Flowchain: A Distributed Ledger Designed for Peer-to-Peer IoT Networks and Real-time Data Transactions

Jollen Chen

Flowchain Open Source Project, Devify Inc.
jollen@flowchain.io

Abstract. Recently, the blockchain for the IoT has considered an emerging technology for creating more secure and more cost-effective IoT systems. Despite a myriad of studies on blockchain IoT, few studies have investigated how an IoT blockchain system works in practice. In this paper, we propose Flowchain, an open source distributed ledger programming framework for peer-to-peer IoT networks and real-time data transactions. The main feature of the Flowchain framework is Virtual Blocks that provides a new blockchain data structure design to ensure the real-time data transactions. This paper also proposes a software architecture that provides peer-to-peer IoT networking and interoperable IoT application framework.

Keywords: Blockchain, Distributed Ledger, IoT, P2P, WoT, JSON-LD

1 Introduction

Bitcoin, a frequently referenced cryptocurrency, uses a distributed database system called a blockchain [5]. The Bitcoin blockchain can operate without any central server that the transactions stored with high trust. Although the Bitcoin blockchain has such decentralized network model, it does not meet the need of current IoT requirements. As the Bitcoin blockchain uses "unverified pool" to queue new transactions, the average waiting time for verifying a transaction could be 15 minutes that not in a real-time manner. Thus, to address this technical challenge, the main aim of the Flowchain programming framework is to provide a dedicated blockchain system for the IoT that can process and record transactions in a real-time manner. Flowchain presents a new mechanism called Virtual Blocks to provide such real-time transactions ability.

Besides, Bitcoin uses a distributed single blockchain model in which all nodes compete to mine new blocks. However, IoT hardware varies, e.g., resource-constrained devices, mobile devices, and high-performance server frames that the computing power varies from devices. Thus, memory-bound functions have been proposed to deal with heterogeneous hardware [7]. The mining process

can avoid "competition" and denial-of-service attacks by using memory-bound hash functions [8], however, this technique cannot be employed in IoT devices. A resource-constrained device has limited computational power and memory resource; therefore, memory-bound hash functions do not perform well on such IoT devices. Consequently, the proposed Virtual Block system can also address such technical challenges.

2 The Flowchain Framework Overview

To facilitate the decentralized IoT model, Flowchain also adopts the Web of Things (WoT) ontology as the underlying layer for interoperable devices along with the peer-to-peer IoT networking and the interoperable IoT programming framework.

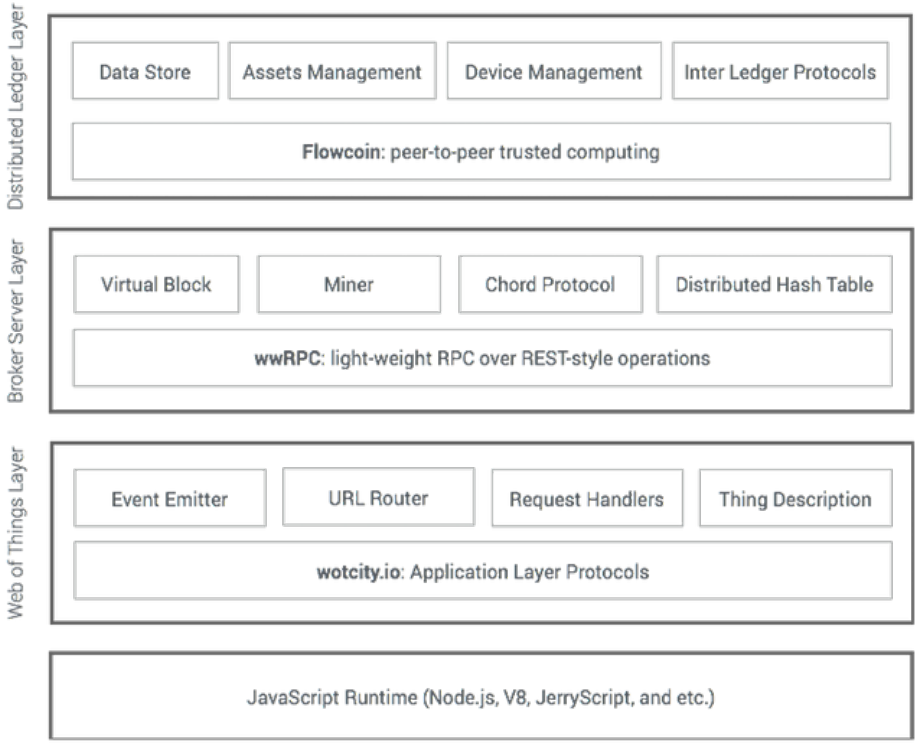

Fig. 1. Flowchain framework overview

Figure 1 shows the Flowchain framework with a full-stack and three-layer architecture design. The first layer (the WoT layer) provides a standards-based model

