Consensus Algorithms in Blockchain: Comparative Analysis, Challenges and Opportunities

Conference Paper · December 2018

DOI: 10.1109/ICOSST.2018.8632190

2 authors:

Natalia Chaudhry
University of the Punjab
9 PUBLICATIONS **70** CITATIONS

SEE PROFILE

Muhammad Murtaza Yousaf
University of the Punjab
46 PUBLICATIONS **273** CITATIONS

SEE PROFILE

Consensus Algorithms in Blockchain: Comparative Analysis, Challenges, and Opportunities

Natalia Chaudhry
Punjab University College of Information Technology
University of the Punjab
Lahore, Pakistan
natalia@pucit.edu.pk

Muhammad Murtaza Yousaf
Punjab University College of Information Technology
University of the Punjab
Lahore, Pakistan
murtaza@pucit.edu.pk

***Abstract*— Blockchain is a distributed ledger that gained a prevalent attention in many areas. Many industries have started to implement blockchain solutions for their application and services. It is important to know the key components, functional characteristics, and architecture of blockchain to understand its impact and applicability to various applications. The most well-known use case of blockchain is bitcoin: a cryptocurrency. Being a distributed ledger, consensus mechanism is needed among peer nodes of a blockchain network to ensure its proper working. Many consensus algorithms have been proposed in literature each having its own performance and security characteristics. One consensus algorithm cannot serve the requirements of every application. It is vital to technically compare the available consensus algorithms to highlight their strengths, weaknesses, and use cases. We have identified and discussed parameters related to performance and security of consensus in blockchain. The consensus algorithms are analyzed and compared with respect to these parameters. Research gap regarding designing an efficient consensus algorithm and evaluating existing algorithms is presented. This paper will act as a guide for developers and researchers to evaluate and design a consensus algorithm.**

Keywords—blockchain, bitcoin, distributed ledger, consensus

I. INTRODUCTION

Blockchain technology emerged to overcome the risks and inefficiencies in business transactions. It has revolutionized the structure of industries and businesses [1] [2] [3] [4] [5] [6]. Blockchain can be defined as a distributed ledger, shared among the nodes of a business network. Transactional data is stored in blocks linked to each other forming a chain. It facilitates assets tracking in a business. An asset can be anything of a value including tangible or non-tangible things. Examples of tangible assets include car, land, house, and cash. Intangible assets include intellectual assets such as copyrights, properties, patents, or branding. Transactions are grouped together in a block according to the block size defined. Block confirms the sequence of transactions using a timestamp. A hash of a previous block is maintained by each block which helps in backtracking while validating new blocks. This strengthens the security of whole blockchain as no one can add a malicious or corrupted block in between valid blocks. This feature is known as immutability in which no existing block can be altered. Each node updates the ledger whenever a transaction occurs. Each full node stores the copy of whole ledger (blockchain). Figure 1 depicts the architecture of blockchain.

As compared to traditional distributed databases, blockchain provides significant advantages such as reduced cost and time, no reliance on a third party, and security of assets [7] [8]. In distributed database systems, each node in the network holds its own ledger of record and relies on some intermediary while performing a transaction. Delay in execution time of agreements and additional fee charged by intermediaries make this design inefficient. This design is vulnerable to security attacks as a central intermediary might be compromised and entire business can suffer. Transactions in blockchain are verifiable and secure. Blockchain design has overcome these issues. It uses a consensus mechanism among nodes of the network to validate the information, eliminating the need for intermediaries [9] [10]. Consensus mechanism is a core concept in blockchain that ensures a tamper free environment in which only one version of truth is agreed upon by all the nodes. All the nodes in a decentralized network must reach at an agreement about state of the blockchain. This makes it difficult for an attacker to introduce a tampered block into the blockchain. Selecting the right consensus algorithm is a vital decision to make while implementing a blockchain solution. Moreover, there are some critical parameters to be considered while designing a consensus algorithm.

This paper discusses the consensus mechanisms in distributed systems and their significance. Various consensus algorithms designed both for distributed environment in general and specifically for blockchain are reviewed. Parameters relevant to consensus algorithm are identified and discussed that will help the developers in evaluating and designing a new consensus algorithm. Rest of the paper is organized as follows: Section 2 describes the blockchain technology and some well-known cryptocurrencies. Section 3 reviews the work done related to the design and comparison of consensus algorithms in blockchain. Section 4 reviews and discusses the consensus mechanism in distributed systems. Section 5 includes the comparative analysis of some recently designed consensus algorithms. Section 6 presents the research opportunities. Section 7 concludes the paper.

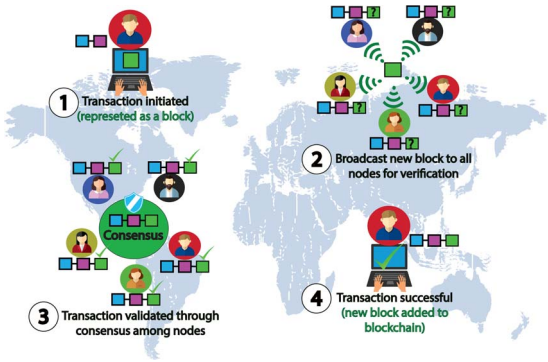

Fig. 1. Blockchain architecture.

II. Blockchain Technology And Crytpocurrencies

Blockchain is a distributed system in which a common ledger of transactions is kept and shared among participants of the network. Transactions are kept in the form of chain of blocks where each block references previous block using a hash value. The participants must agree on a list of transactions. In either case, divergence will happen resulting in forks. Each miner has a local state of blockchain which may vary from miner to miner. The local states will be different when two or more miners simultaneously mine two different blocks. In that case, two blocks will point to the same previous block. The global state of a blockchain is constructed by taking union of all local states. In a global state, the point at which different blocks have a same predecessor block is referred to as fork. To resolve fork, various implementations of blockchain; such as Bitcoin and Ethereum use different mechanisms to identify the main branch of blockchain. Bitcoin resolves fork by considering a deepest branch as a main branch of blockchain. It uses Nakamoto's consensus protocol for this purpose. Main branch is selected as a branch that has the largest number of nodes. On the other hand, a consensus algorithm in Ethereum, Greedy Heaviest Observed Subtree (GHOST) selects heaviest subtree as a main branch.

