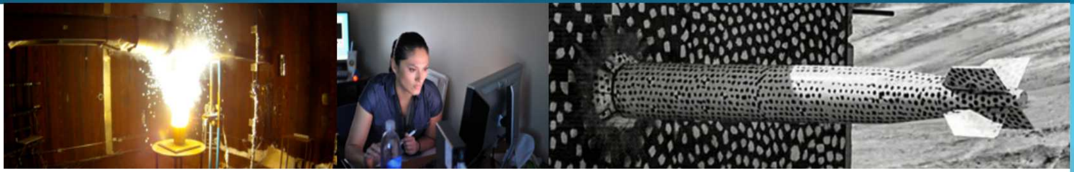
SAND2020-4931C

SPHYNX: Spectral Partitioning for HYbrid aNd aXelerator-based systems



Seher Acer – sacer@sandia.gov
Erik G. Boman – egboman@sandia.gov
Siva Rajamanickam – srajama@sandia.gov

AsHES, IPDPS -- May 18, 2020

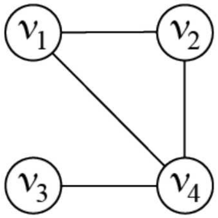
Funded by the ECP project ExaGraph

Highlights

- We propose a graph partitioning tool called Sphynx
- Sphynx uses several Trilinos packages using Kokkos for performance portability
- Sphynx is the first MPI+Cuda hybrid partitioner
- Sphynx implements a spectral partitioning approach
- Sphynx on GPUs is up to 17x faster than Sphynx on CPUs on irregular graphs
- Sphynx is up to 580x faster than the state-of-the art multilevel partitioner on irregular graphs

Graph Partitioning Problem

- Graph $G = (V, E)$: set of vertices V, set of edges E
- For the graph partitioning problem
  - each vertex v_i is assigned a weight value
  - each edge $e_{i,j}$ is assigned a cost value



- Given a K-way partition $\Pi = \{V_1, V_2, \ldots, V_K\}$ of G
  - $e_{i,j}$ is called cut if v_i and v_j are assigned to different parts
  - Cutsize of Π is the sum of the costs of the cut edges
- **Graph partitioning problem** is to find a balanced K-way partition Π of G with minimum cutsize

Motivation for Sphynx

- Applications are moving to accelerators
- No accelerator-enabled graph partitioning tool exists
- Moving the graph to CPUs, partitioning in on CPUs, and moving it back to the accelerators is not practical
- An accelerator-based portable partitioner is needed
- DoE facilities have announced different accelerators
  - AMD, Intel, NVIDIA

Spectral Partitioning in Sphynx

- Form a Laplacian matrix associated with the given graph
  - Normalized Laplacian $L_N = I - D^{-1/2}AD^{-1/2}$
- Find eigenvector x corresponding to smallest nontrivial eigenvalue λ s.t.

