI Modes"

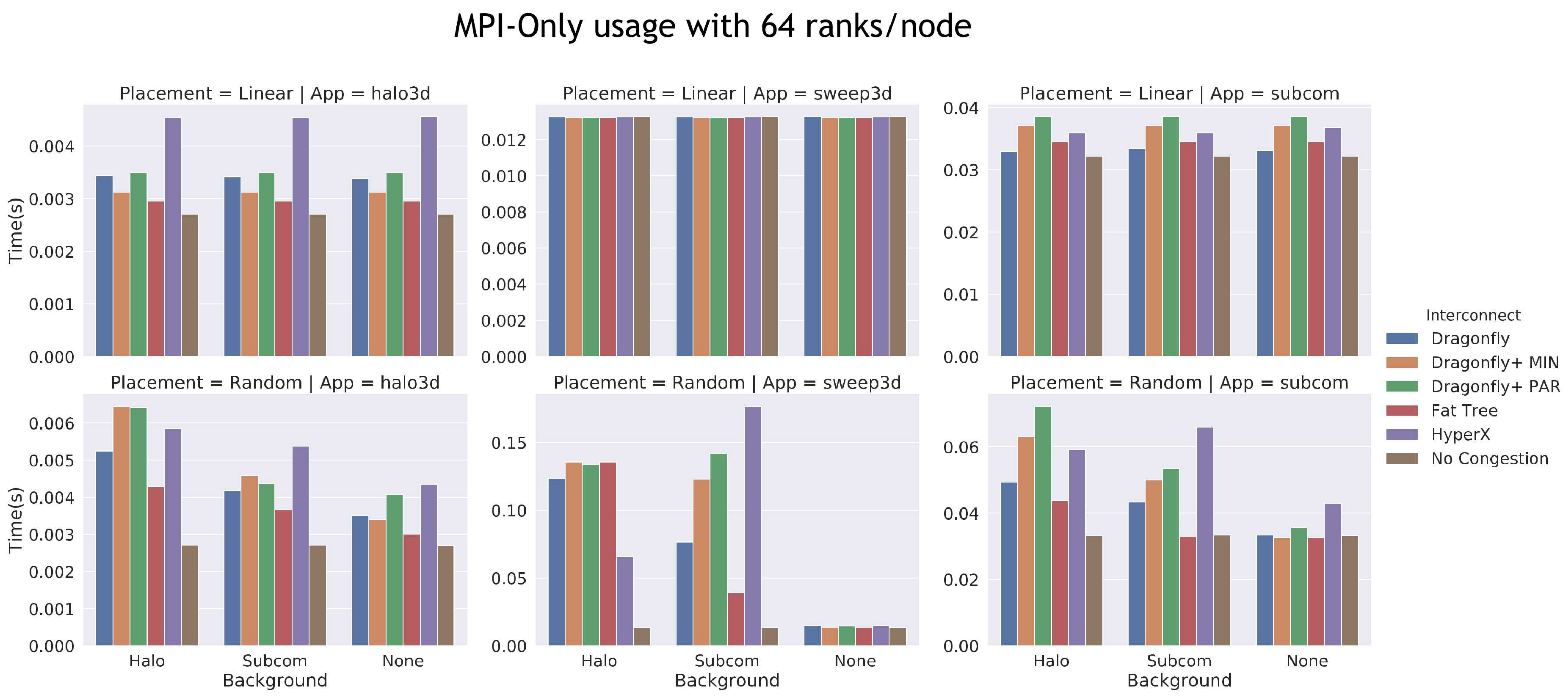
- MPI + OpenMP: 1 rank/node
- Mixed 4 ranks/node
- Mixed 16 ranks/node
- MPI-Only 64 ranks/node

CTS scale (1K nodes) shows room for improvement with all topologies and fat tree consistently the best

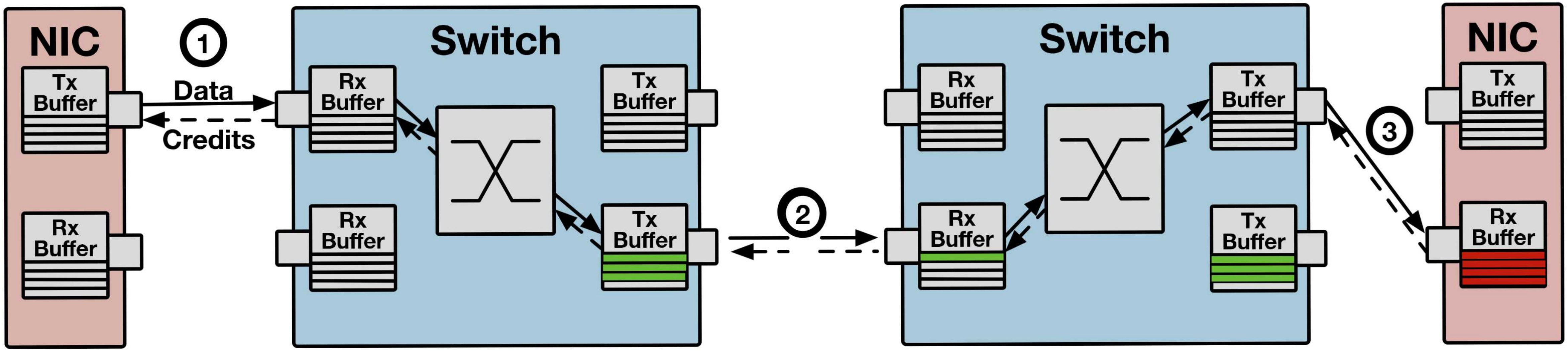
MPI+OpenMP usage with 1 rank/node



CTS scale (1K nodes) story mostly consistent when used with MPI-only communication patterns



Inefficient use of the network primarily manifests in three performance counters on network ports



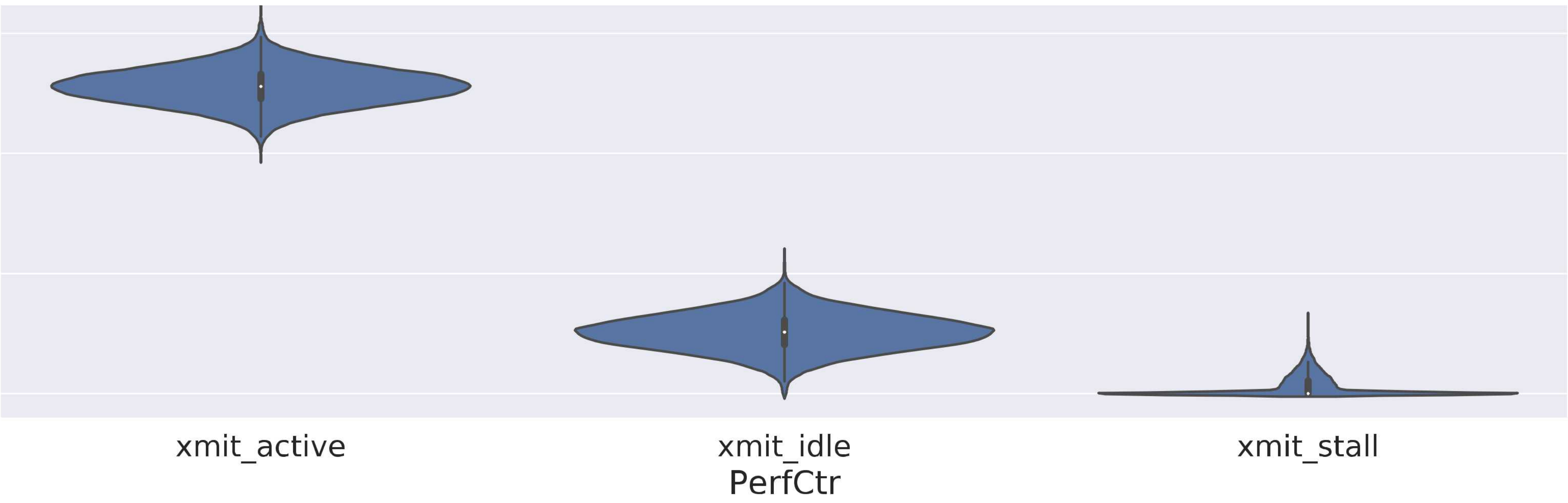
1. Idle: No packet to send from Tx buffer
2. Active: Packet actively sending from Tx buffer to open Rx buffer
3. Stalled: Packets available in Tx buffer, but no open Rx buffer