A. Bitcoin

In Bitcoin or any other cryptocurrency, a transaction is a an atomic operation that represents the transference of money from sender to receiver. A public key and digitally signed hash value of previous transaction defines a current transaction in bitcoin. A transaction is identified by its hash value. For digitally signing the transaction, a private key is used. Public key is used in verification of the transaction. Each node in this peer to peer network possesses a copy of the ledger. If user A wants to transfer some coins to another user B, he has to announce this transaction publically. The network then verifies the correctness of this transaction. To ensure consistency, output of a transaction must not be used as input by entire blockchain more than once. If output is referenced more than once then this results in double spending problem that is strictly prohibited in the network. To eliminate these issues, proof of work is demanded from every node that verifies the transaction (also known as miners). Miners in a blockchain must verify the transaction and sender's public key based signature. They have to perform heavy computation to prove that they are valid entities in the network. The network will remain consistent only if the total computation power possessed by honest nodes is more than the computation power of an attacker. Merkle trees are used in Bitcoin to verify and commit transactions. It is a binary hash tree where each leaf holds transaction's ID and is represented by hash of its child nodes. A transaction is committed with a single hash value stored at the root of Merkle tree. Each user need to store only the root node of the Merkle tree. Bitcoin's transaction processing rate is not scalable. With a high computational power consumption, it can only process 7 transactions per second. Bitcoin's POW mechanism consumes massive power. It is estimated that power consumption of Bitcoin per transaction is around 545 KWh [42].

B. Ethereum

Ethereum is proposed by Vitalik Buterin to address the limitations in Bitcoin. The blocks in Ethereum also hold a list of transactions and the most recent state. Apart from transferring money, Ethereum also ensures execution of smart contracts. It uses GHOST protocol to ensure consensus in the network. It caters the issue of stale blocks that arise when a group of miners possess more computation power than the rest and thus contribute more to the network. This will result into centralization issue. The GHOST protocol integrates the stale blocks in longest blockchain calculation. The stale blocks are rewarded that eliminates the centralization issue. Miners are now rewarded even if they haven't managed to be a part of the main blockchain. As compared to Bitcoin, Ethereum has better block time (15 seconds). It has a transaction processing rate of 11 per second. It is estimated that power consumption of Ethereum per transaction is around 49 KWh [42].

III. Related Work

This section discusses the various consensus algorithms in blockchain that are recently proposed. The research works that have performed comparative analysis of consensus algorithms in blockchain are also reviewed.

A. Secure sharding algorithm for open blockchain [11]

ELASTICO is proposed as a scalable agreement algorithm for permission-less blockchain. Transaction rate scales linearly with the computation involved in mining procedure. That is, with more computation power, more transaction blocks are processed. This algorithm can tolerate with adversaries that possess one fourth of the total computational power. The main idea is to divide a network into small chunks known as committees. Each committee processes a disjoint set of transactions and whole procedure is parallelized. This algorithm is different from the classical byzantine algorithm in a way the agreement condition is implemented in probabilistic manner. Each honest process matches its agreed value with a constraint function and checks its validity. The solution is accepted only if it satisfies the constraint function. All committees perform a classical byzantine consensus. The final committee merges the results (shards) of all committees. Some security properties are defined and algorithm is validated with respect to each of these properties that highlights its security strength. Experiments are run on Amazon EC2 involving up to 1600 nodes. Regarding implementation of this algorithm, additional code is added to the publically available code for Bitcoin [11].

B. SCP: A computationally-scalable byzantine consensus algorithm for blockchain [12]

SCP is proposed as a new computationally scalable blockchain consensus algorithm based on byzantine. Computationally scalable means the algorithm is flexible enough to adjust the bandwidth consumption by altering the computational parameters such as level of difficulty in proof of work (PoW) mechanism. If we increase the computational power then an increase in throughput will be expected. In SCP, miners with less computational power are allowed to become a part of committee and involve themselves in the voting process. Moreover, transactions are divided uniformly among the data blocks resulting in decentralization of network. No central certificate is needed thus eliminating the risk of single point of failure. The SCP algorithm is mapped to scalable cryptocurrency: SCoin. Additionally, Merkel tree is used that avoids double spending problem. Double

spending is prevented locally by checking if a transaction has only one output or not. SCP is proved to scale linearly with the increase in computational capacity without increasing the bandwidth in a quadratic manner. Experiments are run with 80, 40, 20, and 10 cores on Amazon EC2 to assess the computational scalability [12].

C. *Leader-free byzantine consensus algorithm [13]*

Byzantine based consensus problem is discussed for a partially synchronous system. A partial synchronous system is an asynchronous system that comes to synchronous mode eventually. Designing a deterministic leader free algorithm for a partially synchronous system is a challenging task. A leader free algorithm, however, consists of rounds in which all to all communication happens among the nodes. This paper proposed a leader free algorithm to achieve consensus in partial synchronous system. The consensus leader free algorithm for synchronous system is extended for partial synchronous system. The consensus algorithm considered in this paper for synchronous systems is based on interactive consistency problem. Each process computes a set of values with each value representing an element for a process. The correct processes come up with the same set of values. This algorithm is extended in two ways. One is by using parametrized consensus algorithm in which different models of fault are allowed in agreement. The other way proposed in this paper is to make the consensus optimal with strong validation [13].

D. *On the Security and Performance of Proof of Work blockchain [14]*

A quantitative framework is introduced to assess the security and performance of PoW consensus based blockchain with respect to various network and consensus related parameters. PoW consensus is adopted by most of the existing blockchain. The variants of PoW have not received attention in literature with respect to security assessment. The relationship between the performance and security of PoW blockchain is not studied well. This paper has studied this relationship using a simulator to mimic the network and consensus mechanism of a blockchain. Impact of varying the block size and block interval on double spending and selfish mining is studied. It is found that the high block reward for blockchain will result in less chance of double spending attack. With the block size of 1 MB and block interval time of 1 minute, the security is not penalized.

