Graph's Application For Modeling Blockchain Technology

Joseph Salimin and 13516037[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]joseph_salimin@hotmail.com

***Abstract*—Blockchain technology has grown rapidly in these few years. Blockchain itself is a continuosly growing list of records as we call it blocks, and it is secured by using cryptography. This paper will focus in presenting the model of blockchain in the graphs, not how the cryptography helps securing information in blockchain. There are many graphs which can model the blockchain. This papel will focus how to model blockchain using the transaction graph and trust management model. This paper also presents how blockchain grows by using the model of network topology, for example Barabási-Albert model. This model itself is an algorithm for generating random scale-free networks, making the network as a distributed system.**

***Keywords*— Barabási-Albert model, Blockchain, transaction graph, trust management model**

I. INTRODUCTION

We lived in an increasingly interconnected world or usually we call globalization. In this era of globalization, many interactions can be done online. It means peoples can connect without even seeing other people face-to-face. Online interactions also are a common occurrence in the daily routine of most individuals. Billion interactions are made everyday around the world. But how do we know that it is safe to interact in this interconnected world especially when we do online trading?

One of the solution from this problem is to do aunthetication. Authentication is a major enabler of the security on the Internet. It is a basic requirement of almost all secure online transactions to handle loss or crack by some hackers. But it also does not solve the problem as there are many hackers who can crack the centralized system and steal the information in it.

The other problem is consensus problem (Byzantine Generals problem). In asynchronous networks, when many parties are communicating over a network and some fraction may crash or behave maliciosly, it is very hard or even impossible for all the fraction to agree on a common value. A result in 1982 showed it is unsolvable by any deterministic protocol if even one party is at faulty.

These solutions actually can be solved by blockchain technology. As we know, blockchain technology has grown very rapidly over these recent years. One of the few examples is the fast growth and the popularity of Bitcoin, which uses the blockchain technology. These events also indicates that the power of the "blockchain" techonology is real and it is also considered very powerful because the ability of Bitcoin's blockchain to achieve such is considered by some to be one of its most powerful feature.

Blockchain in simple terms is a distributed database that is very secured. The basic concept of blockchain is quite simple: a distributed database that maintains a continuously growing list of ordered records. As in the case of creating blocks of transactions, the core ideas behind Blockchain directly shape creation and maintenance of nodes and edges in graphs. So, we know that graph is very useful to analyze blockchain.

Blockchain is very powerful, but many people also still wonder how it actually work and how this distributed system is very secured over the centralized system. That is why people must know how it actually works. Some theorem can help in modeling the graph of the blockchain. Blockchain primarily can be analyzed using transaction graph. But this paper will also discuss about trust management model dan modeling the network using Barabási-Albert model.

II. GRAPH'S THEORIES AND NETWORKING MODEL

A. *Type of Graph* [1]

The definition of graph is pair of sets (V,E) or we can define it as $G = (V,E)$. V is shown as nonempty sets from the vertices or nodes and E is sets of edges or arcs which connect two nodes. In modeling graph, we can name the nodes and the edges. For example, we can call nodes u and nodes v. The pair (u, v) can be called e (edge). In another way, if e is the edge which connects nodes u and v, then e is (u, v).

Graph can be empty. Null graph of empty graph is a graph which set of edges is an empty set but its nodes must not be empty and can be written as N_n with n is the sum of vertices.

Two nodes in undirected graph also be called adjacent if there is edge which connects those two nodes. In another way, u is adjacent to v if there is (u,v) in graph G. For $e = (u,v)$, we also call it incident to node u and v.

Graph can be grouped in such way based on the viewpoint of grouping. There are many types of graphs, but this section will only present some of them.

Based on the loops and the edges, there are two types of graphs, which is :

1. Simple graph

Simple graph is a graph which does not contain loop or multiple edges. Simple graph is also undirected graph as edges is an unorderer pairs. So we can define simple graph $G = (V,E)$

with *V* is set of nonempty nodes and *E* is set of unordered pairs of nodes.

