SAND2011-5439P

Overlapping Communication and Computation using Cray's Gemini Interconnect

August 1st, 2011

Ben Strohbeen
CSRI Student Intern

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Overlapping Communication and Computation can Improve Application Performance

- A way to hide communication latency
  - Theoretical improvement?
    - Perfect overlap differs between programs
- Important for scalability of large applications
- Hardware and software design issues get in the way of actually achieving this

How the Cray XE6 Supports Overlapping Computation and Communication

- Gemini Interconnect
  - RDMA for direct memory copies between processes
    - No copying to and from buffers
  - Has structures to take advantage of RDMA for both large and small messages
    - Block Transfer Engine (BTE)
    - Fast Memory Access (FMA)
- Asynchronous MPI progress

Making Asynchronous MPI progress is Crucial to Overlapping Computation and Communication

- The effectiveness of nonblocking MPI operations is system and implementation dependent
  - Nonblocking != asynchronous
- Most MPI implementations have nonblocking operations make progress and complete in MPI_Wait or MPI_Test
  - These implementations are not truly asynchronous
    - Cray's MPICH2 should be

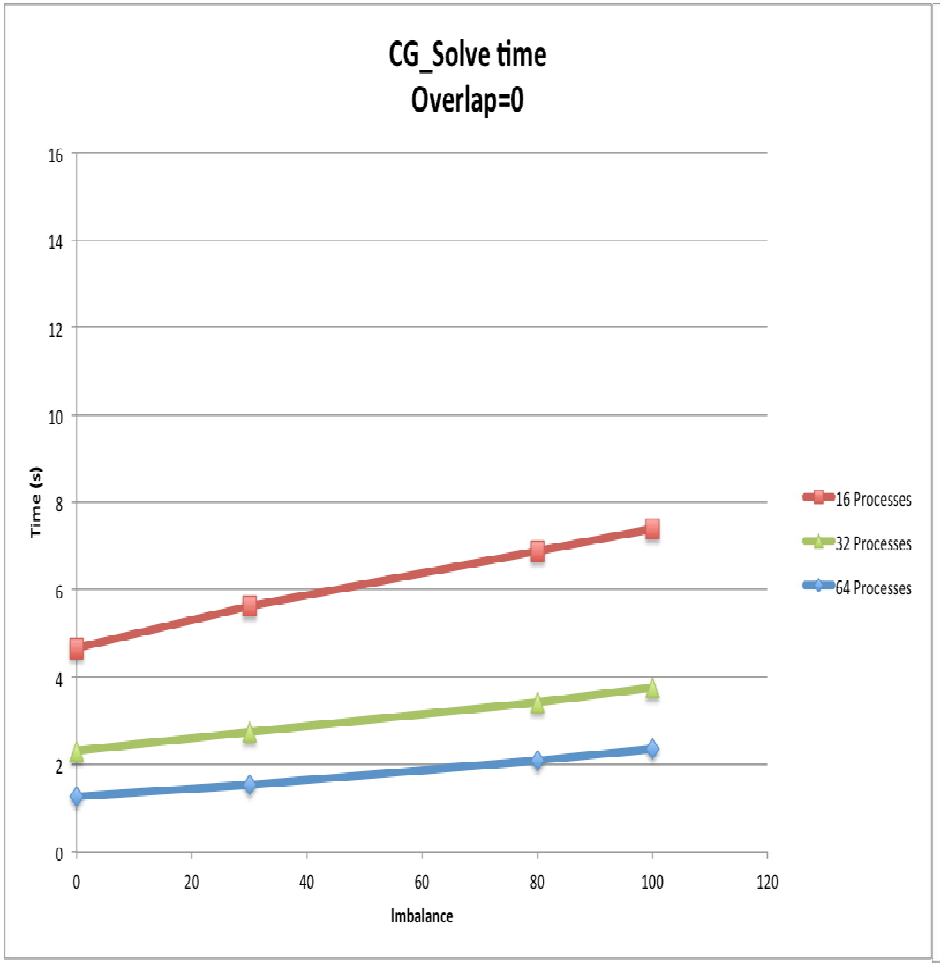
miniFE

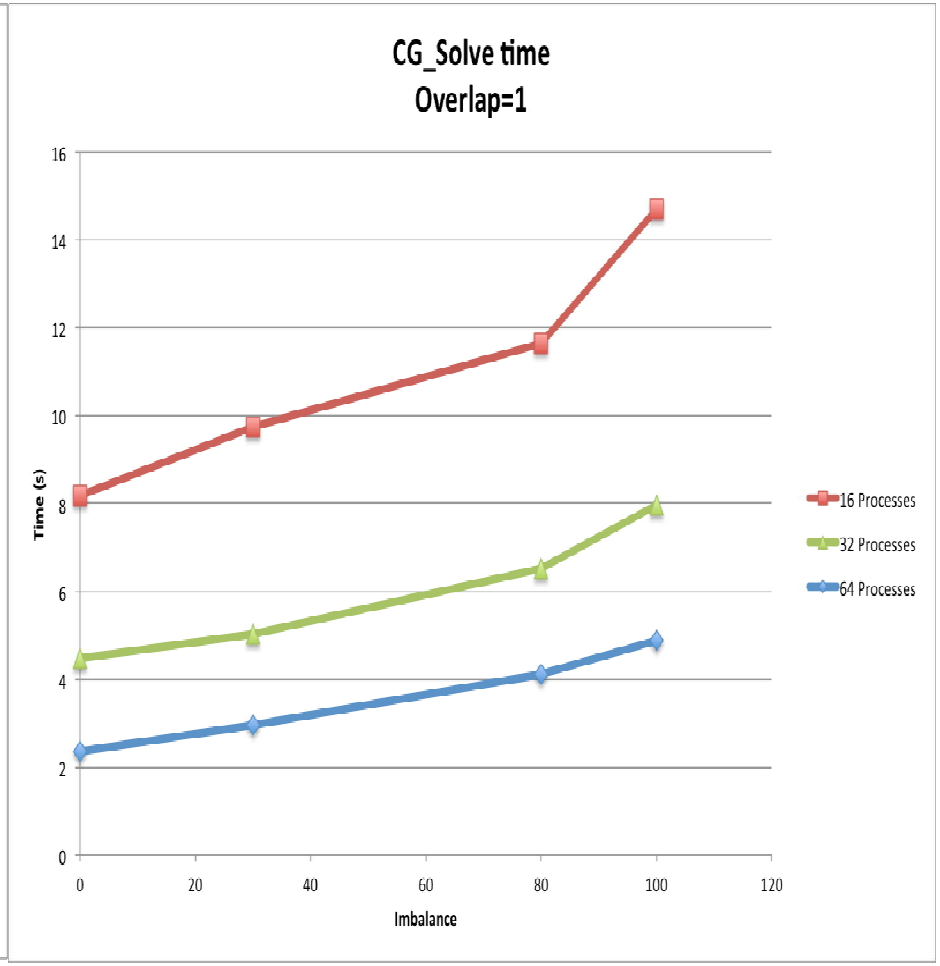
- Implicit finite element miniapp
- Focused on the Conjugate Gradient solver
  - Tried overlapping communication and matrix-vector multiplication
- Instrumented with CrayPat
  - Used Gemini performance counters
- Compared runtime and CrayPat data between miniFE's overlapping mode and non-overlapping mode

miniFE is Negatively Impacted by Trying to Overlap Communication and Computation

- When using nonblocking MPI message passing, miniFE's global dot product becomes greatly slowed down.
  - Seems to be due to MPI_AllReduce
  - Sensitive to load imbalance

