Digital Voting with Blockchain using Interplanetary File System and Practical Byzantine Fault Tolerance

Giddaluru Somasekhar
Department of Computer Science and Engineering, GITAM (Deemed to be University), Hyderabad, India
giddalurisomasekhar@gmail.com (corresponding author)

Sreedhar Jinka
Department of Computer Science and Engineering, GITAM (Deemed to be University), Hyderabad, India
sjinka@gitam.edu

Chinna Kullayappa Kanekal
Department of ECE, PVKK Institute of Technology, Ananthapuramu, India
kanekal.kullayappa1976@gmail.com

Anusha Marouthu
Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Green Fields, Vaddeswaram, Guntur, India
anushaaa9@kluniversity.in

Received: 19 July 2024 | Revised: 18 August 2024 and 13 September 2024 | Accepted: 2 September 2024

ABSTRACT

Traditional voting schemes are often overwhelmed by problems such as deception, influence, and incompetence, which can be resolved by applying blockchain technology with transparency, decentralization, and immutability. This study proposes a safe and indisputable digital voting system with blockchain technology to maintain the integrity of the voting procedure. The reliability and privacy of the voting procedure are upheld with distributed ledger technology and cryptographic techniques. The essence of the proposed method is the immutability of the blockchain ledger, which ensures a tamper-proof record of each cast vote, promoting transparency and offering a way of audit for free verification. The proposed method employs cryptographic protocols to protect individual votes while preserving complete transparency and verifiability of the voting procedure. The InterPlanetary Filesystem (IPFS) is applied to ensure data integrity. Moreover, the practical Byzantine Fault Tolerance (pBFT) consensus algorithm is utilized to remove glitches in distributed settings. The proposed approach provides a decentralized platform where voters can cast their votes from anywhere without difficulty using an internet connection, eliminating the need for physical ballot papers and polling stations. Using immutable ledger and cryptographic security aspects in blockchain, the reliability of the voting procedure can be protected while maintaining voter anonymity and confidentiality. Finally, it is shown that the proposed scheme outweighs other existing approaches.

Keywords-digital voting; blockchain; decentralized; distributed system

I. INTRODUCTION

Voting data have become more susceptible than ever due to the upsurge of advanced technology, which aggravates the challenges of traditional voting schemes. In addition to deception, centralized susceptibility, lack of transparency, physical mistakes, confidentiality concerns, and logistical intricacies, additional challenges arise. Mischievous performers may manipulate defects in electronic voting machines or online platforms to influence or compromise poll results. Such cybersecurity threats pose major risks. Social media spread falsehood and distortion that leads to distrust in democratic methods and a lack of confidence in poll results. Certain demographics are marginalized by availability fences, such as persons in inaccessible zones or those with disabilities, stressing the necessity for comprehensive voting methods. A multidimensional system is required to address these problems by encompassing procedural development, administrative restructuring, collective training, and joint stakeholder efforts to protect the inclusivity, reliability, and capacity of democratic ways in the digital era.

Various challenges are faced by the contemporary situation of voting structures that encumber democratic methods. With paper ballots and centralized systems, several deception practices, such as impersonation, ballot tampering, and data violations, can occur when implementing conventional voting methods. The accuracy and reliability of election outcomes deteriorate with these limitations, affecting the communal faith in the objectivity of democratic ways. Furthermore, the lack of transparency inherent in these methods becomes a problem for stakeholders while validating the authenticity of the votes cast and the veracity of the polling procedure cannot be guaranteed. There is a demand for pioneering solutions to resolve these issues using evolving technologies to create a more transparent, safe, and convenient voting scheme. This study aimed to fill this gap by developing a blockchain-based platform to ensure the faithfulness of the polls, protect voter secrecy, and improve trust in democratic actions. These areas of current voting systems can be largely enriched for the common good of society.

In [1], a cloud-based online voting system was proposed using a hybrid blockchain. In [2], an enterprise blockchain was utilized for electronic voting. In [3], a broad review of blockchain-based electronic voting was presented. In [4], an optimized voting scheme with a combination of blockchain technology and artificial intelligence was presented. In [5], a review of blockchain-based scalable voting was presented. In [6], an electronic voting system was introduced to address legal and technological problems in the blockchain. In [7], some important research schemes and solutions for electronic voting were offered. In [8], the importance of improving blockchain in electronic voting was highlighted. In [9], a decentralized and automated online voting system with the Ethereum blockchain was proposed. In [10], a blockchain-based secure electronic voting system with face recognition and mobile OTP authentication was proposed. In [11], the current status of blockchain-based voting was studied and evaluated. In [12], an effective blockchain-based distributed application for e-voting was introduced with transparency, fairness, and flexibility features. Blockchain has also been investigated for its role in computer science education [13], to secure mobile agents in the Internet of Things (IoT) context [14], and to detect fraud in cryptocurrency transactions [15]. Many researchers investigated the following problems and suggested different solutions [5-10, 12, 15].

