Traffic Management and Optimization

Or Gerlitz & Yiftah Shahar

Voltaire

End in Mind

- To deliver a cost-effective solution for large-scale deployments built out of industry standard servers and networks using a shared file or block storage infrastructure.
- A fabric that can scale-out without performance or service degradation - the ability to build more cost-effective configurations to satisfy the same effective load.

The Challenge

Scale-out architecture:

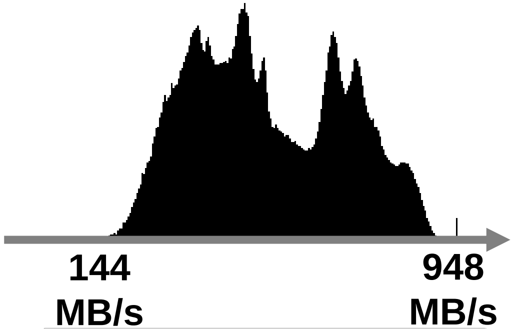
- Consolidation of multiple applications and jobs
- Multiple conflicting traffic classes:
  - Messaging
  - file or block storage
    LAN
  - Management
- Same infrastructure and even on the same wire

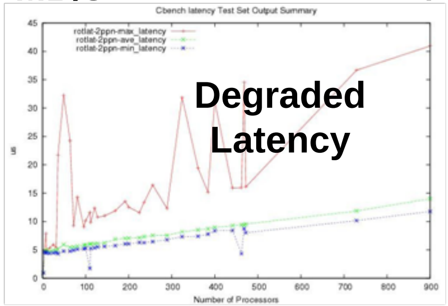
Cluster Efficiency

- Performance degrade due to unbalanced / non-optimized routing
- Congestion on one flow/class can seriously impact others
- Adaptive routing not suitable for all traffic patterns
- Very complex (or even impossible) to manually tune fabric routing

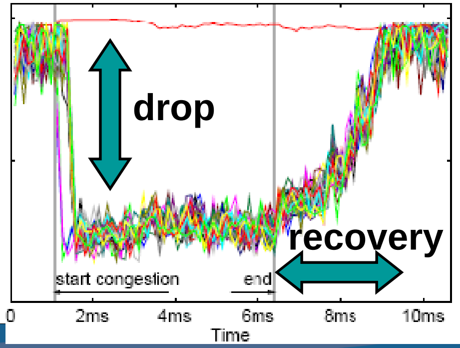
An Intelligent solution is needed that map applications to fabric configuration

Non-uniform Bandwidth





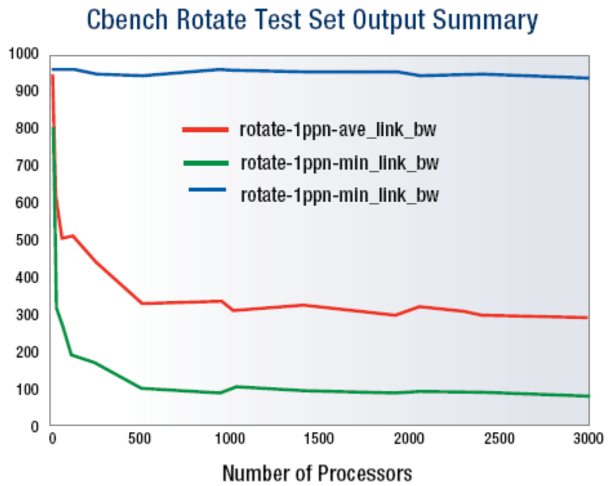
Congestion Spreading

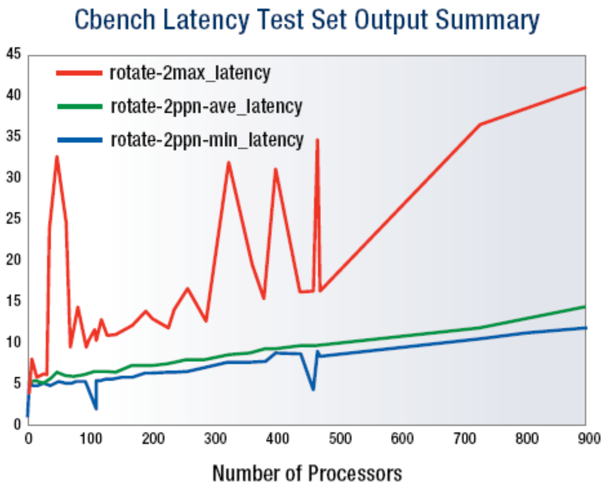


Cluster Efficiency in The Real World

Cluster Name	Server Nodes	Theoretical Bandwidth	Effective Bandwidth
TACC Ranger	3908	FBB	57.5%
LLNL Atlas	1142	FBB	55.6%
SNL Thunderbird	4390	½ FBB	40.6%

