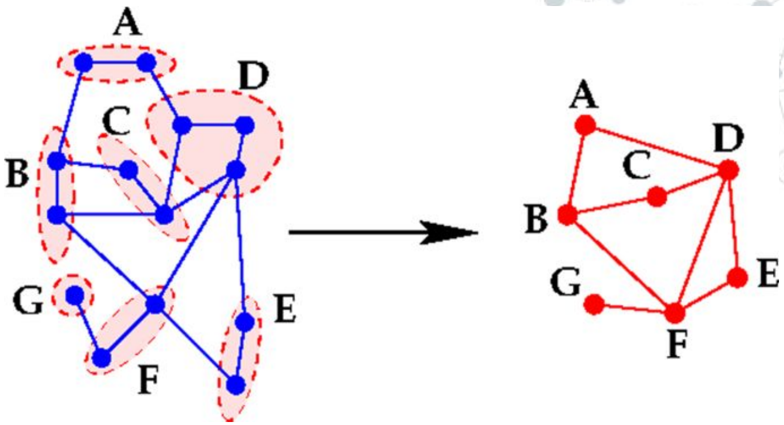
TDA Graph Coarsening Benchmark

Kelly Williams

Background

Motivation for Graph Coarsening

- Large-scale graphs may involve complex structures
  - Difficult to compute and analyze key properties directly from large graphs
- **Coarsened Graph** – Graph of reduced size that preserves important graph properties

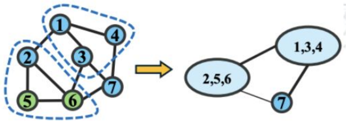


https://link.springer.com/article/10.1007/s40324-021-00282-x

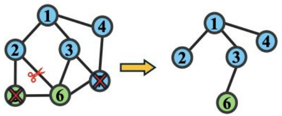
- Applications: GNNs, Biology, Graphics, etc.
- Related: Clustering unstructured point-cloud data
  - Ball Mapper & Mapper

Brief Aside

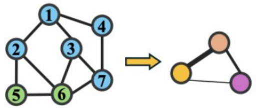
- ***A Comprehensive Survey on Graph Reduction: Sparsification, Coarsening, and Condensation*** (2024) – Hashemi, et al.
- **<u>Graph Reduction</u>** – General term for reducing the size of the graph dataset, including the number of graphs, nodes, and edges
  - **Coarsening** – Group and aggregate similar nodes and edges to construct a smaller graph
  - **Sparsification** – Selecting significant nodes and edges while discarding others
  - **Condensation** – Learn a synthetic graph from scratch



Coarsening



Sparsification



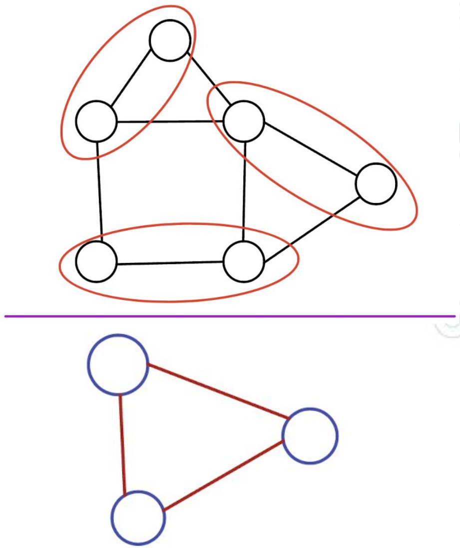
Condensation

Coarsening Algorithms

Maximal Matching Coarsening

Basic Algorithm

- Find a maximal matching in the graph
  1. **Maximal Matching** – maximal set of edges, no two of which are incident on the same vertex



- For each Matching Edge (i; j):
  1. Contract edge to **form new vertex v**
  2. Accumulate **vertex weight** → weight(v) := weight(i) + weight(j)
  3. Connect neighboring edges to new vertex v
  4. If i and j were both adjacent to a neighbor vertex k
     - Accumulate **edge weight**(v; k) := weight(i; k) + weight(j; k)

Hendrickson & Leland (1997)

Spectral Guarantees Coarsening

(a little more complicated…)

