ECS231: Spectral Partitioning

Based on Berkeley's CS267 lecture on graph partition
Definition of graph partitioning

• Given a graph $G = (N, E, W_N, W_E)$
  - $N =$ nodes (or vertices),
  - $E =$ edges
  - $W_N =$ node weights
  - $W_E =$ edge weights

• Ex: $N =$ {tasks}, $W_N =$ {task costs},
  $E =$ {edge (j,k): task j sends $W_E(j,k)$ words to task k}

• Graph partitioning:
  
  choose a partition $N = N_1 \cup N_2 \cup \ldots \cup N_P$ such that
  - The sum of the node weights in each $N_j$ is “about the same”
  - The sum of all edge weights of edges connecting all different pairs $N_j$ and $N_k$ is minimized

• Ex: balance the work load, while minimizing communication

• Special case of $N = N_1 \cup N_2$: graph bisection
Applications

- Telephone network design
  - Original application, algorithm due to Kernighan
- VLSI layout
  - N = \{units on chip\}, E = \{wires\}, W_{E}(j,k) = wire length
- Data mining and clustering
- Physical mapping of DNA
- ...
Load balancing while minimizing communication in HPC:

- **Sparse matrix-vector multiplication**
  - $N = \{1, 2, \ldots, n\}$,
  - $(j, k)$ in $E$ if $A(j, k)$ nonzero,
  - $W_E(j, k) = 1$
  - $W_N(j) = \#$ nonzeros in row $j$

- **Sparse Gaussian elimination**
  - Used to reorder rows and columns to increase parallelism, decrease “fill-ins”
Sparse matrix partition:

Partitioning a Sparse Symmetric Matrix

\[
A = \begin{bmatrix}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\end{bmatrix}
\]
Cost of graph partitioning

- Many possible partitionings to search:

- \(\binom{n}{n/2} \approx \sqrt{\frac{2n}{\pi}} \times 2^n\) bisection possibilities
- Choosing optimal partitioning is NP-complete
  - Only known exact algorithms have cost = exponential(n)
- Need good heuristics
First heuristic: repeated graph bisection

- To partition $N$ into $2^k$ parts
  - bisect graph recursively $k$ times
- Henceforth discuss mostly graph bisection
Coordinate-free partitioning

• Popular techniques for partitioning
  • Breadth-First Search – simple, but not great partition
  • Kernighan-Lin – good corrector given reasonable partition
  • Spectral Method – good partitions, but may be slow

• Multilevel methods
  • Used to speed up problems that are too large/slow
  • Coarsen, partition, expand, improve
  • Can be used with K-L and Spectral methods and others

• Speed/quality
  • For load balancing of grids, multi-level K-L probably best
  • For other partitioning problems (vision, clustering, etc.), spectral may be better
  • Good software available: meshpart, metis, chaco, …
Coordinate-free: spectral bisection

• Definitions
• Basic spectral bisection algorithm
• Fiedler’s theorem
• Implementation via the Lanczos algorithm

Reference: Fiedler (1970s): basic theory,
Pothen, Simon and Liou (1990): one of first "modern” treatments
Basic definitions

- **Definition:** The incidence matrix $\text{In}(G)$ of a graph $G(N,E)$ is an $|N|$ by $|E|$ matrix, with one row for each node and one column for each edge. If edge $e=(i,j)$ then column $e$ of $\text{In}(G)$ is zero except for the $i$-th and $j$-th entries, which are +1 and -1, respectively.

- Slightly ambiguous definition because multiplying column $e$ of $\text{In}(G)$ by -1 still satisfies the definition, but this won’t matter...

- **Definition:** The Laplacian matrix $\text{L}(G)$ of a graph $G(N,E)$ is an $|N|$ by $|N|$ symmetric matrix, with one row and column for each node. It is defined by
  - $\text{L}(G) (i,i) = \text{degree of node } i \text{ (number of incident edges)}$
  - $\text{L}(G) (i,j) = -1 \text{ if } i \neq j \text{ and there is an edge } (i,j)$
  - $\text{L}(G) (i,j) = 0 \text{ otherwise}$
Example of $\text{In}(G)$ and $\text{L}(G)$

**Incidence and Laplacian Matrices**

**Graph $G$**

**Incidence Matrix $\text{In}(G)$**

$$
\begin{bmatrix}
1 & 2 & 3 & 4 \\
1 & -1 & & \\
2 & 1 & -1 & \\
3 & & 1 & -1 \\
4 & & & 1 & -1 \\
5 & & & & 1 \\
\end{bmatrix}
$$

