Fast Spectral Graph Layout on Multicore Platforms

Ashirbad Mishra, Penn State

Shad Kirmani, eBay

Kamesh Madduri, Penn State

INTERNATIONAL CONFERENCE ON PARALLEL PROCESSING

ICPP/2020/EDMONTON/CANADA

AUGUST 17-20, 2020

Outline

Introduction

HDE

Related Work

Methodology: Parallelization and Extensions

Experiments and Results

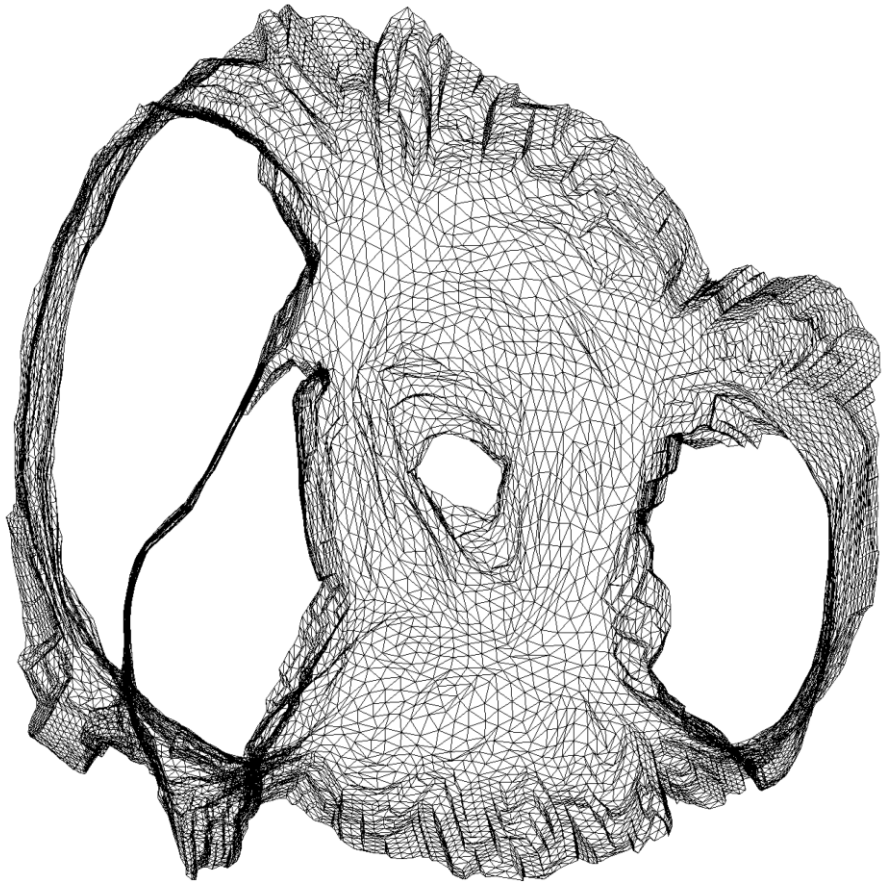
Conclusions and Future Work

Introduction

High Dimensional Embedding (HDE), a spectral graph drawing algorithm

Our work: ParHDE, fast multicore parallelization of HDE

18x speedup over prior state-of-the-art, can process graphs with billions of edges in minutes



ParHDE drawing of barth5

HDE

Constrain drawing axes to lie in a well-chosen subspace

Can be thought of as an approximate spectral graph drawing algorithm

Three phases

- Distance matrix D (breadth-first search from s sources)
- Distance matrix orthogonalization to form S
- Triple product $S^T L S$ (L is graph Laplacian)

Prior Work

An Eigen-based implementation of HDE (DOI: 10.1109/IPDPSW.2018.00053)

PHDE (DOI: 10.1007/3-540-36151-0_20), PCA-based HDE, does not require LS product

PivotMDS (DOI: 10.1007/978-3-540-70904-6_6), another graph drawing algorithm computationally identical to HDE

PHDE and PivotMDS can be viewed as special cases of HDE

ParHDE high-level details

Decouple BFS and Orthogonalization phases

Use direction-optimizing BFS, modifying implementation from the GAP benchmark suite