to represent a physical device as a "Virtual Things" [1, 4]. A Virtual Thing is referred to as a "node" in this paper.

The broker server is the second layer that implements peer-to-peer communications, REST-style RPC operations, and a distributed hash table (DHT). Flowchain uses the Chord protocol and algorithm, the technology initially published in 2001 by MIT [2], for providing the DHTs. Moreover, in the Flowchain framework, CoAP and WebSocket are the primary protocol bindings for RPC operations in the peer-to-peer network. CoAP is an application layer protocol expected for applications on resource-constrained devices. The WebSocket protocol provides a standardized way to facilitate real-time data transfer.

In practice, the application server runs on an IoT device and represents the node as a WoT application server. A Flowchain application server runs on a variety range of the hardware devices, from a high-performance device to a resource-constrained device. Figure 1 shows that the Flowchain framework uses the Node.js JavaScript runtime for a high-performance device, and the JavaScript engines, such as JerryScript [3] for a resource-constrained device. Given JavaScript's current ubiquitous nature, it is natural to use a JavaScript platform for the IoT system. Thus, Flowchain offers a 100% JavaScript platform for programming the distributed ledger of the interoperable IoT devices.

Furthermore, IoT devices in a decentralized IoT network may require a new model to exchange data. The data exchange model must be more secure and ensure data privacy. Thus, Flowchain uses blockchain-based decentralized IoT model to ensure such data security and privacy.

3 The Permissioned Distributed Ledger

The Flowchain peer-to-peer network authorizes nodes intending to join the network that only authorized nodes can operate as a Flowchain node and thus facilitates the permissioned DLT platform. Besides, Flowchain uses a proof-of-stake system to determine the device resource requirements that the peer-to-peer network uses such resources required to validate the reliability for a node intending to join the network.

3.1 The Virtual Block

As described in 1, existing blockchain systems do not act specifically for the IoT. To facilitate the blockchain for the IoT device, Flowchain employs an IoT dedicated blockchain data structure design and implements the Flowcoin system (described in 5) for the peer-to-peer IoT cryptocurrency and trusted computing to approach the secure data exchange required by the IoT device.

The proposed blockchain data structure is called Virtual Blocks, and it aims to provide real-time data transactions. Flowchain initially creates branches for each

node when nodes mine their Virtual Blocks. This design can estimate the block "forks" exception during the mining process. In this way, Flowchain can act in a real-time manner through maintaining "valid and invalid blocks." As shown in Figure 2, the important Flowchain data structure design features are as follows.

- Five IoT devices are labeled N1 to N5, and each device is a "node" in a peer-to-peer network.
- All nodes are mining blocks that use the same genesis block.
- In other words, each node creates a new "branch" for mining; thus, there is no blockchain "fork."
- Every block in each branch is called a Virtual Block.
- Virtual Blocks can be labeled as valid or invalid.
- Only valid blocks are available to record transactions.

The most significant design feature of the Flowchain data structure is that every node can only mine blocks at its branch. Therefore, Virtual Blocks do not need to be synchronized with all nodes because nodes do not "compete" to mine new blocks. Accordingly, Flowchain does not use a proof-of-work system [6].

