A taxonomy of blockchain consensus protocols

Bouraga, Sarah

Published in:
Expert Systems with Applications

DOI:
10.1016/j.eswa.2020.114384

Publication date:
2021

Document Version
Publisher's PDF, also known as Version of record

A Taxonomy of Blockchain Consensus Protocols: A Survey and Classification Framework

Sarah Bouraga
Business Administration Department University of Namur
Namur, Belgium
sarah.bouraga@unamur.be

Abstract

Blockchain, the underlying technology of Bitcoin, refers to the public ledger used in a distributed network. Because blockchain does not rely on a central authority, peers have to agree on the state of the ledger among themselves, i.e., they have to reach a consensus on the state of the transactions. The way nodes reach that consensus has gained incredible attention in the literature. Bitcoin uses the Proof-of-Work (PoW) mechanism, as did Ethereum at first. The latter decided to move from PoW to Proof-of-Stake (PoS) because of the high energy consumption required by PoW. To date, many other consensus protocols have been proposed to address the limitations of the seminal ones.

In this paper, we inform researchers and practitioners about the current state of consensus protocols research. The aim is to provide an analysis of the research introducing new consensus protocols in order to enable a more unified treatment. To that end, we review 28 new consensus protocols and we propose a four-category classification framework: Origin, Design, Performance and Security. We demonstrate the applicability of the framework by classifying the 28 protocols. Many surveys have already been proposed in the literature and some of them will be discussed later in the paper. Yet, we believe that this work is relevant and important for two reasons. Firstly, blockchain being a fast evolving topic, new consensus protocols emerge regularly and improvements are also put forward on a regular basis. Hence, this work aims at reflecting the latest state-of-the-art in terms of consensus protocols. Secondly, we aim to propose a comprehensive classification framework, integrating knowledge from multiple works in the literature, as well as introducing classification dimensions that have not been proposed before.

This work demonstrates that multiple consensus have been proposed in a short period of time, and highlights the differences between these protocols. Furthermore, it is suggested that researchers and practitioners who aim to propose consensus protocols in the future should pay attention to all the dimensions presented in the classification framework.

Keywords: Blockchain, Consensus Protocols, Survey, Proof-of-Work, Proof-of-Stake, Practical Byzantine Fault Tolerance

1 Introduction

Blockchains have gained worldwide attention in the world in the last decade, with the growing popularity of Bitcoin, of which every boom and crash has been profusely documented. But what is Bitcoin and what is its relation to blockchain? Bitcoin is a decentralized cryptocurrency. Using the Bitcoin network, peers can transfer any amount of Bitcoins (BTC) in a transaction without the need for a trusted or central authority. **Blockchain** is the underlying technology of Bitcoin, it is the "decentralized transparent (public) ledger with the transaction records" [1].

Originally, the blockchain technology was solely used for monetary transactions. And even if, at the beginning, Bitcoin was considered as a speculative investment - and might still be considered as such by some - more and more organizations allow their customers to pay using Bitcoin. The interested reader is invited to consult the website coinmap.org, which offers an aggregated view of businesses accepting Bitcoin as a method of payment. Following the success of Bitcoin, many other cryptocurrencies have emerged over the years. Those other cryptocurrencies are called "atlcoins" for "alternative to Bitcoins". There are about 5000 altcoins today, the most popular ones are: Tether, Tezos, Litecoin, Monero, and Maker to name a few.

With time, blockchains progress from a technology enabling exclusively monetary transactions, to a programmable platform. Ethereum was the first blockchain that allowed the development and deployment of smart contracts and decentralized applications. This new evolution has increased the number of opportunities - and challenges - for the adoption of blockchain technology.

Whether we are dealing with a blockchain supporting money transfers and/or the design of decentralized applications, both types work without any central authority. And because blockchains do not rely on a central authority, a mechanism needs to be put in place to ensure trust in the system. This mechanism is the **consensus protocol**. It is an algorithm that ensures all the peers agree on the state of the digital ledger, i.e. on the state of the transactions.

Bitcoin uses the Proof-of-Work (PoW) consensus protocol, where nodes compete with each other to create the next block. In order to mine the next block, nodes participating in the competition need to solve an energy-intensive puzzle. Because this mechanism is not energy efficient, other consensus protocols have been proposed in order to mitigate this limitation. Examples of such consensus mechanisms are the Proof-of-Stake (PoS) and the Practical Byzantine Fault Tolerance (PBFT) and will be explained in Section 2.2.

The aim and corresponding contribution of this paper is to propose a **survey** of the latest consensus protocols, and a **framework** that will allow readers to analyze and compare those protocols alongside various dimensions. Many surveys have already been proposed in the literature and some of them will be discussed later in the paper. However, we believe that our work differs from other surveys as follows:

- Blockchain being a fast evolving topic, new consensus protocols emerge regularly and improvements are also put forward on a regular basis. Hence, this survey aims at reflecting the latest state-of-the-art in terms on consensus protocols.

- Moreover, we want to propose a comprehensive classification framework, integrating knowledge from multiple works in the literature, as well as introducing classification dimensions that have not necessarily been proposed before.

This work has multiple practical implications, for both researchers and practitioners. First of all, it will enable readers to understand the fundamentals of blockchain consensus protocols. Secondly, it will help the community to compare existing protocols and decide on the most appropriate protocol for a specific blockchain system. Finally, the framework could facilitate the design of future protocols.

The contributions and practical implications are in line with Gregor's *Nature of Theory in Information Systems* [2]. More specifically, the author presented five types of theory in Information Systems (IS), each type having distinguishing attributes. The Type 1 is the *Theory for Analyzing* and aims to describe "what is" without the intention to go beyond the analysis and description. Classification schema, frameworks or taxonomies are examples of variants of this theory type, as well as a revision of a previous classification to reflect either the coming and/or discovery of new entities, or another (and preferred) way of categorizing. In order to make a contribution to knowledge with this type of theory, little should be known about the phenomena under study [2]. This work here aims to propose a useful classification framework, which will aid in the analysis of consensus protocols. This framework will build on existing surveys to propose a more comprehensive and integrated taxonomy. Blockchains and by extension consensus protocols are a fairly new and fast-moving technology. Therefore we feel it makes sense to approach these phenomena in light of Gregor's Theory for Analyzing.

The remainder of this paper is organized as follows. The Related Work is presented in Section 2 and is composed of (i) a general background on blockchain in Section 2.1, (ii) a detailed explanation of the seminal consensus algorithms in Section 2.2, and (iii) the existing surveys on consensus protocols, as well as the newly proposed consensus protocols in Section 2.3. Section 3 introduces the classification framework and Section 4 analyzes the protocols discussed earlier using the proposed framework. Finally, Sections 5 and 6 discusses and concludes this paper respectively.

2 Related Work

2.1 Background

In 2008, Satoshi Nakamoto proposed Bitcoin, a decentralized cryptocurrency. Using the Bitcoin network, peers can transfer any amount of Bitcoins (BTC) in a transaction without the need for a trusted or central authority. As mentioned earlier, blockchain is the underlying technology of Bitcoin.

Blockchain is a decentralized public ledger storing the transactions and is "structured into a linked list of blocks" [3]. In a blockchain, transactions are stored in a block. More specifically, the block is composed of a block header and the list of transactions, which are stored in a **Merkle tree** (for more information about this specific data structure, the reader is invited to consult [4,5]). A blockchain offers multiple benefits, namely decentralization, transparency, immutability, security and privacy [6].

Over the years, the blockchain technology has evolved to fit various needs, and three types of blockchains have emerged [3, 7, 8]:

- **Public blockchains**. Examples include Bitcoin and Ethereum.
- **Private blockchains**. Private blockchains are used by a single organization.
- **Consortium blockchains**. Industry consortia using specialized private blockchains. An example is Hyperledger Fabric [9, 10].

These types of blockchains lead to a first way to classify consensus protocols, namely using the incentivized/non-incentivized dichotomy. Public blockchains are open to everyone and thus keep the original philosophy of Bitcoin, while the other two are permissioned schemes. Specifically, in public blockchains, every node can take part in the consensus process (i.e. the process ensuring that all nodes have the same version of the valid transaction history); while in consortium and private blockchains, nodes need to get permission. Public blockchains need to implement an incentive mechanism so that the nodes participating in the network are encouraged to verify and validate the transactions and the blocks (the *mining* process). Indeed, the procedure being time- and energy-consuming, the nodes typically get a fee for every block mined. On the other hand, permission-based blockchains do not necessarily need to implement such complex incentive mechanism. Trusted business relationships are an example of factors ensuring trust in the system [11].

Even if Bitcoin was the pioneer in making blockchain popular, others have also gained in popularity over the years. An example is Ethereum [12], which offers a platform with a built-in Turing-complete programming language which anyone can use to write Smart Contracts and Decentralized Applications.

2.2 Consensus Protocols

The blockchain system consists, among other elements, of nodes, and does not rely on a central authority regulating the content of the ledger. The nodes have to agree on the content themselves. They have to reach a *consensus* about the state of the ledger without any single trusted third-party. More specifically, the nodes have to agree on the blocks to be added to the chain. The consensus was defined by [13] as:

"Blockchain technology makes it possible to build an accurate ledger by relying not on a central authority but on an algorithm involving many independent people or computers called network nodes. This algorithm is called a (decentralized) consensus algorithm."

A consensus will be effective if the following elements are present [14]:

1. The *Legal structure* (i.e. laws, rules, transitions and states) of the blockchain is accepted
2. The *Agent structure* (i.e. nodes, methods and stakeholders applying the legal structure) of the blockchain is accepted
3. The *Equality structure* (i.e.members feel that all members are equal under the consensus laws) is recognized

Various authors highlighted the properties that a consensus protocol should have. These properties are presented in Table 1.

Table 1: Summary of Consensus Properties

Property	**Description**	**Articles**
Safety (or **Consistency**)	If all nodes produce the same and valid (according to the rules of the protocol) outputs, then the consensus protocol can be considered safe.	[15], [16] (drawing on the work of [17])
Liveness	If all non-faulty nodes participating in consensus produce a value, then the consensus protocol can be considered to guarantee liveness.	[15]
Validity	The set of messages delivered by the correct nodes includes all the messages broadcast by the correct nodes.	[16] (drawing on the work of [17])
Integrity.	The set of messages delivered by the correct nodes includes no false messages.	[16] (drawing on the work of [17])
Total order	All correct nodes deliver all the messages in the same order.	[16] (drawing on the work of [17])
Fault Tolerance	If the consensus can recover from the failure of a node, then the consensus can be considered to provide fault tolerance.	[15]

Multiple consensus protocols exist. The seminal ones are: (i) Proof-of-Work (PoW), (ii) Proof-of-Stake (PoS), (iii) Practical Byzantine Fault Tolerance (PBFT). They are briefly described below.

2.2.1 Proof-of-Work.

Bitcoin [18] and originally Ethereum [12] adopt the PoW protocol. The PoW mechanism is based on the resolution of a puzzle. Nodes compete with each other to create the next block. In order to create the next block, a node has to (i) verify the transactions that will be part of the block and (ii) create the header. The **block header** is composed of different elements: (i) the root of the Merkle tree (recall that the transactions are stored in the form of a Merkle tree in the core of the block; as for the block header, it will only store the root of that data structure), (ii) the hash of the previous block (linking the block with the previous one, and hence creating a *chain of blocks*), (iii) the restriction regarding the solution to the puzzle (it is the number of zeros the hash of the header has to have at the beginning of the string, the higher the number of zero, the more difficult the puzzle), (iv) the timestamp, and finally (v) the nonce (Figure 1). The header will be complete once the miner discovers the nonce, i.e. once the miner solves the puzzle (Figure 2). The resolution is based solely on computer power and not logic [19, 20].

