Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

Clustering Ethereum Addresses

Semester Thesis

Dominic Grandjean
grdomini@ethz.ch

Distributed Computing Group
Computer Engineering and Networks Laboratory
ETH Zürich

Supervisors:
Lioba Heimbach, Quentin Kniep
Prof. Dr. Roger Wattenhofer

June 16, 2022

Acknowledgements

I would like to thank Lioba Heimbach and Quentin Kniep for supervising my semester thesis. They let me work independently but were always there when I needed some support. I learned a lot and had a great time.

Furthermore, I would also like to thank Prof. Dr. Roger Wattenhofer for giving me the opportunity to work on this project.

Zurich, June 2022

Dominic Grandjean

Abstract

Whether for fraud detection or deanonymization, address clustering on Bitcoin and its derivatives has been done for many years and is a valuable tool for on-chain data analysis. Of course, there are similar incentives for clustering user addresses on alternative blockchains like Ethereum. However, Ethereum's account-based model requires dedicated clustering heuristics, some of which have been introduced in recent years.

In this thesis, we show how addresses can be clustered in Ethereum yielding entities that are likely in control of multiple addresses. We enhance an existing method for clustering entities based on reused exchange deposit addresses. Our results show that we can cluster 8.8% of all unique addresses ever used on Ethereum, including internally owned addresses. In addition, we can identify 1'410'523 entities which control at least 2 and at most 1'000 addresses. We observe that an average entity owns 4.81 user addresses, is active for 307 days and holds 5.0ETH in all its addresses. Finally, we study their interaction and behaviour towards popular DeFi projects as well as layer-2 rollups and Eth2 staking on the Ethereum Beacon Chain.

Contents

Acknowledgements ii

Abstract ii

1 Introduction 1

2 Related Work 2

3 Background 3

3.1 Account Model . . . 3

3.1.1 Externally Owned Account . . . 3

3.1.2 Smart Contracts . . . 3

3.2 On-Chain Data . . . 4

3.2.1 Event Logs and Topics . . . 4

3.2.2 Centralized Exchanges . . . 4

4 Data Collection 5

4.1 Hardware . . . 5

4.2 Software . . . 5

4.3 Data . . . 5

4.3.1 Miner Addresses . . . 6

4.3.2 Transactions . . . 6

4.3.3 Smart Contract Addresses . . . 6

4.3.4 Centralized Exchange Addresses . . . 7

5 Address Clustering 8

5.1 Deposit Address Reuse . . . 8

5.2 Triplet Extraction . . . 8

5.3 Triplet Merging . . . 10

6 Results **12**
6.1 Entity Analysis . . . 12
6.1.1 Address Distributions . . . 13
6.1.2 Popular Exchanges . . . 15
6.1.3 Entity Activity and Entity Age . . . 16
6.1.4 Entity Funds . . . 17
6.2 DeFi, Layer-2 and Eth2 Interaction Analysis . . . 18
6.2.1 DeFi Behaviour Analysis . . . 18
6.2.2 Layer-2 Interaction . . . 20
6.2.3 Eth2 Beacon Chain Staking . . . 20

7 Conclusion **21**
7.1 Future Work . . . 21

Bibliography **22**

A Individual Project Activities **A-1**
A.1 Decentralized Exchanges . . . A-1
A.2 Lending Protocols . . . A-5
A.3 Layer-2 . . . A-7

Chapter 1

Introduction

The emergence of Bitcoin in 2008 [1] paved the way for an entire cryptocurrency ecosystem. Ethereum, a blockchain proposed in 2013 [2] further expanded the notion of a blockchain and enabled additional functionality on the base layer. With a market dominance of roughly 18% at the time of writing [3], Ethereum is a key driver of innovation in the ecosystem.

A growing number of daily active addresses using Ethereum is one metric indicating a wider adoption of cryptocurrencies over the years [4]. This resulted in a growing market cap of the entire space, reaching US$2.95 trillion at its last peak in November 2021 [3]. The value increase in recent years also attracted the interest of regulatory bodies around the world. The introduction of KYC (Know Your Customer) verification on centralized exchanges can be seen as a direct result. In combination with chain analysis tools, the authorities have the possibility to deanonymize parts of the otherwise pseudonymous network.

In this thesis, we perform clustering of Ethereum addresses into entities. We focus on addresses that have at some point interacted with a deposit address of a centralized exchange. Addresses that belong to the same entity are likely to be owned by the same person or group of people and their identity is likely to be known to the authorities. After analyzing the entities themselves, we then examine the behaviour and interaction of these entities with popular projects in the recently emerged Decentralized Finance (DeFi) space to gain better insight into the clustered entities.

In addition, we determine the funds held by these entities, which gives us a view of the decentralization of funds in the network. For Ethereum, this metric will be even more relevant after the transition from Proof of Work (PoW) to Proof of Stake (PoS), as this change will give more power to entities with large ETH holdings. Especially for on-chain governance on emerging layer-2 solutions, the knowledge and observation of large entities is of importance.

Chapter 2

Related Work

Clustering cryptocurrency addresses into entities has been performed in numerous occasions. Due to its size and dominance in the space, the Bitcoin network has received the most attention [5, 6, 7]. Ethereum did not inherit the UTXO model of Bitcoin. Therefore, the clustering heuristics of Bitcoin cannot be applied to Ethereum. Existing studies on Ethereum mainly focus on the address graph [8, 9], without entity clustering.

In a recent paper [10], Bonifazi et al. discuss an automatic approach to classify users based on their behaviour and interaction with Ethereum. However, they do not cluster individual entities on a large scale.

Friedhelm [11] proposes heuristics that exploit patterns related to deposit addresses, multiple participation in airdrops and token authorization mechanisms. To the best of our knowledge, Friedhelm was first to propose deposit address reuse as a basis of entity clustering on Ethereum. The author shows that his approach can cluster 17.9% of all active, externally owned account addresses. He also finds that there are more than 340,000 entities that likely control multiple addresses. In this thesis, we use his results as a guide for our own clustering algorithm, and propose an enhanced method which also merges entities that do not share a common transaction with the same deposit address.

User behaviour analysis was conducted primarily to understand the general use of smart contracts and user relationships [8], as well as to understand specific behaviours on a subset of popular DeFi projects [12]. However, we could not find evidence for a work that performs analysis of clustered entities and their interaction behaviour with the DeFi or layer-2 spaces on Ethereum.

Chapter 3

Background

In this section we introduce terminologies and explain the basics of Ethereum accounts and on-chain data handling. We also look at the user's interaction with centralized exchanges in terms of on-chain data.

3.1 Account Model

In contrast to the UTXO model of Bitcoin, many blockchains that support smart contracts have adopted an account-based model, where a list of accounts, including all balances, is stored in the global state. If the balance of the sending account is larger than the amount sent, the sending account is debited while the receiving account is credited with the corresponding transaction amount. Accounts in an account-based model are thus similar to regular bank accounts. Two account types exist in Ethereum:

3.1.1 Externally Owned Account

Externally Owned Accounts (EOAs) are controlled by the entity that controls the respective private keys. EOAs can initialize transactions without any external influence, while the transactions between EOAs are limited to value and message transfers.

3.1.2 Smart Contracts

Smart contracts are accounts that are deployed to the network and controlled by code. Unlike an EOA, a smart contract cannot initiate a transaction. To deploy a contract account, on-chain storage space needs to be allocated, for which a fee is charged at initialization.

3.2 On-Chain Data

On-chain data, as the name suggests, can be classified as data stored on the blockchain itself. This data is stored by a network of Ethereum nodes, computers, which run a blockchain specific client software. This data includes the entire history of every valid transaction performed on the network. Besides sender, recipient and value of a transaction, the blockchain also stores metadata related to the transaction.

3.2.1 Event Logs and Topics

If at least one party of a transaction is a smart contract, additional metadata is stored on the blockchain. These so-called event logs provide information about the nature of the interaction between the parties. A transaction can contain more than one log, hence the name logs. Each log consists of both topics and data. Topics are 32-byte (256 bit) "words" that are used to describe what happened in an event. When looking at the topics of a transaction, you can find out what the transaction was about. A block explorer like Etherscan is generally used to translate the 32-byte topic to a concrete event such as a deposit [13], withdrawal or swap.

3.2.2 Centralized Exchanges

Centralized exchanges, which are usually owned and operated by a company, have played and continue to play a major roll as on and off ramps between the fiat and cryptocurrency space. To use such an exchange, a user must open a trading account on the exchange's website. Depending on the volume to be traded, a KYC verification procedure is required. The user then requests a unique deposit address from the exchange, which is linked to the user's account, and deposits the funds. After this deposit, the exchange takes custody of the deposited funds. This means that the user is not in possession of the private keys to the funds. A best practice that is widely used in the industry is the subsequent transfer of the funds by the exchange to a secure cold storage wallet owned by the exchange. This reduces potential attack vectors. An exchange may have multiple addresses where users' funds are collectively stored. After this procedure, two transactions can be found on the blockchain resembling the transfer from the user to cold storage.

Some exchanges, such as Kraken, have taken a slightly different approach where users send funds to a smart contract deployed by Kraken, which then transfers the funds to cold storage. This method reduces transaction costs on Kraken's side. However, as we will see in Chapter 6.1.2, the majority of large exchanges have not yet adopted this approach.

Chapter 4

Data Collection

In this chapter, we explain the hardware and software used for the data collection. In addition, the different methods for the data acquisition are discussed.