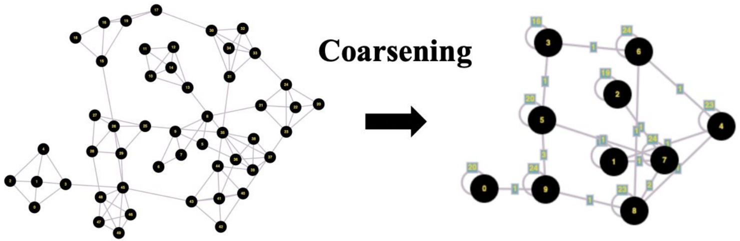
Input: Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{W})$, eigenvectors $\boldsymbol{U}$ of the normalized Laplacian $\boldsymbol{\mathcal{L}}$, target size n.
if $\boldsymbol{\lambda}(N) \leq 1$ **then**
  Set $k_1 \leftarrow n$ ▷ Spectral Clustering
else
  Set $k_1 \leftarrow \arg\min_k \{k : \boldsymbol{\lambda}(k) \leq 1, k \leq n, \boldsymbol{\lambda}(N - n + k + 1) > 1\}$ ▷ Iterative Spectral Coarsening
$k_2 \leftarrow N - n + k_1$.
while $k_1 \leq n$ **do**
  $\boldsymbol{U}_{k_1} \leftarrow [\boldsymbol{U}(1 : k_1); \boldsymbol{U}(k_2 + 1 : N)]$
  Apply k-means clustering algorithm on the rows of $\boldsymbol{U}_{k_1}$ to obtain graph partitions $\mathcal{P}^*_{k_1}$ that optimizes the following k-means cost:

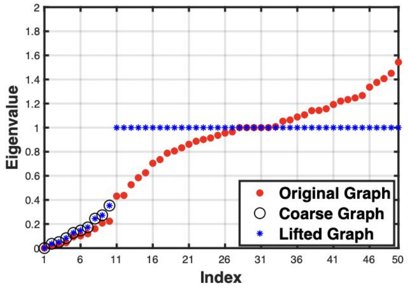
$$\mathcal{F}(\boldsymbol{U}_{k_1}, \mathcal{P}^*_{k_1}) = \sum_{i=1}^{N} \left(\boldsymbol{r}(i) - \sum_{j \in \mathcal{S}_i} \frac{\boldsymbol{r}(j)}{|\mathcal{S}_i|} \right)^2$$

  where $\boldsymbol{r}(i)$ is the i^{th} row of $\boldsymbol{U}_{k_1}$.
  $k_1 \leftarrow k_1 + 1$, $k_2 = N - n + k_1$
return coarse graph $\mathcal{G}_c$ generated with respect to the partitions with minimum k-means clustering cost as

$$\mathcal{P}^* = \arg\min_{k_1} \mathcal{F}(\boldsymbol{U}_{k_1}, \mathcal{P}^*_{k_1})$$

- Build a subgraph that is as structurally similar to the original graph as possible





Jin, Loukas, JaJa 2020

Parallel 2MIS

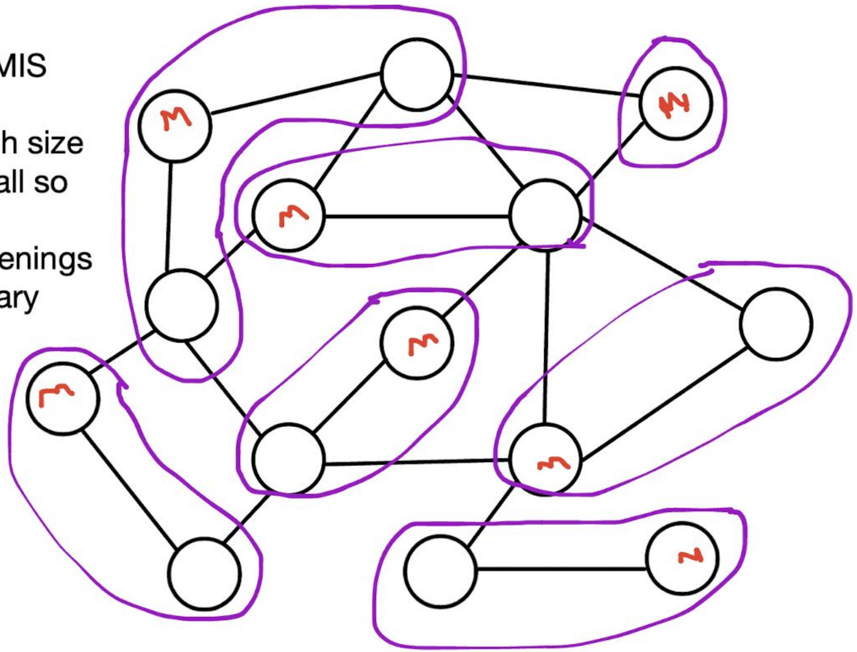
- Find the Maximal Independent Set in parallel to determine which nodes in the **original graph G** are Supernodes