2. Unsimple graph

Unsimple graph is graph which contains multiple edges or loops. There are two types of unsimple graph : multigraph and pseduograph. Multigraph contains multiple edges. These multiple edges can be considered as the same unordered pair of nodes. It means that every simple graph is unsimple graph, meanwhile not every unsimple graph is simple graph. Pseudograph contains both loop or multiple edges.

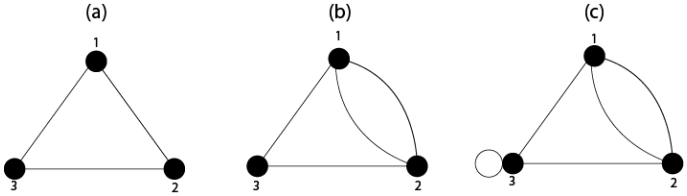

Fig. 1. Graph (a) simple graph, (b) multigraph, and (c) pseudograph

Based on the orientation of the direction, graph can be grouped to two types, which is :

1. Undirected graph

Undirected graph is a graph which edges does not have the direction in them. In undirected graph, edge is an unordered pair of two nodes. It means that (*u*,*v*) is the same as (*v*,*u*).

2. Directed graph

Directed graph is a graph which edges is given an orietation of the direction. These edges are usually called arcs. In directed graph, we consider (*u*,*v*) as a different arc from (*v*,*u*). For (*u*,*v*), *u* is called initial vertex and *v* is called terminal vertex. To draw directed graph, we usually draw the arcs with arrows. This paper will use directed graph to model the blockchain because it is necessary to know the orientation of the transaction.

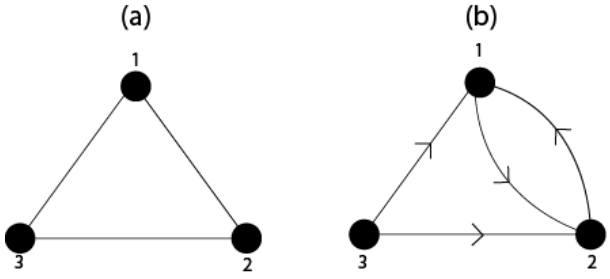

Fig. 2 Graph (a) undirected graph and (b) directed graph

B. Barabási-Albert Model [4]

The recognition that growth and preferential attachment coexist in real networks has inspired a minimal model of it. We called the model as the Barabási-Albert model, which can generate scale-free networks (it is also called BA model). This model tries to explain the existence of such nodes in real networks.

In Barabási-Albert Model, we start with the first nodes or initial nodes, we call it m_0 which is the links between which are chosen arbitarily. This model also incorporates two important general concepts: growth and preferential attachment. Growth means that at each timestep, we can add a new node with m links that connects the new node to another $m \leq m_0$ nodes in the network. Preferential attachment is the probability that a link of the new node connects to a node, we call it node *i*, which depends on the degree k_i.

Formally, the probability (p_i) of the new node to connect to node *i* can be formulated with this formula :

$$p_i = \frac{k_i}{\sum_j k_j} \qquad (1)$$

where k_i is the degree of node *i* and the probability is that degree divided by the sum of the degree of pre-existing node.

As we see in this model, it generates the distributed system in the networks since it is scale-free. For blockchain technology, for example bitcoin network, a new node is usually bootstrapped by connecting to nodes in the seed list. So we can say that blockchain also use this network topology in its network system.

In general, blockchain system consists of nodes which are called blockchain clients. Each nodes in this system is self-sufficient to perform the transactions. Each nodes also can interract with other nodes in the system. The valid transactions done by the nodes then will be included in a structure called block. The block itself is connected to other block in the chain.

