Università della Svizzera italiana
Institute of Computing CI

**Numerical Computing** **2024**

**Student:** POLAD BAKHISHZADE

## Solution for Project 3 **Due date:** Wednesday, 13 November 2024, 11:59 PM

### 1. Implementing Various Graph Partitioning Algorithms [50 points]

Table 1 presents the bisection edge cuts for a selection of toy meshes generated and loaded using the script 'Bench_bisection.m'.

Table 1: Bisection Results for Toy Meshes

| Mesh | Coordinate | Metis 5.0.2 | Spectral | Inertial |
|---|---|---|---|---|
| grid5rec(12,100) | 12 | 14 | 12 | 12 |
| grid5rec(100,12) | 12 | 12 | 12 | 12 |
| grid5recRotate(100,12,-45) | 22 | 12 | 12 | 12 |
| gridt(50) | 72 | 76 | 70 | 72 |
| grid9(40) | 118 | 129 | 128 | 118 |
| Smallmesh | 25 | 13 | 12 | 30 |
| Tapir | 55 | 34 | 18 | 49 |
| Eppstein | 42 | 48 | 42 | 45 |

In the inertial bisection, the center of mass was first calculated, followed by the construction of the inertia matrix $M$. The smallest eigenvector of $M$ was then obtained, and a perpendicular vector to this eigenvector was used to partition points around the median. For the spectral bisection, the Laplacian matrix $L = D - A$ was constructed, and the Fiedler vector (the second smallest eigenvector of $L$) was computed. Partitioning was then performed around zero in the Fiedler vector. Both methods included plotting to visually verify the partitions and their corresponding cut edges.

Initially, both spectral and inertial methods used the median as the partitioning threshold. Replacing the median with zero in the spectral method significantly reduced edge cuts in several meshes, notably gridt(50), Smallmesh, Tapir, and Eppstein. The most substantial improvement was seen in Tapir, where the edge cuts dropped from 58 (median) to 18 (zero). This reduction occurs because setting the threshold to zero aligns with the natural distribution of the Fiedler vector around zero. When the Fiedler vector is asymmetrically distributed, using zero instead of the median reduces imbalances and unnecessary cuts.

In the Metis bisection, the partitioning line is often curved rather than straight, due to the multilevel approach used by the algorithm, which partitions and refines based on the overall graph connectivity rather than strictly geometric alignment. The rotated grid mesh shows that coordinate bisection produces a jagged line because it is constrained to the coordinate axis, unlike spectral bisection, which leverages the Fiedler vector to align more naturally with the graph's geometry and results in a cleaner straight-line cut.

## 2. Recursively bisecting meshes [20 points]

Implemented recursive bisection to partition finite element meshes into 8 and 16 subgraphs using four methods: Coordinate, Metis 5.0.2, Spectral, and Inertial. The algorithm 'rec_bisection' was applied to meshes from 2D and 3D meshes folder, using each method to compute edge cuts. Graph partitioning divides a graph into parts, aiming for balanced partitions and minimal edge cuts. Spectral bisection leverages the Fiedler vector, Metis optimizes local partitions, Coordinate bisection splits by spatial coordinates, and Inertial partitions based on geometrical centers of mass. Table 2 summarizes the edge cuts for each method and mesh, with figures illustrating 8- and 16-way partitions for the 'de-2010' mesh.

Table 2: Recursive bisection results for each method and mesh.

| **Mesh** | **Coordinate** | | **Metis 5.0.2** | | **Spectral** | | **Inertial** | |
|---|---|---|---|---|---|---|---|---|
| | **8** | **16** | **8** | **16** | **8** | **16** | **8** | **16** |
| mesh1e1.mat | 63 | 63 | 55 | 55 | 51 | 61 | 59 | 59 |
| bodyy4.mat | 1065 | 1951 | 965 | 1620 | 1000 | 1675 | 1363 | 2212 |
| de2010.mat | 929 | 1796 | 525 | 952 | 809 | 1512 | 1080 | 1996 |
| biplane-9.mat | 548 | 974 | 467 | 831 | 411 | 794 | 647 | 1092 |
| L-9.mat | 631 | 1028 | 648 | 1016 | 718 | 1121 | 828 | 1378 |

