Spectral Clustering-Based Analysis of ERC-721 Blockchain Transactions

Piyush Anand
CS21B1019
Dept. Of Computer Science and Engineering
IIIT Raichur

Under the guidance of
Dr. Priodyuti Pradhan

Goal

Identify groups /communities of users interacting within the ERC-721 blockchain dataset.

Blockchain: The Foundation of Decentralized Systems

- Distributed, immutable ledger ensuring trustless transactions
- Enables peer-to-peer value exchange without intermediaries
- Basis for advanced applications like NFTs and smart contracts

Feature	Traditional Systems	Distributed Ledger (Blockchain)
Trust Mechanism	Central authority	Cryptography, consensus
Record Immutability	Vulnerable to tampering	Tamper-evident, permanent
Transaction Model	Intermediated	Peer-to-peer, direct
Advanced Applications	Limited, manual contracts	Smart contracts, NFTs, automation
Transparency and Auditability	Restricted, siloed	Universal, real-time

Credits : Perplexity AI

Real-World Applications of Blockchain Technology

- Secure financial transactions (DeFi, cross-border payments)
- Transparent supply chain and asset tracking
- Identity verification and digital voting systems

Image Credits :AI

NFTs: Revolutionizing Digital Ownership

- Unique, indivisible digital assets stored on blockchain
- Enables verifiable ownership of art, music, and collectibles
- Powered by smart contracts on Ethereum & similar platforms

Image Credits :AI

ERC-721: The Standard Behind NFTs

- Ethereum-based standard for non-fungible tokens
- Supports tracking ownership and transfer of unique assets

Image Credits :AI

Dataset Collection: Real NFT Transactions

Publicly available ERC-721 transaction data downloaded from X-Block ETH

Image Credits :Google

Data Pre-processing: Reducing the Data Size

blockNumber	timestamp	transactionHash	tokenAddress	from	to	fromIsContract	toIsContract	tokenId
1001165	1455424860	0x1d11b3ea559	0x55b9a11c2e8	0x38150290c18	0x3d2068aeb96	0	0	200000000
1001165	1455424860	0x1d11b3ea559	0x55b9a11c2e8	0x38150290c18	0x5d2c24efac49	0	1	0
1003181	1455460299	0x26a30aa663f6	0x55b9a11c2e8	0x38150290c18	0x3d2068aeb96	0	0	500000000
1003181	1455460299	0x26a30aa663f6	0x55b9a11c2e8	0x38150290c18	0x5d2c24efac49	0	1	0
1003393	1455463847	0xc0db923ac0c6	0x55b9a11c2e8	0x3d2068aeb96	0xb51446cc4291	0	0	100000000
1003393	1455463847	0xc0db923ac0c6	0x55b9a11c2e8	0x3d2068aeb96	0x5d2c24efac49	0	1	130000
1005878	1455505636	0x113de15964f8	0x55b9a11c2e8	0x38150290c18	0xc0cfc0969d0d	0	0	1000000000
1005878	1455505636	0x113de15964f8	0x55b9a11c2e8	0x38150290c18	0x5d2c24efac49	0	1	0
1005946	1455506683	0x09581f238af3	0x55b9a11c2e8	0xc0cfc0969d0d	0xa220568ace92	0	0	100000000
1005946	1455506683	0x09581f238af3	0x55b9a11c2e8	0xc0cfc0969d0d	0x5d2c24efac49	0	1	130000
1006034	1455508426	0x637e4263f782	0x55b9a11c2e8	0x38150290c18	0xa220568ace92	0	0	200000000
1006034	1455508426	0x637e4263f782	0x55b9a11c2e8	0x38150290c18	0x5d2c24efac49	0	1	0
1006046	1455508554	0x5eac9cbd9cae	0x55b9a11c2e8	0xc0cfc0969d0d	0xa220568ace92	0	0	898000000
1006046	1455508554	0x5eac9cbd9cae	0x55b9a11c2e8	0xc0cfc0969d0d	0x5d2c24efac49	0	1	1167400
1006059	1455508793	0xc71da93bc664	0x55b9a11c2e8	0x38150290c18	0xa220568ace92	0	0	4000000000
1006059	1455508793	0xc71da93bc664	0x55b9a11c2e8	0x38150290c18	0x5d2c24efac49	0	1	0
1007626	1455536210	0x6c6729b40547	0x55b9a11c2e8	0x38150290c18	0xad917335252	0	0	200000000
1007626	1455536210	0x6c6729b40547	0x55b9a11c2e8	0x38150290c18	0x5d2c24efac49	0	1	0
1007645	1455536477	0x0a67fa2e1500	0x55b9a11c2e8	0x38150290c18	0xad917335252	0	0	1800000000
1007645	1455536477	0x0a67fa2e1500	0x55b9a11c2e8	0x38150290c18	0x5d2c24efac49	0	1	0
1007784	1455539020	0x8d55d21c2e7	0x55b9a11c2e8	0x38150290c18	0xad917335252	0	0	8000000000

Timestamp Conversion: UNIX to Human Readable

blockNumber	timestamp	transactionHash	tokenAddress	from	to	fromIsContrac	toIsContract	tokenId
4585326	2017-11-20 0:51:46	0xd467a51842ac6	0x441d1b228cad4	0x000000000	0x12e3da7ef08	0	0	1
4585337	2017-11-20 0:53:12	0x44a00538df1c2	0x441d1b228cad4	0x000000000	0x12e3da7ef08	0	0	2
4585342	2017-11-20 0:55:09	0x034a644c344ff9	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	3
4585351	2017-11-20 0:58:13	0x51103e6d153bf	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	4
4585354	2017-11-20 0:59:11	0x6308acf866c01	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	5
4585356	2017-11-20 0:59:18	0x29e6ae148db8b	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	6
4585356	2017-11-20 0:59:18	0x275350d21811b	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	7
4585357	2017-11-20 0:59:49	0xf5cd3789299ab	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	8
4585359	2017-11-20 1:00:34	0x5807656c322a4	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	9
4585363	2017-11-20 1:01:31	0xf2a40fcf1867ae	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	10
4585363	2017-11-20 1:01:31	0xea997d61cb21f	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	11
4585376	2017-11-20 1:04:56	0x91cd09d8ae62b	0x441d1b228cad4	0xba52c7576	0xe727e07021	0	1	3
4585398	2017-11-20 1:08:49	0x76c2656ef9836	0x441d1b228cad4	0xba52c7576	0xe727e07021	0	1	4
4587251	2017-11-20 8:13:21	0x6f9202d553ebd	0x59061b6f26bb4	0x92d8fce77	0xbba49956f1	0	0	100000000
4587571	2017-11-20 9:20:05	0xa829b36bf3c2e	0x59061b6f26bb4	0x92d8fce77	0xbba49956f1	0	0	108308399587
4588239	2017-11-20 11:54:39	0x94ace52e0424b	0x2ee13cbd30471	0x72855c26e	0x1d6ff33eb64	1	0	1E+21
4588843	2017-11-20 14:13:12	0x4af72fa908eb3	0x59061b6f26bb4	0xfbb1b73c4f	0x0bc8e22a82	0	1	23905000000
4588853	2017-11-20 14:17:15	0x6b630b1e0bbab	0x59061b6f26bb4	0x0bc8e22a8	0xfbb1b73c4f0	1	0	23905000000

Divide Daywise: File for every day

blockNumber	timestamp	transactionHash	tokenAddress	from	to	fromIsContrac	toIsContract	tokenId
4585326	2017-11-20 0:51:46	0xd467a51842ac6	0x441d1b228cad4	0x000000000	0x12e3da7ef08	0	0	1
4585337	2017-11-20 0:53:12	0x44a00538df1c2	0x441d1b228cad4	0x000000000	0x12e3da7ef08	0	0	2
4585342	2017-11-20 0:55:09	0x034a644c344ff9	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	3
4585351	2017-11-20 0:58:13	0x51103e6d153bf	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	4
4585354	2017-11-20 0:59:11	0x6308acf866c01	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	5
4585356	2017-11-20 0:59:18	0x29e6ae148db8b	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	6
4585356	2017-11-20 0:59:18	0x275350d21811b	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	7
4585357	2017-11-20 0:59:49	0xf5cd3789299ab	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	8
4585359	2017-11-20 1:00:34	0x5807656c322a4	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	9
4585363	2017-11-20 1:01:31	0xf2a40fcf1867ae	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	10
4585363	2017-11-20 1:01:31	0xea997d61cb21f	0x441d1b228cad4	0x000000000	0xba52c75764	0	0	11
4585376	2017-11-20 1:04:56	0x91cd09d8ae62b	0x441d1b228cad4	0xba52c7576	0xe727e07021	0	1	3
4585398	2017-11-20 1:08:49	0x76c2656ef9836	0x441d1b228cad4	0xba52c7576	0xe727e07021	0	1	4
4587251	2017-11-20 8:13:21	0x6f9202d553ebd	0x59061b6f26bb4	0x92d8fce77	0xbba49956f1	0	0	100000000
4587571	2017-11-20 9:20:05	0xa829b36bf3c2e	0x59061b6f26bb4	0x92d8fce77	0xbba49956f1	0	0	108308399587
4588239	2017-11-20 11:54:39	0x94ace52e0424b	0x2ee13cbd30471	0x72855c26e	0x1d6ff33eb64	1	0	1E+21
4588843	2017-11-20 14:13:12	0x4af72fa908eb3	0x59061b6f26bb4	0xfbb1b73c4f	0x0bc8e22a82	0	1	23905000000
4588853	2017-11-20 14:17:15	0x6b630b1e0bbab	0x59061b6f26bb4	0x0bc8e22a8	0xfbb1b73c4f0	1	0	23905000000

