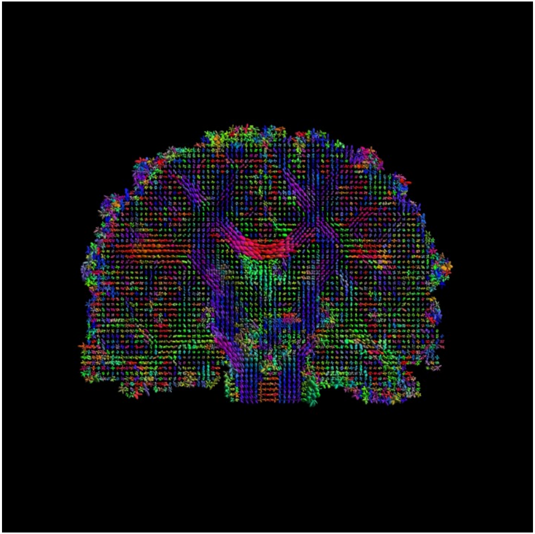
A Multigrid Solver for the Graph Laplacian on Power-Law Graphs

Eric Buras

Advisor: Dr. Matthew Knepley

Rice University
Department of Computational and Applied Mathematics

April 7, 2016

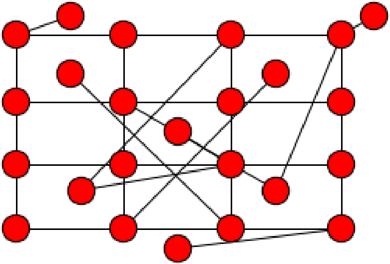


Human brain connectome; coronal view [5]

- Model this with a graph, G, with set of vertices and edges
- Set up graph Laplacian matrix and linear system
- Solve to provide some information about the data

Foundation

Graph: $G = (V, E)$ where V is the set of vertices (nodes) and E is the set of edges.

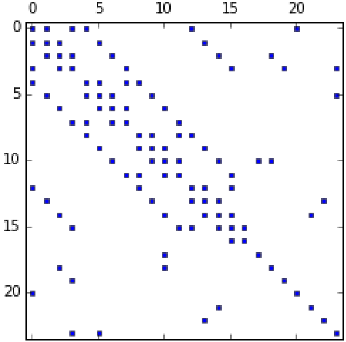


Foundation

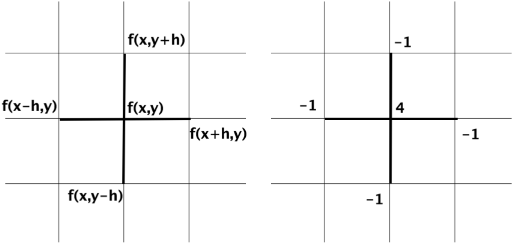
Laplacian matrix of G:

$$L = D - A$$

D: diagonal matrix of node degrees, A: adjacency matrix of G



Why is it called a Laplacian matrix?



Finite difference discretization of the Laplacian on a grid

Why Do We Care About the Laplacian?

- Laplacian operator inverse is weighted average of paths through a network:

$$b + Lb + L^2 b + L^3 b + \ldots = \sum_{i=0}^{\infty} L^i b = (I - L)^{-1} b$$

- PageRank linear system:

$$(I - \alpha L)x = (1 - \alpha)b$$

My Work

- Partition the graph into large locally-connected subgraph and small teleportation subgraph
- Solve the large locally-connected part with algebraic multigrid
- Solve the teleportation part directly
- Linear algebra to combine and solve the entire linear system

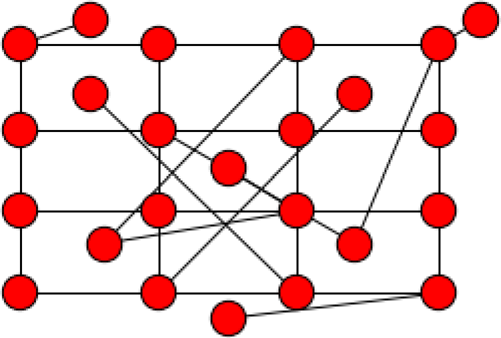
Graph Partitioning (k, l)

