Know Your Transactions: Real-time and Generic Transaction Semantic Representation on Blockchain & Web3 Ecosystem

Zhiying Wu[1], Jieli Liu[1], Jiajing Wu[1,*], Zibin Zheng[1], Xiapu Luo[2], Ting Chen[3]
[1]Sun Yat-sen University, [2]Hong Kong Polytechnic University, [3]University of Electronic Science and Technology of China
{wuzhy95,liujli7}@mail2.sysu.edu.cn,{wujiajing,zhzibin}@mail.sysu.edu.cn
daniel.xiapu.luo@polyu.edu.hk,brokendragon@uestc.edu.cn

ABSTRACT

Web3, based on blockchain technology, is the evolving next generation Internet of value. Massive active applications on Web3, e.g. DeFi and NFT, usually rely on blockchain transactions to achieve value transfer as well as complex and diverse custom logic and intentions. Various risky or illegal behaviors such as financial fraud, hacking, money laundering are currently rampant in the blockchain ecosystem, and it is thus important to understand the intent behind the pseudonymous transactions. To reveal the intent of transactions, much effort has been devoted to extracting some particular transaction semantics through specific expert experiences. However, the limitations of existing methods in terms of effectiveness and generalization make it difficult to extract diverse transaction semantics in the rapidly growing and evolving Web3 ecosystem. In this paper, we propose the **Mo**tif-based **T**ransaction **S**emantics representation method (**MoTS**), which can capture the transaction semantic information in the real-time transaction data workflow. To the best of our knowledge, MoTS is the first general semantic extraction method in Web3 blockchain ecosystem. Experimental results show that MoTS can effectively distinguish different transaction semantics in real-time, and can be used for various downstream tasks, giving new insights to understand the Web3 blockchain ecosystem. Our codes are available at https://github.com/wuzhy1ng/MoTS.

CCS CONCEPTS

• **Applied computing → Digital cash**; • **Computing methodologies → Semantic networks**.

KEYWORDS

Blockchain, Extract-Transform-Load, semantic extraction, network motif, financial data mining, Web3

ACM Reference Format:

Zhiying Wu, Jieli Liu, Jiajing Wu, Zibin Zheng, Xiapu Luo, Ting Chen. 2023. Know Your Transactions: Real-time and Generic Transaction Semantic Representation on Blockchain & Web3 Ecosystem. In *Proceedings of the ACM Web Conference 2023 (WWW'23), May 1–5, 2023, Austin, TX, USA*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3543507.3583537

*Jiajing Wu is corresponding author.

WWW '23, May 1–5, 2023, Austin, TX, USA

ACM ISBN 978-1-4503-9416-1/23/04…$15.00
https://doi.org/10.1145/3543507.3583537

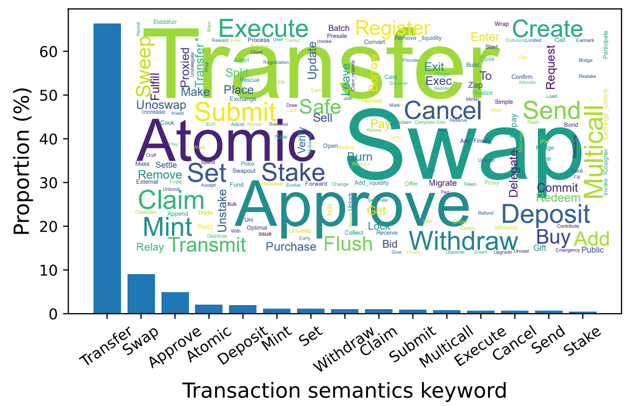


Figure 1: Proportion and keyword cloud of transaction semantics on Ethereum in 2021. More than 30% of transactions in Ethereum purpose to *Swap*, *Deposit*, *Mint*, etc., rather than *Transfer* money.

1 INTRODUCTION

In recent years, Web3 has attracted a lot of attention from industry and academia as a distributed Internet of value without trusted third parties. Massive Web3 applications, e.g. Decentralized Finance (DeFi) and Non-Fungible Tokens (NFT), are emerging and are inextricably linked to the support of blockchains [8, 17], which creates a decentralized and trusted trading environment for transactions of various cryptocurrencies including Bitcoin [15]. Particularly, Ethereum [32] is a typical account-based blockchain with smart contract functionality, being able to encode arbitrary computation logic related to Web3. To trigger the specific logic on Ethereum, users can launch transactions from their accounts to smart contracts. With the rise of Web3 applications, the logic of smart contracts has become increasingly complex, allowing transactions to achieve a variety of purposes. Thus, transactions on Ethereum may carry various transaction semantics depending on the users' intent or the smart contracts being invoked. Before 2019, more than 90% of transactions were made with the purpose of money *Transfer* [42]. While according to a report provided by a famous Ethereum browser called Etherscan [24], as shown in Figure 1, transaction semantics have become rather diverse in recent years and more than 30% of transactions are involved in businesses other than money *Transfer* in 2021. These phenomena indicate that blockchain-based platforms like Ethereum have evolved from cryptocurrency platforms to distributed Web3 application systems.

In traditional financial systems like banks and stock markets, the account information is provided by the customers and verified by managers to conduct the Know-Your-Customer (KYC) process, the purpose being market regulation and risk prevention. However, due to the pseudonymity of blockchain, we usually cannot obtain the identity and registration information of accounts, to infer the intent of related transactions, as traditional trading systems do. Therefore, it is important to understand the intent behind the transactions for the healthy development of the Web3 blockchain ecosystems. In prior work, some researchers recognized the price manipulation semantics [30, 36, 44] and help detect the flash-loan attack transactions, which have caused a total loss more than $364M in 2021 [22]. By identifying the mixing semantics, [35, 39] traced the money flow for the goal of anti-money laundering. Characterizing the money transferring semantics also helps researchers detect phishing scams [4, 34, 38]. In summary, existing work has presented transaction semantic analysis from different perspectives and validated its significant value for understanding various types of behaviors on blockchain [30, 36, 42, 44].

However, the efficiency and generalization of existing transaction semantic extraction methods are still rather limited. On the one hand, semantic extraction relies on the Extraction, Transformation, and Loading (aka. ETL) procedure of the collected blockchain raw data [5, 12, 42]. Yet the efficiency of the current lightweight ETL methods is limited due to the partial support of parallel data acquisition. In addition, the cost of the heavyweight ETL approaches is exceedingly expensive. On the other hand, most existing methods [30, 36, 37, 44] of transaction semantic extraction are proposed based on expert rules of some specific cases, thus lacking universality in the evolving Web3 blockchain ecosystem.

As a result, towards universal applications on large-scale blockchain data, extracting transaction semantics is still a challenging task. ***First***, the extraction procedure must be efficient enough to meet real-time demands. On blockchain trading systems, new transactions and smart contract logic are constantly and rapidly emerging, making it difficult for slow semantic extraction to be of value to downstream tasks or applications. For example, when it comes to anti-money laundering, the ability to access the hidden semantics within the latest transactions determines whether the laundering process can be intercepted timely. ***Second***, the extraction method should be adaptable to multiple transaction semantics, but most of the existing methods focus on a specific kind of semantics. As Web3 blockchain ecosystems grow in popularity, new semantics emerge, and a more general and universal semantics detection method can better meet the needs of practical applications.