4.1 Hardware

A computer equipped with an AMD Ryzen Threadripper 3960X (24 cores, 48 threads), 128GB of RAM, and a 4TB SSD is used for data collection. This computer hosts the Ethereum client described in the next section. To reduce latency, the code that requests data from the client is also executed on this machine. A high core count for parallelization turned out to be beneficial throughout this thesis.

4.2 Software

On top of Ubuntu Server, we use Erigon as the Ethereum client [14]. The Erigon full node database containing the blockchain reached a size of about 2TB and was synced with the rest of the network within 2 days. For the interaction with Erigon the python package Web3.py is used [15]. Whenever possible, requests made to the Erigon client via Web3 are parallelized. A separate database is not required for this thesis but is recommended when dealing with more data. Future blockchain growth should also not be neglected. Installation guides can be found in the GitLab repository [16].

4.3 Data

Multiple separate datasets are created for clustering and subsequent behavioral analysis. The underlying data originates from the Ethereum blockchain stored by the Erigon node and multiple publicly accessible online sources.

4.3.1 Miner Addresses

First, we identify 5'585 addresses that have mined at least one block since Ethereum's inception. These addresses therefore belong to miners. The addresses are collected by going through the blockchain from block 0 to block 14'699'999, reading the address that has mined each individual block.

4.3.2 Transactions

By far the largest dataset created is a collection of all valid transactions recorded on the blockchain, starting from block 0 and ending with block 14'699'999. We parallelize the process, resulting in multiple files which are subsequently merged into a single file of approximately 139GB in size. The final CSV file contains three columns. One of them indicated the block number in which the transaction was recorded. The other two indicate the address of the sending and receiving parties, respectively. No restrictions on the transaction volume, the sender, or the recipient address are posed. In total, we detect and save 1'559'436'843 transactions which matches with secondary sources [17].

4.3.3 Smart Contract Addresses

For a subsequent analysis of entity behaviour in the DeFi space, we collect contract addresses of the most popular projects in DeFi. We also study the interaction of entities with layer-2 projects and the staking contract for staking on the Ethereum Beacon Chain. A breakdown of all projects and the total number of respective contract addresses can be seen in Table 4.1.

Decentralized exchanges (DEXs), in their current form, consist of many smart contracts, each resembling a trading pair, also called a liquidity pool. Such a pool can be created and used by any network participant. Once established on the blockchain, it is immutable. A user can either interact directly with a pool contract or use a routing contract instead. For our analysis we gather pool and routing contract addresses from the decentralized exchange projects UniswapV2, UniswapV3, SushiSwap, Curve and the exchange aggregator 1inch. All UniswapV2, UniswapV3 and SushiSwap pool addresses can be found by scanning all transaction event logs for the respective creation event topic. A UniswapV2 or SushiSwap pool creation event possesses the same topic. Therefore, separating these pools is not straightforward. For this reason, we do not differentiate the two projects in terms of pools. Contract addresses belonging to the 1inch projects, an exchange aggregator that scans decentralized exchanges to find the lowest cryptocurrency prices for traders, are collected from Etherscan [18]. All relevant Curve pools can be found on the project's GitHub page [19].

We also collect addresses of three lending protocol projects, namely Aave, MakerDAO and Compound, open source and non-custodial liquidity protocols for earning interest on deposits and borrowing assets. AaveV1 and AaveV2 contract addresses are taken from their official documentation website [20]. The MakerDAO project maintains a website that makes all active and inactive contract addresses accessible [21]. We collect all addresses that have been active at some point in time. Contract addresses belonging to the Compound project are collected by Etherscan [22].

In addition, we add addresses belonging to the layer-2 rollups Arbitrum and Optimism to the list to determine the ratio of entities already interacting with layer-2. Rollups are Ethereum's short-term scaling solution to increase transaction throughput and a simultaneous reduction in transaction fees. The corresponding addresses can be found on Etherscan [23, 24].

Finally, we are interested in the entity participation in staking on the Ethereum Beacon Chain. By locking 32ETH, participants provide security to the Ethereum network once it moves to its new Proof of Stake consensus mechanism. They earn ETH as a reward and thus have a financial incentive. The Eth2 Deposit Contract can be found on Etherscan [25].

Project	Addresses
Eth2 Deposit Contract	1
Arbitrum	3
Optimism	3
1inch	4
AaveV1	18
Compound	21
AaveV2	36
Curve	102
MakerDAO	387
UniswapV3	6'686
UniswapV2 & SushiSwap	74'143

Table 4.1: Number of contract addresses collected for each project.

4.3.4 Centralized Exchange Addresses

Finally, we create a list of known centralized exchange addresses. An existing list created by Friedhelm [11] in 2018 contains 243 addresses. We extend the list with 191 additional EOA exchange addresses that can be found on Etherscan.

Chapter 5

Address Clustering

Ethereum addresses can be clustered in a variety of ways. In this thesis, we explore a clustering method based on the reuse of exchange deposit addresses. In this chapter, we explain the theory behind this clustering method. We also describe in detail how entity clustering is performed. We start with the datasets from Chapter 4.3 and end with a dataset containing all entities and their user and deposit addresses, as well as the corresponding exchange addresses.

5.1 Deposit Address Reuse

When signing up on a centralized exchange, a user usually only receives one unique Ethereum deposit address to send his funds to. Each time the user makes a deposit to the exchange, the same deposit address is reused. When a user makes deposits to the exchange from multiple personal addresses, a permanent link between them remains in the blockchain. An outside observer looking at all transactions made to the same deposit address can therefore assume that all sender addresses are controlled by the same entity. In a next step, these entities can be merged together if they have used at least one address that is equal. This allows linking entities that use different accounts on the same exchange, and even linking entities across multiple exchanges. This concept is illustrated in Figure 5.1.

5.2 Triplet Extraction

In a first step, all transactions in the transaction dataset discussed in Chapter 4.3.2 are analyzed. We look for traces consisting of two transactions that happened a maximum of 100'000'000 transactions apart. Given an average daily transaction count of about 800'000 [26], similar to Friedhelm [11], we assume that exchanges collect funds sent to a deposit address within approximately four months. Furthermore, the recipient of the second transaction has to be a known

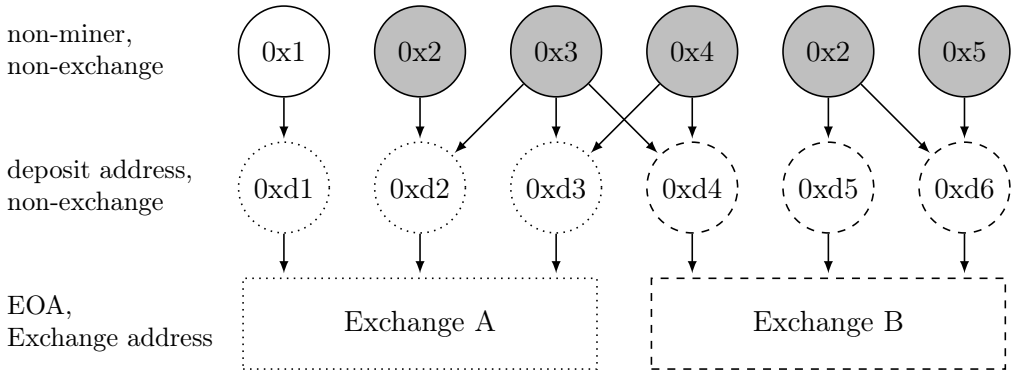

Figure 5.1: If 0xd1 to 0xd6 are forwarding deposit addresses controlled by the two exchanges A and B, we cluster all addresses that use the same deposit addresses and the same address when sending to different deposit addresses. Colors indicate the same entity. We see 4 entities: 2 users [0x1], [0x2 - 0x5] and 2 exchanges with their deposit addresses.

exchange address. It must be part of the exchange dataset mentioned in Chapter 4.3.4. We call each of these traces a triplet. Thus, each triplet consists of a user, a deposit, and an exchange address.

For user and deposit addresses, we only consider addresses that are neither a known exchange nor a miner. Here, we compare each address with the exchange and miner datasets. This minimizes the likelihood of accidentally linking two exchanges. We exclude miner addresses because mining pool participants sometimes request their share to be sent directly to their exchange. This behaviour would cause mining pool addresses to be incorrectly clustered with their participants' deposit addresses.

Two types of transaction tuples are created, each containing two addresses linked by a transaction. Each tuple resembles either a user $\rightarrow$ deposit or a deposit $\rightarrow$ exchange transaction. When looking for said tuples, the above rules are applied. Additionally, it is ensured that a user address is not a deposit address.

In a final step, all transaction tuples are merged into triplets, ensuring that both transactions have occurred within a maximum of 100'000'000 transactions, as previously discussed. All 44'800'605 triplets are combined to a new triplets dataset 5.38GB in size. The merging process is performed using the computer described in Chapter 4.1.

It is important to note that at this point the uniqueness of addresses in the triplet dataset is not given. However, the triplet dataset does not contain triplet duplicates.

5.3 Triplet Merging

In the next phase, the individual triplets are merged into entities with at least two user addresses. In this phase, the deposit addresses play an important role. When looking at all entities, deposit, and user addresses are the only addresses that are likely to be unique in the dataset. The same exchange addresses are likely coupled to multiple different deposit addresses and therefore also user addresses. However, since we want to retain in the final entity dataset the information describing which user address interacted with which exchanges, we choose a dataset structure that does not allow user address uniqueness. Therefore, only the deposit addresses will be unique. This allows us to use deposit addresses as a kind of fixed point for the entire merging process.

