Communication Assessment, Benchmarking, and Application Improvement

Presentation Coordinators:

Prof. Patrick Bridges and Prof. Puri Bangalore

Research Lightning Talks/Posters

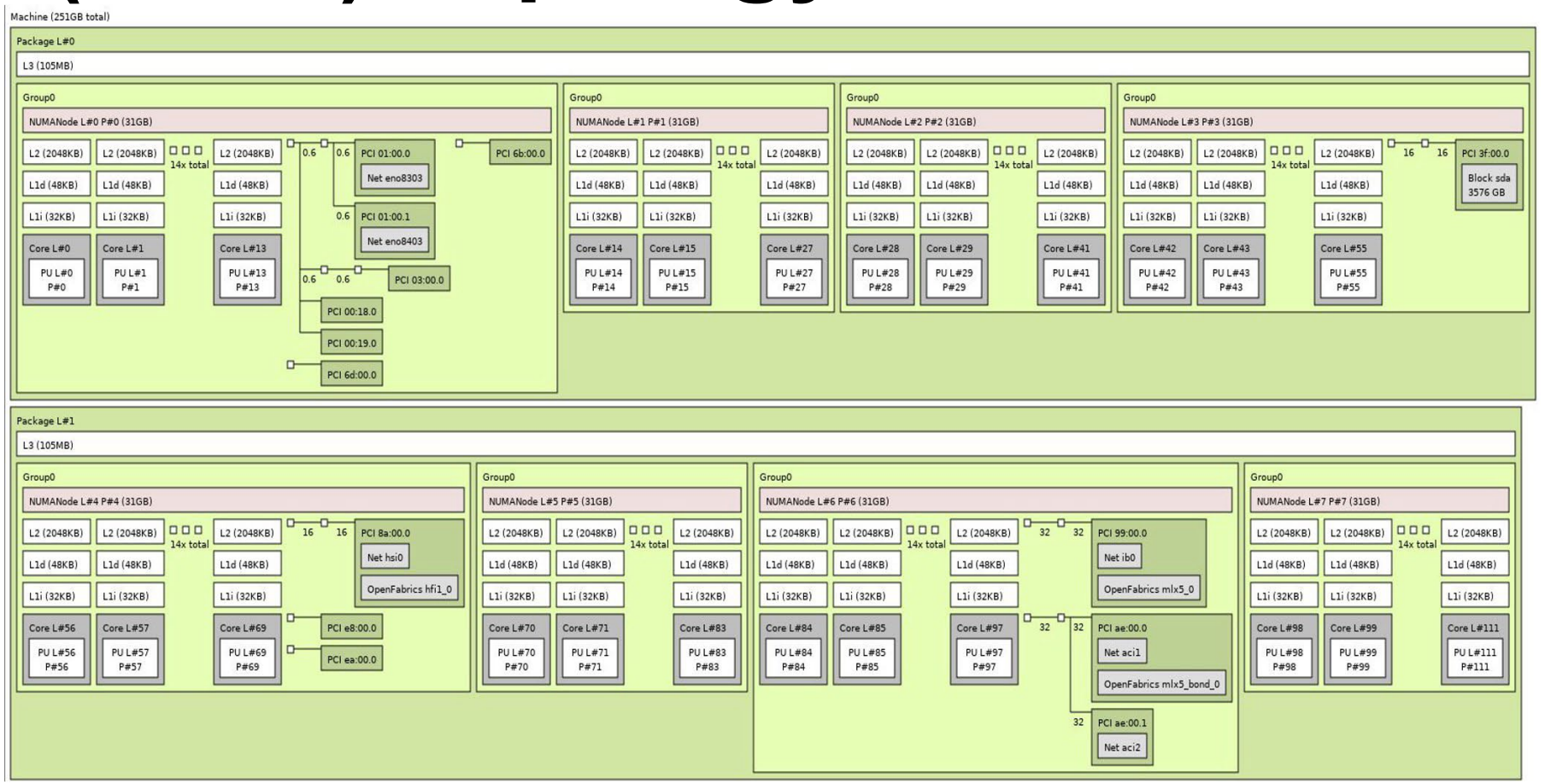
- Jackson Wesley, UNM: Understanding Data Movement on Many-Core Architectures
- Grace Nansamba, TN Tech: Extending Caliper for Greater MPI Performance Understanding
- Derek Schafer, UNM: Vernier: Generalizing Communication Pattern Tracing
- Nicholas Bacon, UNM: Analyzing and Predicting Irregular Communication Performance Improvements
- Jason Stewart, UNM: Beatnik: A Novel Global Communication Mini-Application
- Abdalaziz Raad, UNM: Communication Benchmarking using the Benchpark Experiment Management Framework
- Akhil Alasandagutti, UNM: Scaling Laws for the Workload Throughput of Emerging Heterogeneous Clusters

Understanding Data Movement on Many-Core Architectures

Jackson Wesley

CUP ECS
Center for Understandable, Performant Exascale Communication Systems
THE UNIVERSITY OF NEW MEXICO

Dane (LLNL) Topology

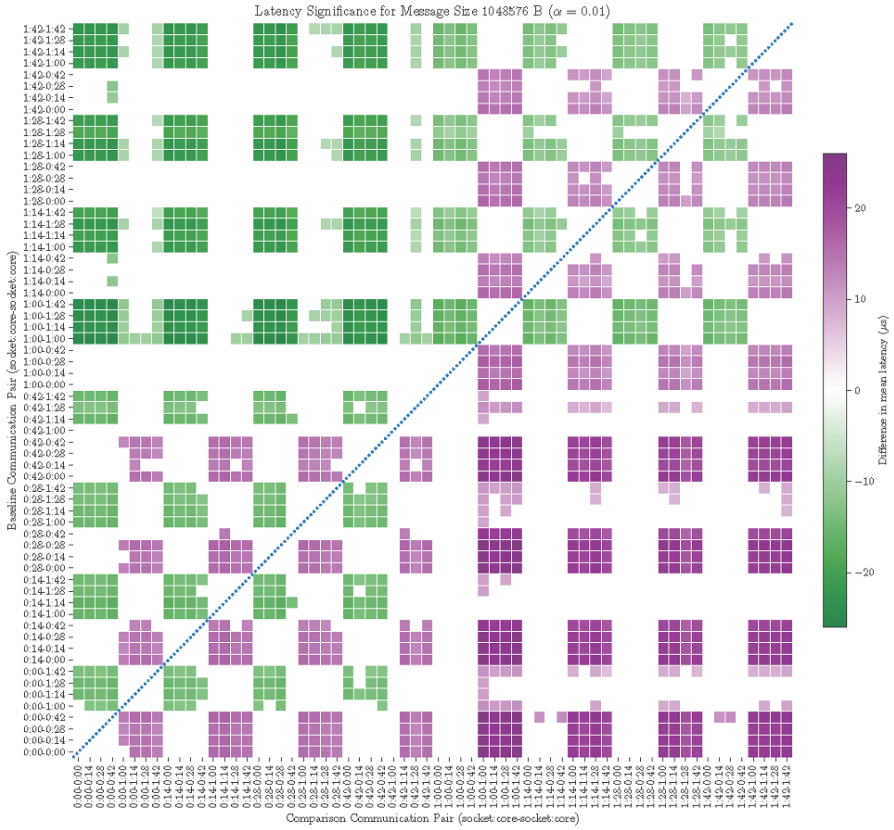


Methodology

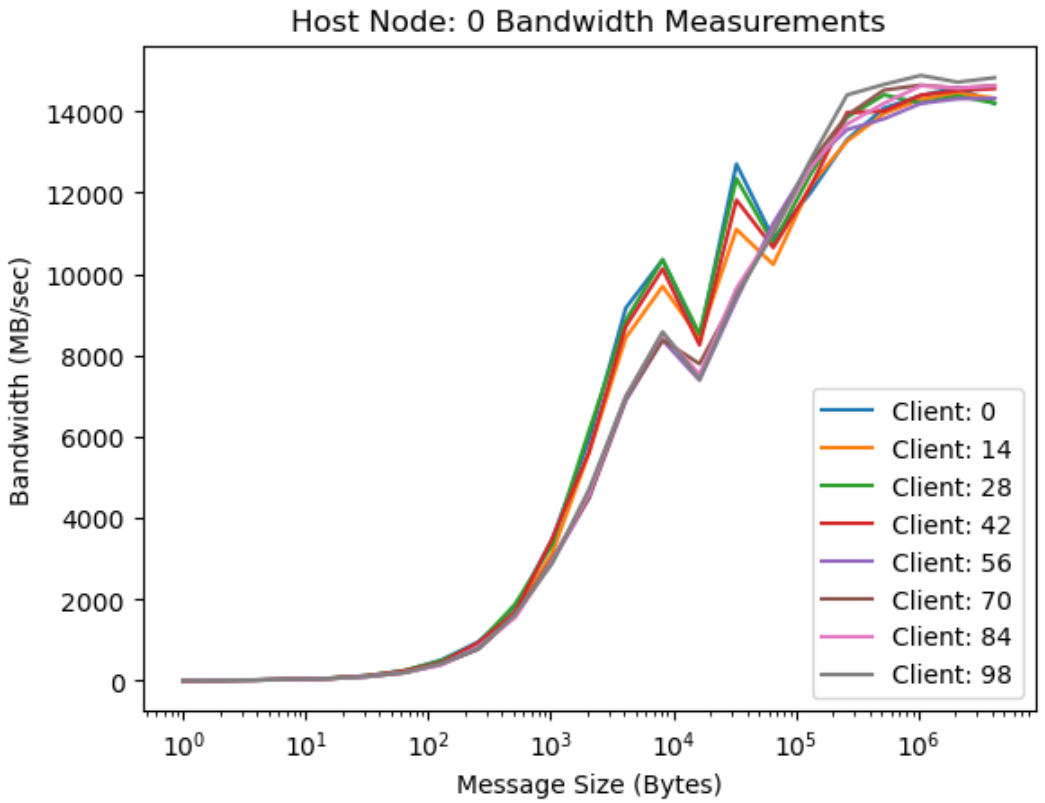
- Pin specific cores to MPI ranks and run OSU benchmarks between them

- **Main Research Question:** Are there performance differences dependent on the physical location of the sending/receiving processes?

