mathematics

MDPI

Article

Axiomatization of Blockchain Theory

Sergey Goncharov † and **Andrey Nechesov** *,†

Sobolev Institute of Mathematics, 630090 Novosibirsk, Russia; s.s.goncharov@math.nsc.ru
* Correspondence: nechesov@math.nsc.ru
† These authors contributed equally to this work.

Abstract: The increasing use of artificial intelligence algorithms, smart contracts, the internet of things, cryptocurrencies, and digital money highlights the need for secure and sustainable decentralized solutions. Currently, the blockchain technology serves as the backbone for most decentralized systems. However, the question of axiomatization of the blockchain theory in the first-order logic has been open until today, despite the efficient computational implementations of these systems. This did not allow one to formalize the blockchain structure, as well as to model and verify it using logical methods. This work introduces a finitely axiomatizable blockchain theory $\mathbb{T}$ that defines a class of blockchain structures $\mathbb{K}$ using the axioms of the first-order logic. The models of the theory $\mathbb{T}$ are well-known blockchain implementations with the proof of work consensus algorithm, including Bitcoin, Ethereum (PoW version), Ethereum Classic, and some others. By utilizing mathematical logic, we can study these models and derive new theorems of the theory $\mathbb{T}$ through automatic proofs. Also, the axiomatization of blockchain opens up new opportunities to develop blockchain-based systems that can help solve some of the open problems in the fields of artificial intelligence, robotics, cryptocurrencies, etc.

Keywords: first order logic; blockchain; blockchain axiomatization; bitcoin; ethereum; XAI; artificial intelligence; AI; robotics; DeFi; IoT; smart contracts

MSC: 03C68

check for updates

Citation: Goncharov, S.; Nechesov, A. Axiomatization of Blockchain Theory. *Mathematics* **2023**, *11*, 2966. https://doi.org/10.3390/math11132966

Academic Editors: Vassily Lyubetsky, Vladimir Kanovei and Su Gao

Received: 6 June 2023
Revised: 25 June 2023
Accepted: 28 June 2023
Published: 3 July 2023

1. Introduction

The revolutionary technology of blockchain [1] is well known for providing a decentralized and transparent way of managing and maintaining information. Most people associate blockchain with cryptocurrencies such as Bitcoin [2–4] and Ethereum [5,6], but it has countless other applications beyond that. One of the most significant potential applications is facilitating the efficient tracking and management of supply chains, especially in fields such as food and pharmaceuticals. The popularity of DeFi [7] further highlights the possibilities of blockchain in finance [8], while governments [9] worldwide are exploring blockchain's potential from identity verification to tax collection. Smart contracts [10] based on blockchain technology could revolutionize the legal industry by providing secure and transparent mechanisms to create and enforce contracts. Some blockchains are designed exclusively for the internet of things (IoT) [11] as an efficient and secure solution for managing a huge amount of data generated by IoT devices.

Artificial intelligence algorithms have significantly impacted our daily routines as they provide prompt answers mostly by learning quickly on the basis of neural networks [12]. Nevertheless, the limitation of such systems lies in their high error rate, particularly when exposed to new data. Additionally, these systems do not explain their results and thus present a black box problem. The need for transparent and humanized AI algorithms is paramount to gaining trust in artificial intelligence. The focus now is on explainable artificial intelligence (XAI) [13] using hybrid methods with neural networks and high-level logic programming languages (such as L [14] and L* [15]) that provide clear execution logic, to ensure accurate results and logical explanations. To protect these programs from code alterations and breaches, blockchain structures [16] secured through consensus algorithms are required. It is imperative to employ operational decentralized blockchain implementations such as Ethereum and Bitcoin, and also express these structures mathematically with axioms for further mathematical analysis, enhancement, and protection.

Attempts to axiomatize the blockchain have been made for a long time. One of the first works in this direction was the work [17], in which a blockchain structure of a special type was defined, and a system of axioms was set using modal operators of non-classical logic. However, it was the simplest structure, not connected in any way with efficient systems such as Bitcoin and Ethereum (PoW version). These were the first attempts at axiomatization, which gave rise to further study in this direction. Three axioms for decentralized protocols were presented in the work [18]: axiom of Anonymity, axiom of Robustness to Sybil Attacks, and axiom of Robust to Merging. Further, axioms for automated market makers within decentralized environments were proposed in the work [19] but all these works almost did not touch upon the axiomatic nature of real blockchain models used in Bitcoin, Ethereum, and other well-known crypto systems.

The problem of axiomatization of the blockchain theory with the help of the first-order logic remained open. Axiomatization would bridge the gap between logic and computable implementations such as Bitcoin and Ethereum blockchains. Moreover, the formalization of the theory makes it possible to use a unified interpretation of signature symbols to build blockchain libraries in high-level programming languages for modeling and verifying various types of blockchains. As shown in related work Section 6, this is one of the key areas of research today. We offer effective tools to solve these problems.

The axiomatization in the work [20] prompted us to construct an axiomatization of blockchain structures. In the current work, the blockchain theory $\mathbb{T}$ is specified using the axioms of the first-order logic. There are several reasons for this. First, most mathematicians work in the terminology of the first-order logic. Second, the main logical results can be transferred here as well. Third, with the help of predicate calculus, one can obtain new statements and prove new theorems. Fourth, this theory defines a class of structures K such that $Th(\mathbb{K}) = \mathbb{T}$. This class will be called the class of blockchain structures.

The paper is structured as follows. Section 2 gives the basic definitions related to the blockchain, and also presents various consensus algorithms. Section 3 is devoted to the axiomatization of the blockchain theory. This section defines blockchain theory $\mathbb{T}$ and gives a list of axioms using first-order logical formulas. It also defines the concept of a class of blockchain structures $\mathbb{K}$. Section 4 shows that the Bitcoin blockchain and the Ethereum blockchain are models of our theory $\mathbb{T}$. Section 5 proves the complete or partial independence of most axioms of our theory. Section 6 describes related works and the potential application of the blockchain axiomatization to them. Section 7 discusses the perspectives and applications of our research. Section 8 is the conclusion.

2. Blockchain Basics

Since many different types of blockchain have appeared [21] in the world over the past 15 years, in this work, by blockchain we mean a special tree of blocks that are interconnected using cryptographic hashes. Each block of this special tree contains a cryptographic hash of the previous block, transaction data (generally represented as a Merkle tree [22,23], where data nodes are represented by leaves), a timestamp, etc. [24]. In this special tree, with the help of some computable procedure, we can select the "true" blockchain (a chain or subtree of blocks with only approved transactions). In the case of Bitcoin, the blockchain coincides with the longest chain of blocks ("true" blockchain) in the tree of blocks. In the case of Ethereum, the blockchain also contains ommer blocks (Figure 1, more details in the works [25–27]) that are not part of the "true" blockchain, but play a key role in the selection of blocks with approved transactions.

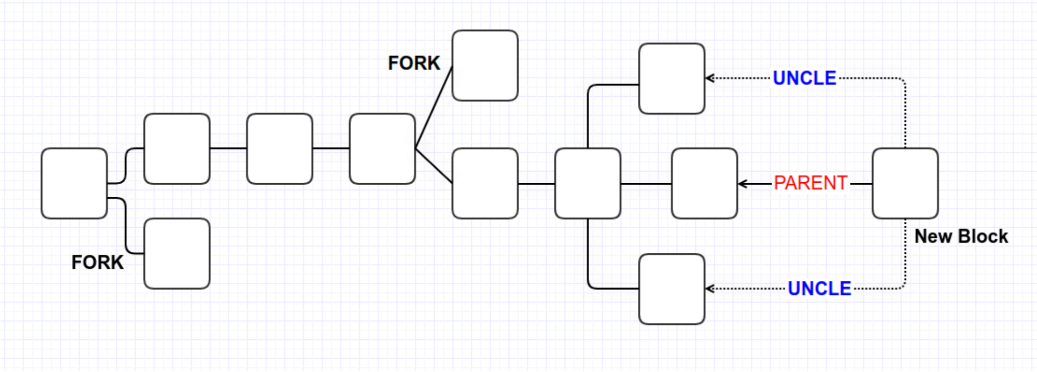


Figure 1. Ommer blocks (or uncle blocks) in the Ethereum blockchain.