k: **maximum path length**
l: **minimum edge-disjoint paths**

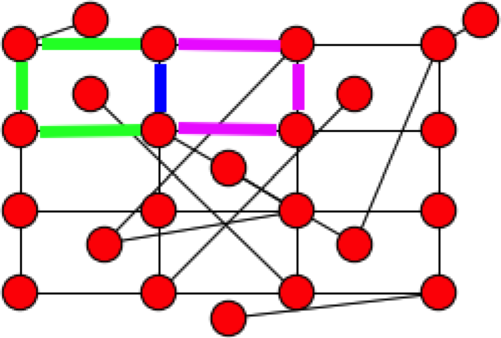
Graph **G**
P = copy(G)
While: $\textit{Deleted\_edges} == \textit{True}$:
    Deleted_edges = False
    For *edge* in P.edges:
        $(\textit{start\_node}, \textit{end\_node}) = \textit{edge}$
        Search through paths of length $i = 2 : k$ from *start_node* to *end_node*
        $\textit{count} =$ number of edge-disjoint paths
        if $\textit{count} \leq l$:
            P.remove(*edge*)
            $\textit{Deleted_edges} =$True

P **is the maximum** k, l**-locally connected subgraph of G**

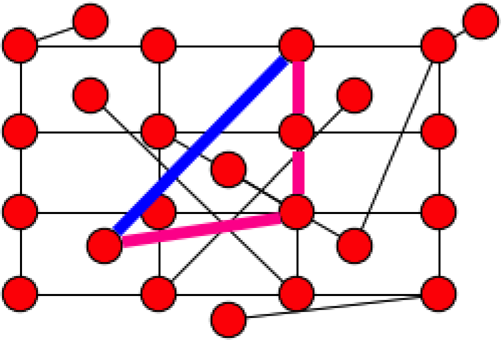
Graph Partitioning



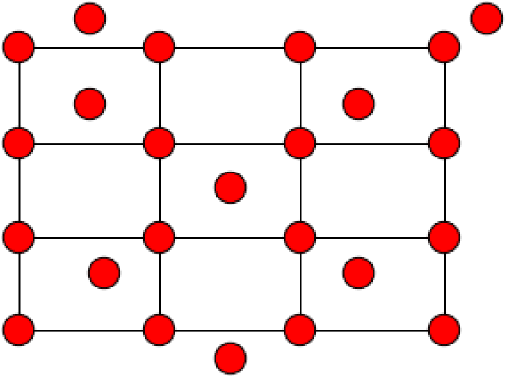
Graph Partitioning



Graph Partitioning

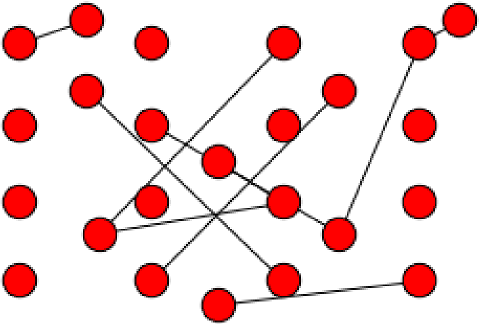


Graph Partitioning



Max locally-connected subgraph using Chung-Lu [1]

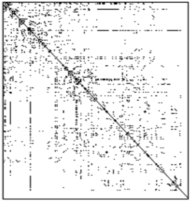
Graph Partitioning



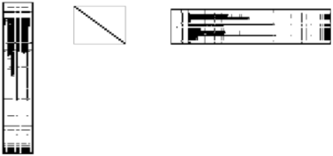
Remaining teleportation subgraph

Linear System Solve

$$L = P_L + T_L$$



$U \quad S \quad V^T$



Linear System Solve

Definition (Woodbury Matrix Identity)

$(P_L + USV^T)^{-1} = P_L^{-1} - P_L^{-1}U(S^{-1} + V^T P_L^{-1} U)^{-1} V^T P_L^{-1}$

$$
\begin{aligned}
Lx &= b \\
x &= L^{-1} b \\
x &= (P_L + T_L)^{-1} b \\
x &= (P_L + USV)^{-1} b \\
x &= (P_L^{-1} - P_L^{-1} U (S^{-1} + V P_L^{-1} U)^{-1} V P_L^{-1}) b \\
x &= P_L^{-1} b - P_L^{-1} U (S^{-1} + V P_L^{-1} U)^{-1} V P_L^{-1} b
\end{aligned}
$$

