Clustering-Based Analysis of ERC-721 Blockchain Transactions

Project Report

Prepared by : Piyush Anand (CS21B1019)
Supervisor: Dr. Priodyuti Pradhan

Dept. of Computer Science & Engineering
Indian Institute Of Information Technology Raichur
Transit in Government Engineerign College, Yeramarus Camp, Raichur, Karnataka, 584135

Declaration

I, Piyush Anand (Roll No: CS21B1019), hereby declare that this report entitled "Clustering-Based Analysis of ERC- 721 Blockchain Transactions", submitted to the Indian Institute of Information Technology Raichur towards partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering, is an original work carried out by me under the supervision of Dr. Priodyuti Pradhan and has not formed the basis for the award of any degree or diploma in this or any other institution or university.

I have sincerely tried to uphold academic ethics and honesty. Whenever external information, statements, or results are used, they have been duly acknowledged and cited.

Place: Raichur
Date: April 26, 2025

Piyush Anand

Acknowledgement

I wish to express my sincerest gratitude to Dr. Priodyuti Pradhan for his continuous guidance and mentorship throughout this project. He showed me the path to achieving my goals by clearly explaining each task and emphasizing the significance and industrial relevance of this work. He was always willing to help and clarify any doubts or challenges I faced. Without his constant support and motivation, the successful completion of this project would not have been possible.

Place: Raichur
Date: April 26, 2025

Piyush Anand

Contents

1 Abstract

This report presents a novel approach to analyzing ERC-721 token transactions on the Ethereum blockchain using spectral clustering techniques. By leveraging a comprehensive dataset from X-Block ETH, we aim to uncover the underlying structures in the network of token transfers and wallet interactions. Traditional clustering methods, such as those based on graphs, often struggle to capture the intricate relationships between wallets and tokens. In contrast, spectral clustering offers a more effective method by leveraging the eigenvalues and eigenvectors of the graph Laplacian to partition the data into meaningful clusters. Our approach utilizes the spectral properties of the graph formed by ERC-721 token transactions, providing deeper insights into the community structure of wallets and the evolution of token transfers over time. The results highlight the effectiveness of spectral clustering in identifying distinct patterns and clusters within the Ethereum blockchain, offering a more nuanced understanding of the interactions between wallets and tokens.

Place: Raichur
Date: April 26, 2025

Piyush Anand

2 Introduction

Over the years, networks have been extensively investigated and have successfully provided insights into the structure and dynamics of physical [1], biological [2, 3, 5, 4], social [6, 7] and several other natural systems [8].

The growing complexity of networks has led researchers to analyze optimal structures and strategies for studying the diverse structural properties and dynamic behaviors of complex networks. Prominent examples relevant to this analysis are interconnected networks represented by multilayer networks [9, 10] and higher-order interactions [11, 12, 13, 14], which are investigated using hypergraphs and simplicial complexes.

Higher-order interactions represent the connections among two or more than two nodes simultaneously in complex systems. Such interactions can be modeled as hyperedges analogous to edges in conventional graphs or simplices in simplicial complexes. Simplicial complex is a specific type of hypergraph with an inclusion property: For a triadic interaction (i, j, k) to exist, the corresponding dyadic interactions (i, j), (i, k) and (j, k) must also be present. In this article, we make use of hypergraphs for our analysis.

Various fields have witnessed the existence of higher-order interactions such as the human brain [15], collaboration networks [17], species interactions [16], cellular networks [18], and drug combinations [19]. Notably, in several real-world networks, presence of higher-order interactions has been found to give new insights, for example, in [20], authors developed a novel method called high-order

functional connectivity, which captures interactions among three or more brain regions across various spatiotemporal scales. This method was applied to detailed EEG and fMRI characterizations of neurodegenerative conditions.

Faes et al. [21] introduced the O-information rate (OIR) metric to evaluate higher-order interactions in multivariate time series. This approach was applied to physiological networks, such as heart period, respiratory variability, and arterial pressure in healthy subjects, as well as brain networks described by electrocorticographic signals in animal experiments during anesthesia. Moreover, a study proposed STHAN-SR, a neural hypergraph architecture for stock selection that models complex relationships between stocks using a hypergraph and temporal Hawkes attention mechanism, resulting in a new spatiotemporal attention [22]. In [23], authors conducted neuronal studies revealing that while pair-wise interactions between neurons explain population responses in sensory areas, higher-order interactions are necessary to explain responses in executive areas involved in decision-making and actions.

Santoro et al. [24] analyzed multivariate time series to uncover higher-order patterns in brain functional activity, epidemics, and financial markets. Additionally, a multivariate signal processing technique was developed to construct higher-order networks from time series, which was applied to resting fMRI signals to identify higher-order communications between different brain regions.

Also, these group interactions give rise to new collective phenomena, including synchronization [25, 26], contagion dynamics [27, 28], diffusion processes [29], and evolutionary games [30, 31]. For example, in the evolutionary dynamics of public good games on hypergraph, it is found that overlap between hyperedges tend to foster cooperative outcomes and the heterogeneity of hyperedges promotes cooperation [32]. Burgio et. al [33] studied the susceptible-infectious-susceptible (SIS) process on a hypergraph and demonstrated that overlap between three body interactions lowers the critical point and

produces smaller spreads.

Cencetti et al. examined the formation and evaluation of higher-order organizations in temporal networks by studying five social network datasets from various social contexts [34].

3 Pre-requisites

3.1 Blockchain

A blockchain is a decentralized ledger that involves blocks of data, usually with several transactions, a date, the previous block's hash, and a nonce for hash verification. Because blocks are encrypted, it means they are linked through their unique and unalterable hash values, thus ensuring integrity. A consensus mechanism validates transactions through a network, where nodes agree on whether the block is valid before adding it to the chain [35]. This process secures the blockchain, preventing any form of tampering, which enables trust in peer-to-peer asset transfers, as seen in Bitcoin, whereby miners validate blocks and are rewarded with cryptocurrency [36].

3.2 Ethereum

Ethereum, proposed in Vitalik Buterin's paper [37], addresses several limitations of Bitcoin's scripting language. Its chief contributions include full Turing-completeness, supporting all kinds of computations, including loops. More importantly, it improves the structure of a blockchain by allowing for transaction states and other such improvements. Abstractly, it provides an additional layer where users can define customized rules for ownership and transaction formats and also state transition functions via smart contracts-cryptographic rules executed when certain conditions are met [38].

3.3 Non Fungible Token

Non Fungible Token (NFTs) are digital tokens issued on the Ethereum blockchain, similar to Bitcoin, in that they're "minted" and sold. But unlike Bitcoin, which is "fungible," that is, indistinguishable from another bitcoin, NFTs are "non-fungible," that is each token unique. This uniqueness assigns a property right over digital assets, for instance Beeple's artwork. Anyone can view Beeple's "Everydays—The First 5,000 Days"; only the owner of the associated NFT can make an ownership claim.

This dynamic introduces exclusivity into digital art. The NFT is recorded on the immutable Ethereum blockchain after being minted, which proves ownership of the art. While digital art can be looked at, copied, and shared, NFTs ensure that ownership cannot be faked, enabling true exclusive ownership of digital art, a phenomenon previously unthinkable [39].

3.4 ERC-721

The ERC-721 Non-Fungible Token (NFT) Standard [40] proposes a structure to represent one-of-a-kind tokens that are not interchangeable within the Ethereum blockchain. In the case of ERC-20 tokens, their uniqueness and interchangeability come to an end, ERC-721 on the other hand has no crops and it plays the role of managing unique asset properties. They afford atomic management for ownership of each unique digital asset, thus finding usefulness with 'one of a kind' digital objects such as NFTs, games, or intellectual property. By default, the standard states that the American Public policy has ten basic functions and three events which should be observed.

ERC-721 standard tokens were created and distributed via smart contracts and placed on the Ethereum network. The standard is free and open by nature, which means that it can be leveraged when creating non-fungible tokens that are aimed at specific digital as-

sets/objects, which can be owned individually and have historical backgrounds. An application of ERC-721 tokens is CryptoKitties which are internet pets available for auction. Thirdly, the ERC 721 standard lays out basic provisions for creating, issuing, managing, transferring and trading of these specific types of tokens, thus introducing new digital forms of scarcity and asset representation [41].

3.5 Smart and Non-Smart Contracts

In digital systems, transactions and agreements between parties can be handled either manually or through automated processes. These two approaches are broadly categorized as non-smart contracts and smart contracts, respectively.

3.5.1 Non-Smart Contracts

Non-smart contracts refer to traditional, manual methods of executing transactions or agreements. In such systems, human intervention is required at each step to initiate, authorize, and complete the process. The actions depend entirely on the users' decisions and manual inputs. These contracts lack automation and require the parties involved to actively participate in every stage of execution. This approach is similar to how conventional payments or physical agreements work — where the outcome is dependent on explicit user approval and execution.

3.5.2 Smart Contracts

Smart contracts, on the other hand, are self-executing digital contracts where the terms of the agreement are written in code and automatically enforced by the system. Once deployed, a smart contract runs on a blockchain and performs predefined actions when certain conditions are met — without needing manual approval each time. This brings automation, trust, and efficiency into the transac-

tion process. Smart contracts reduce the need for intermediaries and minimize the chances of human error or intentional manipulation.

3.5.3 Key Differences

Automation

Smart contracts operate automatically, while non-smart contracts require manual intervention.

Trust

Smart contracts are executed based on code and cannot be altered once deployed, offering higher reliability.

Efficiency

Smart contracts execute instantly once conditions are met, whereas non-smart contracts may be delayed due to human dependency.

3.6 Spectral Clustering

Spectral clustering is a graph-based technique that identifies community structures or tightly connected groups in complex networks by leveraging the spectral (eigenvalue) properties of the graph Laplacian matrix. It is particularly effective in handling data that is non-linearly separable or exhibits manifold structures.

3.6.1 Unnormalized Spectral Clustering

Given an undirected graph $G = (V, E)$, we define: - $A \in \mathbb{R}^{n \times n}$: the adjacency matrix, where $A_{ij} = 1$ if there is an edge between nodes i and j. - $D \in \mathbb{R}^{n \times n}$: the degree matrix, a diagonal matrix where D_{ii} is the degree of node i.

