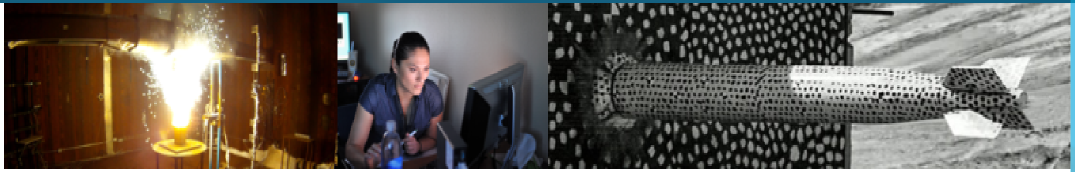
Sandia National Laboratories

SAND2021-2327C

Sphynx: a parallel multi-GPU graph partitioner for distributed-memory systems



Seher Acer
Erik Boman
Christian Glusa
Sivasankaran Rajamanickam

March 4, 2021

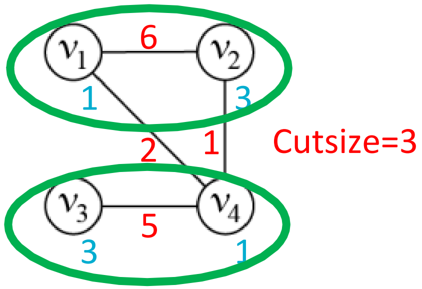
Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Sphynx – Highlights

- Sphynx: Spectral Partitioning for HYbrid aNd aXelarator-based systems
- Sphynx uses several Trilinos packages using Kokkos for performance portability
- Sphynx is the first multi-GPU partitioner for distributed-memory systems
- Compared to ParMETIS, Sphynx is faster on irregular graphs and obtains similar quality partitions on regular graphs

Sphynx – Problem Statement

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



- A K-way partition Π of G
  - is balanced if there is a balance on part weights
  - has a cutsize defined as the sum of the cut-edge costs
- Graph partitioning problem is to find a balanced K-way partition of G with minimum cutsize

Sphynx – Motivation

- Why is this problem important?
  - Used for optimizing parallel performance of scientific applications on distributed-memory systems
    - graph ↔ sparse matrix, mesh, circuit networks, social networks, …
    - vertices ↔ computational tasks
    - edges ↔ dependencies of tasks
    - parts ↔ processors
    - balancing part weights ↔ balancing processor loads
    - minimizing cutsize ↔ minimizing communication volume

Sphynx – Motivation

- We are revisiting graph partitioning problem, because:
  - Applications are moving to accelerators
  - DoE facilities have announced different accelerators
    - AMD, Intel, NVIDIA GPUs
  - No accelerator-enabled graph partitioning tool exists
  - We provide Sphynx to fill this gap
    - Distributed-memory parallel, accelerator-enabled, and portable
- Sphynx is based on a spectral approach, because:
  - Spectral methods use linear-algebra kernels, which are more amenable to parallelization on accelerators
  - Popular combinatorial partitioning methods are inherently sequential

Sphynx – Spectral partitioning

- Eigenvalue problems: combinatorial, generalized, and normalized
- Adjacency matrix $A = (a)_{ij} = \begin{cases} 1 & \text{if } e_{i,j} \in E \\ 0 & \text{otherwise} \end{cases}$
- Degree matrix $D = (d)_{ij} = \begin{cases} \deg(v_i) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$
- Form a Laplacian matrix:
  - Combinatorial Laplacian $L_C = D - A$
  - Normalized Laplacian $L_N = I - D^{-1/2} A D^{-1/2}$
- Find eigenvectors x corresponding to smallest nontrivial eigenvalues λ s.t.
  - $L_C x = \lambda x$, for combinatorial eigenvalue problem
  - $L_C x = D \lambda x$, for generalized eigenvalue problem
  - $L_N x = \lambda x$, for normalized eigenvalue problem