Current Solution Approaches

Multigrid on entire graph: LAMG, CMG, Cascadic

- How do you coarsen the grid? Based on heuristics
- Difficult to get good coarsening ratios

Brute force factorization

- Slow and memory inefficient

Partitioning Complexity

- Graph partitioning complexity relies on the degree distribution of the graph
- Power law degree graphs with degree sequence:

$$P(d) \sim d^{-\gamma}$$

$$E[d] = \frac{\zeta(\gamma - 1)}{\zeta(\gamma)}$$

- Partitioning is $O(iter. \times edges \times E[d]^3)$

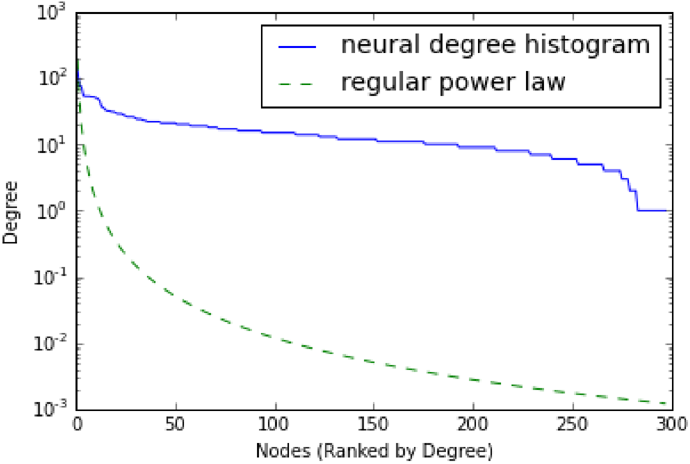
Solve Complexity

Operation	**O(FLOPs)**
$USV = T_L$	$O(n^3)$
S^{-1}	$O(r)$
$y = P_L^{-1} b$ (MG)	O(n)
$y_1 = Vy$	$O(rn)$
$Q = P_L^{-1} U$ (r MG solves)	$O(rn)$
$Q_1 = VQ$	$O(r^2 n)$
$Q_2 = S^{-1} + Q_1$	$O(r^2)$
$y_2 = Q_2^{-1} y_1$	$O(r^3)$
$y_3 = U y_2$	$O(rn)$
$y_4 = P_L^{-1} y_3$ (MG)	$O(n)$
$x = y - y_4$	$O(n)$

Solve Complexity

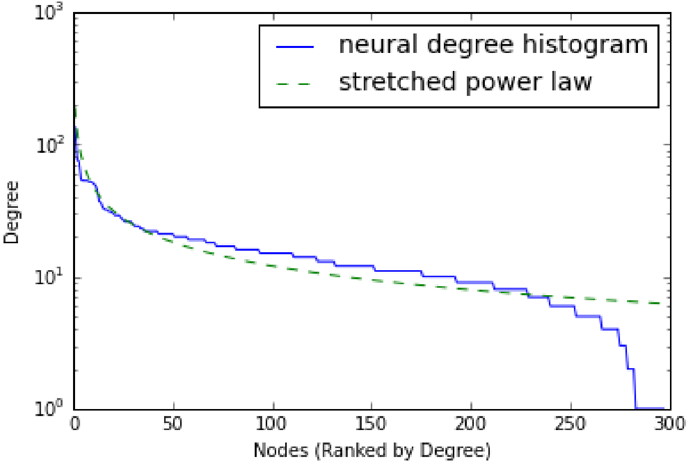
Operation	**O(FLOPs)**	**Optimized**
$USV = T_L$	$O(n^3)$	$O(r^2 n)$
S^{-1}	$O(r)$	
$y = P_L^{-1} b$ (MG)	O(n)	
$y_1 = Vy$	$O(rn)$	
$Q = P_L^{-1} U$ (r MG solves)	$O(rn)$	$\frac{1}{4}$ or $\frac{1}{8}$ MG constant
$Q_1 = VQ$	$O(r^2 n)$	
$Q_2 = S^{-1} + Q_1$	$O(r^2)$	
$y_2 = Q_2^{-1} y_1$	$O(r^3)$	
$y_3 = Uy_2$	$O(rn)$	
$y_4 = P_L^{-1} y_3$ (MG)	$O(n)$	
$x = y - y_4$	$O(n)$	

