Consensus Algorithms for Edge Computing: A Literature Review

Jiakang Chen
Yale University

Abstract

Consensus algorithms play a pivotal role in ensuring secure, efficient, and scalable operations within edge computing environments. As the adoption of Internet of Things (IoT) devices continues to grow exponentially, coupled with the increasing prevalence of latency-sensitive applications such as autonomous systems, industrial IoT, and real-time analytics, the demand for robust consensus mechanisms working on edge computing systems has become more critical than ever. These algorithms are essential for enabling distributed decision-making, maintaining data consistency, and ensuring fault tolerance across resource-constrained and decentralized edge nodes. Moreover, the unique challenges posed by edge computing—such as limited computational resources, network heterogeneity, and the need for real-time responses—necessitate innovative and adaptive consensus protocols that can operate efficiently under these constraints while balancing fault-tolerance, efficiency, and scalability requirements. Understanding recent advancements and emerging trends in this domain is therefore vital to advancing the capabilities of edge computing systems.

1 Introduction

Edge computing has garnered significant attention in recent years [12, 14]. At its core concept, edge computing offers elastic computing resources at the network's edge, enabling client device requests to be processed by nearby edge nodes rather than distant cloud servers. This proximity allows modern applications and services to greatly benefit from the decentralized nature of edge computing [5]. By handling requests closer to the users, edge computing ensures faster response times, reduces network bandwidth usage, and alleviates the peak load on cloud infrastructure. Furthermore, application deployment can be tailored to the geographic distribution of users, optimizing resource utilization on edge nodes. Given these numerous advantages, it is no surprise that more and more applications and technologies are deployed using edge computing platforms.

One key area where edge computing is primed to make a major impact is in the world of Internet of Things (IoT). The scope of IoT has been growing steadily over the past decade, encompassing an ever-expanding ecosystem that spans multiple domains such as smart cities [17, 11], vehicular technology [16, 10], and smart healthcare systems [2, 8]. In each of these domains, IoT systems are evolving towards distributed architectures that move away from traditional cloud-centric models. This evolution has naturally aligned with the principles of edge computing, which provides the necessary infrastructure to support this distributed paradigm shift [6, 14]. Edge computing is particularly suited for IoT applications because it addresses many of the challenges inherent to these systems. By processing data locally, edge nodes can minimize latency and ensure real-time responsiveness, which is critical for applications like autonomous vehicles and emergency healthcare monitoring. Additionally, edge computing enhances data privacy by enabling sensitive information to be processed closer to its source, reducing the risk associated with transmitting data to centralized servers. This is especially significant in domains like smart healthcare, where patient confidentiality is paramount.

Figure 1: Applications in integration of edge computing with IoT technologies [9]

In this literature review, we first provide an overview of what it means to achieve consensus in the distributed edge computing scenario as well some common algorithms that can already be found in production. We then develop a framework of the three main goals that consensus on the edge is trying

to achieve. The first is fault-tolerance in the sense of being resistant to Byzantine attacks, i.e. still being able to reach consensus even when a portion of the machines in the distributed system may be intentionally bad actors and not complying. The second is that the algorithm should be efficient in terms of having high throughput and low latency for an end user interacting with the system. The third and final goal is scalability in the sense that the algorithm should be able to scale with the system, i.e. as the number of edge devices increase, the algorithm shouldn't become significantly slower. We describe and analyze many advancements for consensus algorithms in these three areas in the remaining three sections of this literature review.

2 Preliminaries

To begin this section, we will describe what it means to achieve consensus, before moving into a brief discussion of some of the popular consensus algorithms on edge computing systems currently in production.

2.1 Consensus

Consensus algorithms are one of the key aspects within the design of decentralized networked systems or distributed computing systems. Consensus mechanisms are those algorithms that enable multiple independent agents to reach an agreement on a certain value, operation, transaction, or other types of data. In a distributed and decentralized system, different agents, or nodes, need to be able to trust each other [9]. Achieving consensus in a decentralized system involves ensuring that all participating nodes agree on a consistent state of the system, even in the presence of faulty or malicious nodes. This agreement is crucial for maintaining the reliability and integrity of operations across the network.