Since we have a computable selection procedure for choosing the "true" subtree of blocks, we have some consensus algorithm which has been fixed on the choice of a subtree in the tree of blocks. This consensus can be reached using various algorithms such as: Proof of Work (PoW) [2], Proof of Stake (PoS) [6], Proof of Activity (PoA) [28], etc. This work deals only with the blockchain structures using the PoW consensus algorithm. Moreover, almost all leading cryptocurrency projects, such as Bitcoin, Ethereum (PoW version), Ethereum Classic, and others use a PoW consensus algorithm. All blocks in the tree of blocks are related to each other using a sequence of computable relations that establishes a relationship between two blocks. There is some computable procedure for adding a new block to the tree of blocks. In addition, there is some computable procedure for comparing two trees of blocks in terms of complexity, and so on. All necessary operations and relationships for blockchain structures will be defined below.

Figure 2 shows different options for building a blockchain on a tree of blocks depending on the definition of the blockchain operation:

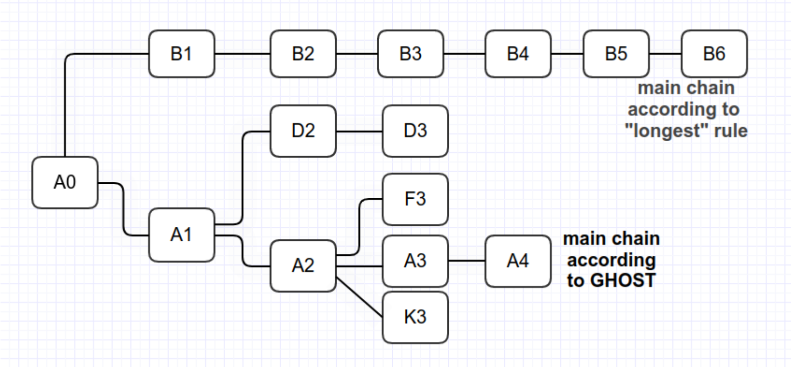

B1
B2
B3
B4
B5
B6
main chain according to "longest" rule
D2
D3
A0
F3
A1
A2
A3
A4
main chain according to GHOST
K3

Figure 2. Longest path or GHOST algorithm.

The main chain according to GHOST (or Greedy Heaviest Observed Sub-Tree) [29]: the longest branch of blocks is inductively chosen so that at the $i+1$-th step, the block b_{i+1} lies in the most branching subtree with the root being the block b_i.

The main chain is selected according to the "longest" rule: the "longest" branch in the tree of blocks is selected.

As can be seen from Figure 2, the GHOST algorithm is sometimes more preferable to the "longest" rule algorithm. There are other blockchain selection algorithms, so we are going to build a single blockchain theory that covers all these situations.

3. Blockchain Axiomatization

In the blockchain theory, there are two main concepts: blocks and trees of blocks. Variables for blocks will be denoted as x, y, z, etc. By default, we assume that in the axioms where such variables are presented, the $Block$ predicate is true for them. Variables for the trees of blocks will be denoted as T, T_1, …, T_n. We also assume that in all axioms, the predicate $Tree_0$ is true for these variables. Additionally, all trees of blocks will have one common root (genesis block), which will be given by the constant symbol 0.

Let σ be the final signature of the following form:

$$\sigma = \{0, Tree_0^{(1)}, Block^{(1)}, B^{(1)}, BC^{(1)}, \leq_T^{(3)}, \in_t^{(2)}, \equiv^{(2)}, \wedge^{(3)}, P^{(3)}, add^{(3)}\} \tag{1}$$

In the following chapters, it will be shown that the interpretation of signature symbols for models of the blockchain theory can be very different. However, in most cases, signature symbols "intuitively" define the following:

- B—function that builds a blockchain from a tree of blocks T.
- BC—function that builds a "true" blockchain from a blockchain $B(T)$.
- $x \in_t T$—binary relation of membership of block x in the tree of blocks T.
- $P(T, x, y)$—parent relationship, which verifies that the block x is the parent for the block y relative to the tree of blocks T. Note that the relation $P(T, x, y)$ can also be true if the block $y \notin_t T$.
- $add(T, x, y)$—function that adds the child block y to block x in the tree of blocks T.
- $\wedge(T, x, y)$—operation of finding the greatest lower block for blocks x and y in the tree of blocks T.
- 0—constant symbol highlighting the genesis block.
- $\leq_T$—linear preorder on trees of blocks.
- $\equiv$—equivalence relation on blocks.

The equality $=$ is understood as a character-by-character comparison of two elements from the main set of the model.

Let us introduce the following notation:

- $\leq_b (T, x, y) : \ \wedge(T, x, y) = x$
- $=_b (T, x, y) : \leq_b (T, x, y) \& \leq_b (T, y, x)$
- $T_1 =_T T_2 : \ (T_1 \leq_T T_2) \& (T_2 \leq_T T_1)$

3.1. Blockchain Axioms

Axioms of Blockchain Theory $\mathbb{T}$

- Blockchain axiom 1: $B(B(T)) = B(T)$
- Blockchain axiom 2: $x \in_t B(T) \to x \in_t T$
- Blockchain axiom 3: $Tree_0(B(T))$
- Blockchain axiom 4: $x \in_t BC(B(T)) \to x \in_t B(T)$
- Blockchain axiom 5: $Tree_0(BC(B(T)))$
- Equiv axiom 1: $x \equiv x$
- Equiv axiom 2: $x \equiv y \to y \equiv x$
- Equiv axiom 3: $(x \equiv y) \& (y \equiv z) \to (x \equiv z)$
- Order axiom 1: $T \leq_T T$
- Order axiom 2: $(T_1 \leq_T T_2) \vee (T_2 \leq_T T_1)$
- Order axiom 3: $(T_1 \leq_T T_2) \& (T_2 \leq_T T_3) \to T_1 \leq_T T_3$
- Order axiom 4: $\forall x \in_t T_1 \forall y \ (add(T_1, x, y) = T_2) \to T_1 \leq_T T_2$
- Zero axiom 1: $0 \in_t T$
- Zero axiom 2: $\forall x \in_t T \ \leq_b (T, 0, x)$
- Boundary axiom 1: $\forall x, y \in_t T \ \wedge (T, x, y) = \wedge(T, y, x)$
- Boundary axiom 2: $\forall x, y, z \in_t T \ \wedge (T, \wedge(T, x, y), z) = \wedge(T, x, \wedge(T, y, z))$
- Boundary axiom 3: $\forall x \in_t T \ \wedge (T, x, x) = x$
- Boundary axiom 4: $\forall x, y, z \in_t T \ \leq_b (T, \wedge(T, x, y), \wedge(T, y, z)) \to \wedge(T, x, y) = \wedge(T, x, z)$
- Boundary axiom 5: $\forall x, y \in_t T \ Block(\wedge(T, x, y)) \& (\wedge(T, x, y) \in_t T)$

- Generation axiom 1:

$$\begin{aligned}&\forall x \in_t T_1 \forall y \exists T_2\ P(T_1,x,y)\&(\forall z \in_t T_1\ z \not\equiv y) \rightarrow\\&\rightarrow (add(T_1,x,y) = T_2 \& y \in_t T_2)\&P(T_2,x,y)\&\\&\&(\forall z\ z \in_t T_1 \leftrightarrow (z \in_t T_2)\&(z \neq y))\&(\forall z,t \in_t T_1\ \ \leq_b (T_1,z,t) \leftrightarrow \leq_b (T_2,z,t))\end{aligned}$$