- Nodes of the new coarsened graph become all nodes within 2 edges of the Supernodes in **G** [2MIS]

- Similar accumulation process to Maximal Matching

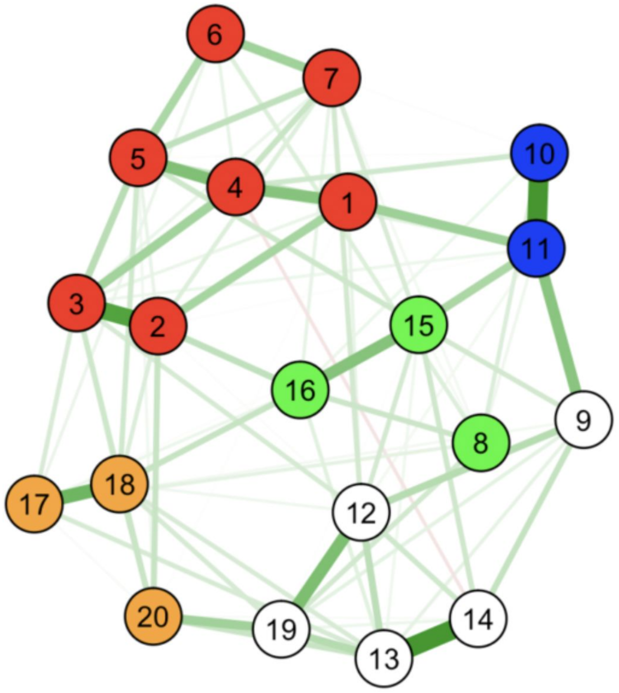
Ex. 1MIS

*Graph size is small so exact coarsenings can vary



Adapted Walktrap Community Detection

- Algorithm based on Pons and Latapy's Community Detection
- **Choose a set of nodes** and **perform random walks** that are a specified set of **steps** away to determine communities
- Adapted for coarsening: the **communities** become **Supernodes** for the coarsened graph
- Chosen for its similarity to the BallMapper algorithm (Dlotko, 2019)
  - Radius of ball in point cloud = Steps from node in structured graph



https://psych-networks.com/r-tutorial-identify-communities-items-networks/

Pons & Latapy, 2005

Benchmark

Implementation

- Graph: **bcspwr10** from Graph500
  - # of Vertices = 5300
  - # of Edges = 8271

- Coarsen the graph with each algorithm and analyze metrics
  - Do 1 round of coarsening
  - If the algorithm requires a coarsening goal, use **75%** of the original graph

- *Some algorithms are self-implemented and may not be in their most efficient state

Current Metrics

- # of Vertices
- # of Edges
- Approximate Connectivity
- Density
- # of Connected Components
- # of Basis Cycles
- *Time → not all implementations are the most efficient

Results

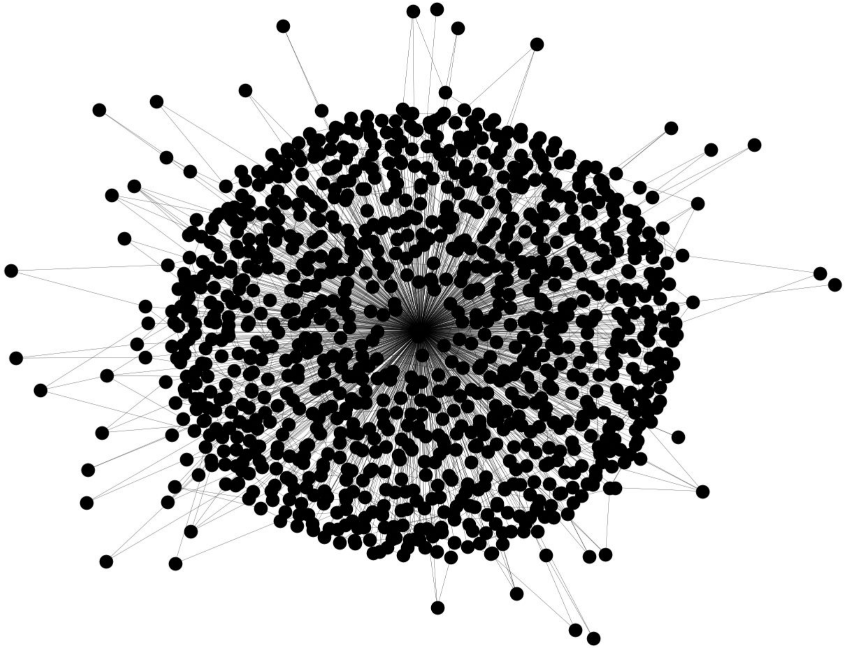
(so far)

