Visualize, Analyze, and Correlate Networking Activities for Your Parallel Programs on InfiniBand HPC Clusters using the OSU INAM Tool

Tutorial at MUG'22

by

Hari Subramoni
The Ohio State University
subramon@cse.ohio-state.edu
http://www.cse.ohio-state.edu/~subramon

Pouya Kousha
The Ohio State University
kousha.2@buckeyemail.osu.edu
https://www.linkedin.com/in/pouya-kousha2/

Ayyappa Kolli
The Ohio State University
kolli.28@buckeyemail.osu.edu
https://www.linkedin.com/in/ayyappa-kolli/

Outline

- **Overview**
  - Motivation
  - Profiling Tools Perspective and Broad Challenges
- OSU INAM Components
- Overhead Analysis
- Downloading, Installing, and Configuring OSU INAM
- Demos
- Future Work and Concluding Remarks

Introduction

- With the recent advances in HPC, providing efficient data movement is critical to have end-to-end high throughput solutions
- A detailed and real-time insight of network level and the communication is needed to identify and alleviate performance bottlenecks
- A newer set of challenges arise as HPC systems are becoming larger, users expect to have better capabilities like real-time profiling at fine granularity and scalability concerns
- The problem of identifying any performance issues with HPC and DL applications is akin to finding “a needle in a haystack” in HPC communication layers

Frontier

Fugaku

Summit

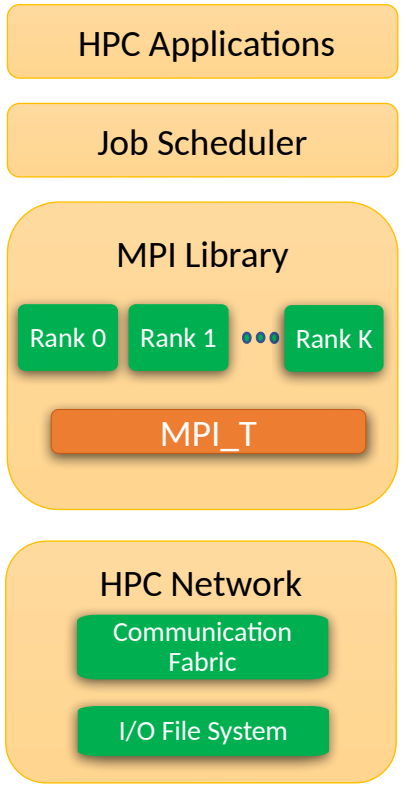
Sierra

Sunway TaihuLight

Profiling Tools Perspective and Broad Challenges

- There are 30+ profiling tools for HPC systems
- System level vs User level
  - User level novelty
- Different set of users have different needs
  - HPC administrators
  - HPC Software developers
  - Domain scientists
- Different HPC layers to profile
  - How to correlate them and pinpoint the problem source?

Unified and holistic view for all users



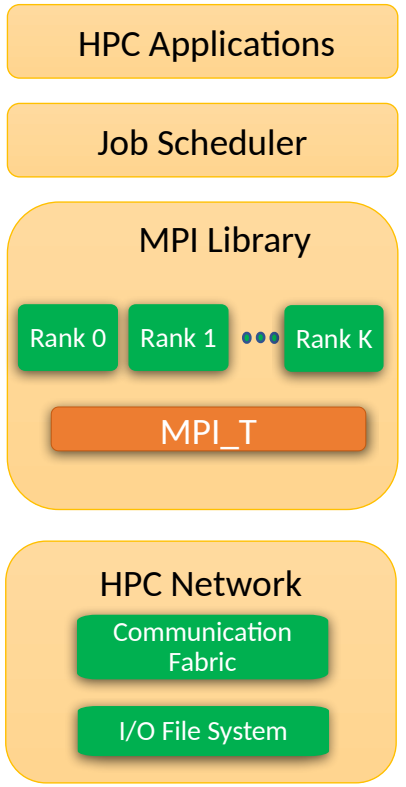
Summary of existing profiling tools and their capabilities

Tools	MPI Runtime		
	Applications	**Network Fabric**	**Job scheduler**
INAM*	✓	✓	✓
TAU	✓	✓	✗
HPCToolkit	✓	✗	✗
Intel Vtune	✓	✗	✗
IPM	✓	✗	✗
mpiP	✓	✗	✗
Intel ITAC	✓	✗	✗
ARM MAP	✓	✗	✗
HVProf	✓	✗	✗
PCP(used by XDMOD)	✗	✓	✓
Prometheus	✗	✓	✓
Mellanox FabricIT	✗	✓	✗
BoxFish	✗	✓	✗
LDMS	✗	✓	✗

* This design has been publicly released on 06/08/2020 and is available for free here https://mvapich.cse.ohio-state.edu/tools/osu-inam/

Profiling Tools Perspective and Broad Challenges

- Understanding the interaction between applications, MPI libraries, I/O and the communication fabric is challenging
  - Find root causes for performance degradation
  - Identify which layer is causing the possible issue
  - Understand the internal interaction and interplay of MPI library components and network level
  - Online profiling



How can we design a tool that enables holistic, real-time, scalable and in-depth understanding of communication traffic through tight integration with the MPI runtime and job scheduler?

Overview of the MVAPICH2 Project

- High Performance open-source MPI Library
- Support for multiple interconnects
  - InfiniBand, Omni-Path, Ethernet/iWARP, RDMA over Converged Ethernet (RoCE), AWS EFA, Rockport Networks, and Slingshot10/11, Broadcom, Cornelis Networks OPX
- Support for multiple platforms
  - x86, OpenPOWER, ARM, Xeon-Phi, GPGPUs (NVIDIA and AMD)
- Started in 2001, first open-source version demonstrated at SC '02
- Supports the latest MPI-3.1 standard
- http://mvapich.cse.ohio-state.edu
- Additional optimized versions for different systems/environments:
  - MVAPICH2-X (Advanced MPI + PGAS), since 2011
  - MVAPICH2-GDR with support for NVIDIA (since 2014) and AMD (since 2020) GPUs
  - MVAPICH2-MIC with support for Intel Xeon-Phi, since 2014
  - MVAPICH2-Virt with virtualization support, since 2015
  - MVAPICH2-EA with support for Energy-Awareness, since 2015
  - MVAPICH2-Azure for Azure HPC IB instances, since 2019
  - MVAPICH2-X-AWS for AWS HPC+EFA instances, since 2019
- Tools:
  - OSU MPI Micro-Benchmarks (OMB), since 2003
  - OSU InfiniBand Network Analysis and Monitoring (INAM), since 2015

- **Used by more than 3,275 organizations in 90 countries**
- **More than 1.61 Million downloads from the OSU site directly**
- Empowering many TOP500 clusters (June '22 ranking)
  - 6th, 10,649,600-core (Sunway TaihuLight) at NSC, China
  - 16th, 448, 448 cores (Frontera) at TACC
  - 30th, 288,288 cores (Lassen) at LLNL
  - 42nd, 570,020 cores (Nurion) in South Korea and many more
- available with software stacks of many vendors and Linux Distros (RedHat, SuSE, OpenHPC, and Spack)
- Partner in the 16th ranked TACC Frontera system
- **Empowering Top500 systems for more than 20 years**

Overview of OSU InfiniBand Network Analysis and Monitoring (INAM) Tool

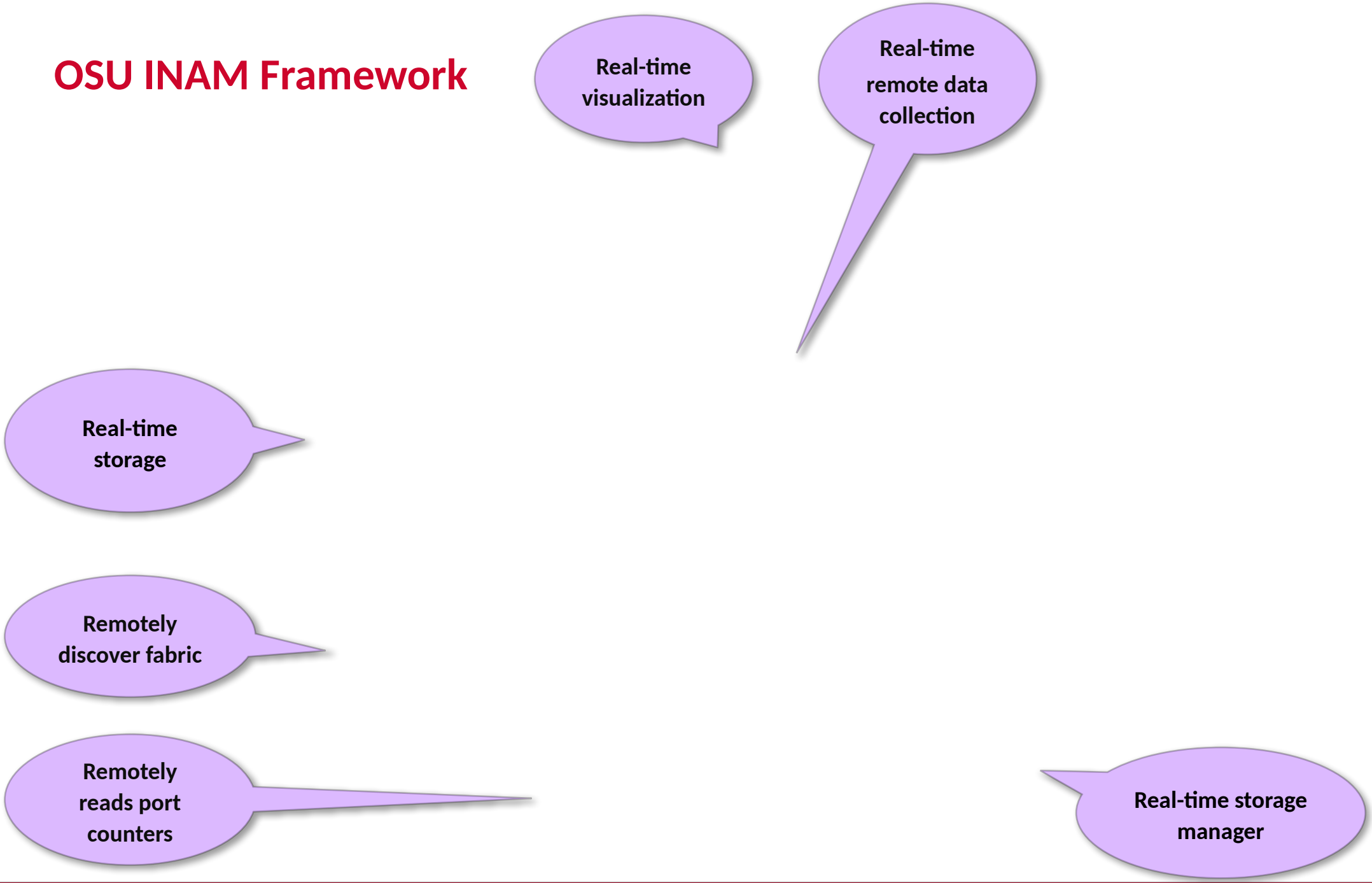
- **A network monitoring and analysis tool that is capable of analyzing traffic on the InfiniBand network with inputs from the MPI runtime**
  - http://mvapich.cse.ohio-state.edu/tools/osu-inam/
- **Monitors IB clusters in real time by querying various subnet management entities and gathering input from the MPI runtimes**
- **Capability to analyze and profile node-level, job-level and process-level activities for MPI communication**
  - Point-to-Point, Collectives and RMA
- **Ability to filter data based on type of counters using "drop down" list**
- **Remotely monitor various metrics of MPI processes at user specified granularity**
- **"Job Page" to display jobs in ascending/descending order of various performance metrics in conjunction with MVAPICH2-X**
- **Visualize the data transfer happening in a "live" or "historical" fashion for entire network, job or set of nodes**
- **Sub-second port query and fabric discovery in less than 10 mins for ~2,000 nodes**

- **OSU INAM 0.9.8 released (08/11/2022)**
  - Support for MySQL and InfluxDB as database backends
  - Enhanced database insertion using InfluxDB
  - Support for continuous queries to improve visualization performance
  - Support for SLURM multi-cluster configuration
  - Significantly improved database query performance when using InfluxDB
  - Support for automatic data retention policy when using InfluxDB
  - Support for PBS and SLURM job scheduler as config time
  - Ability to gather and display Lustre I/O for MPI jobs
  - Enable emulation mode to allow users to test OSU INAM tool in a sandbox environment without actual deployment
  - Generate email notifications to alert users when user defined events occur
  - Support to display node-/job-level CPU, Virtual Memory, and Communication Buffer utilization information for historical jobs
  - Support to handle multiple job schedulers on the same fabric
  - Support to collect and visualize MPI_T based performance data
  - Support for MOFED 4.5, 4.6, 4.7, and 5.0
  - Support for adding user-defined labels for switches to allow better readability and usability
  - Support authentication for accessing the OSU INAM webpage
  - Optimized webpage rendering and database fetch/purge capabilities
  - Support to view connection information at port level granularity for each switch
  - Support to search switches with name and lid in historical switches page
  - Support to view information about Non-MPI jobs in live node page

Outline

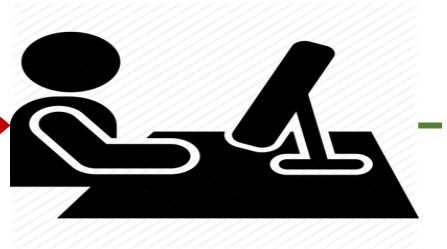
- Overview
- **OSU INAM Components**
- Overhead Analysis
- Downloading, Installing, and Configuring OSU INAM
- Demos
- Future Work and Concluding Remarks

OSU INAM Framework



Flow of Using OSU INAM

Insight and Performance Recommendations



Tuning and Applying Recommendations

REMOTE OSU INAM

- Notifications and Feedback
- Real-time Visualization
- I/O and MPI Traffic Interaction
- Measurements
- Metric Collectors

Job Scheduler

- SLURM
- PBS Torque

HPC/DL Applications

- Co-designed Scientific Applications using Proposed MPI_T Extensions
- TensorFlow
- PyTorch
- MxNet
- Co-designed Horovod using Proposed MPI_T Extensions

MPI Library with Support for Enhanced MPI_T-based Introspection

- Enhanced MPI_T Sessions
- Gathering I/O Traffic Information
- Enhanced PVAR Infrastructure
  - Counters
  - Buckets
  - Timers

HPC Network

- InfiniBand
- Lustre, NFS
- CPU/Mem Usage

Outline

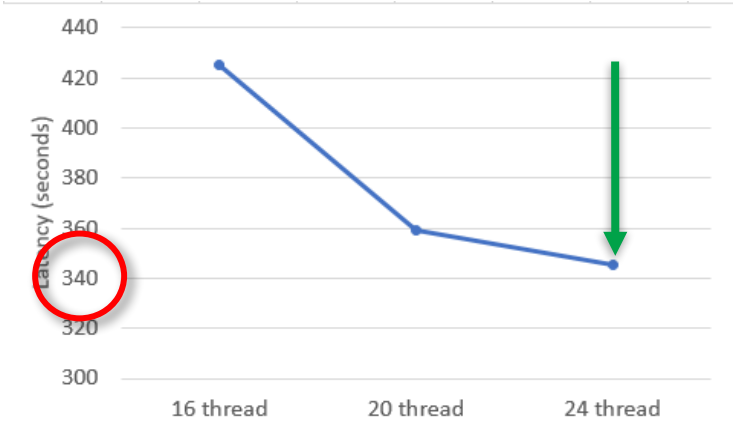
- Overview
- OSU INAM Components
- **Overhead Analysis**
  - **Experiment Setup**
  - **Fabric Discovery Evaluation**
  - **Port Inquiry Evaluation**
  - **Visualization and MPI_T Evaluation**
  - **Application-Level Overhead Analysis and Scaling**
- Downloading, Installing, and Configuring OSU INAM
- Demos
- Future Work and Concluding Remarks