The unnormalized graph Laplacian is defined as:

$$L = D - A$$

The eigenvectors corresponding to the smallest eigenvalues of L are used to embed the nodes into a lower-dimensional space. K-Means clustering is then applied to this embedding. This approach works well when the clusters are well-separated and balanced in size. However, it can be sensitive to variations in node degree and less effective for graphs with large degree disparities.

3.6.2 Normalized Spectral Clustering

To address the limitations of unnormalized Laplacian, normalized versions are used. There are two main forms:

- Symmetric Normalized Laplacian:

$$L_{\text{sym}} = I - D^{-1/2} A D^{-1/2}$$

- Random Walk Normalized Laplacian:

$$L_{\text{rw}} = I - D^{-1} A$$

In this project, the symmetric normalized Laplacian was used. Its spectral decomposition provides eigenvectors that are then row-normalized and fed into K-Means. This normalization makes the method more robust to imbalances in node degree and results in better separation when clusters have different densities.

3.6.3 Why Both Are Considered

The unnormalized version provides intuitive and simple Laplacian behavior, and is computationally lighter in some cases. However, the normalized version is more stable and suitable for real-world data where node degrees vary significantly — such as transaction graphs in blockchain datasets.

Normalized spectral clustering performs better in: - Unevenly sized clusters - Networks with hub nodes or varying node degrees - Situations where balanced partitioning is desired

Feature	**Unnormalized** $L = D - A$	**Normalized** $L = I - D^{-1/2}AD^{-1/2}$
Degree Sensitivity	High	Low
Balanced Clustering	May fail	Better performance
Ease of Computation	Simpler	Slightly more complex
Stability	Less robust	More robust
Preferred When	Degrees are similar	Degrees vary significantly

Table 3.1: Comparison of Unnormalized vs Normalized Spectral Clustering

3.6.4 Comparison Table

3.6.5 Application in This Project

The normalized spectral clustering method was chosen for the majority of experiments due to the presence of hubs and uneven node degrees in blockchain transaction graphs. These graphs tend to have certain wallets that interact with a large number of other wallets (e.g., exchanges or smart contracts), and normalization ensures that clustering results are not biased by such highly connected nodes.

3.7 Hypergraph

A hypergraph is a generalization of a regular graph. While in a standard graph, each edge connects exactly two nodes, in a hypergraph, an edge can connect any number of nodes — not just two. Hypergraphs are useful in scenarios where relationships involve more than two elements.

H denoted by $H = (V; E = (e_i)_{i \in I})$ on a finite set V is a family $(e_i)_{i \in I}$, where I is a finite set of indexes, of subsets of V called hyperedges. Sometimes V is denoted by $V(H)$ and E by $E(H)$ [42].

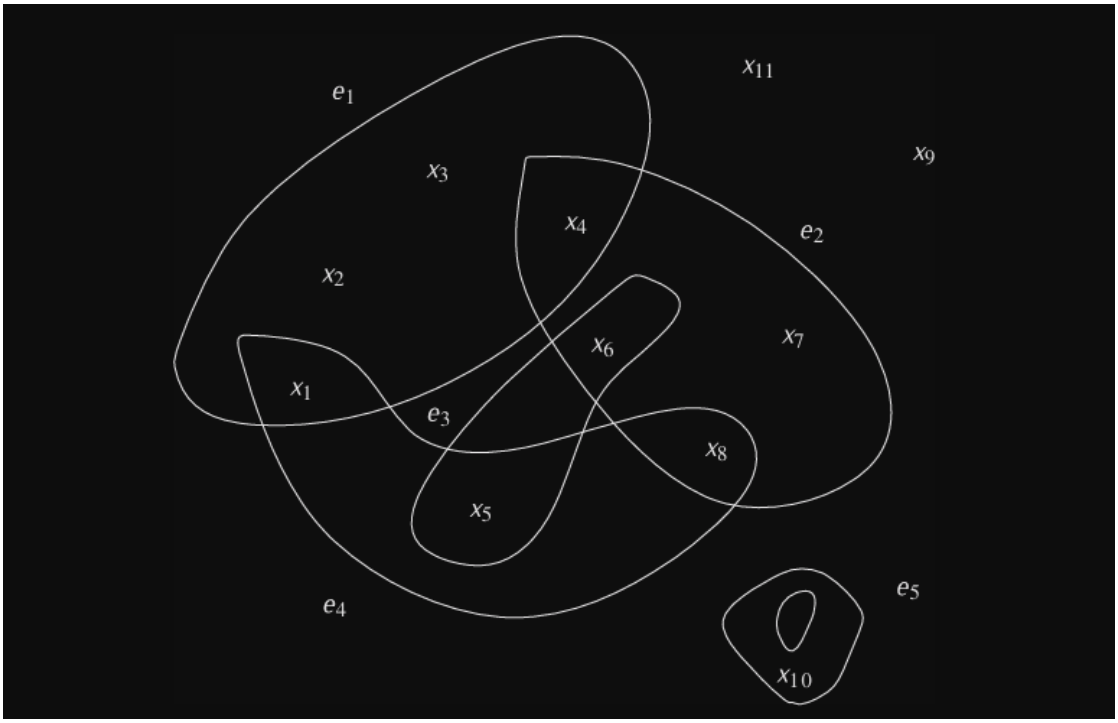

Figure 3.1: The hypergraph H above has 11 vertices, 5 hyperedges e_5, and 2 isolated vertices: x_{11}, x_9. The degree of x_1 is 2 [42].

4 ERC721 Token Transaction Data and Network Modeling

4.1 Transaction Data sets - XBLOCK-ETH [43]

The dataset comprises detailed records of ERC-721 token transactions, which are the foundation for non-fungible tokens (NFTs) on the Ethereum blockchain. Each entry within the dataset provides essential information, including the block number (which indicates where the transaction is recorded on the blockchain), a timestamp (showing when the transaction took place), a unique transaction hash (serving as an identifier), and the addresses of both the sender and recipient.

Additionally, the dataset specifies whether these addresses belong to smart contracts or externally owned accounts, adding another layer of context to the transactions. Spanning from 2015 to 2024, this dataset offers a rich temporal range, enabling analysis of trends and developments in the NFT market over nearly a decade.

4.2 Data Preprocessing

Our analysis began with the preprocessing of the X-Block ETH dataset.

4.2.1 Data Downloading

The initial step in the data preparation procedure was sourcing the needed transaction data found on repository X-Block ETH that contained records of Ethereum transactions. The information contained several key features such as the transaction timing, a respective block to the transaction, the addresses involved and the type of token.

blockNumber	timestamp	transactionHash	tokenAddress	from	to	fromIsContract	toIsContract	tokenId
1001165	1455424860	0x1d11b3ea5593e	0x55b9a11c2e835	0x38150290c18	0x3d2068aeb96	0	0	200000000
1001165	1455424860	0x1d11b3ea5593e	0x55b9a11c2e835	0x38150290c18	0x5d2c24efac49	0	1	0
1003181	1455460299	0x26a30aa663f6e1	0x55b9a11c2e835	0x38150290c18	0x3d2068aeb96	0	0	500000000
1003181	1455460299	0x26a30aa663f6e1	0x55b9a11c2e835	0x38150290c18	0x5d2c24efac49	0	1	0
1003393	1455463847	0xc0db923ac0c6d7	0x55b9a11c2e835	0x3d2068aeb96	0xb51446cc429	0	0	100000000
1003393	1455463847	0xc0db923ac0c6d7	0x55b9a11c2e835	0x3d2068aeb96	0x5d2c24efac49	0	1	130000
1005878	1455505636	0x113de15964f857	0x55b9a11c2e835	0x38150290c18	0xc0cfc0969d0d	0	0	1000000000
1005878	1455505636	0x113de15964f857	0x55b9a11c2e835	0x38150290c18	0x5d2c24efac49	0	1	0
1005946	1455506683	0x09581f238af3eb	0x55b9a11c2e835	0xc0cfc0969d0d	0xa220568ace9	0	0	100000000
1005946	1455506683	0x09581f238af3eb	0x55b9a11c2e835	0xc0cfc0969d0d	0x5d2c24efac49	0	1	130000
1006034	1455508426	0x637e4263f7825d	0x55b9a11c2e835	0x38150290c18	0xa220568ace9	0	0	200000000
1006034	1455508426	0x637e4263f7825d	0x55b9a11c2e835	0x38150290c18	0x5d2c24efac49	0	1	0
1006046	1455508554	0x5eac9cbd9cae1b	0x55b9a11c2e835	0xc0cfc0969d0d	0xa220568ace9	0	0	898000000
1006046	1455508554	0x5eac9cbd9cae1b	0x55b9a11c2e835	0xc0cfc0969d0d	0x5d2c24efac49	0	1	1167400
1006059	1455508793	0xc71da93bc66434	0x55b9a11c2e835	0x38150290c18	0xa220568ace9	0	0	4000000000
1006059	1455508793	0xc71da93bc66434	0x55b9a11c2e835	0x38150290c18	0x5d2c24efac49	0	1	0
1007626	1455536210	0x6c6729b405473e	0x55b9a11c2e835	0x38150290c18	0xad917335252	0	0	200000000
1007626	1455536210	0x6c6729b405473e	0x55b9a11c2e835	0x38150290c18	0x5d2c24efac49	0	1	0
1007645	1455536477	0x0a67fa2e1500f4	0x55b9a11c2e835	0x38150290c18	0xad917335252	0	0	1800000000
1007645	1455536477	0x0a67fa2e1500f4	0x55b9a11c2e835	0x38150290c18	0x5d2c24efac49	0	1	0
1007784	1455539020	0x8d55d21c2e77a8	0x55b9a11c2e835	0x38150290c18	0xad917335252	0	0	8000000000
1007784	1455539020	0x8d55d21c2e77a8	0x55b9a11c2e835	0x38150290c18	0x5d2c24efac49	0	1	0
1008013	1455542892	0xb3a5580191a1c5	0x55b9a11c2e835	0x9ac937835f6a	0x5d2c24efac49	0	1	411
1008013	1455542892	0xb3a5580191a1c5	0x55b9a11c2e835	0x38150290c18	0x9ac937835f6a	0	0	1000000000
1008013	1455542892	0xb3a5580191a1c5	0x55b9a11c2e835	0x38150290c18	0x5d2c24efac49	0	1	0

Figure 4.1: Downloaded Data From XBlock-Eth

4.2.2 Unix to Human Readable Timestamp

One of the earliest activities in the preprocessing process was to manage the timestamps which varied among other formats supposedly stored in Unix epoch. In order to make the reader's work easier, these timestamps were formatted in an easier to comprehend (YY-MM-DD) thereby ensuring that the information in the dataset would be well understood not only by humans but also by the algorithms which process it.

