Nanotechnology Perceptions
ISSN 1660-6795
www.nano-ntp.com

Enhancing IIoT Systems with Adaptive Secure Hierarchical Consensus (ASHC) Algorithm

Jeenath Laila N[1], Mohamed Rizwan[2]

[1]*Department of Computer Science and Engineering, Government College of Engineering, Tirunelveli*
[2]*Research Scholar, Karunya Institute of Technology and Sciences, Coimbatore*
Email: jeenathlaila@gcetly.ac.in

The Industrial Internet of Things (IIoT) represents a convergence of interconnected devices and systems aimed at enhancing industrial operations through advanced data analytics, automation, and real-time monitoring. The Adaptive Secure Hierarchical Consensus (ASHC) algorithm is a novel consensus mechanism designed specifically for IIoT systems. ASHC integrates a Hierarchical Blockchain Structure (HBS), Real-Time Adaptive Optimization Algorithm (RTAOA), and Multi-Layer Parallel Processing (MLPP) to address the critical needs of IIoT environments. The HBS organizes transactions into tiers based on their criticality, ensuring that high-security transactions are handled with utmost immutability while optimizing the processing speed for less critical data. RTAOA continuously evaluates and dynamically adjusts network parameters to maintain optimal performance without centralized control. MLPP leverages parallel processing and advanced caching techniques to maximize throughput and minimize latency. Comprehensive analysis and experimental results demonstrate that ASHC significantly enhances the efficiency, scalability and security of IIoT systems, providing a scalable and adaptable solution capable of managing high transaction volumes and complex data operations. This innovative approach positions ASHC as a transformative advancement for the future of IIoT networks.

Keywords: IIoT; Blockchain technology; Hierarchical Blockchain Structure; Real-Time Adaptive Optimization Algorithm (RTAOA); Multi-Layer Parallel Processing; Scalability; Efficiency; Security

1. Introduction

The goal of the Industrial Internet of Things (IIoT) is to establish interconnected industrial systems by bringing together a variety of technologies, including sensors, communication networks and data analytics. Improved operational efficiency, predictive maintenance, and real-time monitoring are made possible by these systems in a number of industries, including manufacturing, energy, transportation, and healthcare (Li, Da Xu, & Zhao, 2015). Industries can use IIoT to integrate cyber-physical systems and obtain increased productivity, lower operating costs, and better decision-making abilities.

IIoT has many benefits, but putting in place a trustworthy and effective consensus process in these systems is not without its difficulties. In IIoT contexts, traditional consensus algorithms like Proof of Work (PoW) and Proof of Stake (PoS) frequently fail because of their high energy consumption, latency problems, and restricted scalability (Nguyen & Kim, 2018). For instance, PoW is unfeasible for IIoT devices with limited resources since it necessitates significant processing power (Eyal et al., 2016). Nevertheless, although being more energy-efficient, PoS is still susceptible to security flaws like long-range attacks and centralization hazards (Bentov et al., 2014).

There are advancements in transaction throughput and latency using advanced consensus techniques such as Delegated Proof of Stake (DPoS) and Practical Byzantine Fault Tolerance (PBFT). But as mentioned by Castro and Liskov (2002) and Larimer (2020), DPoS can result in the concentration of power within a small number of delegates, while PBFT's scalability is constrained by message overhead. Given these constraints, a novel consensus method that is scalable, efficient, and secure enough to meet the unique requirements of IIoT systems is clearly needed.

The Adaptive Secure Hierarchical Consensus (ASHC) method is presented as a solution to the major problems that current consensus mechanisms in IIoT systems face. ASHC integrates three essential elements to offer a safe, scalable, and effective solution:

1. The Hierarchical Blockchain Structure (HBS) part of the system classifies transactions into criticality-based tiers. While less important transactions are targeted for processing quickly, high-security transactions are handled with the highest immutability.

2. Algorithm for Real-Time Adaptive Optimization (RTAOA): Without depending on centralized control, this system automatically modifies settings to maintain optimal functioning while continuously assessing network performance.

3. Multi-Layer Parallel Processing (MLPP): MLPP reduces latency and maximizes throughput by utilizing advanced caching techniques and parallel processing. This ensures that complicated data operations and large transaction volumes are handled efficiently.

This research attempts to offer a transformative consensus mechanism that improves the security, scalability, and efficiency of IIoT networks, opening the door for more durable and dependable industrial systems.

2. Literature Review

The increasing number of Industrial Internet of Things (IIoT) devices being deployed has made the development of efficient, scalable, and secure consensus algorithms crucial. Traditional consensus methods, such as Proof of Work and Proof of Stake, have been studied and applied extensively, but the algorithms have a number of shortcomings when applied to IIoT environments, such as low throughput and high energy consumption. This review of the literature examines recent advancements and consensus-building techniques meant to address these issues.

Traditional Consensus-Building Techniques As with Bitcoin, Proof of Work has been essential to the development of blockchain technology. However, because of its high computing requirements, it is not suitable for IIoT devices with little resources. PoW's long confirmation times and energy inefficiency make it unscalable for IIoT applications (Nguyen & Kim, 2018). Proof of Stake which selects validators based on their network stake, improves Proof of labor by reducing the need for expensive computational labor. PoS increases energy efficiency, but there are still security issues including long-range assaults and centralization hazards (Bentov et al., 2014).