Pivots chosen using 2-approx. solution to *k*-centers problem

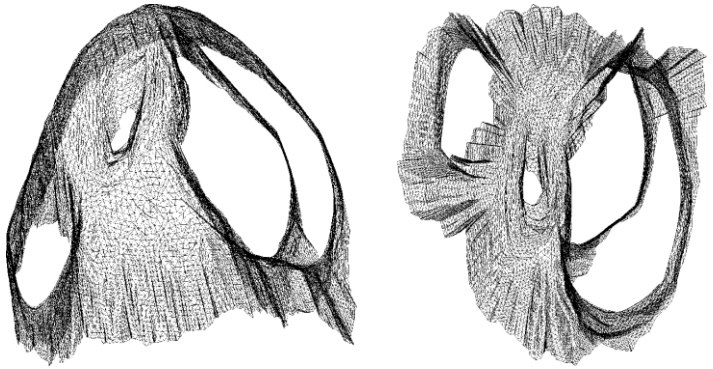
Orthogonalization, experimented with both classical and modified Gram-Schmidt procedures

ParHDE high-level details

LS computation: sparse matrix times a dense vector (SpMM), S not stored explicitly, can be viewed as s sparse matrix vector multiplications (SpMVs)

S^TB computation (where B = LS) using Intel BLAS dgemm

ParHDE extended to encompass PHDE and PivotMDS as well



PHDE and PivotMDS drawings of barth5

Extensions

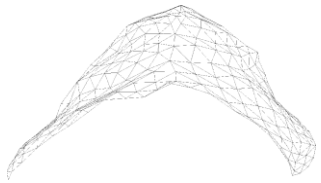
Choosing pivots at random instead of k-centers

Performing BFSes concurrently

Weighted graphs: single-source shortest paths instead of BFS and modified LS computation

Blocked Coarse Gram-Schmidt

"Zoomed" view

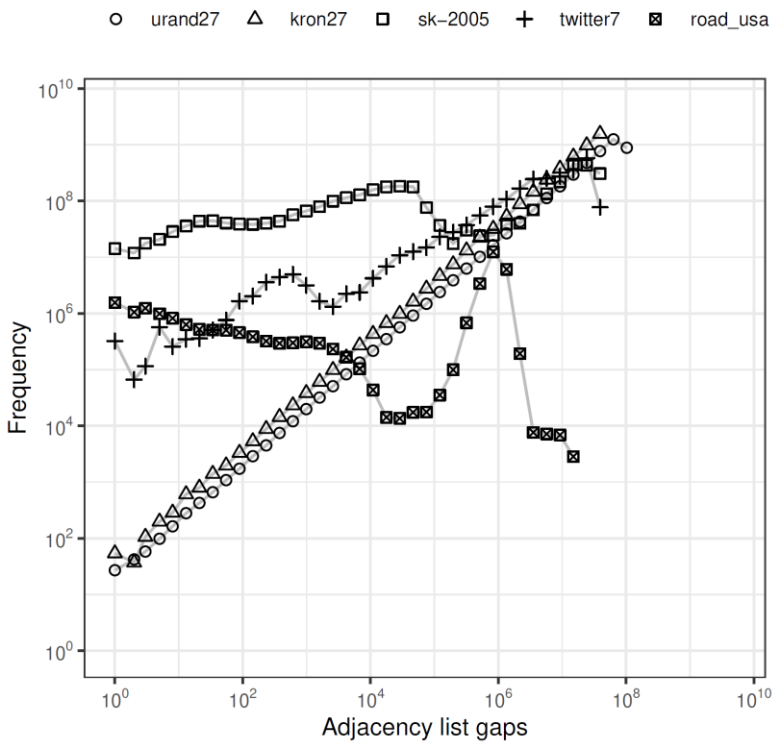


10-hop zoom from a vertex in barth5

Experimental setup

Perf. results on a 28-core Intel Xeon E5-2695 v3 system

Graph	m	n
urand27	2 147 483 376	134 217 728
kron27	2 111 622 405	63 045 458
sk-2005	1 810 050 743	50 634 118
twitter7	1 202 513 046	41 652 230
road_usa	28 854 312	23 947 347
cage14	12 812 282	1 505 785
CurlCurl_4	12 067 676	2 380 515
kkt_power	6 482 320	2 063 494
ecology1	1 998 000	1 000 000
pa2010	1 029 231	421 545