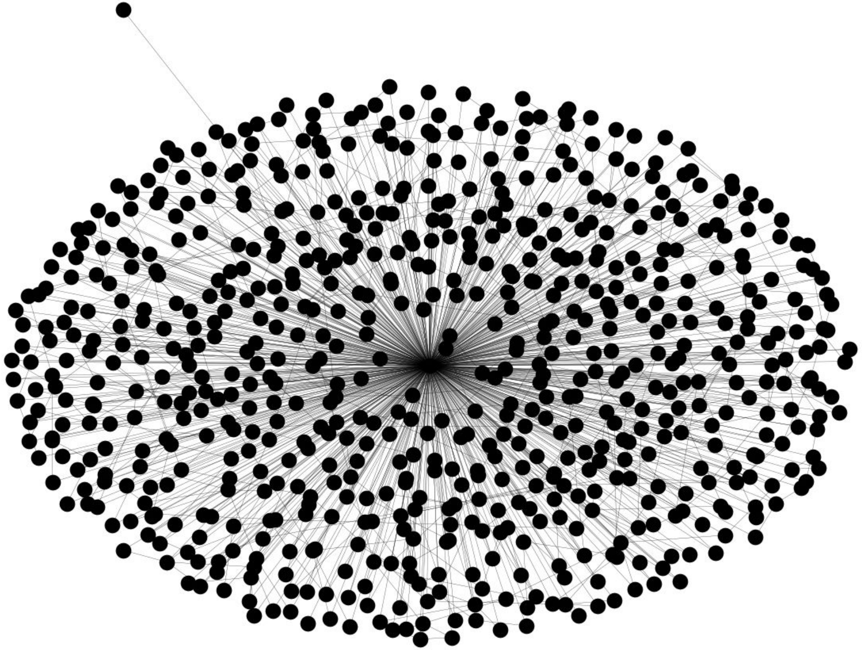
Results Table

	|V|	|E|	~ Connectivity	Density	# CCs	# Basis Cycles	Time (s)
Original	5300	8271	1	0.000589	1	2972	
MaxMatch	2916 (0.55)	5406 (0.65)	1	0.001272	1	2491	0.0110979
Spectral r=0.25	2187 (0.41)	4176 (0.50)	1	0.001747	1	1990	2.0681710
2MIS	1090 (0.21)	4904 (0.59)	1	0.003598	1	1047	0.12111700
Walktrap step=4	47 (0.008)	86 (0.01)	1	0.079556	1	40	0.0636663