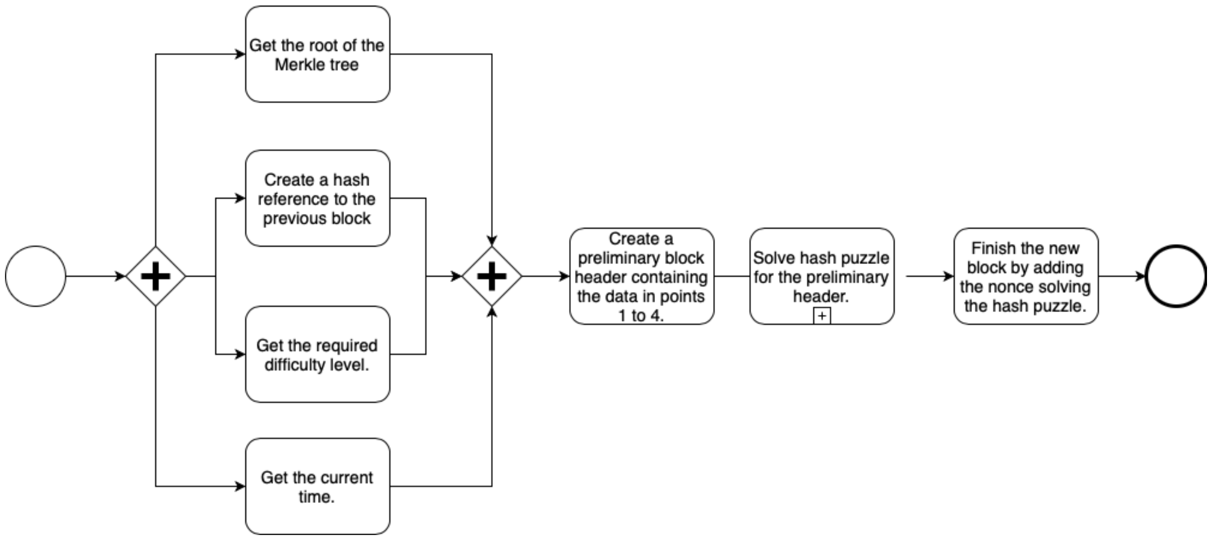
Get the root of the Merkle tree
Create a hash reference to the previous block
Get the required difficulty level.
Get the current time.
Create a preliminary block header containing the data in points 1 to 4.
Solve hash puzzle for the preliminary header.
Finish the new block by adding the nonce solving the hash puzzle.

Figure 1: Process of Block Creation

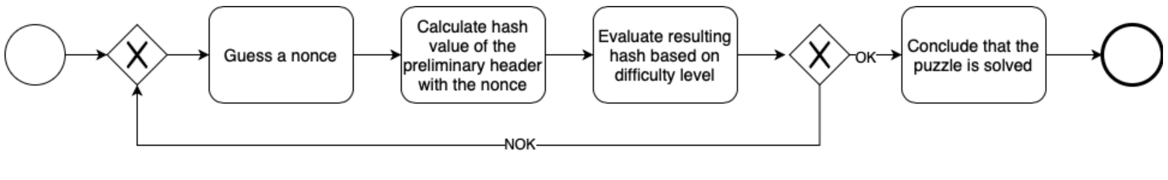
Guess a nonce
Calculate hash value of the preliminary header with the nonce
Evaluate resulting hash based on difficulty level
OK
Conclude that the puzzle is solved
NOK

Figure 2: Sub-process of Puzzle Resolution

Once a node has finalized the block, it is communicated to the other nodes of the network. The nodes verify that the block is correctly created and they add it to the (block)chain, validating its relevance to the transaction history [19].

The main limitations of the PoW consensus are [21]:

- **Tight trade-off between performance and security**. The PoW is known for its low transaction throughput (i.e. the transaction throughput refers to the number of transactions processed by second). In PoW, there is a fixed rate of block generation and there is also a maximum block size. More specifically, a block has to be generated on average every 10 minutes. The reason behind this interval is to make sure that the previous block is sufficiently propagated across the network before the creation of a new block. Options to improve performance could be to: (i) reduce block interval, or (ii) increase block size. However, on the one hand, a reduced block interval would lead to a higher transaction capacity, but also a higher risk of insufficiently propagated blocks. The latter would undermine the security of the network by increasing the likelihood of forks incidents. An increased block size would lead to the same problem. Indeed, a larger block will lead to higher transmission delays and insufficient propagation [21].

- **Energy inefficiency**. The PoW is widely criticized for its huge energy consumption. This is due to the block generation scheme [21].

- **Vulnerability to the Eclipse attack**. An Eclipse attack is an attack where a victim does not receive transactions she is interested in, because an attacker has gained control of a large number of IP addresses and has surrounded the victim with these specific addresses [22].

- **Vulnerability to Selfish mining**. A selfish mining attack occurs when a malicious user or group of users keep all the mined blocks for themselves and only broadcast them when their own chain becomes longer than the main chain of blocks. The attacker private chain thereby becomes the longer - and thus - main chain [23].

- **Mining pools and centralization risk**. First of all, because of the design behind PoW, a miner is more likely to collect a significant mining revenue if she has a significant computing power. Miners who collect a significant mining revenue can then afford more efficient, more powerful mining hardware. This positive network effect can lead to a wealth centralization risk. Moreover, today, it is extremely difficult for a solo miner to successfully participate in a PoW consensus given that she competes against mining pools - i.e. groups of miners pooling their computing resources. Moreover, a study has shown that in 2018, eight mining pools were responsible for the majority of the gross mining power. This also leads to a wealth centralization risk [21].

2.2.2 Proof-of-Stake.

The PoS consensus was first introduced by [24] in order to mitigate the high resource consumption of PoW. Ethereum transitioned from PoW to PoS for that reason.

In PoS, nodes (called validators instead of miners) deposit a stake, corresponding to an amount of coins they own (Step 1). The idea behind PoS is that the higher the stake, the higher the chance to actually validate the block and hence win the competition. The validators generate the block similarly to Nakamoto's PoW. In PoS, the validator still needs to compute the hash of the block's header and meet a certain target. The difference with the PoW is that in PoS, nodes compute the hash over a limited search space, whereas in PoW the hashing operation is done over an unlimited search space [20, 24–27].

The scheme used to calculate the probability of a node producing the next block can differ, i.e. can use different criteria. For example, [28] designed a framework for the sharing of medical imaging and used the PoS consensus mechanism. The author used the number of a specific type of transactions (*Design Study*) as the driver of the probability.

The main limitations of the PoS consensus are [21]:

- **Costless simulation problem**. As PoS does not require intensive computation, any node can simulate any segment of blockchain history costlessly; giving attackers the opportunity to fabricate an alternative blockchain.
- **Nothing-at-stake problem**. A problem where users have nothing to lose by contributing to multiple concurrent blockchains, and thereby creating forks in blockchain and not guaranteeing a single blockchain (and thus a single source of truth) [29]. The idea is that a node on a blockchain will build on every fork possible. This is true for two reasons: (i) Because the PoW is not needed anymore, it does not cost the validator anything to validate transactions on multiple branches of the blockchain; and (ii) Drawing on game theory, it is in the validators' interest to build on every fork.
- **Vulnerability to the posterior corruption**. The posterior corruption is triggered by the transparency regarding the staking history, including stakeholder addresses and staking amounts. An attacker can try to bribe nodes by promising them rewards for supporting an alternative chain containing forged transactions. The targeted nodes would be the ones who owned substantial stakes at some point in time but little at the moment. If the attacker is able to reach a high number of stakeholders, then they, together, could be able to grow an alternative (and malicious) chain that would surpass the main chain [21].
- **Vulnerability to the long-range attack**. A long-range attack occurs when a group of attackers grow a longer valid chain than the main chain, starting a few blocks after the genesis block, instead of starting a few blocks before the current block [21].

- **Vulnerability to the stake-grinding attack**. Attackers can take advantage of the publicly available staking history to distort the randomness of PoS [21].

- **Centralization risk**. This risk is similar to the centralization risk reported above for the PoW [21].

A particular case of PoS is Delegated-Proof-of-Stake (DPoS). In a DPoS consensus, nodes holding stakes vote for block verifiers. In that context, any node has the right or the chance to create blocks, even if she does not hold much stake in the network [20, 27].

2.2.3 Practical Byzantine Fault Tolerance.

The PBFT is based on the work of [30]. Informally, a blockchain adopting the PBFT consensus protocol works as follows. A node - called the client - sends a proposed block to another node - called the primary. The primary then multicasts the proposed block to multiple other nodes - called backups. If a given number of nodes agree on the proposed block, then the block is added to the chain. Otherwise the block is discarded.

More specifically, once a client sends a request (a proposed block) to a primary node, a three-phase process actually starts:

1. Pre-Prepare. The primary creates a "*pre-prepare*" message to send to the backups. The backups will accept the *pre-prepare* message if several conditions regarding the validity of the message are met. Once a backup accepts the *pre-prepare* message, it enters in the *Prepare* phase.

2. Prepare. The backups send a "*prepare*" message to the other nodes (primary and backups). The other nodes check the validity of the *prepare* message. If the *prepare* message is verified, then the nodes enter the *Commit* phase.

3. Commit. The nodes multicast a "*commit*" message to the other nodes. The nodes check and accordingly accept the *commit* message. If the *commit* message is accepted, the nodes execute the request and send a reply to the client.

The client waits for $f + 1$ replies from different nodes with the same result; this is the result of the operation [20, 25, 27, 30, 31]. This discussion is illustrated in Figure 3. For the sake of clarity, some message flows were omitted: the message flows between the primary nodes and the backup nodes, as well as the message flow between the backup nodes and the client.

The main limitation of the BFT consensus is:

- **Limited scalability**. BFT offers a good performance for a small number of replicas [32].

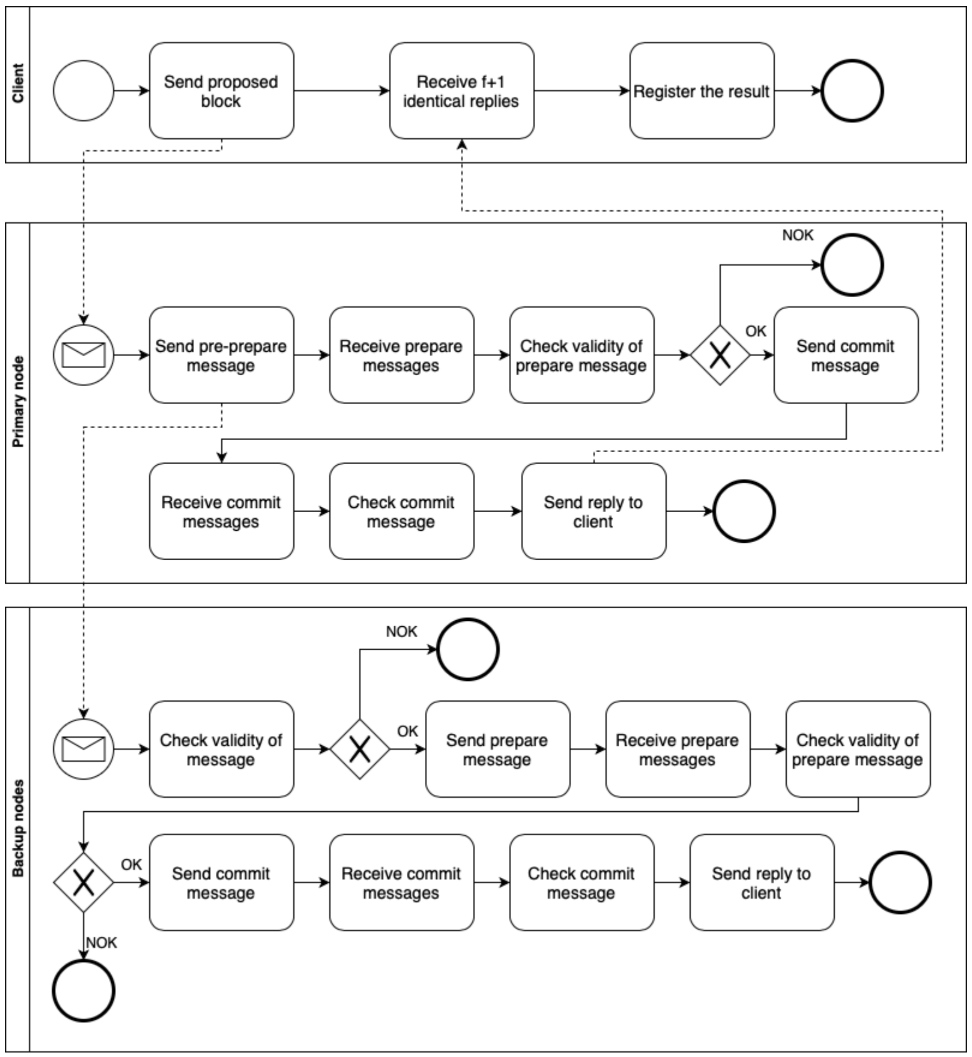

Figure 3: PBFT - Process of Block Creation

2.3 Literature Review

2.3.1 Surveys

Multiple surveys of consensus protocols have been carried out. Multiple papers review and offer a comparison of consensus protocols used in blockchains systems [16, 20, 25, 33–35]. More specifically, [33] compared different consensus for permissionless blockchains. [34] proposed a survey of multiple Proof-Based and Vote-Based consensus. [20] referenced and offered a comparison of PoW, PoS, DPoS, PBFT and Ripple; the protocols used in most blockchain systems. The authors compared the protocols based on various dimensions, namely: (i) fault tolerance, (ii) limitations, (iii) scalability, and (iv) application. [25] executed the same exercise with PoW, Chain-based PoS, BFT-style PoS, Proof-of-Elapsed Time (PoET), PBFT, and Ripple Protocol; and also compared them alongside various dimensions: (i) permission needed, (ii) third party needed, (iii) consensus finality, (iv) connectivity requirement, (v) fault tolerance, and (vi) example. [16] compared a wide range of permissioned blockchain systems (Hyperledger Fabric, Tendermint, Symbiont, R3 Corda, Iroha, Kadena, Chain, Quorum, MultiChain, Sawtooth Lake, Ripple, Stellar, and IOTA); while [35] compared Ripple and Tendermint.

