# Survey on blockchain technology, consensus algorithms, and alternative distributed technologies

Aleksey Fefelov*, Nikita Mishin‖, Vyacheslav Bushev‡, Iakov Kirilenko§ and Daniil Berezun¶
Mathematics and Mechanics Faculty, Saint Petersburg State University, Russia
*st054453@student.spbu.ru, ‖st054748@student.spbu.ru, ‡st054677@student.spbu.ru,
§y.kirilenko@spbu.ru, ¶danya.berezun@gmail.com

***Abstract*—Since its origination, the blockchain technology has attracted tremendous attention from mass media, developers, scientific community and industry. This influenced the fact that it began to actively and quickly grow and develop — new platforms, consensus mechanism, alternative technologies started to appear. There are many information sources about them but they all suffer in one way or another — they are either excessive technical articles or a shallow descriptions. Therefore, as consequence of such immense growth of technology and projects around it, the systematization of "blockchain universe" becomes urgent.**

**The paper provides such taxonomy primarily based on historical motivation and goals that induce to create new algorithms, systems and technologies.**

## I. Introduction

For the first time, the term “blockchain” appeared as a name of a distributed database, implemented in Bitcoin system by S. Nakamoto [1]. The original blockchain in the form as it was initially presented may be considered as an extended distributed digital version of a classical banking promissory note. More precise, blockchain is a linked list of transactions that stores an entire system transactions history.

The key feature of the technology is that it provides history *immutability*, i.e. an immutability of blocks of transactions, and, in most cases, an *ability to verify* the authenticity of a current state of the system by any network participant (aka node).

There are two essentially different kinds of blockchains: those where any node can write to, read from and anyone can join the network are known as *public* or *permissionless*, an opponent is called *private* or *permission*.

Since the original Bitcoin blockchain has been proposed a number of different consensus algorithms were developed, which can be divided into two main groups: *proof-based* (see V) and *voting-based* (see VI). The former is used mainly in *private* blockchains, and the latter is preferred in *public* ones.

When proof-based consensus is used it is possible to encourage network participants, for example, a corresponding commission reward is used for a block creation [1, 2]. However, consensus algorithms of this kind are too energy-consuming to be used in private networks. But for the voting-based algorithms, as a common practice, there is no obvious solution of encouraging network participants, because everyone makes an equal contribution, and one must either pay to everyone involved in consensus problem solving, which would not be very profitable from an economic point of view. At the same time, these algorithms consume significantly less power, known to work well with a smaller number of nodes and are more resistant to attacks[3].

The blockchain was designed as an open, transparent, and independent network, where all participants have equal rights [1]. And the first application for it was limited to a finance sector and few simple use cases. Limited application area of the first blockchain realization, as well as constraints in transaction execution logic and scalability issues, were driving factors, which led to an implementation of new protocols, systems, and architectures. Also, due to an interest to the new technology large companies have developed more centralized forms of the blockchain [4], while maintaining the distribution of data. This new excessive development, in turn, brought to the stage the problem of interaction among different blockchains (interoperability problem) that also is solved by new protocols and systems.

Thus, a systematic study of the blockchain is necessary.

Recently, several surveys on blockchain have been proposed. However they provide a rather narrow perspective of protocols, consensus algorithms, and existing alternatives. For instance, [5, 6] considers consensus mechanisms, ignoring platforms as well as [7], where a technical overview of blockchain technologies and related topics is provided. [8] is focused on security issues and [9] surveys algorithms that are related to the enterprise. Also, some works are constricted to concrete protocols and their analysis, e.g. [10, 11].

In contrast, this paper considers blockchain alternatives while providing a systematic study dedicated to some promising and already well-known platforms. Also while [12] follows a similar approach, it is focused on permissioned blockchains in context of consensus protocols whereas we also describe public ones specifying industry-focused characteristics. We believe that our paper is the first one that provides such a comparable description of platforms. In addition, the paper provides an intuitive description of consensus algorithms.

The paper is organized as follows. Section II describes the most popular blockchain networks and platforms. Section III provides its alternatives. Section IV gives a classification of consensus algorithms. Section V and Section VI present some proof-based and voting-based consensus algorithms, respectively. In Section VII we discuss possible research gaps

and future work. Section VIII concludes the paper.

## II. Blockchain systems and platforms

In this section, we provide an overview of main blockchain systems and platforms. More systems and platforms may be found in a property table on Fig. 1. The table consists of several categories that characterize projects. Some of them are:

- Scalability — a number of transactions that can be processed per second;
- Smart contracts — a presence and a possibility of smart contract creation;
- Governance — who makes key decisions about policy, standards, updating and maintaining a project.

Blockchain was designed to solve several problems:

- A third party (a middle man) when making financial or other operations involving two parties;
- The trust in potentially unsafe decentralized network;
- High transaction fees cost.

The First blockchain system which solved problems above was Bitcoin.

### A. Bitcoin

As mentioned before, described in 2008 [1] by a person or a group of people called S. Nakamoto, Bitcoin has gained popularity over the years. Bitcoin, an electronic coin, is a decentralized digital payment system. It is the first open public blockchain. Nodes[1] in Bitcoin system communicating through peer-to-peer network[2]. Each user has a wallet [13] that contains a set of private and public keys generated by digital signature algorithm (currently ECDSA [14]). These keys fully define users. As a rough analogy username and its password can be considered as a pair of public and private keys correspondingly. Public keys define Bitcoin addresses to which users could send coins. Each user could have many bitcoin addresses: he secretly stores these pairs of keys in one or more of his wallets. Best practices of Bitcoin recommend using different bitcoin addresses for each new transaction in order to preserve pseudonymity[3].

Electronic payments are done by creating transactions that move coins from one address to another. The coin is represented as a chain of digital signatures. Sender digitally signs previous transaction hash with his private key and next owner public key and then adds it to the end of the chain. Using senders public key one may validate the transaction by running through the coin history, a so called *chain of ownership*. This confirms that the sender owns the coins and has not already spent them.

Each transaction in Bitcoin may have multiple inputs and at most two outputs. First output presents the payment and the second one — the change that will be returned to one of the senders addresses. These outputs are called UTXOs (Unspent Transaction Output) [15]. They could be either spent or unspent — there is no way to UTXO be in different states at the same time. UTXO is a blockchain analogy of soft money: buying goods of total cost $n$฿ one pays with several UTXOs (banknotes) of total value $m$฿ : $m \geq n$. The change $(m - n)$฿ is being returned to the customer as a new UTXO (new banknote). UTXO is controlled by a scripting system called Bitcoin Script [16].

Bitcoin Script is stack-based, non Turing-complete language without loops. Turing-incompleteness is very important: it guarantees scripts termination in a limited amount of time. Thus, all transactions and block creations are guaranteed to terminate. All Bitcoin Script programs (scripts) consists of two parts: one, called *scriptPubKey* (aka *locking script*), is embedded into transaction inputs, another, called *scriptSig* (aka *unlocking script*) — into transaction outputs. In order to spend some previous transaction outputs one has to combine new transaction *scriptSig* with the previous transaction *scriptPubKey* into one script. The resulting script is correct iff its execution yields **True**, i.e. it satisfies condition placed by locking script. In the simplest case, *scriptSig* represents the senders signature with the private key, while *scriptPubKey* checks public key correctness and signature validity for the address. In other words, coins can set off conditions on coins spending via locking script. Those conditions must be met in order to spend the UTXO, and the coins receiver must satisfy these conditions by unlocking script. Moreover, Bitcoin Script actually allows some simple smart contracts creation like *multisignature*. In other words, there is a possibility of decentralized application, Dapps, creation on top of Bitcoin. More details could be found in [16].