C. Transaction Graph [2]

Transaction graph or TDAG is a directed acyclic graph (DAG). DAG itself is a directed graph which consists of a finite number of vertices and edges. Each vertex directs to another through an edge [3]. Transaction graph is like many graphs $G = (V,E)$. In the TDAG, the vertices V can be partitioned into state S and witnesses W, that is we can call $V = S \cup W$. The edges E represents the transition between states. The edges also can be partitioned into consuming, observing, and producing edges. So we also call edges E as $Ec \cup Eo \cup Ep$. The definition and the symbol in TDAG itself is :

1. States : State $s \in S$, denotes an atomic state represented by the blockchain and it is depicted by a circle. State models an individual assets, the digital coin, information, etc. The blockchain consists of all states that exist at that particular time. A state can be overwrite by some transactions. There's also special type of state and it is called genesis state which means the initial state of the blockchain.
2. Witnesses : Witness $w \in W$, denotes a witness in the context of a transaction and it is depicted by a rectangle. Witness consists of all the required data which occurs from the transaction. Without it, the transaction will not be valid and there must be only one witness in one transiction.
3. Consuming edges : A consuming edge connects a state to a witness and models that state to be "consumed" or updated or even overwritten by the transaction involving the witness. When a state has already been consumed, it can not be consumed again. Hence, a state can only be consumed once. Consuming edge is depicted by a circle-arc-rectangle.
4. Observing edges : an observing edge also connects a state to a witness but it does not consume the state as it only observe the state. Intuitively a transaction that observes a

state “reads” it. Observing edge is depicted by a circle-dotted-arc-rectangle.

5. Producing edges : a producing edge connects a witness to a state and denotes that a state is created by the transaction corresponding to a witness. Every state apart from genesis state can only be produced once.

From definitions above, we can say that a transaction has input states that are consumed or observed by the transaction and the output states which is the product of the transaction.

We can call a DAG $T = (V,E)$ graph with a of input states, output states, and a witness to be a transaction whenever :

- Every input state is a source.
- Every output state is a sink.
- V = input states ∪ output states ∪ {witness}.
- Every edge is either consuming edge, observing edge which links the input state to witness or a producing edge which links to the output state.

Given a TDAG as the definitions above, there are five types of TDAG, which depends mostly on the number of input and output states.

- INIT is a unique initialization transaction exists in every non-empty TDAG, consisting of consuming edge that links the genesis state to a witness. This is the initialization of the transaction and has special roles because it represents the creation of blockchain. Modeling initialization in some specific transaction is a deliberate design choice,
- SISO is a single input, single output transaction consists of one cosuming edge that links the input to witness and one producing edge that links the witness to the output.
- SIMO is a single input, multiple output transaction consists of one consuming edge that links the input to witness and a set of producing edges to the set of output states.
- MISO is a multi input, single output transaction consists of a set of multiple consuming and observing edges that link the input to witness and one producing edge to output.
- MIMO is a multi input, multi output transaction which consists of a set of multiple consuming and observing edges to witness and set of producing edges to output states.

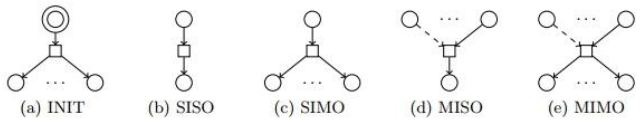

Fig. 3 Graph representation of transactions
Source : https://eprint.iacr.org/2017/1070.pdf

These types above can be combined into more complex TDAG which can be shown in Fig. 5. In Fig. 5, it consists of several type, for example INIT, SIMO, and MIMO. In this model, the transaction must update or overwrite at least one state in order to make it into a simple read queries as blockchain.