Multiple works proposed a framework to analyze and compare protocols. Examples include the *five-component framework* composed of: (i) the block proposal, (ii) the block validation, (iii) the information propagation, (iv) the block finalization, and (v) the incentive mechanism [21]; the *PREStO* framework [36] which stands for and was built along the following axes: (i) Persistence, (ii) Robustness, (iii) Efficiency, (iv) Stability, and (v) Optimality; BLOCKBENCH [26] that allows for a quantitative analysis of consensus protocols; and finally, the work of [37], an evaluation framework for consensus protocols, highlighting the capabilities of protocols, such as the safety and performance characteristics.

Several works proposed an overview of the main consensus protocols [38, 39]. Another noteworthy work is [40] drawing on [41]. The authors proposed a comprehensive survey of the blockchain protocols and an analysis of cryptocurrencies. Finally, various authors focused their work on a summary of consensus protocols applicable to the Internet of Things (IoT) [42, 43]. Other authors proposed an in-depth analysis of the PoW, PoS and/or the BFT protocols [6, 32, 44–49]. [50] identified design challenges and opportunities for consensus protocols. Finally, [51] proposed a model for reasoning about properties of protocols.

Our work here builds on the existing surveys in the literature but differs on two aspects: (i) we will take into account consensus protocols that were not systematically analyzed before, and (ii) we will propose a more comprehensive classification framework. We will draw on the properties discussed before and the dimensions presented in Tables 2 to 5, but we will also propose new classification dimensions.

Table 2: Summary of existing surveys - Part 1

Characteristics	**Authors**
Alternative equilibrium concepts	[36]
Block finalization	[21]
Block generation speed	[6, 34]
Block proposal	[21]
Block validation	[21]
Centralized systems as benchmarks	[36]
Code available	[37]
Committee formation	[37]
Connectivity requirements	[25]
Consensus finality	[6, 20, 25, 32]
Consensus process	[6]
Correctness proof	[32]
Decentralization (Pools of nodes)	[34, 36]
Designing goal	[33]
DoS resistance	[37]
Double spending attack	[34]
Energy consumption	[6, 20, 32, 34, 36, 42, 52]

Table 3: Summary of existing surveys - Part 2

Characteristics	**Authors**
Experimental set up	[37]
Fairness	[36]
Fault-tolerance	[6, 16, 20, 21, 25, 26, 32, 36, 37, 42, 52]
Feature of puzzle design	[33]
Forking	[34]
Governance and sustainability	[36]
Hardware requirement	[6, 34]
Immutability	[42]
Implementation description	[33]
Incentive mechanism	[6, 21, 36]
Information propagation	[21]
Latency	[26, 32, 37, 42]
Leader selection	[6]

Table 4: Summary of existing surveys - Part 3

Characteristics	**Authors**
Liveness	[36]
Multiple committee - Intra committee configuration	[37]
Multiple committee - Intra committee consensus	[37]
Network intensive	[42]
Network synchrony assumption	[6, 32]
Node identity management	[20, 25, 26, 32, 42, 52]
Origin of hardness	[33]
Out-of-protocol incentives	[36]
Privacy	[36, 42]
Recovery from majority attacks	[36]
Safety	[36]
Scalability	[20, 26, 32, 36, 37, 42]
Security issues	[6]
Security metrics	[26]
Simulation of random function	[33]

Table 5: Summary of existing surveys - Part 3

Characteristics	**Authors**
Single committee - Committee configuration	[37]
Single committee - Inter committee consensus	[37]
Strong consistency	[37]
Throughput	[21, 26, 32, 36, 37, 42]
Transaction adding	[6]
Transaction censorship resistance	[37]
Transaction confirmation speed	[6]
Transaction throughput	[6]
Transaction scope	[36]
Trusted third party needed	[25, 42]
Weak and strong persistence	[36]
Zero Knowledge Proof properties	[33]

2.3.2 New Protocols

Multiple authors proposed new consensus protocols in order to mitigate the limitations of the ones discussed in Section 2.2. This section gives a succinct overview of the new consensus protocols. Section 4 will address them in more detail in light of the Classification framework.

First of all, several works built on and addressed the limitations of the **PoW** protocol. We will cover here the Proof-of-Luck consensus; the FruitChains blockchain; Register, Deposit, Vote (RDV); a consensus using a Two-Phase Cooperative Bargaining approach; and a consensus using the Condorcet voting mechanism. The Proof-of-Luck consensus is based on three existing consensus mechanisms - namely PoW, Proof-of-Time and Proof-of-Ownership - and offers multiple benefits such as: ensuring liveness and persistence, providing energy efficient mining and low-latency transaction validation [53]. FruitChains was proposed as a fair blockchain in order to ensure that rewards are evenly distributed among the miners of the blocks [54]. Then, RDV [55] is a solution appropriate for the IoT because of the absence of mining process. A consensus using a Two-Phase Cooperative Bargaining approach was proposed to address the problem of high computational costs of PoW. The proposed model is based on the idea "to split the transactions between multiple shards while processing them in parallel" [56]. Finally, [57, 58] proposed to address the limitations of PoW by using the Condorcet voting mechanism to determine the miner. The authors focused on public blockchains and on specific applications, namely the registration of Internet auction, of sports bets or of energy exchange.

Also, various authors built on, and proposed solutions to mitigate the limitations of **PoS**. We will cover here the following protocols: *Robust Round Robin*, the Fantômette, the CloudPoS, the Trust Consensus Protocol (Trust-CP), Delegated Proof-of-Stake with Downgrade mechanism (DDPoS), the use of weighted voting in the PoS protocol, and the Proof-of-Supply-Chain-Share (PoSCS). The *Robust Round Robin* consensus protocol was designed for permissionless blockchains, and was conceived to address the limitation of PoS of leader selection [59]. The Fantômette protocol encompasses Caucus, a secure leader election protocol, and provides game-theoretic guarantees as well as traditional security properties [60]. In CloudPoS [61], the authors addressed the use of blockchain to the management of data provenance in cloud. They used PoS, where the users stake the cloud computing resources in order to become validators. On the other hand, the Trust-CP calculates a trust score for a peer, which is then sent to the network via a blockchain transaction. The latter is added to a block and the block is validated through a PoS consensus. The goal is a blockchain-based solution to ensure trust between peers [62]. Weighted voting was also used in the PoS protocol to keep the selection and reward allocation scheme, while protecting against the effects of validators who abstain from voting [63]. Next, DDPos [64] builds on both PoW and DPoS. The authors carried out simulation experiments and concluded that the solution performs well in terms of several criteria. Indeed, the solution: (i) is more efficient than PoW and PoS, however not more efficient than DPoS, (ii) offers less decentralization than DPoS, and (iii) ensures the security of the system. Finally, a blockchain-IoT-based food traceability system (BIFTS) [65] uses the PoSCS consensus protocol, which is similar to PoS. The motivation behind the novel consensus was the fact that the existing ones were not appropriate for supply chain management because they are too specific to cryptocurrency.

Other works built on the **BFT** protocol. We will cover here the following protocols: an implicit consensus, FastBFT, Yet Another Consensus (YAC), Tendermint, Block-Supply, and Proof-of-Learning. An implicit consensus was proposed by [66] to mitigate the throughput limitations known in Bitcoin for example. The authors proposed a permissioned blockchain-based solution consisting of four layers: (i) transactions, (ii) individual blockchains, (iii) consensus scheme, and (iv) validation scheme. The novelty in their approach was mainly based on two elements: (i) the nodes stored only the transactions in which they are directly involved, and (ii) they distinguished between two types of blocks, namely the Transaction Blocks and the Check Point Blocks. Only the latter are broadcast to the other nodes in order to reach consensus. FastBFT [67] built on the BFT protocol and used a novel *message aggregation technique*, making the BFT protocol faster and more scalable. YAC [68] intended to solve problems occurring in the BFT protocols, namely inefficient message passing and strong leaders. Another important consensus is Tendermint [69], a novel termination algorithm that benefits from the gossip based nature of communication. Block-Supply [70, 71] is a blockchain system tackling counterfeit goods. The corresponding consensus protocol uses, for each new block, a set of randomly chosen validators. Each time, both the set of validators and the size of the set are different. Finally, Proof-of-Learning [72] is a protocol used in a blockchain serving as an open repository of machine learning models and datasets.

Other authors drew on multiple classic consensus protocols to propose **hybrid protocols** or address the limitations of less known consensus algorithms. Specifically, we will cover here the following protocols: Solida, Hybrid Consensus, Panda, Improved Scalable Consensus Protocol (ISCP). First, Solida [73,74] is based on PoW and Byzantine consensus and its aim was to address two limitations of the Bitcoin protocol, namely limited throughput and long confirmation time of transactions. Then, drawing from Scalable Consensus Protocol (SCP) [75], which originally was an hybrid of PoW and BFT, the authors proposed a two-step consensus mechanism: (i) an intra-committee consensus, and (ii) an inter-committee consensus. A committee is composed of randomly assigned nodes. The intra-committee validates separate sets of transactions and creates sub-blocks. The inter-committee generates the final block by including all sub-blocks. Both committees use BFT. Panda [76] is a consensus designed for permissionless blockchains. The implementation of this solution showed that the consensus performed better than PoW or BFT in terms of time to reach consensus, desired throughput and scalability. Finally, ISCP [77] was designed as an improved version of the SCP consensus protocol.

We will finish this section with the following consensus protocols: Proof-of-Vote, Lightweight consensus protocol, Proof-of-Disease, Proof-of-Play (PoP), and finally a consensus protocol module for Industrial Internet of Things (IIoT).

Proof-of-Vote [78] was designed for consortium blockchains. Participants in the blockchain are assigned one of four roles or identities, based on the idea of a voting campaign and voting mechanism. A lightweight consensus protocol appropriate for private blockchains was proposed in [79]. It is lightweight because the consensus does not rely on extensive calculations, nor an elaborate voting scheme, nor a large amount of cryptocurrency to participate in the system. Instead, the protocol relies on an announcement system, where a miner announces its intention to mine a block in the future. Proof-of-Disease [80] is a consensus applicable in a blockchain for medical decisions. Based on data about a patient's health, medical experts make a decision about a particular disease or the overall health state of that patient. PoP [81] is a consensus protocol for a blockchain-based gaming system. The authors argue that PoW, because of its limitations (high transaction costs and latency), is not appropriate for the gaming application. Moreover, the authors claimed that the other attempts of consensus were not appropriate either because they modify the game itself. Finally, a consensus protocol module for Industrial Internet of Things (IIoT) based on a reputation scheme was proposed in [82]. The idea was that their module can be implemented on top of existing consensus protocols such as PoW or PoS.

3 Classification Framework

This section introduces the classification framework that will be used to analyze and compare consensus protocols. The framework is composed of both dimensions that were already proposed by existing works, as well as new dimensions. The aim is to provide a framework that is as comprehensive as possible.

The framework is composed of four categories, which are further composed of several dimensions. The overview of the framework is proposed in Table 6 and the categories are thoroughly explained below. The Table shows that 14 out of the 23 dimensions come from an integration of the existing surveys, and the other 9 dimensions are new in this framework. For the dimensions Candidates formation, Candidates configuration, Committee formation and Committee configuration, a superscript is added in the Table. In the existing surveys, authors used the dimension Committee Formation or Committee configuration, as reported in Tables 2 to 5. In this classification framework, we separated the aspect of candidates and committee. Hence, the dimensions Candidates formation and Candidates configuration are accompanied by a $^-$ because they are new in the sense that the other surveys did not explicitly mention them (✓), however they were usually encompassed in Committee formation and Committee configuration ($^-$). On the other hand, the dimensions Committee formation and Committee configuration here are accompanied by the symbols $^+$ because they are not new to our classification framework (✗) but what they encompass is usually different from what we can find in other surveys ($^+$).