Ones transaction is created it broadcasts into the network. Then *resourceful nodes* (aka miners) validate and batch it into new blocks. Ones block is created miner adds it into the chain and broadcasts it to the network. Note that some ambiguity may appear on this step. Thus, nodes have to come to a consensus. In Bitcoin proof-of-work (see V-A) is used. When the majority of nodes accepts the created block, the miner created the block got a reward as new coins — currently 12.5฿ [17].

Difficulty of mining [18], i.e. block creation difficulty, is adjusted every 2016 in order to keep the time of new block generation around 10 minutes. It gives constraints on how many transactions per second processed in network. For bitcoin it's around $3 - 5$ tx/s [19].

It should be noted that after the appearance of Bitcoin, a lot of different hard[4] and source code forks appeared. They are caused by some changes in a protocol (larger block size, less generation time for the block) and consensus mechanism (switching to another algorithm, replacing hash function) for different kind of purposes: increasing scalability, fixing bugs,

[1] In peer-to-peer[2] networks *node* stands for any device connected to the network.

[2] *Peer-to-peer* network is a network where all peers (participants) are equal — node is a client and also fulfill server function

[3] Pseudonymity is in a state of anonymity. It allows the user to remain it's identity in secret though digital coins can be tracked through token history and associate with external events compromised user identity (not true anonymity)

[4] A permanent divergence in the block chain, commonly occurs when non-upgraded nodes can't validate blocks created by upgraded nodes that follow newer consensus rules

| Name | Type | Consensus mechanism | Scalability | Smart contracts | Dapps | Programming language | Industry-focus | Governance |
|---|---|---|---|---|---|---|---|---|
| Bitcoin | Public | PoW | 3-5tx/s | Limited | Limited | Script | Financial services | Bitcoin Foundation |
| Litecoin (Bitcoin fork) | Public | PoW | 56tx/s[8] | Limited | Limited | Script | Financial services | Litecoin Foundation |
| Bitcoin Cash (Bitcoin fork) | Public | PoW | 61tx/s[8] | Limited | Limited | Script | Financial services | Bitcoin Cash Foundation |
| Dash (Bitcoin fork) | Public | PoW | 28tx/s[8] | Limited | Limited | Script | Financial services | Decentralized Governance (Public vote) |
| Peercoin | Public | Pow (for coin distribution) then PoS | 3-5tx/s[7] | Limited | Limited | Script | Financial services | Peercoin Foundation |
| Ethereum | Public | PoW (planning to switch to PoS) | 15tx/s | Full support | Full support, plus DAO | Solidity (and other) | Cross-industry | Ethereum developers |
| R3 Corda | Private | pluggable consensus (notary pool cluster may choose any consensus mechanism)[10] | Between 15 and 1678 tx/s depending on measurement[9] | Full support (use pure functions + legally binding) | Full support | Any language that compiled to JVM | Originally Finance industry but may be applied to other industries | R3 consortium |
| Cardano | Public | Ouroboros Proof of Stake | 50-250tx/s[13] | Full support | Full support | Plutus | Financial services | Cardano Foundation IOHK, Emurgo |
| Ripple | Private | RCPA (BFT-based) | 50,000tx/s[1] or 1500tx/s[8] | –[12] | – | – | Financial services | Decentralized Governance (Public vote) |
| Stellar | Private | SCP (BFT-based) | 4,000tx/s[5] or 1000tx/s[8] | –[12] | – | – | Financial services | Decentralized Governance (Public vote) |
| Iroha | Private | YAC (BFT-based) | Is not officially measured | Limited (Built-in commands) | Full support | Built-in commands | Cross-industry | Network Maintainers |
| Fabric | Private | BFT-based (can be chosen) | 3500tx/s[2] | Full support | Full support | General purpose languages (i.e. Go, Java) | Cross-industry | Network Maintainers |
| IOTA | Private | PoW | 1500tx/s[8] | Full support | Full support | Abra | Cross-industry | IOTA Foundation |
| EOS | Public | DPoS | 60+ tx/s[6] | Full support | Full support | EOS | Cross-industry | On-Chain[15] |
| Cosmos | Depends on zone[10] | Tendermint | Possible infinitely scaling by creating new zones[4] | Depends on zone[10] | Depends on zone[10] | Depends on zone[10] | Cross-industry | Holders of native staking token (Atom) may propose submission and vote on proposal by stake token[3] |
| Tezos | Public | Liquid Proof-of-Stake | 40tx/s[11] | Full support + formal verification[14] | Full support | Michelson | Financial services | On-Chain[15] |
| Slimcoin | Public | PoB | Is not officially measured | - | - | - | Financial services | Slimcoin developers |

[1] According to Ripple website, https://ripple.com/xrp/
[2] According to an article, https://www.ibm.com/blogs/research/2018/02/architecture-hyperledger-fabric/
[3] According to Cosmos website, https://cosmos.network/docs/spec/governance/
[4] For example, Ethermint (copy of Ethereum with PoS consensus) provide 200tx/s and further may be improved by a horizontal scaling, https://blog.cosmos.network/latest-in-cosmos-october-update-f8494790fad9
[5] Depending on a hardware, as Stellar founder Jed McCaleb said in an interview with the blockchain-focused podcast Epicenter, https://soundcloud.com/epicenterbitcoin/eb-128
[6] 02/10/2019 60tx/s has been reached, but according to news https://coincodex.com/article/2052/eos-achieves-3000-transactions-per-second peak is 3097tx/s
[7] Similar to Bitcoin. The only difference between BTC and PPC transactions is an added 4 bytes time field as it mentioned on an official Peercoin forum, https://talk.peercoin.net/t/how-many-transactions-per-second-can-ppc-handle/3535
[8] According to Daniel O'Keeffe, creative writer and crypto capitalist, https://medium.com/coinmonks/understanding-cryptocurrency-transaction-speeds-f9731fd93cb3
[9] According to Mike Ward, head of product at R3, https://medium.com/corda/transactions-per-second-tps-de3fb55d60e3
[10] See appropriate article section
[11] According to https://medium.com/tezos/liquid-proof-of-stake-aec2f7ef1da7
[12] Only limited API methods are provided, see appropriate article section
[13] Charles Hoskinson touched that topic in his AMA: 1:01:45 https://youtu.be/pQakcFK72Oc?t=3705
[14] According to tezos developers
[15] According to terminology of https://blockonomi.com/blockchain-governance/

Fig. 1. Systems and platforms Overview

improving security. Some of these forks are presented in the table.

After BITCOIN appearance an idea of using blockchain out of finance area had appeared [20]. From then it was found a number of blockchain applications: banking [21], insurance [22], music [23], real estate[24], internet of things [25], medicine [26–28]. New use cases required new systems which can handle more sophisticated logic.

Also, it can lead to the idea of decentralized applications and even more — decentralized autonomous organizations. Decentralized applications is an application that runs on the decentralized network and not controlled by a single authority and cannot be shut downed. Decentralized autonomous organizations, DAO, are organizations that fully autonomous, decen-

tralized and have no single leader. They run and controlled by programming code. DAO may be a replacement for traditional organization structure. It's owned by everyone who had tokens — tokens represent voting rights. The first time the above ideas were implemented in ETHEREUM (see II-B).

### B. ETHEREUM

ETHEREUM [2] is a platform for creating decentralized applications over blockchain. The key feature of ETHEREUM is smart contracts. A *smart contract* is a programmable object which works on top of the blockchain. The smart contract contains a set of rules and conditions on which participants are agreed upon. If all rules and conditions are met then smart contracts automatically executes after appropriate request or transaction. It is impossible to influence on its execution or change these conditions. Smart contracts are stored in the blockchain and they could keep some additional data for the logic of execution. Usually, smart contracts in ETHEREUM are implemented in SOLIDITY or another high-level programming language which compiles into opcodes (operation codes) of Ethereum Virtual Machine. *Ethereum Virtual Machine*, EVM, is a virtual machine that is isolated from the external world within which all transactions are executed.

