cryptography

Article

A Taxonomy of Blockchain Consensus Methods

Jeff Nijsse * and Alan Litchfield

Service and Cloud Computing Research Lab, Auckland University of Technology; Auckland 1010, New Zealand; alan.litchfield@aut.ac.nz

* Correspondence: jeff.nijsse@aut.ac.nz

Received: 27 September 2020; Accepted: 16 November 2020; Published: 19 November 2020

Abstract: For a blockchain, consensus is the foundation protocol that enables cryptocurrencies such as Bitcoin to maintain state. Additionally, to ensure safety and liveness for a publicly accessible and verifiable ledger, fault tolerance must be robust. However, there appears to be a degree of misunderstanding about how consensus is applied across blockchains. To assist researchers considering variations between them, this study presents a rational classification of consensus methods applied to current blockchains. The study provides a survey of 19 methods classified by the scarce resource they employ: clock-cycles, bits, tokens, votes, time, and biometrics. Blockchain implementations are split between consensus algorithms requiring proof of resource and those that use majority voting to update the ledger.

Keywords: blockchain; consensus; cryptocurrency; taxonomy; Bitcoin; Ethereum; decentralised computing

---

1. Introduction

The emergence of new forms of computing technology tend to follow a fairly predictable cycle. From its initial appearance, if a new technological advancement solves a problem in a unique manner it will capture the attention of the community. Adoption and adaptation will follow and users will define the scope as it manifests and stabilises into a reliable technology. Those that have been involved with computing will have seen a number of significant technological cycles; the emergence of relational databases, the internet, and cloud computing, to name a few. Since 2009, with the arrival of Bitcoin [1], blockchain technology has fit this cycle, borrowing from mature technologies, such as database systems, distributed computing, decentralised systems, cryptography, game theory, and economics.

A blockchain is a data structure representation of an append-only ordered list that usually comes with some additional properties, such as immutability, transparency, censorship resistance, and decentralisation. Every node in the network maintains a copy of the ledger to verify transactions. Security is upheld by the peer-to-peer (p2p) network through incentivising participants to play by the rules. At the heart of a blockchain lies the consensus method, which is a protocol for maintaining the state of the ledger. The role of a consensus method in a blockchain context is two-fold: order the transactions in the ledger, and prevent double-spending. This second role—prevent double-spending—is what makes Bitcoin unique; it was the first implementation for digital cash that did not need a trusted authority to mediate disputes.

In this paper, a review and classification of a fundamental component of a blockchain—consensus methods—is presented. As expected with a new technology cycle, the incidence of peer reviewed published material was initially limited but in the years 2014–2020 the number of articles published about blockchains, cryptocurrencies, and related applications increased rapidly. The rise in blockchain popularity has meant more networks, nodes, and transactions, but has uncovered performance limitations. For example, the scaling of blockchains to accommodate more users and transactions is closely tied to the consensus mechanism built into the protocol [2]. The analysis of consensus mechanisms and the development of a taxonomy created thematic groups to identify opportunities for research. Blockchain research is a growing field and the effort in this study has been to accurately organise consensus algorithms that have emerged in the decade since Bitcoin.

The paper is organised as follows, in Section 2, consensus is introduced in both a centralised and blockchain context. In Section 3 the methodology applied to the development of a taxonomy is presented. In Section 4, the taxonomy of blockchain consensus methods is presented followed by case-studies in Section 5. Themes are discussed in Section 6 with the conclusion in Section 7.

2. Consensus

Many blockchains operate as a distributed computing system, often with orders of magnitude more nodes than their traditional counterpart. How these nodes communicate with one another and maintain truth in the presence of faults is known as consensus. First, consensus relevant to distributed computing is described, followed by the present-day case of blockchains.

One strategy to handle updating the state involves replicating the database across multiple instances and comparing the information. The state of the system is a sequential list of commands that is replicated between all nodes in the network. Coming to consensus on the state means every node executes the linearly ordered log arriving at the same result. A consensus algorithm must maintain consistent copies of the state across all nodes and process updates proposed by any of those nodes. Figure 1 illustrates consensus as a state diagram that seeks to identify who is proposing an update and how agreement is to be achieved.

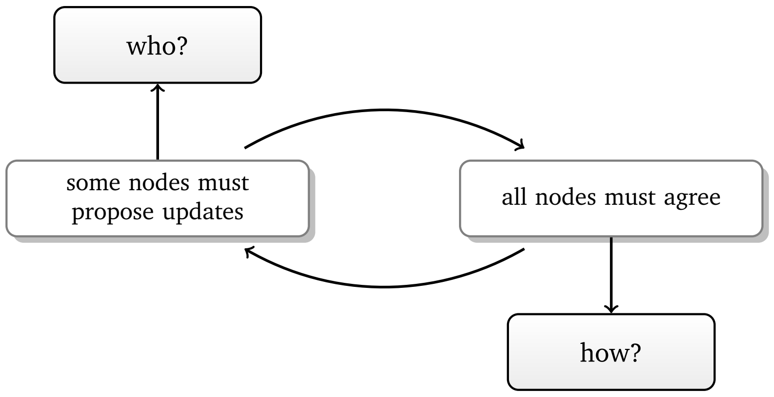

Figure 1. A state diagram for a distributed computing system including a blockchain. A consensus algorithm answers the questions of how? and who?

The difficulty comes from broadcasting local write operations on potentially slow and unreliable networks in a reliable manner [3]. There are two properties that must hold for consensus:

Safety means that two processors will agree on the same value, in addition to the value having been proposed by one of the processors. Agreement is sometimes referred to as consistency, and a processor proposing a value as validity. Valid means the processor decided on a value and proposed it, although it does not necessarily need to be correct.

Liveness is the property that any sent message is eventually delivered. A system that has stalled waiting for a message is "dead" because it cannot make progress. This can be thought of as "eventually something good happens", where *eventually* is loosely defined as finite [4].