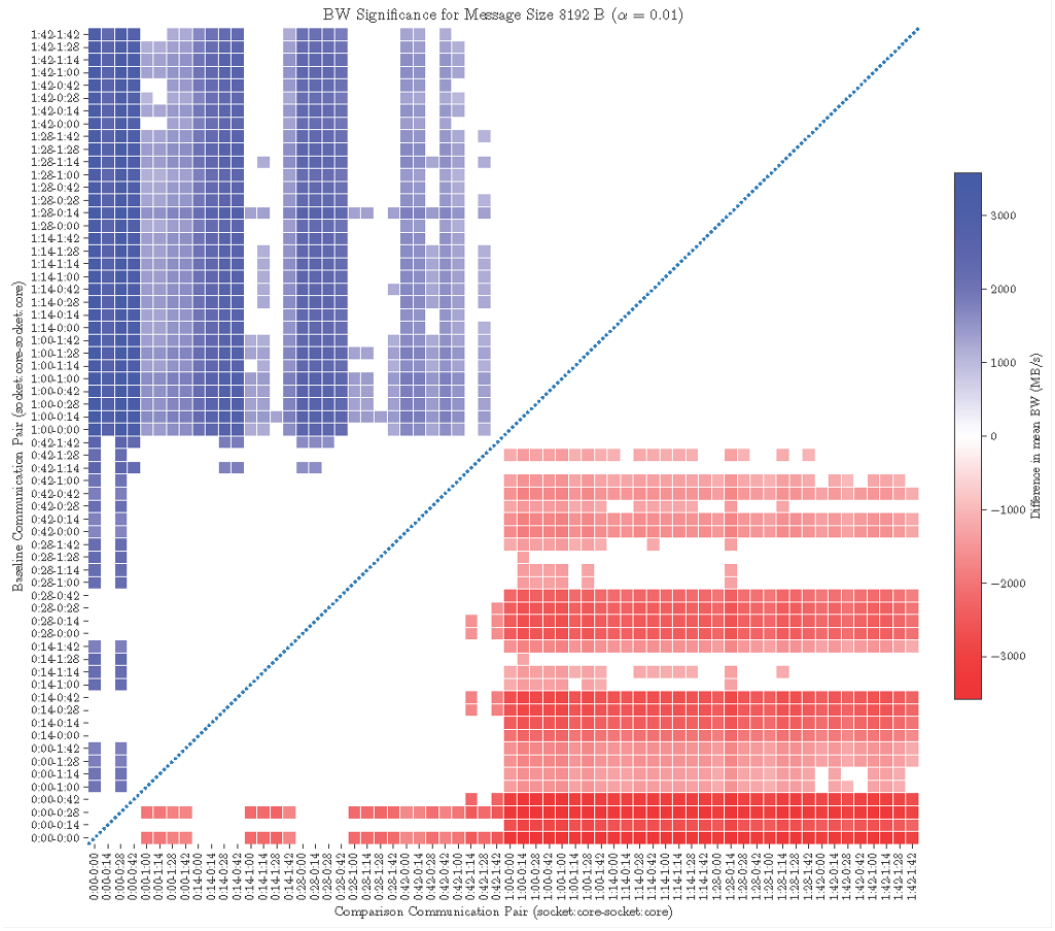
Latency of 1MB Messages



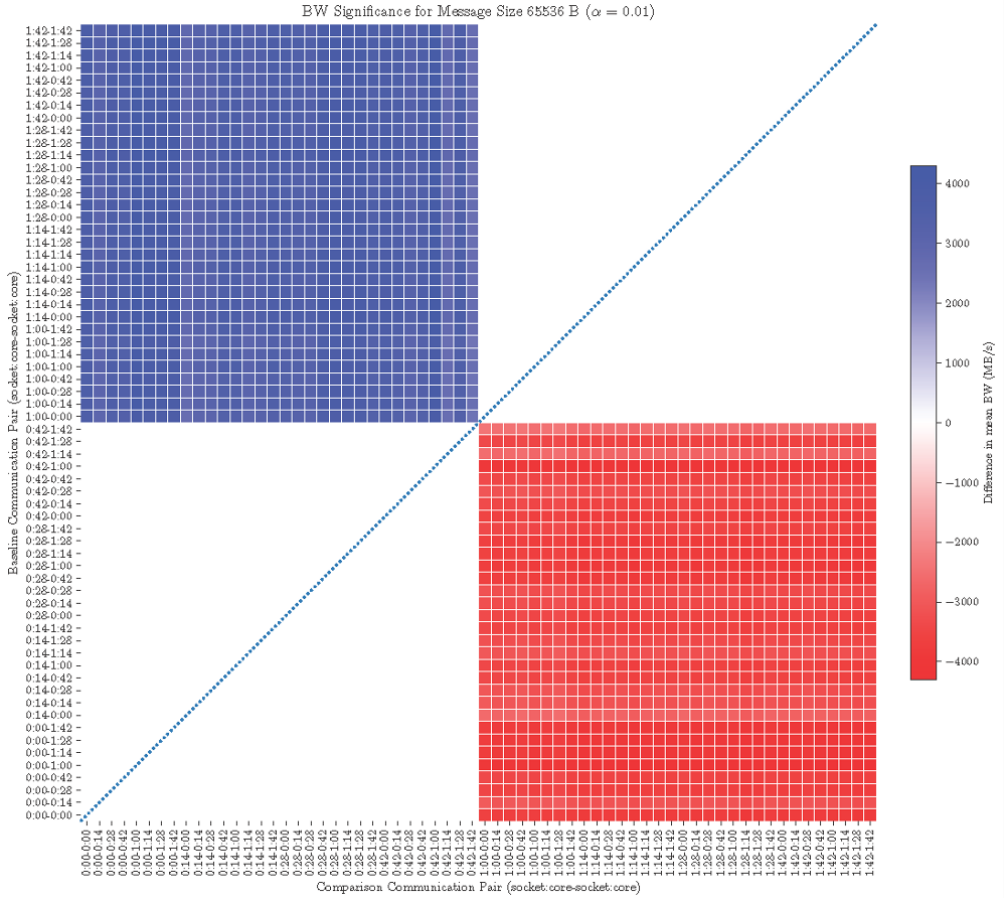
Bandwidth of Core 0, Socket 0, Node 0



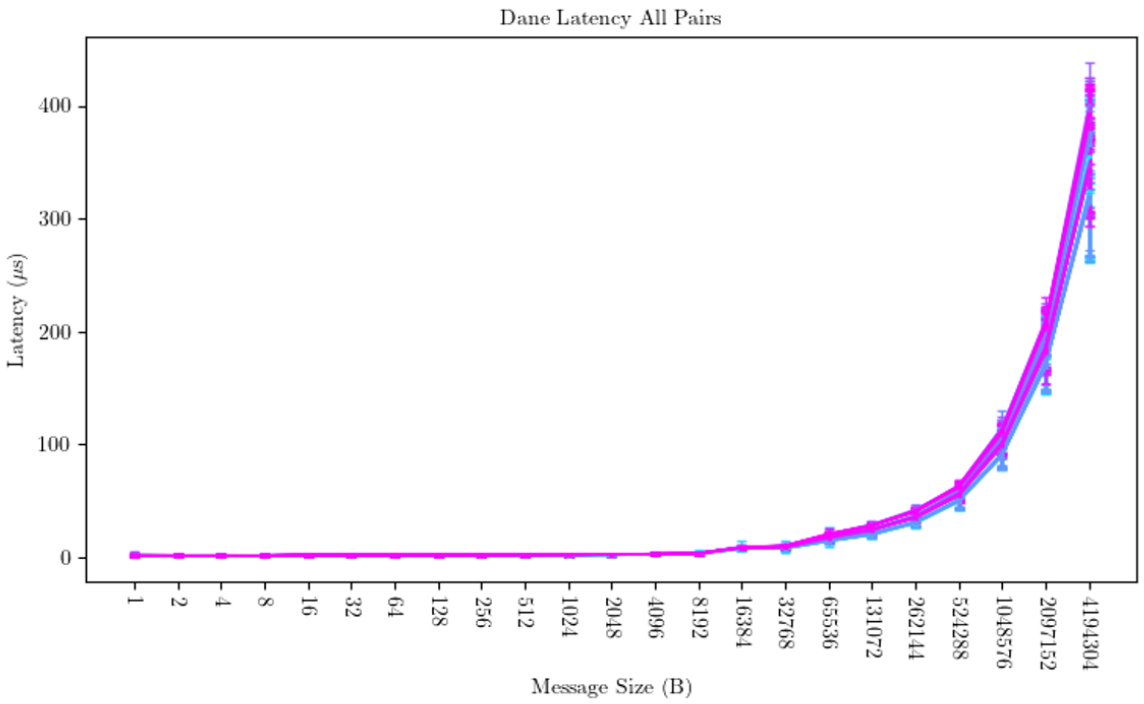
Bandwidth of 8KB Messages

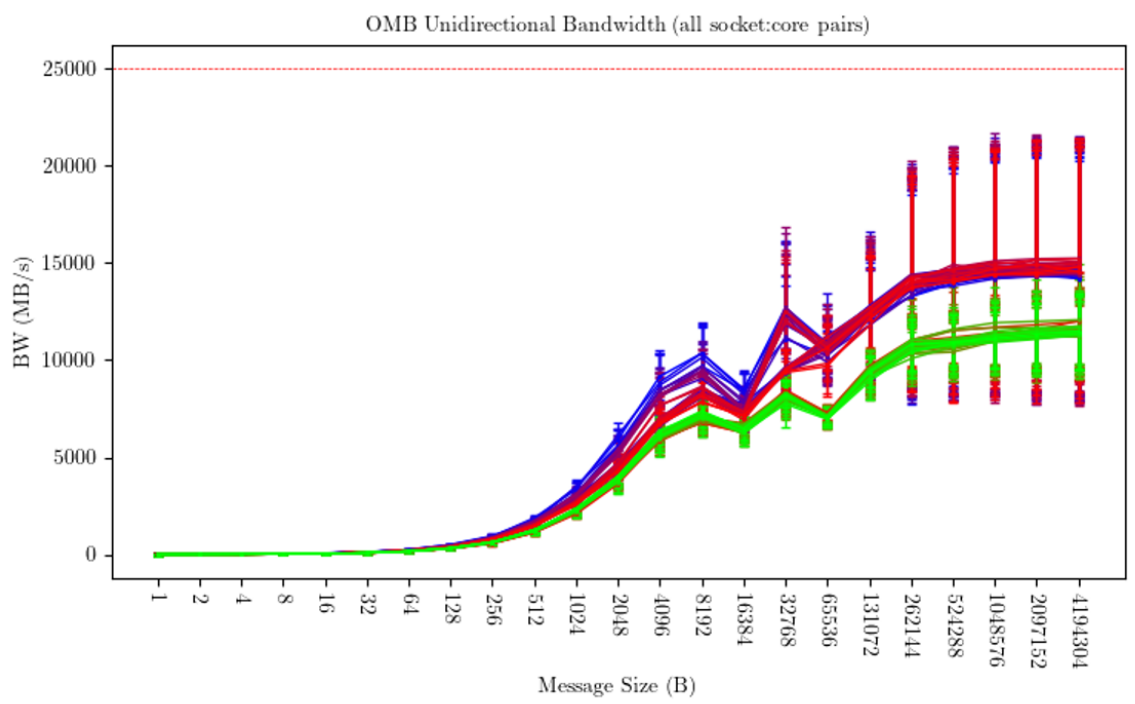


Bandwidth of 65KB Messages



All Pairs Comparisons





Leveraging Caliper and Benchpark to Analyze MPI Communication Patterns

Grace Nansamba, Evelyn Namugwanya, David Boehme[1], Olga Pearce[1], Riley Shipley, Derek Schafer, Anthony Skjellum

[1]Lawrence Livermore National Laboratory

Introduction

Motivation

- Understanding MPI communication patterns is crucial for optimizing HPC performance.
- Halo exchanges, reductions, and collectives dominate communication overhead in many applications.

Approach

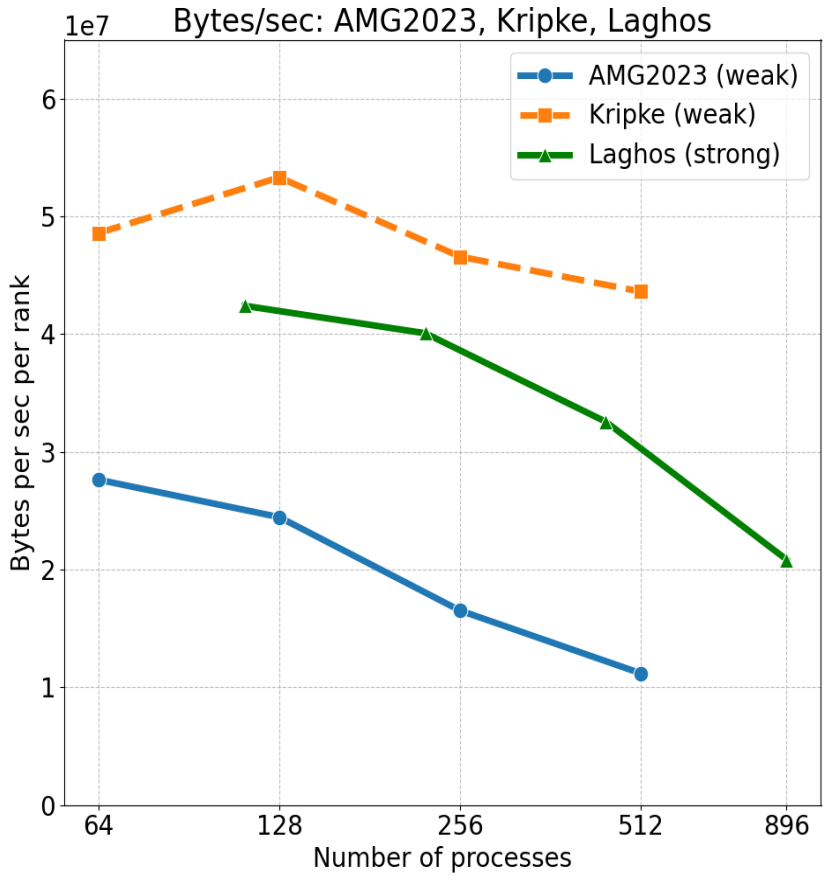
- Extended Caliper with new *'special regions'* and used the modifier in Benchpark benchmarks (AMG2023, Kripke, Laghos).
- Annotated communication regions (e.g., halo_exchange, MatVecComm, sweepComm) using CALI_MARK_COMM macros.
- Captured metrics such as sends, receives, message sizes, source/destination ranks, and Irecv-gap.