E. *Blockchain Consensus: An analysis of Proof-of-Work and its applications [15]*

The authors analyzed proof of work in detail. Average time to mine a block, the number of stale blocks, and average fork length is analyzed with the varying problem complexity. As the problem complexity is increased, mining time for new blocks is also increased. Various applications of blockchain in general are also reviewed. PoW is a widely used consensus mechanism in blockchain solutions. It was first introduced by Bitcoin. In this algorithm, all the nodes vote by solving a proof of work and create new blocks with their own computation power. Bitcoin works with a hash based proof of work to find a nonce value. Security of this algorithm relies on the assumption that no node should possess more than 50% of the total computational power of a network. If this happens then such a node can control the whole system by constructing a longest chain. To ensure an agreement among participating nodes of a network, a voting mechanism is needed. In Nakamoto consensus, a node that wins a lottery is elected as a leader. In bitcoin, first node to solve a puzzle wins a lottery. This leader then broadcasts this block and other nodes vote implicitly for this block acceptance. Traditional Byzantine fault tolerance (BFT) algorithms are also used to achieve consensus among the nodes involving explicitly voting mechanism.

F. *The Blockchain Consensus Layer and BFT [16]*

In this paper, the authors studied the relevance of byzantine fault tolerance algorithms and blockchain algorithms. A layered view of blockchain is presented that highlights its various components. Hybrid solution involving Nakamoto algorithm and BFT is also discussed.

G. *(Leader/Randomization/Signature)-free Byzantine Consensus for Consortium blockchain [17]*

A consensus algorithm without signature, leader, and randomization is presented. The algorithm reduces the multi valued based Byzantine consensus problem to binary Byzantine consensus problem. Binary instances of consensus are run in parallel by which a value is decided in constant time. The optimality of proposed design is proven theoretically.

H. *Implicit Consensus: Blockchain with Unbounded Through- put [18]*

An implicit consensus model is proposed in which each node possesses its individual blockchain. Primary benefit gained with this is an unbounded throughput. Termination property of BFT is replaced with a property of self-interest. Consensus is not guaranteed for every transaction. This makes this scheme scalable with linear message complexity. Instead of consensus of transaction blocks, a special type of blocks are considered, known as check point blocks. The performance and other aspects are analyzed theoretically.

I. *Design and Implementation of a Proof-of-Stake Consensus Algorithm for Blockchain [19]*

This research work discusses the implementation of proof of stake (PoS) consensus algorithm in blockchain. PoW is one of the heavily discussed aspects of Bitcoin. It is used as a way of selecting a block signer based on the computational effort of miners in computing complex mathematical problem. To overcome the energy wastage issue of it, Proof of Stake (PoS) is introduced. The basic idea is that the nodes having more stake will get an opportunity to add blocks to the network more often. A new block signer is elected using a random criteria based on the amount of stake that a miner possess. With this, the power effort in mining a block will get significantly reduced as compared to brute-force like computation in PoW. Ouroboros and Casper are the two well-known PoS based consensus protocols. Ouroboros elects the stakeholder randomly by a secure coin flipping algorithm and an effective time slots synchrony. Casper provides a weaker guarantee regarding how much stake is controlled by adversary to cause disruption.

J. *PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake [20]*

This paper discusses proof-of-stake algorithm that minimizes the energy consumption as it has no long term

impact on the security of a blockchain network. Design of a peer to peer crypto currency: PPcoin is introduced that has no dependence on energy consumption. The design of a proof of stake algorithms is discussed with respect to the concept of coin age. Coin age is the time period for which a currency amount is held. Unlike PoW, creator of a new block is elected deterministically depending on its wealth (stake). There is no reward associated with the block mining or generation. It is more cost effective as miners do not have to compete to approach to the mathematical solution first. The hash target is not fixed for all nodes. It is dependent on the coin age consumed. Blockchain that has highest coin age consumed is treated as a winner. This algorithm makes double spending attack more easy for an attacker as only certain amount of coin age has to be accumulated to introduce a forged block to a blockchain.

K. *A Proof-of-Trust Consensus algorithm for Enhancing Accountability in Crowdsourcing Services [21]*

A consensus algorithm is proposed for crowdsourcing services. Transactions validators are elected based on the trust values of nodes. Two algorithms, Shamirs secret sharing algorithm and RAFT leader election are used in its design. The scalability challenges associated with the traditional BFT algorithms are addressed. The consensus algorithm is divided into 4 phases. Phase 1 is focused on leader election in consortium using Raft leader election algorithm. Next phase selects transaction validators using voting mechanism. In phase 3, group of nodes (that are selected for transaction validation in previous phase) validate the transactions. Last phase assembles the validated transactions and link it with the blockchain. The proposed algorithm can handle the failure of nodes such that $x \geqslant 3y+1$ where x is the number of nodes in the network and y is the number of byzantine nodes. Four categories of nodes are made in consensus procedure. These are consortium leader nodes, gateway nodes, normal ledger management nodes, and validator nodes. The proposed algorithm design is reviewed on the basis of properties: agreement, validity, performance, fairness, liveness, and scalability. Attack scenarios are also discussed. The size of consortium and validator group remains the same in hybrid consortium architecture even with the rapid increase in the number of nodes. This makes the scalability better than BFT. Experiments are conducted using simulation on a single machine. Comparison of PoT with the three consensus algorithms: Ripple, consortium, and joint consensus is done. The effect of concurrent transactions is also studied well. The results indicate that PoT is better than other consensus alternatives with respect to accuracy, scalability, and performance.

