# Design of a Novel Side Chaining Model for Improving the Performance of Security Aware E-Voting Applications


**Hemlata Wamanrao Kohad, Sunil Kumar, and Asha Ambhaikar**

**Abstract** Electronic voting (E-Voting) has been described as one of the most efficient methods of collecting consensus-based decisions about a particular entity. These systems are useful for a wide variety of application scales, which range from selecting candidates at small corporations to nationwide elections. But voting systems face inherent security and quality of service (QoS) issues, which limits their public-domain deployments. Due to many sources of vulnerability, such as mutability, poor traceability, reduced trust levels, and centralized computing design, these systems are susceptible to attack by hackers and other adversaries. The usage of blockchain-based computing models, in which each set of votes is translated into a transaction, and these transactions are kept inside smart contracts, can be used to tackle these problems. These smart contracts are turned into blocks and kept in a decentralized blockchain database for storage. This database uses an improved unidirectional linked list, where each block is connected to the next block via a unique hash value. Due to the uniqueness of connecting hashes, this model exhibits immutability, which is one of the main reasons for its use in e-Voting systems. Secondly, hashes are generated using a decentralized mining mechanism, due to which the blockchain database is stored on multiple nodes, and is resilient against denial of service (DoS), Sybil, masquerading, and other server-based attacks. Similarly, the blockchain model also possesses transparency, and traceability, which makes it an ideal candidate for e-Voting systems. But the delay of voting increases exponentially w.r.t. the number of transactions, which is due to the fact that the addition of each block requires mining nodes to generate a new unique hash, which requires scanning of the entire blockchain. This limits the scalability of the blockchain model, which makes it unusable for larger-scale networks. In order to remove this drawback, a novel sidechaining mechanism is proposed in this text, wherein sidechains are created and managed using a firefly optimization model, which uses a number of parties and cast votes per party parameters. Due to this dynamic model for sidechain creation and management, the

H. W. Kohad (✉)
Kalinga University, Naya Raipur, India
e-mail: hemlata.pangantiwar@gmail.com

S. Kumar · A. Ambhaikar
Naya Raipur, India



N. Marriwala et al. (eds.), *Mobile Radio Communications and 5G Networks*,
Lecture Notes in Networks and Systems 588,
https://doi.org/10.1007/978-981-19-7982-8_2

proposed method is capable of reducing transaction delay by 28% when compared with a single blockchain, and 16% when compared with static sidechain methods. Additionally, the model was tested on medium to large-scale e-Voting applications, and it was discovered that, when compared to other cutting edge models, it is capable of improving throughput by 8% and reducing storage cost by 18%. The proposed sidechain paradigm can be used for a wide range of e-Voting application deployments as a result of these benefits.

**Keywords** Blockchain · Sidechain · e-Voting · Machine learning · Delay · Storage · Throughput

## 1 Introduction

Blockchain is quickly becoming one of the most useful technologies for e-Voting due to its traceability, transparency, high-trustability, immutability, and distributed performance characteristics. While designing a blockchain-based e-Voting system (BES), design of a wide variety of layers is involved. These layers include, but are not limited to, storage layer, auditing layer, aggregation layer, administration layer, election management layer, etc. Figure 1 shows a typical BES model that allows for the visualization of these elements and their internal data flow.

The model initiates with election preparation services, wherein voter list, party list, election rules, etc. are decided. This information is given to an administration

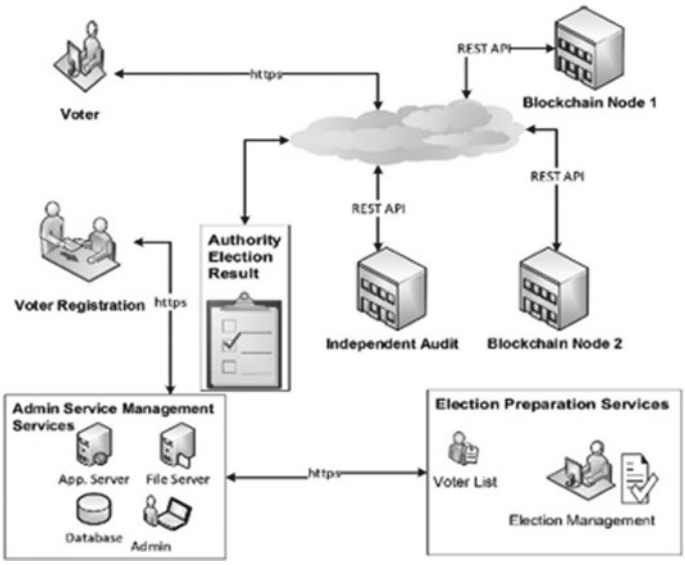


**Fig. 1** Typical blockchain-based e-voting model

module, where details about file server, database, administrator, application server, etc. are stored. These layers are synchronized using Governmental authorities, and election-specific rules are enforced. Based on these rules, the e-voting process is started, and voters are requested to cast their votes using smart phones, and other e-Platforms. Each set of votes is added to a smart contract, and these contracts are stored on multiple blockchain nodes [1]. The following actions are carried out while adding a block to the blockchain: The block is encrypted using high-efficiency encryption techniques like elliptic curve cryptography (ECC) Each block is hashed using a secure hashing algorithm (SHA) Before adding a block, it is checked for uniqueness, wherein the block hash is compared with hashes of already stored blocks. Based on these components, the following delay equation is modeled,