Ideal performance counter use case (and brief introduction to violin plots showing density distributions)

Ports mostly active, activity limited to ”minimal” paths

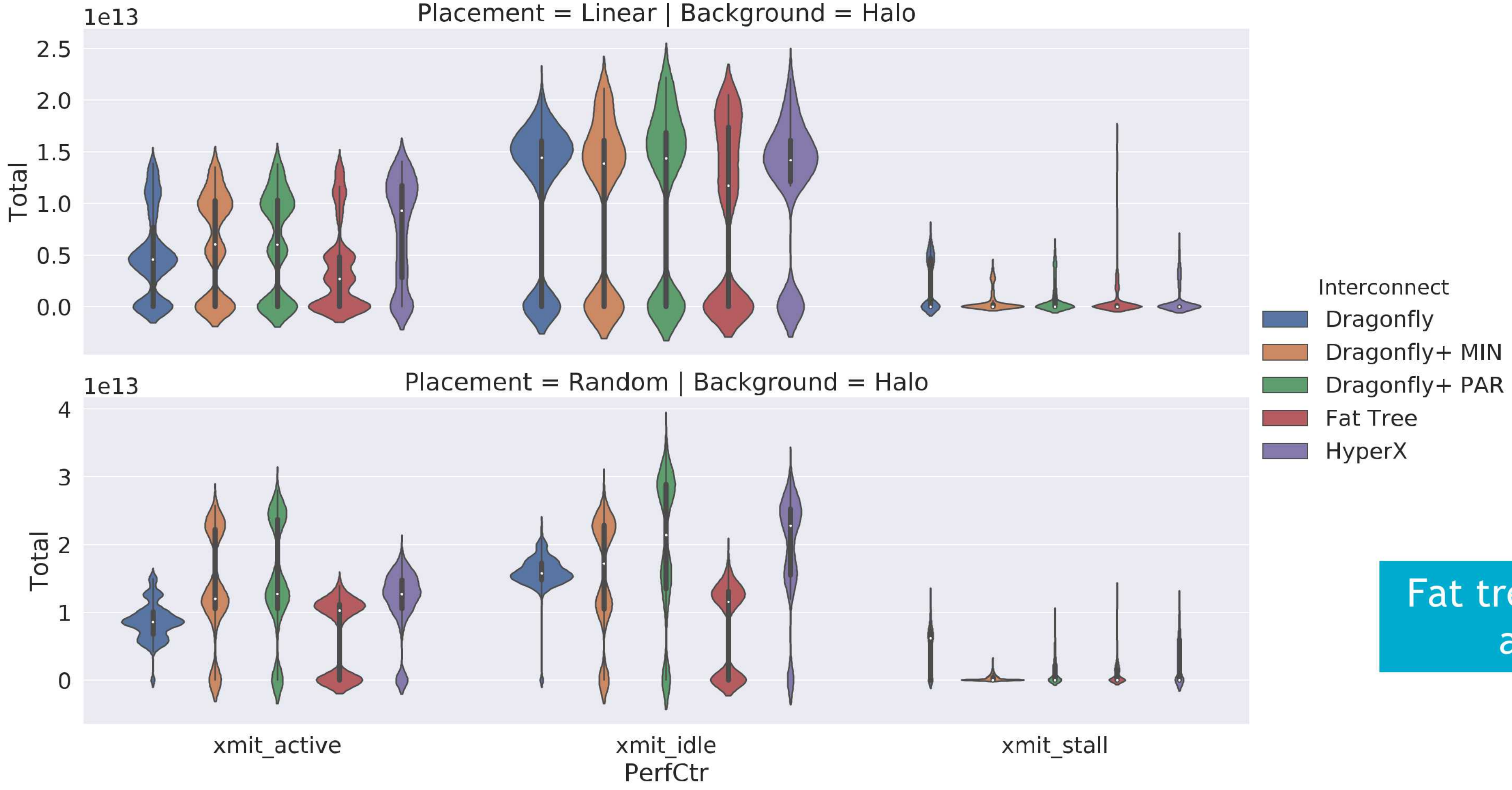
Very few ports idle, all bandwidth in network doing useful work

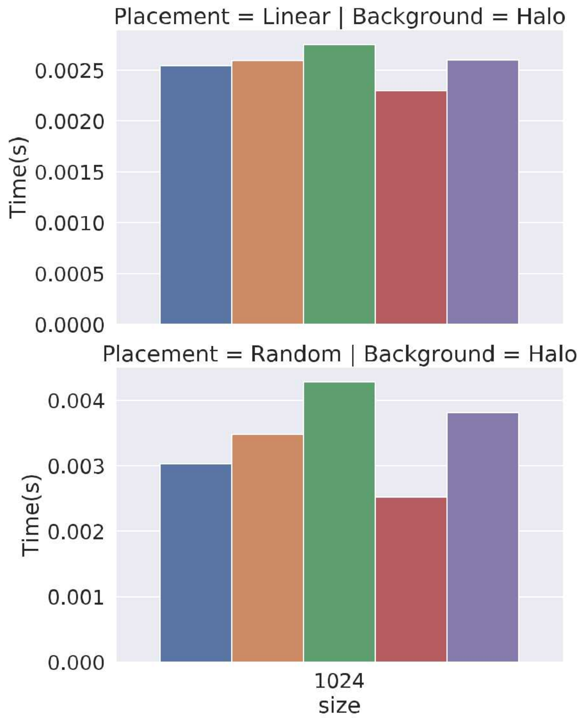
Very few ports stalling due to congestion



Performance counters provide a deeper insight into origins of the performance differences: active, idle, stall

