And., Intl.,J., Appl Sci., Eng., Tech, Vol 06, No 03, November 2026
e-ISSN: 2797-0442

Energy Efficiency of Blockchain Consensus Algorithms for Resource-Constrained Networks: A Comprehensive Review

Hilary U. Ezea*, Uchechukwu R. Ezea, and Joseph Y. Oricha

Federal University Oye Ekiti, Nigeria

*Corresponding author, e-mail: hilary.ezea@fuoye.edu.ng

Abstract— Blockchain technology provides decentralized trust, transparency, and tamper resistance for distributed applications such as the Internet of Things (IoT), smart cities, and cyber-physical systems. However, the high computational and communication overhead of many consensus mechanisms, particularly mining-based approaches, results in excessive energy consumption that limits their deployment in resource-constrained networks. This paper presents a comprehensive review of blockchain consensus algorithms from an energy-efficiency perspective, with emphasis on their suitability for low-power IoT and wireless sensor environments. Consensus mechanisms are categorized into Byzantine fault-tolerant, non-Byzantine fault-tolerant, directed acyclic graph (DAG)–based, and hybrid families, and their operational characteristics are analyzed in terms of computation cost, message complexity, scalability, and power demand. To establish an analytical basis for evaluation, established IoT energy constraint models such as first-order radio, duty-cycling, battery lifetime, energy harvesting, and computation–communication trade-off models—are incorporated to assess protocol behavior under realistic resource limitations. Comparative analysis indicates that computation-intensive schemes such as mining-based protocols are unsuitable for energy-limited devices, whereas leader-based, lightweight, and structured-communication mechanisms significantly reduce power consumption while preserving reliability and security. The study concludes with design guidelines and recommendations to support the selection and development of sustainable consensus strategies for blockchain-enabled IoT systems.

Keywords: Blockchain, Consensus, and energy-constrained networks3.

This article *is licensed under the CC–BY-SA license.*

1. Introduction

Blockchain technology, recognized for its decentralized and tamper-resistant architecture, has become a transformative enabler across domains including Internet of Things (IoT) security, digital rights management, smart contracts, and smart cities [1]. Originally developed as the foundation of Bitcoin, it has since evolved into a versatile platform supporting decentralized applications, supply-chain transparency, and digital identity management. However, despite these benefits, blockchain adoption faces a major drawback: its substantial energy demand, which raises serious environmental and sustainability concerns [2]. The technology has continued to gain popularity both in digital storage and e-commerce where secure and efficient electronic transactions take place. It operates as a distributed data structure in which data is stored in cryptographically connected blocks in such a way that data records are made transparent, tamper-resistant and verifiable without the need of a centralized control. Through decentralizing transaction validation and decision-making, it has fundamentally reshaped the management of digital transactions by improving transparency and strengthening data security [2]. Blockchain's inherent characteristics such as fostering trust, ensuring data privacy, and enhancing transparency, make it a practical and secure platform for enabling the exchange of transactions and information among participants [3]. Its popularity revolves around its ability to provide a decentralized platform for highly secured electronic transactions. The technology finds

Received: March 15, 2026, Revised: April 19, 2026, Accepted: October 9, 2026
https://doi.org/10.25077/aijaset.v6i3.338

application in various fields such as medical field, financial field, wireless sensor networks, internet of things, privacy protection, property rights protection, etc, where security of data and transactions is of great importance [4].

At the core of the integrity of blockchain systems are the consensus algorithms which allow the distributed nodes in a network to reach an agreement on a single consistent state of data or transaction history [5] [6]. In other words, consensus algorithm is the decision-making mechanism which ensures that all the nodes share the same transactional information or data without relying on any central node. Consequently, these algorithms solve trust issues among participating nodes in the open network, thereby preventing any form of malicious interference [6]. Early consensus research in computer science focused on distributed consistency - ensuring that all nodes in a distributed system maintain identical data states and reliably reach agreement on proposed decisions through coordinated communication [7]. A functional diagram of the blockchain technology is presented in Figure 1 below:

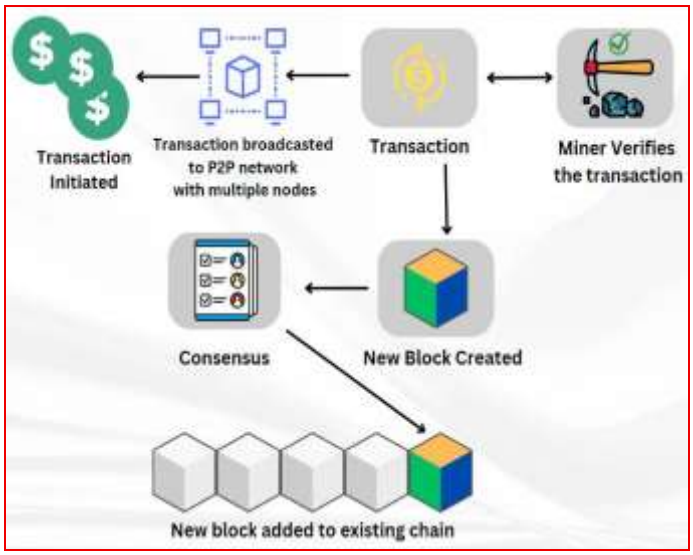


Figure 1. Functional diagram of a typical Blockchain technology [8]

Despite the extensive body of literature on blockchain consensus mechanisms, most existing surveys primarily emphasize security, scalability, and architectural classifications, with limited attention to energy efficiency in resource-constrained environments such as IoT and wireless sensor networks. Moreover, most prior works lack a unified analytical framework that integrates realistic energy consumption models with consensus algorithm performance evaluation. As a result, there remains a critical gap in understanding how different consensus mechanisms behave under strict energy, computation, and communication constraints. This paper distinguishes itself by integrating established IoT energy consumption models with consensus algorithm analysis, thereby providing a unified framework for evaluating energy efficiency in resource-constrained environments. Specifically, this study (i) develops an energy-centric evaluation model, (ii) performs structured comparative analysis across consensus families, and (iii) derives design guidelines tailored to low-power IoT deployments.

2. Classification of consensus families

The consensus algorithms can be categorized broadly into four: Non-Byzantine Fault-Tolerant Consensus algorithms, Byzantine Fault-Tolerant Consensus algorithms, Directed Acyclic Graphs (DAG)- based algorithms and Hybrid Consensus algorithms [4]. Each of these categories shown in Figure 2 has lots of member algorithms which operate based on the underlying operational principles of each category. Each consensus mechanism defines a distinct policy for distributing participating rights and decision-making power among the nodes to ensure fairness, security and decentralization [6].

