# Application performance impact on trimming of a full fat tree InfiniBand fabric

*Siddhartha S. Ghosh*[†1], Davide DelVento[1], Rory Kelly[1], Irfan Elahi[1], Nathan Rini[1], Storm Knight[1], Benjamin Matthews[1], Thomas Engel[1], Ben Jamroz[1] and Shawn Strande[2]

[1]Computational Information Systems Laboratory,
National Center for Atmospheric Research
Boulder, CO, USA
†sghosh@ucar.edu
[2]San Diego Supercomputer Center
San Diego, CA, USA

# HPC clusters for Technical computing

- Multi-processor nodes (typically Xeon)
- Infiniband interconnect in
- Fat-tree topology

Fat-tree topology provides non-blocking communication between all pairs, but

It is expensive for large clusters (cluster of size ~5K nodes it could be ~25% of the cost of whole cluster)

Can we optimize the Fat-tree topology (by trimming) to maximize performance/$ ?

Just going from full-fat-tree to 2:1 we connect same number of equipments with

- 25% less number of Top Of the Rack switches
- 50% less number of core switches

# Outline

- Yellowstone Supercomputer, particularly the fabric
- NCAR Application profile
- Study of IB traffic during heavy IB loads
- Trimming study
- Concluding remarks

# Yellowstone Supercomputer

- IBM IDataPlex cluster
- 4536 dual socket E5-2670 (SandyBridge) nodes (16 cores/node)
- Hyper Thread enabled
- Single rail FDR in Full-fat-tree topology
- 3 – stage fabric
- 1st stage, Top Of the Rack (TOR) Mellanox SX6036 switches with compute nodes on one end and Leafs of SX6536 at the other
- 2nd stage, Leafs connect (TOR) on one end and Spines of Mellanox SX6536 core switches
- Runs IB routing engine (REC) with PQFT routing for the compute part of the fabric

# Building blocks of Yellowstone fabric

## Top Of the Rack (TOR) switch

- 36 port Mellanox SX6036
- Copper cable to nodes
- Fibre-Optics to core switches

## Core Switch

- 648 port Mellanox SX6536
- 29 / 36 leafs populated
- 29 x 18 = 522 ports

# 3 stages of Yellowstone Fat-tree fabric

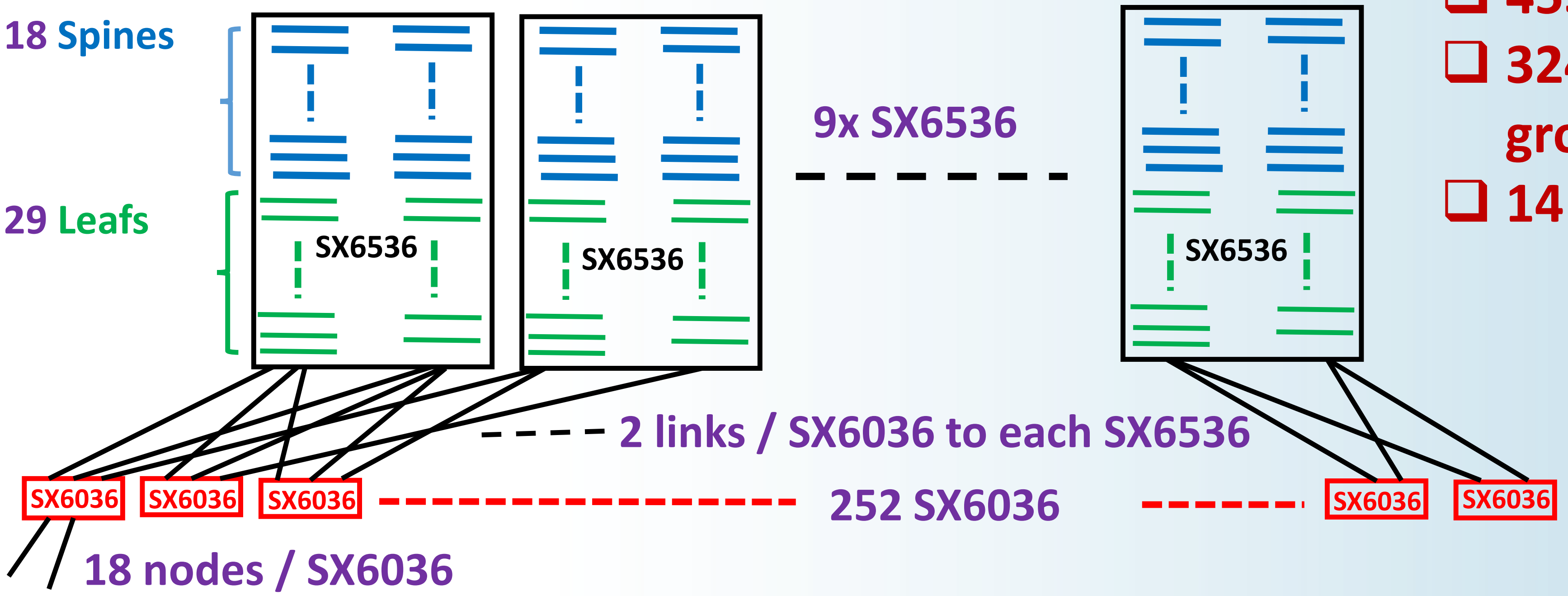
- Stage 1 (TOR) switches (SX6036) nodes are connected on these
- Stage 2 (Leafs) of core switches SX6536 TORs are connected on these devices
- Stage 3 (Spines) of core switches SX6536 Leafs are connected here