Source: http://www.unixer.de/publications/img/hoefler-ib-routing-slides.pdf

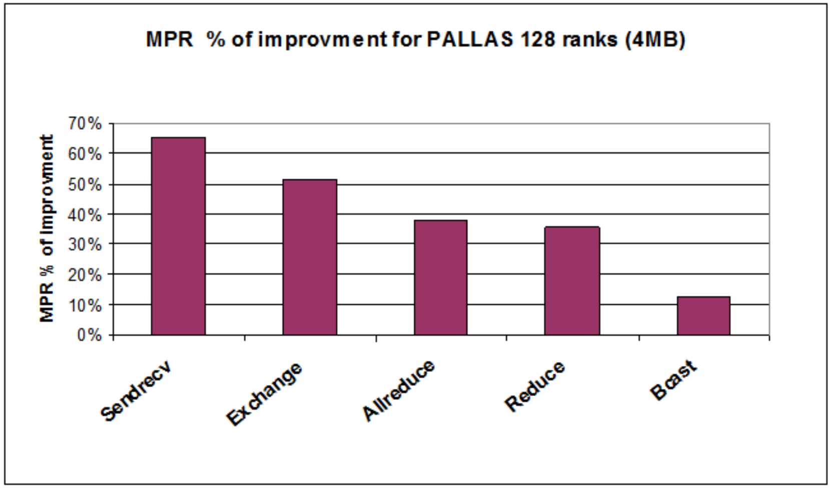




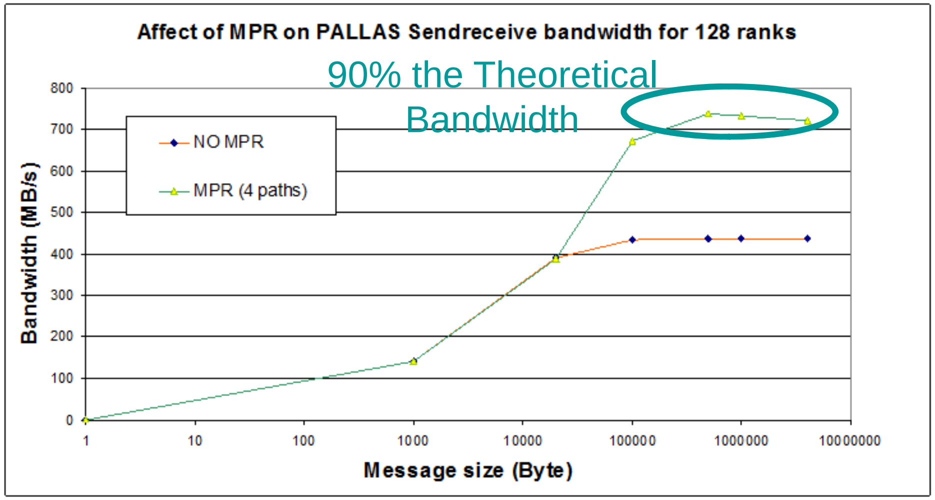
Software Solution Example: MPI Based Multipath Routing

- Implemented as part of OpenMPI & OpenSM (LMC)
- Load balance MPI traffic across fabric, coupled with SM routing

Improvement across all Pallas operations !



Up to 70% Improvement for Sendreceive !

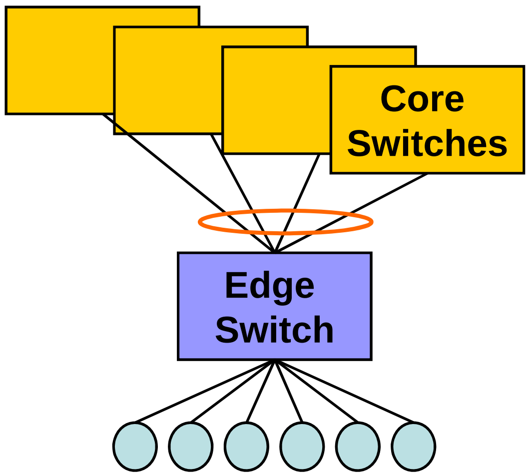


Up to 70% MPI Improvement can be delivered TODAY !