Indicative of a traditional approach, it is assumed in the description above that all nodes are honest and trustworthy, regardless of delays or failures. Any commands that are delivered must be both appended to the log and committed so that if a node crashes, the remaining nodes will maintain the log until it recovers. Examples of the application of crash-fault tolerance include two-phase commit (2PC) [5], Viewstamped Replication [6], and Paxos [7]. Furthermore, derivatives of Paxos include multi-Paxos and fast-Paxos. These can all survive c crash-faults given $2c + 1$ nodes. Both Google's BigTable and Amazon's DynamoDB have been developed based on Paxos.

Crash-fault tolerance does not account for the possibility of nodes behaving arbitrarily. A dishonest or malicious node attempting to have a false value committed to the ledger is exhibiting Byzantine behaviour [8]. To be Byzantine-fault tolerance (BFT), given f faulty processors, the number of honest processors must be $n > 3f + 1$ [9]. Agreement among the n processors is impossible without some timing assumption, such as a message timeout [10].

As the name suggests, a popular BFT implementation is Practical Byzantine Fault Tolerance (PBFT) which guarantees safety and liveness for up to f Byzantine nodes of $3f + 1$ replicas [11]. PBFT is partially-synchronous, meaning there is essentially no bound on when messages may be delivered [12]. PBFT has been implemented on fault-tolerant distributed storage, certificate authorities, secure multi-party computation, and reputedly in the Linux-based systems SpaceX used to dock the Dragon capsule with the International Space Station.

In an open p2p network, every participant exists at an equivalent hierarchical level. This affects the way that agreement is achieved and adds complexity because there is no restriction on who may join or leave the network. The decentralised public network may have a large number of nodes which increases communication complexity, latency, and the attack surface. In this regime there is no upper bound on the time to update state and so consensus cannot be guaranteed deterministically. The question of who gets to propose a block is crucial because there are no trusted identities and a Byzantine actor could attempt to direct consensus.

Blockchain consensus was not possible until the concept of Proof of Work (PoW) where the process of proposing updates by finding cryptographic hashes was applied to a p2p network [1]. The approach was originally conceived as a puzzle designed to limit the spread of email spam by having the user expend some computational work to find a partial collision in a hash function [13,14]. In PoW, the winner of the race condition can append the next block to the chain and is rewarded for doing so. The nodes reach agreement by building on the chain representing the greatest proof of computational work, and should someone attempt to build consensus on a shorter or forked chain they will not be eligible for the block reward, having spent computational clock-cycles in the process. Myriad answers to the questions in Figure 1 have resulted in the variety of methods surveyed here.

Maintaining consensus is vital to a blockchain network. In the short-term, disagreement between nodes will result in a fork of the chain where both branches are valid [15]. Eventually, one branch will become longer and therefore have a higher chance of earning the block reward, resolving the fork. If the protocol has a design flaw, or misaligned incentives, then the chain may continue to fork without reaching consensus. In this worst-case scenario the users will exit the chain, possibly losing any value associated with it, and the chain will stagnate.

3. Methodology and Initial Results

Creating a taxonomy is a multi-stage process. First the literature is collected and assessed in a process similar to a literature review. Present taxonomies in the field of blockchain consensus are identified to ensure a meaningful contribution can be made to the field. The final stage applies the method by Nickerson et al. [16] for developing the taxonomy. This is a well-known method within information systems research for designing and refining a taxonomy [17].

To identify the candidate blockchain consensus methods the AIS, IEEE, and ACM databases were searched for published papers. The search included studies that have been reviewed in a secondary source, plus `blockchain AND consensus AND/OR decentralised AND/OR database AND/OR system`. From the found set, the following exclusion criteria were applied:

- Papers focussed on blockchain applications including, but not limited to finance and internet of things (IoT) as these did not analyse methods themselves, rather their applicability to the chosen application;
- Papers that did not address blockchain consensus or were inconsistent;

- Papers proposing new algorithms (primary research).

Nine surveys have been used as a basis for the taxonomy. Some of the general themes on how classification schemes may be derived emerge in the papers: the grouping of consensus methods as proof-of-any-type method (PoX) versus any other concept such as virtual methods and hybrid models [2,18]; a framework for blockchain consensus consisting of information propagation, incentivisation, block proposal, validation, and finalisation [19], where proposal, validation, and finalisation are the steps for consensus but incentivisation is not relevant to the consensus algorithm; comparing the functionalities provided in private blockchains [20] or by comparing fault models [21]; other means for classification include topics criticising blockchains, such as energy usage [22]; identifying differences between specific blockchains rather than protocols [23–25].

The selected surveys provide 69 consensus methods as empirical data points in the taxonomy. Table 1 lists the blockchains by name (e.g., Hyperledger); however, this could also refer to the protocol maintaining consensus (e.g., BFT SMaRt), or the name of the company (e.g., Ripple). The lack of a naming convention is confusing for researchers looking for a concise summary—for example, Hyperledger is mentioned in six of the surveys but it is not clear what variant is referred to. Hyperledger is an umbrella suite of products and offers support for different consensus mechanisms, including BFT SMaRt and PBFT (Fabric), PoW and PoAuthority (Besu), PoET and PBFT (Sawtooth). The table is sorted by the consensus family that the method is derived from—for example, PoW—is the only method mentioned in every survey but the original version used in Bitcoin is just one of many alternatives. The acronyms used in the table are, in order of appearance:

Po	Proof-of—	PoET	Proof of Elapsed Time
PoW	Proof-of-Work	BFT	Byzantine Fault Tolerant
PoS	Proof-of-Stake	SMaRt	State Machine Replication
PBFT	Practical Byzantine Fault Tolerant	DAG	Directed Acyclic Graph
FBA	Federated Byzantine Agreement	2PC	Two-Phase Commit
DPoS	Delegated Proof-of-Stake		

Table 1. Blockchain projects surveyed and sorted according to the consensus family the protocol was derived from.