- Generation axiom 2:

$$\begin{aligned}&\exists T_1 \exists x,y \in_t T_2\ P(T_2,x,y)\&(\neg\exists z \in_t T_2\ (z \neq y)\& \leq_b (T_2,y,z)) \rightarrow\\&\rightarrow (add(T_1,x,y) = T_2)\&P(T_1,x,y)\&(x \in_t T_1)\&(y \notin_t T_1)\&\\&\&(\forall z\ z \in_t T_1 \leftrightarrow (z \in_t T_2)\&(z \neq y))\&(\forall z,t \in_t T_1\ \ \leq_b (T_1,z,t) \leftrightarrow \leq_b (T_2,z,t))\end{aligned}$$

- Parent axiom:

$$\begin{aligned}&\forall x,y \in_t T\ P(T,x,y) \leftrightarrow (x \neq y)\& \leq_b (T,x,y)\&\\&\&(\neg\exists z \in_t T\ \ \leq_b (T,x,z)\& \leq_b (T,z,y)\&(x \neq z)\&(z \neq y))\end{aligned}$$

- Identity axiom 1: $\forall x,y \in_t T\ \ =_b (T,x,y) \leftrightarrow x = y$
- Identity axiom 2: $\forall x,y \in_t T\ x = y \leftrightarrow x \equiv y$

3.2. Blockchain Axioms: Some Informal Remarks

Blockchain axiom 1 says that once a blockchain $B(T)$ is built for a tree T, then the next application of function B to $B(T)$ will produce the same blockchain.

Blockchain axiom 2 says that all the blocks that lie in the blockchain $B(T)$ lie in the tree T.

Blockchain axiom 3 says that the result of applying the function B to the tree T is the tree.

Blockchain axiom 4 says that all the blocks that lie in the "true" blockchain $BC(B(T))$ lie in the blockchain $B(T)$.

Blockchain axiom 5 says that the "true" blockchain $BC(B(T))$ is a tree.

Order axiom 4 states that for any two trees T_1 and T_2 where T_2 is obtained from T_1 using the addition operation $add(T_1,x,y)$ for block $x \in T_1$ and some block y: the complexity of tree T_1 does not exceed the complexity T_2.

Boundary axioms 1–5 define a tree structure for model elements using the operation $\wedge$.

Generation axiom 1 (*the axiom of building a larger tree*) says that for every tree T_1, every block x of T_1, and every block y is not from T_1 if a number of conditions in Generation axiom 1 are met, then there is a tree T_2 that contains only the blocks of the tree T_1, as well as the block y.

Generation axiom 2 (*the axiom of building a smaller tree*) informally states that for every tree T_2 and any leaf block y in T_2, it is possible to construct a smaller tree T_1 that does not contain block y, such that tree T_2 contains only those blocks that are in T_1 as well as block y.

Parent axiom says that in any tree T and for any blocks x,y from T, the relation $P(T,x,y)$ is true if and only if $\leq_b (T,x,y)$ for $x \neq y$ and there is no block z lying between them with respect to $\leq_b$.

3.3. Axiomatizable Class of Blockchain Structures

Denote by $\mathbb{T}$ the theory given by the axioms of the blockchain described above. This theory $\mathbb{T}$ of signature σ defines a class of structures $\mathbb{K}$ of signature σ for which $Th(\mathbb{K}) = \mathbb{T}$ will be satisfied. The class $\mathbb{K}$ is referred to as *a class of blockchain structures*. Therefore, any structure $<\mathfrak{M},\sigma>$ from $\mathbb{K}$ is referred to as *a blockchain structure*. Moreover, any structure $<\mathfrak{B},\sigma \cup \sigma^*>$ for some finite signature σ^* is referred to as *a blockchain structure* if the structure $<\mathfrak{B},\sigma> \in \mathbb{K}$.

4. Models of Blockchain Theory $\mathbb{T}$

In this section, we will show that the theory $\mathbb{T}$ is consistent. To show this, we will build step by step the model $<\mathfrak{B}, \sigma>$ of the theory $\mathbb{T}$, which is a computably isomorphic real copy of the bitcoin blockchain model [2] where the longest branch in the block tree specifies the blockchain (Figure 3) and the "true" blockchain is exactly this blockchain.

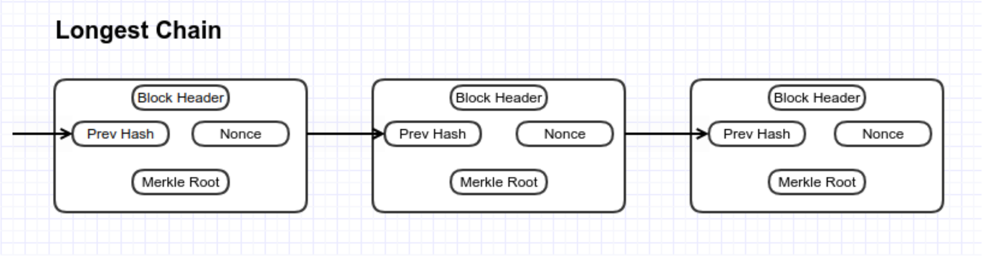


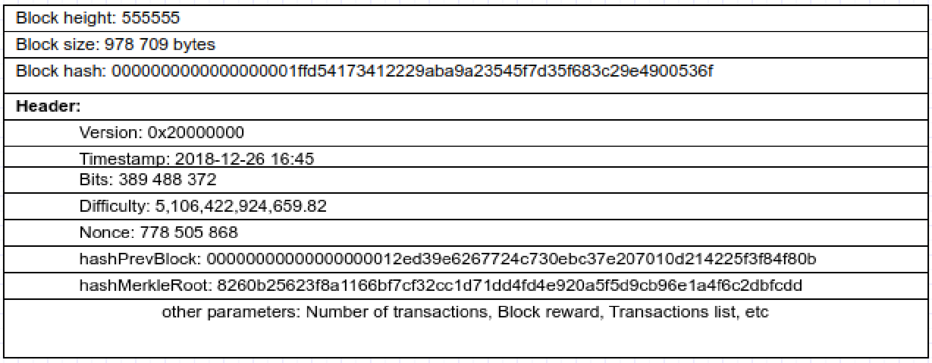
Figure 3. Bitcoin blockchain: the longest branch of the tree of blocks.

Then, using the model $<\mathfrak{B}, \sigma>$, we will construct a model $<\mathfrak{G}, \sigma>$ of the theory $\mathbb{T}$, in which the operation B will be implemented according to the algorithm *GHOST* (Figure 2).

Let Σ be some finite alphabet and the set $M \subseteq \Sigma^*$. Let us describe the hereditarily finite list model $<\mathfrak{B}, \sigma>$ with the following elements: $M \cup HW(M)$, where $HW(M)$ is the set of hereditarily finite lists, defined inductively with the help of elements of the set M (more details in the work [30]).

In this model, using the predicates Block and Tree$_0$, two kinds of list elements will be distinguished: blocks and trees of blocks (consisting of a finite number of blocks). The block b, as shown in Figure 4, consists of a finite set of *name:value*; therefore, the block code has the following form:

$$[b] : << Block\ height, h_b >, < Block\ size, s_b >, \cdots > \tag{2}$$



Block height: 555555
Block size: 978 709 bytes
Block hash: 0000000000000000001ffd54173412229aba9a23545f7d35f683c29e4900536f
Header:
Version: 0x20000000
Timestamp: 2018-12-26 16:45
Bits: 389 488 372
Difficulty: 5,106,422,924,659.82
Nonce: 778 505 868
hashPrevBlock: 00000000000000000012ed39e6267724c730ebc37e207010d214225f3f84f80b
hashMerkleRoot: 8260b25623f8a1166bf7cf32cc1d71dd4fd4e920a5f5d9cb96e1a4f6c2dbfcdd
other parameters: Number of transactions, Block reward, Transactions list, etc

Figure 4. Bitcoin block structure.