4.2.3 Dividing Data into Daywise Files

On completion of the conversion of currencies, the data set was arranged in the form of day wise files. This was done to limit the analysis of the transactions to the specific dates, making it easier to analyze the transactional trends and patterns on a daily basis. All these daywise files were also dated meaning that every single file was clearly marked with the date so that the information was organized properly and could be retrieved easily for more work.

Each daywise file contained a few key columns which were designed to capture important details about the transactions. More specifically, these included the date and time stamp of each transaction, the block in which the transaction's records were added, the respective addresses of the different parties in the transaction (from, to), the address of the token which typified the transfer, and whether or not the addresses that transacted were smart contracts. These flags were of utmost importance as they explained how different addresses operate within their blockchain network.

The last two columns-venturing information on which address was or was not a smart contract remained constant over the course of the processing.

4.2.4 Address Labelling

The 64-bit numerical addresses in the dataset presented a challenge that needed to be addressed in a special way. To further enhance the understanding and make the dataset more interpretable, these numerical addresses were converted into appropriate labels with a junction of a dictionary. As such, it aided to comprehend how the addresses were interconnected in addition to creating a rough outline of the distributions or outliers observed in the changes of the data i.e the mentioned occurrences of certain transfers of tokens to particular wallets or smart contracts from the available ones.

blockNumber	timestamp	transactionHash	tokenAddress	from	to	fromIsContract	toIsContract	tokenId
5918558	2018-07-07 0:01	0xeb8b537664ad398	0x06012c8cf97bead5	0x4fabda075e15e9	0x0acb5378a71	0	0	824413
5918560	2018-07-07 0:01	0x8ac506f81242adcc	0x06012c8cf97bead5	0x9eec7e6391071	0xfa580941ec47	0	0	556396
5918567	2018-07-07 0:03	0xc37b4a2b5c0a979	0x7fdcd2a1e52f10c2	0x1e74456cc75e4	0x2842a67ce34	0	1	3929
5918570	2018-07-07 0:04	0x2a1cf54f8a812cd3	0x06012c8cf97bead5	0xc7af99fe5513eb	0x4fabda075e15	1	0	825908
5918581	2018-07-07 0:05	0xee2280d9243df7e2	0x06012c8cf97bead5	0x4fabda075e15e9	0xb1690c08e21	0	1	831457
5918583	2018-07-07 0:06	0x2deb5153f579991	0x7fdcd2a1e52f10c2	0x54133617ad48a	0x92cace836fa2	0	1	3449
5918584	2018-07-07 0:06	0x1a4f17ffa7394388	0x06012c8cf97bead5	0x0000000000000	0x442dccee684	0	0	833692
5918584	2018-07-07 0:06	0x2001c1e251ee963	0x06012c8cf97bead5	0x0000000000000	0x68b42e44079	0	0	833693
5918584	2018-07-07 0:06	0x803b1644d2d38f8	0x06012c8cf97bead5	0x0000000000000	0x7891f796a5d4	0	0	833694
5918584	2018-07-07 0:06	0x5555e5bc64204a3	0x06012c8cf97bead5	0x0000000000000	0x22d1a32a0be	0	0	833695
5918584	2018-07-07 0:06	0x67a495b4f359d78	0x06012c8cf97bead5	0x0000000000000	0x820c0557307	0	0	833696
5918584	2018-07-07 0:06	0x1a001ce852f4dc8	0x06012c8cf97bead5	0x0000000000000	0xc7a024df3876	0	0	833697
5918584	2018-07-07 0:06	0x2023c3fa1d92e7f6	0x06012c8cf97bead5	0x0000000000000	0x06012c8cf97b	0	1	833698
5918584	2018-07-07 0:06	0x2023c3fa1d92e7f6	0x06012c8cf97bead5	0x06012c8cf97bea	0xb1690c08e21	1	1	833698
5918586	2018-07-07 0:06	0xa225f3cef80c37c2	0x06012c8cf97bead5	0x4fabda075e15e9	0xb1690c08e21	0	1	830720
5918587	2018-07-07 0:07	0x9c991844105e681	0x06012c8cf97bead5	0x4fabda075e15e9	0xb1690c08e21	0	1	830716
5918587	2018-07-07 0:07	0xaf2aeb1b040cb0fc	0x06012c8cf97bead5	0xb1690c08e213a	0x5be7d57e9b1	1	0	833072
5918591	2018-07-07 0:08	0x21aa77639430734	0x06012c8cf97bead5	0x4fabda075e15e9	0xb1690c08e21	0	1	829836
5918592	2018-07-07 0:08	0x507e9c6f5150c1a2	0x1a94fce7ef36bc90	0x0000000000000	0x2e4ee826597	0	0	9521
5918594	2018-07-07 0:08	0xef0637db5d13a01	0x06012c8cf97bead5	0x4fabda075e15e9	0xb1690c08e21	0	1	828948
5918599	2018-07-07 0:10	0xb129c21b597642d	0x06012c8cf97bead5	0x4fabda075e15e9	0xb1690c08e21	0	1	828570
5918600	2018-07-07 0:10	0x746a38f26b80be8	0x06012c8cf97bead5	0x4fabda075e15e9	0xb1690c08e21	0	1	828196
5918603	2018-07-07 0:10	0x5b1d6fc19252734	0x7fdcd2a1e52f10c2	0x5ab10735fd42ec	0x2842a67ce34	0	1	4065
5918604	2018-07-07 0:10	0x2a2738c45a5fc80	0x06012c8cf97bead5	0x4fabda075e15e9	0xb1690c08e21	0	1	828185
5918604	2018-07-07 0:10	0xe3bdda39c984755	0x06012c8cf97bead5	0x4fabda075e15e9	0xb1690c08e21	0	1	828179

Figure 4.2: Daywise Splitting

4.2.5 Hypergraph Format

We stored the Hypergraph in JSON file format, where they Key will be a hyperedge, and the values present in the Key will be the nodes associated with the hyperedge.

Overall, the preprocessing steps were designed to ensure the dataset was clean, well-structured, and ready for further analysis. This approach allowed for easy extraction of meaningful insights from the transaction data and laid the foundation for more complex analyses, such as network analysis, node degree calculations, and hypergraph modeling.

The transition from pairwise relationships to higher-order interactions calls for a shift in how transactional data is structured. Inspired by real-world systems where multiple entities interact simultaneously—such as domain name resolutions involving multiple IPs—a hypergraph format enables the capture of these multi-entity relationships. The ability to preserve both connectivity and context in a

timestamp	fromLabel	toLabel	tokenAddressLabel	fromIsContract	toIsContract	tokenId
2018-07-07 00:01:25	71594	90964	53	0	0	824413
2018-07-07 00:01:50	98051	101339	53	0	0	556396
2018-07-07 00:03:07	87786	101274	381	0	1	3929
2018-07-07 00:04:04	828	71594	53	1	0	825908
2018-07-07 00:05:22	71594	825	53	0	1	831457
2018-07-07 00:06:00	82513	93145	381	0	1	3449
2018-07-07 00:06:06	1179	825	53	1	1	833698
2018-07-07 00:06:49	71594	825	53	0	1	830720
2018-07-07 00:07:33	71594	825	53	0	1	830716
2018-07-07 00:07:33	825	71036	53	1	0	833072
2018-07-07 00:08:12	71594	825	53	0	1	829836
2018-07-07 00:08:46	71594	825	53	0	1	828948
2018-07-07 00:10:01	71594	825	53	0	1	828570

Figure 4.3: Labelled Data

single representation makes hypergraphs ideal for modeling ERC-721 token data, where a single transaction may involve multiple agents (wallets, tokens, or contracts).

{"24":[99,86582],"53":[37889,98051,91143,94220,10005,65048,21017,22042,65820,64031,90657,70179,
14118,21544,22314,14894,93743,815,28465,67638,825,828,40514,74564,94539,68684,71758,68686,12879,
85840,33104,90964,50004,74844,66143,80736,70753,90737,65398,75384,83833,72316,71036,12157,87423,
93570,81284,67717,41094,65162,65164,72844,76430,75923,6292,93844,91540,101270,1179,86687,66720,
8609,101281,66725,28070,95144,92842,71594,95918,94894,50862,65455,53681,101044,91067,30142,101311,
101315,40396,16333,41421,92376,72409,101337,55515,101339,69341,101340,101342,101343,70881,101345,
101346,101348,101349,101350,101351,101352,6632,59114,2283,101354,101355,101356,101357,101360,8945,
93431,65021],"69":[95348,56823],"146":[66720,69399],"266":[72547,72572,72947],"381":[87763,100485,
101256,84617,94989,84625,101274,83486,87753,82476,85934,100398,82992,100787,82995,85814,82873,
100028,101309,101310,82882,82630,85703,82631,87752,87754,87755,87756,87757,87758,87759,87760,82513,
87762,87761,87764,87765,93396,87767,87768,93145,87770,87771,87772,87773,87774,82911,87775,87776,
87778,87779,87777,87766,87780,87786,87787,88829,97790],"396":[101330,12157,84269],"440":[94273,
93697,95235,89961,101353,92141,101277,95471,95856,90778,95964,87965,75743],"463":[88512,85840,
90691,91067,83436,12157,88799],"484":[101248,90816,92228,100486,100429,94993,6292,101338,95394,
101347,100646,95529,94961,95541,94775,92092,92093],"493":[94265,93382,85134,93383],"512":[100617,
99062,99599],"519":[100420,58180,88565,74264,65769,94331,93406]}

Figure 4.4: Storing Hypergraphs as JSON Files

4.3 Spectral Partitioning on a Random Graph

To further demonstrate the effectiveness of spectral clustering, we applied it on a synthetically generated undirected graph with two embedded communities. The process followed was as below:

- A graph with $n = 20$ nodes was created, consisting of two clusters of equal size ($n_1 = n_2 = 10$).
- Intra-cluster connections were added with a high probability ($p_{intra} = 0.8$), while inter-cluster edges were added with a lower probability ($p_{inter} = 0.15$), simulating community structure.
- The adjacency matrix A was constructed accordingly and made symmetric with zero diagonal.
- The unnormalized graph Laplacian L was computed as $L = D - A$, where D is the degree matrix.
- Eigen-decomposition was performed on L, and the second smallest eigenvector (Fiedler vector) was extracted.
- The sign of the Fiedler vector was used to partition the nodes into two groups.