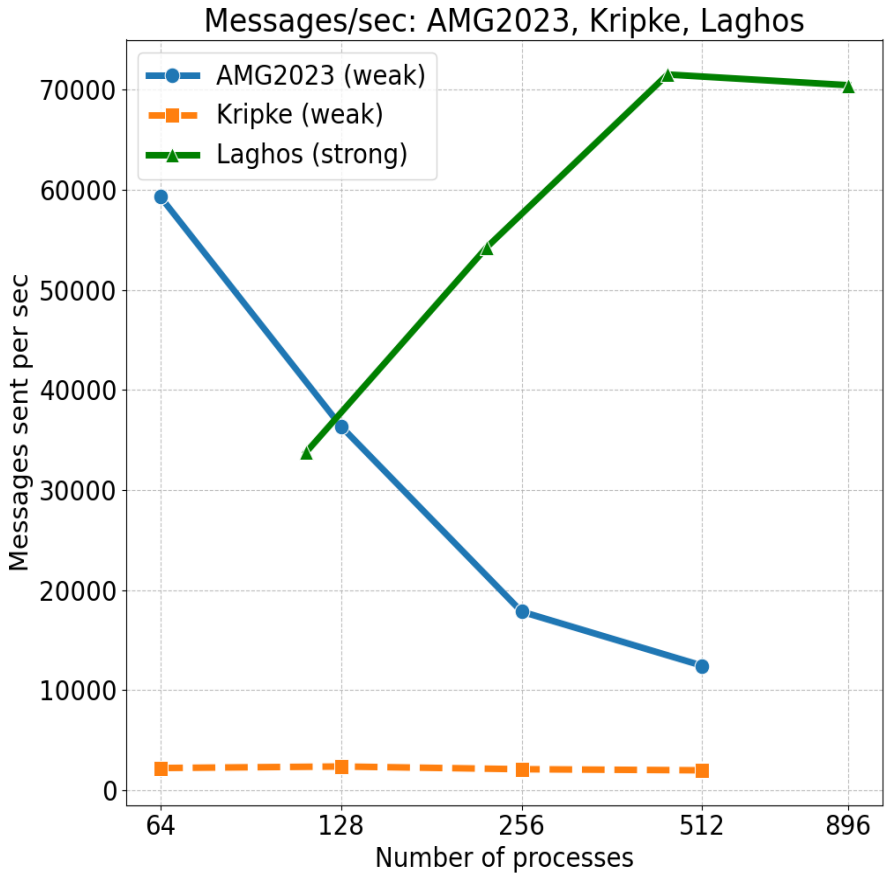
Tools: Caliper, Benchpark, Thicket

Applications: AMG2023, Kripke, Laghos

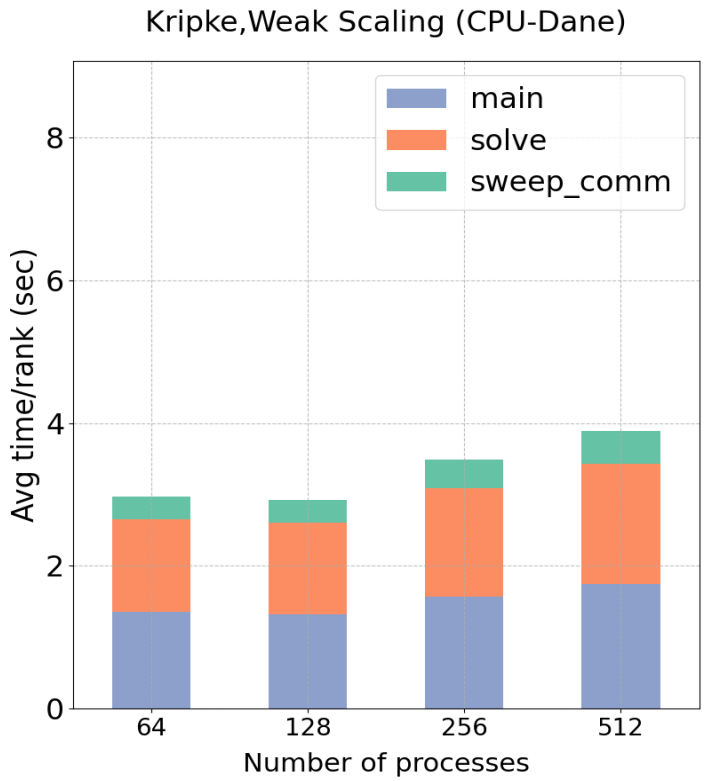
Systems: Dane and Tioga

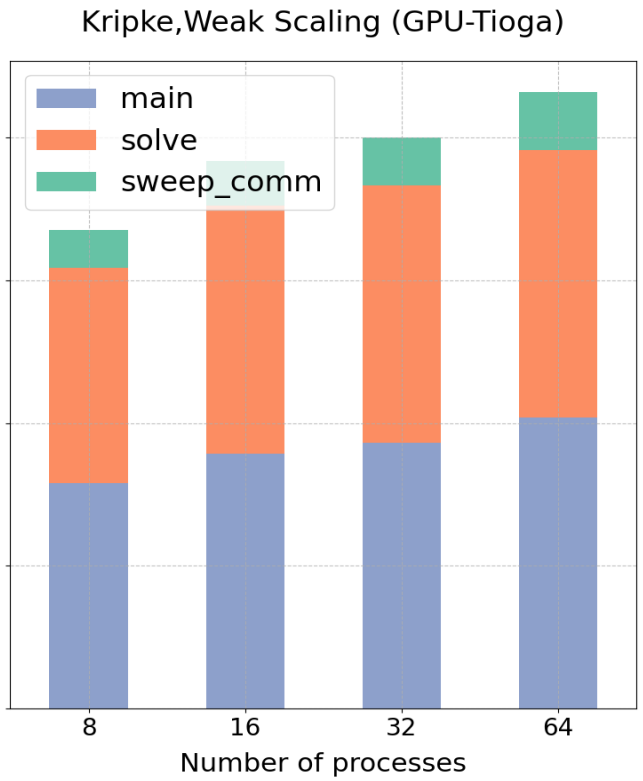
Bandwidth and Message Rates



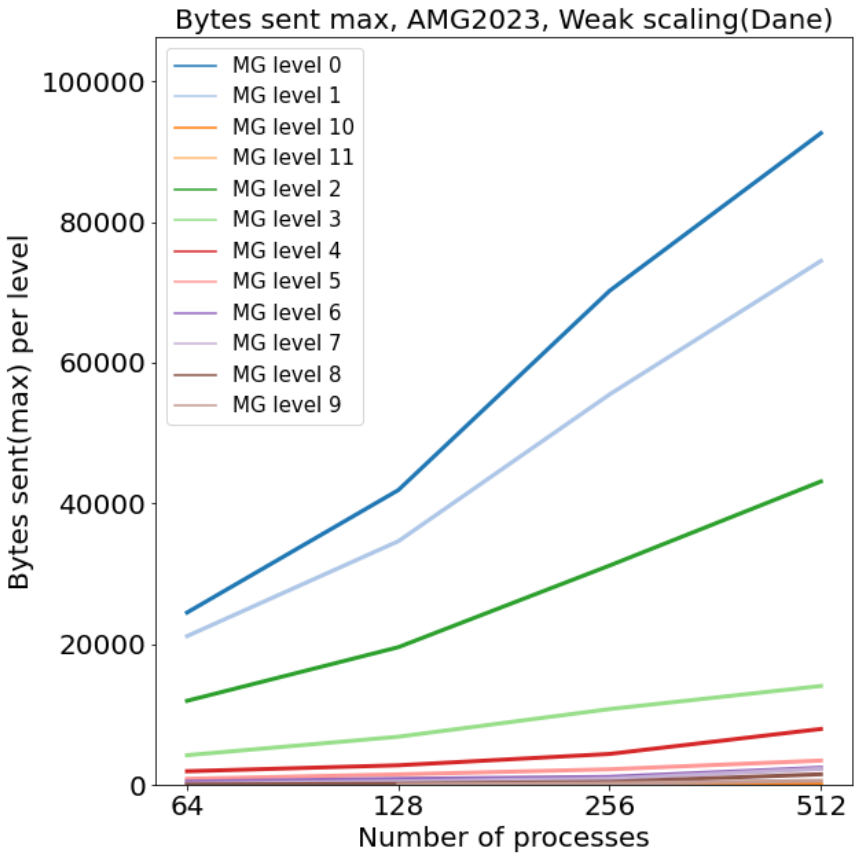


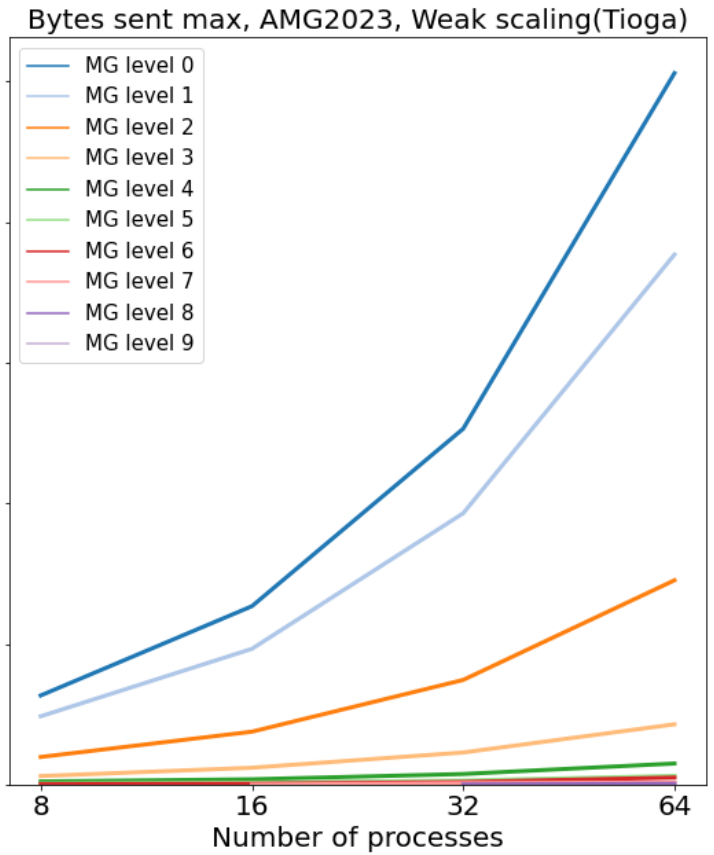
Average time per MPI process in regions





Bytes transferred vs. AMG level





Conclusion

- Extended Caliper with special regions to isolate and profile MPI communication patterns.
- Analyzed AMG2023, Kripke, and Laghos on CPU (Dane) and GPU (Tioga) systems using Caliper, Benchpark, and Thicket.
- Observed distinct communication behaviors across AMG multigrid levels, Kripke particle sweeps, and Laghos mesh updates.
- New insights are possible from the *special region* instrumentation.
- Introduced the Irecv-gap metric to measure delays in non-blocking receives, aiding overlap analysis.