Labelling: For fast computation

timestamp	fromLabel	toLabel	tokenAddressLabel	fromIsContract	toIsContract	tokenId
2017-11-20 1:04:56	766	767	45	0	1	3
2017-11-20 1:08:49	766	767	45	0	1	4
2017-11-20 8:13:21	706	768	24	0	0	100000000
2017-11-20 9:20:05	706	768	24	0	0	1083083995871
2017-11-20 11:54:39	661	769	42	1	0	1E+21
2017-11-20 14:13:12	99	763	24	0	1	23905000000
2017-11-20 14:17:15	763	99	24	1	0	23905000000
2017-11-20 14:53:50	99	770	24	0	1	405000000
2017-11-20 14:56:09	770	99	24	1	0	405000000
2017-11-20 14:57:32	99	770	24	0	1	977603790647
2017-11-20 14:58:54	770	99	24	1	0	977603790647
2017-11-20 18:40:29	766	767	45	0	1	5
2017-11-20 18:42:06	767	765	45	1	0	5
2017-11-20 22:58:40	661	771	42	1	0	1E+21

Why Clustering? Understanding Wallet Behaviors

- Group similar wallets to detect communities or fraud rings
- Reduces complexity of large-scale transaction graphs
- Useful in NFT trend analysis, bot detection, and monitoring
- NULL Nodes found out

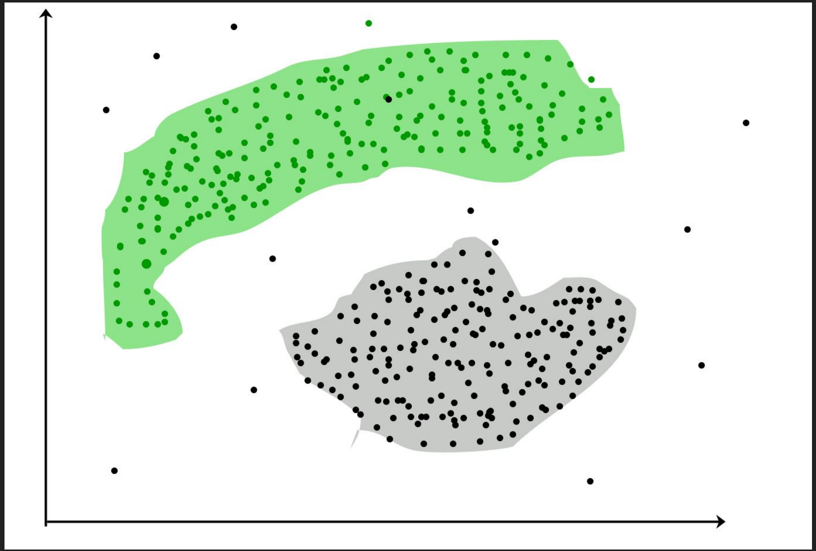
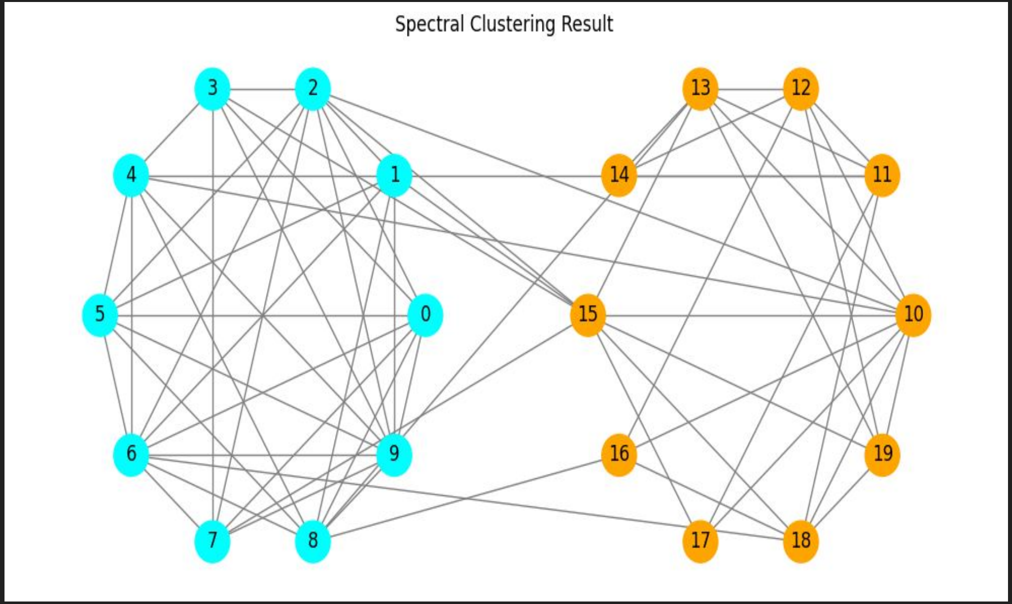
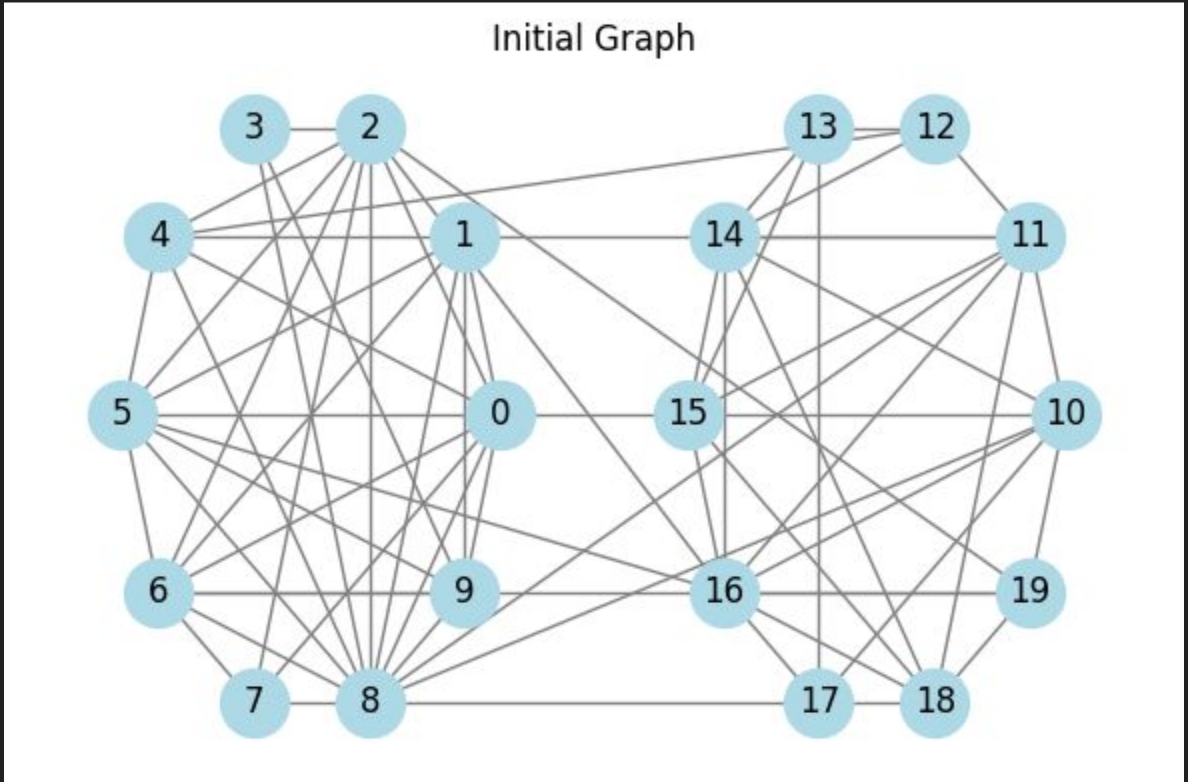


