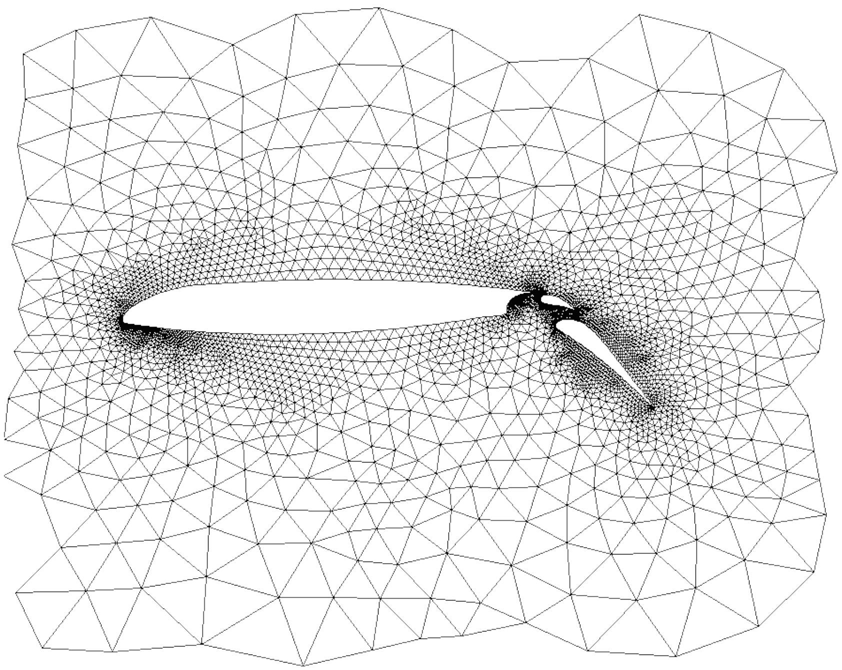
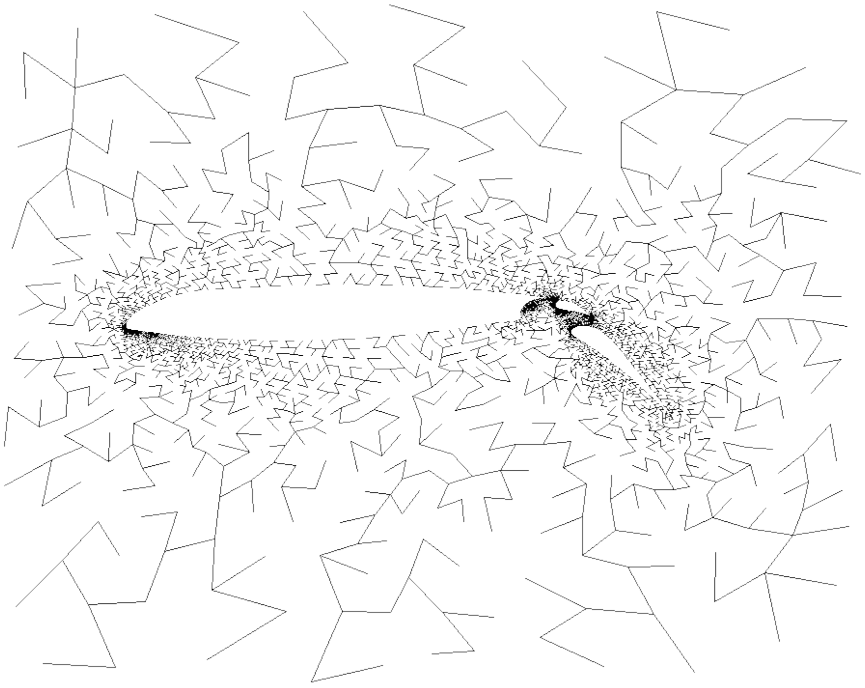
Fast Algorithms for Graph Partitioning, Sparsification, and the Solution of Linear Systems

Daniel A. Spielman

Yale





Joint work with Shang-Hua Teng (BU)

Three $m \log^{O(1)} n$-time algorithms

Solving symmetric,
diagonally-dominant linear systems

Sparsification:
approximating graphs by sparse subgraphs

Partitioning:
Approximately balanced, cutting few edges
By growing clusters, locally, from seed vertices