A. Fraud and Manipulation

Regular voting systems are exposed to many practices of influence and deception. From ballot tampering to illegal access to voting records, these issues can compromise the truthfulness and communal faith in the democratic poll procedure. The centralized authorities in the old polling system are a single point of failure that can lead to exploitation or external intrusion. These problems can be overcome using progressive cryptographic methods and decentralized consensus mechanisms.

B. Lack of Transparency

The reliability of poll procedures is protected by maintaining transparency throughout the process. However, being dependent on physical counting methods and a central system, regular voting practice frequently fails to ensure complete transparency. The exactness and objectivity of poll outcomes become suspicious due to tedious procedures. The security flaws in centralized polling schemes can be exploited by malevolent actors to corrupt election data, manipulate vote counts, or alter the poll process, leading to a great danger to democratic practice. The transparency attainable by the proposed voting method using blockchain extends beyond sheer accessibility to voting records. The stakeholders, including election officials, voters, and regulatory bodies, are empowered to independently validate the legitimacy of poll results.

C. Centralized Control and Vulnerabilities

Centralized voting methods pose considerable concerns. From vulnerability to exploitation and illegitimate access to the potential for external stress or stimulus, centralized authorities can be affected throughout the procedure. These complexities worsen the impartiality and safety of the poll process, raising doubts about the veracity of poll results. Blockchain voting methods eliminate these hazards by decentralizing both the infrastructure and the authority of the voting process.

D. Logistical Complexities

Complex logistic tasks are frequently involved in traditional polling methods. Distribution and collection of physical ballots can lead to delays and shortfalls in the election process. The conversion to a blockchain-established polling system offers a better solution to these complications. The digitization of ballots and the automation of voting procedures are implemented in a blockchain-based voting method to streamline the entire election procedure. Smart contracts and digital ballots are used in the system to remove the need for manual interference, decreasing the jeopardy of logistical mistakes and ensuring poll reliability.

E. Manual Errors and Accessibility

Old polling schemes involve threats of human faults that lead to potential compromise in the correctness of vote tallies at the end. Furthermore, the inaccessibility of polling stations can marginalize certain demographics, discouraging their participation in democratic practice. Using pioneering automation and improved accessibility measures, a voting system using blockchain resolves these issues.

F. Concerns

Privacy is a vital feature in elections. The voters desire the confidentiality of their choices and the protection of their identities. However, in traditional voting mechanisms, there may be many more opportunities to steal sensitive data in a centralized environment. Voter privacy could be enhanced through anonymity features and innovative encryption techniques in voting systems that use blockchain.

II. PROPOSED SOLUTION

In response to the intricate challenges faced by conventional centralized voting data management systems, this study introduces a transformative solution powered by blockchain technology. Traditional voting systems have long grappled with issues ranging from vulnerability to manipulation and unauthorized access to concerns about privacy and inclusivity. These challenges have undermined the integrity and trustworthiness of electoral processes, creating a pressing need for innovation.

This approach is tailored to establish a decentralized ecosystem exclusively dedicated to secure and streamlined management and distribution of voting records. Central to this solution is the adoption of blockchain technology, which is renowned for its decentralized and immutable nature. By leveraging blockchain, this approach reshapes the landscape of voting data management, mitigating the risks associated with centralized control and providing a robust framework for transparent and accountable elections. At the heart of this approach lies the concept of decentralized storage. Instead of relying on a single authority to manage voting data, this responsibility is distributed across a network of nodes. Each node maintains a copy of the blockchain ledger, ensuring redundancy and resilience against potential attacks or manipulation. This decentralized approach eliminates the single point of failure inherent in centralized systems, enhancing the security and integrity of the voting process.

The proposed blockchain-based voting solution presents a comprehensive and innovative approach centered on the core principles of transparency, security, decentralization, and efficiency. Leveraging the transformative capabilities of blockchain technology, this solution addresses the long-standing challenges that have plagued conventional voting systems, paving the way for a more reliable and trustworthy electoral process.

Utilizing blockchain technology, the proposed system capitalizes on its decentralized and transparent nature to create a secure and tamper-proof ledger for recording votes. Each vote is cryptographically sealed, ensuring its immutability and transparency, thus instilling confidence in the accuracy of the electoral process. Security and authentication are paramount in the proposed solution. Advanced security measures, including biometrics and cryptographic keys, are integrated to authenticate voters, safeguarding the integrity of the electoral process. This not only protects against unauthorized access but also ensures that each voter is eligible and authentic, mitigating the risks associated with identity fraud. Figure 1 shows the architecture diagram of the proposed system, which combines smart contracts with the Interplanetary Filesystem (IPFS) and practical Byzantine Fault Tolerance (pBFT).