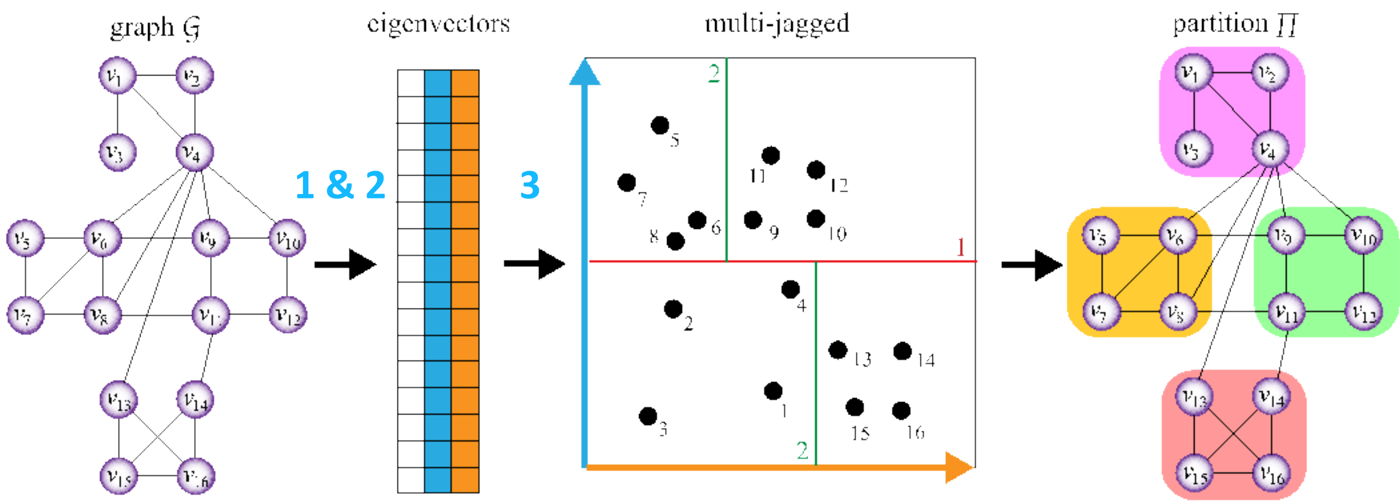
Sphynx – Spectral partitioning

- Traditional spectral methods [1] use recursive bipartitioning. At each bipartitioning step, they
  - compute one eigenvector (Fiedler vector) on the current graph
  - sort the vertices w.r.t. the entries of the eigenvector
  - bipartition the vertices according to the sorted order
- Sphynx computes $(\log K + 1)$ eigenvectors on the Laplacian, all at once
- Computing all eigenvectors at once avoids
  - forming subgraphs and/or corresponding Laplacians
  - moving subgraphs across different processes
  - calling eigensolver multiple times

[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 – Trilinos framework

1. Create Laplacian L for G – Tpetra CrsMatrix, Kokkos parallel_for
2. Compute $(\log K + 1)$ eigenvectors of L using LOBPCG [1] – Anasazi
   - First eigenvector: trivial, not used
   - Remaining vectors: coordinates to embed G into $\log K$-dimensional space
3. Compute a K-way partition 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.