Name	**Consensus Family**	[21]	[2]	[20]	[22]	[23]	[24]	[25]	[18]	[19]
PoW (Bitcoin)	PoW	•	•	•	•	•	•	•	•	•
Bitcoin NG	PoW		•						•	•
PoBurn	PoW				•				•	
Decor + hop	PoW		•							
Ghost	PoW		•							
Scratch-off puzzles	PoW								•	
PoParticipation and Fees	PoW						•			
Spectre	PoW		•							
PoPublication	PoW				•					
PeerCensus	PoW [1]								•	•
ColorCoin	PoW [1]							•		
Counterparty	PoW [1]							•		
Hyperspace (prev. Synereo)	PoW [1]							•		
Multichain	PoW [1]	•								
NameCoin	PoW [1]							•		
Omni Layer (prev. MasterCoin)	PoW [1]							•		
Po eXercise	PoW [2]					•			•	
PoUseful Work	PoW [2]								•	
PoStake	PoS		•	•	•	•	•	•	•	•
Algorand	PoS		•						•	•

Table 1. *Cont.*

Name	Consensus Family	[21]	[2]	[20]	[22]	[23]	[24]	[25]	[18]	[19]
Ouroboros (+ Praos)	PoS		•			•				•
PoActivity	PoS				•				•	•
Snow White	PoS		•						•	•
Casper	PoS								•	•
NXT (Ardor)	PoS							•		•
Chain of Activity	PoS									•
PoStake Velocity	PoS								•	
Hyperledger	PBFT	•	•	•	•			•		•
Implicit Consensus	PBFT						•			
Iroha	PBFT ^{3}	•								
Kadena (Juno)	PBFT ^{4}	•		•						
Honeybadger	PBFT ^{5}	•	•							•
Hybrid Consensus	PBFT + PoW								•	
Omni Ledger	PBFT + PoW		•							
ByzCoin	PBFT + PoW ^{6}		•							•
Solidus	PBFT + PoW ^{6}		•							
Elastico	PBFT ^{7}		•				•			
Chainspace	PBFT ^{8}		•							
Ripple	FBA	•		•	•	•	•	•		•
Stellar	FBA	•				•		•		•
Chain	FBA	•								
DelegatedPoStake	DPoS				•	•		•	•	•
PoAuthority	DPoS			•					•	•
PoImportance	DPoS				•	•				
Dfinity	DPoS			•						
PoVote	DPoS						•			
Sawtooth Lake	PoET	•	•	•	•				•	•
PoLuck	PoET					•			•	
Resource Efficient Mining	PoET		•						•	
PoOwnership	PoET								•	
Raft	Raft	•		•						
PoTrust	Raft						•			
Quorum (JPMorgan)	Raft ^{9}	•								
Tendermint	BFT	•		•	•			•	•	•
Cosmos	BFT							•		
Corda (& Enterprise)	BFT SMaRt + Raft	•						•		
BFT SMaRt	BFT SMaRt	•								•
Symbiont Assembly	BFT SMaRt	•								
PoCapacity	PoCapacity		•		•			•		•
IOTA	Hash DAG	•								•
Hashgraph (Swirlds)	Hash DAG	•								
Paxos	Paxos	•								
PoHumanWork	PoBiometrics								•	
PoMemory (memory-hard)	PoMemory								•	
Ethash	PoMemory								•	
PoSpace	PoSpace		•		•			•	•	•
Filecoin	PoSpaceTime								•	
Peercoin	PoS + Coin Age							•		•
RSCoin	2PC		•							

^{1} based on Bitcoin, ^{2} useful, ^{3} based on Bchain, ^{4} Scalable BFT, ^{5} asynchronous, ^{6} with a committee, ^{7} with sharding, ^{8} flexible, ^{9} Istanbul BFT.

A review of papers that present taxonomies of blockchain consensus protocols shows some common trends. The approaches reviewed are thorough but including everything into a taxonomy, as most try to do, is ultimately unwieldy and fails to include additional elements that may be blockchain-specific but that provide nuance. With the exception of one that categorises blockchain applications rather than the protocol to maintain the ledger [26], most include consensus as just one category. For example, providing a taxonomy from a systems-architecture viewpoint where consensus is one dimension among others, so a systems designer can choose an appropriate blockchain style [27]. Two taxonomies attempt to categorise all the components of a blockchain [25,28] and consensus is included as a dimension among others, like the open-source nature of the codebase and the financial classification of the token. A comprehensive survey is found in [18] but offers no formal classification and has some confusing categories. The authors delineate consensus into four categories—permissionless consensus, PoW-style, proof-of-resources, and proof-of-concept for performance improvement—while categorizing PoS separately. The last survey provides a first-principles approach by creating a classification tree structure with the underlying network assumptions at the root so that point-to-point methods among authenticated channels are separate from the p2p network [29]. In this case, the taxonomy tree illustrates that consensus is branched with public/private setup, computational assumptions, and communication cost. So while examples of PoS in the p2p setting are provided, a fine-grained distinction between blockchain protocols is not present.

In general terms, a taxonomy is an attempt to gain understanding of a group of instances by grouping and categorising apparent features or characteristics and creating an abstract model that may be used to compare with newly found instances. The characteristics, when categorised, allow for the identification of meta-characteristics from which the model is comprised. A new instance, when compared to existing instances, may then be considered in respect of what is already known and a taxonomy may be refined or developed as new knowledge is accrued. Thus, taxonomies tend to emerge once a researcher has sufficient experience with a field to see subtle distinctions between characteristics.

To develop the taxonomy, an iterative three-step approach has been applied that is based on design science [16]. The apparent characteristics identified for the development of this taxonomy reflect the understanding that blockchain consensus is about maintaining the state of a ledger that has been replicated across many nodes in a decentralised system. Implementation specifics are not included in the taxonomy.

Step 1 Derive the principal meta-characteristic for categorisation; the methodology for maintaining a distributed ledger at a high-level. Characteristics, similar to deterministic finality, or committee-based voting, are included in this category.

Step 2 An inductive approach that determines which characteristics may be grouped into dimensions empirically by identifying characteristics within the meta-characteristic and subsequently selecting dimensions to group them together [30]. The conceptual-to-empirical approach provides the opportunity to hypothesise about new dimensions that can be tested with data.

Step 3 Checks whether there are characteristics left unresolved or uncategorised and iterates if necessary, back to either an empirical-to-conceptual or conceptual-to-empirical strategy.

4. Taxonomy

In this section, the taxonomy is described along with the process for building it.

