# BLUE WATERS

SUSTAINED PETASCALE COMPUTING

May 8, 2014

# Toward Understanding the Impact of I/O Patterns on Congestion Protection Events on Blue Waters

**Rob Sisneros**, Kalyana Chadalavada

GREAT LAKES CONSORTIUM FOR PETASCALE COMPUTATION

# Why?

- `Size != Super`
- `HSN   == Super`

- Fast food philosophy
  - What the hell does *super size* mean?

# Why? Less Congestion is Better

- Really?
  - Maybe more congestion means higher utilization
  - Do you have proof?
- Proof: deployed safeguards against congestion

# Congestion Protection Events

- Triggered during network congestion
- Throttles network reducing injection bandwidth

- Everyone suffers (short term)

# Reducing Congestion

- Obvious: all things equal, less congestion is better
- Otherwise: avoiding congestion protection events at least has a positive impact on *perceived* user experience

# An Issue with This Title

- "Toward Understanding the Impact of I/O Events On Congestion Protection Events on Blue Waters"
- Big assumption: we understand I/O patterns
- We don't.

# Toward Fixing the Title: "Toward…"

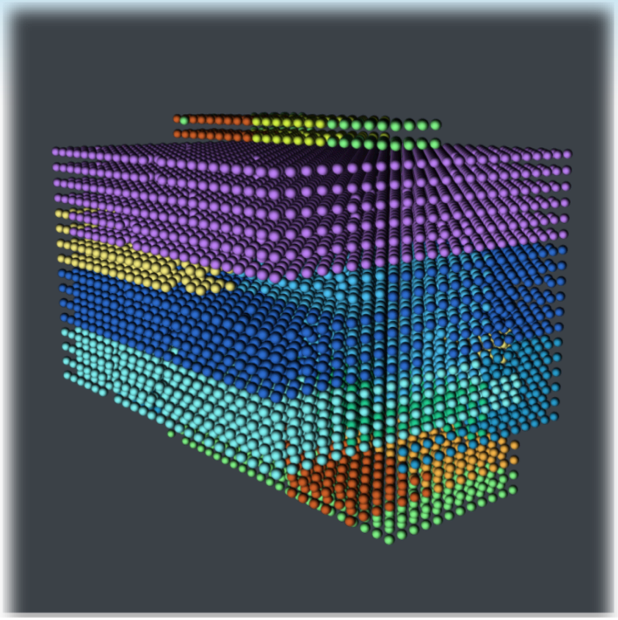
- Groundwork needed to understand/identify HSN network traffic patterns
- Hypothesis: understanding this requires ability to *conceptualize* system-wide traffic patterns
- We need vis.
  - BW network is 3D torus
  - How to visualize that?
  - We spent a lot of time on this, so…

# A Better Title

"Toward Understanding HSN Traffic Patterns on Blue Waters via Visual Analytics"

"and Congestion Protection"

# HSN Visualization Difficulties



- 3D is 4D
- Method: map topological coordinates directly to 3D grid
  - Deployed at BW
  - People are happy!

# The Problems with 3D Mapping

- Distances represented incorrectly
- Visual occlusion
- Boundary conditions (torus wrap-around) ignored

- Still
  - Great for general sense of job layout
  - Could be used for limited traffic visualization (on a small subset of the torus)

# Our Layout: An Aside

- Is Blue Waters' HSN really a 3D Torus?
- Well:
  - Yes, the machine room is a magical place that utilized 4 spatial dimensions
  - Kind of yes, a *practical* 3D torus, after some simplifying assumptions
- Either way:
  - A photograph of the machine is a very effective 2D representation of a 3D torus
  - Let's use something like that!