In this paper, we propose the Motif-based Transaction Semantics representation method (MoTS) to capture the transaction semantic information in the real-time transaction data workflow and give a demonstration of MoTS in Ethereum. Figure 2 shows the framework of this paper. First of all, we propose a lightweight and high-performance ETL tool to collect the transaction data on account-based blockchains (i.e. Transaction data acquisition). Next, we represent the money transfer relationships as a network and utilize network motifs [1], which are high-order network organizations, to extract hidden transaction semantic information. This

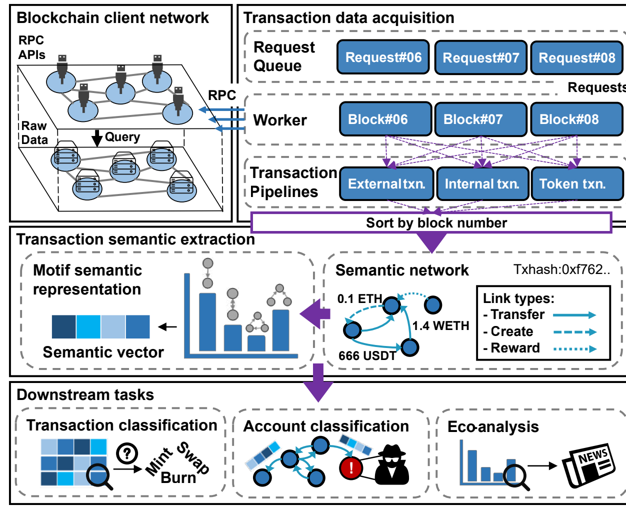


Figure 2: The framework of our work. All purple arrows indicate the data flow in MoTS.

information is then represented as general semantic vectors (i.e. Transaction semantic extraction) and used as additional features to improve the performance of downstream models. The experiments demonstrate that our method can extract transaction semantics in real-time, and capture the various transaction semantics with sufficient discrimination. Although our tool is built for Ethereum, it can be applied in many other account-based blockchains e.g. BNB chain [2], Polygon [23], and Avalanche [19]. In summary, the major contributions are as follows:

- **Data acquisition**. We propose a lightweight and high-performance RPC-based[1] blockchain data ETL tool, with a speed of more than 10 times of similar tools in the same machine configuration. Table 1 shows the differences between our tool and similar methods.
- **Semantic extraction**. We offer an efficient and generic transaction semantic representation approach in real-time ETL workflow based on the network motifs. Experiments indicate that this representation is capable of discriminating between various transaction semantics.
- **Multi-dimension analysis**. We verify the effectiveness of transaction semantic representation in multiple downstream tasks, e.g. transaction classification and account classification. In addition, from the perspective of the network motifs, we analyze the evolutionary trend of transaction semantics in 2021 and identify the boom of NFT.

The rest of this paper is organized as follows. Section 2 reviews the related work. Section 3 proposes our approach. Section 4 evaluates our approach in multiple experiments.

2 RELATED WORK

In this section, we discuss related work on transaction data acquisition, transaction semantic extraction, and semantic-based downstream tasks in account-based blockchains.

[1]Remote Procedure Call (RPC) enables clients invoke functions on the remote hosts.

Table 1: Comparison of our tool and existing methods. ✓ and × denote that the data can or cannot be collected, respectively, and Δ means the data can be obtained partially.

	Ethereum-ETL [13]	EtherQL [12]	Etherscan [24]	BigQuery [11]	OpenEthereum [26]	DataEther [5]	**Ours.**
Lightweight	✓	✓	✓	✓	×	×	✓
Streaming ETL	✓	✓	×	×	✓	✓	✓
External transaction	✓	✓	✓	✓	Δ	✓	✓
Internal transaction	✓	✓	✓	✓	Δ	✓	✓
Token transfer (ERC20, ERC721 & ERC1155)	Δ	Δ	✓	✓	Δ	Δ	✓
Transaction semantics	×	×	Δ	×	×	×	✓

2.1 Transaction data acquisition

Due to the public accessibility of blockchain transaction records and the mining value embedded in these data, researchers have proposed many blockchain data acquisition methods. There are three kinds of typical ways to acquire transactions on blockchains: invoking Remote Procedure Call (RPC) interfaces, crawling website APIs, and instrumenting full nodes.

As the most popular data acquisition approach, ETL through RPC interfaces is lightweight. Most clients of Ethereum such as Geth[2] and OpenEthereum [26] provide RPC interfaces for users interacting with data on chain. To simplify the deployment of RPC interface servers, some online platforms including Infura[3] and Alchemy[4] also provide free RPC interfaces for users. Based on the RPC interfaces, ETL approaches [3, 12, 42, 43] were designed to acquire blockchain transaction data, where the most famous RPC-based ETL tools is Ethereum-ETL [13]. However, the performance of existing RPC-based approaches is limited by the supporting library, Web3.py [14], which provides a batch-level parallel invocation.

Optimized for specific query tasks, some website APIs, e.g. BigQuery [11] and Etherscan, were employed in existing studies [9, 12] to fetch transaction data. However, third-party website APIs usually set anti-crawler rules to restrict the query frequency of users, thus making it difficult to query all the transaction data through the website APIs. Additionally, ETL workflows based on website APIs usually cannot switch APIs easily because different website APIs are usually called in different ways.

For faster and more complete data acquisition, existing work [5, 6, 36] proposed to perform a transaction data ETL workflow via an instrumented full node on blockchain. Compared with other ETL approaches, running a instrumented full node for collecting transaction data is relatively heavyweight, requiring times of resources. Moreover, when the full node is upgraded, the instrumented full node based ETL approach has to abandon the old modified node and hook the code of the upgraded full node again manually.

[2]https://geth.ethereum.org
[3]https://infura.io
[4]https://www.alchemy.com

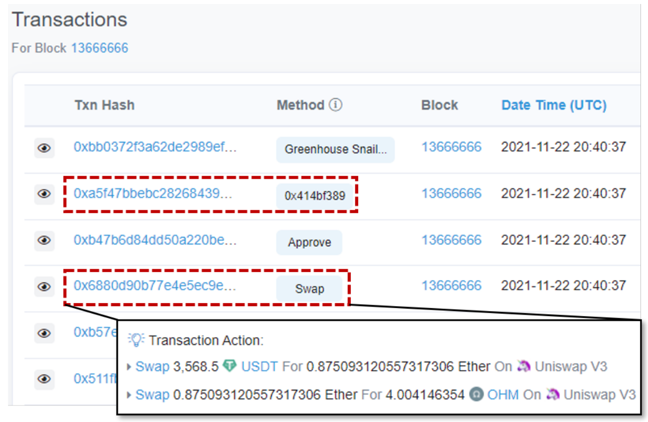


Figure 3: An example of transaction semantics on Etherscan. The red dotted blocks mark two noteworthy transactions, where the former cannot be recognized while the latter can.