4.1. Derivation of Categories

As described in the previous section, a three-step process is applied to derive the taxonomy where the meta-characteristic is defined as a high-level methodology for maintaining state of a distributed ledger. Table 1 lists consensus categories that provide an empirical starting point for the taxonomy. The blockchains in Table 1 are sorted by the number of occurrences in literature to derive a prominence ranking (Table 2). While the number of mentions is a crude means for deriving prominence, it does afford some degree of conscious acknowledgement of importance in the community. Nineteen consensus categories are identified and provide a basis for the first iteration. The taxonomy incorporates the methods from Table 2.

Table 2. Consensus methods ranked by occurrence in literature. For the acronyms refer to the list above Table 1.

Consensus Method	Occurrences	(Continued)	
PoW	33	BFT SMaRt	4
PoS	27	PoCapacity	4
PBFT	20	Hash DAG	3
FBA	13	Honeybadger	3
DPoS	12	PoCoinAge	2
PoET	11	PoMemory	2
Raft	8	2PC	1
Tendermint (BFT)	7	Paxos	1
PoSpace	5	PoBiometrics	1
		PoSpaceTime	1

As a feature, PoW was the key to solving the double-spending problem for a cryptocurrency by incentivising the longest-chain rule. Thus, this is the first dimension in the taxonomy and represents a finite resource (that is, a limited resource from which value may be apportioned). Characteristics of this dimension can be determined directly from Table 2, for example: the stake in PoS represents the proportion of the total tokens that a node has dedicated to consensus, therefore *tokens* is the characteristic.

At least one iteration is required when a new dimension is identified. The approach applied is to begin with a concept and drill down to derive empirical outcomes. One survey begins with a distinction made between networks that operate in an open p2p manner and those that require trusted channels [29]. Thus, a dimension for *network communication* is included with the characteristics of *trusted channels* and *p2p*.

4.2. Derivation of Dimensions

Seven dimensions are derived by their characteristics and are listed in the taxonomy of blockchain consensus methods (Table 3). The dimensions are scarce resource, fault tolerance, block proposal, transaction finality, network timing, network accessibility, and network communication, as defined below.

Scarce Resource A resource that is difficult to replace, access, or create and thus scarce, accrues value as it is consumed or exploited. The potential for the growth of value is a motivation that encourages acceptable behaviour in the emergence of consensus. Six types of resource are identified.

1. *Clock-cycles* have non-zero cost, meaning the computational work done to find a hash value that is subject to a target requirement. The computational work cannot be recovered or undone. A cryptographic requirement is that the hash function is one-way, so the originating data cannot be inferred, but is easily verifiable.
2. *Tokens* held by a user may be committed to maintaining a protocol. PoS allocates users a stake in the system proportionate to their number of tokens. In general, tokens that are held as stake cannot be used in transactions. Honesty is encouraged by both advantaging nodes that validate transactions and create blocks at the risk of forfeiting their stake (slashing) for dishonest behaviour. A variation, DPoS, hands block production responsibility to a set of validators, or delegates, which are a subset of the network. Often the validators must be known to the network and thus subject to ridicule if they behave maliciously as in proof-of-authority.
3. *Votes* are used to determine a majority and is a common method for gaining consensus with nearly half of the methods in Table 3 employing votes as the scarce resource. A BFT system must determine consensus by the replicas voting on the state. Votes have no tradable value and generally a node is permitted one vote per round, although an additional vote may be permitted in each subsequent round. Classical consensus methods that maintain state in a non-blockchain system, such as PBFT, use voting or a round-robin style to elect leaders.

4. *Bits* represent the state of a transistor and occupy a finite amount of space in storage. PoCapacity methods allocate a user some stake in the system by requiring that a proportion of storage is kept aside. If the computer is a blockchain node, the allocation cannot be used for other purposes. PoMemory requires access to volatile storage; Ethash (Ethereum) commits a pseudorandom dataset to a DAG which grows linearly resulting in access that is limited by the memory bandwidth [18]. Storage volumes tend to scale more slowly than computer processor clock-cycles and can reduce the domination of performance-enhanced processor architectures, such as Application Specific Integrated Circuits (ASICs).
5. *Time* is independent of computing advances. As clock-cycles and read/write times reduce, blockchains secured by these resources may be exposed to unforeseen factors. Thus, PoET processors provide additional execution environments or enclaves, that cannot be accessed by the system. These modules can return a random delay to a process that can then be used to assign block proposers.
6. *Biometrics* are a range of indicators that can verify identity or life. Similar to a PoW hash function requiring a known average number of clock-cycles, a blockchain based on proof-of-biometrics can require a unique biological solution. While biometrics are typically understood to be consistent parameters, such as facial, iris, voice, and fingerprint recognition systems, other systems may include affordances available only to humans or to specific humans. To be successful, such a system must be easy to use, require minimal input, and validate or authenticate rapidly. For example, the Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA) system is easy for humans to solve but difficult for a computer.

Fault Tolerance At the base level, fault tolerance refers to crash-fault tolerance where a node can fail and will resume operation once it has been brought back online. These nodes cannot exhibit arbitrary behaviour, such as sending faulty information and so if a single node becomes compromised, the entire system is compromised. Consensus requires a majority of nodes—for example, with 2PC up to c nodes can crash, requiring $2c + 1$ replicas. Systems, such as Google's Bigtable, are guaranteed five replicas and can tolerate two failures. If the replicas can be subverted and send incorrect information the system must be Byzantine-fault tolerant. PBFT and its derivatives can handle up to f Byzantine faults of $3f + 1$ replicas.

Above, f and c are integers; whereas for a proofing type method, fault tolerance is the percentage of total resource that may be sacrificed before consensus is lost. P2p systems assume bad actors will attempt to subvert the network, possibly colluding with each other, and may only be held off for as long as there is an honest majority. A 51% attack occurs when adversaries obtain >50% control of the scarce resource and can then alter, control, predetermine, or direct the consensus process. The selfish mining strategies of [31] have shown that an actor with <50% can withhold blocks and earn more of the reward; however, this does not affect liveness or safety.