$$\text{New}_{\text{delay}} = N * \left(\text{Read}_{\text{delay}} + \text{Hash}_{\text{delay}} + \text{Check}_{\text{delay}}\right) + \text{Write}_{\text{delay}} \quad (1)$$

where $N$ represents a number of blocks already present in the chain, while New, Read, Hash, Check, and Write represents delay for adding a new block, delay for reading old blocks, hashing delay, delay for comparing current block with other blocks, and delay for writing the block to the blockchain. It can be shown from Eq. 1 that as the number of blocks grows, so does the time required to add a new block to the blockchain. It has been noted that the delay for adding blocks using distributed blockchain (DB) [2], consortium blockchain [3], or smart contracts [4] follows a roughly exponential curve in relation to the number of blocks. Thus, various techniques are proposed for reducing this delay, which includes side chain creation, increasing block length, etc. The next section reviews these techniques and discusses their intricacies, benefits, drawbacks, and potential applications in further research. The design of the suggested firefly-based side chain creation and management paradigm, which aids in enhancing the functionality of E-Voting systems, is then presented. In Sect. 4, the suggested model is tested, validated, and contrasted with several state-of-the-art methods across a range of applications. This essay concludes with some thought-provoking remarks on the suggested model and offers a number of suggestions for improvements to increase its capacity for real-time performance.

## 2 Literature Review

For e-Voting with side chains, a wide range of system models are put out, and each of these models is used for a specific user scale. For instance, to increase the scalability of e-Voting systems, the work in [5, 6] suggests trust-enhanced blockchain and access control-based blockchain. Similar to this, the work in [7] suggests many uses for an Oracle-based open-source decentralized blockchain that can be utilized to boost performance. This performance is measured in terms of quality of service (QoS) parameters including transaction delay, memory requirement, and throughput. Blockchain are further used for flying ad-hoc networks [8],

corporate security systems [9], and industrial IoT [10] for solving multiple security and performance issues. All these models are equally applicable for e-Voting and are used while designing and optimizing block structures. Methods like delegated proof of stake (DPoS) [11], privacy preservation using smart contracts [12], Proof-of-Quality-Factor (PoQF) [13], and dynamic topology-aware blockchains [14] are further proposed by researchers for multiple application security. These models also find their usage in e-Voting via the reduction in delay and improvement in node-level scalability. In [15–17], a use case for these models is presented, in which a Blockchain-Assisted Certificateless Key Agreement Protocol is suggested for Internet of Vehicle (IoV) based voting scenarios, Secure Software-Defined Industrial Network voting scenarios, and Consensus Managed Reputation and Contract Theory-based IoV voting scenarios. By eliminating topological dependence, these protocols help to increase QoS and scalability performance.

Due to an increase in blockchain length, existing models adapt to side chaining, wherein the main blockchain is divided into multiple subparts, and each subpart is managed independently for secure communications. The work in [18–21] proposes such models, wherein researchers have discussed usage of Reputation-Based Coalition Formation, Streamlined Consensus models, Traffic Classification Services, and Trust Model for Malicious Node classification using side chains. These models assist in selecting side chaining as an alternative to single chained blockchains, which improves transaction speed while reducing dependency on singular chains. Based on this observation, the work in [22] proposes an Ethereum Smart Contracts-based E-Voting model, which reduces voting delay, and improves the efficiency of vote counting by storing multiple votes on a single block structure. A high trust model that uses Adjusted Blockchain Technology for trust (ABTT)-based voting is proposed in [23], wherein miner nodes are selected depending upon their location and computational efficiency. Similar models are put out in [24–26], where researchers talk about using IoT-based E-Voting systems, the privacy-preserving automatic voting tally mechanism (PriScore or PS), and Anti-Quantum E-Voting with an integrated Audit Function. These models are highly scalable, and reduce E-Voting complexity via truly distributed computing models. Based on these observations, a novel dual Genetic Algorithm (GA)-based model for the next section of this paper proposes side chain-based E-Voting. As seen in Sect. 4 of this text, where performance evaluation in terms of transaction delay, throughput, and storage capacity is exhibited for various scaling situations, this model is assessed on a wide variety of applications and compared with various state-of-the-art methodologies.

## 3 Proposed Novel Side Chaining Model for Improving Performance of Security-Aware E-Voting Applications

According to the literature review, the majority of current blockchain solutions for electronic voting are either not scalable or perform worse in terms of quality of service

(QoS) when compared to non-blockchain alternatives. Some side chain implementations are proposed for this purpose, but their complexity increases with respect to increase in E-Voting data. In order to design a highly scalable E-Voting system with lower complexity, this section discusses a novel side chaining model that is able to create and manage side chains depending upon the number of cast votes. Figure 2 shows the suggested model and specifics of how it operates internally, as well as how side chains are created and managed. The current blockchain is observed to be reviewed for each new request, and either new sidechains are created or old ones are updated.