Experimental Setup

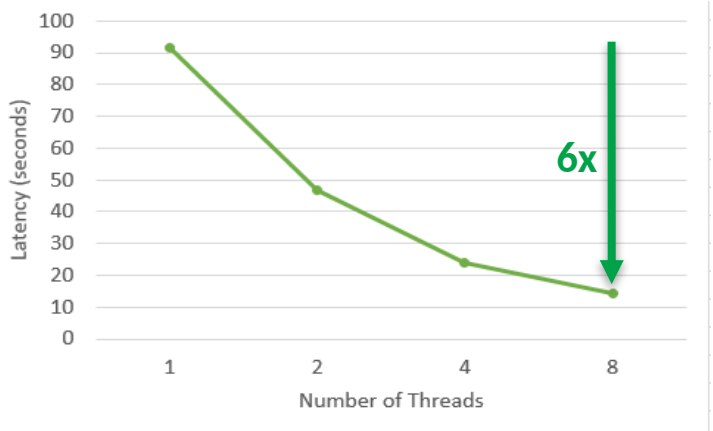
Cluster/Configuration	RI	OSC
Overview	Equipped with MT26428 QDR ConnectX-2 HCAs with PCI-Express interfaces	3 heterogeneous clusters connected to the same InfiniBand fabric
#Nodes	146	1,428
#Links	542	3,402
#Switches	20 (36 ports/switch)	114 (36 ports/switch)
CPU	Intel Xeon 2.53Ghz	Intel(R) Xeon(R) CPU E5-2680 v3 @ 2.50GHz
Cores/node for OSU INAM	8	24
L3 cache	12MB	30MB
Memory/node	12GB	128GB
Switch Technology	Mellanox MTS3610QDR	Two clusters have Mellanox EDR InfiniBand (100Gbps) and one with FDR/EN (56Gbps)
Job Scheduler	SLURM	PBS

Fabric Discovery Evaluation

- We evaluated the impact of multithreading and performance variability
- To be fair and mimic the behavior of real deployment, we set some threads for port inquiry while measuring performance of fabric inquiry.
- The timings measured include discovery of fabric and insertion of data into the database



Impact of multi-threading on fabric discovery module on OSC cluster with hourly interval for 1428 nodes



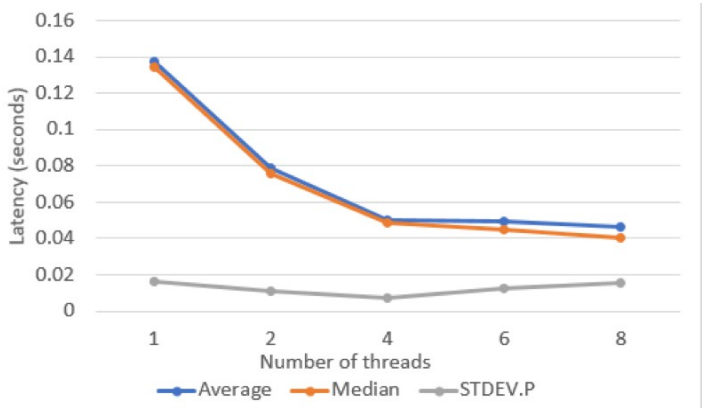
Impact of multi-threading on fabric discovery module on RI cluster with 1-minute interval

Histogram of the time taken for fabric discovery module using 8 threads on RI cluster. Port inquiry frequency is set to 1 sec

- Enhanced performance for fabric discovery using optimized OpenMP-based multi-threaded designs with **6x speedup compared to serial version**

Port Inquiry Evaluation

- We evaluated the impact of multithreading and performance variability
- To be fair and mimic the behavior of real deployment, we set some threads for fabric module while measuring performance of port inquiry.
- The timings measured include reading port metrics and insertion of data into the database



Impact of multi-threading on the latency of port inquiry sweep with 1 second query interval on the OSC clusters. OSC has 3,402 active ports

Impact of multi-threading on latency of port inquiry sweep with 1 second querying interval for RI cluster. RI has 542 active port

The latency of port inquiry sweep for 35,000 samples for OSC cluster with query interval of 250ms and using 16 thread

- Ability to remotely gather InfiniBand performance counters at sub-second granularity for very large (>2,000 nodes) clusters

MPI_T and Visualization Overhead

- Overhead of PVAR Collection – per MPI Process
  - Query interval of 1s

Very Low Overhead for Query Phase

OVERHEAD OF COLLECTING PVAR DATA AT NANOSECOND GRANULARITY

Metrics	Average	Min	Max	STDDEV.p
Collecting PVARs	517.63 ns	140 ns	16,204 ns	305.91 ns

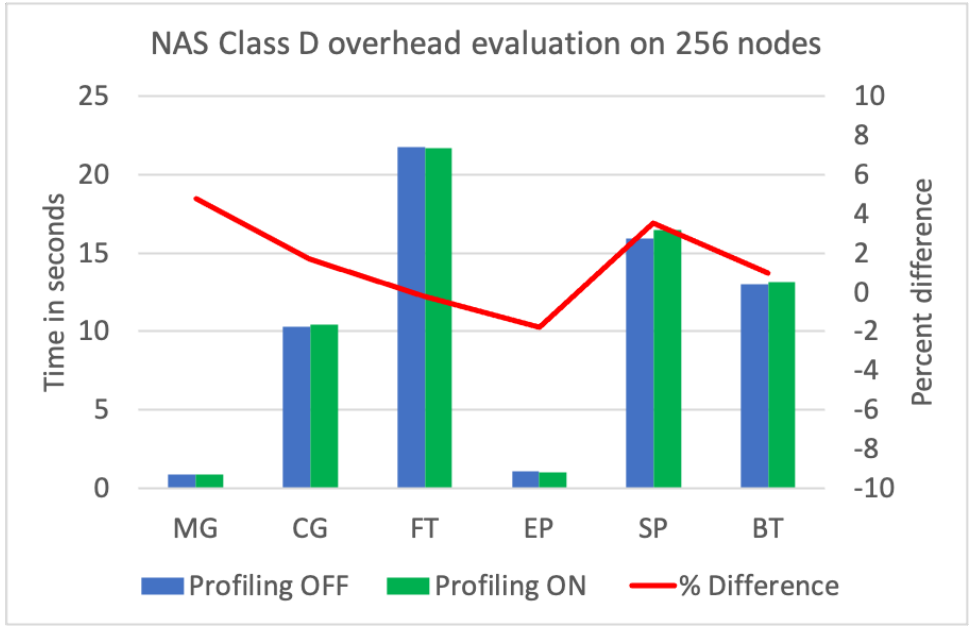
- Overhead of Visualization and Rendering
  - Evaluating the timing of accessing and visualizing on web UI

NETWORK AND LIVE JOBS VIEW GENERATION TIMING ON OSC WITH 1K JOBS

View	Average	Min	Max	STDEV.p
Network View	196.15 ms	187 ms	206.09 ms	5.75 ms
Live Jobs View	18.17 ms	16 ms	20 ms	1 ms

Application-Level Analysis and Session Scaling

- Introspect performance of different NAS Class D benchmarks with and without MPI profiling (PVARs) with an interval of 5 seconds for 4,096 processes across 256 nodes
- Numbers are average of 25 integrations
- There is less than 5% performance degradation.



- Network overhead of adding additionall profiling sessions running on 3DStencil application
  - Compared to disabled MPI_T mode

#Sessions	Bytes sent for profiling	Bytes sent by application	Overhead
1	165 KB	306 MB	0.05%
4	390 KB	306 MB	0.12%
8	689 KB	306 MB	0.22%

Outline

- Overview
- OSU INAM Components
- Overhead Analysis
- **Downloading, Installing, and Configuring OSU INAM**
  - **Downloading and Architecture Support**
  - **Deciding on Intervals for Query Data from HPC Communication Stack**
  - **Storage Data Modeling**
  - **OSU INAM Configuration**
  - **Best Practices for Resource Allocation among INAM Components**
  - **Configuration for Profiling MPI Jobs**
  - **Running Example MVAPICH2-X Jobs with INAM**
- Demos
- Future Work and Concluding Remarks

Downloading and Architecture Support

- OSU INAM is available at http://mvapich.cse.ohio-state.edu/downloads/

OSU INAM

OSU InfiniBand Network Analysis and Monitoring (OSU INAM) Tool v0.9.6 (06/08/20)

- The OSU INAM package is distributed under the BSD License .
- A detailed user guide with instructions to build, install and run OSU INAM is available here. This document also contains guidelines for troubleshooting and best practice deployment.
- Please see CHANGES for the full changelog.
- To estimate the expected size of the database on your system, please see the Database Size Calculator section.
- If you need OSU INAM for specific architecture and OFED version please email us at mvapich-help@cse.ohio-state.edu .
- These RPMs contain the OSU-INAM software on the corresponding distro. Please note that the RHEL RPMs are compatible with CentOS as well. For Debian/Ubuntu users, please follow the instructions in the install section in the userguide.

Environment	Download link
x86_64 MOFED 4.5	Download EL7 MOFED 4.5 RPM (RHEL7/CentOS7)
x86_64 MOFED 4.6 & MOFED 4.7	Download EL7 MOFED 4.6 & 4.7 RPM (RHEL7/CentOS7)
x86_64 MOFED 5.0	Download EL7 MOFED 5.0 RPM (RHEL7/CentOS7)
ARM MOFED 4.5	Download ARM EL7 MOFED 4.5 RPM (RHEL7/CentOS7)

Please note that, all RPMs were built using GNU (GCC) 4.8.5

Modeling of Aggregated InfiniBand Data on the Network and Storage

- How do we identify the amount of traffic added to the network from our tool?

$$Traffic_{(tool)} = Traffic_{(PC\_total)} + Traffic_{(FB\_total)}$$

$$Traffic_{(PC\_total)} = 200Bytes \times QueryFrequency \times NumberofSwitches \times NumberofSwitchPorts$$

- For fabric discovery, it depends on the topology of the network and the diameter of the network

$$Traffic_{(FB\_total)} = NetworkDiameter \times 20Bytes \times NumberofSwitchPorts \times QueryFrequency \times NumberofSwitches$$

- Considering OSC (114 switches and 1,428 compute nodes connected through 3,402 links)
  - Port Inquiry traffic will be 1,603 KB/sec with interval of half second
  - fabric discovery traffic will be 801 KB/sec when using interval of 1 second for updating fabric
  - Given that the OSC clusters are capable of 56 Gbps to 100 Gbps, this is an insignificant amount of data.

Modeling of MPI_T Data Storage Requirement

- For profiling of MPI Performance Variables (PVARs) in MPI jobs:

$$NumProcs = NumNodes \times ProcessesPerNode \times AverageClusterLoad$$

$$Traffic_{PVARs} = NumPVAR(300) \times RecSize_{PVAR}(255B) \times$$

$$NumProcs \times (1 + Sessions) \times Frequency$$

- For CPU, Memory and basic processor information we have:

$$Traffic_{ProcessInfo} = NumProcs * RecSize(336B) * Frequency$$

- Considering the full-load for cluster size of 2,000 nodes with 28 processes per node and 1Hz profiling query frequency
  - Assuming somehow one could activate all PVAR records per MPI process
  - Maximum PVARs traffic/storage will be 4.28 GB/s per profiling session on cluster level
  - 18.8 MB/s for MPI process information on cluster level

OSU INAM Configuration

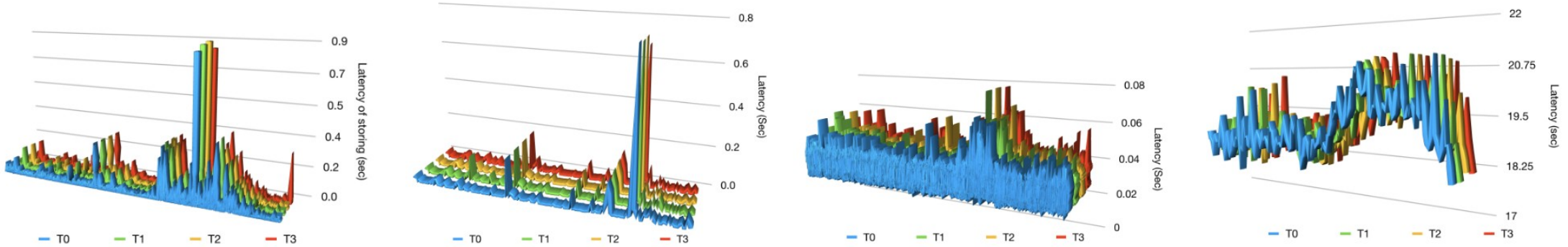
- OSU INAM Daemon configuration specifies resource allocation for components of Daemon
  - Running only on one node!
  - OSU INAM Daemon configuration file is located at $OSU_INAM_INSTALL_PREFIX/etc/osu-inamd.conf
- MySQL configuration can be optimized to use caching based on cluster size
  - See user guide: http://mvapich.cse.ohio-state.edu/userguide/osu-inam/#_mysql_tuning_parameters
  - MySQL configuration file is the same as default located at /etc/my.conf
- OSU INAM web is configurable for access and visualization setting
  - User Authentication, Read-Interval for updating visualizations, etc.
  - OSU INAM Web configuration file is located at /etc/osu-inam.properties.

Best Practice for OSC INAM and Thread Load balancing

- What is the proper allocation of number of thread based on number of CPU cores for each module inside OSU INAM Daemon?

Cluster size	fabric discovery	performance port inquiry	MPI_T and job thread	Purge thread
< 500	2	2+	1	2
500< size <1000	4	8+	1	2
> 1000	8	16+	2	2

- Load Balancing of Threads in Port Inquiry



(a) Timing of write phase for each thread for fabric discovery (b) Timing of write phase for each thread for port inquiry (c) Timing of query phase for each thread for port inquiry (d) Timing of query phase for each thread for fabric discovery

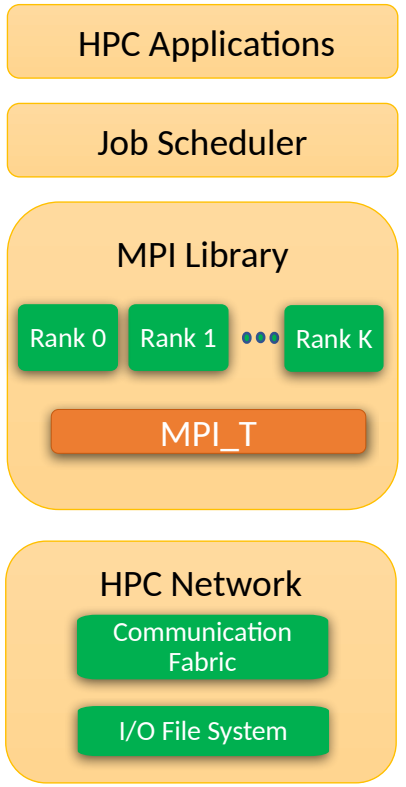
Configuration for Profiling MPI Jobs

- OSU INAM Daemon generates a configuration file that users need to pass to MPI jobs to send the information to OSU INAM Daemon
- This file is located in /tmp/mv2_mpit.conf or $OSU_INAM_INSTALL_PREFIX/etc/osu-inam.conf
- Query intervals passed to the jobs are user-defined variables and can be edited in this file
- Unless specified they are system-defaults as specified in daemon config file

```
(base) kousha.2@head:~$ cat inam.conf
MV2_TOOL_QPN=190564
MV2_TOOL_LID=10
MV2_TOOL_COUNTER_INTERVAL=30
MV2_TOOL_REPORT_CPU_UTIL=1
MV2_TOOL_REPORT_MEM_UTIL=1
MV2_TOOL_REPORT_IO_UTIL=1
MV2_TOOL_REPORT_COMM_GRID=1
MV2_TOOL_REPORT_LUSTRE_STATS=0
MV2_TOOL_REPORT_PVARS=1
(base) kousha.2@head:~$
```

Deciding on Intervals for Query Data from HPC Communication Stack

- Deciding on Query interval is a key factor for profiling jobs
- Ideal Query interval should be the same across stacks
- Selecting Interval has a trade-off
  - Low intervals (1-5 s) of query result in massive amount of profiling data
    - Fine grained profiling
    - Results in slower database
    - Delay in web interface
  - Large (>60s)of query interval results in hiding concurrency bugs across layers or missing information
    - Faster web interface specifically for historical jobs
    - coarse-grained profiling
- You will explore this as part of Demo 0!