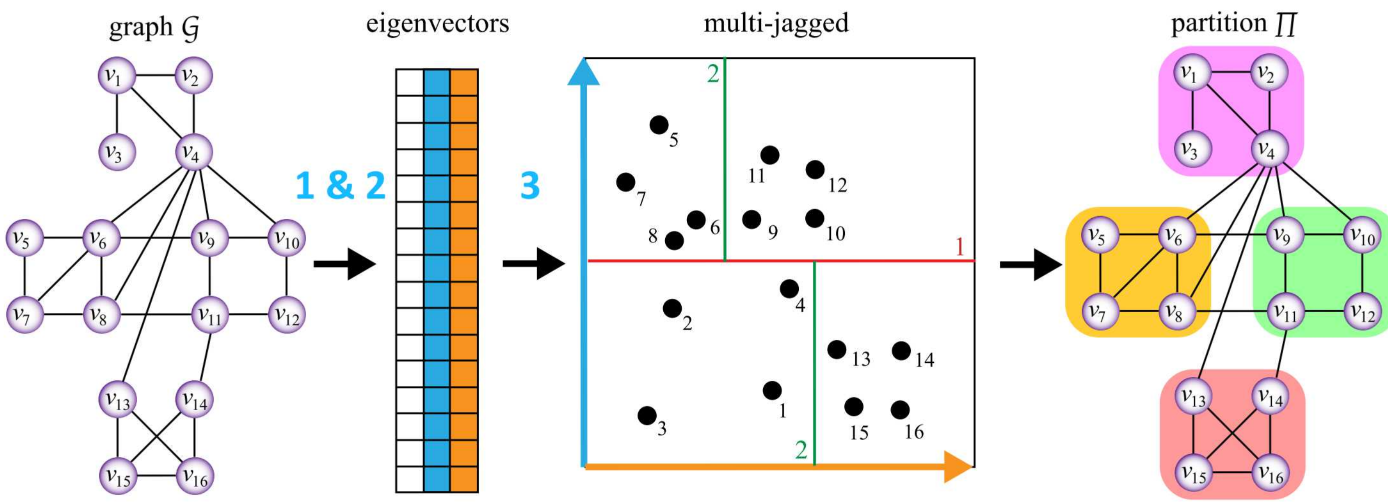
$$Lx = \lambda x$$

- Traditional spectral methods [1] use recursive bipartitioning approach
  - Find an eigenvector, bipartition the graph using it, and recursively repeat
- Sphynx finds $(\log K + 1)$ eigenvectors on the Laplacian, all at once
- Computing all eigenvectors at once avoids
  - forming subgraphs and/or corresponding Laplacians
  - moving subgraphs across different processes or nodes
  - calling eigensolver multiple times, on different graphs

[1] A. Pothen, H. Simon, and K. Liou, "Partitioning sparse matrices with eigenvectors of graphs," SIAM J. Matrix Anal., vol. 11, pp. 430–452, July 1990.

Sphynx

1. Form Laplacian L of G – Tpetra, Kokkos
2. Find $(\log K + 1)$ eigenvectors of L using LOBPCG [1] – Anasazi
   - First eigenvector: trivial, not used
   - Remaining vectors: coordinates to embed G into $\log K$-dimensional space
3. Find a K-way partition Π on coordinates using multi-jagged [2] – Zoltan2



[1] A. V. Knyazev,"Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method," SIAM Journal on Scientific Computing, vol. 23, no. 2, pp. 517–541, 2001.

[2] M. Deveci, S. Rajamanickam, K. D. Devine, and U. V. Catalyurek, "Multi-jagged: A scalable parallel spatial partitioning algorithm," IEEE Transactions on Parallel and Distributed Systems, vol. 27, pp. 803–817, March 2016.

Experiments

- Performance comparisons in terms of running time and quality
- Tested 20 highly irregular graphs from SuiteSparse collection [1]
- Evaluations:
  1. Sphynx on GPUs vs Sphynx on CPUs
  2. Parameter sensitivity of Sphynx
  3. Running time breakdown of Sphynx into three steps
  4. Sphynx vs ParMETIS [2]
- Performed on Summit at ORNL
  - Each node contains 6 NVIDIA Volta V100 GPUs
- Used Same Trilinos driver for both Sphynx and ParMETIS
  - Zoltan2 provides an interface for ParMETIS
  - 1D block partitioning is used as the distribution of the graph

[1] T. A. Davis and Y. Hu, "The University of Florida sparse matrix collection," ACM Trans. Math. Softw., vol. 38, pp. 1:1–1:25, Dec. 2011.

[2] G. Karypis and V. Kumar, "ParMETIS: Parallel graph partitioning and sparse matrix ordering library," tech. rep., Dept. Computer Science, University of Minnesota, 1997.

Experiments

- On 24 MPI processes (4 nodes)
- 10^{-2} as convergence tolerance of LOBPCG in the 1st table
- MPI+Cuda in the 2nd table