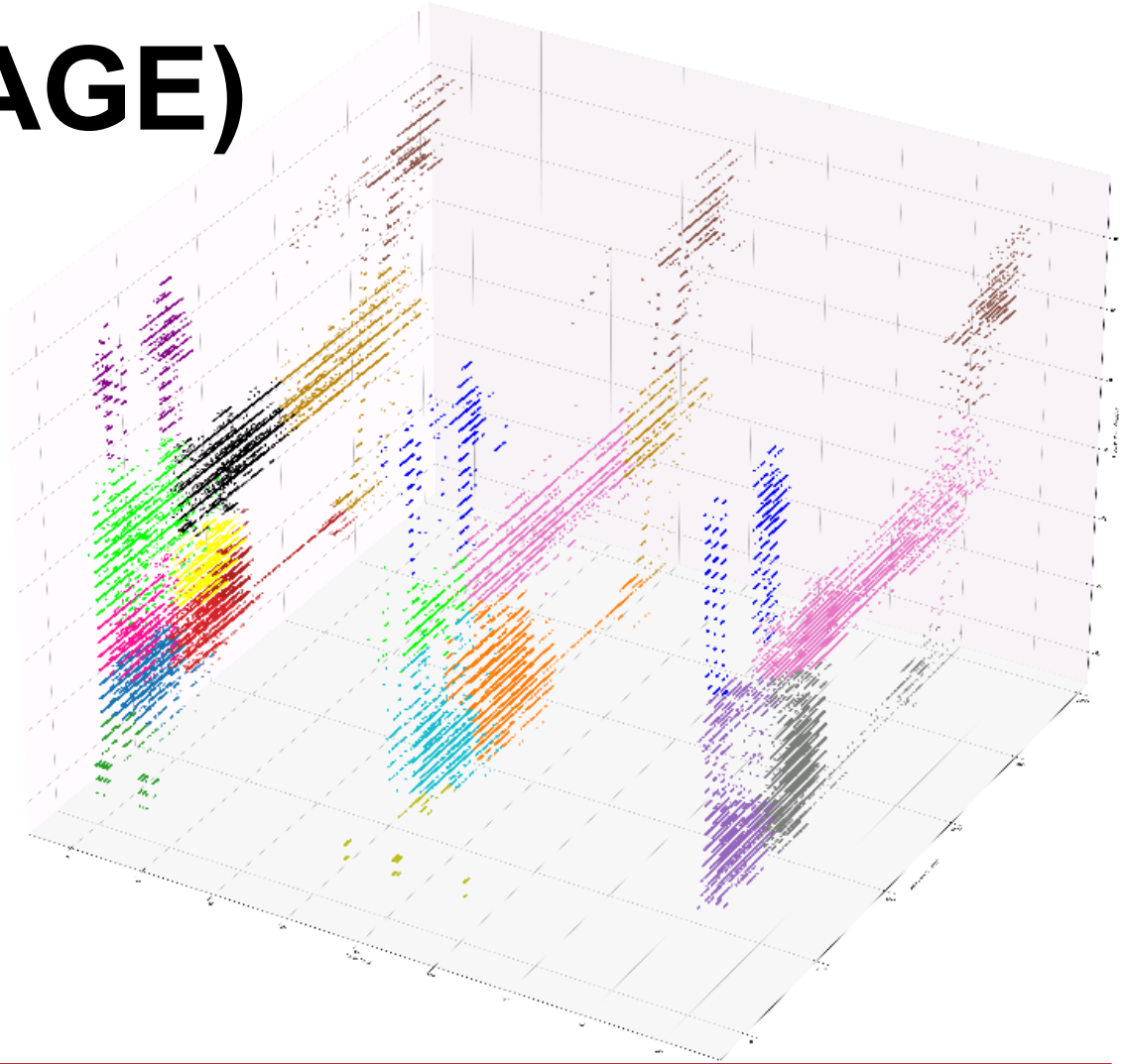
Vernier: Generalizing Communication Pattern Tracing

Derek Schafer, Riley Shipley,

Patrick Bridges

First Approach (in xRAGE)

- Added manual annotations to xRAGE
  - In both the communication setup and communication execution phases
  - Sample output:
    - Rank 0: 0|B1:1-200,T2:53,F4:90|1.23|T0
    - Rank 1: 0|B0:200-1,B2:5-40|0.97|T0
- Produced visualizations like the one on the right
  - Number of exchanges
  - Size of exchanges
  - Number of ranks in exchanges
- Wanted to generalize approach to be used in other applications



Introducing Vernier

- Goal: Understand the characteristics of specific application-defined communication patterns and see how they change as the application runs.
- By tracing the exact pattern and capturing key metrics, we can focus on understanding and optimizing communication without also having to worry about input decks or application specific roadblocks
- Unique technical features of Vernier lies in the key details captured within each pattern:
  - What MPI calls are in the pattern (and which MPI calls are captured can be tweaked at runtime)
  - Size of messages
  - To/from ranks, communicators involved
- With this data, we can build tools to help understand the patterns characteristics
- Caliper captures similar data, why not use Caliper?
  - Caliper provides data in aggregate, even at a “per rank” level
  - Vernier can provide the same macro calls as Caliper for simpler* switching.

Vernier Example Data Capture

Code Sample

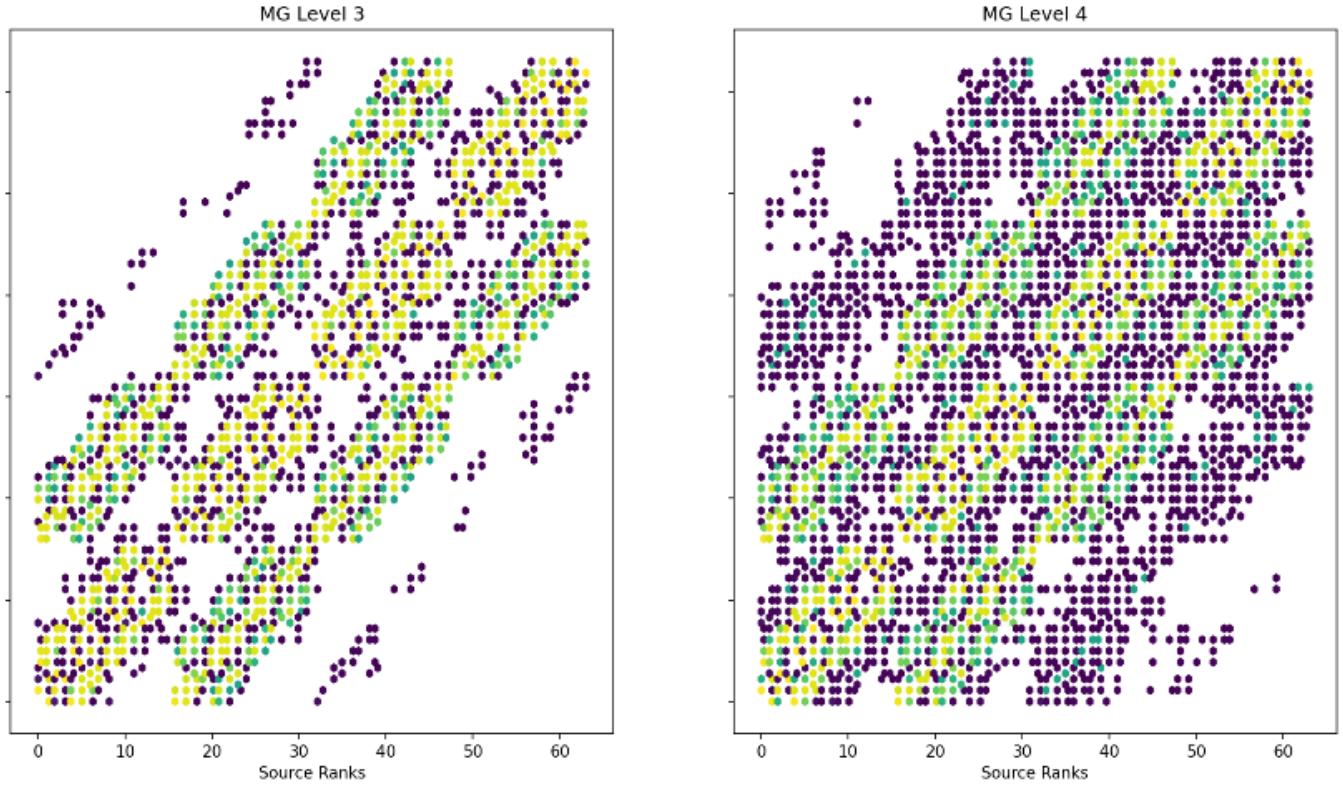
```
CALI_MARK_COMM_REGION_BEGIN("Hello World");
MPI_Comm_split(MPI_COMM_WORLD, …, &new_comm);
MPI_Type_contiguous(4000, MPI_BYTE, &c_type);
MPI_Request request;
MPI_Irecv(…, &request);
CALI_MARK_COMM_REGION_BEGIN("Nested pattern");
MPI_Send(…, new_comm);
CALI_MARK_COMM_REGION_END("Nested pattern");
MPI_Wait(&request, …);
CALI_MARK_COMM_REGION_END("Hello World");
```

Vernier Output

- Rank 0:
  - Hello World:0,Nested pattern:1
  - 2|1{0}
  - 0|IR:1:0:4000:0,*1*,W:0|114.559
  - <1>1|S:1:0:4000|17.874
- Rank 1
  - Hello World:0,Nested pattern:1
  - 2|1{1}
  - 0|IR:1:0:4000:0,*1*,W:0|110.421
  - <1>1|S:1:0:4000|22.443

What's Next?

- Create visualizations of the communication patterns
  - Communication intensity between process pairs
  - Irregular pattern communication clustering
- Tie-in opportunities with other CUP-ECS projects:
  - Visualize how many messages are crossing boundaries for potential aggregation opportunities
  - Look to see if any of our custom neighborhood collectives make any sense to be integrated
  - Determine distribution of key communication metrics for a given pattern, feed to irregular communication benchmark
  - Feed captured patterns to SST
- Working on a scalable version that focuses on calculating distributions instead of tracing
  - In collaboration with Jered Dominguez Trujillo, LANL staff and former CUP-ECS student)
  - Goal is to integrate into Caliper



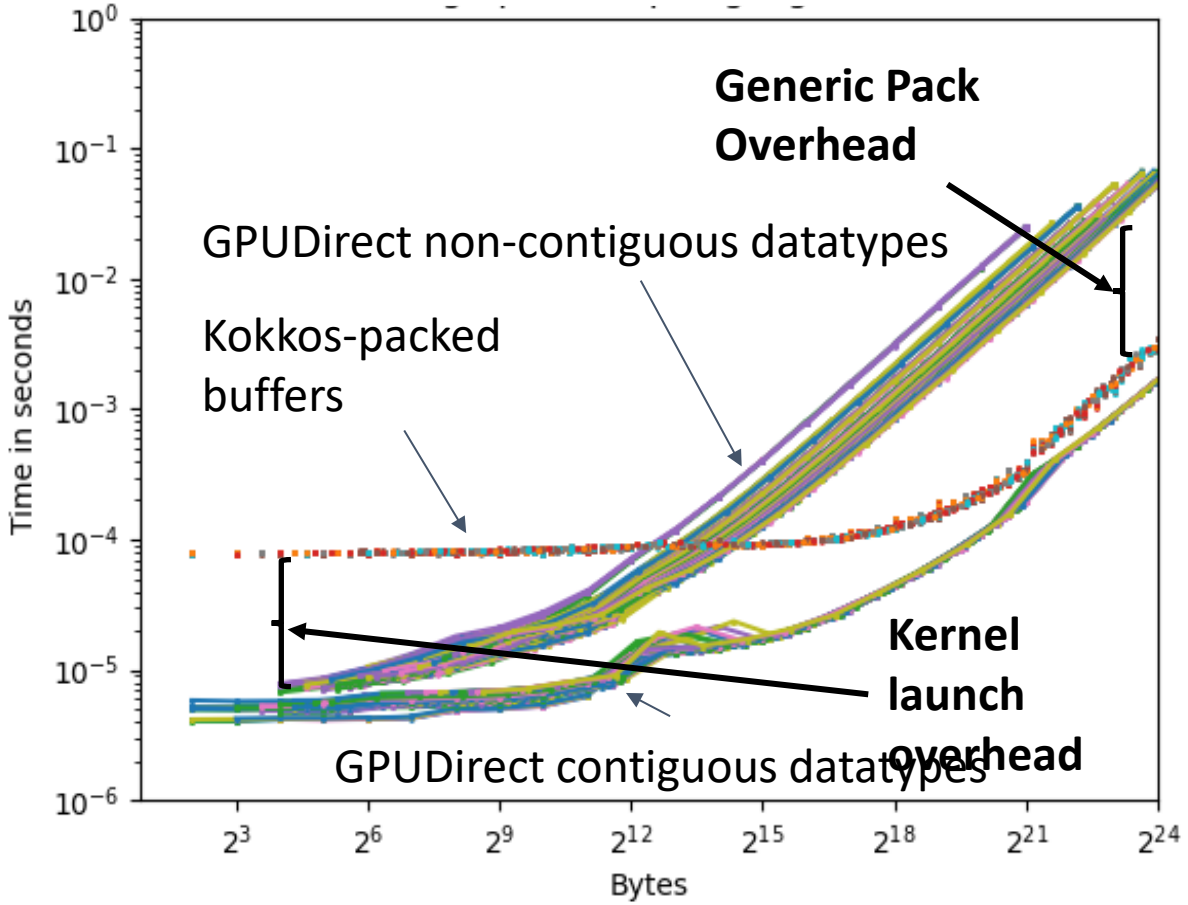
Combined AMG2023 Communication Intensity per AMG Level

