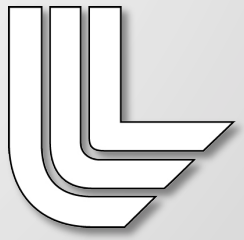
Ravel: Ordering Traces Logically to Identify Lateness in Parallel Programs



Martin Schulz, Lawrence Livermore National Laboratory

Joint work with: K. Isaacs, P.-T. Bremer, I. Jusufi, T. Gamblin, A. Bhatele, B. Hamann
LLNL & UC Davis

Petatools Workshop ◆ August 4th, 2014

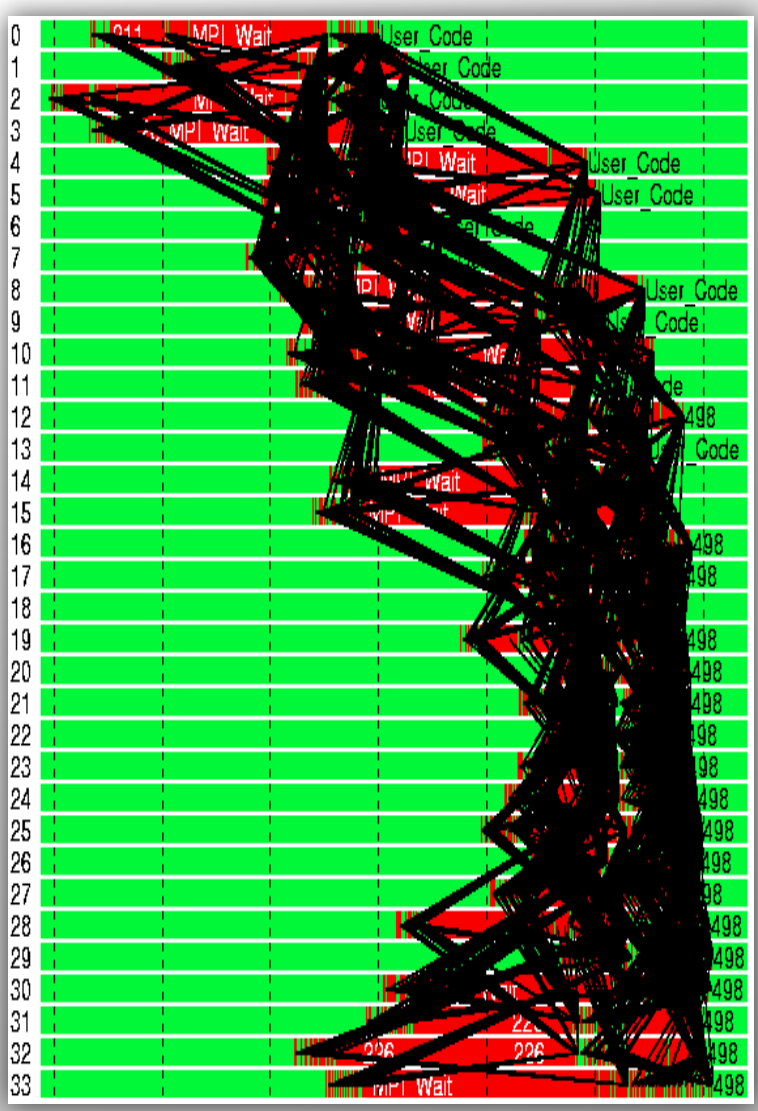
LLNL-PRES-658255

http://scalability.llnl.gov/

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

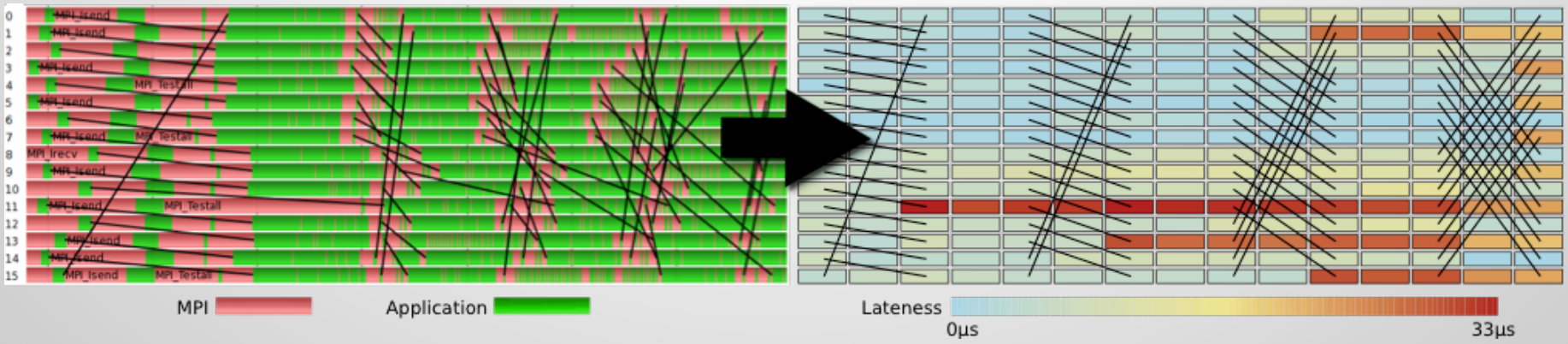
How to Analyze Traces? –OR- How to Comb the Hairball?

- **Communication traces**
  - Highly useful for detailed analysis
  - Capture all details
- **Timeline views of traces**
  - Standard way to show traces
  - But: hard to interpret
  - Also: hard to scale
    - In number of processes (each process is one timeline)
    - In time (how to find interesting parts)



