A Fast Adaptive Blockchain Consensus Algorithm via Wlan Mesh Network

Mingzhe Liu[1,2], Xin Jiang[1], Feixiang Zhao[1], Xuyang Feng[2], Ruili Wang[3]

[1] State Key Laboratory of Geohazard Prevention and Geoenvironment Protection, Chengdu University of Technology, China
[2] School of Cyberspace Security, Chengdu University of Technology, China
[3] School of Natural and Computational Sciences, Massey University, New Zealand
liumz@cdut.edu.cn, jiangxin@cdut.edu.cn, zhaofeixiang@cdut.edu.cn, xuyang.f@foxmail.com, Ruili.wang@massey.ac.nz

Abstract

This paper presents a decentralised and fast adaptive block chain's consensus algorithm with maximum voter privacy using wlan mesh network. The algorithm is suitable for consortium blockchain and private blockchain, and is written as a smart contract for Hyperledger Fabric. Unlike previously proposed blockchain's consensus protocols, this is the first implementation that does not rely on any trusted authority to compute the tally or to protect the voter's privacy. Instead, the algorithm is a fast adaptive protocol, and each voter is in control of the privacy of their own vote such that it can only be breached by a full collusion involving all other voters. The execution of the protocol is enforced using the consensus mechanism that also secures the Fabric blockchain. The implementation on Fabric's official test network is conducted to demonstrate its feasibility. Also, this paper provides a computational breakdown of its execution cost.

Keywords: Consensus algorithm, Blockchain, Wlan mesh, Hyperledger fabric

1 Introduction

Hyperledger Fabric is the most popular cryptocurrency of blockchain as of 2018. It relies on the same innovation behind Bitcoin [1]: namely, the blockchain is an append-only ledger maintained by a decentralised and open-membership peer-to-peer network. The purpose of the blockchain is to remove the centralised role of banks for maintaining a financial ledger. Many researchers are trying to reuse the Blockchain to solve some open problems such as coordinating the Internet of Things [2], carbon dating [3], and healthcare [4]. However, How to reach fast consensus has become a bottleneck in the development of blockchain in above fields.

In this paper, we focus on decentralised internet consensus of blockchain in wlan mesh network. This algorithm is designed as a voting protocol. E-voting protocols support verifiability which normally assumes the existence of a public bulletin board that provides a consistent view to all voters. In practice, an example of implementing the public bulletin board can be seen in the yearly elections of the International Association of Cryptologic Research (IACR) [5]. They use the Helios voting system [6] in which bulletin board is implemented as a single web server. This server is trusted to provide a consistent view to all voters. Instead of such a trust assumption, we explore the feasibility of using the blockchain as a public bulletin board. Furthermore, we consider a decentralised election setting in which the voters are responsible for coordinating the communication amongst themselves.

2 Related Works

In recent years, the unique advantages of blockchain technology have been attracted much attention from academic fields. There are some research methods of blockchain in consensus algorithm. BBLAST system solved the problem of low trust centralized ledger held by a single third party, high trust decentralized forms held by different entities, or in other words, verification nodes [7].

Gramoli [8] discussed the mainstream blockchain consistency algorithm and the classical Byzantine consensus to re-examine the blockchain context. To against bitcoin and the Ethum consensus algorithms, Ref. [9] worked to prove consensus algorithm of mining dilemma in the process of analyzing PoW (Proof of the work) strategy choice for the existence of Nash equilibrium. In general, this solution came from the perspective of game theory, analyzed the PoW consensus algorithm, and provided new ideas and methods for the further design of the consensus

*Corresponding Author: Mingzhe Liu; E-mail: liumz@cdut.edu.cn
DOI: 10.3966/160792642020032102020

algorithm based on game theory.

To overcome performance problem, [10] proposed a public supply chain system based on double-chain structure. The results have shown that the two-chain structure of supply chain based on agricultural product supply chain could consider the openness and security of transaction information as well as the privacy of enterprise information, and could achieve rent-seeking and matching adaptively. Ref. [11] proposed a new blockchain consistency algorithm, introduced the two-stage delivery and quorum vote process, and solved the legitimacy verification problem in decentralized environment by using the distributed ledger feature of the blockchain protocol.

Compared with the traditional Byzantine consensus, the algorithm reduced the number of message passing, improved system fault tolerance. [12] instantiated a provably secure OKSA solution at transport stage, reduced a round interaction and constant communication cost. The analysis and evaluation showed that the search chain could remain reasonably cost effective without loss of retrieval privacy.

In [13] a security certification scheme is proposed for managing human-centric solutions (SAMS) to verify resource information in participating mobile devices and processing in MRM resource pools. In order to verify the SAMS of MRM, the data of FalsifICA was tested. The test results showed that data tampering was impossible. To achieve the fault tolerance of the XFT consensus algorithm, the solution proposed by [14] presented a Byzantine consensus algorithm based on the Gossip protocol, which could enable the system to tolerate less than half of the nodes as byzantines. At the same time, the system has better scalability, and it is helpful to identify malicious nodes for correct nodes in the blockchain system since the unified data structure is adopted.

In order to improve operation efficiency of the consensus algorithm in blockchain system, [15] introduced various potential of blocks in the chain of consensus optimization scheme by combining with aggregate signature technology and bilinear mapping technology. The optimized aggregate dBFT consensus algorithm can effectively reduce the space complexity of signature in blockchain system. Ref. [16] explored how different network conditions can change the results of consistency between nodes. Besides, Bach's analysis in [17] focused on algorithm steps: the scalability algorithm adopted by each consistent algorithm, and the time and security risks of the algorithm reward validator verification block in the algorithm.

Under various assumptions with the advent of chains, various consensuses have come to high performance. However, the Byzantine fault tolerance (BFT) protocol is not suitable for this. For a high number of participants, it must be accommodated. Regarding to performance and security, one can use novel network techniques and trusted computing in [18-19].

3 Method

3.1 Consensus Algorithm

The consensus algorithm is a decentralized two-round protocol designed for small-scale boardroom voting. In the first round, all voters register their intention to vote in the election, and in the second round, all voters cast their votes. The system assumes an authenticated broadcast channel available to all voters. The self-tallying property allows anyone (including non-voters) to compute the tally after observing messages from the other voters. In this paper, we only consider an election with two options, e.g., yes/no. Extending to multiple voting options, and a security proof of the protocol can be referred in [20].