The admin adds candidates and voters via the administration subsystem. After the admin opens the elections, the voter views candidates and votes via the voter subsystem. Admin and voter both are associated with their corresponding smart contracts. The IPFS, integral to the system, securely stores voting data through distinctive hashes for streamlined retrieval, linked to corresponding blockchain records for data authenticity. Scalability is considered, addressing potential gas expenses and transaction thresholds, while IPFS caters to expanding data demands. After voting, each vote is tested for validity using the pBFT algorithm, which checks the manipulation of the voting process by vote alteration. The voter can view the results after the voting process. An attempt is made to overcome the limitations of the existing approaches mentioned in Table I, such as overhead due to a traditional file system, more maintenance cost, less security, etc. In addition, emphasis was placed on reducing the authentication delay, vote alteration, and latency values.

The seamless assimilation of IPFS revolutionizes the storage of actual voting data, empowering secure, decentralized storage while ensuring data integrity through innovative content addressing mechanisms and IPFS hash linkages to blockchain records. User authentication and access control are fortified mechanisms, utilizing ReactJS and token-based secure authentication methods to manage user registration, authentication, and authorization. Smart contracts are imbued with intelligence to allocate roles and permissions to voting records, preventing unauthorized data access. Figure 8 shows the implementation of smart contract functions.

The voting system triggers temporal frontend storage, followed by hash generation and blockchain record creation, ensuring report veracity. IPFS, an integral part of the system, securely stores voting data through distinctive hashes for streamlined retrieval, linked to the corresponding blockchain records for data authenticity. Robust data access mechanisms within the frontend bestow authorized users, including admins and voters, with exclusive access to data based on permissions validated by smart contracts. An innovative feature empowers voters to transparently view voting data and manage access control settings, epitomizing voter-centricity. To ensure security, the architecture employs secure communication between the frontend, blockchain, and IPFS nodes, maintaining data privacy while storing sensitive information such as encrypted hashes and IPFS links. Scalability is considered, addressing potential gas expenses and transaction thresholds, while IPFS caters to expanding data demands.

The pBFT consensus algorithm is applied to overcome node failures and mischievous activities in the blockchain network. With this, the number of transactions processed per second can be very high, increasing the scalability of the network. In addition, transactions can be confirmed rapidly, decreasing latency.

III. RESULTS AND DISCUSSION

The proposed system was tested on a sample of 30,000 voters and compared with existing e-voting approaches. Parameters such as the average gas cost for the implementation of all smart contract functions, the latency of transaction execution, the delay in transaction authentication, and the percentage of possible vote alteration were measured and compared, as shown in Figures 2, 3, 4, and 5, showing that the proposed system outperformed existing approaches.

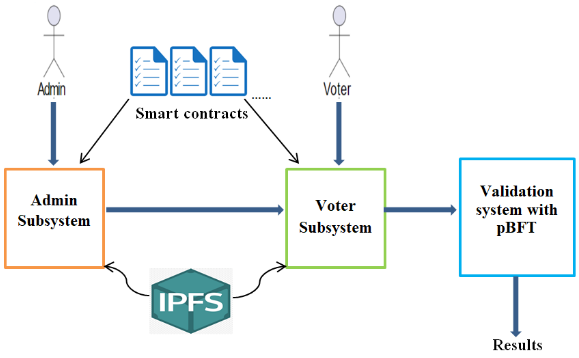


Fig. 1. The proposed system architecture.

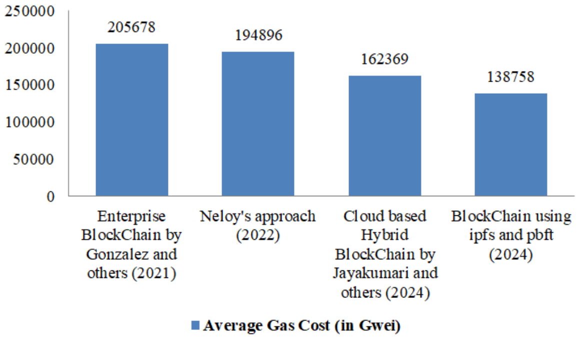


Fig. 2. Average gas cost comparison.

For implementation, the proposed solution leveraged modern frontend technologies such as HTML, CSS, JavaScript, and React.js. On the backend, it utilized the Third-Web framework, integrating solidity smart contracts on the Ethereum blockchain. This tech stack ensures a user-friendly interface, seamless interactions, and a robust, decentralized infrastructure that aligns with the principles of transparency and security. Effective cost reduction functions and IPFS were used to reduce the transaction cost related to smart contracts. The average gas cost was obtained by taking the average gas costs of all smart contracts executed in the voting process. For each approach, the latencies for the samples of 10,000, 20,000, and 30,000 voters were measured, and the average value was taken to compare with other approaches. The average authentication delay and average vote alteration percentage were also measured in the same way. Compared to the approaches in [1, 2, 4] tabulated in Table I, the proposed method obtained 15%, 33%, and 29% reduction, respectively, in average gas cost (Figure 2).