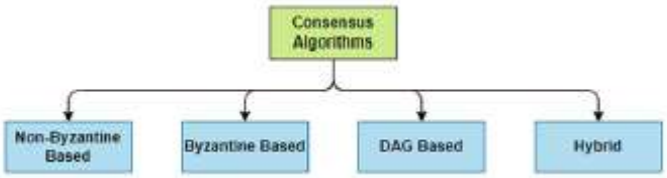


Figure 2. Categories of consensus algorithms [4]

2.1 Byzantine Fault-Tolerant Consensus algorithms

For this category of consensus algorithms, nodes repeatedly communicate, verify and vote on a proposed value until a decision is reached. These algorithms derive from byzantine generals' problem which models a distributed system where there are tendencies of some of the participating nodes to send conflicting messages, collude, act maliciously, lie or crash. Consequently, decisions are reached through a quorum-based policy in which sufficiently large majority of nodes agree on a value thereby overriding the influence of the minority faulty or malicious nodes [4]. This category includes the following algorithms:

2.1.1 Proof of Work (PoW)

The Proof of Work (PoW) algorithm was originally developed as a countermeasure against spam and denial-of-service attacks, requiring a sender to expend significant computational resources to solve a security-related mathematical problem. In the context of blockchain systems, PoW functions as a consensus mechanism in which participating nodes compete to solve a computationally intensive cryptographic puzzle to obtain the right to append the next block to the chain. The first node to discover a valid solution broadcasts the proposed block, after which other nodes promptly verify the accuracy and integrity of the contained information before collectively accepting and adding it to the blockchain [6]. By deliberately making block generation computationally costly, PoW deters malicious actors and strengthens network security. Although this approach provides strong protection and robustness, it is also associated with substantial energy consumption [9].

2.1.2 Delegated Proof of Work (DPoW)

Delegated Proof of Work (DPoW) is a consensus algorithm designed to enhance the energy efficiency and security of blockchain systems while addressing the limitations of traditional Proof of Work (PoW) [9]. It combines PoW's cryptographic robustness with a delegated or hierarchical validation structure to improve scalability and performance. Rather than allowing all nodes to engage in energy-intensive mining, the network elects or appoints a limited set of trusted delegates to carry out the PoW process on behalf of others. These delegates generate blocks through computational work, while the rest of the network verifies and validates the outcomes. By confining mining responsibilities to selected nodes, DPoW reduces energy consumption and communication overhead while maintaining the attack resistance of conventional PoW. Delegates remain accountable and can be replaced if they behave maliciously or inefficiently, introducing a governance mechanism. Although this approach balances security and efficiency, it may introduce partial centralization due to reliance on a smaller group of block producers.

2.1.3 Proof of Stake (PoS)

The principle underlying Proof of Stake (PoS) differs from Proof of Work in that nodes (miners) do not compete for block generation rights by solving computationally intensive puzzles. Instead, the authority to create and validate new blocks is determined by the amount of cryptocurrency a node has committed, or staked, in the network [9][6]. Participants, known as validators, deposit coins as collateral to demonstrate commitment to the system. A validator is then selected, typically at random, to propose the next block, with the probability of selection proportional to the size of each participant's stake. Notably, while the protocol incentivizes honest behavior through rewards, dishonest or malicious actions attract penalties, including partial or total loss of the staked assets

(slashing). Consequently, PoS achieves network security through economic risk rather than computational expenditure.

2.1.4 Delegated Proof of Stake (DPoS)

Delegated Proof of Stake (DPoS) is a modification of the PoS in which token holders elect a small group of trusted representatives (delegates) and assign their voting power to them to create new blocks and validate transactions on the blockchain [11]. These delegates are chosen by the stakers through a voting process, and the number of votes a delegate receives is determined by the number of tokens they hold [4]. The delegates with the most votes are responsible for creating new blocks and validating transactions, while the remaining delegates act as backups [12][13]. In other words, consensus is achieved through stake-weighted voting and representative validation. DPoS is intended to be more efficient and scalable than other consensus algorithms such as Proof-of-Work (PoW) or Proof-of-Stake (PoS), as it allows a small group of delegates to handle the majority of the network's transactions. It also gives token holders more influence over the network, as they can vote for or against specific delegates. However, on the downside, DPoS requires active and well-coordinated community participation for effective governance and voting, but its heavy reliance on sustained and informed engagement which often is difficult to ensure in practice, introduces non-technical risks that can significantly impact the blockchain's operation. It can also be more centralized as the power to create new blocks and validate transactions is held by a small group of delegates, making it susceptible to collusion and manipulation [14].

2.1.5 Practical Byzantine Fault Tolerance (PBFT)

Practical Byzantine Fault Tolerance (PBFT) is a consensus algorithm that enables distributed systems to achieve agreement despite the presence of faulty or malicious (Byzantine) nodes. Instead of relying on computational puzzles or economic incentives, it employs coordinated message exchange and majority voting. A designated primary (leader) proposes client requests or blocks, while replica nodes validate them through three sequential phases—pre-prepare, prepare, and commit—to ensure consistency and correctness. Consensus is reached when at least two-thirds of the nodes agree, requiring (3f + 1) nodes to tolerate up to (f) Byzantine faults.

Originally proposed by Castro and Liskov in the late 1990s, PBFT reduces the complexity of earlier Byzantine fault-tolerant approaches from exponential to polynomial, making it practical for real-world deployment. Its deterministic design provides low latency and high throughput, which makes it well suited to permissioned, consortium, or private blockchains and other resource-constrained environments. However, because communication overhead grows quadratically with the number of nodes, its scalability is limited for large public networks [10]

2.1.6 Proof of Authority (PoA)

Proof of Authority (PoA) is a consensus algorithm designed to optimize transaction validation in permissioned networks. Unlike PoS, which selects block producers based on cryptocurrency stake, PoA assigns block validation rights to a small set of pre-approved, identifiable nodes (authorities or validators), based on their real-world identity or reputation staked in the network [11][5]. Prospective validators undergo a notarization process linking their on-chain and off-chain identities, while existing authorities can vote to admit additional participants. Validators are rewarded for certifying and ordering transactions, incentivizing honest behavior. Because consensus is reached by a limited number of trusted nodes rather than computational work or coin staking, PoA achieves low latency, high throughput, and minimal energy consumption. This identity-at-stake approach aligns well with enterprise or consortium blockchains, where trustworthiness and reputation are critical. However, the model inherently centralizes authority, which can reduce censorship resistance and decentralization. In open public networks, centralization may allow authorities to blacklist or deny transactions, creating potential trust issues. Nevertheless, in controlled environments, PoA offers fast, deterministic, and energy-efficient consensus.

2.1.7 Proof of Burn (PoB)

Proof of Burn is a consensus mechanism in which participants gain the right to mine or validate blocks by permanently "burning" cryptocurrency, sending it to an address from which it cannot be retrieved [12]. By sacrificing coins, validators demonstrate long-term commitment to the network, aligning incentives with the blockchain's success while avoiding the high computational costs of Proof-of-Work or the collateral requirements of Proof-of-Stake. In PoB, the probability of being selected to create the next block increases with the amount of cryptocurrency burned. Validators may burn either the native currency of the blockchain or tokens from another chain, such as Bitcoin, to improve their chances of selection. This short-term sacrifice signals dedication to the network and offers potential long-term rewards [13]. To prevent early participants from gaining disproportionate advantage, PoB incorporates mechanisms that periodically adjust or decrease the impact of previously burned coins, creating a deflationary effect as the total currency supply gradually diminishes, potentially increasing its value over time. PoB provides a secure, energy-efficient alternative to traditional PoW systems, but it can be slower in reward cycles and is sometimes criticized as "wasteful" since the burned coins are permanently destroyed. The mechanism is particularly useful for bootstrapping new networks and mitigating Sybil attacks while preserving blockchain security in a cost-effective manner.