In the edge computing setting, it is especially key for these consensus algorithms to be fault-tolerant, efficient, and scalable. This is because edge nodes are inherently distributed and harder to control, making them more susceptible to malicious behavior or Byzantine faults, where a subset of nodes may act intentionally or unintentionally in adversarial ways. Efficiency is critical as edge computing is often employed to ensure low latency and high responsiveness, which are essential for applications like real-time analytics, autonomous systems, and industrial IoT. Lastly, scalability is indispensable since edge computing frequently involves massive networks with thousands or even millions of nodes, requiring consensus mechanisms that can handle the growing scale without significant degradation in performance or communication overhead. These traits collectively enable edge computing systems to operate securely, reliably, and efficiently under challenging conditions.

2.2 Popular Algorithms

Some of the most popular algorithms used in edge computing environments to achieve consensus include Proof of Work (POW), Proof of Stake (PoS), and Practical Byzantine Tolerance (PBFT) [9]. For the rest of this section, we will briefly explore their designs and drawbacks.

2.2.1 Proof of Work (PoW)

Proof of Work (PoW) is one of the earliest and most well-known consensus mechanisms, popularized by its implementation in Bitcoin. PoW requires nodes, known as miners, to solve complex cryptographic puzzles to validate transactions [3]. The process is computationally intensive, ensuring network security by making it expensive for malicious actors to perform attacks such as double-spending. However, PoW's heavy reliance on computational resources leads to low efficiency and limited scalability, which are significant drawbacks in resource-constrained edge computing environments [9].

2.2.2 Proof of Stake (PoS)

Proof of Stake (PoS) offers an alternative approach to consensus in certain scenarios (such as cryptocurrenyc) by requiring validators to "stake" collateral. Instead of solving computational puzzles, nodes are selected to validate transactions based on the size of their stake, which correlates with their ownership in the network. This mechanism significantly increases the efficiency of the algorithm compared to PoW and is more suited to edge computing scenarios. However, PoS can introduce centralization risks if a small number of nodes hold a majority of the stake, and thus could be vulnerable to Byzantine attacks [9].

2.2.3 Practical Byzantine Fault Tolerance (PBFT)

Practical Byzantine Fault Tolerance (PBFT) is a consensus algorithm designed to ensure reliability in distributed systems, even when some nodes act maliciously. PBFT relies on a set of validating nodes to achieve consensus through a series of communication rounds [1]. Unlike PoW and PoS, PBFT does not require mining or staking, making it less computationally stressing. However, its reliance on frequent communication between nodes results in high overhead, limiting its scalability in large networks [9].

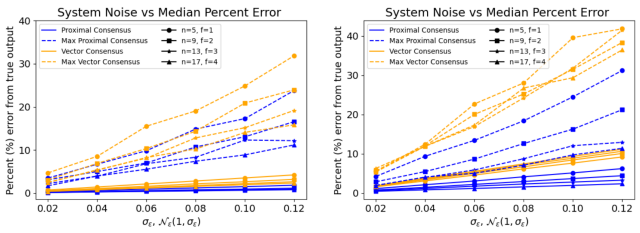
3 Fault-Tolerant Consensus

In edge computing environments, achieving fault tolerance is critical due to the inherent unpredictability and potential failures of edge devices and networks. Fault-resistant consensus mechanisms address this challenge by ensuring system reliability and integrity even in the presence of Byzantine nodes, network noise, or other sources of faults. In this section we

will explore the Proximal Byzantine Consensus Algorithm, which attempts to be a more reliable and fault-tolerant consensus algorithm.

3.1 Proximal Byzantine Consensus (PC)

Proximal Byzantine Consensus (PC) is a novel approach that extends traditional Byzantine Fault Tolerance (BFT) to address challenges unique to edge environments, such as noisy feedback data and volatile network conditions. Unlike geometric methods like convex hulls, PC employs statistical inference to identify the most probable consensus value based on received inputs. This approach tolerates noise and Byzantine behavior while providing a probabilistic interval guarantee (IG) for the consensus value, ensuring high confidence in its accuracy. The protocol provides confidence intervals around the consensus value, quantifying uncertainty and ensuring robustness against noise and malicious behavior [13]. Compared to traditional fault tolerance techniques, proximal byzantine consensus also scales better making it more suitable for complex, distributed edge systems.



(a) Error rates (no attacks). (b) Error rates (optimal attacks).

Figure 2: Evaluation results comparing the accuracy of PC and VC [13].

In Figure 2, we can see compared to other methods, such as the convex hull based vector consensus (VC), PC performs well against Byzantine attacks and noisy environments, such as those that may be encountered in an edge computing environment. Simulation results show a 56%-65% reduction in median error compared to vector consensus under no attack scenarios and a 31%-78% reduction under optimal Byzantine attacks. Additionally, PC maintains robust performance across varying noise levels, ensuring its reliability in diverse edge computing contexts.