The delay in transaction execution is termed latency and is directly proportional to the number of voters participating in the election. However, when compared to the approaches in [1, 2, 4], the proposed method achieved 25%, 55%, and 41% improvement in average latency, respectively, (Figure 3). Authentication delay is the time taken from voter entry to voter validation. Compared to the approaches in [1, 2, 4], the proposed method obtained 35%, 71%, and 60% improvement, respectively, in average authentication delay (Figure 4). Due to deceptive functionality, the votes can be altered after casting, leading to errors in results. Using the pBFT mechanism, the votes cast can be validated and errors can be eliminated. Compared to the approaches in [1, 2, 4], the proposed method achieved 29%, 67%, and 56% improvement in average vote alteration, respectively (Figure 5).

TABLE I. BENEFITS AND LIMITATIONS OF DIFFERENT E-VOTING SYSTEMS INCLUDING THE PROPOSED

System	Benefits	Shortcomings / Limitations
[1]	Improvement in authentication delay, vote transfer, latency, and so on.	Much overhead due to the traditional file system, less secure, incurs more gas cost
[2]	Flexible	More maintenance cost, less reliable
[4]	Improvement in cost reduction	Lack of efficient cost-reduction mechanisms for large networks, no focus on authentication delay, vote alteration, latency, and so on, less secure
Other approaches outlined in section I.	Can be used as a base to design novel blockchain models	Not suitable in real-time scenarios, lack of scalability
Proposed method	Overcomes the shortcomings of the above approaches	Maintenance cost

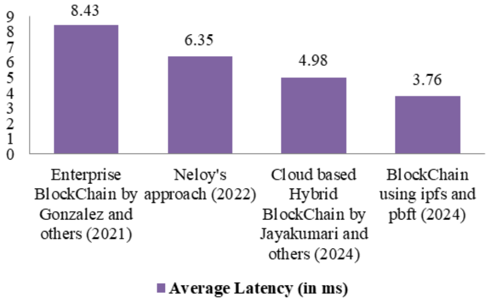


Fig. 3. Average latency comparison.

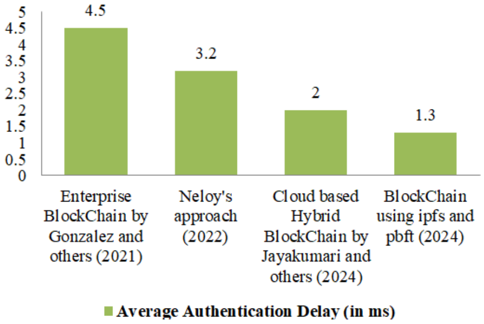


Fig. 4. Average authentication delay comparison.

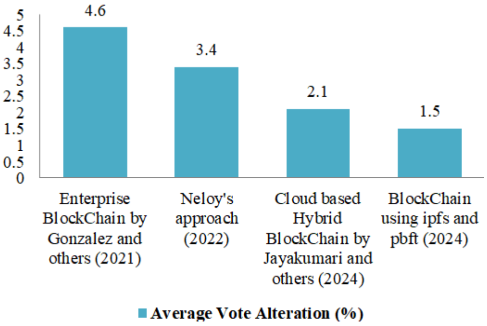
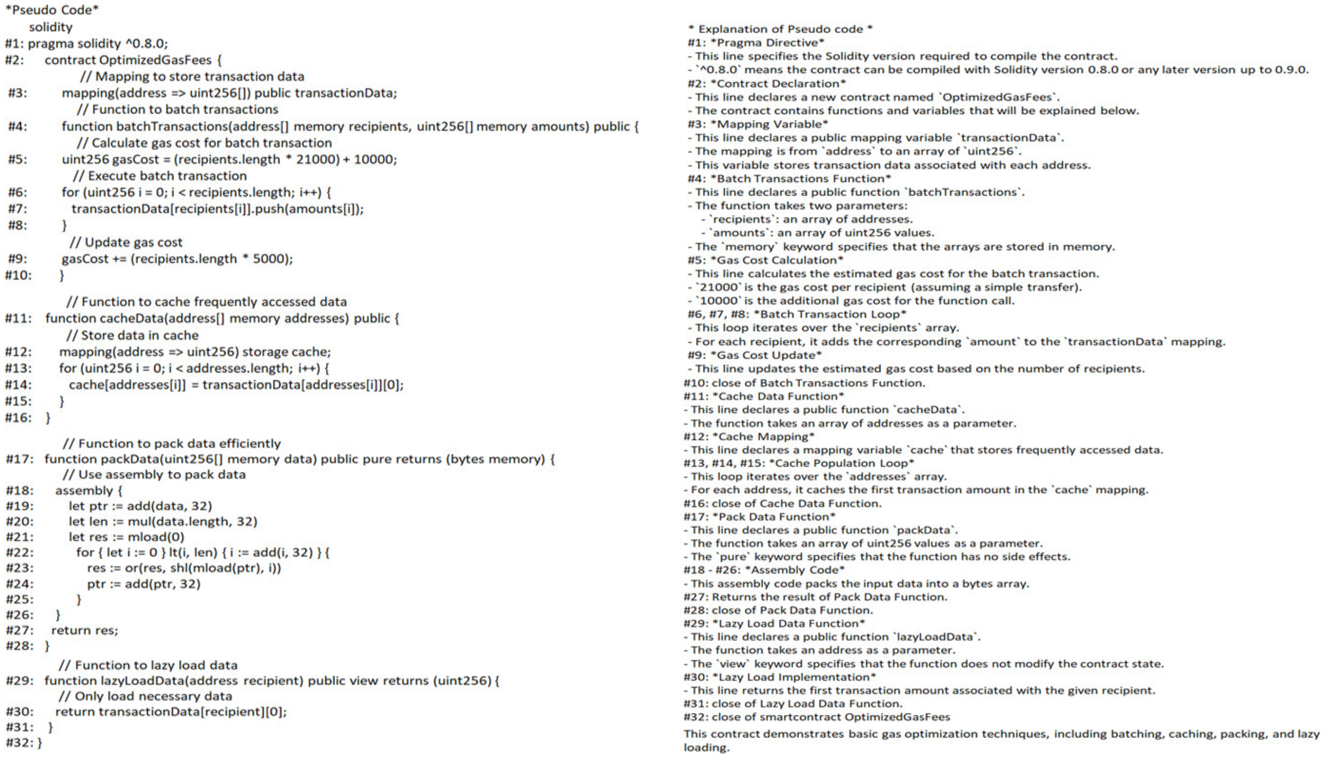