**Cabling from TOR to leafs are**

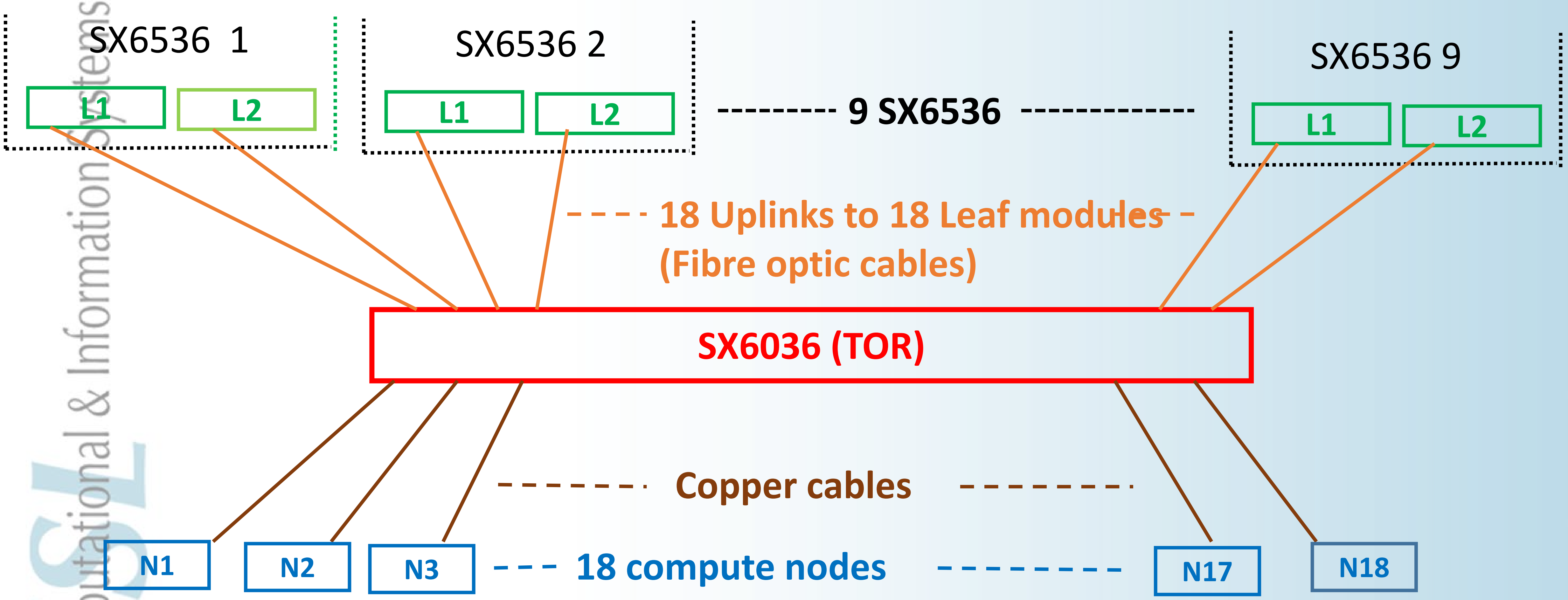
- Quasi Fat-Tree (QFT)  TOR to Leaf cables are spread out to different leafs as against
- True Fat-tree (TOR to leaf cables go to same leaf in a core switch)

# Yellowstone Fabric (schematic)

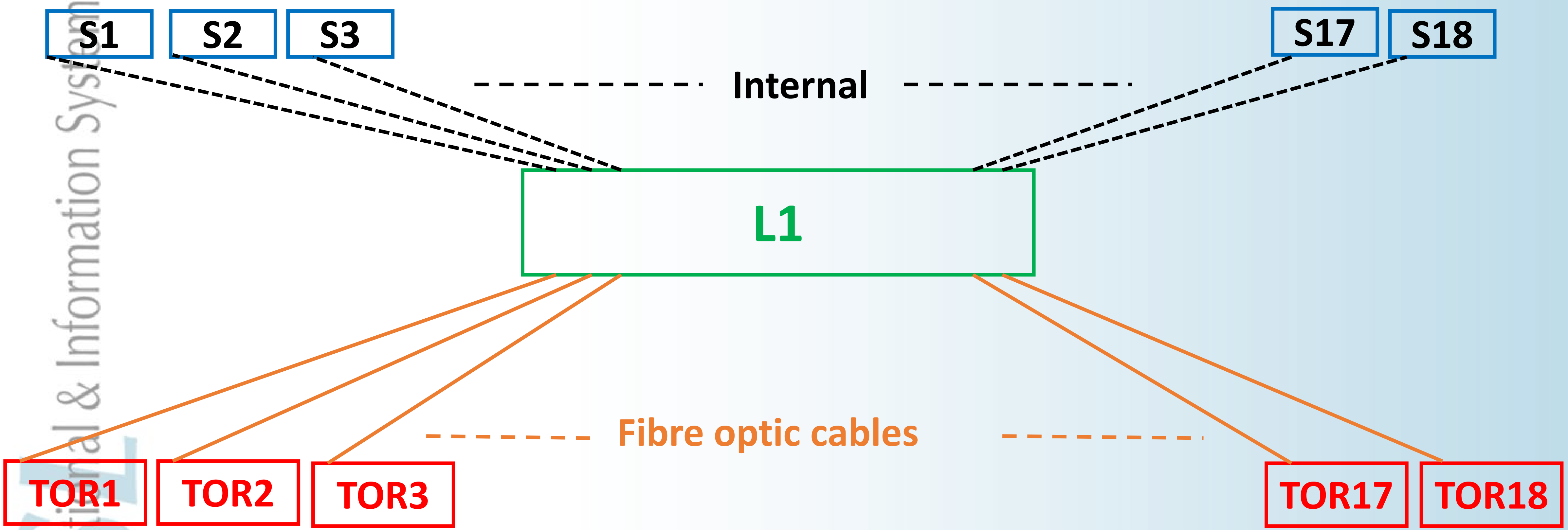
- 18 nodes / TOR (A – group)
- 4 TORs / Rack
- 72 nodes / Rack
- 63 Racks
- 4536 nodes
- 324 nodes (B – group)
- 14 B groups