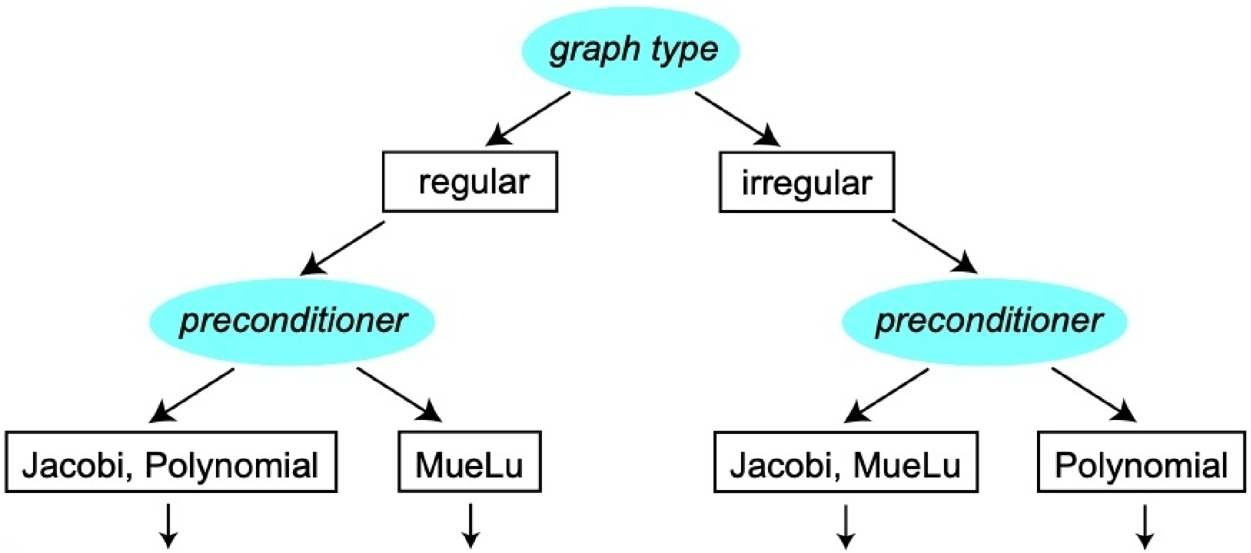
Sphynx – Preconditioning

- Number of iterations in LOBPCG is a bottleneck
- LOBPCG allows using a preconditioner
- Sphynx uses three preconditioners

1. Jacobi: $M = diag(A)^{-1}$ (Ifpack2)
   - scaling each row by the inverse of the diagonal, easy to parallelize
2. Polynomial: $M = p_k(A)$ (Belos)
   - SpMV to apply, highly parallel
   - based on GMRES polynomial
3. (Algebraic) Multigrid: $A_{\ell+1} = RA_{\ell}P$ (MueLu)
   - multilevel, captures more global information
   - costlier setup

Sphynx – Parameters

Default values for different graph types and preconditioners:



tolerance	1e-3	1e-2	1e-2	1e-2
problem type	combinatorial	combinatorial	generalized	normalized

Sphynx – Experiments

- The GPU focus: MPI+Cuda
- Performed on Summit and used 24 GPUs
  - Desired number of parts = K = 24
- Each GPU is exclusively used by one MPI rank (default)
- Device allocations in the Unified Virtual Memory (default)
- Initial distribution of the test graphs: 1D block
  - This is the default distribution with Tpetra CrsMatrix
- Parameter sensitivity and comparison against the state of the art
  - Performance metrics: cutsize and runtime