2.2 Transaction semantic extraction

Recent work has devoted to understanding the semantics of transactions. Etherscan, the Ethereum browser, supports extracting and displaying the semantics of transactions by handcrafted rules. As shown in Figure 3, Etherscan recognizes two swap actions in the fourth transaction successfully but fails to identify semantic of the second transaction. Based on specific transaction patterns predefined by experts, [30, 44] extracted attack transaction semantics like flash-loan and re-entrancy hidden in massive transactions.

DeFiRanger [36] modeled the money transfer relationships as Currency Flow Tree (CFT) for classifying DeFi transaction semantics. Utilizing The Graph [25], a website provided indexing APIs, [37] queries some specific DeFi semantics e.g. *Mint*, *Swap*, and *Burn*.

However, existing semantic extraction methods are usually based on the analysis of some specific semantics by experts, and are therefore deficient in universality, especially for the transactions with rich semantics in Web3 applications, e.g. DeFi and NFT.

2.3 Semantic-based downstream tasks

The semantics of transactions has important implications for information extraction and behavioral understanding of on-chain data. Existing downstream tasks of blockchain data mining based on transaction semantics can be divided into three categories, i.e., transaction classification, account classification, and eco-analysis.

Transaction classification aims to identify the intent or involved business of given transactions. Based on extracted semantic information of transactions, Wang et al. recognized the flash loan on the massive blockchain transaction data firstly [30], Wu et al. detected the price manipulation actions towards DeFi applications [36], and Zhou et al. identified various attacks e.g. reentrancy, honeypot, etc. For anti-money laundering, Yousaf et al. [39] and Wu et al. [35] proposed detective methods to classify the mixing transaction.

For blockchain trading systems like Ethereum, the main purpose of account classification is to recognize the identity or type of the accounts in the blockchain trading systems. Wang et al. [29] classified the accounts on Ethereum into diverse categories, e.g.

miner, oracle and etc. Some researchers [4, 34, 38] aimed at detecting the phishing accounts and giving previous warning to reduce losses.

Eco-analysis attempts to explore the blockchain evolutionary trend and offer more insights for research and investment. Wang et al. [29] found the arbitrage behavior between different exchanges on Ethereum. Based on the money transfer semantics, Zhao et al. [41] conducted an evolution study of Ethereum ecosystem.

3 PROPOSED APPROACH

Summarizing the discussions in the previous section, research in transaction semantic recognition, which has rich Web3 application scenarios, is still in a relatively preliminary stage. In particular, existing transaction semantic recognition methods have limitations in terms of efficiency and cost of data acquisition as well as the universality of transaction semantic extraction. To this end, this paper proposes a data acquisition tool that balances lightweight and speed performance, and a generic semantic representation method based on transaction pattern mining. Here we apply the concept of *motif*, a high-order network organization, to portray transaction patterns, and thus name the proposed approach as **Mo**tif-based **T**ransaction **S**emantics representation method (**MoTS**).

First, we propose a parallelized RPC-based ETL for request processing to achieve high-speed and lightweight blockchain data fetching. Then, based on the concept of motif, we propose an efficient and generalized method for higher-level semantic extraction.

3.1 Transaction data acquisition

Through RPC interfaces, ETL tools can obtain transaction data on blockchains. Blockchain trading systems, such as Ethereum, allow users to join the network with the use of clients that implement the P2P protocol. Common Ethereum clients like Geth and OpenEthereum provide JSON-RPC interfaces to access the raw data on the blockchain. Three types of transactions, i.e. external transactions, internal transactions, and token transfers, can be extracted from the raw data on the blockchain through the RPC interfaces, *eth_getBlockByNumber*, *trace_block* and *eth_getLogs* respectively. Usually, externally owned accounts actively create external transactions to transfer money or execute smart contract logic. Executing smart contract logic could result in more internal transactions and token transfers that are also for the purpose of transferring money.

To improve the speed of data acquisition, our ETL tool makes parallel invocations to the RPC interfaces. Figure 4 illustrates two kinds of parallel invocation, namely ***batch-level parallel invocation*** and ***request-level parallel invocation***. Most of current ETL methods for blockchain data parallelly invoked the JSON-RPC interfaces at the batch level, which initiates multiple calls at the same time. However, as shown in Figure 4, the next batch-level parallel invocation can only be executed after all previous calls are completed, which increases the additional waiting time. To this end, we choose the request-level parallel invocation to design our RPC-based ETL tool. As Figure 4 shows, with the same load of invocation, the request-level parallel invocation reduces the wasted waiting time for other calls to complete. In this way, the request-level parallel invocation can obviously speed up the ETL procedure.

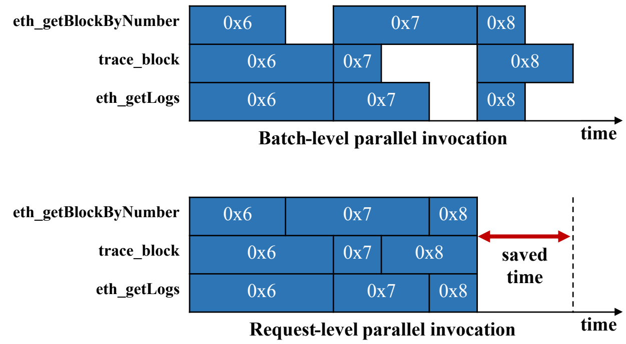


Figure 4: Two kinds of parallel invocation in RPC-based ETL tools. 0x6, 0x7, and 0x8 indicate the invocation order.

Referring to Figure 2, the proposed RPC-based ETL tool basically consists of three components, namely the request queue, worker, and pipeline. During the data acquisition, the RPC-based ETL tool fetchs transaction data, according to the order of block numbers, which is consistent with the order of data generated on blockchain trading systems. Therefore, for the first step, the request queue continuously generates RPC data access requests in block number order and delivers them to workers. After that, multiple workers parallelly invoke the JSON-RPC interfaces at the request level to download the raw data, parse it into external transactions, internal transactions, and token transfers, and then distribute the transactions to the pipelines. Finally, the pipelines sorts the output transactions according to the block number, ensuring the order of transactions output by pipelines is the same as that generated on chain.

Additionally, many blockchain clients, e.g. BNB chain and Polygon, have RPC interfaces that adhere to the protocol of Ethereum clients, making it possible to utilize our ETL tool on these Ethereum-like blockchain trading platforms without any modifications.

3.2 Transaction semantic extraction

This work discusses two kinds of transaction semantics: *low-level semantics* and *high-level semantics* [36]. Low-level transaction semantics refer to the explicit labels of transactions which are defined by the blockchain trading systems or contract standards, such as money transfer, contract creation, etc. Among various low-level semantics, the money *Transfer* semantics, carries out the most basic transaction action. Note that the money *Transfer* semantics can be expressed by an external transaction, an internal transaction, or a token transfer. On the contrary, the higher-level semantics are expressed through a combination of a series of external transactions, internal transactions and token transfers. Consider *Swap*, a high-level semantics in DeFi, enabling an account to swap one token for another in accordance with a set exchange rate. Figure 5 shows how Uniswap [45], a DeFi application, achieves *Swap* (high-level semantic) by executing multiple money *Transfer* (low-level semantic) transactions according to the smart contract logic.