- **Goal: new ways to visualize communication traces**

Main Concept: Visualize Using Virtual Time

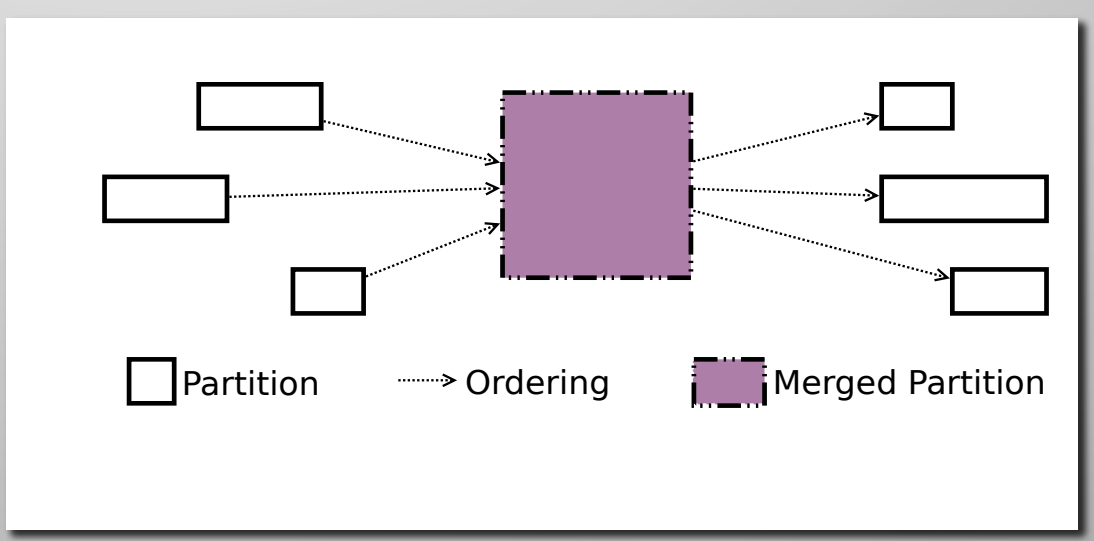


- **Advantage: expose logical communications structure**
- **Challenges**
  - Detecting logical structure
  - Integrating wall clock time information
  - Displaying information in a scalable way
  - Creating an interactive tool with linked views

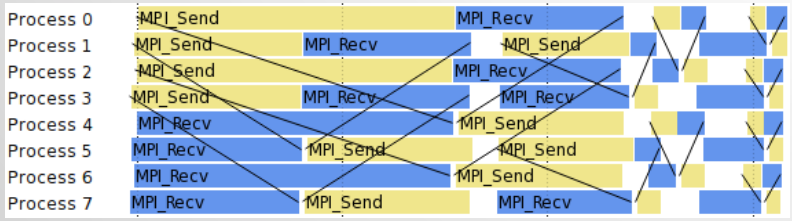
This lead us to Ravel, a new interactive trace visualizer

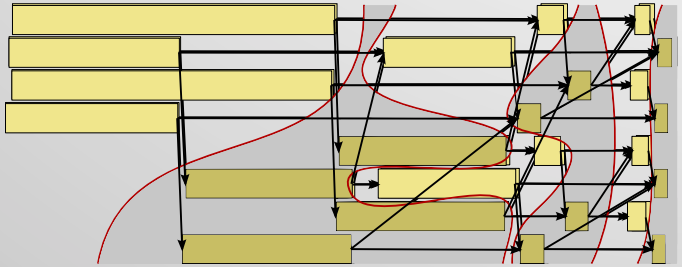
Detecting and Extracting Logical Structure

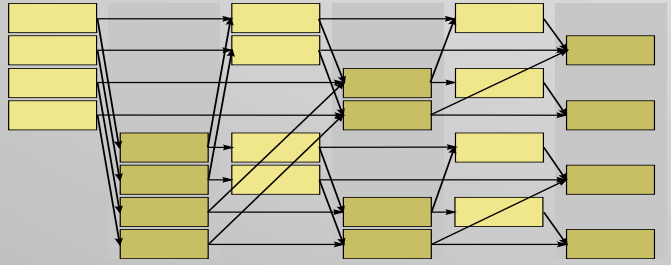
- **Goal**
  - Identify operations that logically belong together (e.g., a stencil)
  - Capture the developer's intention and organization
- **Partitioning based on "Happens-before Relationships"**
  - Grow single events into connected groups
  - Merge groups with cyclic dependencies
- **Further heuristics**
  - Join events connected by MPI_Waitall
  - Ensure job-wide communication per phase/component
- **Detects most cases**

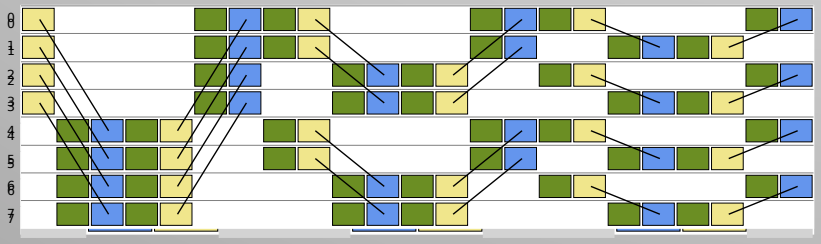


Local and Global Step Assignments





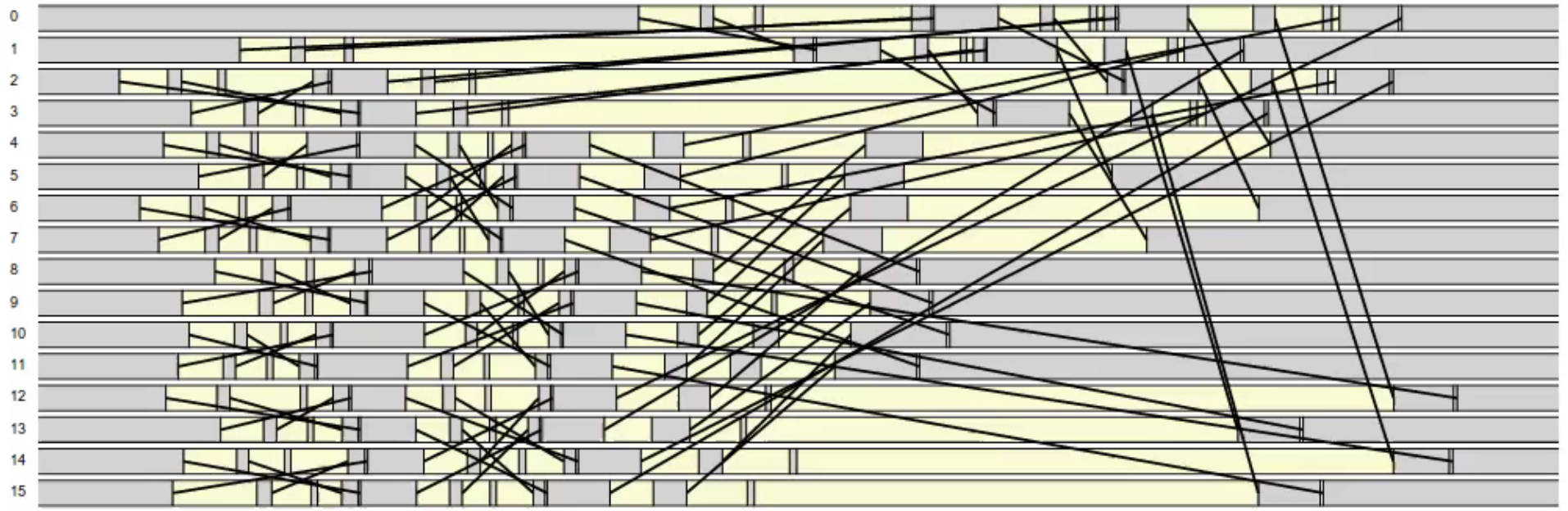




- Collect VampirTrace data (OTF)
- Create Reduced Happens-Before Trace based on Send Events
- Insert partitions based on component graph distance
- Align components
- Split Send and Receive
- Add Blocks for Computation Time