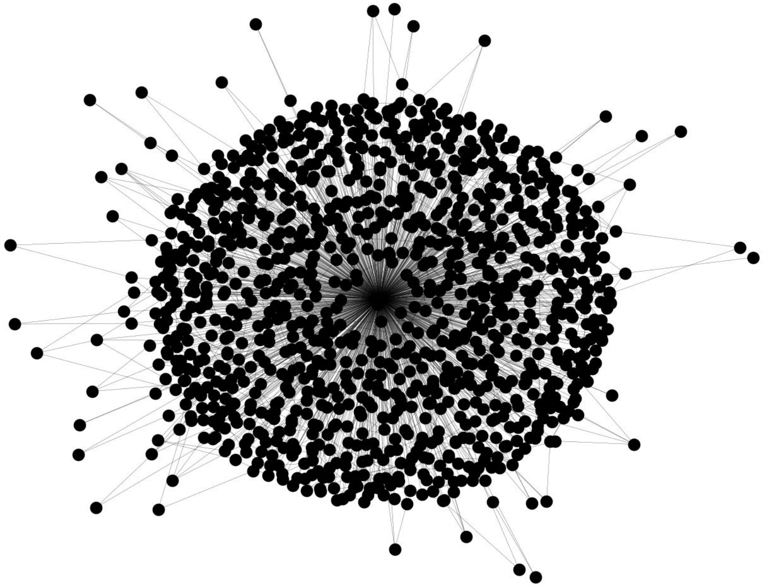
*Estrada Index is not normalized

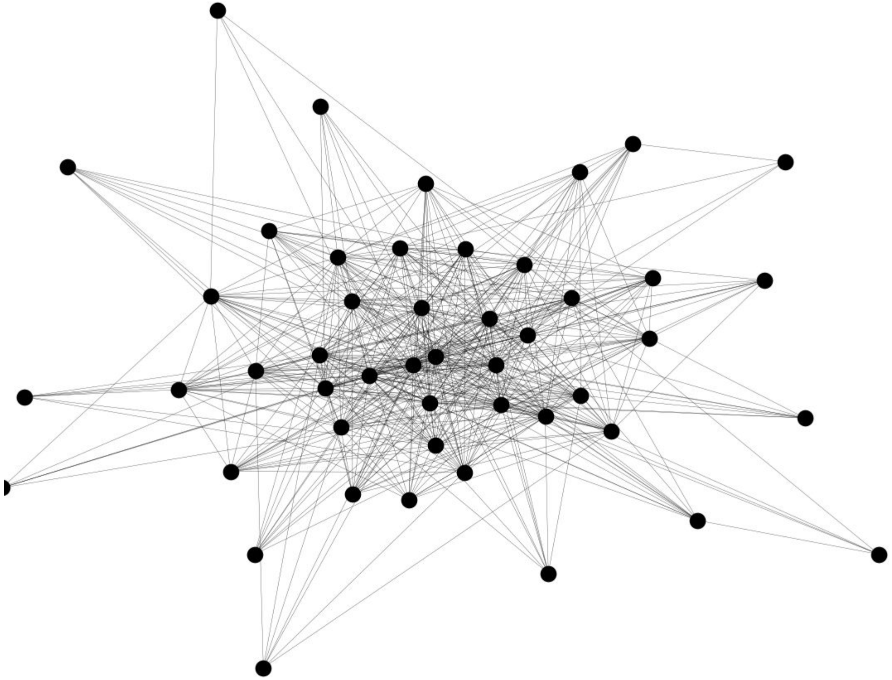
Maximal Matching Coarsening



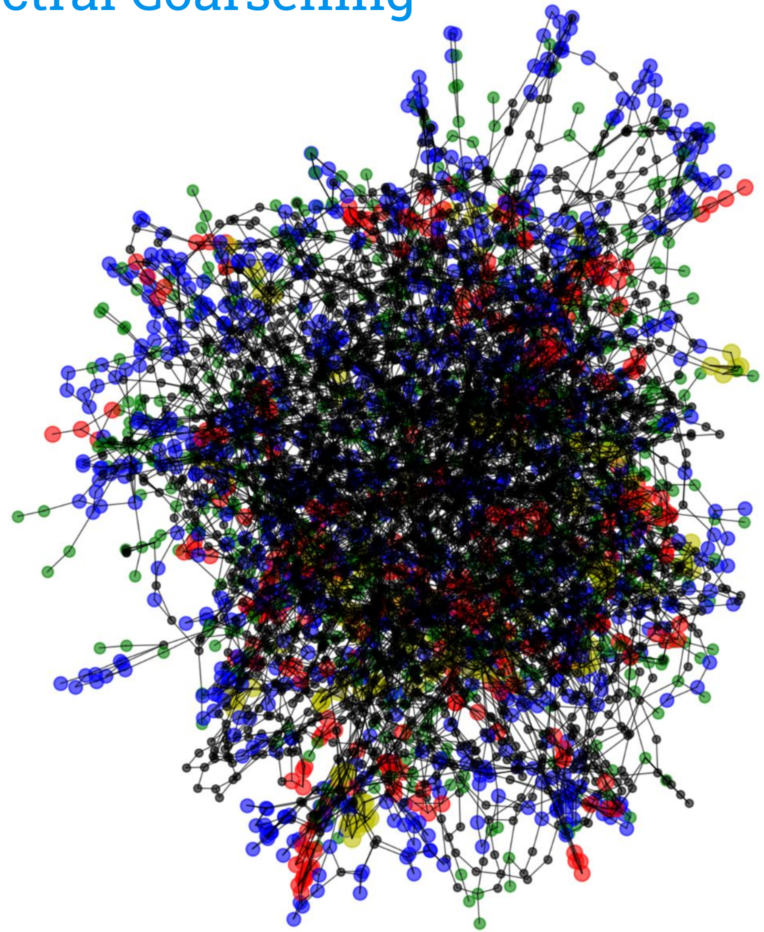


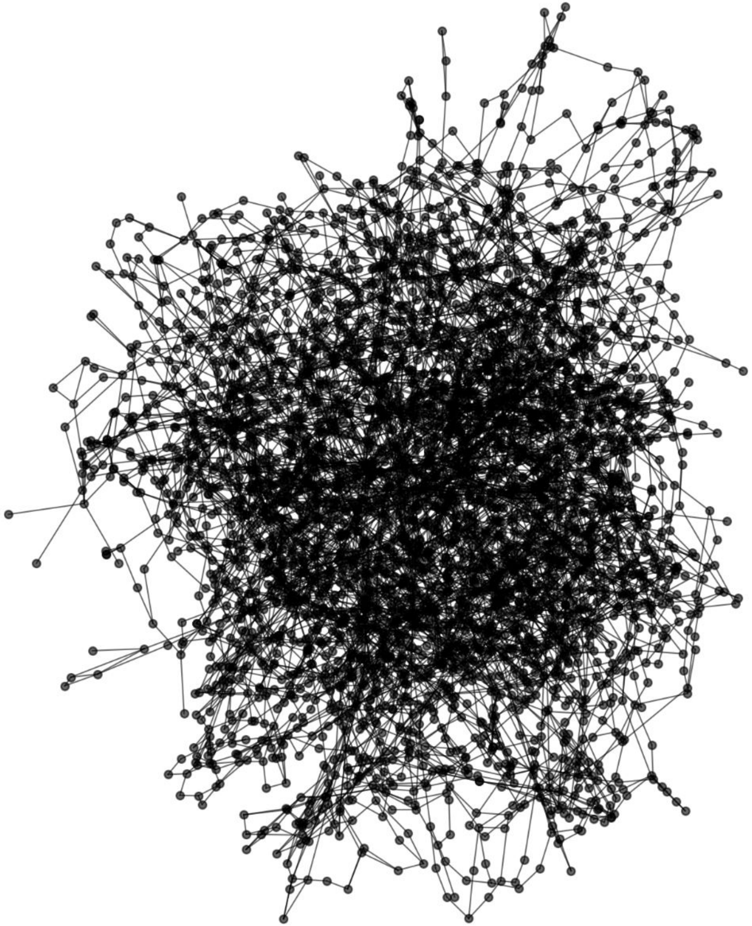
Walktrap Coarsening





Spectral Coarsening





Time Complexity (Theoretical)

- Maximal Matching Coarsening = **$O(M) * O(k)$**
  - M = # of edges in Maximal Matching
  - k = time to edit graph (depends in data structures)
- Spectral Coarsening = **$O(KTNn^2)$**
  - K = # of while loop k-cluster iterations w/ $K \leq n$
  - TNn^2 = the complexity of the k-means clustering
    - T bounds the # of k-means iterations
- Parallel 2MIS* = **$O(V \log V + E \log V + V \log^2 V)$**
  - Parallel algorithms are complex and this does not cover the whole coarsening process
- Walktrap = **$O(n^2 \log n) * O(k)$**
  - k = time to edit graph (depends in data structures)

Future Work

Future Work

- Parallelize the algorithms more when applicable
- Adjust Walktrap step size if possible
- Explore at least 1 more algorithm in the current benchmark
- Include more metrics and include nicer visuals to better represent the results
- Compare the algorithms in C/C++ in order to compare performance on different architectures
- Apply benchmark to more complex algorithms [Zollner, 2011]

Final Report Goals (stay tuned)

- Involve persistence to track features as graph is coarsened
- See if Ball Mapper can be applied to data that is structured and has relations
- Add one or two more algorithms
- Analyze more metrics based on the coarsenings

References

- Jin, Loukas, JaJa. (2020). **Graph Coarsening with Preserved Spectral Properties**.
- Pons, Latapy. (2005). **Computing communities in large networks using random walks.**
- Hedrickson, Leland. (1997). **A Multilevel Algorithms for Partitioning Graphs.**
- Kelley, Rajamanickam. (2022). **Parallel, Portable Algorithms for Distance-2 Maximal Independent Set and Graph Coarsening**.
- Hashemi, Gong, Ni, et al. (2024). **A Comprehensive Survey on Graph Reduction: Sparsification, Coarsening, and Condensation.**
- Zollner. (2011). **A Potts model for junction limited grain growth.**
- Madukpe, et al. (2024). **Comparative analysis of Ball Mapper and conventional Mapper in investigating air pollutants' behavior.**

Thank You