Power Law vs. Neural Network Degree Sequence



$\gamma = 2.1$

Stretched Power Law



stretched degree is 0.6

Graphs and Results

Graph (nodes, edges)	Avg. Deg.	Rank T_L	Part. (s)	Iter.
Neural (297, 2148) [6, 7]	14.46	22	1.6	3
Metabolic (453, 2025) [2]	8.94	49	2.3	3
Protein (912, 22738) [4]	49.86	26	53	2
Facebook (4039, 88234) [3]	43.69	180	480	3
Power (4941, 6594) [6]	2.67	4284	.33	3

Graphs and Results

Graph (vertices, edges)	Part. (s)	Nx Part. (s)
Neural (297, 2148)	1.6	11
Metabolic (453, 2025)	2.3	12
Protein (912, 22738)	53	1915
Facebook (4039, 88234)	480	11593
Power (4941, 6594)	.33	2.1

Solve Component Results

These are the limiting solve components according to theoretical complexity and timed results:

Operation	$O(r,n)$	**Neur.**	**Meta.**	**Prot.**	**FB**	**Pow.**
$USV = T_L$	$O(n^3)$	.0334	.0737	.4183	38.47	72.86
$Q = P_L^{-1}U$ ($r\times$MG)	$O(rn)$	.3292	.7360	.5190	23.11	106.9
$Q_1 = VQ$	$O(r^2n)$	.0006	.0036	.0013	.4124	285.1
$y_2 = Q_2^{-1}y_1$	$O(r^3)$	.0023	.0025	.0013	.0099	86.49
Total	$O(n^3)$	**.51**	**.966**	**1.44**	**64.46**	**560**

What Do These Solutions Mean?

C. Elegans data:

- Important regions in the worm nervous system
- Key metabolic processes
- Unobserved protein-phenotype relationships

Facebook friend networks:

- Origins of influence in a social group

Power Grid:

- Flow of electricity through urban infrastructure

Summary

Complete, working graph Laplacian solver

- Simple and easy to replicate
- Utilizes maximum locally-connected subgraph algorithm
- Provided baseline complexity model for graphs of certain class

Optimized partitioning

- 50x speedup from NetworkX library function for $k, l = 3$
- Make open source contribution

Future Work: Theory

- Identify class of graphs with low-rank T_L
- Establish provably good performance for multigrid on P_L

Future Work: Implementation

- Vectorize multiple rhs multigrid
- Utilize rank-revealing SVD
- Write parallelizable C code and run on larger machines
- Compare to current heuristic-based multigrid methods

Future Work: Application

- Find and replicate Laplacian solvers on data
- Test larger graphs; more complex biological systems

Thanks

Thank you:

- Dr. Knepley for advising me on this project
- Dr. Schaefer and Dr. Shrivastava for being on my committee
- Everyone for attending my defense!

References

Fan Chung and Linyuan Lu.
Complex Networks, chapter The Small World Phenomenon in Hybrid Power Law Graphs, pages 89–104.
Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

Jordi Duch and Alex Arenas.
Community detection in complex networks using extremal optimization.
Physical review E, 72(2):027104, 2005.

Julian J McAuley and Jure Leskovec.
Learning to discover social circles in ego networks.
In *NIPS*, volume 2012, pages 548–56, 2012.

Nicolas Simonis, Jean-François Rual, Anne-Ruxandra Carvunis, Murat Tasan, Irma Lemmens, Tomoko Hirozane-Kishikawa, Tong Hao, Julie M Sahalie, Kavitha Venkatesan, Fana Gebreab, et al.
Empirically controlled mapping of the caenorhabditis elegans protein-protein interactome network.
Nature methods, 6(1):47–54, 2009.

Arthur W Toga, Kristi A Clark, Paul M Thompson, David W Shattuck, and John Darrell Van Horn.
Mapping the human connectome.
Neurosurgery, 71(1):1, 2012.

Duncan J Watts and Steven H Strogatz.
Collective dynamics of 'small-world' networks.
nature, 393(6684):440–442, 1998.

JG White, E Southgate, JN Thomson, and S Brenner.
The structure of the nervous system of the nematode caenorhabditis elegans: the mind of a worm.
Phil. Trans. R. Soc. Lond, 314:1–340, 1986.