The result is visualized in Figure 4.5, where nodes belonging to different clusters are colored distinctly. The Fiedler vector effectively separates the two planted communities, confirming the utility of spectral methods in graph partitioning.

Impact of Different Eigenvectors in Spectral Partitioning

In the previous example, spectral partitioning was performed using the second smallest eigenvector (Fiedler vector) of the graph Laplacian. To study how other eigenvectors influence the partitioning, we performed clustering based on different non-trivial eigenvectors (up to the 6th one).

For each eigenvector $\mathbf{v}_k$, nodes were divided into two groups based on the sign of the corresponding component: nodes with $v_k > 0$ were colored red, and those with $v_k \leq 0$ were colored blue.

Figure 4.6 shows the resulting partitions, demonstrating how higher-order eigenvectors provide increasingly fine or unstable separations, highlighting the significance of using the Fiedler vector (second smallest eigenvector) in practice.

4.3.1 Spectral Clustering on ERC-721 Token Transfer Hypergraph

In this study, we performed spectral clustering on the ERC-721 token transfer dataset. The hypergraph was constructed where wallets act as vertices and token transactions represent hyperedges connecting them.

Since traditional spectral methods are defined for simple graphs, the hypergraph was first converted into a simple undirected graph by connecting nodes that co-occur in at least one hyperedge. Although this simplifies computation, it introduces certain limitations by discarding higher-order interaction information inherent to hypergraphs.

The graph Laplacian L was constructed, and eigen-decomposition was performed to extract its eigenvalues and eigenvectors. Nodes

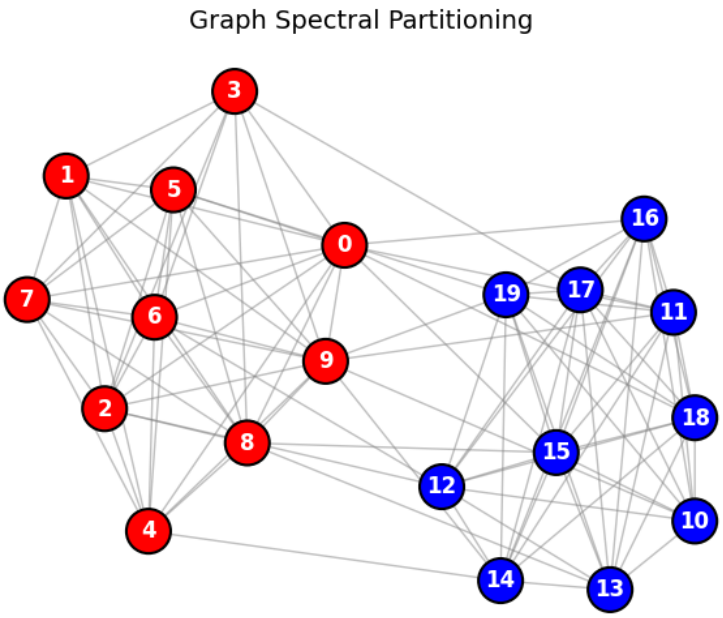

Figure 4.5: Spectral partitioning of a synthetic graph with planted community structure. Nodes are grouped based on the Fiedler vector derived from the Laplacian matrix.

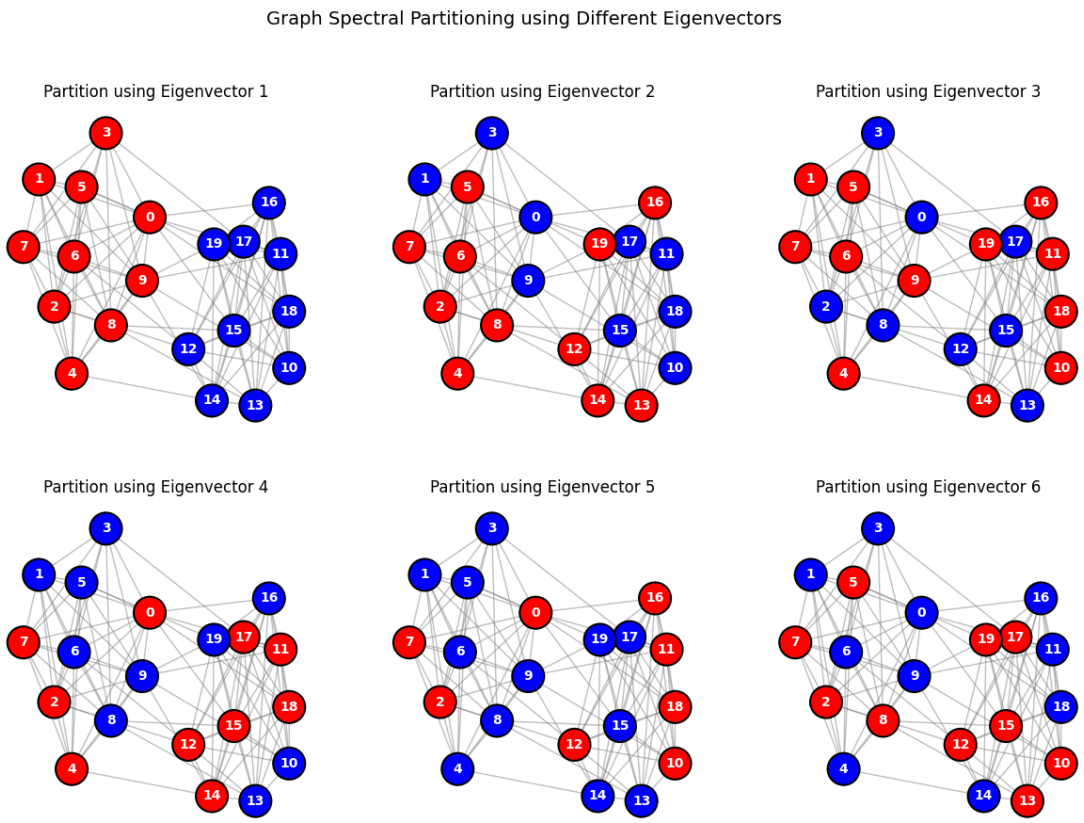

Figure 4.6: Spectral partitioning using different eigenvectors of the graph Laplacian. The first subplot uses the Fiedler vector (eigenvector 2), while subsequent plots use eigenvectors 3 to 7. The differences in partitioning demonstrate how eigenvector selection impacts clustering.

were embedded using the eigenvectors corresponding to the smallest non-trivial eigenvalues. Clustering was then performed on these embeddings using the KMeans algorithm.

Impact of Different Eigenvectors on Clustering

To analyze the influence of different eigenvectors, we experimented with using multiple eigenvectors (up to the sixth smallest) for clustering. For each eigenvector $\mathbf{v}_k$, nodes were split into three clusters based on their component values using KMeans.

Figure 4.6 shows the resulting partitions. It is observed that lower-order eigenvectors (especially the second and third smallest eigenvectors) capture the major community structure. However, as higher-order eigenvectors are used, the partitions become increasingly unstable and noisy.

This highlights the importance of carefully selecting eigenvectors in spectral clustering. Lower-order eigenvectors typically correspond to global structure, while higher-order ones capture finer, sometimes insignificant, variations.

Spectral Clustering on ERC-721 Token Transfer Network

In this work, we performed spectral clustering on the ERC-721 token transfer dataset by modeling it as a simple undirected graph. Each wallet address was treated as a node, and an undirected edge was created between two nodes if they participated together in a token transfer.

After constructing the undirected graph, the normalized graph Laplacian L was computed. Spectral embedding was then obtained by taking the eigenvectors corresponding to the smallest non-trivial eigenvalues. Clustering was performed in the spectral space using the KMeans algorithm, aiming to group nodes exhibiting similar transaction behaviors.

To study the effect of different eigenvectors on the clustering, we

experimented with using embeddings based on multiple eigenvectors (up to the sixth smallest eigenvalue). For each case, nodes were divided into three clusters.

Figure 4.7 illustrates the clustering results. It is observed that the second and third eigenvectors capture meaningful communities within the network, whereas higher-order eigenvectors lead to finer or unstable partitions, consistent with the theory of spectral clustering.

Limitations

One limitation of this approach is that the original hypergraph structure of token transfers was reduced to a simple undirected graph, losing the multi-party interaction information inherent in the dataset. This simplification may affect the accuracy and depth of the discovered communities, as complex higher-order relationships between wallets are not fully captured. Future extensions could explore clustering directly on the hypergraph without flattening it into pairwise connections.