Block Proposal The question of who gets to propose new blocks (Figure 1) is fundamental to consensus. Selecting a validator must be fair and secure. A decentralised open system allows any participant the opportunity to propose blocks and a fair way to accomplish this is by random selection. If peers do not find out who proposed the block until after it is proposed, this is a leader-free scenario. Leader election can also be accomplished by using randomized processes at which point the lead replica is responsible for coordinating the subsequent update. These systems are usually private as replicas require known IDs. A committee-based system relies on a predetermined set of validators to be responsible for updates. Participants may join the network but not necessarily be part of the committee.

The exception occurs in the case of the DAG, in which no node is responsible or chosen for updates and there is no block proposer. A DAG links transactions in a similar manner to a blockchain with the exception that a graph node can have n outgoing verticies. A traditional family tree satisfies this condition; however, so does a PoW chain in the presence of n forks (not necessarily in consensus). With IOTA, the graph breadth continues to grow.

Table 3. Taxonomy of blockchain consensus methods

	Scarce Resource												
Consensus Method	bits	time	clock-cycles	tokens	biology	votes	**Fault Tolerance**	**Transaction Finality**	**Network Timing**	**Block Proposal**	**Network Accessibility**	**Network Comm.**	**Example Blockchain**
PBFT						•	3f+1	D	partial	election	public/private	trusted	Hyperledger Sawtooth
BFT-SMaRt						•	3f+1	D	partial	election	private	trusted	R3 Corda
Federated BA						•	5f+1 to 3f+1	D	partial	committee	consortium	both	Stellar, Ripple
Honeybadger						•	3f+1	P	async.	random	public/private	both	POA Network
2PC						•	2c+1	D	async.	none	private	trusted	RSCoin
Paxos						•	2c+1	D	async.	election	private	trusted	none
Raft						•	2c+1	D	async.	election	private	trusted	Quorum
Hashgraph						•	3f+1	P	async.	none	private	p2p	Hashgraph, IOTA
DPoS				•		•	3f+1 to 2f+1	P	synch.	committee	all	trusted	Steemit, EOS, Bitshares
PoS				•			3f+1	P	s,p	committee	all	trusted	Ethereum, Decred
Tendermint				•			3f+1	D	partial	election	private	both	Cosmos
PoW			•				50%	P	synch.	random	public	p2p	Bitcoin
PoCoinAge		•		•			50%	P	synch.	random	public	p2p	Peercoin/Ppcoin
PoElapsedTime		•					2c+1	P	synch.	random	public	trusted	Hyperledger Sawtooth
PoRetrievability	•						50%	P	synch.	random	public	p2p	Permacoin (PoR)
PoSpace	•						50%	P	synch.	random	public	p2p	Spacemint
PoSpaceTime	•	•					50%	P	synch.	random	public	p2p	Filecoin
PoMemory	•		•				50%	P	synch.	random	public	p2p	Zcash, Beam
PoHumanWork		•			•		-	-	-	-	-	-	none

Transaction Finality There are two approaches to determining a transaction is complete. The first is deterministic and guarantees the data are written and committed to the blockchain as soon as the block is posted. The second is probabilistic, where a transaction is confirmed with increasing probability as more blocks are added—e.g., with PoW. There is a chance that transactions are added to a fork of the chain that does not represent the most computational work. Over time the longest chain will emerge and forked blocks become orphaned. Bitcoin's proof-of-work is often called *Nakamoto* or *emergent* consensus because the more chain work that is done after a transaction is in a block, the more likely it is to be committed to the ledger. PoS systems, such as Ethereum's Casper [32] have checkpoints before which all data finality is probabilistic. After a checkpoint is reached, the transactions are finalised.
Network Timing The timing considerations may be synchronous, asynchronous, or partially synchronous. Networks such as PoW and proof-of-capacity are generally considered synchronous because they are guaranteed to update the state upon completion of every round. What is not guaranteed is that every round will complete because there are no timing assumptions; the protocol will run as long as necessary to append a block. Recently it has been shown that Nakamoto consensus, operating in a dynamic participant pool maintains safety and liveness in bounded-delay networks [33]. A fully asynchronous network has no known upper bound for message delivery. A limitation of 2PC is that it will stall if a message is delayed for an arbitrarily long amount of time [10]. A partially synchronous model may employ timeouts, rules, learning algorithms, and predetermined hierarchies to resolve deadlocks.
Network Accessibility Categories for blockchain network access are public, private or a combination of the two. In a public network, anyone can join the network and participate, then leave the network without penalty. Most decentralised blockchains are public and so they need to be secured against Byzantine behaviour. Private blockchains require nodes to be validated and external participants may not be able to participate or view activity. Enterprise blockchains are typically private for a range of reasons, including efficiency and security. A consortium is comprised of a group of parties, such as financial institutions that share access to a network. The reasons for establishing a consortium vary and include common goals, sharing of resources, mutual agreement on consensus methods, development opportunities, and so on. Individual consensus methods, such as PoS, may have instantiations of different access types—for example PoS may be public as in Decred, or consortium as in EOS. Any public PoS system can be adapted for private use.
Network Communication Nodes exchange transaction or block data via a range of network communication methods. A trusted setup requires nodes to validate each other through a key-exchange procedure or similar. Open p2p networks communicate via a gossip protocol flooding nearest-neighbours with information until all nodes are in agreement. Network Communication could be both point-to-point and peer-to-peer—for example, trusted channels can be built on top of a peer-to-peer network [34].

In a few cases, the characteristics that identify methods in Table 3 are not exclusive but lead to a range of blockchain implementations. While there may be an argument for additional dimensions (such as performance, scalability, message complexity, etc.), the contribution is not sufficient to warrant their inclusion here. Performance criteria apply to specific cases: the Example Blockchains in Table 3, or the Validation Cases in Table 4; thus they are characteristics not only of the protocol but the implementation. For example, transactions per second (TPS) is often cited as a performance metric [2,19]; Bitcoin and Litecoin both use PoW but utilise different hashing algorithms and due to design factors such as blocktime Bitcoin can handle $\approx$7 TPS, and Litecoin $\approx$56 TPS. These variations have not been included in Table 3 to keep the focus on consensus.

Table 4. Taxonomy validation results.