2.1.8 Proof of Space (PoSpace)

Proof of Space, also called Proof of Capacity, is a consensus mechanism in which participants demonstrate their eligibility to create new blocks by allocating unused hard drive storage instead of performing intensive computations (as in PoW) or staking coins (as in PoS) [4][14]. Miners plot their storage with precomputed solutions to cryptographic challenges. When a new block is to be created, the network selects the miner whose stored solution is closest to the challenge, allowing computations to be reused rather than discarded. PoSpace is energy-efficient because it relies on low-power storage rather than continuous computation, making it a suitable choice for environmentally-conscious blockchain networks. It is also highly decentralized, but may be vulnerable to monopolization if a participant controls a large portion of available storage. Security is maintained through the difficulty of producing fraudulent proofs and the inherent randomness in block selection, helping to preserve trust and fairness in the network.

2.1.9 Proof of Authentication (PoAuth)

Proof of Authentication (PoAuth) is a consensus mechanism in which only nodes with verified identities are permitted to participate in block validation. Instead of relying on computational effort, as in Proof of Work, or economic stake, as in Proof of Stake, PoAuth establishes trust through cryptographic authentication, access control, and permissioned participation. Nodes must first authenticate themselves using digital credentials, public–private key pairs, or certification by trusted authorities before they can propose or validate blocks. Once authenticated, approved nodes collaboratively validate transactions and reach consensus through coordinated verification or voting. Because participation is restricted to known and trusted entities, PoAuth achieves low latency, high throughput, and minimal energy consumption, making it well suited for private, enterprise, or consortium blockchain environments. However, its dependence on identity management introduces centralization and reduced openness compared with fully decentralized approaches.

Proof-of-Authentication has also been proposed as a lightweight consensus solution for resource-constrained devices [15]. In this model, nodes generate transactions and assemble them into blocks, which are digitally signed using private keys and broadcast to the network. Authentication relies on asymmetric cryptography (e.g., ElGamal), where public keys are used to verify signatures while private keys remain secret. Designated trusted nodes validate block authenticity by checking signatures and message authentication codes (MACs), and only authenticated blocks are propagated and appended to the chain. This process minimizes computational overhead while maintaining integrity and security, making PoAuth suitable for scalable and low-power blockchain deployments.

2.1.10 Proof of Importance

Proof of Importance (PoI) is a consensus approach that determines block validators using both stake ownership and the participant's actual involvement in network operations [11]. First deployed by the NEM blockchain, it builds on Proof of Stake concepts but goes further by incorporating activity-based indicators so that influence depends on meaningful participation rather than simply holding tokens. Eligibility to create blocks - called *harvesting* in NEM, requires a minimum amount of vested tokens (e.g., 10,000 XEM). Once this threshold is met, the system assigns each account an importance score, which represents its relative contribution to the ecosystem. This score is computed heuristically from three main elements: (i) the amount and duration of token holdings, (ii) net outbound transactions that encourage circulation of funds and discourage self-transfer manipulation, and (iii) the degree of interaction with other nodes, reflecting broader and more consistent network engagement.

Accounts with higher importance scores are more likely to be selected for block production and to earn transaction fees. By coupling economic stake with transactional activity and connectivity, PoI promotes fairness, liquidity, and decentralization. Nevertheless, accurately defining and tuning the scoring metrics introduces additional complexity and may affect transparency if not carefully designed.

2.2 Non-Byzantine Fault-Tolerant Consensus algorithms

For this category of consensus algorithms, nodes may stop working or send incorrect data but such error is not intended. Such error is classified as non-byzantine error which describes a simple, predictable or well-behaved failure in a distributed network where a node fails without malicious or inconsistent behaviour. Non-Byzantine Fault-Tolerant Consensus algorithms are therefore designed to tolerate non-byzantine errors, although they cannot guarantee data security and system stability [6][4]. Consequently, these consensus algorithms are usually used in closed environment where the risk associated with open networks characterized by diverse nodes and low credibility, is minimal. The algorithms under this category include:

2.2.1 Paxos consensus algorithm

The operational principle of the Paxos consensus algorithm centers on enabling a group of distributed nodes to agree on a single proposed value despite node failures or message delays [16]. Rather than relying on a central authority, Paxos ensures that all correct nodes ultimately decide on the same value, and that this value must have been legitimately proposed, even in the presence of partial system failures [4].

Within Paxos, nodes may assume multiple logical roles, including proposers, acceptors, and learners. Proposers generate candidate values for agreement, acceptors evaluate and vote on these proposals, and learners are responsible for determining the final chosen value. A proposal is considered selected once it is accepted by a majority quorum of acceptors, after which learners are notified to update their state accordingly. Although Paxos offers strong consistency guarantees and computational efficiency under crash-fault assumptions, its applicability is generally limited to permissioned or private environments that do not account for malicious (Byzantine) behavior [6][4].

2.2.2 Multi-Paxos algorithm

Multi-Paxos algorithm is a very important variant of the Paxos consensus protocol which improves efficiency when distributed services require a continuous sequence of operations [17]. Unlike the basic Paxos algorithm, where each consensus instance determines a single proposed value independently, Multi-Paxos reaches agreement on an ordered sequence of values across successive instances.

The enhanced efficiency of Multi-Paxos stems primarily from its one-time leader election mechanism. Once a leader is established, it can coordinate multiple subsequent proposals without repeating the full two-phase negotiation required by basic Paxos. As a result, later consensus decisions typically complete in a single communication round rather than two, reducing round-trip delays and message overhead, and thereby increasing overall system throughput.

2.2.3 WPaxos consensus algorithm

WPaxos is a distributed consensus protocol specifically designed to improve latency and throughput in Wide-Area Network (WAN) environments, where communication delays vary significantly across geographic regions [18]. To address these variations, WPaxos supports multiple concurrent leaders, with each region served by a local leader, enabling faster request processing and reducing cross-region communication overhead.

Furthermore, WPaxos adopts the concept of flexible quorums, allowing the accept phase to be completed locally using a smaller subset of acceptors within a leader's jurisdiction. This design minimizes the need for wide-area coordination and substantially lowers WAN communication costs [19]. Another notable feature of WPaxos is its adaptive dynamic partitioning mechanism, through which leadership over data objects is reassigned based on workload patterns. By dynamically adjusting object ownership among leaders, the protocol balances load and further enhances performance.