L. *RMBC: Randomized Mesh Blockchain Using DBFT Consensus Algorithm [22]*

Diversity of opinion based BFT consensus algorithm is proposed and discussed. This algorithm mainly addressed the two problems in PoW and BFT algorithms respectively. In PoW, there is a cost overhead. With BFT, as number of malicious entities in a network increases by a large number then invalid transactions cannot be prevented. There are two layers in consensus agreement process. The first phase involves a BFT algorithm. Second phase performs grouping of related departments. After that, a verifier is elected from each department. Transaction is confirmed if the first and second agreement phase are identical. The proposed design of algorithm is not validated extensively using experiments. A short quantitative comparison with PoW and BFT solutions is provided.

M. *PoPF: A Consensus Algorithm for JCLedger [23]*

Developers can customize the cloud specific services using a model known as JointCloud. JCLedger is a blockchain based solution for JointCloud. This paper discusses the proposed consensus algorithm for JCLedger, known as PoPF (Proof of Participation and Fees). The computing overhead of PoW makes it difficult to get adopted by JCLedger. Therefore, a consensus algorithm is proposed that consumes much less computing power. The candidates for mining are elected using two factors: fee paid by participant and number of times the participant appeared as accountant. The authors have experimentally evaluated the algorithm through simulation with respect to distribution of accountants. The accuracy and performance related aspects are not involved in experiments.

N. *The Ripple algorithm Consensus Algorithm [24]*

A consensus algorithm, Ripple is presented that uses subnetworks that are collectively trusted within the whole network. The protocol is executed every few seconds to ensure correctness of the network. After consensus is reached, the ledger becomes closed. The last closed ledger possessed by all the network nodes should be identical if no fork happens. This protocol runs in rounds. At the start, each node publically announces a candidate list having all the valid transactions that it has perceived before the consensus round. Each node then votes on the correctness of all transactions and unites the candidate sets of all nodes. Number of votes are compared to a threshold value and accept/ discard the transaction on the basis of it. Then, a final round is carried out which requires that at least 80% of the nodes must agree on a transaction. All transactions confirming to this criteria are validated and applied to the ledger, constituting a new closed ledger.

O. *Proof of Vote: A High-Performance Consensus algorithm Based on Vote Mechanism & Consortium Blockchain [25]*

Consensus algorithm, named as Proof of Vote (POV) is pro- posed that is efficient than POW. Verification of the blocks is carried out using voting mechanism. Four roles are defined in a consortium network model. These are commissioner, butler candidate, butler, and ordinary user. The proposed algorithm has shown best performance in terms of power consumption.

P. *Proof of Work [26]*

In PoW, miners that solve a mathematical puzzle first are rewarded. The complexity of a puzzle is decided by the overall power of the blockchain network. In order to mine, miners have to possess huge computational power to solve a mathematical puzzle first. The main disadvantage of this algorithm is the overhead of computations involved.

Q. *Qualitative comparisons*

In [27], authors have done a short comparison of PoS, PoW, BFT, PoET, and Federated BFT on the basis of some qualitative parameters. The research work [28] compared variants of Proof of exercise (PoX) and some hybrid

consensus algorithms qualitatively. In [29], five consensus algorithms, PBFT, PoW, DPoS, PoS, and RAFT are compared briefly with respect to Byzantine and crash fault tolerance, verification speed, throughput, and support for scalability. The research work [30] briefly compared some consensus algorithms qualitatively. However, some recent algorithms proposed are not involved in comparison. Giang et al. [43] qualitatively compared the two main categories of consensus mechanisms in blockchain, vote-based and proof-based. The authors only briefly discussed the working mechanism of various consensus algorithms that belong to these two categories. The categories are generically reviewed with respect to agreement, joining of nodes, number of executing nodes, decentralization, trust, identities of nodes, and security threats. In [44], author compared PoW based solutions to BFT state machine replication based approaches. High level comparison is done on the basis of node identity management, consensus finality, scalability, performance, power consumption, adversary tolerance, network synchrony, and availability of correctness proofs. In [45], authors reviewed some consensus protocols on the basis of consensus resilience properties. Some prominent consensus protocols focused on permissioned systems are discussed. Elli et al. [46] discussed the horizontal scaling of permissioned blockchain using sharding. In permissioned blockchain, the power of adversary is relaxed. Confidentiality in the presented design is also reviewed. Sharding is adapted for permissioned asset management environment.

There exist some research gaps in the fore-mentioned research works. The proposed algorithms are not evaluated in detail with respect to performance and security parameters. Performance and security are two core aspects that must be involved in evaluating the quality of blockchain consensus solutions. There is a need to cover these aspects to analyze the consensus solutions. Research works that have compared the consensus algorithms in blockchain have not incorporated some critical performance and security parameters. The comparison is done briefly that does not help in evaluating maturity of a consensus solution. Most of the comparison studies have evaluated the algorithms qualitatively without considering quantitative performance related aspects. Moreover, some recent consensus algorithms are not involved in comparative analysis. In this paper, we have identified some critical parameters related to the performance and security of a consensus algorithm. Various recently proposed consensus algorithms are compared and analyzed with respect to the identified parameters. This research study will help developers in selecting appropriate algorithm design for their blockchain application. This will also highlight the research gap regarding evaluation of consensus algorithms.

IV. Consensus In Distributed Systems And Blockchain

Consensus is a core concept in distributed systems and not restricted to blockchain. It applies to scenarios in which multiple processes or nodes have to maintain a common state of data item. Permission-less and permissioned blockchain are two major types of blockchain. In permission-less blockchain, nodes are anonymous. A new tampered transaction block can be added resulting in a fork. Fork occurs where a valid transaction mismatches with the invalid one. The primary goal of consensus algorithm is to achieve agreement among the nodes such that each node agrees on one true value. In permissioned blockchain, the nodes are not anonymous and considered as known entities.

Consensus is still important to achieve in that case as the nodes are not trustworthy. Figure 2 depicts the architecture of blockchain. In general, consensus is worth to consider where the nodes of a network are faulty or they communicate in an unreliable way. Consensus is achieved in distributed systems with regard to some failures. Further, while defining consensus system communication model such as synchronous or asynchronous is also considered relevant. There are various categories of failures: crash failure, transient failure, omission failure, security failure, software failure, byzantine failure, temporal failure, and environmental perturbations.