Fig. 5. Average vote alteration (%) comparison.



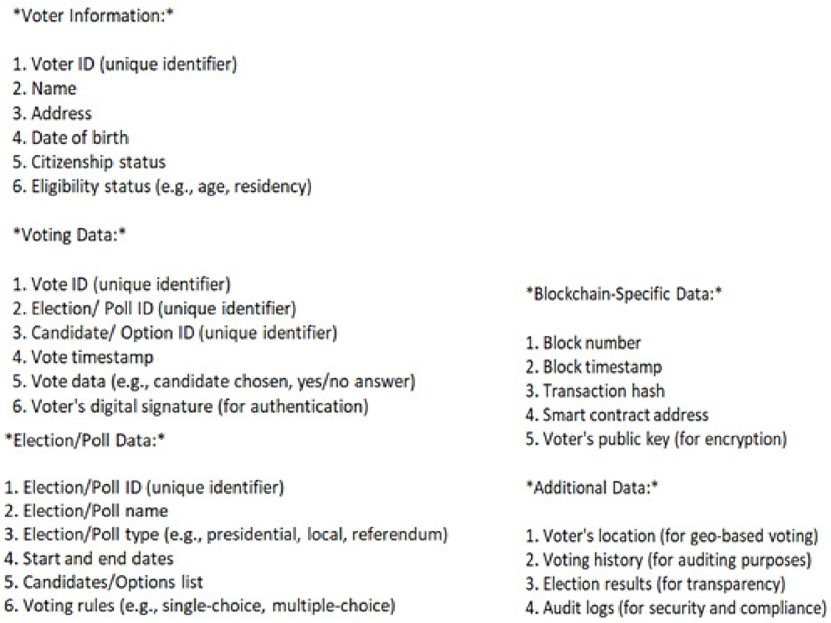
```
*Pseudo Code*
     solidity
#1: pragma solidity ^0.8.0;
#2:     contract OptimizedGasFees {
                // Mapping to store transaction data
 #3:        mapping(address => uint256[]) public transactionData;
               // Function to batch transactions
 #4:        function batchTransactions(address[] memory recipients, uint256[] memory amounts) public {
               // Calculate gas cost for batch transaction
 #5:        uint256 gasCost = (recipients.length * 21000) + 10000;
              // Execute batch transaction
 #6:        for (uint256 i = 0; i < recipients.length; i++) {
 #7:          transactionData[recipients[i]].push(amounts[i]);
 #8:        }
             // Update gas cost
 #9:        gasCost += (recipients.length * 5000);
#10:       }

            // Function to cache frequently accessed data
#11:   function cacheData(address[] memory addresses) public {
            // Store data in cache
#12:      mapping(address => uint256) storage cache;
#13:      for (uint256 i = 0; i < addresses.length; i++) {
#14:        cache[addresses[i]] = transactionData[addresses[i]][0];
#15:      }
#16:   }

          // Function to pack data efficiently
#17:   function packData(uint256[] memory data) public pure returns (bytes memory) {
          // Use assembly to pack data
#18:     assembly {
#19:       let ptr := add(data, 32)
#20:       let len := mul(data.length, 32)
#21:       let res := mload(0)
#22:        for { let i := 0 } lt(i, len) { i := add(i, 32) } {
#23:          res := or(res, shl(mload(ptr), i))
#24:          ptr := add(ptr, 32)
#25:        }
#26:     }
#27:    return res;
#28:  }
         // Function to lazy load data
#29:  function lazyLoadData(address recipient) public view returns (uint256) {
         // Only load necessary data
#30:    return transactionData[recipient][0];
#31:   }
#32: }
```