Trees of blocks will also be stored as lists, which are defined inductively as follows:
Base of Induction: the tree T consists of one block b, then its list code will be $< [b] >$.
Step of induction: if the tree T with the root b consists of more than one block, and $T_1, \ldots, T_n$ are all various subtrees listed from left to right, whose roots are directly connected to the root block b of the tree T. Then, the code of the tree T has the following form:

$$[T] : < [b], [T_1], \ldots, [T_n] > \tag{3}$$

Remark 1. *Everywhere below, by blocks and trees of blocks, we will mean their list codes, so we will omit the square brackets for these elements.*

The interpretation of the signature symbols in the model $<\mathfrak{B}, \sigma>$ is as follows:

- The predicate Block selects those and only those list elements that have the form (2):

- The value of the parameter *Block hash* is obtained as the value of the standard hash function f [31] for Bitcoin, which is shown in Figure 5:

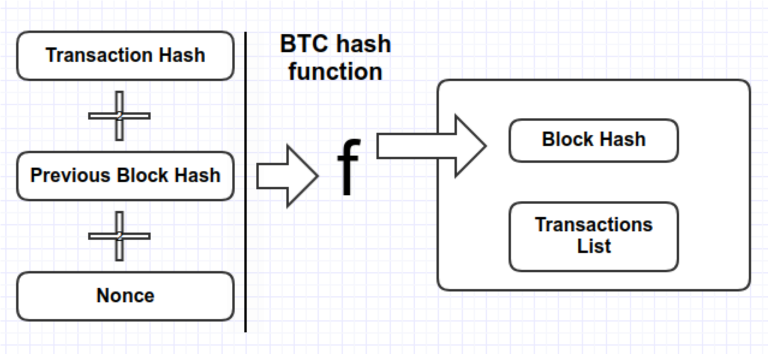


Figure 5. Bitcoin hash function f.

- The number of zeros at the beginning of each hash must be consistent with the current block difficulty
- Merkle hash [22] for transactions list in the block must match the parameter *hashMerkleRoot*.
- The constant symbol 0 highlights the genesis block [32] in the Bitcoin blockchain with the following parameter and the value:

$$\text{Block hash: } 000000000019d6689c085ae165831e934ff763ae46a2a6c172b3f1b60a8ce26f \quad (4)$$

- The predicate $Tree_0$ checks the correctness and consistency of all blocks in T:
  - The root of the tree must be a genesis block.
  - The tree is checked for the correctness of the list code (3).
  - The tree is checked for equivalent blocks.
  - The complexity of each block must satisfy the following formula [33]:

    $$newDiffuculty = oldDifficulty \otimes \frac{20,160}{t}$$

    The difficulty is recalculated on each block with the parameter *Block height* equal to n, where $n-1$ is a multiple of 2016, t is the time in minutes spent getting the last 2016 blocks.
  - The tree is checked for consistency of the *Block hash* and *hashPrevBlock* parameters with the tree structure, where the child block must immediately follow the parent block.
  - All blocks in the tree T, all addresses, transactions, amounts, signatures, and public keys must be agreed upon.
- The value of the operation $B(T)$ is the first leftmost longest branch found in the tree T.
- The function BC is defined as follows: $BC(B(T)) = B(T)$ (“true” blockchain is equal to blockchain).
- The binary relation $T_1 \leq_T T_2$ is true if the length of the blockchain $B(T_1)$ does not exceed the length of the blockchain $B(T_2)$.
- The membership relation $\in_t$ of block b in the tree T is true if the list code of the block b is included in the list code of the tree T as an element participating in the construction of this list code of the tree T.
- An equivalence relation $\equiv$ on blocks is true if the hashes of blocks are the same.
- The operation $add(T,x,y)$ of adding a block y to the block x in the tree T is uniquely defined. Without loss of generality, we assume that the operation is defined as follows: a new block y is added to the list code of the block x as the rightmost child if the result of this operation is a new tree for which the formula $Tree_0(add(T,x,y))$ is true, else $add(T,x,y) = T$.

- The predicate $P(T, a, b)$ is implemented using the following algorithm:
  - If $a, b \in_t T$, the predicate $P(T, a, b)$ is true if and only if the block b immediately follows the block a in the tree T.
  - If a block $a \in_t T$ and a block $b \notin_t T$, then the following should hold:

$$\mathfrak{B} \models P(T, a, b) \Leftrightarrow \mathfrak{B} \models Tree_0(add(T, a, b)) \tag{5}$$

- The operation of the greatest lower block $\wedge(T, x, y)$ is defined as standard [20]. The result of the operation $\wedge(T, x, y)$ where $x, y \in_t T$ is the such block z, which is the greatest lower block for blocks of x and y by relation $\leq_b$ in the tree of blocks T.

Theorem 1. *<$\mathfrak{B}, \sigma$> is a model of the theory $\mathbb{T}$.*

Proof. Let us show that in <$\mathfrak{B}, \sigma$> all axioms of the theory $\mathbb{T}$ are satisfied.

Blockchain axioms 1–5 are satisfied by construction.

Equiv axioms 1–3 are satisfied: blocks are considered equivalent if their hashes coincide.

Order axioms 1–3 are satisfied by construction.

Order axiom 4 follows from the interpretation of the relation $\leq_T$ that the length of $B(T_1)$ does not exceed the length of $B(T_2)$. Hence, $T_1 \leq_T T_2$.

Zero axioms 1–2 are satisfied due to the interpretation of the $\wedge$ operation.

Boundary axioms 1–5 are satisfied by construction.

Generation axiom 1 is satisfied due to the given interpretations of the signature symbols on the model <$\mathfrak{B}, \sigma$>.

Generation axiom 2 is satisfied due to the given interpretations of the signature symbols on the model <$\mathfrak{B}, \sigma$>.

Obviously, the Parent axiom, given the above interpretation of the signature symbols, is satisfied on the model <$\mathfrak{B}, \sigma$>.

Identity axioms 1–2 are satisfied by definition of the $=_b$, $=$ and $\equiv$ relations.

Thus, all the axioms are satisfied, which means that the theory $\mathbb{T}$ is satisfied on the model <$\mathfrak{B}, \sigma$>. □

Let the model <$\mathfrak{G}, \sigma$> be obtained from <$\mathfrak{B}, \sigma$> as a result of redefining the operations B and BC, and all other interpretations of signature symbols remain unchanged. The operation B is specified via the *GHOST* algorithm (returns the heaviest subtree). The operation BC produces the longest branch ("true" blockchain) in the heaviest subtree $B(T)$.

Corollary 1. *The theory $\mathbb{T}$ is satisfiable in the model <$\mathfrak{G}, \sigma$>.*

Proof. The operations B and BC only appear in Blockchain axioms 1–5. Due to the fact that the interpretations of the remaining signature symbols remained unchanged, all other axioms are satisfied in <$\mathfrak{G}, \sigma$>.

Blockchain axioms 1–5 are also satisfied by construction. □

Consider the model <$\mathbb{ETH}, \sigma$> that is a computably isomorphic copy of Ethereum blockchain (PoW version). This model is built similarly to the model <$\mathfrak{B}, \sigma$>, but with its own parameters that satisfy the ethereum blockchain properties: hash functions, block generation time, block structure, smart contracts, ommer blocks, etc.