Analyzing and Predicting Irregular Communication Performance Improvements

Nicholas Bacon, Patrick Bridges,
Scott Levy[1] and Kurt Ferreira[1]
[1]Sandia National Laboratories

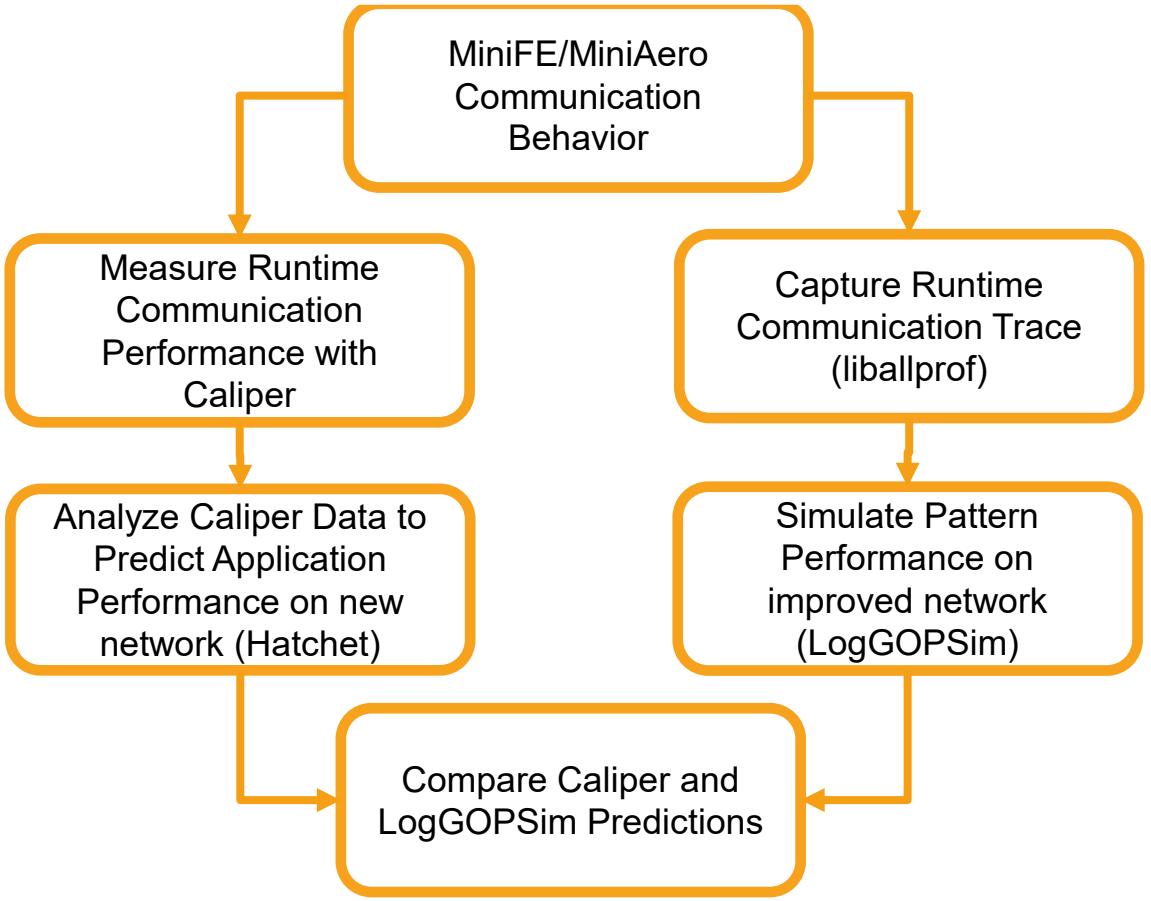
Analyzing and Modeling Costs of Irregular Data Movement

- Irregular communication uses complex data layouts and are difficult
  - Generic datatypes can be faster for small irregular data
  - Generic datatypes prohibitively slow for large irregular data
- Measured irregular data type handling (figure at right)
  - The improvements to the right were not reproducible (mvapich bugs!).
  - The slowdowns to the right were.
- Also modeled performance of irregular communication using LogGOP
- Simple performance studies showed benefits of complex APIs for this not worth it.
  - Key remains getting rid of unnecessary synchronization for datatype handling
  - Stream and kernel triggering of communication is the right way to do this, not convoluted APIs

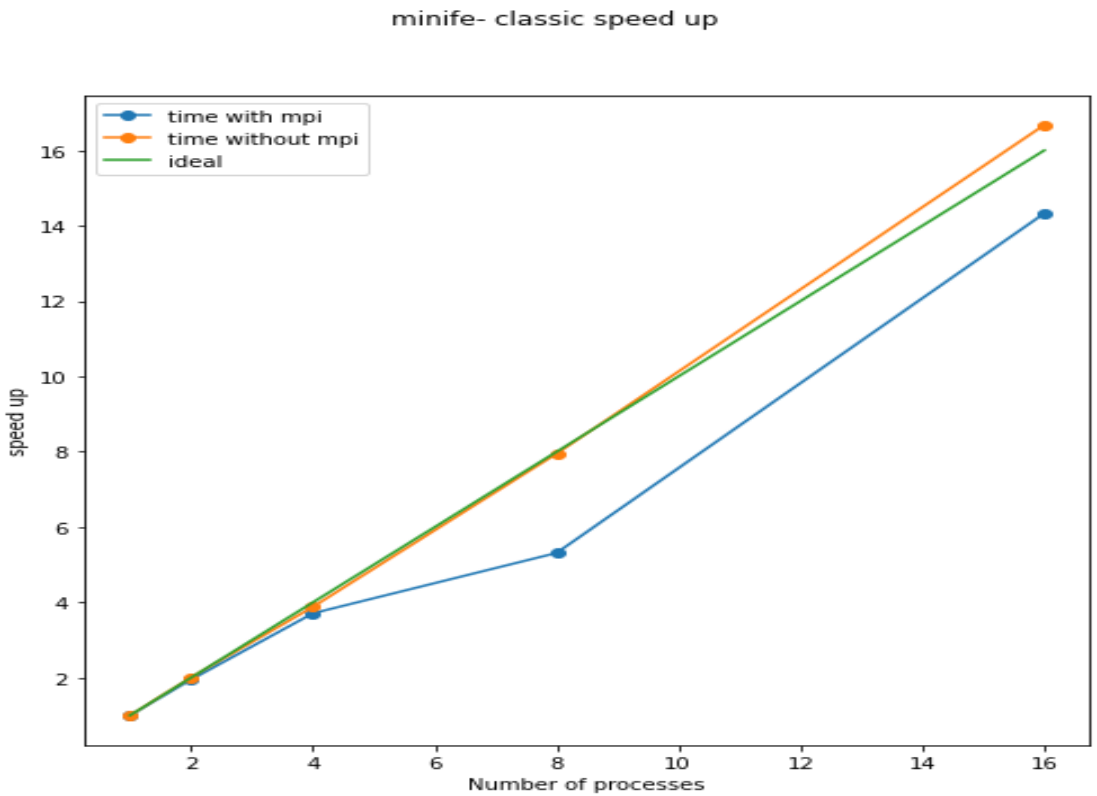


Using Caliper/Hatchet and LogGOPSim to Predict Network Speedup

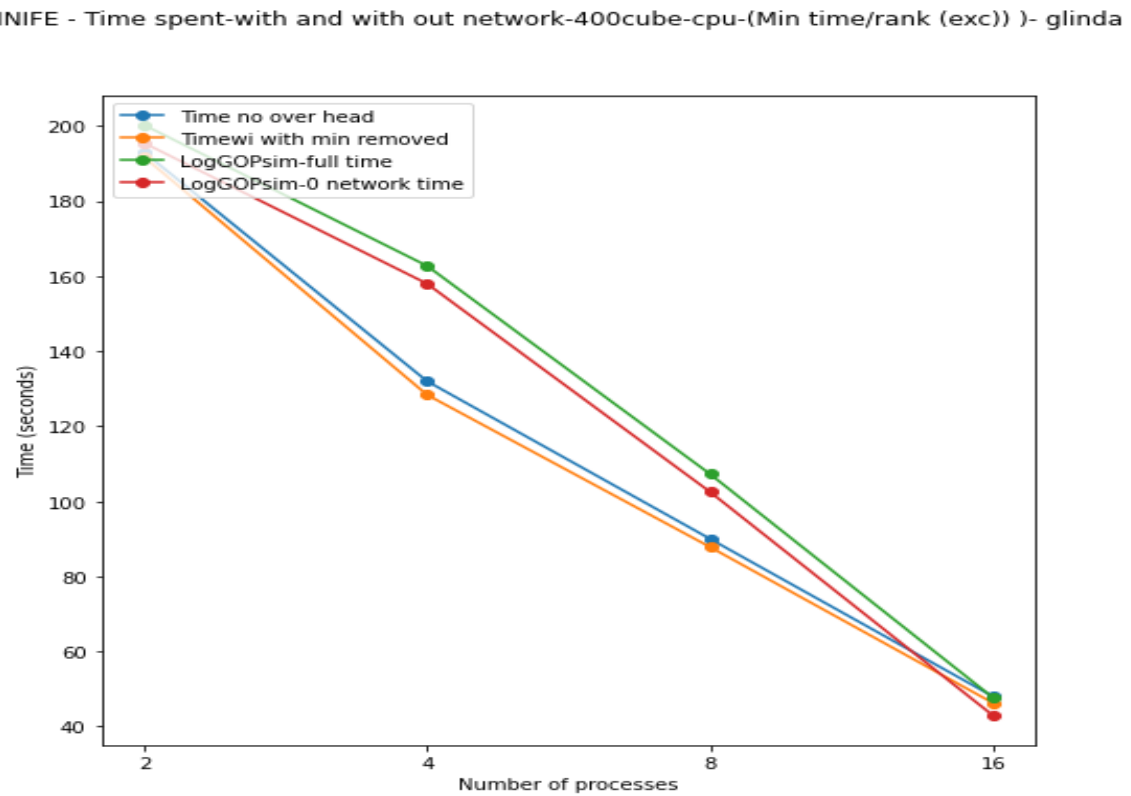
- Goal: Understand how accurately simple Caliper/Hatchet communication analyses can predict the impact of communication optimizations
- Simple hypothetical optimization: predict performance gain of infinitely fast network.
- Proposed methods for estimating application performance improvement:
  - LogGOPSim+liballprof (baseline)
  - Caliper+Hatchet remove MPI average (all functions)
  - Caliper+Hatchet remove MPI average but add back the min time per call
  - Caliper+Hatchet remove MPI min time per call