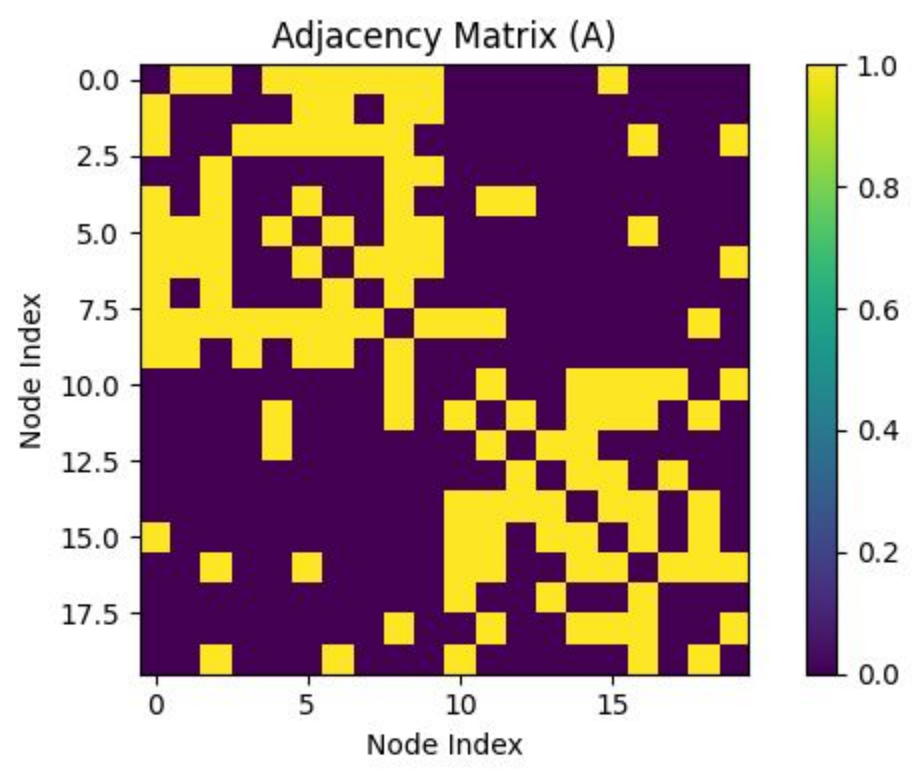
Image Credits :GFG

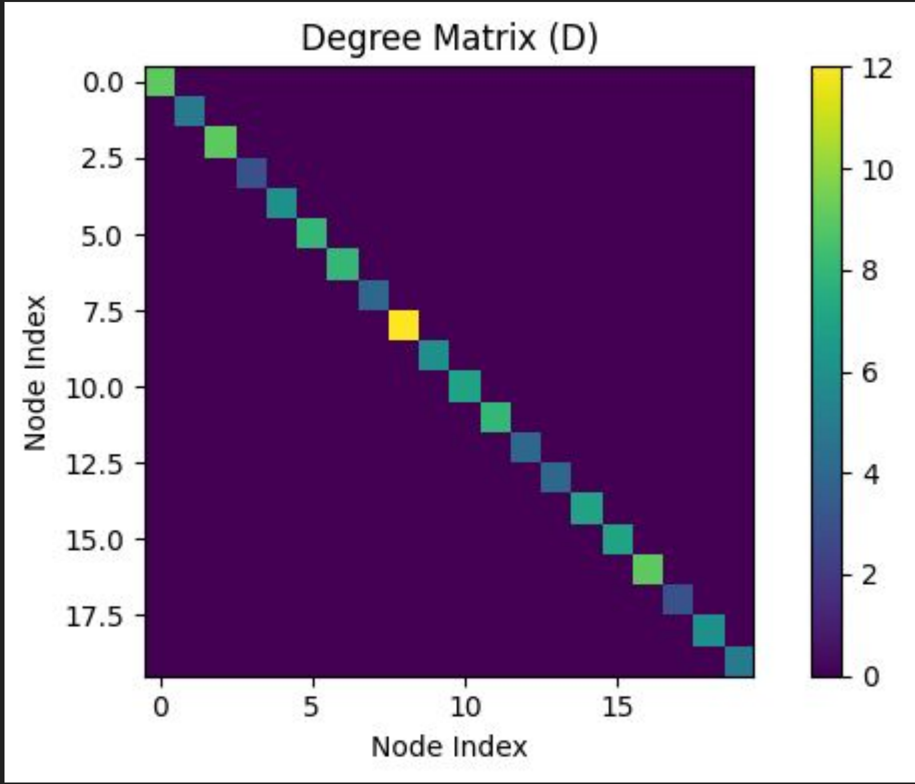
Spectral Clustering: Graph-based Group Discovery

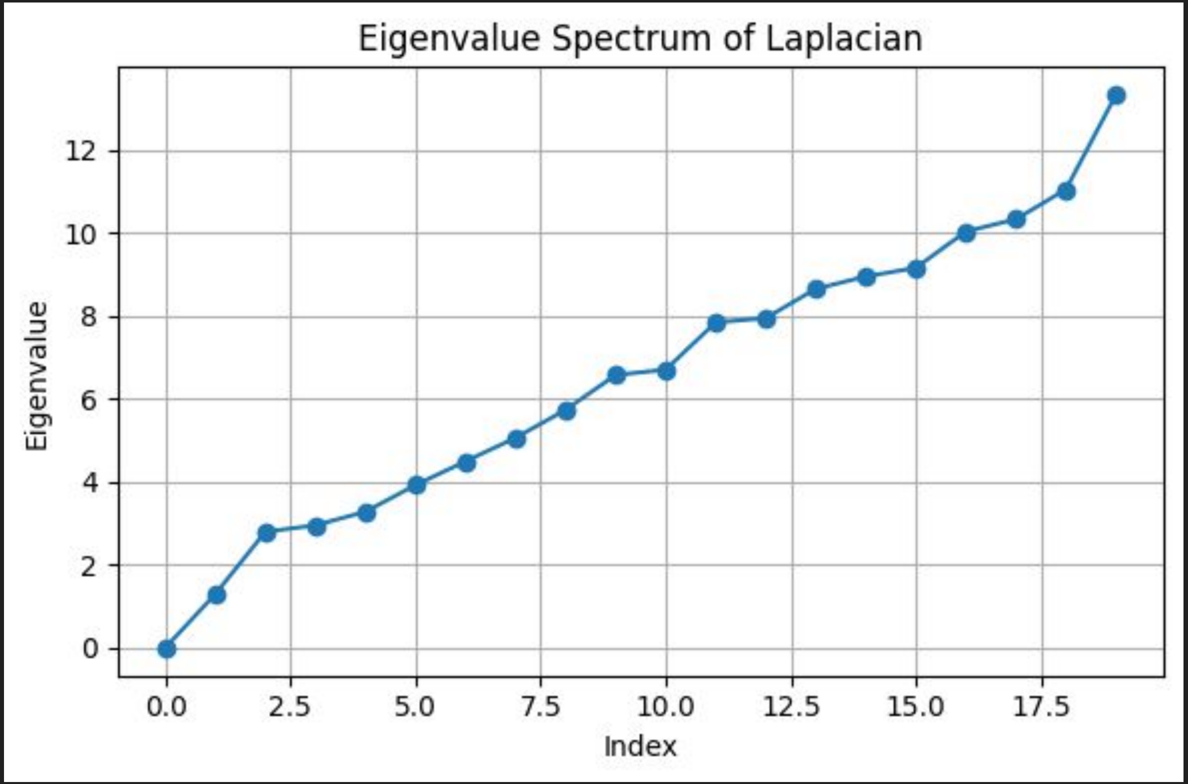
- Partition nodes into clusters based on graph structure
- Uses the Spectrum of Graph Laplacian Matrix
- Effective when clusters are not well separated.



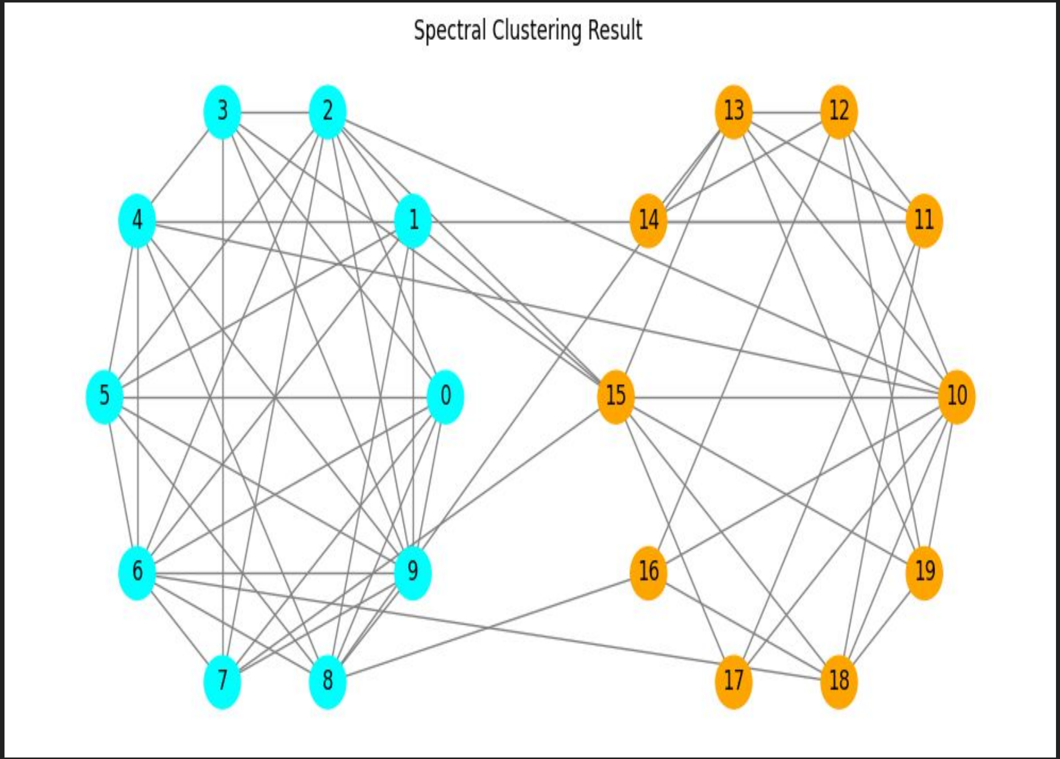

Initial Graph
3
2
13
12
4
1
14
11
5
0
15
10
6
9
16
19
7
8
17
18


Adjacency Matrix (A)
Node Index
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
0
5
10
15
Node Index
1.0
0.8
0.6
0.4
0.2
0.0


Degree Matrix (D)
Node Index
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
0
5
10
15
Node Index
12
10
8
6
4
2
0

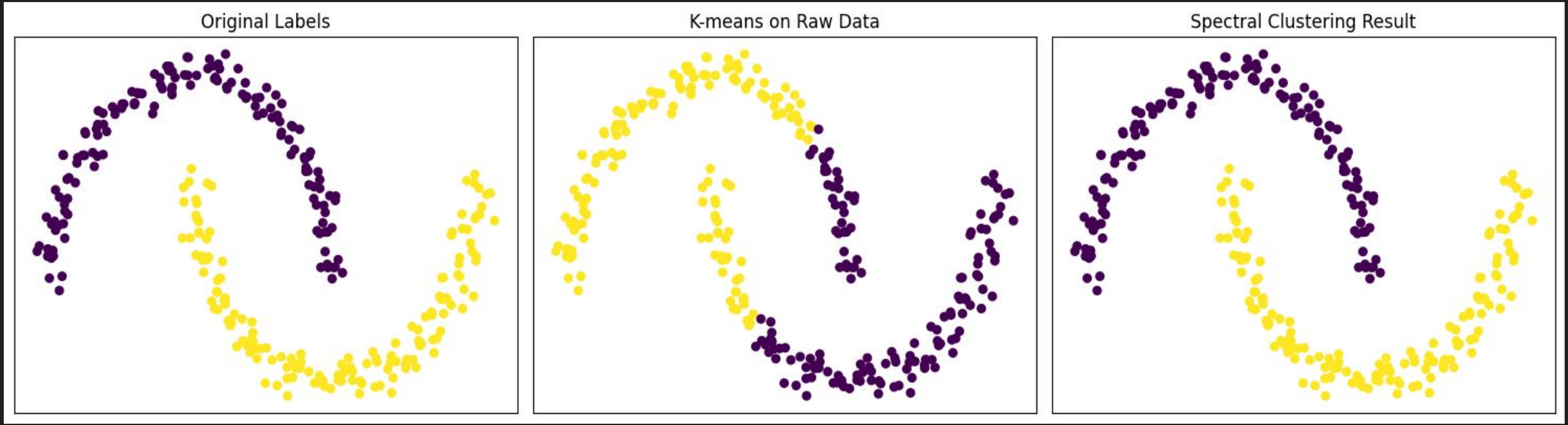

Eigenvalue Spectrum of Laplacian
Eigenvalue
Index
0
2
4
6
8
10
12
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