Thus, two different Genetic Algorithm models are required for managing the blockchains. The Genetic Algorithm (GA) is designed for new sidechain creation, in order to optimize sidechain length, and the number of votes stored per chain uses the original blockchain and divides it into multiple parts. The proposed GA model employs the following steps to function:

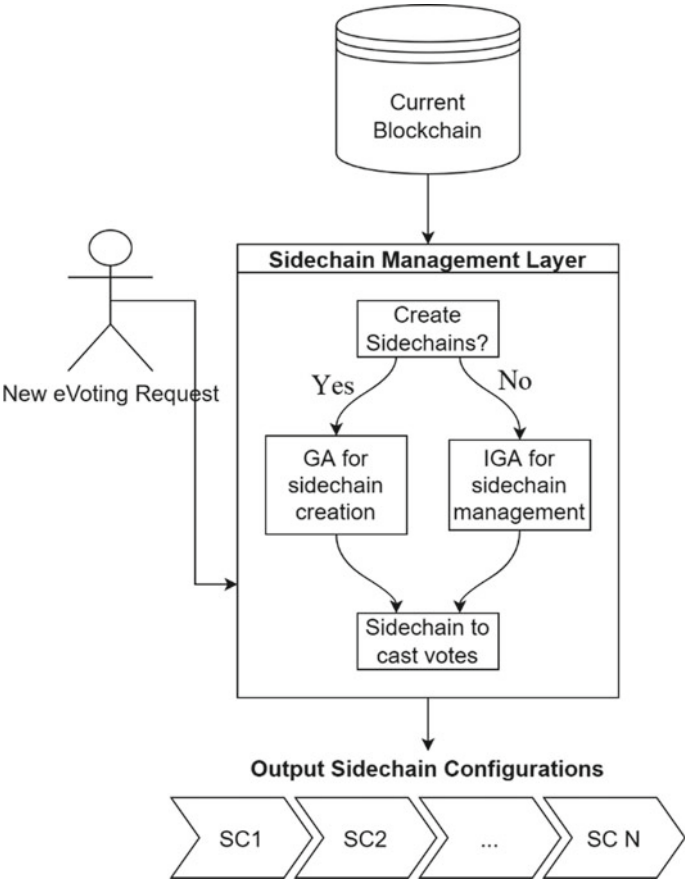


**Fig. 2** Overall structure of the suggested model

Initialize GA parameters

Number of iterations ($N_i$)
Number of solutions ($N_s$)
Learning rate ($L_r$)
Voting time limit (max DV)
The blockchain's maximum length (max BL)

Mark each answer as "to be modified."
For each of the 1 to $N_s$ solutions, from 1 through $N_i$ iterations.
Go to the next solution if the first one is marked as "not to be modified."
Otherwise, create a new answer using the procedures below.
Using Eq. 2, divide the current blockchain into $n$ random, unequal-sized chunks (Fig. 2).

$$n = \text{random}\left(\max_{\text{BL}} * L_r, \max_{\text{BL}}\right) \tag{2}$$

Number of blocks (NB) in each part is estimated using Eq. 3,

$$\text{NB} = \text{random}\left(\frac{\max_{\text{BL}}}{n}, B_{\text{REM}}\right) \tag{3}$$

where $B_{\text{REM}}$ represents number of blocks remaining to be allocated in sidechains. Select a random chain $r$ out of these sidechains, and cast a dummy vote into this chain.

While adding a vote, estimate its voting delay using Eq. 4,

$$\text{DV} = \left(D_{\text{read}_r} + D_{\text{check}_r} + D_{\text{hash}_r}\right) * \text{NB}_r + D_{\text{write}_r} \tag{4}$$

where DV, $D_{\text{read}_r}$, $D_{\text{check}_r}$, $D_{\text{hash}_r}$, $\text{NB}_r$, and $D_{\text{write}_r}$ represents voting delay, delay to read one block from current chain, delay to compare hashes of one block from current chain, delay to hash the current block, number of blocks in current chain, and delay to write a block to the current chain, respectively.

Accept this solution, only if $\text{DV} < \max_{\text{DV}}$.

Else, again split the blockchain into multiple parts, and regenerate a new solution.

If, $N_s * N_i$ combinations have been evaluated, and solution is not found, then create a new sidechain and cast all votes into this sidechain.

Evaluate fitness for this solution using Eq. 5,

$$f = \text{DV} * \frac{n}{\text{BL}^2} * \frac{\sum_{i=1}^{r} \text{NB}_i}{r} \tag{5}$$

where $r$ represents number of sidechains created by this solution.

Repeat this method for each solution, and then use Eq. 6 to estimate fitness threshold.

$$f_{\text{th}} = \frac{\sum_{i=1}^{N_s} f_i * L_r}{N_s} \tag{6}$$

where fitness exceeds the threshold, mark all solutions as "to be mutated," and "not to be altered." Choose the solution with the lowest fitness value after all iterations, and estimate the following parameters: ($N_{SC}$) Number of sidechains produced Each sidechain's total number of votes ($N_{V_{SC}}$)

Selected sidechain for casting the vote ($\text{Sel}_{SC}$)

Cast current vote into this sidechain, and use the given configuration for casting future votes. While casting these votes, check DV levels, and if DV $>$ $\max_{DV}$, then a new sidechain must be selected or created. The following Iterative Genetic Algorithm (IGA) model is triggered to carry out this task: establishing IGA parameters.

Number of iterations ($N_i$)
Number of solutions ($N_s$)
Learning rate ($L_r$)
Voting time limit (Max DV)

Mark each answer as "to be modified."

From 1 through $N_i$ iterations, for each of the 1 to $N_s$ solutions go to the next solution if the first one is marked "not to be modified." Otherwise, create a new solution using the procedures below.

Select a random chain *r* out of the current sidechains, and cast a dummy vote into this chain.