Results of Caliper/Hatchet and LogGOPSim Comparison



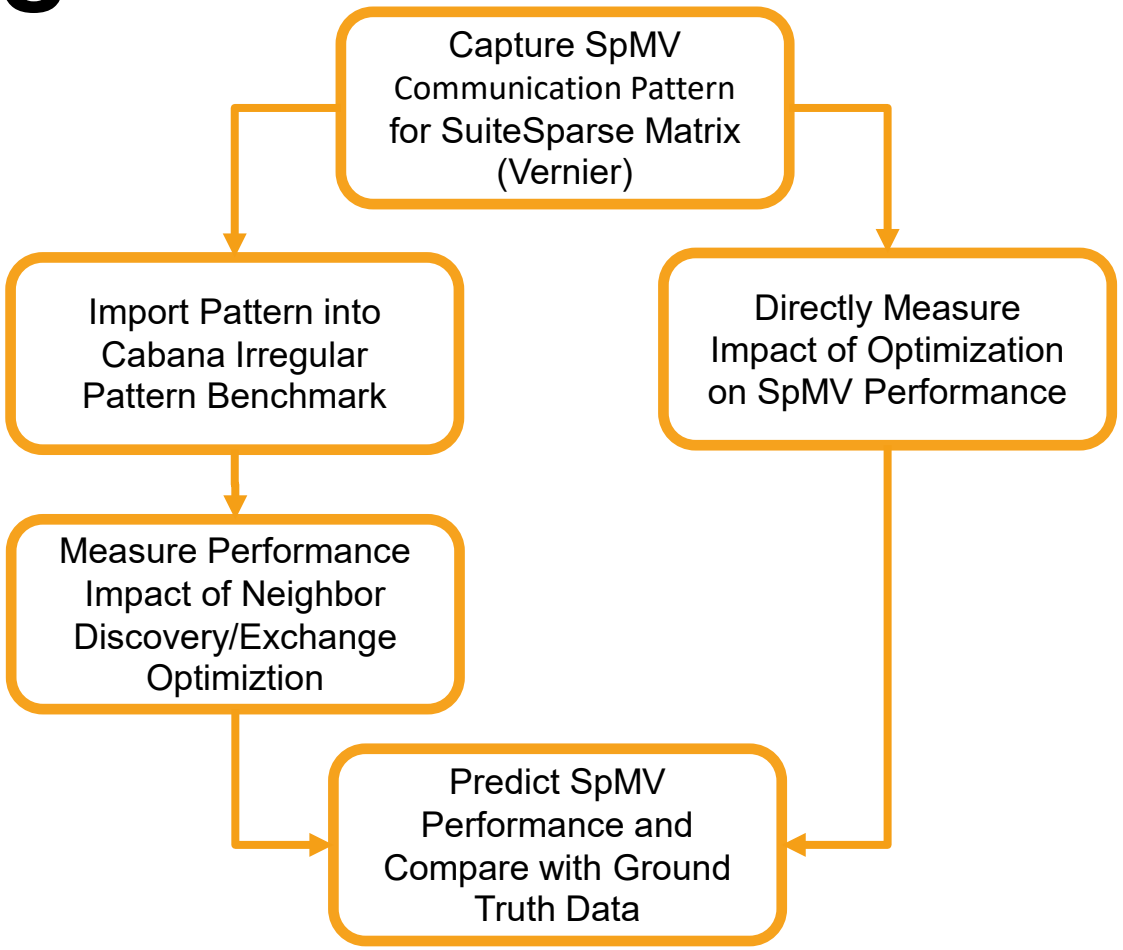
Initial Caliper prediction of MiniFE performance with infinite bandwidth network



Comparison of Caliper and LogGOPSim MiniFE performance predictions

Predicting Optimization Impact on Production Applications

- Challenges with Caliper and LogGOPSim comparisons caused us to change target optimization and approach
- Revised workflow (shown at right) focuses in neighbor discovery and neighbor exchange optimizations in Sparse Matrix/Vector multiplication (SpMV) benchmark
- Created Cabana irregular pattern benchmark to be able to reply Vernier-captured irregular communication patterns
- Currently developing analyses for Vernier data captures to drive irregular pattern benchmark and analyze results
- Goal is to predict SpMV performance improvement and extend this to MiniFE, AMG2023, MueLU, and MiniEM

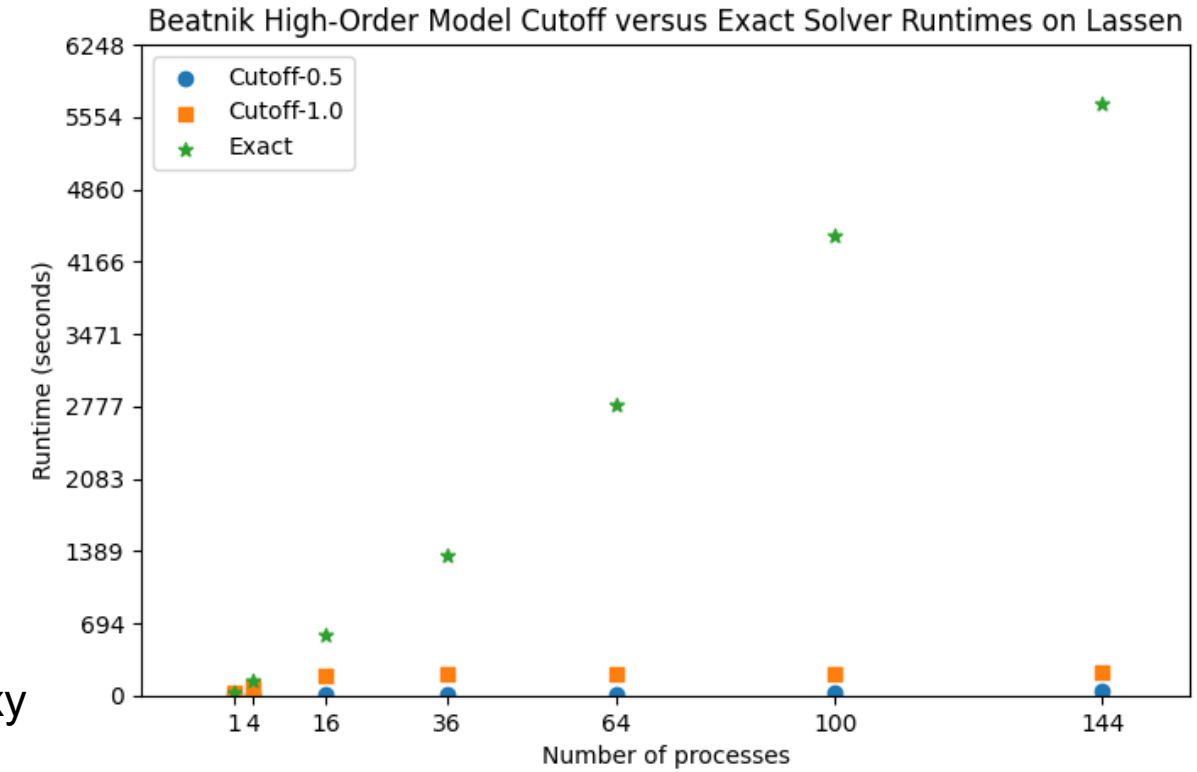


Beatnik: A Proxy for Remapping and Global Communication

Jason Stewart, **Patrick Bridges**

Beatnik: A Proxy for Remapping and Global Communication

- Release 1.0 available
  - https://github.com/CUP-ECS/beatnik
  - Scalable low-order solver (uses HeFFTe)
  - Non-scalable brute force high-order solver
  - Structure for implementing other solvers
- Current development – Jason's Poster
  - New cutoff-based solver that remaps between surface and spatial distributions
  - Adding support for irregular spatial meshes
  - Collecting communication profiles
  - Developing additional input decks
- Potential Future:
  - Addition of surface remeshing support
  - Integration with CFD miniapp for multi-scale proxy

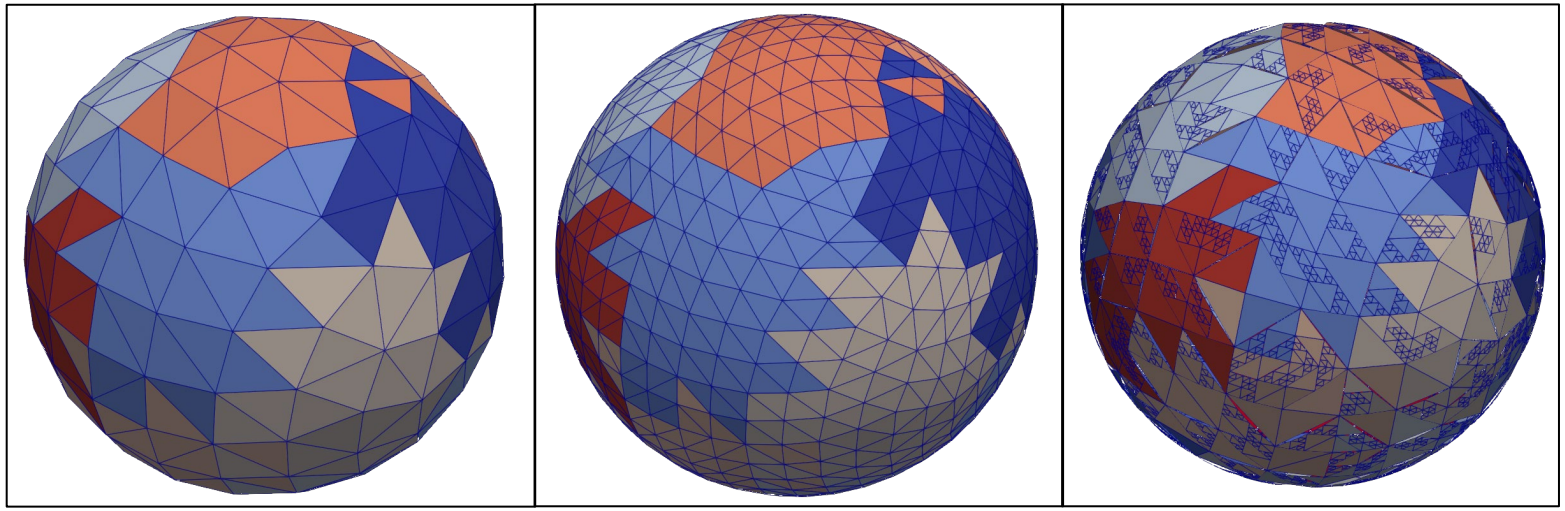


Performance Profiling of Beatnik

- Beatnik is limited by network bandwidth when weak scaling FFTs.
- Latencies associated with small all-to-all communication and small GPU FFTs form bottleneck when strong scaling.
- Cutoff solve strong scaling: Load imbalances develop as particles are pulled into the rollup.
- Beatnik highlights tradeoffs of HeFFTe parameter choice.
- **See the poster for runtime results and time spent in MPI.**

Added support in Beatnik for closed interface surfaces and adaptive mesh refinement

- Enable research of communication patterns associated with dynamic, non-uniform meshes
- Enable modeling of closed interface surfaces
- Working with LANL Postdoc Ian May on unstructured, adaptive mesh support for Beatnik (e.g., rising bubble problem)



16 processes, no face refinement

16 processes, uniform face refinement

16 processes, non-uniform face refinement

Colors represent rank of ownership

Future work includes load balancing and addition of more far-field force algorithms

- Incorporate Z-Model mathematics into unstructured and adaptive mesh.
- Implement distributed, fast-multipole-like algorithm for far-field force calculations on the unstructured mesh, taking advantage of hierarchal structure.
- Implement dynamic load balancing of mesh.
- Performance profiling using different communication spaces in Cabana.