# Our Torus Layout

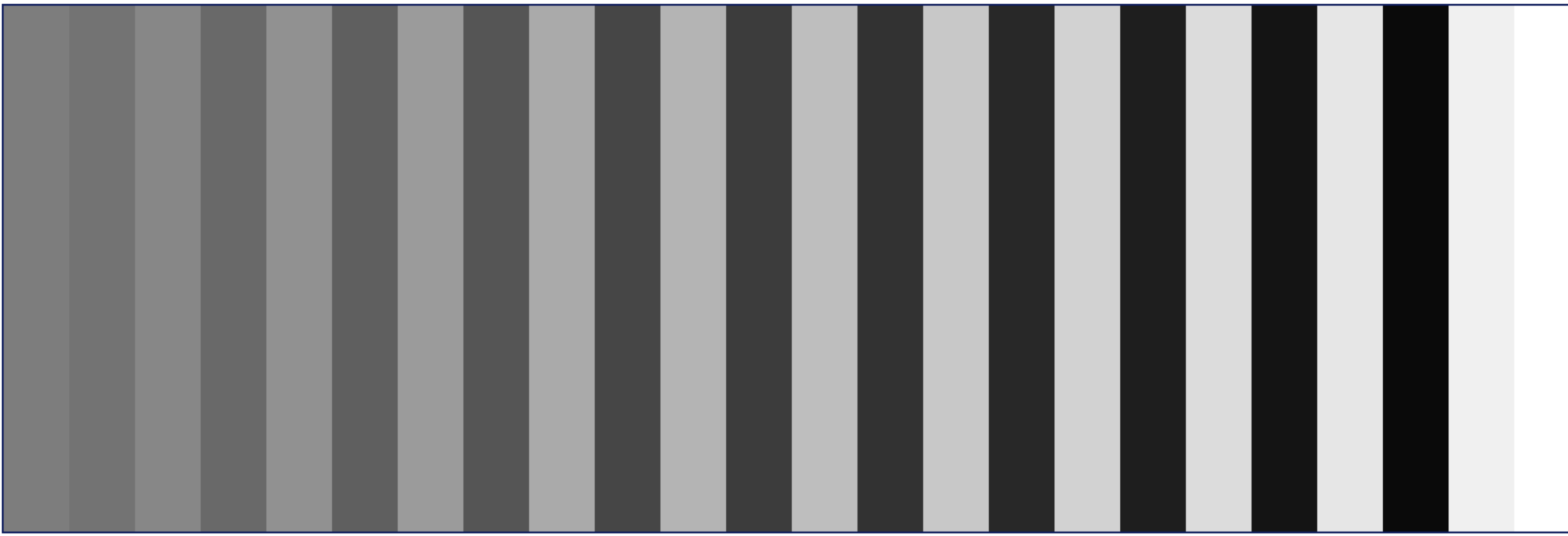
- Blue Waters cabinets are mapped to 2D corresponding to their layout on the machine room floor
- This X&Y space is widened so that the Z dimension (throughout a cabinet) can be flattened into this space

# Creating Vis: Another Aside

- For once, we are the vis developers *and* the targeted user group!!!

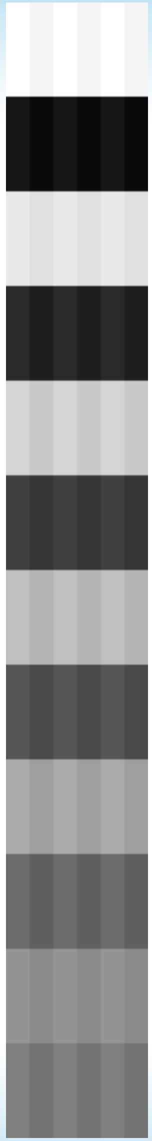
- And we want clarity.