Table 6: Overview of the Classification Framework

Categories	**Dimensions**	**Novelty**
Origin	Existing PoX	✓
	Existing Theory	✓
	Answer to limitations	✓
	Accessibility	✗
	Application	✓
Design	Third party needed	✗
	Incentive	✗
	Consensus finality	✗
	Candidates formation	✓$^{-}$
	Candidates configuration	✓$^{-}$
	Leader selection	✗
	Committee formation	✗$^{+}$
	Committee configuration	✗$^{+}$
	Formal development	✓
Performance	Throughput	✗
	Latency	✗
	Fault tolerance	✗
	Scalability	✗
	Experimental evaluation	✗
Security	Sybil attack	✓
	DoS attack	✗
	Double spending attack	✗
	Eclipse attack	✓

3.1 Origin

The **Origin** category allows us to answer the following questions: *where does the protocol come from?* and *why was it proposed?*

1. **Existing Proof-of-*X* (PoX)**: On which seminal consensus protocol is the new consensus protocol based? The possible values are: PoW, PoS, and (P)BFT.

2. **Existing Theory**: On which existing theory(ies) - other than a consensus protocol - is the new consensus protocol based? Usually works will draw on existing and validated concepts to build the new algorithm.

3. **Answer to known limitation(s)**: what known limitation(s) does the new consensus attempt to solve? Indeed, if a new solution is proposed, it is usually because the existing protocols suffer from limitations that were not acceptable for the given purpose. What are those limitations?

4. **Accessibility**: Does the protocol work with a permissioned or permissionless setting?

5. **Application**: For what type of applications is the consensus protocol designed? It could be a general purpose, or a specific domain (e.g. for IoT).

3.2 Design

The **Design** category allows us to answer the following question: *how is the new protocol developed?*

1. **Third party needed**: Does the protocol rely on a trusted third party for a common service (typically, for access management)? Normally, a blockchain solution should be fully decentralized. However, in particular settings, a third-party might be needed.

2. **Incentive**: what are the incentives given to the nodes/miners to participate in the protocol? As explained earlier, a consensus protocol used in a public blockchain should have an incentive mechanism in place. This dimension documents that incentive.

3. **Consensus finality**: Vukolic̀ defined consensus finality as

   > "If a correct node p appends block b to its copy of the blockchain before appending block b', then no correct node q appends block b' before b to its copy of the blockchain."

   The consensus finality has two possible values: **probabilistic** and **deterministic** (or absolute). Probabilistic finality means that *"all written blocks (except the genesis block) are prone to revocation, although with small probabilities"* [25], while *"Deterministic means all written blocks will never be revoked"* [25].

4. **Candidates Formation**: Bano et al. [37] explained that we can observe a shift from a single node towards a group of nodes (called committee) driving the consensus. This shift is explained by two shortcomings of the single node consensus: (i) poor performance, and (ii) safety limitations. The authors in [37] proposed to analyze consensus protocols using the **criteria used to allow nodes to join a committee**. The classification framework proposed here further distinguishes this notion of committee. The dimension Candidates formation addresses the way potential block miners are selected and how they thus become "Candidate block miners". The possible values are: (i) Free access, (ii) Lottery, (iii) Permission, (iv) PoW, (v) Score, (vi) Self-interest, (vii) Stake, (viii) Vote.

5. **Candidates Configuration**: Bano et al. [37] also proposed to use the way the committee was configured as a dimension. In this framework, we look at how the block miner candidates are configured, i.e. how does the pool of candidates develop after a block has been added to the chain. The possible values are: (i) Static, (ii) Dynamic, (iii) Rolling (single), (iv) Rolling (multiple), and (v) Full swap. The values (iii) to (v) are special cases of Dynamic.

6. **Leader Selection**: If candidate block miners are grouped in a set, a candidate has to be selected eventually to mine the next block. This chosen miner is called here a *leader*. The dimension Leader selection looks at the way the next miner is effectively chosen. The idea is that, unless the next miner is chosen completely randomly, an element in the consensus setting will increase (or decrease) the likelihood of a node to be selected to mine the next block. The possible values are: (i) Lottery, (ii) None, (iii) PoW, (iv) Rank, (v) Stake, (vi) Vote.

7. **Committee Formation**: Following [37], the dimension Committee Formation examines how the nodes validating the block are selected. The candidate block miners are selected, the leader is identified, and the block is created. Now, other nodes have to validate or invalidate the block newly created. Here, we look at how those nodes are selected. The possible values are: (i) Free access, (ii) Lottery, (iii) Permission, (iv) PoW, (v) Score, (vi) Self-interest, (vii) Stake, (viii) Vote.

8. **Committee Configuration**: Following [37], the dimension Committee Configuration examines how the set of validating nodes is configured. The possible values are: (i) Static, (ii) Dynamic, (iii) Rolling (single), (iv) Rolling (multiple), and (v) Full swap. The values (iii) to (v) are special cases of Dynamic.

9. **Formal development**: does(do) the author(s) provide a formal development for her (their) proposal? This formal development can be in the forms of an algorithm and/or a proof.

3.3 Performance

The **Performance** category allows us to answer the following question: *how well does the new consensus protocol perform?*

1. **Throughput**: What is the number of successful transactions per second starting from the first transaction deployment time [83]?

2. **Latency**: For each transaction, what is the difference between the completion time and the deployment time $(t_2 - t_1)$ [83]?

3. **Fault Tolerance**: The speed and efficiency of network operations in such a way that network operations are not dependent on the non-failure of any specific node or server [14]. If the consensus can recover from the failure of a node, then the consensus can be considered to provide fault tolerance [15].

4. **Scalability**: To what extent does the protocol accommodate an increasing number of nodes and/or process growing volumes of transactions and blocks [84]?

5. **Experimental evaluation**: does(do) the author(s) provide an experimental evaluation of her (their) proposal?

3.4 Security

The **Security** category allows us to answer the following question: *Does the protocol address the protection against the following common attacks, and if it does, how efficient is it?*

1. **Sybil attack**: "An entity attempting to influence the P2P network by way of creating multiple identities and controlling" multiple nodes [27].

2. **DoS attack**: "A denial-of-service (DoS) attack is intended to prevent users from accessing a service" [27].

3. **Double Spending attack or 51 Percent attack**: "A malicious node gains control of more than 50 percent of a blockchain network's hash rate and is able to alter and manipulate blocks" [27].

4. **Eclipse Attack on the P2P Network**: "The attackers gain control over a peer's access to information in the P2P network by manipulating the network so that nodes communicate only with malicious nodes. The attacker can then manipulate the mining and the consensus mechanism" [27].

4 Application of the Framework

Tables 7 to 14 summarize the comparisons of protocols. In the Tables, the symbol ✗ indicates that the authors did not mention the aspect in their work. Also, in Table 14, the symbol ✓ is used to indicate that the consensus protects the blockchain against a specific attack, while the symbol ✗✗ indicates that it does not.

4.1 Application of the Origin Category

For the **Origin** category, we can see from Tables 7 to 9 that all three seminal algorithms were used as the basis for the development of almost all new protocols. Exceptions include Proof-of-Reputation-X (PoRX) [82], which can be implemented on top of various consensus protocols, such as PoW and PoS; and Fantômette [60] which is compatible with PoS, but could also be used for other PoX. The authors do not draw on a particular protocol. Next, about half of the new solutions drew on another existing theory (ranging from Fuzzy logic to Condorcet voting mechanism), while the other half did not. Almost all of the new algorithms aimed to address the limitations of the common consensus, namely: resource consumption (energy and/or time), limited throughput, high latency, security, to name a few. The Permissionless and Permissioned settings are well distributed; and finally, the new solutions were proposed both for general purpose and for specific applications, such as IoT.

4.2 Application of the Design Category

For the **Design** category, the discussion is summarized in Tables 10 to 12. For all consensus designed for permissioned blockchains, there need to be a permission management module. Hence, for [28, 29, 61, 65–71, 78–80, 82], a third party will be documented. In Robust Round Robin [59], identity creation can be handled in two different ways: (i) Bootstrap from Existing Infrastructures, (ii) Mining identities. In Table 10 , we document thus (i) and we also indicate there is also the possibility to manage the creation without any third party (✗✗). Other consensus that rely on a third party are [65, 72].

As far as the incentive mechanism is concerned, some consensus mechanisms explicitly state that a fee or reward will be given to the selected miner. It is the case for [28, 29, 54, 56–61, 63, 69–74, 76, 78, 80, 82]. Some consensus rely on the self-interest of the nodes [66]. Another type of incentive is a concept that influences the selection of the miner, such as the level of reputation or trust. Examples of such consensus are [55, 62, 82]. Others plan a punishment if a node behaves maliciously [55, 60, 62, 64, 70, 71, 76, 77, 82]. Finally, some works do not mention explicitly the incentive mechanism in place [53, 65, 67, 68, 75, 79, 81].

Usually, the consensus finality was not explicitly expressed by the authors. However, following the definitions provided in Section 3, we can state that [29, 53, 54, 57–59, 61, 63, 64, 69–71, 75–77, 79, 82] are probabilistic, while [55, 56, 60, 62, 65–68, 72–74, 78, 80, 81] are deterministic.

Table 7: Application of Classification Framework - Origin - Part 1

Consensus	**PoX**	**Theory(ies)**	**Limitations**	**Accessibility**	**Application**
Medical image sharing [28]	PoS	✗	✗	Permissioned	Medical image sharing
FruitChains [54]	PoW	✗	Lack of fairness	Permissionless	General
Proof-of-Luck [53]	PoW	TEE	Resource consumption, Latency	Permissionless	Decentralized electronic currency designs
Cooperative Bargaining [56]	✗	Sharding solution and Game theory	Non-supervision, Energy consumption	Permissionless	General
Condorcet [57,58]	PoS	Condorcet voting mechanism	Latency, Energy consumption	Permissionless	Auction, bet
Robust RR [59]	PoS	Round Robin	Leader selection problem	Permissionless	General
Fantômette [60]	PoS	BlockDAG	Resource consumption	Semi-permissionless	General
CloudPoS [61]	PoS	✗	PoS only effective in cryptocurrency domains	Permissioned	Data operations occurring in cloud environment
Trust-CP [62]	PoS	Trust Evaluation System	Energy consumption	Permissionless	M2M Communication
Weighted Voting [63]	PoS	Multiplicative weights updating method	Efficiency, Robustness	Permissionless	General
DDPoS [64]	PoW and DPoS	✗	Security, Stability, Efficiency	Permissionless	General

Table 8: Application of Classification Framework - Origin - Part 2

Consensus	**PoX**	**Theory(ies)**	**Limitations**	**Accessibility**	**Application**
BIFTS [65]	PoS	Fuzzy logic	Inappropriate mechanism for IoT food traceability	Permissioned	IoT for food traceability
Implicit [66]	BFT	✗	Throughput	Permissioned	Value exchange
FastBFT [67]	BFT	TEE	Scalability	Permissioned	General
YAC [68]	BFT	✗	Inefficient message passing, Strong leaders	Permissioned	Infrastructural or IoT projects
Tendermint [29, 69]	PBFT	DLS [85]	Energy consumption, Latency, Security	Permissioned	General
Block-Supply [70, 71]	Tendermint	Game theory	Efficiency	Permissioned	Anti-counterfeiting
Proof-of-Learning [72]	Algorand Byzantine Agreement approach	✗	Energy consumption	Permissionless	Machine learning tasks
Solida [73, 74]	PoW and BFT	✗	Throughput, Latency	Permissionless	General
Hybrid [75]	PoW and BFT	✗	Efficiency	Permissionless	General
Panda [76]	DPoS and BA	DAG	Energy consumption	Permissionless	Digital assets for sale and transaction
ISCP [77]	PoW and BFT	SCP	Security, Efficiency	Permissionless	General

Table 9: Application of Classification Framework - Origin - Part 3

Consensus	**PoX**	**Theory(ies)**	**Limitations**	**Accessibility**	**Application**
Proof-of-Vote [78]	✗	Voting mechanism	Resource consumption	Permissioned	General
Lightweight [79]	✗	✗	Hardware specific, Energy consumption	Permissioned	General
RDV [55]		✗	Latency, Energy consumption	Permissionless	Low-level energy devices and IoT
Proof-of-Disease [80]	✗	✗	✗	Permissioned	Medical decisions
Proof-of-Play [81]	✗	✗	Transaction cost and latency	Permissionless	Games
PoRX [82]	✗	Proof of Reputation	Security, Efficiency	Permissioned	IIoT

In [28], the author introduced a consensus for medical image sharing, and used a PoS scheme where the stake is driven by the number of Define transactions initiated by a node (*Candidates formation is Permission and Leader Selection is based on Stake*). For each block, the candidates pool will stay the same, but the stakes will change (*Candidates configuration is Static*). The author does not explicitly explain how the blocks are then validated.