We split the merging of triplets to entities into two phases: First, we extract all 10'466'065 unique deposit addresses from the triplets database. Multiple parallel processes are created, in our case 300 processes. Each process receives a unique chunk of unique deposit addresses. All processes simultaneously go through each of their deposit addresses and look for triplets in the triplets dataset that contain a transaction to their deposit address. Once at least 2 and at most 1'000 user addresses are found for a deposit address, they are combined to an entity and stored in a new CSV file. Because of computational resource constrains we use a lower limit of 2 and an upper limit of 1'000 user addresses as proposed by Friedhelm [11]. We perform this first merger on the ETH Euler Cluster [27]. The 300 cores reduce the computation time to approximately 4 hours, instead of 1.5 months when using a single process. Each row in the entity dataset file resembles an entity, each following the same structure. To be able to distinguish the addresses in an entity without consulting a deposit and exchange dataset each time, we introduce the pipe symbol | as a positional reference. It is placed between the deposit and exchange address of an entity. This pipe symbol plays a crucial role after the next merging step. In the following example, *deposit*, *exchange*, and *user* all resemble addresses linked to the same entity, where $user_1$ and $user_2$ are two addresses the entity likely controls.

$$deposit, |, exchange, user_1, user_2, ...$$

Finally, it is possible that entities with the same user address interact with multiple deposit addresses from the same exchange. This means that they are likely to have multiple accounts on the same exchange. It is also possible that users interact with multiple different exchanges. In order to detect such behaviour, the entities created in the previous step can be merged into a larger entity. Here, entities that contain at least one equal user address are being merged. Again, we drop entities that exceed the 1'000 user addresses per entity limit.

The following example illustrates the merger of two entities by concatenating both entities.

$deposit_1, |, exchange_1, user_1, user_2$

$deposit_2, |, exchange_2, user_3, user_2$

The second row is appended to the firs which results in:

$deposit_1, |, exchange_1, user_1, user_2, deposit_2, |, exchange_2, user_3, user_2$

The pipe symbol | can now be used to navigate through the different parts of an entity. All pipe symbols clearly indicate the position of the deposit and exchange addresses.

Unfortunately, parallelizing this processing step results in merging opportunities that are overlooked by the processes working independently. To achieve an optimal result, this step needs to be performed by a single process. This will take several weeks, as the currently implemented algorithm has a time complexity of $O(n^2)$. However, the usage of hash maps could reduce the time complexity to $O(n)$. Neither of these merging algorithms are optimal in terms of runtime, but are rather designed with easy verifiability in mind. Both algorithms should be seen as a reference for future optimizations.

Chapter 6

Results

All in all, we find 44'800'605 triplets among the 1'559'436'843 transactions, which we can merge to 1'410'523 entities with at least 2 and at most 1'000 user addresses. In a next step, we analyze the clustered entities themselves. Finally, we investigate their behaviour against the projects introduced in Chapter 4.3.3. For all further calculations, we use block 14'699'999 as a reference point.

6.1 Entity Analysis

We can see that all 1'410'523 entities collectively own 6'788'215 addresses, which is approximately 3.48% of all 195'011'000 unique addresses ever used on Ethereum, including internally owned addresses [28]. In addition, all entities collectively deposit to a total of 1'553'060 deposit addresses, representing approximately 0.8% of all addresses. Out of the 434 exchange addresses we track, only 284 are used by our entities. We assume the rest are used internally by exchanges and are listed on Etherscan for transparency reasons. As a result, an average entity owns 4.81 user addresses and deposits funds on 1.1 deposit addresses on 1.09 exchange addresses. For user addresses, the median is 3.0, suggesting that a few large entities are skewing the picture. All results discussed above are summed up in Table 6.1.

	Count	Average Entity	Of all Addresses
Entities	1'410'523	-	-
User Addresses	6'788'215	4.81	3.48%
Deposit Addresses	1'553'060	1.1	0.8%
Exchange Addresses	284	1.09	-

Table 6.1: This table shows the total number of entities, their user addresses, interacting deposit addresses and unique exchange addresses. The number of addresses belonging to an average entity, and finally a comparison of the combined addresses with all active addresses in the network up to block 14'699'999.

Furthermore, if we count the 284 exchanges and their 10'466'065 unique deposit addresses as clustered entities we end up with 1'410'807 entities which collectively own 17'254'280 addresses. They account for approximately 8.8% of all unique addresses ever used on Ethereum, including internally owned addresses. Unfortunately, the total number of existing smart contracts is not easy to determine. However, according to Schoellen [29] there existed 48'083'448 smart contract addresses at block 14'000'000. If we take this value as a lower bound, we know for certain that we cluster at least 11.7% of all unique EOA addresses.

6.1.1 Address Distributions

Looking at the distributions of user, deposit, and exchange address on a log-log scale, we can clearly see a linear decrease in the number of entities that have multiple addresses and users that have several deposit and exchange addresses. We find a power-law behaviour with an exponent of approximately -2.7, -3.6, and -4.2 for distributions of user, deposit, and exchange addresses. In Figure 6.1 the x-axis ends at 1'000 addresses because we set an upper bound in Chapter 5.3.

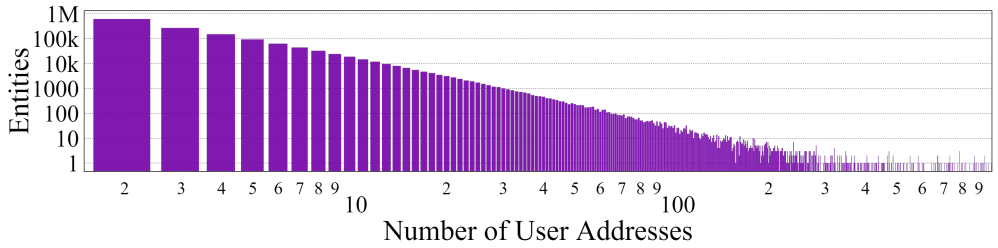

Figure 6.1: This figure shows how many entities control X number of user addresses, where 43.6% of all entities control 2, 19% control 3, and 10.6% control 4 user addresses.

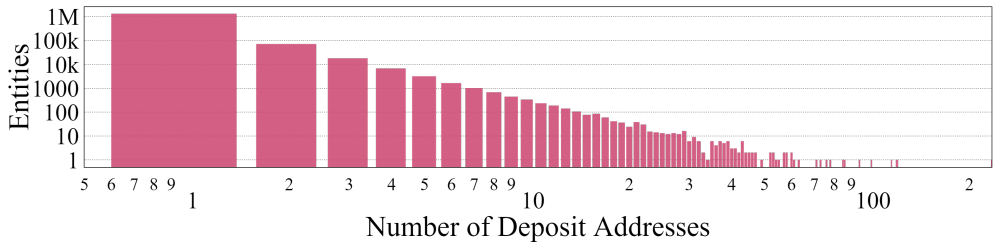

Figure 6.2: In this figure we see to how many deposit addresses an entity sends funds, where 92% of all entities deposit to 1, 5% to 2, and 1.3% deposit to 3 deposit addresses.

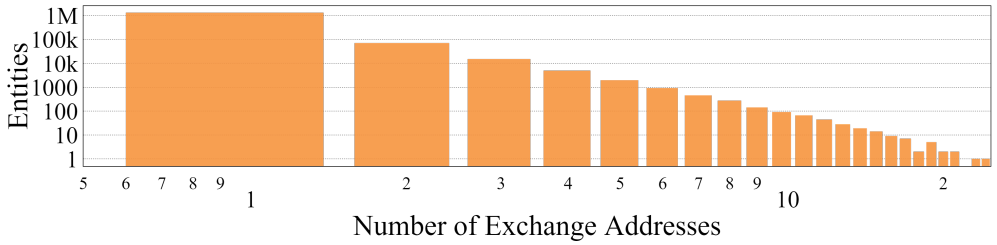

Figure 6.3: This figure shows the number of exchange addresses an entity interacts with, where 93.3% of all entities interact with 1, 5% interact with 2, and only 1.1% interact with 3 different exchange addresses. We find one entity that interacts with 24 exchange addresses.

It is important to note that Figure 6.3 does not specify the number of unique exchanges that an entity interacts with, but rather the number of unique exchange addresses. In the next chapter we further analyze exchange addresses and the number of deposit addresses that interact with exchanges.

6.1.2 Popular Exchanges

In total, we identify 10'466'065 unique deposit addresses. In Figure 6.4 we illustrate the 10 exchange addresses that we can link to the most unique deposit addresses. When we combine all exchange addresses that belong to the same exchange, we see that Binance owns 23.8% of all detected deposit addresses, followed by ShapeShift and Coinbase with a 16.8% and 8.5% share respectively.

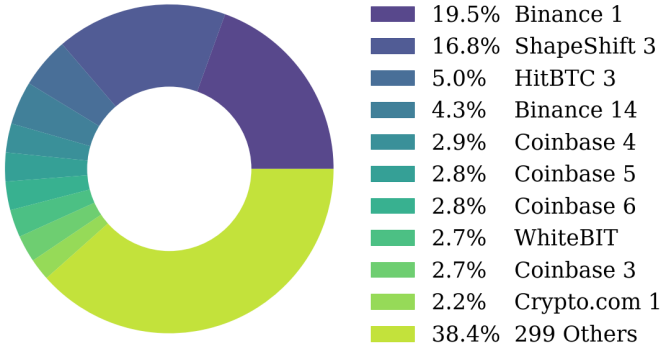

Figure 6.4: Illustrates the 10 exchange addresses which we can link to the most unique deposit addresses.