Adding Beatnik Support to the Benchpark Experiment Management Framework

Abdalaziz Raad, Patrick Bridges

Motivation

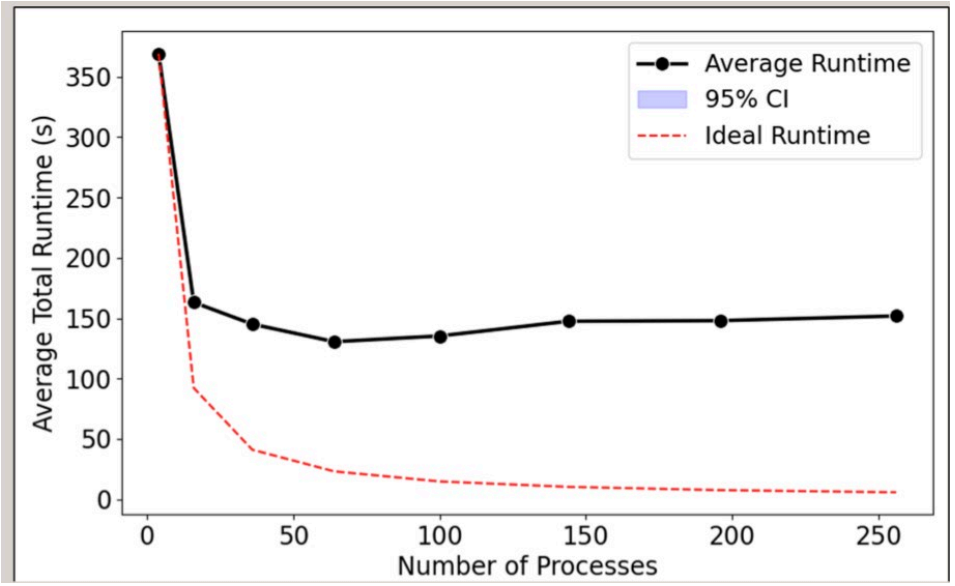
Goal: Demonstrate the ability to manage complex communication experiments.

- Started off with running and benchmarking Beatnik on its own.
- Worked with ReFrame to automate and track testing of Beatnik.

Findings:

- ReFrame doesn't offer fine-grained experiment tracking.
- Need to track linking inputs, outputs, configuration versions, and metadata.

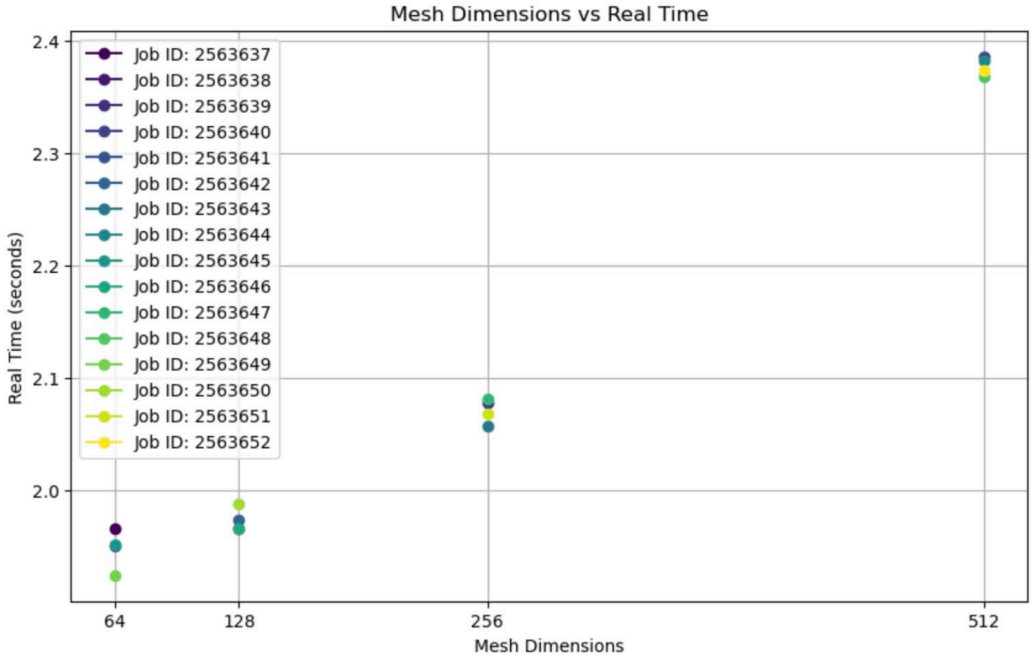
Result: Transitioned to using Benchpark to manage benchmarking, which is integrated with Spack and Ramble.



Example: Beatnik scalability data that we want be able to generate systematically and reproducibly.

Current Work

- Studied STREAM and Kripke Benchpark configurations to understand how it works.
- Using Ramble to automate job control and environment setup.
- Learning to use Benchpark to store the performance data
  - Structured and searchable format
  - Result tracking
  - Performance comparisons across different setups.



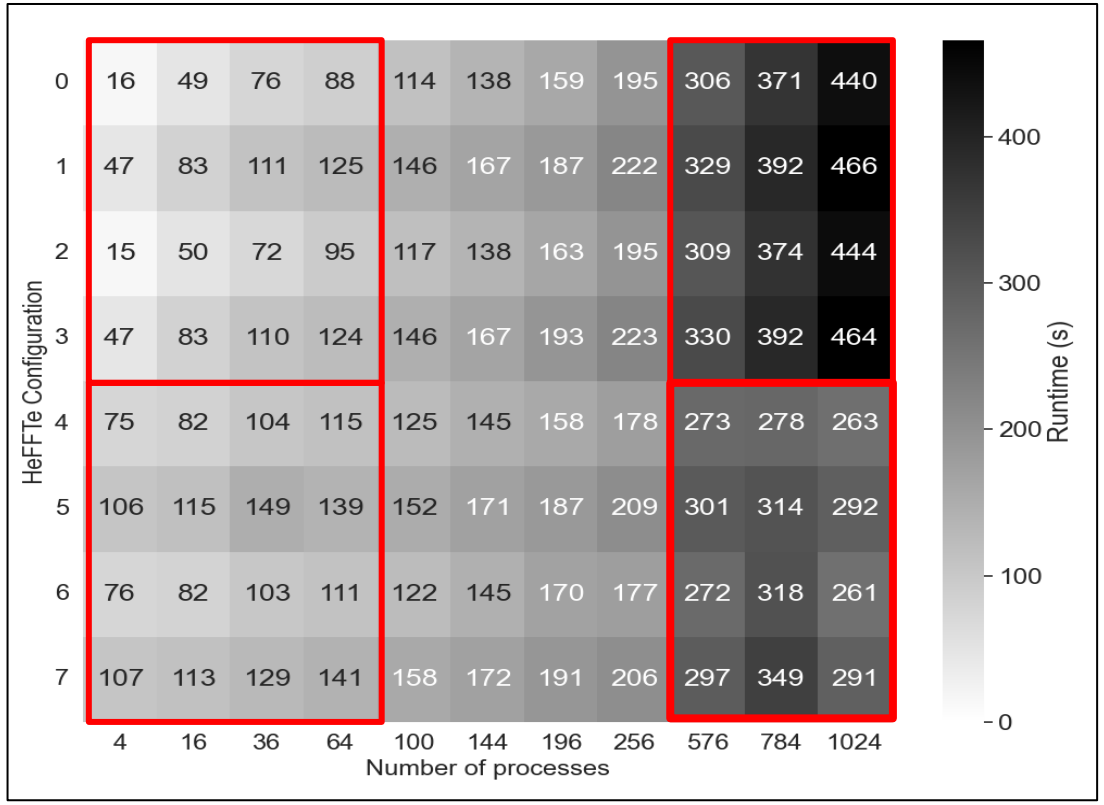
Example: Beatnik performance data from multiple runs to store in Benchpark

Additional Beatnik Results to Reproduce

The rows correspond to different HeFFTe configurations (numbered 0–7), while the columns represent increasing numbers of parallel processes. The red boxes highlight important regions:

- Left section (processes 4–64): Here, we see strong differences in runtime depending on the configuration. Configurations 2 and 0 perform the best at low process counts.
- Right section (processes 576–1024): At very high process counts, runtimes increase again for some configurations — likely due to communication overhead outweighing computational gains.
- Bottom section (configs 4–7): These configurations perform better at higher process counts, showing scalability improvements over configurations 0–3.

Beatnik Benchpark experiments will help identify which FFT configurations are optimal for different levels of parallelism.



Runtime (in seconds) of different HeFFTe configurations across varying process counts.

Next Steps

- Developing a full configuration that defines how Beatnik runs.
- Setting up the workflow so Beatnik can be easily executed across different systems without needing to manually edit scripts or settings
- Ensuring compatibility with any system that supports Ramble and Benchpark, regardless of hardware or environment
- Planning on running Beatnik on multiple systems and adding it to the Benchpark application library.

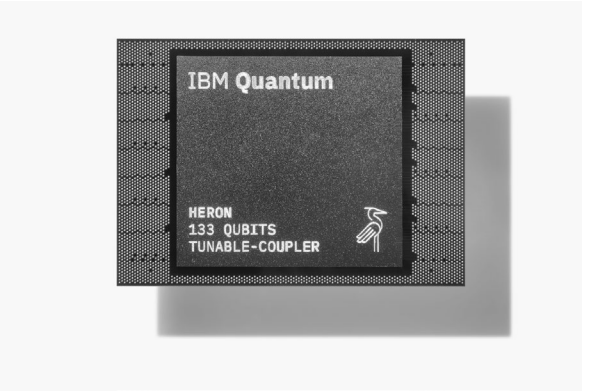
Scaling Laws for the Workload Throughput of Emerging Heterogeneous Clusters

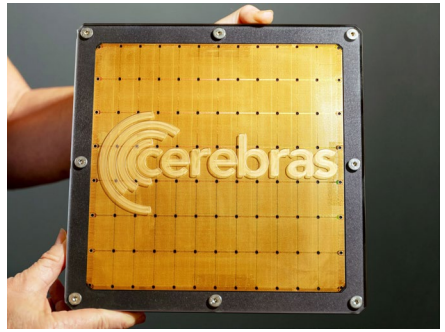
Akhil Alasandagutti, Joshua Suetterlein[2], Jesun Firoz[2], Stephen Young[2], Joseph Manzano[2], Jason Stewart, Patrick Bridges, Trilce Estrada[1], Kevin Barker[2]

The University of New Mexico[1], Pacific Northwest National Laboratory[2]

Motivation

- HPC systems incorporate heterogeneous components such as GPUs and FPGAs for performance improvements as Moore's Law ends.
- Next generation systems will likely incorporate a broader array of specialized accelerators co-located on the same network
  - Quantum Co-processors
  - Neuromorphic Accelerators
  - AI Accelerators
  - Dataflow Accelerators
- New challenges
  - System Utilization
  - Resource Allocation
  - System Provisioning





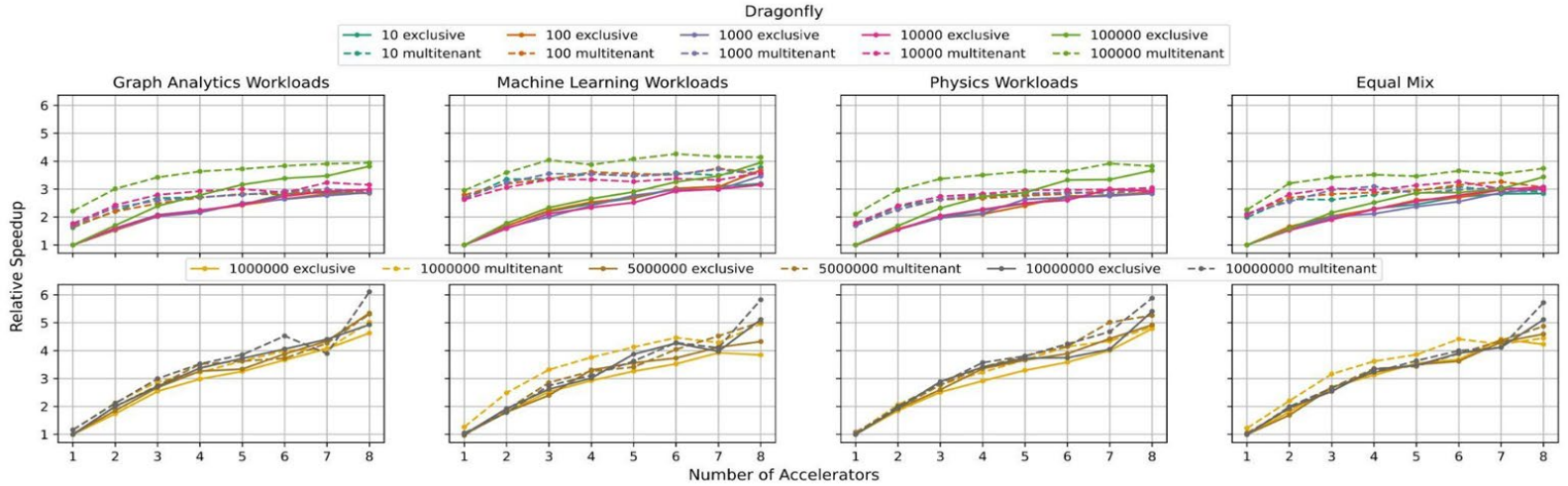
Methodology

Goal: Simulate and model approaches for sharing network-attached accelerators on different network and hardware configuration