# Connections across a TOR Switch (stage 1)



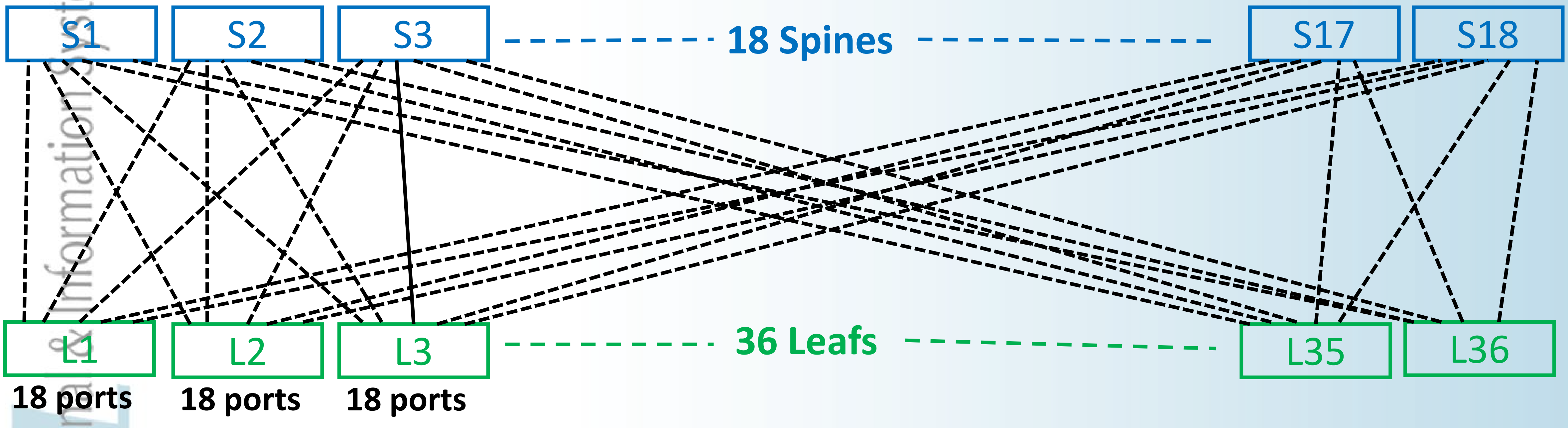
# Connectivity across a leaf (stage 2)




**Pattern**

- 19 th, 21st, 23rd, .. 35th port of TOR connects to n-th Leaf
- 20th, 22nd, 24th, … 36th port of TOR connects to (n+1)-th Leaf

# Internal connections in SX6536

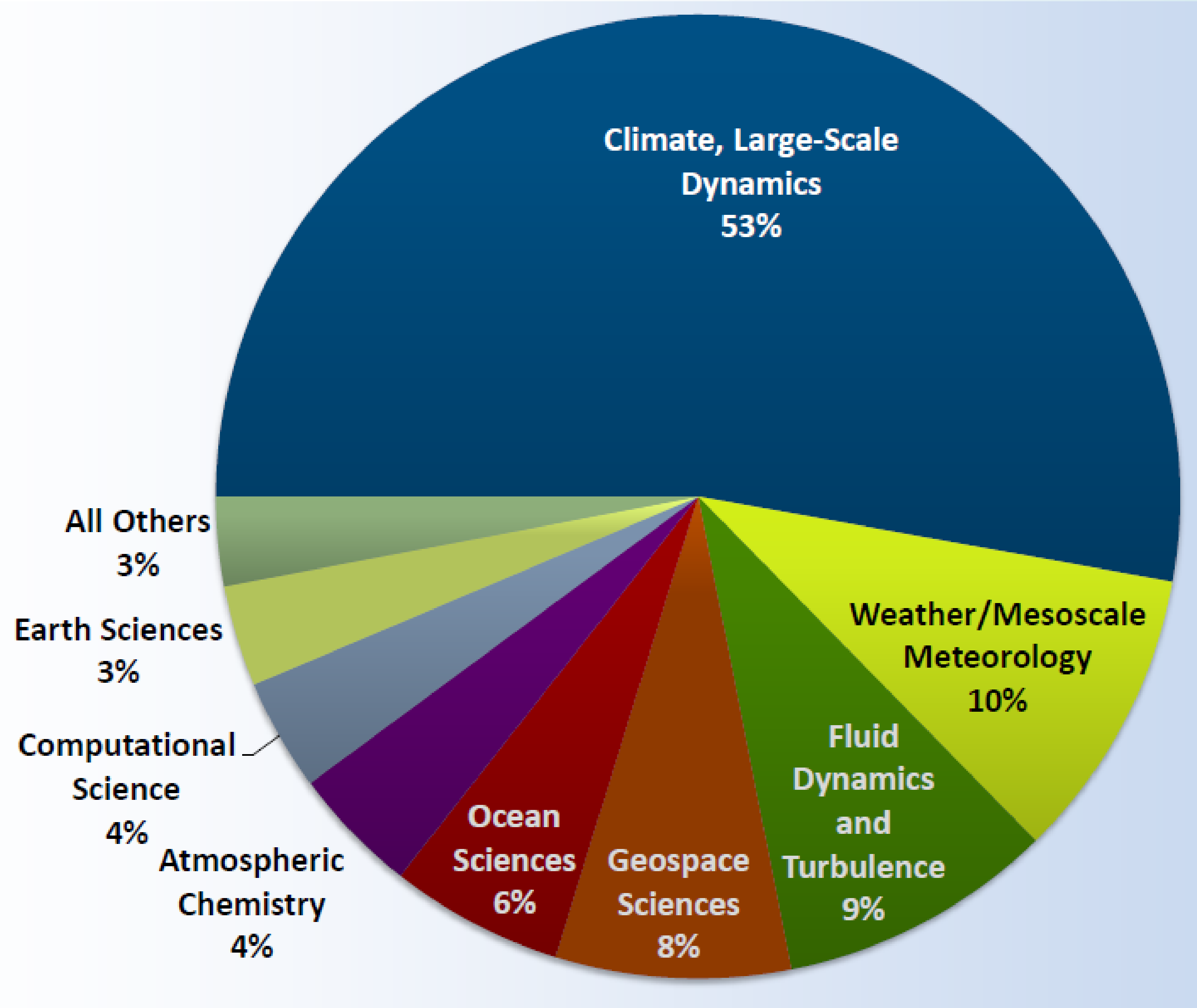



**We only have 29 leafs**

**Pattern**

- ❑ 19 th port of all the Leafs connect to S1
- ❑ $20^{th}$ port of all the leafs connect to S2 and so on

