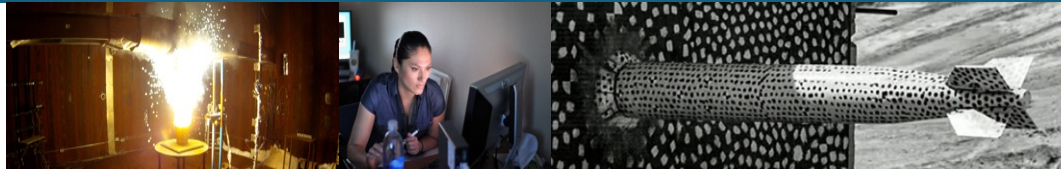




Sandia  
National  
Laboratories

# Randomized Linear Algebra can Accelerate Graph Partitioning



*Erik G. Boman*  
Heliezer Espinoza  
Jennifer Loe

SIAM CS&E 23, Amsterdam, NL  
Feb. 27 – Mar. 3, 2023



Sandia National Laboratories is a multi-mission 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.

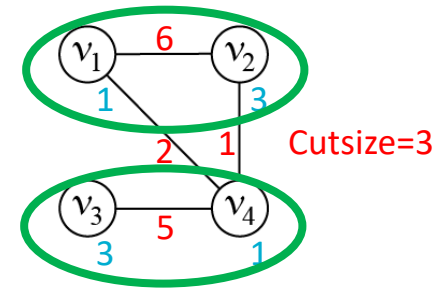


- Graph Partitioning
- Spectral Partitioning
- Sphynx: A Parallel Spectral Partitioner
- A Randomized Eigensolver
- Results
  
- *Thanks to Seher Acer for initial version of Sphynx*

# 3 Graph Partitioning Background



- 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**  $\Pi$  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**



# Motivation - Sphynx



- 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**
  - Can potentially speed up algorithm with **randomized linear algebra**