Running Example MVAPICH2-X Jobs with INAM

- **MVAPICH2 Running Example without Performance Variables (PVARS)**

```
$ mpirun_rsh -rsh -np 4 n0 n0 n1 n1 MV2_ON_DEMAND_THRESHOLD=1
MV2_TOOL_INFO_FILE_PATH=inam.conf MV2_TWO_LEVEL_COMM_THRESHOLD=1
MV2_USE_RDMA_CM=0 ./test
```

- **MVAPICH2 Running Example with Performance Variables (PVARS)**

```
$ mpirun_rsh -rsh -np 4 n0 n0 n1 n1 MV2_ON_DEMAND_THRESHOLD=1
MV2_TOOL_INFO_FILE_PATH=inam.conf MV2_TWO_LEVEL_COMM_THRESHOLD=1 MV2_USE_RDMA_CM=0
MV2_TOOL_REPORT_PVARS=1 MV2_ENABLE_PVAR_TIMER=1 MV2_ENABLE_PVAR_COUNTER=1
MV2_ENABLE_PVAR_TIMER_BUCKETS=1 MV2_ENABLE_PVAR_COUNTER_BUCKETS=1
MV2_TOOL_REPORT_SESSIONS=1 MV2_TOOL_SESSIONS_DEFAULT_ALL_HANDLES=1 ./test
```

Outline

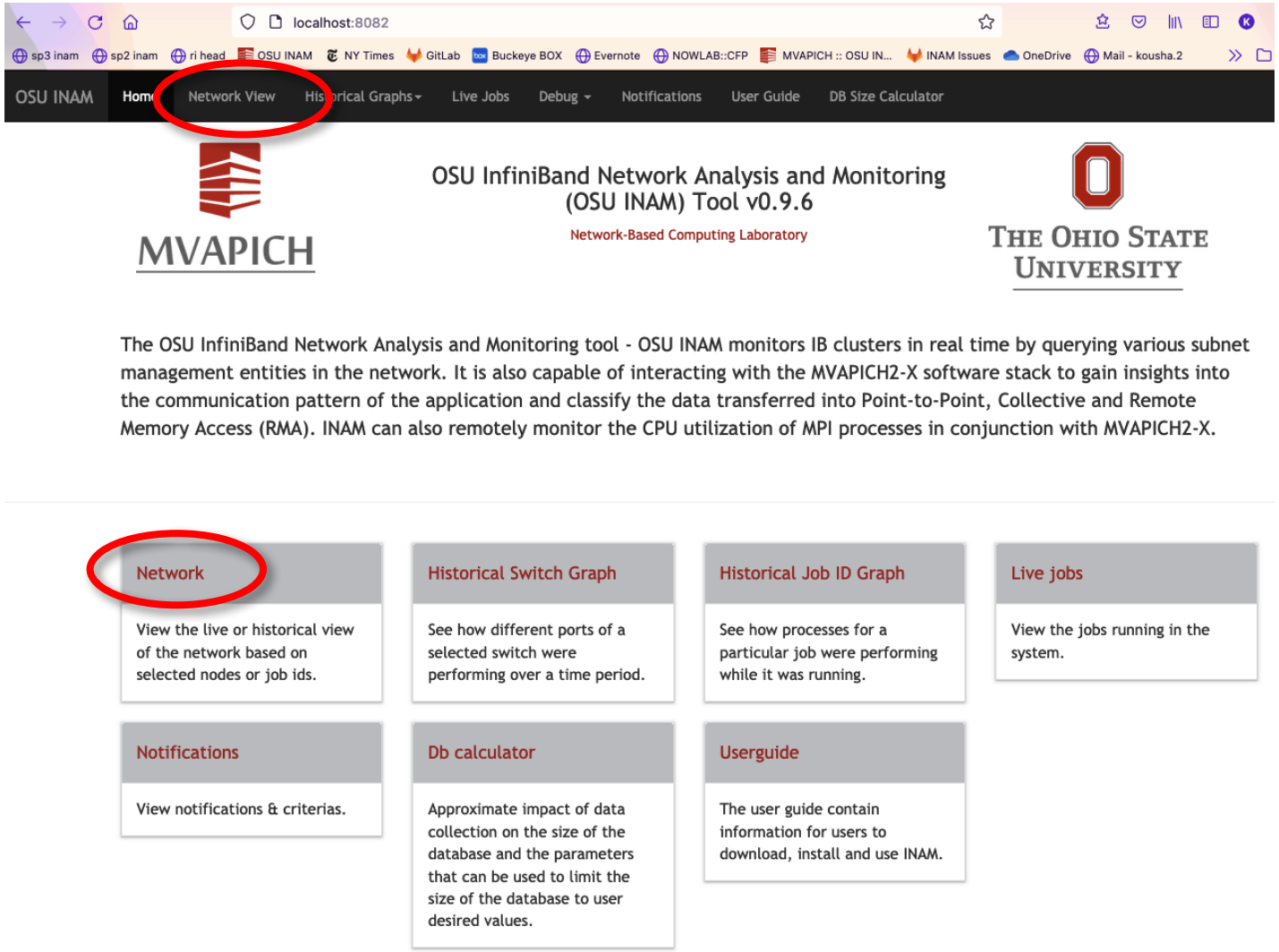
- Overview
- OSU INAM Components
- Overhead Analysis
- Downloading, Installing, and Configuring OSU INAM
- **Demos**
  - **Demo 1: Visualizing Entire Network**
  - **Demo 2: Visualizing Specific Job**
  - **Demo 3: Understanding Live Jobs on Cluster**
  - **Demo 4: Setting up Notifications**
- Future Work and Concluding Remarks

Demo 1 - Visualizing Entire Network

- Objectives
  - How to visualize network traffic at system level?
  - How to find congested links in a cluster?
  - How to view network traffic flow for historical jobs in a cluster?
- Tasks:
  - System level network traffic visualization
  - Viewing live node information
  - Finding nodes in the cluster
  - Finding most used links
  - Historical replay of network traffic

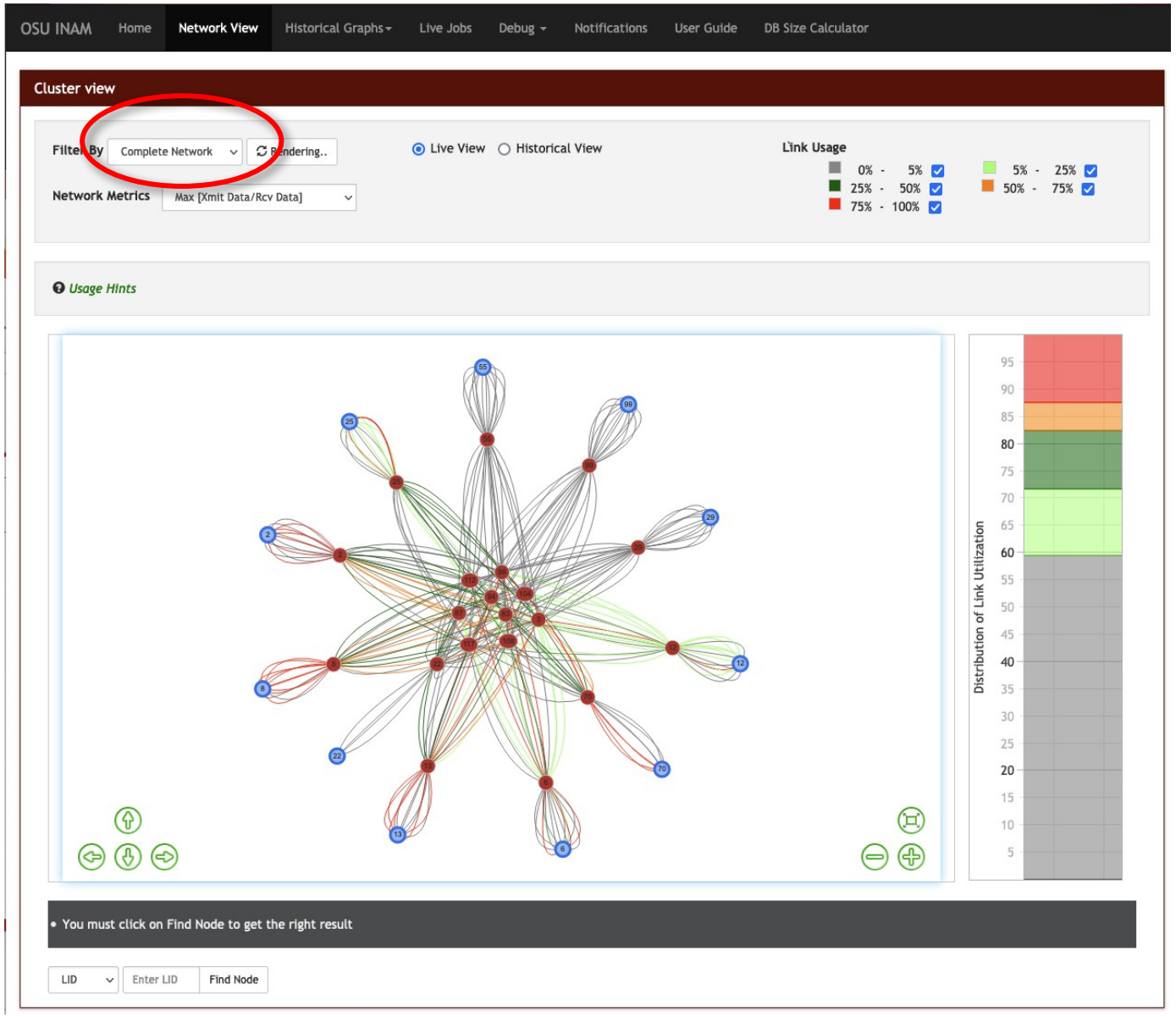
Demo 1: Visualizing Entire Network

- Visualize the entire network
  - Represents a logical layout of the HPC network
  - Need not show how the compute nodes and switches are physically laid out on the data center
  - Use physics-based visualization package for better layout
  - Use advanced caching schemes to accelerate visualization time for very large clusters



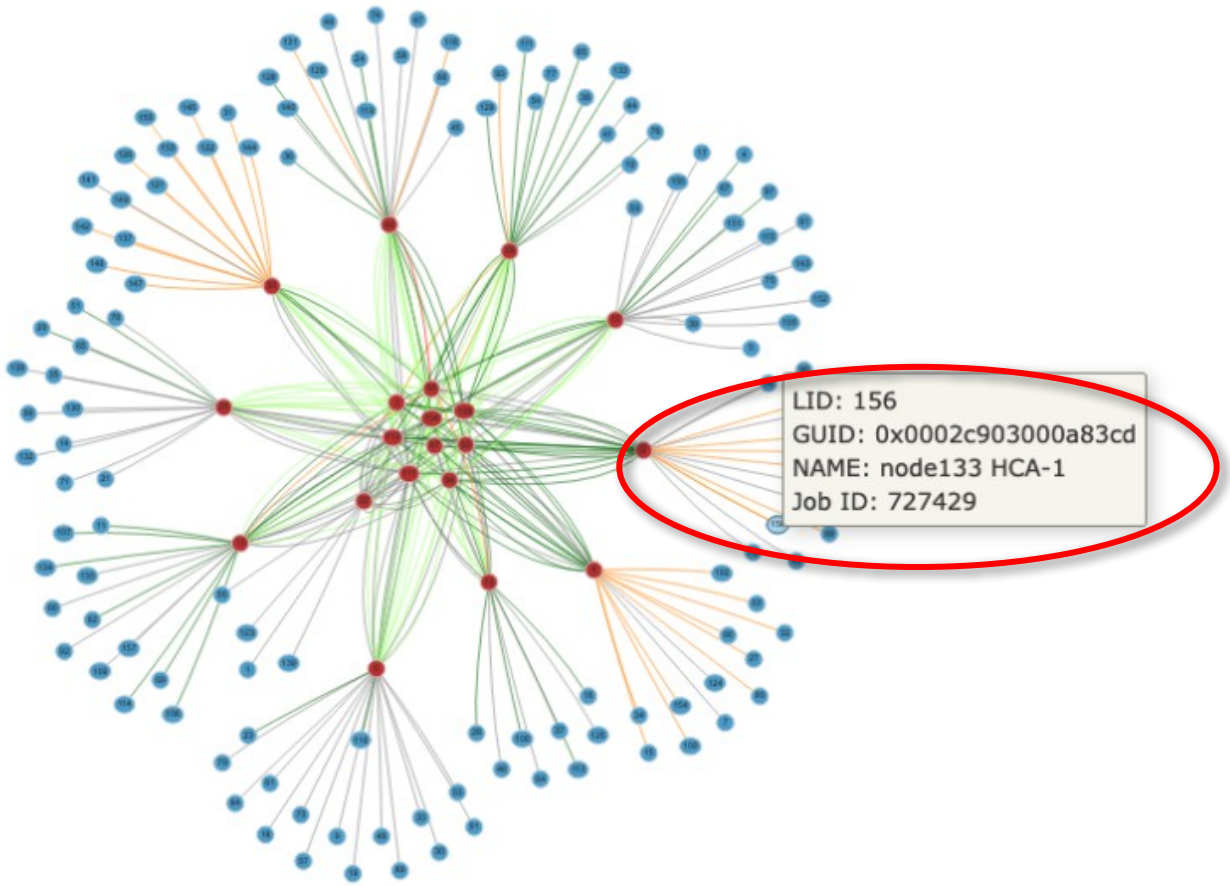
Demo 1: Visualizing Entire Network – System Level Visualization

- Visualize the entire network
  - Contains two types of nodes
    - Switches (Red)
    - Compute Nodes (Blue)
  - Links represent physical IB connectivity between switches and compute nodes
  - Color on the links represent link usage
    - Grey (0% - 5%)
    - Light Green (5% - 25%)
    - Dark Green (25% - 50%)
    - Orange (50% - 75%)
    - Red (75% - 100%)
  - Link utilization information is obtained in a live fashion from the fabric



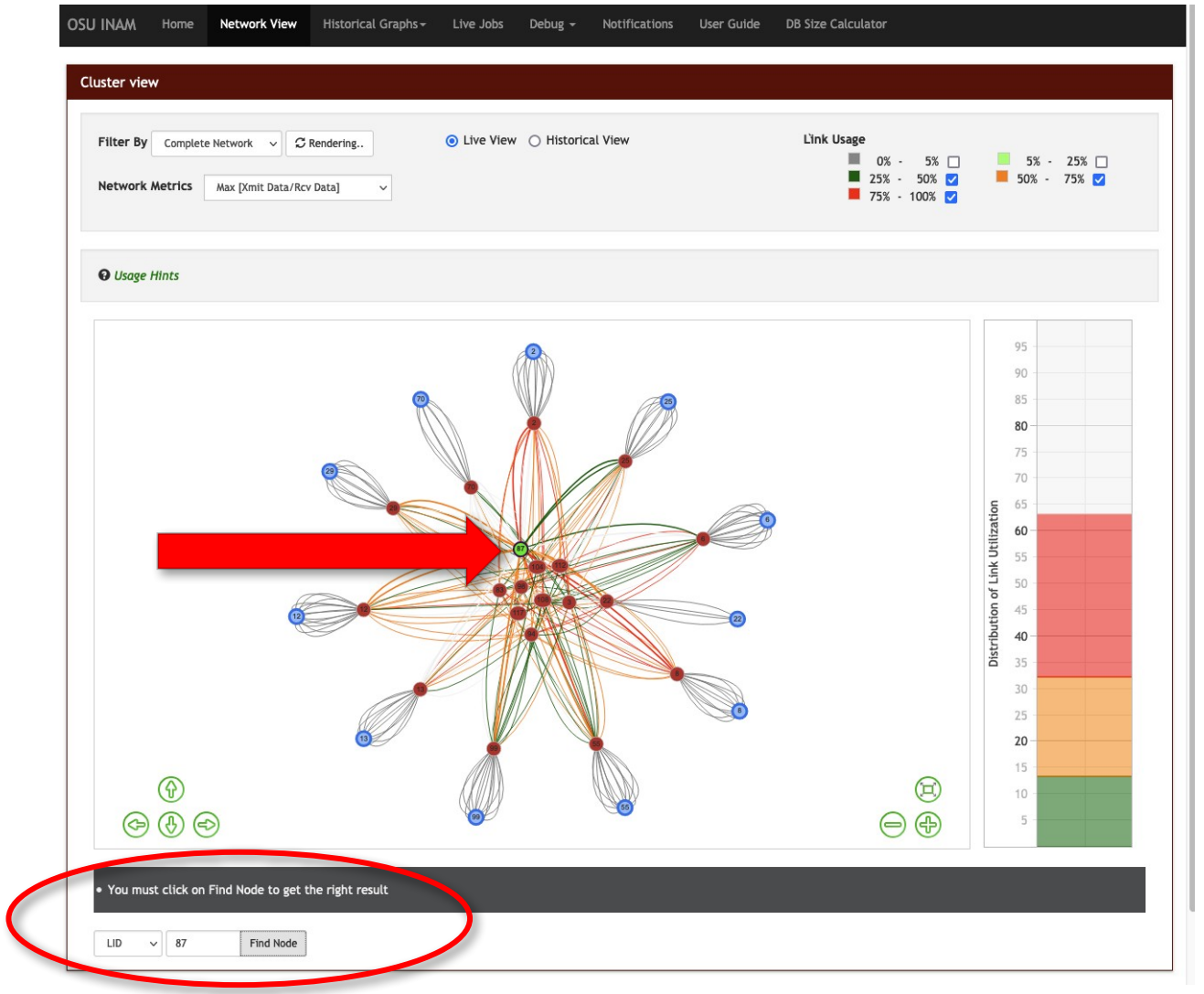
Demo 1: Visualizing Entire Network – View Node Information

- Users can interact with the different elements on the network view
- Hovering over the switches and nodes gives more information about specific element
- Provides information gathered and combined from job scheduler (e.g., Job ID) and IB fabric (e.g., link utilization)