Weighted Graphs and Laplacian Matrices

$$\begin{pmatrix} 1 & -1 & 0 \\ -1 & 5 & -4 \\ 0 & -4 & 4 \end{pmatrix}$$

Laplacian matrix of weighted graph

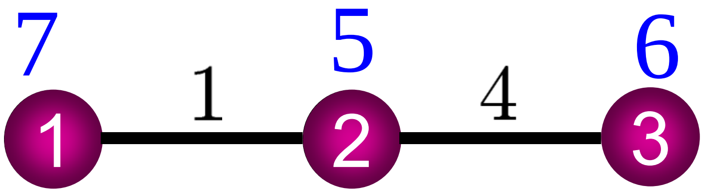
$L_{i,j}$ = negative of weight from i to j

diagonal = weighted degree

Corresponding Quadratic Form:

$$Q(x) = x^T L x = \sum_{(i,j) \in E} w_{i,j} (x_i - x_j)^2$$

Weighted Graphs and Laplacian Matrices



$$\begin{pmatrix} 1 & -1 & 0 \\ -1 & 5 & -4 \\ 0 & -4 & 4 \end{pmatrix}$$

Corresponding Quadratic Form:

$$Q(x) = x^T L x = \sum_{(i,j)\in E} w_{i,j}(x_i - x_j)^2$$

Example:

$$(x_1, x_2, x_3) = (7, 5, 6)$$

$$1 \cdot (7-5)^2 + 4 \cdot (5-6)^2 = 8$$

Graphic Inequalities and Approximating Graphs

$$Q(x) = x^T L x = \sum_{(i,j)\in E} w_{i,j}(x_i - x_j)^2$$

$G \succcurlyeq H$ if $Q_G(x) \geq Q_H(x), \forall x$

iff $L_G - L_H$ is positive semi-definite

For example, if H is a subgraph of G

H is quality κ approximation of G if

$$H \preccurlyeq G \preccurlyeq \kappa \cdot H$$

Iterative Methods

$$\|x - A^{-1}b\|_A < \epsilon$$

Preconditioned Conjugate Gradient

Find easy-to-solve B that approximates A

Solve in time

$$\sqrt{\kappa_f(A,B)}\,(\mathrm{nnz}(A) + \mathrm{solve}(B))\log(1/\epsilon)$$

Quality of approximation

Time to solve ***By = c***

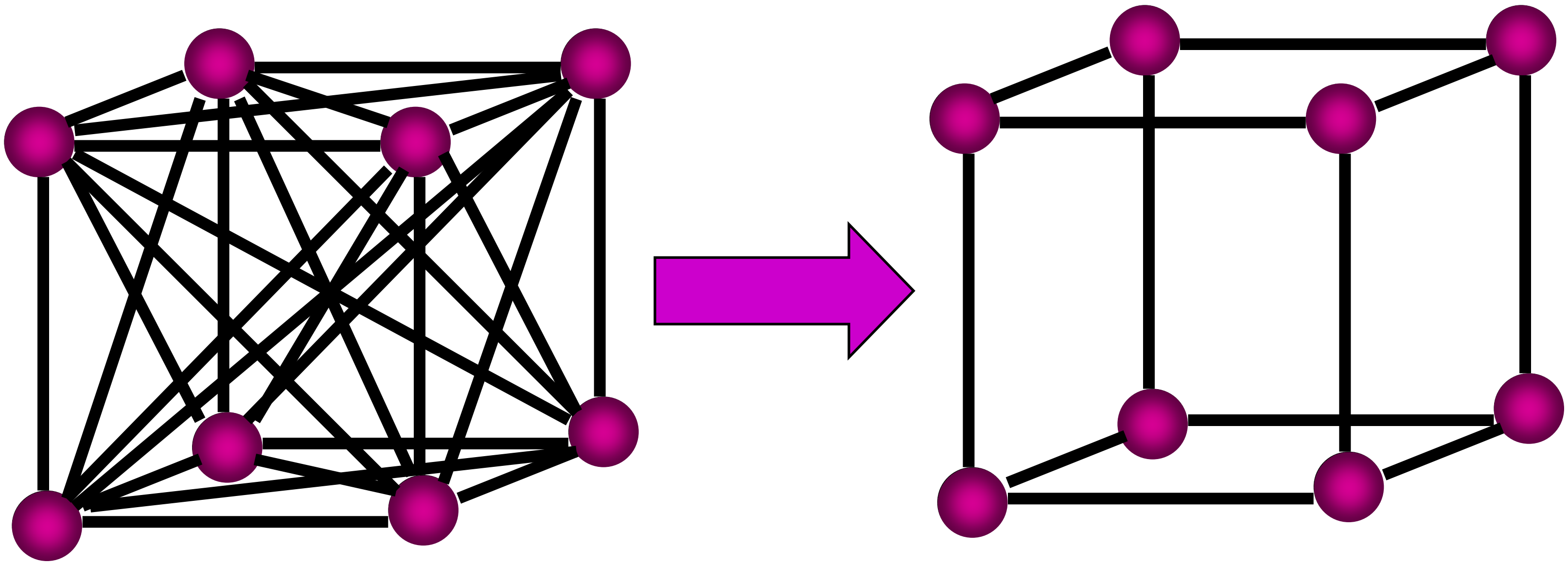
Accuracy

$$\kappa_f(A,B) = \min_{\kappa} \text{ such that } \exists c : c \cdot B \preccurlyeq A \preccurlyeq \kappa c \cdot B$$

Sparsification Theorems

Feder-Motwani ’91, Benczur-Karger ‘96

Every graph can be well-approximated by a sparse graph



Sparsification Theorems

Sparsifier: Given G, find weighted subgraph H s.t.

$$H \preccurlyeq G \preccurlyeq (1+\epsilon) \cdot H$$

#edges(H) < $\left(n/\epsilon^2\right) \log^{O(1)} n$

in time $m \log^{O(1)} n$

Ultra-Sparsifier:

$$H \preccurlyeq G \preccurlyeq k \cdot H$$

#edges(H) < $n + (n/k) \log^{O(1)} n$

in time $m \log^{O(1)} n$

tree + few edges, so can solve H quickly

Linear System Solvers

For symmetric, diagonally dominant A, any b

Find $\|x - A^{-1}b\|_A < \epsilon$ in time

$$m \log^{O(1)} n \log(1/\epsilon)$$

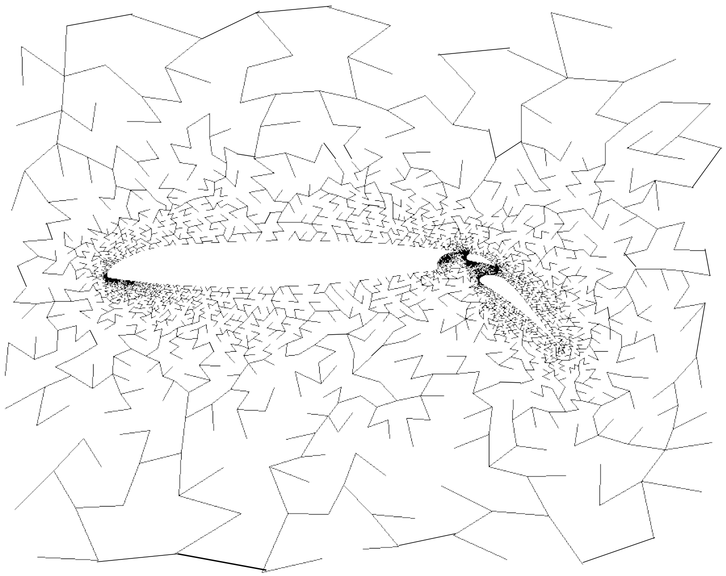
If planar:

$$O(n \log^2 n \log \log n \log(1/\epsilon))$$

no other assumptions

(m is number of non-zeros in A)

Linear System Solvers from subgraph preconditioners (Vaidya '90)



(Gremban, Miller '96)

(Joshi '97), (Reif '98)

(Bern, Boman, Chen, Gilbert, Hendrickson, Nguyen, Toledo '01)
(Boman, Hendrickson '01)
(S, Teng '03)

Most nodes have degree 1 or 2,
so can Cholesky factor to smaller system,
and solve recursively