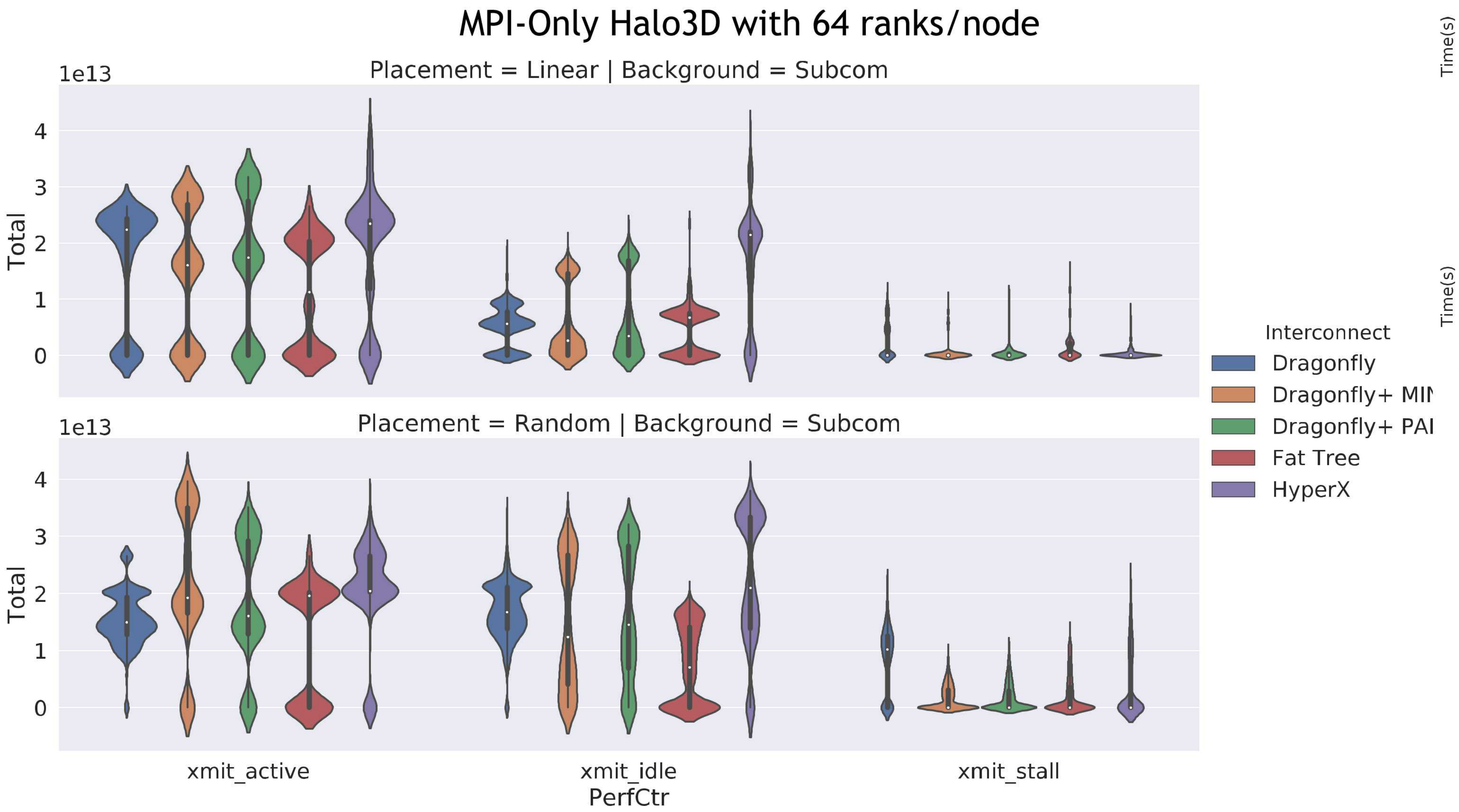
MPI+OpenMP Halo3D with 1 rank/node

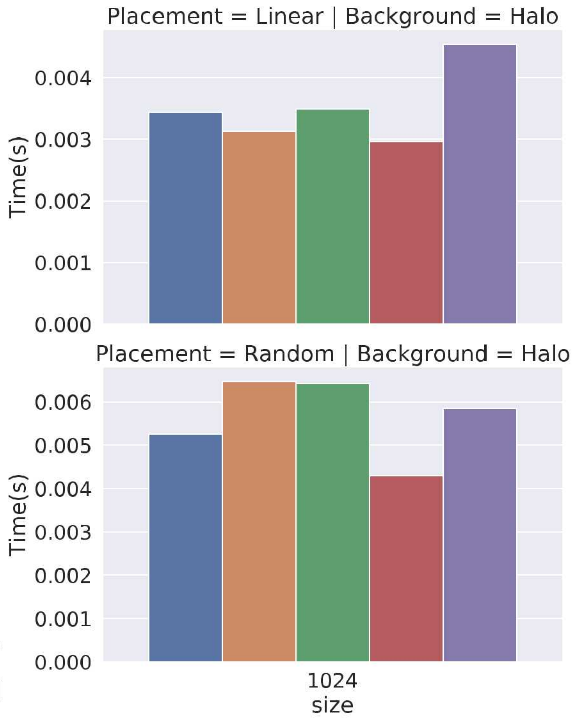




Fat tree has less activity and less idling

MPI-only usage mode tells similar story as MPI+OpenMP traffic injection





CTS-scale (1K nodes) Fat Tree is reliable choice and Dragonfly looks like a strong alternative

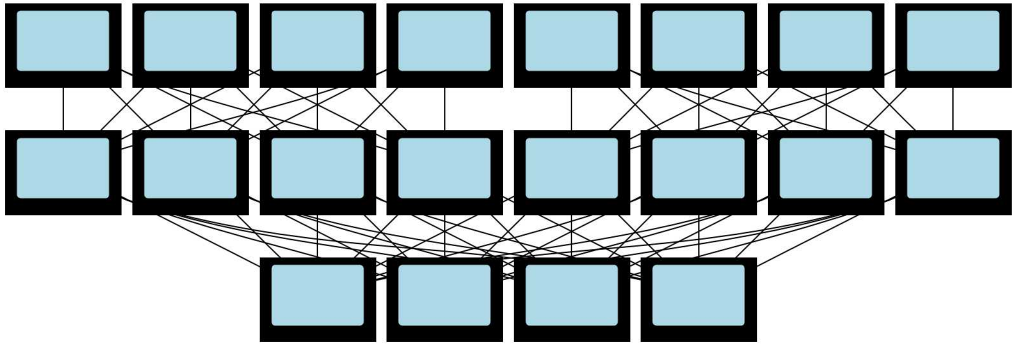


Figure: 4 paths to neighbor, 16 bisection paths
Actual: 32 paths to neighbor, 1K bisection paths

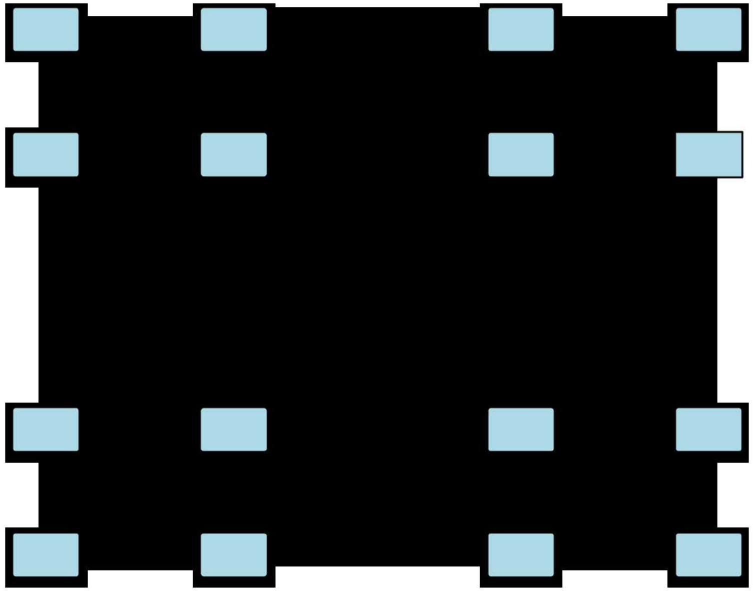
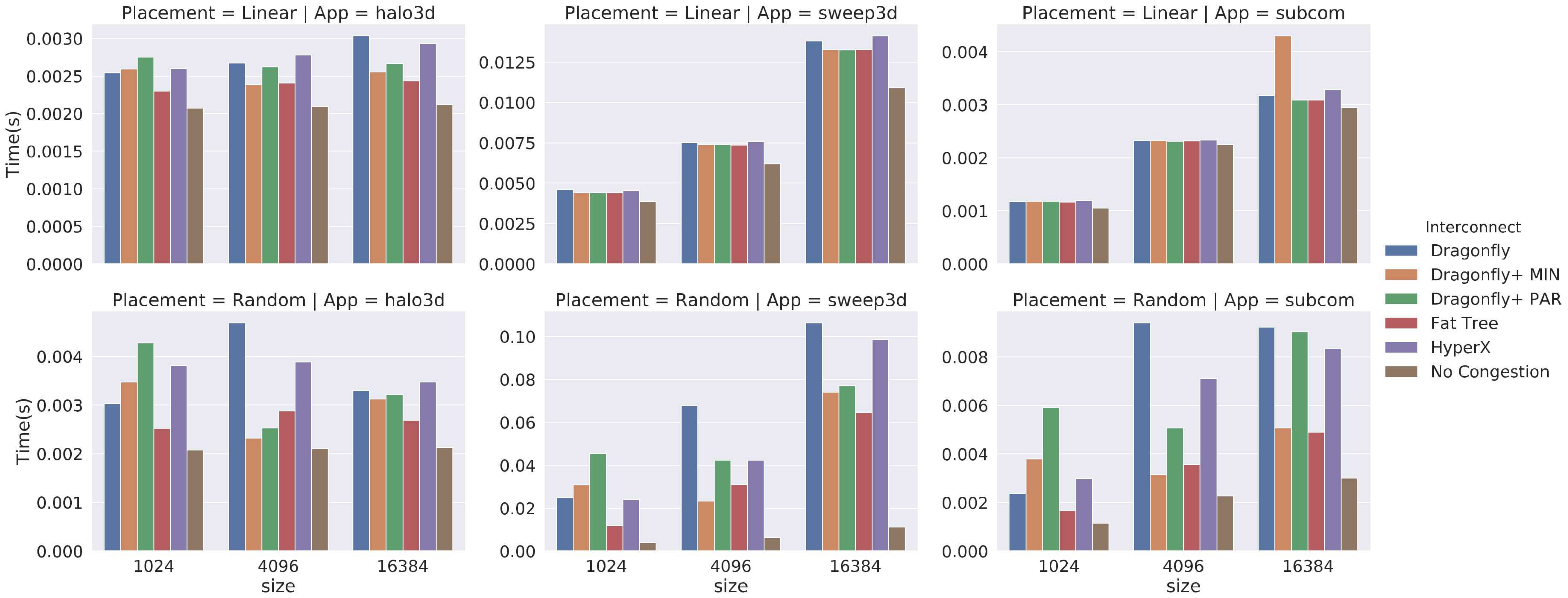