# Clarity in X



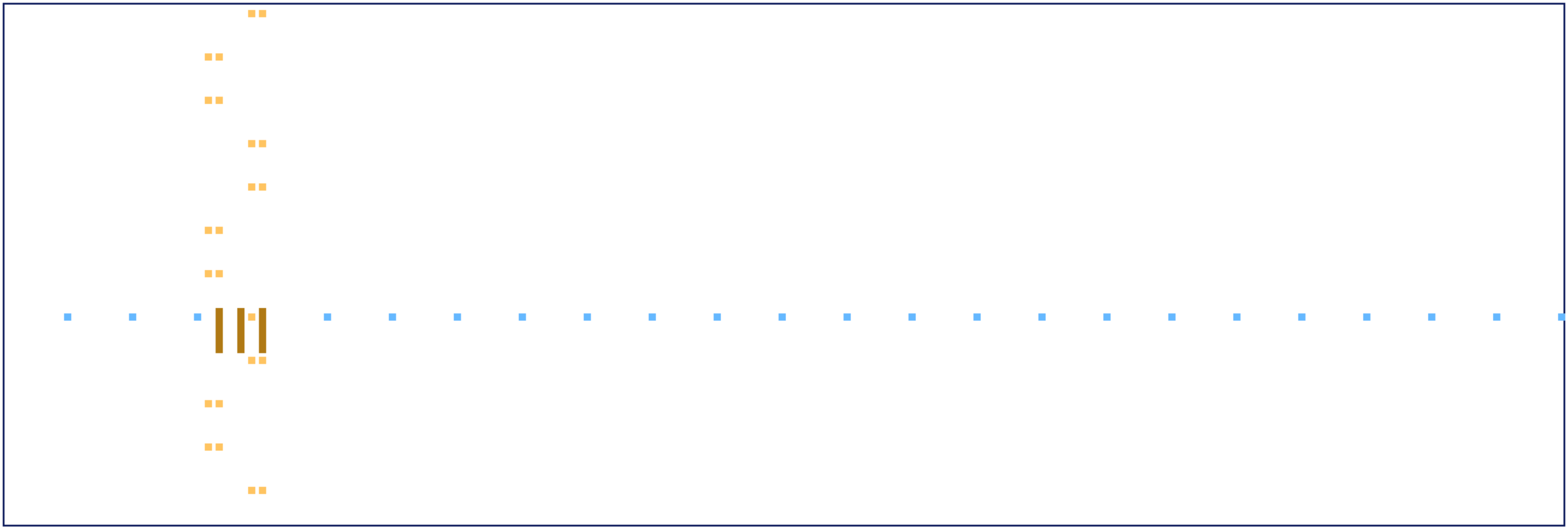
# Clarity in Y and Z

Y at X=0:



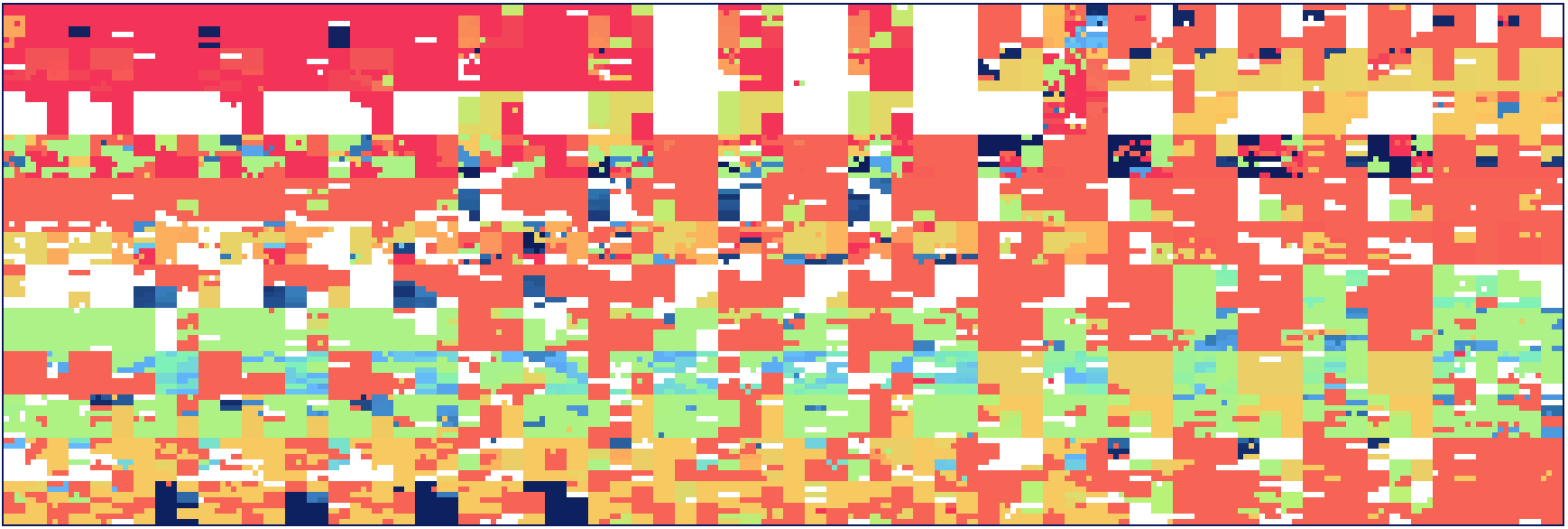
Z at X=0, Y=0:

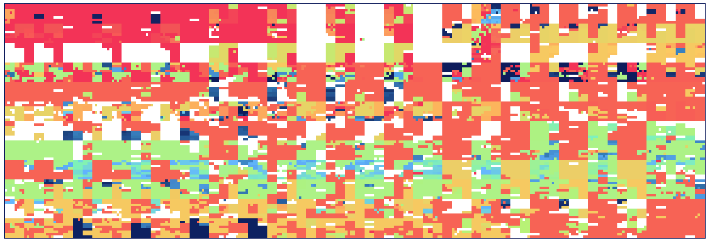
# Clarity Continued: A Trip Around the Torus



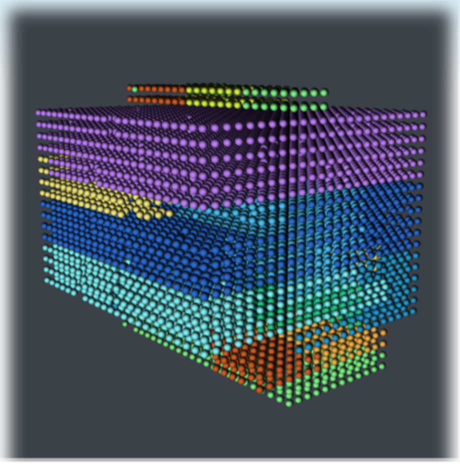
# Clarity Continued: A Job Layout

# The End of Clarity: 352 Running Jobs





vs.