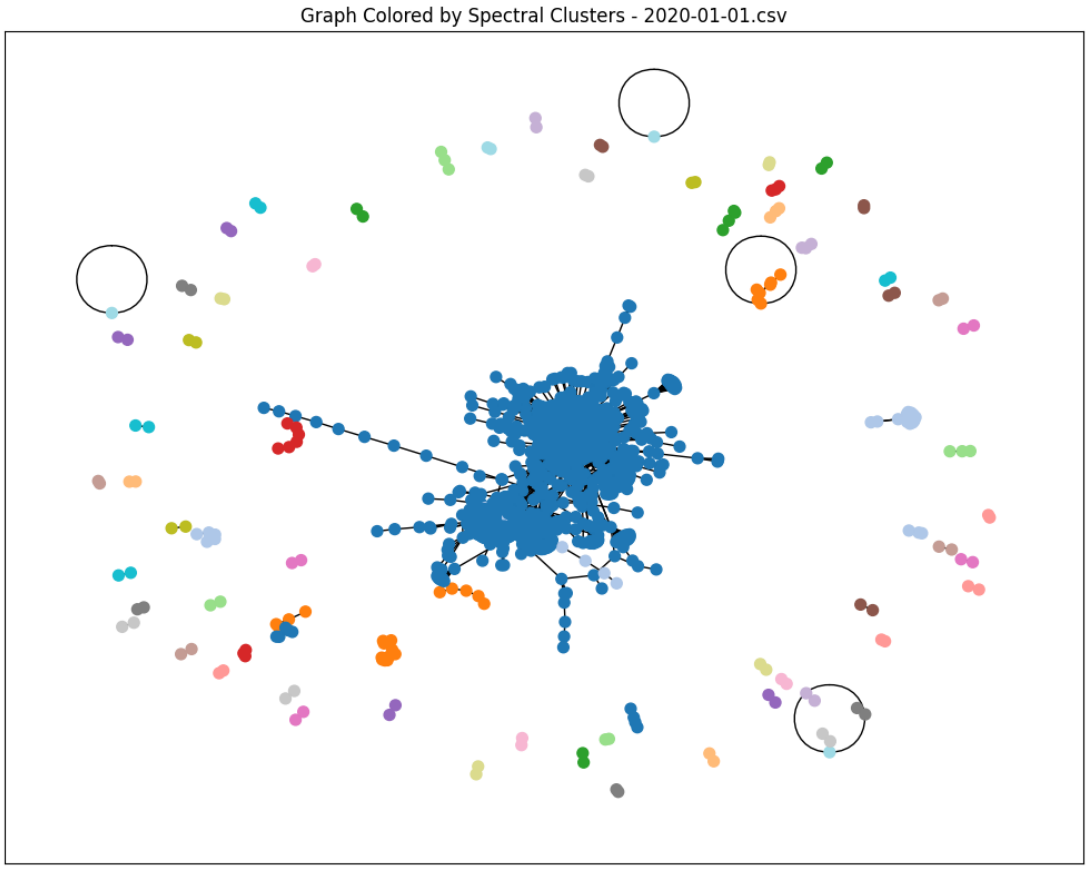

Figure 4.7: Spectral clustering of ERC-721 transactions on 1st January 2021. The network was partitioned into multiple clusters based on spectral embeddings. Cluster sizes varied significantly: the largest cluster (Cluster 0) contained 1683 nodes, while most other clusters were very small, typically containing 2 to 7 nodes. A few notable cluster sizes were: Cluster 1 (16 nodes), Cluster 4 (5 nodes), Cluster 7 (7 nodes), Cluster 8 (5 nodes), Cluster 11 (7 nodes), and several singleton or near-singleton clusters with 1–2 nodes each. This demonstrates a highly imbalanced clustering structure, where a dominant community is present alongside many sparsely populated groups.

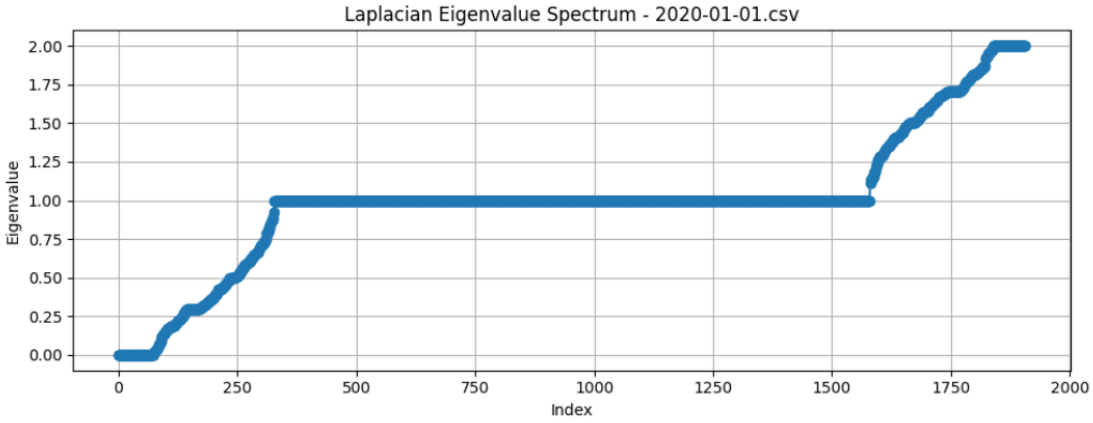

Figure 4.8: Laplacian eigenvalue spectrum for the ERC-721 transaction graph on 1st January 2021. The plot shows the sorted eigenvalues of the normalized graph Laplacian. A large flat region around eigenvalue 1 indicates many disconnected or loosely connected components, suggesting that the graph is highly fragmented. This structure is consistent with the observed clustering behavior, where one dominant cluster exists along with many small clusters.

4.4 Spectral Clustering on Hypergraphs

Spectral clustering provides a powerful framework for partitioning data, especially when modeled as a graph or hypergraph. Traditional spectral clustering, as formulated by Ng, Jordan, and Weiss, involves embedding the nodes of a graph into a lower-dimensional space using the top eigenvectors of a graph Laplacian, followed by k-means clustering in the embedded space.

Given a hypergraph $H = (V, E)$, where V is the set of nodes and E the set of hyperedges, we construct an incidence matrix $\mathbf{H} \in \mathbb{R}^{|V|\times|E|}$, where

$$H(v, e) = \begin{cases} 1, & \text{if } v \in e, \\ 0, & \text{otherwise.} \end{cases}$$

The degree matrices are defined as:

- Vertex degree matrix $\mathbf{D}_v \in \mathbb{R}^{|V|\times|V|}$, where $D_v(i, i) = \sum_{e\in E} w(e)H(i, e)$
- Hyperedge degree matrix $\mathbf{D}_e \in \mathbb{R}^{|E|\times|E|}$, where $D_e(e, e) = |e|$
- Hyperedge weight matrix $\mathbf{W} \in \mathbb{R}^{|E|\times|E|}$, typically diagonal

The normalized Laplacian for the hypergraph is then defined as:

$$\mathbf{L} = \mathbf{I} - \mathbf{D}_v^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T\mathbf{D}_v^{-1/2}$$

To perform spectral clustering:

1. Compute the Laplacian matrix $\mathbf{L}$
2. Find the k smallest eigenvalues and their corresponding eigenvectors to form the matrix $\mathbf{U} \in \mathbb{R}^{|V|\times k}$
3. Normalize the rows of $\mathbf{U}$
4. Apply K-means clustering on the rows of $\mathbf{U}$

This technique allows the identification of tightly-knit communities or structurally coherent groups in the hypergraph, even when traditional methods fail due to the high-order nature of the connections.

Ng et al. justify this method by demonstrating that, under ideal separability, the eigenvectors of the Laplacian recover the true cluster structure. For noisy or real-world hypergraphs, this embedding still captures significant topological structure, which is why we adopt this method in our pipeline.

This approach not only preserves high-order semantics inherent in hypergraphs but also aligns with the proven theoretical properties of spectral embeddings, making it highly effective for partitioning ERC-721 transaction data.

Recursive Spectral Clustering Visualization

To demonstrate the effectiveness of spectral clustering in revealing the underlying structure of a dataset, we applied recursive spectral partitioning on the well-known `dartboard2.csv` dataset — a benchmark dataset commonly used in clustering experiments due to its visually complex but mathematically tractable distribution.

The method followed includes:

- **Adjacency Matrix Construction:** We computed pairwise Euclidean distances between all points and used the Gaussian kernel to derive the similarity matrix:

$$A_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$$

  with $\sigma = 0.1$.

- **Laplacian Matrix Formation:** The degree matrix D was computed, and the unnormalized Laplacian was formed as:

$$L = D - A$$

- **Spectral Decomposition:** We solved the eigenvalue problem $L\vec{x} = \lambda\vec{x}$, and used the Fiedler vector (second smallest eigenvector) to bisect the graph.

- **Recursive Splitting:** This process was repeated recursively to form 4 clusters (2 levels of depth).

The figure below shows the result of this clustering process applied to the `dartboard2.csv` dataset.

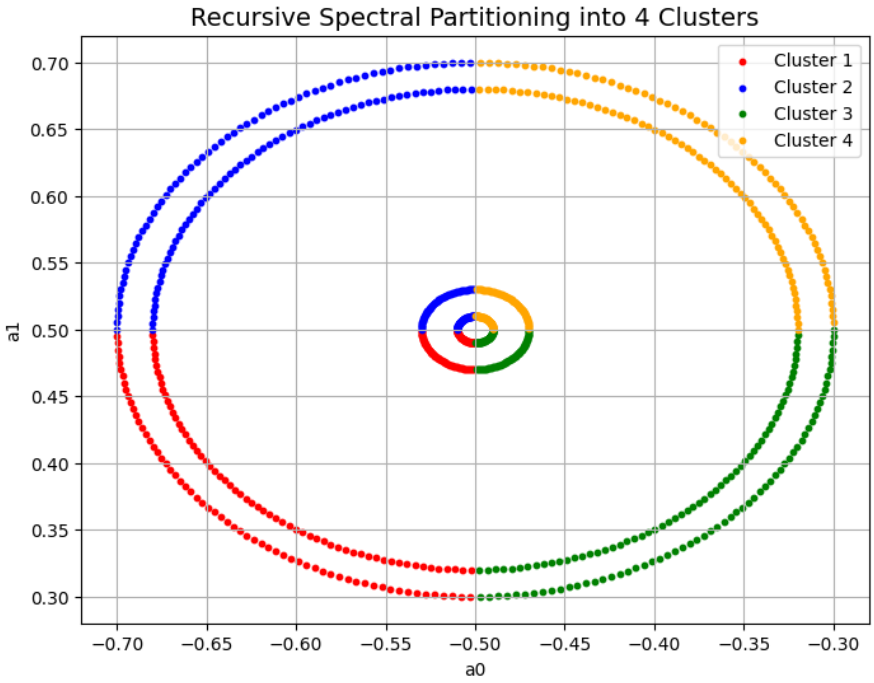

Figure 4.9: Recursive Spectral Clustering applied to the benchmark `dartboard2.csv` dataset. Four clusters were identified using two levels of Laplacian-based partitioning.