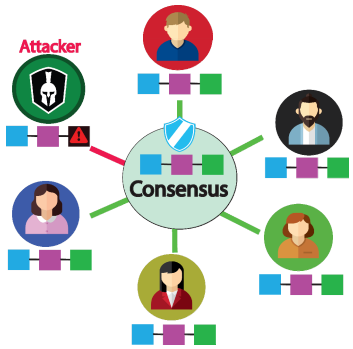

Fig. 2. Consensus in distributed environment.

- **Crash failure:** In crash failure, process halts in an irreversible way. Timeout measures will help to detect this failure in synchronous environment whereas in asynchronous environment, it is difficult to detect crash failure.
- **Transient failure:** These faults are non-deterministic and permanent. In terms of hardware it can be a fault due to weak batteries or power surge. With respect to software, these errors could be bugs in internal codes that occur rarely and are not detected during the testing phase.
- **Omission failure:** These type of failures occur due to transmission issues such as buffer overflow, collisions, or malfunctioning of transmitter.
- **Security failure:** Security failures are induced as a result of security attacks and impersonation. Data might be corrupted due to it.
- **Software failure:** Software failures are due to the design and modeling flaws. This type of failure can induce other type of failures such as crash or omission.
- **Byzantine failure:** Byzantine failure refers to the fault that presents different symptoms to all members of the system. It hinders all members to reach a consensus or agreement. These failures confuse the system and makes it difficult for the system to tolerate the failures. For example, a server may appear failed to one observer and functioning to the other. The server cannot be entitled as failed because both the observers will not reach consensus because they have conflicting information.
- **Temporal failure:** Temporal failures occur due to latency in meeting the deadline. That is, correct results might be generated but too late to be considered useful. This category is quite relevant in real-time systems.
- **Environmental perturbations:** This type of failure occurs if solution is not made adaptable to the environment changes. Environment change might make a correct result wrong [31].

Blockchain are not managed by a central authority and maintains a decentralized ledger. Malicious entities might be given huge incentives to try causing faults. Thus considering Byzantine problem and its solutions will be relevant to blockchain. Byzantine fault refers to the fault that presents different symptoms to all members of the system. It hinders all members to reach a consensus or agreement. These failures confuse the system and makes it difficult for the system to tolerate the failures. For example, a server may appear failed to one observer and functioning to the other. The server cannot be entitled as failed because all the observers will not reach consensus. The term Byzantine is derived from Byzantine general's problem. It is an agreement problem in which group of generals head an army encircling a city. The generals want to decide upon a plan regarding attacking a city. Preference of each general will be either to retreat or attack. Each general should agree on a common decision. Issue is created when there is a traitor in a group of generals. The traitor will vote for sub-optimal strategy and do this act selectively to create confusion. For example, if there are 11 generals with one traitor and ten honest generals. If five of them vote for retreat and five of them vote for attack, then the traitor will send a vote of retreat to the group supporting retreat and vote of attack to the group supporting attack. In this case, half of the army will retreat and half will attack. Consensus problem is difficult to be solved in asynchronous networks. To cater this, some randomized consensus solutions are proposed in literature that relax the three mandatory properties of Byzantine consensus, agreement, termination, and validity. Some well-known randomized solutions include Ben-Or's consensus algorithm and Rabin's consensus algorithm. There are many flavors of randomized algorithms such as Monte Carlo consensus and Las Vegas consensus algorithms. There is another category of consensus algorithm that deals with a leader selecting a probability for itself by election process. Monte Carlo consensus is further classified on the basis of relaxed validity property. Randomized consensus solutions have a drawback that they violate the safety properties of the distributed system. The main categories of consensus algorithms are randomized [32] [33] [34], deterministic [35] [36], Monte Carlo [37] [38], Las Vegas [39], leader based [40], and leader-free algorithms [13]. Apart from this, each algorithm is also categorized based on communication model such as synchronous, partial synchronous, and asynchronous. Randomized consensus provides guarantee to agreement, validity, and termination properties but with some probability value. If no probability is associated then the solutions are referred to as deterministic. Monte Carlo consensus works by running Monte Carlo algorithm on each process or node with specific ranged data. Values from various processes are combined to a global consensus value. Las Vegas is another category in which each consensus round has a probability. The difference between Monte Carlo and Las Vegas is in the running time. Monte Carlo has deterministic running time whereas Las Vegas has a probabilistic running time. There also exist leader based consensus algorithms in which a leader is needed to make termination on consensus. Leader-free consensus algorithms are also proposed. Figure 3 depicts the generic architecture and categorization of consensus mechanisms in distributed systems.

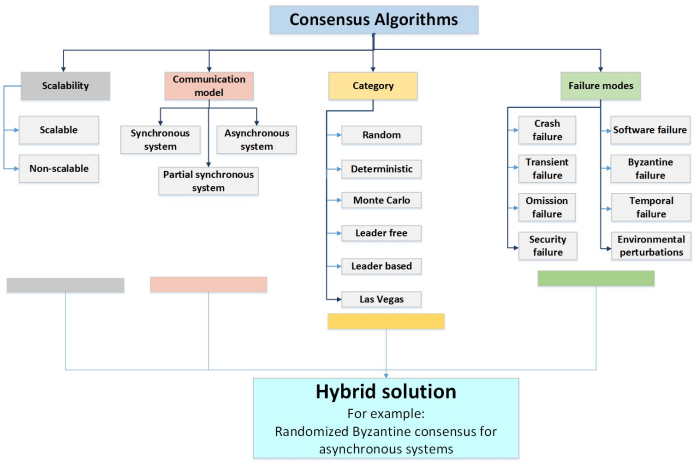

Fig. 3. Categorization of consensus algorithms.