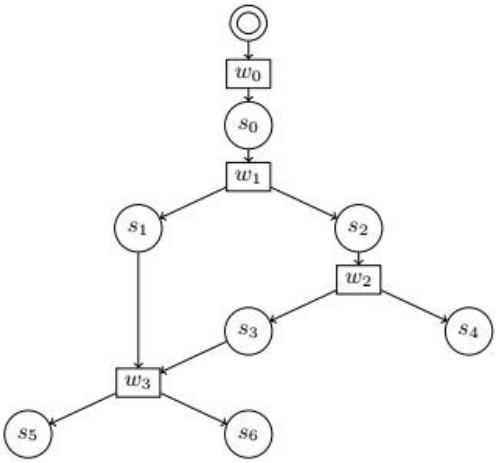

Fig. 4 Example of TDAG
Source : https://eprint.iacr.org/2017/1070.pdf

D. *Trust Management Model* [5]

Blockchain in its application also uses security to secure its information which is distributed throughout the network. A general approach is necessary to address the common security issues of authentication Trust Management (TM) systems which can be modeled by a graph.

We say we model a TM system for authentication, as a directed multigraph $G = (V,E)$ with vertices representing entities that participate in the system, and edges representing trust relations. Entities can be perceived as public keys in the network system.

For a trust relation (TR) itself, we call TR as a sequence $<A, B, c, v, \alpha, t>$ where :

- A is the trustor, the individual who give the trust to another.
- B is the trustee, the individual who is given the trust by the trustor.
- c is the context of the relation, an indentifier consists of the relevant information.
- $v \in [0, 1]$ is the actual value of the trust. 0 means no trust and 1 means total trust.
- α is a set of cryptographic artifacts, most commonly digital signature to assure integrity and authenticity of the relation.
- t is the logical time component, shows the time where the relation occurs.

There’s also TM Network which express the whole ecosystem of the relations, which includes all trust relations between each individuals. For the definition, we can call TM Network or trust graph is a directed multigraph $G = (V,E)$ where :

- Each $v \in V$ is an entity as the definitions above.
- Each $e \in E$ is a trust relations in the network.

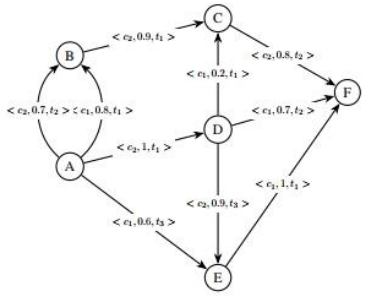

Fig. 5 Example of TM network
Source : https://arxiv.org/pdf/1711.04591.pdf

The goal of this TM network is to enable the assessment of trust in different contexts which is called trust assessment. Trust assessment is the result of the calcution of trust A to B in a context c. We defined it as :

$$T^{c}_{A\to B} \coloneqq P(c,H), \qquad P : c \times H \subseteq G \to [0,1]$$

where P is the program which parameter is context c and the subgraph H, which represents the view of the graph by A and it outputs the trust value to B. This trust value can take binary, discrete, or continues values and that output it in the interval of [0,1]. For the security itself, it also depends on P as it is used to derivethe trust assessment. This security could be defined as, whether the decisions made are correct.

III. MODELING THE BLOCKCHAIN

A. *Blockchain Algorithm* [7]

Blockchain consists of blocks of transactions which can be verified and confirmed without a central authority [6]. These blocks contain the information of the transaction made during a set or period. It is needed to keep record of all track and verify the transactions. That is why in blockchain, we can take an example from Bitcoin network, they deal this by collectiong all the transactions during a set period of time into a list and it is the miner's job to confirm the transactions and write them into the general ledger.

Blockchain algorithm allow the distributed network to gain consensus using distributed proof-of-work by solving transaction issues. For example, consider two transactions withdrawing digital currency from wallet entered into the distributed network at the same time. The problem is the network must decide which transactions must occur first. This is how miner's work. Miners select transactions and put them into a "block" and by using difficult probabilistic algorithm, they generate the transactions and put it into the block. New block contains the hash of the previous block, the new data to be added to the blockchain, and a random nonce. The first node to publish a block wins and the winner will get rewards. That is why there are many competitions between the miners.