Metis consistently minimized edge cuts across the majority of the meshes, indicating its optimization capabilities. This is noticeable in meshes such as de2010.mat and mesh1e1.m, where Metis consistently provided the most efficient partitions. Its approach of optimizing local partitions effectively reduced the number of cut edges, especially in more complex structures. Spectral partitioning also demonstrated strong performance, particularly for simpler meshes. For example, in biplane-9.mat and mesh1e1.mat, the Spectral method achieved cut sizes close to those of Metis. By leveraging the Fiedler vector, Spectral partitioning ensured balanced cuts, especially when dealing with meshes that are less complex in structure.

When it comes to figures below, the Coordinate method produced partitions that generally align with the coordinate axes, resulting in rectangle-like shapes for both 8 and 16 partitions. In contrast, the Inertial method generated partitions that resemble triangle-like shapes, influenced by the center of mass and the distribution of mass within each partition. This distinction in partition shapes highlights the effect of each method's underlying partitioning strategy on the geometry of the final partitions.

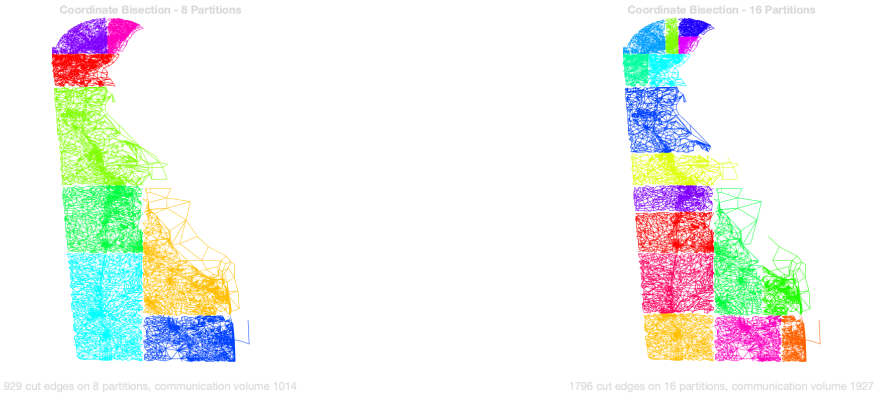


Figure 1: Coordinate Bisection: 8-partitions (left) and 16-partitions (right).

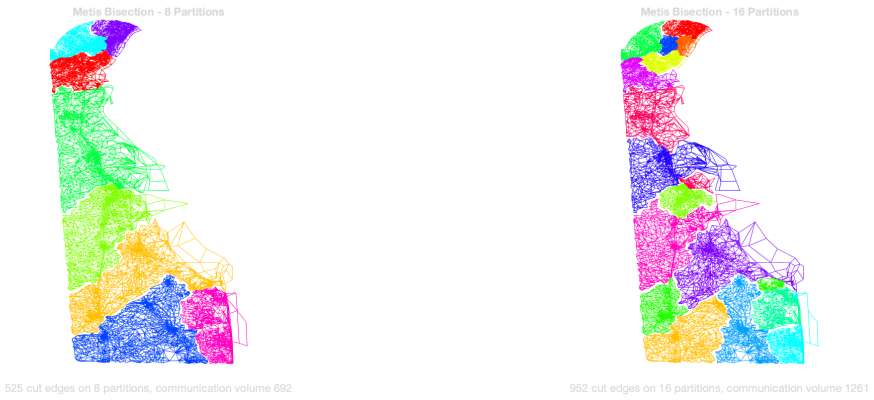


Figure 2: Metis Bisection: 8-partitions (left) and 16-partitions (right).