ETHEREUM relates to public blockchain since anyone can connect to the network, create a transaction or a smart contract, mine blocks or validate them without any kind of permission or registration. It can be seen as a decentralized transactional state machine where the global state is moved by each transaction from one state to another, which must be agreed upon by the majority of nodes. ETHEREUM global state consists of accounts and states associated with them (it can be seen as a mapping from accounts addresses to their states). There are two account types: user accounts and contracts. Like BITCOIN accounts, user accounts in ETHEREUM are defined by a pair of private and public keys (also uses ECDSA). Unlike BITCOIN where users balances are defined by UTXO, in ETHEREUM accounts are fully defined by its global state. Contracts also have balances but they are controlled by associated with them contracts codes. Moreover, contracts could store an additional data for their execution. As a rough analogy, contracts can be seen as a singleton in object-oriented paradigm with their own states and methods, which can be created by a user or other contract transaction. Note that both account types contain some other important components which we omit in the paper since they are irrelevant for the current discussion. More details can be found in ETHEREUM's yellow paper [2].

As mentioned before, user accounts interact with contracts or other user accounts via transactions. There are two different transaction types: message calls and contract creations. Message call is an act of passing a message, an arbitrary set of data and value (Ether), from one account to another. If the last one is associated with a user account then a specified amount of money is just transferred to this user account like in BITCOIN. If it is a contract account then it receives message and runs its code. Contract creation is an act of creating and deploying a smart contract. Each transaction that occurs between two accounts trigger changes in ETHEREUM global state, thereby leading it to a new state. Both types of transactions contain some other fields except mentioned above, e.g. value, data and account address [2]. In order to provide a detailed description of how *Ethereum* works we will consider two more transaction fields below: `gasPrice` and `gasLimit`.

*Ethereum Virtual Machine*, EVM, is a simple virtual stack-based Turing-complete bytecode machine. EVM has its own language called *EVM bytecode* which operates with opcodes. As mentioned, usually smart contracts are written in a high-level language such as SOLIDITY, and then SOLIDITY code is being compiled to EVM bytecode. EVM has several limitations programmer has to take in mind all the time. For example, the size of EVM stack is limited to 1024 and a program execution bounded by *gas*. These limitations are made to workaround halting problem for Turing-complete machine. More precisely, all EVM operations, including storing or reading from memory, have prices expressed in the amount of gas [2]. This limitation not only prevents from infinite execution which lead to disruption of node activity and the whole system (all nodes execute all transaction), but also prevents the network from clogging that may lead to fetching down the ETHEREUM network. It is important to note that release of memory occupied by the contract or it's data is encouraged by protocol — some money are being refunded to the creator. This is done so that the size of the blockchain does not increase too much. Thus, `gasLimit` transaction field contains the limit of gas that may be used during transaction execution. If both executions are not finished and the transaction is out-of-gas then a *rollback* occurs — all changes that were made through the transaction execution are being reverted. The same is true for all other paths which don't not lead to acceptable and correct state. While `GasLimit` limits computation time, `GasPrice` serves as a market characteristic. The higher the price indicated in `GasPrice`, the faster the transaction will be carried out since miners get fees for each transaction which is equal to the product of `gasPrice` and the total gas being spent by the transaction.

Similarly to BITCOIN, transactions in ETHEREUM are batched into the blocks. First, miners validate transaction, i.e. checks whenever account has enough money, signature to be valid and so on. Then transaction is being executing on EVM. During the execution ETHEREUM records information that is needed after transaction termination [2]. If the execution succeed, i.e. it leads to a valid state, this state becomes the new global state. Then all steps above are being repeated for other transaction while there is enough gas, note that blocks also have `gasLimitS` that are being decreased after each successful transaction. It also is dictated by the protocol since an absence of gas limit for blocks will potentially lead to blocks with a huge size because the more transactions miner will add to a block, the more fee he will receive. Besides, it will also lead to reducing the number of processed transactions per second. Current network "speed" is around 15tx/s [29].

Ethereum uses a proof-of-work algorithm called ETHASH [30] which was specifically created for it. This

algorithm has a dynamic difficulty. It is adjusted whenever a period of two block times changes: the difficulty is being increased when time makes smaller and vice versa. Not that ETHEREUM creators set "ice age" or "difficulty bomb", difficulty function will slowly exponentially increase, in order to switch to proof-of-stake consensus algorithm called Casper [31] in future. In this situation, proof-of-work serve for initial distribution of coins in network.

Since now ETHEREUM uses proof-of-work, miners get a reward for mining block, but as the time for generation block is short compared to Bitcoin, part of reward also gets to *ommers*, aka *uncle*, block miners. Ommers got smaller reward than a full block. It is made to encourage nodes to stick to main chain because each such ommer contribute to the security by augmenting the amount of work waste on creating stale blocks. This total amount of work is applied when there need to choose canonical blockchain path (path with the most computation done upon it).

Public blockchains are not appropriate for many enterprises. They need some specific criteria to be met when using this technology:

- High transaction speed and scalability;
- Constraints on access to the network;
- Constraints on performing different actions in the network:
  - observing records in a ledger,
  - making a specific type of transaction;
- Constraints on involvement in consensus mechanism.

Public blockchains do not meet these criteria. While ETHEREUM provides the possibility to create permission network, scalability and speed are still an issue. HYPERLEDGER consortium [32] was formed by LINUX FOUNDATION in order to solve these problems for organizations, i.e. to create enterprise blockchains. It has several projects such as IROHA, FABRIC and others.

### C. HYPERLEDGER IROHA

IROHA is a private network where access to all the data can be controlled. Permissions can be set for commands, queries and even joining the network. IROHA is built around domains and accounts. An *account* is an entity that is able to perform a specified set of actions [33], while a *domain* is just a set of accounts. Permission collections (*roles*) are given to accounts. Moreover, there are special grantable permissions, which allow controlling other accounts.

IROHA uses BFT consensus algorithm called YAC (Yet Another Consensus) [34], which is considered as an advantage of this network, because of its high work speed. Nevertheless, the more nodes joining the network, the more time it takes to reach the consensus.

IROHA has built-in commands for easy creation of new digital assets, registering an account, transfering assets and so on. IROHA clients interact with any peer similarly to a single server, i.e. one doesn't need to ask different peers to make sure that transaction has been written to the ledger. This network also supports multisignature transactions, which can not succeed until every responsible account confirms the transaction should proceed.

IROHA has no ability to write smart contracts[5]. Right now it is still in the designing stage [36].

### D. HYPERLEDGER FABRIC

FABRIC is one of the famous growing hyperledger projects. It was designed to satisfy enterprise requirement, that's why it is permissioned and supports channels network [37]. There are a few well-know methods to avoid private data leaks on a blockchain. The first one is encrypting data. And the second one is zero-knowledge proof (ZKP). A zero-knowledge proof system is an interactive protocol between a prover and a verifier, which aims at demonstrating that some statement is true [38]. The prover should demonstrate that he knows a proof, but without showing the proof itself. FABRIC offers *payment channels* (a way of making multiple transactions without committing all of them to the blockchain [39]). Instead of zero-knowledge proof (ZKP) or encrypting data, setting up a channel between nodes is a more flexible way to avoid private data leaks because encrypting data being stored on every node can be eventually decrypted and ZKP can require time and computational resources.

