

High-Performance Computing 2025

Graph Partitioning

On to “Graph Partitioning”

Combinatorial matrix theory is a rich branch of mathematics that combines combinatorics, **graph theory**, and linear algebra.

Motivation

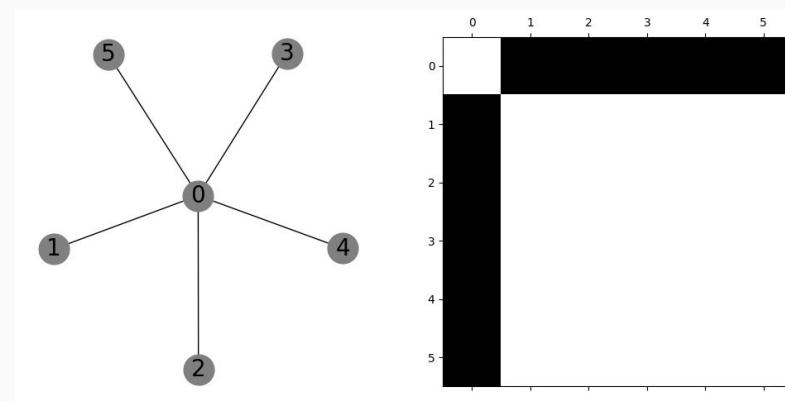
What is a graph?

Simple undirected graph is defined as $G=(\mathcal{V}, \mathcal{E})$

- The vertices \mathcal{V} are $\{1, 2, \dots, n\}$.
- The edge $\{i, j\} \in \mathcal{E}$ connect an unordered pair of vertices.
- Adjacency Matrix $A_{ij} \neq 0 \Leftrightarrow \{i, j\} \in \mathcal{E}$

What if the connections between nodes differ in magnitude? (weighted)

What if $i \rightarrow j$ but not the other way?
(directed)

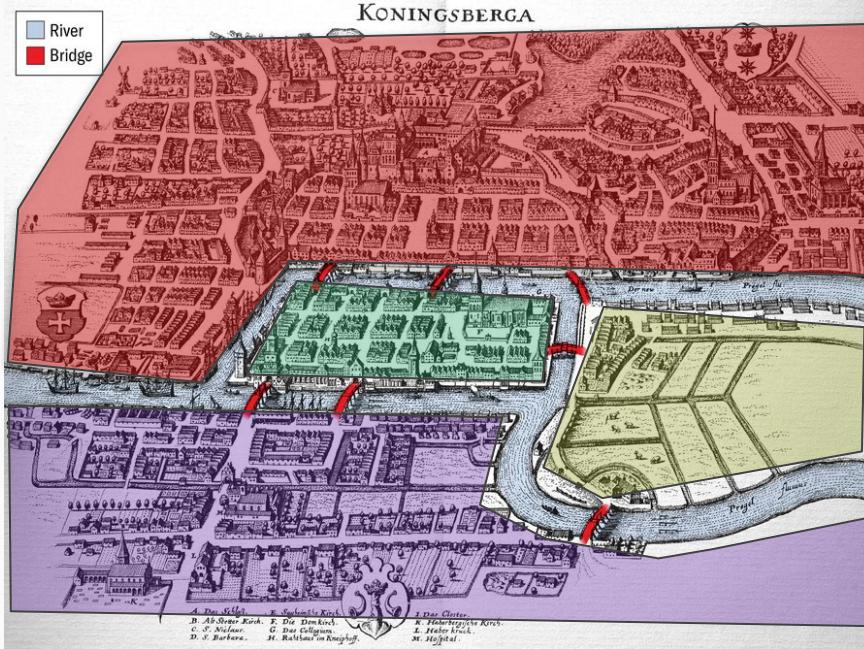


Graphical Representation

Adjacency Matrix **A**

Motivation

What is graph theory about?



Königsberg Bridge Problem

"... walk through the city cross each bridge once and only once"

Leonhard Euler: *Solutio problematis ad geometriam situs pertinentis* [[Link](#)]

This work is considered the beginning of modern "Graph Theory".

Motivation

What is graph partitioning?

“Bisection” to form two partitions ...
Repeated recursively.