In a next step, we again rank all exchange addresses with the most linked deposit addresses, but this time we only count deposit addresses owned by our entities with 2 to 1'000 user addresses. As we can see in Figure 6.5, they prefer a different set of exchanges. Out of the 10 exchange addresses, 6 are new addresses. Again, and more clearly this time, Binance tops the list with 59.2% of all deposit addresses owned by entities, followed by two new exchanges, KuCoin and Bitfinex with 7.4% and 2.4% respectively.

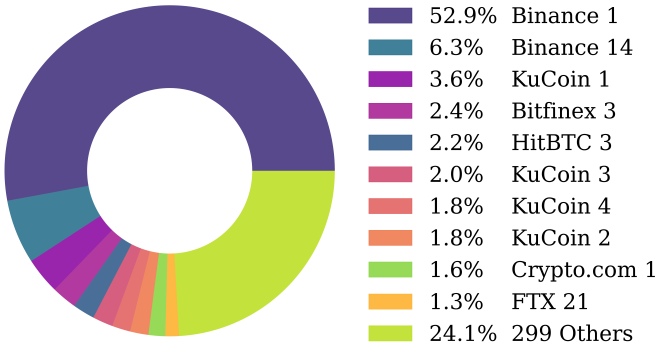

Figure 6.5: Illustrates the top 10 exchange addresses by linked deposit addresses, where all deposit addresses are owned by entities with 2 to 1'000 user addresses.

We conclude that entities which have 1 or more than 1'000 addresses are more likely to use Coinbase and ShapeShift, while Binance, KuCoin, and Bitfinex are used less by said entities. Our findings reassemble the top five exchanges by daily trading volume at the time of writing [30], Binance, Coinbase, FTX, Kraken, and KuCoin. As discussed in Chapter 3.2.2, Kraken uses smart contracts instead of regular deposit addresses and is therefore not visible in Figures 6.4 and 6.5. ShapeShifts presence in Figure 6.4 is likely because of its popularity in the early years of Ethereum.

6.1.3 Entity Activity and Entity Age

Next, we look for the block numbers at which all entities made their first and last transactions. Using this data, we visualize the number of entities that emerge or vanish at any given time in Figure 6.6. For reference: 1'000'000 blocks are added to the blockchain in approximately 10 months.

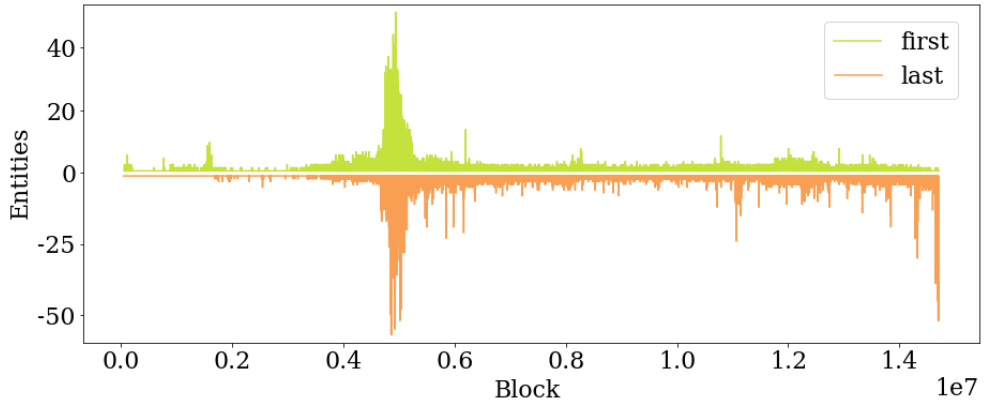

Figure 6.6: This figure shows the block heights at which entities perform their first (green) and last (orange) transactions. The peaks around block 5'000'000 occurred during the bull market in late 2017, early 2018. The orange peak on the far right originates from entities that have been active in recent weeks.

Subtracting the corresponding block numbers results in an entity age, denominated in number of blocks. We measure an average entity age of 1'998'054 blocks, which corresponds to about 307 days, and a median of 589'604 blocks, which is approximately 90 days. Figure 6.7 illustrates the age distribution of all entities. Note, because of the log scale on the x-axis, the average age is not easy to derive.

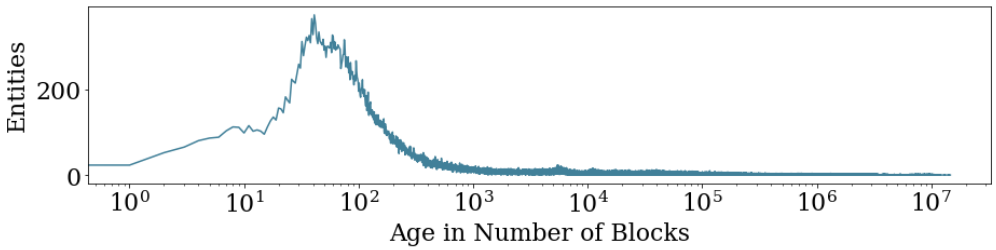

Figure 6.7: The age distribution of all entities on a logarithmic scale reveals an accumulation around the age of approximately 45 blocks, or 10 minutes. Furthermore, 3% of all entities are active for only less than one hour, 7.4% for less than a day, and 18.6% for less than one week. However, the average age is 1'998'054 blocks, which is not easy to see because of the log scale.

6.1.4 Entity Funds

We are also interested in the total funds an entity controls. We add up all funds held in an entity's user addresses around block 14'924'000. Only the six most frequently used currencies, ETH, WETH, WBTC, DAI, USDT, and USDC are considered and converted to ETH.

On average, an entity controls 5.0ETH. The median is practically 0. However, if we ignore all addresses that contain 0 funds, we observe an average of 13.2ETH and a median of 0.02ETH. We assume that the reasons for the discrepancy between the average and the median are a few large holders and many addresses with near-zero balances, coupled with high gas fees, that make it uneconomical to accumulate all balances at a single address. Most of the wealth is held by a minority of entities. The richest entity we detect owns 289'000ETH. However, we must keep in mind that entities can also be companies and institutions.

We close this chapter with Figure 6.8, which relates all the metrics discussed in the previous chapters.

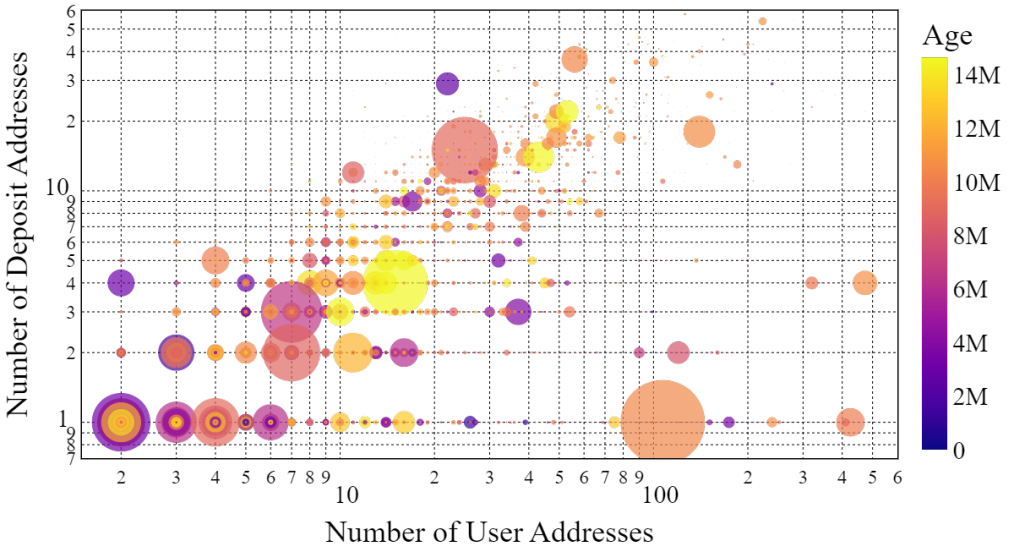

Figure 6.8: In this figure we relate an entities size, resembled in the number of user and deposit addresses, as well as the age of entity and the amount of funds. Fund size correlates proportionally with bubble size. This is a slightly zoomed-in view. A handful of small entities at the top of both axes are not visible.

We conclude, entities with multiple user addresses are more likely to also interact with multiple deposit addresses. Apart from a few exceptions, many wealthy entities with more than 10 user addresses tend to be older than 8 million blocks. In combination with Figure 6.10 we see that many young entities only interact with one deposit address.

6.2 DeFi, Layer-2 and Eth2 Interaction Analysis

In this section, we show our results regarding the interaction of entities with smart contracts introduced in Chapter 4.3.3. In total, we surface and analyze 15'400'116 interactions of 231'712 entities. We can thus say that 16.4% of all entities interacted with DeFi at some point. Their interactions are responsible for 1% of all transactions on Ethereum. An average entity makes 66.4 transactions to a DeFi smart contract. The median is 11.0 transactions. The entity with most interactions performed a total of 153'228 transactions.

6.2.1 DeFi Behaviour Analysis

First, we focus exclusively on DeFi. Figure 6.9 visualizes a comparison between all DeFi projects in the smart contract dataset in terms of the number of transactions sent by entities to the smart contracts. We detect a total of 11'112'068 interactions with the UniswapV2 routing contract, representing 73.6% of all interactions. It is important to note that UniswapV2 was launched in May 2020 and has a head start of approximately one year compared to UniswapV3. Moreover, it benefited from favourable market conditions in its first year. As mentioned in Chapter 4.3.3, we do not differentiate between UniswapV2 and SushiSwap pools.