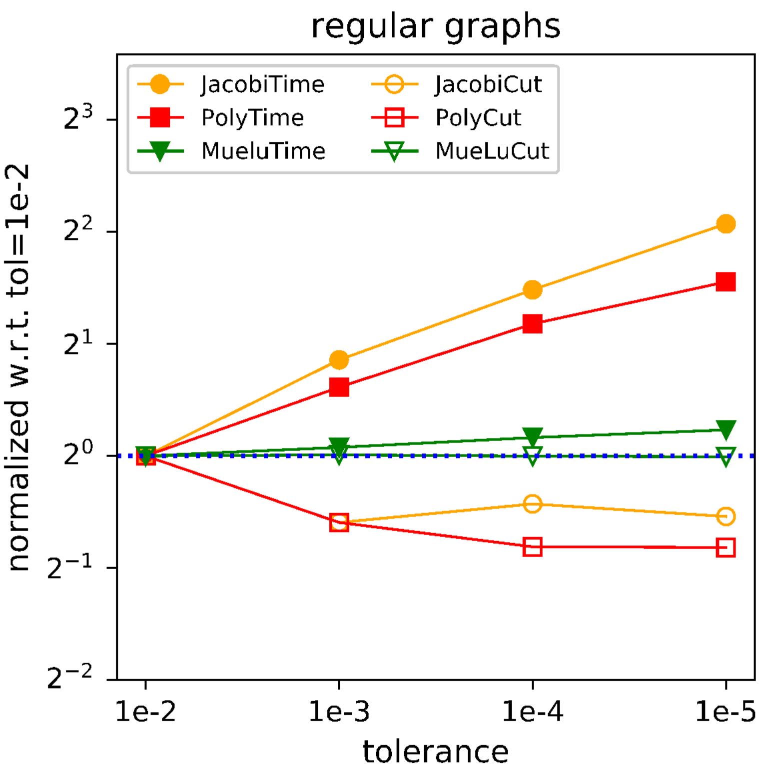
Sphynx – Dataset

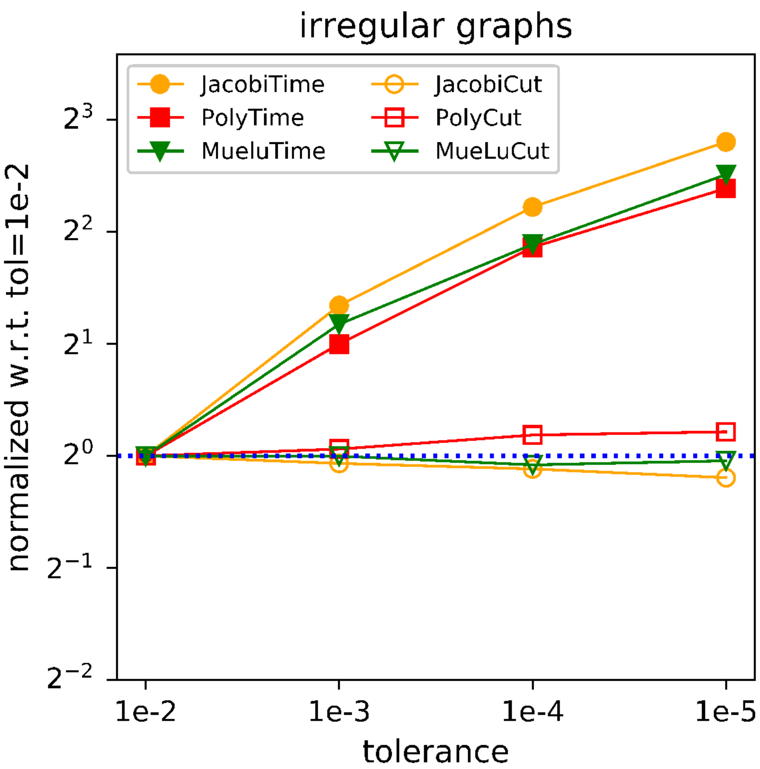
	graph	#vertices	#edges	degree	
				max	avg
regular	ecology1	1,000,000	4,996,000	5	5
	dielFilterV2real	1,157,456	48,538,952	110	42
	thermal2	1,227,087	8,579,355	11	7
	Bump_2911	2,852,430	127,670,910	195	45
	Queen_4147	4,147,110	329,499,284	81	79
	100^3	1,000,000	26,463,592	27	26
	200^3	8,000,000	213,847,192	27	27
	400^3	64,000,000	1,719,374,392	27	27
irregular	hollywood-2009	1,069,126	113,682,432	11,468	106
	com-Orkut	3,072,441	237,442,607	33,314	77
	wikipedia-20070206	3,512,462	88,261,228	187,672	25
	cit-Patents	3,764,117	36,787,597	794	10
	com-LiveJournal	3,997,962	73,360,340	14,816	18
	wb-edu	8,863,287	97,233,789	25,782	11
	uk-2005	39,252,879	1,602,132,663	1,776,859	41
	it-2004	41,290,577	2,096,240,367	1,326,745	51
	twitter7	41,652,230	2,446,678,322	2,997,488	59
	com-Friendster	65,608,366	3,677,742,636	5,215	56
	FullChip	2,986,914	26,621,906	2,312,481	9
	circuit5M	5,555,791	59,519,031	1,290,501	11