# Yellowstone Workload distribution



# Application profile

- CESM (Community Earth System Model)
  - more than 50% of our resource is spent running CESM
  - coupled earth system model with components
  - Community Atmosphere Model (CAM), most compute intensive
  - Parallel Ocean Program (POP), usually needs less resource than CAM
  - Community Land Model (CLM), Sea-Ice, River run-off etc. needs much less resource
- WRF (Weather Research Forecast)
  - about 10% resource is spent on weather prediction
  - MPAS (Model for Prediction across Scales), probably future of WRF
- Earth/Geo science, Computation science, Atmospheric chemistry, Solar and planetary science consumes rest of the pie

# Community Atmosphere Model (CAM)

- ❑ **Contains two major pieces,**
- ❑ **Dynamical core**
  - **Governs the dynamics**
  - **Supports several types of dynamics e.g. Spectral, Finite Volume, Spectral Element (SE) etc.**
  - **Solves equations within 3D spherical Shell in few**
  - **Grids (e.g. lat-lon, cubed sphere)**
  - **Resolutions ($2^{\circ}$ to $1/4^{\circ}$) in the horizontal directions**
  - **Typically it is 2D decomposition**
  - **Near neighbor communication pattern (*dominant*)**
  - **Most efficient configuration is Run in 1-task/core (16-tasks/node) and 2-threads / core**
- ❑ **Physics / Chemistry**
  - **Mostly columnar**
- ❑ ***Locality of communication* through Space Filling Curve**

# Parallel Ocean Program (POP)

- 2D decomposition over sphere
- Near neighbor and also some global communication
- Pure MPI, 1-task/core or 16-tasks/node
- Load imbalance is a problem due to non-rectangular distribution of oceanic area over globe
- Locality of communication through space filling curve

# Weather Research Forecast (WRF)

- ❑ Rectangular (lat-lon) grid
- ❑ 2D decomposition
- ❑ Dominant near neighbor communication pattern
- ❑ No special algorithm for locality of communication but usually jobs are not too big

# Model Prediction Across Scales (MPAS)

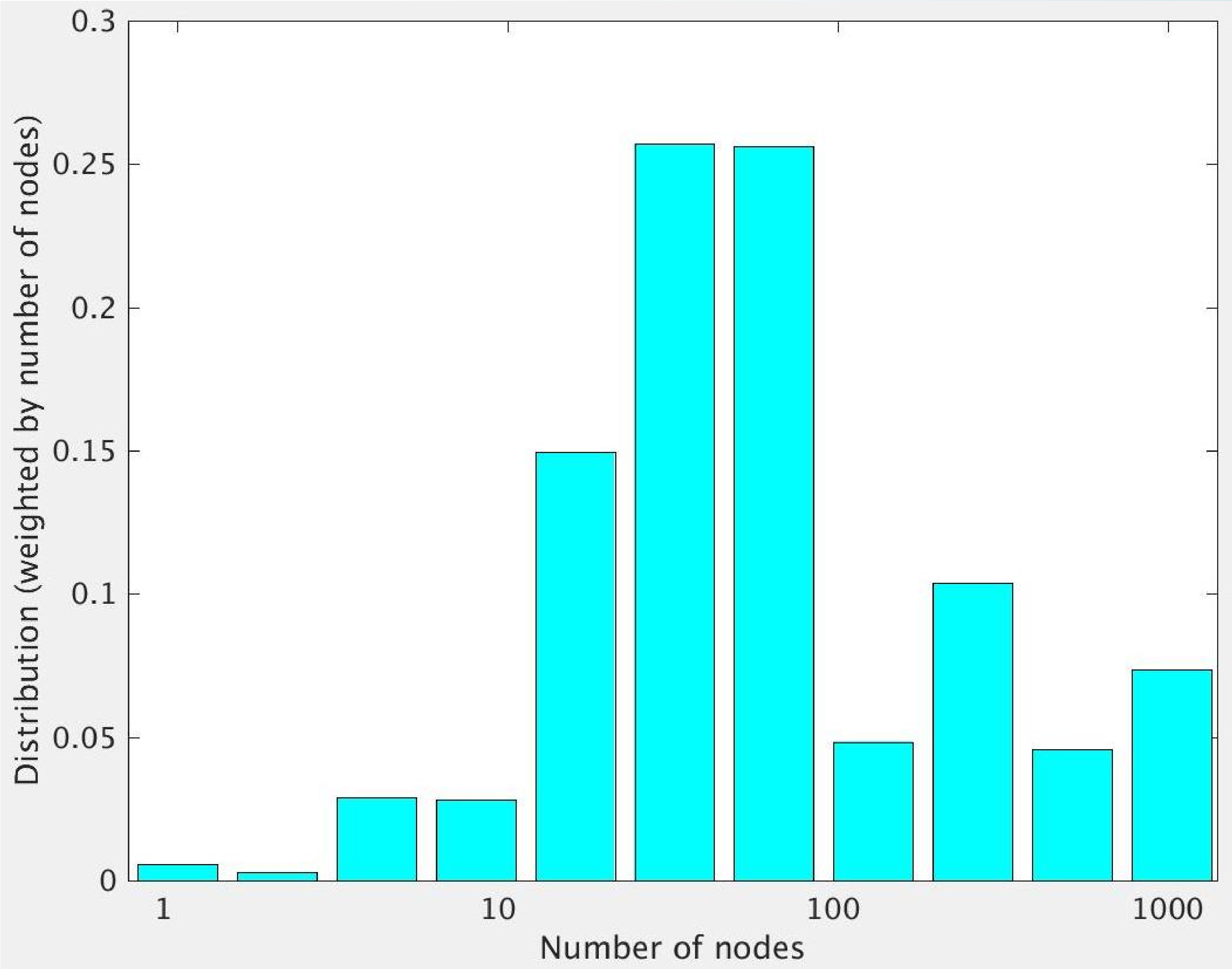
- ❑ Global grid using Voronoi Polyhedron
- ❑ 2D decomposition
- ❑ Dominant near neighbor communication pattern
- ❑ METIS applied for *communication locality*
- ❑ Overall communication overhead is relatively smaller than computation compared with other models