A description of the open vote network is as follows. All n voters agree on (G, g) where G denotes a finite cyclic group of prime order q which the Decisional Diffie-Hellman (DDH) problem is intractable, and g is a generator in G. A list of eligible voters $(P_1, P_2, \ldots, P_n)$ is established and each eligible voter P_i selects a random value as their private voting key.

3.1.1 Election register

Every voter P_i broadcasts their voting key g^{x_i} and a (non-interactive) zero knowledge proof $ZMP(x_i)$ to prove knowledge of the exponent x_i on the public bulletin board. $ZMP(x_i)$ is implemented as a Schnorr proof [21] made non-interactive using the Fiat Shamir heuristic [22].

At the end, all voters check the validity of all zero knowledge proofs before computing a list of reconstructed keys:

$$Y_i = \prod_{j=1}^{i-1} g^{x_j} / \prod_{j=i+1}^{n} g^{x_j} \quad \textbf{(1)}$$

Implicitly setting $y_i = g^{y_i}$, the above calculation ensures $\Sigma_i x_i y_i$.

3.1.2 Vote round

Every voter broadcasts $g^{x_i y_i} g^{v_i}$ and a (non-interactive) zero knowledge proof to prove that v_i is either no or yes (with respect to 0 or 1) vote. This one-out-of-two zero knowledge proof is implemented using the Cramer, Damgrd and Schoenmakers (CDS) technique.

All zero knowledge proofs must be verified before computing the tally to ensure the encrypted votes are well-formed. Once the final vote has been cast, then

anyone (including non-voters) can compute $\prod_i g^{x_i y_i} g^{v_i}$ and calculate $g^{\Sigma_i v_i}$ since $\prod_i g^{x_i y_i} = 1$. The discrete logarithm of $g^{\Sigma_i v_i}$ is bounded by the number of voters and is a relatively small value. Hence the tally of yes votes can be calculated subsequently by exhaustive search.

Note that for the election tally to be computable, all the voters who have broadcasted their voting key in Round 1 must broadcast their encrypted vote in Round 2. Also note that in Round 2, the last voter to publish their encrypted vote has the ability to compute the tally before broadcasting their encrypted vote (by simulating that he would send a no-vote). Depending on the computed tally, one may change his/her vote choice. In our implementation, we address this issue by requiring all voters to commit to their votes before revealing them, which adds another round of commitment to the protocol.

The decentralised nature of the consensus algorithm makes it suitable to implement over wlan mesh network. Wlan mesh could be used as the private blockchain to store the voting data for the consensus algorithm.

3.2 Structure of Implementation

There are two smart contracts that are both written in Fabric's Solidity language. The first contract is called the voting contract. It implements the voting protocol, controls the election process and verifies the two types of zero knowledge proofs we have in the Open Vote Network. The second contract is called the cryptography contract. It distributes the code for creating two types of zero knowledge proofs 3. This provides all voters with the same cryptography code that can be used locally without interacting with the Fabric network. We have also provided three HTML5/JavaScript pages for the users, as shown in the Figure 1.

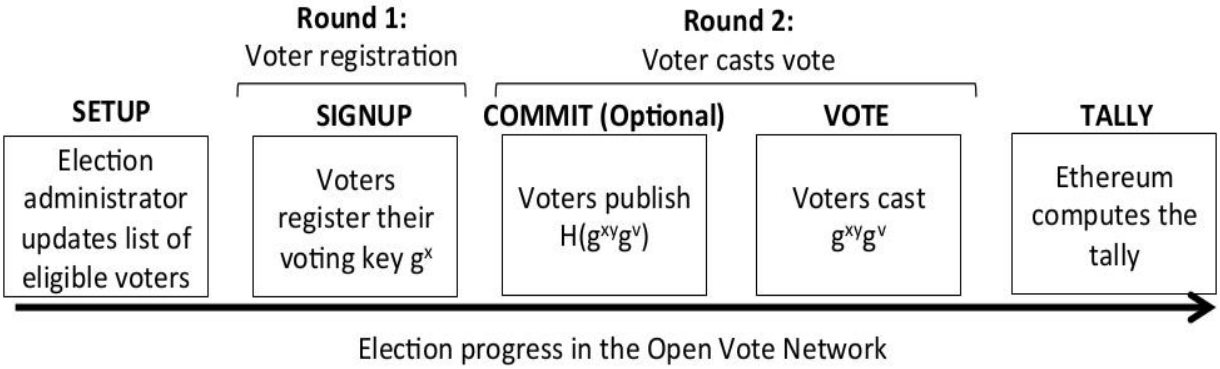

Figure 1. There are five stages to the election

We assume that voters and the election administrator have their own Fabric accounts. The Web3 framework is provided by the Ethereum Foundation to faciltiate communication between a user's web browser and their Fabric client. The user can unlock their Fabric account (decrypt their Fabric private key using a password) and authorise transactions directly from the web browser. There is no need for the user to interact with an Ethereum wallet, and the Fabric client can run in the background as a daemon.

3.3 Election Administrator

This includes establishing the list of eligible voters, setting the election question, and activating a list of timers to ensure the election progresses in a timely manner. The latter includes notifying Ethereum to begin registration, to close registration and begin the election, and to close voting and compute the tally, which is shown in Figure 2.

Voter can register for an election, and once registered must cast their vote. Observer can watch the election's progress consisting of the election administrator starting and closing each stage and voters registering and casting votes. The running tally is not computable.

3.4 Setup Scenarios

We defined 2 typical user setup scenarios: (1) New nodes – How to privatize your nodes first time. (2) Privatized nodes – How to re-privatize your nodes. When first open one node never been privatized, our nodes will be in public mesh state, user can use phone APP to privatize the devices and setup AP configurations. The new devices setup sequence is shown in Figure 3. The step of joining a privatized group is shown in Figure 4.

Definitions, Acronyms and Abbreviations: WMD is defined as WIFI manager daemon. Master is defined as the device which is in private mesh network and connected to AP. Slave: The device is in private mesh network. Public Mesh: Mesh network with all known mesh ID. Private Mesh: Mesh network with privatized mesh ID.