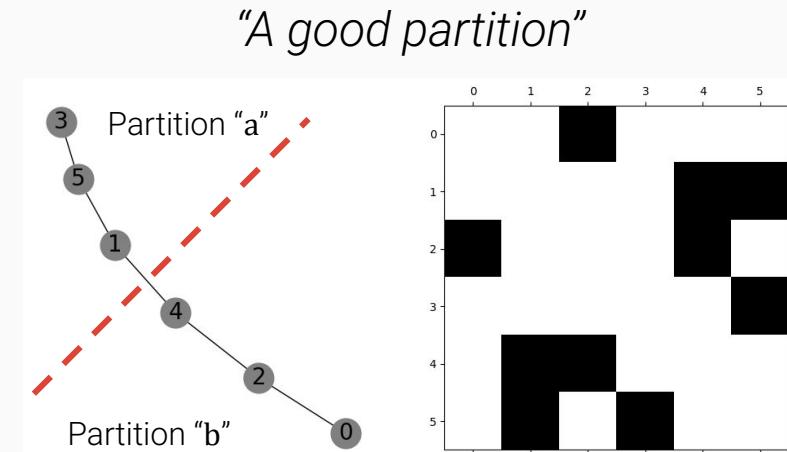
Desired attributes of bisection

- 1) Make two partitions of (roughly) equal size:

$$\mathcal{V} = \mathcal{V}^a \cup \mathcal{V}^b \text{ s.t. } |\mathcal{V}^a| = |\mathcal{V}^b|$$

- 2) With the minimum edge-cut:

$$\min \{ \{i,j\} \in \mathcal{E} : i \in \mathcal{V}^a \text{ and } j \in \mathcal{V}^b \}$$



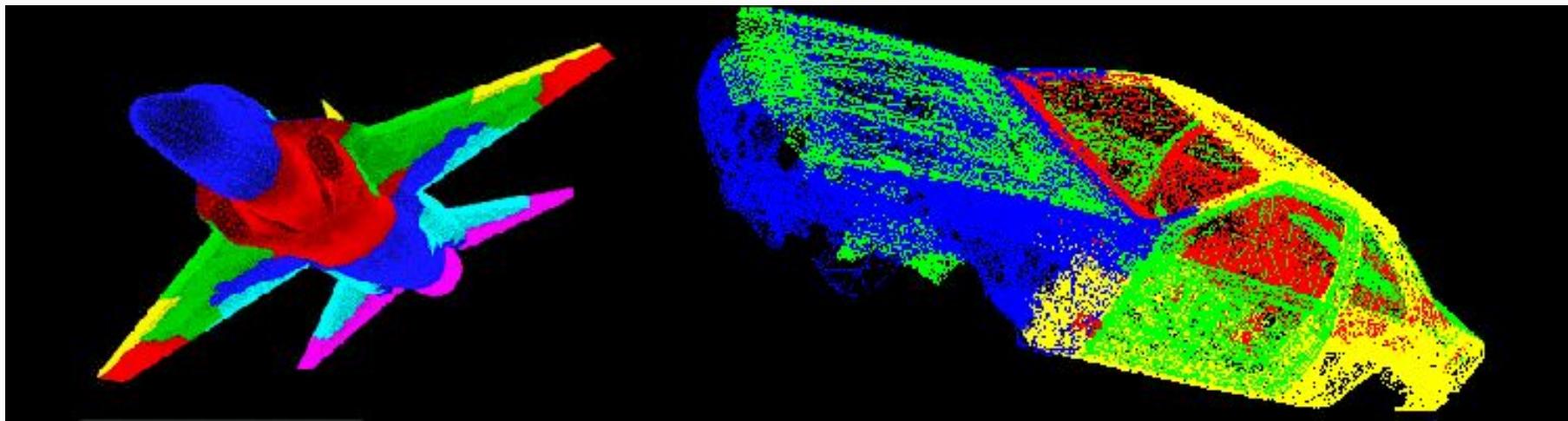
Think of a bad or the worst partitioning?

Motivation

Why is it used?

Parallel computing and load balancing

Perform parallel computation while minimizing communication.

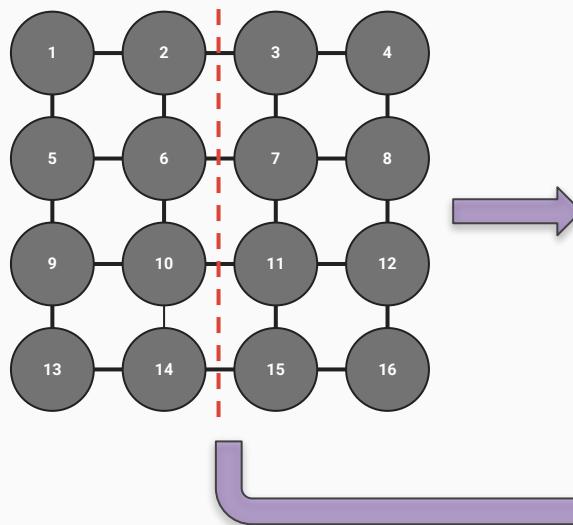


You already did a basic version of this in the MPI PDE mesh discretization.

