Analyzing the Performance of a Sparse Matrix Vector Multiply for Extreme Scale Computers

Amanda Bienz, Jon Calhoun, Luke Olson, Marc Snir, and William D. Gropp
University of Illinois at Urbana-Champaign

Abstract

Extreme scale computers require algorithms that are scalable to millions of cores. To scale to such core counts algorithms need algorithms need to become asynchronous

Sparse Matrix-Vector Multiply (SpMV) is fundamental to a large class of HPC applications, but its performance and scalability is limited at large core counts by network contention

An asynchronous SpMV can improve performance by 2.7x

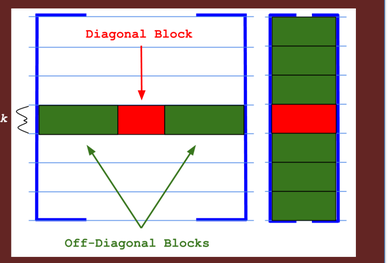
Motivation

A parallel SpMV requires communication to collect vector components needed for the off-diagonal block

Most linear algebra packages collect vector components with a receive per process

Computation is preformed once all components are received

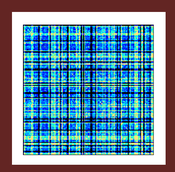
Network contention at large scale increases communication time



Test Setup

- Run on Vulcan and Blue Waters with 16 processes per node
- Test using the Dalaunay and Kron matrices from the University of Florida Sparse Matrix Collection[1] with approximately 10,000 degrees-of-freedom per process
- Compare SpMV performance of our Asynchronous SpMV to the SpMV in Hypre[2], Trilinos[3], and PETSc[4]

delaunay_n22



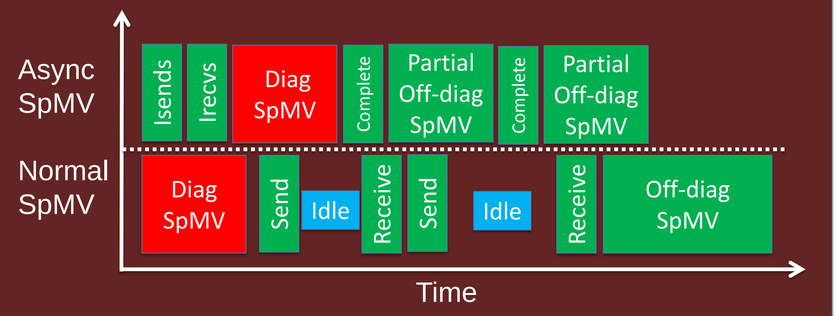
kron_g500-logn19

Asynchronous Algorithm

1. Post a MPI_Irecv for each process we need to collect vector components from
2. Do local diagonal SpMV
3. Do a SpMV as each Irecv completes

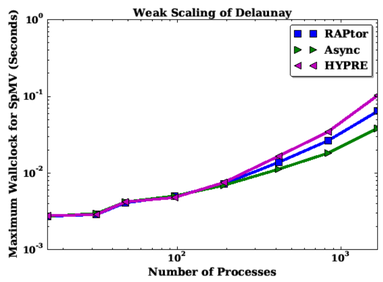
Goal:

Remove idle time in the off-diagonal communication by scheduling part of the off-diagonal computation as soon as a receive completes

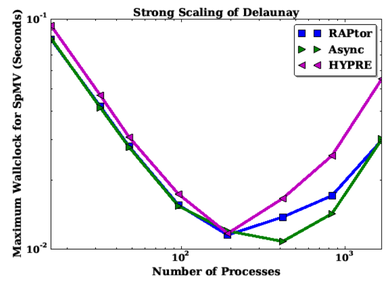


Results

Vulcan

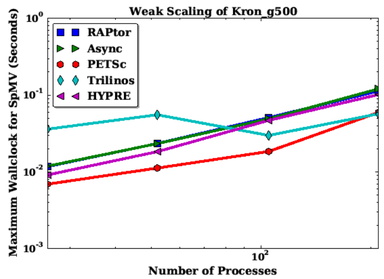


Weak Scaling Study – Delaunay Matrices

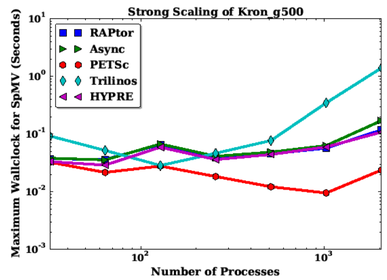


Strong Scaling Study – Delaunay Matrix

Blue Waters



Weak Scaling Study – Kron Matrices



Strong Scaling Study – Kron Matrix

- **Vulcan** – our asynchronous SpMV can significantly improve performance in both weak and strong scaling studies
- **Blue Waters** – our asynchronous SpMV was outperformed by our synchronous SpMV as well as the other methods

	Vulcan	Blue Waters
α (latency)	4.319e-06	1.812e-06
β (1/bandwidth)	4.274e-09	1.845e-09

Conclusions

- SpMV performance is improved with an asynchronous algorithm
- Sparsity pattern in the off diagonal portion of the local matrix determines amount of network contention and dictates which SpMV algorithm is best

Future Work

- Test other forms of communication available on modern HPC systems, e.g. one sided, active messages
- Build a performance model to better address the asynchronous SpMV performance
- Investigate heuristics to determine which sparsity patterns and network parameters permit the asynchronous SpMV

This work is sponsored by the Air Force Office of Scientific Research under grant FA9550-12-1-0478.
This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant Number DGE-1144245.
This research is part of the Blue Waters sustained-petascale computing project, which is supported by the National Science Foundation (awards OCI-0725070 and ACI-1238993) and the state of Illinois. Blue Waters is a joint effort of the University of Illinois at Urbana-Champaign and its National Center for Supercomputing Applications.

1: http://www.cise.ufl.edu/research/sparse/matrices/index.html
2: http://acts.nersc.gov/hypre/
3: https://trilinos.org/
4: http://www.mcs.anl.gov/petsc/