You get everything, maybe nothing

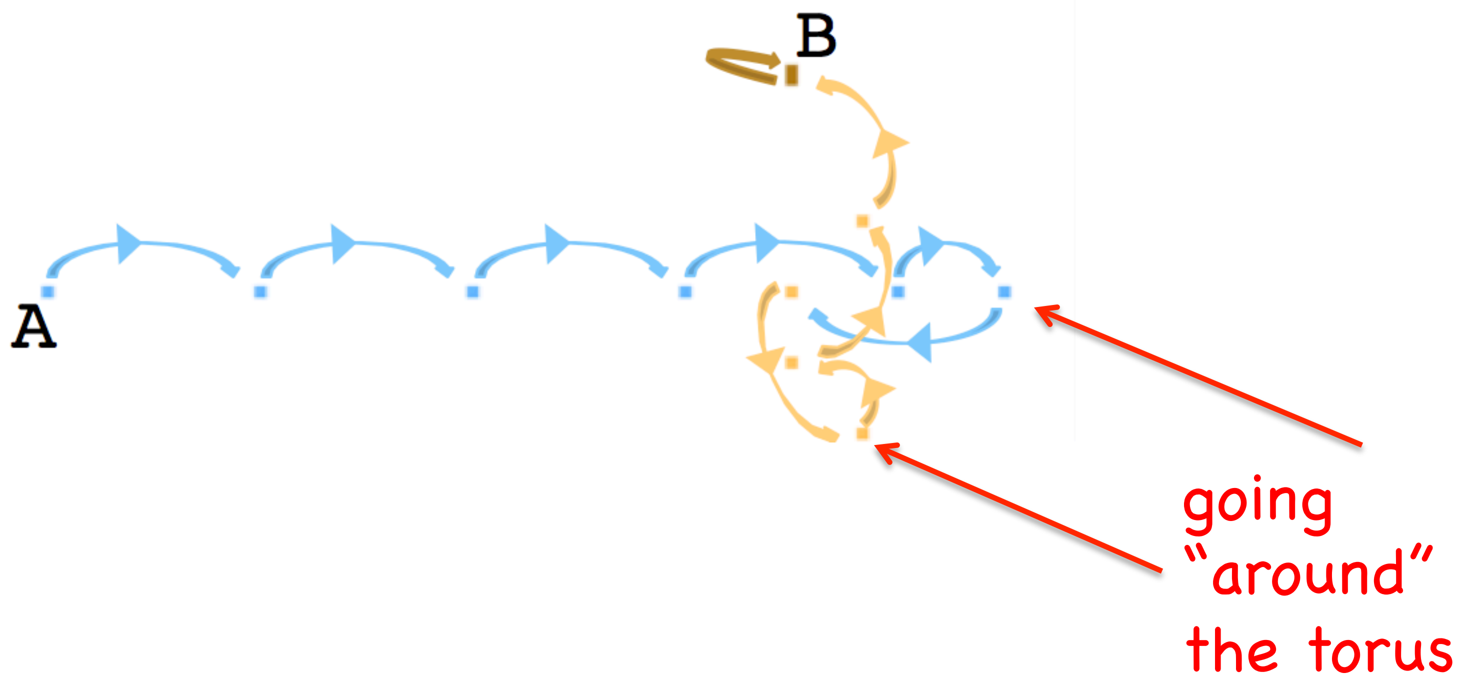
You get something, just not everything

# Finally Seeing Network Traffic

- Implemented an infrastructure to read and draw communications
- Communications are, timestamp, start node, end node, and message size
- The full path through the torus is calculated for each communication
- This is accumulated across all communications for a single image

# Example: Hops from (5,15,6) to (23,6,7)

# Clarity Returns



# Are We Looking in the Right Place?

- Viewing communications requires significant additional code profiling
- We have no way to leverage historical data to analyze several logged significant events
- But we want to.

# More on Congestion Protection Events

- When a congestion protection event occurs
  - Traffic into each node is logged
  - Nodes are ranked by traffic
  - The compute nodes with most traffic are mapped back to their jobs to suggest contributors
  - Service nodes are ignored

# Ignoring Service Nodes

- Our initial motivation:

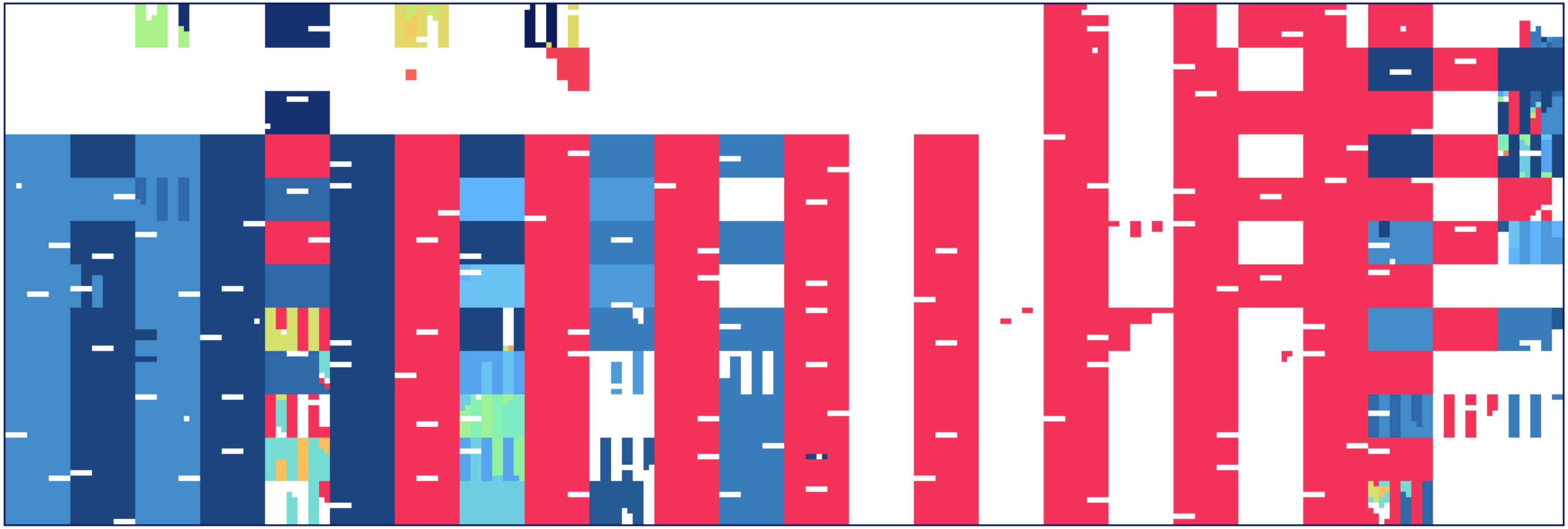
  "The top 50 nodes with highest traffic are service nodes, guess there is a lot of I/O going on."