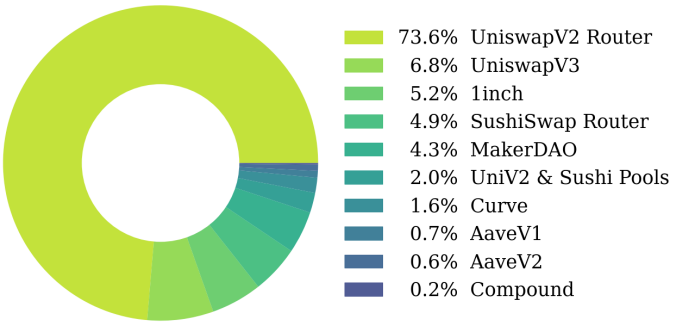

Figure 6.9: A comparison of all 9 DeFi projects in terms of entity-initiated interactions.

The dominance of UniswapV2 is expected, yet impressive. At the time of writing, the ratios between the DEXs UniswapV2, UniswapV3 and SushiSwap closely match the ratios of all time DEX activity on Etherscan [31]. We can further deduce that about 8.0% of the total trading activity on UniswapV2 stems from our clustered entities.

Except for the y-axis, Figure 6.10 is similar to Figure 6.8. It illustrates the relationships between entity age, size, funds, and the number of transfers to DeFi projects. It shows us that there is no clear trend in terms of age, assets and the number of DeFi project interactions. However, a differentiation between old and new entities with regards to the number of user addresses is more pronounced than in Figure 6.8.

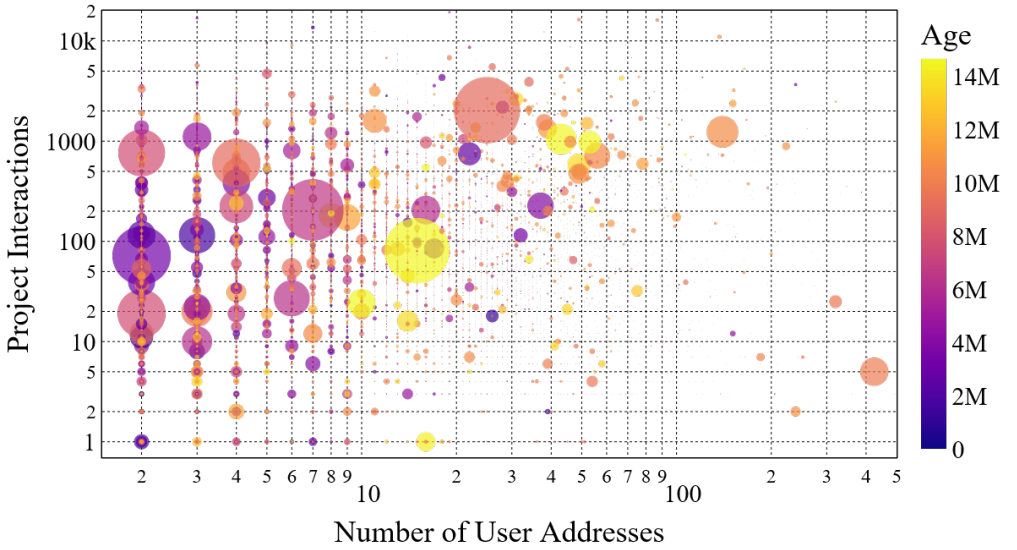

Figure 6.10: This figure is similar to Figure 6.8, but with the number of interactions with DeFi projects on the y-axis. With regards to age, assets, and the number of DeFi project interactions no clear trend is observable. This is a slightly enlarged view, some small entities are not visible. Additional plots, considering each DeFi project individually, can be found in the Appendix A.1 and A.2.

Going one step further, we also analyze the nature of the interactions that entities have with the DeFi protocols. For this, we examine all log topics that specify the transaction between entity and smart contract. The ratio of all topic occurrences is visible in Figure 6.11. We find a total of 12'847'152 *swap* topics.

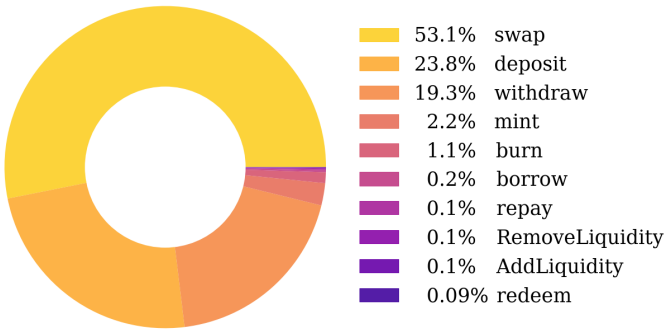

Figure 6.11: A comparison of the occurrences of individual topics in transactions between entities and DeFi projects.

The logs of a transaction are likely to contain multiple topics. We use all topics of the transaction for our comparison, except for duplicates. This means that the sum of all displayed tropics is greater than all DeFi transactions. Additionally, but not visible in Figure 6.11, we find 327 liquidation calls initiated by our entities.

6.2.2 Layer-2 Interaction

With respect to Ethereum's layer-2 rollups Arbitrum and Optimism, we mainly track bridging activity. Overall, we can say that 2.6% of all entities have used Arbitrum or Optimism at least once. A detailed comparison of the two rollups can be seen in Table 6.2.

	Arbitrum	Optimism
Total Interactions	63'104	25'775
Total Entities	22'509	13'741
Entity Interactions Average	2.8	1.9
Entity Interactions Median	2.0	1.0

Table 6.2: This table shows a comparison of the two layer-2 rollups Arbitrum and Optimism in terms of entity bridging activity.

We assume that the low average number transactions per entity is due to users trying to avoid moving funds repeatedly between the base chain and layer-2 because of the high gas fees. However, at the time of writing, both rollups had only been in operation for approximately 9 months, which likely also contributes to the low average interactions. For a further activity comparison, two figures similar to Figure 6.10, but referring exclusive to the Arbitrum and Optimism rollups can be found in Appendix A.3.

6.2.3 Eth2 Beacon Chain Staking

We find a total of 16'900 staking events performed by our entities initiated by 754 distinct entities. This means that only 0.05% of entities participate in solo staking on the Beacon Chain. The five entities that have made the most deposits are responsible for 13'230 of the 16'900 deposits. Furthermore, we can identify the entity which accounts for 6'000 deposits as part of Binance. Each deposit consists of exactly 32ETH, which results in a total of 540'800ETH staked. At the time of data retrieval, which happened at block 14'850'043, approximately 12'700'000ETH were staked in the contract [32]. Therefore, all entities contribute 4.3% of the entire staked ETH amount. If we ignore all Binance deposits, we end up with 2.8%. It is important to note that we only track solo staking. Many users prefer to stake their funds via a staking protocol such as Lido or Rocket Pool.

Chapter 7

Conclusion

In this thesis, we enhanced an existing method for clustering entities on Ethereum based on reused exchange deposit addresses. Moreover, we merge entities that do not share a common transaction with the same deposit address. Overall, we are able to cluster 8.8% of all unique addresses ever used on Ethereum, including internally owned addresses. We can identify 1'410'523 entities that control at least 2 and at most 1'000 addresses.

In addition, we analyzed entity properties, such as their size, age, and total funds held. We observed that an average entity owns 4.81 user addresses, is active for 307 days, and holds 5.0ETH in all its addresses.

Finally, we examined their interaction and behavior towards popular DeFi projects as well as Layer-2 rollups and Eth2 staking on the Beacon Chain. We concluded that 16.4% of all entities interacted with DeFi projects and that 2.6% bridged at least once onto Layer-2 rollups. Only 0.05% of said entities participated in solo-staking on the Ethereum Beacon Chain.

7.1 Future Work

Deposit address reuse is not the only applicable clustering method. Various other heuristics, such as multiple participation in airdrops, can be used to cluster additional entities. The size of the Ethereum blockchain is growing daily. Optimizing parts of the merging algorithms could simplify the process and future-proof this method. Other usage patterns related to NFTs and on-chain governance can be studied.

We believe it as likely that user activity will shift more and more from the Ethereum base chain to its layer-2 rollups. However, getting the relevant transaction data from future rollups is likely to be more difficult and might increase clustering complexity. Account abstraction, a topic currently discussed in the Ethereum community, which allows a contract to be the top-level account that pays the fees and initiates the execution of transactions, could further complicate entity clustering.

Bibliography

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, 2008.

[2] V. Buterin, "Ethereum white paper: A next generation smart contract & decentralized application platform," 2013. [Online]. Available: https://github.com/ethereum/wiki/wiki/White-Paper

[3] "Cryptocurrency market caps," 2022. [Online]. Available: https://coinmarketcap.com/charts/

[4] "Daily active addresses on ethereum," 2022. [Online]. Available: https://etherscan.io/chart/active-address

[5] M. Jourdan, S. Blandin, L. Wynter, and P. Deshpande, "Characterizing entities in the bitcoin blockchain," 2018. [Online]. Available: https://arxiv.org/abs/1810.11956

[6] D. Ron and A. Shamir, "Quantitative analysis of the full bitcoin transaction graph," pp. 6–24, 2013.