# Background: Spectral partitioning

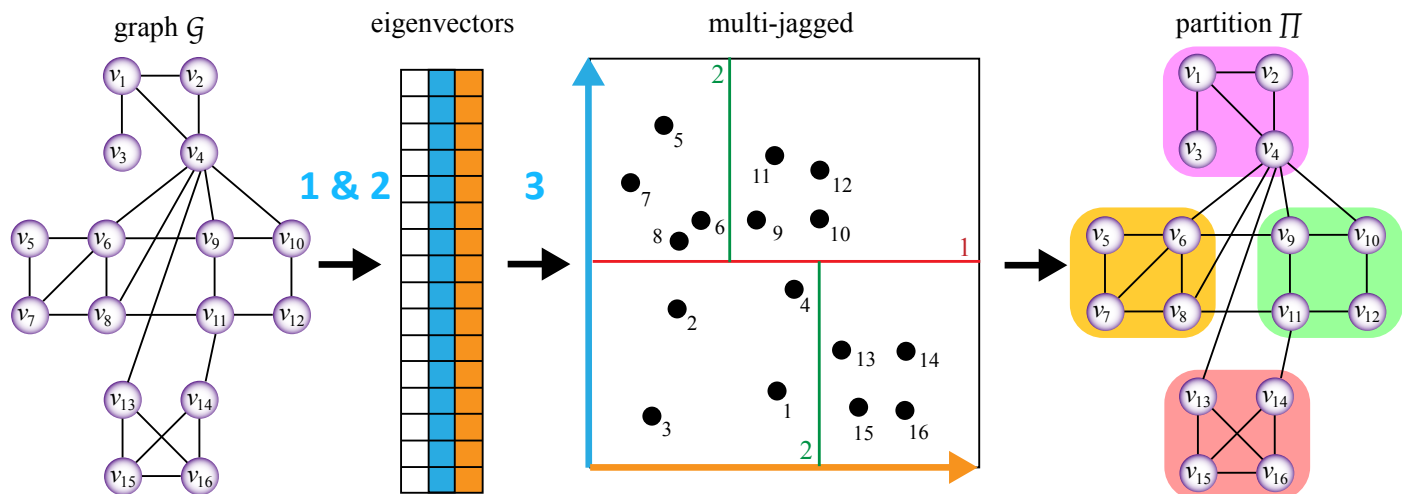


- Proposed by Pothen, Simon, Liou ('90)
  - Based on Donath & Hoffman ('73), Fiedler ('73)
- Eigenvalue problems: **combinatorial**, and **normalized**
- Adjacency matrix  $A = (a)_{ij} = \begin{cases} 1 & \text{if } e_{i,j} \in E \\ 0 & \text{otherwise} \end{cases}$
- Degree matrix  $D = (d)_{ij} = \begin{cases} \text{deg}(v_i) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$
- Form a Laplacian matrix:
  - Combinatorial** Laplacian  $L_C = D - A$
  - Normalized** Laplacian  $L_N = I - D^{-1/2}AD^{-1/2}$
- Find eigenvectors  $x$  corresponding to smallest nontrivial eigenvalues  $\lambda > 0$  s.t.
  - $L_C x = \lambda x$ , for **combinatorial** eigenvalue problem
  - $L_N x = \lambda x$ , for **normalized** eigenvalue problem

# 6 Sphynx: Parallel partitioner in Trilinos



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.

# 7 Sphynx – Preconditioning



- Number of iterations in LOBPCG is a bottleneck
- LOBPCG allows using a **preconditioner**
- Sphynx supports three preconditioners
  1. Jacobi:  $M = \text{diag}(A)^{-1}$  (**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 – Experiments



- The GPU focus: MPI+Kokkos (Cuda/HIP)
- 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



# 9 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

# 10 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	cutsizes	runtime	cutsizes
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.

# 11 Randomized Eigensolvers



- The most expensive phase (90-95%) in spectral partitioning is the eigensolver
- Fairly low accuracy is sufficient to obtain good partitioning
- Key idea: We can use a randomized eigensolver instead of LOBPCG
  - Randomized methods often get low-accuracy solutions very fast
  - We follow the approach in Halko, Martinsson, Tropp (20XX)
- We have explored this approach in a prototype
  - HPC implementation in Trilinos/Sphynx still to do

# Randomized Method: Phase I



- Here we will use the normalized Laplacian,  $L_N$
- We estimate the largest eigenvalues of the normalized adjacency matrix, which correspond to the smallest eigenvalues of  $L_N$
- First, we approximate the range of  $A_N$ , where  $A_N = D^{-1/2} A D^{-1/2}$ .
- Draw a random Gaussian (normal) matrix  $\Omega$
- Form  $Y = A_N^q \Omega$
- Compute skinny QR:  $QR = Y$

# Randomized Method: Phase 2

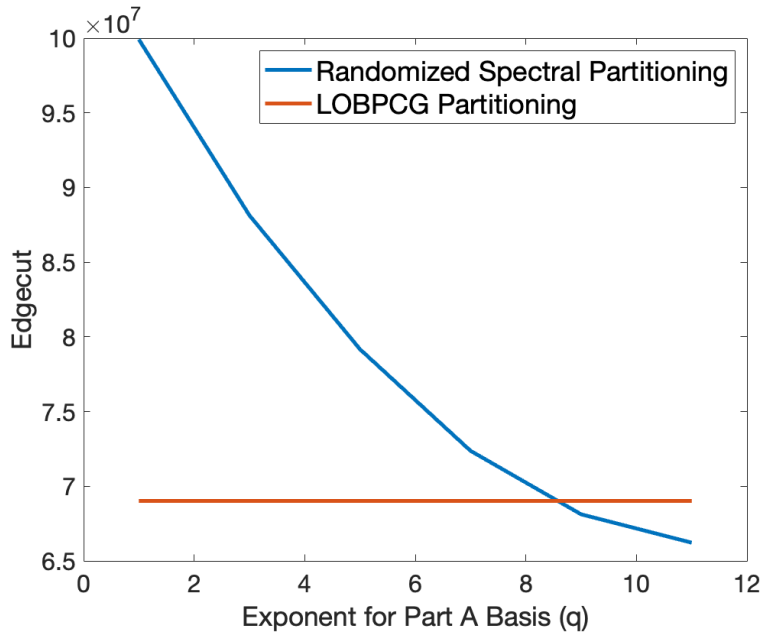


- Second, compute eigenvalues on the projected problem.
- Compute projection  $B = Q^T A Q$
- Solve eigenproblem for B:  $B = V \tilde{\lambda} V^T$
- Project back:  $U = QV$
  
- We only need to solve a small, dense eigenproblem for B
  - Fast!
  - No longer need LOBPCG (or any sparse eigensolver)