Similarly, we can build the model <$\mathbb{ETC}, \sigma$> for Ethereum Classic blockchain.

Corollary 2. *<$\mathbb{ETH}, \sigma$> is a model of the theory $\mathbb{T}$.*

Proof. The main difference between Ethereum blockchain and Bitcoin blockchain is the definition of blockchain operations B and BC.

In the model <$\mathfrak{B}, \sigma$> these operations output the longest branch of the tree, but in the model <$\mathbb{ETH}, \sigma$> operation B outputs the subtree with ommer blocks (Figure 1) and operation BC outputs the heaviest chain in this subtree.

By construction, we get that the theory $\mathbb{T}$ is satisfiable in the model $<\mathfrak{G}, \sigma>$. □

Corollary 3. $<\mathbb{ETC}, \sigma>$ *is a model of the theory* $\mathbb{T}$*.*

Proof. The proof of this statement completely coincides with the proof of Corollary 2. □

5. Independence of the Axioms of the Theory $\mathbb{T}$

In this section, for some axioms of the theory $\mathbb{T}$, we will show their complete or partial independence. Due to the fact that in the model $<\mathfrak{B}, \sigma>$ all axioms of our theory $\mathbb{T}$ are satisfied, then to prove the independence of one or another axiom, we will take its negation and build an appropriate model using the model $<\mathfrak{B}, \sigma>$ as a basic one, in which we will change the interpretation of some signature symbols. If all the remaining axioms and the negation of the basic axiom are satisfied in the constructed model, then the independence will follow.

Lemma 1. *Blockchain axiom 1 is independent.*

Proof. Using the model $<\mathfrak{B}, \sigma>$, we construct the model $<\mathfrak{B}^{\neg bla1}, \sigma>$. Below, we redefine the interpretation of some signature symbols in the model $<\mathfrak{B}, \sigma>$.

The relation B is defined as follows:

- If the tree T has incomparable blocks with respect to $\leq_b$, then $B(T)$ will return the first longest branch of T.
- If the tree T consists of a single block, then $B(T) = T$.
- If all blocks in the tree T are comparable with each other with respect to $\leq_b$ and the number of blocks in the tree T is greater than one, then $B(T)$ will produce this tree without a maximum element with respect to $\leq_b$.

The interpretations of the rest of the signature symbols will be exactly the same as in the model $<\mathfrak{B}, \sigma>$.

Let us show that the negation of Blockchain axiom 1 is satisfied on the model $<\mathfrak{B}^{\neg bla1}, \sigma>$:

$$\mathfrak{B}^{\neg bla1} \models \exists T \; B(B(T)) \neq B(T) \tag{6}$$

The expression (6) will be true if T is any tree with the number of blocks greater than 2.

Blockchain axioms 2–5 are satisfied by construction.

The remaining axioms of the theory $\mathbb{T}$ are satisfied because the relation B does not participate in them. □

Lemma 2. *Blockchain axiom 2 is independent.*

Proof. As before, we build the model $<\mathfrak{B}^{\neg bla2}, \sigma>$ using the model $<\mathfrak{B}, \sigma>$.

Let us change the block condition in the predicate $Tree_0$ by removing the check for the parameter *Block hash* on the number of leading zeros. Now, it is not necessary to require the number of zeros at the beginning of *Block hash* depending on the value of the parameter *Diffuculty*.

Let n_0 be some natural number. Since all trees of blocks in the main set of the model $<\mathfrak{B}^{\neg bla2}, \sigma>$ have a finite number of blocks, we define operation B as follows:

- Find the first longest branch in the tree T.
- Find a leaf element in this branch.
- Replace the value *Nonce* in this block with n_0.
- Find the new value *Block hash* for this element using the hash function f.
- If there was no block with this name in the tree T, then we replace this block with a block with new parameters *Nonce* and *Block hash*, otherwise we leave the leaf element unchanged.

It is clear that with this interpretation, taking into account the hashing width of the hash function f, there will be a tree T for which the negation of Blockchain axiom 2 will be satisfied.

All other signature symbols will be interpreted in the same way as in the model $<\mathfrak{B}, \sigma>$. Changing the requirement for the number of leading zeros in the *Block Hash* does not affect the remaining axioms in any way. Therefore, all remaining axioms will be satisfied in $<\mathfrak{B}^{\neg bla2}, \sigma>$. □

Lemma 3. *Blockchain axiom 3 is independent.*

Proof. As before, we build the model $<\mathfrak{B}^{\neg bla3}, \sigma>$ using the model $<\mathfrak{B}, \sigma>$.

The function $B(T)$ of the new model for the tree T is defined through the function $B(T)$ in the model $<\mathfrak{B}, \sigma>$, where the final value is obtained by replacing the largest element with zero block 0.

The function BC is defined as a constant function, and its value will be the subtree with only zero block 0.

Now, we can easily check the satisfiability of all the remaining axioms of the blockchain. In this case, the negation of Blockchain axiom 3 will be satisfied. □

Lemma 4. *Blockchain axiom 4 is independent.*

Proof. To prove this statement, it suffices to define the value of the function BC as a tree consisting of two elements of the zero block 0 and a new block, which was not in the tree T. □

Lemma 5. *Blockchain axiom 5 is independent.*

Proof. Let the function BC replace the zero block 0 in the tree $B(T)$ with some other block. □

Lemma 6. *Order axiom 4 is independent.*

Proof. To prove this lemma, it suffices to define the relation $\leq_T$ in the new model $<\mathfrak{B}^{\neg oa4}, \sigma>$ as follows:

$$T_1 \leq_T T_2 \leftrightarrow |B(T_1)| \geq |B(T_2)|$$

where $|B(T)|$—the number of blocks in the blockchain $B(T)$.

If the rest of the signature symbols are interpreted as in the model $<\mathfrak{B}, \sigma>$, then in this case in the model $<\mathfrak{B}^{\neg oa4}, \sigma>$ all the axioms of the blockchain theory $\mathbb{T}$ will be satisfied, except for Order axiom 4. Instead, the negation of its negation will be satisfied. □

The axioms of Generation 1–2 and Parent axiom depend very strongly on each other and are quite large, so we will show that each of them is independent with respect to the set of axioms, which is obtained from the set of axioms S of the theory $\mathbb{T}$ as a result of deleting the axioms of Generation 1–2 and Parent axiom. Denote the new set as S^{-G}.

Lemma 7. *Generation Axiom 1 is independent with respect to the set of axioms S^{-G}.*

Proof. Let us build the $<\mathfrak{B}^{-G}, \sigma>$ model using the model $<\mathfrak{B}, \sigma>$.

In this model, all axioms from the S^{-G} set will be satisfied, as well as the negation of the Generation Axiom 1 axiom, which has the following form:

$$\begin{aligned}
&\exists T_1 \exists x \in_t T_1 \exists y \forall T_2\ P(T_1, x, y) \& (\forall z \in_t T_1\ z \not\equiv y) \& \\
&\& (\neg(add(T_1, x, y) = T_2 \& y \in_t T_2) \vee \neg P(T_2, x, y) \vee \\
&\vee \neg(\forall z\ z \in_t T_1 \leftrightarrow (z \in_t T_2) \& (z \neq y)) \vee \\
&\vee \neg(\forall z, t \in_t T_1\ \leq_b (T_1, z, t) \leftrightarrow \leq_b (T_2, z, t)))
\end{aligned}$$

The predicate $Tree_0$ will be redefined in the same way as in Lemma 2, removing the check of the parameter *Block hash* for the number of leading zeros in the tree of blocks.