Figure: Diameter-2 dragonfly (all-to-all group connectivity)
Actual: Diameter-3 dragonfly at 16K nodes (need intra-group hops to global gateways)

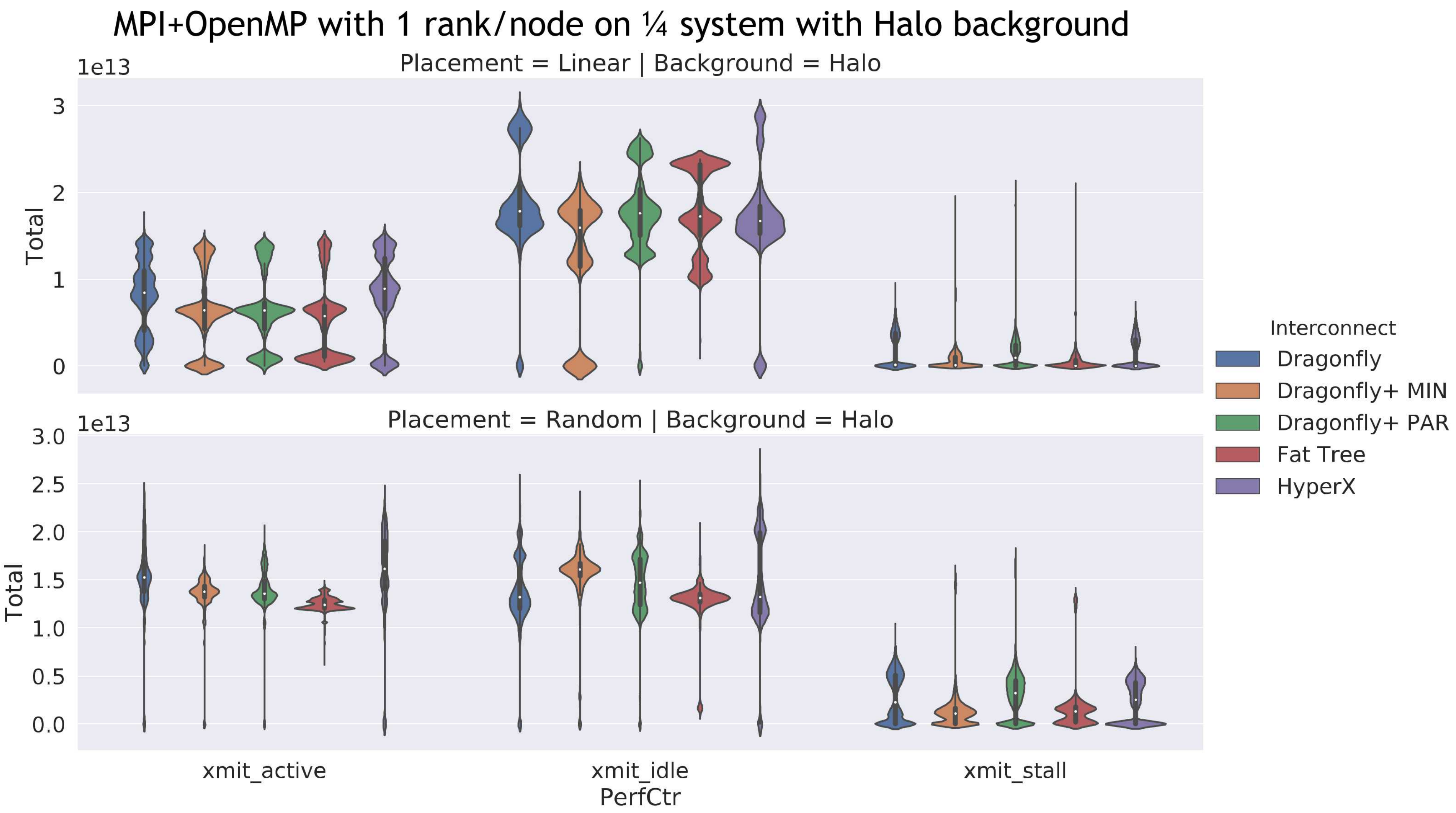
Scaling to ATS (4K-16K nodes) system sizes dramatically changes performance behavior

MPI+OpenMP with 1 rank/node on ¼ system with Halo background



Dragonfly goes from a diameter 2 to a diameter 3 network, complicating adaptive routing

In Dragonfly and HyperX ports are working hard (random placement) or hardly working (linear) for 16K nodes



The simulation methodology includes leading HPC network topologies and communication patterns

Three network topologies
- Dragonfly (Cray)
- Fat-tree (3-level)
- HyperX
- 64-port routers
- Link speed 100 Gbps

Four communication patterns
- Communication **only**
- Halo3d-26 (27-point stencil)
- Sweep3d ("pencil")
- Subcom2d-coll (collectives/Qbox)
- Subcom3d-a2a (all-to-all)

- 32,768 MPI ranks*
- Adaptive routing (best)
- Using simulator TraceR-CODES
  - TraceR: MPI trace traffic generation
  - CODES: network modeling