Resulting Physical -> Logical Time Transformation

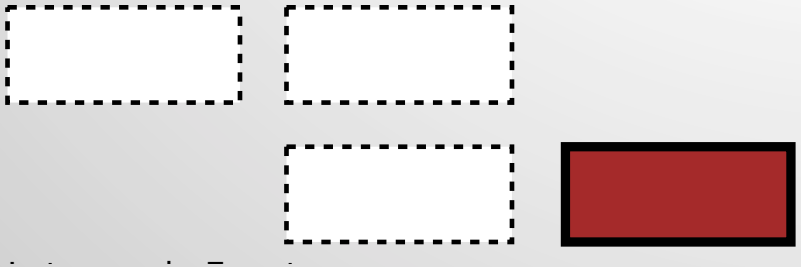
- **Example: MG on 16 processors**



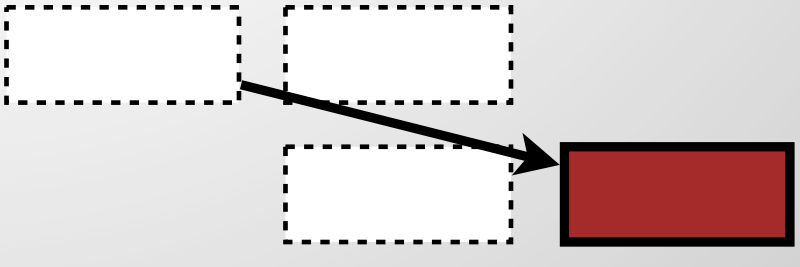
Lateness Metric to Re-Include Time Information

- **Logical time exposes the structure of communication**
  - Alignment of logical grouped events
  - "Clean" views
- **Problem: time information lost**
  - Critical for performance optimization
  - Need to concentrate on events that cause or propagate delays
- **New lateness metric**
  - Compare exit times within aligned components
  - Difference to first process exiting the component
  - Shows delays (or lateness) relative to communication structure
- **Additional metric: Differential lateness**
  - Identify causes for lateness by showing where lateness increases