Motivation

You already did this !

Discretized Problem



Finite Difference Operator

4	-1	0	0	-1	0	0	0	0	0	0	0	0	0
-1	4	-1	0	0	-1	0	0	0	0	0	0	0	0
0	-1	4	-1	0	0	-1	0	0	0	0	0	0	0
0	0	-1	4	0	0	0	-1	0	0	0	0	0	0
-1	0	0	0	4	-1	0	0	-1	0	0	0	0	0
0	-1	0	0	-1	4	-1	0	0	-1	0	0	0	0
0	0	-1	0	0	-1	4	-1	0	0	-1	0	0	0
0	0	-1	0	0	-1	4	0	0	0	-1	0	0	0
0	0	0	0	-1	0	0	0	4	-1	0	0	-1	0
0	0	0	0	0	-1	0	0	-1	4	-1	0	0	-1
0	0	0	0	0	0	-1	0	0	-1	4	-1	0	0
0	0	0	0	0	0	-1	0	0	-1	4	0	0	-1
0	0	0	0	0	0	0	-1	0	0	0	4	-1	0
0	0	0	0	0	0	0	0	-1	0	0	-1	4	-1
0	0	0	0	0	0	0	0	0	-1	0	0	-1	4
0	0	0	0	0	0	0	0	0	0	-1	0	0	-1

The Adjacency Matrix **A**? How did we do the bisection/partitioning?

Classification of Graphs

Coordinate-based vs Non-coordinate-based Graphs

Coordinate Graphs:

- Vertices are in physical space with coordinates (2D or 3D).
- Example, PDE meshes and particle simulations, where spatial positioning is crucial.
- Partitioning often emphasizes spatial node distribution rather than edge-cut.

Non-Coordinate Graphs:

- Vertices are not in physical space.
- Example, abstract and relationship-based, common in social networks.
- Partitioning prioritizes minimizing edge-cut to reduce communication overhead, crucial due to complex connectivity patterns without spatial constraints.

We will look at key algorithms for different types of graphs.

Coordinate Based

Coordinate Bisection

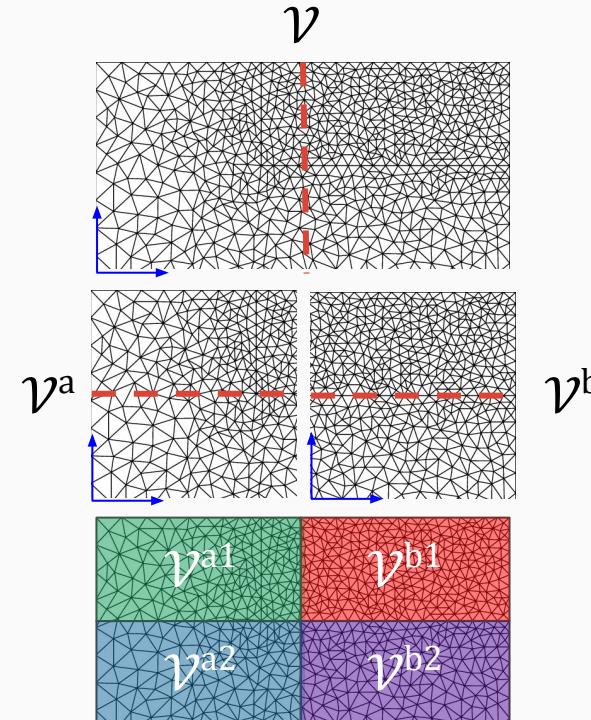
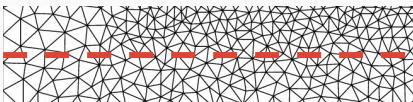
Partition along the **standard basis**.

1. **Cut** with even division of points X (median).
We can alternate **basis** between the different axes to keep recursive partitions.

Where is edge-cut calculated?

What is this procedure in 1D?

What can we say about this cut?



