Enabling Sparse Matrix Computation in Multi-locale Chapel

Tyler Simon

Laboratory for Physical Sciences, College Park, MD

Amer Tahir

Milton Halem

University of Maryland Baltimore County, MD

Motivation, Objectives & Related Work

Motivation

- Large sparse matrices often appear in Science or Engineering problems, Social Network analysis, Topic modeling, Graph analytics, Sentiment analysis, Cyber security and so on.
- Storage and processing of these matrices is not possible on a single computer, so high performance computing (HPC) systems are used.
- HPC systems are evaluated with benchmarks. Conjugate Gradient algorithm on sparse matrices is used in popular HPC benchmarks, HPCG and NAS CG.
- Chapel, an emerging PGAS (partitioned global address space) language built for parallel computation offers flexibility and abstraction – significantly less lines of code compared to existing solutions (MPI/OpenMP).

Aims

- *"To provide a data structure in the Chapel programming language that enables the implementation of CG benchmark for compressed large sparse matrices"*
- Chapel programming language currently unable to deal with distributed compressed/sparse matrices over multiple locales.
- This work develops MSBD, a Multilocale Sparse Block Distribution for Chapel.

Related Work

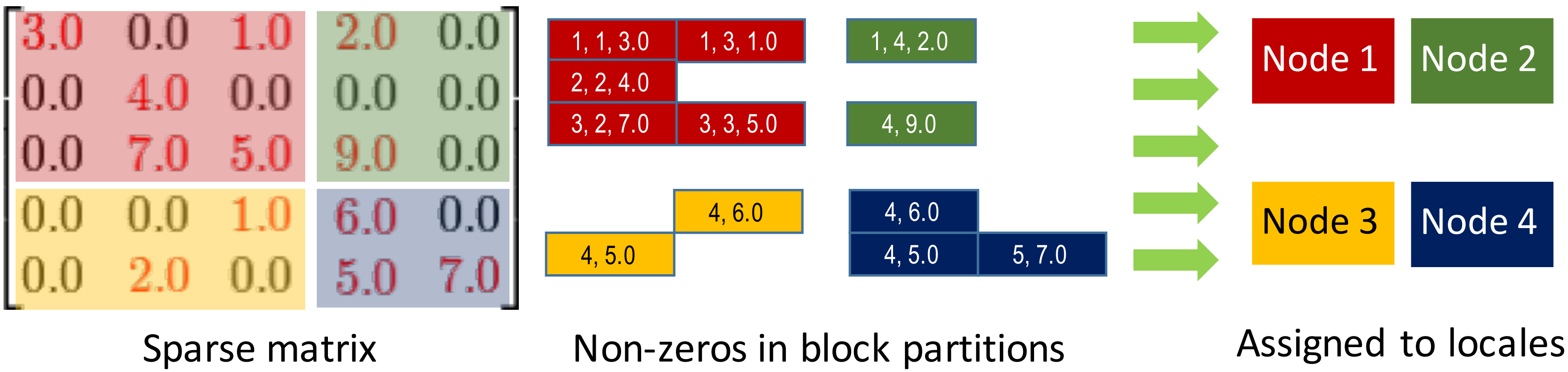
- HPCG & NASCG
  - Sequential, OpenMP and MPI reference implementations
- Chapel port of NAS CG - uses CSR
  - Single-locale only – Doesn't scale to multiple nodes!
- Unified Parallel C (UPC) and Titanium
  - UPC implementation offers better speed than MPI but doesn't scale as well as MPI based matrix multiplication

MSBD Overview

- Proposed solution is a custom Chapel distribution for sparse matrices
- Behaves like Block distribution but only non-zeros stored locally at compute nodes in Coordinate format (COO) – matrix values as [i, j, x], where x is the non-zero at row i and column j in the sparse matrix
- Local-to-global mapping of indices and values done at each node as a communication optimization

MSBD Overview

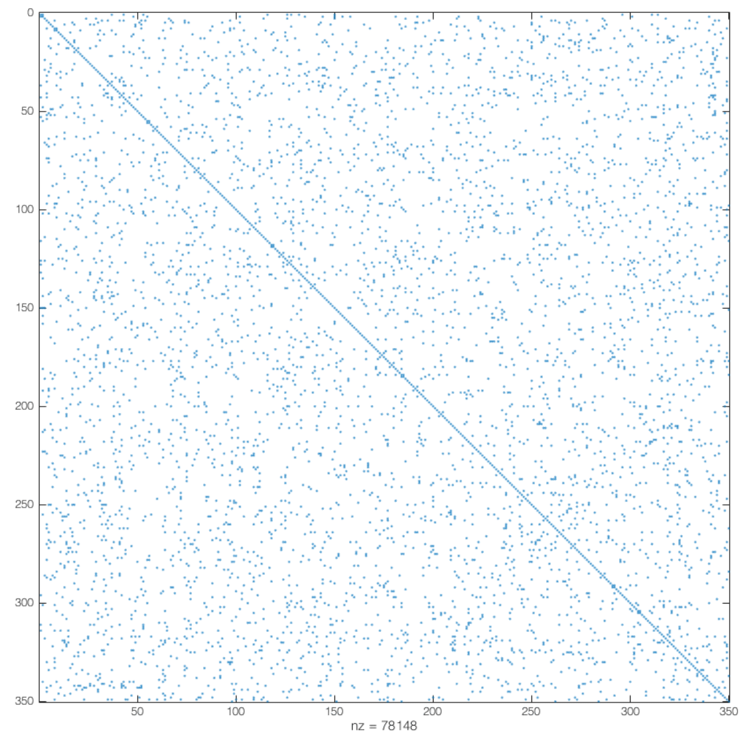
MSBD distributes sparse matrix by partitioning it over nodes:



- Sparse matrix is mapped into fixed boundary partitions to locales
- Sparse matrix values are accessed/modified only when required – reduces extra communication overhead

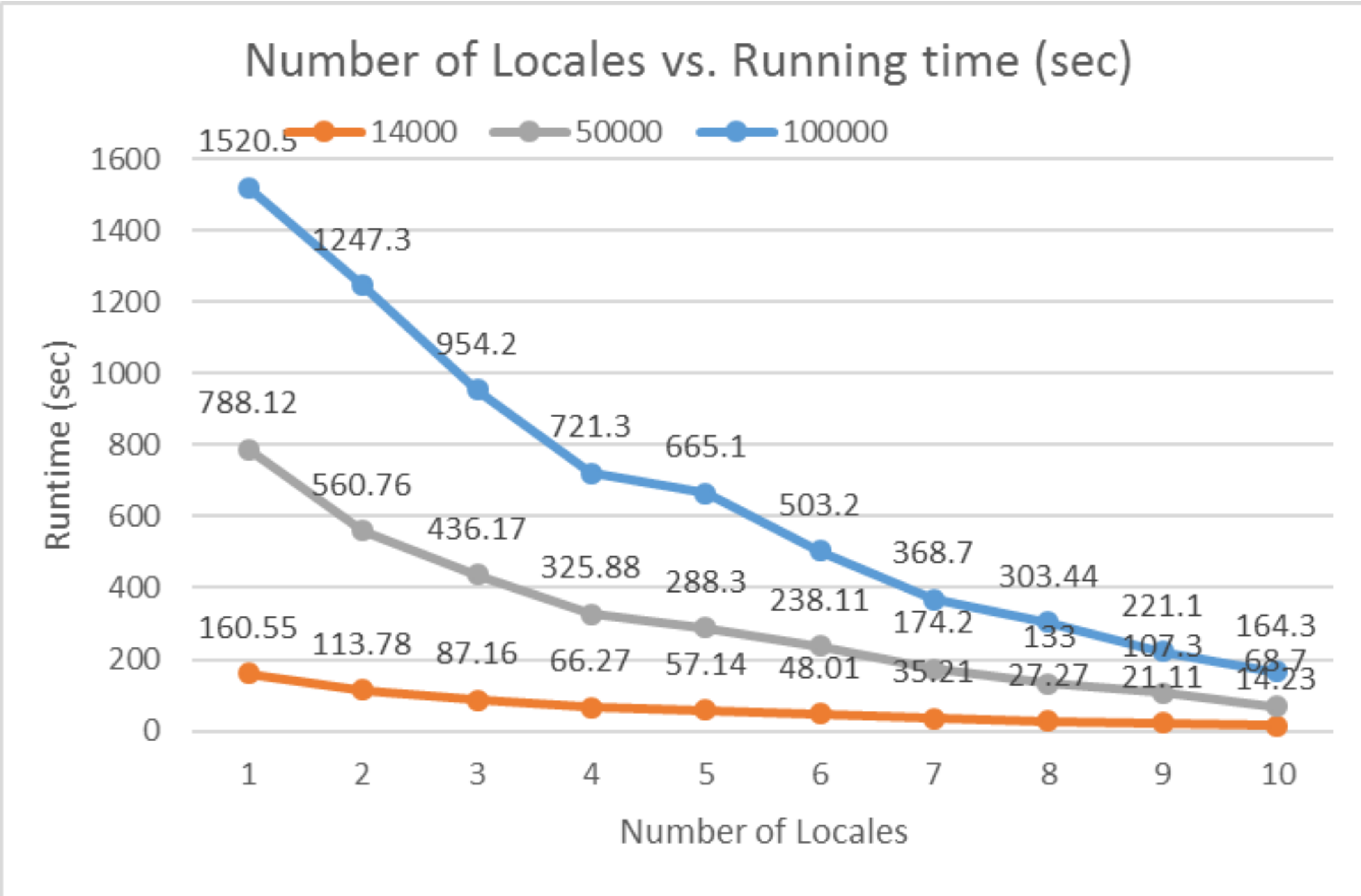
Evaluation

- MSBD evaluated by using NAS CG algorithm in Chapel
  - Sparse matrix A is a synthetic positive definite square matrix of 4% sparseness where each element $\in$ (0.0, 1.0)
  - CG algorithm consists of 25 iterations
  - At the end of iterations, final result is compared with predefined values given in NAS CG benchmark for error



- Multi-locale CG algorithm that uses MSDB is run parallel on varying number of nodes, 1 – 10. Multiple tests are done each for different size of the matrix, 14000, 50000 and 100000. The objective is to show scalability of the proposed MSBD.

Results


Number of Locales vs. Running time (sec)
14000
50000
100000
Runtime (sec)
Number of Locales
1600
1400
1200
1000
800
600
400
200
0
1
2
3
4
5
6
7
8
9
10
1520.5
1247.3
954.2
721.3
665.1
503.2
368.7
303.44
221.1
164.3
788.12
560.76
436.17
325.88
288.3
238.11
174.2
133
107.3
68.7
160.55
113.78
87.16
66.27
57.14
48.01
35.21
27.27
21.11
14.23

Conclusion

- This work presents a generalized multi-locale sparse block distribution for Chapel, MSBD
- MSBD partitions 2-D sparse data into blocks compressed in COO format that are assigned to nodes in the cluster
- Using a Chapel NAS CG algorithm, MSBD is evaluated on UMBC's Bluewave cluster and shown to be scalable

Contact

tasimon@lps.umd.edu