Case	SR [1]	FT [2]	BP [3]	TF [4]	NT [5]	NA [6]	NC[7]
Avalanche	votes	$1-\epsilon$	none	probabalistic	S [9]	public	p2p
Thunderella	cc[8], votes	25%–50%	election	deterministic	A [10], S	public, private	p2p, trusted
LibraBFT	votes	$3f+1$	committee	deterministic	P [11]	consortium	trusted
Gemini	cc	50%	random	probabalistic	S	public	trusted

[1] Scarce Resource, [2] Fault Tolerance, [3] Block Proposal, [4] Transaction Finality, [5] Network Timing, [6] Network Accessibility, [7] Network Communication, [8] Clock-cycles, [9] Synchronous, [10] Asynchronous, [11] Partially synchronous.

5. Taxonomy Evaluation

To evaluate the taxonomy, the dimensions should encapsulate novel methods. Four cases are considered and applied to the taxonomy: Thunderella, Avalanche, LibraBFT, and Gemini. The cases are selected because they are sufficiently different from each other that if there emerged issues with the taxonomy, they would arise during the evaluation. Additionally, none of the cases were used in the development and refinement of the taxonomy. The evaluation assesses the cases against the taxonomy to determine whether they are well associated and if there are features in the cases that are not addressed by the taxonomy. Table 4 provides the results of the evaluation and the findings suggest that the taxonomy applies to a range blockchain types.

Avalanche [35] uses a probabilistic safety guarantee similar to PoW but without the resource intensive dependence on PoW. The probability that safety will be upheld is set by a parameter, ϵ, and determined by the designer. Avalanche uses an append-only DAG to maintain the public ledger.

Thunderella [36] is a hybrid of classical asynchronous consensus methods with synchronous blockchain methods. To increase throughput, Thunderella seeks to improve the performance of PoW chains by using an optimistic path that can tolerate 25% Byzantine nodes in an asynchronous environment. In the event that this fails, it and can fallback to 50% BFT in a synchronous environment. A committee is selected for leader election progressing in rounds in a permissioned environment or by a PoW oracle in a permissionless environment.

LibraBFT [37], otherwise known as the Libra blockchain, applies BFT consensus and is partially synchronous. Developed to be used on the Facebook platform, Libra is intended to operate as a global payments system. Ledger state is maintained in a PoS style by trusted validators that can tolerate up to one-third faulty nodes.

The Gemini dollar [38] is a cryptographic token pegged to the US dollar. Known as a stable-coin, they can be used in confidence by merchants without the volatility of a cryptocurrency. The Gemini dollar is an application built on the Ethereum platform and therefore must adhere to the rules of the Ethereum consensus mechanism.

6. Discussion

This taxonomy applies to researchers and analysts investigating the current state in blockchain consensus to classify various types and variations. Researchers may be looking to identify areas for future development or optimisation while analysts could be looking for a starting point in adopting a consensus method to use or to base a blockchain design on. While this classification system is derived from existing sources and represents a current review of types, it may also be subject to change and update as new methods or approaches emerge.

Three categories are identified in Table 3: (1) blockchains based on traditional consensus and dependent on replica voting; (2) the need to demonstrate that some resource has been consumed, exploited, or set aside; (3) dependency on tokenised representation. What is apparent is that all the methods have valid uses but there is no ideal method. Furthermore, to accommodate more users and activity, much research is focussed on the use of distributed systems as a framework or means for scaling protocols.

PBFT, BFT SMaRt, and FBA are grouped together and have well known blockchains. Depending on client need, Hyperledger incorporates a number of consensus algorithms. R3 Corda can use BFT SMaRt, which is an optimised BFT algorithm that can achieve a high throughput [39] while Hyperledger Sawtooth has a PBFT implementation. FBA has a federation of permissioned validators whereby each validator determines its own set of nodes to trust for consensus. The pool of validators in Ripple is called a unique node list and in Stellar a quorum slice.

Paxos, 2PC, and Raft are categorised together and are not well represented in this space due to the algorithms being crash-fault (not Byzantine) tolerant, and requiring a network with known participants. Of note, 2PC splits communication into a prepare phase and a commit phase. In the prepare phase nodes are made aware of the state update and the central coordinator tallies the votes. The commit phase involves another round trip whereby each node updates their ledger. Paxos itself has no known blockchain implementations and is included in the taxonomy as a reference point. Raft is found in Quorum by JPMorgan and is an exception.

PoCapacity schemes that monopolise disk storage are interesting because of their resource efficiency versus PoW and their application to distributed storage. An alternative seen in proof-of-retrievability requires that a participant verify they have stored a portion of a file for later use. This example could be applied to a large public dataset where storage provides a distributed archive, such as Permacoin. Alternatively, proof of space as proposed for Spacemint seeks to apply disk space in a mining-like capacity where the user dedicates a portion of disk space to a large file and can verify that they have done the computational work to pebble a DAG from the file [40]. Lastly, Filecoin uses a time dimension in their proof of spacetime consensus method by taking into account an amortisation period. The prover must show a recursive proof of retrievability to show they have maintained the storage for a period of time [41].

The use of biometrics in providing evidence of a scarce resource holds promise because individual people are unique and that combined with time implies that a transaction cannot be replicated or accelerated by computing advances. However, time is difficult to verify computationally due to advances in hardware and parallel processing. PoET uses special enclaves in a chip architecture called trusted execution environments such as Intel's Software Guard eXtension and ARM's TrustZone. Hyperledger Sawtooth has a PoET implementation. Network accessibility in Table 3 is listed as public, but a node is required to have the specific hardware module to participate. A PoW blockchain can act as a proxy for proof of ”time” as it is a linked list in the manner of secure time stamping although a good PoET implementation can be more energy efficient.

7. Conclusions

The paper presents a taxonomy in which blockchains are categorised by consensus family across seven dimensions: scarce resource, fault tolerance, block proposal mechanism, transaction finality, network timing assumptions, network accessibility, and network communication. Much of the literature reviewed did not meet the conditions for quality for inclusion or it had been influenced by corporate marketing. The taxonomy provides a robust contribution to the field that includes 69 blockchains that are presented in peer reviewed literature. While the taxonomy offers a high level explanatory view, it is concise because it is limited to seven dimensions and concentrates on the meta-characteristic of maintaining the state of a distributed ledger. Lastly, to accommodate future algorithms, the taxonomy is extensible and to demonstrate this, as well as to evaluate the taxonomy on cases that were not involved in its development, four case studies are applied.