Simplest Sparsification: Complete Graph

If A is Laplacian of K_n,
all non-zero eigenvalues are n

If B is Laplacian of Ramanujan expander
all non-zero eigenvalues satisfy

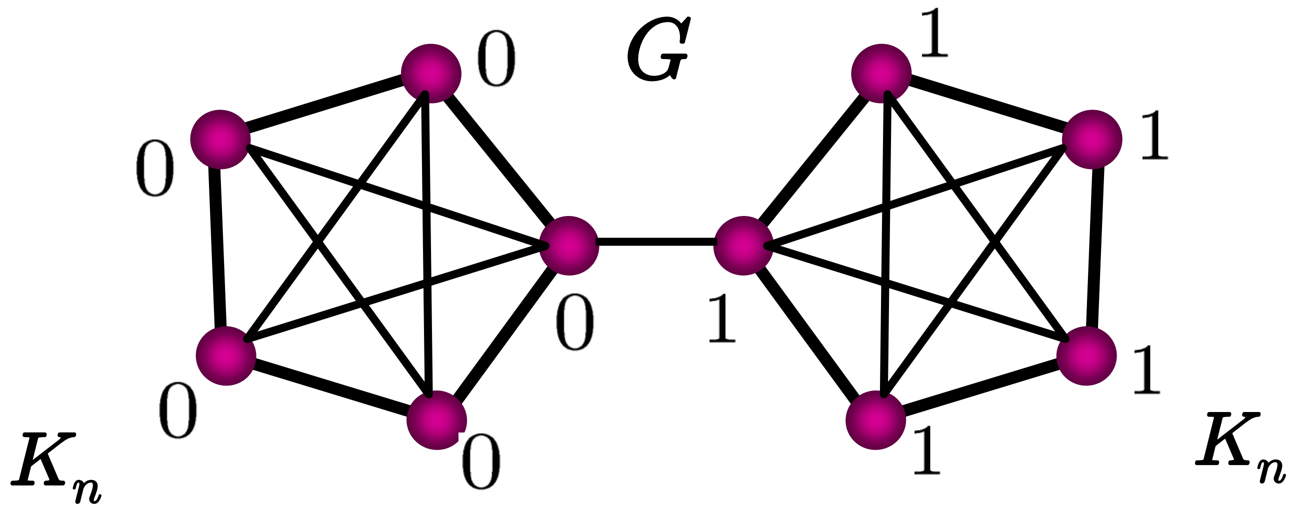
$$\lambda \in \left(d - 2\sqrt{d-1}, d + 2\sqrt{d-1} \right)$$

And so

$$\frac{n}{d + 2\sqrt{d-1}} \cdot B \preccurlyeq A \preccurlyeq \frac{n}{d - 2\sqrt{d-1}} \cdot B$$