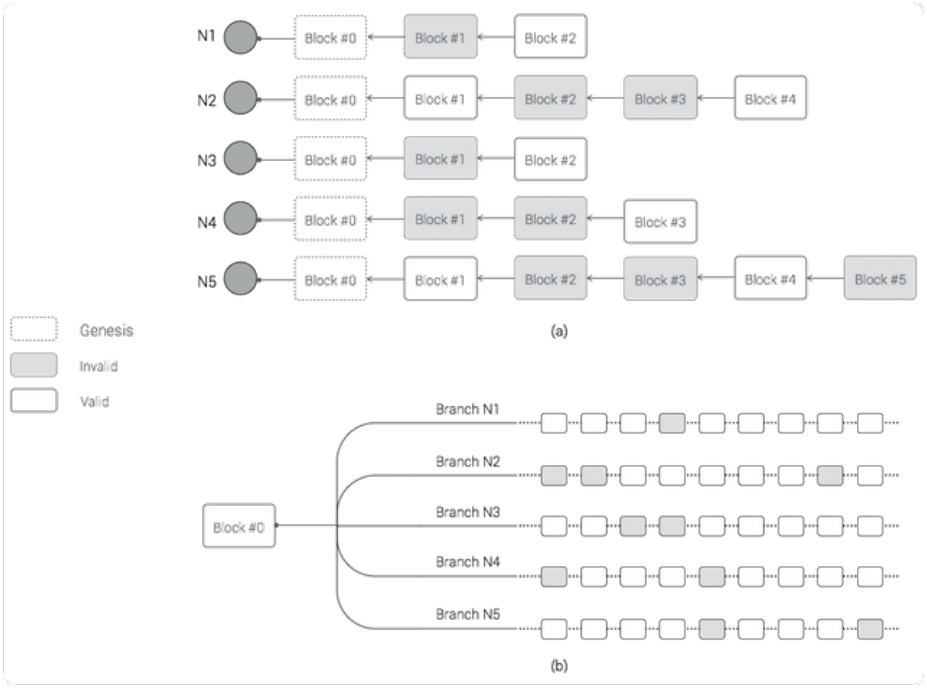

Fig. 2. Virtual blocks data structure

Flowchain also proposes a mechanism called “Inventory” by which administrators can perform a branch merging operation that all blocks merge imperceptibly into a single blockchain. For example, the administrator can merge all valid Virtual Blocks across all IoT devices in the same peer-to-peer network into a distributed ledger. In short, a Flowchain distributed ledger is built upon a series of the databases that allows IoT devices participated in the peer-to-peer networks to create, disseminate and store transactional data in an efficient and secure manner.

3.2 Difficulty Control and the Miner

Numerous blockchain systems, including the Satoshi blockchain, are intended to facilitate peer-to-peer currency transactions. Furthermore, “mining” is a mechanism and distributed consensus system that can verify and record such transactions. In Flowchain, the Virtual Block system can label blocks as valid or invalid. Valid blocks act as a secure ledger that stores transaction records. Although Flowchain and Bitcoin use the same SHA-256 hash algorithm, Flowchain has a very different mining algorithm design. The proposed design allows an IoT device to operate more stably when mining blocks. As shown in Listing 1, a node receives a key-value pair through the peer-to-peer network and then stores it in a valid block.

Listing 1. Flowchain virtual block algorithm

```
Node.on(‘message’, function(key, value) {
  // Get a valid block of the device’s blockchain
  N = GetOneValidBlock(chains)

  // Put key-value pair in block "N"
  PutToBlock( N, { key: value } );
});
```

Listing 2. Mining difficulty table

```
Difficulties = [
  ‘0000FFFFFFFFFFFF’, // [0.0, 0.2)
  ‘000FFFFFFFFFFFFF’, // [0.2, 0.4)
  ‘00FFFFFFFFFFFFFF’, // [0.4, 0.6)
  ’0FFFFFFFFFFFFFFF’, // [0.6, 0.8)
  ’FFFFFFFFFFFFFFFF’  // [0.8, 1.0)
]
```

Moreover, Flowchain will use the probability distribution as a mechanism to update the mining difficulty [9] and thus Flowchain can have a cost-effective

mining system. As we have seen above, Flowchain implements a mining-based proof-of-stake consensus system, and use the probability density function of the normal distribution to ensure a cost-effective difficulty control system as well as two properties:

- Reliability probability - A probability calculation can directly reference an IoT device's "reliability."
- Probability density - Use the reliability as the variance input of the probability density function.

Furthermore, to facilitate a faster and more cost-effective mining algorithm, a predefined difficulty table can easily implement such an algorithm. For example, the leading zeros will increase the degree of difficulty. The mining becomes increasingly difficult with more leading zeros. Listing 2 shows that the miner can simply search the difficulty table and pick a value according to the probability. Also, the miner is timed in a fixed number calculation per second in which the Flowchain can comprise a proof-of-stake mechanism.