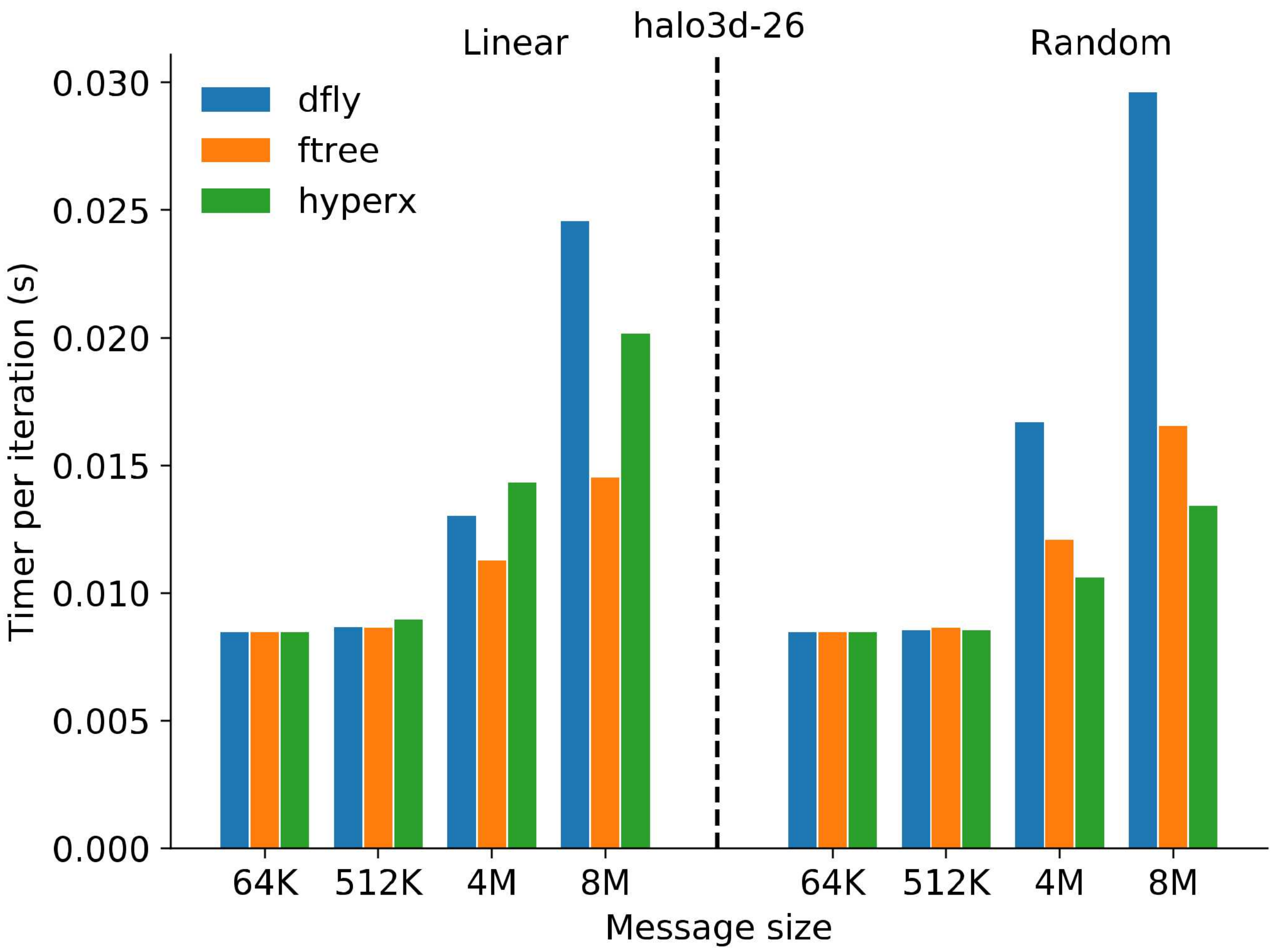
*ALL RANKS COMMUNICATE THROUGH NETWORK (WORST-CASE ANALYSIS)

Summarizing the results on halo3d-26
Fat-tree performs best for large message sizes otherwise topologies perform similarly