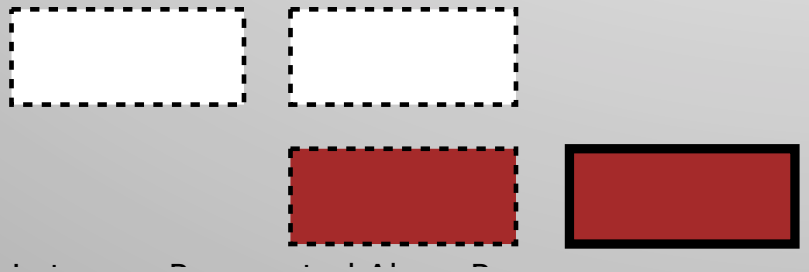
Lateness Creation and Propagation



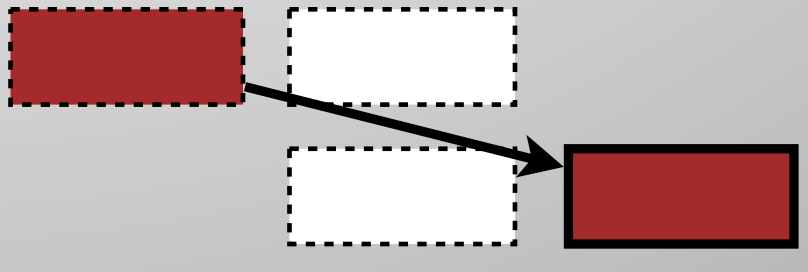
Event causing lateness



Message causing lateness

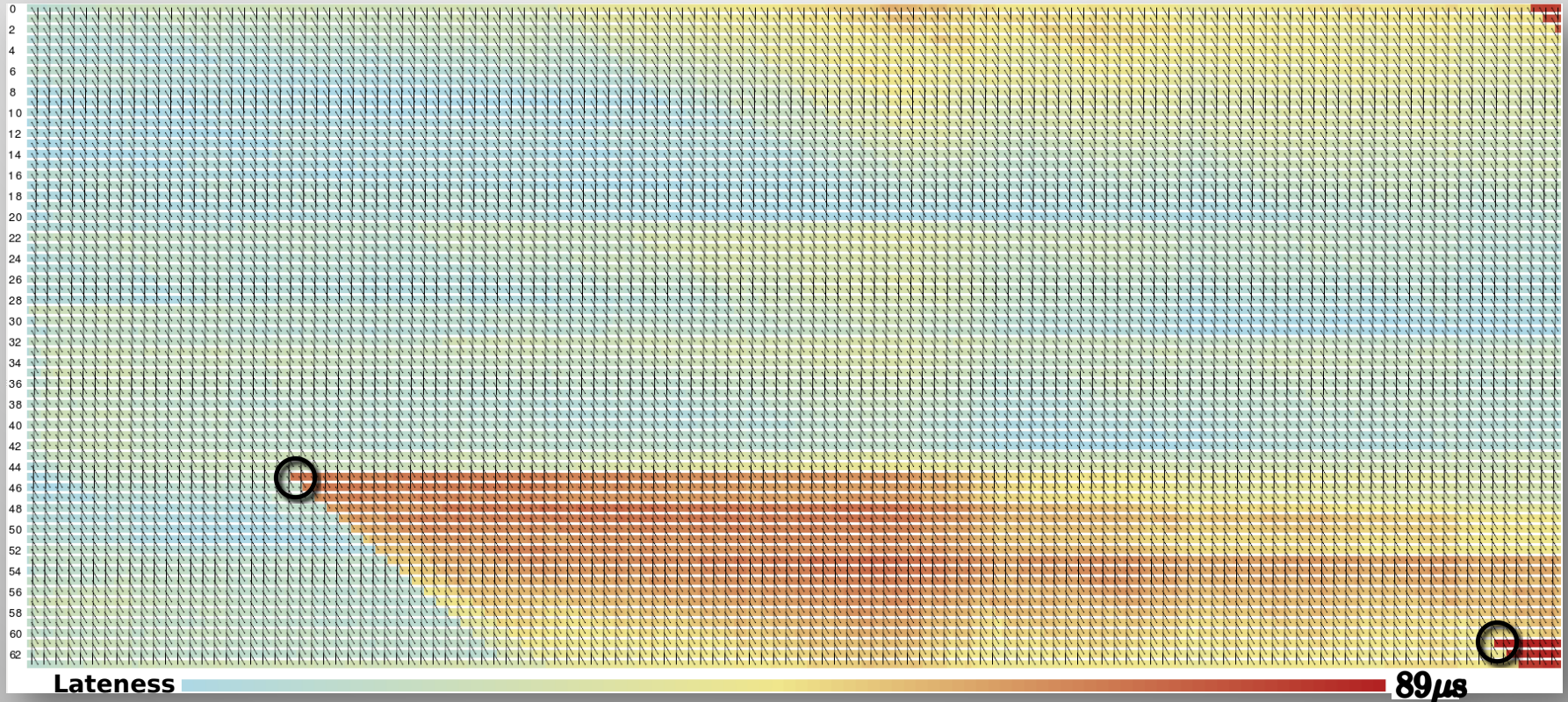


Event propagating lateness



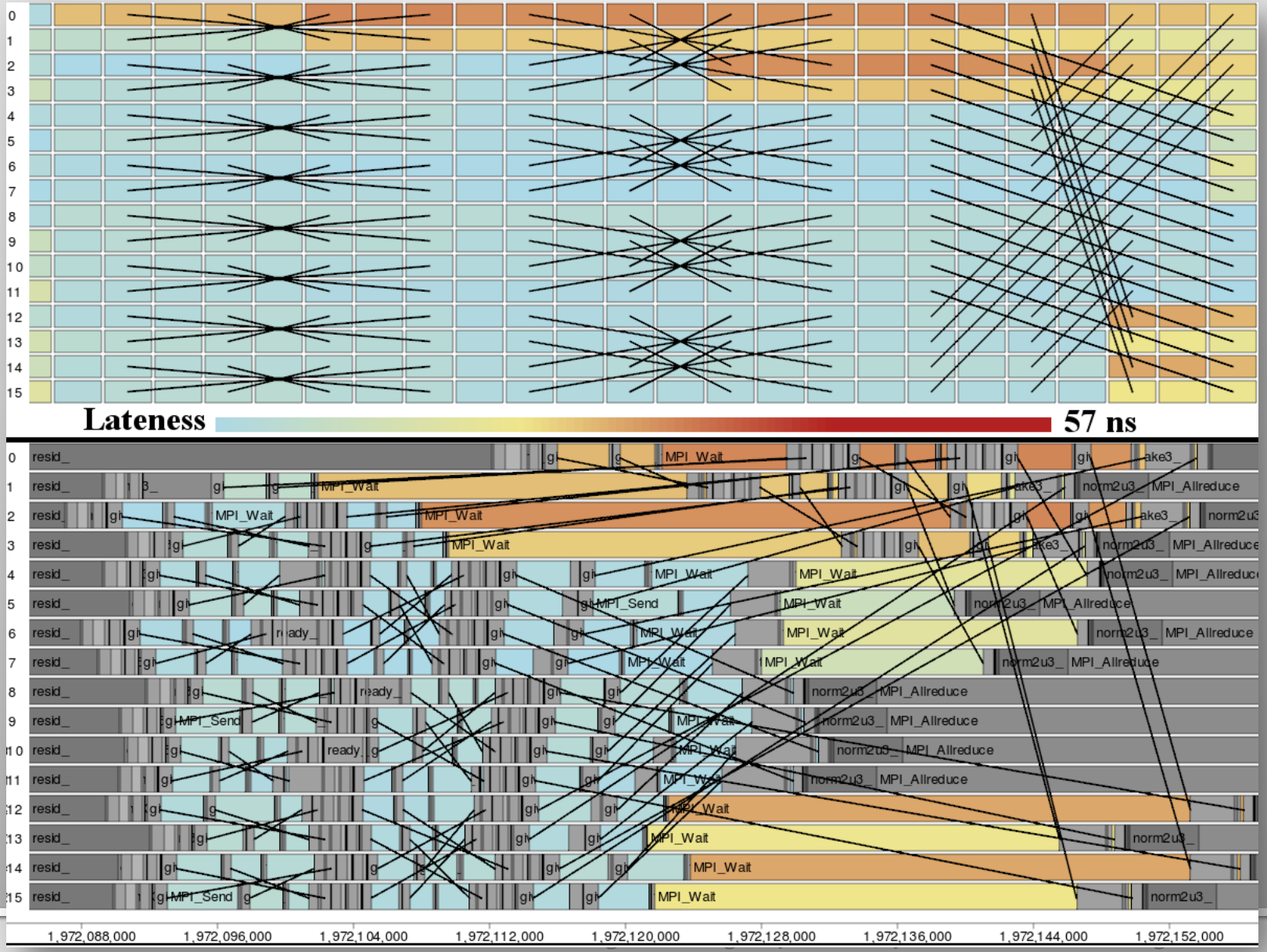
Message propagating lateness

Lateness Example: Allreduce



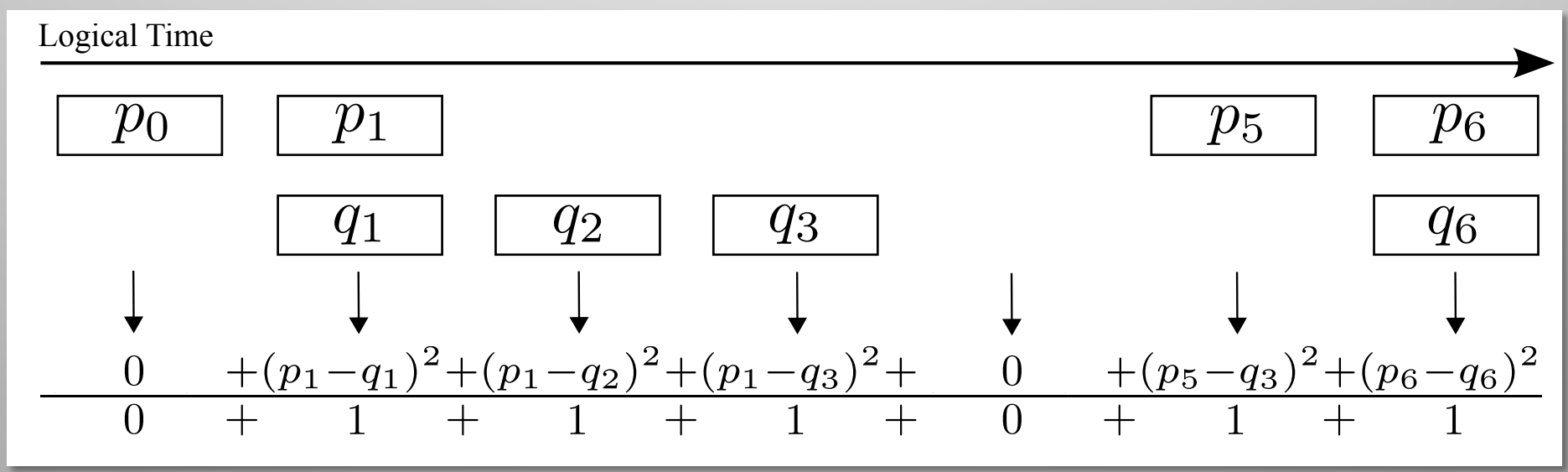
- **Ring based MPI_Allreduce using libNBC on 64 processes**
  - Logical time algorithm detected alignment of communication steps
  - Lateness spreads from process with rank 45