4 Valid and Invalid Control

4.1 Invalid Conditions

The miner labels new virtual blocks found as valid, and as any invalid condition occurs, the current in use virtual block becomes invalid. Invalid virtual blocks are treated as deleted, and they will no longer become valid again. The invalid conditions can vary between different types of IoT hardware. For example, the operating system on a resource-constrained device may enter the starvation status due to the resource leaks. In general, invalid conditions are dependent on a result from such starvation problems, application process abnormal termination (e.g., crash, restart), the operating system exceptions (e.g., out of memory, out of disk space), and the program errors, such as the network disconnection error. Furthermore, there are also specific types of invalid conditions encountered by the IoT hardware, i.e., low battery warnings.

4.2 The Single Valid Block Model

To reduce the complexity of maintaining valid and invalid blocks, Listing 3 shows an O(1) implementation that labels the latest block as a Most Recently Used (MRU) block; thus, every IoT device will have only a single valid block.

Listing 3. Flowchain MRU block algorithm

```
Node.on('message', function(key, value) {
  // N is the length of the blockchain.
  // Put payload in the latest block in the blockchain.
  // This is to say; only the latest block is valid for use.
  PutToBlock( chains[N-1], { key: value } );
});
```

Hardware devices such as laptops work stably without certain invalid conditions, such as low battery warnings and out of memory exceptions. In theory, these systems can simply condition the impossibility of starvation, abnormal, exceptions and errors as mentioned earlier so that Flowchain can employ this single valid block model.

5 Flowcoin: Peer-to-Peer Trusted Computing

Flowcoin, Flowchain's cryptocurrency subsystem, is responsible for real-time transactions and recording trusted data. Flowchain provides time series and streaming data capabilities required by the IoT device. Furthermore, the Flowchain framework treats each data slice (the "chunk" data) in a time series or streaming data as a separate transaction, and Flowchain transfers each transaction to the peer-to-peer network. As such, Flowchain employs the Chord algorithm to exchange chunk data over the peer-to-peer network. In summary, Flowchain can ensure the IoT data security by using this chunk data model in which the distributed ledger stores transactional data across different IoT devices.

5.1 Simulating the Data Transaction

Each data slice is hashed by a double SHA-256 hash function to the corresponding data "key." Chunk data comprise sliced data and the data key. Then, Flowchain forwards the chunk data to the chunk data's "successor" node over the Chord ring. The Chord protocol and algorithm organize all IoT devices as a peer-to-peer network in a "ring" topology. The successor node lookup via the DHT with the data key processes the chunk data: Create a new transaction from the chunk data and store it in a valid block after verification.

Moreover, given the data key's hash generation algorithm, it is natural that the successor node is random and difficult to predict. In other words, the time series and streaming data are stored and distributed across IoT devices. Figure 3 shows the successor(key) function of the Chord algorithm that finds the data key's node through the peer-to-peer network. The successor node is represented as N'. When N' receives the chunk data, it combines the valid block ID and the data key to generate a transaction ID. To ensure data privacy, N' can also sign the transaction with its private key embedded in the hardware. Finally, N' creates