# NCAR application and scheduling characteristics

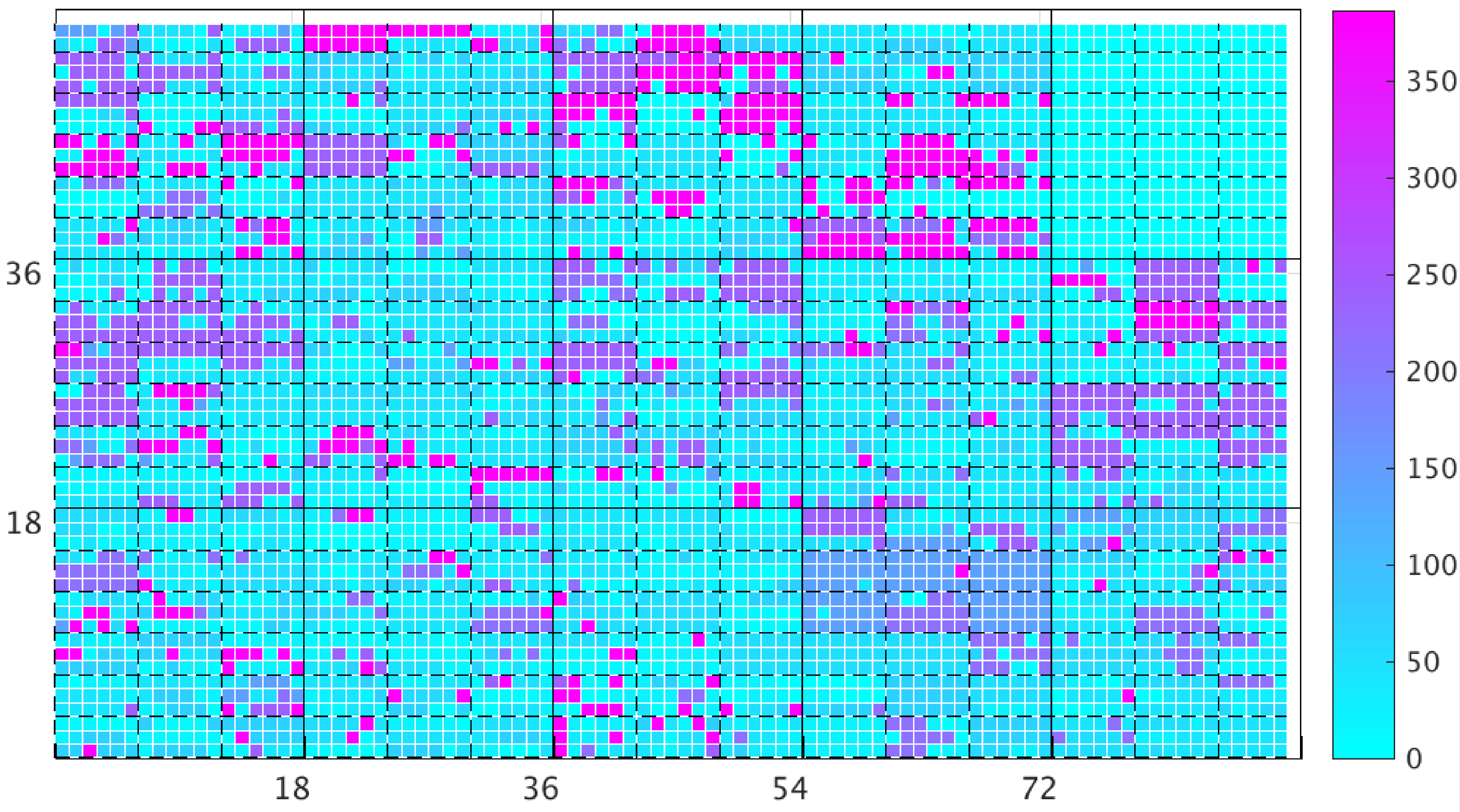
- Dominant Communication pattern is local due to
- 2D decomposition
- Near neighbor communication
- Most often Locality ensured through some utilities like SFC or METIS
- NCAR scheduler tries to schedule in index order or chunks of nodes (i.e. tries to minimize fragmentation)