Topologies and time-per-iteration on par for smaller message sizes of 64K, 512K

Fat-tree scales better for larger message sizes 4M, 8M

- Followed by HyperX, Dragonfly

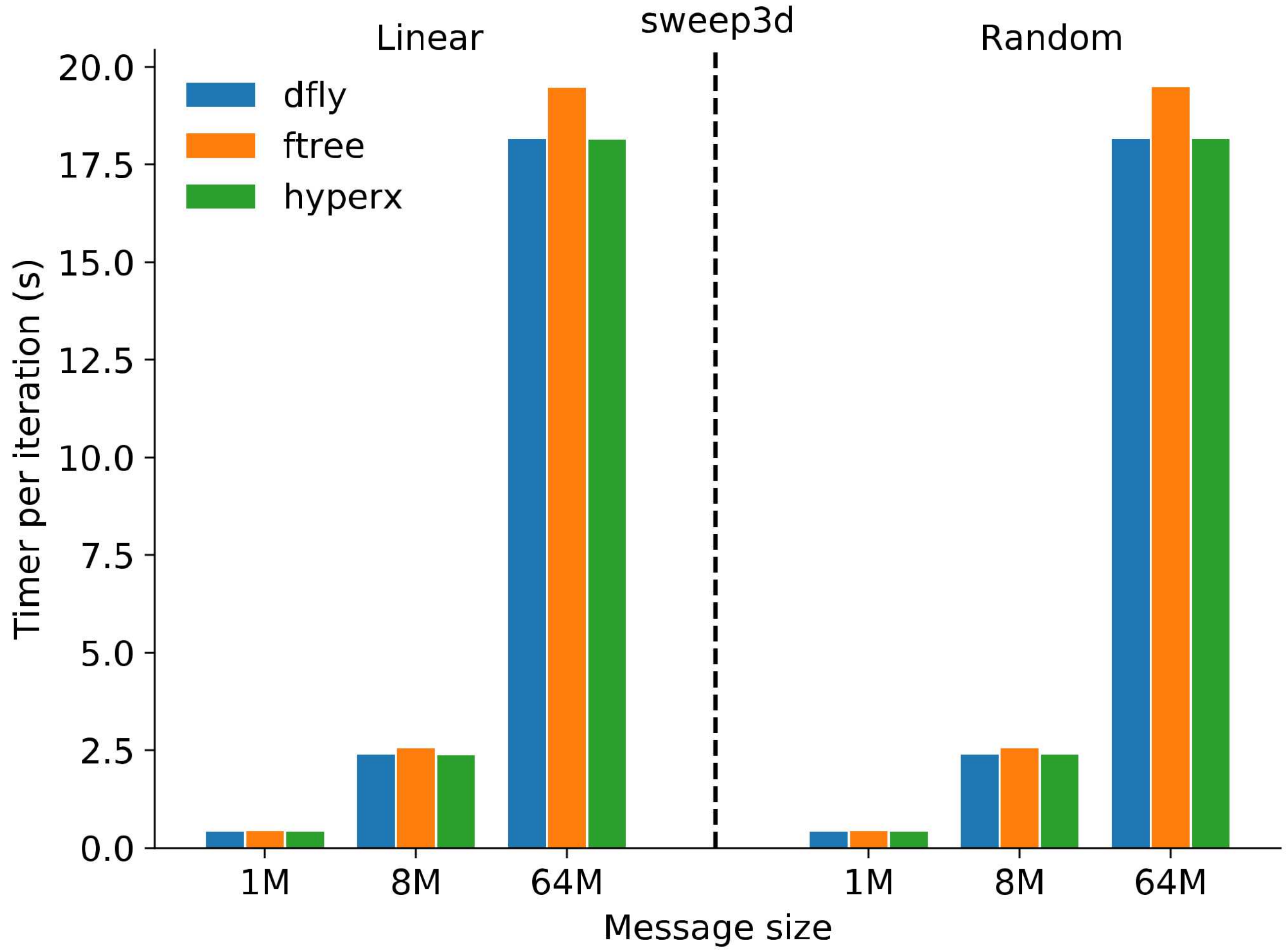


Summarizing the results on sweep3d All topologies perform similarly

Time-per-iteration scales linearly with message size

Topologies are on par

Insensitive to linear or random mapping



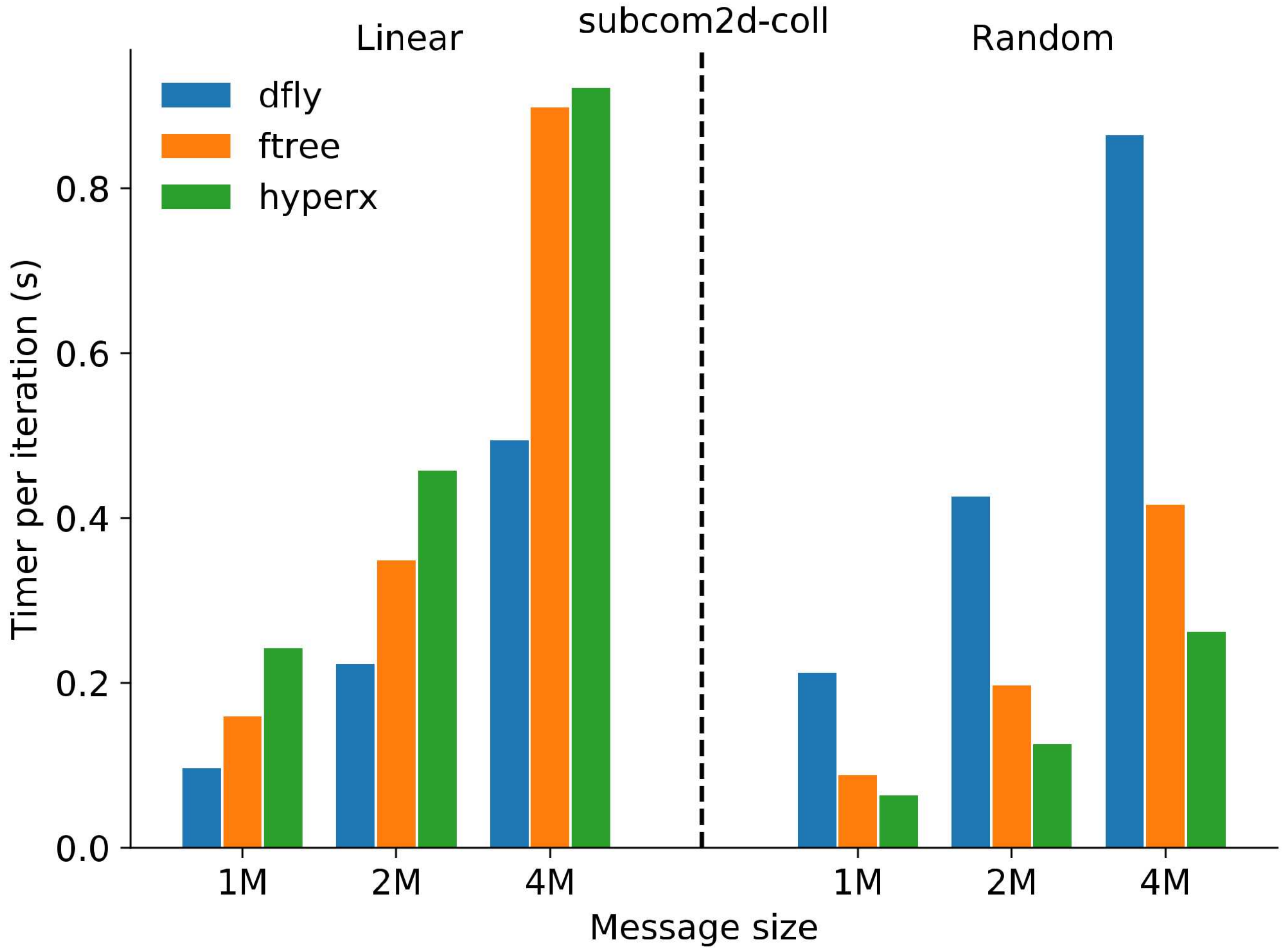
Summarizing the results on subcom2d-coll Dragonfly is best for linear mapping while HyperX is best for random (and Fat-tree is in-between)

Dragonfly scales best for linear mapping

- Followed by Fat-tree, HyperX

Time-per-iteration increases super-linearly

Random mapping reverses the order of performance of topologies

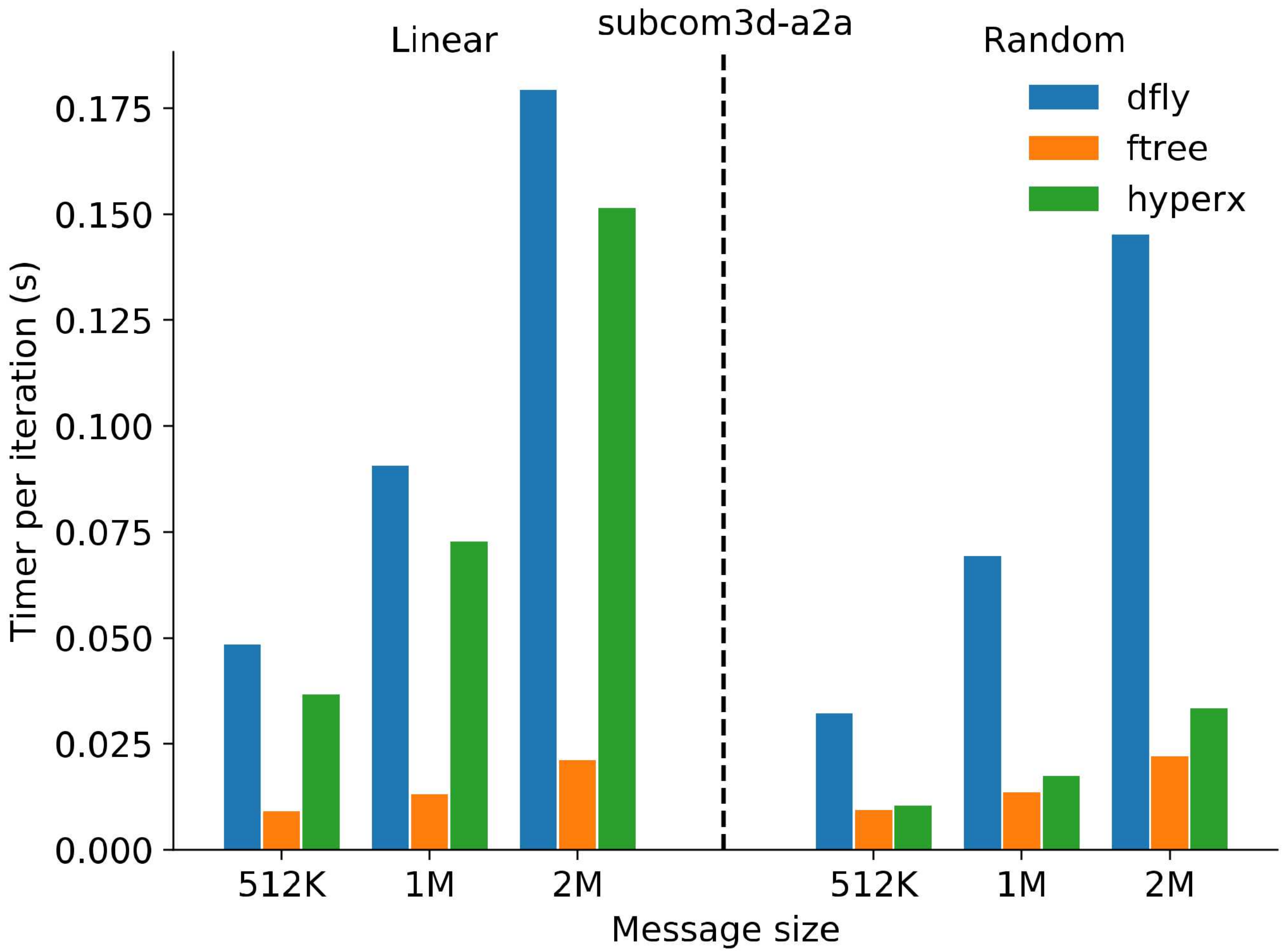


Summarizing the results on subcom3d-a2a Fat-tree performs best for all message sizes and mappings