Hardware aware adaptive routing

- Edge switches distribute traffic between cores based on actual load (while preserving flow affinity)
- Spread load across fabric and reduce hot-spots

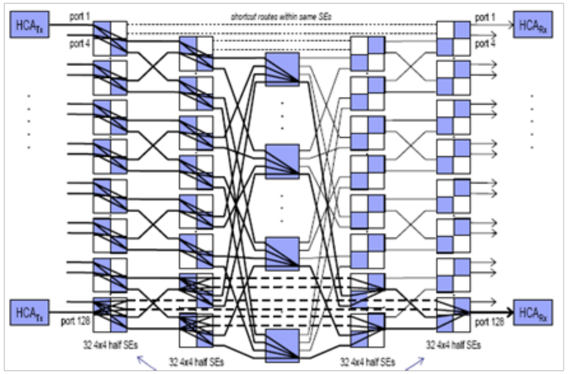
- Efficiency is heavily dependent on traffic type and parameter tuning (otherwise can degrade performance)



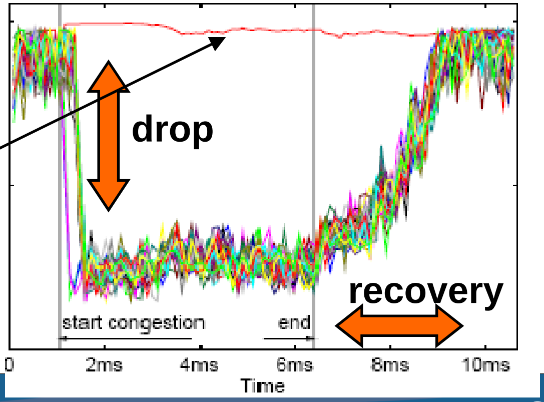
Performance Degradation - Congestion

- Nodes may receive data from multiple sources
  - Bursts of traffic arriving simultaneously
  - Collectives or multicast traffic
  - Storage traffic (many to few)
- Leading to credit starvation throughout the fabric
  - Destination ports are saturated, blocking up stream ports
  - Congestion spread across the fabric
- Cause significant performance degradation
  - Flows drop to **20%** of capacity, Latency increase
  - Take long time to recover

Congestion Spreading in a fat-tree topology



Bandwidth per flow under congestion (each flow = different color)

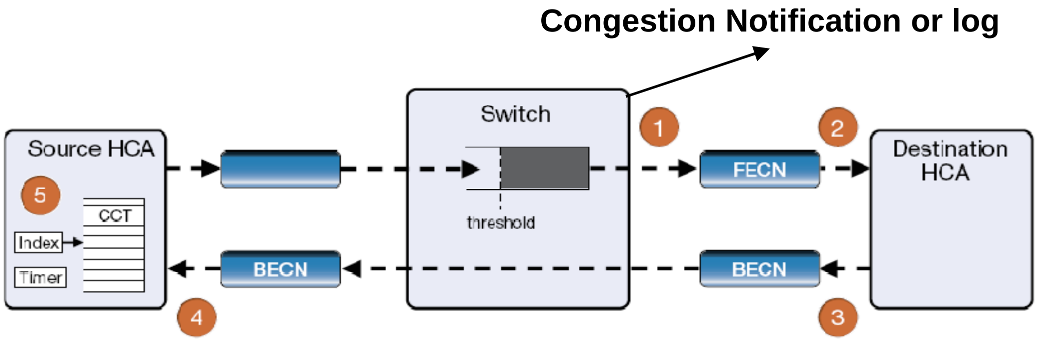


The oversubscribed flow is at max BW

Source: "Solving Hot Spot Contention Using InfiniBand Architecture Congestion Control"
www.cercs.gatech.edu/hpidc2005/presentations/GregPfister.pdf

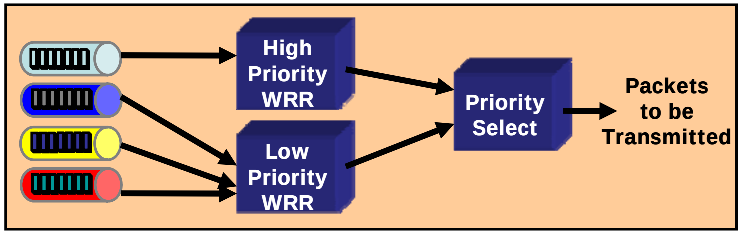
IB Congestion Control (CCA)

- When congestion is detected packets are marked (FECN)
- The returned message (BECN) cause the sources to slow down
- Notifications can be sent for tracking
- Parameters must be tuned correctly to avoid oscillations