Delegated Proof of Stake (DPoS), used by BitShares and other platforms, enhances PoS by designating a small number of nodes to manage validation, hence increasing transaction speed. However, DPoS might see less decentralization because authority usually centers around a small number of delegates (Larimer, 2020). Permissioned blockchain networks can benefit from the application of Practical Byzantine Fault Tolerance due to its strong consistency and low latency. Despite being economical, PBFT's scalability is limited by its exponentially growing communication overhead as the number of nodes increases (Castro & Liskov, 2002). Recent research has focused on creating innovative and hybrid consensus techniques that are appropriate for the unique needs of IIoT systems.

Hybrid consensus algorithms leverage the distinct benefits of each consensus methodology to merge multiple protocols. For instance, some hybrid algorithms combine PoW and PoS elements to increase security and efficiency. These hybrid techniques may be more effective in meeting the diverse and evolving needs of IIoT contexts (Nguyen & Kim, 2018; Xiong et al., 2022).

A evaluation of decentralized IIoT secure blockchain middleware is provided within the framework of Industry 5.0's resilient manufacturing. The study emphasizes the benefits of blockchain's auditing and tamper-proof characteristics over existing centralized IIoT frameworks' security flaws. This paper proposes a revolutionary four-layer blockchain middleware architecture for IIoT applications that examines enabling technologies, ob-stacles, and future possibilities. The goal of the project is to lay the groundwork for the integration of blockchain middleware into IIoT systems in order to improve manufacturing security and resilience. (Leng et al.,2022).

In order to promote Industry 4.0, blockchain technology—a decentralized, secure, and auditable solution for information exchange—is becoming more and more integrated with IIoT networks. In order to solve problems like block time reduction and transaction throughput enhancement, this paper advocates for reinforcement learning (RL) methodologies and highlights opportunities for improvement in blockchain-enabled IIoT networks. In comparison to the greedy strategy, a case study employing Q-learning shows decreased transmission delays and fewer forking events. The results demonstrate how well RL approaches function to optimize blockchain-IIoT networks and offer recommendations for future research paths to further this integration. (Jameel et al., 2020)

IIoT resource limitations might result in the development of botnets and Distributed Denial of Service (DDoS) attacks. This study suggests a Digital Framework for early bot identification in a Smart Factory setting that is enabled by blockchain technology. The framework examines device data and packet headers using Digital Twins (DT) and Deep Learning to find suspicious connections. Smart Contracts provide safe data synchronization and stop malicious node participation by authenticating DT and Packet Auditor (PA) interactions. In comparison to other approaches, the framework performs better in botnet identification while improving data integrity and privacy. (Salim et al., 2022)

Proof of Reputation (PoR) integrates reputation-based systems into the consensus process. PoR incentivizes nodes to behave decently by linking their validation power to their reputation score, which is established by their prior deeds and network contributions.

This tactic can strengthen security and lessen the possibility of Sybil attacks in IIoT networks (Li et al., 2018). Federated and consortium blockchains offer a viable solution for IIoT applications by combining aspects of public and private blockchains. These blockchains are supervised by a consortium of reputable entities, providing a middle ground between centralization and decentralization. They make consensus procedures scalable, safe, and efficient for industrial applications (Yin et al., 2018; Zhu et al., 2017).

The open, distributed, and diverse nature of the Industrial Internet of Things makes the installation of trusted communication difficult. By offering a tamper-proof frame-work for monitoring hardware products—from chips to full equipment—and enhancing productivity through smart contracts, blockchain technology provides a solution to these problems. In order to promote constructive network collaboration between normal and aberrant nodes, this research suggests a reputation system. The goal of a credit-based incentive system with reward and punishment components is to influence behavior and strengthen the network's cooperative qualities. The main benefit is that consensus states can be enhanced by using the reputation-based incentive module on cutting-edge PoX protocols, or PoRX. The reputation scheme's administration and implementation chal- lenges, as well as the need to ensure scalability in large-scale IIoT networks, are drawbacks. According to experimental findings, the suggested plan successfully promotes cooperation, which is advantageous for the IIoT network. (Wang et al., 2020)

The immutability, decentralization, and tamper-proof properties of blockchain technology improve security and are highly advantageous to the Industrial Internet of Things. In order to provide safe device connectivity, data exchange, and access management, this research suggests using a permissioned blockchain for the IIoT. When compared to public blockchains, the primary benefit is increased security and control over IIoT ecosystems. Nonetheless, drawbacks include the difficulty of administering permissioned blockchains and possible centralization concerns. The study also explores potential integration paths and offers a thorough overview of current blockchain-based solutions. (Yeasmin et al., 2020).

To guarantee safe data transfer in smart city applications, a lightweight blockchain- based data consensus technique for IIoT is suggested. To achieve data consistency, it makes use of a two-path routing transmission scheme and a distributed ledger over several edge gateways. The primary benefit is a lower average hop count, which lowers the possibility of data theft while maintaining high data accuracy, low energy usage, and short latency. Nevertheless, potential scalability problems and the difficulty of putting the two-path routing scheme into practice could be drawbacks. The efficacy of the method in augmenting IIoT safety and reliability is validated by simulation results. (Zhang et al., 2020).