One of the exclusivity of FABRIC is its modular architecture. In fact, instead of using the default BFT, a network creator can use any consensus protocol even one that requires no cryptocurrencies. But not only consensus can be embedded. FABRIC has many modular components, for example:

- A pluggable ordering service establishes consensus on the order of transactions and then broadcasts blocks to peers;
- A pluggable membership service provider responsible for associating entities in the network with cryptographic identities; There are different actors in a blockchain network include peers, client applications, administrators and more; Each of them has a digital identity;
- An optional peer-to-peer gossip service disseminates the blocks output by ordering service to other peers;
- A pluggable endorsement and validation policy enforcement that can be independently configured per application;
- The ledger can be configured to support a variety of database management system.

FABRIC supports smart contracts, which are called *chaincodes*. They run in isolation within a container environment and can be written in general-purpose programming languages like JAVA, GO, and others, but they do not have a direct access to the ledger state. It is well known that smart contracts must be deterministic to have the same behavior on every node, because of this many platforms require to use specific language like SOLIDITY. Typically smart contracts are executed by all the nodes of the network, this is not an optimal approach. A new approach in FABRIC is "execute-order-validate". Smart

[5]Hyperledger Iroha allows users to perform common functions, such as creating and transferring digital assets, by using prebuilt commands [35].

contracts are executed by limited amount of peers, then transactions are ordered and validated before committing to the ledger. The first phase eliminates non-determinism because smart contracts are not executed on other nodes with a different network state, that's why we can use general-purpose programming languages for writing smart contracts.

The systems described above solve scalability problem[6] by sacrificing security or decentralization. For example, techniques such as *Side Chains* [40], *Off-chain payments* [41], *Sharding* [42] may be applied. However, with the emergence of a large number of blockchains the *interoperability* [43] problem appears.

*Interoperability* (in blockchains) is an ability of blockchains to communicate and freely exchange and share information with each other. Now there exist several projects that solve this problem such as WANCHAIN [44], EOS [45], COSMOS [46], Polkadot [47].

### E. COSMOS

The aim of COSMOS project is to create "Internet of Blockchains" [48]. This will solve interoperability and scaling problems. COSMOS offers new architecture which consists of several independent parallel blockchains, called *zones*. Each zone is powered by *Tendermint core* [49] and is attached to the main blockchain called the *hub*. Tendermint is related to Practical Byzantine Fault Tolerance consensus algorithm (its variation). For now, it requires mining as in PoW and guarantees safety when $2/3$ of nodes are not compromised. It is much faster in term of transactions per seconds then PoW.

Blockchains communicate with each other through inter-blockchain communication (IBC) protocol. All IBC goes through the hub which keeps track of all tokens exist in each zone. Horizontal scaling through sharding mechanism may be applied to solve scalability problem with such architecture — we just have several parts of blockchain (zones) divided to shards that communicate through IBC protocol.

COSMOS architecture allows extending an existing network by implementing new blockchains with a custom users consensus mechanism. It proposes a fine tuning of each zone and the whole network in case of governance. Each zone can have its own tunable governance mechanism. Governance of Cosmos ecosystem is implementing through a voting mechanism. Network ecosystem itself is governed by voting. Holders of native stacking tokens (called Atoms) may propose submission and vote on proposals by staking tokens [50].

It's worth to mention that the main launch of the network already happened [51].

## III. BLOCKCHAIN ALTERNATIVES

The blockchain shows that distributed ledger technologies (DLT[7]) may be applied to many areas. It was the first fully functional DLT technology. Due to interest, there are appeared several DLTs that are fulfilled requirements. Now there exist several alternative DLTs for blockchain. In this section, we provide a description of some of them.

### A. R3 CORDA

CORDA [52] is a decentralized global database. It is designed especially for creating so-called CORDAPPS — applications that bring new facilities to the database. The problem CORDA is intended to solve is a situation when network participants, such that individual people, institutions or companies, trust each other enough to trade and complete agreements but not enough to rely on a counterparty to operate all the records. Note, that CORDA possesses a lot of blockchain features while it does not appear as a blockchain system.

In Corda, *states* represent shared facts at a specific moment in time. They are immutable objects, stored in the ledger and known by one or more nodes. States have several fields such as the reference to a smart contract that govern them over time and participants who shared this fact and, arbitrary data for representing this fact. These states are passed around the network between nodes and each node maintains a database called Vault where it tracks current and historic states. Corda uses UTXO model which means that each transaction may have zero or more inputs and zero or more outputs. States are passed to transactions as inputs. Through transaction execution, new states are produced as outputs and inputs are marked as historic. In other words, a chain of states is created. The transaction could be seen as a proposal for changes in a ledger. This proposal would be committed if and only if the transaction does not contain double-spends, signed by all involved parties of it and valid in terms of a smart contract.

Smart contracts in CORDA are pure deterministic functions that can be written in any language compiled to JVM. Note, that contracts are executed in the deterministic version[8] of JVM. CORDA's smart contracts can be explicitly linked to legal contracts in the ledger, and thus such contracts will have legal power. In CORDA, nodes can only see those transactions in which they are directly involved or the ones that are able to influence them. Thus since nodes can only see a limited set of ledger records, transactions are kept in a private manner. Moreover, the system scalability is improved since there is no need to store or process transactions by all the nodes.

Since there are no miners in CORDA, so called *notaries nodes* fulfill their function. They provide transaction uniqueness checking, that also prevents the double-spending problem, whereas incoming transactions validation, i.e. correctness and authorization checking, is still processed by ordinary nodes.

In CORDA, in order to join the network, nodes must obtain an identity signed by a root authority. It is important to

[6] scalability problem in terms of blockchain referred to limits of transactions per second that network can process

[7] it is a digital system for storing and recording transactions in multiple places at the same time and has neither the central administration nor data storage

[8] The special deterministic version of JVM is needed since contracts execution has to be purely deterministic: contracts has to run, fail, terminate with an exception (the same one on all machines), go to an infinite loop, and even go out of gas, etc. simultaneously on all machines. In other words, contracts execution has to be so fully specified that it has to be impossible to see any difference both outside and inside the system in all circumstances.

note that since there is no blockchain in CORDA, there may be several independent networks. As the independent, each such network is free to choose own consensus algorithm and governance rules. They can be further be merged by establishing a connection between their nodes, setting *notaries nodes*, consensus algorithm.

CORDAPPS are distributed applications that run on Corda platform (same functionality as DAPPS). They one of the key feature of CORDA and, as an opposite to HYPERLEDGER FABRIC, CORDAPPS are able to communicate, exchange assets with each other through the platform.

Nevertheless, there was still a need for a platform that would be designed specifically for fast international transactions with low commission currency exchange for the banking sector. In other words, a fast exchange and payment system were needed. A solution was proposed by RIPPLE project.

### *B.* RIPPLE

RIPPLE network is one of the most popular today [53]. It has many modern features like supporting permissions, payment channels, multisignature transactions, decentralized exchange and the ability to lock up *XRP* (native Ripple asset) until a declared time passes. RIPPLE also has a smooth upgrade — users can vote for changes, which means that Ripple's technology can evolve over the time without intervention.

RIPPLE consensus algorithm is similar to BFT, but has own advantages. The key feature is that it is based on collectively trusted sub-networks. RIPPLE considers that this approach significantly decreases a consensus latency, and proves that a single XRP transaction takes about 4 seconds to be confirmed against BITCOIN transactions [54]. Sub-networks are created through *Unique Node List* (UNL). It is actually a list of trusted servers (validators) for a current user of the network. A consensus achieving process is simple. Firstly, servers agreed on transactions set by a simple rule: to agree in cases when the trusted majority of validators agree and to disagree otherwise. Then, when the final version of a ledger is known, each server independently computes a new ledger and finally, results are compared. If a validated result differs, then the node should download the latest ledger version from other nodes in the network.