Clustered Graph



Spectral Clustering vs K-Means: Key Differences

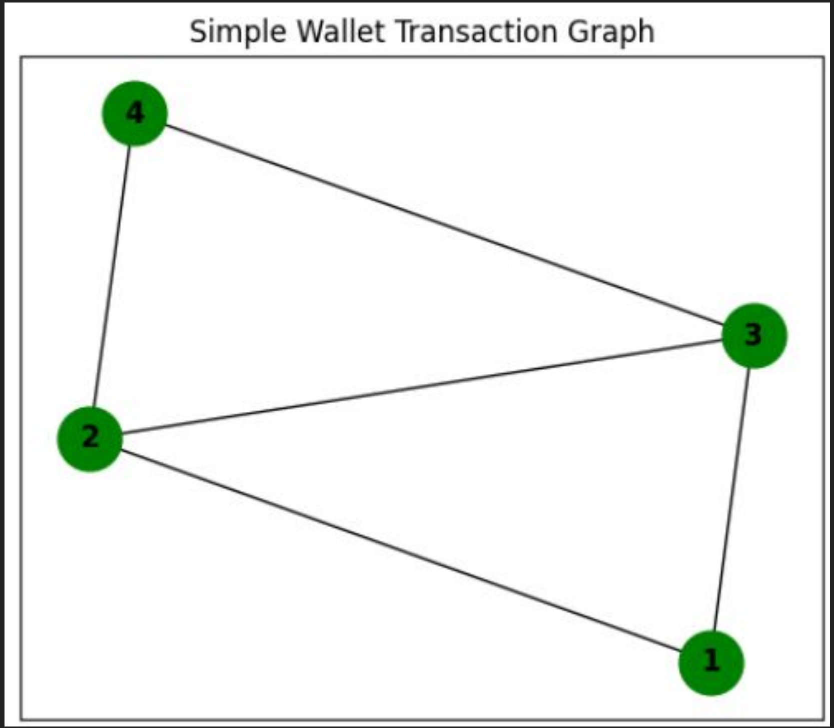
- Spectral works on graph Laplacian; K-Means uses Euclidean distance
- Handles non-convex clusters better than K-Means
- Doesn't assume spherical clusters or equal sizes



Graph Construction: Wallets as Nodes, Transfers as Edges

Created undirected graphs from token transfers

timestamp	fromLabel	toLabel	tokenAddressLabel	fromIsContract	toIsContract
2020-01-27 00:00:00	1	2	111	0	0
2020-01-27 00:01:00	2	3	112	0	0
2020-01-27 00:02:00	3	4	113	0	0
2020-01-27 00:03:00	1	3	111	0	0
2020-01-27 00:04:00	2	4	112	0	0
2020-01-27 00:05:00	3	1	113	0	0



Min-Cut Problem in Graphs

- The Min-Cut problem aims to split a graph into two sets such that the number (or weight) of edges between them is minimized.
- It is a classic NP-hard problem, widely used in clustering and graph partitioning tasks.
- Spectral methods approximate Max-Cut solutions by leveraging eigenvalues of the Laplacian matrix.

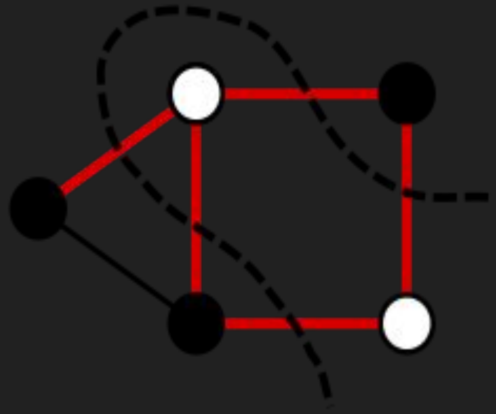


Image Credits :Wikipedia

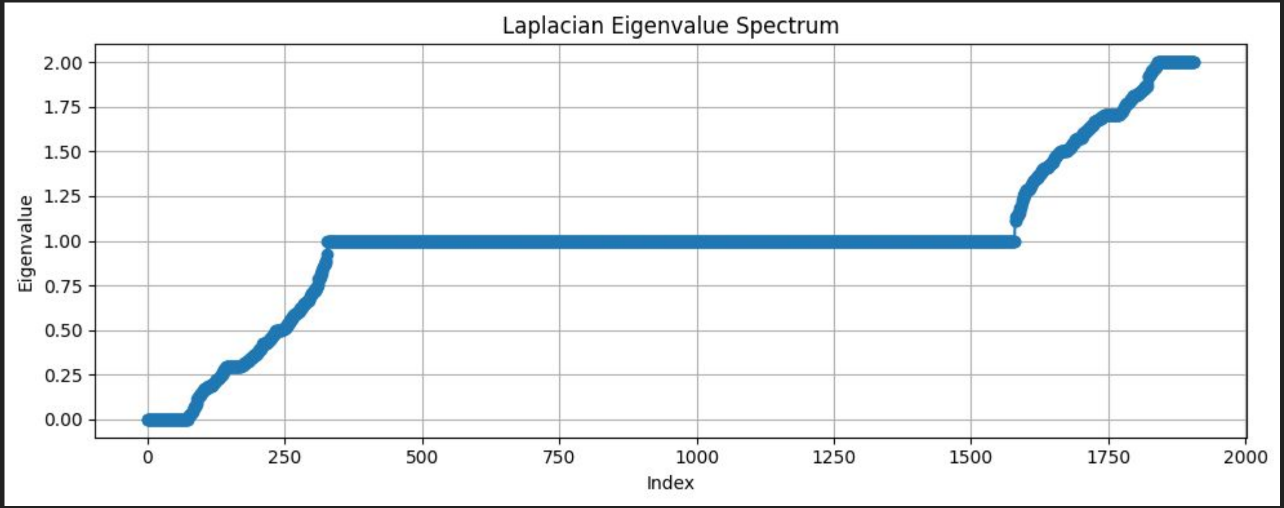
Spectral Clustering Results: Distinct Wallet Groups Found

- Clear clustering observed in low-dimensional spectral space
- High intra-cluster similarity and low inter-cluster connections
- Validated results using silhouette scores and modularity

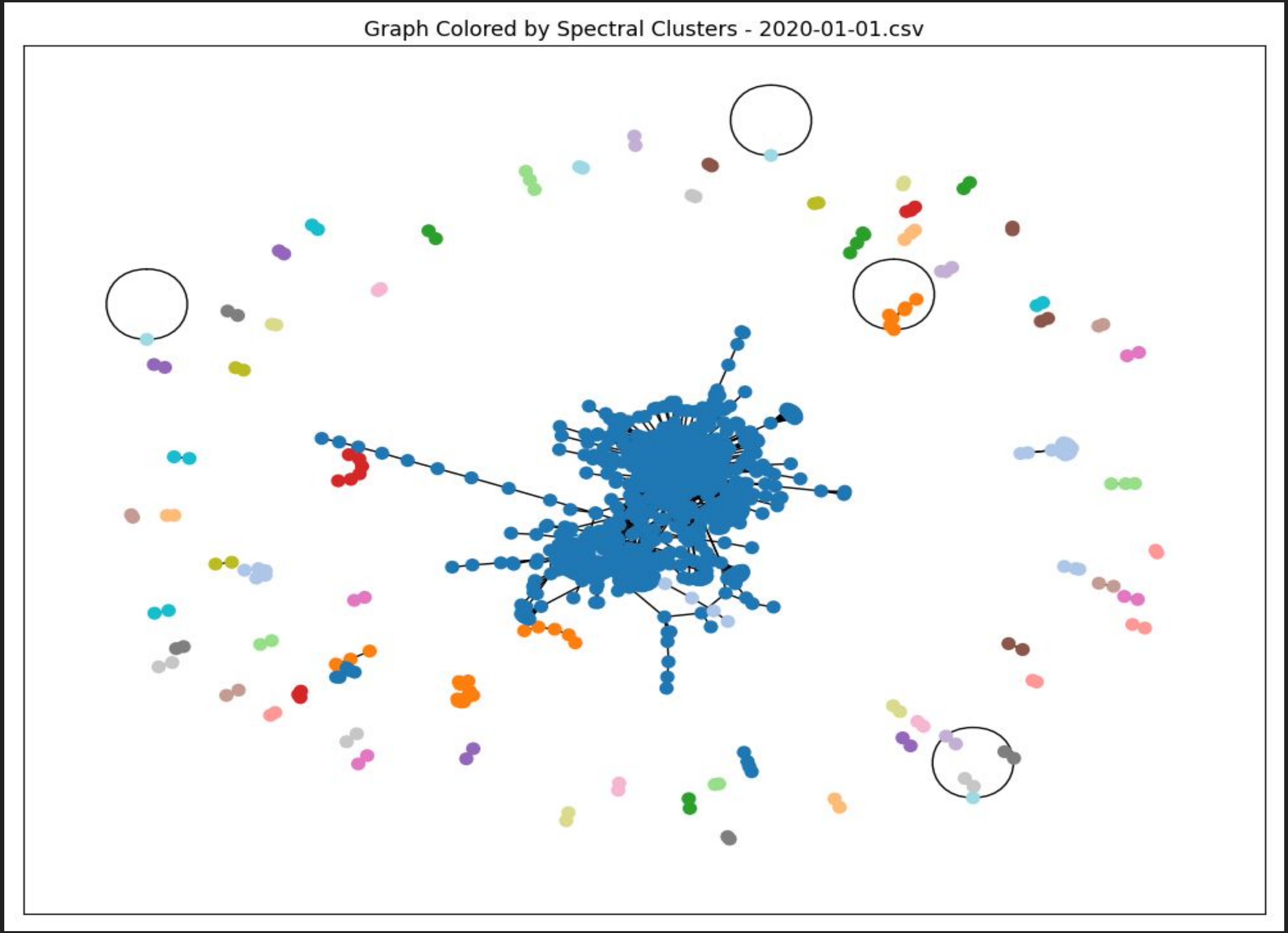
Eg : 1st Jan, 2020

Total Transactions :
18904

Total Unique Wallets:
1904



Clustered Graph



Clustering Result Summary of 1st Jan

Estimated Number of Clusters: 75

Cluster Size Distribution Highlights:

Largest Cluster (Cluster 0): 1683 nodes

Small Clusters (Size ≤ 3): Majority

Clusters with Size ≥ 5:

- Cluster 6 → 22 nodes
- Cluster 1 → 16 nodes
- Cluster 10 → 11 nodes
- Cluster 7, 11 → 7 nodes each
- Cluster 17 → 6 nodes

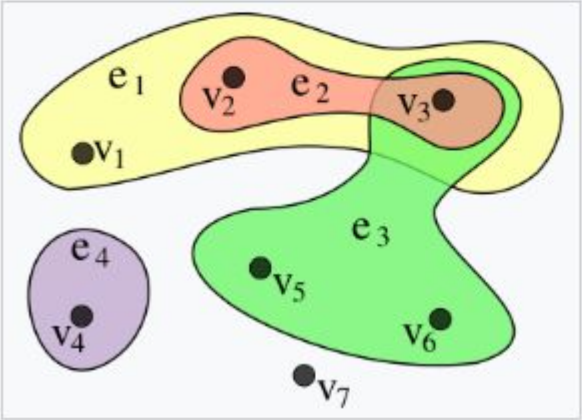
Cluster 4, 8, 2 → 5 nodes each

Observation:

The clustering is highly imbalanced, with one dominant cluster and many tiny clusters, indicating potential community concentration or sparsity in some areas of the graph.