- While adding a vote, estimate its voting delay using Eq. 4,
- Accept this solution, only if DV $<$ $\max_{DV}$
- Else, select a new sidechain for casting the vote.
- If, $N_s$ combinations have been evaluated, and solution is not found, then create a new sidechain and cast all votes into this sidechain.
- Evaluate fitness for this solution using Eq. 5.
- Carry out this procedure for each solution, and then use Eq. 6 to get the fitness threshold.
- Mark all solutions where fitness exceeds threshold as "to be mutated," and mark other solutions as "not to be mutated."
- Select the sidchain with the lowest fitness value after all iterations, and estimate the following parameters:
- Sidechain number ($\text{Num}_{SC}$)
- Number of votes in current sidechain ($N_{V_{CSC}}$)

Cast current vote into selected sidechain, and use the given configuration for casting future votes. Repeat the process if DV $>$ $\max_{DV}$, which assists in formation of newer sidechains. Based on this process, a wide variety of sidechains with different sizes, and different votes per chain can be created. These sidechains are tested on numerous e-Voting applications, and the results are tallied in terms of transaction delay, throughput, and storage expense. In the following section of this article, it is

described how the proposed model stacks up against alternative techniques based on these parameters.

## 4 Results Analysis and Comparison

Different optimization methods are used for sidechain creation and maintenance in the proposed dual GA architecture. The proposed sidechain concept was put to the test in a variety of e-Voting scenarios to gauge its effectiveness, including

- Small-scale e-Voting, which is used to form consensus for less than 5 parties.
- Moderate-scale e-Voting, which is used to form consensus for less than 10 parties.
- Large-scale e-Voting, which is used to form consensus for more than 10 parties (Fig. 3).

Based on these scenarios, smart contracts were deployed using Ethereum blockchain in Solidity. Each scenario was evaluated terms of transaction delay (TD), throughput (Th), and storage cost (S). These values were estimated for the proposed model by varying number of votes casted by users, and were compared with DB [2], ABTT [23], and PS [25] for validating its performance. For small-scale voting application, number of votes casted were varied between 200 and 2000, and 5 participating candidates were considered during evaluation. Based on these parameters, transaction delay was estimated (Fig. 4).

The suggested model was found to be 15% quicker than DB [2], 8% faster than ABTT [23], and 26% faster than PS [25] for small-scale voting applications due to the usage of numerous GA models that are customized to reduce transaction delay. Similarly, the throughput (measured in Megabits per second, or Mbps), which represents the number of blocks mined in a unit of time, was seen for each of these models in Fig. 5.

The suggested model was found to be 61% quicker than DB [2], 9% faster than ABTT [23], and 28% faster than PS [25] for small-scale voting applications due to the usage of numerous GA models that are tweaked to optimize sidechain length. The storage cost (in Megabits or Mb), which represents the average number of blocks stored per chain, was similarly displayed for each of these models in Fig. 6.

The proposed model was observed to be 80% better than DB [2], 15% better than ABTT [23], and 40% better than PS [25] for small-scale voting applications in terms of storage cost due to the use of multiple GA models that are tuned to optimize sidechain length and number of blocks per sidechain. These benefits enable the suggested methodology to be implemented in small-scale e-Voting systems.

Similarly, for the moderate-scale voting application scenario, number of votes casted were varied between 8000 and 100 k, and 10 participating candidates were considered during evaluation. Based on these parameters, transaction delay was shown in Fig. 7.

The suggested model was found to be 25% faster than PS [25], 26% faster than ABTT [23], and 18% faster than DB [2] for intermediate scale voting applications