Fig. 6 Example of block
Source: https://arxiv.org/ftp/arxiv/papers/1707/1707.04558.pdf

The process of generating a block to add to the blockchain, known as mining, can be performed by any node in the distributed network. We can assume that any node in the network can publish to all the other nodes in the network. It is also critical that node validates all the data sent to it before commiting them to block.

There is also some conditions to mine a block. For example, we can say that the nonce must have given a number of leading 0s. Then, the difficulty of the block is determined by how many leading 0s when it is being encoded.

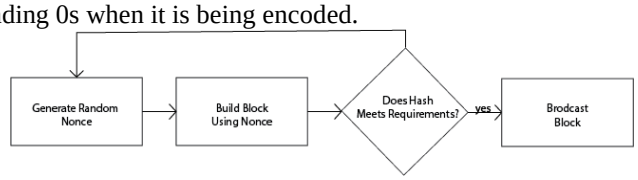

Fig. 7 Simple Blockchain

This paper will present simplest algorithm to generate (or mine) a block in blockchain. First of all, we need to make a structure of the block. As the definition says before, the block will have the index, the data (transaction), time, previous hash and its hash right now. Let us see the the program and the execution below.

(a)

```
class Block:
    def __init__(self, index, timestamp, data, previous_hash):
        self.index = index
        self.timestamp = timestamp
        self.data = data
        self.previous_hash = previous_hash
        self.hash = self.hash_encoded(random.randint(0, 3))

    # To generate the hash
    def hash_block(self):...

    def hash_encoded(self, difficulty_level):...

    def print_block(self):...

# Genesis block is the first block of the blockchain
genesis_block = Block(0, date.datetime.now(), random.randint(0, 45), '0')
genesis_block.print_block()
blockchain = [genesis_block]

# Generate all the block
for n in range(1, 10):
    block = Block(n, date.datetime.now(), random.randint(0, 45), blockchain[n-1].hash)
    blockchain.append(block)
    block.print_block()
    time.sleep(1)
```

(b)

```
---------------------------------
Block 0
Time 2017-12-03 11:40:08.461598
Transaction 28
Previous Hash 0
Hash z4PhNX7vuL3xVChQ1m2AB9Yg5AULVxXcg/SpIdNs6c5HONE8XYXysP+DGNKHfuwvY7kxvUdBeoGlODJ6+SfaPg==
---------------------------------
Block 1
Time 2017-12-03 11:40:08.461598
Transaction 40
Previous Hash z4PhNX7vuL3xVChQ1m2AB9Yg5AULVxXcg/SpIdNs6c5HONE8XYXysP+DGNKHfuwvY7kxvUdBeoGlODJ6+SfaPg==
Hash 0z4PhNX7vuL3xVChQ1m2AB9Yg5AULVxXcg/SpIdNs6c5HONE8XYXysP+DGNKHfuwvY7kxvUdBeoGlODJ6+SfaPg==
---------------------------------
Block 2
Time 2017-12-03 11:40:09.462788
Transaction 28
Previous Hash 0z4PhNX7vuL3xVChQ1m2AB9Yg5AULVxXcg/SpIdNs6c5HONE8XYXysP+DGNKHfuwvY7kxvUdBeoGlODJ6+SfaPg==
Hash 00z4PhNX7vuL3xVChQ1m2AB9Yg5AULVxXcg/SpIdNs6c5HONE8XYXysP+DGNKHfuwvY7kxvUdBeoGlODJ6+SfaPg==
---------------------------------
```

Fig. 8 (a) Program to generate blockchain and (b) the execution of the program

In program below, first we make the class of Block which structure is as we have defined before. The hash encoded is decided by random difficulty from 0 to 3. The first block is called genesis block is the head of the blockchain and contain no previous hash (just like in linear list). After the genesis block, then we can generate new block which previous hash is the hash of the previous block. This process then will make the blockchain and each block will have the data transaction and link to them to (We can make it like a graph).