Apart from the general categorization of consensus algorithms in distributed environment, the categorization can be done with regard to blockchain. Specific to blockchain, consensus algorithms can be categorized into two groups: proof-based consensus and vote-based consensus. In proof-based consensus, node that wants to join the network must prove that it is more qualified than the rest in carrying out the appending work. In vote-based consensus, each node in the network is required to communicate and exchange the result of new transaction block that it verifies to the rest of the network nodes. Final decision is made only after considering results of the majority. For example, a node A will manage to get a block x appended to the blockchain only if at least T nodes append the same block x to it. T is a threshold parameter decided by the system.

A. Comparative analysis of consensus algorithms in blockchain

This section discusses the parameters relevant in evaluating the consensus algorithms in blockchain. Blockchain type, transaction rate, scalability, adversary tolerance model, experimental setup, latency, throughput, bandwidth, communication model, communication complexity, attacks, energy consumption, mining, consensus category, and consensus finality are identified as a critical parameters for comparing the various consensus algorithms for blockchain. Comparative analysis of some recently proposed algorithms: ELASTICO [11], leader-free Byzantine consensus [17], implicit consensus [18], blockchain with unbounded throughput [18], Proof of trust (PoT) [21], DBFT consensus [22], PoPF [23], Ripple [24], Proof of Vote [25], and Proof of work [26] is performed. The comparative view of the consensus algorithms is presented in table I-III. The identified parameters and comparison of the consensus algorithms with respect to them is presented as under.

- **Blockchain type:** There are three types of blockchain, public, private, and consortium [41]. The type of blockchain defines the membership control in the consensus algorithm. This has to be considered while evaluating the consensus algorithms to check what kind of membership is assumed in the design. The type of blockchain should be chosen according to the nature of business application.

- **Scalability:** Scalability is a core requirement to deal with big data in today's environment. Scalability is achieved if increasing the number of nodes results in more transaction blocks being processed. Proof of trust and ELASTICO are scalable. Implicit consensus and PoW are not scalable solutions. Other algorithms involved in comparisons are not tested with respect to their scalability yet.
- **Adversary tolerance model:** Adversary model controls the fraction of blockchain network that can withstand failure or attack without affecting consensus. The algorithm proposed for consensus in blockchain come with threshold value for this adversary model. Higher value for adversary threshold is better. ELASTICO has best adversary control than the rest of the algorithms.
- **Performance related parameters:** Some of the existing consensus algorithms are not experimentally evaluated. They are only compared theoretically using correctness proofs. However, along with this there is a need to have a quantitative analysis in place that reviews the performance and security of these consensus algorithms. Latency, throughput, and bandwidth are the three core performance aspects that must be focused on for each of the consensus algorithms. Except ELASTICO, other algorithms are not experimentally evaluated with respect to these performance aspects.
- **Communication model and complexity:** In synchronous communication, sender waits for recipient to acknowledge the request. In asynchronous communication, sender does not need to wait for the response from recipient and continue with the communication. For real time applications that cannot afford delays, PoW, PoT, Ripple, and implicit consensus can be considered. If there are more read operations expected in an application then synchronous model must be chosen as it gives immediate response. ELASTICO and leader free consensus algorithm [17] has synchronous communication model assumed in consensus algorithm design. Leader free consensus algorithm [17] has linear and better cost of communication than ELASTICO and PoT. Communication cost of the rest of the algorithms is not analyzed yet in literature.
- **Attacks:** Ripple is prone to Sybil attack in which single attacker controls multiple network nodes by creating different IP addresses, virtual machines, and user accounts. Leader free consensus, PoT are secure against this attack. The algorithms are not evaluated and analyzed with respect to the number of security attacks possible in a blockchain network. It is important to review the algorithms with respect to security attacks.
- **Energy consumption:** Energy consumption defines the amount of energy or electricity that the hardware infrastructure consumes in the blockchain network. The energy consumption of almost all of the consensus algorithms is not experimentally evaluated.
- **Mining and consensus category:** It defines how the process of mining is carried out in the blockchain network. It is closely related with how the verification process occurs. Proof-based consensus is ideal for networks with large number of nodes. Vote-based consensus, on the other hand, works best with limited number of nodes. If the network has large number of nodes then it is preferable to use ELASTICO, PoW, PoPF, and implicit consensus. In other case, the rest of the consensus protocols (involved in comparison) would fit best.
- **Consensus finality:** Finality in blockchain indicates that the transaction is finalized and will not be reverted back. It is an important aspect to consider while designing a consensus protocol for blockchain. Probabilistic finality and absolute finality are the two categories of consensus finality. In probabilistic finality, as the block gets deeper into the chain the chance of its reverting decreases. However, in absolute finality, a transaction is immediately finalized after its inclusion in the blockchain. If absolute consensus finality is to be achieved then ELASTICO, PoPF, and implicit consensus are recommendable variants in consensus.

In summary, number of features exist to evaluate a consensus algorithm with respect to its working, performance, and security related aspects.

V. CONCLUSION AND FUTURE WORK

With the recent trend towards blockchain, a lot of applications and businesses are moving towards blockchain based solutions. This made it important to comprehensively analyze blockchain with respect to performance and security characteristics. Recently, some research efforts are done regarding comparing the existing consensus algorithms used in blockchain and proposing a new one. In this paper, we have discussed in detail the consensus mechanisms, their categories, and importance in distributed environment. Consensus mechanisms are discussed generally for a distributed system and specifically for blockchain. We compared some recently proposed consensus algorithms with respect to number of parameters that have a significant impact on consensus algorithm. The parameters identified for comparison covers both security and performance aspects. The algorithms are then discussed along with each of the identified parameters. Apart from these, a number of other aspects are also important to be considered. Network topology (e.g. fully connected graph), transaction rate, consistency achieved by the consensus solution, concurrency check, transaction verification speed, and complexity of rounds (if the consensus algorithm involved multiple rounds or phases). For future research, these parameters can be incorporated to make a more detailed comparison. The comparative view provided in this paper has highlighted the parameters with respect to which some recent algorithms need evaluation and analysis. A detailed qualitative and quantitative comparison can be carried out that covers up the missing areas in comparison presented in this paper. Experiments must be carried out in cluster environment to truly evaluate the strengths and weaknesses of the consensus algorithms with respect to big data needs.