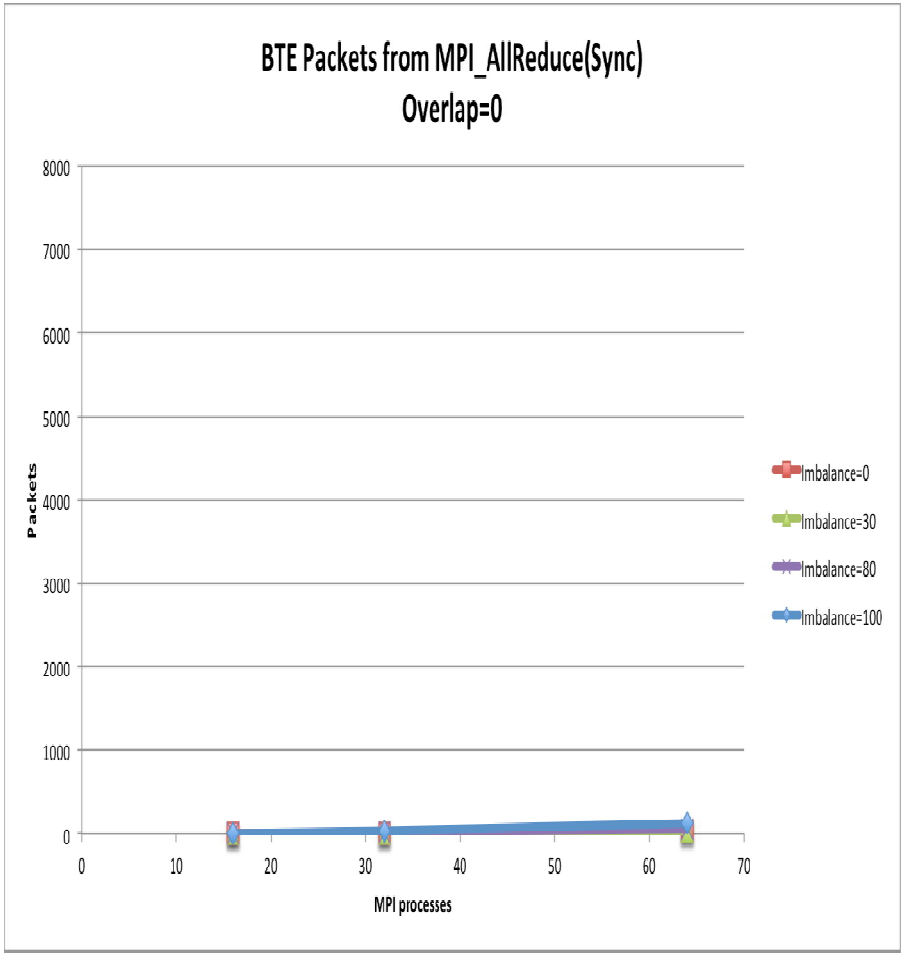
Total Conjugate Gradient Solve Times

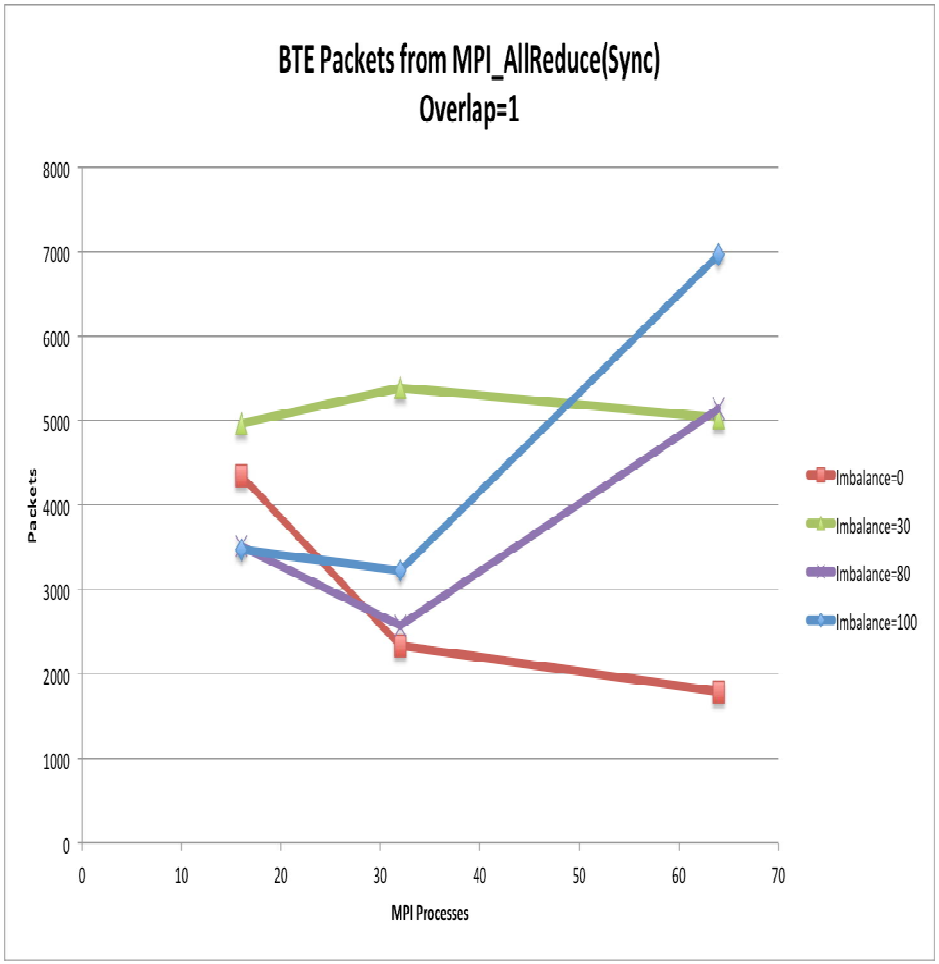