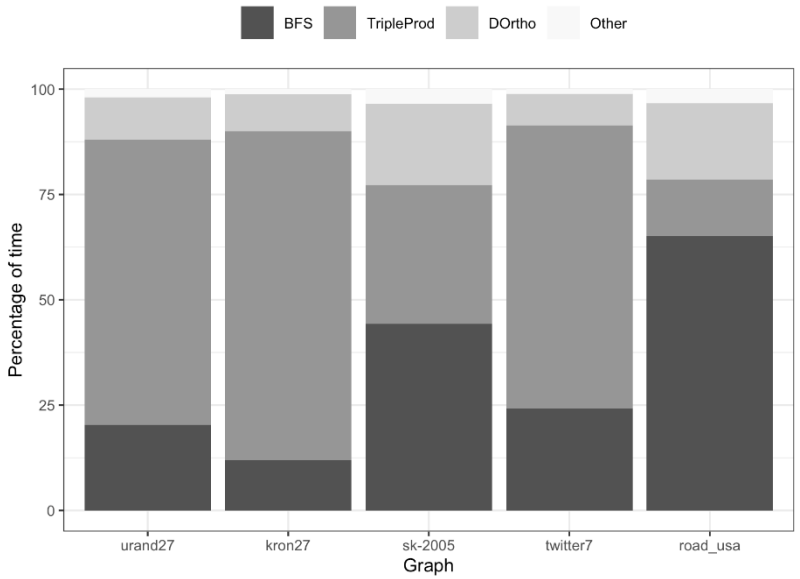


Comparison to prior art and rel. speedup

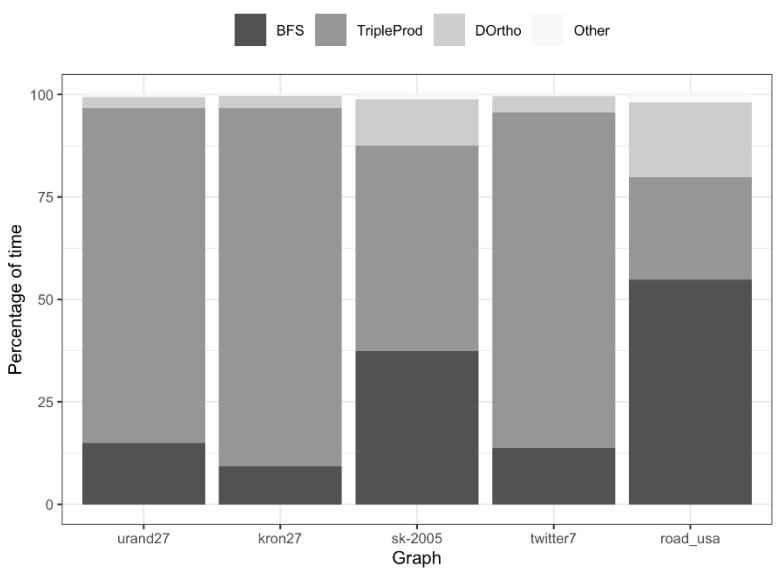
Graph	Time (s)		Speedup
	ParHDE	Prior Par.	
urand27	72	1301	18.0×
kron27	47	688	14.7×
sk-2005	18	131	7.3×
twitter7	34	372	10.9×
road_usa	13	36	2.9×

Graph	Time (s)	Rel. Speedup
urand27	52.5	24.5×
kron27	34.3	14.8×
sk-2005	9.9	11.3×
twitter7	23.8	11.0×
road_usa	4.6	7.1×
CurlCurl_4	0.6	5.8×
kkt_power	0.5	8.1×
cage14	0.3	9.1×
ecology1	0.3	4.2×
pa2010	0.1	4.2×