- The confusion
  - I/O traffic routes through compute nodes
  - MPI traffic routes through service nodes

# Can MPI Traffic Elevate Service Nodes?

- Experiment: simulate worst case scenario during actual time period of a congestion protection event that had heavy service node traffic
- The Event
    - March 25, 2014 at 7:46 AM
    - 32 jobs running
    - Service nodes had heaviest traffic

# 32 Jobs

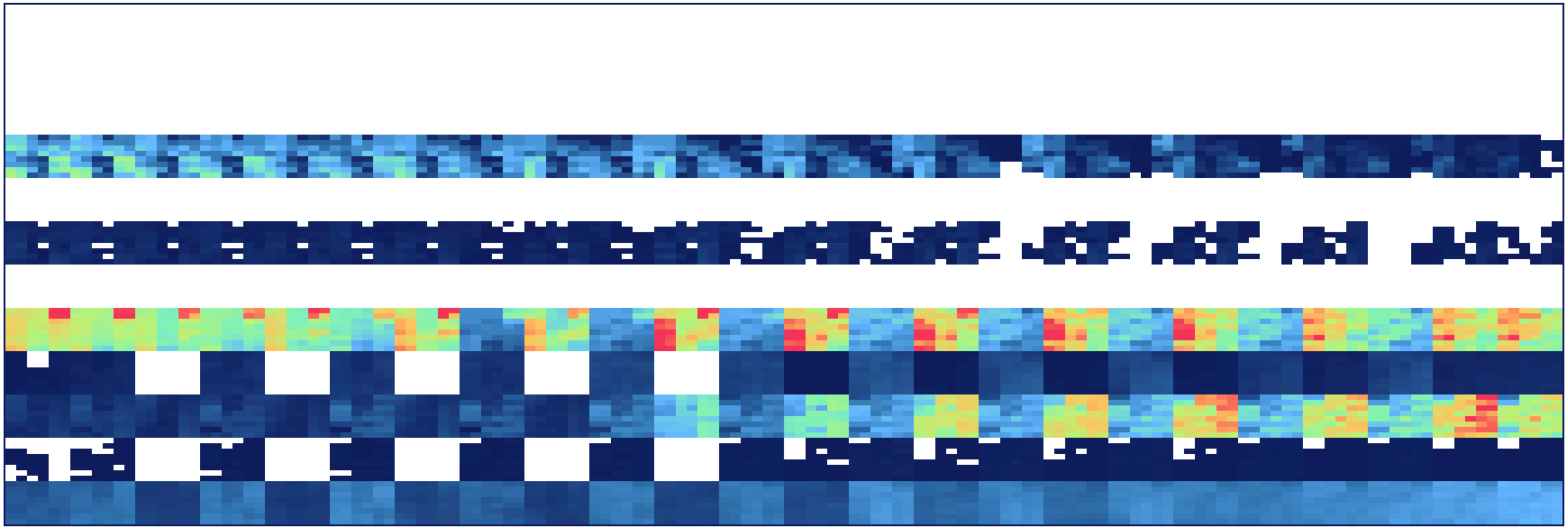


# The Worst Case Scenario

- All to all communications
- Rough heuristic to calculate all to all
  - Take upper triangle of the matrix of all possible communications among nodes of a job
  - Calculate for each job
  - Accumulate across the system