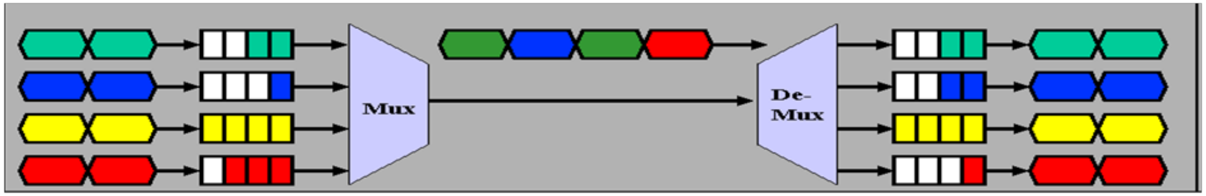


IB QoS HW architecture

- InfiniBand support up to 15 ***Virtual Lanes*** (VLs) for data
  - Each virtual lane has dedicated resources namely **separate buffering & flow control**
  - Virtual lanes are **arbitrated** at each host/switch by a dual-priority weighted round robin scheme
- Flows are classified into ***Service Levels*** (SLs) at ***end nodes***
  - Each packet sent is marked with the corresponding SL in its LRH (IB L2 header)
  - Packets mapped to VLs in each link according to their SL



H/L Weighted Round Robin (WRR) VL Arbitration



source: “QoS in OFED 1.3”
Liran Liss, Sonoma 2008

IB QoS SW architecture

1. Administrator configures the IB network QoS
   - Fabric QoS – for all IB HW devices/ports
     - SL-to-VL mappings
     - VL arbitration
   - QoS manager policy
2. Host applications communicate with the QoS manager at the SM/SA for getting their QoS level
3. The IB fabric enforces QoS while the packet is transmitted

Observations

- Blocking in cut through networks is an issue
- Different traffic classes have different requirements:
  - Collectives and storage require congestion control
  - IPC requires low-latency (high-priority)