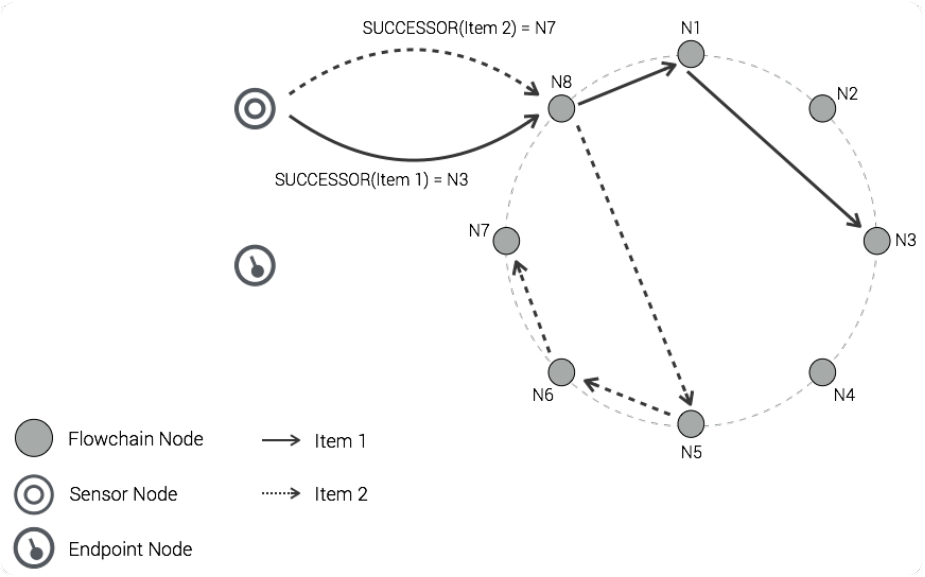

Fig. 3. Flowchain peer-to-peer networking

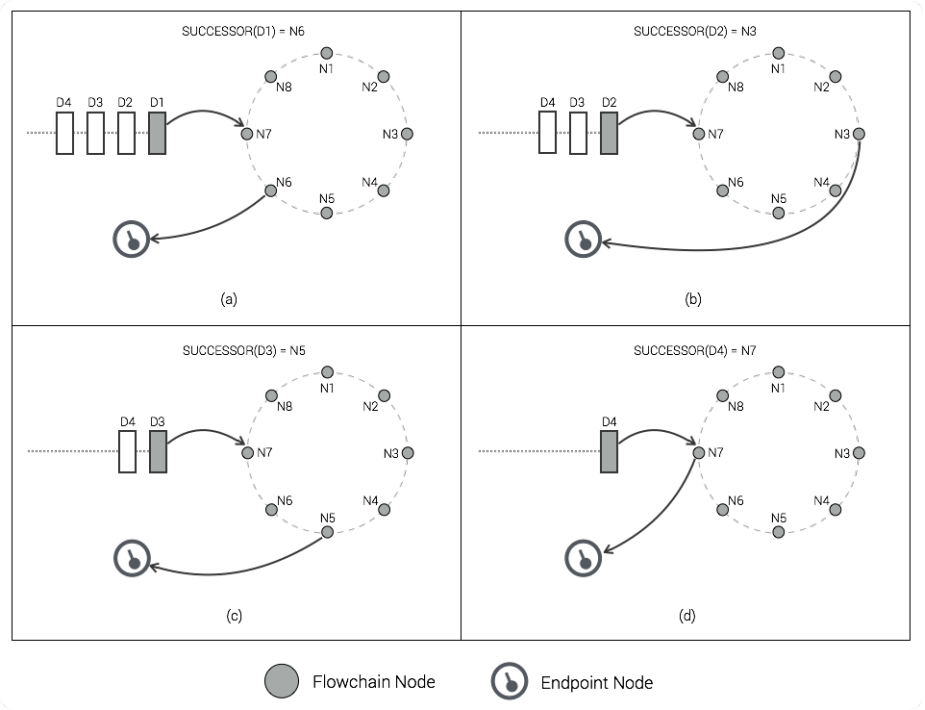

Fig. 4. Simulating data transactions

a record that comprises the transaction ID and the chunk data and stores the record in a valid block.

Figure 4 simulates four transactions from a time-series that each transaction is forwarded to the peer-to-peer network in sequence. Regarding the simulation process, it is evident that the successor node of each transaction is unpredictable. Thus, this design helps to ensure data security.

5.2 Linked Data for Transactions

This paper presents an initial work on creating a Linked Data document to support time series database (TSDB) via the semantic web technology. Time series data stored across the distributed ledgers requires the ability of fast access to the data store and retrieve, thus, Flowchain uses JSON-LD as the primary linked data technology to structure the transaction into a simple key-value document to make access to data more efficient. Furthermore, several studies [14, 15] have presented NoSQL databases as the high-performance key-value stores; thus, Flowchain uses Google LevelDB [16], a lightweight NoSQL database, as the backend engine to implement such TSDB technology.

Listing 4. Transaction generation algorithm

```
N'.PutToBlock(block, doc) {
  db = DatabaseAdapter.getDatabase();

  txID = SHA256( SHA256( block.id + doc.key ) );

  tx = new Transaction( doc.value );
  tx.sign( privateKey );

  record = {
    "@context": "http://flowchain.io/ledger-context.jsonld",
    "txID": txID,
    "tx": tx
  };

  db.put( record );
}
```

Listing 4 shows that the IoT node *N'* presents a transaction record in the JSON-LD document format. The use of Linked Data for Distributed Ledgers together with a NoSQL engine as the TSDB backend ensures the data access efficiency. Besides, another key advantage of Linked Data and Semantic Web technologies is that the TSDB can use the semantic object in `@context` as the NoSQL schema design to index transaction records. Figure 5 shows that columns are indexed (named) that this significant design ensures the data access performance rather than access columns as separate fields.