Hypergraphs

A hypergraph is a generalization of an ordinary graph in which an edge can connect more than two vertices together



An example of an undirected hypergraph, with $X = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$ and $E = \{e_1, e_2, e_3, e_4\} = \{\{v_1, v_2, v_3\}, \{v_2, v_3\}, \{v_3, v_5, v_6\}, \{v_4\}\}$. This

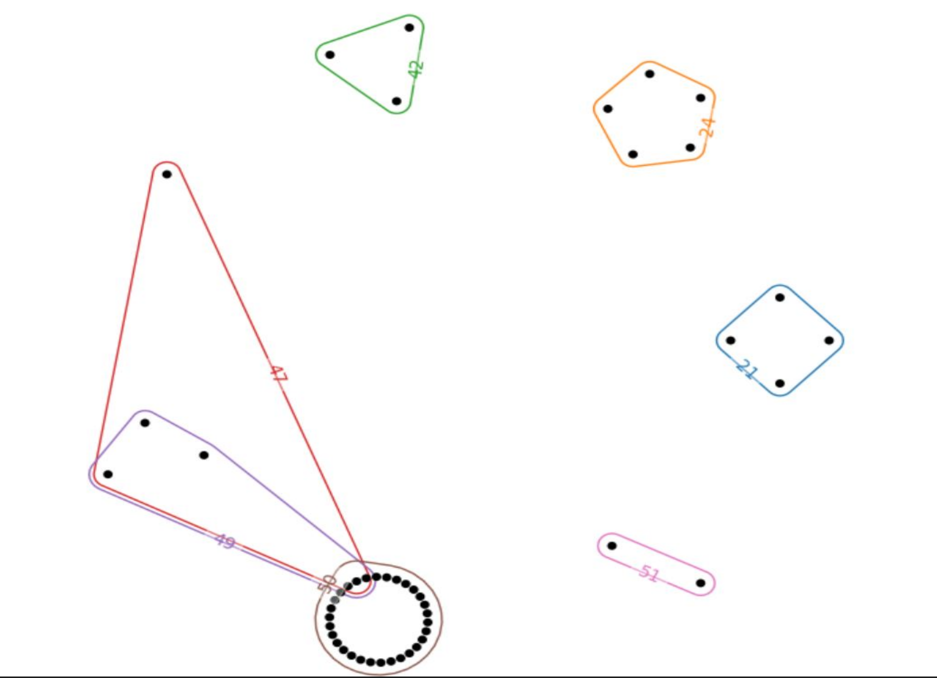
Credits : Wikipedia

Hypergraph Vs Normal Graph

Aspect	Normal Graphs	Hypergraphs
Edge Connectivity	Two vertices per edge	Any number of vertices per hyperedge
Group Interaction Modeling	Requires many pairwise edges	Captures the entire group with one hyperedge
Representation Clarity	Can become cluttered in group settings	More natural and compact for group data
Example Scenario	Friendships, pairwise collaborations	Research groups, social chat groups

Hypergraphs: Capturing Multi-Wallet Token Interactions

- Nodes: wallets
- Hyperedges: token contracts
- Encodes many-to-many relationships missed by simple graphs
- Better suited for understanding NFT community structures



20th November, 2017

Hypergraph Clustering

- Matrix size scales with number of hyper edges × vertices
- Eigen decomposition on incidence matrix more computationally expensive
- Less distinct clusters observed due to overlap in token usage

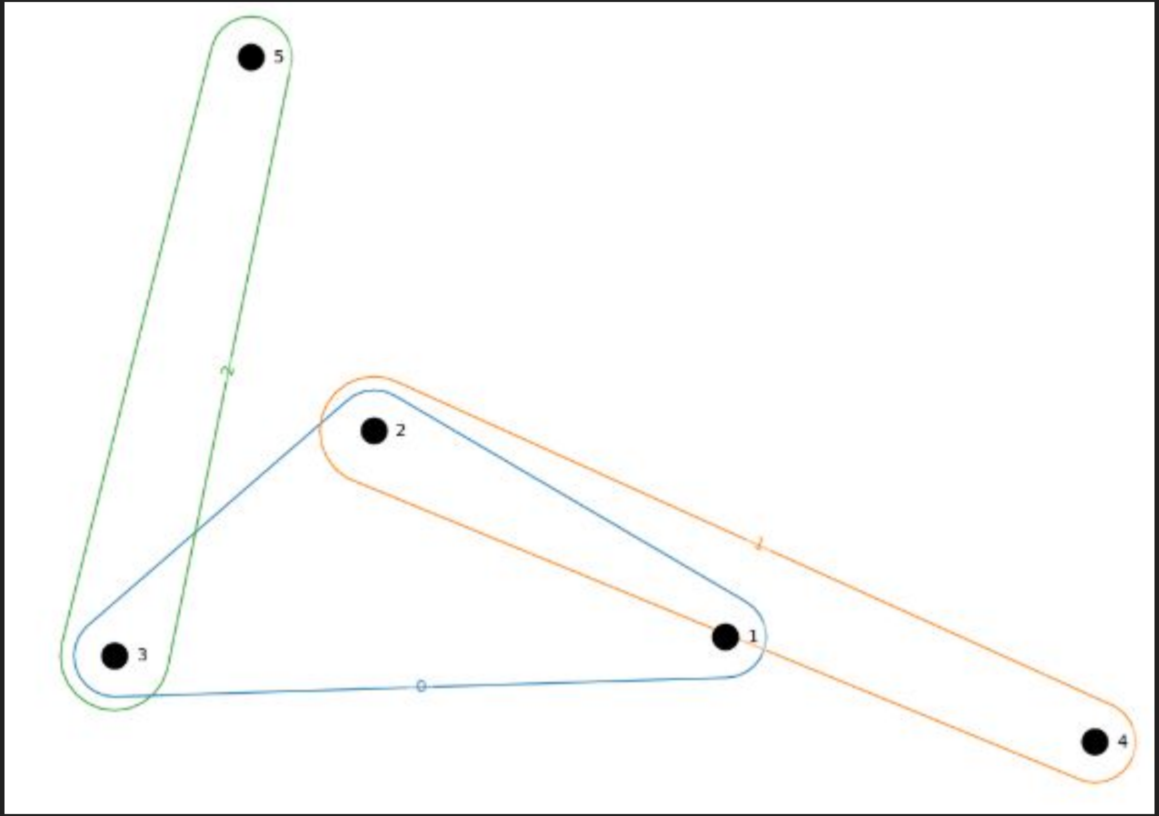
Laplacian Of Hypergraph

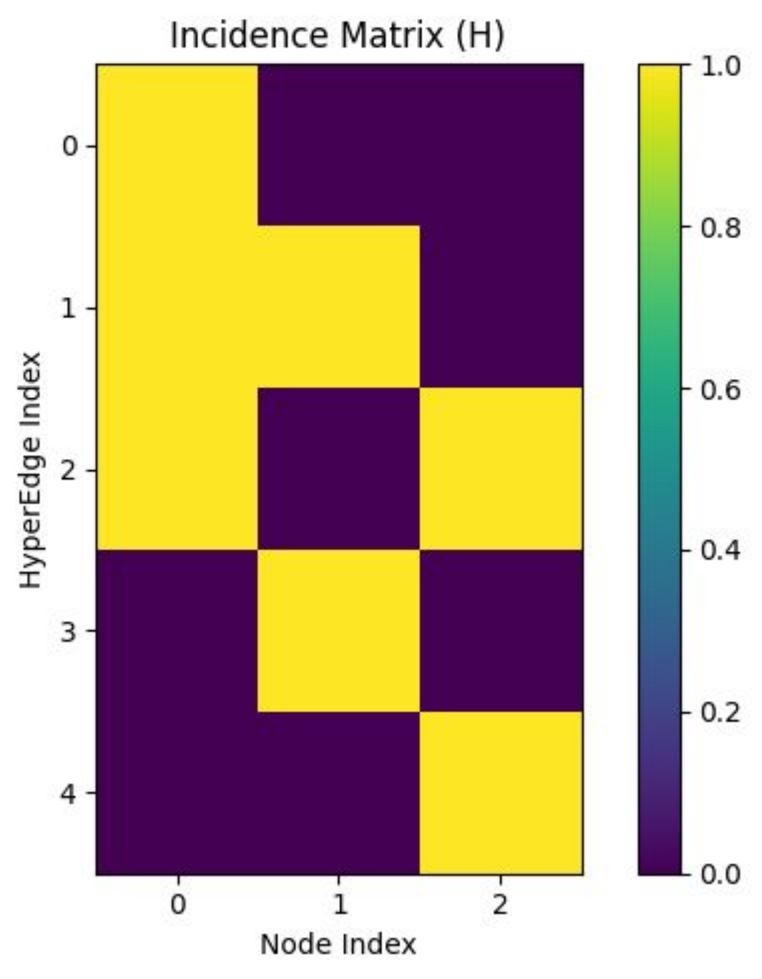
- Dv: Degree matrix of vertices
- De: Degree matrix of hyperedges
- H: Incidence matrix
- W: Weight matrix