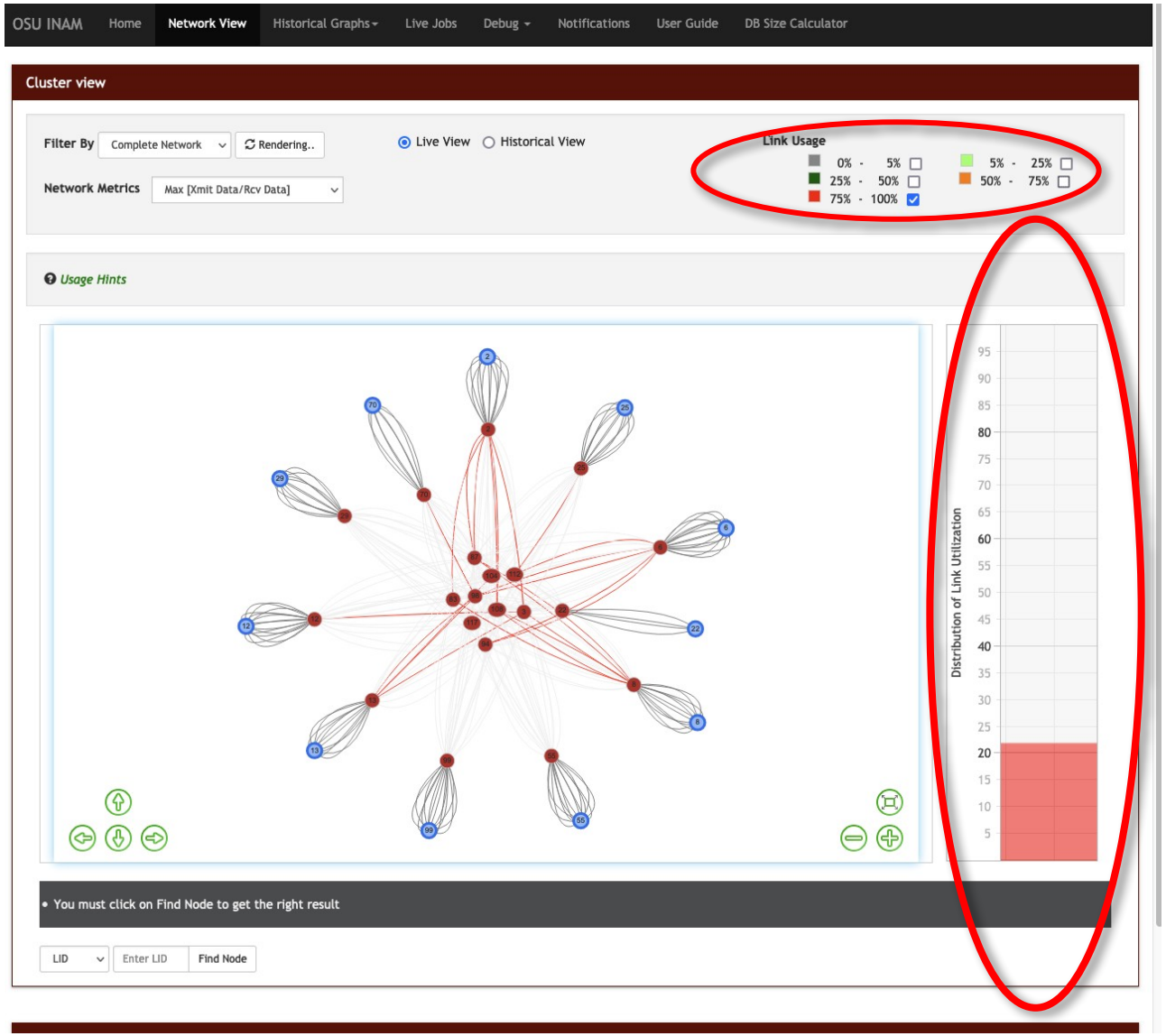
Demo 1: Visualizing Entire Network – Finding Nodes

- To find specific nodes, users can use search bar at bottom
- Start typing the node names, the interface provides suggestions on possible options using data from IB fabric
- Highlights desired node in live network view



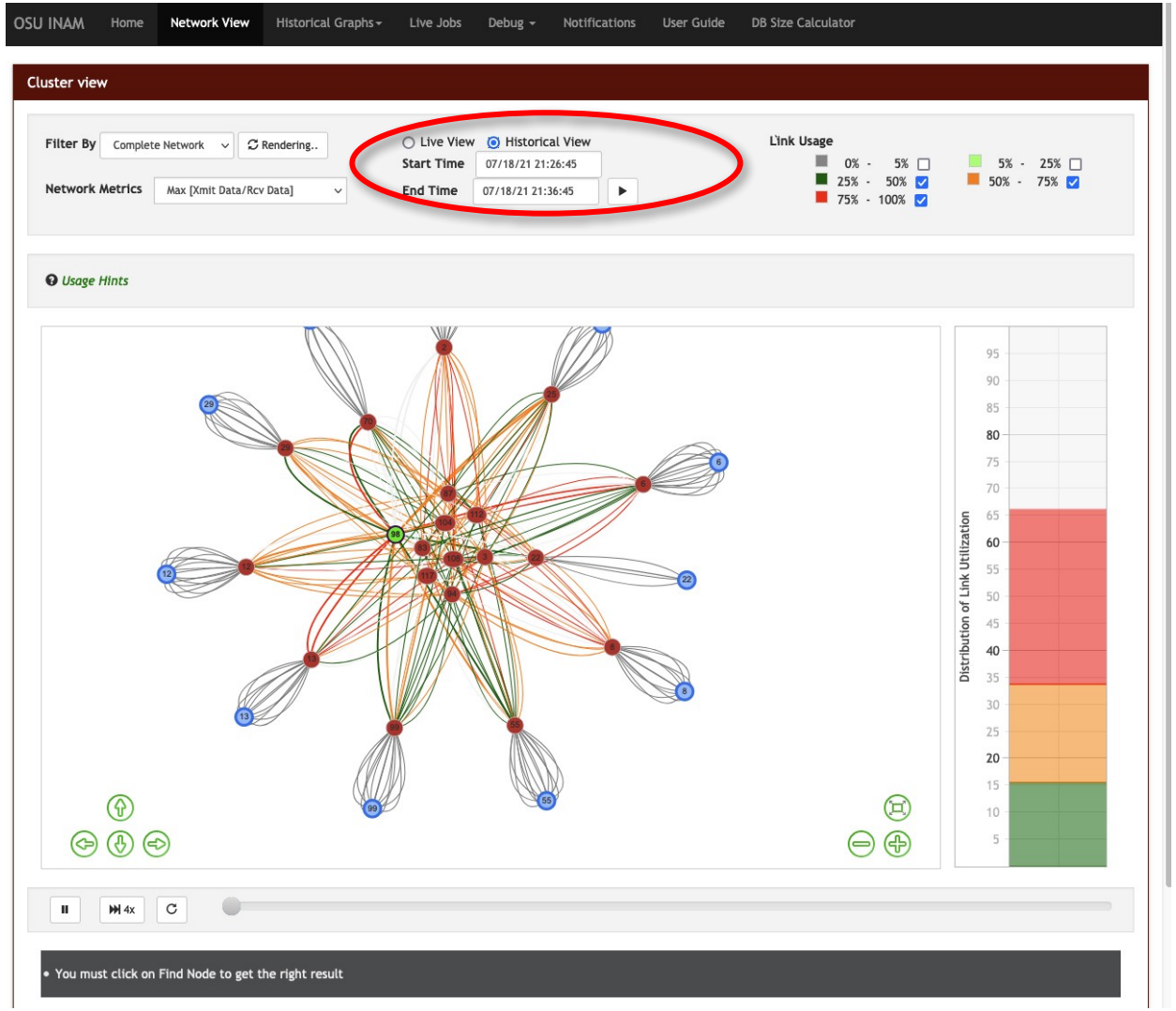
Demo 1: Visualizing Entire Network – Finding Most-used Links

- The IB links can be filtered by selecting/deselecting appropriate values in the “Link Usage” legend
- Provides an easy way to find out/visualize specific categories of links in the user interface



Demo 1: Visualizing Entire Network – Historical Replay

- OSU INAM provides a live view as well as historical replay feature
- Users can select desired date and time ranges from the user interface
- Data is retrieved from the database and displayed as a moving timeline
- Time taken to render/fetch the data depends on the size of the cluster and amount of data requested
  - Start with a small interval like one hour to get an idea

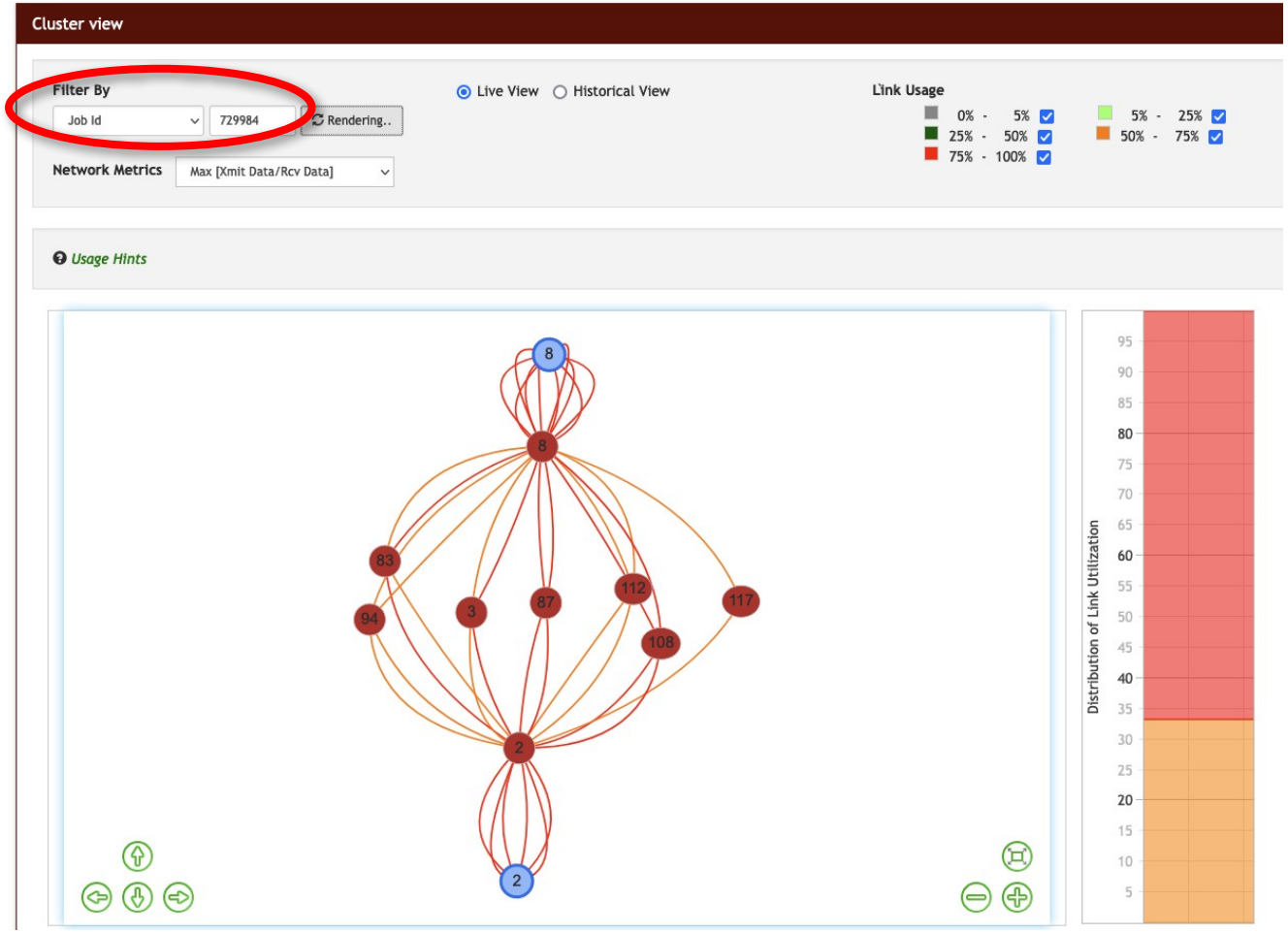


Demo 2 - Visualizing Specific Job

- Objectives
  - How to visualize a specific job's traffic in the cluster?
  - How to find most congested links in a job with respect to system level traffic?
  - How to understand job's topology to explain performance jitters?
- Tasks:
  - Job level network traffic visualization
  - Understanding job topology
  - Finding participating nodes in the job and accessing their information
  - Finding most used links in the job
  - Historical replay of job level network traffic

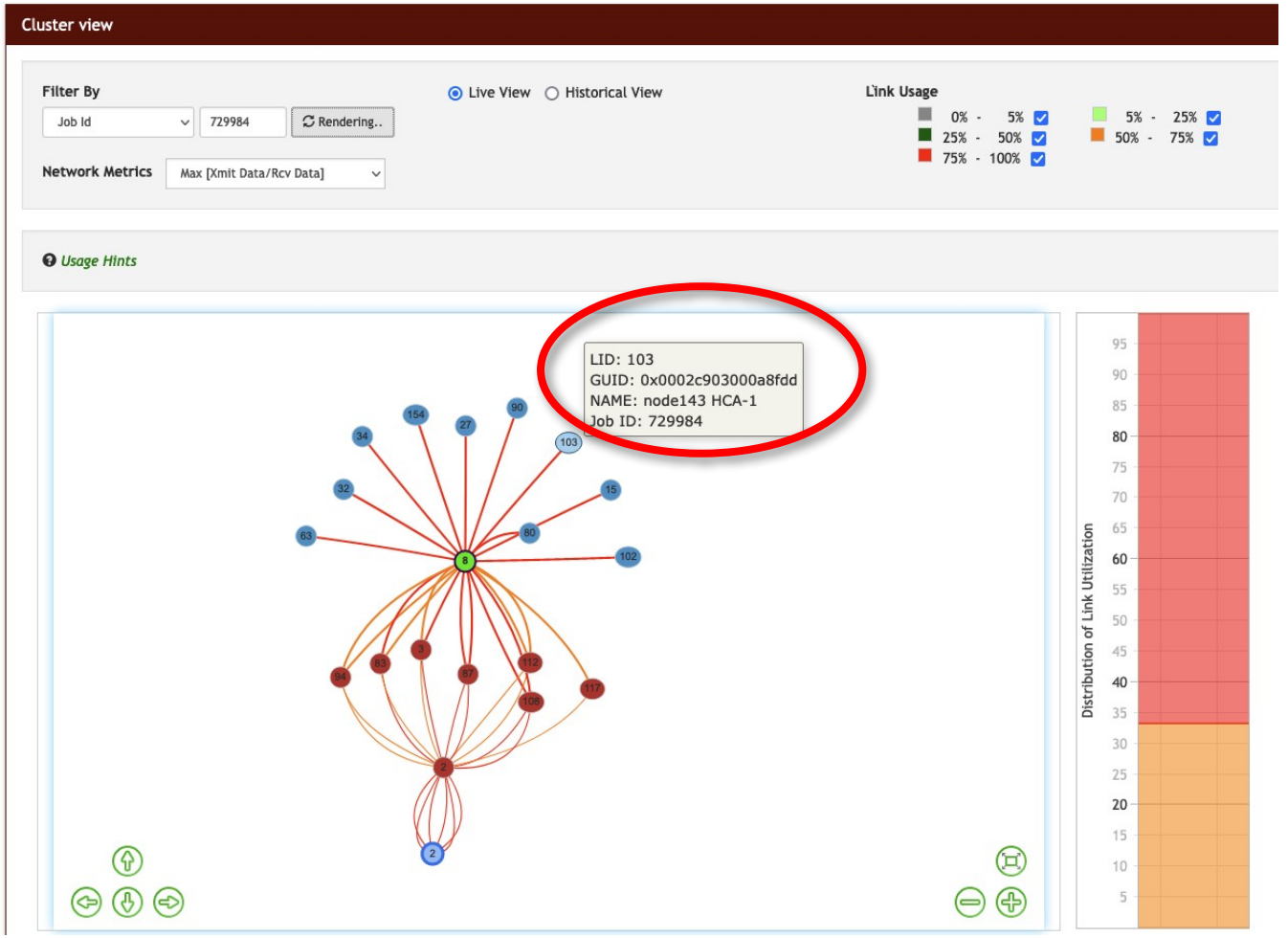
Demo 2: Visualizing Specific Job – Understanding Job Topology

- Live network can be viewed at multiple granularities and in different ways
  - Entire network
  - Nodes and switches/links participating in a specific job
  - User specified nodes and the switches/links interconnecting them
- Job-level view can be obtained by filtering the network elements using the job ID
  - If users do not know the ID of the job, it can be obtained from the “Live Jobs” page
  - More details about the “Live Jobs” page and associated details will be provided in Demo 3