It is known that RIPPLE consensus algorithm will tolerate up to 20% of fraudulent nodes since a transaction is only approved if 80% of the UNL of a server agrees with it. This property is also called correctness. All non-faulty nodes will reach consensus on the same set of transactions regardless of their UNLs [55] when:

$$\forall i, j. |\mathrm{UNL}_i \cap \mathrm{UNL}_j| \geq \frac{1}{5} max(|\mathrm{UNL}_i|, |\mathrm{UNL}_j|).$$

It is important for any consensus algorithm to converge in a finite time. RIPPLE uses an upper bound for the termination of the algorithm. It considers communication latency between nodes. If latency is larger than a defined bound, nodes are removed from all UNLs.

For now, RIPPLE gives a recommended set of transaction validators, which means that by default the network is not actually fully decentralized because you rely on the given servers (of course, they can be changed). But eventually, RIPPLE indents to entirely remove themselves from the process of selecting UNLs.

RIPPLE has no native support of smart contracts, there are only a few API methods available [56]. Developers can use CODIUS platform, which was created by Ripple company to remove this flaw. With CODIUS, smart programs can hold assets of their own on any or multiple cryptography-based ledgers, such as XRP and BITCOIN as well [57].

### *C.* STELLAR

STELLAR network is similar to RIPPLE, a consensus algorithm works the same way, but UNLs are replace by *Quorum Slices*. Stellar has a set of modern features like multisigning, time bounds and batching for transactions, but has no smart contracts (a few API methods are given [58]).

STELLAR introduces a concept of custom assets besides native Stellar assets (*Lumans*). If someone issues an asset, he becomes an anchor, and than can require an authorization to access his assets. Anchors are bridges between currencies and the STELLAR network. Users can send assets after setting up a *trustlines*. The trustline is a way to tell the network that the user trusts an asset. Stellar cannot control things out of the network, every user should explicitly trust a creator of the asset he uses.

Blockchain is usually represented as a linked list, and there are projects, which consider that blockchain itself is a problem. They introduce an alternative method of data storing, based on an efficient data structure.

### *D.* DAG-*based approach*

Directed acyclic graph, DAG, is a directed graph data structure with a topological ordering on nodes [59]). DAGs are used as an alternative method for storing decentralized data. Recall that usually blockchain is represented as a linked list. While many blockchain projects are trying to optimize chains using payment channels, hubs or other approaches, the DAG-based approach changes the entire data structure.

Blocks cannot be created simultaneously in blockchain. Moreover, creating and validating a block usually takes a long time. The key idea of DAG-based approach is to divide transactions between chains. The DAG-based network doesn't link new transaction with old ones. It would make the network too wide and validation more complicated. The goal of the approach is to keep the network width within a certain range that can support quick transaction validations. Instead, DAG-based currencies have algorithms to construct chains in such a way that the most recent transactions are likely to be approved. In case if a double or incorrect spending occurs, one of the correct graph chains will eventually grow heavier than another, and the lighter one will be abandoned. However, it means that transactions cannot be considered as confirmed ones right after adding into a chain.

Let us look at DAG-based networks, which are successfully using this technology.

IOTA is a cryptocurrency that is based on the network uses a special DAG called TANGLE [60]. IOTA's network has its native cryptocurrency called MIOTA and an ability to attach some custom data to transactions. This cryptocurrency eliminates mining. Users of the network perform the role of a validator and an issuer at the same time. In order to confirm a transaction two previous transactions should be validated, but to prevent spamming of the network some proof of work is required too. This concept excludes expensive transaction fees and allows to perform micro-transactions. It also means that the more people are online, the faster transactions will be validated.

IOTA also uses its own cryptographic algorithm, but on September 2017, MIT issued a report about serious vulnerability, and it is unknown if this will happen again [61].

An another IOTA's disadvantage is that its consensus algorithm is vulnerable to the 34% attack. The network currently is using *the coordinator* node to prevent malicious activity [62]. The coordinator controls DAG-tree, and every transaction goes through the coordinator. For the network users, this means that transactions will be processed slowly then it was expected. For now, the coordinator is a part and parcel of the network, but it is planned that some day IOTA will have enough activity to start working without it [63]. Finally, IOTA doesn't support native smart contracts.

However, thanks to fast transaction validation and low-cost fees, it is possible to process smallest amounts with TANGLE, which makes IOTA an excellent payment network.

## IV. CLASSIFICATION OF CONSENSUS ALGORITHMS

One of the most important blockchain parts is a *consensus algorithm* it is based on. The *Consensus* is a problem in distributed systems where network participants need to come to a single point of view, i.e. reach an agreement, on the current system state, provided that there are faulty and unscrupulous nodes in the network whose purpose is to disable the system or even more — to get control over it.

There are already a lot of different consensus algorithms. We do not set goals to strictly describe all existing consensus algorithms. We will only give a classification and an intuitive understanding of the essence of existing algorithms. For now, there are two fundamentally different types of consensual algorithms: *proof-based* and *voting-based*. The detailed description of them is presented below in the corresponding subsections.

## V. PROOF-BASED ALGORITHMS

Algorithms of this type are based on the idea that node which created a block must provide some kind of evidence in order to show the block correctness and to force the blockchain to accept the node into the chain. The exact form of the evidence depends on the concrete algorithm.

### A. Proof of Work

The first and the most popular consensus algorithm is *Proof of Work* (PoW) [1].

Actually, this is a group of algorithms whose general idea is that a group of independent nodes solves a complex computational problem. The problem should be formulated as follows: its solution should be difficult to find and easily verifiable. For example, find $x$, knowing $f(x)$. Obviously, if we know x, we can easily check whether it is a solution or not, however, finding a solution is usually an extremely difficult computational task. It is used by such systems as BITCOIN [1], ETHEREUM [2], etc. Now there are a lot of different algorithms based on PoW [64].

There are several main drawbacks of PoW consensus algorithms. First, they require enormous computing power to form and adopt a block in a circuit [65]. Second, they are subject for various types of attacks such as "51 percent" attack [3], Double-Spending [66], etc. [67, 68]. Finally, with this approach, it is necessary to stimulate the network participants, aka *miners*, and to pay them a commission fee for block creation and transaction execution.

The first two problems were partially solved by *Proof-of-Stake* algorithm.

### B. Proof of Stake

A concept of Proof-of-Stake (PoS) was first introduced in [69]. It is designed in order to solve the PoW problem, associated with the need to spend a huge amount of electricity. The basic principle of operation of this algorithm is that the probability of a participant forming a new unit is proportional to its share of assets of the total. Thus, the probability of forming a new unit does not depend on the power of the equipment. However, this approach gives a motivation to accumulate funds in one hand, which, potentially, can lead to network centralization [70]. Note that such an attack becomes too expensive and unprofitable from an economic point of view. ETHEREUM is planning to move from PoW to PoS in the nearest future [2].

However, PoS also have a lot of drawbacks. For example, as a systems, based on PoW, systems, based on PoS also vulnerable to "51 percent" attack, i.e. it is enough to have more than 50 percent of the digital assets. Besides, the rich always get richer and the poor get poorer. Thus, the system tends to become centralized.

There are many variants of proof-based consensus algorithms, including combining PoW and PoS. For example, Proof-of-Burn (see V-C), Delegated Proof-of-Stake (see V-D), Delayed Proof-of-Work (see V-E), Proof-of-Authority (see V-F). Below we give a brief description of the above algorithms.