2.2.4 Viewstamped Replication (VR) Consensus algorithm

The VR consensus algorithm is designed to maintain consistency and fault tolerance across a set of distributed replicas by combining state replication with leader-based agreement. In essence, VR performs coordinated log replication among nodes to ensure that system state remains consistent across all replicas [6]. It guarantees that every replica executes the same operations in the same order, as determined by the primary node, even in the presence of node failures.

The primary node establishes the global ordering of client requests, which is subsequently accepted and applied by the backup replicas. To ensure continuity under failures, VR incorporates a recovery mechanism known as a view change. When the primary node becomes unavailable, the remaining nodes collaboratively elect a new primary. The newly selected primary resumes operation using the replicated log and view numbers inherited from the previous configuration, thereby preserving both ordering and consistency guarantees [4]. Although VR effectively tolerates node crashes and network partitions while maintaining strong consistency and safety properties - making it suitable for fault-tolerant distributed systems, its scalability remains limited.

2.3 RAFT

RAFT is a leader-based distributed consensus algorithm that uses the log replication of each node to achieve unification of transactions and consensus [6]. In the RAFT consensus algorithm, participating nodes assume one of two primary roles: leader or follower [20]. The leader coordinates the system by accepting client requests, proposing new log entries that represent state changes, and managing their replication across the cluster. Followers maintain replicated logs, respond to the leader's requests, and enforce protocol rules to ensure consistency.

Consensus proceeds through a leader-driven log replication process. The leader broadcasts proposed entries to followers, which validate and append them to their local logs. Once a majority of followers acknowledge an entry, it is considered committed and becomes durably applied to the system state. The leader then finalizes the commitment and ensures all replicas eventually converge to the same state.

To maintain fault tolerance, RAFT incorporates an automated leader election mechanism. If the current leader fails or becomes unreachable, followers initiate an election to select a replacement. The candidate with the most up-to-date and complete log is preferred, ensuring that the new leader preserves previously committed entries and maintains system safety.

2.4 Directed Acyclic Graph (DAG)- based consensus algorithm

DAG-based consensus algorithm is a distributed consensus method where transactions or blocks are organized in a directed acyclic graph and agreement is reached by referencing, validating or voting on multiple past events instead of extending a single chain. While the transactions are represented as nodes within the DAG, the interrelationship between the transactions are represented by the edges between the nodes [21]. One of the main advantages of DAG-based consensus algorithms over conventional blockchain-based counterparts is their relative flexibility and fast processing of transactions.

2.5 Hybrid consensus mechanisms

Hybrid consensus mechanisms emerged as innovative solutions to address the observed limitations of the traditional consensus mechanisms. These algorithms integrate two or more consensus mechanisms to exploit their complementary strengths, thereby improving transaction and block verification while reducing individual limitations. [29][30]. These hybrid algorithms blend mechanisms in parallel or sequentially to achieve better security, scalability, energy efficiency. However, hybrid algorithms also inherit the computational complexity and energy demands of their constituent mechanisms, resulting in performance and efficiency trade-offs during deployment

2.5.1 Proof-of-Activity (PoAct)

Proof-of-Activity is a hybrid consensus mechanism that combines aspects of Proof-of-Work (PoW) and Proof-of-Stake (PoS) to secure a blockchain while improving energy efficiency. In PoAct, miners first use computational power to create an initial “empty” block header, similar to PoW. Once the block header is mined, the system transitions to a PoS phase, where validators are randomly selected to sign and complete the block. The probability of selection is proportional to the amount of cryptocurrency staked, prioritizing participants with a long-term interest in the network [4][22].

A block is finalized and added to the blockchain once the required number of validator signatures is obtained, with rewards distributed to both the initial miner and contributing validators. By combining computational work and stake-based validation, PoAct mitigates risks such as 51% attacks while balancing security and energy efficiency. While more complex to implement and potentially less secure than pure PoW or PoS, PoAct effectively aligns incentives with network longevity and reduces wasteful energy expenditure [23].

2.6 Comparative Analysis of Consensus Algorithms

Table 1 shows the comparative performance of the consensus algorithms in terms of computational cost, communication complexity, energy consumption profile, scalability and suitability for resource-constrained networks. The evaluation metrics used are defined as follows:

i. **Energy per transaction (J/tx):** Energy per transaction in consensus algorithms denotes the average amount of electrical energy, typically measured in joules, required to validate and permanently record a single transaction on a blockchain. It serves as a key indicator of the environmental sustainability and energy efficiency of distributed ledger systems [24].
ii. **Communication complexity ($O(n^2)$, $O(n)$):** This refers to the total amount of information exchange among participating nodes in a consensus process, commonly quantified by the number of messages transmitted (required) to reach agreement. Communication complexity reflects the trade-off between communication overhead and the speed or efficiency of consensus formation.

iii. **Computational cost:** The computational cost of a consensus algorithm represents the aggregate resources required to achieve agreement among distributed nodes, including processing power, execution time, energy consumption, and communication bandwidth involved in validating transactions and forming new blocks.

iv. **Scalability:** Scalability in consensus algorithms describes the capability of a blockchain system to sustain or improve its performance – particularly with respect to transaction throughput and confirmation latency, as the number of participating nodes increases, without causing significant degradation in efficiency[25] .

Table 1. Comparative performance evaluation of the consensus mechanisms

Algorithm	Computation cost	Energy Consumption	Communication Complexity	Scalability	Suitability for resource-constrained networks
PoW	Very high	Very high	Low	Low	Not suitable
DPoW	Low	Reduced	Reduced	High	Suitable
PoS	Low	Low	Low	High	Suitable
DPoS	Low	Very low	Very low	High	Suitable
PBFT	Low	Very high	Very high	Low	Not suitable
PoA	Very low	Very low	Low	High	Highly suitable
PoB	Low	Low	Moderate	Low	Limited
PoSpace	High	Very High	High complexity	Low	Not suitable
PoAuth	Low	Very low	Low complexity	High	Highly suitable
PoI	Low	Low	High complexity	High	Not suitable
Paxos	Low	Low	Moderate	Moderate	Suitable
Multi-Paxos	High	Low	Reduced complexity	Moderate	Limited
WPaxos	Moderate	Reduced	Low	High	Not suitable
VR	Low	Low	Moderate	Moderate	Not suitable
RAFT	Low	Low	Moderate	Moderate	Suitable
DAG-based	Moderate	Low	Low	Moderate	Suitable
PoAct	Moderate	Moderate	Moderate	High	Conditional

From the table, while algorithms such as the PoA and PoAuth are highly suitable for resource-constrained networks such as WSN and IoT networks owing to their low power consumption profiles, low communication overhead, and low computational requirements, PoW, PoSpace, PoI, WPaxos, and VR algorithms are not suitable for such networks.