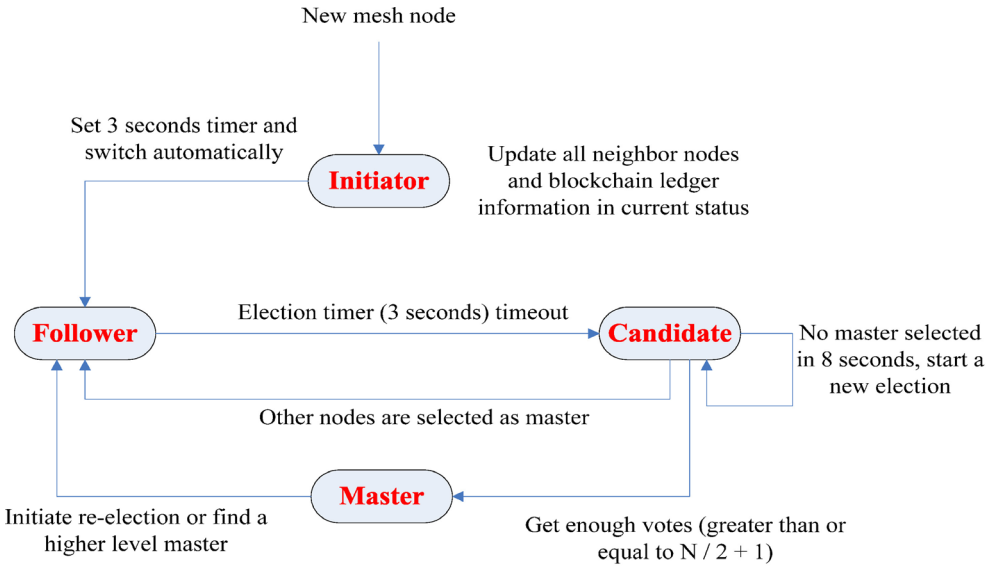

Figure 2. The process of Election administrator

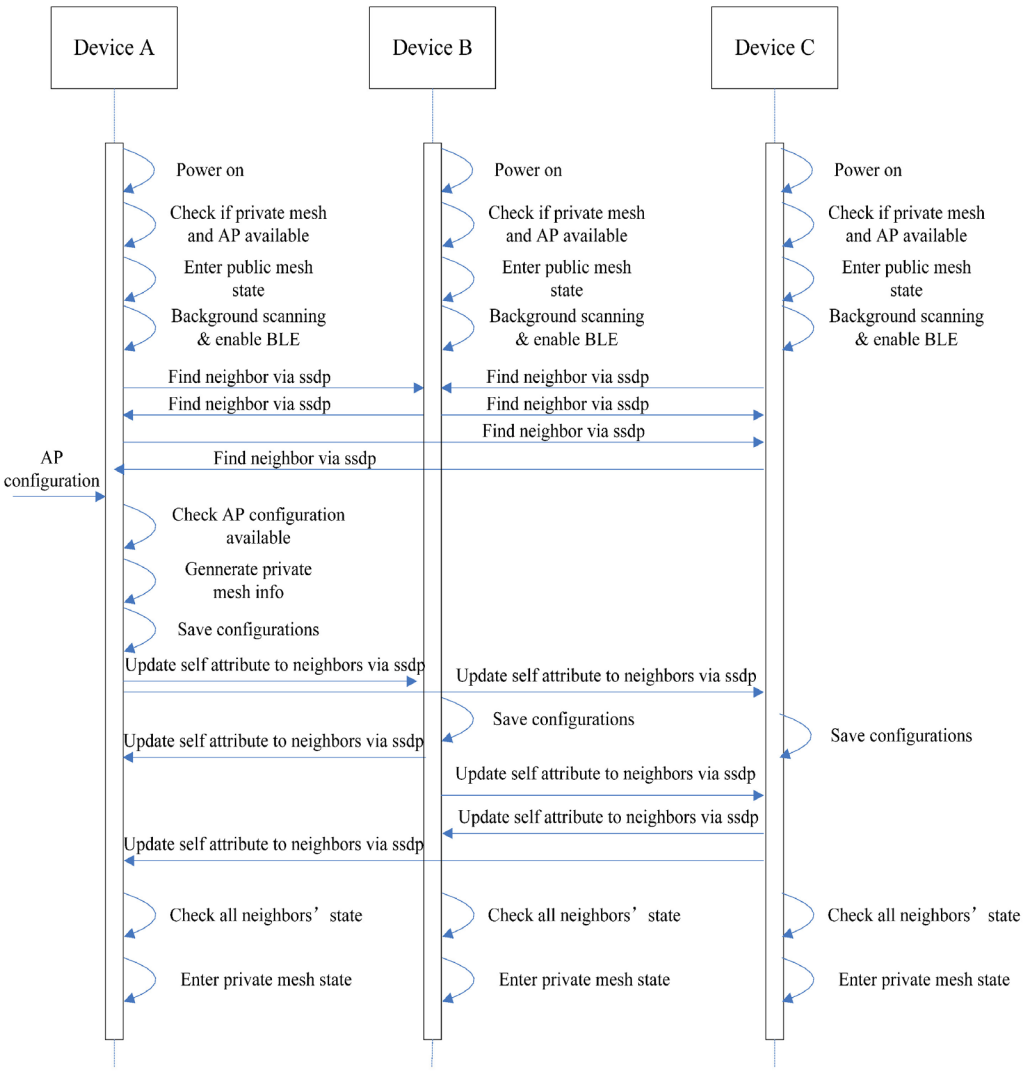

Figure 3. Flow of new nodes

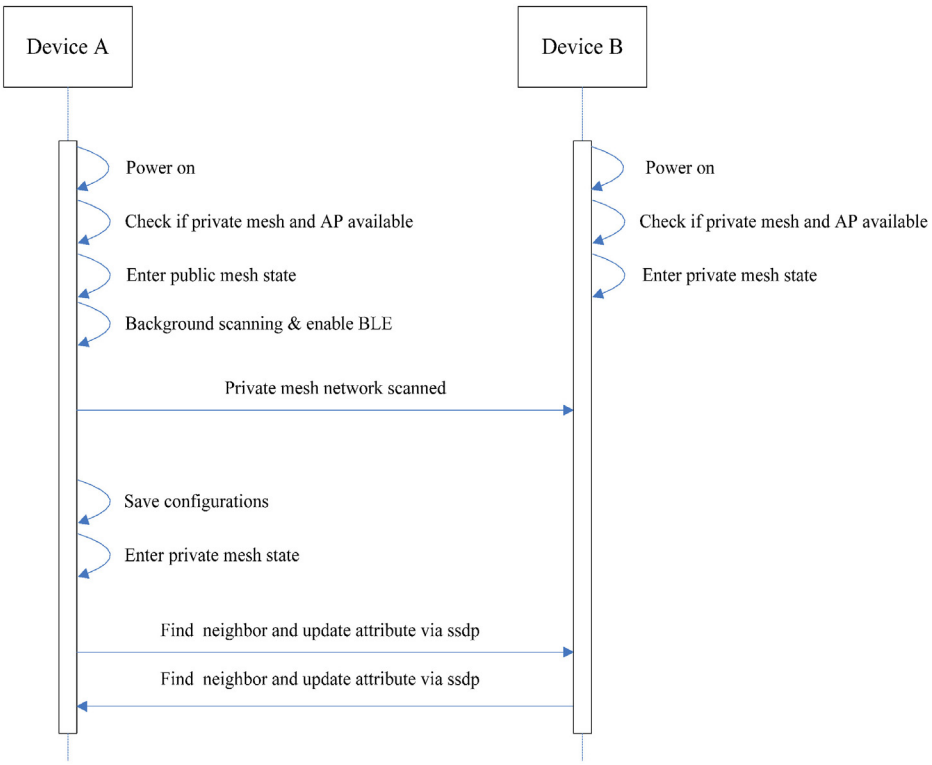

Figure 4. Flow of resetting privatized nodes

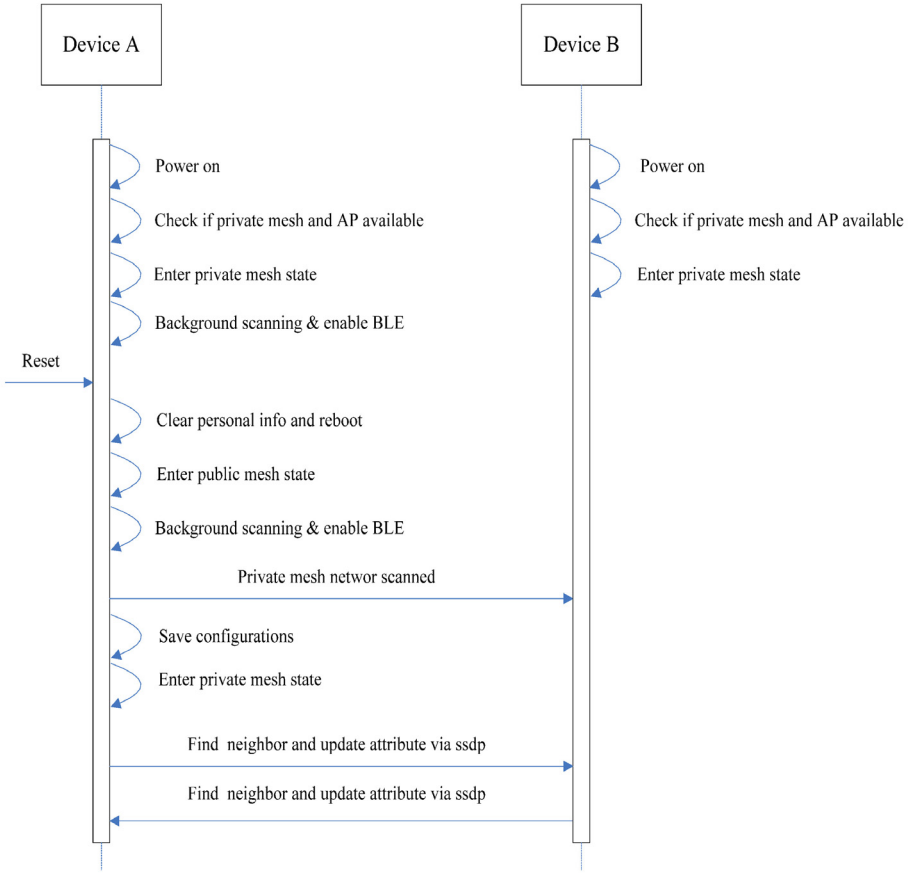

Figure 5. Flow of new nodes

If there is a privatized device to join another privatized group, what we need to do is press reset button to clear privatized information and reboot. Figure 5 is an example step when a privatized device joins in another privatized group:

3.5 Node Discovery

We use SSDP protocol to discovery node, the structure of SSDP is designed as Figure 6.

According to the protocol, when a control point (client) accesses the network, it can send "ssdp: discover" messages to a SSDP port with a specific multicast address using the M-SEARCH method. When the device listens for messages sent by the control point on the reserved multicast address, it analyzes the service requested by the control point. If it provides the service requested by the control point itself, the device will directly respond to the request by unicast, see Figure 7.