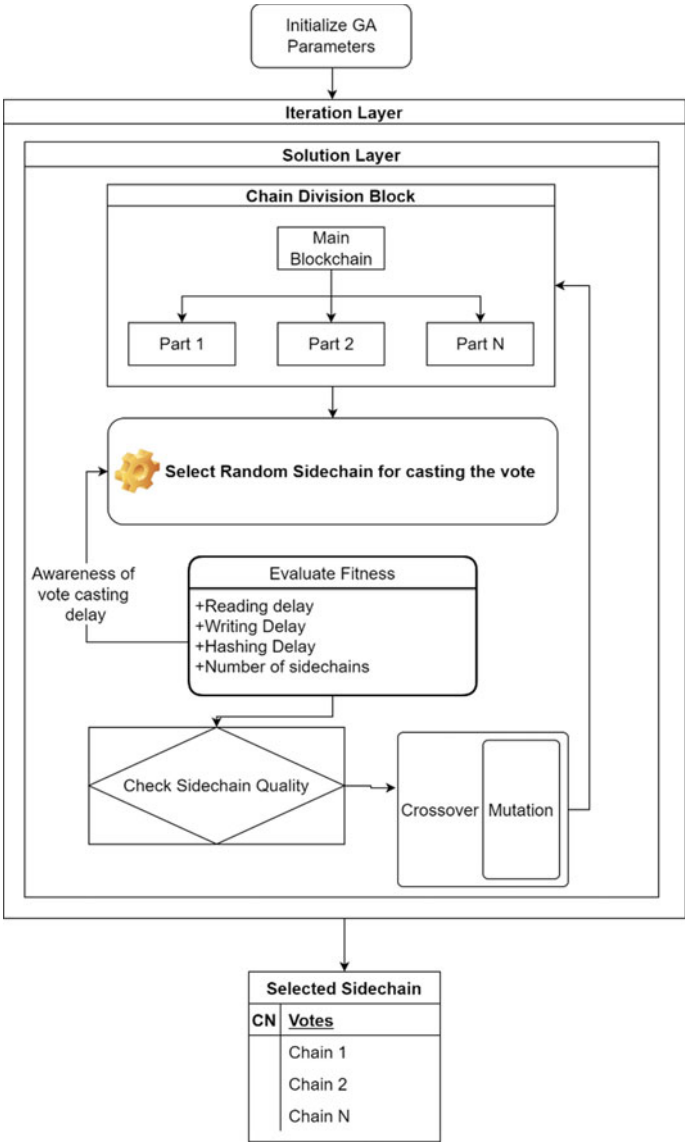


**Fig. 3** Flow of the designed sidechain creation and management genetic algorithm (GA) model

due to the usage of numerous GA models that are optimized to reduce transaction delay. Similar to this, Fig. 8's throughput (in Megabits per second or Mbps), which represents the number of blocks mined per unit time, was displayed for each of these models.

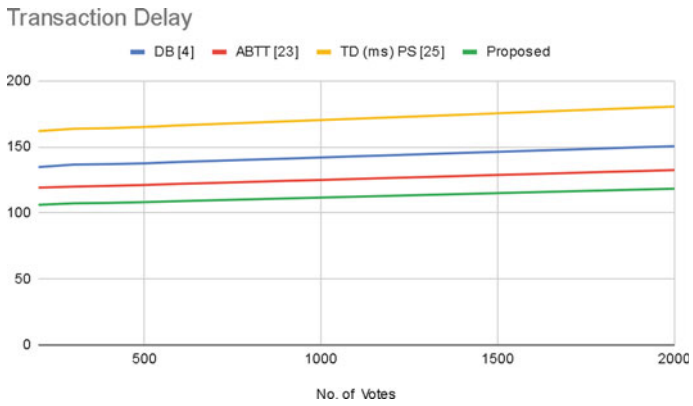


**Fig. 4** Transaction delay performance for different algorithms on small-scale voting applications

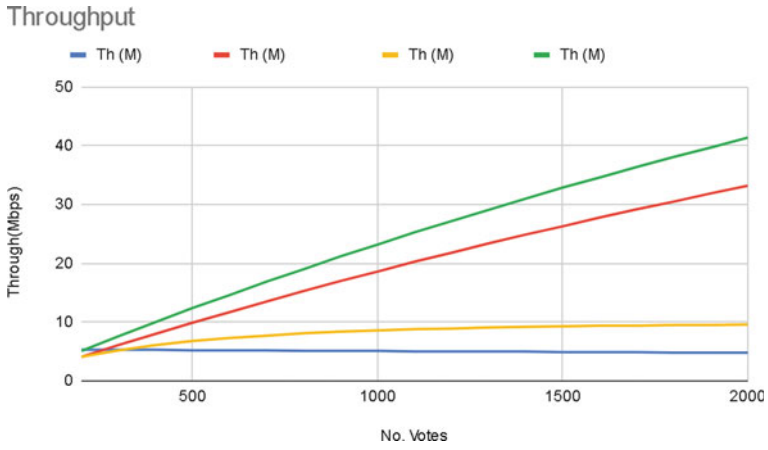


**Fig. 5** Throughput performance for different algorithms on small-scale voting applications

The suggested model was found to be 20% faster than DB [2], 6% faster than ABTT [23], and 19% faster than PS [25] for moderate-scale voting applications due to the usage of numerous GA models that are tweaked to optimize sidechain length. For each of these models, the storage cost (in Megabits or Mb), which reflects the average number of blocks stored per chain, is presented in Fig. 9.

In terms of storage cost, the suggested model was found to be 20% better than DB [2], roughly the same as ABTT [23], and 10% better than PS [25] due to the usage of various GA models that are modified to optimize sidechain length & number of blocks per sidechain. These benefits make the suggested model suitable for deployment in any intermediate scale e-Voting application.