(a)

```
* Explanation of Pseudo code *
#1: *Pragma Directive*
- This line specifies the Solidity version required to compile the contract.
- `^0.8.0` means the contract can be compiled with Solidity version 0.8.0 or any later version up to 0.9.0.
#2: *Contract Declaration*
- This line declares a new contract named `OptimizedGasFees`.
- The contract contains functions and variables that will be explained below.
#3: *Mapping Variable*
- This line declares a public mapping variable `transactionData`.
- The mapping is from `address` to an array of `uint256`.
- This variable stores transaction data associated with each address.
#4: *Batch Transactions Function*
- This line declares a public function `batchTransactions`.
- The function takes two parameters:
  - `recipients`: an array of addresses.
  - `amounts`: an array of uint256 values.
- The `memory` keyword specifies that the arrays are stored in memory.
#5: *Gas Cost Calculation*
- This line calculates the estimated gas cost for the batch transaction.
- `21000` is the gas cost per recipient (assuming a simple transfer).
- `10000` is the additional gas cost for the function call.
#6, #7, #8: *Batch Transaction Loop*
- This loop iterates over the `recipients` array.
- For each recipient, it adds the corresponding `amount` to the `transactionData` mapping.
#9: *Gas Cost Update*
- This line updates the estimated gas cost based on the number of recipients.
#10: close of Batch Transactions Function.
#11: *Cache Data Function*
- This line declares a public function `cacheData`.
- The function takes an array of addresses as a parameter.
#12: *Cache Mapping*
- This line declares a mapping variable `cache` that stores frequently accessed data.
#13, #14, #15: *Cache Population Loop*
- This loop iterates over the `addresses` array.
- For each address, it caches the first transaction amount in the `cache` mapping.
#16: close of Cache Data Function.
#17: *Pack Data Function*
- This line declares a public function `packData`.
- The function takes an array of uint256 values as a parameter.
- The `pure` keyword specifies that the function has no side effects.
#18 - #26: *Assembly Code*
- This assembly code packs the input data into a bytes array.
#27: Returns the result of Pack Data Function.
#28: close of Pack Data Function.
#29: *Lazy Load Data Function*
- This line declares a public function `lazyLoadData`.
- The function takes an address as a parameter.
- The `view` keyword specifies that the function does not modify the contract state.
#30: *Lazy Load Implementation*
- This line returns the first transaction amount associated with the given recipient.
#31: close of Lazy Load Data Function.
#32: close of smartcontract OptimizedGasFees
This contract demonstrates basic gas optimization techniques, including batching, caching, packing, and lazy
loading.
```

(b)

Fig. 6. Pseudocode (a) for average gas cost reduction and (b) explanations.



Voter Information:

1. Voter ID (unique identifier)
2. Name
3. Address
4. Date of birth
5. Citizenship status
6. Eligibility status (e.g., age, residency)

Voting Data:

1. Vote ID (unique identifier)
2. Election/ Poll ID (unique identifier)
3. Candidate/ Option ID (unique identifier)
4. Vote timestamp
5. Vote data (e.g., candidate chosen, yes/no answer)
6. Voter's digital signature (for authentication)

Election/Poll Data:

1. Election/Poll ID (unique identifier)
2. Election/Poll name
3. Election/Poll type (e.g., presidential, local, referendum)
4. Start and end dates
5. Candidates/Options list
6. Voting rules (e.g., single-choice, multiple-choice)

(a)

Blockchain-Specific Data:

1. Block number
2. Block timestamp
3. Transaction hash
4. Smart contract address
5. Voter's public key (for encryption)

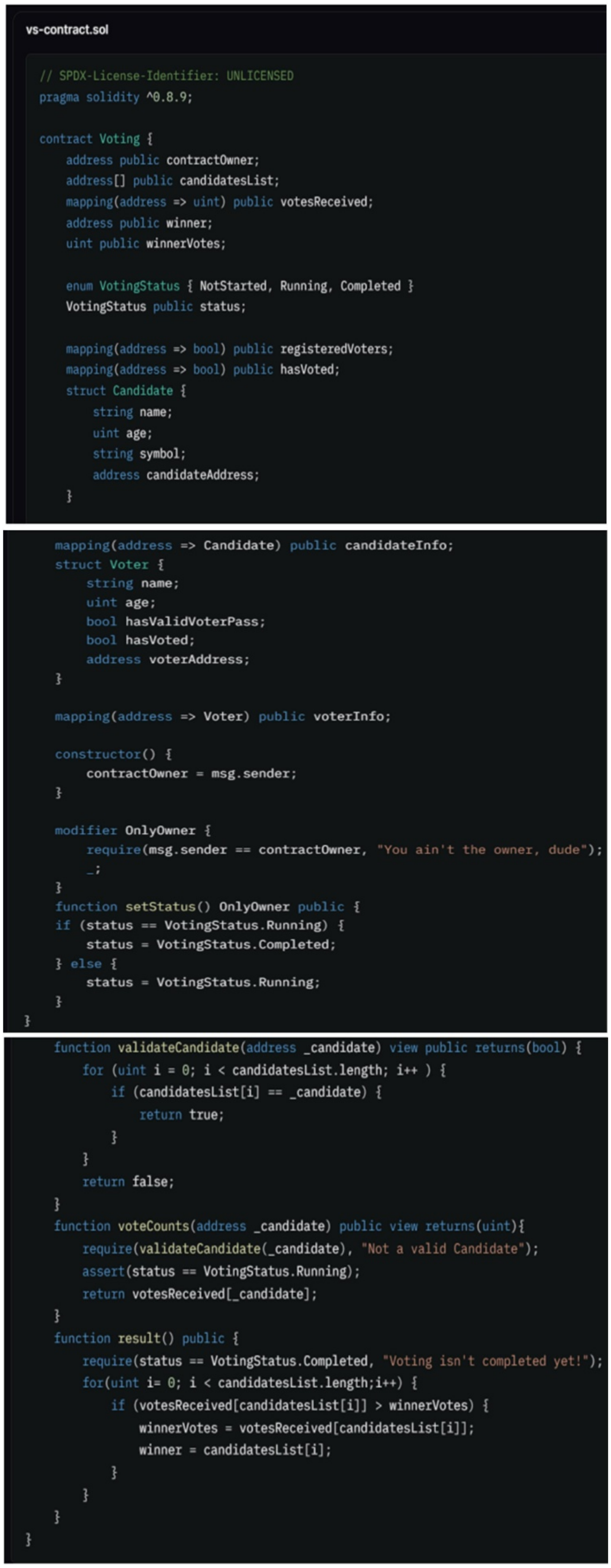
Additional Data:

1. Voter's location (for geo-based voting)
2. Voting history (for auditing purposes)
3. Election results (for transparency)
4. Audit logs (for security and compliance)

(b)

Fig. 7. Information about the data in each block.

As the parameters such as average gas cost, average latency, average authentication delay, and average vote alteration are improved compared to existing approaches, the proposed system outperforms them. The pseudocode for average gas cost reduction using basic gas optimization techniques and its explanation are shown in Figure 6. Figure 7 shows the information about data in each block.



vs-contract.sol

```
// SPDX-License-Identifier: UNLICENSED
pragma solidity ^0.8.9;

contract Voting {
    address public contractOwner;
    address[] public candidatesList;
    mapping(address => uint) public votesReceived;
    address public winner;
    uint public winnerVotes;

    enum VotingStatus { NotStarted, Running, Completed }
    VotingStatus public status;

    mapping(address => bool) public registeredVoters;
    mapping(address => bool) public hasVoted;
    struct Candidate {
        string name;
        uint age;
        string symbol;
        address candidateAddress;
    }
```

```
    mapping(address => Candidate) public candidateInfo;
    struct Voter {
        string name;
        uint age;
        bool hasValidVoterPass;
        bool hasVoted;
        address voterAddress;
    }

    mapping(address => Voter) public voterInfo;

    constructor() {
        contractOwner = msg.sender;
    }

    modifier OnlyOwner {
        require(msg.sender == contractOwner, "You ain't the owner, dude");
        _;
    }
    function setStatus() OnlyOwner public {
    if (status == VotingStatus.Running) {
        status = VotingStatus.Completed;
    } else {
        status = VotingStatus.Running;
    }
}
```

```
    function validateCandidate(address _candidate) view public returns(bool) {
        for (uint i = 0; i < candidatesList.length; i++ ) {
            if (candidatesList[i] == _candidate) {
                return true;
            }
        }
        return false;
    }
    function voteCounts(address _candidate) public view returns(uint){
        require(validateCandidate(_candidate), "Not a valid Candidate");
        assert(status == VotingStatus.Running);
        return votesReceived[_candidate];
    }
    function result() public {
        require(status == VotingStatus.Completed, "Voting isn't completed yet!");
        for(uint i= 0; i < candidatesList.length;i++) {
            if (votesReceived[candidatesList[i]] > winnerVotes) {
                winnerVotes = votesReceived[candidatesList[i]];
                winner = candidatesList[i];
            }
        }
    }
}
```

Fig. 8. Implementation of smart contract functions.

IV. CONCLUSION

In the process of finding a good alternative to the limitations of conventional voting processes, various blockchain-based approaches for digital voting were investigated, due to the inherent security and transparent nature of blockchain. As existing blockchain-based methods face some limitations, this study attempted to overcome them and propose an improved approach. Using IPFS and efficient smart contracts, vote alteration, latency, and authentication delay were improved. Further security was obtained with pBFT, and the validation of each vote was also performed efficiently. Efficient cost reduction measures were taken in the design of the proposed high-quality smart contracts, resulting in fewer gas costs. The novelty of the proposed method lies in the perfect blend of efficient smart contracts, IPFS, and pBFT in the implementation of blockchain. The results showed that the proposed system outperformed existing methods.