The Fruitchains [54] is similar to the classical PoW, but in Fruitchains, the transactions are stored in fruits, fruits in turn are stored in blocks. The miner still has to solve a PoW (*Candidates formation is Free access and Leader selection is PoW*) and the other nodes have to validate the block proposed by the miner (*Committee formation is Free access*). Fruitchains works with permissionless blockchains, nodes can join or leave the blockchain whenever they want (*Candidates and Committee configurations are Dynamic*).

In Proof-of-Luck [53], all participants are required to use a Trusted Execution Environments (TEE), and the trusted platform vendor controls the correct execution of the algorithm inside each participant's TEE. Participants can prepare a block (*Candidates formation is Permission* by the trusted platform vendor), and the winning block will be determined randomly (*Leader selection is Lottery*). The winning block will be broadcast to other participants for validation. Each round, all nodes can be candidates and be part of the committee. Hence, the *Candidates and the Committee configurations are Static*, while the *Committee formation is Permission.*

In [56], shards are formed randomly for each epoch. In each shard, nodes create subblocks and send it to the so-called adjusting shard. The latter will combine and validate all the subblocks running a standard byzantine protocol, and finally this adjusting shard will broadcast the final to the rest of the network (*Candidates and Committee formations are Lottery, Leader selection is Lottery*). When another epoch starts, the whole process starts again too (*Candidates and Committee configurations are Dynamic*).

In [57,58], nodes need to place their service offer in the form of voting tokens to the previous miner (*Candidates formation is Stake*). The future miner will be selected based on a score using four criteria: voting token, age of the last block, reputation, and random (*Leader selection is Rank*). For the next block, the process starts again (*Candidates configuration is Dynamic*). The authors do not explicitly explain how the blocks are then validated.

In Robust Round Robin [59], each identity is assigned an age, i.e. an integer referring to the number of rounds since its enrollment or last block creation. A small set of oldest identities are chosen, the *Candidates formation* can thus be considered *Score-based*, and the *Candidates configuration Rolling single.* Then a set of endorsers is selected among the recently active identities (*Committee formation is Lottery*) in order to choose the node that will actually append its block to the chain. The miner who receives q confirmations from the endorsers will be selected as leader (*Leader selection is based on Votes*). Since the endorsers will change with each new round, we can consider that the *Committee configuration is Dynamic.*

In Fantômette [60], candidates need to place a security deposit in order to be considered as potential leaders (*Candidate formation is Stake*) and then as voters (*Committee formation is Stake*). The leader is then selected randomly using a Verifiable Random Function (VRF) and broadcast (*Leader selection is Lottery*). The authors usually made the assumption of a dynamic committee where participants can leave and join the set of participants (*Candidates and Committee configuration are Dynamic*).

In CloudPoS [61], the consensus is executed in an epoch. In each epoch, validators need to stake resources in order to be electable as leader (*Candidates formation is Stake*). The leader is selected stochastically based on the individual stakes (*Leader selection is Lottery*). The block created by the leader is then broadcast to the other validators who will either validate and add the block to the chain, or reject the block (*Committee formation is Stake*). The whole process starts again with a new epoch (*Candidates and Committee configurations are Dynamic*).

In Trust-CP [62], a five-step process is in place. First, an algorithm selects a set of nodes having a trust score higher than a given threshold (*Candidates Formation is Score-based*). Then, the block miner is randomly selected from that set of nodes (*Leader Selection is Lottery*). All other nodes (are incentivized to) participate in the validation of the block (*Committee formation is Free access*). The trust score of the miner will be adapted according to the decision of the validating committee. Since the trust score is used to filter out peers and create the set of candidate block miners, we can consider that *Candidates configuration is Dynamic.* For the next round of block creation, another node will be selected as block creator while the validating committee will stay the same except for one node entering and one node leaving (*Committee configuration is Rolling single*).

In [63], a node deposits a stake to take part in the mining process (*Candidates formation is Stake*) and the miner will be chosen "proportionally to their stake by a pseudo-random mechanism" (*Leader selection is Lottery*). This pseudo-random mechanism also selects a set of validators who will then validate the proposed block (*Committee formation is Stake + Lottery*). The whole process starts again when a new block has to be created. We can thus consider that the *Candidates and Committee configurations are both Dynamic.*

In [64], the nodes generating and validating the blocks are selected based on a two-step process: (i) PoW to select a first set of at least 201 nodes, and (ii) stake voting to select 201 nodes. From the set of 201 nodes, the top 101 - called witness nodes - will be the nodes actually generating the blocks (*Candidate formation is PoW and Stake voting*). The witness nodes take turns to generate a block (*Candidates configuration is Rolling single and Leader selection is Rank*) however, it should be noted that the witness nodes list is updated daily (*Candidates configuration* could be considered *Dynamic*). Once the block is created, the witness node broadcasts it to the other $(201 - 1)$ nodes from step two of the process (*Committee formation is PoW and Stake voting*). If the witness node gets more votes validating the block than votes invalidating the block, then the block is added to the chain. Finally, the committee members are selected based on PoW, the *Committee configuration is Dynamic.*

In [65], the creator of the block is selected based on its stake in the supply chain, evaluated by considering various factors, such as the shipment transit time, the stakeholder assessment and the shipment volume, and a roulette wheel selection (*Candidate formation is Permission and Leader Selection is based on Stake*). The stake is calculated and a miner is selected every time a new block has to be created *(Candidates configuration is Dynamic*). The authors do not explicitly explain how the blocks are then validated.

In [66], the Transaction Blocks are created by the node having an interest in the transactions composing the block (*Candidates formation is Self-interest, Leader selection is Self-interest, and the Candidates configuration is Static*). The consensus is reached on the Check Points, using a BFT algorithm. However, the authors do not mention explicitly how the validation committee is formed nor configured.

Liu et al. [67] build on BFT to propose FastBFT. Following BFT, the *Candidates formation* is considered to be *Permissioned*, the *Leader selection* - here the client - *Rank*, and the *Committee formation* is considered to be based on *Permission*. We can consider that the *Committee configuration* is a *Full swap* since the primary replica has to choose $f+1$ new active replicas; and the *Candidates configuration* is *Static* for a given set of nodes (although, the set of nodes allowed to join the network can evolve over time).

In [68], the authors present a consensus, YAC, used in Hyperledger, a permissioned blockchain. In YAC, the ordering service (OS) is responsible for the collection of transactions and the creation of a block proposal. This OS is a set of nodes, forming an abstract entity which is defined upon network creation [10,68] (*Candidates formation is Permission and Candidates configuration is Static* for a given set of nodes). There is no real leader selection, since the block is created collectively by the OS (*Leader selection is None*). The peers then validate the block proposal (*Committee formation is Permission and Committee configuration is Static* for a given set of nodes).

In Tendermint [29, 69], nodes willing to participate in the consensus have to deposit an amount of bonded coins; and in doing so, the nodes become validators. This amount corresponds to their voting power (*Candidates and Committee formations are based on Stake*). In each round, a block proposer is selected in weighted round-robin fashion such that a validator with more voting power is selected more frequently as proposer (*Leader selection is Rank*). The validators then validate or not the block created by the proposer. The validators are allowed to unlock their coins at any time (*Candidates and Committee formations are Dynamic*).

In Block-Supply chain [70, 71], the next block is proposed by the node that currently has the product (*Candidates formation is Self-interest, Candidates configuration is Dynamic, and Leader selection is Self-interest*). Each block proposer is mapped with four validation-leader nodes who randomly select nodes for the validation of the block (*Committee formation is Lottery*). Those validators change with every new block (*Committee configuration is Full Swap*).

In Proof-of-Learning [72], nodes are randomly selected to become validators. A node is more likely to become a validator the more data it stores (*Candidates formation is Lottery*). All validators then create a block and broadcast it to the other validators. If all validators agree on the block it is added to the blockchain (*Committee formation is Lottery*). There is not one single leader, since all validators create the block simultaneously (*Leader Selection is None*). The fee will thus be evenly distributed between all validators. The whole process starts again when a new task has to be handled (*Candidates and Committee configurations are Dynamic*).

Solida [73,74] is a hybrid protocol, combining PoW and PBFT. Nodes try to solve a PoW, the first one to bring the solution to the puzzle will be selected as miner (*Candidates formation is PoW* and *Leader selection is Rank*). The PoW is also used for joining the validating committee (*Committee formation is PoW*). Both *Candidates and Committee configurations* take the value *Rolling Single*: a node submitting a PoW to the validating committee leads to a committee reconfiguration, i.e. a node will leave the committee while another (the node submitting the PoW) will join it.

In [75], the authors propose a hybrid consensus which runs a PoW consensus to (re-)elect committee members (*Committee formation is PoW*), where the latter consist of recently online miners (*Candidates formation is PoW* and *Leader selection is PoW*). This process will be executed daily, which makes the *Committee configuration a Full swap*, while the *Candidates configuration is Dynamic*. Indeed, the authors propose a consensus for a permissionless setting. The candidates configuration is thus driven by the activity of the miners.

In Panda [76], a block is composed of only one transaction and is created by the initiator of that transaction (*Candidates formation is Self-interest, Leader selection is Self-interest and Candidates configuration is Dynamic*). A voting committee is only created when a fork is identified. In order to participate in the resolving of the fork, a node calculates its "consensus identity(ies)" based on its voting power (*Committee formation is Score-based*). The process starts over every time a new fork is observed (*Committee configuration is Dynamic*).

In [77], the operation is split into epochs. In each epoch, nodes are divided into multiple sub-committees based on a PoW. The block is then generated in a two-step process: (i) each sub-committee generates a sub-block and broadcasts it to the other sub-committees, (ii) each sub-committee agrees on the final block composed of all the accepted sub-blocks. Because the nodes are split into sub-committees using a PoW, both *Candidates and Committee formation are PoW*. The two-step process starts again with a new epoch (both *Candidates and Committee configurations are Dynamic*). Since the final block is generated using all accepted sub-blocks from the sub-committees, there is not one single leader (*Leader selection is None*).

In Proof-of-Vote [78], *butlers* are the nodes creating the blocks and the *commissioners* are the nodes validating the blocks. To become a butler, a node needs to register an account, submit a recommendation letter, submit a deposit, and win an election (*Candidates formation is Vote*). Once a node becomes a butler, it will be chosen randomly to produce a series of blocks (*Leader selection is Lottery*). Once the current butler has created the next block, the block has to be validated by the commissioners. The butler will create multiple blocks during a given period of time before switching to the next butler (*Candidates* *configuration is Rolling single*). Proof-of-Vote is meant to be used by consortium blockchains. In that context, a commissioner is one of the members of the consortium (*Committee formation is Permission, and Committee configuration is Static*).

In Lightweight [79], miners have to pre-announce their desire to mine blocks (*Candidates formation is Message*). This message must contain, among other elements, the miner identification number (MIN) which is the hash value of its public key concatenated with its unique address identifier. The miner who will mine the next block is the miner with a MIN closer to the hash value of the preceding block, or some other similar values (*Leader selection is Lottery*). Nodes can submit their mining application whenever they want (*Candidates configuration is Dynamic*). The author does not explicitly explain how the block is then validated.

In [55], if a node wants to vote and actively participate in the network, it needs to register by depositing a part of its coins (*Candidates formation is Stake*). The *Candidates configuration* can be considered *Dynamic* since the evolution of the committee is driven by the desire of nodes to leave registration mode. In RDV, the distinction between block creation and block validation is not really present, since the voters first vote on the transactions and they then all create the block (*Leader selection is None*). Hence, the same values apply for *Committee formation* and *Committee configuration* as for the Candidates formation and configuration.

In Proof-of-Disease [80], medical experts have to validate results from a diagnostic and add the information in the blockchain (*Candidates formation is Permission* and *Candidates configuration is Static*). The authors do not explicitly explain how the actual miner is selected nor how blocks are then validated.

In Proof-of-Play [81], if a node wants to create the next block, it must have paid enough effort in the game (*Candidates formation is PoW* and *Candidates configuration is Dynamic*). The leader is selected randomly (*Leader selection is Lottery*). The authors do not explicitly explain how the blocks are then validated.

PoRX [82] is a module that works on top of other consensus. Thus, the candidates and committee will be formed and configured according to the initial consensus (*Candidates and Committee Formation and Configuration* are not documented). However, the module is used to exploit the reputation of the nodes to enhance its likelihood to be selected as the next block miner (*Leader selection is Rank*).