Sandia National Laboratories

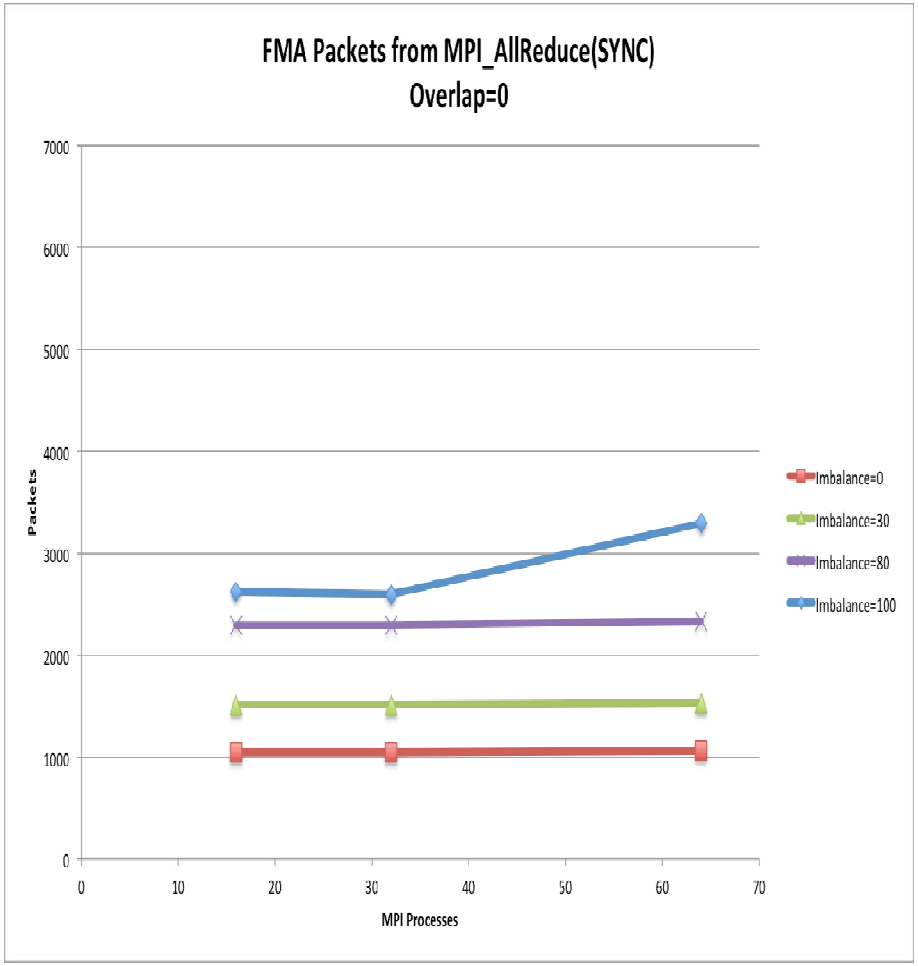
Packets from MPI_AllReduce(SYNC) sent through the Block Transfer Engine