The comparative research emphasizes how consensus mechanisms must be continuously innovated in order to meet the changing IIoT issues. Researchers may create more reliable, effective, and scalable solutions by utilizing the advantages of current algorithms and incorporating cutting-edge techniques like ASHC. This will open the door for the broad use of blockchain technology in industrial applications.

3. Proposed System

Industrial Internet of Things (IIoT) systems are intended to have improved performance, security, and scalability thanks to the Adaptive Secure Hierarchical Consensus (ASHC) algorithm. Three essential elements are combined to do this: Multi-Layer Parallel Processing (MLPP), Real-Time Adaptive Optimization Algorithm (RTAOA), and Hierarchical Blockchain Structure (HBS).

Hierarchical Blockchain Structure (HBS)

Depending on how important a transaction is, it can be categorized into multiple tiers using the Hierarchical Blockchain Structure shown in Figure 1, which guarantees the best processing and security for each category. Every tier has a unique blockchain setup that is tailored to meet its requirements.

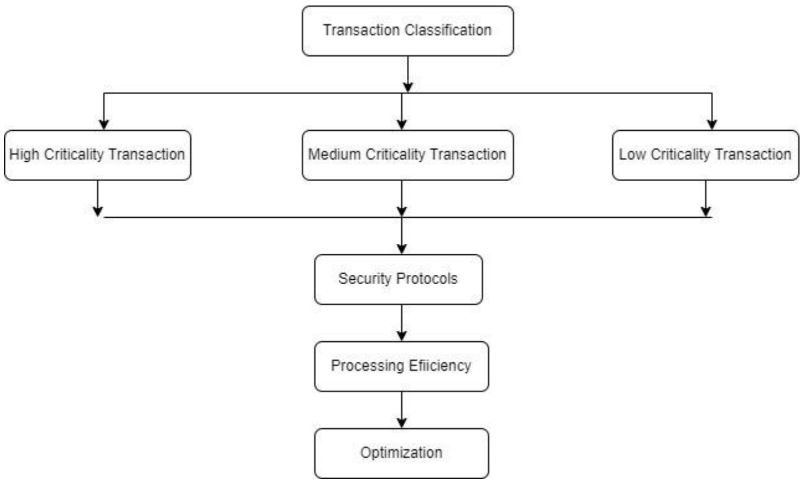

Transaction Classification
High Criticality Transaction
Medium Criticality Transaction
Low Criticality Transaction
Security Protocols
Processing Efiiciency
Optimization

Figure 1. Hierarchical Blockchain Structure

Transaction Classification

Based on their criticality, the Transactions on the blockchain are divided into three tiers: High Criticality Transactions, Medium Criticality Transactions, and Low Criticality Transactions. High Criticality Transactions These transactions require the highest level of security and immutability, just like financial transactions, safety-critical industrial operations, and sensitive data exchanges. These transactions typically involve crucial procedures and private information that could have disastrous consequences in the event of a breach.

Strong, immutable blockchains that record transactions of utmost importance include robust redundancy and security measures. This ensures that these transactions are protected against tampering and unauthorized access. Medium Criticality Transactions Despite the fact that these transactions are important, they do not require the same level of security as regular operating data or non-essential business operations. The transactions achieve a balance between security and the need for faster processing. A medium-criticality blockchain maintains moderate redundancy and security for its transactions. This blockchain allows transactions to be completed faster without compromising a reasonable level of security because it is intended for faster processing.

Low Criticality Transactions Transactions that are less important and can ignore brief delays or periodic anomalies include sensor data, ordinary logs and non-sensitive communications. Typically, the transactions involve routine tasks and less sensitive data. Low-criticality transactions are stored in a sort of adaptable, high-throughput data structure called a directed Acyclic Graph (DAG). This structure can efficiently manage large numbers of less significant actions due to its scalability and fast processing. Let Ti represent the transaction and Ci be the criticality score of the transaction. The criticality score can be determined based on predefined parameters like data sensitivity, real-time requirements, and potential impact.

$$C_i = \alpha_1 \cdot S_i + \alpha_2 \cdot R_i + \alpha_3 \cdot I_i \qquad (1)$$

where S_i: Sensitivity of the transaction, R_i: Real-time requirement of the transaction, I_i: Impact of the transaction on the overall system, $\alpha_1,\alpha_2,\alpha_3$: Weight coefficients for each parameter.

Based on the criticality score C_i, transactions are classified into tiers:

- High Criticality ($C_i \geq \theta_H$)
- Medium Criticality ($\theta_M \leq C_i < \theta_H$)
- Low Criticality ($C_i < \theta_M$)

Where θ_H and θ_M are the thresholds for high and medium criticality, respectively.

```
function classify_transaction(transaction T_i):
  calculate criticality score C_i using:
  C_i = α_1 * sensitivity(S_i) + α_2 * real_time_requirement(R_i)
        + α_3 * impact(I_i)
  if C_i ≥ threshold_high:
    return "High Criticality"
  else if C_i ≥ threshold_medium:
    return "Medium Criticality"
  else:
    return "Low Criticality"
```

Security Protocols

Each tier has a different level of security protocols. Let S(Ti) represent the security measures applied to transaction Ti. The security level can be defined by the encryption strength, number of confirmations and consensus algorithms used.