  - Storage may use more bandwidth and not be latency sensitive
  - Hardware based adaptive routing not efficient with bursts or storage traffic
- Job layout can influence routing decisions:
  - IPC traffic typically stays within a job
  - Storage traffic fan into storage nodes
  - Management spread into all nodes
- Hardware capabilities can be destructive if used inappropriately
  - E.g. mis-configured adaptive routing or congestion management

What Is Needed

- Create multiple Classes of Service with different behavior (e.g. storage, MPI, Multicast)
  - Will dictate which mechanisms will be applied and how

- Provide flexible routing mechanism that take into account application layout and requirements

- Provide tools to effectively monitor fabric congestion and utilization, and allow application tune-up

Fabric Performance Requirements

- Traffic Classes (storage, MPI, etc')
  - Low latency, high bandwidth, burstiness, relative BW
  - Per class routing and congestion control preferences

Class of Service - CoS

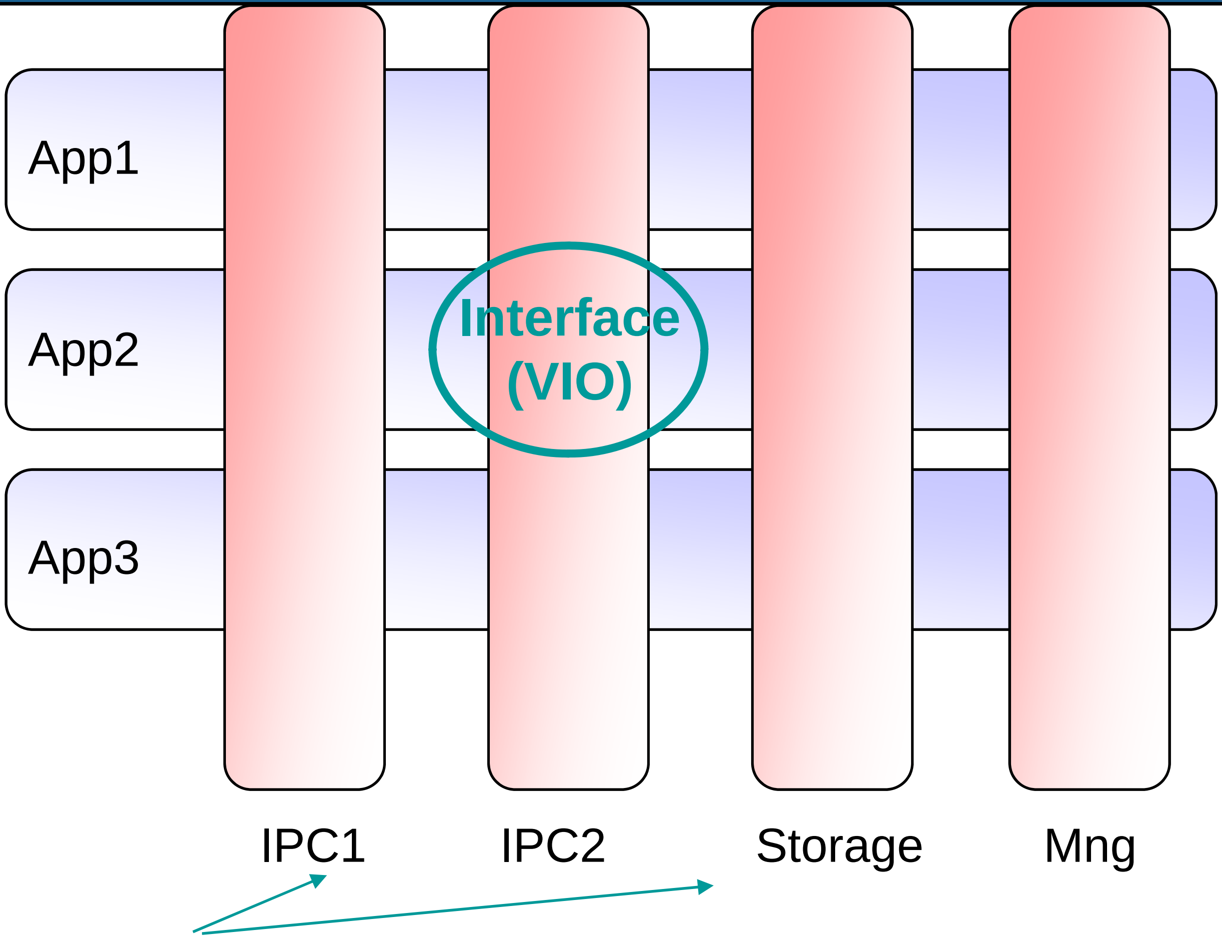
- Application layout and communication patterns
  - Which endpoints communicate, how much do they talk
  - Typical Pattern: All-to-all, many-to-one, one-to-many
  - Bandwidth requirements and limit