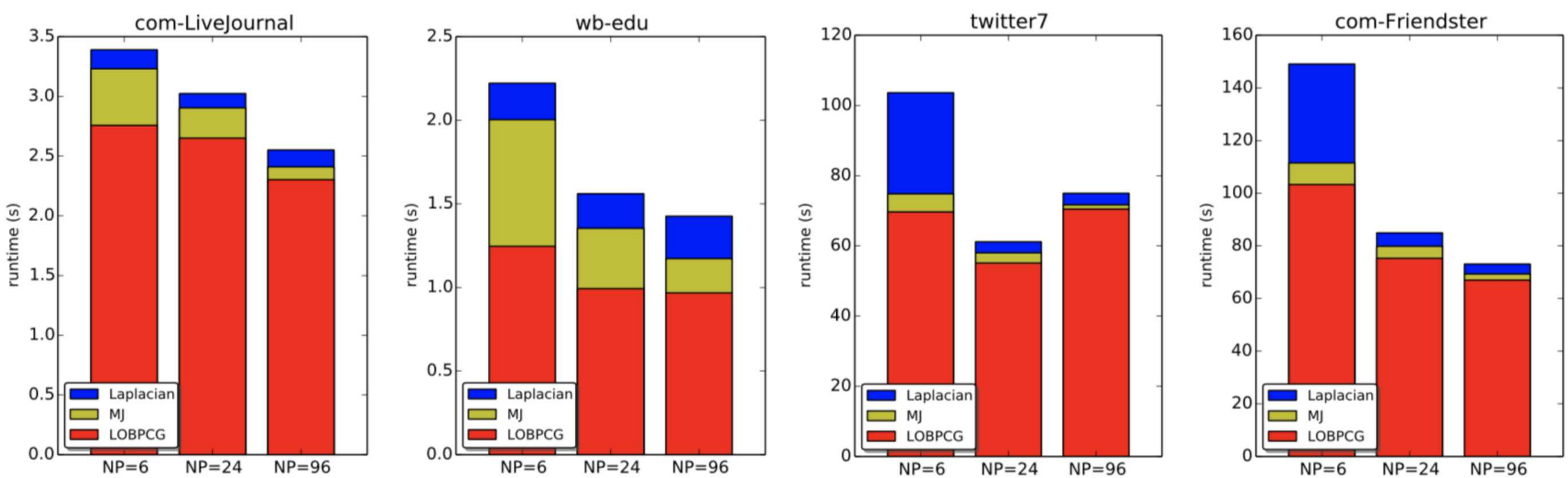
Speedup of MPI+Cuda over MPI-only		
	all graphs	**> 1M vertices**
geomean	3.27	6.83
maximum	17.77	17.77

Results of 10^{-3} normalized w.r.t. those of 10^{-2}			
	cutsize	**runtime**	**#iterations**
geomean	1.19	2.77	3.89
maximum	1.48	4.68	6.22

Experiments

- Running time breakdown into three steps
- 24 GPUs, 10^{-2} as LOBPCG tolerance

Percentages of the running times of the three steps			
	Laplacian	**LOBPCG**	**MJ**
geomean	6.2	84.2	5.0
maximum	28.1	95.7	23.1

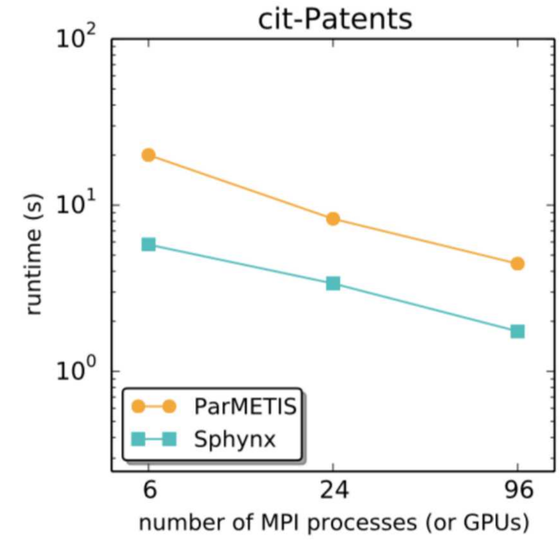


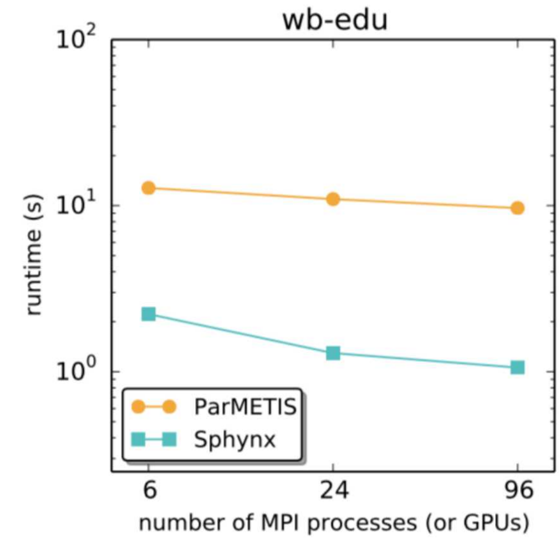
Experiments -- Results

- Sphynx on 24 GPUs vs ParMETIS [1] on 24 MPI processes
- ParMETIS does not finish in 2 hours in largest 4 graphs while Sphynx does

Sphynx compared to ParMETIS		
	cutsize deterioration	speedup
geomean	4.36	12.68
maximum	39.21	581.03

- Scalability is limited due to high irregularity of these graphs and 1D block partitioning





[1] G. Karypis and V. Kumar, "ParMETIS: Parallel graph partitioning and sparse matrix ordering library," tech. rep., Dept. Computer Science, University of Minnesota, 1997.