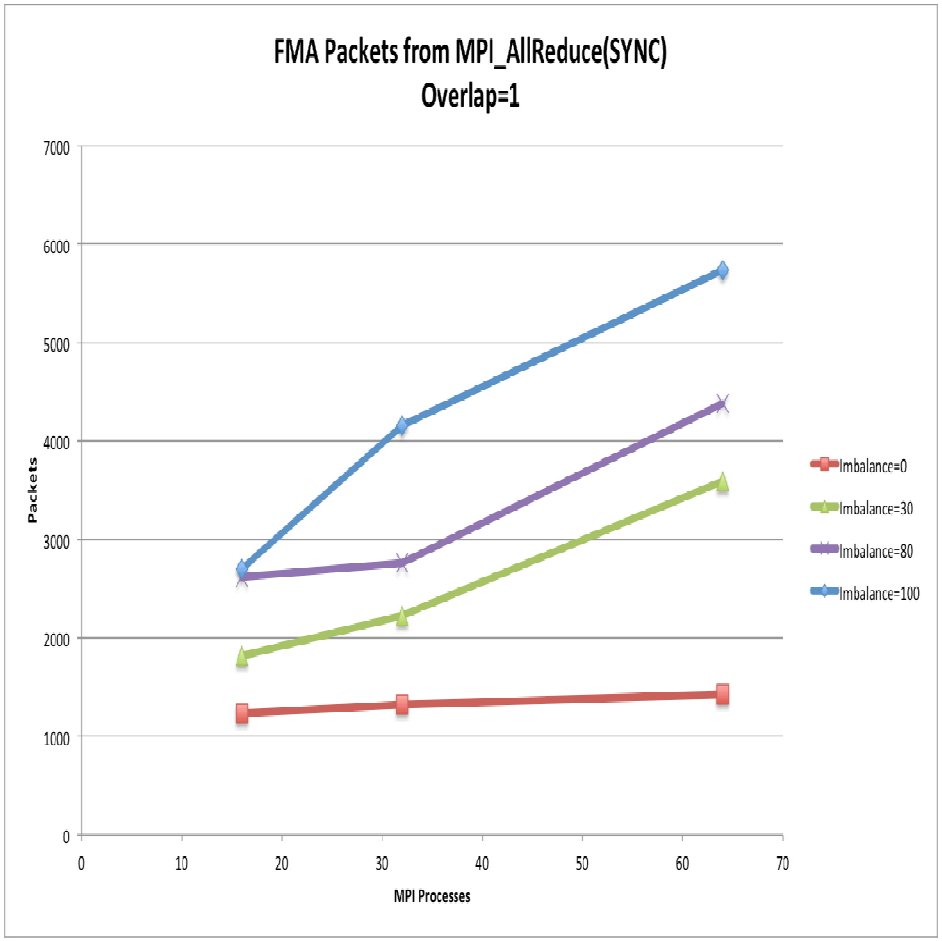




Sandia National Laboratories

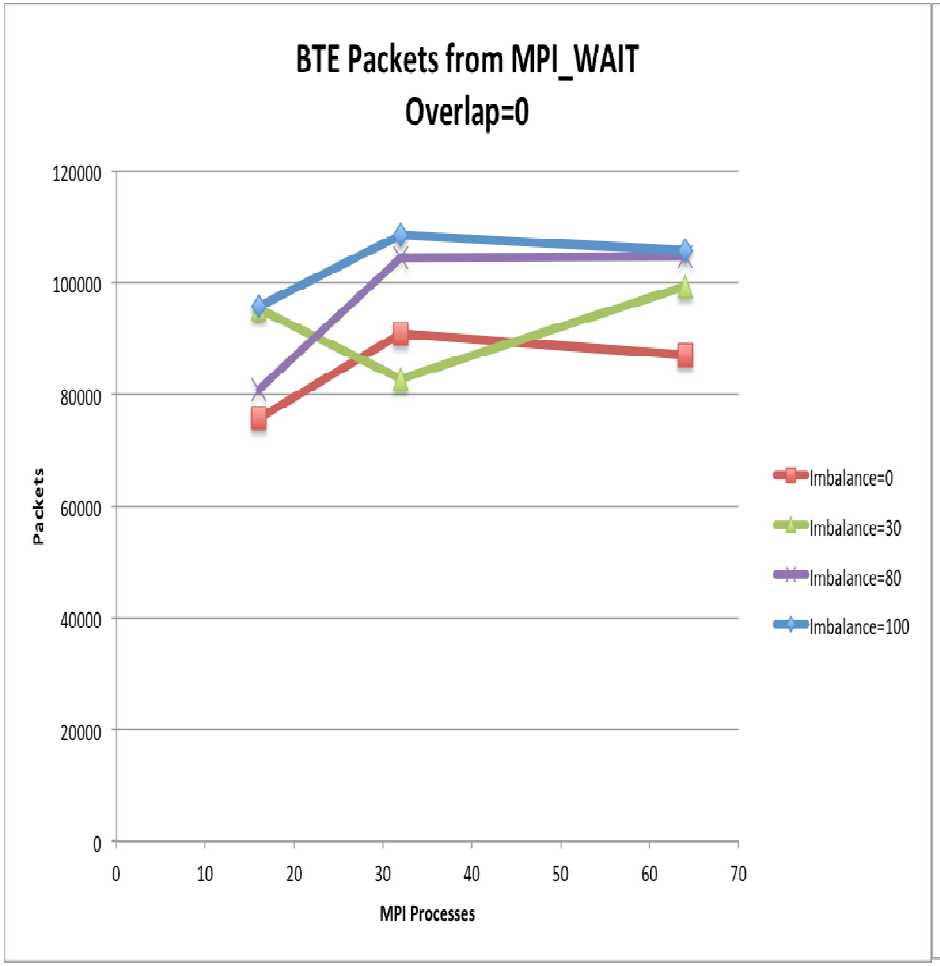
Packets from MPI_AllReduce(SYNC) sent using Fast Memory Access