Lateness Example: MG on 16 processes

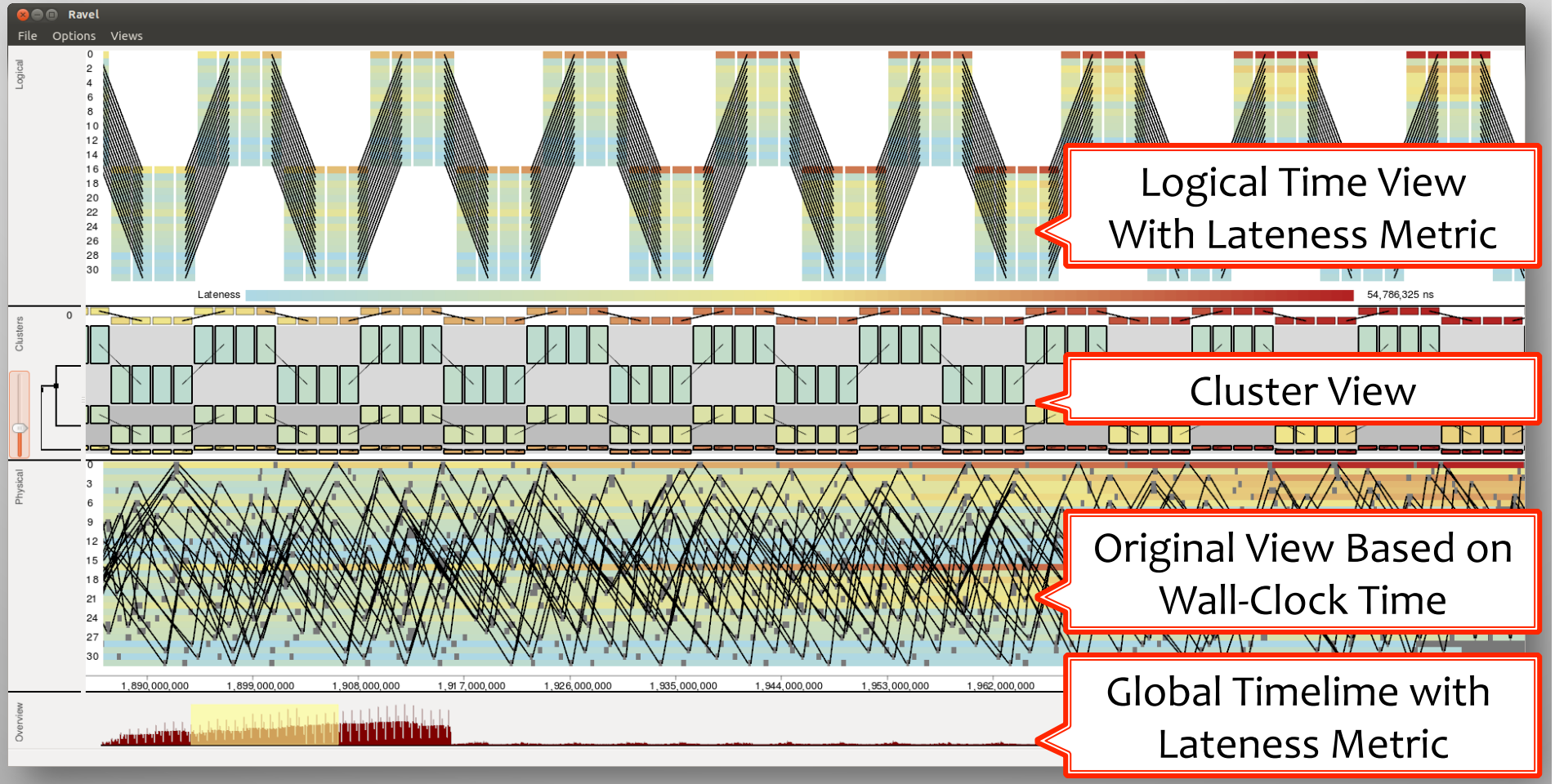


Clustering Processes with Similar Traces

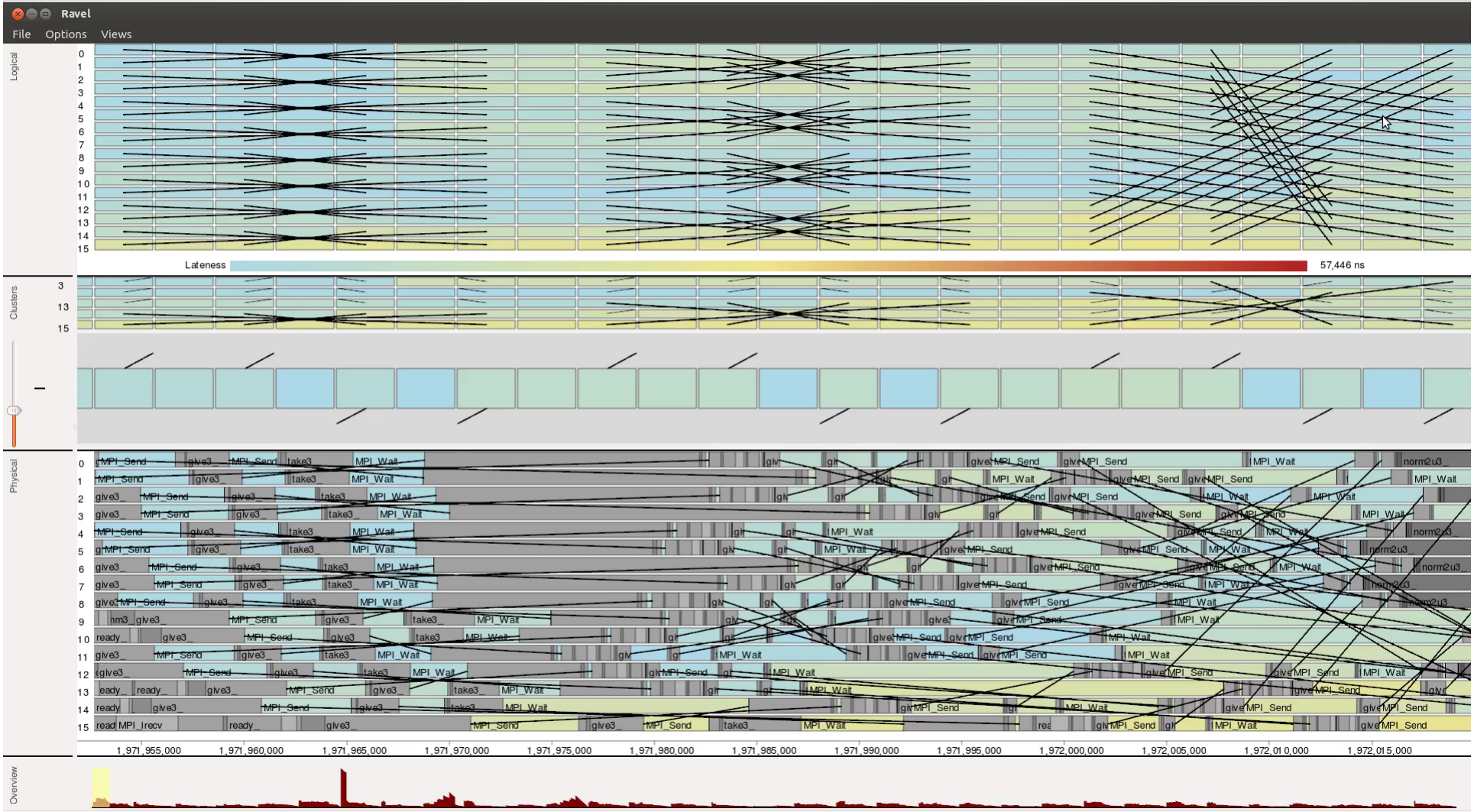
- **Challenge: Scalability**
  - The logical timeline view still uses 1 timeline per process