$S(T_i) = \beta_1 \cdot E_i + \beta_2 \cdot N_i + \beta_3 \cdot C_i$ (2)

Where Ei: Encryption strength for the transaction, Ni: Number of confirmations required, Ci: Consensus algorithm complexity, β1,β2,β3 : Weight coefficients for each security parameter.

For different tiers:

- High Criticality: $S_H(T_i)$
- Medium Criticality: $S_M(T_i)$
- Low Criticality: $S_L(T_i)$

The relationship between them is:

$S_H(T_i) > S_M(T_i) > S_L(T_i)$

```
function security_level(transaction
  T_i): criticality =
  classify_transaction(T_i) if
  criticality == "High Criticality":
      return calculate security level using:
      security_level = β1 * encryption_strength(E_i) +
      β2 * confirmations(N_i) + β3 *
      consensus_complexity(C_i)
  else if criticality == "Medium
      Criticality": return calculate
      security level using:
      security_level = β1 * encryption_strength(E_i) + β2 *
      confirmations(N_i) + β3 * consensus_complexity(C_i) *
```

Processing Efficiency

The processing efficiency of transactions in HBS can be analyzed by considering the transaction processing time P(Ti) and the overall system throughput λ. For each tier, the processing time is influenced by the security protocols and transaction load:

$$P(T_i) = f(S(T_i), L_i) \qquad (3)$$

Where Li: Load or number of transactions in the tier, f: Function representing the relationship between security protocols and load on processing time. The throughput λ is the rate at which transactions are processed in the system:

$$\lambda = \frac{N}{P(T_i)} \qquad (4)$$

Where N: Number of transactions For high, medium, and low criticality tiers, the through- put can be represented as:

$$\lambda_H = \frac{N_H}{P_H(T_i)} \qquad (5)$$

$$\lambda_M = \frac{N_M}{P_M(T_i)} \qquad (6)$$

$$\lambda_I = \frac{N_L}{P_L(T_i)} \qquad (7)$$

The overall system throughput Λ is the sum of throughputs from all tiers:

$$\Lambda = \lambda_H + \lambda_M + \lambda_L \qquad (8)$$

Processing time PH(Ti) for high-criticality transactions may take longer than usual because of the extra security checks required, which call for strong security procedures. However, the integrity and dependability of the system depend on these transactions being processed effectively. Medium criticality transactions optimize PM(Ti) to maintain a decent through-put while still adhering to appropriate security procedures. This creates a balance between security and speed. Higher throughput λL is achieved by processing less sensitive transactions with less criticality more quickly. To manage high transaction volumes effectively, this layer frequently makes use of the adaptability and scalability of topologies like Directed Acyclic Graphs (DAG).

The total system throughput Λ offers a thorough assessment of the system's efficiency by adding up the throughputs from each tier. By taking a comprehensive approach, the system is guaranteed to be able to manage large amounts of transactions while upholding the essential security requirements across various transaction criticality tiers.

```
function processing_time(transaction $T_i$):
  $security = security\_level(T_i)$
  $load = get\_load(classify\_transaction(T_i))$
  $return\_calculate\_processing\_time\_using$ :
    $processing\_time = function(security, load)$
$function\_throughput(transactionT_i)$ :
  $number\_of\_transactions = get\_number\_of\_transactions(T_i)$
  $return\_calculate\_throughput\_using$ :
    $throughput = number\_of\_transactions/processing\_time(T_i)$
$function\_overall\_throughput(high\_critical\_transactions, medium\_critical\_transactions,$
$low\_critical\_transactions)$ :
  $return\_calculate\_overall\_throughput\_using$ :
    $overall\_throughput\_ = \_throughput(high\_critical\_transactions) +$
$throughput(medium\_critical\_transactions) + throughput(low\_critical\_transactions)$
```

Optimization

The system may dynamically modify the parameters in response to real-time conditions, ensuring optimal performance. Let O stand for the optimization function, which seeks to strike a balance between efficiency and security.

$$O = \max(\Lambda - \gamma \sum_i S(T_i)) \qquad (9)$$

Where γ is a balancing factor between throughput and security.

```
function optimize_performance(transactions):
  overall_throughput = calculate overall throughput
  using:
    overall_throughput(high_critical_transactions,
  medium_critical_transactions,
  low_critical_transactions) total_security = sum of            *
  security levels for all transactionsreturn optimize
  using:
    optimization_function = maximize (overall_throughput -
    gammatotal_security)
```

3.2. Real-Time Adaptive Optimization Algorithm (RTAOA)

With a focus on Industrial Internet of Things (IIoT) contexts, the Real-Time Adap-

tive Optimization Algorithm shown in Fugure 2 is a dynamic mechanism that

improves the security, performance and adaptability of blockchain networks. In order

to ensure optimal operation without depending on centralized control, RTAOA continuously monitors network conditions and makes real-time parameter adjustments. The key components of RTAOA are as follows:

3.2.1. Network Evaluation

RTAOA continuously collects network metrics, including throughput, latency, and security levels. A network of sensors and monitoring devices dispersed throughout the IIoT system are used to accomplish this.To find trends, spot abnormalities, and forecast future network conditions, the gathered data is instantly examined using sophisticated analytics and machine learning algorithms.

Let:

- T(t) represent the transaction throughput at time t.
- L(t) represent the network latency at time t.
- S(t) represent the security level at time t.