Bisection

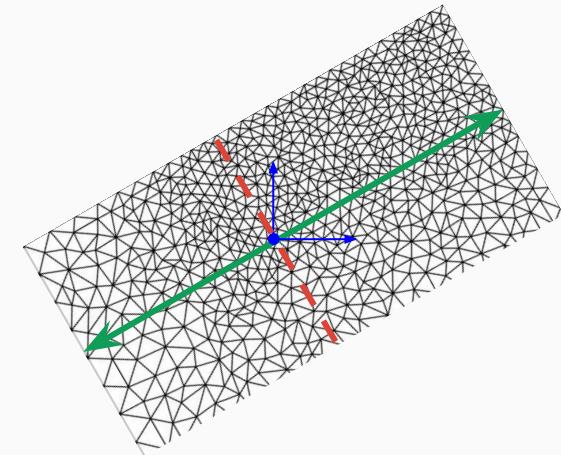
Recursive

Coordinate Based

Inertial Partitioning—Minimal Rotational Inertia Line

Partition along line orthogonal to **some other line**.

1. **Center** points \mathbf{X} .
2. Compute the **axis** corresponding to the direction of maximum variance of \mathbf{X} ; i.e., the axis that minimizes the rotational moment of inertia of the points \mathbf{X} ; a.k.a:
 - a. Principal component of $\text{cov}(\mathbf{X})$.
 - b. Eigvec of $\text{cov}(\mathbf{X})$ associated with the largest eigval.
3. Projecting \mathbf{X} on the line (new **axis**).
4. The **cut** is then taken at median of the line, partition is orthogonal to the line (new **axis**).



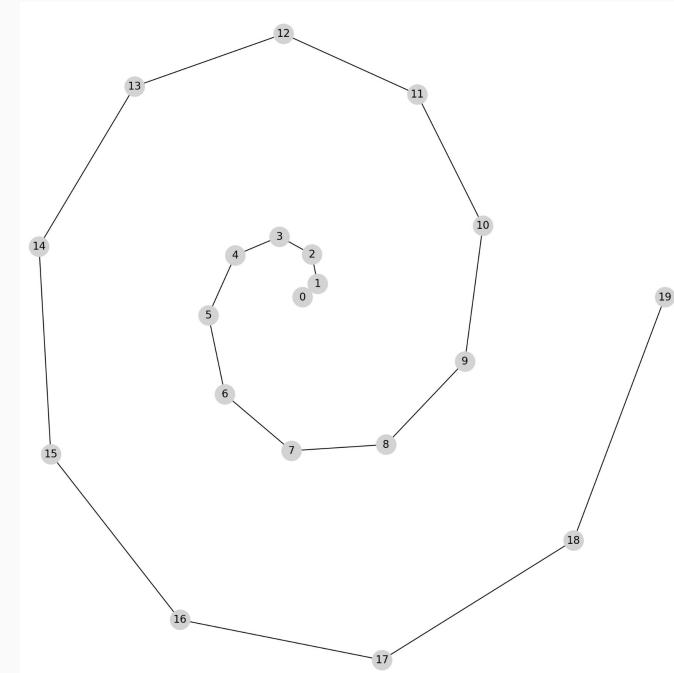
Project \mathbf{X} on the eigenvector corresponding to the largest eigenvalue of $\mathbf{X}\mathbf{X}^T$, then perform a 1D “Coordinate Bisection.”

Where is edge-cut calculated? What is the complexity of this solution method?

Coordinate Based Limitations

What does Inertial Bisection
do here ?

Nothing good...



Non-Coordinate Graphs

Greedy-algorithm for “Refinement”

Kernighan-Lin (70's) cost $\mathcal{O}(|\mathcal{V}|^3)^*$ & **Fiduccia-Mattheyses** (80's) cost $\mathcal{O}(|\mathcal{E}|)$

Refine the edge-cut of the partitions \mathcal{V}^a and \mathcal{V}^b

- Both assign nodes to partitions based on local rules.
- The KL algorithm operates on a pairwise exchange basis, while the FM algorithm simplifies this approach by not relying on pairwise exchanges.

See details [here](#).

* Naive implementation.