Finally, the framework reports the presence or absence of a formal development in the work presenting the new consensus. This formal development can take the form of algorithms and/or proofs confirming the validity of the proposed approach. We can see in Tables 10 to 12 that it is usually the case, except for [62, 72, 77, 80, 81].

Table 10: Application of Classification Framework - Design - Part 1 - The following abbreviations are used in the header: Candidates Formation (Can Form), Candidates Configuration (Can Conf), Committee Formation (Com Form), Committee Configuration (Com Conf), Formal Development (FD); and the following abbreviations are used in the Table: Rolling Single (RS) and Rolling Multiple (RM)

Consensus	**1/3 Party**	**Incentives**	**Finality**	**Can Form**	**Can Conf**	**Leader**	**Com Form**	**Com Conf**	**FD**
Medical image sharing [28]	Permission management	Reward	✗	Permission	Static	Stake	✗	✗	✗✗
Fruitchains [54]	✗✗	Reward	Probabilistic	Free access	Dynamic	PoW	Free access	Dynamic	✓
Proof-of-Luck [53]	Trusted platform vendor	✗	Probabilistic	Permission	Static	Lottery	Permission	Static	✓
Cooperative Bargaining [56]	✗✗	Reward	Deterministic	Lottery	Dynamic	Lottery	Lottery	Dynamic	✓
Condorcet [57, 58]	✗✗	Reward	Probabilistic	Stake	Dynamic	Rank	✗	✗	✓
Robust Round Robin [59]	Permission management or ✗✗	Reward	Probabilistic	Score	RS	Votes	Lottery	Dynamic	✓
Fantômette [60]	✗✗	Reward and Punishment	Deterministic	Stake	Dynamic	Lottery	Stake	Dynamic	✓
CloudPoS [61]	Permission management	Reward	Probabilistic	Stake	Dynamic	Lottery	Stake	Dynamic	✓
Trust-CP [62]	Trust Evaluation System	Reward and Punishment	Deterministic	Score	Dynamic	Lottery	Free access	RS	✗

Table 11: Application of Classification Framework - Design - Part 2 - The following abbreviations are used in the header: Candidates Formation (Can Form), Candidates Configuration (Can Conf), Committee Formation (Com Form), Committee Configuration (Com Conf), Formal Development (FD); and the following abbreviations are used in the Table: Rolling Single (RS) and Rolling Multiple (RM)

Consensus	**1/3 Party**	**Incentives**	**Finality**	**Can Form**	**Can Conf**	**Leader**	**Com Form**	**Com Conf**	**FD**
Weighted Voting [63]	✗✗	Reward	Probabilistic	Stake	Dynamic	Lottery	Stake + Lottery	Dynamic	✓
DDPoS [64]	✗✗	Punishment	Probabilistic	PoW and Stake voting	RS	Rank	PoW and Stake voting	Dynamic	✓
BIFTS [65]	IoT Monitoring Module and Fuzzy Food Quality Evaluation Module	✗	Deterministic	Permission	Dynamic	Stake	✗	✗	✓
Implicit [66]	Permission management	Self-interest	Deterministic	Self-interest	Static	Self-interest	✗	✗	✓
FastBFT [67]	Permission management	✗	Deterministic	Permission	Static	Rank Round Robin	Permission	Full Swap	✓
YAC [68]	Permission management	✗	Deterministic	Permission	Static	None	Permission	Static ✓	
Tendermint [29, 69]	Permission management	Reward	Probabilistic	Stake	Dynamic	Rank	Stake	Dynamic	✓
Block-Supply [70, 71]	Permission management	Reward and Punishment	Probabilistic	Self-interest	Dynamic	Self-interest	Lottery	Full swap	✓
PoL [72]	IPFS	Reward	Deterministic	Lottery	Dynamic	None	Lottery	Dynamic	✗✗

Table 12: Application of Classification Framework - Design - Part 3 - The following abbreviations are used in the header: Candidates Formation (Can Form), Candidates Configuration (Can Conf), Committee Formation (Com Form), Committee Configuration (Com Conf), Formal Development (FD); and the following abbreviations are used in the Table: Rolling Single (RS) and Rolling Multiple (RM)

Consensus	**1/3 Party**	**Incentives**	**Finality**	**Can Form**	**Can Conf**	**Leader**	**Com Form**	**Com Conf**	**FD**
Solida [73, 74]	✗✗	Reward	Deterministic	PoW	RS	Rank	PoW	RS	✓
Hybrid [75]	✗✗	✗	Probabilistic	PoW	Dynamic	PoW	PoW	Full swap	✓
Panda [76]	✗✗	Reward and Punishment	Probabilistic	Self-interest	Dynamic	Self-interest	Score	Dynamic	✓
ISCP [77]	✗✗	Punishment	Probabilistic	PoW	Dynamic	None	PoW	Dynamic	✗✗
Proof-of-Vote [78]	Permission management	Reward	Deterministic	Vote	RS	Lottery	Permission	Static	✓
Lightweight [79]	Permission management	✗	Probabilistic	Message	Dynamic	Lottery	✗	✗	✓
RDV [55]	✗✗	CTR Reward and Punishment	Deterministic	Stake	Dynamic	None	Stake	Dynamic	✓
Proof-of-Disease [80]	Permission management	Reward	Deterministic	Permission	Static	✗	✗	✗	✗✗
Proof-of-Play [81]	✗	✗	Deterministic	PoW	Dynamic	Lottery	✗	✗	✗✗
PoRX [82]	Permission management	Reward and Punishment	Probabilistic	✗	✗	Rank	✗	✗	✓

4.3 Application of the Performance Category

The third category of the framework is the **Performance** category (Table 13). Most of the works considered here proposed an experimental evaluation (Exp in Table 13) of their solution, which facilitates the analysis of the various dimensions: throughput, latency, fault tolerance and scalability (respectively TP, L, FT and S in the header of Table 13).

The figures for the Robust Round Robin [59] protocol are reported for a block of size 2MB and transaction size of 250 bytes (similar to Bitcoin). The authors evaluated the evolution of the performance over a growing network site as well as a growing committee size. CloudPoS [61] provides a block latency of 5 to 10 milliseconds with 15 validators. The authors evaluated the evolution of the performance over a growing network site. To avoid any malicious behaviour, the consensus makes sure that every validator is recorded. With its default network size (103 replicas), FastBFT [67] achieves a throughput of about 500 operations per second, and a latency of 4 ms to answer a request with 1 KB payload. The authors evaluated the performance with a growing number of replicas. FastBFT requires $2f + 1$ replicas to tolerate f (Byzantine) faults. Block-Supply [70, 71] offers a latency of 16ms for committing one block, for a network size of 100 nodes. The authors evaluated the evolution of the performance over a growing network site. The consensus tolerates up to 1/3 of Byzantine nodes. In [73, 74], the authors documented the time necessary for a reconfiguration decision regarding the committee, but they do not inform on the throughput nor latency of the transactions or the blocks, nor discuss the scalability issue in depth. Solida is fully Byzantine fault tolerant, i.e. it can tolerate up to 1/3 (33.33%) of faulty nodes. In Panda [76], the throughput is close to 1200 TPS with a network size of 100, while the latency of transaction is instantaneous (0s in the Table). The authors evaluated the evolution of the performance over a growing network site. In the PoRX solution, mining competition mode can maintain 50% fault-tolerant rate [82].

Several authors provide an analysis of their consensus, nevertheless, without any experimental evaluation, we cannot report any figure in the Table. This is the case for the following works [29,53,55,59,66,69,75,78]. It can be stated that the implicit consensus [66] is more scalable than the ones on which it is based, because it reduces the message complexity to $O(N)$. RDV [55] also provides low latency because it is based on a voting approach (deterministic) instead of a lottery (probabilistic) approach.

Other authors carry out a simulation but tested other elements than the performance dimensions [56–58, 60, 63–65, 77, 81]. For instance, in [60], the authors focused on the game-theoretic aspects in the experiments. In [56], the author makes use of sharding, which allows for a better throughput and a better scalability. In [68], the authors try to find the optimal value of a given parameter, namely the vote step delay, for different network configurations. The authors also proved that YAC will function as long as there are not more than f faulty validating peers out of at least $3f + 1$ peers on the network.

Finally, the works in [28, 54, 72, 79, 80] do not address any performance dimension nor provide a rationale for them.

Table 13: Application of Classification Framework - Performance.

Consensus	**TP**	**L**	**FT**	**S**	**Exp**
Medical image sharing [28]	✗	✗	✗	✗	✗
Fruitchains [54]	✗	✗	✗	✗	✗
Proof-of-Luck [53]	✓	✓	✗	✓	✗
Cooperative Bargaining [56]	✓	✗	✗	✓	✓
Condorcet [57, 58]	✗	✗	✓	✗	✗
Robust Round Robin [59]	1500	1min	✗	✓	✓
Fantômette [60]	✗	✗	✗	✗	✓
CloudPoS [61]	✗	10-15ms	✓	✗	✓
Trust-CP [62]	✗	✗	84%	✗	✓
Weighted Voting [63]	✗	✗	✗	✗	✓
DDPoS [64]	✗	✗	✓	✗	✓
BIFTS [65]	✗	✗	✗	✓	✓
Implicit [66]	✓	✗✗	✗	✓	✗
FastBFT [67]	500	4ms	<50%	✓	✓
YAC [68]	✓	✓	33%	✗	✓
Tendermint [29, 69]	✗	✗	33%	✗	✗
Block-Supply [70, 71]	✗	16ms	33%	✓	✓
Proof-of-Learning [72]	✗	✗	✗	✗	✗
Solida [73, 74]	✗	✗	33%	✗	✓
Hybrid [75]	✓	✗	33%	✓	✗
Panda [76]	1200	0s	✗	✗	✓
ISCP [77]	✓	✗	33%	✗	✓
Proof-of-Vote [78]	✓	✓	<50%	✗	✗
Lightweight [79]	✗	✗	✗	✗	✗
RDV [55]	✓	✓	✗	✗	✗
Proof-of-Disease [80]	✗	✗	✗	✗	✗
Proof-of-Play [81]	✗	✗	✗	✗	✓
PoRX [82]	✗	✗	50%	✗	✓

4.4 Application of the Security Category

Finally, let's have a look at the **Security** category. None of the protocols considered here address all four attacks, and some of them do not address any attack at all, to focus on other aspects of the consensus [67, 68, 79]. Other works address an attack, but do not provide any proof that their solution is protected against such attack ([53] and [63] for the Sybil and Eclipse attacks). The new solutions are mostly protected against the Sybil and the double spending attacks; then the DoS attack. Only one protocol explicitly states that it is protected against the Eclipse attack.

Table 14: Application of Classification Framework - Security

Consensus	**Sybil**	**DoS**	**51%**	**Eclipse**
Medical image sharing [28]	✗	✗	✗	✗
Fruitchains [54]	✓	✗	✗	✗
Proof-of-Luck [53]	✗	✗	✗	✗
Cooperative Bargaining [56]	✗	✗	✗	✗
Condorcet [57, 58]	✓	✗	✗	✗
Robust Round Robin [59]	✗	✗✗	✓	✗
Fantômette [60]	✓	✓	✗	✗
CloudPoS [61]	✗	✗	✓	✗
Trust-CP [62]	✓	✗	✗	✗
Weighted Voting [63]	✗	✗	✗	✗
DDPoS [64]	✗	✗	✗	✗
BIFTS [65]	✗	✗	✓	✗
Implicit [66]	✗	✗	✓	✗
FastBFT [67]	✗	✗	✗	✗
YAC [68]	✗	✗	✗	✗
Tendermint [29, 69]	✗	✗	✓	✗
Block-Supply [70, 71]	✗	✓	✗	✓
Proof-of-Learning [72]	✗	✓	✗	✗
Solida [73, 74]	✓	✗	✗	✗
Hybrid consensus [75]	✓	✗	✗	✗
Panda [76]	✓	✓	✓	✗
ISCP [77]	✗	✗	✗	✗
Proof-of-Vote [78]	✗	✗	✗	✗
Lightweight [79]	✗	✗	✗	✗
RDV [55]	✗	✗	✓	✗
Proof-of-Disease [80]	✗	✗	✗	✗
Proof-of-Play [81]	✓	✗	✓	✗
PoRX [82]	✓	✗	✓	✗

5 Discussion

In this Section, we will discuss the results of Section 4 by highlighting some trends/patterns observed following the application of the framework. We also discuss the practical implications as well as the limitations of this research.