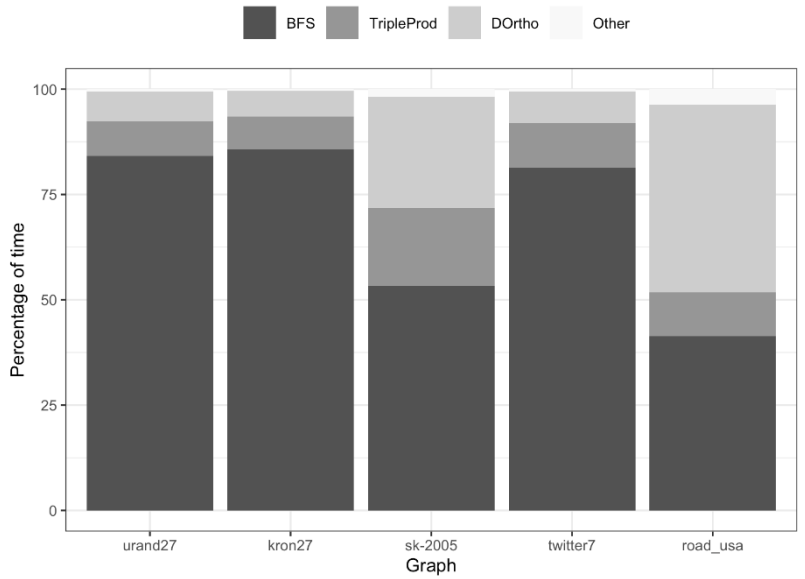
Execution time breakdown



ParHDE 28-core

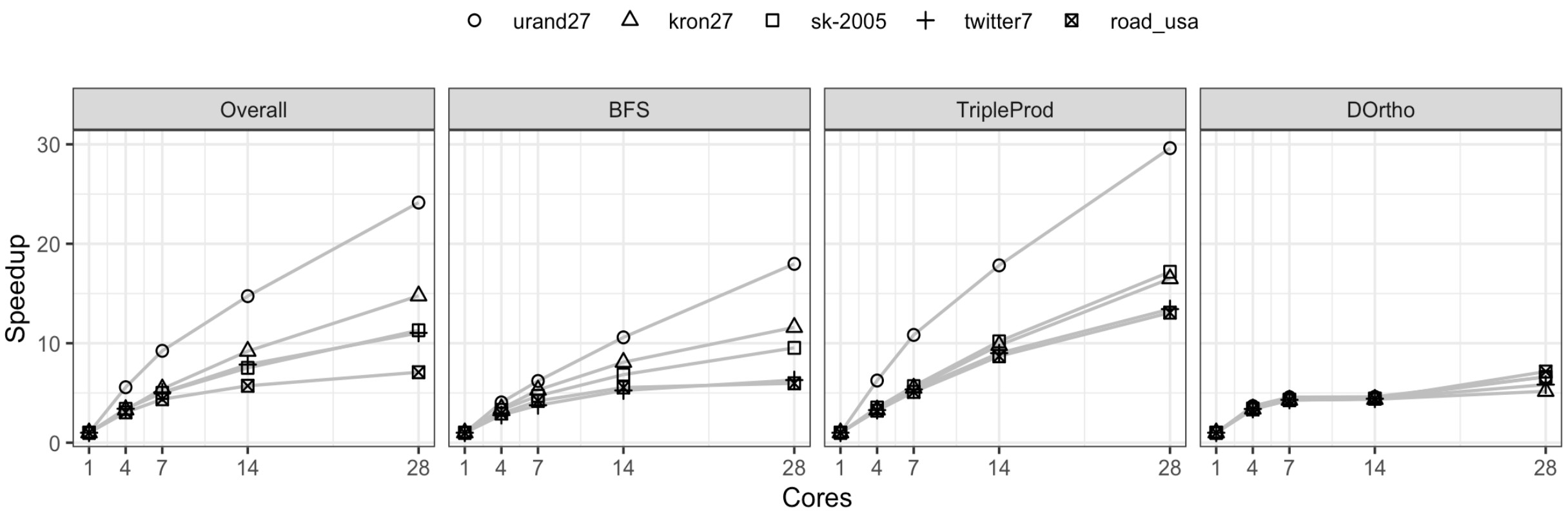


ParHDE serial