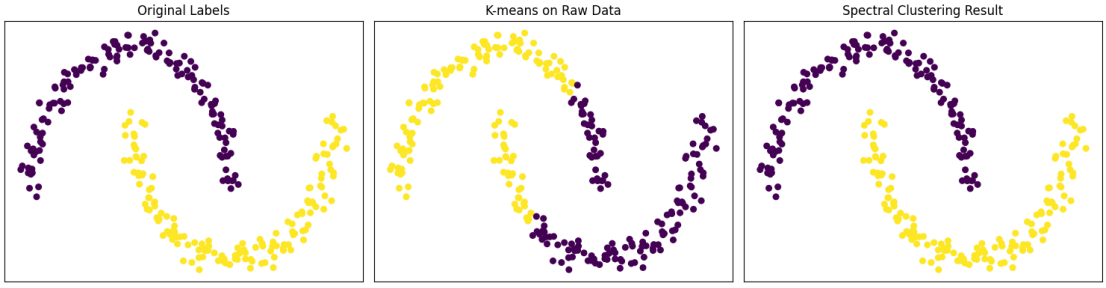

Figure 4.10: Comparison of clustering results: (Left) Ground truth labels of the `make_moons` dataset, (Middle) K-means applied directly on raw data, and (Right) Spectral clustering using eigenvectors of the normalized Laplacian. Spectral clustering effectively captures the non-linear structure.

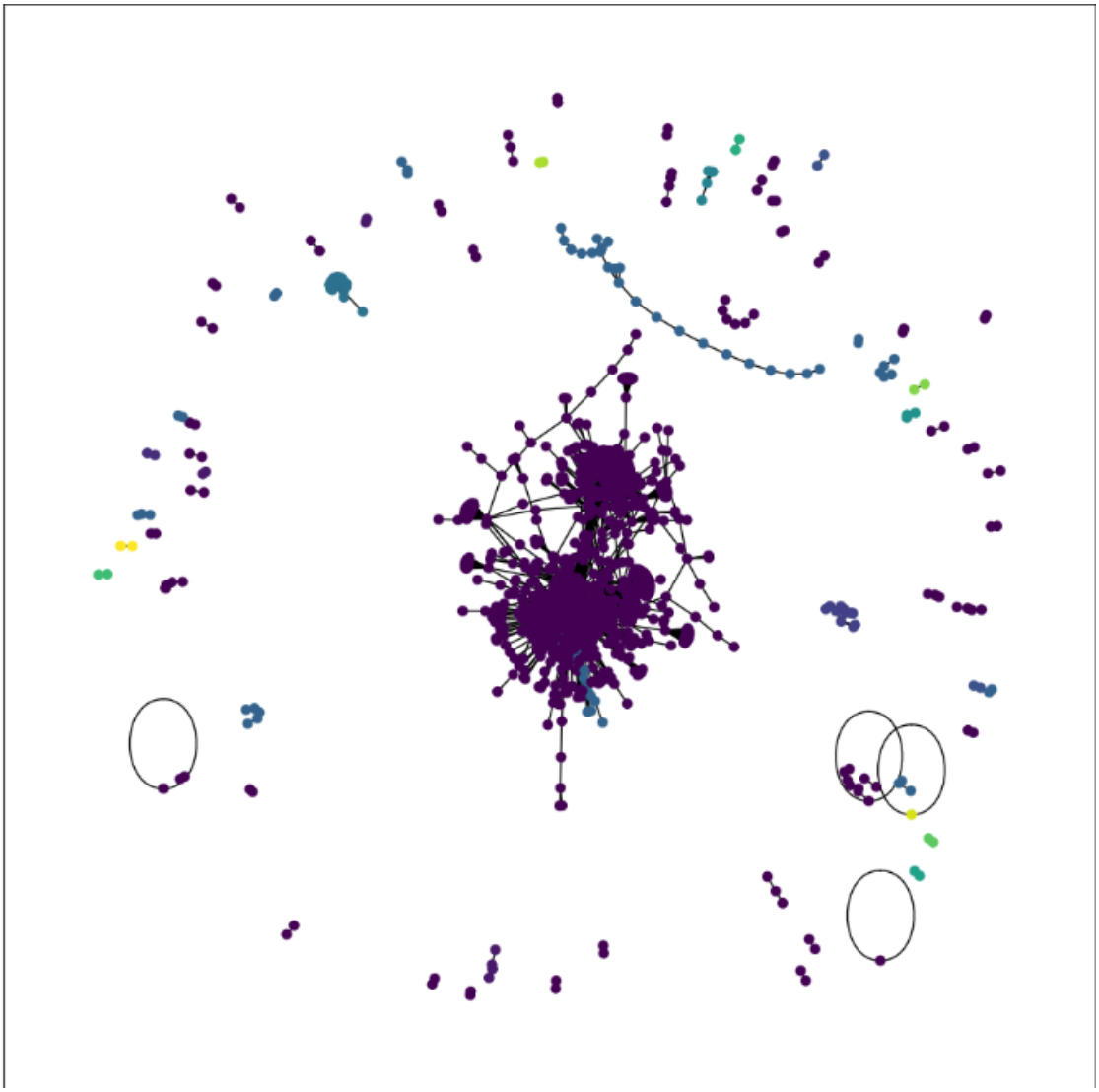

Figure 4.11: Hypergraph clustering visualization based on ERC-721 transactions on 1st January 2020. The hypergraph consists of 17 detected clusters. The largest cluster (Cluster 0) dominates the structure with 1765 nodes, indicating a major component of interconnected wallets. Several smaller clusters exist, with most clusters having very few nodes (e.g., clusters with only 2 or 3 nodes), revealing the fragmented nature of transactions outside the main network. This distribution is consistent with the Laplacian eigenvalue spectrum, where a large multiplicity around eigenvalue 1 was observed, suggesting loosely connected or disconnected components.

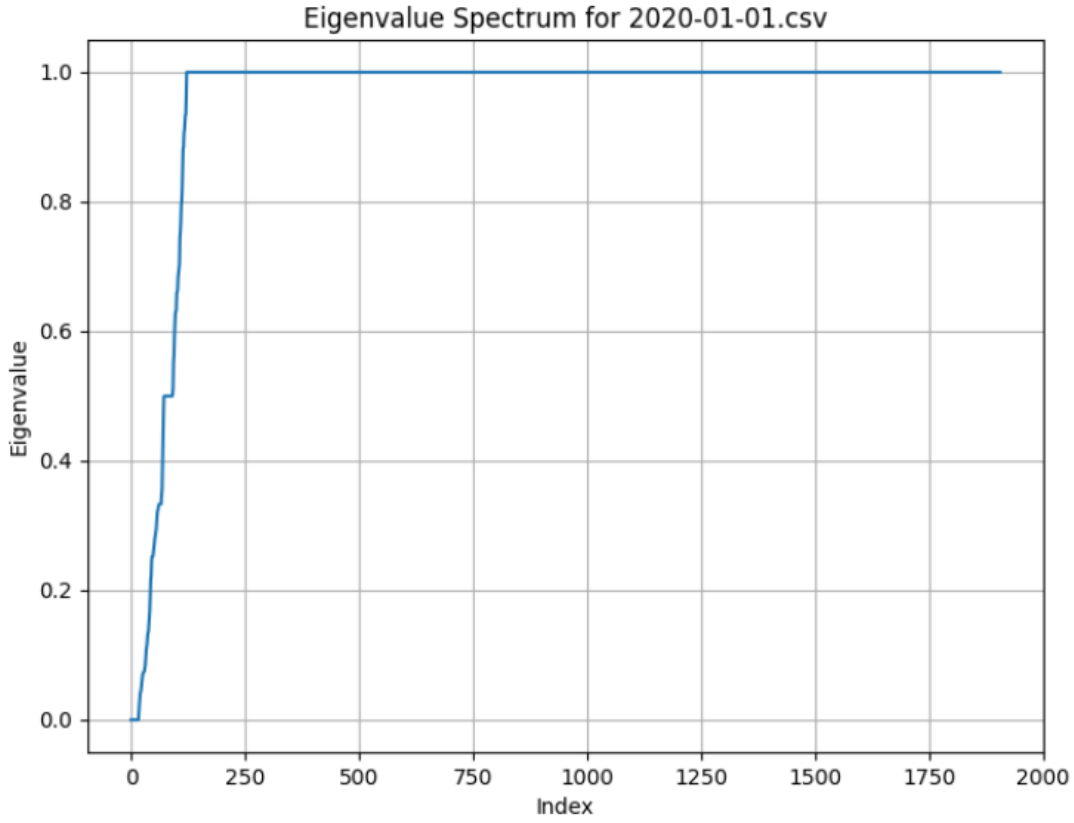

Figure 4.12: Eigenvalue spectrum of the hypergraph Laplacian for transactions on 1st January 2020. The plot shows a rapid rise of eigenvalues from 0 to 1, with the majority of eigenvalues exactly equal to 1. This indicates the presence of a large number of disconnected or weakly connected components in the hypergraph. The steep jump suggests that only a small subset of nodes are tightly connected (forming a major cluster), while the rest remain isolated or form very small clusters. This eigenvalue behavior is consistent with the cluster distribution observed in the hypergraph visualization.

5 Results

In this chapter, we present the comparative analysis of clustering performance between traditional graphs and hypergraphs. Different evaluation metrics were considered, including cluster size, number of clusters, and standard clustering validation indices.