2.6.1 Energy consumption profiles of the consensus algorithms

The energy consumption profiles of the different algorithms were evaluated using simulation of 100, 1000, and 10,000 network nodes in a c++ environment. The simulation parameters for the first scenario include power per node (50W), consensus time (2s), message cost (0.05J), number of messages (50), PoW hash rate factor (51), PoSpace storage factor (10^9 – to accommodate large number of participants), number of DPoS delegates (51 – for reduced risk and balanced performance). On the other hand, the simulation parameters for the second scenario include power per node (25W), consensus time (1.6s), with message cost, number of messages, PoW hash rate factor, PoSpace storage factor, and number of DPoS delegates retaining the same values as in the first scenario.

Table 2 shows the results of simulation carried out to evaluate the energy consumption profiles of the consensus mechanisms.

Table 2. Energy consumption profiles of the consensus mechanisms

	Ist scenario			**2nd scenario**		
	100 Nodes	**1000 Nodes**	**10000 Nodes**	**100 Nodes**	**1000 Nodes**	**10000 Nodes**
PoW	510128	5.10E+06	5.10E+07	204128	2.04E+06	2.04E+07
DPoW	7001.75	70001.8	700002	2801.75	28001.8	280002
PoS	3000.75	30000.8	300001	1200.75	12000.8	120001
DPoS	5102.5	5102.5	5102.5	2042.5	2042.5	2042.5
PBFT	10502.5	150002	6.00E+06	4502.5	90002.5	5.40E+06
PoA	2000.5	20000.5	200000	800.5	8000.5	80000.5
PoB	5001.25	50001.2	500001	2001.25	20001.2	200001
PoSpace	1.00E+13	1.00E+14	1.00E+15	4.00E+12	4.00E+13	4.00E+14
PoAuth	2500.62	25000.6	250001	1000.62	10000.6	100001
PoI	3500.88	35000.9	350001	1400.88	14000.9	140001
Paxos	10012.5	100102	1.00E+06	4012.5	40102.5	401002
Multi-Paxos	10007.5	100052	1.00E+06	4007.5	40052.5	400502
Wpaxos	8002	80002	800002	3202	32002	320002
VR	10010	100078	1.00E+06	4010	40077.5	400752
RAFT	10007.5	100052	1.00E+06	4007.5	40052.5	400502
DAG-based	6001.5	60001.5	600002	2401.5	24001.5	240002
PoAct	6501.62	65001.6	650002	2601.62	26001.6	260002

The simulation result shown in Table 2 aligns with the contents of Table 1. Table 2 reveals that delegation-based and authority-driven consensus mechanisms (e.g., DPoS, PoA) exhibit superior energy efficiency and scalability compared to mining-based protocols such as PoW. This could be attributed to the fact that the delegation-based algorithms remove redundant participation thereby reducing drastically, energy consumption. Classical consensus algorithms like Paxos and RAFT demonstrate stable linear energy growth and as such, are predictable, while PBFT suffers from exponential energy escalation due to communication overhead. Notably, PoSpace shows prohibitively high energy consumption due to energy tied to storage operations, rendering it impractical for energy-constrained deployments.

3. Energy/IoT constraints models

These models describe analytical frameworks used to characterize and manage the limited resources of Internet of Things (IoT) devices - particularly energy, computation, memory, bandwidth, and latency. These constraint models are employed in designing communication protocols, consensus mechanisms, and task allocation strategies that operate efficiently under strict resource budgets. At the core, energy models quantify how sensing, processing, transmission, and idle states consume power, often expressing total energy as the sum of computation energy and communication energy, with wireless transmission typically dominating consumption. IoT constraint models also account for limited battery capacity, intermittent energy harvesting, low processing capability, small storage, and unreliable connectivity. Consequently, lightweight algorithms, low-complexity cryptography, reduced message exchanges, and duty-cycling techniques are emphasized to prolong network lifetime.

Such models guide the development of energy-aware routing, scheduling, and consensus schemes, particularly in blockchain-enabled IoT systems- ensuring that security and functionality are achieved without exceeding device capabilities.

3.1 First Order Radio Model

The First-Order Radio Model is widely employed to estimate the energy consumed during data transmission and reception in wireless sensor networks [26]. It characterizes the energy required to send and receive packets between neighboring nodes or a base station [27]. The model assumes symmetric transmission energy for a given signal-to-noise ratio (SNR), uniform sensing rates across nodes, and channel loss proportional to the square of the transmission distance (d^2).

A representative application of this model is the Low Energy Adaptive Clustering Hierarchical (LEACH) protocol, which organizes nodes into clusters based on received signal strength, with cluster heads serving as routers that forward aggregated data to the sink [28]. Although LEACH extends network lifetime through energy-aware clustering, its reliance on single-hop communication limits scalability. Cluster heads located far from the base station expend excessive energy, making the protocol less suitable for large-area deployments.

According to this model, the energy consumed to transmit a n-bit message over a distance d is given by [27]:

$$E_{Tx(n,d)} = E_{Tx-elec}(n) + E_{Tx-amp}(n,d) \quad (1)$$

$$E_{Tx(n,d)} = E_{elec} * n + \epsilon_{amp} * n * d^2 \quad (2)$$

Then energy consumed to receive a n-bit message is given by:

$$E_{Rx(n)} = E_{Rx-elec}(n) \quad (3)$$

$$E_{Rx(n)} = E_{elec} * n \quad (4)$$

Where n is the number of information bits, d is the distance between the communicating parties, $E_{Tx-elec}$ is the transmitter electronics, E_{Tx-amp} is the transmitter amplifier, E_{elec} is the energy consumed by the radio circuitry regardless of distance, ϵ_{amp} is the amplifier coefficient, d^2 represents path loss effect under free-space propagation

It could be seen in equations 3 and 4 that in this model, the receiver consumes less energy than the transmitter because the aspect of signal amplification seen at the transmitter does not apply as the receiver merely processes and forwards the received information [27].

3.2 Duty-cycling / sleep ↔ wake energy model

The Duty-cycling (sleep ↔ wake) energy model is an energy-management approach widely adopted in resource-constrained wireless sensor and IoT networks, where nodes periodically switch between active and low-power sleep states to extend battery life [29]. In this scheme, the radio, processor, and sensing units are enabled only during short, scheduled intervals for communication or computation, while the node remains in sleep or idle mode for the majority of the time.

During active periods, nodes listen, transmit, or receive data; during sleep periods, communication ceases and power consumption drops to a minimal level [28]. Each node wakes periodically to check for pending transmissions and returns to sleep if no activity is detected [30]. When network traffic is low, longer sleep durations are possible, leading to substantial energy savings. Conversely, increased traffic reduces sleep opportunities and raises energy usage due to more frequent transmissions.

Because radio transmission and idle listening dominate energy expenditure, minimizing active time, often by batching communications into bursts and sleeping between them, significantly reduces overall consumption. Consequently, total energy use is typically modeled as a weighted sum of sleep, idle, transmit, and receive states, with extended sleep intervals yielding the greatest savings. By lowering power demand, duty-cycling prolongs network lifetime and enhances sustainability, making

it particularly suitable for IoT, wireless sensor, and energy-harvesting systems. However, intermittent node availability may introduce latency, synchronization overhead, and reduced throughput. Duty cycling can assume adaptive pair generation mechanism in which two nodes are paired to be sleeping and waking in alternation, thereby achieving the needed security purpose complementarily with reduced energy expenditure [31]. Energy saving is defined in [30] as:

$$E_s = \frac{T_{sleep}}{T_{frame}} = 1 - \frac{T_{listen}}{T_{frame}} \quad (5)$$

Where T_{frame} is a complete cycle of listen and sleep.