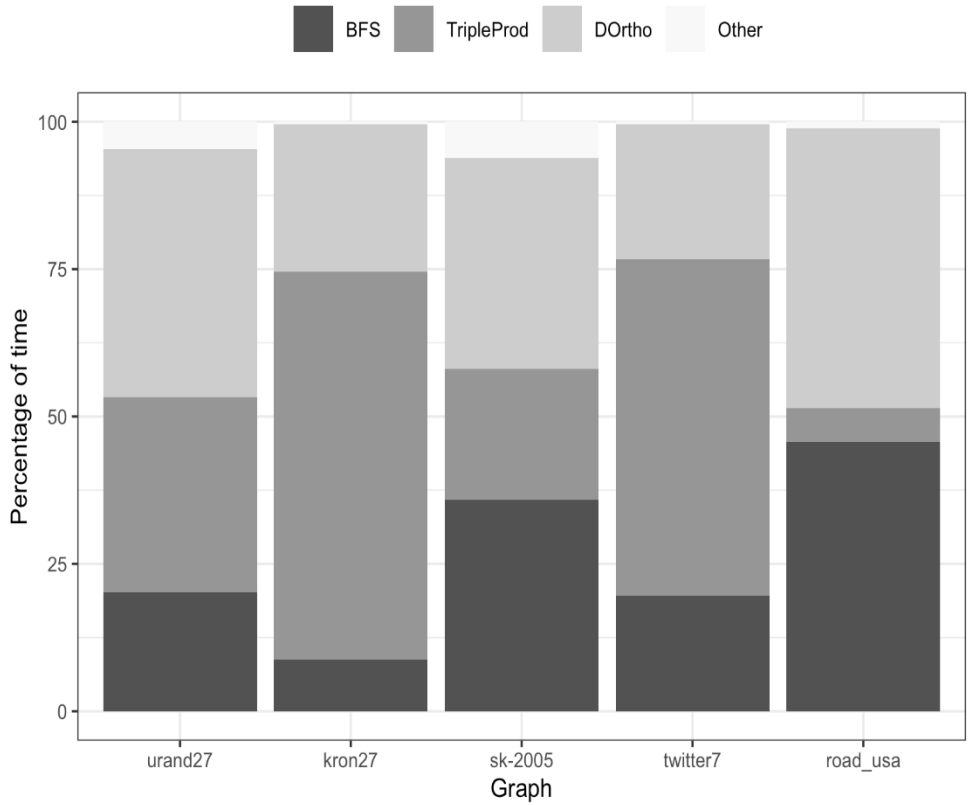


Prior art on an 80-core system

Parallel scaling



ParHDE additional analysis

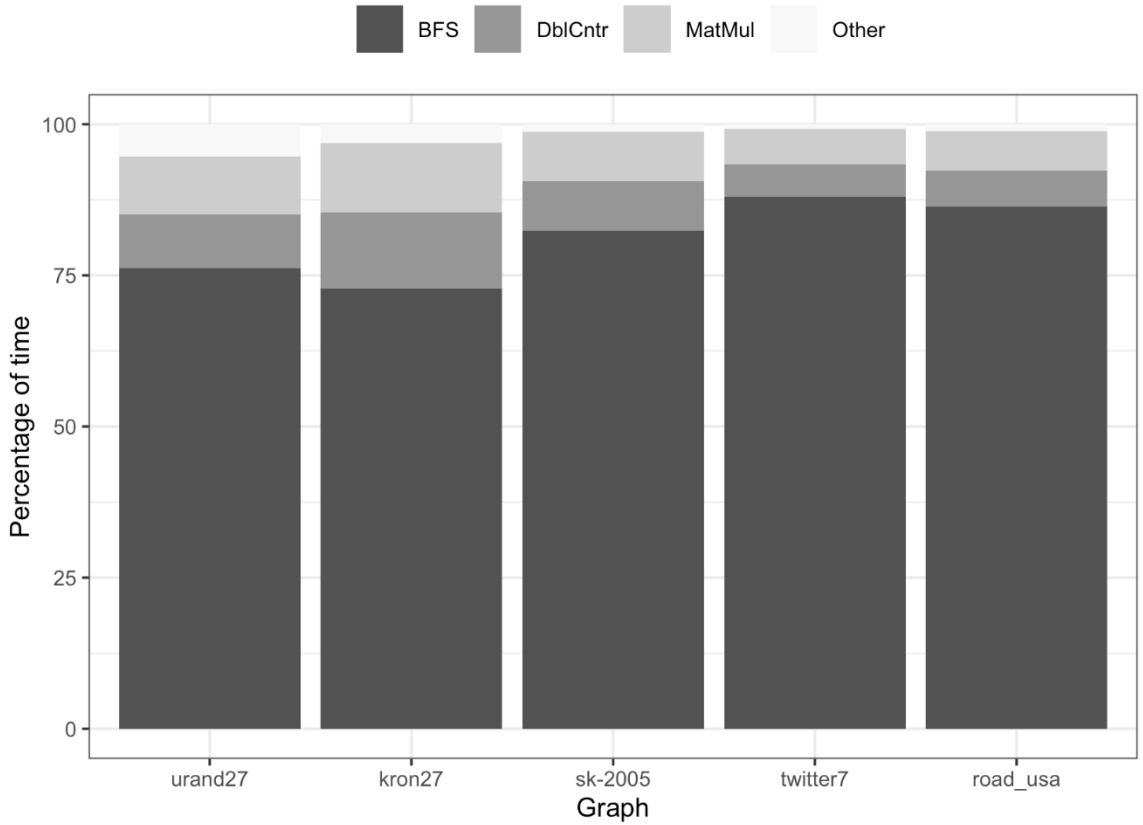


ParHDE (s = 50)