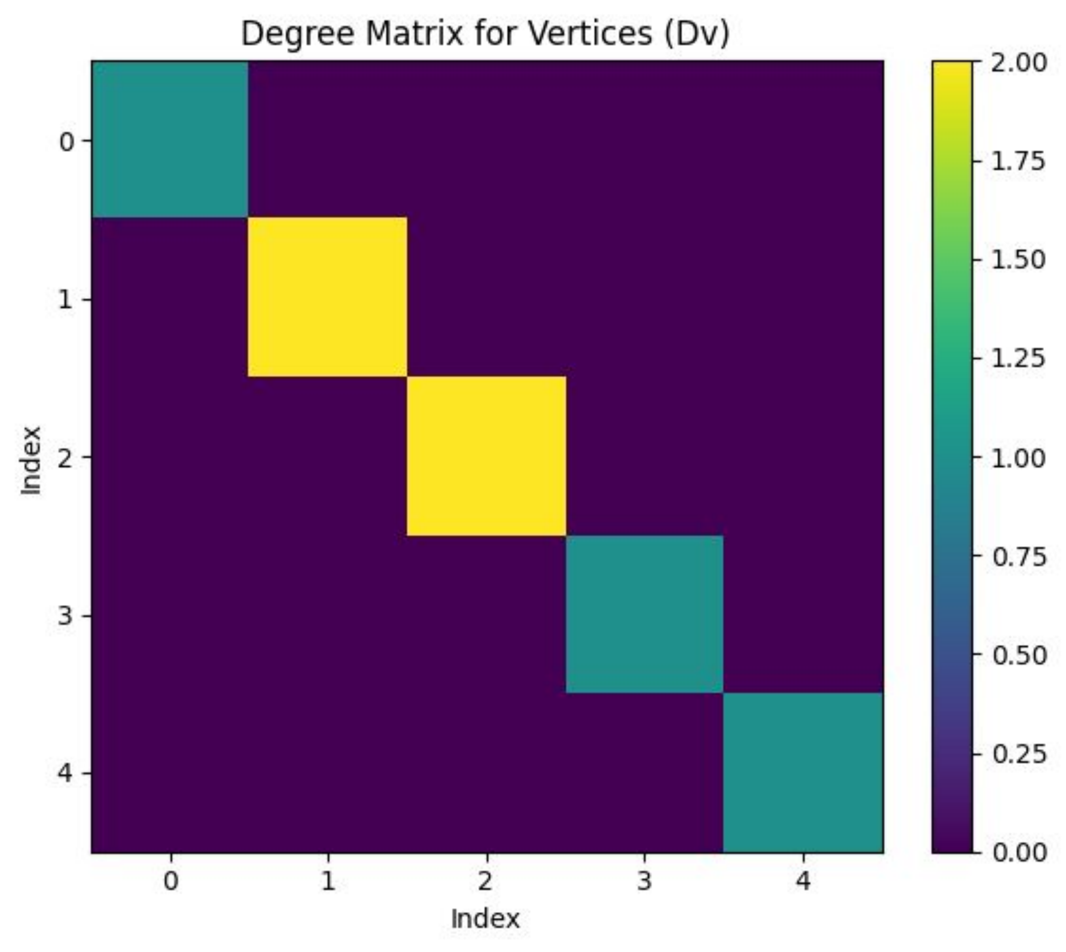
$$\Delta = \mathbf{D}_v - \mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T.$$

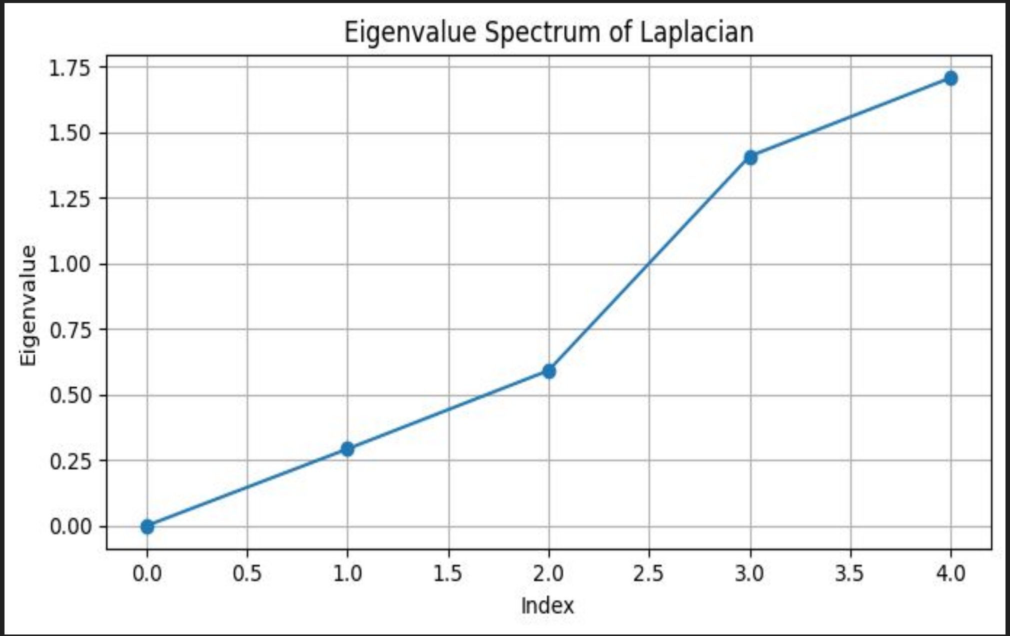
Ref: Gao, Yue, et al. "Hypergraph learning: Methods and practices." IEEE Transactions on Pattern Analysis and Machine Intelligence 44.5 (2020): 2548-2566.

Example : Hypergraph




Incidence Matrix (H)
HyperEdge Index
Node Index
0
1
2
3
4
0
1
2
1.0
0.8
0.6
0.4
0.2
0.0


Degree Matrix for Vertices (Dv)
Index
Index
0
1
2
3
4
0
1
2
3
4
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00

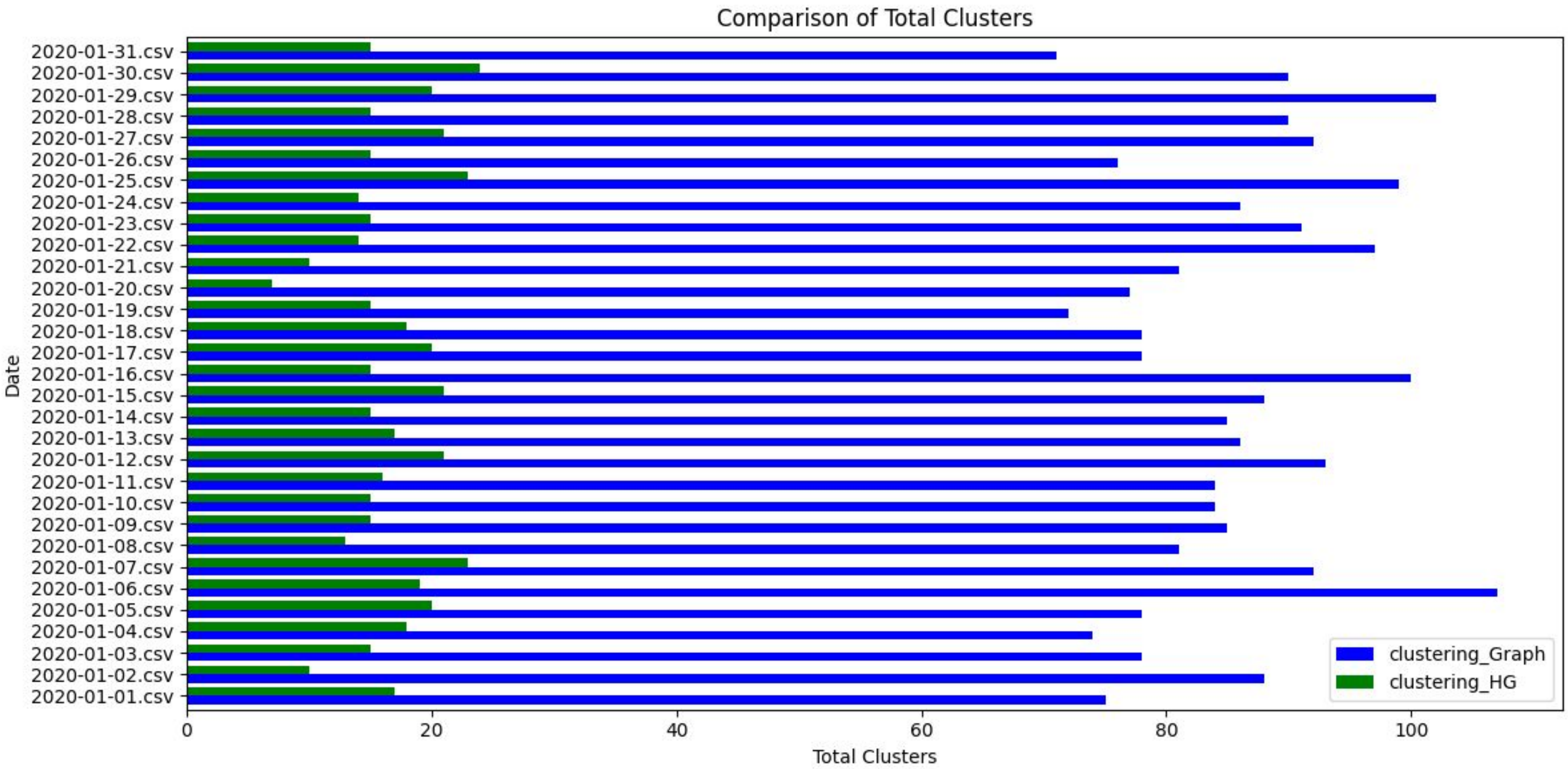



```
Cluster 1: [1]
Cluster 2: [2, 4]
Cluster 0: [3, 5]
```