5.1 Results Analysis

From the applications of the Origin and the Design categories, we can observe that the authors who raised the problems of high energy/resource consumption and lack of efficiency of current protocols, proposed to use the **Stakes** [29, 55, 60, 63, 69], **Permission** [53], or the **Lottery** [56, 72] for both Candidates and Committee formations, a **Score** either for the Candidates formation [62] or for the Committee formation [76], or finally a **Vote and Permission** for the Candidates formation and Committee formation respectively [78]. In all these works, the leader is selected either by **Lottery** or **Rank**. These configurations make sense if the authors want to mitigate the energy consumption caused by PoW.

Other authors focused on the efficiency problem, including the throughput and latency of existing protocols. Their solutions are based on the use of **PoW** [73–75, 77], **Permission** [53, 68],and **Stake** [29, 55, 69] for both Candidates and Committee formations, or the use of **Self-interest** for the Candidates formation [66, 70, 71].

The application of the Security category shows the typical **threat model** used by authors. We can observe that the Sybil and the DoS attacks are mostly mentioned by consensus designed for permissionless blockchains. This is not surprising since permissioned blockchains control who has access to the network, mitigating the risk of duplicitous identities. On the other hand, the 51% attack is in majority mentioned by consensus proposed for Permissioned settings.

5.2 Implications

We believe that this research will be helpful for researchers and practitioners in that it can facilitate the development of new consensus mechanisms. The paper highlights the relevant constructs that we consider important when designing a new consensus. Also, this research can help readers analyze and compare existing and future consensus.

Organizations wanting to select a consensus algorithm can, as a first step, consult the flowchart on Figure 4. The flowchart was built on analyzing the results of the classification framework in Section 4. The process was built "manually" because we wanted to take advantage of the software engineering aspect of the process. More specifically, when designing any new software application, one has to take into account the requirements and constraints communicated by the user, but also be aware of the necessary trade-offs. It is indeed essential to set priorities when collecting and analyzing requirements. If the process had been generated automatically, it would have been more difficult to use that knowledge. For instance, a decision tree could have been generated but the underlying algorithm would have used the features without taking into account this logic behind an organization's choice.

When an organization wants to choose a suitable consensus for its own application, the first element to consider is the Accessibility of the blockchain: does the organization want a Permissioned or a Permissionless setting?

The process goes on with the salient features regarding the Design, the Performance, and the Security; highlighting our discussion about the prioritization of requirements: does the organization want to have control on the Leader selection process or the Candidates or Committee Formation (the Design part of the consensus), or rather on the Performance or even the Security? Depending on its priority, the organization will follow the chart which will provide some indication about a suitable consensus for its application. The reader will notice that the Incentives and Application dimensions are not present in the process. Indeed, when analyzing the values for both dimensions, one can see that these factors are not discriminant enough. Furthermore, we feel the presence or absence of incentives is more of a consequence of the consensus rather than a driver in selecting it.

This work builds on existing works to enhance and reflect the new advancement of the field. The presentation of the Classification Framework and its Applications in Sections 3 and 4 show that the results are consistent with past surveys. Indeed, for a given consensus protocol, we recorded the same values for the same dimension. This paper differs from existing works because: (i) we integrated multiple surveys to offer a unified view to the reader, (ii) we added new dimensions to offer an extended view to the reader, and (iii) we consider the most recent consensus protocol to offer a current view to the reader. Another aspect distinguishing this paper from past works is the decision process provided here, which can help an organization to choose an appropriate protocol given its requirements and priority.

5.3 Limitations

We need to acknowledge the limitations of this research. First, we purposefully, did exclude an important aspect related to the blockchain technology, namely the **block structure/content**. While this aspect is clearly important, the aim of this paper was to focus on the consensus part of the blockchain. Hence, we analyzed the flow applied by the nodes to agree on the state of transactions instead of focusing on the way transactions are actually stored. Second, we did not include **all** of the consensus protocols proposed in the literature, but focused on the most recent ones. Applying the framework to more articles would have strengthened the conclusions, yet we believe that the trends and patterns would not be significantly different.

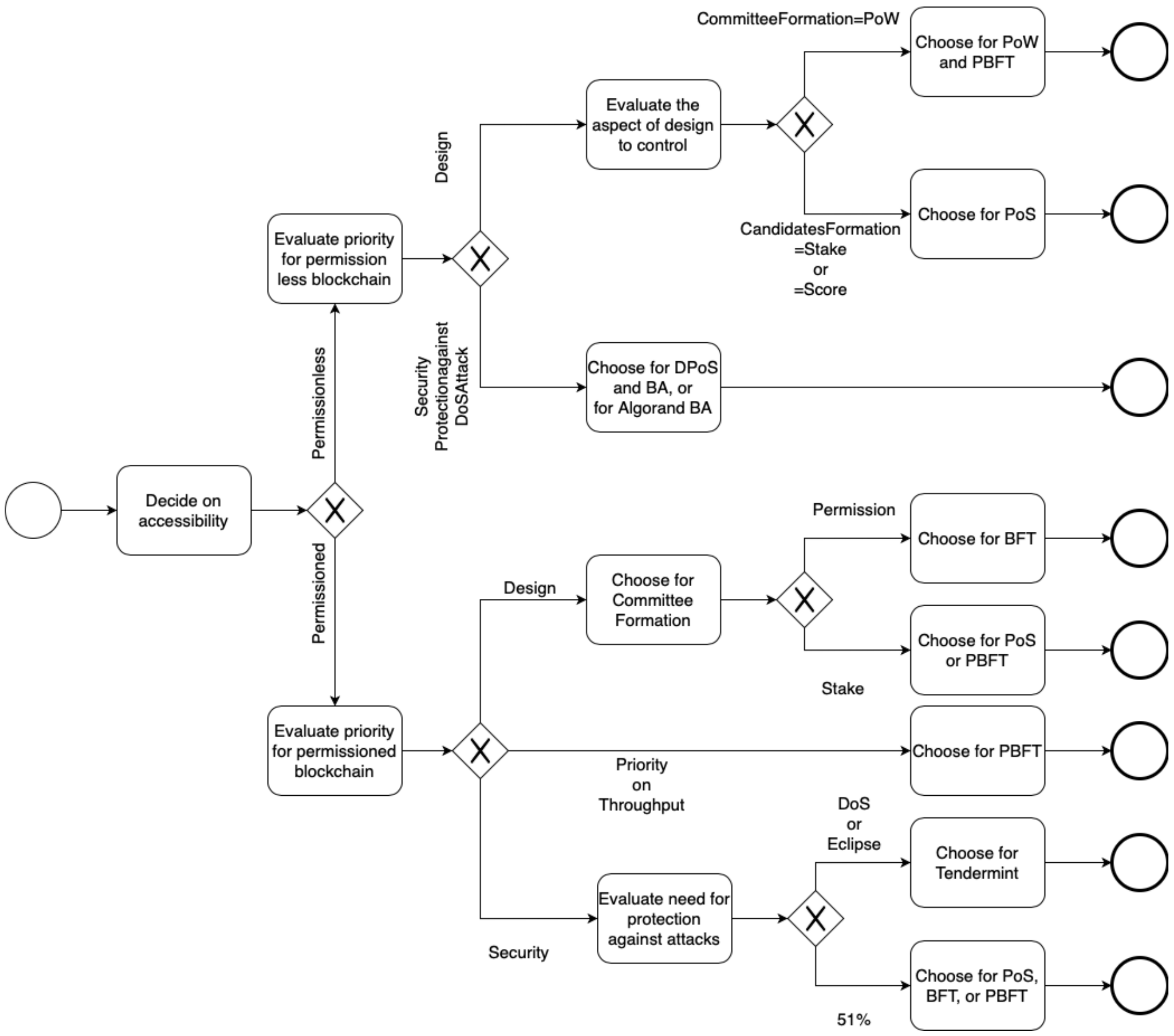
Decide on accessibility
Permissionless
Permissioned
Evaluate priority for permission less blockchain
Design
Security Protectionagainst DoSAttack
Evaluate the aspect of design to control
CommitteeFormation=PoW
Choose for PoW and PBFT
CandidatesFormation =Stake or =Score
Choose for PoS
Choose for DPoS and BA, or for Algorand BA
Evaluate priority for permissioned blockchain
Design
Choose for Committee Formation
Permission
Choose for BFT
Stake
Choose for PoS or PBFT
Priority on Throughput
Choose for PBFT
Security
Evaluate need for protection against attacks
DoS or Eclipse
Choose for Tendermint
51%
Choose for PoS, BFT, or PBFT

Figure 4: Process of Suitable Consensus Selection

6 Conclusion

Consensus is a mechanism of great importance in blockchain, where the trust in the system is not ensured by a central authority but instead where it needs to be programmatically enabled. Consensus protocols have received much attention by researchers and practitioners, because the seminal protocols (PoW, PoS and PBFT) present a series of limitations.

From the works reviewed here, we observed that most of the new protocols provide a contribution regarding the Candidates formation and configuration, Leader selection and Committee formation and configuration.

We believe that this research will be helpful for researchers and practitioners in that it can facilitate the development of new consensus mechanisms (by pinpointing the relevant constructs), but also it can help readers analyze existing and future consensus. Finally, an organization looking for guidance on how to choose a suitable consensus can use the flowchart presented in Section 5.2.

References

[1] M. Swan, *Blockchain: Blueprint for a new economy.* " O'Reilly Media, Inc.", 2015.

[2] S. Gregor, "The nature of theory in information systems," *MIS quarterly*, pp. 611–642, 2006.

[3] X. Xu, I. Weber, and M. Staples, *Architecture for blockchain applications.* Springer, 2019.

[4] R. C. Merkle, "Protocols for public key cryptosystems," in *1980 IEEE Symposium on Security and Privacy.* IEEE, 1980, pp. 122–122.

[5] ——, "A certified digital signature," in *Conference on the Theory and Application of Cryptology.* Springer, 1989, pp. 218–238.

[6] C. T. Nguyen, D. T. Hoang, D. N. Nguyen, D. Niyato, H. T. Nguyen, and E. Dutkiewicz, "Proof-of-stake consensus mechanisms for future blockchain networks: fundamentals, applications and opportunities," *IEEE Access*, vol. 7, pp. 85 727–85 745, 2019.

[7] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: A survey," *International Journal of Web and Grid Services*, vol. 14, no. 4, pp. 352–375, 2018.

[8] O. Delgado-Mohatar, J. Fierrez, R. Tolosana, and R. Vera-Rodriguez, "Blockchain and biometrics: A first look into opportunities and challenges," in *International Congress on Blockchain and Applications.* Springer, 2019, pp. 169–177.

[9] C. Cachin *et al.*, "Architecture of the hyperledger blockchain fabric," in *Workshop on distributed cryptocurrencies and consensus ledgers*, vol. 310, 2016, p. 4.

[10] J. Sousa, A. Bessani, and M. Vukolic, "A byzantine fault-tolerant ordering service for the hyperledger fabric blockchain platform," in *2018 48th annual IEEE/IFIP international conference on dependable systems and networks (DSN)*. IEEE, 2018, pp. 51–58.

[11] P. Lipovyanov, *Blockchain for Business 2019: A user-friendly introduction to blockchain technology and its business applications*. Packt Publishing Ltd, 2019.

[12] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, vol. 3, p. 37, 2014.

[13] K. Omote and M. Yano, "Bitcoin and blockchain technology," in *Blockchain and Crypt Currency*. Springer, 2020, pp. 129–136.

[14] V. Morabito, "Business innovation through blockchain," *Cham: Springer International Publishing*, 2017.

[15] A. Baliga, "Understanding blockchain consensus models," *Persistent*, vol. 2017, no. 4, pp. 1–14, 2017.

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

[17] V. Hadzilacos and S. Toueg, "Fault-tolerant broadcasts and related problems," in *Distributed systems (2nd Ed.)*, 1993, pp. 97–145.

[18] S. Nakamoto *et al.*, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[19] D. Drescher, *Blockchain basics*. Springer, 2017, vol. 276.

[20] S. Zhang and J.-H. Lee, "Analysis of the main consensus protocols of blockchain," *ICT Express*, 2019.

[21] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, "A survey of distributed consensus protocols for blockchain networks," *IEEE Communications Surveys & Tutorials*, 2020.

[22] D. Valdeolmillos, Y. Mezquita, A. González-Briones, J. Prieto, and J. M. Corchado, "Blockchain technology: A review of the current challenges of cryptocurrency," in *International Congress on Blockchain and Applications*. Springer, 2019, pp. 153–160.

[23] V. Dedeoglu, R. Jurdak, A. Dorri, R. Lunardi, R. Michelin, A. Zorzo, and S. Kanhere, "Blockchain technologies for iot," in *Advanced Applications of Blockchain Technology*. Springer, 2020, pp. 55–89.