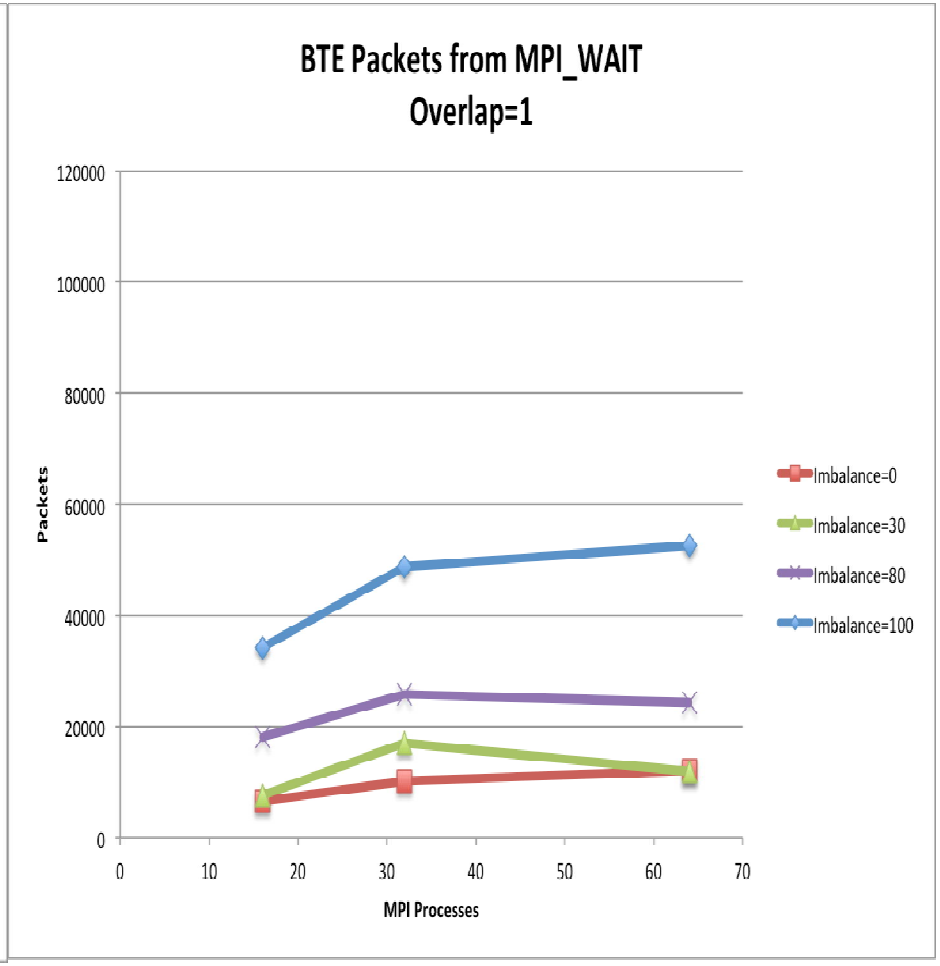




Sandia National Laboratories

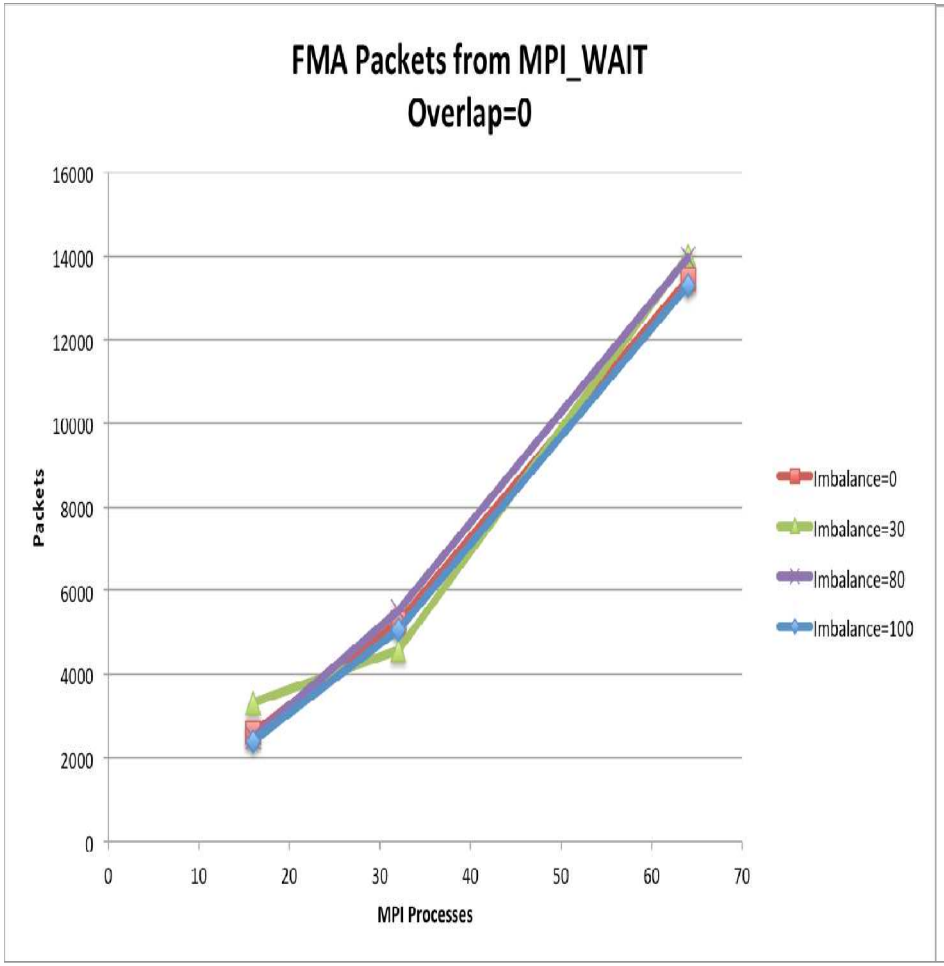
Packets from MPI_Wait sent through the Block Transfer Engine