Application Interface

Application Service Based Optimization

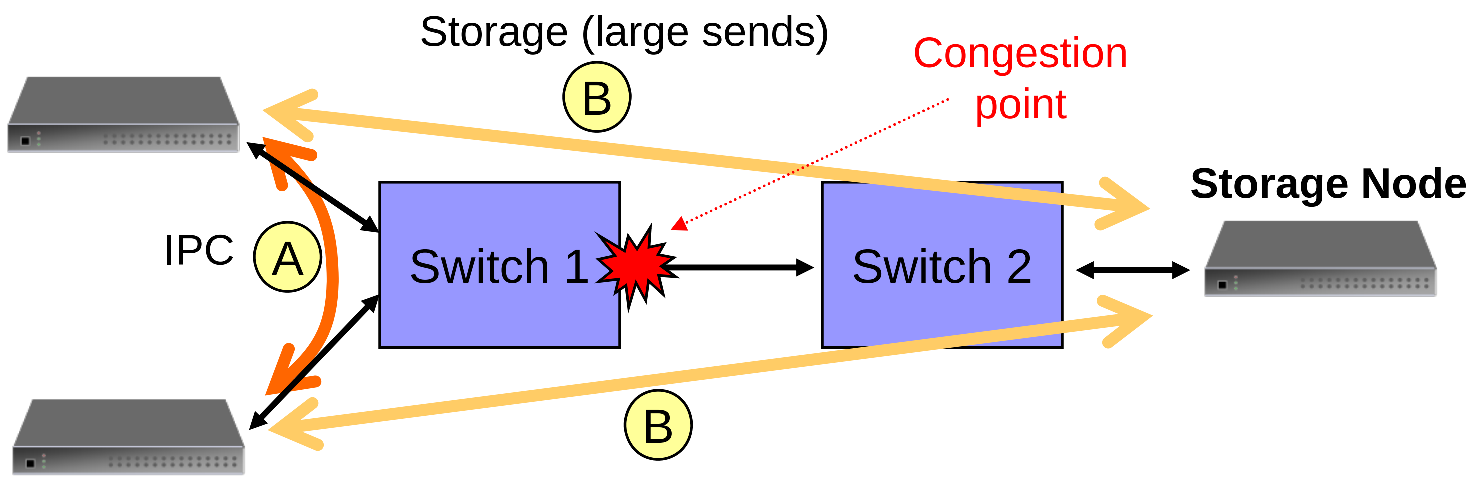
OPENFABRICS ALLIANCE

Applications/Jobs
associate with nodes and application patterns



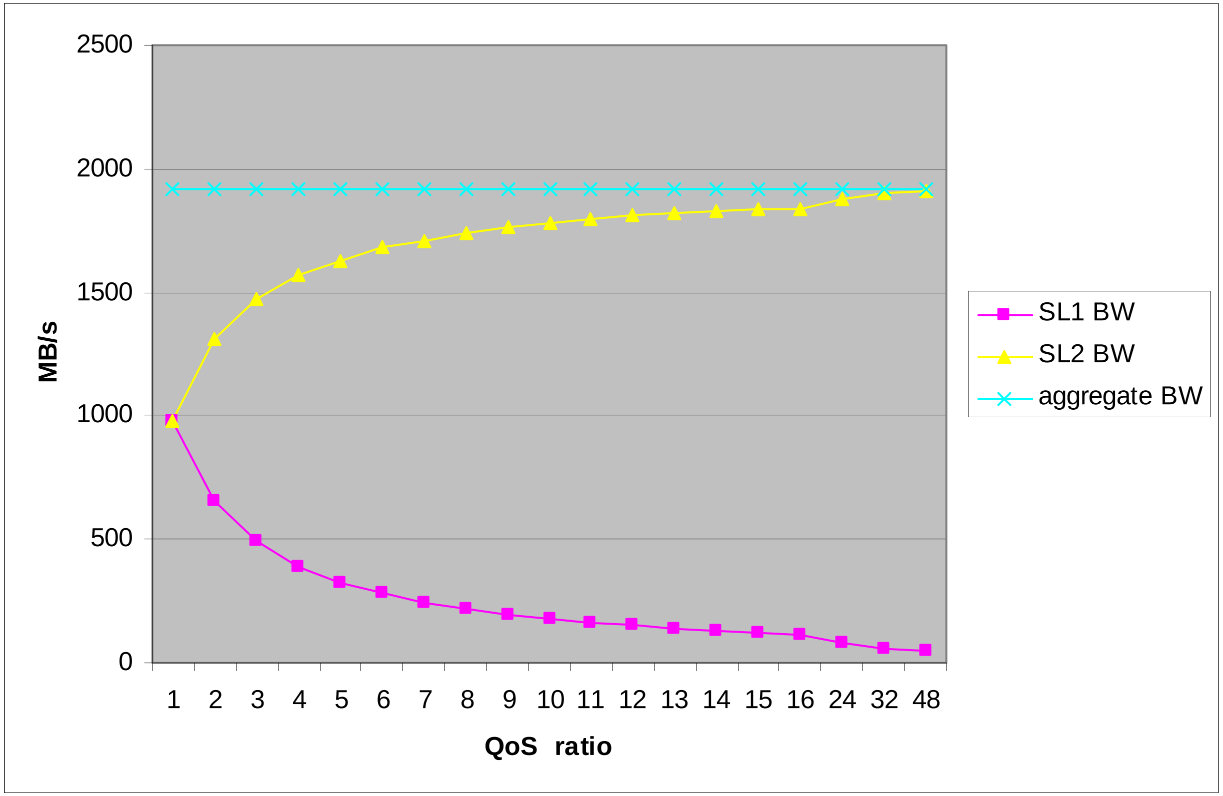
Applications Use Multiple CoS or Logical Network
Each have different behavior