In most cases, a high-level semantics may contain a series of transactions that carry lower-level semantics, but do not rely on a

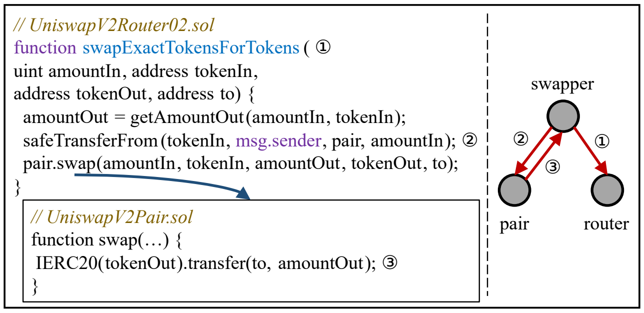


Figure 5: The illustration of *Swap*, a kind of high-level transaction semantics. A simplified logic of *Swap* in Uniswap-V2 using Solidity is present, where the numbers indicate the trigger order of money transfers. Especially, an M_{14} can be found here, and the links are marked in red.

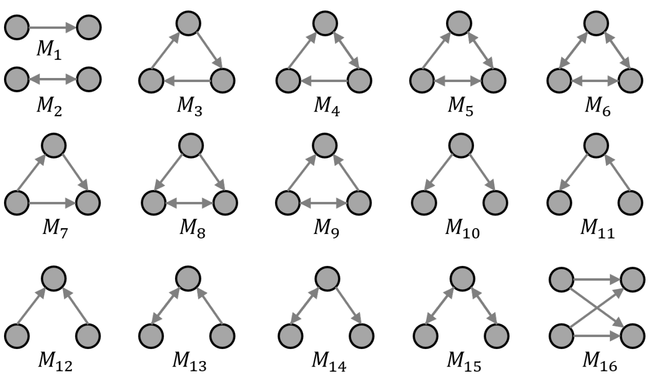


Figure 6: Network motifs used to characterize the semantic network.

simple overlay of transactions, but follow a specific form of transaction combination. By modeling accounts as nodes and transactions as links, we can explore and mine the relationships of these transactions from a network perspective. For example, a triangle transaction pattern can be found in *Swap*, as shown in Figure 5. As a result, in order to gain a comprehensive representation and understanding of high-level semantics, we construct a semantic network $G = (V, E)$ for each external transaction along with its triggered internal transactions and token transfers after acquiring the transactions through our RPC-based ETL tool. The node set V consists of accounts and the link set E represents a series of transactions. It is worth mentioning that in order to speed up and simplify the subsequent network analysis, here we define that a semantic network consisting of internal transactions and token transfers generated by the contract logic triggered by one external transaction, and only consider transactions defined as *Transfer* in the system.

Prior studies [36, 44] have demonstrated that different semantics usually correspond to distinct network patterns. Therefore, we calculate the representation vector of a high-level semantics by mining its corresponding semantic network. To efficiently capture the topological patterns of semantic networks, we considered the 16 motifs shown in Figure 6. These motifs have been widely considered in much previous work and have been shown to reveal the characteristics of various complex networks. In this way, we use a 16-dimensions semantic vector x to represent the semantics of a transaction, where the x_i means the frequency of the i-th network motifs in a semantic network. In [1], Bensen et al. proposed an approach to count the frequency of network motifs on the networks and obtain the motif adjacency matrix approach.

The method proposed in [1] is a generic method designed for simple directed networks without multiple links between node pairs. In this paper, the calculation approach of [1] is adapted to the scenarios discussed by considering the multiplicity of links and the realistic patterns of the transaction semantics. The details of the approach are as follows. Consider the adjacency matrix A of a semantic network G, the element A_{ij} denotes the number of links from nodes i to j. In addition, the element of $A \odot A^i_M$ means how many times the i-th network motifs appears, where A^i_M denotes the i-th motif adjacency matrix in A. The computations of the motif adjacency matrix are given in Table 2, in which D means the simple directed adjacency matrix. Consider the sign function sgn, the element of D is given as

$$D_{ij} = sgn(A_{ij}). \tag{1}$$

Therefore, the i-th element of x can be computed by Equation 2, where the function $sum()$ sums all elements for the given matrix, and $|E^i_M|$ means the link number of the i-th network motifs.

$$x_i = \frac{sum(A \odot A^i_M)}{|E^i_M|}. \tag{2}$$

The proposed semantic vectors can be calculated with a low computational cost. We give the analysis for the computational complexity of the transaction semantic vectors in Theorem 3.1 (see appendix for the proof).

Theorem 3.1. *Considering the gas limit of a block is g_b and a gas cost per log is g_l, the computational complexity of the transaction semantic vectors in a block is $O((\frac{g_b}{g_l})^2)$.*

As of October 2022, the gas limit for each block on Ethereum is about 30 million, according to Etherscan, and the gas cost per log is about 1,000 [31]. As a result, computing the semantic vectors in each block requires at most 900 million calculations, which is quite easy for any CPU with a clock speed of more than 1Ghz.

4 EXPERIMENTS

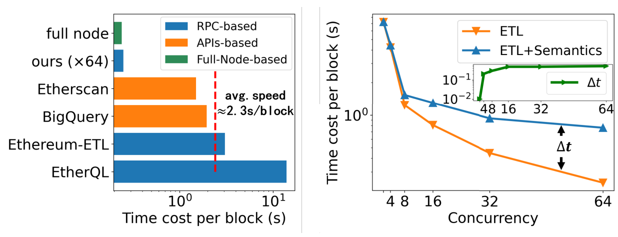
To evaluate the performance of MoTS, this section presents the experimental results. Experiments in this work aim at answering the questions as follow:

(1) Is our RPC-based ETL workflow fast enough to acquire the transactions and extract the transaction semantics? **(Q1)**
(2) Whether the semantic vectors can distinguish different transaction semantics? **(Q2)**
(3) What downstream tasks can the semantic vectors be used for? **(Q3)**

Our experiments are conducted on 3 cloud computation nodes, where each node is equipped with 8 vCPUs, 64GB RAM, 2TB disk. Additionally, we deploy Galaxybase [40], a distributed parallel

Table 2: Computations for part of the motif adjacency matrix, where i-th row corresponds to A^i_M and C_i is the intermediate variable. Other computations based on D following [1].