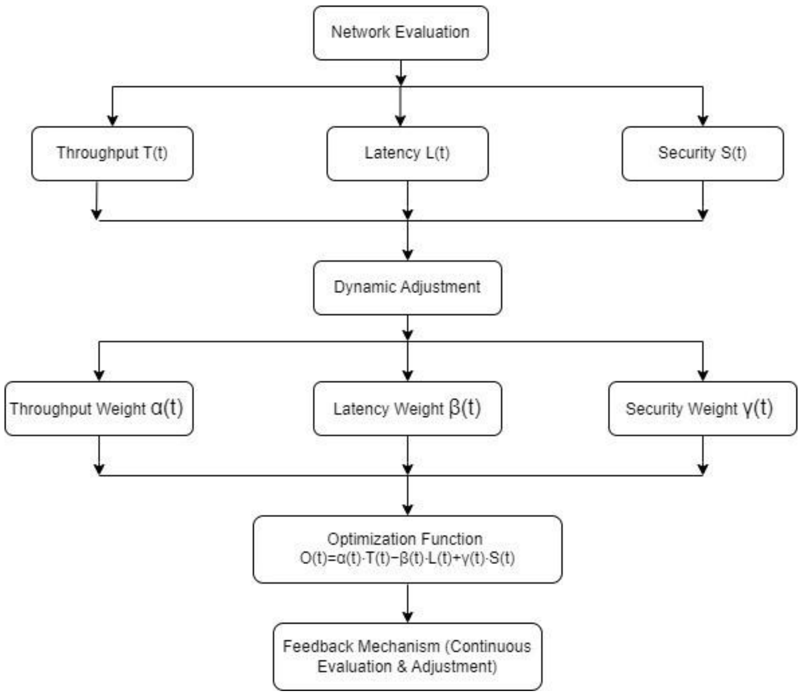


Figure 2. Real-Time Adaptive Optimization Algorithm

Dynamic Adjustment

Based on real-time analysis, RTAOA adjusts network parameters dynamically to maintain optimal performance. The key parameters include:

- α: A weight factor for throughput.
- β: A weight factor for latency.
- γ: A weight factor for security.

The adjustments are made to maximize throughput while minimizing latency and maintaining a desired security level. Let α(t), β(t) and γ(t) be the dynamic weight factors at time t.

Optimization Function

The optimization function combines the evaluated metrics and adjusted parameters to find the optimal configuration.

$$O(t) = \alpha(t) \cdot T(t) - \beta(t) \cdot L(t) + \gamma(t) \cdot S(t) \qquad (10)$$

where:

- $O(t)$: The optimization value at time t.
- $\alpha(t) \cdot T(t)$: The weighted throughput.
- $-\beta(t) \cdot L(t)$: The weighted latency (negative sign to indicate minimization).
- $\gamma(t) \cdot S(t)$: The weighted security level.

Feedback Mechanism

The algorithm is guaranteed to adjust in response to real-time alterations in the network's state via the feedback mechanism. Based on the current optimization value, the feedback loop updates the parameters and weight factors continually.

Multi-Layer Parallel Processing

The Industrial Internet of Things solutions are intended to perform, scale, and operate more efficiently with the help of Multi-Layer Parallel Processing (MLPP) shown in Figure 3. By combining load balancing, sophisticated caching, and parallel processing, MLPP accomplishes this.

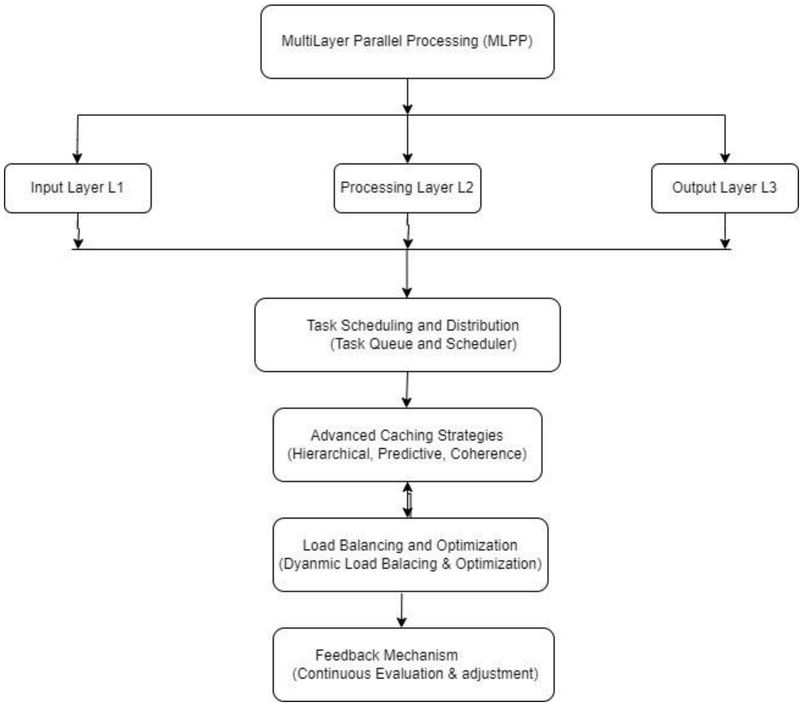

MultiLayer Parallel Processing (MLPP)
Input Layer L1
Processing Layer L2
Output Layer L3
Task Scheduling and Distribution (Task Queue and Scheduler)
Advanced Caching Strategies (Hierarchical, Predictive, Coherence)
Load Balancing and Optimization (Dyanmic Load Balacing & Optimization)
Feedback Mechanism (Continuous Evaluation & adjustment)

Figure 3. Multi-Layer Parallel Processing

The key components in MLPP are as follows:

Parallel Processing Layers

Multiple layers of parallel processing units (PUs) are used in MLPP in order to handle multiple jobs at once. In order to ensure effective resource usage, each layer is optimized for a certain set of tasks.