[7] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. M. Voelker, and S. Savage, "A fistful of bitcoins: Characterizing payments among men with no names," in *Proceedings of the 2013 Conference on Internet Measurement Conference*. New York, NY, USA: Association for Computing Machinery, 2013, p. 127–140. [Online]. Available: https://doi.org/10.1145/2504730.2504747

[8] T. Chen, Z. Li, Y. Zhu, J. Chen, X. Luo, J. C.-S. Lui, X. Lin, and X. Zhang, "Understanding ethereum via graph analysis," *ACM Trans. Internet Technol.*, vol. 20, no. 2, apr 2020. [Online]. Available: https://doi.org/10.1145/3381036

[9] S. Ferretti and G. D'Angelo, "On the ethereum blockchain structure: A complex networks theory perspective," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 12, aug 2019. [Online]. Available: https://doi.org/10.1002%2Fcpe.5493

[10] G. Bonifazi, E. Corradini, D. Ursino, and L. Virgili, "Defining user spectra to classify ethereum users based on their behavior," *Journal of Big Data*, 2022. [Online]. Available: https://doi.org/10.1186/s40537-022-00586-3r

[11] V. Friedhelm, "Address clustering heuristics for ethereum," p. 17, 2018.

[12] L. Heimbach, Y. Wang, and R. Wattenhofer, "Behavior of liquidity providers in decentralized exchanges," 2021. [Online]. Available: https://arxiv.org/abs/2105.13822

[13] "Etherscan," 2022. [Online]. Available: https://etherscan.io/

[14] "Erigon client implementation," 2022. [Online]. Available: https://github.com/ledgerwatch/erigon

[15] "Web3.py," 2022. [Online]. Available: https://github.com/ethereum/web3.py

[16] "Thesis code on gitlab," 2022. [Online]. Available: https://gitlab.ethz.ch/disco-students/fs22/grdomini_clustering_ethereum_addresses

[17] "Totoal transaction count on blockchair," 2022. [Online]. Available: https://blockchair.com/ethereum/charts/total-transaction-count

[18] "1inch exchange contract addresses on etherscan," 2022. [Online]. Available: https://etherscan.io/accounts/label/1inch-exchange

[19] "Curve.fi pools on github," 2022. [Online]. Available: https://github.com/curvefi/curve-contract/tree/master/contracts/pools

[20] "Aave contracts," 2022. [Online]. Available: https://docs.aave.com/developers/v/2.0/deployed-contracts

[21] "Makerdao contracts," 2022. [Online]. Available: https://chainlog.makerdao.com/api.html

[22] "Compound contract addresses on etherscan," 2022. [Online]. Available: https://etherscan.io/accounts/label/compound

[23] "Arbitrum contract addresses on etherscan," 2022. [Online]. Available: https://etherscan.io/accounts/label/arbitrum

[24] "Optimism contract addresses on etherscan," 2022. [Online]. Available: https://etherscan.io/accounts/label/optimism

[25] "Eth2 deposit contract on etherscan," 2022. [Online]. Available: https://etherscan.io/address/0x00000000219ab540356cbb839cbe05303d7705fa

[26] "Ethereum transactions per day," 2022. [Online]. Available: https://ycharts.com/indicators/ethereum_transactions_per_day

[27] "Eth euler cluster," 2022. [Online]. Available: https://scicomp.ethz.ch/wiki/Euler

[28] "Unique addresses chart on etherscan," 2022. [Online]. Available: https://etherscan.io/chart/address

[29] Schoellen, "Mining and visualizing ethereum," 2022.

[30] "Coinmarketcap exchange ranking," 2022. [Online]. Available: https://coinmarketcap.com/rankings/exchanges/

[31] "Dex activity chart on etherscan," 2022. [Online]. Available: https://etherscan.io/stat/dextracker?range=1

[32] "Staket eth chart on beacon chain," 2022. [Online]. Available: https://beaconcha.in/charts/staked_ether

Appendix A

Individual Project Activities

We show some additional scatter plots similar to Figure 6.10 but for each DeFi project and layer-2 rollup individually. Entities only change their position on the y-axis, which makes tracking of single entities across multiple projects easier. All figures have the same range on both x and y-axis. Because of that some small entities can be cut off.

A.1 Decentralized Exchanges

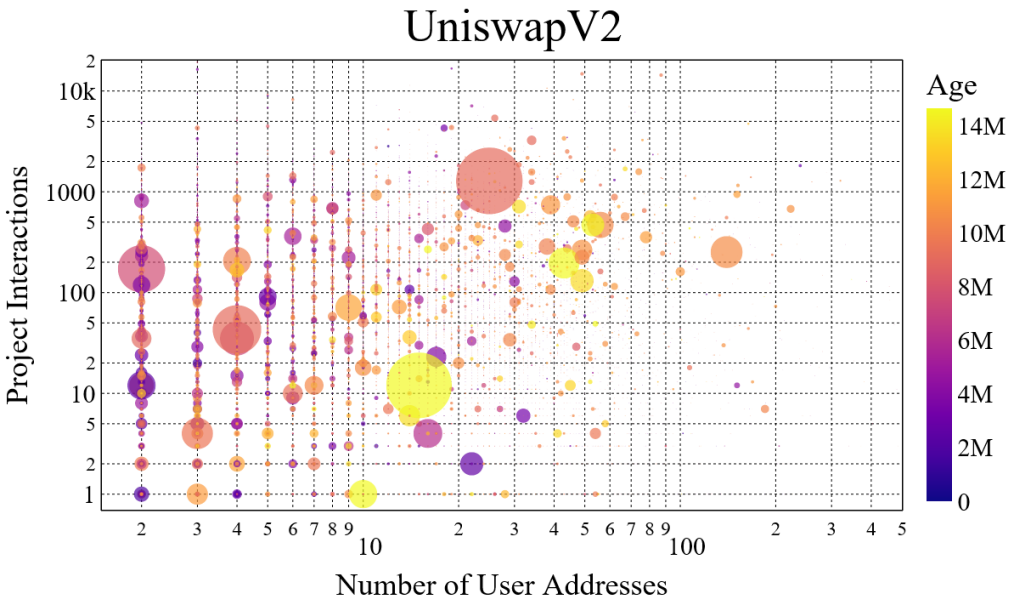

Figure A.1: UniswapV2 entity interaction activity.

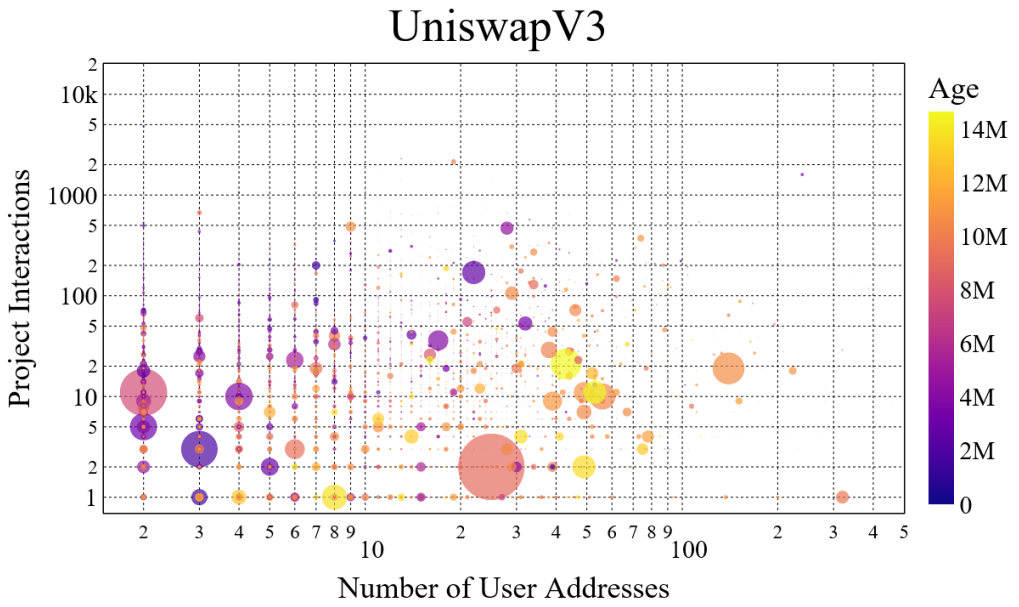

Figure A.2: UniswapV3 entity interaction activity.

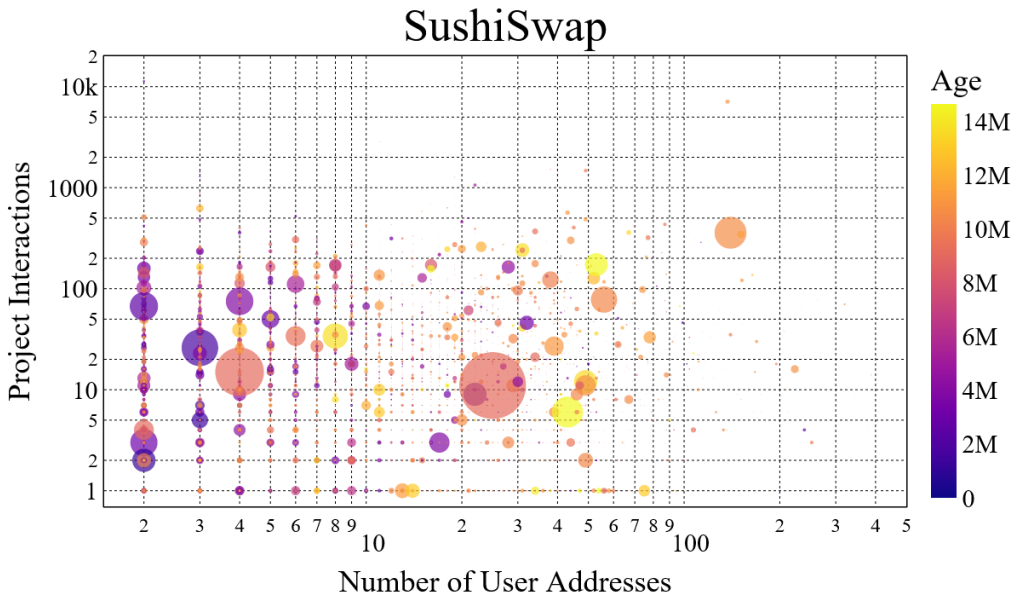

Figure A.3: SushiSwap entity interaction activity.

