Summary on Byzantine Fault Tolerence

Blockchain consensus technology relies on two different protocols: Proof of Work (PoW) and Byzantine Fault Tolerance (BFT).
Proof of Work is considered fully decentralized, while the BFT protocol is often seen as a more centralized approach.

The first part of this paper focuses on the differences between PoW and BFT consensus, highlighting the strengths and weaknesses of each protocol. The second part explores various methods to address the main challenges of PoW and BFT.

Broadly speaking, PoW is more scalable in terms of the number of nodes, whereas BFT can handle a higher transaction throughput and achieves consensus finality faster compared to PoW-based blockchains.

Below is a table that provides a more precise comparison between PoW and BFT consensus.

	PoW consensus	BFT consensus
Node identity management	**open, entirely decentralized**	permissioned, nodes need to know IDs of all other nodes
Consensus finality	no	**yes**
Scalability (no. of nodes)	**excellent (thousands of nodes)**	limited, not well explored (tested only up to $n \leq 20$ nodes)
Scalability (no. of clients)	**excellent (thousands of clients)**	**excellent (thousands of clients)**
Performance (throughput)	limited (due to possible of chain forks)	**excellent (tens of thousands tx/sec)**
Performance (latency)	high latency (due to multi-block confirmations)	**excellent (matches network latency)**
Power consumption	very poor (PoW wastes energy)	**good**
Tolerated power of an adversary	$\leq 25\%$ computing power	$\leq 33\%$ voting power
Network synchrony assumptions	physical clock timestamps (e.g., for block validity)	**none for consensus safety** (synchrony needed for liveness)
Correctness proofs	no	**yes**

There are ongoing efforts to improve blockchain technology by enhancing both Proof of Work and Byzantine Fault Tolerance.
Some methods aimed at improving the performance of PoW-based blockchains include:

- **GHOST protocol**
- **Bitcoin-NG**
- **Block DAF approach**

On the other hand, several approaches have been proposed to address scalability issues in BFT-based blockchains, including:

- **Reducing communication overhead**
- **Switching to randomized BFT protocols**
- **Using a hybrid PoW-BFT approach**, which combines aspects of both PoW and BFT to achieve consensus finality and enhance blockchain performance.

Practical Byzantine Fault Tolerance Algorithm

Now, we will discuss a practical Byzantine Fault Tolerance algorithm.

This approach involves setting up a model and an environment to simulate a blockchain network with both faulty nodes exhibiting Byzantine behavior and non-faulty nodes. Faulty nodes are assumed to be capable of adopting Byzantine failures, delaying communication, or duplicating messages.

The algorithm provides several essential services in blockchain systems, including:

- **Safety**: Ensuring that at any time, fewer than one-third of the nodes are faulty and that all non-faulty nodes maintain the same state.
- **Liveness**: Guaranteeing that clients always receive responses to their requests and that the system state remains updated.
- **Optimal Resiliency**: To tolerate *f* faulty replicas, a total of *3f + 1* replicas is required. This ensures that even if some replicas are non-responsive or faulty, a majority (*2f + 1*) of responses will come from non-faulty nodes.

Optimizations for the Algorithm

Several methods have been implemented to optimize the algorithm while preserving both liveness and safety properties:

1. **Reducing communication overhead** by:
    - Avoiding the transmission of large replies.
    - Reducing message delays for operations.
    - Improving the performance of read-only operations that do not modify the service state.
2. **Cryptographic Enhancements** to ensure safety:
    - Each node shares a secret key with every other node, allowing for efficient computation of Message Authentication Codes (MACs) using MD5.
    - Truncating MACs to reduce size while enhancing security.
    - Achieving significant speed improvements over traditional RSA signatures.