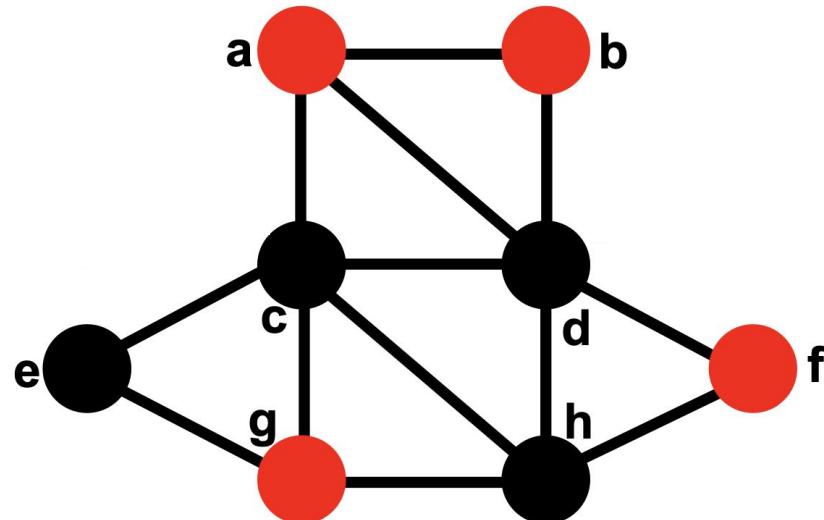
Non-Coordinate Graphs

Fiduccia-Mattheyses—A walk through

We ask “what if we swapped a vertex”...

Fiduccia-Mattheyses Algorithm

1. Sweep vertices:
 - a. For $v_1 \in \mathcal{V}^a$ select the largest decrease in edge-cut “if” $v_1 \in \mathcal{V}^b$.
 - b. Compute edge-cut.
 - c. Get $v_2 \in \mathcal{V}^b$ select the largest decrease in edge-cut “if” $v_2 \in \mathcal{V}^a$.
 - d. Compute edge-cut.
2. After sweep of all vertices, select the configuration with the lower edge-cut.

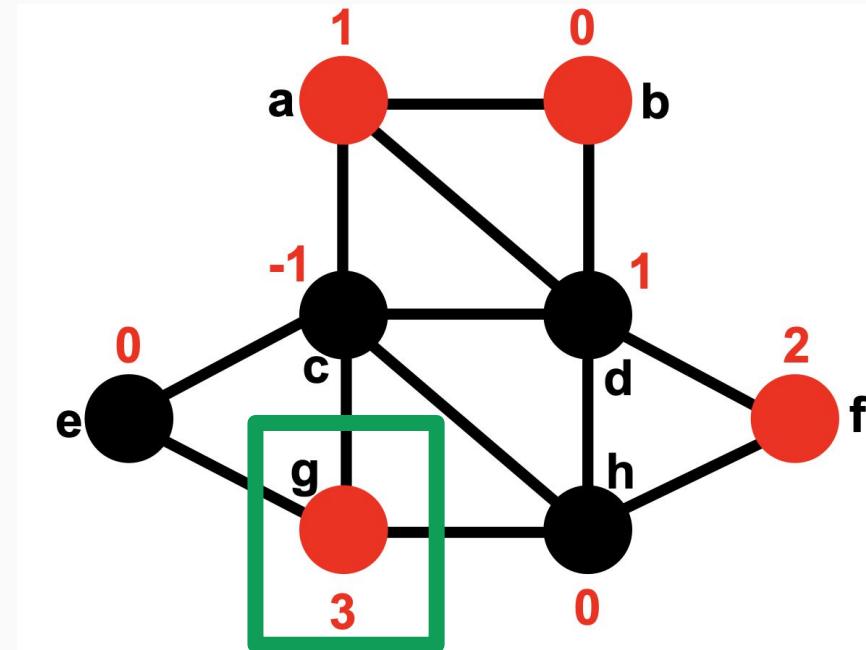


Non-Coordinate Graphs

Fiduccia-Mattheyses—A walk through

Partition \mathcal{V}^a

- The edge-cut of the graph is 8.
- In red, we show the values of swapping partitions or the “diff-value” of a vertex.
- Consider vertex “a”: the direct neighbors of “a” form a graph with an edge-cut of 2. If “a” changes partitions to black, the edge-cut becomes 1. Thus, the diff-value of “a” is 1.
- The node “g” yields the largest decrease.