***Can we hope to see these reflected in distribution of IB network traffic load ?***

# CPU-hr distribution over number of nodes in jobs



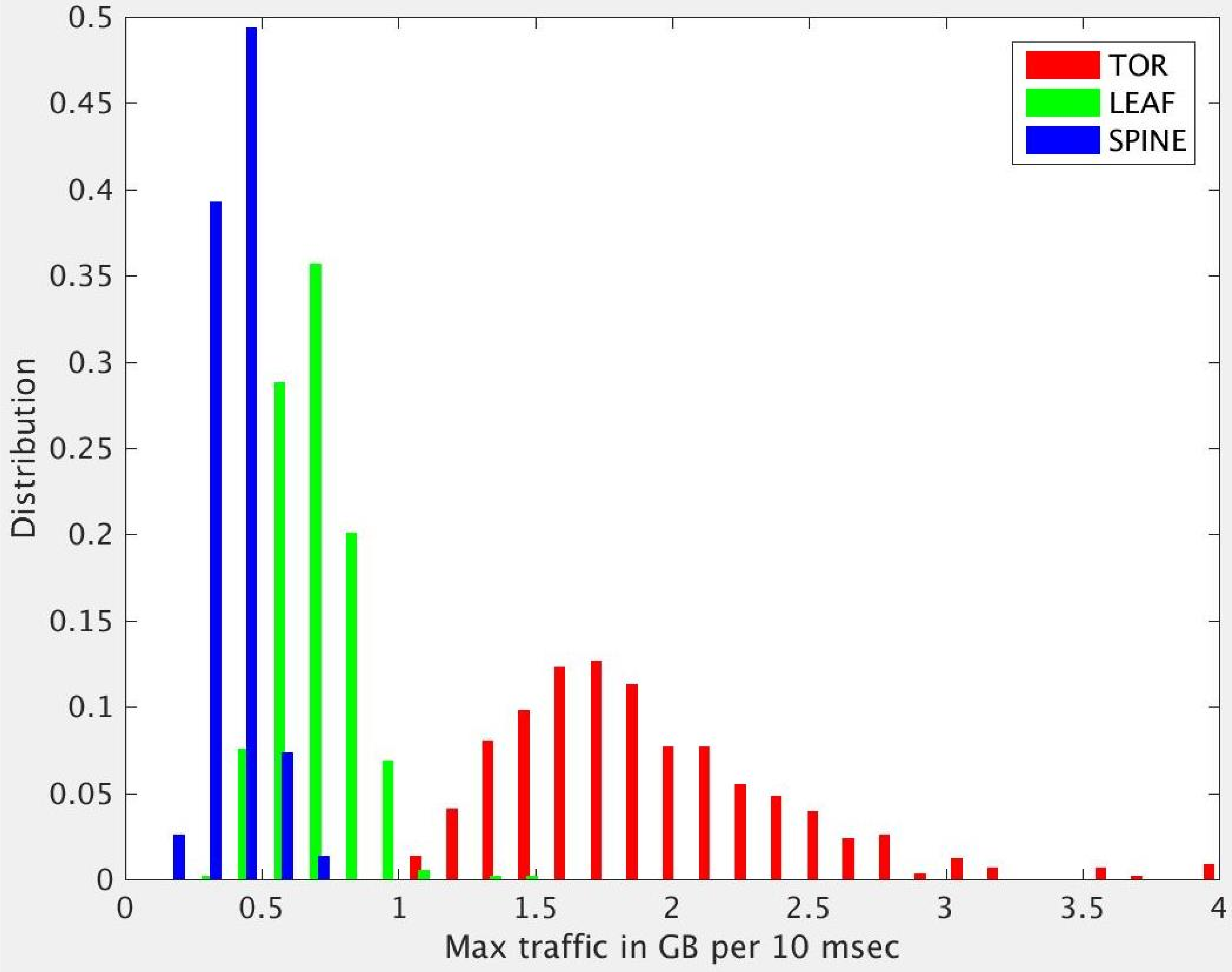
# Network Locality of node distribution in parallel jobs



# IB Traffic distribution

- Using Mellanox  OFED utility ***perfquery***
- Specifically watching 32 bit counter
  1. ***PortXmitData***
  2. ***PortRcvData***
- Across all the ports of a given stage of devices
  1. TOR
  2. LEAF and
  3. SPINE
- For many 10 milli-sec sample during heavy load
- We find …

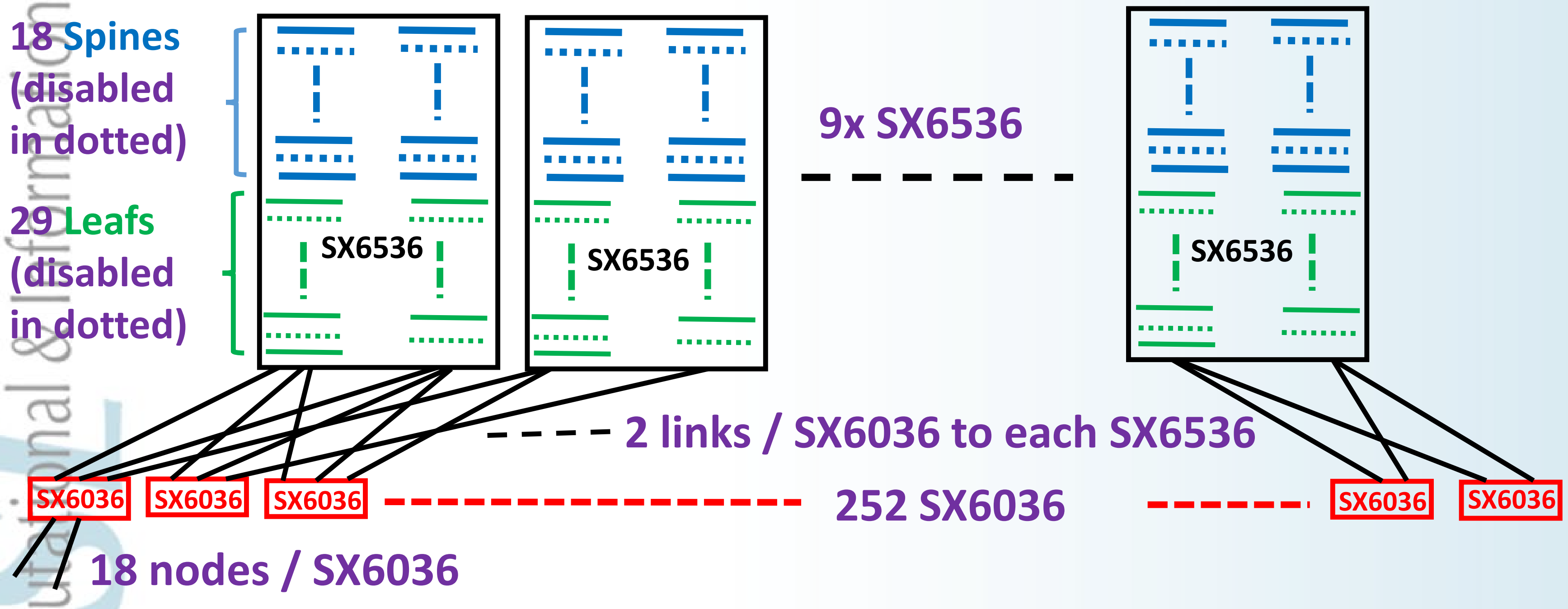
# IB traffic load distribution across devices