Matrix computations	$A^i_M =$
$C_1 = D - C_2$	C_1
$C_2 = D \odot D^\top$	C_2
$C_3 = (C_1 \cdot C_1) \odot C_1^\top$	$C_3 + C_3^\top$
$C_4 = ((C_2 \cdot C_1) + (C_1 \cdot C_2)) \odot C_1^\top + (C_1 \cdot C_1) \odot C_2$	$C_4 + C_4^\top$
$C_5 = (C_2 \cdot C_2) \odot C_1 + ((C_2 \cdot C_1) + (C_1 \cdot C_2)) \odot C_2$	$C_5 + C_5^\top$
$C_6 = (C_2 \cdot C_2) \odot C_2$	C_6
$C_7 = ((C_1 \cdot C_1) + (C_1 + C_1^\top) + (C_1^\top + C_1)) \odot C_1$	$C_7 + C_7^\top$
$C_8 = (C_1 + C_2) \odot C_1 + (C_2 \cdot C_1^\top) \odot C_1^\top + (C_1^\top \cdot C_1) \odot C_2$	C_8
$C_9 = (C_1^\top \cdot C_2) \odot C_1^\top + (C_2 \cdot C_1) \odot C_1 + (C_1 \cdot C_1^\top) \odot C_2$	C_9



(a) Comparison of speed (b) Concurrency and speed

Figure 7: (a) Comparison of different approaches in terms of the time cost of transaction acquisition for each block. ×64 means the maximum concurrency is 64 when parallelly invokes RPC interfaces. (b) The extraction speed with different concurrencies when the transaction semantics computation is enabled or not.

graph database, to store the external transactions, internal transactions, and token transfers on Ethereum prior to 13.25 million blocks, for a total of more than 4 billion records.

4.1 Is our RPC-based ETL fast enough?

To investigate the efficiency of our RPC-based ETL approach, a comparative experiment on transaction acquisition speed is conducted in this section. Particularly, we consider Ethereum-ETL v1.10.1 [13] and EtherQL [12], two RPC-based approaches, Etherscan APIs [24] and BigQuery [11], two website API-based approaches, and a full-node-based approach OpenEthereum v3.3.2 [26], as baselines. Note that Ethereum-ETL does not support multi-core computing. For fairness, all comparison methods in the experiment are run on single-core. In addition, we use the RPC interfaces provided by Alchemy for the RPC-based approaches and all comparison methods turn on the streaming mode with default settings, which enables the output transactions to be in the order of block numbers. Finally, we randomly choose 10,000 consecutive blocks four times, and all methods extract external transactions, internal transactions, and token transfers on Ethereum and report the average speed.

Experimental results shown in Figure 7(a) indicate that our RPC-based tool achieves State-of-Art transaction acquisition speed. First, by request-based parallel invocation, our RPC-based ETL is faster than the batch-based parallel RPC approaches, which turns out to be consistent with our analysis in Section 3.1 and Figure 4. Specifically, compared with the fastest existing RPC-based method, Ethereum-ETL, our tool achieves about x10 speedup. Secondly, the website APIs-based methods provided by Etherscan are faster than Ethereum-ETL, due to the specific optimization of transaction queries. Compared with the website APIs-based methods, our tool can also get about x5 speed up, because the frequency of invoking APIs is usually limited by the anti-crawler rules of providers. Interestingly, the transaction acquisition speed of our RPC-based ETL tool is similar to the full-node-based methods, which run on at least 16GB of memory, but our tool uses just roughly 1GB.

In addition, our RPC-based ETL method satisfies the real-time transaction acquisition demands on multiple blockchain platforms. As of the submission time of this paper, the block generating speed on Ethereum is about 12s per block, which means that all of the above methods are fast enough to acquire transactions on Ethereum. However, many other Ethereum-like blockchains generate blocks faster than Ethereum. Therefore, we calculate the average block production speed of the Ethereum-like blockchains, such as BNB chain [2], Polygon [23], and Avalanche [19] from BlockScan[5], and plot it in Figure 7(a), i.e., 2.3s/block. Our RPC-based ETL method is still fast enough when acquiring transactions on the Ethereum-like blockchains as Figure 7(a) shown, but other RPC-based methods are unable to timely access the latest transactions because the data acquisition speed is lower than the block production speed.

Our RPC-based tool is also fast enough to extract transaction semantics in real time while acquiring transactions from raw data. As shown in Figure 7(b), when the concurrency of RPC parallel invocation is high enough (>32), the extraction per block can be finished within 1s whether transaction semantics computation is enabled or not. Additionally, comparing the speed when transaction semantics computation is enabled or not, we estimate the cost Δt of transaction semantic computation around 0.5s per block, which fits our cost analysis in Theorem 3.1.

4.2 Whether the semantic vectors are distinguished?

To bring the answer to **Q2**, we conduct unsupervised experiments to prove that the semantic vectors corresponding to different semantics can be distinguished. To verify the results, we collected the transaction semantic labels in 2021 from Etherscan. Be aware that, although occasionally unable to provide precise labels, Etherscan is still the only tool that, as of right now, offers large-scale transaction semantic labels. The distribution of transaction semantic labels is given in Figure 1. With the exception of *Transfer*, we selected other top 9 transaction semantics on Etherscan, and more than 60% of non-*Transfer* semantics fall into these categories. Note that *Transfer* is filtered because *Transfer* has been recognized by definite rules in the ETL workflow.

As Figure 8 shows, the boundary among different types of semantic vectors after dimensionality reduction by T-SNE [27] is clear, indicating that the semantic vectors can distinguish different transaction semantics. At least five kinds of transaction semantics

[5]https://blockscan.com/

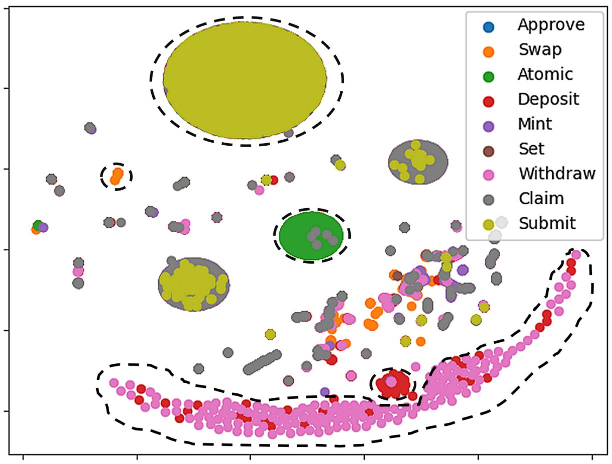


Figure 8: Low-dimensions visualization of semantic vectors using t-SNE. The classification boundaries of *Swap, Atomic, Deposit, Withdraw,* and *Submit* are marked by dotted lines.

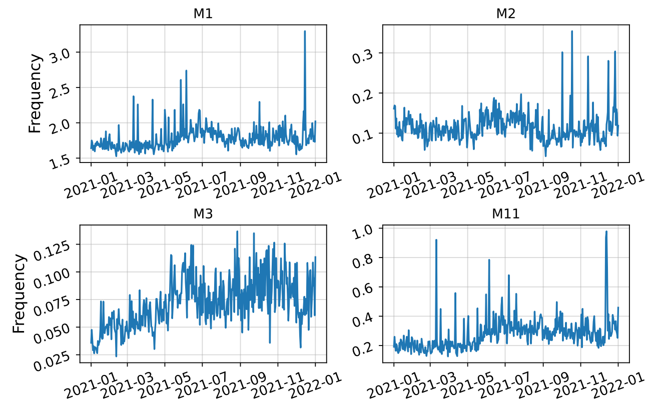


Figure 9: Evolution of the frequencies of four motifs (M_1, M_2, M_3, and M_{11}) over time in 2021.

categories i.e. *Swap*, *Atomic*, *Deposit*, *Withdraw*, and *Submit* can be distinguished through the low dimension visualization, and we mark the boundary among the transaction semantics with dotted lines in Figure 8.

