# Reducing Communication Costs in the Parallel SpMV

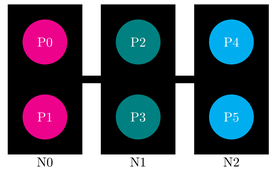
Amanda Bienz[1]
In collaboration with: Luke Olson[1], William D. Gropp[1]
[1]University of Illinois at Urbana-Champaign

## Abstract


- **Parallel sparse matrix-vector multiplication (SpMV)**
- Dominant operation in many linear solvers
- Often dominated by communication costs
- Cost can be reduced by grouping messages together based on topology of super computer


## Parallel Communication

- Multiple processes per node
- Communication cost = start-up + byte transport
- **Inter-node communication is *much* more costly than intra-node**
- Baseline Costs:




Three nodes, each containing two processes

| | Intra | Inter |
|---|---|---|
| Start-Up | 7.759e-07 | 2.577e-06 |
| Byte Transport | 4.090e-09 | 9.751e-09 |

Costs of large messages

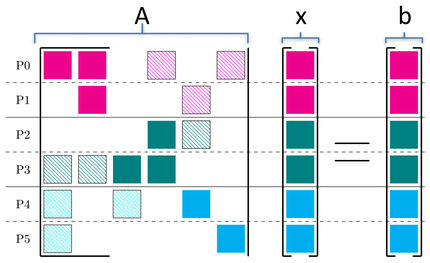
| | Intra | Inter |
|---|---|---|
| Start-Up | 2.439e-06 | 7.623e-06 |
| Byte Transport | 7.890e-10 | 1.971e-09 |

Costs of small messages

- Additional costs: network contention, injection bandwidth limits, etc.

## Standard Parallel SpMV

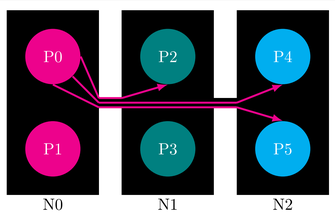
- System distributed across processes in example system
- Each process communicates it's vector values to every process that needs them



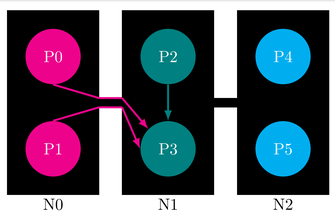

An example system partitioned across four processes.

- **Solid blocks**: non-zeros corresponding to vector values stored on node
- **Striped blocks**: associated with vector entries located on distant processes; require communication

- Multiple messages are often sent to / received from the same node




For example system, P0 must send same value to P3, P4, and P5.



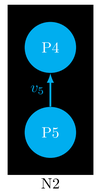

For example system, P3 receives from P0, P1, and P2

## Reducing Global Communication

- **Goal: Remove duplicate data and combine small messages**

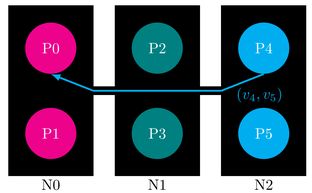
1. Process p on node n collects all on-node values that go to some node m (*intra-node)*

**Example:** P4 collects all values to send to node N0 ( $v_4$ and $v_5$ )



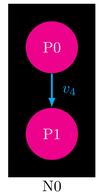

2. Process p sends collected values to process q on node m (*inter-node*)

**Example:** P4 sends all collected values ( $v_4$ and $v_5$ ) to process P0




3. Process q distributes to all on-node processes that need values from node n (*intra-node)*

**Example:** P0 sends collected value $v_4$ to process P1

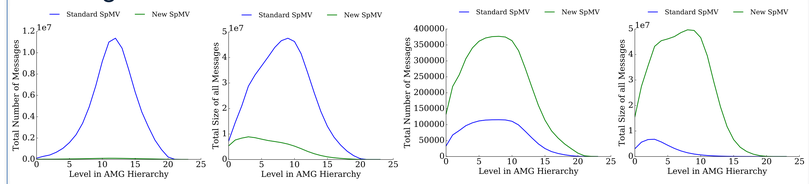



## SpMV Tests

- Linear Elasticity AMG Hierarchy from MFEM + Hypre
  - **Finer levels:** Small number of large messages
  - **Coarser levels:** Large number of small messages
- Florida Sparse Matrix Collection
  - Subset of largest real matrices (excluding the 4 largest due to large I/O requirements)
- **Blue Waters, 8192 Processes (512 nodes)**

## Communication Analysis

- Greatly reduce the number and size of inter-node messages
- Large increase in the amount of local communication



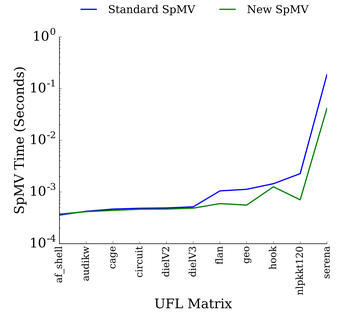

**Inter-node messages**: total number (left) and total size (right) for each level in linear elasticity hierarchy

**Intra-node messages**: total number (left) and total size (right) for each level in linear elasticity hierarchy

## Results

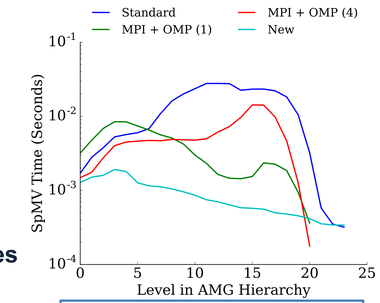
- Reduction in SpMV time for systems with a large number of small messages, ***often by an order of magnitude***
- Improvement on many UFL matrices


Linear elasticity (MFEM) hierarchy




Subset of UFL Matrices

- MPI + OMP (1)
  - PPN = 1
  - 16 OpenMP threads per process
- MPI + OMP (4)
  - PPN = 4
  - 4 OpenMP threads per process
- **New SpMV (only MPI) improved over both OMP cases**
- ***Note:*** *OpenMP hierarchies were slightly different due to different number of MPI processes*




Linear elasticity (MFEM) hierarchy (with OpenMP)

## Future Work

- Extend idea – inter-Gemini is more expensive than intra-Gemini
- Look at communication in other operations, such as matrix-matrix multiplication
- Test other large unstructured systems, as UFL collection contains relatively small matrices

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant Number DGE-1144245.