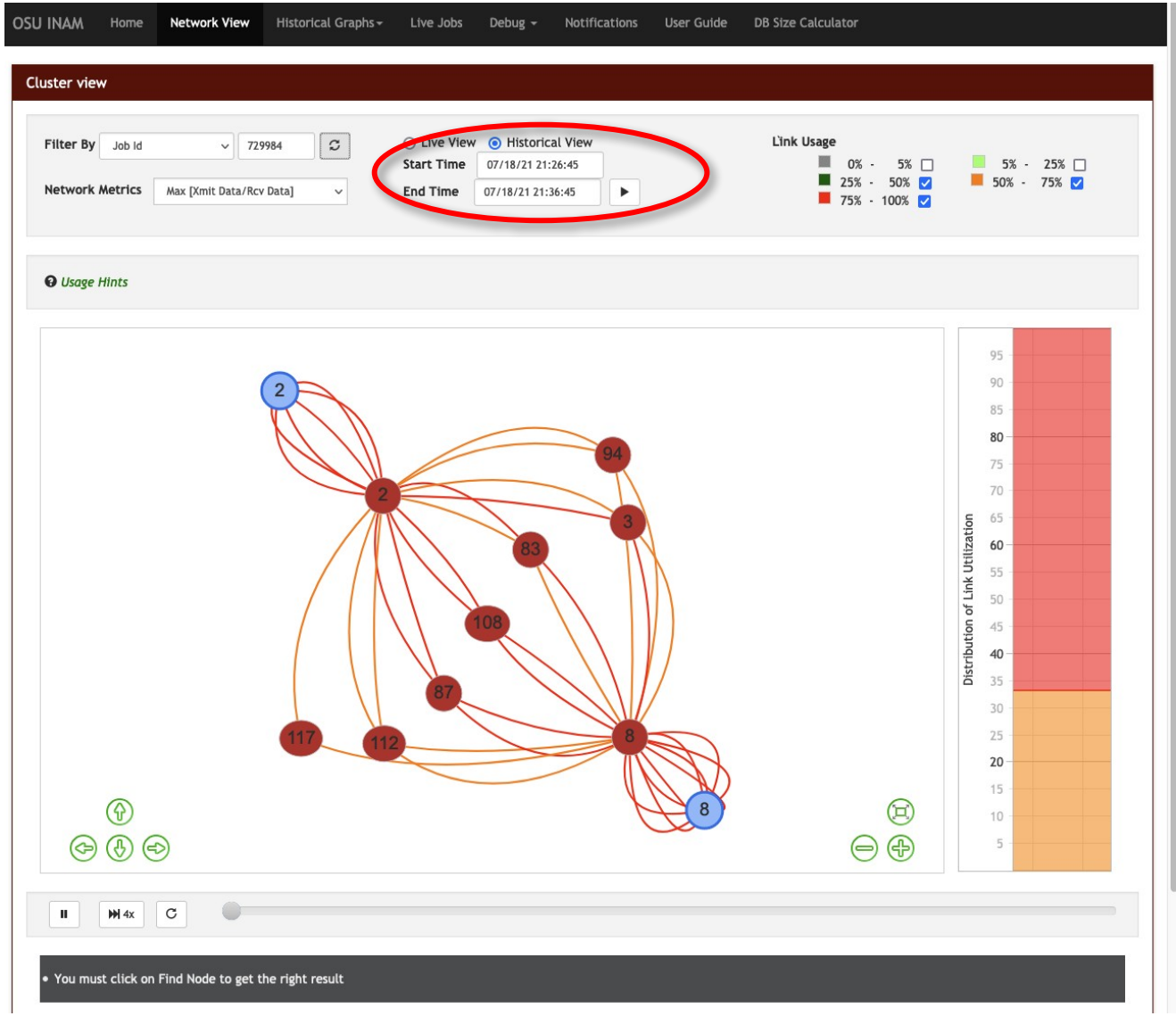
Demo 2: Visualizing Specific Job – Finding Node Information

- Users can interact with different elements in the Job-Level view of the live network page
- Hovering over the node provides details of the node using data collected from the job scheduler (e.g., Job ID) and the IB fabric (e.g., LID, GID, HCA name etc)

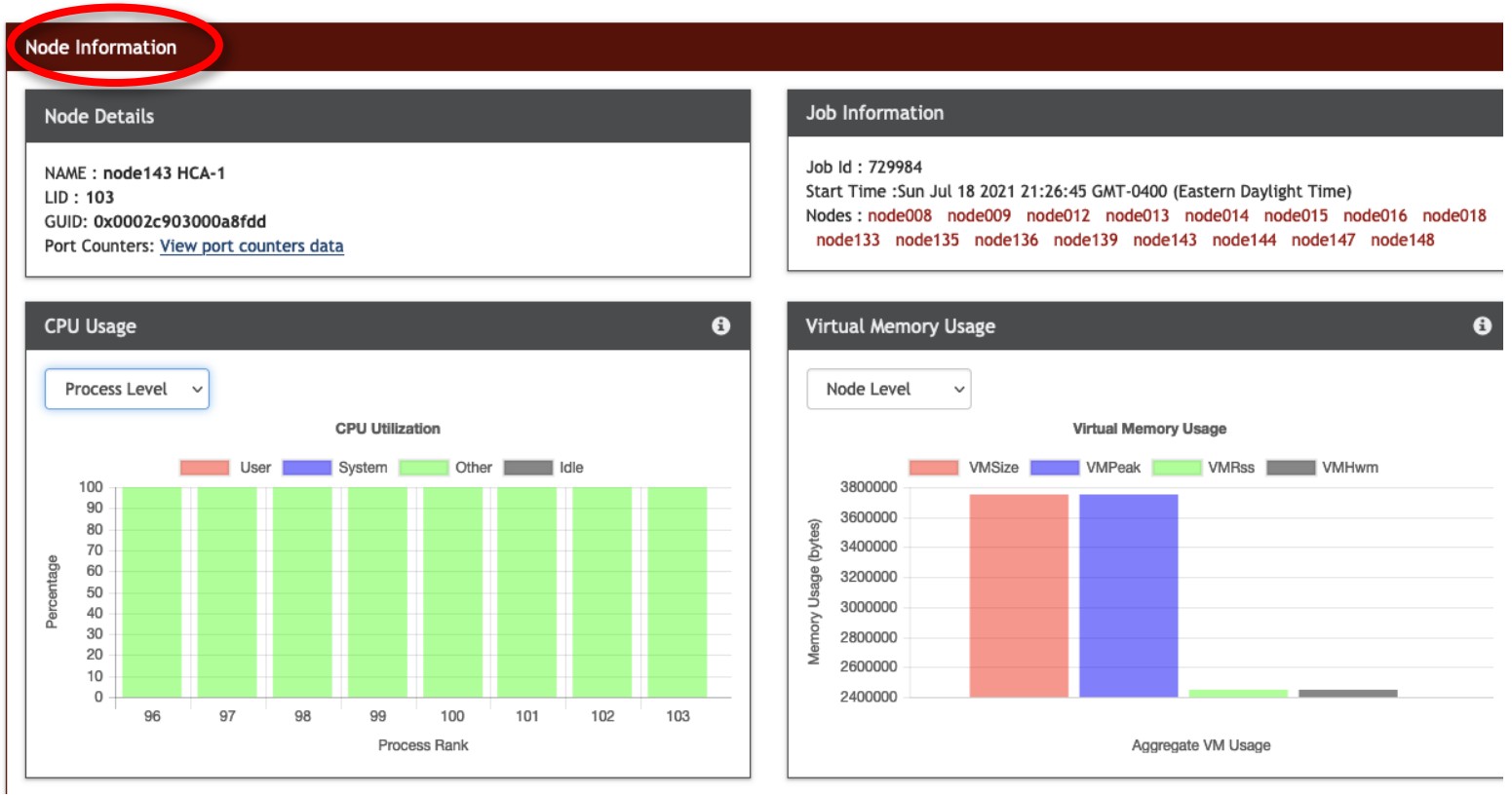


Demo 2: Visualizing Jobs– Historical Replay

- OSU INAM provides historical replay feature at the job-level filter of the live network view
- Users can select desired date and time ranges from the user interface
- Data is retrieved from the database and displayed as a moving timeline
- Time taken to render/fetch the data depends on the size of the job and amount of data requested
  - Start with a small interval like one hour to get an idea



Demo 2: Visualizing Specific Job – Live Node View

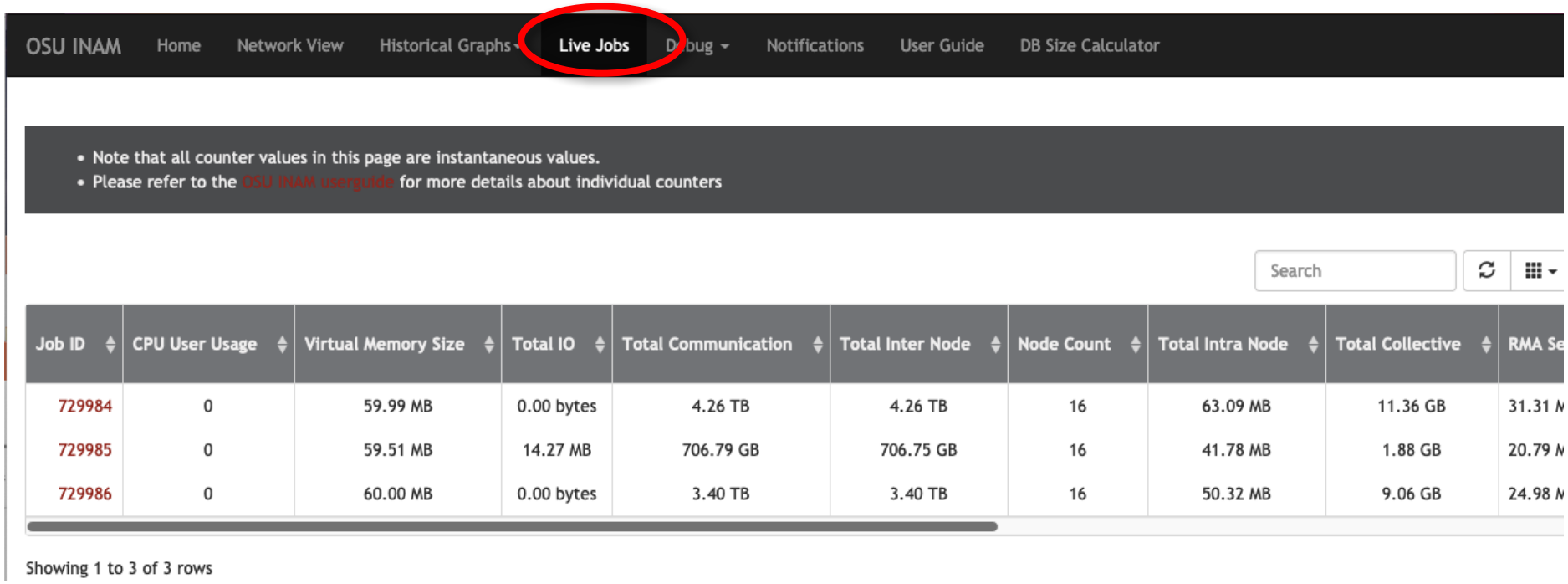


- From the Live Network or Jobs page, users can interact with different elements (switches and nodes) to get more information about them by double clicking on the element
- Double clicking on the node takes one to the live "Node-Level" view
- Uses information from job scheduler, MPI library, and IB fabric as available

Demo 3 – Understanding Live Job on Cluster

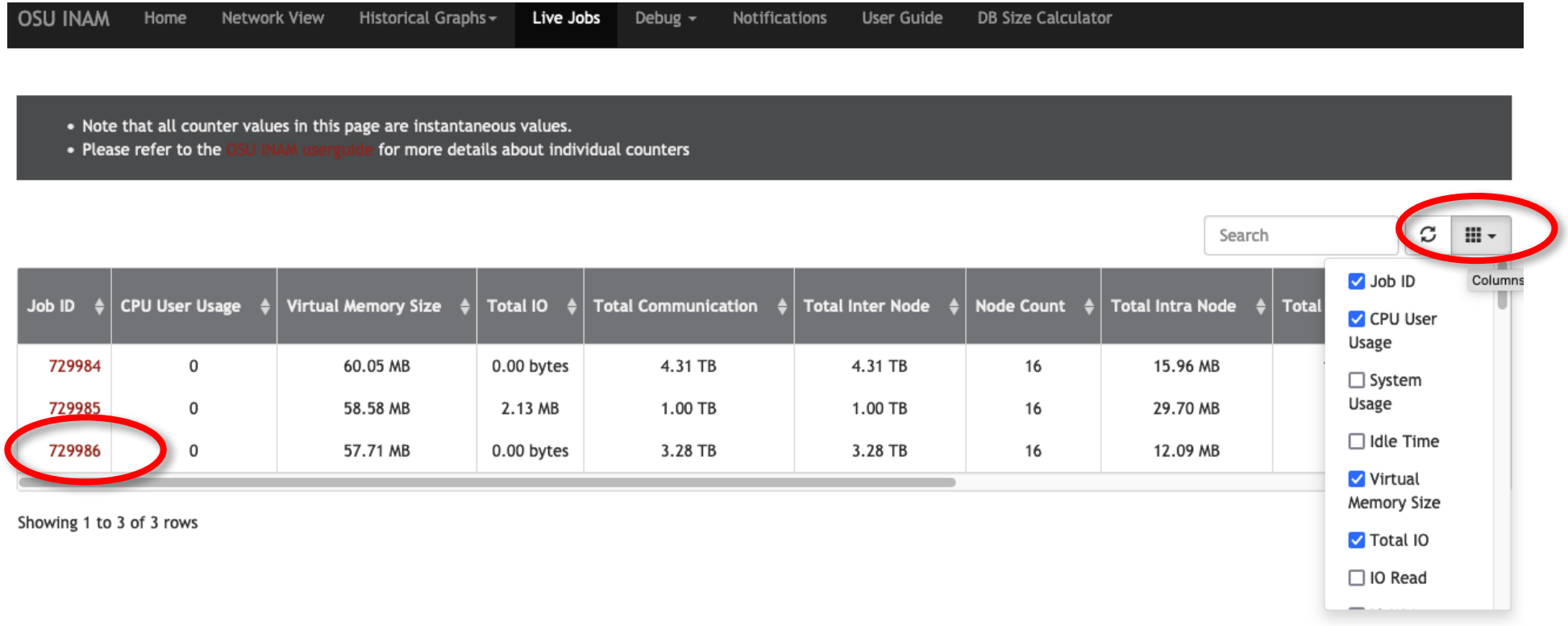
- Objectives
  - How to access running jobs statistics in an easy manner?
  - How to sort live jobs on a cluster based on different metrics?
  - How much traffic does each job generate at system-level?
  - How to view CPU usage, memory usage, MPI information usage and IB level counters for live jobs running in the cluster?
- Tasks:
  - Viewing the list of running jobs
  - Filtering jobs and sorting jobs based on different metrics