4.3 What can the semantic vectors do?

To answer **Q3** and demonstrate the potential of semantic vectors in data mining, we employ semantic vectors in supervised tasks, i.e., transaction and account classification, and economy analysis.

4.3.1 Transaction classification. The novel Web3 application of blockchain trading systems, e.g. DeFi, NFT, etc, are closely related to transactions. Therefore, recognizing various semantics of the transactions enables us to comprehend the Web3 applications on blockchain and resolve some important issues. Taking the DeFi semantics classification as examples, this section shows the powerful ability of semantic vectors in transaction classification tasks.

With the rapid growth since 2020, DeFi introduces a new business model to blockchain, carrying 224 billion USD transaction volume from April 2021 to April 2022 [21]. For the purposes of investment and minimizing risk, identifying DeFi transactions in blockchain trading systems is critical. Firstly, we crawl the DeFi transaction semantic labels in 2021 from Etherscan and collect the semantic vectors of transactions as features for classifying. Five kinds of typical machine learning models [16] are selected as classifiers, including decision tree (DT), support vector machine (SVM), multilayer perceptron (MLP), random forest (RF), and lightGBM. Table 3 shows the classification results of various models. Interestingly, even with the most simplest DT, different DeFi transactions can be identified well, demonstrating that there is a significant different transaction semantics differ in the high-dimensional semantic vector space.

4.3.2 Account classification. As a classic task in blockchain data mining, account classification aims to categorize the accounts [37], recognize the identity of transaction peers, and expose potential trading risks, e.g. rug pull scams [20]. This section conducts the rug pull scam token task, demonstrating that semantic vectors can significantly improve the performance of existing models.

In order to execute rug pull scams, scammers usually deploy the scam token contract in advance [37]. Therefore, identifying scam token contracts in advance and front-end to warn users can reduce losses. To this end, we fetch the scam token contracts label dataset [37], and collect all external transactions related to the scam token contracts and their creators, to build a transaction network. Specifically, the links in the transaction network denote external transactions, owning the features of amount, time, and semantics. Furthermore, the nodes in the transaction network denote accounts, with the features of the maximum, minimum, mean, variance, and sum of the features in related links. Two types of models, i.e. machine learning models [16] and graph neural network models [7, 10, 28], employed in scam detection [4, 37], are selected as classifiers. Table 4 shows that incorporating transaction semantic information improves the performance of all models.

4.3.3 Eco-analysis. Analyzing the Web3 ecosystem of blockchain helps identify new hot spots and provides insights into the evolution trend of blockchain [33]. The following analysis demonstrates that semantic vectors can be employed to explain the evolution trend of the Web3 blockchain ecosystem from a micro perspective, which can assists researchers and investors to make decisions.

Increasingly usage of smart contracts. With the rise of various smart contract-based Web3 applications on Ethereum, users can trigger diverse application logic to achieve various purposes through external transactions, rather than just using the blockchain system as a tool to transfer money. From the perspective of motifs, the number of M_1 is the same as the number of links of the semantic networks. As shown by the evolution trend of M_1 in Figure 9, the average number of links in the semantic network exceeds 1, indicating that a sizable portion of external transactions trigger the smart contract logic, rather than carrying out money *Transfers* (M_1 should be 1 at this time). Additionally, a typical contract triggering pattern can be found in M_{11}, where one *Transfer* simultaneously causes another *Transfer*. The doubling of the average frequency of M_{11} in 2021, as shown in Figure 9, also further proves the phenomenon that the usage of smart contracts on Ethereum is increasing.

Table 3: Use machine learning models to classify DeFi transaction semantic categories. All metrics are given in percentage.

Model	Mint		Burn		Remove liquidity		Add liquidity		Swap		F1-macro
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	
DT	87.98	77.65	93.66	55.98	98.60	86.84	99.87	78.32	95.62	98.34	85.93
SVM	68.87	34.15	12.62	83.75	34.07	83.53	79.00	75.15	90.05	93.45	57.30
MLP	87.92	77.50	93.10	55.07	99.06	86.20	98.72	78.19	95.59	98.33	85.59
RF	87.98	77.65	93.72	55.96	98.63	86.85	99.93	78.33	95.62	98.34	85.94
lightGBM	87.80	77.46	87.77	55.41	99.02	86.26	99.25	77.89	95.57	98.27	85.32

Table 4: The results of scam token contract classification. $*$ means that the models add semantics vectors as features.

Model	Precision	Recall	Accruacy	Δ Acc.
RF	91.41±0.25	91.37±0.68	91.76±0.32	+1.25
RF*	**93.02±0.85**	**93.12±0.87**	**93.01±0.88**	
SVM	88.77±1.42	88.48±2.23	88.00±0.92	+2.57
SVM*	**91.08±0.39**	**90.17±0.93**	**90.57±0.35**	
MLP	83.66±5.98	81.64±9.92	79.60±6.30	+6.3
MLP*	**86.30±1.95**	**85.89±2.99**	**85.63±1.46**	
SAGE	88.71±0.34	88.98±0.35	88.79±0.27	+0.47
SAGE*	**89.27±0.11**	**89.52±0.12**	**89.26±0.11**	
GAT	88.22±0.43	88.45±0.44	88.15±0.39	+1.53
GAT*	**90.01±0.79**	**90.22±0.80**	**89.68±0.69**	
APPNP	89.19±3.96	88.31±6.92	88.28±5.53	+3.19
APPNP*	**91.48±0.00**	**91.67±0.00**	**91.47±0.00**	
GPRGNN	78.87±2.97	75.61±4.72	73.66±2.82	+10.34
GPRGNN*	**86.10±2.37**	**86.23±3.25**	**84.00±2.74**	

```
function executeFundsTransfer( ①
Order memory buy,
Order memory sell) returns (uint) {
 // pay for seller and fees
 transferTokens(sell.paymentToken, seller, price); ②
 transferTokens(sell.paymentToken, exchange, fee); ③
 // transfer token executing by delegatecall
 token.safeTransferFrom(
  seller, buyer, tokenId, amount); ④
}
```

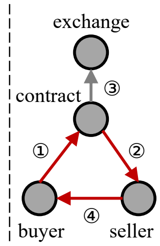


Figure 10: Illustration of a key function in NFT trading. An M_3 can be found, whose links are marked in red.

NFT becomes popular. NFT trading platforms represented by OpenSea exploded since 2021, and Wyvern [18], an open-source project, provides the contract logic for many other similar NFT trading platforms. In Wyvern, the key function to implement NFT trading is *executeFundsTranfer*. Figure 10 gives a simplified illustration of the *executeFundsTransfer* function and shows that triggering this function by an external transaction can generate a semantic network containing at least one $M3$, whose frequency quadruple in 2021 in Figure 9. Moreover, we collected the daily NFT market volume in 2021 from NFTGo[6]. The Pearson correlation coefficient, a data correlation metrics, between daily NFT market volume and M_3 frequency, reaches 0.82, which exhibits the close correlation between M_3 and the boom of NFT.