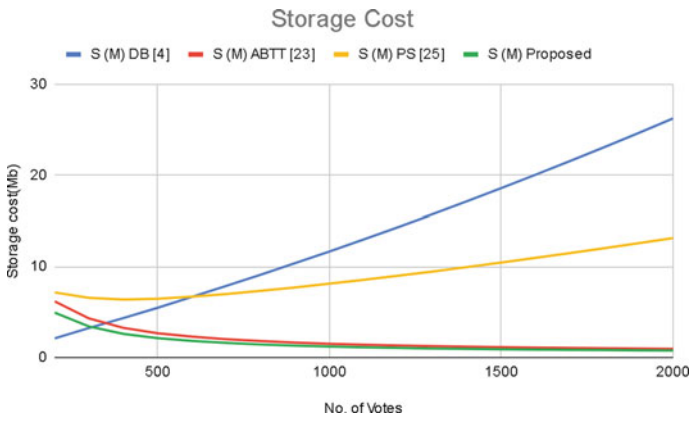


**Fig. 6** Storage cost for different algorithms on small-scale voting applications

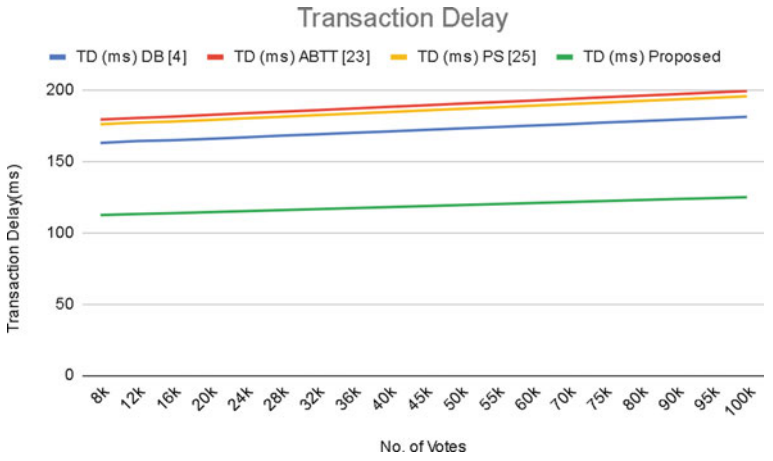


**Fig. 7** Transaction delay performance for different algorithms on moderate-scale voting applications

Similarly, for the large-scale voting application scenario, number of votes casted were varied between 500 k and 20 M, and 45 participating candidates were considered during evaluation. Based on these parameters, transaction delay is shown in Fig. 10.

The suggested model was found to be 18% quicker than DB [2], 16% faster than ABTT [23], and 10% faster than PS [25] for large-scale voting applications due to the usage of numerous GA models that are customized to reduce transaction delay. For each of these models, the throughput (measured in Megabits per second, or Mbps), which represents the number of blocks mined per unit time, is displayed in Fig. 11.

Due to use of multiple GA models, which are tuned to optimize sidechain length, the proposed model was observed to have16% better throughput than DB [2], 25%

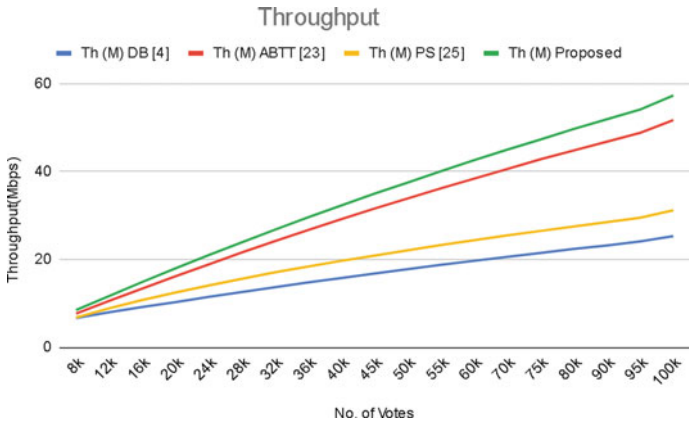


**Fig. 8** Throughput performance for different algorithms on moderate-scale voting applications

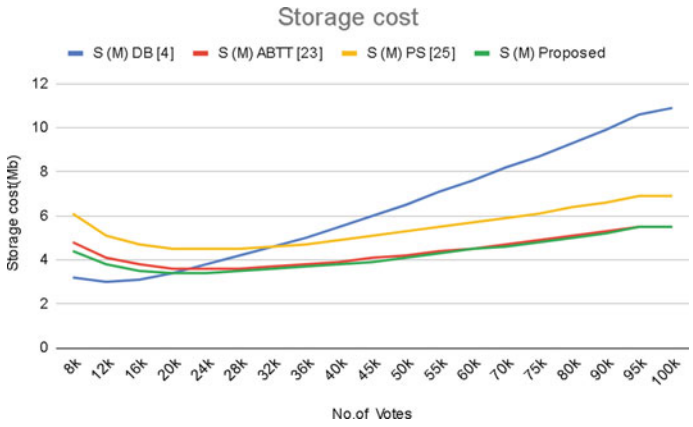


**Fig. 9** Storage cost for different algorithms on moderate-scale voting applications

better throughput than ABTT [23], and 26% better throughput than PS [25] for large-scale voting applications. Similarly, the storage cost (in Megabits or Mb), which indicates average number of blocks stored per chain, for each of these models is shown in Fig. 12.

The proposed model was observed to be 25% better than DB [2], 10% better than ABTT [23], and 8% better than PS [25] for large-scale voting applications in terms of storage cost due to the use of multiple GA models that are tuned to optimize sidechain length and number of blocks per sidechain. These benefits make the suggested approach suitable for use in any form of extensive e-Voting application. The usage of multiple GA models for sidechain construction and maintenance causes the suggested model to be seen as being superior than a number of state-of-the-art techniques. These models enable the sidechain management technique to be