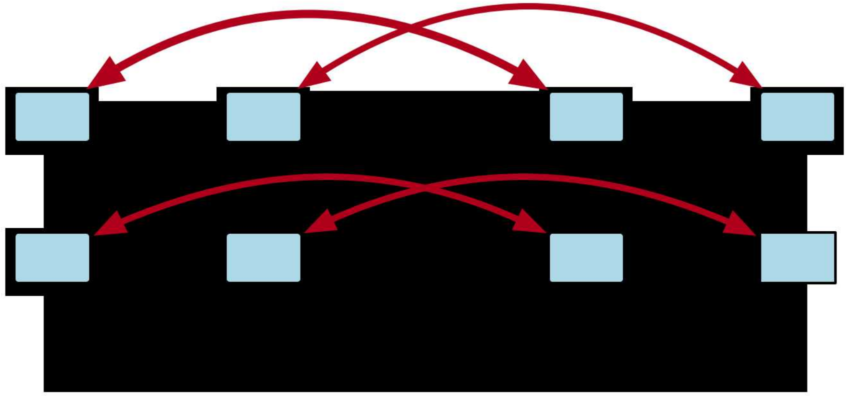
Fat-tree scales best for all-to-all communication
- Followed by HyperX, Dragonfly

Time-per-iteration scales more than linearly

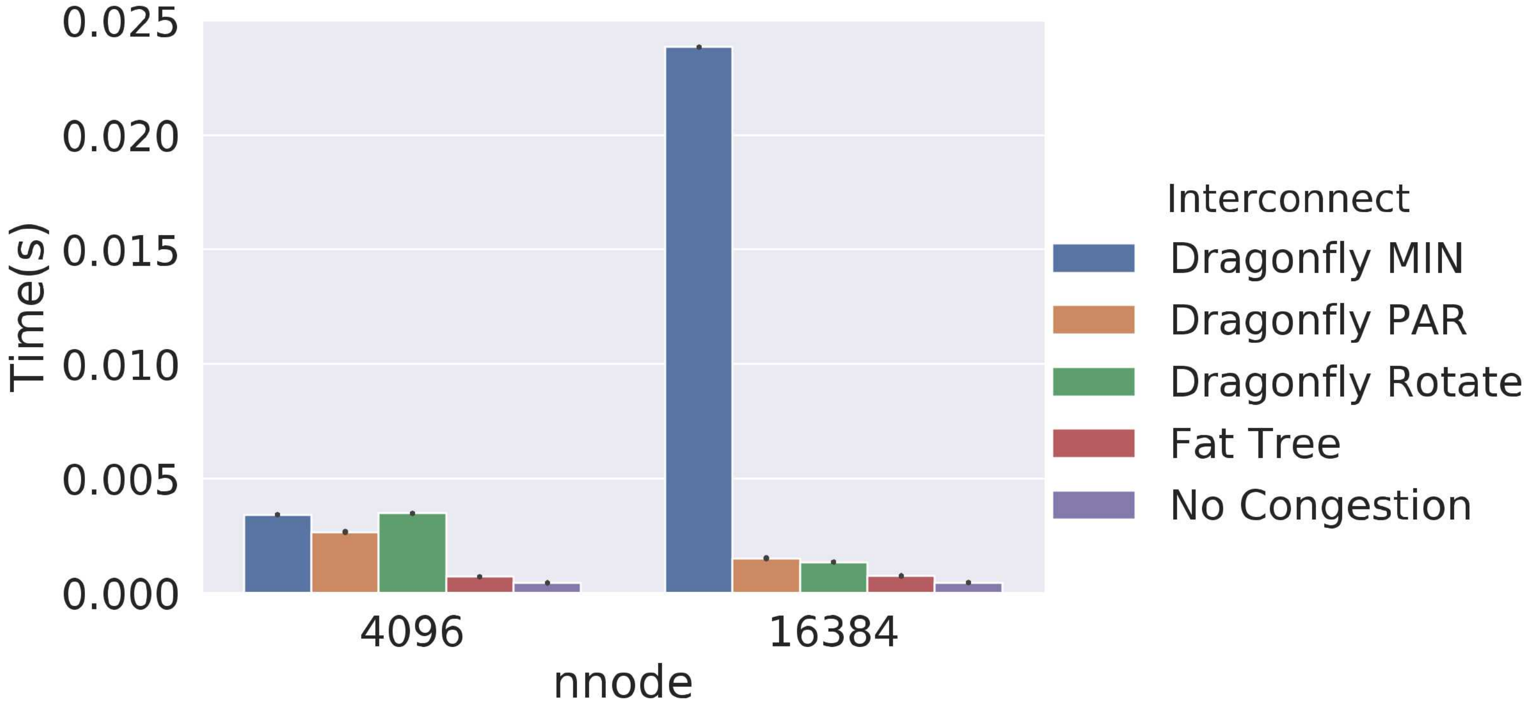
Under random mapping HyperX performs close to Fat-tree



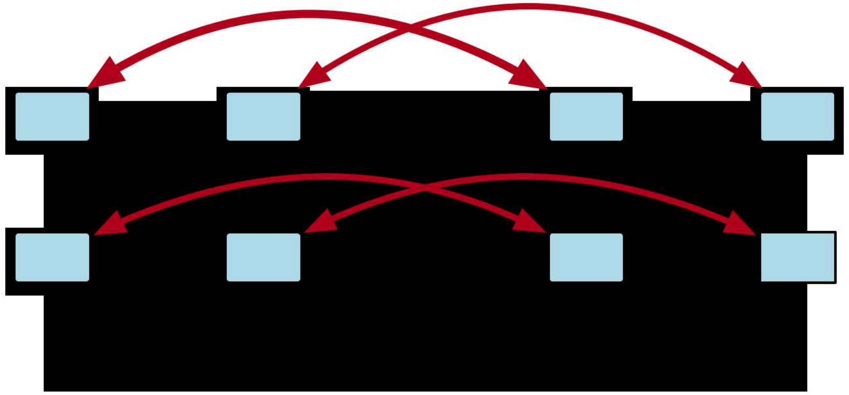
Worst-case traffic pedagogical example clearly shows challenges in dragonfly implementation



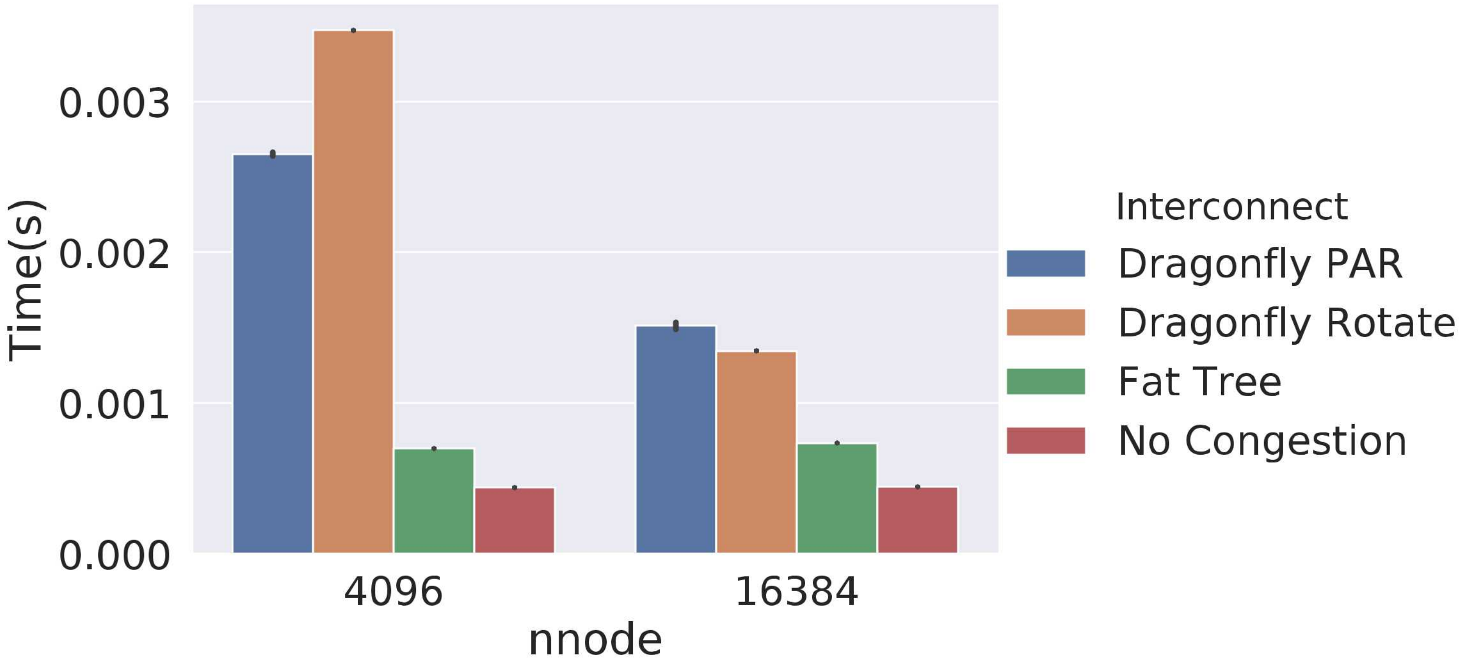
Worst-case traffic sends entirely from one group to another across ”scarce” global links