  - Logical timeline provides good base for clustering
- **Use lateness as distance metric**
  - Cluster between aligned events
  - Lateness is assumed to propagate where events are missing



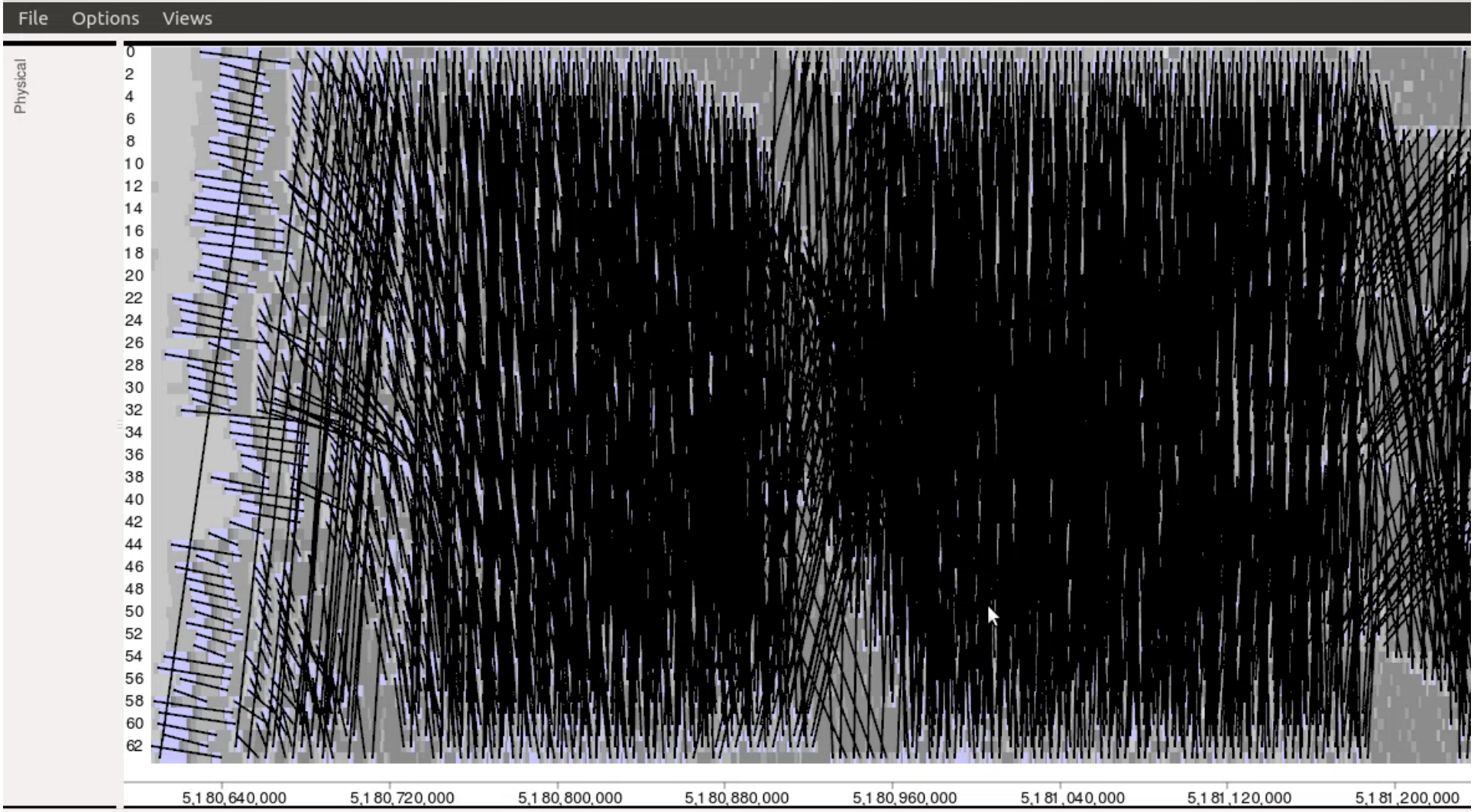
Ravel: Trace Visualization Using Logical Time