# How about 2:1 trimming starting from TOR

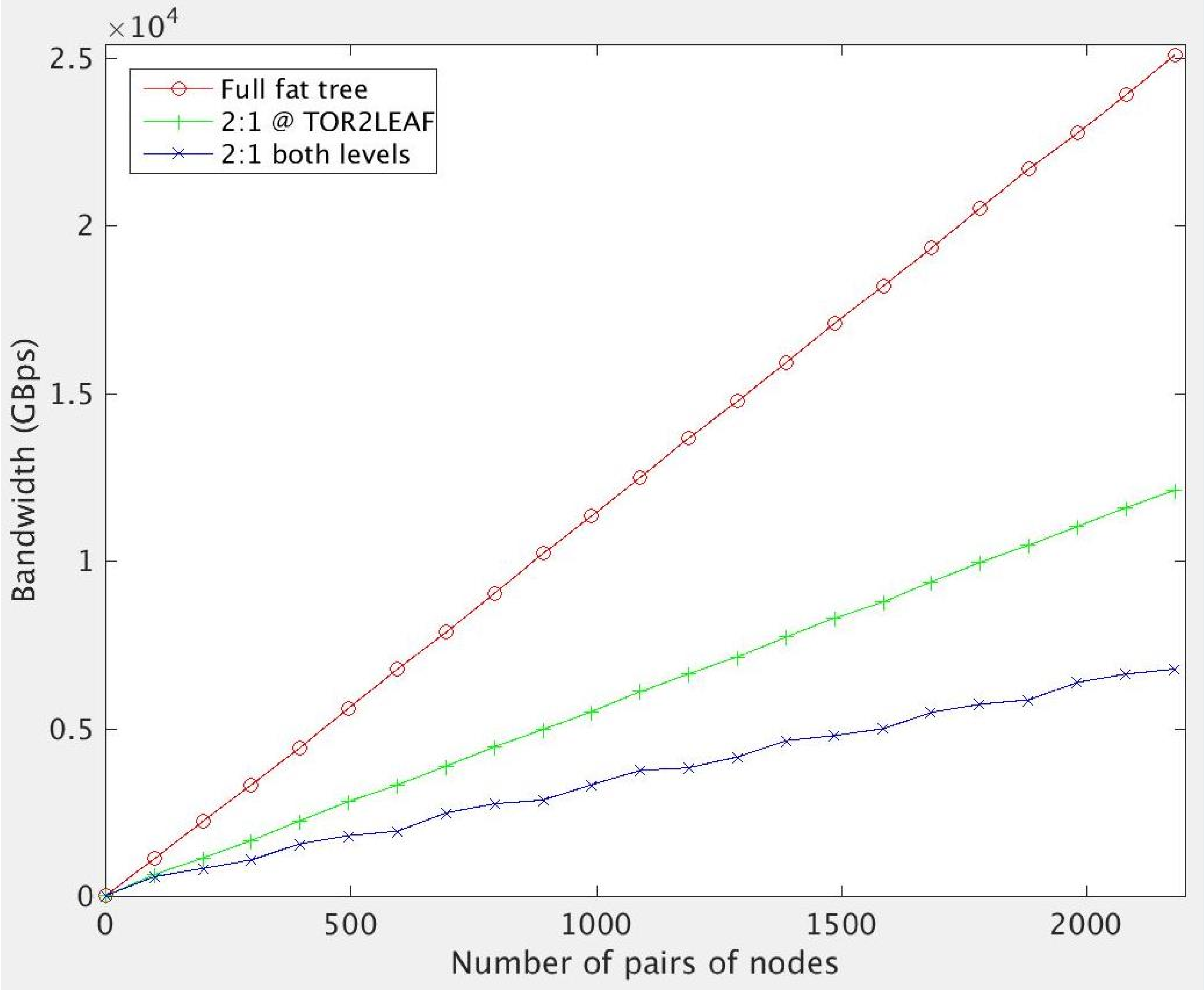
- For our experiment  we
  1. Disable 9 uplinks from TOR (in practice did not consider half the leafs while evaluating the Ftree routes)
  2. Also disable 9 uplinks from Leafs (in practice did not consider half the spines while evaluating the Ftree routes)
- Compare performance of our application kernels with baseline where everything was functional