### C. Proof-of-Burn

To date, there is no algorithm that solves all the problems of PoS, but the Proof-of-Burn (PoB) [71] algorithm partially solves one of them —the "51 percent" attack.

The principle of operation of this algorithm is similar to the PoS algorithm. The difference is that everyone who wants to become a miner must donate, or *burn*, any amount of his assets. Burning occurs by sending money to an address from

which it is guaranteed it is not able to spend them. After that, the probability that a participant will be able to form a new block is proportional to his stake in the donated (*burned*) assets of the total number of burned assets (but not his percent of total assets as opposed to PoS).

With this approach, with an increase of miners number, the “51 percent" attack obviously becomes extremely expensive, but theoretically, it is still possible. Currently, PoB is used by SLIMCOIN [72].

### D. Delegated Proof-of-Stake

Another evolution of Proof-of-Stake algorithm is *Delegated Proof-of-Stake* (DPoS) [73]. A distinctive feature of this algorithm is that systems using it are partially centralized. In addition to the usual roles in the system, there is one new role — “leaders”, which can be nodes that have publicly declared their readiness to continuously support the operation of a full-fledged network node, timely perform transaction verification and form new blocks.

The consensus based on the modified proof-of-stake works according to such a rule that each user can optionally put up his candidacy for the post of verifying workstation by validator nodes, or *validators*. Then among all users a vote is taken for candidates, where the weight of each vote is determined by the sum of the assets of the voter. According to the results of voting, $N$, a natural number that selects the community, usually $20 - 50$, candidates who are given the right to form new blocks of transactions. Protocol rules guarantee correct decision making if most of the assets participating in the voting are controlled by honest users.

The validators selected as a result of voting are mixed in a pseudo-random manner, forming a queue. Mixing is performed by a special algorithm, so it is impossible to predict the queue in advance, but it turns out the same for all honest members of the network. Next, there is a time period for which each of the validators must form one block, respectively, of the queue. Moreover, each validator in this period is allocated a strictly limited time interval, usually 1 sec. Either the validator manages to check the new transactions and form a new unit based on the previous one, or the next validator in the queue will do this work. After the time period is completed, the validators are again mixed and form a new queue.

It is also important to note that asset holders can re-vote candidates for an arbitrary time. Consequently, the current group of validators may change and a new queue of validators will be formed by a different composition. In addition, one asset holder can vote for more than one candidate, distributing the weight of his coins proportionally among several candidates. Features of this algorithm are:

- Low energy consumption;
- The system is partially centralized;
- High block adoption speed (due to relatively few validators);
- Participants with a large share of stake can vote for themselves to become validators (It will become too expensive when the network grows).

### E. Delayed Proof-of-Work

To ensure maximum security, there has been proposed Delayed Proof-of-Work algorithm. Delayed Proof-of-Work [74] is a consensus based on the fact that one blockchain can take advantage of the security of another blockchain. Blockchain system based on this consensus algorithm has *notary* nodes selected by other nodes, which add data from the first to the second blockchain.

Blockchain systems based on this consensus algorithm have high safety, but only blockchains based on PoW, PoS can be a part of this algorithm.

Currently used in Komodo [75].

### F. Proof-of-Authority

The basic idea behind Proof-of-Authority is that the rights for blocks creation and transaction verification are granted only to the most trusted validators.

Anyone who has publicly declared his or her or their readiness to maintain network operation uninterruptedly can become a validator. Also the following conditions must be met:

1) Validators identity must be verified.
2) The right to become a validator should be difficult to obtain. For example, potential validators should obtain a license to carry out this activity.
3) There should be complete uniformity in all check and receipt documents procedures for validators.

Thus, transactions can only process by validators. Blocks are created using only the computing power of validators.

Features of this algorithm are:

- Low energy consumption
- High speed
- A bit centralized (used mainly in private blockchains)
- If you capture server validator nodes, you can gain control of the network.

## VI. VOTING-BASED ALGORITHMS

Algorithms of this class are based on *voting*. The typical behavior of this class of algorithms is as follows.

One of the network nodes forms a new unit and invites the rest to accept it into the network. Next comes a vote among network members, according to the results of which the block is either accepted into the network or rejected. This type of consensus algorithm is mainly used in private networks.

The task of the *Byzantine generals* (Byzantine Fault Tolerance [76]) is the classical problem of the interaction with remote objects, which is relevant in the voting-based consensus algorithms.

The essence of the problem is as follows. Let there be n generals who besiege the fortress. Each of them can retreat or attack, it is important that they all have to make the same decision. Generals can communicate with each other only through messengers (they may not walk), and besides, there may be traitors among the generals. If one general attacks without remaining, then the offensive ends in defeat.

There are many solutions to this problem, differing in the principle of voting and the choice of the node generating new blocks. We will consider the main ones:

- Practical Byzantine Fault Tolerance (pBFT) [77];
- Federated Byzantine Agreement (FBA) [78];

### A. Practical Byzantine Fault Tolerance (pBFT)

pBFT is one of the first solutions to the problem of the Byzantine generals, which was proposed by Miguel Castro and Barbara Liskov in 1999 [79].

The pBFT model mainly focuses on providing a practical Byzantine ultimate replication machine (the process of transferring information from one node to all others), which allows Byzantine faults (malicious nodes), assuming that there are independent node failures and manipulated messages distributed by specific, independent nodes . The algorithm is designed to work in asynchronous systems and is optimized for high performance.

The essence of this algorithm is that all nodes in the pBFT model are ordered, with one of the nodes being the main (leader), and all the other side (backup) ones. They all communicate with each other through message passing. The main goal of the system is that all nodes come to a common agreement (consensus) on the state of the system. Nodes are constantly interacting and they need not only to prove that messages came from a particular node, but also to check that this message has not changed in the process of transmission.

Blockchains based on this algorithm have a high bandwidth but are partially centralized.

Pros: high throughput.

Cons: partially centralized, private.

Currently used in Hyperledger Fabric [80].

### B. Federated Byzantine Agreement

FBA [78] is another class of solutions for BFT problems. Used in Stellar and Ripple.

The main difference between FBA and pBFT is that each node decides who to trust and to whom not to. Thus, quorum areas are formed, that is, decision groups (Quorum Slice) and the network is divided into many such overlapping groups. A node may consist of several such groups. Decisions are first made within the trusted group and then distributed to the entire network.

FBA is considered one of the best distributed consensus due to its low transaction costs and high bandwidth.

## VII. Discussion

Beyond dispute, blockchain is now one of the most popular technology trends. Many companies and organizations start using blockchain on a wave of this popularity even if they do not really need it. Also, there are many publications describing various use cases of the blockchain [81]. In many proposed cases blockchain is not the most appropriate solution and may be replaced by other corresponding technology. It raises the important question: whether blockchain's use cases are bounded by the financial sector or not?

Another question relates to the scalability problem of blockchains. For example, Visa can process around 24000 tx/s[82]. It still faster than many blockchains. Thus, to satisfy the needs of users, blockchain must be scalable. Indeed there exist several solutions to overcome this problem but each solution comes with some trade-offs. The creation of new approaches is not only theoretical but also of practical interest due to users needs.

Blockchain provide honored immutability, decentralization, and security, but systems that are based on this technology are less flexible in case of required rollbacks and history updates. Vivid examples are the DAO Hack [83] and the lost or an unrecoverable password, for example, in case of the death of a wallet owner[84]. Thus, the more flexible solutions are needed but it is just a trade-off.

The interesting topic in research may be found in designing blockchains in a such way that the energy consumed by the network would have value not only within the network. The good example of this is GridCoin [85] which used so-called Proof-of-Research. In GridCoin users provide their computing power for scientific research purposes and earn tokens.