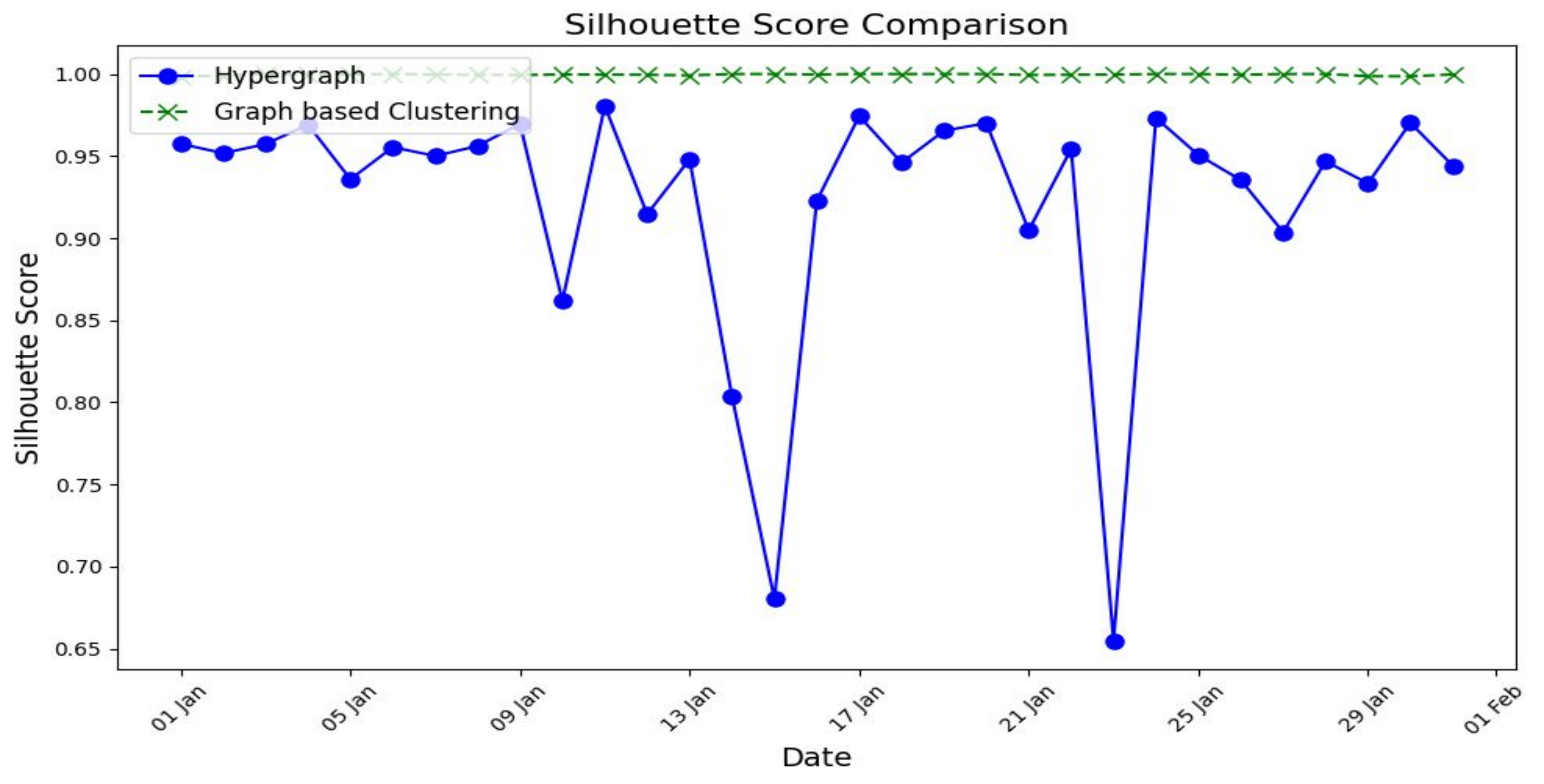
```
Laplacian Matrix (L):
[[ 0.66666667 -0.33333333 -0.33333333  0.          0.        ]
 [-0.33333333  1.16666667 -0.33333333 -0.5         0.        ]
 [-0.33333333 -0.33333333  1.16666667  0.         -0.5       ]
 [ 0.         -0.5         0.          0.5         0.        ]
 [ 0.          0.         -0.5         0.          0.5       ]]
```