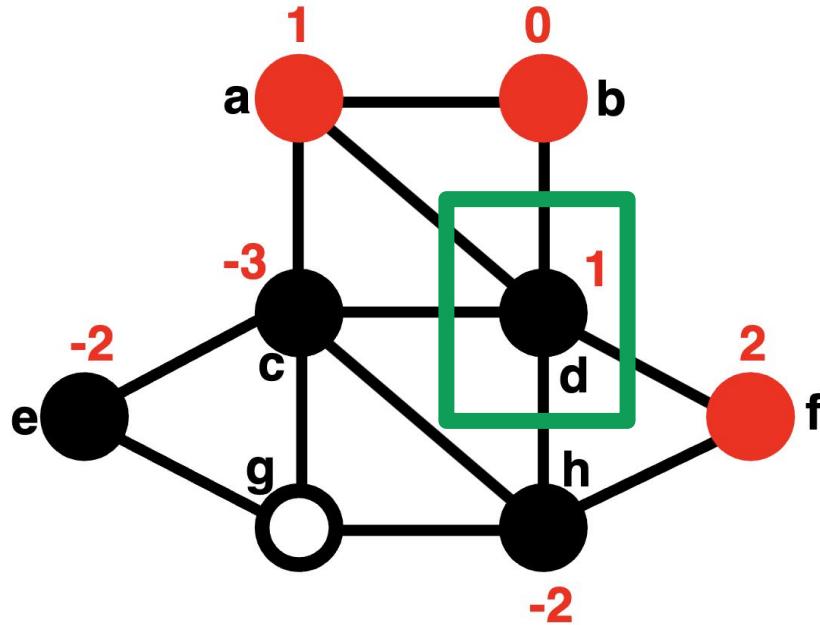


Non-Coordinate Graphs

Fiduccia-Mattheyses—A walk through

Partition \mathcal{V}^b

- We look to the other partition.
- The edge-cut of the graph is 5.
- The biggest decrease is “d”.

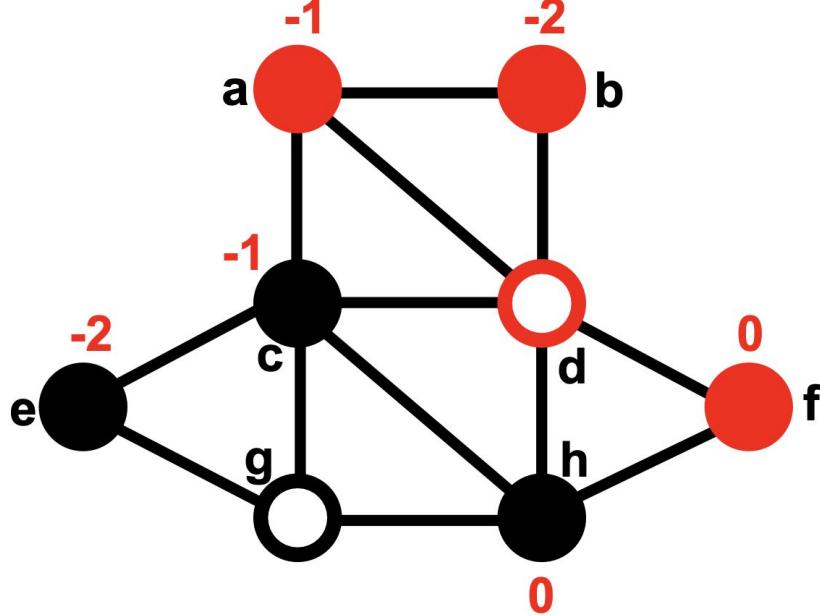


Non-Coordinate Graphs

Fiduccia-Mattheyses—A walk through

Repeat ...

- The edge-cut is now 4.
- Repeat until we have selected all i and j .
- The next value selected would be “f”.



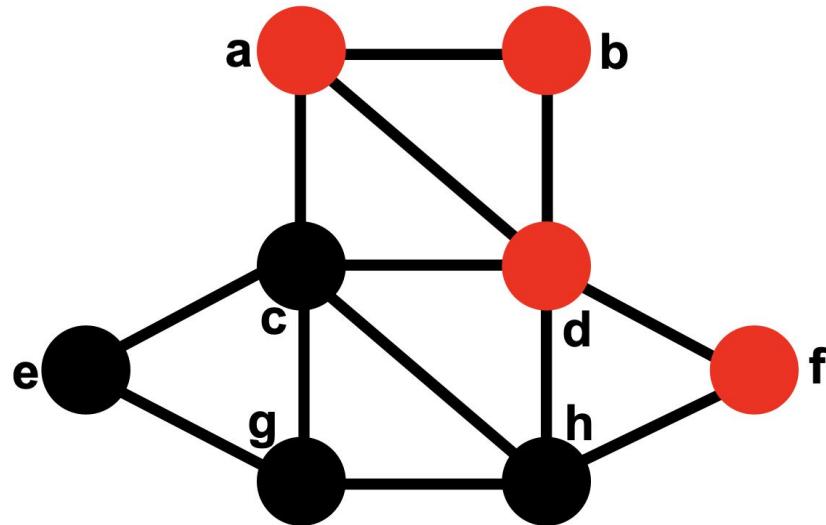
Non-Coordinate Graphs

Fiduccia-Mattheyses—A walk through

Select configuration with the lower edge-cut

- The history of edge cuts are $\mathbf{C}=\{8,4,5,9\}$.
- Select the configuration $k=\arg \min(\mathbf{C})=2$.
- Selected configuration has edge-cut of 4.