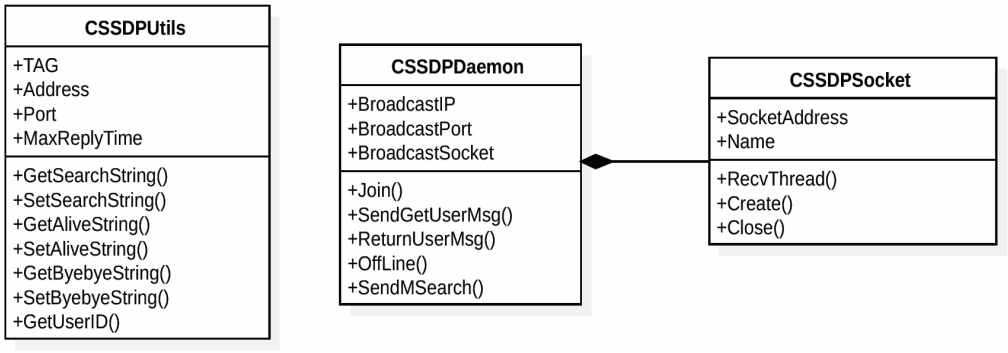

Figure 6. Internal structure of SSDP

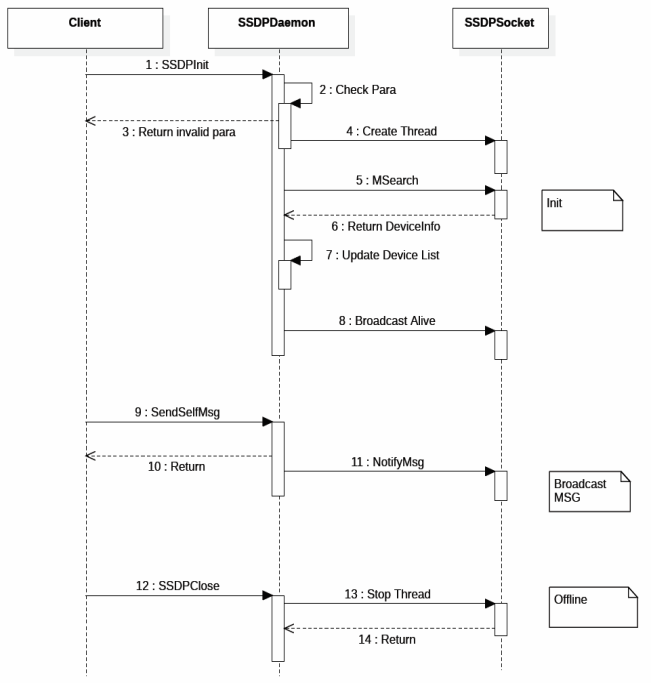

Figure 7. Setup Flow of SSDP

3.6 Master Vote Strategy

3.6.1 Initial Stage

When the vote is beginning, voter first send an election request and broadcast its own RSSI and blockchain height through SSDP protocol. The process is shown as Figure 8, every candidate will keep the waiting status untill other followers reture the result through TCP protocol.

3.6.2 Vote stage

This paper first define Parameters as:

Role: define the role of the node.

AP & RSSI: define the rssi of the AP found (if there are multi AP, there should be multi RSSI for different AP)

Master: every node in the group should save the master's UUID and IP.

Self: every node should save their own UUID and IP in the service.

Master vote strategy is presented as Figure 9.

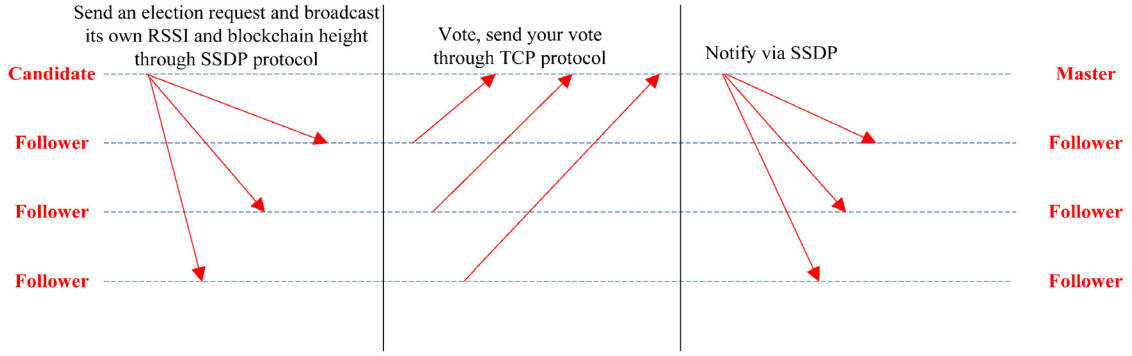

Figure 8. The process of Initial stage from master vote strategy

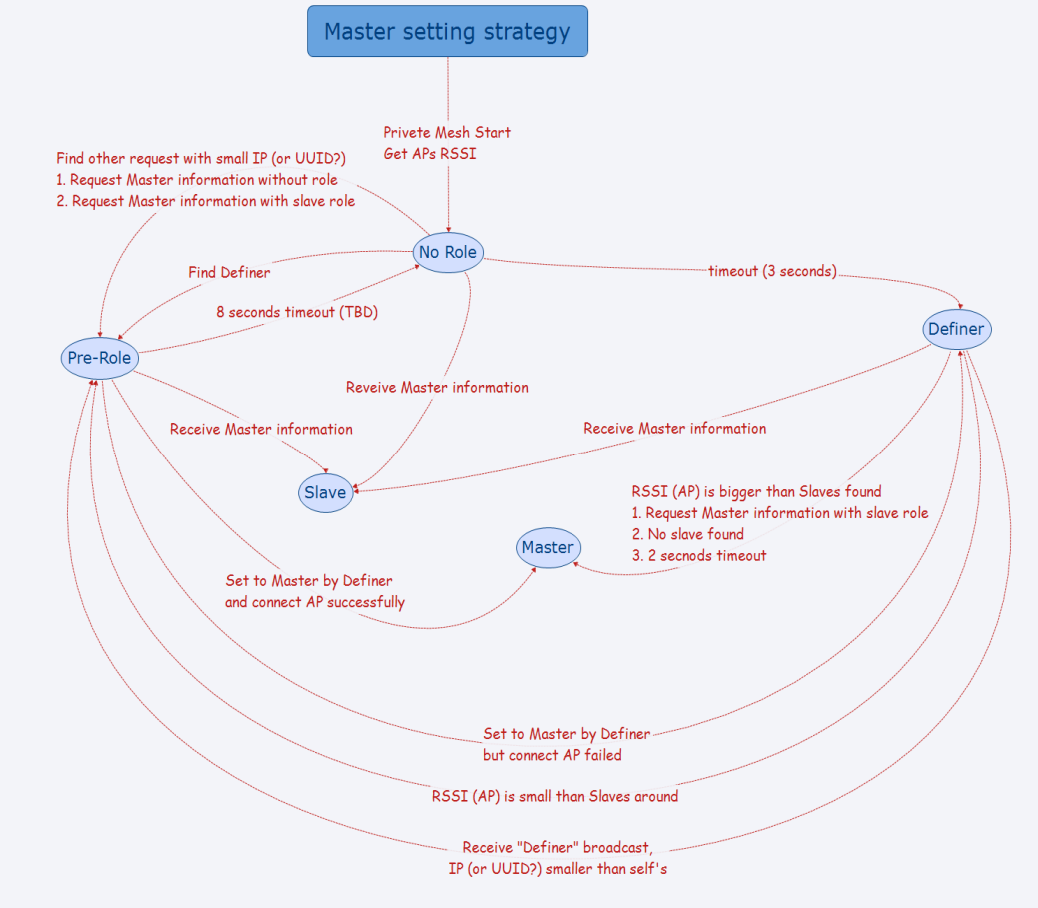

Figure 9. Master vote strategy

No Role state: Role is None; Send broadcast "request master information without role"; Receive "request master information without role" broadcast, compare with self IP. If self is smaller, then keep state. If self is bigger, then transfer to "Pre-Role" state. Receive "request master information with slave role" broadcast, compare with self IP. If self is smaller, then keep state. If self is bigger, then transfer to "Pre-Role" state. Receive "Definder" broadcast, change to "Pre-Role" state. Master or slave responses the broadcast to set the master information, then save master information and change to "Slave" state. After 8 seconds timeout, change to "Definer" state.