[24] S. King and S. Nadal, "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake," *self-published paper, August*, vol. 19, 2012.

[25] Y. Xiao, N. Zhang, J. Li, W. Lou, and Y. T. Hou, "Distributed consensus protocols and algorithms," *Blockchain for Distributed Systems Security*, vol. 25, 2019.

[26] T. T. A. Dinh, R. Liu, M. Zhang, G. Chen, B. C. Ooi, and J. Wang, "Untangling blockchain: A data processing view of blockchain systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 7, pp. 1366–1385, 2018.

[27] E. Elrom, *The Blockchain Developer.* Springer, 2019.

[28] V. Patel, "A framework for secure and decentralized sharing of medical imaging data via blockchain consensus," *Health informatics journal*, vol. 25, no. 4, pp. 1398–1411, 2019.

[29] J. Kwon, "Tendermint: Consensus without mining," *Draft v. 0.6, fall*, vol. 1, no. 11, 2014.

[30] M. Castro, B. Liskov *et al.*, "Practical byzantine fault tolerance," in *OSDI*, vol. 99, no. 1999, 1999, pp. 173–186.

[31] B. Singhal, G. Dhameja, and P. S. Panda, *Beginning Blockchain: A Beginner's Guide to Building Blockchain Solutions.* Springer, 2018.

[32] M. Vukolić, "The quest for scalable blockchain fabric: Proof-of-work vs. bft replication," in *International workshop on open problems in network security.* Springer, 2015, pp. 112–125.

[33] W. Wang, D. T. Hoang, P. Hu, Z. Xiong, D. Niyato, P. Wang, Y. Wen, and D. I. Kim, "A survey on consensus mechanisms and mining strategy management in blockchain networks," *IEEE Access*, vol. 7, pp. 22 328–22 370, 2019.

[34] G.-T. Nguyen and K. Kim, "A survey about consensus algorithms used in blockchain." *Journal of Information processing systems*, vol. 14, no. 1, 2018.

[35] K. Ambili, M. Sindhu, and M. Sethumadhavan, "On federated and proof of validation based consensus algorithms in blockchain," in *IOP Conference Series: Materials Science and Engineering*, vol. 225, no. 1. IOP Publishing, 2017, p. 012198.

[36] S. Leonardos, D. Reijsbergen, and G. Piliouras, "Presto: A systematic framework for blockchain consensus protocols," *arXiv preprint arXiv:1906.06540*, 2019.

[37] S. Bano, A. Sonnino, M. Al-Bassam, S. Azouvi, P. McCorry, S. Meiklejohn, and G. Danezis, "Sok: Consensus in the age of blockchains," in *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, 2019, pp. 183–198.

[38] S. Pahlajani, A. Kshirsagar, and V. Pachghare, "Survey on private blockchain consensus algorithms," in *2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT)*. IEEE, 2019, pp. 1–6.

[39] 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.

[40] M. Sadek Ferdous, M. Jabed Morshed Chowdhury, M. A. Hoque, and A. Colman, "Blockchain consensus algorithms: A survey," *arXiv*, pp. arXiv–2001, 2020.

[41] M. J. M. Chowdhury, M. S. Ferdous, K. Biswas, N. Chowdhury, A. Kayes, M. Alazab, and P. Watters, "A comparative analysis of distributed ledger technology platforms," *IEEE Access*, vol. 7, pp. 167 930–167 943, 2019.

[42] M. Salimitari and M. Chatterjee, "A survey on consensus protocols in blockchain for iot networks," *arXiv preprint arXiv:1809.05613*, 2018.

[43] Q. He, N. Guan, M. Lv, and W. Yi, "On the consensus mechanisms of blockchain/dlt for internet of things," in *2018 IEEE 13th International Symposium on Industrial Embedded Systems (SIES)*. IEEE, 2018, pp. 1–10.

[44] R. Pass, L. Seeman, and A. Shelat, "Analysis of the blockchain protocol in asynchronous networks," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2017, pp. 643–673.

[45] I. Abraham, D. Malkhi *et al.*, "The blockchain consensus layer and bft," *Bulletin of EATCS*, vol. 3, no. 123, 2017.

[46] S. De Angelis, L. Aniello, R. Baldoni, F. Lombardi, A. Margheri, and V. Sassone, "Pbft vs proof-of-authority: Applying the cap theorem to permissioned blockchain," 2018.

[47] V. Gramoli, "From blockchain consensus back to byzantine consensus," *Future Generation Computer Systems*, 2017.

[48] J. Garay and A. Kiayias, "Sok: A consensus taxonomy in the blockchain era," in *Cryptographers' Track at the RSA Conference*. Springer, 2020, pp. 284–318.

[49] F. Saleh, “Blockchain without waste: Proof-of-stake,” *Available at SSRN 3183935*, 2020.

[50] D. K. Tosh, S. Shetty, X. Liang, C. Kamhoua, and L. Njilla, “Consensus protocols for blockchain-based data provenance: Challenges and opportunities,” in *2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON)*. IEEE, 2017, pp. 469–474.

[51] G. Pîrlea and I. Sergey, “Mechanising blockchain consensus,” in *Proceedings of the 7th ACM SIGPLAN International Conference on Certified Programs and Proofs*, 2018, pp. 78–90.

[52] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, “An overview of blockchain technology: Architecture, consensus, and future trends,” in *2017 IEEE international congress on big data (BigData congress)*. IEEE, 2017, pp. 557–564.

[53] M. Milutinovic, W. He, H. Wu, and M. Kanwal, “Proof of luck: An efficient blockchain consensus protocol,” in *proceedings of the 1st Workshop on System Software for Trusted Execution*, 2016, pp. 1–6.

[54] R. Pass and E. Shi, “Fruitchains: A fair blockchain,” in *Proceedings of the ACM Symposium on Principles of Distributed Computing*, 2017, pp. 315–324.

[55] S. Solat, “Rdv: An alternative to proof-of-work and a real decentralized consensus for blockchain,” in *Proceedings of the 1st Workshop on Blockchain-enabled Networked Sensor Systems*, 2018, pp. 25–31.

[56] S. Kim, “Two-phase cooperative bargaining game approach for shard-based blockchain consensus scheme,” *IEEE Access*, vol. 7, pp. 127 772–127 780, 2019.

[57] D. Vangulick, B. Cornélusse, and D. Ernst, “Blockchain for peer-to-peer energy exchanges: design and recommendations,” in *2018 Power Systems Computation Conference (PSCC)*. IEEE, 2018, pp. 1–7.

[58] ——, “Blockchain: A novel approach for the consensus algorithm using condorcet voting procedure,” in *2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON)*. IEEE, 2019, pp. 1–10.

[59] M. Ahmed and K. Kostiainen, “Don’t mine, wait in line: Fair and efficient blockchain consensus with robust round robin,” *arXiv preprint arXiv:1804.07391*, 2018.

[60] S. Azouvi, P. McCorry, and S. Meiklejohn, “Betting on blockchain consensus with fantomette,” *arXiv preprint arXiv:1805.06786*, 2018.

[61] D. Tosh, S. Shetty, P. Foytik, C. Kamhoua, and L. Njilla, "Cloudpos: A proof-of-stake consensus design for blockchain integrated cloud," in *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*. IEEE, 2018, pp. 302–309.

[62] B. Shala, U. Trick, A. Lehmann, B. Ghita, and S. Shiaeles, "Novel trust consensus protocol and blockchain-based trust evaluation system for m2m application services," *Internet of Things*, vol. 7, p. 100058, 2019.

[63] S. Leonardos, D. Reijsbergen, and G. Piliouras, "Weighted voting on the blockchain: Improving consensus in proof of stake protocols," in *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 2019, pp. 376–384.

[64] F. Yang, W. Zhou, Q. Wu, R. Long, N. N. Xiong, and M. Zhou, "Delegated proof of stake with downgrade: A secure and efficient blockchain consensus algorithm with downgrade mechanism," *IEEE Access*, vol. 7, pp. 118 541–118 555, 2019.

[65] Y. P. Tsang, K. L. Choy, C. H. Wu, G. T. S. Ho, and H. Y. Lam, "Blockchain-driven iot for food traceability with an integrated consensus mechanism," *IEEE Access*, vol. 7, pp. 129 000–129 017, 2019.

[66] Z. Ren, K. Cong, J. Pouwelse, and Z. Erkin, "Implicit consensus: Blockchain with unbounded throughput," *arXiv preprint arXiv:1705.11046*, 2017.

[67] J. Liu, W. Li, G. O. Karame, and N. Asokan, "Scalable byzantine consensus via hardware-assisted secret sharing," *IEEE Transactions on Computers*, vol. 68, no. 1, pp. 139–151, 2018.

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

[69] E. Buchman, J. Kwon, and Z. Milosevic, "The latest gossip on bft consensus," *arXiv preprint arXiv:1807.04938*, 2018.

[70] N. Alzahrani and N. Bulusu, "Towards true decentralization: A blockchain consensus protocol based on game theory and randomness," in *International Conference on Decision and Game Theory for Security*. Springer, 2018, pp. 465–485.

[71] ——, "A new product anti-counterfeiting blockchain using a truly decentralized dynamic consensus protocol," *Concurrency and Computation: Practice and Experience*, p. e5232, 2019.

[72] F. Bravo-Marquez, S. Reeves, and M. Ugarte, "Proof-of-learning: a blockchain consensus mechanism based on machine learning competitions," in *2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON)*. IEEE, 2019, pp. 119–124.

[73] I. Abraham, D. Malkhi, K. Nayak, L. Ren, and A. Spiegelman, "Solidus: An incentive-compatible cryptocurrency based on permissionless byzantine consensus," *CoRR, abs/1612.02916*, 2016.

[74] ——, "Solida: A blockchain protocol based on reconfigurable byzantine consensus," *arXiv preprint arXiv:1612.02916*, 2016.

[75] R. Pass and E. Shi, "Hybrid consensus: Efficient consensus in the permissionless model," in *31st International Symposium on Distributed Computing (DISC 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[76] T. Zhou, X. Li, and H. Zhao, "Dlattice: A permission-less blockchain based on dpos-ba-dag consensus for data tokenization," *IEEE Access*, vol. 7, pp. 39 273–39 287, 2019.

[77] Z.-C. Li, J.-H. Huang, D.-Q. Gao, Y.-H. Jiang, and L. Fan, "Iscp: An improved blockchain consensus protocol." *IJ Network Security*, vol. 21, no. 3, pp. 359–367, 2019.

[78] K. Li, H. Li, H. Hou, K. Li, and Y. Chen, "Proof of vote: A high-performance consensus protocol based on vote mechanism & consortium blockchain," in *2017 IEEE 19th International Conference on High Performance Computing and Communications; IEEE 15th International Conference on Smart City; IEEE 3rd International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. IEEE, 2017, pp. 466–473.

[79] K. Finlow-Bates, "A lightweight blockchain consensus protocol," *Computer Security Resource Center [accessed 8 July 2018]. Available at: https://csrc. nist. gov/CSRC/media/Publications/nistir/8202/draft/documents/nistir8202-draft. pdf*, 2017.

[80] A. K. Talukder, M. Chaitanya, D. Arnold, and K. Sakurai, "Proof of disease: A blockchain consensus protocol for accurate medical decisions and reducing the disease burden," in *2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*. IEEE, 2018, pp. 257–262.

[81] H. Y. Yuen, F. Wu, W. Cai, H. C. Chan, Q. Yan, and V. C. Leung, "Proof-of-play: A novel consensus model for blockchain-based peer-to-peer gaming system," in *Proceedings of the 2019 ACM International Symposium on Blockchain and Secure Critical Infrastructure*, 2019, pp. 19–28.

[82] E. K. Wang, Z. Liang, C.-M. Chen, S. Kumari, and M. K. Khan, "Porx: A reputation incentive scheme for blockchain consensus of iiot," *Future Generation Computer Systems*, vol. 102, pp. 140–151, 2020.

[83] S. Pongnumkul, C. Siripanpornchana, and S. Thajchayapong, "Performance analysis of private blockchain platforms in varying workloads," in *2017 26th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 2017, pp. 1–6.

[84] A. B. Bondi, "Characteristics of scalability and their impact on performance," in *Proceedings of the 2nd international workshop on Software and performance*, 2000, pp. 195–203.

[85] C. Dwork, N. Lynch, and L. Stockmeyer, "Consensus in the presence of partial synchrony," *Journal of the ACM (JACM)*, vol. 35, no. 2, pp. 288–323, 1988.