B. *Bitcoin's Transaction with TDAG* [2]

Each application use different structure in how they store

assets to blockchain. This paper will only present on the most prominent blockchain, that is Bitcoin, the prototype of all blockchain systems in this world. Bitcoin combines the transaction, coin mining, and agreement on the ledger called Nakamoto Protocol. A block in Bitcoin can hold two types of transactions, which is:

- A coinbase transaction that transfers the unmined bitcoin to an address chosen by the miner of the corresponding block. This transaction can be done if there is solution to the proof-of-work puzzle for successful mining and if a numbers of Bitcoin transferred is correct to the height of the block. The first reward for the first mined block is 50 bitcoins.
- A regular transaction is usual transaction with the fees to put it in the block. It transfers Bitcoin from a set of input addresses to a set of output addresses. A regular transaction is valid if it includes a confirmation for each input and if it does not create new bitcoins.

From the reference [2], we say that in the TDAG modeling Bitcoin, there is a set of tuple which consist of (*addr*, *val*, *hash*, *height*) where *addr* is the address of the user, *val* is the amount of the bitcoin held in the state, *hash* is the cryptographic for security system and *height* is the index of the block or the number of the block.

The transaction fees and unmined bitcoin is associated to an address. The form of witness for coinbase transaction is the solution for the proof-of-work. Meanwhile, for regular transaction, the witness is the set of confirmations for the transfer of the bitcoin over the input and output addresses.

The TDAG for Bitcoin also contains producing and consuming edges with no observing edges. For a coinbase transaction, the input states are the unconsumed state of the unmined bitcoins and there is also a reward of fee states from the transactions included. One producing edge leads to a state for collecting the fees and the mining reward, another one to a state containing the remaining unmined bitcoins with its witness is the mining proof. For a regular transaction, the input states represents the transaction inputs and the output is the state corresponding to it with its witness is a set of confirmations. Also there is output state of the fee.

We say that each block $B_i = (MP, \{CBTX, RTX_1 .. RTX_N\})$ is a tuple composed by successful mining proof and a set of transaction containing coinbase transaction and a set regular transactions. Coinbase transaction contain a Bitcoin address and regular transaction contain address input, output, and set of confirmations.

In TDAG, we can represent a state as s a Bitcoin address that holds a group of bitcoins, a transaction fee or unmined bitcoins although a fee and unmined bitcoin practically not associated with the address. The genesis state represents a Bitcoin address holding the total unmined bitcoins in the Bitcoin system. Each witness represents either as a proof of successful mining or as a set of confirmation of the transaction occurs. There are also producing edges and consuming edges for each transaction.

We can model an execution of the Bitcoin system as the graph defined as follows :

- Each state is defined as a tuple (*addr*, *val*, *hash*, *height*) as definition aboves.
- Each witness is defined by a tuple (*txtype*, *F*) where *txtype* is the type of the transaction and determines the contents of *F*.
- Each edge is either consuming or producing edge.

The transaction graph determines the modeling of the possible transactions in a Bitcoin execution. For a coinbase transaction, we can model it as SIMO transaction. For a regular transaction, we can model it as SISO, SIMO, MISO, or MIMO transaction depend of the set of input and output addresses.

Then, to consider each transaction is valid or true, can be proven by one of these following conditions :

- If it is a regular transaction, then witness must contains a valid confirmation of each input state.
- If it is a coinbase transaction, then witness must constains a proof of successful mining.
- Each output state represents a positive number of bitcoins.
- The sum of bitcoins held at the input states must be equal to the sum of bitcoins held at the output states.
- Each output state contains the evaluation of the hash function over input states and the witness.