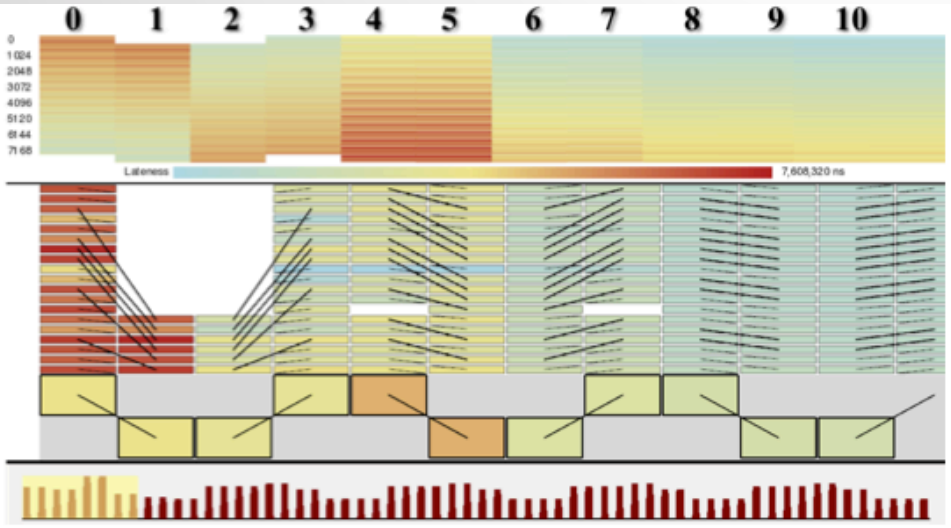
Interactive Usage of Ravel (on MG)



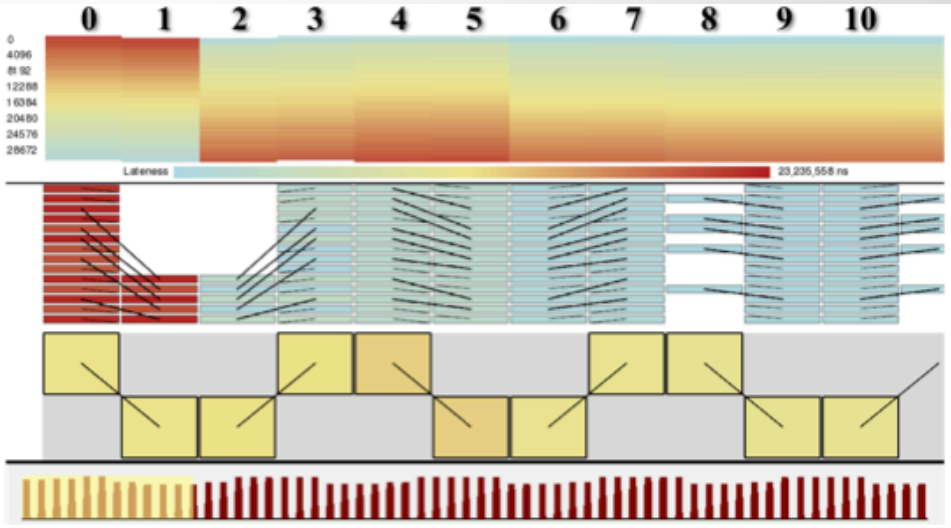
Case Study: MPI_Allreduce



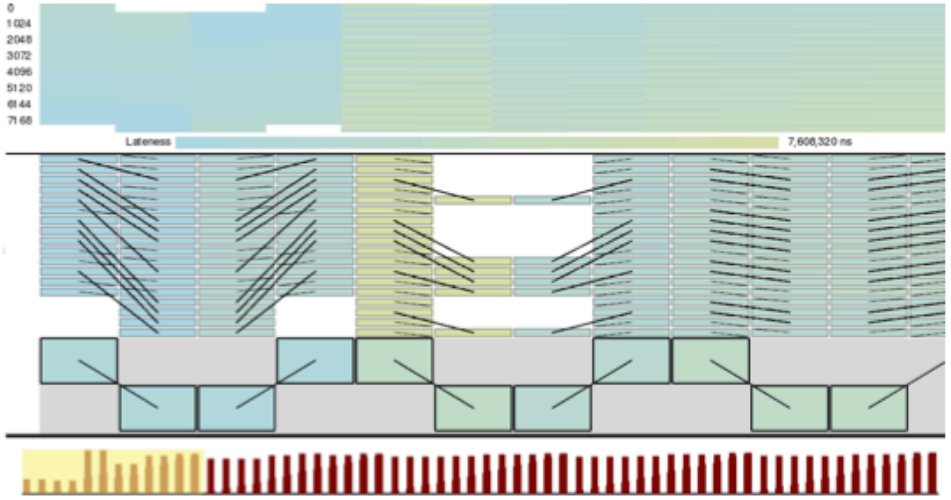
Case Study: LLNL Laser-Plasma Interaction Benchmark



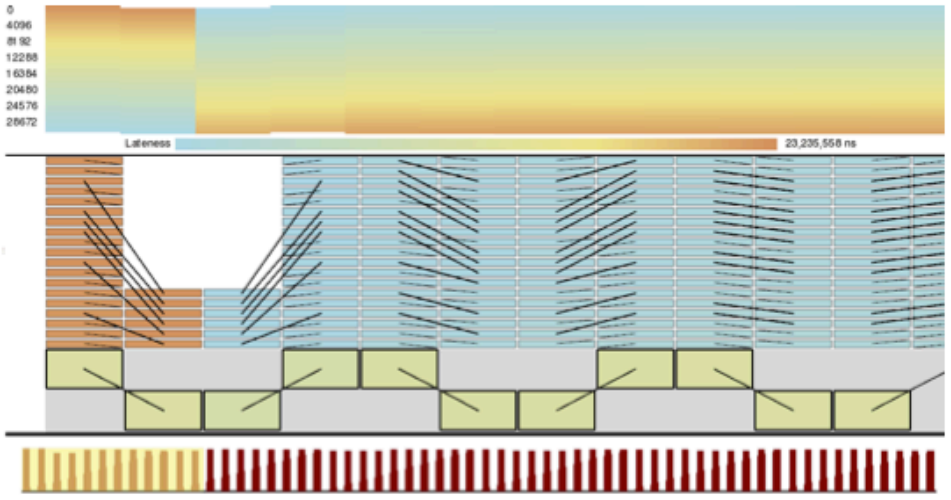
8,192 processes, default mapping.