The SuiteSparse Matrix Collection, https://sparse.tamu.edu/

Sphynx – Results

LOBPCG Convergence Tolerance:





<u>Default:</u> 1e-2 for MueLu

1e-3 for others

<u>Default:</u> 1e-2 for all

Sphynx – Results

Eigenvalue Problem:

Average results normalized w.r.t combinatorial					
	preconditioner	generalized		normalized	
		runtime	cutsize	runtime	cutsize
regular	Jacobi	0.81	1.15	0.43	2.26
	Polynomial	0.73	1.21	0.54	2.45
	MueLu	0.99	1.12	0.95	2.20
irregular	Jacobi	0.75	0.83	0.26	1.36
	Polynomial	0.36	0.84	0.02	0.83
	MueLu	0.71	0.90	0.31	1.68

<u>Default:</u> combinatorial for regular graphs,

generalized for irregular graphs with Jacobi and MueLu, and

normalized for irregular graphs with Polynomial.

Sphynx – Results

Preconditioner:

Average results normalized w.r.t. Jacobi				
	Polynomial		MueLu	
	runtime	cutsize	runtime	cutsize
regular	0.46	1.03	**0.42**	**0.91**
irregular	**0.62**	**1.71**	1.91	0.94

<u>Suggested:</u> MueLu for regular graphs,

Polynomial for irregular graphs.

Sphynx – Results

- Comparison against ParMETIS [1] and XtraPuLP [2]
  - ParMETIS and XtraPuLP **do not run** on GPUs
- Application-friendly comparison on 24 MPI ranks
  - Sphynx uses 6 MPI ranks per node and 1 GPU per rank
  - ParMETIS uses 6 MPI ranks per node
  - XtraPuLP uses 6 MPI ranks per node and 7 OpenMP threads per rank

Average results normalized w.r.t Sphynx				
	ParMETIS		XtraPuLP	
	runtime	cutsize	runtime	cutsize
regular	0.33	0.81	0.31	6.36
irregular	23.95	0.30	1.24	0.45

- ParMETIS execution **did not finish** in 2 hours on 4 graphs
  - Largest irregular graphs: uk-2005, it-2004, twitter7, com-Friendster

[1] G. Karypis, V. Kumar, Parmetis: Parallel graph partitioning and sparse matrix ordering library, Tech. rep., Dept. Computer Science, University of Minnesota, 1997.
[2] G. M. Slota, S. Rajamanickam, K. Devine, K. Madduri, Partitioning trillion-edge graphs in minutes, IPDPS, 2017.

Sphynx – Results

- Comparison against nvGRAPH’s spectral partitioner [1]
  - runs on a single GPU
  - minimizes a ratio cut metric, does not enforce strict balancing
- Sphynx: on a single MPI rank (i.e., on a single GPU)
- Number of parts: 24

Average results of Sphynx normalized w.r.t. nvGRAPH			
	runtime	cutsize	max part weight
regular	0.45	0.94	0.54

- nvGRAPH did not run on large graphs (most irregular graphs)

[1] M. Naumov, T. Moon, Parallel spectral graph partitioning, Tech. rep., NVIDIA tech. rep. NVR-2016-001 (2016).

Sphynx – Conclusion

- First multi-GPU partitioner on distributed-memory systems
  - Many knobs to tune the performance: preconditioners, problem type, etc.
- Built on top of other Trilinos packages, intelligent code reuse
  - Improvements in Anasazi, MueLu, Tpetra, etc. will improve Sphynx
- Released as a subpackage of Zoltan2 in Trilinos:

  https://github.com/trilinos/Trilinos/tree/master/packages/zoltan2/sphynx