We define the relation $P(T_1, x, y)$ as follows:

- $x, y \in_t T_1$—the truth of this relation remains the same as in the model $<\mathfrak{B}, \sigma>$
- $x \in_t T_1, y \notin_t T_1$—the truth of this relation will be given as follows:
  - check the possibility of adding block y as in $<\mathfrak{B}, \sigma>$.
  - checks the additional possibility of adding block t as a child block to y with an empty set of transactions. The parameter *Nonce* in the block t contains the smallest natural number such that the parameter *Hash block* of the block t does not coincide with the hash of any block from the tree T_1 or with the hash of the block y.

We define the operation $add(T_1, x, y)$ as follows:

- if the relation $P(T_1, x, y)$ is true and y not in T_1, then the result of the function $add(T_1, x, y)$ for y not in T_1 is a tree T_2, where in the tree T_1 the block y is added to the block x, and the block t is added to the block y.

The negation of Generation Axiom 1 will be satisfied by construction, due to the fact that if there is some tree T_2 where $add(T_1, x, y)$ equals T_2 and y not in T_1 then in T_2 addition to y, there is also a block t not from T_1. □

Lemma 8. *Generation Axiom 2 is independent with respect to the set of axioms $S^{\{-G\}}$.*

Proof. Let us show that if we take the constructed model $<\mathfrak{B}^{-G}, \sigma>$ from the proof in Lemma 7 then the negation of Generation Axiom 2 will be satisfied.

Informally, Generation axiom 2 says that for any tree T_2 with n vertices where $n \geq 1$ there exists a tree T_1 with $n-1$ vertices and blocks $x, y \in T_2$ such that $add(T_1, x, y) = T_2$.

Take the negation of Generation Axiom 2:

$$\begin{aligned}
&\exists T_2 \forall T_1 \forall x, y \in_t T_2\ P(T_2, x, y)\& \\
&\&(\neg \exists z \in_t T_2\ (z \neq y)\& \leq_b (T_2, y, z))\& \\
&\&(\neg(add(T_1, x, y) = T_2) \vee \neg P(T_1, x, y) \vee \neg(x \in_t T_1) \vee \\
&\vee \neg(y \notin_t T_1) \vee \neg(\forall z\ z \in_t T_1 \leftrightarrow (z \in_t T_2)\&(z \neq y)) \vee \\
&\vee \neg(\forall z, t \in_t T_1\ \leq_b (T_1, z, t) \leftrightarrow \leq_b (T_2, z, t)))
\end{aligned}$$

Taking into account the interpretation of the relation P and the operation add in the model $<\mathfrak{B}^{-G}, \sigma>$, we construct a tree T_2 that satisfies the negation of Generation Axiom 2.

Let us take a tree T_2 consisting of two nodes: the genesis block and the block referring to it. For this tree, taking into account the interpretation of the operation add with an additional block t the negation of Generation axiom 2 is satisfied. The remaining axioms from the set S^{-G} are also satisfied. □

Lemma 9. *Parent Axiom is independent with respect to the set of axioms $S^{\{-G\}}$.*

Proof. The independence of Parent axiom comes from the fact that there are no more axioms in $S^{\{-G\}}$ with the relation P. Therefore, the relation P can be defined in any way. Including so that the negation of Parent axiom will be satisfied and all other axioms $S^{\{-G\}}$ are satisfied. □

6. Related Works and Axiomatization Capabilities

Today, there are so many works on blockchain technology and so few on its axiomatization. Therefore, it makes sense to consider related works on the topic of blockchain, multi-party blockchain-based systems, consensus algorithms, smart contracts, and protocols. Let us start with the work [34] on modeling multi-party blockchain-based systems. The authors present two schemes for modeling such systems using logical transitions on their diagrams, which make it possible to better understand the advantages of blockchain over other data storage and processing solutions. Also note that these schemes do not fully describe all types of blockchains. From the point of view of modeling, in our work, blockchain axioms can be viewed as abstract modeling in which the main properties of the blockchain are revealed. Without an explicit analysis of these properties, it is not always clear what role blockchain plays in multi-party blockchain-based systems.

In work [35], the authors present a way to build a rolling blockchain as a viable alternative to the classic one. Rolling blockchain solves the problem of scalability by reducing the amount of information stored in the blockchain. In our opinion, the axiomatization of the rolling blockchain using first-order logic formulas would also help to more clearly see the possibilities of this technology and its security and scalability.

In work [36], with the help of logical programs of a special language based on logical rules, the bytecode is split into blocks, which allows us to control the computational costs and other parameters of the execution of smart contracts on the Ethereum Virtual Machine. Modeling the smart contract implementation as well as the blockchain behavior using a BIP (Behavior Interaction Priorities) framework based on finite state automata is presented in work [37]. Paper [38] presents the blockchain simulator SIMBA, which is based on open source and allows us to test already developed blockchains.

In paper [39], the authors explore the main PoW blockchains and how they affect the reliability of systems built on top of them. First of all, they include issues of the availability of functions that such systems need and properties that cause their negative consequences. The issues of reducing the number of unconfirmed transactions in blockchains such as Bitcoin and Ethereum are also being explored. We would like to note that there would be fewer problems if the logic for adding such transactions were correctly described. Therefore, the correct construction of the theory and its further axiomatics are also important in these directions.

The next two papers [40,41] are similar in their subject matter. In the first one, the model-based formalization is presented for the blockchain protocol by SDL formalism of Telegonic Tau. In the second, smart contracts are verified where finite automata simulate the smart contract execution. Both in the first and second case, mathematical methods are used for formalization and verification. Moreover, verification using finite automata can be transferred to the language of logic since there is an axiomatization [42,43] of the theory of finite automata.

The following paper [44] explores the application of a model-checking approach to Ethereum blockchain applications based on smart contracts. They used model-checking to check whether a smart contract conforms to its specification. Just like in that work, the blockchain axioms presented in our work are a kind of verification for those blockchain structures that are being created. Moreover, with the help of axioms, it is easy to check the correctness of the given basic operations and relations.

Works on formal modeling and verification [45–48] as well as stochastic modeling [49,50] use various methods based on mathematical approaches. More and more works are coming out on the topic of describing the blockchain and other ledgers using already known mathematical theories such as the category theory [51], the group theory [52], and others. However, any mathematical theory is based on axioms, so we can try to apply the axiomatic approach. Our approach naturally extends this line of research. It also helps to better understand the relationship between other mathematical theories and the blockchain theory.

7. Discussion

The axiomatization of blockchain theory opens up new perspectives in the study of the computational complexity of blockchain structures and the identification of new properties and methods. The independence of some axioms of this theory gives an understanding of the key properties of the blockchain. The constructed models of the blockchain theory $\mathbb{T}$ show that this theory is consistent and can be developed further. This research focuses on the study of models with PoW consensus algorithms, but these approaches can be used for models with other consensus algorithms such as PoS and PoA.

Approaches and methods for modeling and formalizing blockchains, smart contracts, and other blockchain-based systems described in related works can be considered from the point of view of the axiomatization of such structures in the first-order logic. Axioms help us understand what types of structures we work with, what properties they have, and what methods and approaches to use for modeling and verification. Moreover, the axioms define the skeleton of these structures. This makes it easy to translate these structures into other environments. This is important because most of the related work uses mostly graphical interpreters in which transitions are specified using the simplest logical rules, but this is clearly not enough.

We would especially like to highlight the possibility of applying logic and axioms to the creation of smart contracts that work on blockchain structures. We have already developed a p-complete logical object-oriented programming language L^* for creating smart contracts. We have developed a virtual machine to execute them. Blockchain axiomatics allows us to create special blockchain libraries for our language and apply ready-made solutions in such areas as artificial intelligence, finance, swarm intelligence, etc.