5 CONCLUSION

In this paper, we studied the real-time transaction semantic extraction problem on Web3 blockchain ecosystems and proposed a semantic representation method called MoTS, which consists of two main stages: transaction data acquisition and transaction semantic extraction. In terms of data acquisition, we proposed an RPC-based tool which can meet both lightweight and efficiency requirements compared to existing methods. Moreover, unlike some existing recognition methods designed for some specific semantics, our proposed motif-based transaction semantic representation method can be generalized to various semantics. Specifically, we sped up the RPC-based ETL tool for data acquisition dramatically by parallel invocation at the request level. After that, we modeled the money transfers as low-level semantic networks and obtained semantic representation vectors by calculating the frequency of 16 network motifs. The theoretical analysis and multi-dimension experimental results demonstrated the efficiency and effectiveness of our work. In particular, the experiments show that the semantic vectors extracted by MoTS can be employed to i) distinguish transactions with different intents, ii) effectively enhance transaction and account classification performance, iii) analysis and predict the Web3 ecosystem. Yet our approach still has some limitations, such as considering only basic transfer transactions to construct semantic networks and not utilizing contract code information. Therefore, in the future, we will further delve into transaction semantic representation by integrating more information like contract bytecode and other basic transaction semantics. Last but not least, this paper is the *first* attempt on generic semantic extraction of blockchain transactions, which provides a new perspective for understanding the blockchain and Web3 ecosystems.

[6]https://nftgo.io/

ACKNOWLEDGMENTS

The work described in this paper is supported by the National Natural Science Foundation of China (62032025, 61973325), the Natural Science Foundations of Guangdong Province (2021A1515011661), the Guangzhou Basic and Applied Basic Research Project (202102020616), and the Hong Kong RGC Project (No. PolyU15222320).

REFERENCES

[1] Austin R Benson, David F Gleich, and Jure Leskovec. 2016. Higher-order organization of complex networks. *Science* 353, 6295 (2016), 163–166.

[2] Binance. 2020. *BNB Chain, Build Web3 dApps on the Most Popular Blockchain.* Retrieved July 8, 2022 from https://www.bnbchain.org/

[3] Giorgio Camozzi, Felix Härer, and Hans-Georg Fill. 2021. Multidimensional Analysis of Blockchain Data Using an ETL-based Approach. In *17th International Conference on Wirtschaftsinformatik.* AiSeL, Nürnberg, Germany, 28.

[4] Liang Chen, Jiaying Peng, Yang Liu, Jintang Li, Fenfang Xie, and Zibin Zheng. 2020. Phishing scams detection in ethereum transaction network. *ACM Transactions on Internet Technology (TOIT)* 21, 1 (2020), 1–16.

[5] Ting Chen, Teng Hu, Jiachi Chen, Xiaosong Zhang, Zihao Li, Yufei Zhang, Xiapu Luo, Ang Chen, Kun Yang, Bin Hu, Tong Zhu, and Shifang Deng. 2019. DataEther: Data Exploration Framework For Ethereum. In *39th IEEE International Conference on Distributed Computing Systems.* IEEE, Dallas, TX, USA, 1369–1380. https://doi.org/10.1109/ICDCS.2019.00137

[6] Ting Chen, Yufei Zhang, Zihao Li, Xiapu Luo, Ting Wang, Rong Cao, Xiuzhuo Xiao, and Xiaosong Zhang. 2019. TokenScope: Automatically Detecting Inconsistent Behaviors of Cryptocurrency Tokens in Ethereum. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security.* Association for Computing Machinery, New York, NY, USA, 1503–1520. https://doi.org/10.1145/3319535.3345664

[7] Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. 2021. Adaptive Universal Generalized PageRank Graph Neural Network. In *9th International Conference on Learning Representations, ICLR 2021.* OpenReview.net, Virtual Event, Austria.

[8] Sam Gilbert. 2022. *Crypto, web3, and the Metaverse.* Retrieved July 8, 2022 from https://www.bennettinstitute.cam.ac.uk/wp-content/uploads/2022/03/Policy-brief-Crypto-web3-and-the-metaverse.pdf

[9] Voon Hou Su, Sourav Sen Gupta, and Arijit Khan. 2022. Automating ETL and Mining of Ethereum Blockchain Network. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining.* ACM, Virtual Event / Tempe, AZ, USA, 1581–1584.

[10] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. 2019. Predict then Propagate: Graph Neural Networks meet Personalized PageRank. In *7th International Conference on Learning Representations, ICLR 2019.* OpenReview.net, New Orleans, LA, USA.

[11] Valliappa Lakshmanan and Jordan Tigani. 2019. *Google BigQuery: The Definitive Guide: Data Warehousing, Analytics, and Machine Learning at Scale.* O'Reilly Media.

[12] Yang Li, Kai Zheng, Ying Yan, Qi Liu, and Xiaofang Zhou. 2017. EtherQL: a query layer for blockchain system. In *Database Systems for Advanced Applications - 22nd International Conference.* Springer, Suzhou, China, 556–567.

[13] Evgeny Medvedev and Allen Day. 2018. *Ethereum-ETL in BigQuery: how we built this dataset.* Retrieved July 8, 2022 from https://cloud.google.com/blog/products/data-analytics/ethereum-bigquery-how-we-built-dataset

[14] Piper Merriam and Jason Carver. 2018. *Web3.py is a Python library for interacting with Ethereum.* Retrieved July 8, 2022 from https://web3py.readthedocs.io/en/v5/index.html

[15] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. https://bitcoin.org/bitcoin.pdf.

[16] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research* 12 (2011), 2825–2830.

[17] Jason Potts and Ellie Rennie. 2019. Web3 and the creative industries: how blockchains are reshaping business models. In *A Research Agenda for Creative Industries.* Edward Elgar Publishing, 93–111.

[18] Protinam and Jeff Wentworth. 2018. *Project Wyvern Ethereum Smart Contracts.* Retrieved July 8, 2022 from https://docs.projectwyvern.com/

[19] Avalanche Team. 2020. *Avalanche: Blazingly Fast, Low Cost, and Eco-Friendly Dapps Platform.* Retrieved July 8, 2022 from https://www.avax.network/

[20] Chainalysis Team. 2022. *Crypto Crime Trends for 2022: Illicit Transaction Activity Reaches All-Time High in Value, All-Time Low in Share of All Cryptocurrency Activity.* Retrieved July 8, 2022 from https://blog.chainalysis.com/reports/2022-crypto-crime-report-introduction/

[21] Chainalysis Team. 2022. *DeFi-Driven Speculation Pushes Decentralized Exchanges' On-Chain Transaction Volumes Past Centralized Platforms.* Retrieved July 8, 2022 from https://blog.chainalysis.com/reports/defi-dexs-web3/

[22] Chainalysis Team. 2022. Defi Hacks Are on the Rise. https://blog.chainalysis.com/reports/2022-defi-hacks/.