Note the FM operates on local heuristics of the graph, i.e., local selection rules.



Non-Coordinate Graphs

Spectral Partitioning—An Eigenvalue Problem

Fielder (70s) using eigenvalues, cost of $\mathcal{O}(|v|^3)$

Transform the discrete partitioning problem into a relaxed, continuous optimization problem we can have global partitioning method.

See details and derivations [here](#).

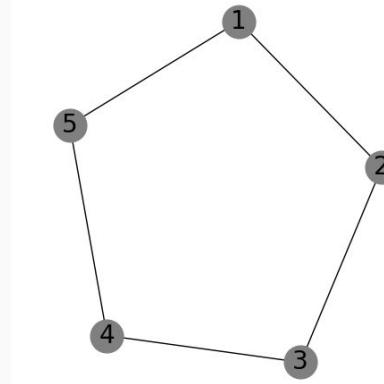
Non-Coordinate Graphs

Spectral Partitioning—An Eigenvalue Problem

The Laplacian Matrix $\mathbf{L} := \ell(\mathcal{G}) \in \mathbb{R}^{n \times n}$

Simple graph: the *Laplacian* has degrees on the diagonal (**D**) and minus the adjacency (-**A**) on the off-diagonals.

- Defined as $\mathbf{L} := \mathbf{D} - \mathbf{A}$
- Positive semi-definite $\mathbf{L} \geq 0$
- Row sum is 0, $\text{Null}(\mathbf{L}) = \mathbf{1}$, i.e. $\mathbf{L}\mathbf{1} = \mathbf{0}$
- Symmetric $\mathbf{L} = \mathbf{L}^T$



Adjacency Matrix

```
[[0 1 0 0 1]
 [1 0 1 0 0]
 [0 1 0 1 0]
 [0 0 1 0 1]
 [1 0 0 1 0]]
```

Laplacian Matrix

```
[[ 2 -1  0  0 -1]
 [-1  2 -1  0  0]
 [ 0 -1  2 -1  0]
 [ 0  0 -1  2 -1]
 [-1  0  0 -1  2]]
```

The construction of this matrix is very important ...

Non-Coordinate Graphs

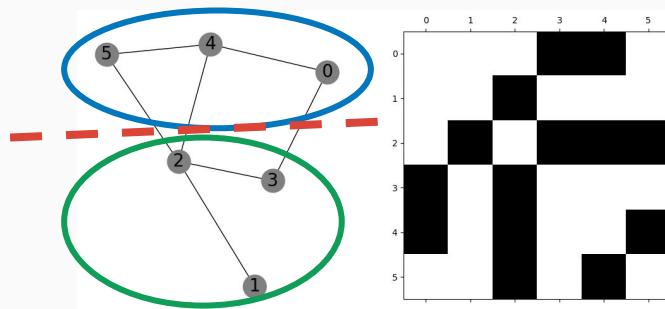
Spectral Partitioning—An Eigenvalue Problem

Consider $\mathbf{x} \in \{1, -1\}^n$ as the partition “label”, $\mathbf{x}^T \mathbf{L} \mathbf{x} = “4 \text{ times the edge-cut}”^*$

$$\text{nnz}(\mathbf{A}) = 2|\mathcal{E}|$$

$$\text{nnz}(\mathbf{A}) = 2|\mathcal{E}^a| + 2|\mathcal{E}^b| + 2|\mathcal{E}^r|$$

$$14 = 2(2+2+3)$$



Discrete Combinatorial Problem

1. Two partitions of equal size:
 $|\mathcal{V}^a| = |\mathcal{V}^b| \Leftrightarrow \mathbf{x}^T \mathbf{1} = 0$

2. Minimum edge-cut:
 $\mathbf{x}^* = \arg \min \{\mathbf{x}^T \mathbf{L} \mathbf{x}\}$

No polynomial-time algorithm is known to exist.