Using CoS



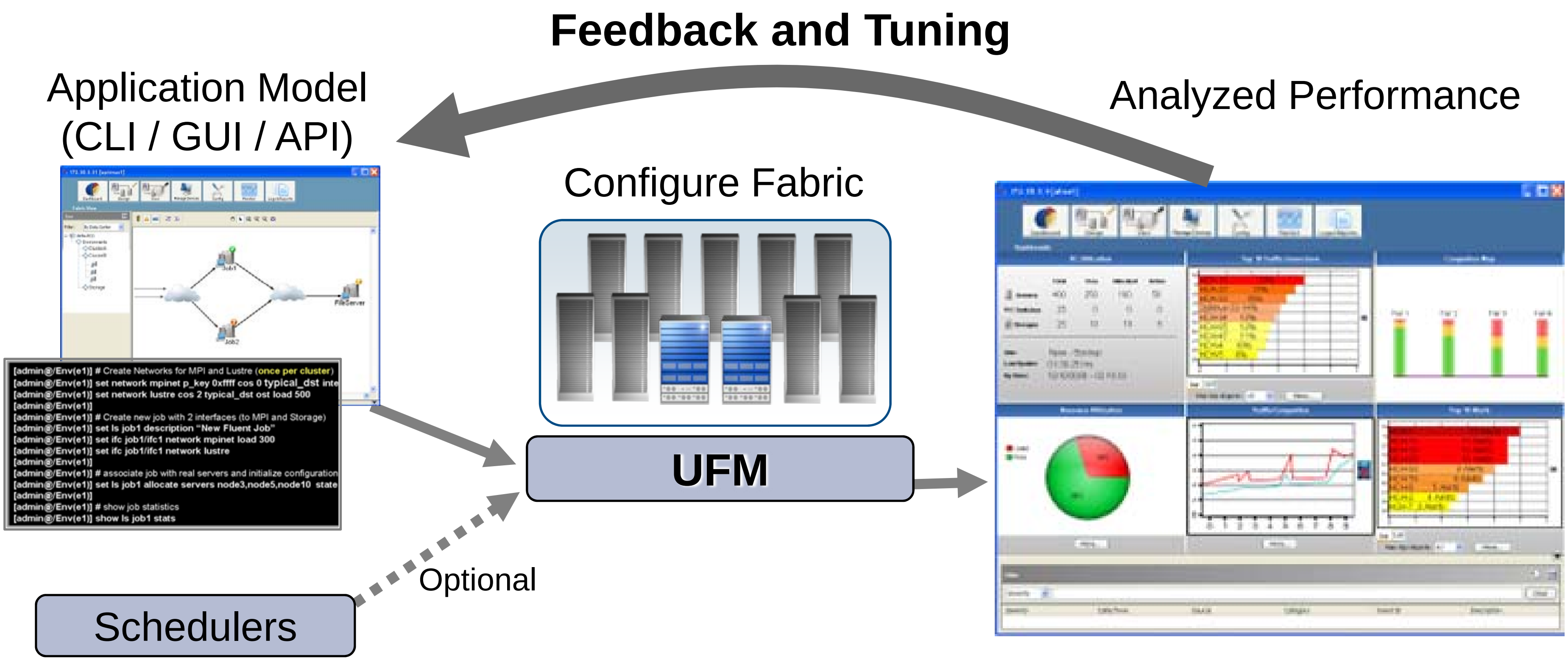
Case	A CoS	B CoS	IPC (A) Latency	
			Typ	Max
Just IPC traffic	0	None	1.2us	1.7us
IPC+Storage use the same CoS	0	0	18us	20us
IPC+Storage separate CoS	0	1	1.5us	2us

Bandwidth ratio using VL arbitration



- Two BW streams from host A to host B connected by switch
- QoS ratio - the ratio between the weight given to each VL e.g 1:1,1:2, …1:16,1:32,1:48
- For ratios > 8 used multiple instances of the VL

Configuring & Tuning System Performance With UFM



Characterize application Traffic and priorities → Optimize routing, configure QoS,… → Show traffic distribution and congestion info

Conclusions

- InfiniBand has a very reach set of capabilities yet to be explore and use
- Understand the Requirements, Design, Monitor & Troubleshoot
- You need to have unified approach for fabric management
- All of the above will be also relevant for DCB/CEE solutions

Thank You

QoS SW arch - IB Host stack

- IB L2 addresses (LIDs) – are assigned by the SM (routing)
  → SA path query is needed to establish session between nodes

- Apps / ULPs issue SA Path Queries
  - The path query was extended by IBA to specify QoS fields
  - input: Path query <SGID, DGID, PKEY, Service-ID, QoS-Class>

- SA consults QoS manager before replying
  - output: Path record <SLID, DLID, **SL, MTU, Rate, Packet life time**>

- ULPs use path record values for sending traffic
  - Configure the returned QoS elements to the IB Queue-Pair

Fabric Optimization Flow

- Fabric Optimization requires system wide solution
  - Simplistic/component level solutions can lead to worse results
- First, Gather application and I/O requirements
  - MPI, Storage, and collectives create different congestion effects, which should be addressed by different mechanisms or parameters
  - Correct job placement or job specific routing/handling reduce blocking
- Than, Increase Visibility, identify bottlenecks
  - Show meaningful real-time on about fabric performance & utilization
- Lastly, Automatically optimize traffic with “smart” software
  - Use deterministic or adaptive routing according to usage model
  - Identify congestion, Control sources for congestion when appropriate
  - Use virtual lanes and QoS to isolate congestion and prioritize traffic