3.3 Battery capacity and lifetime models

Battery capacity and lifetime models characterize how finite energy storage in resource-constrained devices (e.g., sensors, IoT nodes) determines operational longevity and influences protocol design. In these models, each node is assumed to have a limited battery capacity, and its lifetime is defined by how quickly that energy is consumed through sensing, computation, transmission, reception, and idle listening. The core idea is to relate total energy consumption to time until energy depletion, enabling design and optimization of energy-aware protocols such as duty-cycling, low-power routing, and lightweight consensus [32].

Energy consumption is often unevenly distributed across the network. For example, serving as a cluster-head requires additional communication, aggregation, and forwarding tasks, which drain the battery faster [33]. To prevent early node failure, cluster-head responsibilities are periodically rotated, and nodes with higher residual energy are preferentially selected. This rotation balances energy usage and prolongs overall network lifetime.

Battery lifetime is strongly affected by the discharge rate, that is, the current drawn from the battery. Exceeding the rated current capacity accelerates energy depletion and reduces effective lifetime because electrochemical processes cannot replenish active materials at the same rate they are consumed. Sustained high current therefore shortens usable battery life, even when charge remains. Consequently, minimizing peak current and frequently reducing the node to low-power or shutdown states can significantly extend lifetime. The radio subsystem typically dominates energy consumption and operates in four modes: transmit, receive, idle, and sleep. Idle listening often consumes nearly as much power as reception [34], making it inefficient to remain idle when no communication occurs. Switching to sleep mode greatly reduces power draw, although state transitions introduce small startup costs. Hence, maximizing sleep duration and minimizing unnecessary radio activity are key to conserving energy.

In energy-constrained networks, node energy use is commonly modeled as the sum of power consumed in distinct operating states:

$$E_{Total} = P_{sleep}t_{sleep} + P_{idle}t_{idle} + P_{tx}t_{tx} + P_{rx}t_{rx} + P_{cpu}t_{cpu} \quad (6)$$

Consequently, the average power consumption over a period of time, T, is:

$$P_{avg} = \frac{E_{Total}}{T} \quad (7)$$

It therefore follows that given a node with initial battery capacity - $E_{battery}$, the estimated lifetime of the node is:

$$N_{lifetime} = \frac{E_{battery}}{P_{avg}} \quad (8)$$

3.4 Energy-harvesting (EH) models

Energy-harvesting (EH) models describe networks in which nodes replenish energy from ambient sources (solar, thermal, vibration, or RF) rather than relying solely on finite batteries, whose limited capacity and frequent replacement increase maintenance cost and reduce operational lifetime [35] [36]. By converting environmental energy into electrical power through suitable transduction and management circuits, EH techniques enable long-term and potentially maintenance-free monitoring [37] [38]. Subject to an energy-causality constraint that permits operation only when sufficient energy is stored, these models relate harvested energy $E_{harvested}(t)$, storage limits E_{max}, and consumption dynamics $E_{consumed}(t)$ to node availability through an energy balance equation:

$$E(t+1) = \min\left(E_{max}, E(t) + E_{harvested}(t) - E_{consumed}(t)\right) \quad (9)$$

While EH can substantially prolong or sustain network lifetime, variability in energy supply may introduce intermittent operation and latency, necessitating adaptive power management.

3.5 Computation–communication trade-off model

These models address the fundamental tension between limited energy availability and desired functionality in resource-constrained IoT and sensor nodes, where form-factor limitations restrict active power, energy reserves, processing, and communication capabilities [44]. Because wireless transmission typically consumes far more energy than local computation, these models promote shifting workload from communication to processing through techniques such as compression, aggregation, filtering, and local decision-making to minimize transmitted data and overall energy use. For example, the hardware–software co-design framework proposed in [44] increases on-sensor computation, reduces communication volume, and maximizes inference completion in energy-harvesting WSNs by employing a store-and-execute strategy and task-aware coreset construction to transmit only compact features to a coordinating mobile host.

3.6 Comparative analysis of the energy models

Table 3 presents a structured comparative framework for evaluating consensus mechanisms under different energy-constraint models. The table consolidates the resource limitations captured by each model, their implications for energy-efficient operation, typical deployment scenarios, and the consensus protocols best suited to those conditions. This mapping provides a systematic basis for assessing the relative suitability of various algorithms for resource-constrained and IoT environments

Table 3. Comparative framework for evaluating consensus mechanisms

Constraint Model	What it Captures	Why Suitable for Energy-Efficient Consensus	Typical Consensus Use-Case	Consensus mechanisms
First-order radio energy model	Energy consumed for transmit/receive vs distance; communication dominates energy consumption	Consensus is message-heavy; modeling Tx/Rx cost helps minimize broadcast rounds and optimize message aggregation. Prefer algorithms with few messages, short rounds, small broadcasts	Clustered WSNs, IoT swarms, edge devices	Paxos, Multi-Paxos, WPaxos, Viewstamped Replication (VR), RAFT, PBFT , PoAh
Duty-cycling / sleep–wake model	Sleep, idle, active power states; Idle listening & always-on radios waste power	Allows nodes to sleep between consensus rounds, reducing idle listening and standby energy. Prefer leader-based or scheduled communication so nodes sleep most of the time	Battery-powered IoT, periodic sensing systems	RAFT, Multi-Paxos, WPaxos, PoA, DPoS, PoAh
Battery lifetime/capacity model	Long-term energy budget, fairness of consumption	Enables role rotation (leader/validator switching) to prevent early node death and extend	Long-lived deployments	RAFT, Paxos family, VR, PBFT, PoA, PoAh

		network lifetime; Preference for lightweight computation + predictable message load; avoid mining		
Computation – communication trade-off model	Balance CPU work vs message overhead	Guides use of lightweight cryptography and local aggregation instead of excessive communication or heavy hashing. Works when limited computation can reduce communication; avoid heavy cryptographic puzzles	Lightweight blockchain, IoT cryptography, signature batching, Edge/fog computing, gateways	PBFT, DAG-style consensus, Hybrid consensus, WPaxos, PoS/DPoS (light staking)
Energy harvesting model	Dynamic harvested energy and storage	Consensus participation can adapt to available energy; nodes with surplus energy take more responsibility	Solar/RF-powered IoT blockchains, remote deployments	DPoS, PoS, PoA, Hybrid consensus, DAG, WPaxos