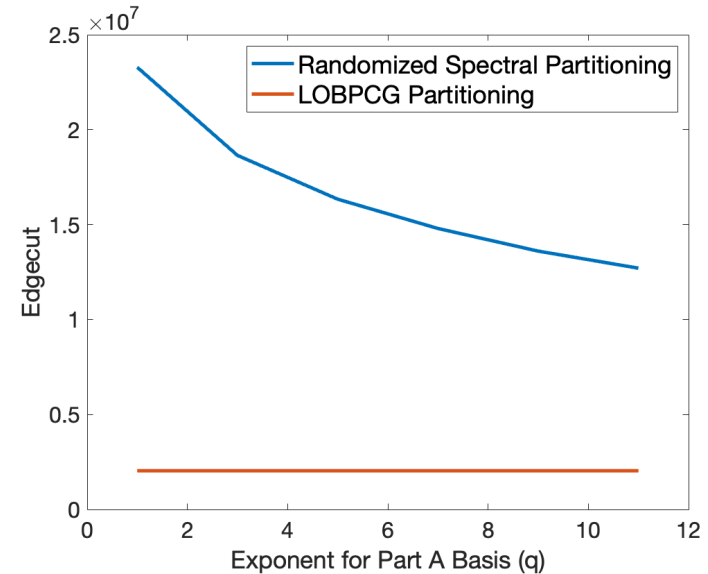
# Randomized partitioning quality



## Irregular: Hollywood



## Regular: Brick3d



The randomized eigensolver actually works *better* than LOBPCG for partitioning with sufficiently large  $q$  (for irregular graphs)!

# Results: Quality



Sphynx edge cuts with LOBPCG vs randomized solver with  $L=20$ .

Matrix	LOBPCG	q=1	q=3	q=5	q=7	q=9	q=11
cube100	2,036,942	23,280,878	18,643,406	16,339,232	14,763,242	13,620,082	12,645,466
hollywood	69,010,728	99,913,222	88,136,872	79,170,890	72,369,774	68,120,278	66,220,826
wikipedia	70,372,336	82,607,424	80,328,958	77,783,740	74,965,720	71,660,478	69,205,042
FullChip	19,837,100	21,925,964	18,723,240	17,296,392	16,483,600	15,890,706	15,466,674
Circuit5M	40,918,466	52,793,074	49,117,806	42,290,548	33,731,544	31,757,878	31,563,734

Randomized method (q=11) often gives lower (better) cuts than LOBPCG !

# 16 Results: Run Time



Sphynx run times (CPU) with LOBPCG vs randomized solver with L=20.

Matrix	LOBPCG	q=1	q=3	q=5	q=7	q=9	q=11
cube100	18.19	1.601	2.185	2.779	3.357	3.907	4.465
hollywood	360.6	5.865	10.88	15.69	20.44	25.33	30.35
wikipedia	923.3	18.59	33.28	47.75	62.15	76.45	90.84
FullChip	162.8	5.00	6.24	7.50	8.79	10.13	11.38
Circuit5M	821.5	11.14	13.98	16.23	18.27	21.04	23.93

Our randomized method is 10X-80X faster (q=1) than LOBPCG!  
Also much faster with q=11 (good cuts)





- Randomized eigensolver can dramatically speed up (5-80X) a spectral partitioner
- Works well for many irregular graphs (e.g., web graphs)
  - but not so well for more regular graphs (e.g., meshes)
  - Perhaps related to separation of the Laplacian eigenvalues?
- Trade-off in computational cost vs quality
- Sphynx has been released in Trilinos/Zoltan2
  - Randomized method in progress (will be released soon)
- All spectral methods have some weaknesses
  - Often slower than multilevel methods and sometimes worse cuts
  - Collaboration with K. Madduri and M. Gilbert (Penn State) may address this
- Future work:
  - Add refinement: Will improve cut quality but take more time
  - Alternatives to Multijagged geometric partitioning (QRCP?)

This research was supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration.

# Extra slides



# Why Revisit Spectral Methods Now?



Weren't they abandoned in the '90s and replaced by multilevel methods? Yes, but...