5.1 Maximum Cluster Size Comparison

The maximum cluster size refers to the size of the largest cluster formed during the clustering process. A smaller maximum cluster size indicates that the clusters are more evenly balanced, avoiding dominance by a few large groups. In applications such as community detection or pattern analysis, balanced clustering helps to preserve meaningful structures.

The maximum cluster size is simply defined as:

$$\text{Max Cluster Size} = \max\left(|C_i|\right) \quad \text{for all clusters } C_i$$

Where C_i represents the set of points in cluster i, and $|C_i|$ is the size (number of points) of cluster i.

In traditional graph clustering, the dominance of a few large clusters may lead to inefficient results, while hypergraph clustering tends to promote smaller, more evenly distributed clusters, leading to a better-balanced partitioning.

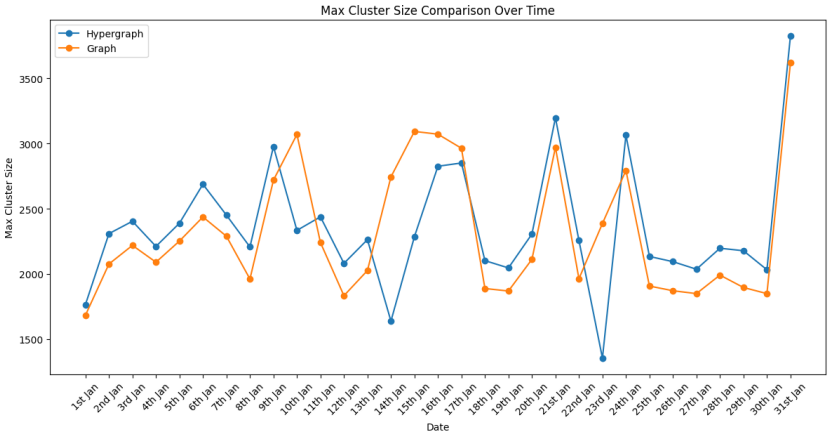

Figure 5.1: Comparison of maximum cluster size between Graph and Hypergraph. The Graph structure leads to smaller maximum cluster sizes, indicating a more balanced and homogeneous clustering compared to the Graph. A smaller maximum cluster size suggests that the data points are more evenly distributed among clusters rather than being dominated by a few large groups.

5.2 Number of Clusters Comparison

The number of clusters represents how finely the dataset is partitioned. More clusters suggest that the clustering method can capture finer structures or subtle communities, while fewer clusters might indicate coarse grouping. The ability to adjust to the underlying distribution of the data is crucial for a good clustering algorithm.

The number of clusters is a simple count of how many clusters are formed during the clustering process. If $C_1, C_2, \ldots, C_k$ represent all the clusters, the number of clusters N is:

$$N = k \quad \text{where } k = \text{number of clusters}$$

This metric is essential for understanding how well the algorithm captures the finer structure of the data. A higher number of clusters can be beneficial when detecting smaller or more nuanced communities.

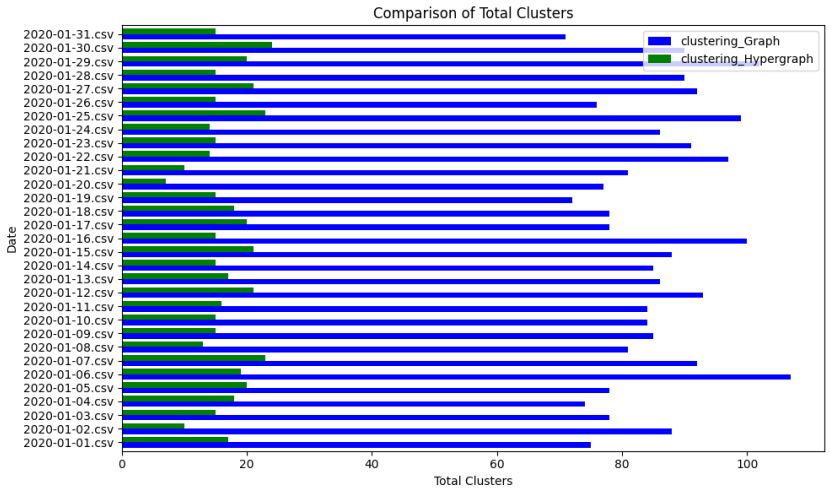

Figure 5.2: Comparison of the number of clusters formed in Graph and Hypergraph. Graphs tend to produce a larger number of clusters, suggesting finer-grained partitioning of the dataset. A higher number of clusters indicates that the model is capable of capturing subtle community structures.

5.3 Silhouette Score Analysis

The Silhouette score measures how similar a data point is to its own cluster compared to others. It combines cohesion (within-cluster similarity) and separation (between-cluster dissimilarity).

- The score ranges from -1 (poor) to +1 (excellent). - Higher Silhouette scores indicate dense and well-separated clusters.

The Silhouette score for a data point x_i is calculated as:

$$s(x_i) = \frac{b(x_i) - a(x_i)}{\max(a(x_i), b(x_i))}$$

Where: - $a(x_i)$ is the average distance from point x_i to all other points in the same cluster. - $b(x_i)$ is the average distance from point x_i to all points in the nearest cluster (i.e., the nearest cluster that x_i is not part of).

The overall Silhouette score for the entire dataset is the average of the individual Silhouette scores of all data points:

$$S = \frac{1}{N} \sum_{i=1}^{N} s(x_i)$$

Where N is the total number of points in the dataset. A higher Silhouette score indicates that the points within clusters are more similar to each other and that clusters are well separated.

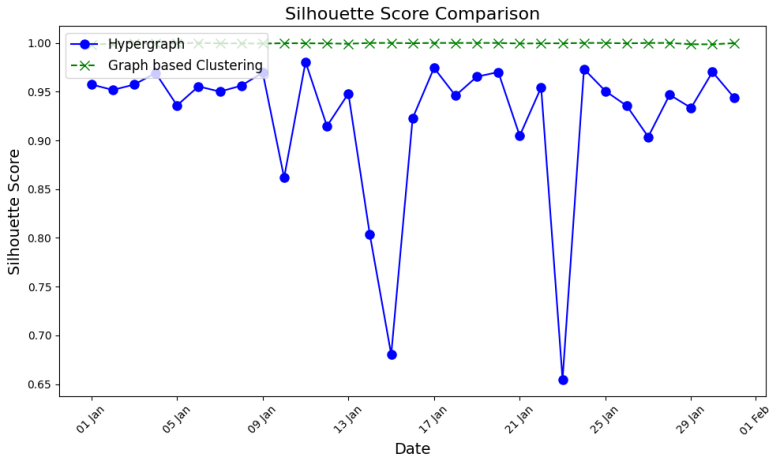

Figure 5.3: Silhouette score comparison between Graph and Hypergraph clustering. The Silhouette score ranges from -1 to +1, where a higher value indicates better clustering. It measures how similar an object is to its own cluster compared to other clusters. Graphs exhibit higher Silhouette scores, indicating that clusters are more compact and well-separated.

5.4 Davies–Bouldin (DB) Score Analysis

The Davies–Bouldin (DB) score is a clustering evaluation metric that focuses on cluster compactness and separation. - Lower DB score means better clustering. - It calculates the average similarity between each cluster and its most similar one (bad if clusters overlap too much).

The Davies-Bouldin score for two clusters C_i and C_j is:

$$R(C_i, C_j) = \frac{s(C_i) + s(C_j)}{d(C_i, C_j)}$$

Where: - $s(C_i)$ is the average distance between the points in cluster C_i and the centroid of C_i (i.e., the cluster's internal compactness). - $d(C_i, C_j)$ is the distance between the centroids of clusters C_i and C_j (i.e., the separation between clusters).

The Davies-Bouldin score for the entire clustering solution is the average of the maximum similarity between each cluster and all other clusters:

$$DB = \frac{1}{k} \sum_{i=1}^{k} \max_{j \neq i} R(C_i, C_j)$$

Where k is the number of clusters. A lower DB score indicates that clusters are more compact and well-separated, which is a desirable property.

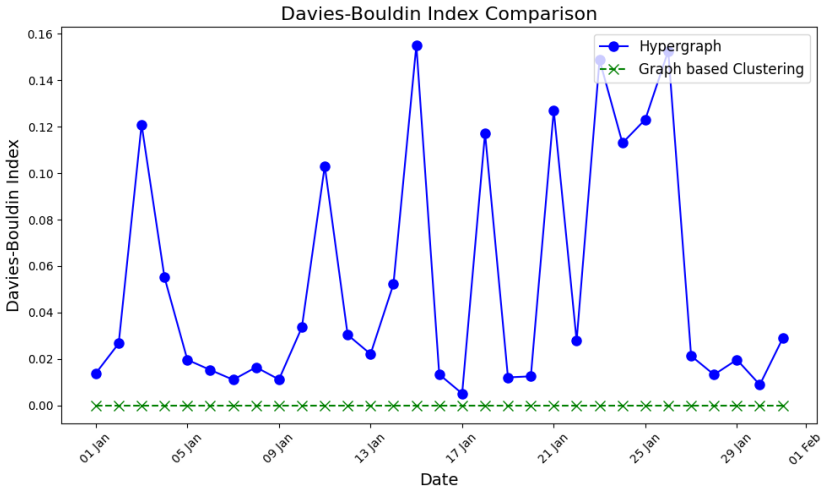

Figure 5.4: Davies–Bouldin score comparison for Graph and Hypergraph. The Davies–Bouldin score evaluates intra-cluster similarity and inter-cluster differences, where a lower value indicates better clustering quality. Graphs achieve lower DB scores, demonstrating that their clusters are more compact internally and better separated externally.

5.5 Calinski–Harabasz (CH) Score Analysis

The Calinski–Harabasz (CH) index, also known as the Variance Ratio Criterion, evaluates clustering performance based on the dispersion between clusters versus within clusters. - Higher CH score means better-defined clusters. - It measures the ratio of between-cluster dispersion to within-cluster dispersion.