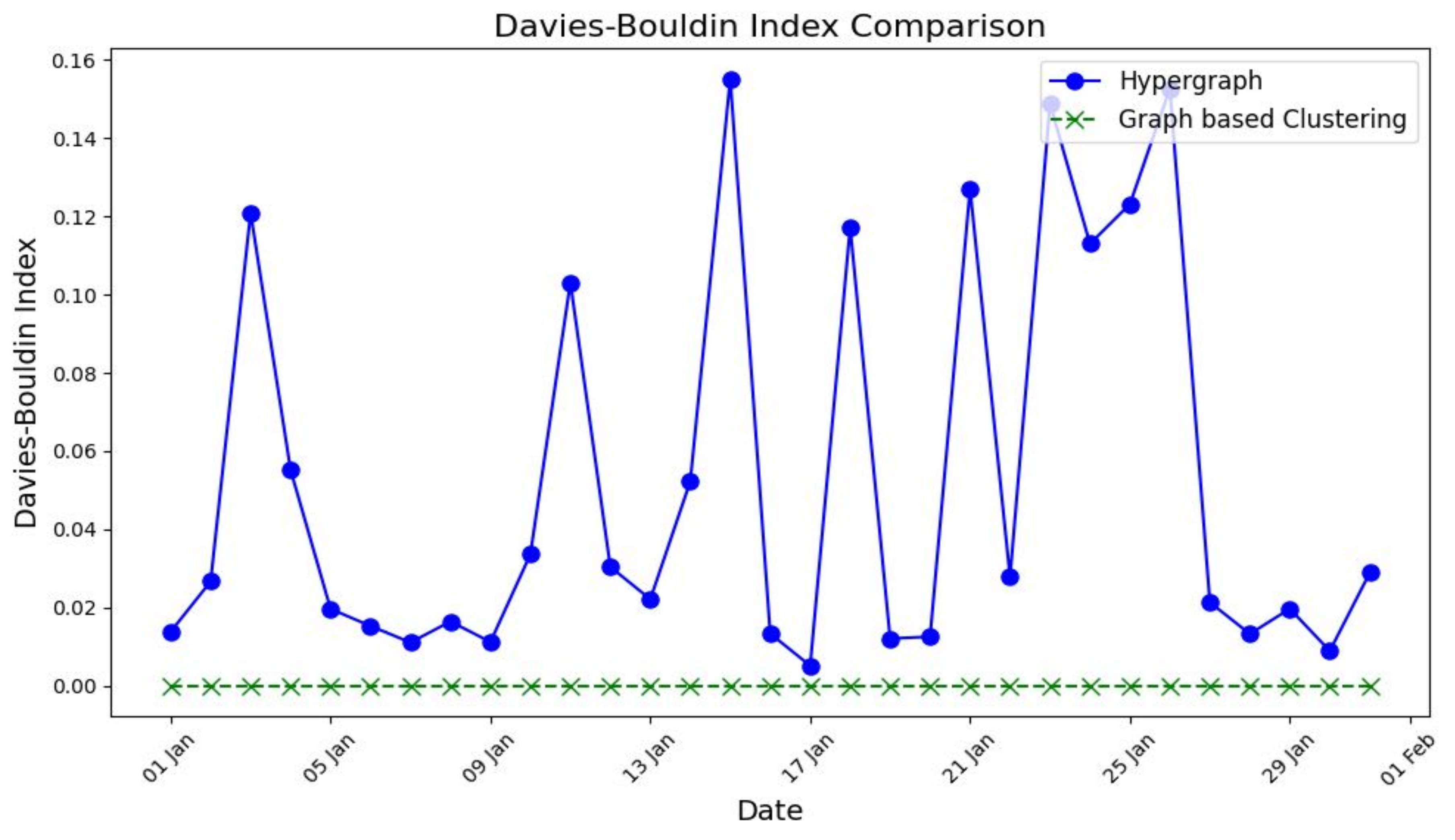
Results

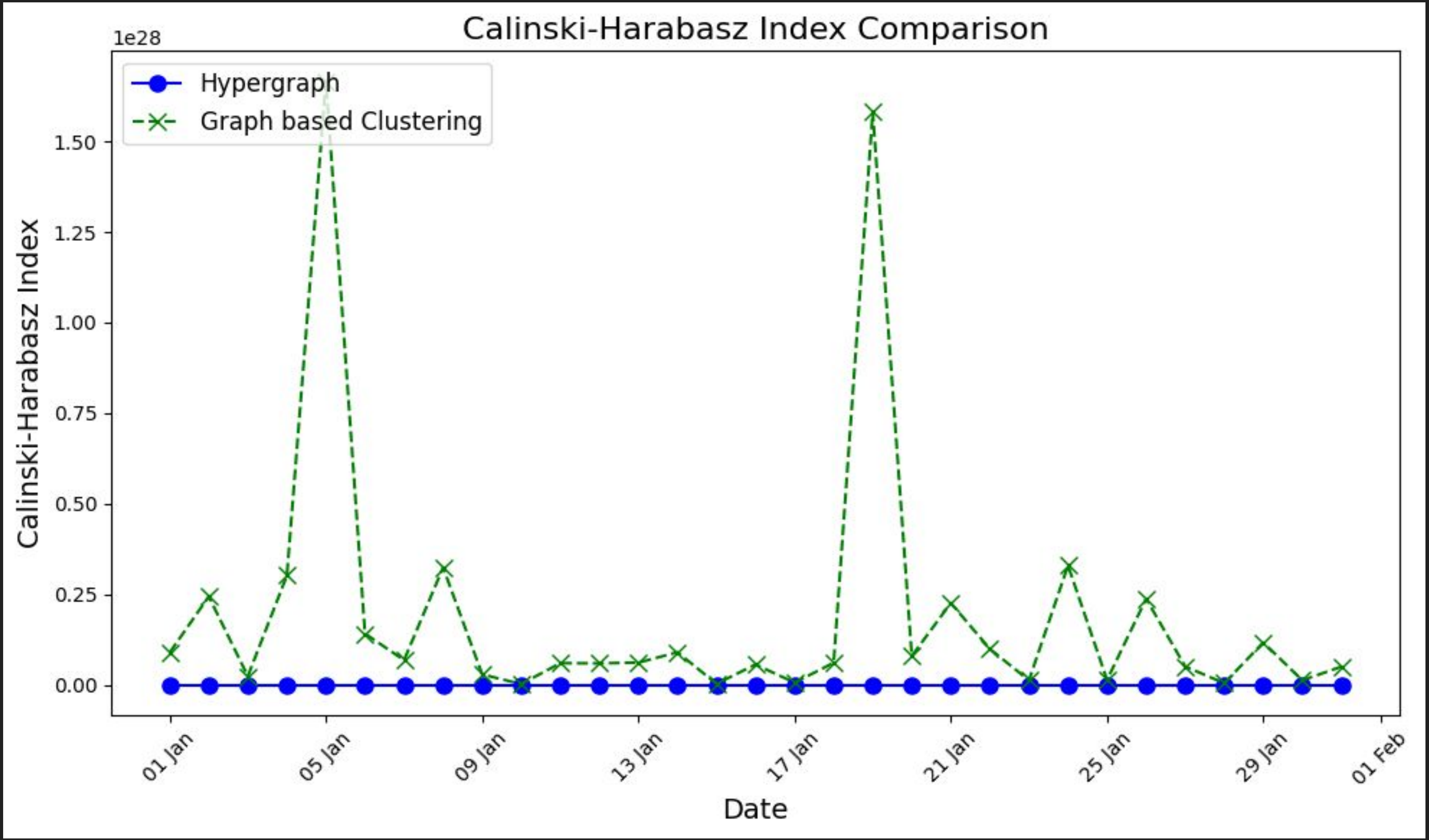

Max Cluster Size Comparison Over Time
Hypergraph
Graph
Max Cluster Size
1500
2000
2500
3000
3500
Date
1st Jan
2nd Jan
3rd Jan
4th Jan
5th Jan
6th Jan
7th Jan
8th Jan
9th Jan
10th Jan
11th Jan
12th Jan
13th Jan
14th Jan
15th Jan
16th Jan
17th Jan
18th Jan
19th Jan
20th Jan
21st Jan
22nd Jan
23rd Jan
24th Jan
25th Jan
26th Jan
27th Jan
28th Jan
29th Jan
30th Jan
31st Jan


Comparison of Total Clusters
Date
2020-01-31.csv
2020-01-30.csv
2020-01-29.csv
2020-01-28.csv
2020-01-27.csv
2020-01-26.csv
2020-01-25.csv
2020-01-24.csv
2020-01-23.csv
2020-01-22.csv
2020-01-21.csv
2020-01-20.csv
2020-01-19.csv
2020-01-18.csv
2020-01-17.csv
2020-01-16.csv
2020-01-15.csv
2020-01-14.csv
2020-01-13.csv
2020-01-12.csv
2020-01-11.csv
2020-01-10.csv
2020-01-09.csv
2020-01-08.csv
2020-01-07.csv
2020-01-06.csv
2020-01-05.csv
2020-01-04.csv
2020-01-03.csv
2020-01-02.csv
2020-01-01.csv
0
20
40
60
80
100
Total Clusters
clustering_Graph
clustering_HG

Comparison of Graph vs Hypergraph Clustering




Davies-Bouldin Index Comparison
Hypergraph
Graph based Clustering
Davies-Bouldin Index
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
01 Jan
05 Jan
09 Jan
13 Jan
17 Jan
21 Jan
25 Jan
29 Jan
01 Feb
Date



Calinski-Harabasz Index Comparison
1e28
Hypergraph
Graph based Clustering
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Calinski-Harabasz Index
01 Jan
05 Jan
09 Jan
13 Jan
17 Jan
21 Jan
25 Jan
29 Jan
01 Feb
Date

Observations Across Graph Types (Normal & Hypergraph)

In both normal and hypergraph clustering:

- One very large cluster dominates
- Many other clusters are extremely small (often just 2–3 wallets)

What This Means !!

- The big cluster likely represents:
  - A large number of wallets interacting with popular NFT contracts
  - Centralized activity around trending collections or marketplaces
  - Possibly an NFT drop, airdrop, or active trading on major platforms

The small clusters reflect:

- Individual trades
- Niche or low-activity wallets
- Potentially bot wallets, private sales, or side projects

Clustering Observations :

- Hypergraphs have multi-node relationships (hyperedges), making clustering more complex.
- Graphs rely on pairwise connections, forming denser and clearer clusters.
- Hyperedges create sparse, higher-order connections, leading to fewer clusters.
- Clustering in hypergraphs is more challenging due to intricate node interactions.

Future Scope: Scaling and Real-time Analytics

- Fraud Detection Using ML Models: Integrate supervised learning on top of clustering to identify anomalous patterns or flag suspicious wallets.
- Time-based Dynamic Clustering: Perform clustering across different time windows to detect evolving communities or changing behavior over time.
- Visual Dashboard: Build a real-time visualization tool to track clusters, token flows, and interactions between wallet communities.

Thank You!

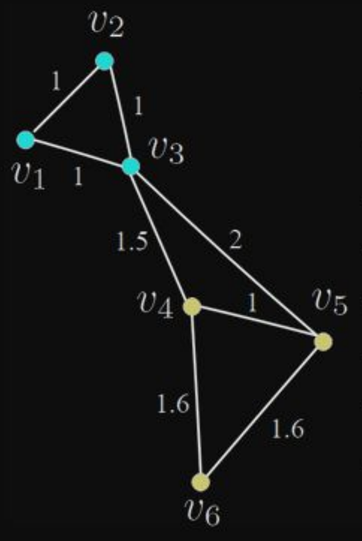
Hypergraph Generation Methods

- Distance-Based Hypergraph Generation
- Representation-Based Hypergraph Generation
- Attribute-Based Hypergraph Generation
- Network-Based Hypergraph Generation

- Graph: Faster, better-defined clusters, scalable
- Hypergraph: Richer data, but high computation and less clean separation
- Visual and statistical comparison shows spectral graph wins here

Distance-Based Hypergraph Generation

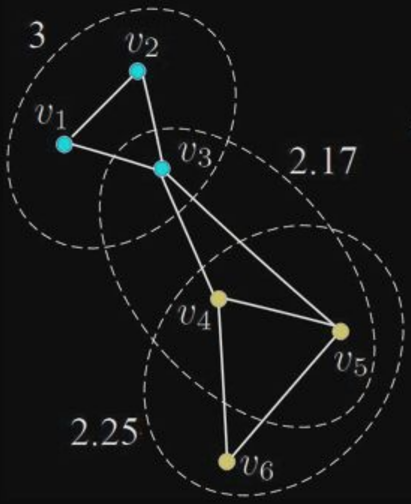
- Nodes grouped based on spatial distance.
- Example: Clustering similar points.
- Pros: Simple, effective for spatial data.
- Cons: Requires a threshold, sensitive to distance metric.



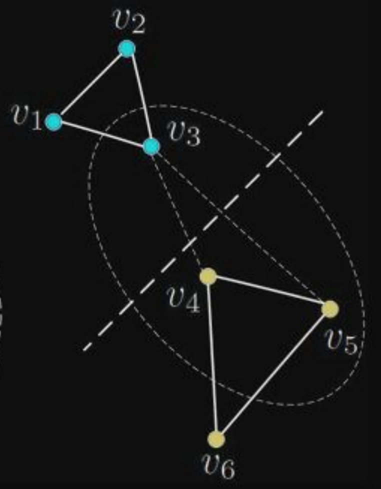
(a)

	e_1	e_2	e_3	e_4	e_5	e_6
v_1	1	1	1	0	0	0
v_2	1	1	1	0	0	0
v_3	1	1	1	1	0	0
v_4	0	0	0	1	1	1
v_5	0	0	0	1	1	1
v_6	0	0	0	0	1	1

(b)



(c)



(d)

Figure 2. (a): A simple graph representing the interrelationships among six points in the 2-D space. Pairwise distances between every vertex and its neighbors are marked on the corresponding edges. (b) The H matrix representing the relationship between the vertices set and the hyperedge sets. The entry (v_i, e_j) is set to 1 if e_j contains v_i, or 0 otherwise. (c): The hypergraph which illustrates the complete relationships in the matrix H. The hyperedge weight is defined as the sum of reciprocals of all the pairwise distances in a hyperedge. (d) A hypergraph partition which is made on e_4.

Credits : Huang, Y., Liu, Q., & Metaxas, D. (2009).]Video object segmentation by hypergraph cut. 2009 IEEE Conference on Computer Vision and Pattern Recognition. doi:10.1109/cvpr.2009.5206795

Representation-Based Hypergraph Generation

- Constructs hyperedges based on shared representations.
- Example: Word embeddings in NLP.
- Pros: Works well with ML models.
- Cons: Needs a pre-trained model, computationally expensive.

How Hyperedges Are Formed?

Compute vector representations of words using an embedding model.Group words that are close in the embedding space into the same hyperedge.

Words with similar meanings (e.g., king, queen, monarch) will belong to the same hyperedge

Attribute-Based Hypergraph Generation

- Groups nodes based on shared attributes.
- Example: Users with common interests in social networks.
- Pros: Intuitive, good for categorical data.
- Cons: Attribute selection matters.

Network-Based Hypergraph Generation

- Extracts hyperedges from existing networks.
- Example: Overlapping groups in citation networks.
- Pros: Captures complex relationships.
- Cons: Requires pre-existing network structures, computationally heavy.

Comparison of Methods

- Distance-Based: Simple, effective for spatial data, but threshold-sensitive.
- Representation-Based: Uses ML models, needs pre-trained embeddings.
- Attribute-Based: Good for categorical data, requires careful attribute selection.
- Network-Based: Captures real-world structures, but computationally expensive.

Credits : Gao, Yue, et al. "Hypergraph learning: Methods and practices." IEEE Transactions on Pattern Analysis and Machine Intelligence 44.5 (2020): 2548-2566.