Continuous research and development efforts are integral to the proposed solution. Recognizing the dynamic nature of technology and potential challenges, the proposed work commits itself to ongoing research and development. This ensures that the system remains adaptive to emerging technologies, scalable to accommodate growing user bases, and compliant with evolving legal standards. In the future, the use of advanced gas optimization techniques will be considered to further reduce the gas cost.

REFERENCES

[1] B. Jayakumari *et al.*, "E-voting system using cloud-based hybrid blockchain technology," *Journal of Safety Science and Resilience*, vol. 5, no. 1, pp. 102–109, Mar. 2024, https://doi.org/10.1016/j.jnlssr.2024.01.002.

[2] C. D. González, D. F. Mena, A. M. Muñoz, O. Rojas, and G. Sosa-Gómez, "Electronic Voting System Using an Enterprise Blockchain," *Applied Sciences*, vol. 12, no. 2, Jan. 2022, Art. no. 531, https://doi.org/10.3390/app12020531.

[3] M. H. Berenjestanaki, H. R. Barzegar, N. El Ioini, and C. Pahl, "Blockchain-Based E-Voting Systems: A Technology Review," *Electronics*, vol. 13, no. 1, Art. no. 17, Jan. 2024, https://doi.org/10.3390/electronics13010017.

[4] M. N. Neloy *et al.*, "A remote and cost-optimized voting system using blockchain and smart contract," *IET Blockchain*, vol. 3, no. 1, pp. 1–17, 2023, https://doi.org/10.1049/blc2.12021.

[5] U. Jafar, M. J. Ab Aziz, Z. Shukur, and H. A. Hussain, "A Systematic Literature Review and Meta-Analysis on Scalable Blockchain-Based Electronic Voting Systems," *Sensors*, vol. 22, no. 19, Jan. 2022, Art. no. 7585, https://doi.org/10.3390/s22197585.

[6] F. Þ. Hjálmarsson, G. K. Hreiðarsson, M. Hamdaqa, and G. Hjálmtýsson, "Blockchain-Based E-Voting System," in *2018 IEEE 11th International Conference on Cloud Computing (Cloud)*, San Francisco, CA, USA, Jul. 2018, pp. 983–986, https://doi.org/10.1109/cloud.2018.00151.

[7] S. El Kafhali, "Blockchain-Based Electronic Voting System: Significance and Requirements," *Mathematical Problems in Engineering*, vol. 2024, 2024, Art. no. 5591147, https://doi.org/10.1155/2024/5591147.

[8] F. Rabia, A. Sara, and G. Taoufiq, "Review on blockchain-based e-voting systems," in *Proceedings of the 2023 9th International Conference on Computer Technology Applications*, Vienna, Austria, May 2023, pp. 151–156, https://doi.org/10.1145/3605423.3605435.

[9] P. Sanjeeva, M. S. Sathwik, G. S. Prasad, G. P. Reddy, V. Sajwan, and B. Ganesh, "Decentralized and Automated Online Voting System using Blockchain Technology," in *15th International Conference on Materials*

Processing and Characterization (ICMPC 2023), 2023, vol. 430, Art. no. 01046, https://doi.org/10.1051/e3sconf/202343001046.

[10] A. Parmar, S. Gada, T. Loke, Y. Jain, S. Pathak, and S. Patil, "Secure E-Voting System using Blockchain technology and authentication via Face recognition and Mobile OTP," in *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, Kharagpur, India, Jul. 2021, pp. 1–5, https://doi.org/10.1109/ICCCNT51525.2021.9580147.

[11] U. Jafar, M. J. A. Aziz, and Z. Shukur, "Blockchain for Electronic Voting System—Review and Open Research Challenges," *Sensors*, vol. 21, no. 17, Jan. 2021, Art. no. 5874, https://doi.org/10.3390/s21175874.

[12] Md. R. Ahmed, F. M. J. M. S. Shamrat, Md. A. Ali, M. R. Mia, and Mst. A. Khatun, "The Future of Electronic Voting System Using Blockchain," *International Journal of Scientific & Technology Research*, vol. 09, no. 02, pp. 4131–4134, Feb. 2020.

[13] I. Purdon and E. Erturk, "Perspectives of Blockchain Technology, its Relation to the Cloud and its Potential Role in Computer Science Education," *Engineering, Technology & Applied Science Research*, vol. 7, no. 6, pp. 2340–2344, Dec. 2017, https://doi.org/10.48084/etasr.1629.

[14] B. E. Sabir, M. Youssfi, O. Bouattane, and H. Allali, "Towards a New Model to Secure IoT-based Smart Home Mobile Agents using Blockchain Technology," *Engineering, Technology & Applied Science Research*, vol. 10, no. 2, pp. 5441–5447, Apr. 2020, https://doi.org/10.48084/etasr.3394.

[15] S. S. Taher, S. Y. Ameen, and J. A. Ahmed, "Advanced Fraud Detection in Blockchain Transactions: An Ensemble Learning and Explainable AI Approach," *Engineering, Technology & Applied Science Research*, vol. 14, no. 1, pp. 12822–12830, Feb. 2024, https://doi.org/10.48084/etasr.6641.