- Quality of spectral partitioners are often only slightly worse than multilevel.
- Preconditioned eigensolvers (LOBPCG) came later.
- Linear algebra operations have been optimized for GPU.
- Spectral partitioning is robust for large #proc (#GPU), as eigenvectors don't change

# Sphynx – Spectral partitioning



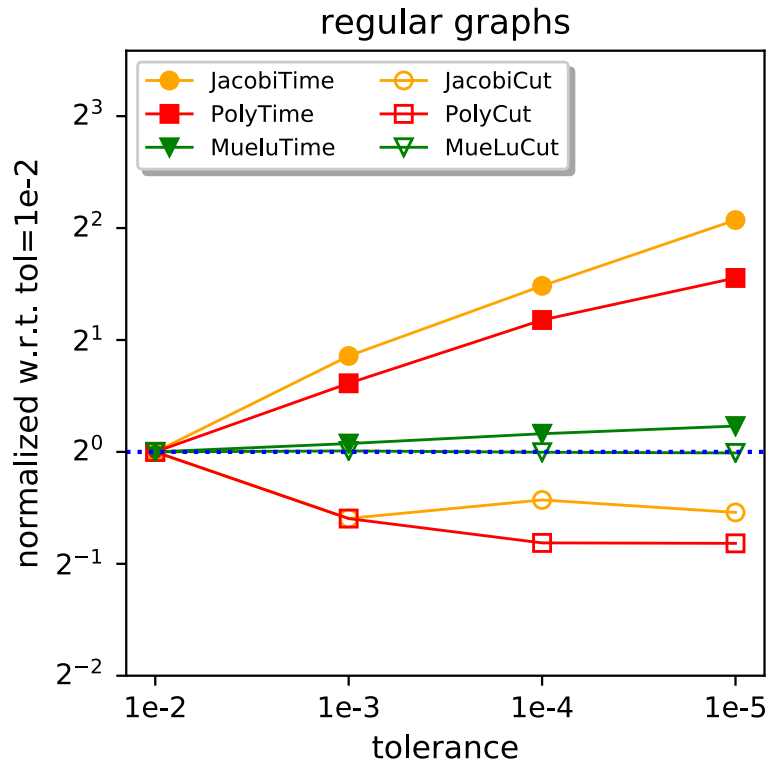
- Traditional spectral methods [1] use recursive bipartitioning. At each bipartitioning step, they
  - compute one eigenvector (Fiedler vector) on the current graph
  - sort the vertices w.r.t. the entries of the eigenvector
  - bipartition the vertices according to the sorted order
- Sphynx computes  $(\log K + 1)$  eigenvectors of 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.

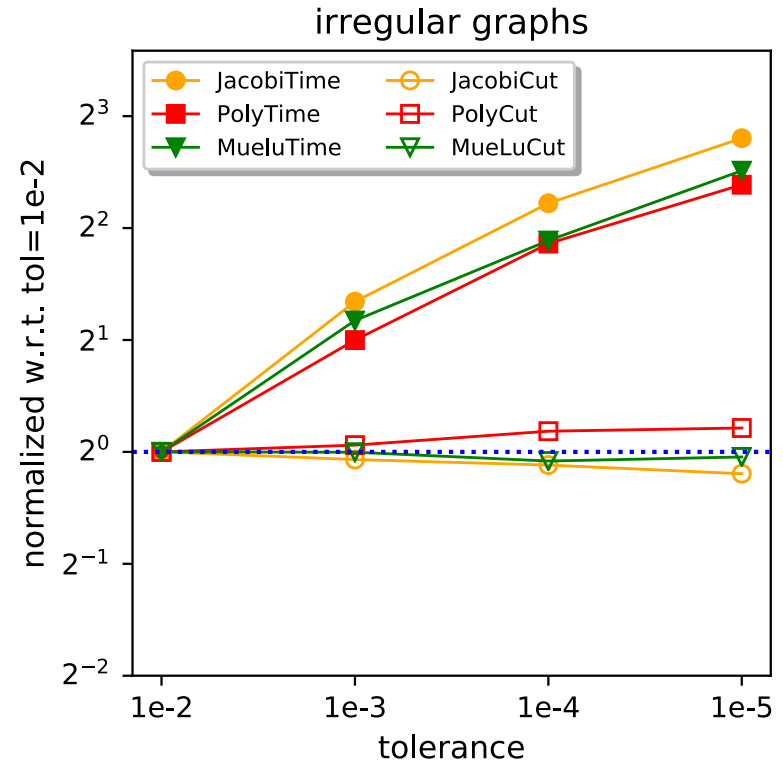
# Sphynx – Results



## LOBPCG Convergence Tolerance:



Default: 1e-2 for MueLu  
1e-3 for others



Default: 1e-2 for all



Eigenvalue Problem:

Average results normalized w.r.t combinatorial					
	preconditioner	generalized		normalized	
		runtime	cutsizes	runtime	cutsizes
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

Default: combinatorial for regular graphs,  
 generalized for irregular graphs with Jacobi and MueLu, and  
 normalized for irregular graphs with Polynomial.