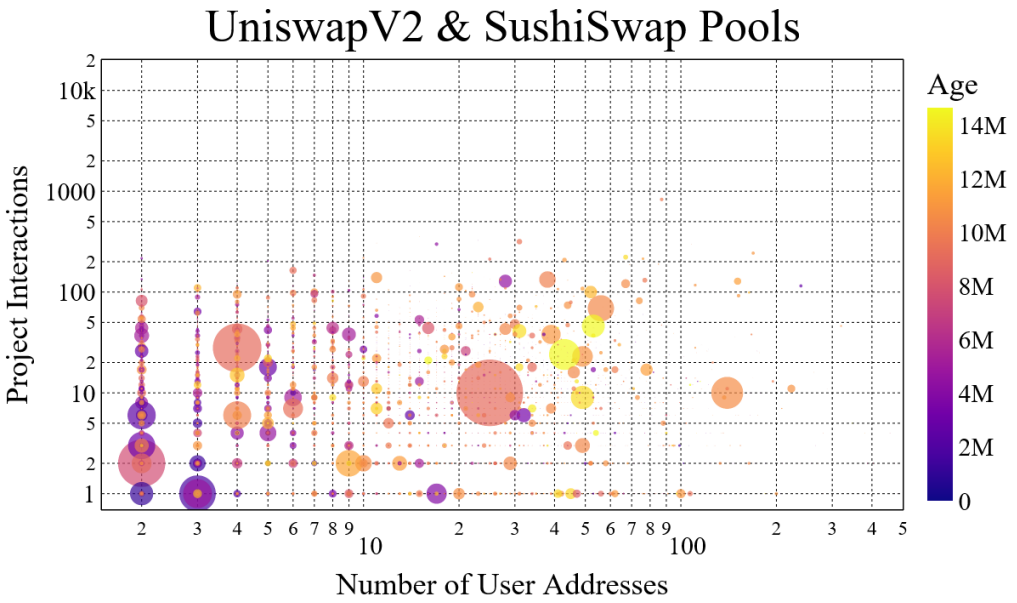

Figure A.4: UniswapV2 and SushiSwap pool entity interaction activity.

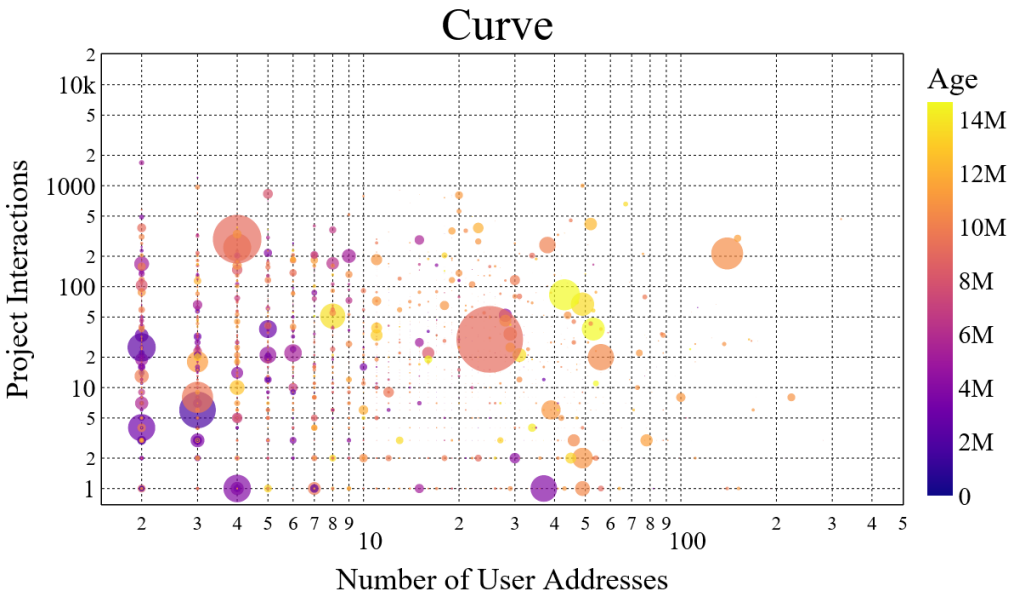

Figure A.5: Curve entity interaction activity.

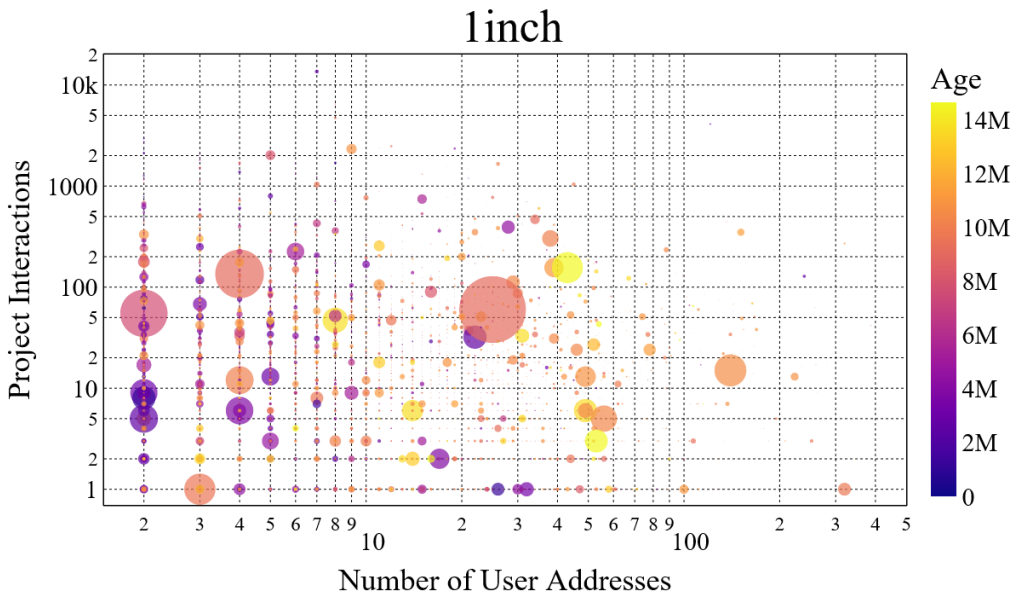

Figure A.6: 1inch entity interaction activity.

A.2 Lending Protocols

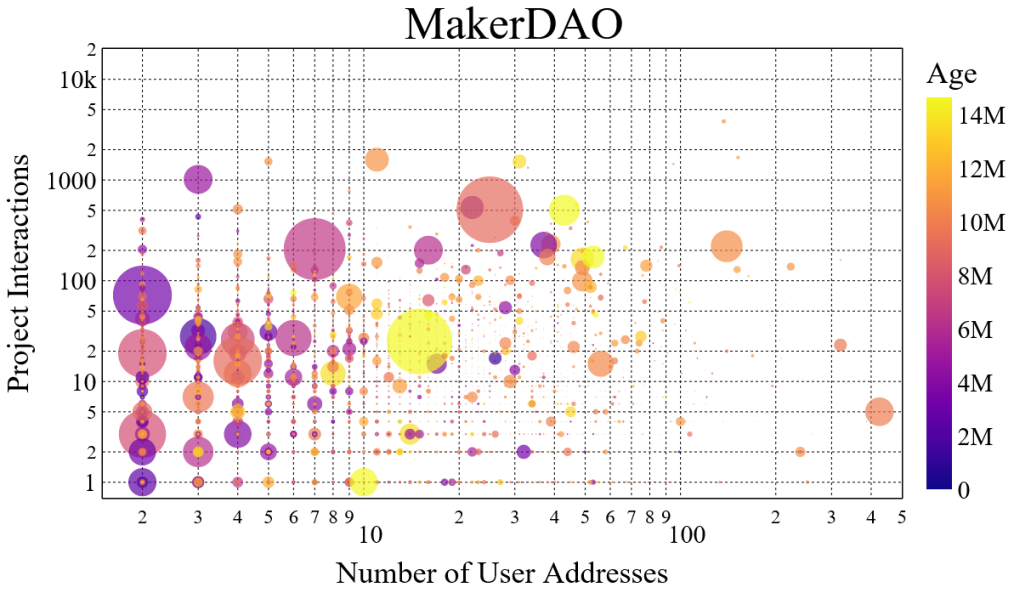

Figure A.7: MakerDAO entity interaction activity.