Demo 3: Viewing the list of running jobs



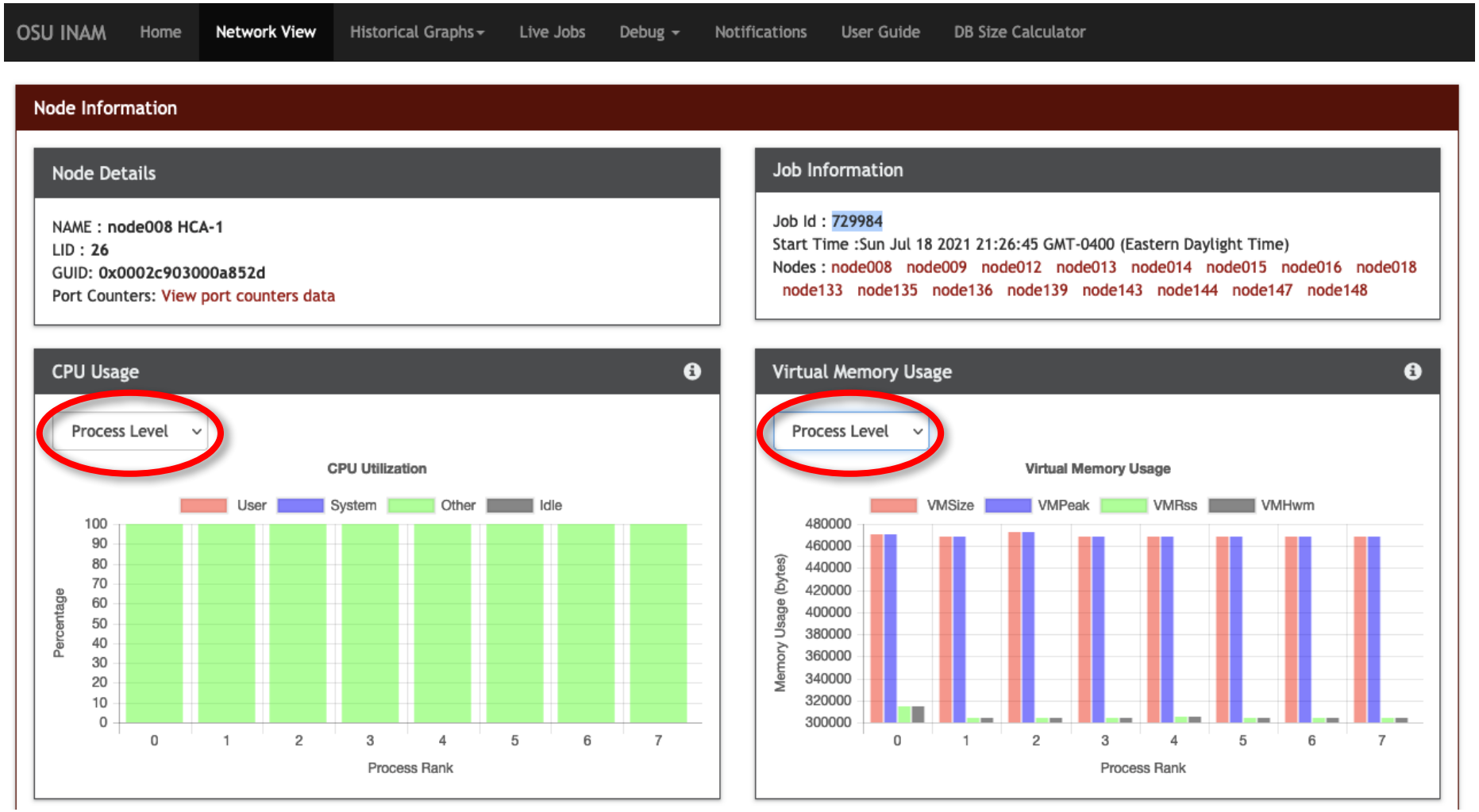
- Users can click on the “Live Jobs” tab to obtain list of all the jobs running currently on the cluster
- Uses information from job scheduler, MPI library, and IB fabric as available

Demo 3: Filtering jobs and sorting jobs based on different metrics



- Different metrics are available, and users can select which metrics to be listed
- Ability to sort table in ascending or descending order based on different metrics

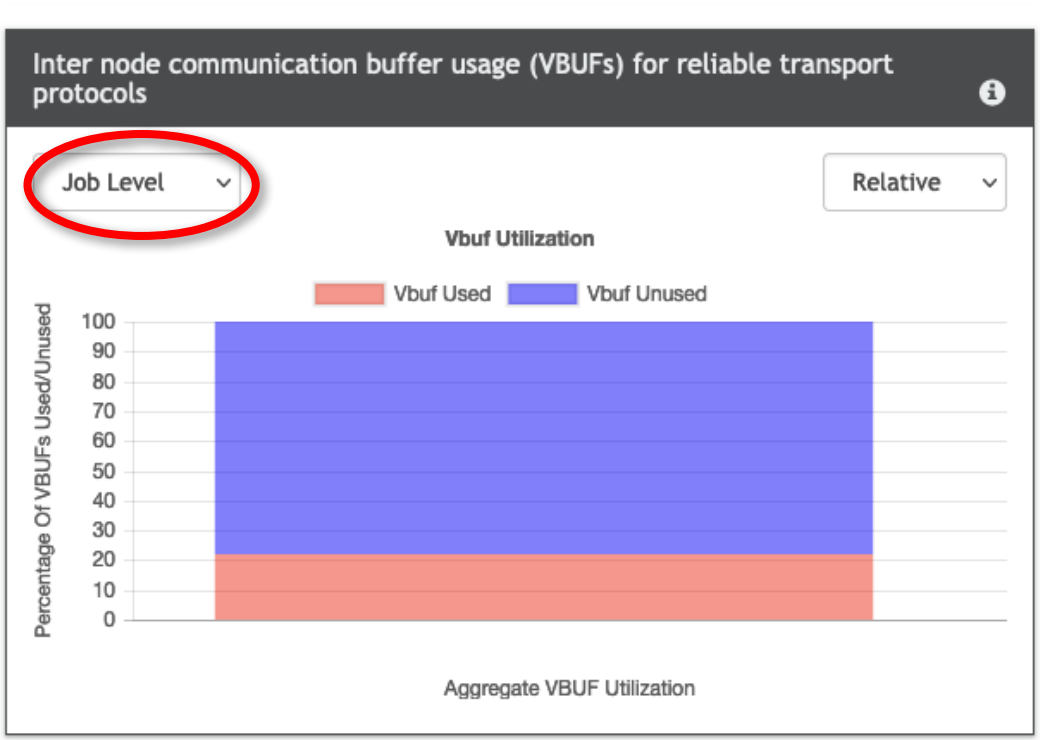
Demo 3: Live Job View – Job Info and CPU/Mem Usage

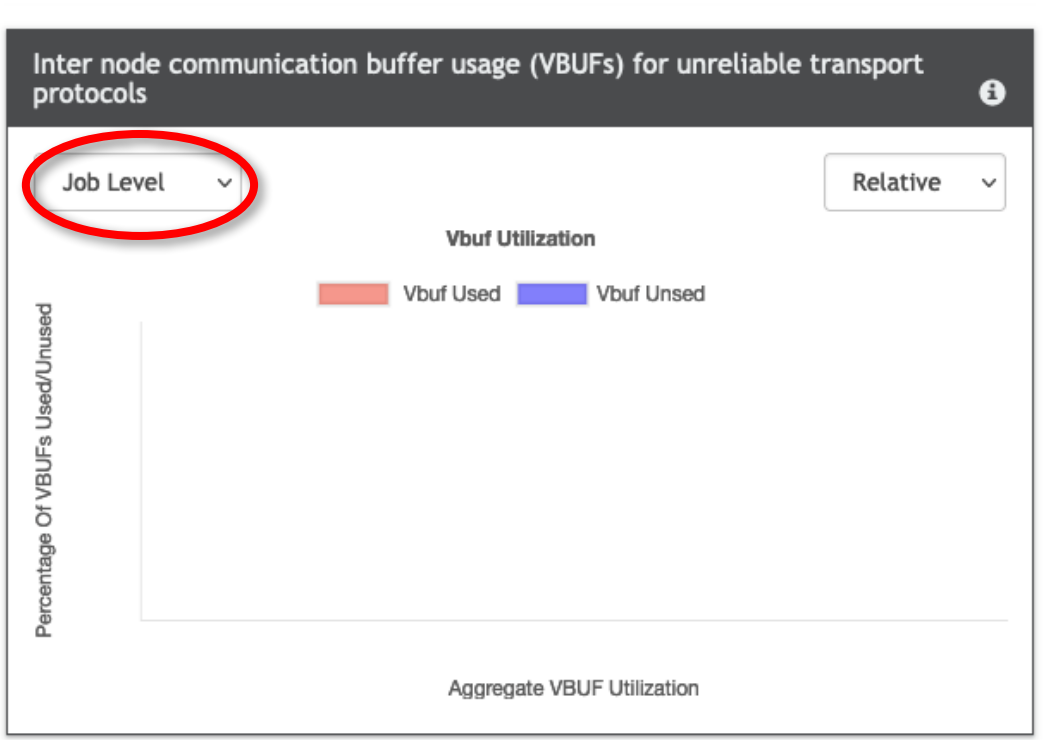


- Users can go to any level of granularity and visualize elements relevant to that level

Demo 3: Live Job View – Vbuf Usage for UD and RC

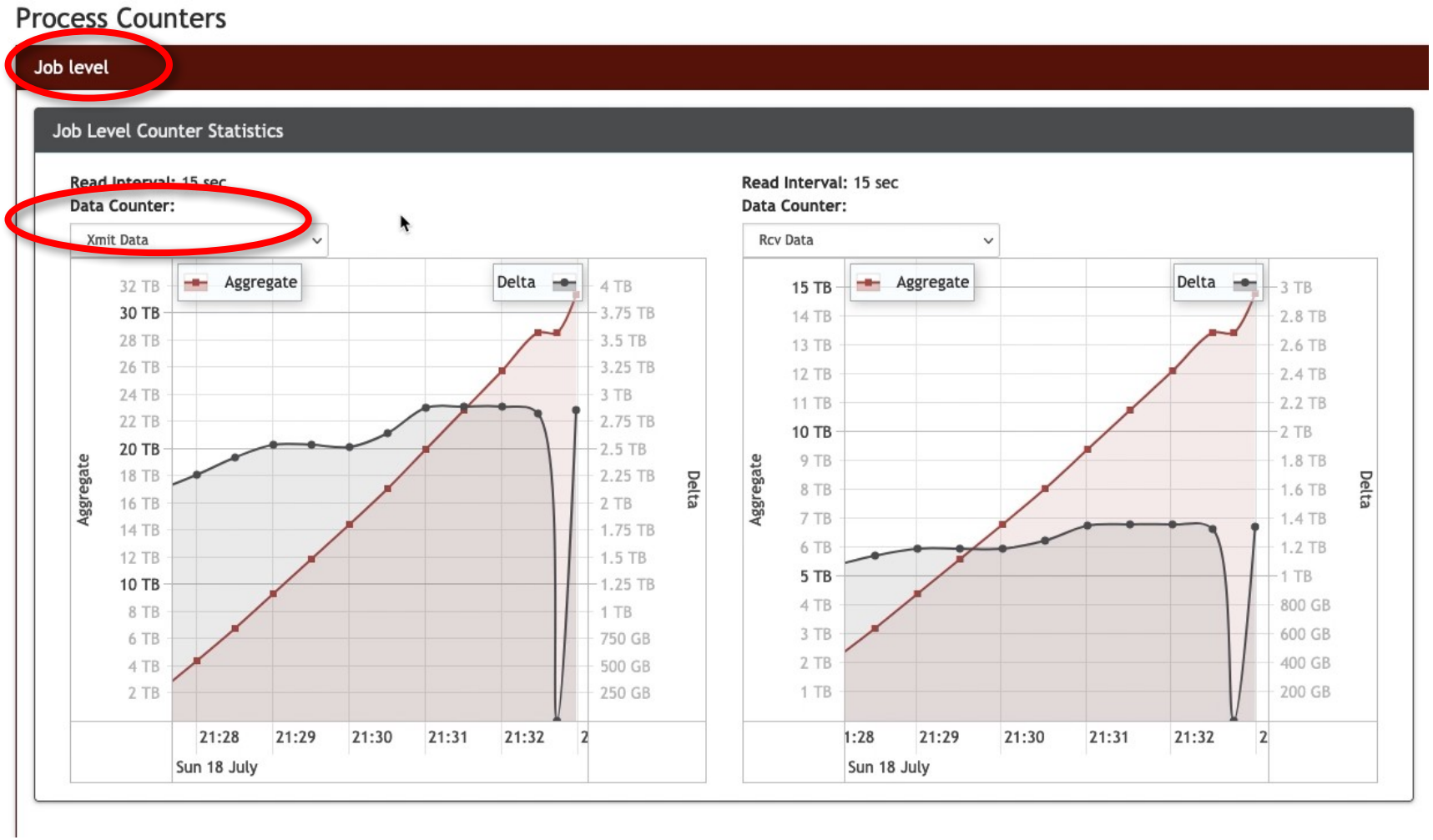
- Users can go to any level of granularity and visualize elements relevant to that level
  - Shows MPI internal communication buffer usage for different transport protocols
    - Supports Reliable Connected (RC) and Unreliable Datagram (UD)





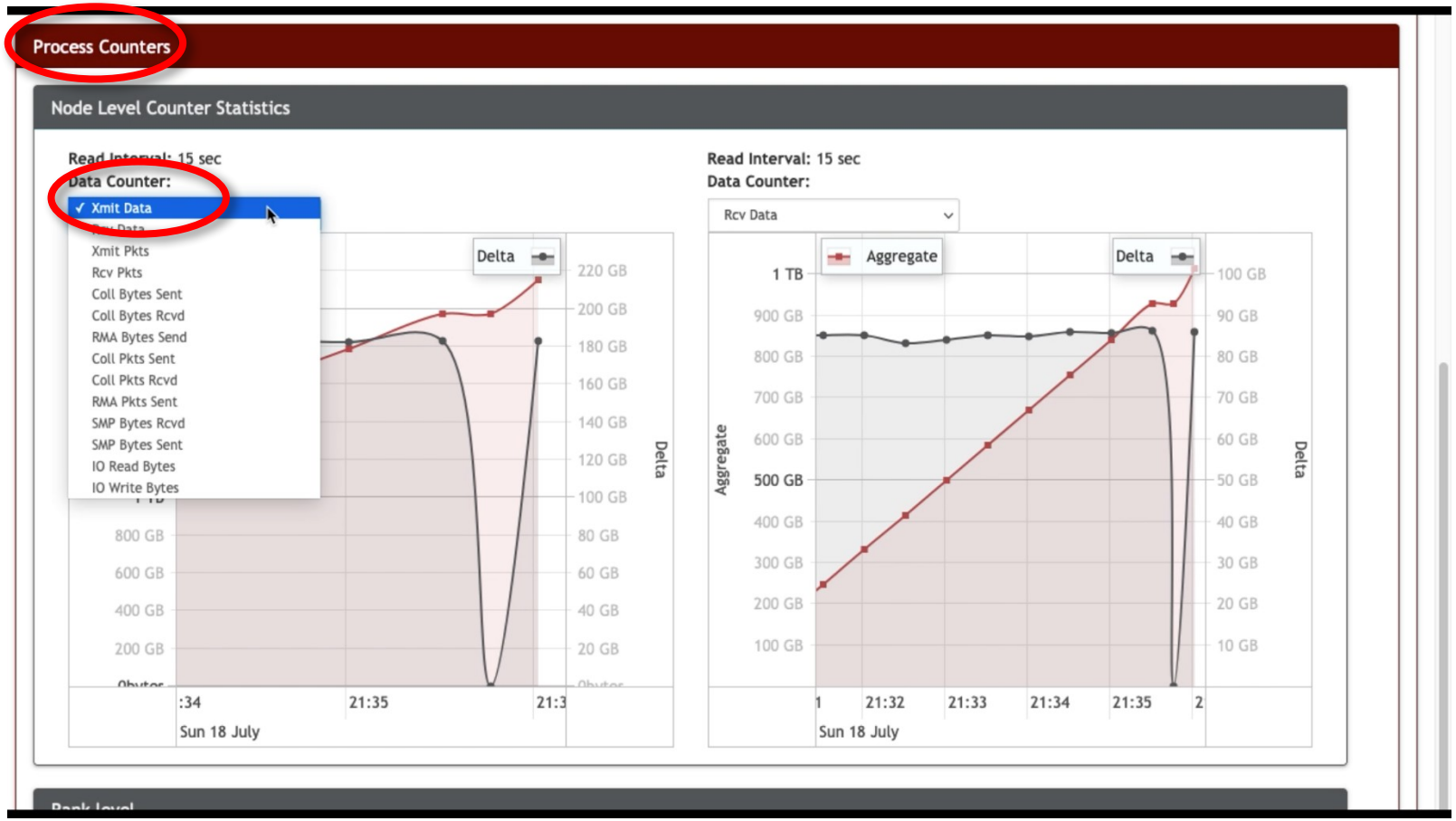
Demo 3: Live Job View – Job level Counters

- Provides counters collated from MPI runtime as well as IB fabrics at different granularities
  - Job Level
  - Node Level
  - Process Level
- Users can select the type of counter from the dropdown box



Demo 3: Live Job View – Node Level Counters

- Provides counters collated from MPI runtime as well as IB fabrics at different granularities
  - Job Level
  - Node Level
  - Process Level
- Users can select the type of counter from the dropdown box