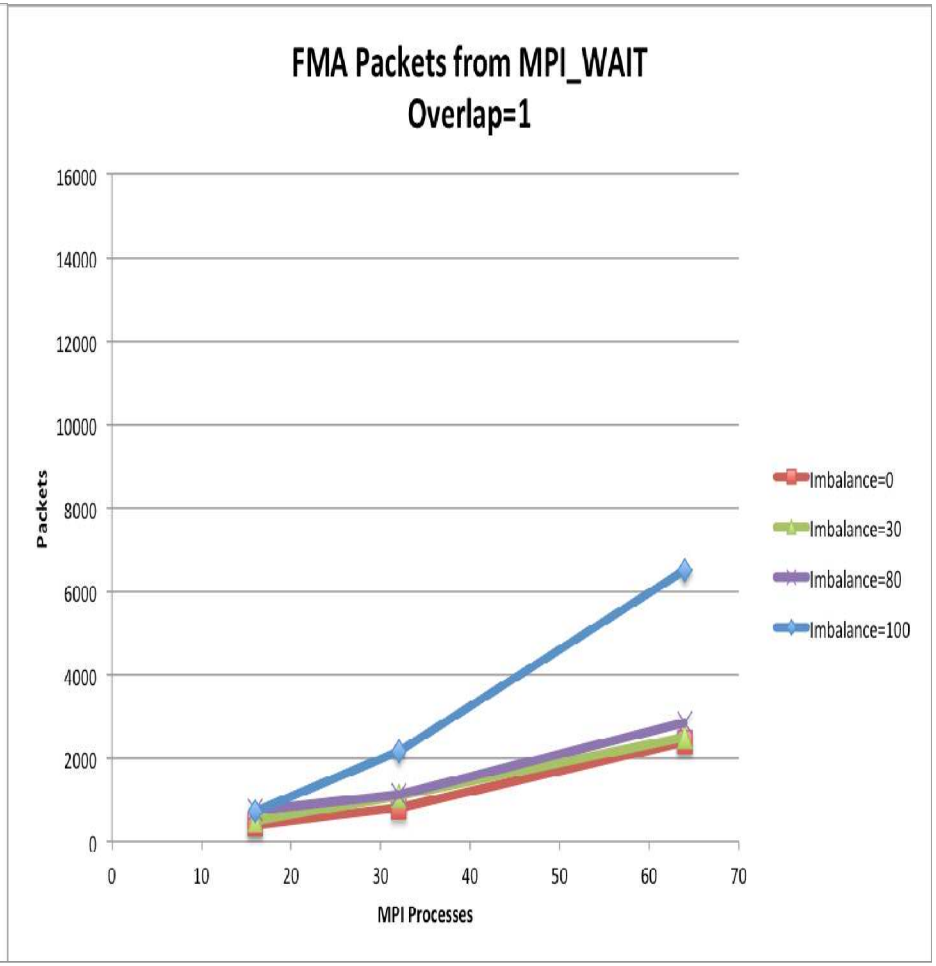




Sandia National Laboratories

Packets from MPI_Wait sent using Fast Memory Access





Sandia National Laboratories

MPI Collective Operations Decrease Overlapping Potential

- The bottleneck due to a MPI Collective operation makes sense
  - No support for nonblocking collectives
  - Processes enter the reduction at different times
    - Even though the collective call is outside of the nonblocking calls, it still has a major impact
- Nonblocking collectives could probably help this bottleneck

miniGhost

- Difference stencil miniapp
  - Much simpler communication pattern (in general)
- Problems with overlapping computation with exchanging diagonal ghost boundary values
  - Communication/ logic issue, wouldn't be simply fixed by a better MPI implementation like miniFE possibly would.

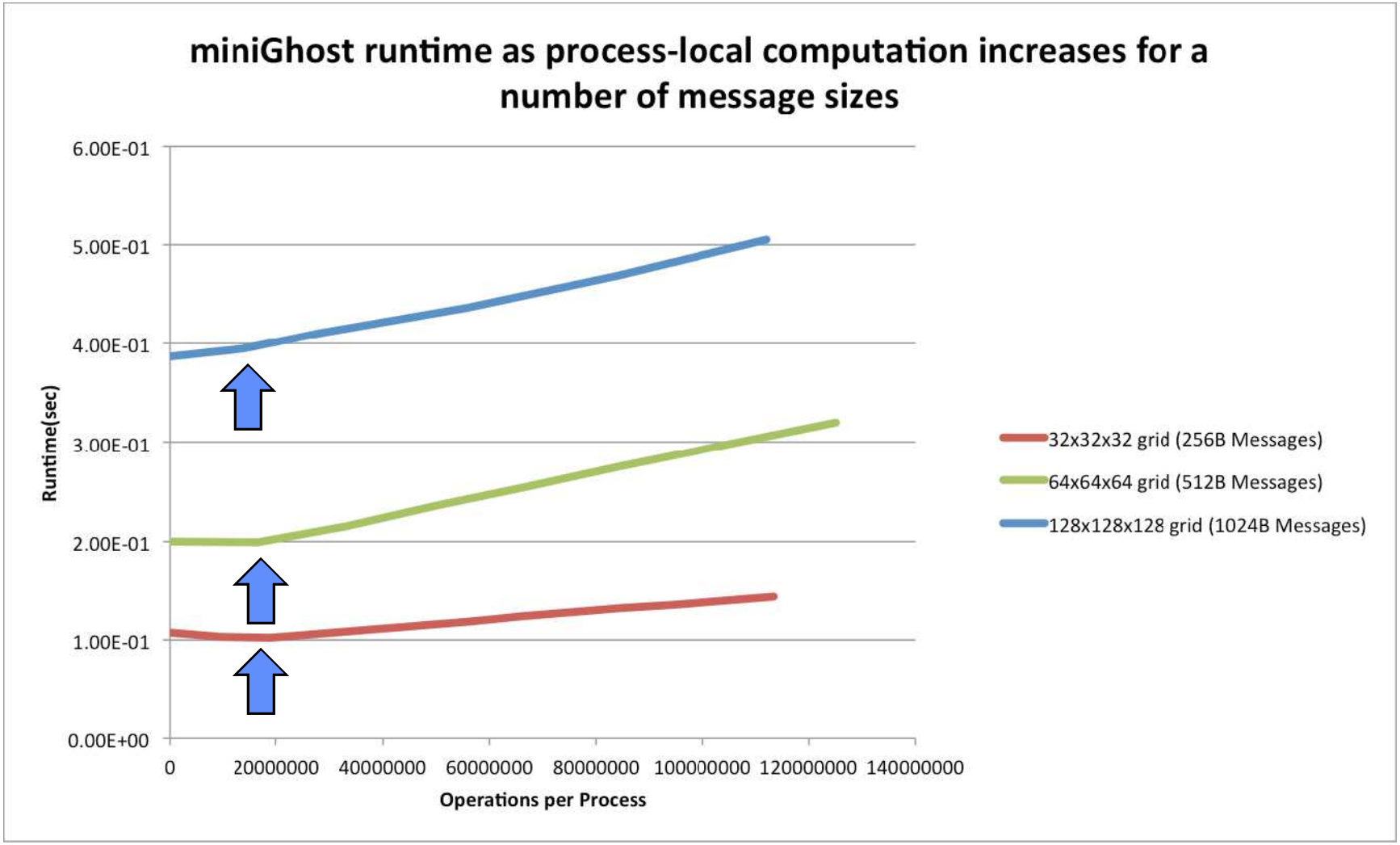
miniGhost Receives Data Values before MPI_Wait is Called