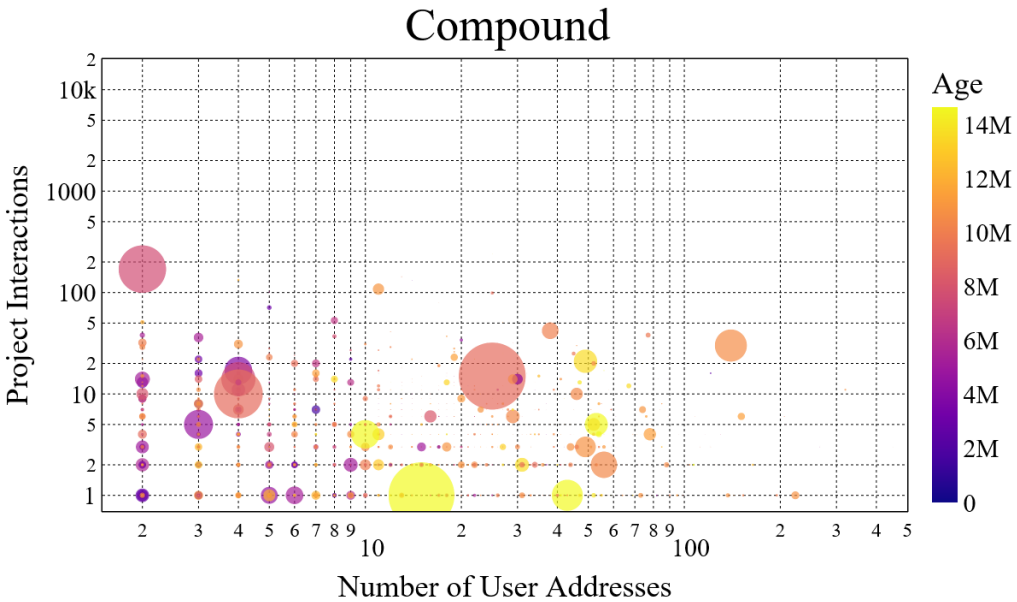

Figure A.8: Compound entity interaction activity.

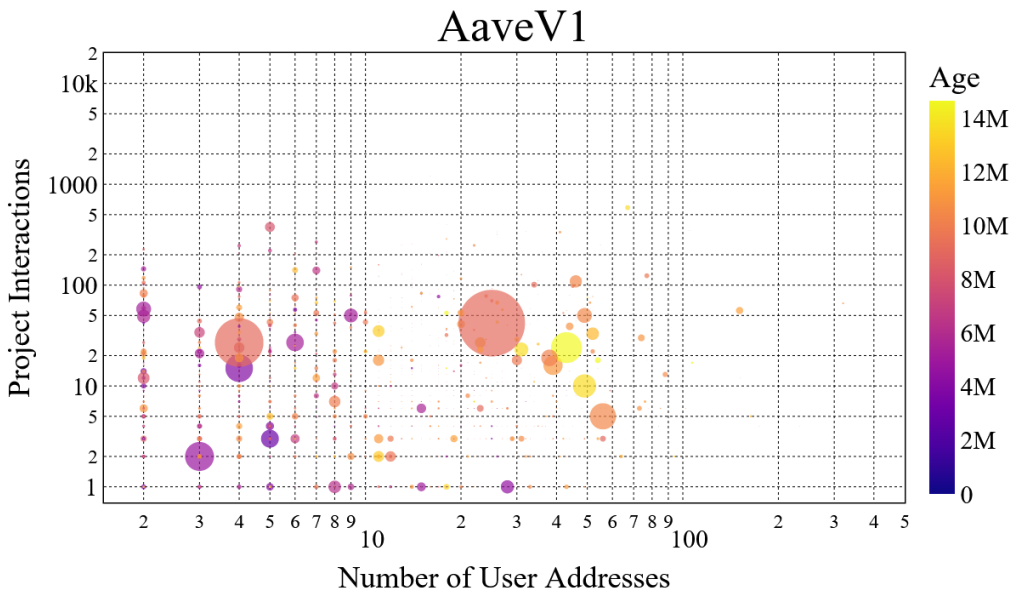

Figure A.9: AaveV1 entity interaction activity.

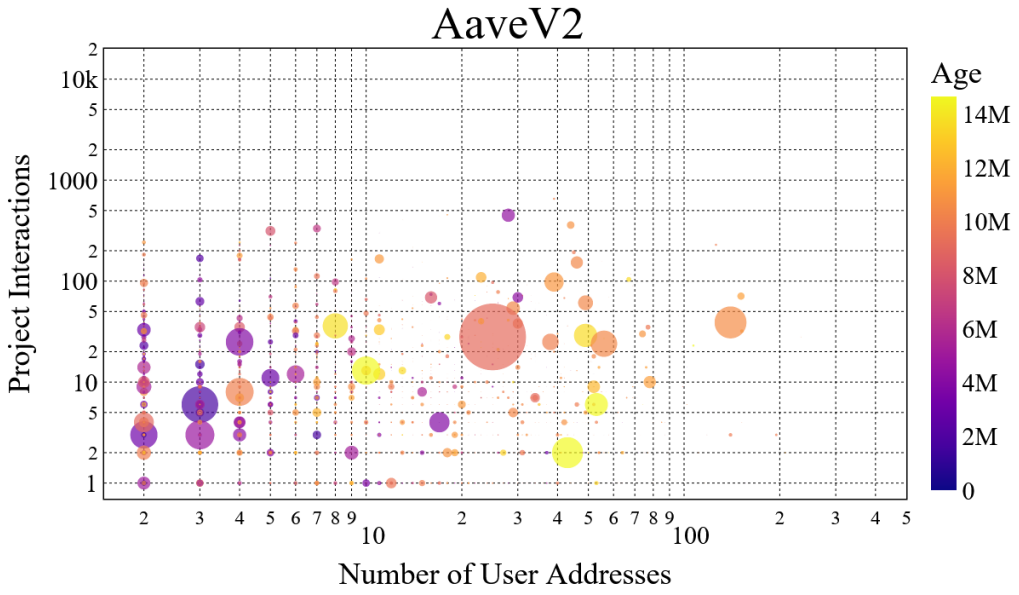

Figure A.10: AaveV2 entity interaction activity.

A.3 Layer-2

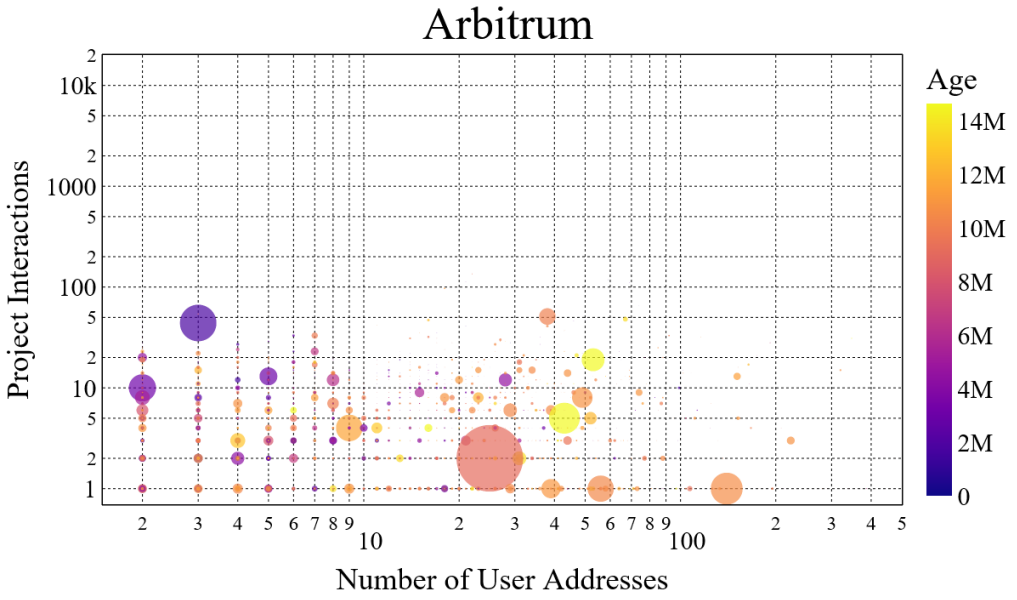

Figure A.11: Arbitrum entity interaction activity.

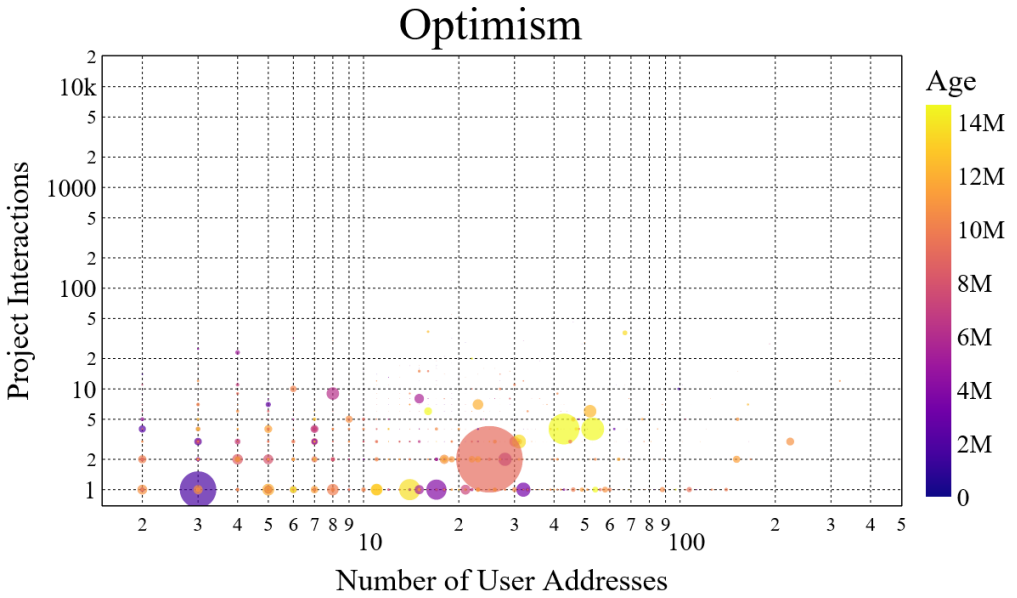

Figure A.12: Optimism entity interaction activity.