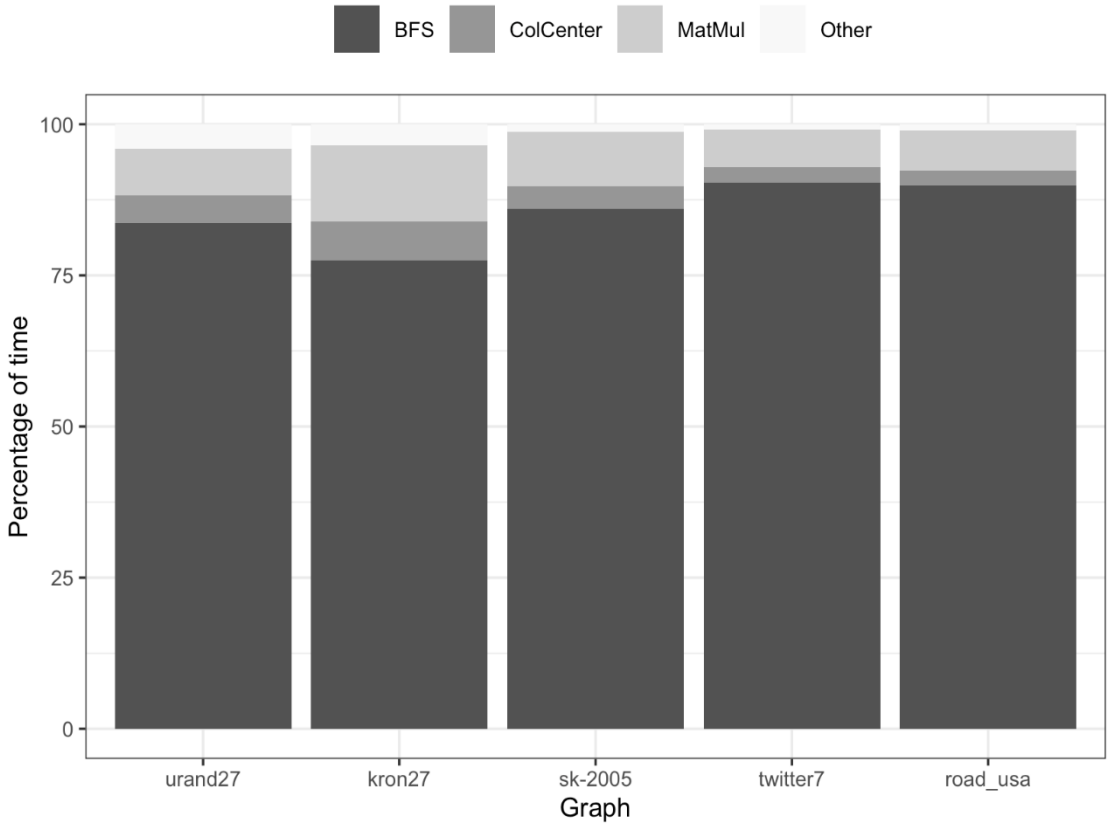
Graph	Default Alg. (MGS) Time(s)	CGS Time (s)	Rel. Speedup
urand27	5.9	2.7	2.2×
kron27	3.0	1.1	2.8×
sk-2005	2.0	0.8	2.5×
twitter7	1.8	0.7	2.5×
road_usa	0.8	0.4	2.1×

Using CGS instead of MGS for orthogonalization

PivotMDS and PHDE performance



PivotMDS



PHDE

Extensions: Performance Analysis

Our implementation of LS is on average 2.5x faster than *mkl_sparse_d_mm*

On a weighted version of the road_usa graph, the SSSP phase is 3.7x slower than a BFS phase on the unweighted graph

Graph	Default Alg. Time (s)	Rand. Pivots Time (s)	Rel. Speedup
CurlCurl_4	0.91	0.33	2.8×
kkt_power	1.10	0.66	1.7×
cage14	0.66	0.47	1.4×
ecology1	0.88	0.09	10.1×
pa2010	0.42	0.05	9.1×

Performance impact of picking pivots at random

Conclusions and Future Work

ParHDE can generate coordinates for a 2.1 billion-edge graph in 53 seconds

We analyze performance impact of number of sources s

SpMM performance is dependent on vertex labeling

Future work: preprocessing for eigensolvers and graph partitioning

Acknowledgments

National Science Foundation (NSF) grants CCF-1439057 and OAC-1253881

NSF XSEDE project for access to the Bridges supercomputer

Reviewers