REFERENCES

[1] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the internet of things," Ieee Access, vol. 4, pp. 2292–2303, 2016.

[2] J. L. Zhao, S. Fan, and J. Yan, "Overview of business innovations and research opportunities in blockchain and introduction to the special issue," 2016.

[3] M. Crosby, P. Pattanayak, S. Verma, and V. Kalyanaraman, "Blockchain technology: Beyond bitcoin," Applied Innovation, vol. 2, pp. 6–10, 2016.

[4] D. Yermack, "Corporate governance and blockchains," Review of Finance, vol. 21, no. 1, pp. 7–31, 2017.

[5] T. J. MacDonald, D. W. Allen, and J. Potts, "Blockchains and the boundaries of self-organized economies: Predictions for the future of banking," in Banking Beyond Banks and Money. Springer, 2016, pp. 279–296.

[6] M. Pilkington, "11 blockchain technology: principles and applications," Research handbook on digital transformations, p. 225, 2016.

[7] T. McConaghy, "Blockchain, throughput, and big data," Bitcoin Startups Berlin, Oct, vol. 28, 2014.

[8] D. Brandon, "The blockchain: The future of business information systems," International Journal of the Academic Business World, vol. 10, no. 2, pp. 33–40, 2016.

[9] J. Mattila et al., "The blockchain phenomenon–the disruptive potential of distributed consensus architectures," The Research Institute of the Finnish Economy, Tech. Rep., 2016.

[10] H. Watanabe, S. Fujimura, A. Nakadaira, Y. Miyazaki, A. Akutsu, and J. J. Kishigami, "Blockchain contract: A complete consensus using blockchain," in Consumer Electronics (GCCE), 2015 IEEE 4th Global Conference on. IEEE, 2015, pp. 577–578.

[11] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena, "A secure sharding protocol for open blockchains," in Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2016, pp. 17–30.

[12] L. Luu, V. Narayanan, K. Baweja, C. Zheng, S. Gilbert, and P. Saxena, "Scp: A computationally-scalable byzantine consensus protocol for blockchains." IACR Cryptology ePrint Archive, vol. 2015, p. 1168, 2015.

[13] F. Borran and A. Schiper, "A leader-free byzantine consensus algorithm," in International Conference on Distributed Computing and Networking. Springer, 2010, pp. 67–78.

[14] A. Gervais, G. O. Karame, K. Wu¨st, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work blockchains," in Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2016, pp. 3–16.

[15] A. Porat, A. Pratap, P. Shah, and V. Adkar, "Blockchain consensus: An analysis of proof-of-work and its applications."

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

[17] T. Crain, V. Gramoli, M. Larrea, and M. Raynal, "(leader/randomization/signature)-free byzantine consensus for consortium blockchains," arXiv preprint arXiv:1702.03068, 2017.

[18] Z. Ren, K. Cong, J. Pouwelse, and Z. Erkin, "Implicit con- sensus: Blockchain with unbounded throughput," arXiv preprint arXiv:1705.11046, 2017.

[19] E. Garcia Ribera, "Design and implementation of a proof-of-stake consensus algorithm for blockchain," B.S. thesis, Universitat Polite`cnica de Catalunya, 2018.

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

[21] J. Zou, B. Ye, L. Qu, Y. Wang, M. A. Orgun, and L. Li, "A proof-of-trust consensus protocol for enhancing accountability in crowdsourcing services," IEEE Transactions on Services Computing, 2018.

[22] S. Jeon, I. Doh, and K. Chae, "Rmbc: Randomized mesh blockchain using dbft consensus algorithm," in Information Networking (ICOIN), 2018 International Conference on. IEEE, 2018, pp. 712–717.

[23] X. Fu, H. Wang, P. Shi, and H. Mi, "Popf: A consensus algorithm for jcledger," in Service-Oriented System Engineering (SOSE), 2018 IEEE Symposium on. IEEE, 2018, pp. 204–209.

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

[25] 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 High Performance Computing and Communications; IEEE 15th International Conference on Smart City; IEEE 3rd Interna- tional Conference on Data Science and Systems (HPCC/SmartCity/DSS), 2017 IEEE 19th International Conference on. IEEE, 2017, pp. 466– 473.

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

[27] A. Baliga, "Understanding blockchain consensus models," Persistent, 2017.

[28] W. Wang, D. T. Hoang, Z. Xiong, D. Niyato, P. Wang, P. Hu, and Y. Wen, "A survey on consensus mechanisms and mining management in blockchain networks," arXiv preprint arXiv:1805.02707, 2018.

[29] D. Mingxiao, M. Xiaofeng, Z. Zhe, W. Xiangwei, and C. Qijun, "A review on consensus algorithm of blockchain," in Systems, Man, and Cybernetics (SMC), 2017 IEEE International Conference on. IEEE, 2017, pp. 2567–2572.

[30] N. Chalaemwongwan and W. Kurutach, "State of the art and challenges facing consensus protocols on blockchain," in Information Networking (ICOIN), 2018 International Conference on. IEEE, 2018, pp. 957–962.

[31] "Homepage.divms.uiowa.edu," [online] http://homepage.divms.uiowa.edu/ ghosh/16612.week10.pdf.

[32] S. Toueg, "Randomized byzantine agreements," in Proceedings of the third annual ACM symposium on Principles of distributed computing. ACM, 1984, pp. 163–178.

[33] G. Bracha and O. Rachman, "Randomized consensus in expected o (n 2 log n) operations," in International Workshop on Distributed Algorithms. Springer, 1991, pp. 143–150.

[34] J. Aspnes, "Randomized protocols for asynchronous consensus," Distributed Computing, vol. 16, no. 2-3, pp. 165–175, 2003.

[35] M. J. Fischer, N. A. Lynch, and M. S. Paterson, "Impossibility of distributed consensus with one faulty process," Journal of the ACM (JACM), vol. 32, no. 2, pp. 374–382, 1985.

[36] D. Dolev, C. Dwork, and L. Stockmeyer, "On the minimal synchronism needed for distributed consensus," Journal of the ACM (JACM), vol. 34, no. 1, pp. 77–97, 1987.