The table summarizes how different energy-constraint models guide the selection of blockchain consensus mechanisms for resource-constrained networks by linking each model to the type of resource limitation it captures, its relevance to energy-efficient operation, and the most appropriate deployment scenarios. The first-order radio model emphasizes the dominance of communication energy over computation, motivating message-efficient protocols such as Paxos, Multi-Paxos, WPaxos, Viewstamped Replication, Raft, Practical Byzantine Fault Tolerance, and Proof of Authentication for clustered IoT and edge systems. The duty-cycling model captures sleep–wake behavior and highlights the benefits of leader-based or scheduled communication to minimize idle listening, favoring lightweight schemes such as Raft, Multi-Paxos, WPaxos, Proof of Authority, Delegated Proof of Stake, and Proof of Authentication for battery-powered devices. Battery lifetime models address long-term energy fairness and recommend role rotation and predictable workloads, again aligning with lightweight, non-mining protocols. The computation–communication trade-off model encourages shifting effort toward local processing and aggregation to reduce costly transmissions, supporting Practical Byzantine Fault Tolerance, DAG-style, hybrid, and light-staking approaches. Finally, energy harvesting models account for variable energy availability and promote adaptive participation, where nodes with surplus power assume greater responsibility, making delegated, stake-based, hybrid, or DAG mechanisms suitable for solar or remotely powered IoT deployments.

Consensus algorithms that demand high computation (continuous hashing and mining), therefore, are very poor candidates for IoT-based systems. Such algorithms drain battery quickly and require always-on hardware. The algorithms that fall into this category include Proof of Work, Proof of Burn, and Proof of Activity. On the other hand, algorithms with low computation as well as structured communication are energy-friendly and as such are very good candidates for energy-constrained networks. Algorithms within this Category include Paxos, Multi-Paxos, WPaxos, Viewstamped Replication, RAFT, Delegated PoS, Proof of Authority, Proof of Authentication, Proof of Importance, and PBFT (small network). These findings align with the energy-consumption profiles of some of the consensus algorithms as stated in [2]- with Proof of Work consuming energy up to 707.6 kWh per transaction while Delegated PoS and Proof of Authority consume 0.0015 kWh per transaction and 0.00022 kWh per transaction, respectively

4. Conclusion

This review demonstrates that the energy footprint of blockchain consensus mechanisms is the primary limiting factor for adoption in IoT and other resource-constrained environments. Mining-based approaches that rely on continuous hashing and intensive computation impose excessive power demands and rapidly deplete limited energy reserves, making them impractical for battery-powered or energy-harvesting devices. In contrast, consensus protocols characterized by lightweight computation, structured or leader-based communication, reduced message exchange, and adaptive participation offer substantially lower energy consumption while preserving reliability and security. Mechanisms such as Paxos-based protocols, RAFT, PBFT for small networks, Proof of Authority, and delegated or

authentication-based schemes exhibit strong potential for sustainable deployment. Integrating energy-aware models during protocol design enables balanced trade-offs between security, latency, and power efficiency. Overall, energy-efficient consensus selection is critical to enabling scalable, long-lived, and environmentally sustainable blockchain systems for IoT and edge networks.

Data availability

The c++ code that supports the findings of this study is available from the corresponding author upon reasonable request.

Acknowledgment

The authors acknowledge the encouragement given to them by Dr. Shereefdeen Sanni.

References

[1] G. Tripathi, M. Abdul, and G. Casalino, "A comprehensive review of blockchain technology : Underlying principles and historical background with future challenges," *Decis. Anal. J.*, vol. 9, no. October, p. 100344, 2023, doi: 10.1016/j.dajour.2023.100344.

[2] M. Pineda, D. Jabba, W. Nieto-bernal, and A. Perez, "Sustainable Consensus Algorithms Applied to Blockchain : A Systematic Literature Review," *Sustainability*, vol. 16, no. 23, p. 10552, 2024, doi: https://doi.org/10.3390/su162310552.

[3] T. Pawar, J. Rao, and P. Patil, "Cyber-shield machine learning : an intrusion detection system with blockchain-powered secure log storage across network nodes," *Sigma J. Eng. Nat. Sci.*, vol. 43, no. 3, pp. 714–725, 2025, doi: 10.14744/sigma.2025.000.

[4] Z. Hussein, M. A. Salama, and S. A. El Rahman, "Evolution of blockchain consensus algorithms : a review on the latest milestones of blockchain consensus algorithms," *Cybersecurity*, vol. 6, no. 30, 2023, doi: 10.1186/s42400-023-00163-y.

[5] D. Çatal and M. K. Ulukök, "A Comparative Evaluation of Blockchain Consensus Algorithms in Terms of Energy Efficiency and Performance Metrics," in *International Symposium on AI-Driven Engineering Systems (ISADES2025)*, Tokat, Türkiye: SETSCI, 2025, pp. 132–135. doi: https://doi.org/10.36287/setsci.22.33.001.

[6] H. Xiong, M. Chen, C. Wu, Y. Zhao, and W. Yi, "Research on Progress of Blockchain Consensus Algorithm : A Review on Recent Progress of Blockchain Consensus Algorithms," *Futur. Internet*, vol. 14, no. 2, p. 47, 2022, doi: https://doi.org/10.3390/fi14020047.

[7] Z. Fan, "Research on Blockchain Hybrid Consensus Algorithm Based on Internet of Things," *Int. J. Internet Things Big Data*, vol. 4, no. 1, pp. 31–44, 2019, doi: http://dx.doi.org/10.21742/ijtbd.2019.4.1.05.

[8] A. A. Monrat, O. Schel, and K. Andersson, "A Survey of Blockchain from the Perspectives of Applications , Challenges and Opportunities," *IEEE Access*, vol. 7, pp. 117134–117151, 2019, doi: 10.1109/ACCESS.2019.2936094.

[9] M. Kara *et al.*, "A Novel Delegated Proof of Work Consensus Protocol," in *2021 International Conference on Artificial Intelligence for Cyber Security Systems and Privacy (AI-CSP)*, El Oued, Algeria: IEEE, 2021, pp. 1–7. doi: 10.1109/AI-CSP52968.2021.9671096.

[10] X. Zheng and W. Feng, "Research on Practical Byzantine Fault Tolerant Consensus Algorithm Based on Blockchain," *J. Phys. Conf. Ser. Pap.*, 2021, doi: 10.1088/1742-6596/1802/3/032022.

[11] P. Zhang, D. C. Schmidt, J. White, and A. Dubey, *Consensus mechanisms and information security technologies*, 1st ed. Elsevier Inc., 2019. doi: 10.1016/bs.adcom.2019.05.001.

[12] K. Karantias, A. Kiayias, and D. Zindros, "Proof-of-Burn," in *24th International Conference on Financial Cryptography and Data Security 2020*, J. Bonneau and N. Heninge, Eds., Kota Kinabalu, Malaysia: Springer, 2020, pp. 523–540. doi: 10.1007/978-3-030-51280-4_28.

[13] J. Yusoff, Z. Mohamad, and M. Anuar, "A Review : Consensus Algorithms on Blockchain," *J. Comput. Commun.*, vol. 10, pp. 37–50, 2022, doi: 10.4236/jcc.2022.109003.