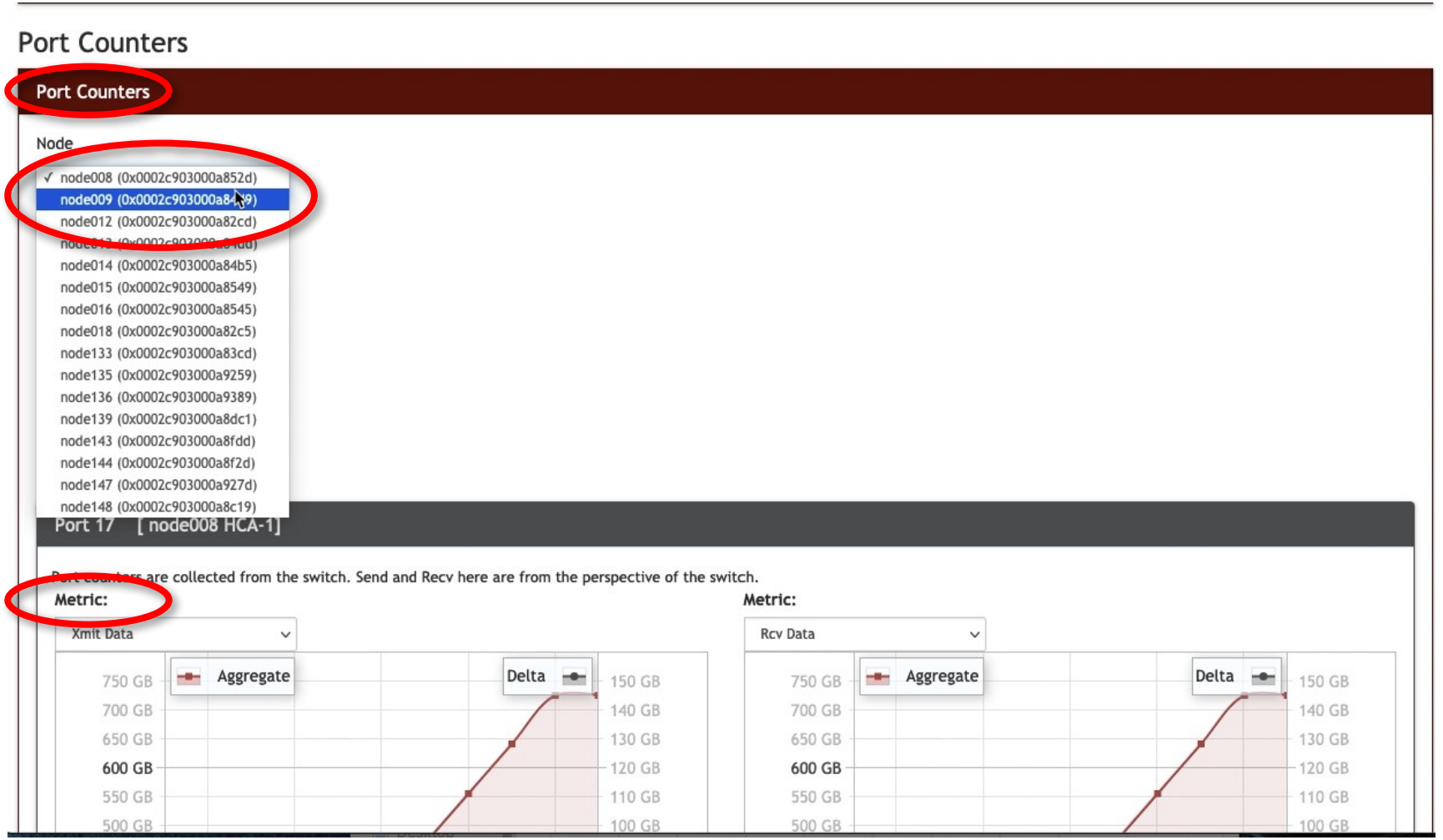


Demo 3: Live Job View – MPI Rank level Counters

- Provides counters collated from MPI runtime as well as IB fabrics at different granularities
  - Job Level
  - Node Level
  - Process Level
- Users can select the type of counter from the dropdown box

Demo 3: Live Job View – InfiniBand Level Counters

- The counters are applicable to the switches as well as the compute nodes
- Only IB level counters provided at switch level

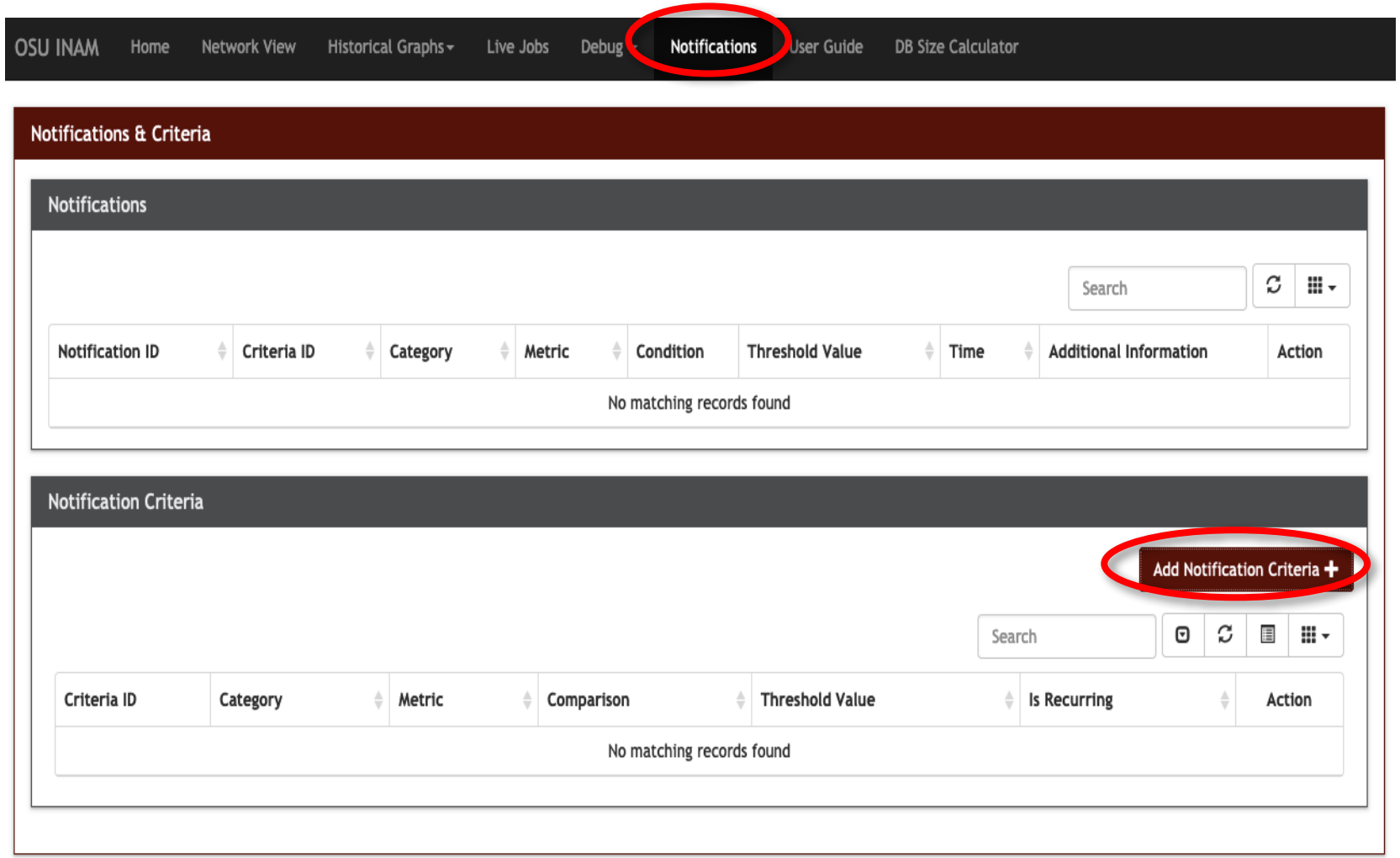


Demo 4 – Performance Notifications

- Objectives:
  - Understand how to set up customized notifications
  - Understand how to manage and delete notifications
  - Understand how to setup recurring and non-recurring notifications
- Tasks:
  - Accessing Notifications
  - Creating Notifications
  - Managing notifications
  - Notifications occurrence view

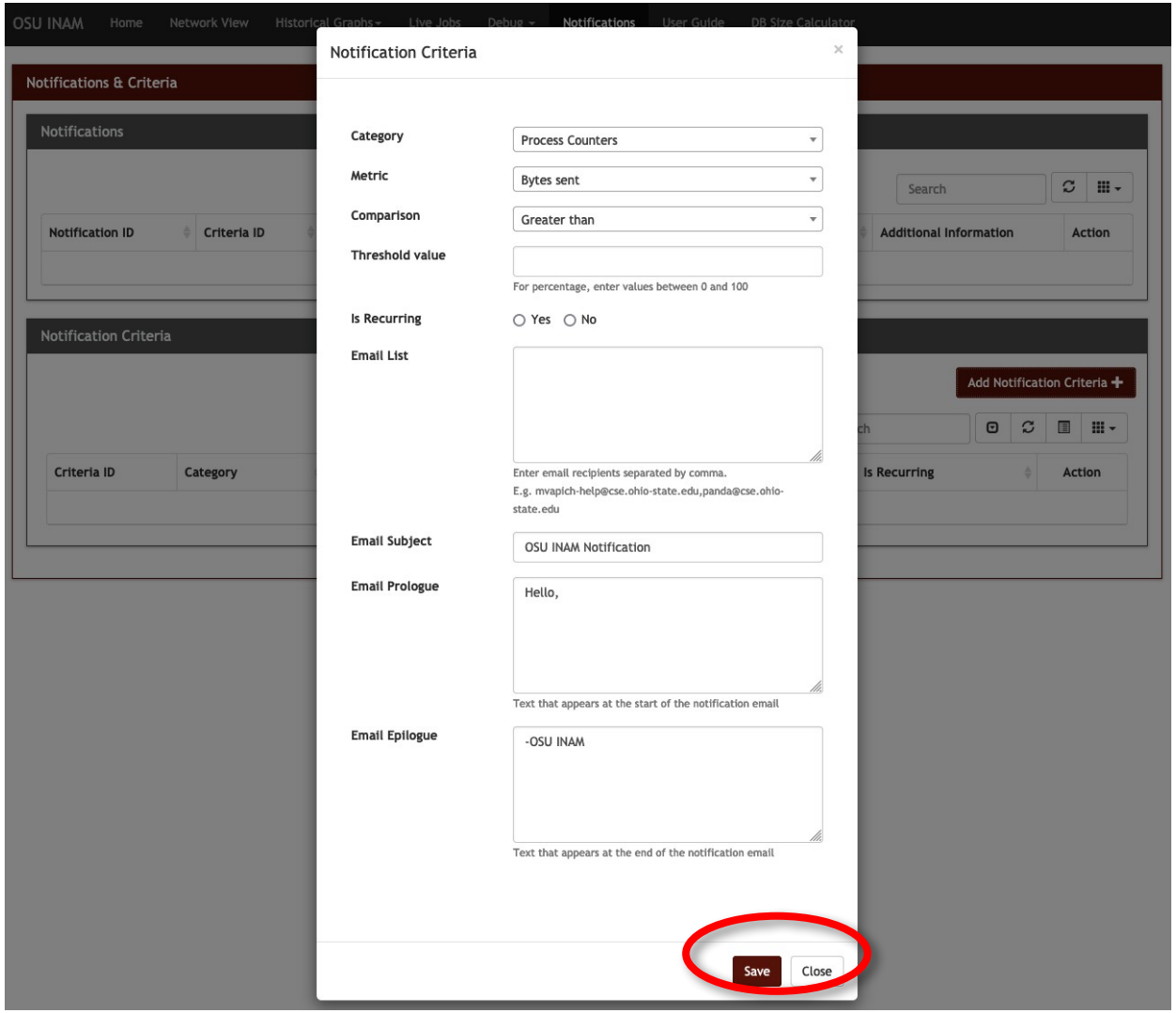
Demo 4: Setting up Notifications – Notifications

- Choose Notifications tab from tabs
- This tab shows the notifications system
- The top half shows the notifications that have already been created/is in effect
- The bottom half allows to create a new notification
- Choose to add a new notification criteria



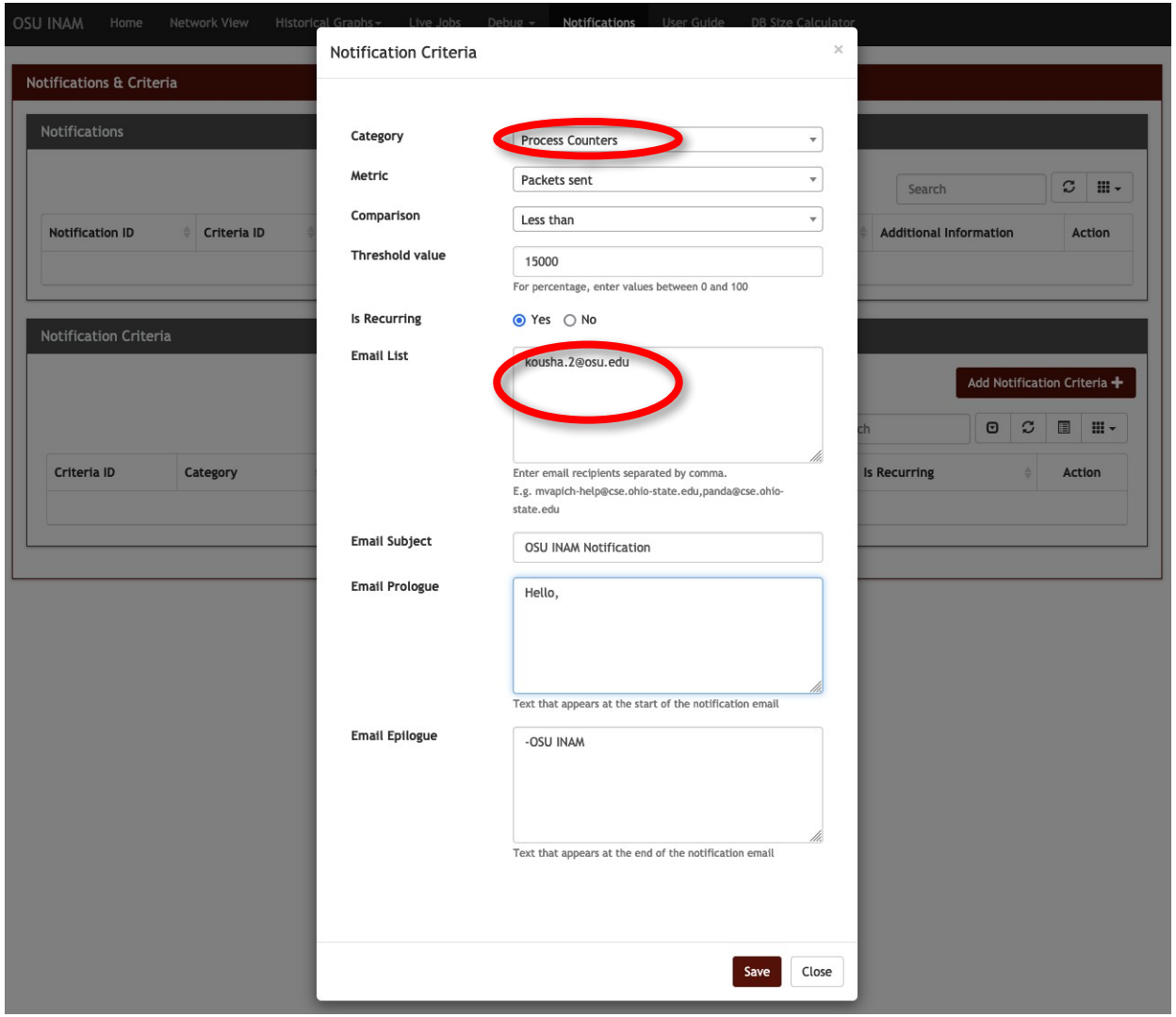
Demo 4: Setting up Notifications – Adding Notifications

- Upon Click a new window should appear for adding notification
- Explore the available options as this part is under development
- Request for feedback: What type of notifications are you interested to have support here?
- Future Work: Ability to create multi-variable notifications.