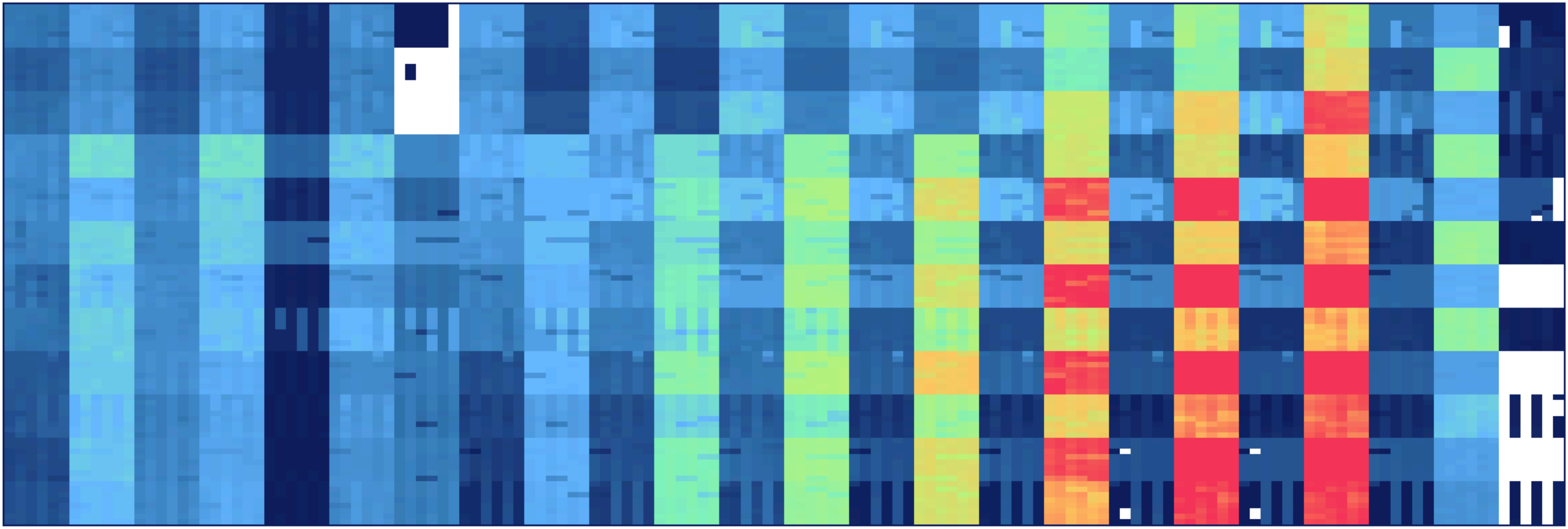
# Easing Into It: A Single Job

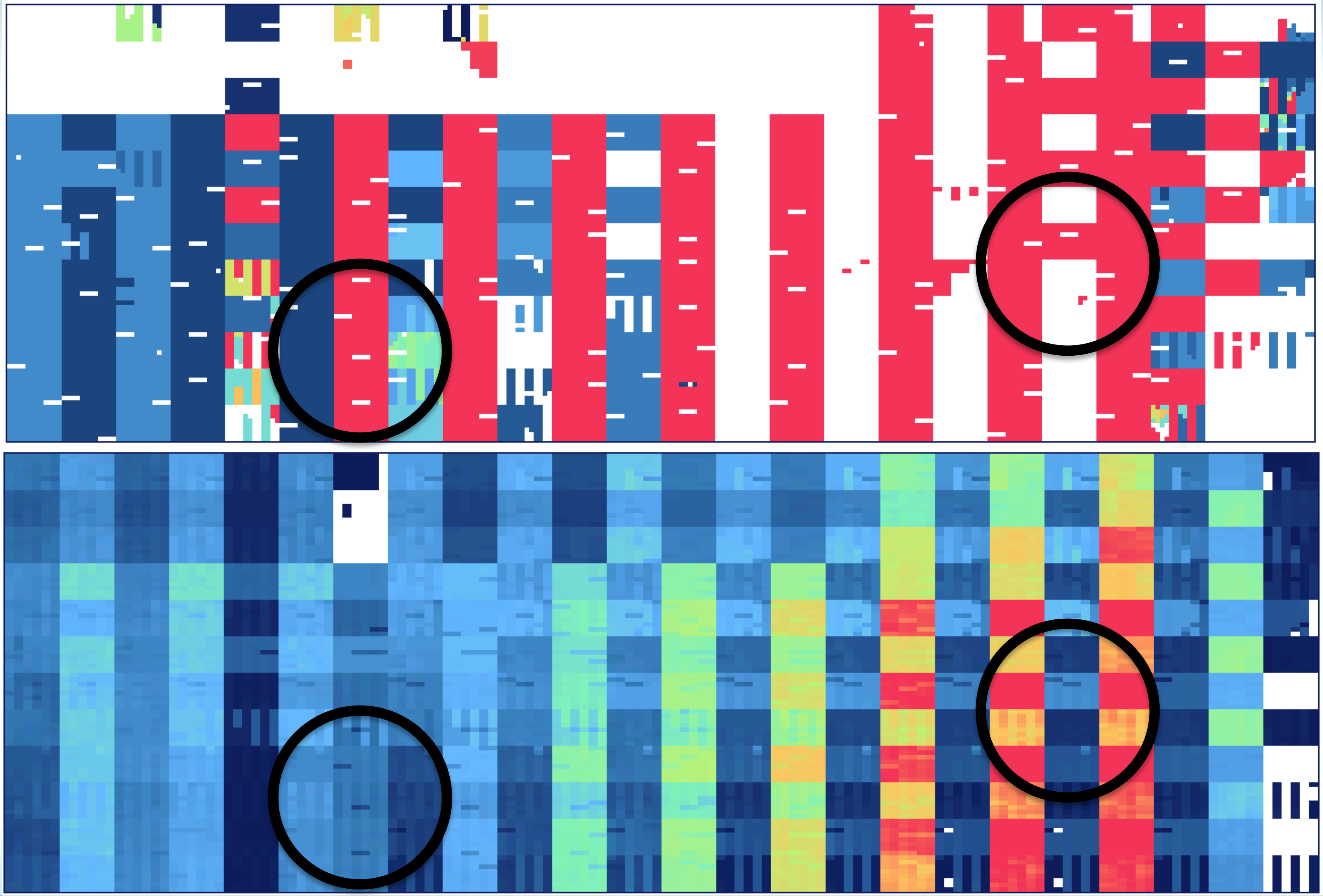
# Easing Into It: A Single Job



# Proof of Concept – Nice!

# All to All for 32 CPE Jobs





# The Point

- Not surprising that this message pattern floods the system
- Is surprising that some service nodes remain distinguishable from compute nodes while others do not
- Ignoring nodes may be good practical idea, but may also be the wrong direction for best potential analysis

# The Possibilities

- Implement different communication patterns for simulation with historical data
- Refinement
  - Use actual implementation for communications rather than rough heuristics
  - Implement code-specific patterns
- Deploy traffic viewer
  - Increased I/O profiling
  - Real-time via OVIS data

# Wrapping Up

- OVIS data?
  - Yes!
  - Solves problem of requiring advanced profiling
  - Does *not* solve problem understanding traffic
    - Easy to pinpoint which nodes have heavy traffic
    - Still not easy to find which nodes are actually responsible for it
  - System must still be modeled to find

# Questions

- Now? Ask away.

- Later?
  - sisneros@illinois.edu
  - kalyan@illinois.edu