**Laplacian Matrix $\text{L}(G)$**

$$
\begin{bmatrix}
1 & 1 & -1 & & & & & & & & & \\
1 & -1 & 2 & -1 & & & & & & & & \\
1 & -1 & 2 & -1 & & & & & & & & \\
1 & & & 1 & & & & & & & & \\
1 & & & & 1 & & & & & & & \\
1 & & & & & 1 & & & & & & \\
\end{bmatrix}
$$

**Nodes numbered in black**

**Edges numbered in blue**
Properties of incidence and Laplacian matrices

1. \( \text{In}(G) \times (\text{In}(G))^T = L(G) \). This is independent of the signs chosen for each column of \( \text{In}(G) \).

2. \( L(G) \) is symmetric. (This means the eigenvalues of \( L(G) \) are real and its eigenvectors are real and orthogonal.)

3. Let \( e = [1, \ldots, 1]^T \), i.e. the column vector of all ones. Then \( L(G) \times e = 0 \).

4. Suppose \( L(G) \times v = \lambda \times v \), so that \( v \) is an eigenvector and \( \lambda \) an eigenvalue of \( L(G) \). Then

\[
\lambda = \frac{\| \text{In}(G)^T \times v \|^2}{\| v \|^2} = \sum \{ (v(i)-v(j))^2 \text{ for all edges } e=(i,j) \} / \sum v(i)^2
\]

5. The eigenvalues of \( L(G) \) are nonnegative:

\[
0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n
\]

6. The number of connected components of \( G \) is equal to the number of \( \lambda_i \) equal to 0. In particular, \( \lambda_2 \neq 0 \) if and only if \( G \) is connected.

Definition: \( \lambda_2(L(G)) \) is the algebraic connectivity of \( G \)
Basic spectral bisection algorithm

• Compute eigenvector $v_2$ corresponding to $\lambda_2(L(G))$
• For each node $j$ of $G$
  • if $v_2(j) < 0$
    put node $j$ in partition $N-$
  • else
    put node $j$ in partition $N+$
Spectral bisection algorithm (cont’d)

Why does this make sense?

Fiedler’s theorem:

1. Let $G$ be connected, and $N^{-}$ and $N^{+}$ defined as above. Then $N^{-}$ is connected. If no $v_2(j) = 0$, then $N^{+}$ is also connected.

2. Let $G_1(N,E_1)$ be a subgraph of $G(N,E)$, so that $G_1$ is “less connected” than $G$. Then $\lambda_2(L(G_1)) \leq \lambda_2(L(G))$, i.e. the algebraic connectivity of $G_1$ is less than or equal to the algebraic connectivity of $G$. 
Computing $v_2$ and $\lambda_2$ of $L(G)$ using Lanczos

- Given any $n$-by-$n$ symmetric matrix $A$ (such as $L(G)$). Lanczos computes a $k$-by-$k$ “approximation” $T$ by doing $k$ matrix-vector products, $k << n$

Choose an arbitrary starting vector $r$

$b(0) = ||r||$

$j = 0$

repeat

$j = j + 1$

$q(j) = r / b(j-1)$ … scale a vector

$r = A \cdot q(j)$ … matrix vector multiplication, the most expensive step

$r = r - b(j-1) \cdot v(j-1)$ … “saxpy”, or scalar*vector + vector

$a(j) = v(j)^T \cdot r$ … dot product

$r = r - a(j) \cdot v(j)$ … “saxpy”

$b(j) = ||r||$ … compute vector norm

until convergence … details omitted

$T = \begin{bmatrix} a(1) & b(1) \\ b(1) & a(2) & b(2) \\ b(2) & a(3) & b(3) \\ \vdots & \vdots & \vdots \\ b(k-2) & a(k-1) & b(k-1) \\ b(k-1) & a(k) \end{bmatrix}$

- Approximate $A$’s eigenvalues/vectors using $T$’s
Summary

• Laplacian matrix represents graph connectivity
• Second eigenvector gives a graph bisection
  • Roughly equal “weights” in two parts
  • Weak connection in the graph will be separator
• Implementation via the Lanczos algorithm
First references


Meshpart toolbox

- Meshpart: Matlab mesh partitioning and graph separator toolbox by J. Gilbert and S. Teng
- http://www.cerfacs.fr/algor/Softs/MESHPART

Demo:
- Load Graphpartdata.mat
- Gplot(Tapir,Txy)
- [part1,part2]=specpart(Tapir,Txy);
- Main files: specpart.m, fiedler.m