Limitations that are present in the taxonomy are that it is a snapshot of the present state of consensus and while blockchain research is expanding, blockchain variants are proposed faster than they appear in academic sources. Examples of blockchain implementations are given for reference; however, this is not a complete listing nor does the taxonomy classify blockchains.

Opportunities for development and research present themselves as further in-depth analyses. For example, a large number of blockchains use some form of proofing method to attempt to publicly verify that a scarce resource is secured but others utilise BFT methods from distributed computing. At this point, there is no clear future direction where consensus will be focussed. In the meantime, other methods such as DAG with a gossip protocol or asynchronous BFT have yet to be tested at scale.

Additionally, biometrics and time exploit opportunities sociotechnical systems and offer interesting areas for future development. For example, social status or reputation provide a strong incentive to maintain integrity in a network by providing increased social scores for good behaviours and attempts to subvert are be negatively reinforced by punishing a social score that took time to accrue.

Author Contributions: J.N. contributed the taxonomy under the supervision and guidance of A.L. who also submitted and prepared the draft submission of this article. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

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

References

1. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. Available online: https://bitcoin.org/bitcoin.pdf (accessed on 6 May 2020).
2. Bano, S.; Sonnino, A.; Al-Bassam, M.; Azouvi, S.; McCorry, P.; Meiklejohn, S.; Danezis, G. SoK: Consensus in the Age of Blockchains. Available online: https://arxiv.org/pdf/1711.03936.pdf (accessed on 6 May 2020).
3. Lindsay, B.G.; Selinger, P.G.; Galtieri, C.; Gray, J.N.; Lorie, R.; Price, T.G.; Putzolu, F.; Traiger, I.L.; Wade, B.W. *Notes on Distributed Databases*; Technical Report RJ2571; IBM: San Jose, CA, USA, 1979.
4. Attiya, H.; Welch, J. *Distributed Computing: Fundamentals, Simulations and Advanced Topics*, 2nd ed.; John Wiley & Sons, Inc.: New York, NY, USA, 2004; doi:10.1002/0471478210. [CrossRef]
5. Gray, J.N. Notes on Data Base Operating Systems. In *Operating Systems: An Advanced Course*; Springer: Berlin/Heidelberg, Germany, 1978; pp. 393–481, doi:10.1007/3-540-08755-9_9. [CrossRef]
6. Oki, B.M.; Liskov, B.H. Viewstamped Replication: A New Primary Copy Method to Support Highly-Available Distributed Systems. In Proceedings of the Annual ACM Symposium on Principles of Distributed Computing, Toronto, ON, Canada, 15–17 August 1988; pp. 8–17, doi:10.1145/62546.62549. [CrossRef]
7. Lamport, L. The part-time parliament. *ACM Trans. Comput. Syst.* **1998**, *16*, 133–169, doi:10.1145/279227.279229. [CrossRef]
8. Lamport, L.; Shostak, R.; Pease, M. The Byzantine Generals Problem. *ACM Trans. Program. Lang. Syst.* **1982**, *4*, 382–401, doi:10.1145/357172.357176. [CrossRef]
9. Pease, M.; Shostak, R.; Lamport, L. Reaching Agreement in the Presence of Faults. *J. Assoc. Comput. Mach.* **1980**, *27*, 228–234, doi:10.1145/322186.322188. [CrossRef]
10. Fischer, M.J.; Lynch, N.A.; Paterson, M.S. Impossibility of Distributed Consensus with One Faulty Process. *ACM* **1985**, *32*, 374–382. [CrossRef]
11. Castro, M.; Liskov, B. Practical Byzantine Fault Tolerance. In Proceedings of the Third Symposium on Operating Systems Design and Implementation, New Orleans, LA, USA, 22–25 February 1999; Volume 99, pp. 173–186.
12. Cachin, C. State Machine Replication with Byzantine Faults. In *Lecture Notes in Computer Science*; Charron-Bost, B., Pedone, F., Schiper, A., Eds.; Replication; Springer: Heidelberg, Germany, 2010; Volume 5959, doi:10.1007/978-3-642-11294-2_9. [CrossRef]
13. Dwork, C.; Naor, M. Pricing Via Processing or Combatting Junk Mail. In Proceedings of the 12th CRYPTO '92 Annual International Cryptology Conference on Advances in Cryptology, Santa Barbara, CA, USA, 16–20 August 1992; p. 11.
14. Back, A. Hashcash—A Denial of Service Counter-Measure. Available online: http://www.hashcash.org/ (accessed on 6 May 2020).
15. Schär, F. *Blockchain Forks: A Formal Classification Framework and Persistency Analysis*; World Scientific Europe: London, UK, 2020; doi:10.13140/RG.2.2.27038.89928/1. [CrossRef]
16. Nickerson, R.C.; Varshney, U.; Muntermann, J. A Method for Taxonomy Development and Its Application in Information Systems. *Eur. J. Inf. Syst.* **2013**, *22*, 336–359, doi:10.1057/ejis.2012.26. [CrossRef]