* Assuming simple unweighted graph.

Non-Coordinate Graphs

Spectral Partitioning—An Eigenvalue Problem

Let's relax the problem ... now with $\mathbf{z} \in \mathbb{R}^n$

Continuous Problem

1. Two partitions of (roughly) equal size:

$$|\mathcal{V}^a| = |\mathcal{V}^b| \Leftrightarrow \mathbf{z}^T \mathbf{1} = 0$$

2. Minimum edge-cut:

$$\mathbf{z}^* = \operatorname{argmin} \{ \mathbf{z}^T \mathbf{L} \mathbf{z} \}$$

*What is the problem here?
(Zero Vector!)*

Eigenvalue Problem (equivalent)

$$\mathbf{z}^* = \operatorname{argmin} \{ \mathbf{z}^T \mathbf{L} \mathbf{z} / \mathbf{z}^T \mathbf{z} \} \text{ s.t.}$$

$$\mathbf{z}^T \mathbf{1} = 0$$

- \mathbf{z}^* is the eigenvector associated with the smallest nonzero eigenvalue of \mathbf{L} (**Fiedler Vector**).
- For every zero eigenvalue, there is a \mathbf{z} corresponding to a partition with zero edge-cut.

How to convert \mathbf{z}^* to labels, e.g., {0,1}? (using median)

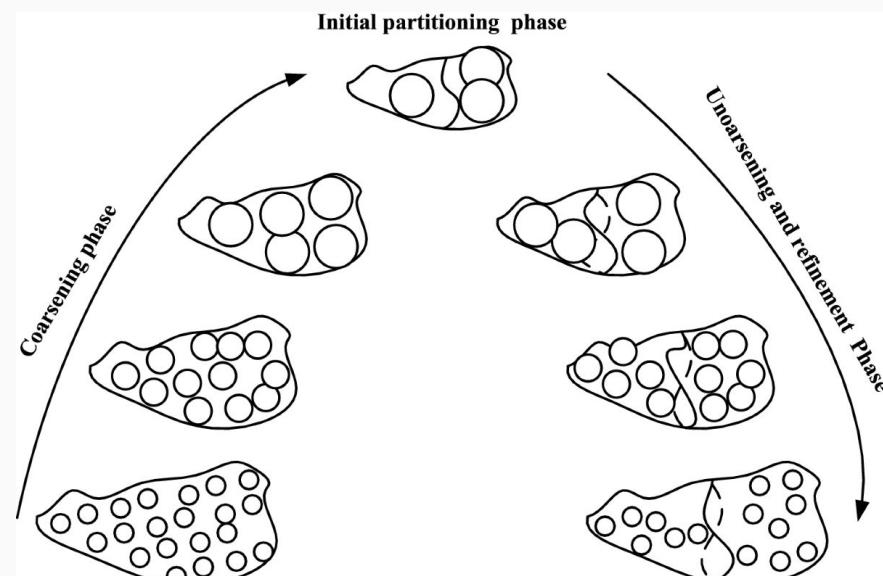
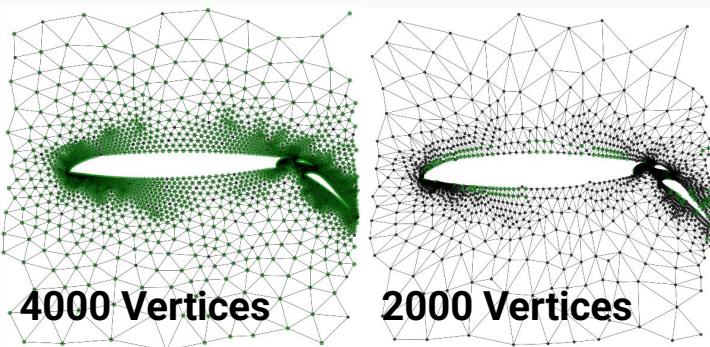
What is the difference between this eigenvalue problem and inertial partitioning?

Non-Coordinate Graphs

Multilevel Partitioning (e.g. METIS or KaHIP)

Reduce graph size via coarsening, compute **partitions on coarse levels**, and **refine** during **uncoarsening**.

- **Coarsening:** Aggregate vertices to form progressively smaller graphs.
- **Coarse Partitioning:** Compute an initial cut on the smallest graph (e.g., spectral).
- **Uncoarsening + Refinement:** Project upward and refine locally (e.g., FM).



<https://link.springer.com/article/10.1007/s00607-020-00834-5>