Example: Random Sampling Fails

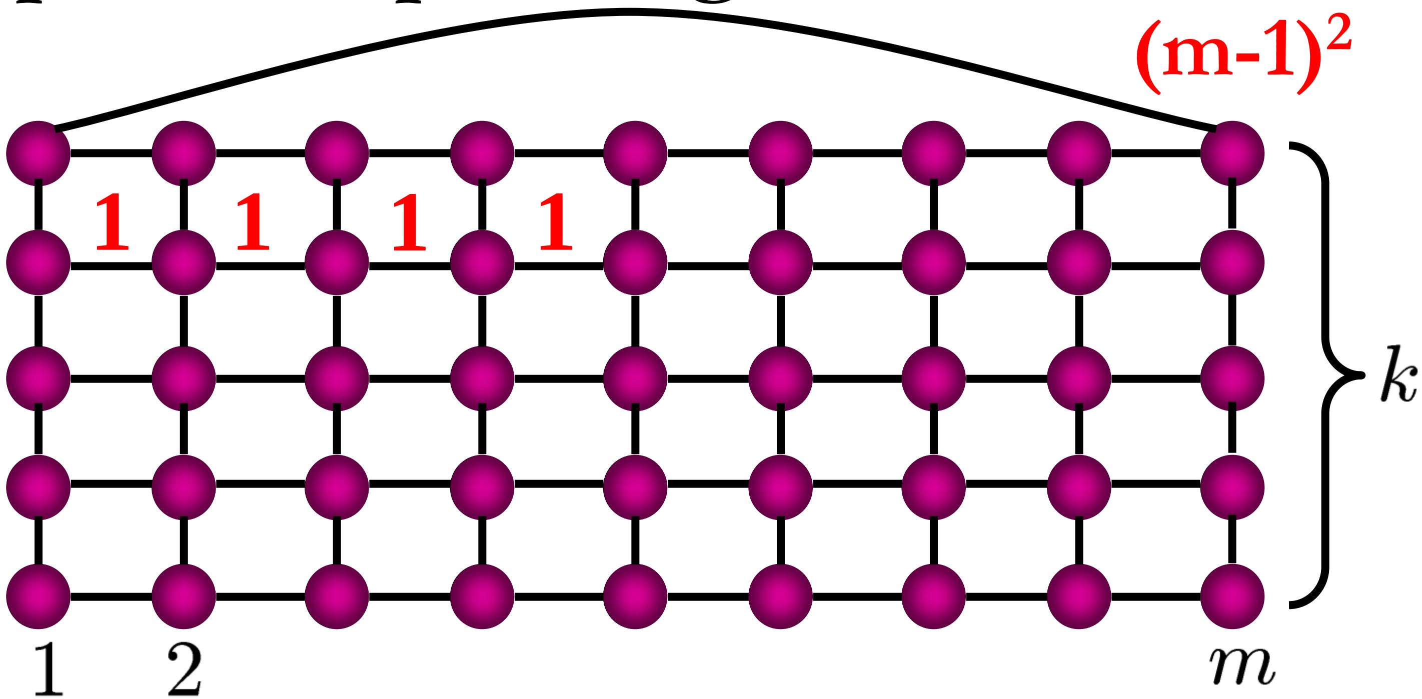
$$Q(x) = x^T L x = \sum_{(i,j) \in E} w_{i,j}(x_i - x_j)^2$$



If H does not contain middle edge,

$$\nexists c : G \preccurlyeq c \cdot H$$

Example: Grid plus edge

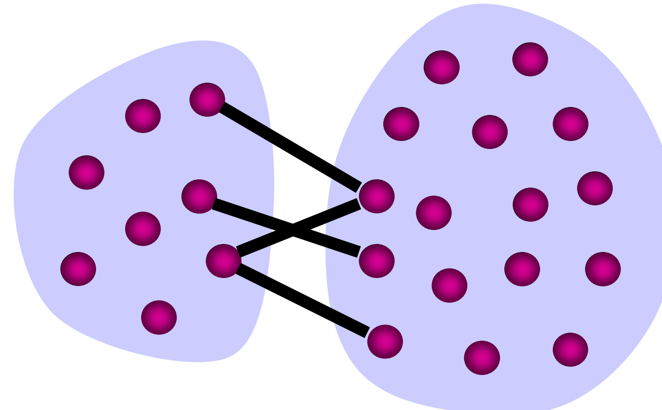


$$x^T A x = \sum_{(i,j) \in E} (x_i - x_j)^2 = (m-1)^2 + k(m-1)$$

Conductance

Cut = partition of vertices

S



Conductance of S

$$\Phi(S) \stackrel{\text{def}}{=} \frac{\#\text{ edges leaving } S}{\text{sum of degrees on smaller side}}$$

Conductance of G

$$\Phi_G \stackrel{\text{def}}{=} \min_S \Phi(S)$$

$$\Phi_G^2/2 \leq \lambda_2(D^{-1}L) \leq \Phi_G$$

Conductance and Sparsification

If conductance high (expander)
    can precondition by random sampling

If conductance low
    can partition graph by removing few edges

Decomposition:
    Partition of vertex set into big pieces
        remove few edges
        graph on each partition has high conductance