17. Szopinski, D.; Schoormann, T.; Kundisch, D. Because Your Taxonomy Is Worth It: Towards a Framework for Taxonomy Evaluation. In Proceedings of the 27th European Conference on Information Systems (ECIS), Uppsala, Sweden, 8–14 June 2019; pp. 1–19.
18. Wang, W.; Hoang, D.T.; Hu, P.; Xiong, Z.; Niyato, D.; Wang, P.; Wen, Y.; Kim, D.I. A Survey on Consensus Mechanisms and Mining Strategy Management in Blockchain Networks. *IEEE Access* **2019**, *7*, 22328–22370, doi:10.1109/ACCESS.2019.2896108. [CrossRef]
19. Xiao, Y.; Zhang, N.; Lou, W.; Hou, Y.T. A Survey of Distributed Consensus Protocols for Blockchain Networks. Available online: http://arxiv.org/abs/1904.04098 (accessed on 6 May 2020).
20. Dinh, T.T.A.; Liu, R.; Zhang, M.; Chen, G.; Ooi, B.C.; Wang, J. Untangling Blockchain: A Data Processing View of Blockchain Systems. *IEEE Trans. Knowl. Data Eng.* **2018**, *30*, 1366–1385, doi:10.1109/TKDE.2017.2781227. [CrossRef]
21. Cachin, C.; Vukolic, M. Blockchain Consensus Protocols in the Wild. Available online: http://arxiv.org/pdf/1707.01873v2.pdf (accessed on 6 May 2020).
22. Chalaemwongwan, N.; Kurutach, W. State of the art and challenges facing consensus protocols on blockchain. In Proceedings of the 2018 IEEE International Conference on Information Networking (ICOIN), Chiang Mai, Thailand, 10–12 January 2018; pp. 957–962, doi:10.1109/ICOIN.2018.8343266. [CrossRef]
23. Bach, L.; Mihaljevic, B.; Zagar, M. Comparative Analysis of Blockchain Consensus Algorithms. In Proceedings of the 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 21–25 May 2018; pp. 1545–1550, doi:10.23919/MIPRO.2018.8400278. [CrossRef]
24. Chaudhry, N.; Yousaf, M.M. Consensus Algorithms in Blockchain: Comparative Analysis, Challenges and Opportunities. In Proceedings of the 2018 12th International Conference on Open Source Systems and Technologies (ICOSST), Lahore, Pakistan, 19–21 December 2018; pp. 54–63, doi:10.1109/ICOSST.2018.8632190. [CrossRef]
25. Tasca, P.; Tessone, C. Taxonomy of Blockchain Technologies. Principles of Identification and Classification. *Ledger* **2019**, *1*, 1–39, doi:10.5195/LEDGER.2019.140. [CrossRef]
26. Glaser, F.; Bezzenberger, L. Beyond Cryptocurrencies—A Taxonomy of Decentralized Consensus Systems. In Proceedings of the 23rd European Conference on Information Systems (ECIS), Münster, Germany, 26–29 May 2015; pp. 1–18, doi:10.18151/7217326. [CrossRef]
27. Xu, X.; Weber, I.; Staples, M.; Zhu, L.; Bosch, J.; Bass, L.; Pautasso, C.; Rimba, P. A Taxonomy of Blockchain-Based Systems for Architecture Design. In Proceedings of the 2017 IEEE International Conference on Software Architecture, ICSA 2017, Gothenburg, Sweden, 3–7 April 2017; pp. 243–252, doi:10.1109/ICSA.2017.33. [CrossRef]
28. Wieninger, S.; Schuh, G.; Fischer, V. Development of a Blockchain Taxonomy. In Proceedings of the 2019 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC), Sophia Antipolis, France, 17–19 June 2019; pp. 1–9, doi:10.1109/ice.2019.8792659. [CrossRef]
29. Garay, J.A.; Kiayias, A. SoK: A Consensus Taxonomy in the Blockchain Era. Available online: https://eprint.iacr.org/2018/754.pdf (accessed on 6 May 2020).
30. Bailey, K.D. Typologies and Taxonomies: An Introduction to Classification Techniques. In *Quantitative Applications in the Social Sciences*; Sage Publications: Newbury Park, CA, USA, 1994; pp. 7–102.
31. Eyal, I.; Sirer, E.G. Majority Is not Enough: Bitcoin Mining Is Vulnerable. *Financ. Cryptogr. Data Secur.* **2014**, 436–454, doi:10.1007/978-3-662-45472-5_28. [CrossRef]
32. Buterin, V.; Griffith, V. Casper the Friendly Finality Gadget. Available online: http://arxiv.org/abs/1710.09437 (accessed on 6 May 2020).
33. Garay, J.; Kiayias, A.; Leonardos, N. Full Analysis of Nakamoto Consensus in Bounded-Delay Networks. https://eprint.iacr.org/2020/277.pdf (accessed on 19 November 2020).
34. Miller, A.; Xia, Y.; Croman, K.; Shi, E.; Song, D. The Honey Badger of BFT Protocols. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security—CCS'16, Vienna, Austria, 24–28 October 2016; ACM Press: New York, NY, USA, 2016; pp. 31–42, doi:10.1145/2976749.2978399. [CrossRef]
35. Rocket, T. Snowflake to Avalanche: A Novel Metastable Consensus Protocol Family for Cryptocurrencies. Available online: https://ipfs.io/ipfs/QmUy4jh5mGNZvLkjies1RWM4YuvJh5o2FYopNPVYwrRVGV (accessed on 25 October 2020).

36. Pass, R.; Shi, E. Thunderella: Blockchains with Optimistic Instant Confirmation. In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 3–33, doi:10.1007/978-3-319-78375-8_1. [CrossRef]
37. Baudet, M.; Ching, A.; Chursin, A.; Danezis, G.; Garillot, F.; Li, Z.; Malkhi, D.; Naor, O.; Perelman, D.; Sonnino, A. State Machine Replication in the Libra Blockchain. Available online: https://developers.libra.org/docs/state-machine-replication-paper (accessed on 6 May 2020).
38. The Gemini Dollar: A Regulated Stable Value Coin. Available online: https://gemini.com/static/dollar/gemini-dollar-whitepaper.pdf (accessed on 6 May 2020).
39. Bessani, A.; Sousa, J.A.J.; Alchieri, E.E.P. State Machine Replication for the Masses with BFT-SMART. In Proceedings of the 2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, Atlanta, GA, USA, 23–26 June 2014; pp. 355–362, doi:10.1109/DSN.2014.43. [CrossRef]
40. Park, S.; Alwen, J.; Fuchsbauer, G.; Gazi, P.; Pietrzak, K. SpaceMint: A Cryptocurrency Based on Proofs of Space. Available online: https://eprint.iacr.org/2015/528.pdf (accessed on 6 May 2020).
41. Benet, J.; Greco, N. Filecoin: A Decentralized Storage Network. Available online: https://filecoin.io/filecoin.pdf (accessed on 6 May 2020).