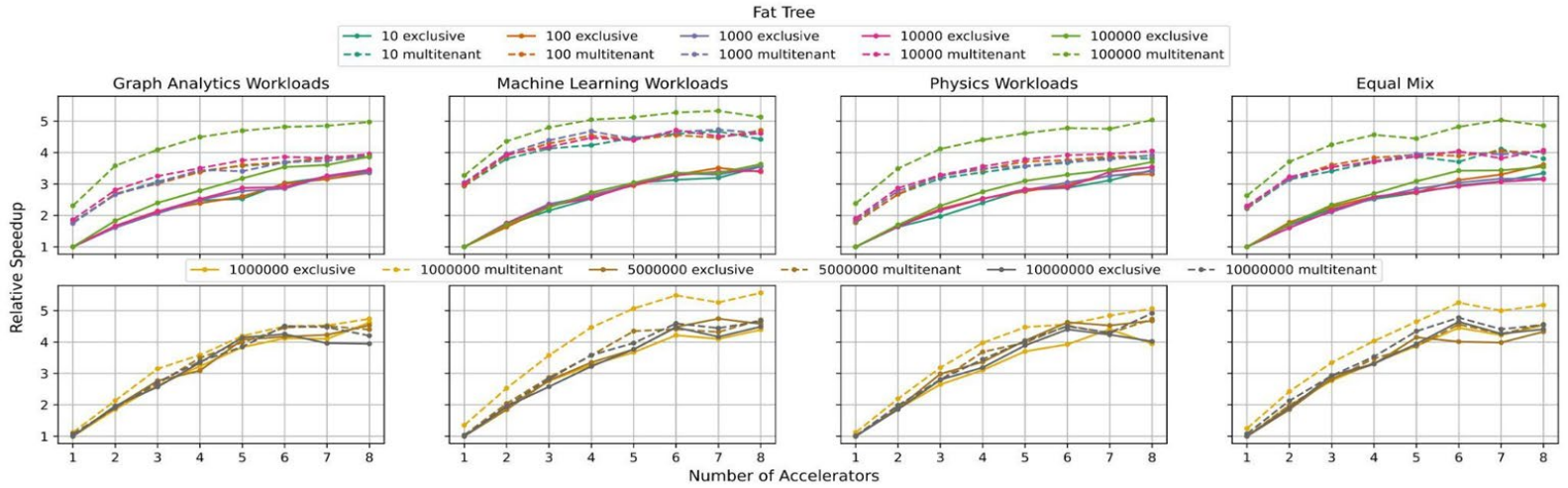
1. Use SST Macro to examine how different policies, network topologies, and accelerator task times (ATTs) impact overall application speedup and utilization.
2. Develop analytical performance model to predict how changes in workload, scheduling policy, and accelerator availability impact utilization and throughput
3. Validate against simulated networked accelerators on real systems

Example Simulation Data: Relative Speedups

- Relative to 1 Accelerator using Exclusive Mode.
- Fat Tree shows continued improvement beyond 6 accelerators. Likely due to larger size (686 vs 512 nodes).
- For shorter ATTs, multi-tenant mode offers significantly better performance for the same number of accelerators.
- No significant difference in performance between scheduling modes for longer ATTs



Homogeneous workloads



Performance Model

- γ is some workload and system-dependent parameter.
- *k* is the number of accelerators.
- (1 + ϵ) is the required threshold for diminishing returns upon the addition of an accelerator.
- *k* can be solved algebraically to determine the number of accelerators required for a system with a given γ and (1 + ϵ).
- Formal derivation of this model is given in the paper.

$$\frac{1}{1-\gamma+\frac{1}{k}\gamma}$$

Speedup relative to 1 accelerator

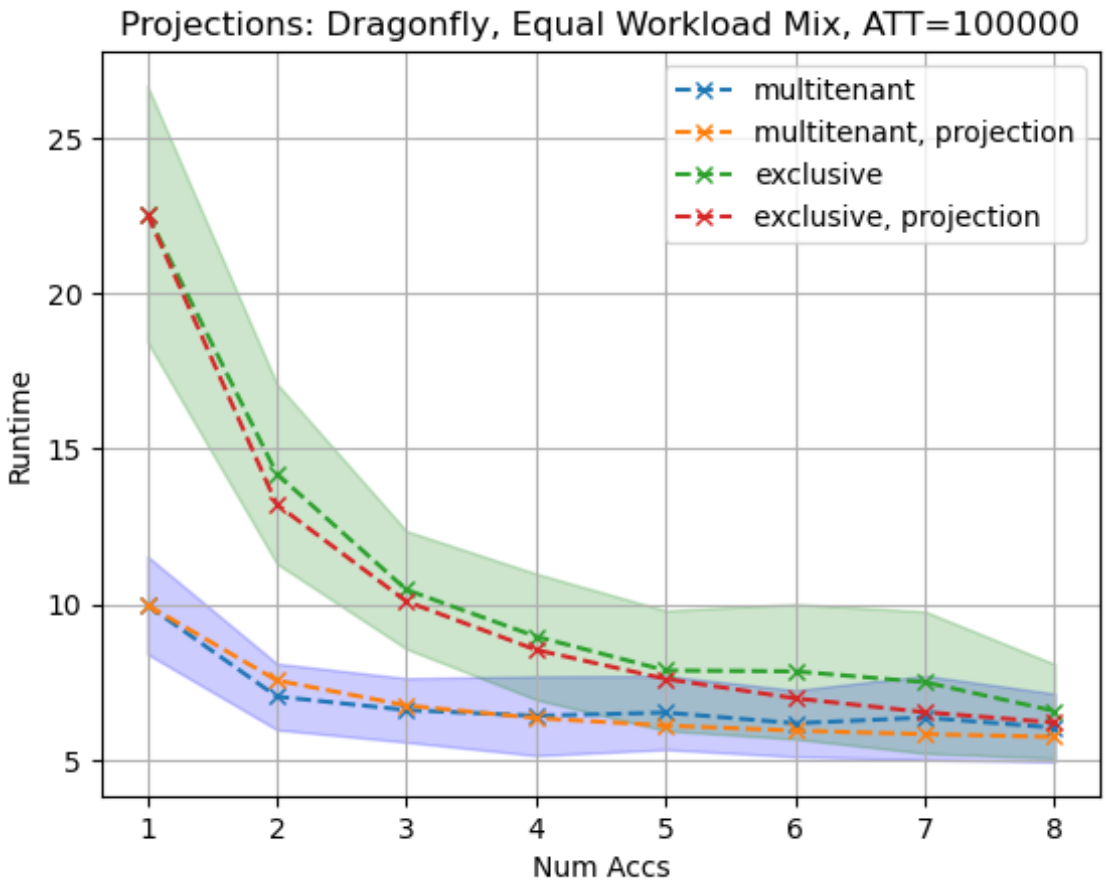
$$\frac{1-\gamma+\frac{1}{k}\gamma}{1-\gamma+\frac{1}{k+1}\gamma} < (1+\epsilon)$$

Inequality for diminishing returns

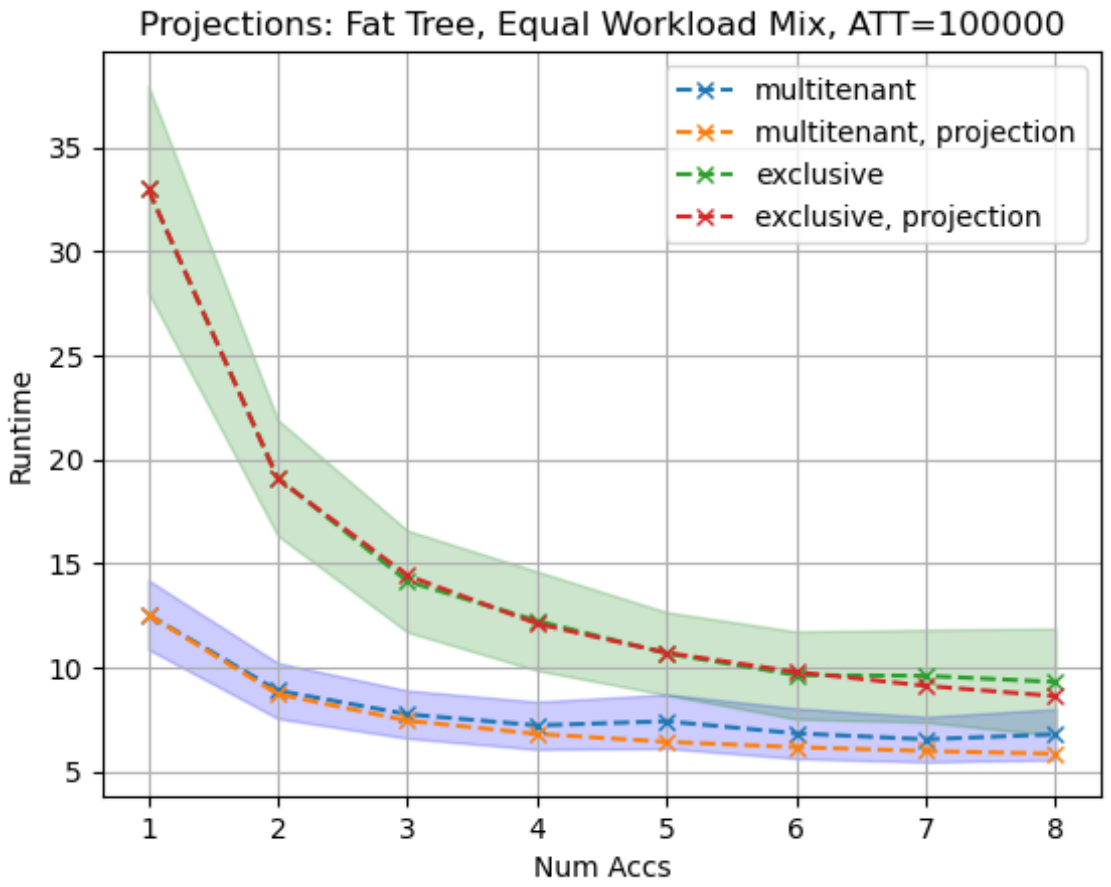
$$k = \frac{-1+\sqrt{1+4\frac{\gamma(1-\gamma)}{\epsilon}}}{2(1-\gamma)}$$

Solving k for the required number of accelerators

Example Performance Model Results



Model Projections on Dragonfly for ATT = 10^5 µs



Model Projections on Fat Tree for ATT = 10^5 µs

Validation Results

- The median ATT of 10^5 µs resulted in a clear separation of performance between exclusive and multi-tenant scheduling modes.
- There is no observable performance difference between exclusive and multi-tenant modes in the projections for the top end ATT of 10^7 µs.
- Our analytical models predict performance well, with all of the predicted points falling within the confidence interval of 1 standard deviation.
- Z-scores indicate that exclusive mode predictions are marginally more accurate than multi-tenant mode predictions.
- There is no observable difference in prediction accuracies for any of the other parameters.
- The increased variability in real system runs can be attributed to network interference and other system effects not present in SST/macro simulations.

Poster-based Discussion Time

Break at 3:15 PM