# Yellowstone Fabric (schematic)

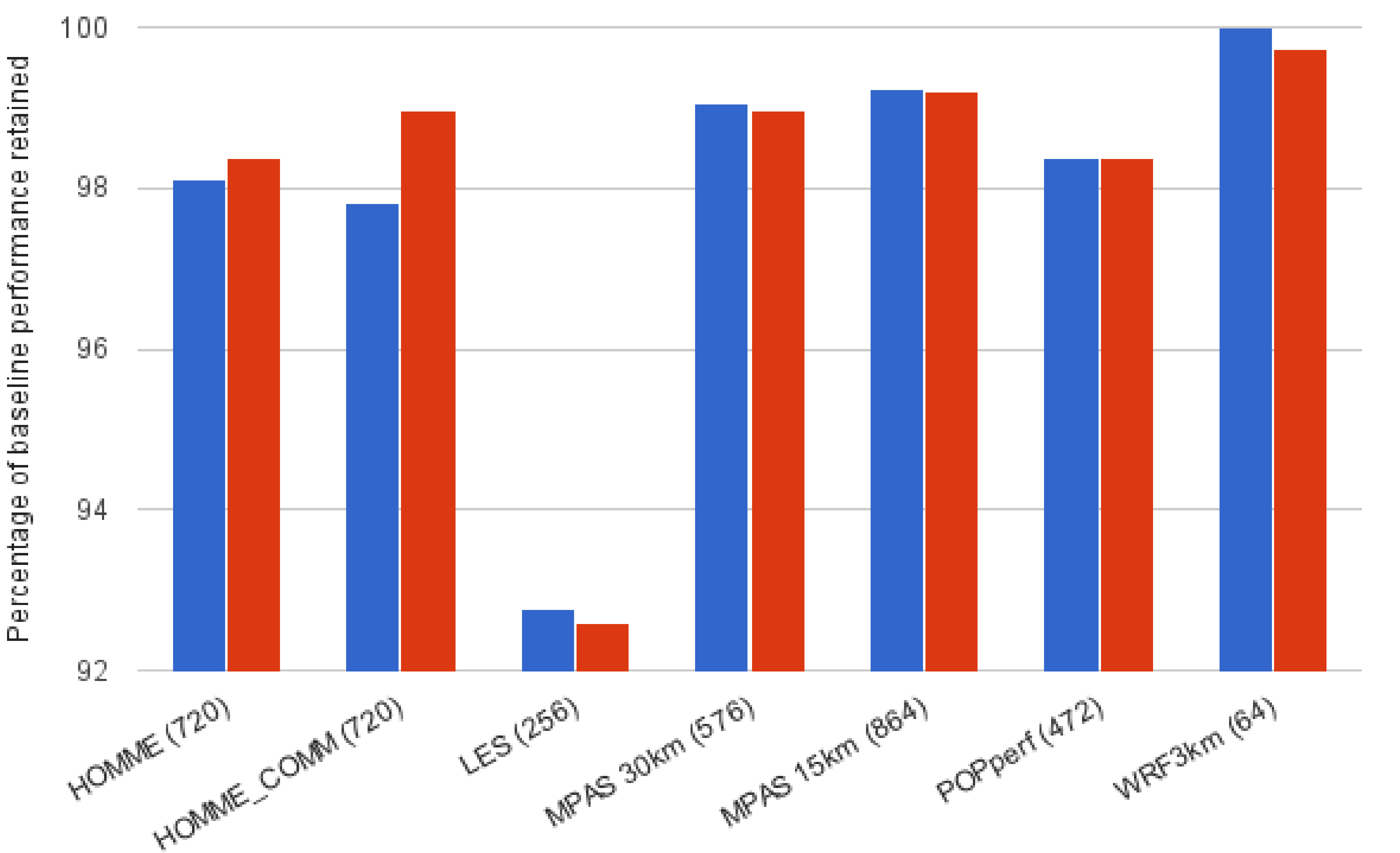



- 18 nodes / TOR (A – group)
- 4 TORs / Rack
- 72 nodes / Rack
- 63 Racks
- 4536 nodes
- 324 nodes (B – group)
- 14 B groups

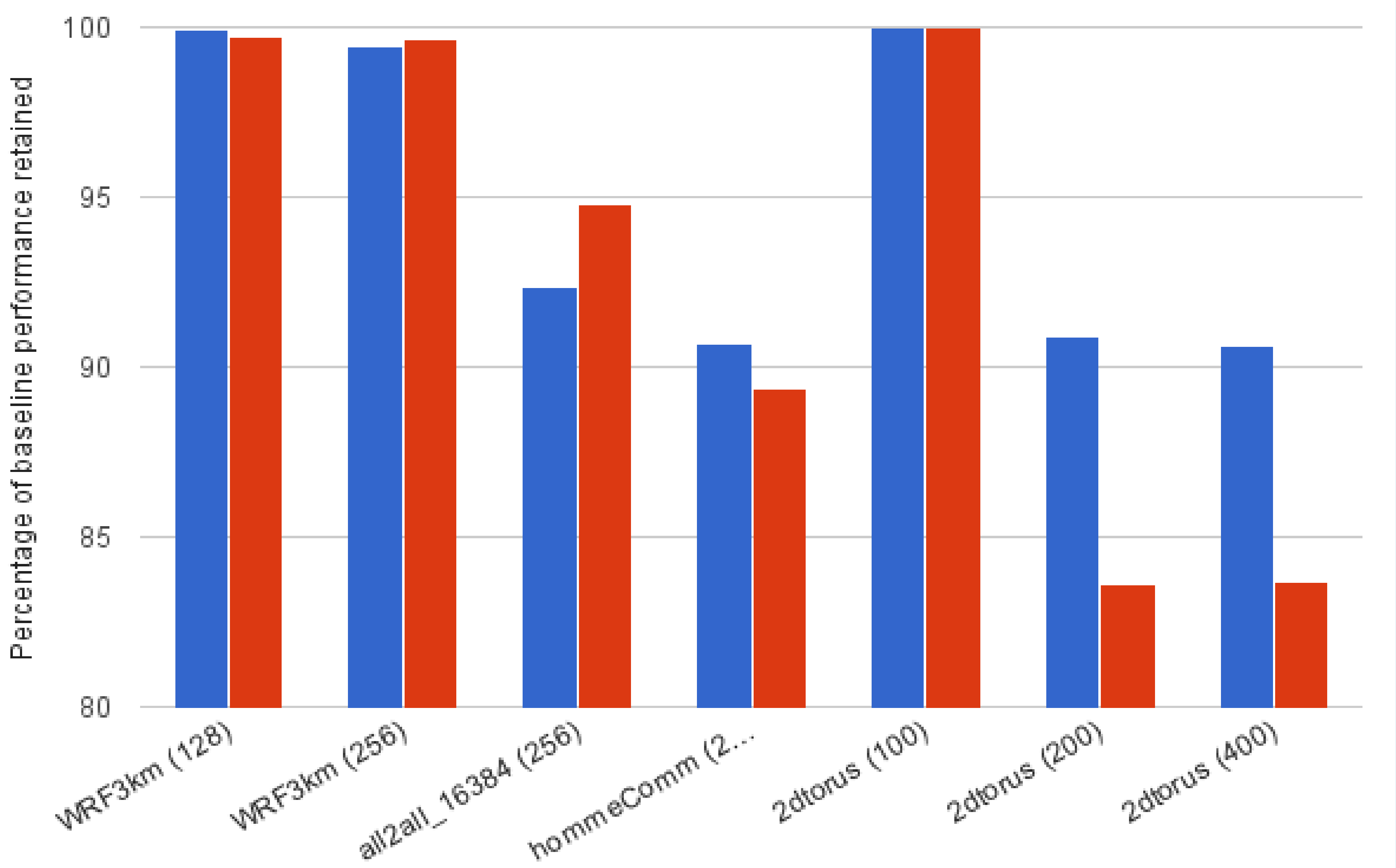
# Bisection bandwidth across the fabric



# Application performance impact



# Application performance impact



# Concluding remarks

Given our workload and distribution of jobs within fabric

- ❑ It will be cost effective to be able to trim the fabric, especially at the TOR level
- ❑ The perfquery based study is pretty non-invasive and may be of interest to others
- ❑ In practice 2:1 trimming at the leaf level is tricky, unless switch vendors consider such cost effective trimmed core switches

Questions, comments ?