C. Modeling the Bitcoin Execution

Let us take an example. We assume that the block reward is fixed to a value of 50 bitcoins (first reward of the system). We can describe the process of the execution to TDAG although it will take longer time because it is more complex to draw the execution as TDAG. Let us see the Fig. 9

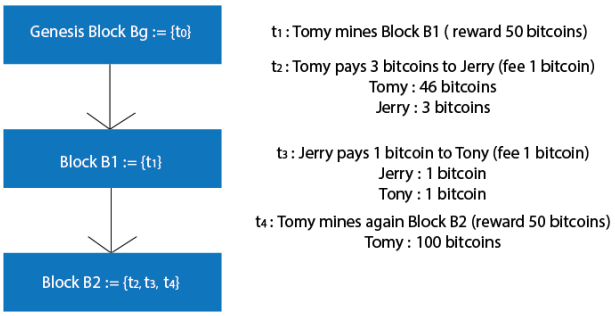

Fig. 9 Example of bitcoin execution

In the Fig. 9 above, we can make it to the TDAG model. There are 3 blocks with B_g is the genesis block, B_1 is the first block contains 1 transaction (coinbase transaction) and B_2 constains 2 regular transactions and 1 coinbase transaction. There are also t_0, t_1, t_2, and so on which will be interpreted below:

- $t_o := (\{s_g, s_0, w\}, \{(s_g, w), (w, s_0)\})$ represents the initialization of the transaction.
- $t_1 := (\{s_0, s_1, s_2, w_1\}, \{(s_0, w_1), (w_1, s_1), (w_1, s_2)\})$. This transaction SIMO issues a 50 bitcoins to Tomy after he has succesfully mined the block 1. s_1 contains the address of the numined bitcoind and s_2 contains the address of Tomy and his bitcoins.
- $t_2 := (\{s_2, s_3, s_4, s_5, w_2\}, \{(s_2, w_2), (w_2, s_3),(w_2, s_4),(w_2, s_5)\})$. This SIMO transaction shows that Tomy (s_2) pays 3 bitcoins to Jerry with fee of 1 bitcoin (s_3). Now Tomy has 46 bitcoins (s_4) and Jerry has 3 bitcoins (s_5).
- $t_3 := (\{s_4, s_6, s_7, s_8, w_3\}, \{(s_4, w_3), (w_3, s_6), (w_3, s_7), (w_3, s_8)\})$. This MIMO transaction shows that Jerry (s_5) pays 1 bitcoins to Charles with fee of 1 bitcoin (s_6). Now Jerry

has 1 bitcoin (s_7) and Charles also has 1 bitcoin (s_8).

- $t_4 := (\{s_1, s_3, s_6, s_9, s_{10}, w_4\}, \{(s_1, w_4), (s_3, w_4), (s_6, w_4), (w_4, s_9), (w_4, s_8)\})$. This MIMO transaction depicts the issue where Tomy has successfully mined new block (say block 2). Tomy claims the reward and also the fees (s_3, s_6) from transaction before. Now Tommy has 100 bitcoins (s_{10}).

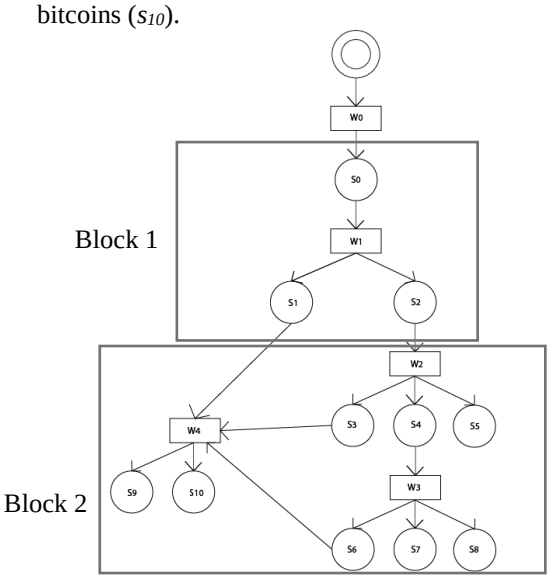

Fig. 10 The TDAG of the execution

D. *Example of Bitcoin Blocks in Real Life*

In practical use, blockchain is much more complex than the model in this paper. There are many ways to model the blockchain and to understand it. The ways above are only some ways to decribe how transaction in blockchain works.