The Calinski-Harabasz index is calculated as:

$$CH = \frac{\text{tr}(B_k)}{\text{tr}(W_k)} \cdot \frac{N - k}{k - 1}$$

Where: - $\text{tr}(B_k)$ is the trace of the between-cluster dispersion matrix. - $\text{tr}(W_k)$ is the trace of the within-cluster dispersion matrix. - N is the total number of data points. - k is the number of clusters.

A higher CH score indicates that the clustering solution has well-separated and compact clusters, which is generally a sign of a good clustering solution.

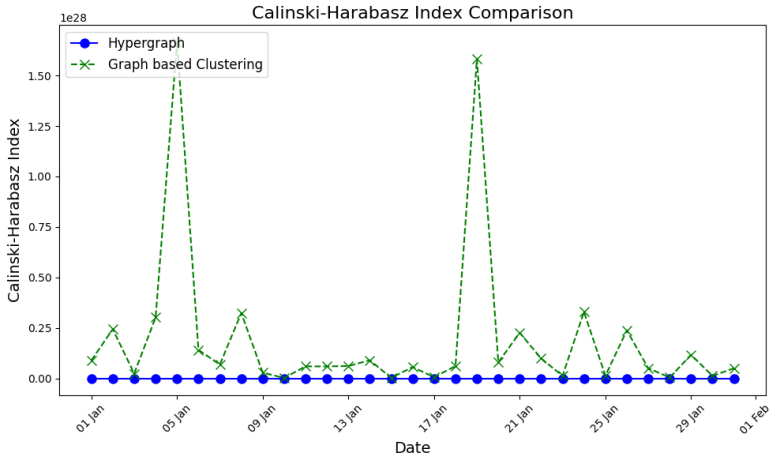

Figure 5.5: Calinski–Harabasz score comparison between Graph and Hypergraph. The Calinski–Harabasz index, also called the Variance Ratio Criterion, measures cluster dispersion: a higher score indicates better clustering. It is calculated as the ratio of between-cluster dispersion to within-cluster dispersion. Graphs achieve higher CH scores, reflecting a clear separation between clusters with minimal internal variance.

In conclusion, we observe that in our case, Graphs provide better clustering results compared to Hypergraphs. While Hypergraphs show improved performance in some aspects such as the number of clusters and Silhouette score, Graph-based clustering yields more compact and well-separated clusters. This is reflected not only in the higher Calinski-Harabasz score but also in the Silhouette and Davies-Bouldin scores. The higher Silhouette score for Graphs indicates that the clusters are more cohesive and better separated, while the lower Davies-Bouldin score for Graphs suggests that the clusters are more compact and distinct. This collectively suggests that the traditional graph structure is better suited for clustering in our dataset, with more defined clusters and better overall clustering quality.

Bibliography

[1] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez and D. U. Hwang. Complex networks: Structure and dynamics. Physics Reports. 424, 175-308, (2006).

[2] C. Koch , G. Laurent. Complexity and the nervous system. Science. 1999, **284**(5411) 96-98.

[3] H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai , A.-L. Barabási. The large-scale organization of metabolic networks. Nature.

[4] H. Jeong, S.P. Mason, A.-L. Barabási, Z.N. Oltvai. Lethality and centrality in protein networks. Nature. 2001, **411**, 41–42.

[5] G. Weng, U. S. Bhalla, R. Iyengar. Complexity in biological signaling systems. Science. 1999, **284**(5411), 92-96.

[6] M. E. J. Newman. The structure of scientific collaboration networks. PNAS. 2001, **98**(2), 404-409.

[7] S. Redner. How popular is your paper? An empirical study of the citation distribution. Eur. Phys. J. B. 1998, **4**, 131–134.

[8] M.E.J. Newman. The Structure and Function of Complex Networks. SIAM Review. **45**(2), (2003).

[9] M. De Domenico, C. Granell, M. A. Porter and A. Arenas. The physics of spreading processes in multilayer networks. Nature Phys 12, 901–906, (2016).

[10] M. Kivel, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno and M A Porter. Multilayer networks, Journal of Complex Networks, (3), 203–271, (2014).

[11] S. Boccaletti, P. De Lellis, C.I. del Genio, K. Alfaro-Bittner, R. Criado, S. Jalan, M. Romance. The structure and dynamics of networks with higher order interactions. Physics Reports, 1018, 1-64, (2023).

[12] F. Battiston et al. Networks beyond pairwise interactions: structure and dynamics. Phys. Rep. 874, 1 (2020).

[13] R. Lambiotte, M. Rosvall, and I. Scholtes. From networks to optimal higher- order models of complex systems. Nat. Phys. 15, 313–320 (2019).

[14] A. R. Benson, D. F. Gleich, J. and Leskovec. Higher-order organization of complex networks. Science. 353, 163–6, (2016).

[15] S. Gu, M. Yang, J.D. Medaglia, R.C. Gur, R.E. Gur, T.D. Satterthwaite and D.S. Bassett. Functional hypergraph uncovers novel covariant structures over neurodevelopment. Human brain mapping, 38, (2017).

[16] J. Grilli, G. Barabás, M. J. Michalska-Smith, and S. Allesina, Higher-order interactions stabilize dynamics in competitive network models, Nature (London) 548, 210 (2017).

[17] A. Patania, G. Petri, and F. Vaccarino, EPJ Data Sci. 6, 1 (2017).

[18] S. Klamt, U.-U. Haus, and F. Theis, PLoS Comput. Biol. 5, (2009).

[19] A. Zimmer, I. Katzir, E. Dekel, A. E. Mayo, and U. Alon, Proc. Natl. Acad. Sci. U.S.A. 113, 10442 (2016).

[20] R. Herzog, F. E. Rosas, R. Whelan et. al., Genuine high-order interactions in brain networks and neurodegeneration. Neurobiology of Disease. 175, 105918 (2022).

[21] L. Faes et al., A New Framework for the Time- and Frequency-Domain Assessment of High-Order Interactions in Networks of

Random Processes. IEEE Transactions on Signal Processing, 70, 5766-5777, (2022)

[22] R. Sawhney, S. Agarwal, A. Wadhwa, T. Derr, and R. R. Shah. Stock Selection via Spatiotemporal Hypergraph Attention Network: A Learning to Rank Approach. Proceedings of the AAAI Conference on Artificial Intelligence, 35(1), 497-504, (2021).

[23] M. I. Chelaru, S. Eagleman, A. R. Andrei, R. Milton, N. Kharas and V. Dragoi. High-order interactions explain the collective behavior of cortical populations in executive but not sensory areas. Neuron, 109, 3954-3961, (2021).

[24] A. Santoro F. Battiston, G. Petri and E. Amico. Unveiling the higher-order structure of multivariate time series. arXiv:2203.10702.

[25] P. S. Skardal and A. Arenas, Commun. Phys. 3, 1 (2020).

[26] A. P. Millán, J. J. Torres, and G. Bianconi, Phys. Rev. Lett. 124, 218301 (2020).

[27] S. Chowdhary, A. Kumar, G. Cencetti, I. Iacopini, and F. Battiston, J. Phys. 2, 035019 (2021).

[28] L. Neuhäuser, A. Mellor, and R. Lambiotte, Phys. Rev. E. 101, 032310 (2020).

[29] M. T. Schaub, A. R. Benson, P. Horn, G. Lippner, and A. Jadbabaie, SIAM Rev. 62, 353 (2020).

[30] U. Alvarez-Rodriguez, F. Battiston, G. F. de Arruda, Y. Moreno, M. Perc, and V. Latora, Nat. Hum. Behav. 5, 586 (2021).

[31] A. Civilini, N. Anbarci, and V. Latora, Phys. Rev. Lett. 127, 268301 (2021).

[32] D. Wang, P. Yi, Y. Hong, J. Chen, and G. Yan. Evolutionary game on any hypergraph. arXiv:2404.03305.

[33] G. Burgio, S. Gómez, A. and Arenas. Triadic Approximation Reveals the Role of Interaction Overlap on the Spread of Complex Contagions on Higher-Order Networks. Phys. Rev. Lett. 132, 077401, (2024).

[34] G. Cencetti, F. Battiston, B. Lepri, et al. Temporal properties of higher-order interactions in social networks. Sci Rep 11, 7028 (2021).

[35] Glaser, Florian and Glaser, Florian and Bezzenberger, Luis, Beyond Cryptocurrencies - A Taxonomy of Decentralized Consensus Systems (March 1, 2015). 23rd European Conference on Information Systems (ECIS), Münster, Germany, 2015, Available at SSRN: https://ssrn.com/abstract=2605803

[36] Nofer, M., Gomber, P., Hinz, O. et al. Blockchain. Bus Inf Syst Eng 59, 183–187 (2017). https://doi.org/10.1007/s12599-017-0467-3

[37] Buterin, Vitalik. "Ethereum white paper." GitHub repository 1 (2013): 22-23.

[38] D. Vujičić, D. Jagodić and S. Ranđić, "Blockchain technology, bitcoin, and Ethereum: A brief overview," 2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH), East Sarajevo, Bosnia and Herzegovina, 2018, pp. 1-6, doi: 10.1109/INFOTEH.2018.8345547.

[39] Kugler, Logan. "Non-fungible tokens and the future of art." Communications of the ACM 64, no. 9 (2021): 19-20.

[40] W. Entriken, D. Shirley, E. Evans, and N. Sachs, "ERC-721 non-fungible token standard," 2018, accessed 2019-10-12.

[41] M. d. Angelo andG. Salzer, “Tokens, Types, andStandards: Identification andUtilization inEthereum,” in 2020 IEEE International ConferenceonDecentralizedApplicationsandInfrastructures(DAPPS), 2020,pp.1–10.

[42] Bretto, A. (2013). Hypergraph Theory. Mathematical Engineering. doi:10.1007/978-3-319-00080-0

[43] Zheng, Peilin, Zibin Zheng, Jiajing Wu, and Hong-Ning Dai. "Xblock-eth: Extracting and exploring blockchain data from ethereum." IEEE Open Journal of the Computer Society 1 (2020): 95-106.