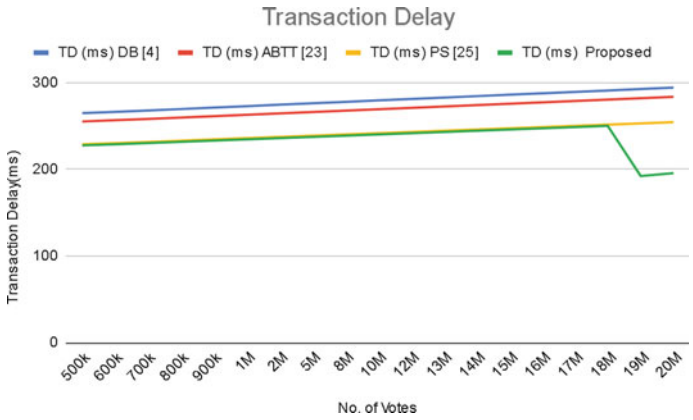


**Fig. 10** Transaction delay performance for different algorithms on large-scale voting applications

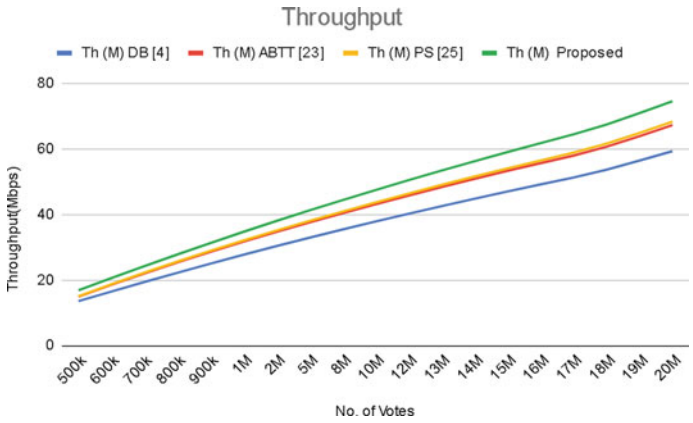


**Fig. 11** Throughput performance for different algorithms on large-scale voting applications

presented to choose configurations that decrease transaction time and increase e-Voting throughput. As a result, the suggested model can be used for small, medium, and large-scale e-Voting systems.

## 5 Conclusion and Future Scope

The suggested concept combines two separate types of gas to create and manage sidechains. The sidechain creation GA is capable of selecting optimum sidechain lengths, and main blockchain divisions, which are optimized for improving transaction speed. The created sidechains are managed using another GA method, which

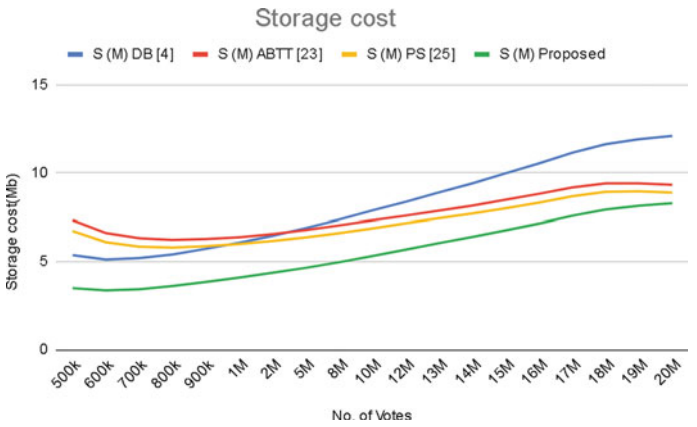


**Fig. 12** Storage cost for different algorithms on large-scale voting applications

assists in optimal creation and usage of existing sidechain configuration model. Both the GA models utilize transaction delay, and sidechain lengths in order to improve e-Voting performance for small, medium, and large-scale deployments. As a result, it was found that the proposed model was 15% faster than DB [2], 8% faster than ABTT [23], and 26% faster than PS [25] for small-scale voting applications, 18% faster than DB [2], 26% faster than ABTT [23], and 25% faster than PS [25] for medium-scale voting applications, and 18% faster than DB [2], 16% faster than ABTT [23], and 10% faster than PS [25] for large-scale. For small-scale voting applications, the suggested model was found to be 61% faster than DB [2], 9% faster than ABTT [23], and 28% faster than PS [25], while for moderate-scale voting, it was shown to be 20% faster than DB [2], 6% faster than ABTT [23], and 19% faster than PS [25]. For large-scale voting applications, the throughput was 26% higher than PS [25], 25% higher than ABTT [23], and 16% higher than DB [2]. Similar findings were reached regarding the cost of storage, which makes the suggested approach highly scalable and applicable to a variety of e-Voting systems. Future evaluations of the suggested methodology for more blockchain applications will help determine its versatility. The performance of the proposed model can be further enhanced by applying deep learning techniques like reinforcement learning and Q-learning, which will help in optimizing sidechain parameters through reward-based learning mechanisms, accelerating transaction performance, and further improving throughput for a variety of side chain applications.