To know from the TDAG whether individual has successfully mined the block is actually simple. We can see that if someone has successfully mined new block, the TDAG will show SIMO or can be MIMO transaction with the sum of the output is 2. It means that the output states consist of state of the unmined bitcoins and the other is the state of the set of addresses chosen by the miner.

There are also many rules in blockchain. For example Bitcoin, there is only one block which can be mined in 10 minutes. So in 1 day, there are only 144 block which can be mined. The block is consists of its hash, the time it is mined, previous hash, the index, and all the transactions included. In the program, we can think the structure of the block with the linked list of the transaction, date type, the hash, etc. The example can be looked at Fig. 7.

One of the implementation of the block is using linked list in its structure data. Each block in blockchain contains the list of all the transactions done. Each transactions in the list also contains the set of input addressess and the set of the output addresses, the type of the transactions, and the nominal.

Nowadays, Bitcoin the most prominent blockchain application in the world. Many transactions are done with using bitcoin. In 1 block, there are many transactions bundled. There are at least 20 transactions in 1 blocks. It is still a small number considering many transactions are done in bitcoin and only one blocks can be generated in 10 minutes. The other problem is the maximum memory the block can accommodate. That is why blockchain still needs to be developed.

IV. Conclusion

Graph has many applications in it. Graph can be used to model how the blockchain works. With network topology, transaction graph, and also trust management, we can model it into blockchain technology. Network topology can show that blockchain is a distributed network system. Transaction graph also helps to understand how the transaction is done in the execution of the blockchain. Trust management also helps in maintaining the trust and the security system. There are also many simple algorithm which can show how the blockchain works. Overall, these methods only show a simple model of blockchain as blockchain is very complicated and still new to all of us.

V. Acknowledgment

The author thanks Dr. Rinaldi Munir as the lecturer of the author's Discrete Mathematics class, for guidance in preparing this paper. The author would also thank for all the people who has submitted their papers in the internet as the reference to this paper.

References

[1] Rinaldi Munir, "Matematika Diskrit (Book style with paper title)", 5th ed. Bandung : Informatika Bandung, 2014, pp. 357-376

[2] Christian Cachin, *et al*., "The Transaction Graph for Modeling Blockchain Semantics (Submitted for paper)". [Online]. Available : https://eprint.iacr.org/2017/1070.pdf (Retrived 2 December 2017).

[3] Lior Yaffe, "Scaling a Blockchain vs Scaling a Tangle". [Online]. Available : https://medium.com/@lyaffe/scaling-a-blockchain-vs-scaling-a-tangle-8b7182eda980 (Retrieved 2 December 2017).

[4] Albert-László Barabási, "Network Science Th Barabási-Albert Model (Book style with paper title)". [Online]. Available : http://barabasi.com/f/622.pdf (Retrieved 2 December 2017).

[5] Nikolaos Alexopoulos, et al., "Beyond the Hype: On Using Blockchains in Trust Management for Authentication (Submitted for paper)". [Online]. Available : https://arxiv.org/pdf/1711.04591.pdf (Retrieved 2 December 2017).

[6] Cuneyt Gurcan Akcora, Yulia R. Gel, Murat Kantarcioglu, "Blockchain: A Graph Primer (Submitted for paper)". [Online]. Available : https://arxiv.org/pdf/1708.08749v1.pdf (Retrieved 2 December 2016).

[7] Jake Billings, "Image-based Proof of Work Algorithm for the Incentivization of Blockchain Archival of Interesting Images (Submitted for paper)". [Online]. Available : https://arxiv.org/ftp/arxiv/papers/1707/1707.04558.pdf (Retrieved 2 December 2017).

Pernyataan

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 3 Desember 2017

Joseph Salimin
13516037