Pre-Role state: Role is "slave without master information" (could be slave role but without master parameters values.); Send broadcast "request master information with slave role"; Definer responses the broadcast to get the AP RSSI value; Definer responses the broadcast to set it to master role, then connect the AP and change to "Master" state. If connect AP failed, it will transfer to "Definer" state. Master or slave responses the broadcast to set the master information, then save master information and change to "Slave" state. If there is no change on the state after 8 seconds timeout, will transfer back to "No Role" state.

Definer state: Role is definer (or slave but send "definer" broadcast); Send broadcast "definer"; Receive "request master information with slave role" to get the AP RSSI value from the broadcaster; Receive "Definer" broadcast, check IP (or UUID), if small than self's, then change to "Pre-Role" state; After 2 seconds timeout, set the highest RSSI slave to the master and change to "Pre-Role" state (or change to "Slave" state), or change self to the "Master" state if self is the highest RSSI one or no slave found; Master or slave responses the broadcast to set the master information, then save master information and change to "Slave" state.

Slave state: Role is slave with master information saved; Receive "request master information without role/with slave role" and "definer" broadcast, set the master information to the broadcaster; Receive master's heartbeat packet.

Master state: Role is master with master information saved; Receive "request master information without role/with slave role" and "definer" broadcast, set the master information to the broadcaster; Send out heartbeat packet.

Special case: When master leave actively, master set new master and send to neighbors' the change; When master leave suddenly, slave has timeout and change to "No Role" state. Only neighbor slaves monitor the master's heartbeat packets. If master leaves suddenly, only neighbor slaves change to "No Role" state. Next jump slaves will be set to "Pre-Role" by neighbor slaves or next jump slaves.

3.6.3 Leadership Transfer

After the master gets the leader through election, it is responsible for managing the blockchain for normal operation, but the master may need to transfer the leadership to other nodes for some reasons to ensure the availability of the network. Therefore, in the election part of our algorithm, the master leadership transfer part is added. The process is designed as Figure 10.

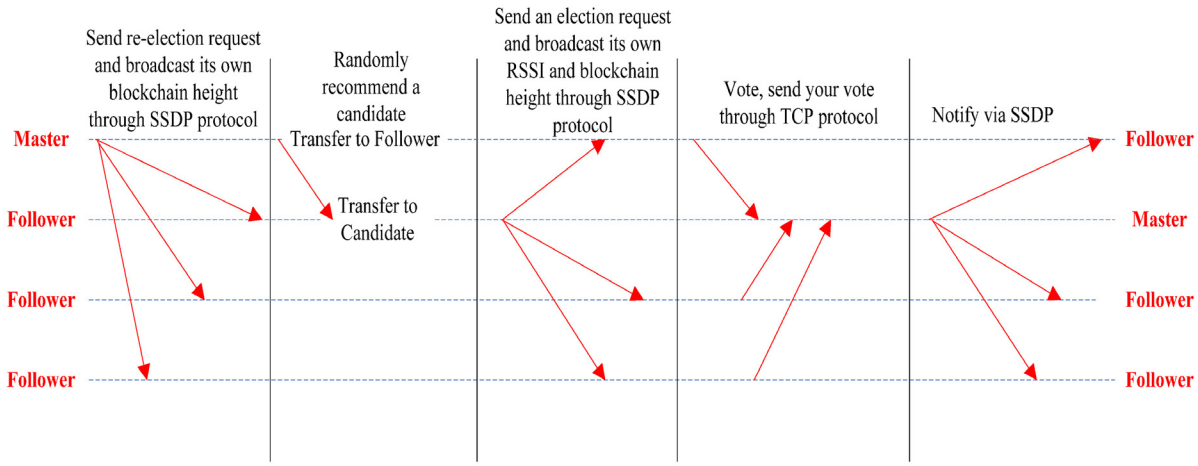

Figure 10. The process of leadership transfer

4 Experiments and Evaluation

4.1 Experiments

Our implementation was deployed on Fabric's official test network that can mimic the production network. We implemented the proposed method on the mesh AP using hardware as shown in Figure 11. The mesh network is formed by multiple non-homogeneous nodes, where each node is built by a different manufacturer, but all of them have access to the same network. The tested hardware includes, a surface pro3 (rtl8192ce) - Node 1, a Samsung Galaxy S7 (mvl8787) - Node 2, a Raspberry pi (wcn36xx) - Node 3, a Samsung4412 dev board (ath9k) - Node 4 and a laptop computer equipped with Intel Core i5-460M processor, 2GB memory and Intel Centrino Advanced-N wireless chip - Node 5.

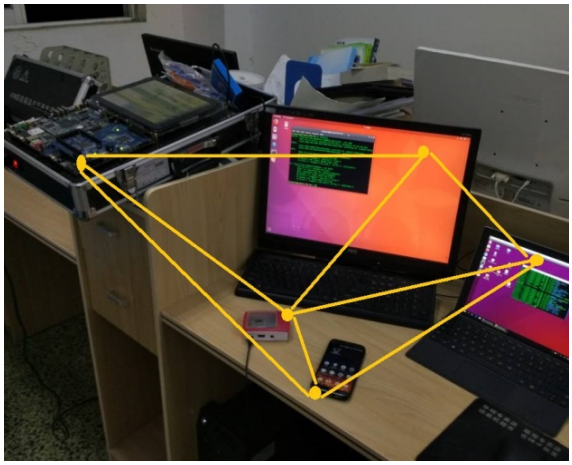

Figure 11. WLAN mesh network

We sent various transactions to simulate block chain simultaneously in the protocol. Figure 12 demonstrates the results of our experiment and highlights the breakdown of the consensus time consumption. Except for opening registration, the time cost for each task increases linearly with the number of transactions. This means that the contract can be modified to perform the processing in batches and allow multiple transactions to complete the task.