2 LID per HCA port – Congestion Problem

Pair BW test:
- *Node A <--> Node C*
- *Node B <--> Node E*

Link contention: ½ of the available BW for each pair

Switch 3

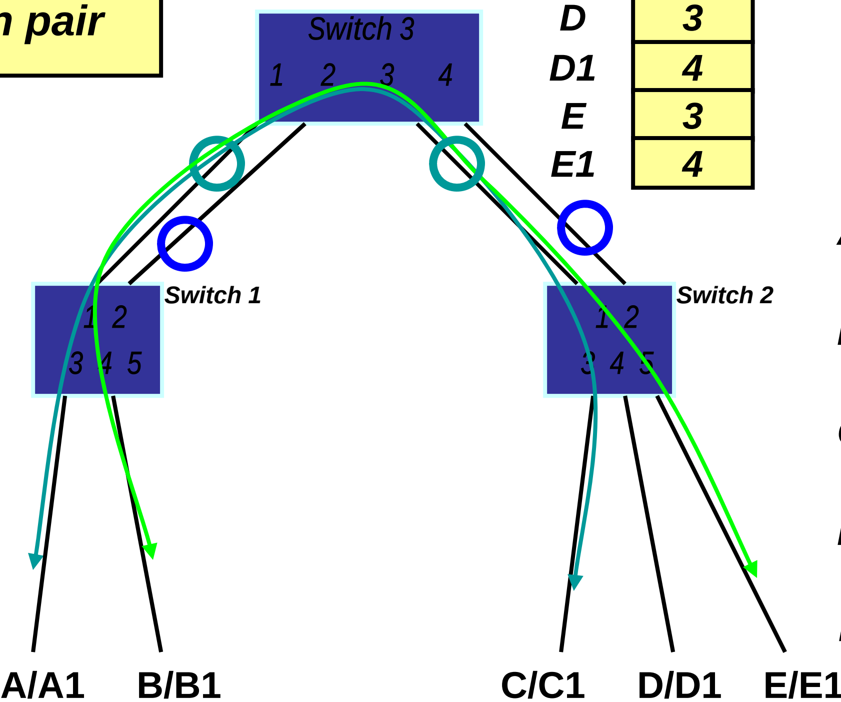
A	1
A1	2
B	1
B1	2
C	3
C1	4
D	3
D1	4
E	3
E1	4

Switch 1

A	3
A1	3
B	4
B1	4
C	1
C1	2
D	1
D1	2
E	1
E1	2

Switch 2

A	1
A1	2
B	1
B1	2
C	3
C1	3
D	4
D1	4
E	5
E1	5



Under utilize link

A <--> C
B <--> E
Over utilize link
Under utilize link

2 LID per HCA port:
Message Striping over available path

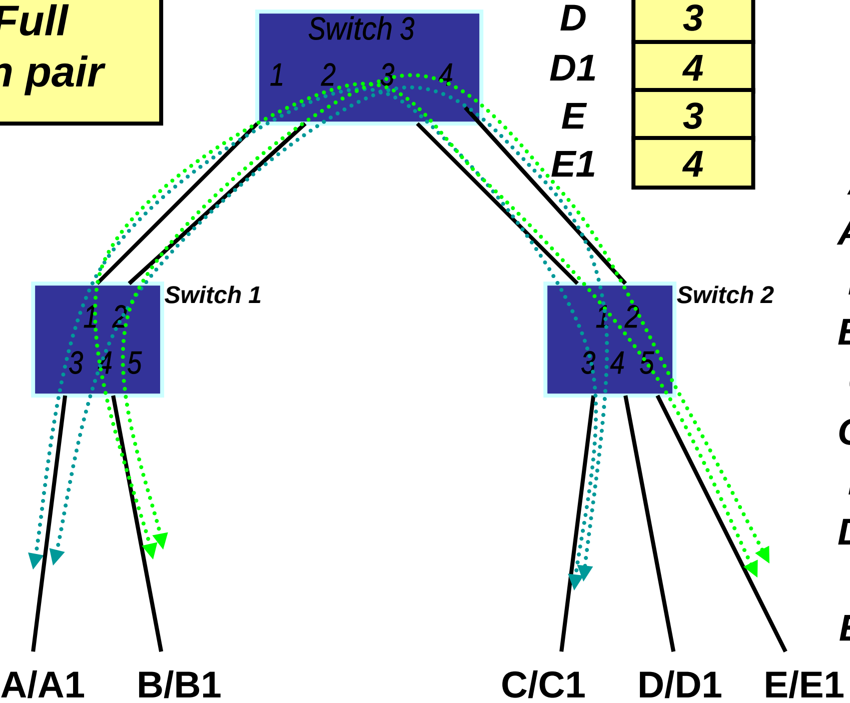
Pair BW test using large message striping over the 2 available remote LIDs:

- *Node A <--> Node C*
- *Node B <--> Node E*

NO Link contention: Full available BW for each pair

A	**1**
A1	**2**
B	**1**
B1	**2**
C	**3**
C1	**4**
D	**3**
D1	**4**
E	**3**
E1	**4**

A	**3**
A1	**3**
B	**4**
B1	**4**
C	**1**
C1	**2**
D	**1**
D1	**2**
E	**1**
E1	**2**



A	**1**
A1	**2**
B	**1**
B1	**2**
C	**3**
C1	**3**
D	**4**
D1	**4**
E	**5**
E1	**5**

A <--> C
B <--> E