4 Efficient Consensus

In edge computing environments, achieving efficiency in consensus mechanisms is essential to meet the stringent performance requirements of latency-sensitive applications. In this section we will discuss three new consensus algorithms that improve on metrics such as latency and throughput when compared to existing popular edge computing consensus algorithms, and analyze their performance.

4.1 Hedera

Hedera employs a hybrid consensus approach, combining Proof-of-Capacity (PoC) and asynchronous Byzantine Fault Tolerance (BFT) to achieve high performance in edge environments [15]. The architecture consists of a two-layer structure:

- A stem chain which utilizes PoC for decentralized and fair committee selection. Nodes contribute disk space and burn tokens as proof of capacity, mitigating Byzantine attacks.
- A leaf chain which implements an asynchronous BFT protocol to process transactions, ensuring high throughput and low latency.

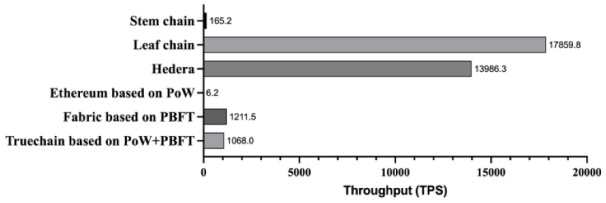


Figure 3: Throughput of different consensus algorithms compared to Hedera [15].

From Figure 3, we can see the dynamic interaction between the stem and leaf chains enables efficient operation allowing Hedera to have a high throughput of 13,986 TPS and minimal latency, outperforming traditional blockchain mechanisms like PoW and PBFT.

4.2 EdgeCons

EdgeCons is a Paxos based consensus algorithm that has been adapted to deal with event ordering in large-scale edge networks by adapting. Unlike traditional PaxosEdgeCons dynamically assigns leadership based on recent performance history, avoiding the bottlenecks of static leader assignment; incorporates a backend cloud as a reliable conflict resolver, ensuring progress even in the event of edge node failures; and supports non-FIFO links, enhancing robustness against network unpredictability. [5]

Simulations demonstrate EdgeCons' significant latency reduction compared to alternatives like Multi-Paxos or E-Paxos, particularly in scenarios with geographically distributed clients and high event volumes (see Figure 4).

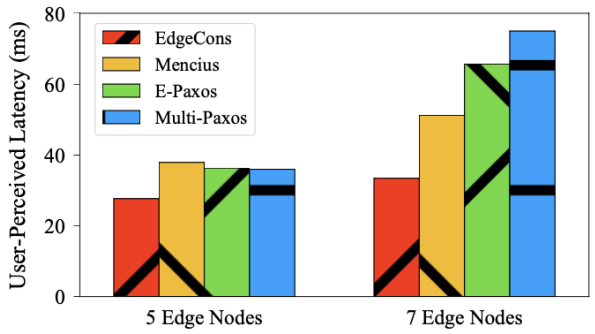


Figure 4: A comparison of the user-perceived latency compared to EdgeCons [5].

4.3 FIBFT

FIBFT enhances the scalability and efficiency of Byzantine consensus mechanisms through K-medoids clustering. Nodes are grouped into clusters based on multidimensional state evaluations (e.g., location, computational power, and network delay). Each cluster performs independent Byzantine consensus, reducing communication complexity. A reputation mechanism quantifies node reliability, integrating subjective logic and environmental priors [4]. Compared to PBFT and other traditional mechanisms, FIBFT achieves higher throughput and lower latency.

From Figure 5, we can see that FIBFT outperforms outher Byzantine Fault Tolerance (BFT) algorithms in terms of both throuhgput and latency, especially as the edge network grows in size. This is attributed to FIBFT's reduced inter-cluster communication and optimized clustering algorithms.

5 Scalable Consensus

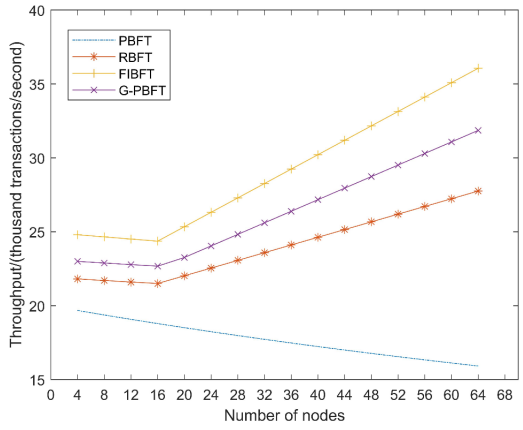
In edge computing environments, scalability is one of the most critical aspects to consider when designing consensus mechanisms. As the network grows in size, traditional approaches like PBFT encounter limitations in their ability to handle large-scale systems efficiently. To address these challenges, the Efficient Fault-Tolerant Consensus (EFT) protocol introduces a novel approach that significantly enhances scalability while maintaining strong correctness guarantees.