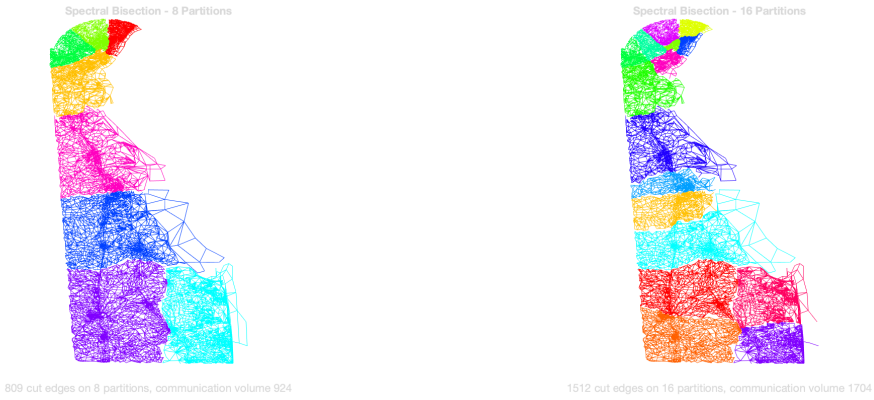


Figure 3: Spectral Bisection: 8-partitions (left) and 16-partitions (right).


Figure 4: Inertial Bisection: 8-partitions (left) and 16-partitions (right).

## 3. Comparing recursive bisection to direct $k$-way partitioning [15 points]

This task involves comparing recursive bisection and direct k-way partitioning using Metis 5.0.2 on two example meshes: *helicopter* and *skirt*. Recursive bisection divides the graph in a step-by-step manner, while direct k-way partitioning uses a global optimization approach aimed at minimizing cut edges across all partitions. The algorithm was implemented with Metis 5.0.2 called via function `metismex` for both recursive bisection and direct k-way partitioning. Each method was applied to the helicopter and skirt meshes, with partitions set to 16 and 32. The Rotate 3D function was enabled to better visualize the results, particularly useful for the 3D helicopter mesh.

Table 3 presents the edge cut results for each partitioning method. Figures 5 and 6 illustrate the 32-partition results for both the recursive and direct partitioning methods.

Table 3: Bisection results

| **Partitions** | **Helicopter** | **Skirt** |
|---|---|---|
| 16 - Recursive Bisection | 343 | 3119 |
| 16 - Way Direct Partition | 324 | 3393 |
| 32 - Recursive Bisection | 538 | 6345 |
| 32 - Way Direct Partition | 531 | 6284 |

Overall, direct k-way partitioning achieved slightly lower edge cuts than recursive bisection, as expected. However, the differences were generally not significant. For example, in the 16-partition case, recursive bisection on the helicopter mesh had 343 edge cuts, while direct partitioning had 324, a minor difference. Interestingly, for the skirt mesh at 16 partitions, recursive bisection (3119) actually achieved fewer cuts than the direct partition (3393), contradicting the general expectation.

In the 32-partition case, direct partitioning again achieved slightly fewer cuts for the helicopter mesh (531 vs. 538), though the difference was marginal. The most noticeable difference was for the skirt mesh with 32 partitions, where direct partitioning produced 6284 cuts compared to 6345 for recursive bisection, showing a slight improvement.


Figure 5: Helicopter: 32-Recursive Bisection (left) and 32-Way Direct Partition (right).


Figure 6: Skirt: 32-Recursive Bisection (left) and 32-Way Direct Partition (right).

Direct k-way partitioning generally provided lower edge cuts than recursive bisection, but the differences were often minimal, especially in the helicopter mesh. The most notable improvement

was observed in the 32-partition case for the skirt mesh, where direct partitioning outperformed recursive bisection with a reduction of 61 edge cuts. These results suggest that while direct k-way partitioning has slight advantages, the recursive bisection approach can still yield comparable results, as observed in the 16-partition case for the skirt mesh.