In this section, we will also discuss such structures as multi-blockchains. Multi-blockchains are a revolutionary development in the world of blockchain technology that has various applications in different industries such as finance, healthcare, and supply chain management. Multi-blockchains utilize multiple blockchain networks to create a more scalable, secure, and efficient solution. They can be used to streamline processes, reduce transaction costs, and improve data privacy. In order to provide a formal and precise description of the properties of multi-blockchains, axiomatization in the first-order logics is necessary. Axiomatization enables the formulation of precise, rigorous, and unambiguous statements about the characteristics and behavior of multi-blockchains, providing a clear understanding of their potential benefits and limitations. This approach can help facilitate the development and deployment of multi-blockchain systems, paving the way for a more innovative and reliable technological landscape.

8. Conclusions

In this work, an axiomatization of the blockchain theory has been proposed using the logic of first-order predicates. We have divided all these axioms into groups and shown that most of them are independent. This makes it possible to highlight the key properties of blockchain structures. We have also shown that both Bitcoin blockchain and Ethereum blockchain are models of our theory. We have also studied related works by other authors and the possibility of applying our results in them. Many of these works are based on mathematical concepts and theories, which have their own axiomatics as well.

Axiomatization made it possible to clearly define the concept of a blockchain structure, which allows modeling and verification of such structures to be used. It also allows enriching the theory with new axioms and creating new types of data storage. Moreover, signature operations and relations clearly define the structure of libraries for creating high-level libraries for their further use in various blockchain-based systems. This has opened up new possibilities for creating innovative solutions that can help address some of the most pressing challenges of our time. We hope that the axiomatization of blockchain in first-order logic marks a significant milestone in the evolution of blockchain technology, and its impact is likely to be felt for many years to come.

Overall, the future use of blockchain technology in AI, robotics, DeFi, IoT, chipping, medicine, logistics, space industry, and other areas is set to bring significant benefits. These benefits will include increased efficiency, transparency, security, and cost-effectiveness. The future is bright for the integration of blockchain, and it is expected to transform different sectors. In some cases, blockchain technology becomes critical. For example, as artificial intelligence systems continue to evolve and mature, there is a growing concern that they may eventually go beyond the limits set by their developers. This could lead to unintended consequences, ranging from minor glitches to catastrophic failures that could put lives at risk. To prevent this, some experts are calling for the use of blockchain technology as a controlling authority. By leveraging the transparency and immutability of the blockchain, developers and regulators can work together to establish clear rules and parameters for AI systems. These rules can be encoded into the blockchain, creating a tamper-proof record of the system's behavior. In this way, the blockchain can help ensure that AI systems operate within safe and ethical boundaries, giving us greater confidence in their continued development and deployment.

Author Contributions: Conceptualization, S.G. and A.N.; methodology, S.G. and A.N.; formal analysis, S.G.; validation, S.G.; investigation, S.G. and A.N.; writing—original draft preparation, A.N.; writing—review and editing, A.N.; supervision, S.G.; project administration, S.G.; software, A.N. All authors have read and agreed to the published version of the manuscript.

Funding: The work was performed within the state task of the Sobolev Institute of Mathematics (project No. FWNF-2022-0011). This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Singhal, B.; Dhameja, G.; Panda, P. *Beginning Blockchain*; Apress: New York, NY, USA, 2018; 386p, ISBN: 978-1-4842-3444-0.
2. Nacamoto, S. *Bitcoin: A Peer-to-Peer Electronic Cash System*. 2008. Available online: https://bitcoin.org/bitcoin.pdf (accessed on 25 March 2023).
3. Narayanan, A.; Bonneau, J.; Felten, E.; Miller, A.; Goldfeder, S.; Clark, J. *Bitcoin and Cryptocurrency Technologies*; Princeton University Press: Princeton, NJ, USA, 2016; 336p, ISBN: 978-0-6911-7169-2
4. Henrique, C. A Complete Decoding of the Bitcoin Block. 2021. Available online: https://levelup.gitconnected.com/a-complete-decoding-of-the-bitcoin-block-578904267142 (accessed on 25 March 2023).
5. Buterin, V. *Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform*; 2014. Available online: https://ethereum.org/669c9e2e2027310b6b3cdce6e1c52962/Ethereum_Whitepaper_-_Buterin_2014.pdf (accessed on 1 June 2023).
6. Ethereum.org. Proof-of-Stake (PoS). Available online: https://ethereum.org/en/developers/docs/consensus-mechanisms/pos/ (accessed on 4 April 2023).
7. Zetzsche, D.A.; Arner, D.W.; Buckley, R.P. Decentralized Finance (DeFi). *J. Financ. Regul.* **2020**, *6*, 172–203. [CrossRef]
8. Jurgelaitis, M.; Drungilas, V.; Ceponiene, L.; Butkiene, R.; Vaiciukynas, E. Modelling principles for blockchain-based implementation of business or scientific processes. In Proceedings of the International Conference on Information Technology, Shanghai, China, 20–23 December 2019.
9. Lykidis, I.; Drosatos, G.; Rantos, K. The Use of Blockchain Technology in e-Government Services. *Computers* **2021**, 10, 168. [CrossRef]
10. Zakrzewski, J. *Towards Verification of Ethereum Smart Contracts: A Formalization of Core of Solidity*; Part of the Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2018; Volume 11294. [CrossRef]
11. Kumar, S.; Tiwari, P.; Zymbler, M. Internet of Things is a revolutionary approach for future technology enhancement: A review. *J. Big Data* **2019**, *6*, 111. [CrossRef]
12. Goncharov, S.S.; Nechesov, A.V. Polynomial-Computable Representation of Neural Networks in Semantic Programming. *J* **2023**, *6*, 48–57. [CrossRef]
13. Xu, F.; Uszkoreit, H.; Du, Y.; Fan, W.; Zhao, D.; Zhu, J. *Explainable AI: A Brief Survey on History, Research Areas, Approaches and Challenges*; Natural Language Processing and Chinese Computing; NLPCC 2019; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2019; Volume 11839. [CrossRef]
14. Goncharov, S.S.; Nechesov, A.V. Solution of the problem P = L. *Mathematics* **2021**, *10*, 113. [CrossRef]
15. Goncharov, S.S.; Nechesov, A.V. Semantic programming for AI and Robotics. In Proceedings of the IEEE International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON), Yekaterinburg, Russia, 11–13 November 2022; pp. 810–815. [CrossRef]
16. Nechesov, A.V. Blockchain Structures and Smart Contracts in Semantic Programming. In Proceedings of the International Conference "Maltsev Readings 2022", Novosibirsk, Russia, 14–18 November 2022; p. 210.
17. Brünnler, K.; Flumini, D.; Studer, T. A logic of blockchain updates. *J. Log. Comput.* **2020**, *30*, 1469–1485. [CrossRef]
18. Leshno, J.D.; Strack, P. Bitcoin: An Axiomatic Approach and an Impossibility Theorem. *Am. Econ. Rev. Insights* **2020**, *2*, 269–286. [CrossRef]
19. Bichuch, M.; Feinstein, Z. Axioms for Automated Market Makers: A Mathematical Framework in FinTech and Decentralized Finance. *arXiv* **2022**, arXiv: 2210.01227. https://doi.org/10.48550/arXiv.2210.01227.