Object ID	Timestamp	Transaction ID (offset 0-31)	Transaction Data
		+0	+32
001	1492041600	c20c44b5ba7a34e7ddd7ec8cbd2203d3	tx1
002	1492041610	21b15ca036d6c144fc14b6b7fb201290	tx2
003	1492041620	905a862315e2c58fcb8038792be3951b	tx3
004	1492041630	685433e9d3fa916fdca73ebe8efd878a	tx4
005	1492041640	7b46c90b7ac12d3ad47c4cb7645c5741	tx5
006	1492041650	7d2c0a88166f5b267613ea51f455adf7	tx6
007	1492041660	fb2cc23fe3f078aea367123a48799dd5	tx7

Fig. 5. Colums are indexed by the semantic object

5.3 Handling Churn in the DHT

We use the periodic nature of Chord stabilization algorithm to handle churn in the DHT [12]. The algorithm periodically checks its successor node and updates the DHT. Therefore, the IoT node selects its successor node for measuring churn and simply uses the timeout calculation technique for handling churn [13].

In Flowcoin, the IoT node indicates successor node failure by routing a *CHECK_SUCESSOR* request to its successor node and maintain the *TTL* time. If the TTL drops to zero, Flowcoin handles churn by removing the successor node from the peer-to-peer network and subsequently updating the DHT. Also, the failure nodes can spontaneously intend to join the peer-to-peer network, and the periodic stabilization algorithm will update the DHTs. Moreover, the TTL value also determines the latency of the DHT lookup.

5.4 The Transaction Approval Process

Differing from Bitcoin's transaction process [10], Flowchain uses a "mining-transaction-approval-verify" process which forwards the transactional data to the endpoint before verification rather than the typical "transaction-mining-verify" process. Regarding the process (6) of Figure 5, N' forwards the chunk data to the endpoint after recording the transaction in the distributed ledger. At this time, Flowchain will not label this transaction as a "verified transaction." Subsequently, in the process (7), the endpoint requests "approval" via one node of the peer-to-peer network. The previously mentioned transaction will only become a verified transaction if the peer-to-peer network successfully verifies it. In conclusion, Flowcoin will recognize the transaction as a verified transaction when the endpoint requests to approve it. Thus, the Flowchain transaction process represents a "mining-transaction-approval-verify" model. This mechanism is the

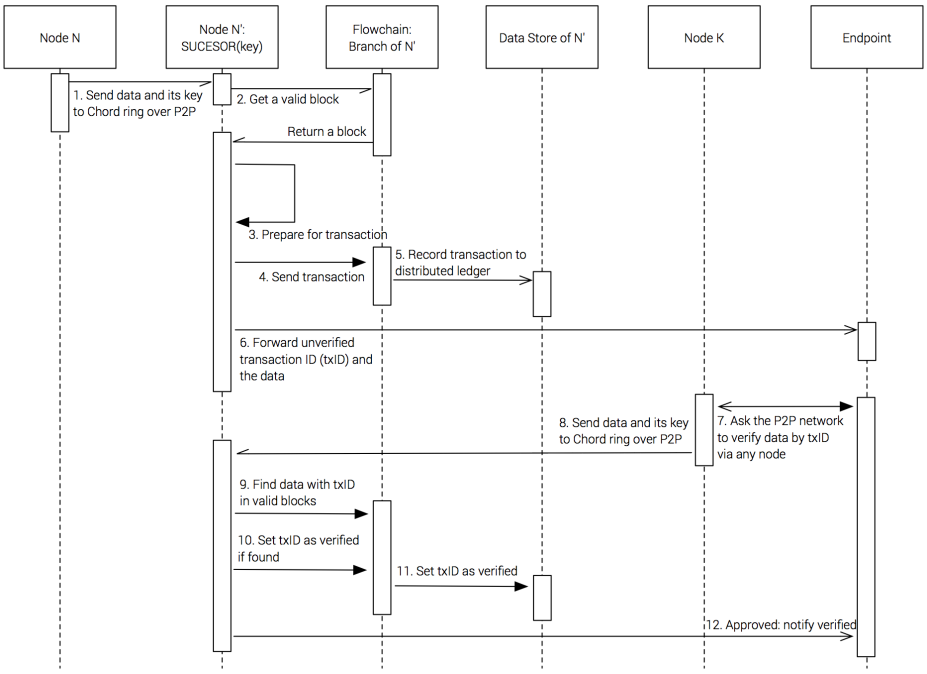

Fig. 6. Flowchain "mining-transaction-approval-verify" process

most important Flowchain design element. Moreover, this mechanism attempts to provide a time-series and streaming data model for current IoT requirements.