Worst-case traffic pedagogical example clearly shows challenges in dragonfly implementation



Worst-case traffic sends entirely from one group to another across "scarce" global links



"Rotate" emulates fat-tree oblivious routing on a Dragonfly

Performance counters shows dragonfly adaptive routing is failing to find uncongested paths

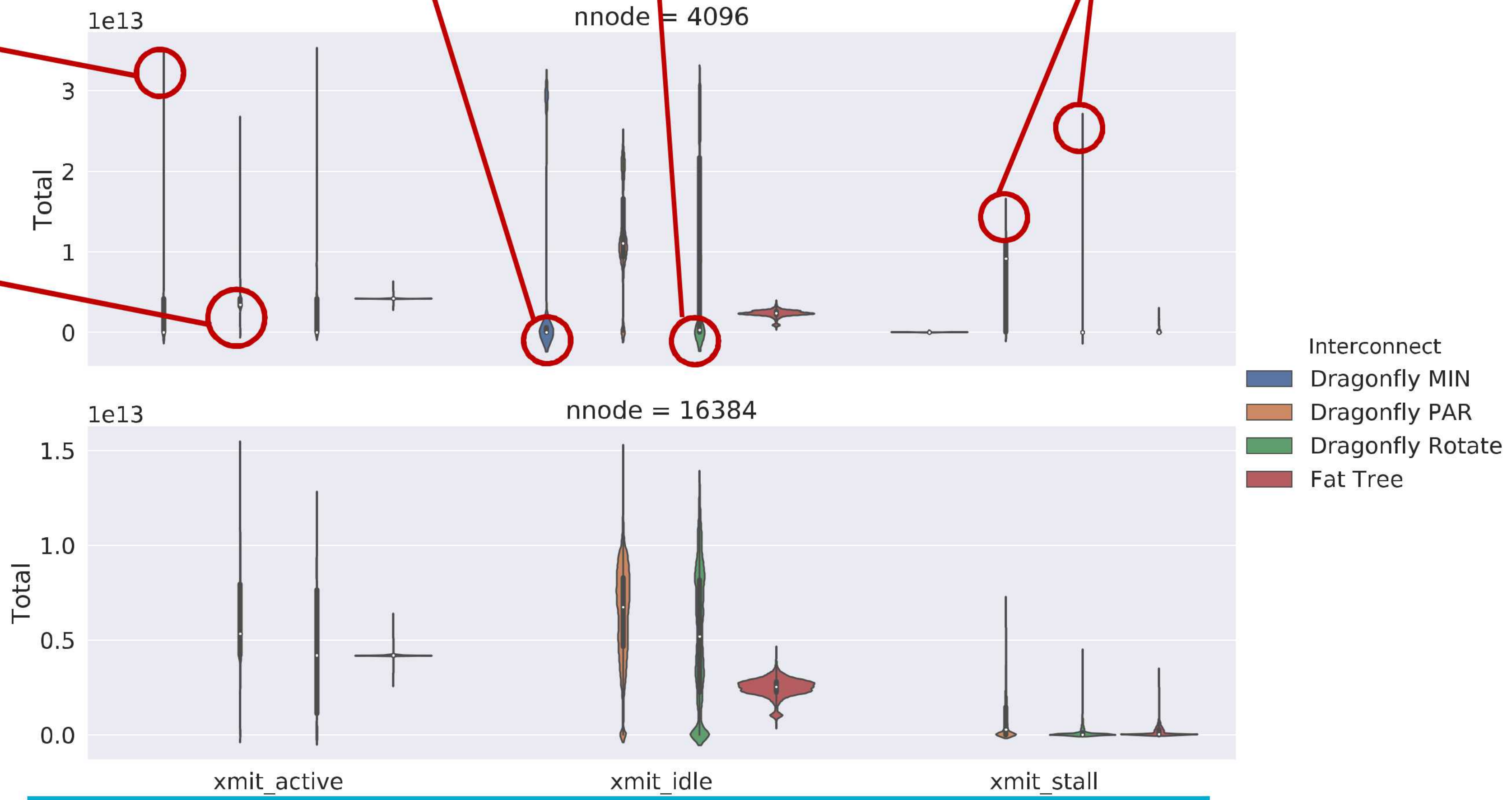
Minimal has "long tail" of active ports

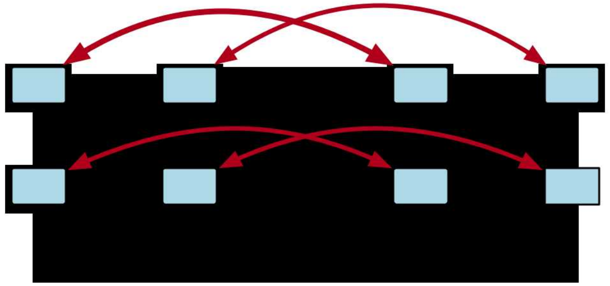
Adaptive has more total activity, but clustered lower

Minimal has many idle ports and a few never idle

Rotate strategy reduces idleness relative to adaptive

Adaptive "misroutes" into contention creating more stalls

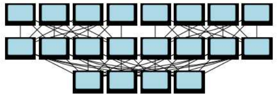
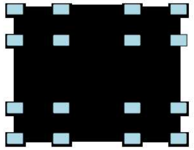
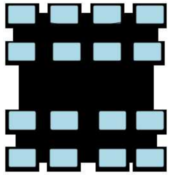
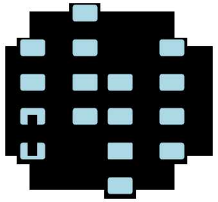




Worst-case traffic sends entirely from one group to another across "scarce" global links

At 16K diameter goes from 2->3,
PAR breaks down trying to find unused global bandwidth

Other features of the network could dramatically change the landscape of performance for each interconnect design

	Topology	Congestion Control	QoS	Routing Metrics
		More efficient on low diameter AND minimal routing	More efficient on simpler geometry with fewer VCs	
	Fat tree	Diameter=5, minimal routing can be efficient	Single VC for deadlock	
	Dragonfly	Diameter=3, multipath routing complicates detection	>= 3 VCs for deadlock	More sophisticated schemes could avoid so many idle ports
	Dragonfly+	Diameter=4, minimal routing can be efficient	Single VC for minimal, 2 VCs for adaptive	
	HyperX	Diameter=3, multipath routing complicates detection	1-3 VCs depending on implementation	More sophisticated schemes could avoid so many idle ports

Very Bayesian conclusions chosen very carefully

- If conditions match those used in simulation:
  - Fat tree is both simplest and most robust, despite some extra cost (might be mitigated with tapering)
- If using commodity IB switches with limited minimal routing AND QoS is important
  - Fat tree is clear winner
- If vendor adaptive routing are effective and workloads are allocated close to “linear”:
  - Dragonfly becomes appealing option, particularly for CTS scale
- If all areas for improvement are combined:
  - HyperX has many desirable for properties, most interesting target for optimization
- If packaging issues are not a problem:
  - Dragonfly+ is an interesting middle-of-the-road option

Acknowledgments

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.