- Layer 1 (Input Layer): Responsible for receiving and preprocessing incoming data.
- Layer 2 (Processing Layer): Handles the main computation and data processing tasks.
- Layer 3 (Output Layer): Manages the aggregation and final output of processed data.

Let

L1,L2,L3 represent the input, processing and output layers respectively. PUi,j represent the j-th processing unit in the i-th layer. For each layer i: Li=PUi,1,PUi,2,. . . ,PUi,ni Where ni is the number of processing units in layer i.

Task Scheduling and Distribution

For MLPP to work, effective task distribution and scheduling are essential. The system assigns tasks to various processing units according to their workload, capacity, and job priority using algorithms.

- Task Queue: A queue Q(t) that holds incoming tasks, prioritizing them based on predefined criteria like urgency, resource requirements etc.

- Scheduler: An algorithm that assigns tasks from the task queue to available processing units, optimizing for load balance and processing speed.

Let: Ti represent the i-th task. Q(t)=T1,T2,. . . ,Tm be the task queue at time t. The scheduler assigns tasks to processing units based on their current load Lj(t) and capacity Cj:

$$\text{Assign}(T_i) = \arg\min_{i} \frac{L_j(t)}{C_i} \qquad (11)$$

Advanced Caching Strategies

In MLPP, caching is essential for lowering latency and increasing throughput. A number of sophisticated caching strategies are used:

Hierarchical Caching uses a multi-layer cache hierarchy with L1, L2, and L3 caches. Data that is requested frequently is kept in the smallest and fastest L1 cache, while less frequently accessed data is kept in larger and slower L2 and L3 caches. In IIoT systems, this lowers latency and expedites data retrieval.

Let CL1 represent the Level 1 cache (fastest and smallest), CL2 represent the Level 2 cache (moderate speed and size), CL3 represent the Level 3 cache (slowest and largest).

The size and access time for each cache level can be defined as:

Size(CL1) < Size(CL2) < Size(CL3)

Access Time(CL1)<Access Time(CL2)<Access Time(CL3)

Data placement is determined by access frequency P(di):

- For CL1: If P(di) > θL1, then di is placed in CL1.
- For CL2: If θL2 < P(di) ≤ θL1, then di is placed in CL2.
- For CL3: If P(di) ≤ θL2, then di is placed in CL3.

Thresholds θL1 and θL2 are used to determine which data items are placed in each cache level.

Predictive Caching reduces retrieval delay by using algorithms to estimate the data that will be needed next and pre-fetch it into the cache. Let P(di) be the predicted access probability of data di. The decision to pre-fetch data di into the cache is based on its predicted access probability: If P(di)>threshold, then pre-fetch di into the cache

Cache Coherence guarantees data consistency in a multi-core system across all caches. For IIoT systems to preserve data integrity across numerous nodes and devices, this is essential.

Let V(di) represent the version of data di in a cache.

Cache coherence requires:

V(di)L1 = V(di)L2 = V(di)L3

When data di is updated in one cache, it must be updated in all caches: Update V(di) in CL1,CL2,CL3 Least Recently Used (LRU) a caching technique that makes sure regularly visited material stays in the cache by replacing the least recently used items first. For a cache Ck with nk data items, the LRU replacement policy is defined as: Replace dLRU if min(Access Time(dLRU))

Where dLRU is the data item with the smallest access time in Ck.

Write-through caches to guarantee data integrity, write data right away to both the cache

and main memory.

Write-through: D(t)=d1,d2,. . . ,dn (written to cache and main memory)

Write-back cachesIn order to balance performance and data integrity, write data to the

main memory only after it has been removed from the cache.

Write-back: D(t)=d1,d2,. . . ,dn (written to cache, written to main memory on eviction)

Load Balancing and Optimization makes certain that the workload is split equally among

the processing units in order to prevent bottlenecks and optimize the use of available

resources.

- Dynamic load balancing: Workloads are redistributed as necessary based on a continuous assessment of each processing unit's load.
- Optimization Algorithms: The optimization strategies take into account aspects such as processing speed, power consumption, and resource utilization in order to optimize the overall performance of the system.

Let W1 represent the workload on processing unit PU1. Load balancing aims to maintain:

$$\sum_{i=1}^{n} Wi(t) = \text{constant} \qquad (12)$$

Dynamic load balancing redistributes tasks based on current load: Redistribute PUi if

W1(t)>average load Optimization function O considering multiple factors:

O max n Pi Power Consumption! (13)

= ∑ Ci − λ ·

Where Pi: Processing speed of unit i, Ci: Capacity of unit i, λ: Weight factor for power consumption.