5.1 EFT Algorithm

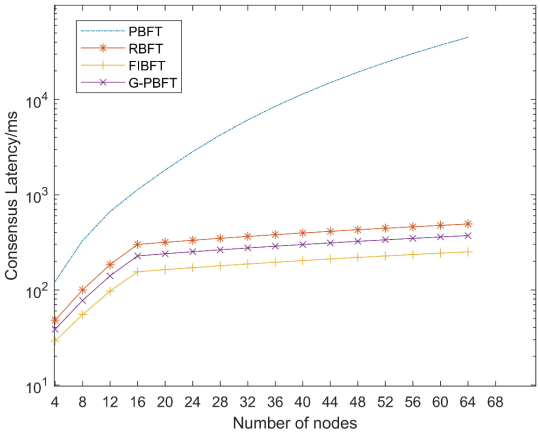
The EFT protocol is specifically designed to achieve an (a,b)-majority consensus in a Byzantine fault environment, which encompasses faults at the physical, protocol, and data layers [7]. The algorithm operates in two main stages:

- In this stage, nodes iteratively broadcast and acknowledge their opinions using programmable transceivers. This phase employs a randomized power selection mechanism to mitigate interference and jamming caused by Byzantine nodes. An acknowledgment scheme ensures that opinions are only broadcast once, preventing repeated interference from malicious actors.

- Once all opinions are collected, nodes execute a greedy algorithm to select the opinion with the most supporters as the consensus. In cases where multiple opinions have equal support, the one with the smallest identifier is chosen.



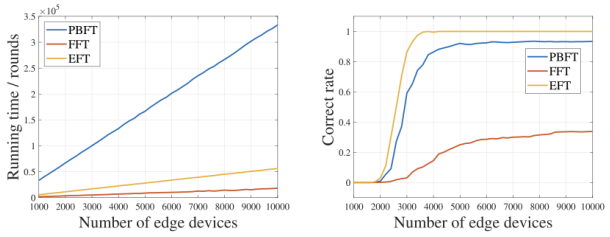
(a) Throughput Comparison



(b) Latency Comparison

Figure 5: Comparison of FIBFT and other consensus algorithms in terms of (a) throughput and (b) latency [4].

EFT achieves its fault-tolerant and scalable properties by leveraging a distributed design that minimizes communication overhead and by adopting a time complexity of $O(n)$, when compared to the much slower $O(n^2)$ message exchanges that PBFT relies on for consensus, which results in significant delays as the network scales. Referencing Figure 6, simulation studies validate the efficiency and robustness of EFT. When



(a) Running time (b) Correct rate

Figure 6: A comparison of EFT with PBFT and FFT [7].

tested in networks with varying sizes and adversarial behaviors, EFT consistently outperforms both PBFT and FFT (Fast Fault Tolerance algorithm) in terms of runtime and success rate. For instance, EFT achieves consensus in networks with over 1,000 nodes in a fraction of the time required by PBFT; and it maintains near 100% correctness in environments with many nodes, significantly surpassing FFT in terms of fault-tolerance, demonstrating its usefulness as the edge computing environment scales.

6 Conclusion

This literature review highlights the critical role of consensus algorithms in ensuring fault-tolerant, efficient, and scalable operations in edge computing environments. As IoT adoption and latency-sensitive applications grow, robust mechanisms are essential to support distributed decision-making, data consistency, and fault tolerance in resource-constrained and decentralized networks. Innovations such as Proximal Byzantine Consensus (PC) enhance fault resistance, Hedera, EdgeCons, and FIBFT address efficiency through low latency and high throughput, and Efficient Fault-Tolerant Consensus (EFT) achieves scalability and correctness guarantees in large-scale networks.

While significant advancements have been made, challenges remain in achieving dynamic adaptability, minimizing energy consumption, and integrating with emerging technologies. Future efforts should focus on hybrid and modular consensus frameworks that balance scalability, efficiency, and resilience to enable the next generation of edge computing systems, unlocking their potential across diverse applications.