One more problem is centralization problem. For example, Bitcoin was supposed to be a decentralized network but, as time goes, mining pools take over control of the most of computing power of the network which leads to its centralization[86]. Concentration or delegation tokens that could be a computing power in case of PoW, or special coins in case of PoS, in hands of a node or a group of nodes — this is a common problem for many consensus algorithms.

Formal verification is also a valuable issue for consensus algorithms. It provides more credibility and reliability for blockchains, which use these algorithms.

For smart contracts there is an important question related to its Turing completeness: is the Turing completeness necessary for writing smart contracts? Programs are written in Turing incomplete languages easier to verify then programs written in Turing complete languages.

Moreover, the existence of smart contracts in many blockchain platforms with different languages can raise the question about universal language for writing smart contracts. There exist many language workbenches such as MPS [87], Xtext [88], Meta-Edit+ [89]. They provide tools to create DSLs and hierarchy of languages raising the level of abstraction. It can decrease difficulty, reduce the number of errors and vulnerability, and even allows non-programmers to write smart contracts. Upon the provided base language several layers of languages specific use cases could be constructed. For example, a rigorous language with limited features there may be used for a simple ordinary money-transfer application, that is verifiable. Nevertheless, the problem of semantic and correctness of programs written in the base language arises, because these properties are determined by the language of the target platform. Also, there is a need to prove the correctness of a translation between languages.

Due to the increased usage of smart contracts legal issues

figures prominently. At the time of writing the legal status of smart contracts is unclear in most jurisdictions. Because of the distributed nature of blockchains, unified inter-state legal status for smart contracts is highly desirable. Otherwise, the smart contract should be consistent with the local state laws for any node that significantly complicates a development process. And there other many questions that must be solved. How to handle a case when a participant of the transaction does not provide an identity? What about taxes in case of trade operations?

It is worth noting in the context of smart contracts about DAO. Right know it is unclear whether or not DAO can replace traditional organization structure. The following questions are still open and are actively discussed.

- Can an organization be successfully governed by smart contracts and controlled via tokens of shareholders? (democracy form of voting)
- How is the legal status of such organization defined?
- How can be vulnerabilities of DAO avoided in terms of programming code?

## VIII. Conclusion

In this paper, we have described the most popular blockchain networks and consensus algorithms. We did not set a goal to provide a detailed description in the limited space of the article. We just tried to give a general idea of the described blockchain systems and an intuitive understanding of consensus algorithms with our observations and direct links to corresponding publications.

## References


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

[2] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," 2017. URL https://ethereum.github.io/yellowpaper/paper.pdf

[3] "After ethereum classic suffers 51% hack." URL https://www.forbes.com/sites/ginaclarke/2019/01/09/after-ethereum-classic-suffers-51-hack-experts-consider-will-bitcoin-be-next/#3a80131ca56b

[4] Forbes, "Big blockchain: The 50 largest public companies exploring blockchain." URL https://www.forbes.com/sites/michaeldelcastillo/2018/07/03/big-blockchain-the-50-largest-public-companies-exploring-blockchain/#3018097d2b5b

[5] Z. Zheng, S. Xie, H.-N. Dai, and H. Wang, "Blockchain challenges and opportunities: A survey," *Work Pap.–2016*, 2016.

[6] G.-T. Nguyen and K. Kim, "A survey about consensus algorithms used in blockchain." *Journal of Information processing systems*, 2018.

[7] D. Yaga, P. Mell, N. Roby, and K. Scarfone, "Blockchain technology overview," National Institute of Standards and Technology, Tech. Rep., 2018.

[8] X. Li, P. Jiang, T. Chen, X. Luo, and Q. Wen, "A survey on the security of blockchain systems," *Future Generation Computer Systems*, 2017.

Last access to links: 9.02.2019

[9] E. B. Hamida, K. L. Brousmiche, H. Levard, and E. Thea, "Blockchain for enterprise: overview, opportunities and challenges," in *The Thirteenth International Conference on Wireless and Mobile Communications (ICWMC 2017)*, 2017.

[10] L. S. Sankar, M. Sindhu, and M. Sethumadhavan, "Survey of consensus protocols on blockchain applications," in *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS)*. IEEE, 2017, pp. 1–5.

[11] F. Tschorsch and B. Scheuermann, "Bitcoin and beyond: A technical survey on decentralized digital currencies," *IEEE Communications Surveys & Tutorials*, 2016.

[12] C. Cachin and M. Vukolić, "Blockchain consensus protocols in the wild," *arXiv preprint arXiv:1707.01873*, 2017.

[13] "Storing bitcoins." URL https://en.bitcoin.it/wiki/Storing$_{\text{b}}$itcoins

[14] D. Johnson, A. Menezes, and S. A. Vanstone, "The elliptic curve digital signature algorithm (ECDSA)," *Int. J. Inf. Sec.*, 2001.

[15] S. Delgado-Segura, C. Pérez-Sola, G. Navarro-Arribas, and J. Herrera-Joancomartı, "Analysis of the bitcoin utxo set," in *Proceedings of the 5th Workshop on Bitcoin and Blockchain Research Research (in Assocation with Financial Crypto 18), Lecture Notes in Computer Science*, 2018.

[16] "Bitcoin script." URL https://en.bitcoin.it/wiki/Script

[17] "Bitcoin block reward halving countdown." URL https://www.bitcoinblockhalf.com/

[18] "Bitcoin difficulty." URL https://en.bitcoin.it/wiki/Difficulty

[19] "Bitcoin transcation chart." URL https://www.blockchain.com/en/charts/transactions-per-second

[20] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, 2014.

[21] Y. Guo and C. Liang, "Blockchain application and outlook in the banking industry," *Financial Innovation*, vol. 2, no. 1, p. 24, 2016.

[22] M. Raikwar, S. Mazumdar, S. Ruj, S. S. Gupta, A. Chattopadhyay, and K.-Y. Lam, "A blockchain framework for insurance processes," in *New Technologies, Mobility and Security (NTMS), 2018 9th IFIP International Conference on*. IEEE, 2018, pp. 1–4.

[23] C. Sitonio and A. Nucciarelli, "The impact of blockchain on the music industry," 07 2018.

[24] A. Spielman, "Blockchain: digitally rebuilding the real estate industry," Ph.D. dissertation, Massachusetts Institute of Technology, 2016.

[25] M. Conoscenti, A. Vetro, and J. C. De Martin, "Blockchain for the internet of things: A systematic literature review," in *Computer Systems and Applications (AICCSA), 2016 IEEE/ACS 13th International Conference of*. IEEE, 2016, pp. 1–6.

[26] X. Yue, H. Wang, D. Jin, M. Li, and W. Jiang, "Healthcare data gateways: found healthcare intelligence on blockchain with novel privacy risk control," *Journal of medical systems*, vol. 40, no. 10, p. 218, 2016.

[27] D. M. Maslove, J. Klein, K. Brohman, and P. Martin, "Using blockchain technology to manage clinical trials data: A proof-of-concept study," *JMIR medical informatics*, vol. 6, no. 4, p. e11949, 2018.

[28] A. F. da Conceição, F. S. C. da Silva, V. Rocha, A. Locoro, and J. M. Barguil, "Eletronic health records using blockchain technology," *arXiv preprint arXiv:1804.10078*, 2018.