Preconditioner:

Average results normalized w.r.t. Jacobi				
	Polynomial		MueLu	
	runtime	cutsizes	runtime	cutsizes
regular	0.46	1.03	<b>0.42</b>	<b>0.91</b>
irregular	<b>0.62</b>	<b>1.71</b>	1.91	0.94

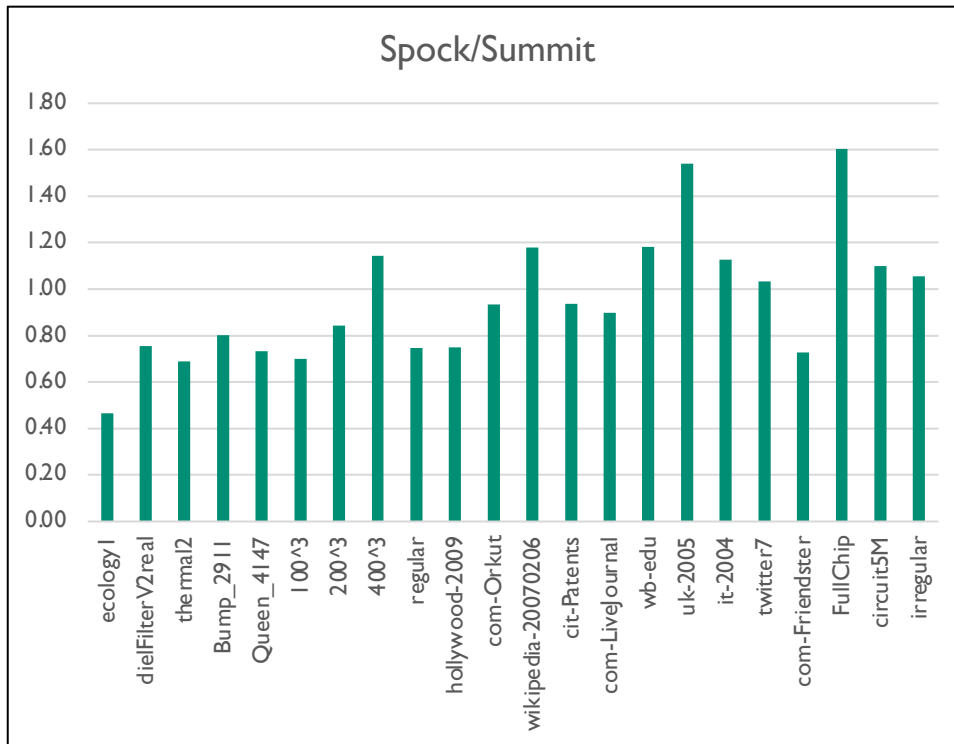
Suggested: MueLu for regular graphs,  
Polynomial for irregular graphs.

# Sphynx: Exascale Systems



Sphynx Running Time on OLCF systems:

- Summit: Nvidia Volta V100
- Spock: AMD MI200



	RUNNING TIME (s)		
	Summit	Spock	Spock/Summit
ecology1	1.40	0.65	0.47
dielFilterV2real	2.13	1.61	0.75
thermal2	1.78	1.22	0.69
Bump_2911	1.68	1.35	0.80
Queen_4147	2.20	1.61	0.73
100^3	1.39	0.97	0.70
200^3	2.11	1.78	0.84
400^3	6.78	7.75	1.14
<b>geomean</b>			<b>0.75</b>
hollywood-2009	4.79	3.60	0.75
com-Orkut	8.06	7.52	0.93
wikipedia-20070206	15.66	18.44	1.18
cit-Patents	8.27	7.75	0.94
com-LiveJournal	8.70	7.81	0.90
wb-edu	5.54	6.55	1.18
uk-2005	89.31	137.59	1.54
it-2004	90.24	101.58	1.13
twitter7	482.85	499.43	1.03
com-Friendster	186.16	135.20	0.73
FullChip	48.36	77.56	1.60
circuit5M	43.55	47.87	1.10
<b>geomean</b>			<b>1.05</b>