32,768 processes, default mapping.

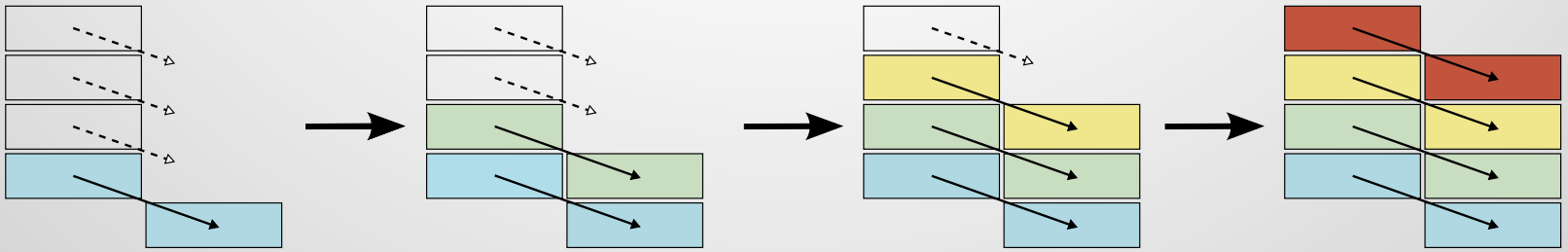


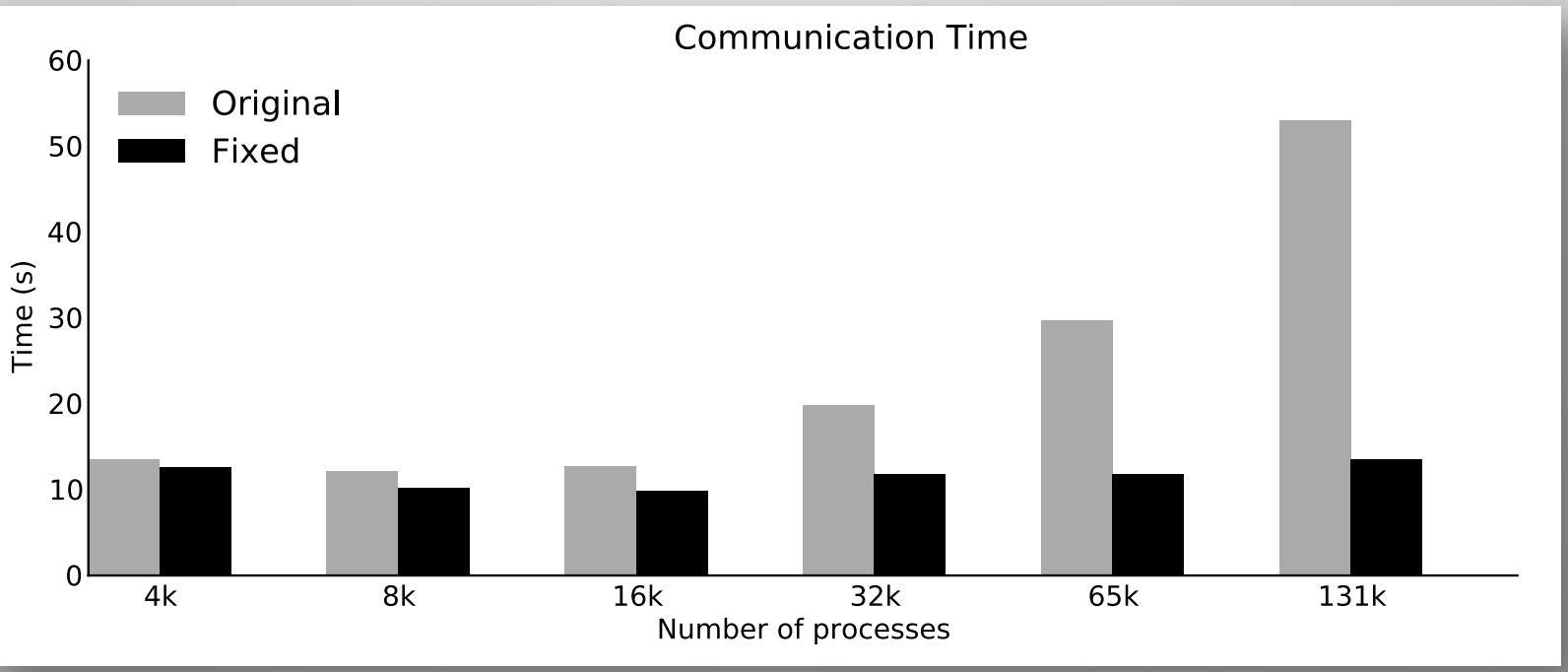
8,192 processes, round-robin mapping.



32,768 processes, round-robin mapping.

Optimizing the Laser-Plasma Code





Next Steps

- **Planned improvements**
  - Adaptation to newer (OTF2) trace generators
    - Working on Score-P support
  - Release of Ravel coming soon
- **Inclusion of higher level MPI information**
  - Communicators / subgroup communication
  - Improve automatic partitioning
  - Support for process re-ordering
- **Adaptation to other types of traces**
  - Arbitrary communication traces, e.g., from Charm++
  - Task dependency traces

Summary [Isaacs et al., IEEE InfoVis 2014]

Combing the Communication Hairball: Visualizing Parallel Execution Traces using Logical Time

Katherine E. Isaacs, Peer-Timo Bremer, Ilir Jusufi, Todd Gamblin, Abhinav Bhatele, Martin Schulz, and Bernd Hamann

Katherine E. Isaacs, Ilir Jusufi and Bernd Hamann are with the University of California, Davis. Emails: {keisaacs,jusufi,bhamann}@ucdavis.edu.
Peer-Timo Bremer, Todd Gamblin, Abhinav Bhatele, and Martin Schulz are with Lawrence Livermore National Laboratory. E-mails: {ptbremer,tgamblin,bhatele,schulzm}@llnl.gov.