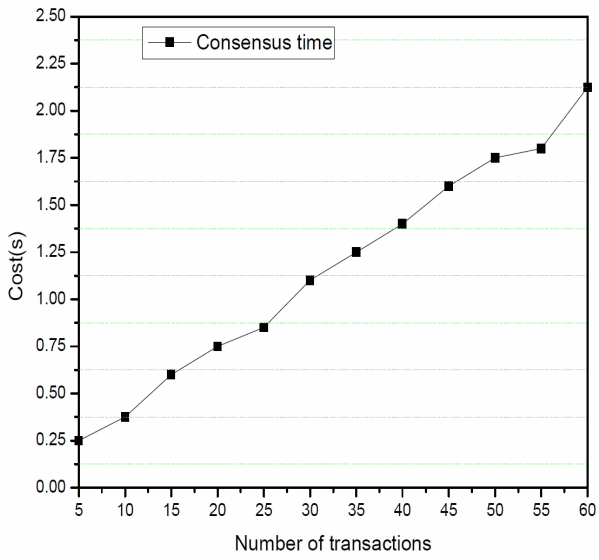

Figure 12. The time cost for the consensus based on the number of transactions sending simultaneously

4.2 Evaluation

To evaluate the effectiveness of our algorithm, we compare the Raft protocol with our algorithm in three aspects: i. Within timeout of 600ms and 900ms, count the probability of re-election of master when the

master node in the network fails and cannot notify the slave nodes. ii. When two device failures occur in the network (non-master nodes, and the number of failures is less than half of the number of nodes), compare the time it takes for the network to restore external services. iii. When nodes move, network sparseness or signal attenuation occur in the network, and other reasons often lead to network interruption and network segmentation. At this time, two or more masters appear. When the network is restored, compare the time it takes to for the network return to normal state.

Table 1 and Table 2 outline the probability measurements for selecting master node of different algorithms, according to the timeout value of slave state nodes. The experimental results show that our algorithm significantly shortens the time of selecting master node in the network, and then improves the probability.

Table 1. Success rates of selecting Master node in 600ms of Timeout of Raft and our algorithm

Test Counts	*Raft* (%)	*Our* algorithm (%)
12 tests	0	59.73
25 tests	17.62	44.79
50 tests	82.64	95.31
100 tests	97.41	99.13
200 tests	85.19	93.13

Table 2. Success rates of selecting Master node in 900ms of Timeout of Raft and our algorithm

Test Counts	*Raft* (%)	*Our* algorithm (%)
12 tests	5.10	86.43
25 tests	23.21	89.11
50 tests	85.43	98.93
100 tests	98.49	99.99
200 tests	99.99	99.99

We set up five groups of experiments. In the experiment, the same two devices in the network are powered off at each time, and then the time for the network to resume external service is tested. As can be seen from Figure 13, the recovery time of our algorithm is significantly less than that of Raft algorithm.

In the practical consensus network, many networks will not be in the same area, but still provide the same service to the outside. Therefore, we simulate the network partition failure in the network, and set up five groups of experiments, each time we let the same two devices appear partition, and then restore the network connection, so as to measure the time of restoring the external service, shown in Figure 14.

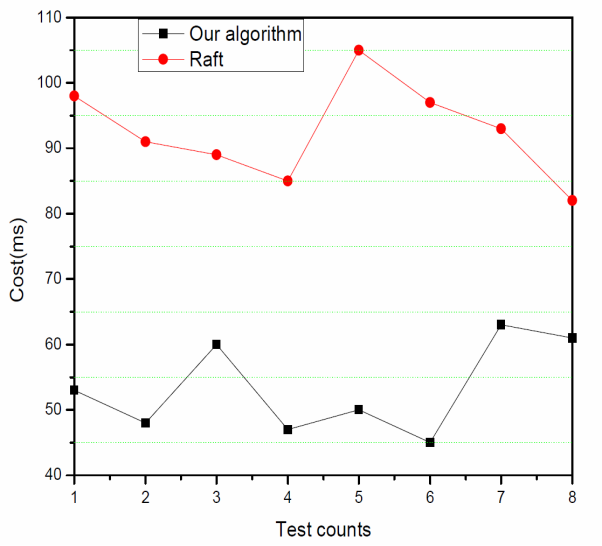

Figure 13. Time required for the network to restore the service in the case of minority of nodes failure

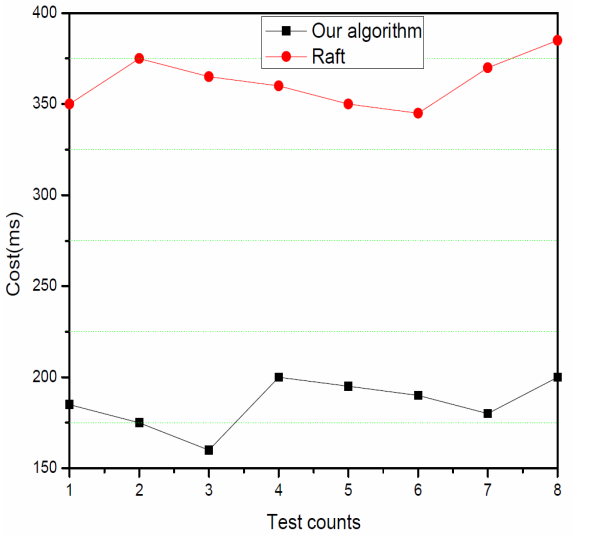

Figure 14. Time required for the network to restore the service in the case of network partition failure

5 Conclusion

In this paper, we developed a fast adaptive blockchain consensus algorithm implementation by using maximum voter privacy that runs on Fabric. Our implementation was tested on the official Fabric test network with forty simulated voters. We have shown that our method can be readily used with minimal setup for elections at a little cost per voter. The cost can be considered reasonable as this voting protocol provides maximum voter privacy and is publicly verifiable. This is the first implementation of a decentralised internet consensus protocol running on wlan mesh network. It uses the Fabric blockchain not just as a public bulletin board, but more importantly, as a platform for consensus computing that enforces the correct execution of the voting protocol.