Graph Partitioning Algorithms

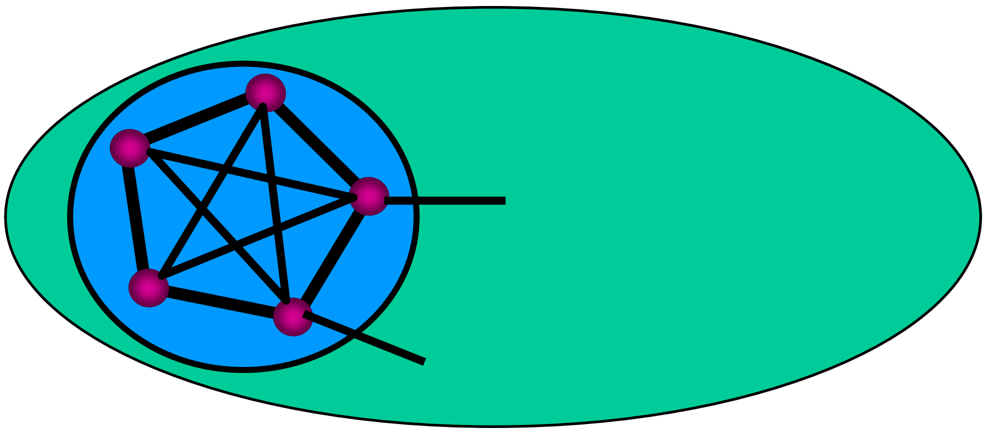
Linear Programming: too slow

Spectral: one cut quickly,
but can be unbalanced ⟹ many runs

Multilevel (Chaco/Metis): can't analyze,
miss small sparse cuts

Local Clustering

Cluster: Set of verts S $\Phi(S) \leq \phi$



when $\phi = \dfrac{1}{\log^{O(1)} n}$

Given random vertex inside cluster S,
find a cluster T of size at most 2|S|
mostly inside S
time $|T| \log^{O(1)} n$

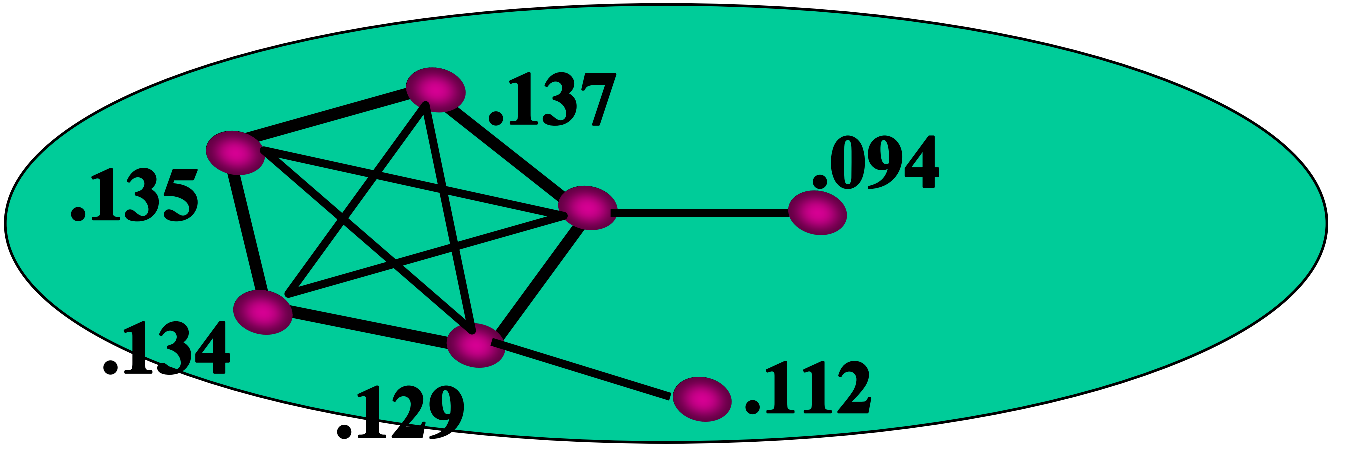
$$\Phi(T) \leq \phi^{1/3}$$

Local Clustering by Truncated Diffusion

Compute probability distribution of
random walk, rounding small values to zero

Lovasz-Simonovits Theorem (modified)

If slow convergence, then low conductance
And, can find the cut from highest probability nodes



Future Work

Practical Local Clustering

Other applications of sparsification

Practicality of solvers
(Khandekar-Rao-Vazirani?)

Computing eigenvectors

Solvers for other families of linear systems

To learn more

"Nearly-Linear Time Algorithms for Graph Paritioning, Sparsification, and Solving Linear Systems" (STOC '04, Arxiv)

Will be split into two papers:
numerical and combinatorial
available mid-July

My lecture notes for
Spectral Graph Theory and its Applications