Figure 6 shows the process (11) that N' marks txID as verified after completing the approval request of the endpoint. Then, Flowchain grants one Flowcoin currency to N'. Note that N' can obtain more Flowcoin currency by completing more approval jobs. In this manner, Flowcoin enables Flowchain to comprise a resource-based proof-of-stake [11] mining approach to mine new blocks. An IoT node can deposit "coins" by joining and completing "approval" jobs. The miner ensures the node's minimum resource requirements, such as network bandwidth, battery level, Wi-Fi signal strength, and the "coins." Thus, the Flowchain difficulty algorithm uses the number of coins owned by a node along with the resource requirements to calculate the reliability probability.

6 Conclusions

In conclusion, the Flowchain designs the DLT for IoT-specific types of applications that the sharing of individual transactional data in ledgers is limited to the branch merging algorithms design of the administrators. Furthermore, the design ensures the interoperability between different ledgers within the same

IoT peer-to-peer network, and the design has a DLT transactional protocol built upon the Chord algorithm and protocol. Also, this paper has practically implemented Flowchain, a decentralized software framework for developing the blockchain on interoperable IoT devices over a peer-to-peer network. All work is currently available on GitHub as the Flowchain open source project that consists in three sub-projects at (a) wotcity.io (https://github.com/wotcity), (b) devify.io (https://github.com/DevifyPlatform), and (c) flowchain.io (https://github.com/flowchain).

References

1. Dave R.: An introduction to the Web of Things Framework. W3C (2015).
2. Stoica, I., Morris, R., Liben-Nowell, D., Karger, D., Kaashoek, M., Dabek, F., Balakrishnan, H.: Chord: a scalable peer-to-peer lookup protocol for internet applications. IEEE/ACM Transactions on Networking. 11, 17-32 (2003).
3. Scheller, T.: JerryScript - An ultra-lightweight JavaScript engine for the Internet of Things. OpenIoT Summit Europe 2016 (2016).
4. Web of Things Interest Group Charter, https://www.w3.org/2014/12/wot-ig-charter.html.
5. Satoshi Nakamoto: Bitcoin: A Peer-to-Peer Electronic Cash System (2009).
6. Proof-of-work system, https://en.wikipedia.org/wiki/Proof-of-work_system.
7. Coelho, F.: Exponential Memory-Bound Functions for Proof of Work Protocols. TR Ecole des mines de Paris A/370/CRI. Cryptology ePrint Archive, Report 2005/356 (2005).
8. Liu, D.Camp, L.: Proof of Work can Work. The Fifth Workshop on the Economics of Information Security (2006).
9. Kraft, D.: Difficulty control for blockchain-based consensus systems. Peer-to-Peer Networking and Applications. 9, 397-413 (2016).
10. Michael, Nachiappan (et al.): BlockChain technology beyond bitcoin. Sutardja Center for Entrepreneurship & Technology Technical Report (2015).
11. Sunny King, Scott Nadal: PPcoin: Peer-to-peer crypto-currency with proof-of-stake (2012).
12. Rhea, S., Geels, D., Roscoe, T., Kubiatowicz, J.: Handling churn in a DHT. Proceedings of the annual conference on USENIX Annual Technical Conference. 10-10 (2004).
13. Stutzbach, D., Rejaie, R.: Understanding churn in peer-to-peer networks. Proceedings of the 6th ACM SIGCOMM on Internet measurement - IMC '06. (2006).
14. Forfang, C., Bratsberg, S.: Evaluation of High Performance Key-Value Stores (2014).
15. Cattell, R.: Scalable SQL and NoSQL data stores. ACM SIGMOD Record. 39, 12 (2011).
16. LevelDB, http://leveldb.org.