[23] Polygon Team. 2019. *Bring the World to Ethereum, Polygon.* Retrieved July 8, 2022 from https://polygon.technology/

[24] The Etherscan Team. 2022. *Etherscan Simple API Calls.* Retrieved July 8, 2022 from https://api.etherscan.io/apis

[25] The Graph Team. 2022. *APIs for a vibrant decentralized future.* Retrieved July 8, 2022 from https://thegraph.com/

[26] The OpenEthereum Team. 2022. *OpenEthereum Documentation.* Retrieved July 8, 2022 from https://openethereum.github.io/

[27] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (11 2008), 2579–2605.

[28] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *stat* 1050 (2017), 20.

[29] Chaofan Wang, Xiaohai Dai, Jiang Xiao, Chenchen Li, Ming Wen, Bingbing Zhou, and Hai Jin. 2022. Demystifying Ethereum account diversity: observations, models and analysis. *Frontiers Comput. Sci.* 16, 4 (2022), 164505. https://doi.org/10.1007/s11704-021-0221-3

[30] Dabao Wang, Siwei Wu, Ziling Lin, Lei Wu, Xingliang Yuan, Yajin Zhou, Haoyu Wang, and Kui Ren. 2021. Towards a first step to understand flash loan and its applications in DeFi ecosystem. In *Proceedings of the Ninth International Workshop on Security in Blockchain and Cloud Computing.* Association for Computing Machinery, New York, NY, USA, 23–28.

[31] G Wood. 2019. Ethereum Yellow Paper: a formal specification of Ethereum, a programmable blockchain. *Accessed on: Mar* 6 (2019).

[32] Gavin Wood et al. 2014. *Ethereum: A secure decentralised generalised transaction ledger.* Technical Report. ETHCore.

[33] Jiajing Wu, Jieli Liu, Yijing Zhao, and Zibin Zheng. 2021. Analysis of cryptocurrency transactions from a network perspective: An overview. *Journal of Network and Computer Applications* 190 (2021), 103139.

[34] Jiajing Wu, Qi Yuan, Dan Lin, Wei You, Weili Chen, Chuan Chen, and Zibin Zheng. 2020. Who are the phishers? phishing scam detection on ethereum via network embedding. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 52, 2 (2020), 1156–1166. https://doi.org/10.1109/TSMC.2020.3016821

[35] Mike Wu, Will McTighe, Kaili Wang, Istvan A Seres, Nick Bax, Manuel Puebla, Mariano Mendez, Federico Carrone, Tomás De Mattey, Herman O Demaestri, et al. 2022. Tutela: An Open-Source Tool for Assessing User-Privacy on Ethereum and Tornado Cash. *arXiv preprint arXiv:2201.06811* abs/2201.06811 (2022).

[36] Siwei Wu, Dabao Wang, Jianting He, Yajin Zhou, Lei Wu, Xingliang Yuan, Qinming He, and Kui Ren. 2021. DeFiRanger: Detecting Price Manipulation Attacks on DeFi Applications. *CoRR* abs/2104.15068 (April 2021).

[37] Pengcheng Xia, Haoyu Wang, Bingyu Gao, Weihang Su, Zhou Yu, Xiapu Luo, Chao Zhang, Xusheng Xiao, and Guoai Xu. 2021. Trade or Trick? Detecting and Characterizing Scam Tokens on Uniswap Decentralized Exchange. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 5, 3 (2021), 1–26.

[38] Yijun Xia, Jieli Liu, and Jiajing Wu. 2022. Phishing Detection on Ethereum via Attributed Ego-Graph Embedding. *IEEE Transactions on Circuits and Systems II: Express Briefs* 69, 5 (2022), 2538–2542.

[39] Haaroon Yousaf, George Kappos, and Sarah Meiklejohn. 2019. Tracing transactions across cryptocurrency ledgers. In *28th USENIX Security Symposium (USENIX Security 19).* USENIX Association, Santa Clara, CA, USA, 837–850.

[40] Chen Zhang. 2018. *Galaxybase: Native Distributed Parallel Graph Platform.* Retrieved July 8, 2022 from https://www.galaxybase.com/

[41] Lin Zhao, Sourav Sen Gupta, Arijit Khan, and Robby Luo. 2021. Temporal analysis of the entire ethereum blockchain network. In *Proceedings of the Web Conference 2021.* ACM / IW3C2, Virtual Event / Ljubljana, Slovenia, 2258–2269.

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

[43] Weilin Zheng, Zibin Zheng, Hong-Ning Dai, Xu Chen, and Peilin Zheng. 2021. XBlock-EOS: Extracting and exploring blockchain data from EOSIO. *Information Processing & Management* 58, 3 (May 2021), 102477.

[44] Shunfan Zhou, Zhemin Yang, Jie Xiang, Yinzhi Cao, Min Yang, and Yuan Zhang. 2020. An Ever-evolving Game: Evaluation of Real-world Attacks and Defenses in Ethereum Ecosystem. In *29th USENIX Security Symposium (USENIX Security 20)*, Srdjan Capkun and Franziska Roesner (Eds.). USENIX Association, Boston, USA, 2793–2810.

[45] Noah Zinsmeister. 2022. *Peripheral smart contracts for interacting with Uniswap V2.* Retrieved July 8, 2022 from https://github.com/Uniswap/v2-periphery

6 APPENDICES

6.1 Proof of Theorem 3.1

Proof. For the account-based blockchains, the execution of any logic requires gas, and each block gas is capped, so the size of transaction data within each block is also limited [31]. In addition, the gas for triggering per transaction, including external transactions, internal transactions, and token transfers is different. Due to the fact that token transfers are recorded in logs, a low-cost storage medium, the gas to trigger per token transfer is the lowest among the above three kinds of transactions [31]. Thus each block has at most $O(\frac{g_b}{g_l})$ transactions. Suppose that the transactions in a block construct n ($n \geq 1$) semantic networks and the k-th semantic network G_k has $O(r_k \frac{g_b}{g_l})$ links where r_k s.t.

$$\sum_{k=1}^{n} r_k = 1, r_k \in (0, 1]. \tag{3}$$

The cost for computing the simple directed adjacency matrix of G_k is $O(r_k \frac{g_b}{g_l})$ and the cost for computing the motif adjacency matrix $\forall i, A_M^i$ is $O((r_k \frac{g_b}{g_l})^2)$, according to [1]. In this way, the computational cost of all semantic networks satisfies

$$\sum_{k=1}^{n} O((r_k \frac{g_b}{g_l})^2) \leq O(\sqrt{\sum_{k=1}^{n} r_k^4 \sum_{k=1}^{n} (\frac{g_b}{g_l})^4}), \tag{4}$$

according to the Cauchy Inequation. The inequation 4 take the equal sign when

$$\frac{r_1^2}{(g_b/g_l)^2} = \frac{r_2^2}{(g_b/g_l)^2} = ... = \frac{r_n^2}{(g_b/g_l)^2}, \tag{5}$$

that is

$$r_k = \frac{1}{n}, k = 1, 2, ..., n, \tag{6}$$

and the equivalence is

$$O(\frac{1}{n}(\frac{g_b}{g_l})^2). \tag{7}$$

Therefore, when it comes to $n = 1$, the computational complexity of the transaction semantic vector for all semantic networks is

$$O((\frac{g_b}{g_l})^2). \tag{8}$$

□