- Instrumented miniGhost to check if ghost boundary values changed before or after MPI_Wait was called
  - Indirect way of checking for asynchronous MPI
  - Tested Pre-posting receives and posting sends first
    - Posting sends first for large messages should be an issue with the BTE

miniGhost and Computation Inflation

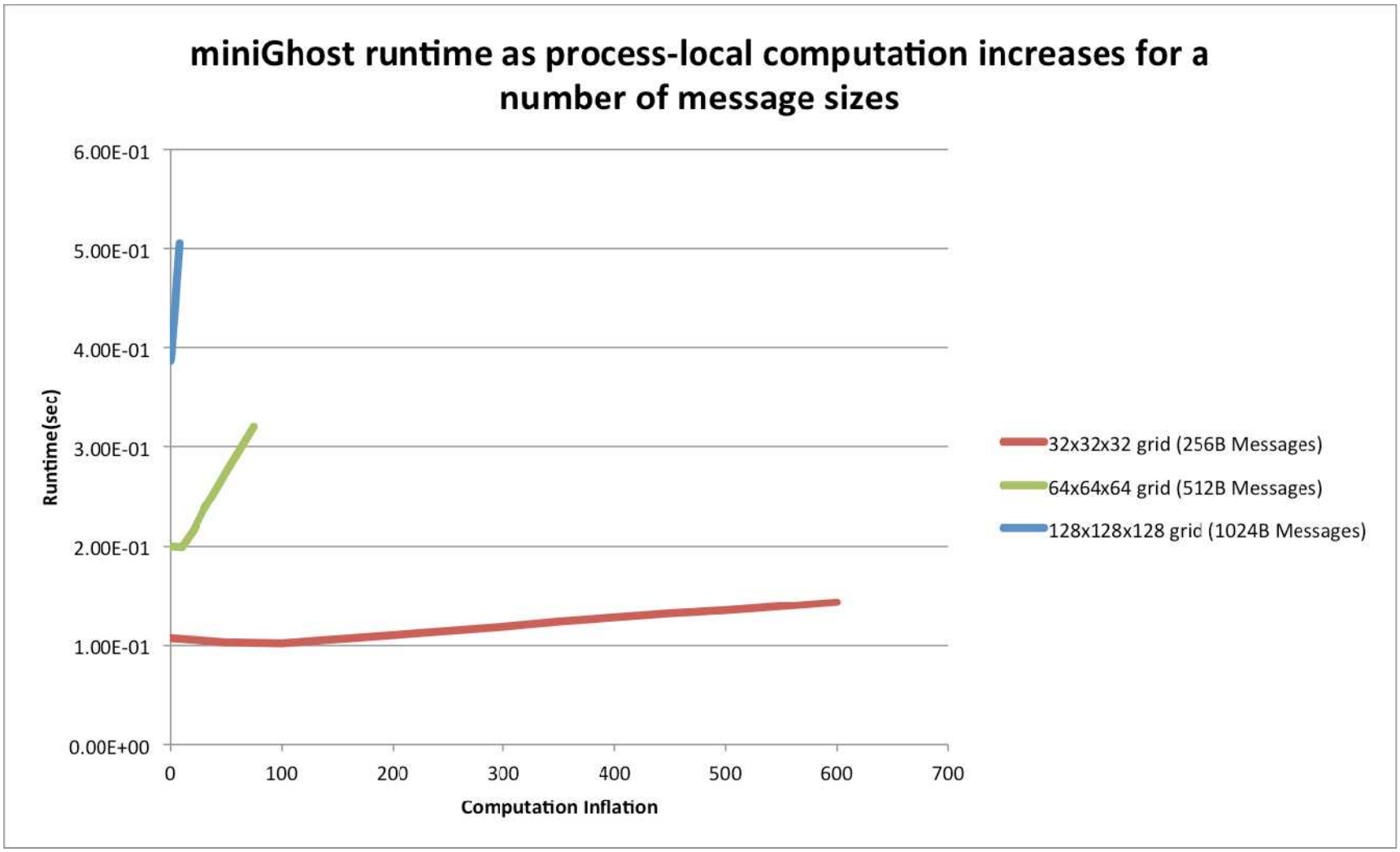
- Runtime should be constant if computation time is less than communication time
- Stripped away all the code so miniGhost would just send, do an inflated amount of computation, then receive
  - Compared miniGhost's runtime for different levels of computation inflation
    - Differences in linear scaling between grid sizes

miniGhost Demonstrates Computation Overlap for Small Messages



Sandia National Laboratories

miniGhost Demonstrates Computation Overlap for Small Messages



Sandia National Laboratories

A Similar Experiment Produced Contradictory Results

- On an XE6, the researchers checked total runtime against computation time
  - Found no evidence of overlap
- The researchers used 80 MB messages
  - Relevant for the applications they were targeting
  - Wanted to avoid complications with Eager / Rendezvous protocols at smaller message sizes

Conclusions

- On Gemini, overlapping communication and computation seems to be possible for real applications
  - Keeping in mind certain restrictions
- It could be useful to quantify a performance increase by configuring a fully nonblocking version of miniGhost and comparing.