[37] S. L. Scott et al., "Comparing consensus monte carlo strategies for distributed bayesian computation," Brazilian Journal of Probability and Statistics, vol. 31, no. 4, pp. 668–685, 2017.

[38] S. L. Scott, A. W. Blocker, F. V. Bonassi, H. A. Chipman, E. I. George, and R. E. McCulloch, "Bayes and big data: The consensus monte carlo algorithm," International Journal of Management Science and Engineering Management, vol. 11, no. 2, pp. 78–88, 2016.

[39] H. Ishii and R. Tempo, "Las vegas randomized algorithms in distributed consensus problems," in American Control Conference, 2008. IEEE, 2008, pp. 2579–2584.

[40] A. Mostefaoui and M. Raynal, "Leader-based consensus," Parallel Processing Letters, vol. 11, no. 01, pp. 95–107, 2001.

[41] "Different types of blockchains in the market and why we need them," [online] https://coinsutra.com/different-types-blockchains/.

[42] "Digiconomist - Cryptocurrency Fraud and Risk Mitigation" [online] https://digiconomist.net/

[43] Nguyen, Giang-Truong, and Kyungbaek Kim. "A Survey about Consensus Algorithms Used in Blockchain." Journal of Information Processing Systems 14.1 (2018).

[44] Vukolić, Marko. "The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication." International Workshop on Open Problems in Network Security. Springer, Cham, 2015.

[45] Cachin, Christian, and Marko Vukolić. "Blockchains consensus protocols in the wild." arXiv preprint arXiv:1707.01873 (2017).

[46] Androulaki, Elli, et al. "Channels: Horizontal Scaling and Confidentiality on Permissioned Blockchains." European Symposium on Research in Computer Security. Springer, Cham, 2018.

TABLE I. COMPARISON OF THE CONSENSUS ALGORITHMS WITH RESPECT TO GENERIC PARAMETERS

Parameters	**Consensus algorithms**				
	References	***Date***	***Blockchain type***	***Mining***	***Consensus category***
ELASTICO	[11]	2016	Permission-less	Based on computational power	Proof based
Leader- free Byzantine Consensus	[17]	2017	consortium	Non-proof of work based mining	Vote based
Implicit Consensus	[18]	2017	permissioned	Proof based mining	Proof based
Proof of trust (PoT)	[21]	2018	Permission-based consortium	Probability and vote based mining	Vote based
DBFT Consensus Algorithm	[22]	2018	permissioned	Non-proof of work based mining (random selection of verifier)	Vote based
PoPF	[23]	2018	permissioned	Selection on the basis of accounting right	Proof based
Ripple	[24]	2014	Permissioned	Vote based mining	Vote based
Proof of Vote (PoV)	[25]	2017	consortium	Vote based mining	Vote based
Proof of Work (PoW)	[26]	2008	Permission-less	Based on computational power	Proof based

TABLE II. COMPARISON OF THE CONSENSUS ALGORITHMS WITH RESPECT TO PERFORMANCE RELATED PARAMETERS

Parameters	**Consensus algorithms**						
	Scalability	***Latency***	***Throughput***	***Bandwidth***	***Experimental setup***	***Communication model***	***Communication complexity***
ELASTICO	Linear	103-110 seconds with around 400-800 nodes	1-6 blocks/epoch with 100-1600 nodes	Constant, 5MB/ node	Simulation, Amazon EC2 with 1600 nodes	Synchronous	$O(nc+nc^3)$ where c is a committee size and n is the number of nodes
Leader- free Byzantine Consensus	-	-	-	-	Theoretically evaluated the design	Synchronous	Termination time: O(t) where t is the number of faulty process
Implicit Consensus	Not scalable	-	-	-	Theoretically evaluated the design	Asynchronous	O(n) where n is the number of nodes
Proof of trust (PoT)	Scalable	-	-	-	Simulation, Single machine	Asynchronous	$O(m^2)$ where m is the number of consortium ledger management nodes
DBFT Consensus Algorithm	-	-	-	-	Proposed solution is not validated through experiments	Asynchronous	-
PoPF	-	-	-	-	Simulation, Single machine	-	-
Ripple	-	-	-	-	Simulation, Single machine	Asynchronous	-
Proof of Vote (PoV)	-	0.25 minutes for transaction verification	-	-	Simulation, Single machine	-	-
Proof of Work (PoW)	Not scalable	-	Less than 100 seconds	-	Real implementation done using Bitcoin	Asynchronous	-

TABLE III. COMPARISON OF THE CONSENSUS ALGORITHMS WITH RESPECT TO SECURITY RELATED PARAMETERS

Parameters	**Consensus algorithms**				
	Adversary tolerance model	***Prone to attacks***	***Secure against attack***	***Energy consumption***	***Consensus finality***
ELASTICO	Faulty process can have up to $1/4^{th}$ of the computational capacity	Double spending attack	-	-	Absolute/ instant irreversibility
Leader- free Byzantine Consensus	Number of faulty process < number of processes/3	-	Sybil attack	-	Probabilistic
Implicit Consensus	Number of faulty process ≤ number of processes/3	DDoS attack	-	-	Absolute
Proof of trust (PoT)	n≥ 3b+1 where n is the number of nodes and b is the number of byzantine nodes	DDoS attack	Sybil attack, collusion attack	-	Probabilistic
DBFT Consensus Algorithm	-	-	-		Probabilistic
PoPF	No node can possess more than 50% of the computational power.	-	51% attack	Less energy consumption claimed as compared to PoW	Absolute
Ripple	faulty processes¡= (processes- 1)/5	DoS attack, Sybil attack	-	-	Probabilistic
Proof of Vote (PoV)	faulty processes ¡ 50% of processes	-	-	-	Probabilistic
Proof of Work (PoW)	No node can possess more than 50% of the computational power.	Selfish mining, DoS attack, Sybil attack, bribe attack	-	538 KWh electricity consumption	Probabilistic