[29] "Ethereum transcation chart." URL https://bitinfocharts.com/comparison/ethereum-transactions.html

[30] "Ethereum ethash." URL https://github.com/ethereum/wiki/wiki/Ethash

[31] "Casper protocol." URL https://github.com/ethereum/research/blob/master/papers/casper-basics/casper$_{\text{b}}$asics.pdf

[32] "Hyperledger consortium." URL https://www.hyperledger.org/

[33] "Hyperledger iroha glossary." URL https://iroha.readthedocs.io/en/latest/core$_{\text{c}}$oncepts/glossary.html

[34] F. Muratov, A. Lebedev, N. Iushkevich, B. Nasrulin, and M. Takemiya, "Yac: Bft consensus algorithm for blockchain," *arXiv preprint arXiv:1809.00554*, 2018.

[35] "Overview of iroha." URL https://iroha.readthedocs.io/ru/latest/overview.html

[36] "Hyperledger iroha smart contract design." URL https://github.com/hyperledger/iroha/issues/249

[37] "Hyperledger fabric." URL https://hyperledger-fabric.readthedocs.io/en/release-1.3/

[38] G. Couteau, "Zero-knowledge proofs for secure computation. cryptography and security," PSL Research University, 2017.

[39] "Micropayment channel." URL https://bitcoin.org/en/developer-guide#micropayment-channel

[40] "Sidechains. how to scale and improve blockchains safely." URL https://www.forbes.com/sites/forbestechcouncil/2018/11/27/sidechains-how-to-scale-and-improve-blockchains-safely/

[41] "Solutions to scalability. can off-chain payments scale blockchain." URL https://etherworld.co/2018/06/24/solutions-to-scalability-can-off-chain-payments-scale-blockchain/

[42] "Sharding: what it is and why so many blockchain protocols rely on it." URL https://www.computerworld.com/article/3336187/blockchain/sharding-what-it-is-and-why-so-many-blockchain-protocols-rely-on-it.html

[43] "The importance of blockchain interoperability." URL https://medium.com/wanchain-foundation/the-importance-of-blockchain-interoperability-b6a0bbd06d11

[44] "Wanchain." URL https://wanchain.org/

[45] "Eos." URL https://eos.io/

[46] "Cosmos." URL https://cosmos.network/

[47] "Polkadot: vision for a heterogeneous multi-chain framework." URL https://polkadot.network/PolkaDotPaper.pdf

[48] "Cosmos white papeer." URL https://github.com/cosmos/cosmos/blob/master/WHITEPAPER.md

[49] "Tendermint core." URL https://media.readthedocs.org/pdf/tendermint/v0.21.0/tendermint.pdf

[50] "Cosmos governance." URL https://cosmos.network/docs/spec/governance/

[51] "A blockchain to connect all blockchains, cosmos is officially live." URL https://www.coindesk.com/a-blockchain-to-connect-all-blockchains-cosmos-is-now-officially-live

[52] M. Hearn, "Corda: A distributed ledger," *Corda Technical White Paper*, 2016.

[53] "Coin market cup." URL https://coinmarketcap.com/

[54] "Xrp. the digital asset for payments." URL https://ripple.com/xrp/

[55] D. Schwartz, N. Youngs, A. Britto *et al.*, "The ripple protocol consensus algorithm," *Ripple Labs Inc White Paper*, vol. 5, 2014.

[56] "rippled api reference." URL https://developers.ripple.com/rippled-api.html

[57] "What is codius?" URL https://codius.org/docs/overview/what-is-codius

[58] "Horizon reference overview." URL https://www.stellar.org/developers/reference/

[59] S. Lee, "Explaining directed acylic graph (dag), the real blockchain 3.0." URL https://www.forbes.com/sites/shermanlee/2018/01/22/explaining-directed-acylic-graph-dag-the-real-blockchain-3-0

[60] "Meet the tangle. a quick introduction to the data structure behind iota's distributed ledger and protocol." URL https://www.iota.org/research/meet-the-tangle

[61] E. Heilman, N. Narula, T. Dryja, and M. Virza, "Iota vulnerability report: Cryptanalysis of the curl hash function enabling practical signature forgery attacks on the iota cryptocurrency," 2017.

[62] "Consensus on the tangle." URL https://docs.iota.org/introduction/tangle/consensus

[63] I. Foundation, "Coordinator. part 1: The path to coordicide." URL https://blog.iota.org/coordinator-part-1-the-path-to-coordicide-ee4148a8db08

[64] G. Nguyen and K. Kim, "A survey about consensus algorithms used in blockchain," *JIPS*, vol. 14, no. 1, pp. 101–128, 2018.

[65] "Inefficiency proof of work," 2015. URL https://motherboard.vice.com/en$_{u}$s/article/ae3p7e/bitcoin-is-unsustainable

[66] G. Karame, E. Androulaki, and S. Capkun, "Two bitcoins at the price of one? double-spending attacks on fast payments in bitcoin." *IACR Cryptology ePrint Archive*, vol. 2012, no. 248, 2012.

[67] "A survey on security and privacy issues of bitcoin." URL https://ieeexplore.ieee.org/document/8369416

[68] X. Li, P. Jiang, T. Chen, X. Luo, and Q. Wen, "A survey on the security of blockchain systems," *Future Generation Computer Systems*, 2017.

[69] S. King and S. Nadal, "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake," *self-published paper, August*, 2012.

[70] "On stake and consensus." URL https://download.wpsoftware.net/bitcoin/pos.pdf

[71] Slimcoin, "A peer-to-peer crypto-currency with proof-of-burn," 2014.

[72] "Slimcoin." URL http://slimco.in/about/

[73] K. S. Myles Snider and T. Jain, "Delegated proof of stake: Features and tradeoffs," 2018. URL https://multicoin.capital/wp-content/uploads/2018/03/DPoS_Features-and-Tradeoffs.pdf

[74] Komodo, "advanced blockchain technology, focused on freedom," 2018. URL https://komodoplatform.com/wp-content/uploads/2018/06/Komodo-Whitepaper-June-3.pdf

[75] "Komodo." URL https://komodoplatform.com

[76] L. Lamport, R. E. Shostak, and M. C. Pease, "The byzantine generals problem," *ACM Trans. Program. Lang. Syst.*, 1982.

[77] M. Castro and B. Liskov, "Practical byzantine fault tolerance and proactive recovery," *ACM Trans. Comput. Syst.*, vol. 20, no. 4, pp. 398–461, 2002.

[78] J. Innerbichler and V. Damjanovic-Behrendt, "Federated byzantine agreement to ensure trustworthiness of digital manufacturing platforms," in *Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems, CRYBLOCK@MobiSys 2018, Munich, Germany, June 15, 2018*, 2018, pp. 111–116.

[79] M. Castro and B. Liskov, "Practical byzantine fault tolerance," in *Proceedings of the Third USENIX Symposium on Operating Systems Design and Implementation (OSDI), New Orleans, Louisiana, USA, February 22-25, 1999*, 1999.

[80] V. B. Elli Androulaki, Artem Barger and other, "Hyperledger fabric: A distributed operating system for permissioned blockchains," 2018.

[81] K. Zīle and R. Strazdiņa, "Blockchain use cases and their feasibility," *Applied Computer Systems*, vol. 23, no. 1, pp. 12–20, 2018.

[82] "Visa." URL https://usa.visa.com/run-your-business/small-business-tools/retail.html

[83] "50 million hack just showed dao human." URL https://www.wired.com/2016/06/50-million-hack-just-showed-dao-human/

[84] "Cryptocurrency exchange founder's death locks 140 millions." URL https://www.bbc.com/news/world-us-canada-47123371

[85] "Greed coin." URL https://www.gridcoin.us/

[86] "Bitcoin mining pools." URL https://www.buybitcoinworldwide.com/mining/pools/

[87] "Mps." URL https://www.jetbrains.com/mps/

[88] "Xtext." URL https://www.eclipse.org/Xtext/

[89] "Meta edit+." URL https://www.metacase.com/