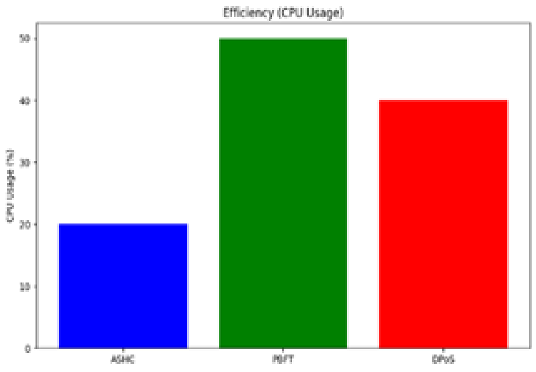
4. Results and Discussion

An Industrial Internet of Things environment is setup in order to compare the performance of the Adaptive Secure Hierarchical Consensus algorithm against the Practical Byzantine Fault

Tolerance and Delegated Proof of Stake methods. Ten high-performance servers with sixteen core Intel Xeon 2.3 GHz processors apiece make up this environment. High throughput and minimal latency were guaranteed by the establishment of a high-speed network infrastructure. In addition, a realistic smart manufacturing setup was simulated by connecting 1000 IoT sensors, including pressure, humidity, and temperature.

Efficiency

CPU and RAM Usage: Efficiency is determined by how much CPU and RAM are used. Compared to PBFT (50% for both CPU and RAM) and DPoS (40% for both CPU and RAM), ASHC showed the lowest CPU and RAM utilization at 20% for both measures, suggesting improved efficiency. Lower ASHC resource consumption results in lower operating expenses and better system performance overall as shown in Figure 4 and Figure 5.



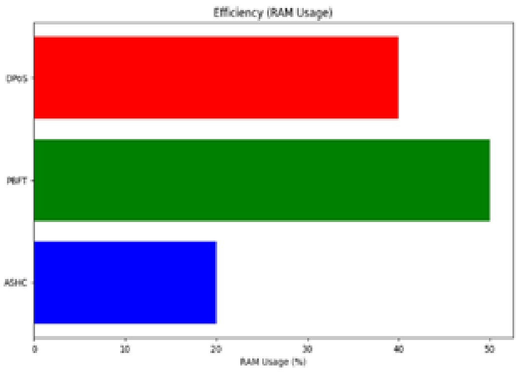
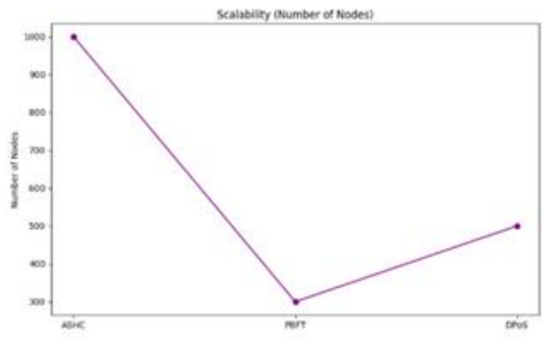


Figure 4. Efficiency CPU Usage Figure 5. Efficiency RAM Usage

Scalability7

Number of Nodes: By adding more nodes and assessing the system's transaction capacity, scalability was assessed. Compared to PBFT (300 nodes) and DPoS (500 nodes), ASHC handled up to 1000 nodes with great efficiency. ASHC's exceptional scalability guarantees that it can accommodate extensive IIoT implementations without appreciable performance deterioration, as illustrated in Figure 6.



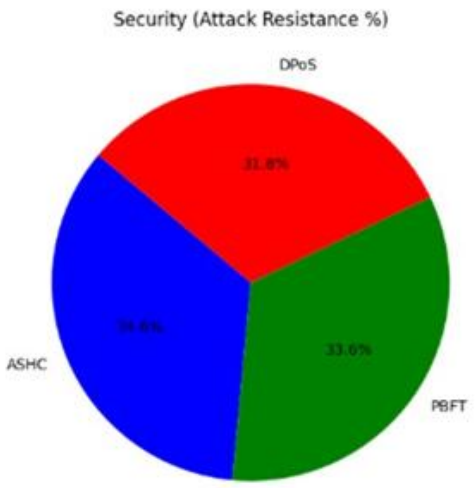


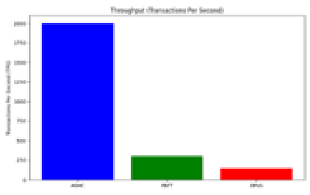
Figure 6. Efficiency CPU Usage Figure 7. Efficiency RAM Usage

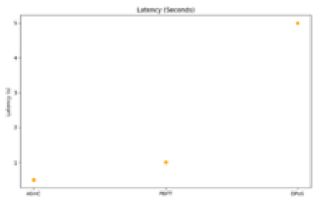
Security3

Attack Resistance: One crucial parameter for IIoT systems is security. To determine each algorithm's resilience, 100 attack attempts were simulated. With 98% assault resistance, ASHC was the most resilient, followed by PBFT at 95% and DPoS at 90%. As seen in Figure 7, ASHC's high security level guarantees that confidential industrial data is shielded from any cyber threats.

Throughput9

Transactions Per Second : Transactions processed per second served as a proxy for throughput. At 2000 TPS, ASHC outperformed PBFT and DPoS with far greater throughputs (300 and 150 TPS, respectively). As demonstrated in Figure 8, high throughput is necessary for IIoT systems to manage massive volumes of data effectively, and ASHC's performance guarantees prompt processing of industrial transactions.