In future work, we will investigate the feasibility of running a national-scale election over the blockchain. Based on the knowledge gained from this paper, we believe that if such a perspective is ever considered possible, its implementation will certainly require a dedicated blockchain. For example, this can be an Fabric-like blockchain that only stores the e-voting smart contract. The new blockchain can have a larger block size to store more transactions on-chain and may be maintained in a centralised manner similar to RSCoin [9].

Acknowledgements

This work is supported by National Natural Science Foundation of China under grant Nos. U19A2086, 61802033, 61662016.

References

[1] J. A. Garay, A. Kiayias, N. Leonardos, The Bitcoin Backbone Protocol: Analysis and Applications, *Theory and Application of Cryptographic Techniques*, Sofia, Bulgaria, 2015, pp. 281-310.

[2] M. Samaniego, R. Deters, Blockchain as a Service for IoT, *Green Computing and Communications*, Chengdu, China, 2017, pp. 433-436.

[3] J. Clark, A. Essex, CommitCoin: Carbon Dating Commitments with Bitcoin, *Financial Cryptography and Data Security*, Kralendijk, Bonaire, 2012, pp. 390-398.

[4] M. Mettler, Blockchain Technology in Healthcare: The Revolution Starts Here, *International Conference on e-health Networking, Applications and Services*, Munich, Germany, 2016, pp. 14-16.

[5] O. Lafe, Data Compression and Encryption Using Cellular Automata Transforms, *Engineering Applications of Artificial Intelligence*, Vol. 10, No. 6, pp. 581-591, December, 1997.

[6] B. Adida, Helios: Web-based Open-audit Voting, *USENIX Security Symposium*, San Jose, CA, USA, 2008, pp. 335-348.

[7] G. T. Nguyen, K. Kim, A Survey about Consensus Algorithms Used in Blockchain, *Journal of Information Processing Systems*, Vol. 14, No. 1, pp. 101-128, January, 2018.

[8] V. Gramoli, From Blockchain Consensus Back to Byzantine Consensus, *Future Generation Computer Systems*, Sydney, Australia, 2017, pp. 1-20.

[9] C. Tang, Z. Yang, Z. L. Zheng, Z. Chen, X. Li, Game Dilemma Analysis and Optimization of PoW Consensus Algorithm, *Acta Automatica Sinica*, Vol. 43, No. 9, pp. 1520-1531, September, 2017.

[10] K. Leng, Y. Bi, L. Jing, H. C. Fu, I. V. Nieuwenhuyse, Research on Agricultural Supply Chain System with double Chain Architecture Based on Blockchain Technology, *Future Generation Computer Systems*, Vol. 86, No. 9, pp. 641-649, September, 2018.

[11] T. Wu, K. Huang, X. L. Zhou, N. Kong, Research on Blockchain Consistency Algorithm with State Legality Verication, *Computer Engineering*, Vol. 44, No. 1, pp. 160-164, January, 2018.

[12] P. Jiang, F. Guo, K. Liang, J. Lai, Q. Wen, Searchain: Blockchain-based Private Keyword Search in Decentralized Storage, *Future Generation Computer Systems*, pp. 1-16, September, 2017.

[13] H. Kim, Y. Jeong, Secure Authentication Management Human Centric Scheme for Trusting Personal Resource Information on Mobile Cloud Computing with Blockchain, *Human-centric Computing and Information Sciences*, Vol. 8, No. 1, pp. 11-22, December, 2018.

[14] S. Zhang, J. Cai, Z. Chen, H. He, Byzantine Consensus Algorithm Based on Gossip Protocol, *Computer Science*, Vol. 45, No. 2, pp. 20-24, February, 2018.

[15] C. Yuan, M. Xu, X. Si, Optimization Scheme of Consensus Algorithm Based on Aggregation Signature, *Computer Science*, Vol. 45, No. 2, pp. 53-56, February, 2018.

[16] M. Vukolic, The Quest for Scalable Blockchain Fabric: Proof-of-Work vs. BFT Replication, *International Workshop on Open Problems in Network Security*, Zurich, Switzerland, 2015, pp. 112-115.

[17] L. Bach, B. Mihaljevi, M. Agar, Comparative Analysis of Blockchain Consensus Algorithms, *41st International Convention for Information and Communication Technology, Electronics and Microelectronics*, Opatija, Hrvatska, 2018, pp. 1545-1550.

[18] L. Feng, H. Zhang, W. T. Tsai, S. Sun, System Architecture for High-performance Permissioned Blockchains, *Frontiers of Computer Science*, Vol. 13, No. 6, pp. 1151-1165, December, 2019.

[19] D. Schwartz, N. Youngs, A. Britto, *The Ripple Protocol Consensus Algorithm*, Ripple Labs Inc., 2018.

[20] F. Hao, P. Y. Ryan, and P. Zielinski, Anonymous Voting by Two-round Public Discussion, *IET Information Security*, Vol. 4, No. 2, pp. 62-67, June, 2010.

[21] C. P. Schnorr, Efficient Signature Generation by Smart Cards, *Journal of cryptology*, Vol. 4, No. 3, pp. 161-174, January, 1991.

[22] A. Fiat, A. Shamir, How to Prove Yourself: Practical Solutions to Identication and Signature Problems, *Crypto86*, Vol. 263, No. 5, pp. 186-194, March, 1987.

Biographies

Mingzhe Liu received his MSc and Ph.D. in Computer Science from Massey University, New Zealand. He is a Professor of School of Network Security, Chengdu University of Technology, China. His research interests include intelligent information processing, information security.

Xin Jiang received his B.Sc in Information and Computing Science from Chengdu University of Technology, China, in 2012. He is currently studying for a doctoral degree in Chengdu University of Technology, China. His research interests include medical imaging, deep learning, and cyberspace security.

Feixiang Zhao received his B.Sc in Measurement, Control Technology and Instrumentation from Chengdu University of Technology in 2018. He is studying for his master's degree in Chengdu University of Technology. His research interests include digital image processing and machine learning.

Xuyang Feng is studying for his bachelor degree in Chengdu University of Technology. His research interests includes cyberspace security.

Ruili Wang is a Professor of Artificial Intelligence at Massey University, Auckland, New Zealand. He received the PhD degree in Computer Science from Dublin City University (Dublin, Ireland), the Bachelor Degree from Huazhong University of Science and Technology (Wuhan, China), Masters' Degree from Northeastern University (Shenyang, China). His current research areas include language and speech processing, machine learning and data mining, computer vison and image processing.