[14] M. F. Stollenga, "Embracing Hellman : A Simple Proof-of-Space Search consensus algorithm

with stable block times using Logarithmic Embargo," *Cryptol. ePrint Arch.*, pp. 1–16, 2022.
[15] D. Puthal, S. P. Mohanty, P. Nanda, E. Kougianos, and G. Das, "Proof-of-Authentication for Scalable Blockchain in Resource-Constrained Distributed Systems," in *2019 IEEE International Conference on Consumer Electronics (ICCE)*, Las Vegas, NV, USA: IEEE, 2019, pp. 1–5. doi: https://doi.org/10.1109/ICCE.2019.8662009.
[16] L. Lamport, "Paxos Made Simple," *ACM SIGACT News (Distributed Comput. Column)*, vol. 32, no. 4, pp. 51–58, 2001.
[17] S. Chand, Y. A. Liu, and S. D. Stoller, "Formal Verification of Multi-Paxos for Distributed Consensus," in *Lecture Notes in Computer Science 9995*, Springer, 2019, pp. 119–136.
[18] A. Ailijiang, A. Charapko, M. Demirbas, and T. Kosar, "WPaxos : Wide Area Network Flexible Consensus," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 1, pp. 211–223, 2020, doi: https://doi.org/10.1109/TPDS.2019.2929793.
[19] H. Howard, D. Malkhi, and A. Spiegelman, "Flexible Paxos : Quorum Intersection Revisited," in *42nd Conference on Very Important Topics (CVIT 2016)*, Q. John and R. Joan, Eds., 2016, pp. 1–17. doi: 10.4230/LIPIcs.CVIT.2016.23.
[20] D. Kim, I. Doh, and K. Chae, "Improved Raft Algorithm exploiting Federated Learning for Private Blockchain performance enhancement," in *2021 International Conference on Information Networking (ICOIN)*, Jeju Island, Korea (South): IEEE, 2021, pp. 828–832. doi: https://doi.org/10.1109/ICOIN50884.2021.9333932.
[21] T. Chen, W. Huang, P. Kuo, H. Chung, and T. Chao, "DEXON : A Highly Scalable , Decentralized DAG-Based Consensus Algorithm," 2018. doi: https://doi.org/10.48550/arXiv.1811.07525.
[22] M. Kaur, M. Z. Khan, S. Gupta, A. Noorwali, C. Chakraborty, and S. K. Pani, "MBCP : Performance Analysis of Large Scale Mainstream Blockchain Consensus Protocols," *IEEE Access*, vol. 9, pp. 80931–80944, 2021, doi: 10.1109/ACCESS.2021.3085187.
[23] B. Shrimali and H. B. Patel, "Blockchain state-of-the-art : architecture , use cases , consensus , challenges and opportunities," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 34, no. 9, pp. 6793–6807, 2022, doi: 10.1016/j.jksuci.2021.08.005.
[24] C. Stoll, L. Klaaßen, and U. Gallersdorfer, "The Carbon Footprint of Bitcoin," *Joule*, vol. 3, pp. 1647–1661, 2019, doi: 10.1016/j.joule.2019.05.012.
[25] M. Vukoli, "The Quest for Scalable Blockchain Fabric : Proof-of-Work vs . BFT Replication," in *Open Problems in Network Security. iNetSec 2015. Lecture Notes in Computer Science(), vol 9591*, J. Camenisch and D. Kesdo, Eds., Springer, 2016, pp. 112–125. doi: 10.1007/978-3-319-39028-4.
[26] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," in *Proceedings of the 33rd Hawaii International Conference on System Sciences - 2000*, Maui, HI, USA: IEEE, 2000, p. 10. doi: https://doi.org/10.1109/HICSS.2000.926982.
[27] Z. Yousif, I. Hussain, S. Djahel, and Y. Hadjadj-aoul, "A Novel Energy-efficient Clustering Algorithm for more Sustainable Wireless Sensor Networks enabled Smart Cities Applications," *J. Sens. Actuator Networks*, vol. 10, no. 3, p. 50, 2021, doi: https://doi.org/10.3390/jsan10030050.
[28] S. Rajanarayanan, R. S. Dipasupil, and S. R. Dipasupil, "Energy Efficient Radio Access Technologies and Networking Wireless Access Network," *ICTACT J. Commun. Technol.*, vol. 9, no. 3, pp. 1846–1857, 2018, doi: 10.21917/ijct.2018.0269.
[29] J. M. Lebreton, S. Kandukuri, N. Murad, and R. Lorion, "An Energy-Efficient Duty-Cycled Wake-Up Radio Protocol for Avoiding Overhearing in Wireless Sensor Networks," *Wirel. Sens. Netw.*, vol. 8, no. August, pp. 176–190, 2016, doi: 10.4236/wsn.2016.88015.
[30] W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," *Wirel. Sens. Network, 2008*, vol. 1, no. June, pp. 1–11, 2008.
[31] A. Grover, "Optimal Duty Cycling with Sleep-wake Schedule between Paired Nodes and Flexible Routing across Pairs," *Int. J. Comput. Appl.*, vol. 144, no. 8, pp. 20–24, 2016.
[32] E. Azoidou, Z. Pang, Y. Liu, D. Lan, G. Bag, and S. Gong, "Battery Lifetime Modelling and Validation of Wireless Building Automation Devices in Thread," *IEEE Trans. Ind. Informatics*,

vol. 14, no. 7, pp. 2869–2880, 2018, doi: 10.1109/TII.2017.2773066.

[33] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," in *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, Maui, HI, USA: IEEE, 2000, p. 10. doi: 10.1109/HICSS.2000.926982.

[34] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, "Energy-Aware Wireless Microsensor Networks," *IEEE Signal Process. Mag.*, vol. 19, no. 2, pp. 40–50, 2002, doi: https://doi.org/10.1109/79.985679.

[35] C. M. Vigorito, D. Ganesan, and A. G. Barto, "Adaptive Control of Duty Cycling in Energy-Harvesting Wireless Sensor Networks," in *2007 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, San Diego, CA, USA: IEEE, 2007, pp. 21–30. doi: https://doi.org/10.1109/SAHCN.2007.4292814.

[36] H. T. Tran, C. V Nguyen, H. T. T. Nguyen, and M. T. Nguyen, "Energy harvesting for devices in wireless sensor networks : A Review," *EAI Endorsed Trans. Internet Things*, vol. 9, no. 2, pp. 1–11, 2023.

[37] M. U. Mushtaq, H. Venter, A. Singh, and M. Owais, "Advances in Energy Harvesting for Sustainable Wireless Sensor Networks : Challenges and Opportunities," *Hardware*, vol. 3, no. 1, pp. 1–26, 2025, doi: 10.3390/ hardware3010001.

[38] T. Sanislav, G. D. Mois, S. Zeadally, and S. C. Folea, "Energy Harvesting Techniques for Internet of Things (IoT)," *IEEE Access*, vol. 9, pp. 39530–39549, 2021, doi: https://doi.org/10.1109/ACCESS.2021.3064066.