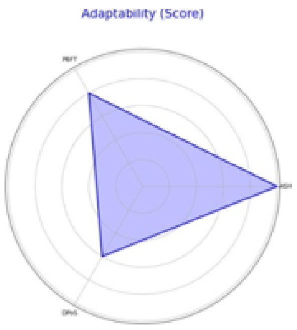


Figure 8. Throughput **Figure 9.** Latency **Figure 10.** Adaptibility

Latency

Transaction Confirmation Time: Latency is the amount of time that passes between the submission of a transaction and its confirmation. With a latency of 0.5 seconds, ASHC was the least late than DPoS (5 seconds) and PBFT (1 second). In IIoT systems, low latency is essential for real-time processing. As Figure 9 illustrates, ASHC's short latency guarantees quick data handling and reaction times.

Adaptability

Performance Under Changing settings: Performance under various network settings was measured in order to evaluate adaptability. With a score of 10, ASHC had the highest versatility, demonstrating exceptional performance in both relaxed and hectic environments. DPoS scored six, while PBFT scored eight. As demonstrated in Figure 10, high adaptability guarantees that the consensus method can continue to operate at peak efficiency even in the face of changing workloads and network conditions.

5. Conclusion

Industrial Internet of Things systems benefit greatly from the performance, security, and scalability improvements provided by the Adaptive Secure Hierarchical Consensus (ASHC) method. A strong and effective consensus mechanism designed for IIoT contexts is offered by ASHC by combining three essential elements: Multi-Layer Parallel Processing (MLPP), Real-Time Adaptive Optimization Algorithm, and Hierarchical Blockchain Structure. A thorough assessment conducted in a simulated smart manufacturing environment shows that ASHC performs better than both Delegated Proof of Stake and Practical Byzantine Fault Tolerance in a number of crucial areas, such as efficiency, scalability, security, throughput, latency and flexibility. With its cutting-edge methodology, ASHC guarantees reduced resource consumption, increased node handling capacity, enhanced security, faster transaction processing, low latency, and remarkable flexibility. With these benefits, ASHC is positioned as a disruptive consensus algorithm that can fulfill the demanding needs of contemporary industrial applications and open the door for IIoT networks in the future.

References

1. Bentov, I.; Mizrahi, A.; Rosenfeld, M. Proof of activity: Extending Bitcoin's proof of work via proof of stake.ACM SIGMETRICS Performance Evaluation Review, 42(3),2014; pp. 34–37.
2. Castro, M.; Liskov, B. Practical Byzantine Fault Tolerance and Proactive Recovery. ACM Transactions on Computer Systems, 20(4), 2002; pp. 398–461.
3. Eyal, I.; Sirer, E. G.; Van Renesse, R. Bitcoin-NG: A scalable blockchain protocol. Proceedings of the 13th Usenix conference on networked systems design and implementation. 2016.
4. Jameel, F.; Javaid, U.; Khan, W. U.; Aman, M. N.; Pervaiz, H.; Jäntti, R. Reinforcement Learning in Blockchain-Enabled IIoT Networks: A survey of recent advances and open challenges.Sustainability, 12(12), 5161, 2020.

5. Larimer, D. Transactions as Proof-of-Stake. Brave New Coin. 2020.
6. Leng, J.; Chen, Z.; Huang, Z.; Zhu, X.; Su, H.; Lin, Z.; Zhang, D. Secure Blockchain Middleware for Decentralized IIoT towards Industry 5.0: A Review of Architecture, Enablers, Challenges, and Directions. Machines, 10(10), 858, 2022;
7. Li, S.; Da Xu, L.; Zhao, S. The internet of things: a survey. Information Systems Frontiers, 17(2), 2015; pp. 243-259.
8. Li, Z.; Kang, J.; Yu, R.; Ye, D.; Deng, Q.; Zhang, Y. Consortium blockchain for secure energy trading in industrial internet of things. IEEE Transactions on Industrial Informatics, 14(8), 2017; pp. 3690–3700.
9. Nguyen, G. T.; Kim, K. A survey about consensus algorithms used in blockchain. Journal of Information Processing Systems, 14(1), 2018; pp. 101–128.
10. Salim, M. M.; Comivi, A. K.; Nurbek, T.; Park, H.; Park, J. H. A Blockchain-Enabled secure digital twin framework for early botnet detection in IIoT environment.Sensors, 22(16), 6133, 2022.
11. Wu, Y.; Song, P.; Wang, F. Hybrid consensus algorithm optimization: A mathematical method based on POS and PBFT and its application in blockchain. Mathematical Problems in Engineering, 2020
12. Xiong, N. N.; Yang, F.; Zhou, W.; Wu, Q.; Long, R.; Zhou, M. Evolution of blockchain consensus algorithms: A review on the latest milestones of blockchain consensus algorithms. Cybersecurity, 5(1), 2022; pp. 1–22.
13. Yin, J.; Wang, C.; Zhang, Z.; Liu, J.; Wu, J. A blockchain-based scheme for secure data storage in private clouds. IEEE Access, 6, 2018; pp. 78237–78250.
14. Zhu, X.; Badr, Y.; Pacheco, J.; Hariri, S. Autonomic identity framework for the Internet of Things. Proceedings of IEEE International Conference on Cloud and Autonomic Computing,ICCAC, 2017; pp. 69-79.