20. Reed, R.C. A decidable Ehrenfeucht theory with exactly two hyperarithmetic models. *Ann. Pure Appl. Log.* **1991**, *53*, 135–168. [CrossRef]
21. Sultan, K.; Ruhi, U.; Lakhani, R. Conceptualizing Blockchains: Characteristics & Applications In Proceedings of the 11th IADIS International Conference on Information Systems, Lisbon, Portugal, 14–16 April 2018.
22. Majeed, A.; Ahmad, A.; Doss, R. Malicious Node Traceback in Opportunistic Networks Using Merkle Trees. In Proceedings of the IEEE International Conference on Data Science and Data Intensive Systems, Sydney, Australia, 11–13 December 2015; pp. 147–152. [CrossRef]
23. Lin, I.; Sung, C. An Efficient Source Authentication for Multicast Based on Merkle Hash Tree. In Proceedings of the Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Darmstadt, Germany, 15–17 October 2010; pp. 5–8. [CrossRef]
24. Blockchain. Wikipedia. Available online: https://en.wikipedia.org/wiki/Blockchain (accessed on 15 May 2023).
25. Frankenfield, J. What Is an Ommer (Uncle) Block in Cryptocurrency? *Investopedia*, 31 May 2023. Available online: https://www.investopedia.com/terms/u/uncle-block-cryptocurrency.asp (accessed on 8 May 2023).
26. Seberino, C. Ethereum Classic Technical Reference. 2020. Available online: https://buildmedia.readthedocs.org/media/pdf/etc-tech-ref/latest/etc-tech-ref.pdf (accessed on 8 May 2023).
27. Orphan Blocks, Stale Blocks and the GHOST Protocol. 2022. Available online: https://www.minima.global/post/orphan-blocks-stale-blocks-and-the-ghost-protocol (accessed on 25 May 2023).
28. Bentov, I.; Lee, C.; Mizrahi, A.; Rosenfeld, M. Proof of Activity: Extending Bitcoin's Proof of Work via Proof of Stake. *ACM SIGMETRICS Perform. Eval. Rev.* **2014**, *42*, 34–37. [CrossRef]
29. Sompolinsky, Y.; Zohar, A. Secure High-Rate Transaction Processing in Bitcoin. In *Financial Cryptography and Data Security*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 507–527. [CrossRef]
30. Goncharov, S.S.; Nechesov, A.V. Polynomial analogue of Gandy's fixed point theorem. *Mathematics* **2021**, *9*, 2102. [CrossRef]
31. Nitsche, N. What Is Bitcoin Mining and How Does It Actually Work? 2020. Available online: https://paymentandbanking.com/what-is-bitcoin-mining-and-how-does-it-actually-work/ (accessed on 10 May 2023).
32. Bitcoin.it. Available online: https://en.bitcoin.it/wiki/Genesis_block (accessed on 28 April 2023).
33. Oettler, M. In-Depth Explanation of Bitcoin Mining Difficulty. Blockchain Academy Mittweida. 2021. Available online: https://blockchain-academy.hs-mittweida.de/courses/blockchain-introduction-technical-beginner-to-intermediate/lessons/lesson-14-bitcoin-mining-difficulty/topic/in-depth-explanation-of-bitcoin-mining-difficulty/ (accessed on 10 March 2023).
34. Lo, S.K.; Staples, M.; Xu, X. Modelling schemes for multi-party blockchain-based systems to support integrity analysis. *Blockchain Res. Appl.* **2021**, *2*, 100024. [CrossRef]
35. Dennis, R.; Owenson, G.; Aziz, B. A Temporal Blockchain: A Formal Analysis. In Proceedings of the International Conference on Collaboration Technologies and Systems (CTS), Orlando, FL, USA, 31 October–4 November 2016; pp. 430–437. [CrossRef]
36. Amani, S.; Begel, M.; Bortin, M.; Staples, M. Towards Verifying Ethereum Smart Contract Bytecode in Isabelle/HOL. In Proceedings of the 7th ACM SIGPLAN International Conference on Certified Programs and ProofsJanuary, Los Angeles, CA, USA, 8–9 January 2018; pp. 66–77. [CrossRef]
37. Abdellatif, T.; Brousmiche, K.L. Formal verification of smart contracts based on users and blockchain behaviors models. In Proceedings of the International Conference on New Technologies, Mobility and Security (NTMS), Paris, France, 26–28 February 2018; IEEE: Piscataway, NJ, USA, 2018.
38. Fattahi, S.; Makanju, A.; Fard, A. SIMBA: An Efficient Simulator for Blockchain Applications. In Proceedings of the 50th Annual IEEE-IFIP International Conference on Dependable Systems and Networks-Supplemental Volume (DSN-S), Valencia, Spain, 29 June–2 July 2020; pp. 51–52. [CrossRef]
39. Weber, I.; Gramoli, V.; Ponomarev, A.; Staples, M.; Holz, R.; Tran, A.; Rimba, P. On Availability for Blockchain-Based Systems. In Proceedings of the IEEE 36th Symposium on Reliable Distributed Systems (SRDS), Hong Kong, China, 26–29 September 2017. [CrossRef]
40. Bai, X.; Cheng, Z.; Duan, Z.; Hu, K. Formal Modeling and Verification of Smart Contracts. In Proceedings of the 7th International Conference on Software and Computer Applications, Kuantan, Malaysia, 8–10 February 2018; pp. 322–326. [CrossRef]
41. Duan, Z.; Mao, H.; Chen, Z.; Bai, X.; Hu, K.; Talpin, J.-P. Formal Modeling and Verification of Blockchain System. In Proceedings of the 10th International Conference on Computer Modeling and Simulation, Sydney, Australia, 8–10 January 2018; pp. 231–235. [CrossRef]
42. Esik, Z.; Kuich, W. *Equational Axioms for a Theory of Automata*; Formal Languages and Applications. Studies in Fuzziness and Soft Computing; Springer: Berlin/Heidelberg, Germany, 2004; Volume 148. [CrossRef]
43. Piedeleu, R.; Zanasi, F. *A String Diagrammatic Axiomatisation of Finite-State Automata*; Foundations of Software Science and Computation Structures; FOSSACS 2021; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2021; Volume 12650. [CrossRef]
44. Nehai, Z.; Piriou, P.Y.; Daumas, F. Model-checking of smart contracts. In Proceedings of the IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Halifax, NS, Canada, 30 July–3 August 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 980–987.
45. Yang, Z.; Dai, M.; Guo, J. Formal Modeling and Verification of Smart Contracts with Spin. *Electronics* **2022**, 11, 3091. [CrossRef]

46. Tolmach, P.; Li, Y.; Lin, S.; Liu, Y.; Li, Z. A Survey of Smart Contract Formal Specification and Verification. *ACM Comput. Surv. (CSUR)* **2020**, *54*, 1–38. [CrossRef]
47. Afzaal, H.; Imran,M.; Janjua, M.; Gochhayat, S. Formal Modeling and Verification of a Blockchain-Based Crowdsourcing Consensus Protocol. *IEEE Access* **2022**, *10*, 8163–8183. [CrossRef]
48. Leo, B.; Ghodous, P.; Gelas, J.-P.; Silva, C. Modelling of Decentralised Blockchain Applications Development. In Proceedings of the 2020 International Conference on High Performance Computing & Simulation (HPCS 2020), Barcelone, Spain, 10–14 December 2020. Available online: https://hal.science/hal-03340842 (accessed on 14 January 2023).
49. Zhang, Z.; Zargham, M.; Preciado, V.M. On modeling blockchain-enabled economic networks as stochastic dynamical systems. *Appl. Netw. Sci.* **2020**, *5*, 19. [CrossRef]
50. Bistarelli, S.; Nicola, R.; Galletta, L.; Laneve, C.; Mercanti, I.; Veschetti, A. Stochastic Modelling and Analysis of the Bitcoin Protocol in the Presence of Block Communication Delays. *Concurr. Comput. Pract. Exp.* **2023**, *35*, e6749. [CrossRef]
51. Nester, C. A Foundation for Ledger Structures. In Proceedings of the 2nd Tokenomics Conference: On Blockchain Economics, Security and Protocols, Online, 26–27 October 2020. Available online: https://arxiv.org/pdf/2010.08337.pdf (accessed on 7 February 2023).
52. Zhao, D. Algebraic Structure of Blockchains: A Group-Theoretical Primer. *arXiv* **2020**, arXiv:2002.05973.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.