## References


1. Li H, Li Y, Yu Y, Wang B, Chen K (2021) A blockchain-based traceable self-tallying e-voting protocol in AI era. IEEE Trans Netw Sci Eng 8(2):1019–1032. https://doi.org/10.1109/TNSE.2020.3011928
2. Zaghloul E, Li T, Ren J (2021) d-BAME: distributed blockchain-based anonymous mobile electronic voting. IEEE Internet of Things J 8(22):16585–16597. https://doi.org/10.1109/JIOT.2021.3074877
3. Sun G, Dai M, Sun J, Yu H (2021) Voting-based decentralized consensus design for improving the efficiency and security of consortium blockchain. IEEE Internet of Things J 8(8):6257–6272. https://doi.org/10.1109/JIOT.2020.3029781
4. Panja S, Bag S, Hao F, Roy B (2020) A smart contract system for decentralized Borda count voting. IEEE Trans Eng Manage 67(4):1323–1339. https://doi.org/10.1109/TEM.2020.2986371
5. Wu D, Ansari N (2021) A trust-evaluation-enhanced blockchain-secured industrial IoT system. IEEE Internet of Things J 8(7):5510–5517. https://doi.org/10.1109/JIOT.2020.3030689
6. Bera B, Saha S, Das AK, Vasilakos AV (2021) Designing blockchain-based access control protocol in IoT-enabled smart-grid system. IEEE Internet of Things J 8(7):5744–5761. https://doi.org/10.1109/JIOT.2020.3030308
7. Nelaturu K et al (2020) On public crowdsource-based mechanisms for a decentralized blockchain Oracle. IEEE Trans Eng Manage 67(4):1444–1458. https://doi.org/10.1109/TEM.2020.2993673
8. Tan Y, Liu J, Kato N (2021) Blockchain-based key management for heterogeneous flying Ad Hoc network. IEEE Trans Industr Inf 17(11):7629–7638. https://doi.org/10.1109/TII.2020.3048398
9. Huang D, Ma X, Zhang S (2020) Performance analysis of the raft consensus algorithm for private blockchains. IEEE Trans Syst Man Cybern: Syst 50(1):172–181. https://doi.org/10.1109/TSMC.2019.2895471
10. Jiang S, Cao J, Wu H, Yang Y (2021) Fairness-based packing of industrial IoT data in permissioned blockchains. IEEE Trans Industr Inf 17(11):7639–7649. https://doi.org/10.1109/TII.2020.3046129
11. Xu G, Liu Y, Khan PW (2020) Improvement of the DPoS consensus mechanism in blockchain based on vague sets. IEEE Trans Industr Inf 16(6):4252–4259. https://doi.org/10.1109/TII.2019.2955719
12. Tran QN, Turnbull BP, Wu H-T, de Silva AJS, Kormusheva K, Hu J (2021) A survey on privacy-preserving blockchain systems (PPBS) and a novel PPBS-based framework for smart agriculture. IEEE Open J Comp Soc 2:72–84. https://doi.org/10.1109/OJCS.2021.3053032
13. Ayaz F, Sheng Z, Tian D, Guan YL (2021) A proof-of-quality-factor (PoQF)-based blockchain and edge computing for vehicular message dissemination. IEEE Internet of Things J 8(4):2468–2482. https://doi.org/10.1109/JIOT.2020.3026731
14. Li X, Han B, Li G, Luo L, Wang K, Jiang X (2021) Dynamic topology awareness in active distribution networks using blockchain-based state estimations. IEEE Trans Power Syst 36(6):5185–5197. https://doi.org/10.1109/TPWRS.2021.3070390
15. Chattaraj D, Bera B, Das AK, Saha S, Lorenz P, Park Y (2021) Block-CLAP: blockchain-assisted certificateless key agreement protocol for internet of vehicles in smart transportation. IEEE Trans Veh Technol 70(8):8092–8107. https://doi.org/10.1109/TVT.2021.3091163
16. Singh M, Aujla GS, Singh A, Kumar N, Garg S (2021) Deep-learning-based blockchain framework for secure software-defined industrial networks. IEEE Trans Industr Inf 17(1):606–616. https://doi.org/10.1109/TII.2020.2968946
17. Kang J, Xiong Z, Niyato D, Ye D, Kim DI, Zhao J (2019) Toward secure blockchain-enabled internet of vehicles: optimizing consensus management using reputation and contract theory. IEEE Trans Veh Technol 68(3):2906–2920. https://doi.org/10.1109/TVT.2019.2894944

18. Asheralieva A, Niyato D (2020) Reputation-based coalition formation for secure self-organized and scalable sharding in IoT blockchains with mobile-edge computing. IEEE Internet Things J 7(12):11830–11850. https://doi.org/10.1109/JIOT.2020.3002969
19. Santiago C, Ren S, Lee C, Ryu M (2021) Concordia: a streamlined consensus protocol for blockchain networks. IEEE Access 9:13173–13185. https://doi.org/10.1109/ACCESS.2021.3051796
20. Qi H, Wang J, Li W, Wang Y, Qiu T (2021) A blockchain-driven IIoT traffic classification service for edge computing. IEEE Internet of Things J 8(4):2124–2134. https://doi.org/10.1109/JIOT.2020.3035431
21. She W, Liu Q, Tian Z, Chen J-S, Wang B, Liu W (2019) Blockchain trust model for malicious node detection in wireless sensor networks. IEEE Access 7:38947–38956. https://doi.org/10.1109/ACCESS.2019.2902811
22. Giraldo FD, Barbosa Milton C, Gamboa CE (2020) Electronic voting using blockchain and smart contracts: proof of concept. IEEE Latin Am Trans 18(10):1743–1751. https://doi.org/10.1109/TLA.2020.9387645
23. Shahzad B, Crowcroft J (2019) Trustworthy electronic voting using adjusted blockchain technology. IEEE Access 7:24477–24488. https://doi.org/10.1109/ACCESS.2019.2895670
24. Rathee G, Iqbal R, Waqar O, Bashir AK (2021) On the design and implementation of a blockchain enabled e-voting application within IoT-oriented smart cities. IEEE Access 9:34165–34176. https://doi.org/10.1109/ACCESS.2021.3061411
25. Yang Y, Guan Z, Wan Z, Weng J, Pang HH, Deng RH (2021) PriScore: blockchain-based self-tallying election system supporting score voting. IEEE Trans Inf Forensics Secur 16:4705–4720. https://doi.org/10.1109/TIFS.2021.3108494
26. Gao S, Zheng D, Guo R, Jing C, Hu C (2019) An anti-quantum e-voting protocol in blockchain with audit function. IEEE Access 7:115304–115316. https://doi.org/10.1109/ACCESS.2019.2935895