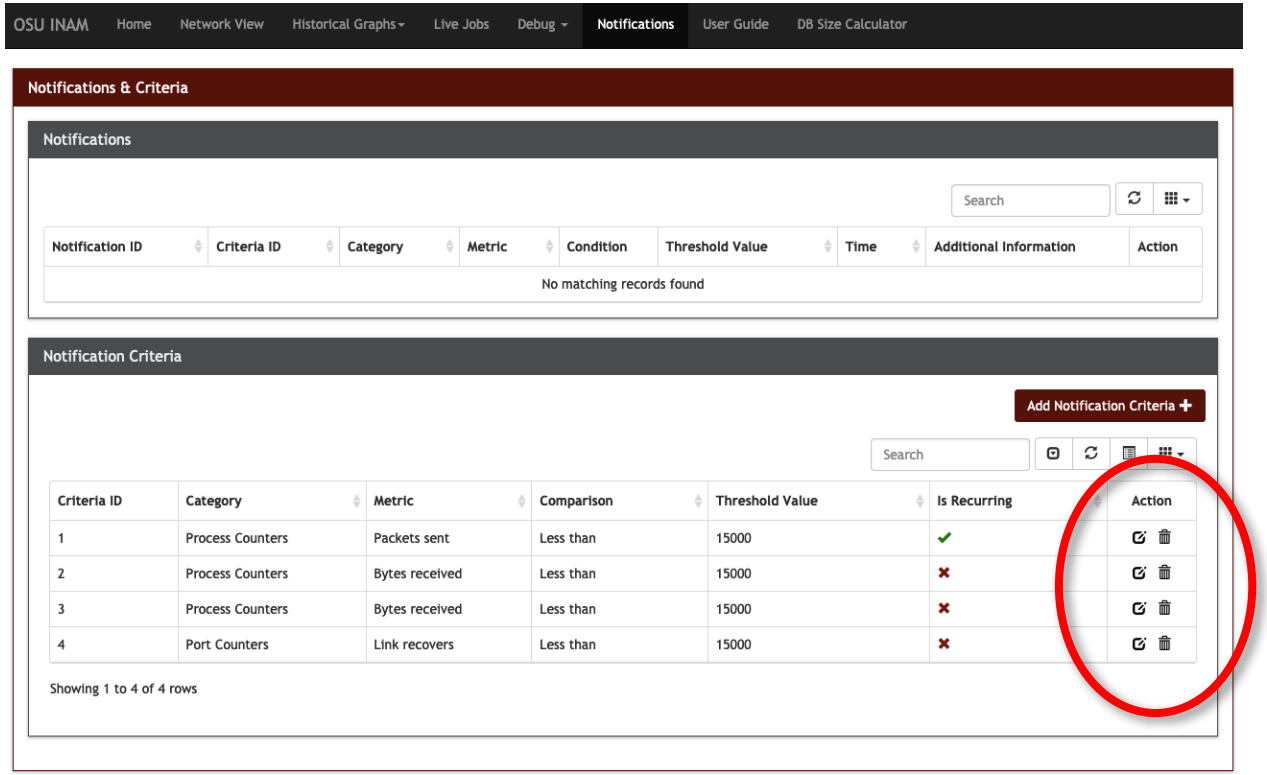
Demo 4: Setting up Notifications – Setting Up Notifications

- Try to add a new notification by choosing your favorite values
- Two modes support Recurring and nonrecurring
- The text for the message and subject could be customized
- Click save after you finish



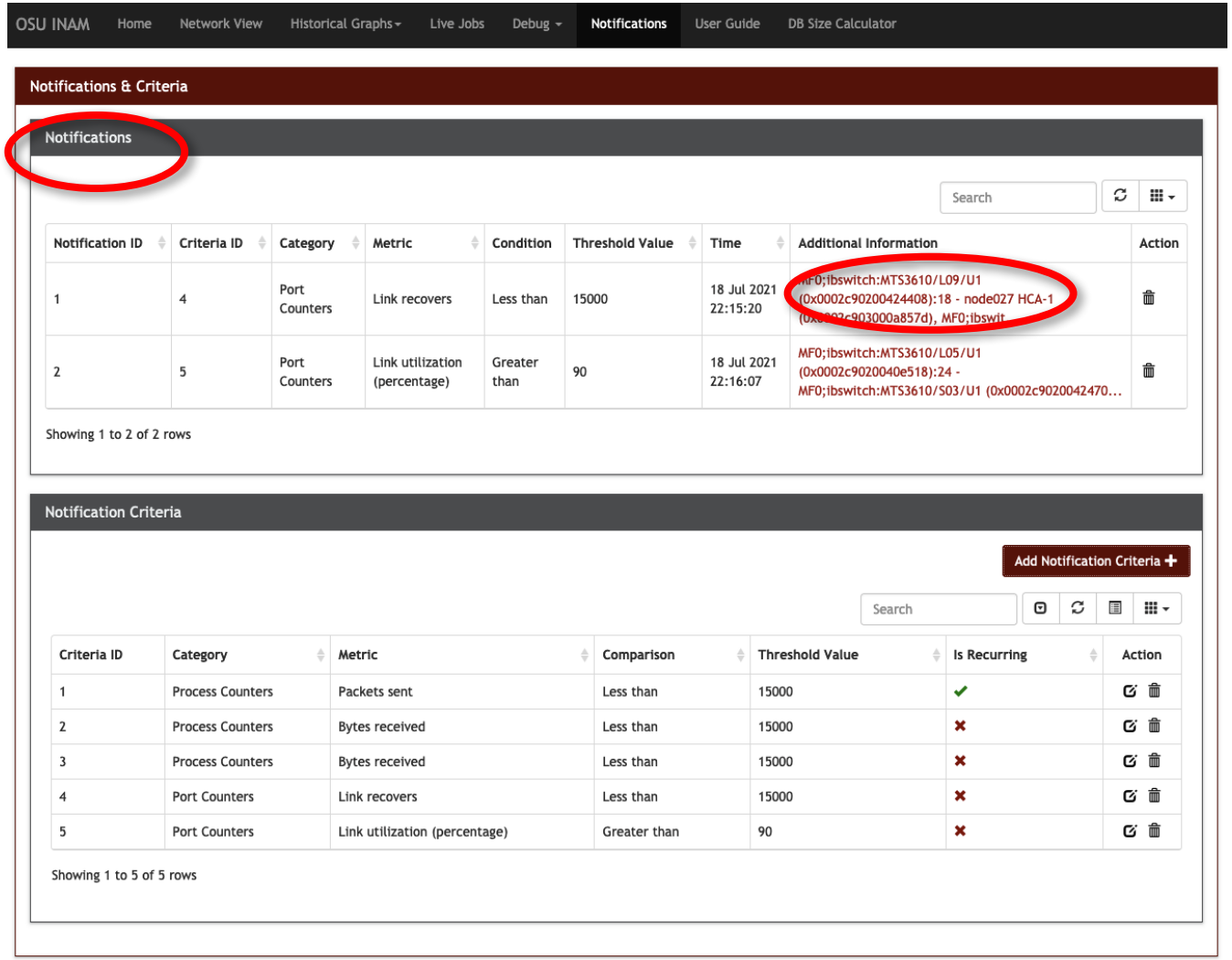
Demo 4: Setting up Notifications – Managing Notifications

- Your saved notification should show in Notification page
- You can edit or delete the notification and view the details of it
- Request for feedback: What would you suggest to improve managing notifications section?



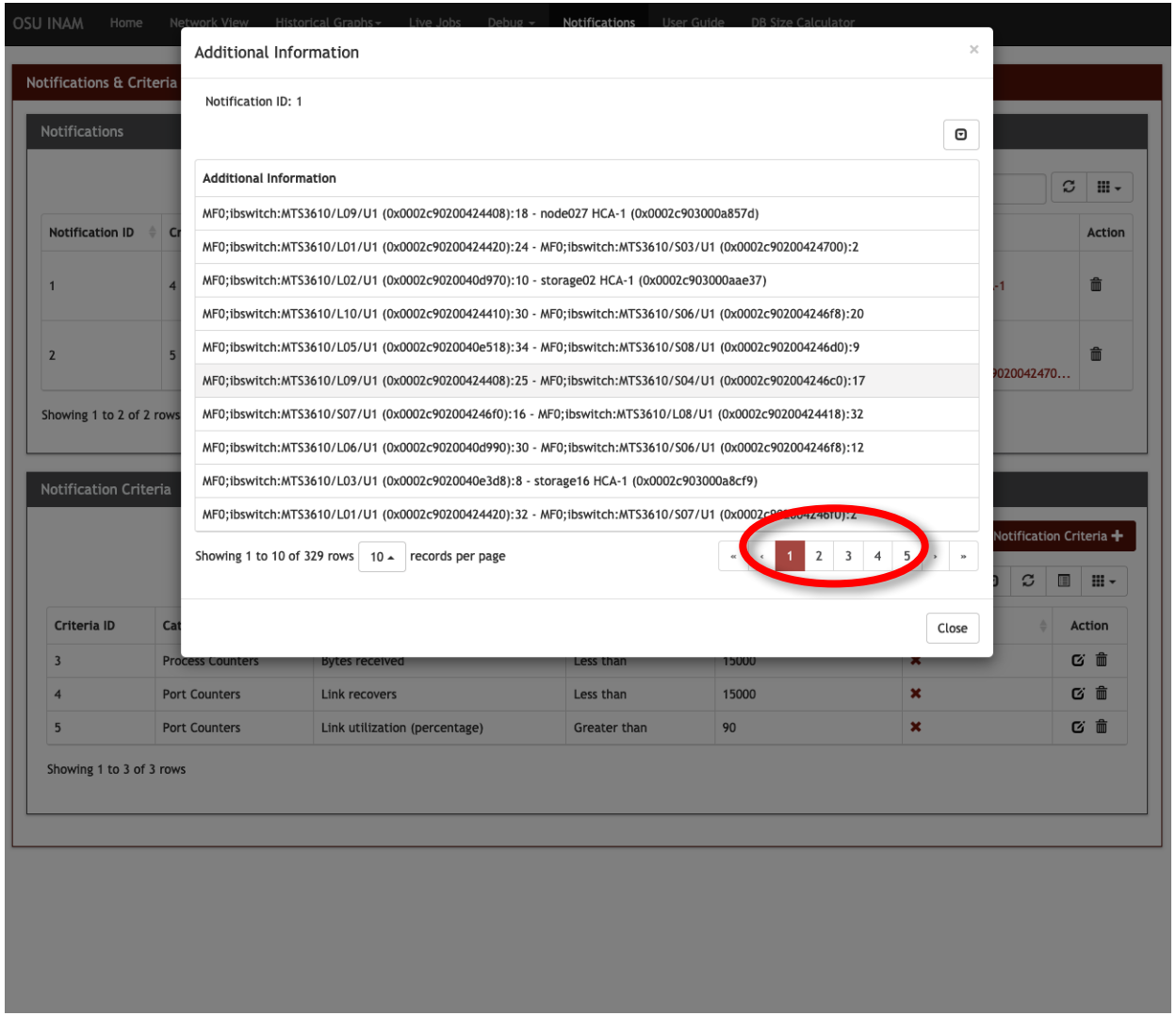
Demo 4: Setting up Notifications - Viewing Occurrences

- Upon occurring of event that was specified in the notification you should see a history of them here
- You can click on additional information to see the details



Demo 4: Setting up Notifications – Viewing Occurrences' Details

- After clicking you can see the details of node GUID and names with their ports for which the notification has happened
- Request for feedback: What would you suggest to improve this notifications section?



Outline

- Overview
- OSU INAM Components
- Overhead Analysis
- Downloading, Installing, and Configuring OSU INAM
- Demos
- **Concluding Remarks and Future Work**

Concluding Remarks and Future Work

- Presented details of OSU INAM tool capable of analyzing the communication traffic on the InfiniBand network with inputs from the MPI runtime. Major features include:
  - Analyze and profile network-level activities with many parameters (data and errors) at user specified granularity
  - Capability to analyze and profile node-level, job-level and process-level activities for MPI communication
    - Point-to-Point, Collectives and RMA
  - Remotely monitor CPU utilization of MPI processes at user specified granularity
  - low overhead (5%), scalable (4K processes on 256 nodes), and fined-granularity profiling of HPC communication stack
  - Visualize the data transfer happening in a "live" or historical fashion for
    - Entire Network, Particular Job One or multiple Nodes, One or multiple Switches
- The designs have been publicly released as a part of v0.9.6 and are being used by many users
  - Downloaded over 5000 times form the project website
  - Community engagement with
    - OSC @ USA, NOAA @ USA, U. of Utah @ USA, CAE Services @ Germany, Pratt & Whitney, Ghent University @ Germany, Cyfronet @ Poland, and Georgia Tech Univ @ USA
- As future work we aim to
  - Add support to profile and analyze GPU-based communication
  - Address the storage bottleneck by exploring high-performance database engines
  - Support more intra-node metrics and power consumption profiling

Funding Acknowledgments

Funding Support by

Equipment Support by

Acknowledgments to all the Heroes (Past/Current Students and Staffs)

Current Students (Graduate)

- N. Alnaasan (Ph.D.)
- Q. Anthony (Ph.D.)
- C.-C. Chun (Ph.D.)
- N. Contini (Ph.D.)
- A. Jain (Ph.D.)
- K. S. Khorassani (Ph.D.)
- P. Kousha (Ph.D.)
- B. Michalowicz (Ph.D.)
- B. Ramesh (Ph.D.)
- K. K. Suresh (Ph.D.)
- A. H. Tu (Ph.D.)
- S. Xu (Ph.D.)
- Q. Zhou (Ph.D.)
- K. Al Attar (M.S.)
- L. Xu (Ph.D.)
- H. Ahn (Ph.D.)
- G. Kuncham (Ph.D.)
- R. Vaidya (Ph.D.)
- J. Yao (Ph.D..)
- M. Han (M.S.)
- A. Guptha (M.S.)

Current Research Scientists

- M. Abduljabbar
- A. Shafi

Current Students (Undergrads)

- V. Shah
- T. Chen

Current Faculty

- H. Subramoni

Current Software Engineers

- B. Seeds
- N. Pavuk
- N. Shineman
- M. Lieber

Current Research Specialist

- R. Motlagh

Past Students

- A. Awan (Ph.D.)
- A. Augustine (M.S.)
- P. Balaji (Ph.D.)
- M. Bayatpour (Ph.D.)
- R. Biswas (M.S.)
- S. Bhagvat (M.S.)
- A. Bhat (M.S.)
- D. Buntinas (Ph.D.)
- L. Chai (Ph.D.)
- B. Chandrasekharan (M.S.)
- S. Chakraborthy (Ph.D.)
- N. Dandapanthula (M.S.)
- V. Dhanraj (M.S.)
- C.-H. Chu (Ph.D.)
- T. Gangadharappa (M.S.)
- K. Gopalakrishnan (M.S.)
- J. Hashmi (Ph.D.)
- W. Huang (Ph.D.)
- W. Jiang (M.S.)
- J. Jose (Ph.D.)
- M. Kedia (M.S.)
- S. Kini (M.S.)
- M. Koop (Ph.D.)
- K. Kulkarni (M.S.)
- R. Kumar (M.S.)
- S. Krishnamoorthy (M.S.)
- K. Kandalla (Ph.D.)
- M. Li (Ph.D.)
- P. Lai (M.S.)
- J. Liu (Ph.D.)
- M. Luo (Ph.D.)
- A. Mamidala (Ph.D.)
- G. Marsh (M.S.)
- V. Meshram (M.S.)
- A. Moody (M.S.)
- S. Naravula (Ph.D.)
- R. Noronha (Ph.D.)
- X. Ouyang (Ph.D.)
- S. Pai (M.S.)
- S. Potluri (Ph.D.)
- K. Raj (M.S.)
- R. Rajachandrasekar (Ph.D.)
- D. Shankar (Ph.D.)
- G. Santhanaraman (Ph.D.)
- N. Sarkauskas (B.S. and M.S)
- N. Senthil Kumar (M.S.)
- A. Singh (Ph.D.)
- J. Sridhar (M.S.)
- S. Srivastava (M.S.)
- S. Sur (Ph.D.)
- H. Subramoni (Ph.D.)
- K. Vaidyanathan (Ph.D.)
- A. Vishnu (Ph.D.)
- J. Wu (Ph.D.)
- W. Yu (Ph.D.)
- J. Zhang (Ph.D.)

Past Post-Docs

- D. Banerjee
- X. Besseron
- M. S. Ghazimeersaeed
- H.-W. Jin
- J. Lin
- M. Luo
- E. Mancini
- K. Manian
- S. Marcarelli
- A. Ruhela
- J. Vienne
- H. Wang

Past Senior Research Associate

- J. Hashmi

Past Programmers

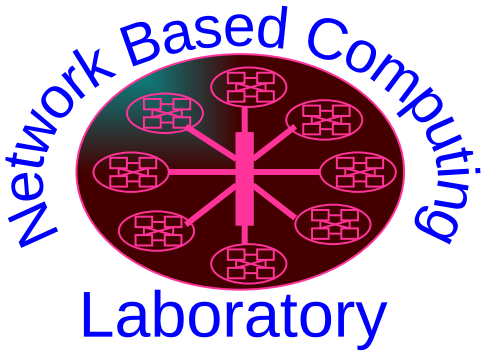
- A. Reifsteck
- D. Bureddy
- J. Perkins

Past Research Specialist

- M. Arnold
- J. Smith

Thank You!

subramoni.1@osu.edu, kousha.2@buckeyemail.osu.edu, kolli.28@buckeyemail.osu.edu



Network-Based Computing Laboratory
http://nowlab.cse.ohio-state.edu/

The High-Performance MPI/PGAS Project
http://mvapich.cse.ohio-state.edu/

The High-Performance Big Data Project
http://hibd.cse.ohio-state.edu/

The High-Performance Deep Learning Project
http://hidl.cse.ohio-state.edu/