References

[1] Miguel Castro and Barbara Liskov. "Practical Byzantine fault tolerance". In: *Proceedings of the Third Symposium on Operating Systems Design and Implementation*. OSDI '99. New Orleans, Louisiana, USA: USENIX Association, 1999, pp. 173–186. ISBN: 1880446391.

[2] Charalampos Doukas and Ilias Maglogiannis. "Bringing IOT and cloud computing towards Pervasive Healthcare". In: *2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing* (July 2012). DOI: 10.1109/imis.2012.26.

[3] Cynthia Dwork and Moni Naor. "Pricing via Processing or Combatting Junk Mail". In: *Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology*. CRYPTO '92. Berlin, Heidelberg: Springer-Verlag, 1992, pp. 139–147. ISBN: 3540573402.

[4] Ningjie Gao et al. "FIBFT: An Improved Byzantine Consensus Mechanism for Edge Computing". In: *2023 IEEE Wireless Communications and Networking Conference (WCNC)*. 2023, pp. 1–6. DOI: 10.1109/WCNC55385.2023.10118628.

[5] Zijiang Hao, Shanhe Yi, and Qun Li. "EdgeCons: Achieving Efficient Consensus in Edge Computing Networks". In: *USENIX Workshop on Hot Topics in Edge Computing (HotEdge 18)*. Boston, MA: USENIX Association, July 2018. URL: https://www.usenix.org/conference/hotedge18/presentation/hao.

[6] Yun Chao Hu et al. "ETSI White Paper #11 Mobile Edge Computing - A key technology towards 5G". In: *ETSI (European Telecommunications Standards Institute)* 11.11 (2015). ISBN: 979-10-92620-08-5, pp. 1–16.

[7] Guanlin Jing et al. "Efficient Fault-Tolerant Consensus for Collaborative Services in Edge Computing". In: *IEEE Transactions on Computers* 72.8 (2023), pp. 2139–2150. DOI: 10.1109/TC.2023.3238138.

[8] Jorge Peña Queralta et al. "Edge-AI in LoRa-based Health Monitoring: Fall Detection System with Fog Computing and LSTM Recurrent Neural Networks". In: July 2019. DOI: 10.1109/TSP.2019.8768883.

[9] Jorge Peña Queralta and Tomi Westerlund. *Blockchain for Mobile Edge Computing: Consensus Mechanisms and Scalability*. 2020. arXiv: 2006.07578 [cs.DC]. URL: https://arxiv.org/abs/2006.07578.

[10] Jorge Peña Queralta et al. "Collaborative Mapping with IoE-based Heterogeneous Vehicles for Enhanced Situational Awareness". In: *2019 IEEE Sensors Applications Symposium (SAS)*. 2019, pp. 1–6. DOI: 10.1109/SAS.2019.8706110.

[11] Ola Salman et al. "IOT survey: An SDN and fog computing perspective". In: *Computer Networks* 143 (Oct. 2018), pp. 221–246. DOI: 10.1016/j.comnet.2018.07.020.

[12] Mahadev Satyanarayanan. “The Emergence of Edge Computing”. In: *Computer* 50.1 (2017), pp. 30–39. DOI: 10.1109/MC.2017.9.

[13] Roy Shadmon, Daniel Spencer, and Owen Arden. *Proximal Byzantine Consensus*. 2024. arXiv: 2402.12577 [cs.DC]. URL: https://arxiv.org/abs/2402.12577.

[14] Weisong Shi et al. “Edge Computing: Vision and Challenges”. In: *IEEE Internet of Things Journal* 3.5 (2016), pp. 637–646. DOI: 10.1109/JIOT.2016.2579198.

[15] Yu Tang et al. “Hedera: A Permissionless and Scalable Hybrid Blockchain Consensus Algorithm in Multiaccess Edge Computing for IoT”. In: *IEEE Internet of Things Journal* 10.24 (2023), pp. 21187–21202. DOI: 10.1109/JIOT.2023.3279108.

[16] Celimuge Wu et al. “Spatial Intelligence toward Trustworthy Vehicular IoT”. In: *IEEE Communications Magazine* 56.10 (2018), pp. 22–27. DOI: 10.1109/MCOM.2018.1800089.

[17] Li Da Xu, Wu He, and Shancang Li. “Internet of Things in Industries: A Survey”. In: *IEEE Transactions on Industrial Informatics* 10.4 (2014), pp. 2233–2243. DOI: 10.1109/TII.2014.2300753.