Connected vehicular network using Hashgraph

Divyansh Jaina, Dara Nanda Gopala Krishnaa, Pranav Goyala, Vikas Hassija1,∗

aDepartment of CSE and IT, JIIT, Noida, India

Abstract

Internet of things(IoT), Internet of vehicles(IoV) are the buzz words today. buliding a smart vehicle running on a smart road that could respond to the very crucial requests such as, reporting an accident ahead or basic requests such as downloading a song at the blink of an eye is the hottest topic researchers and computer scientists are working on. The latest testing and later the rolling out of 5G technology will prove to be a boon in providing the required speed which will further ensure safety as well as Quality-of-service(QoS) concerns. Hashgraphs, a superior distributed ledger technology are used to create communication network between various vehicles and other essential parameters. The decentralized system will check any delay in responses through the inherent consensus process which is the USP of hashgraph. Scheduling the requests according to the priorities so as to provide a better Quality-of-Service quotient is done using hashgraph only. We have also compared why hashgraph surpasses other counter-parts such as bitcoin or ethereum. The proposed model is simulated using Omnetpp simulation by making proper design and network description files. Messages can be seen getting transferred between the vehicles and hashgraph is implemented for prioritizing the messages.

Keywords: Internet-of-vehicles, hashgraph, distributed applications, consensus, blockchain, VANET

1. Introduction

The recent rolling out of fifth-generation communication technology (5G) will be a gi- galeap in the world of technology. This next generation cellular technology will provide reliable communication with ultra low latency. 5G will hopefully form the backbone for budding technologies such as Internet-of-Vehicles (IoV), Internet-of-things (IoT) and ma- chine to machine communications. IoV can act as an imperative option for communication between vehicles for ensuring immaculate order on roads. It can respond to versatile re- quests such as warning messages like do not cross ceratin speed limit or emergency message like report nearest accident and potentially help in achieving smart roads with information

∗Corresponding author

Email addresses: [suramya.sharma2@gmail.com (Suramya Sharma),](mailto:sharma2@gmail.com) [avidhi07@gmail.com (Vidhi](mailto:avidhi07@gmail.com)

Agarwal), [dhruv21097@gmail.com (Dhruv Gupta),](mailto:dhruv21097@gmail.com) [vikas.hassija@jiit.ac.in (Vikas Hassija)](mailto:hassija@jiit.ac.in)

Preprint submitted to Future Generation Computing Systems October 4, 2019

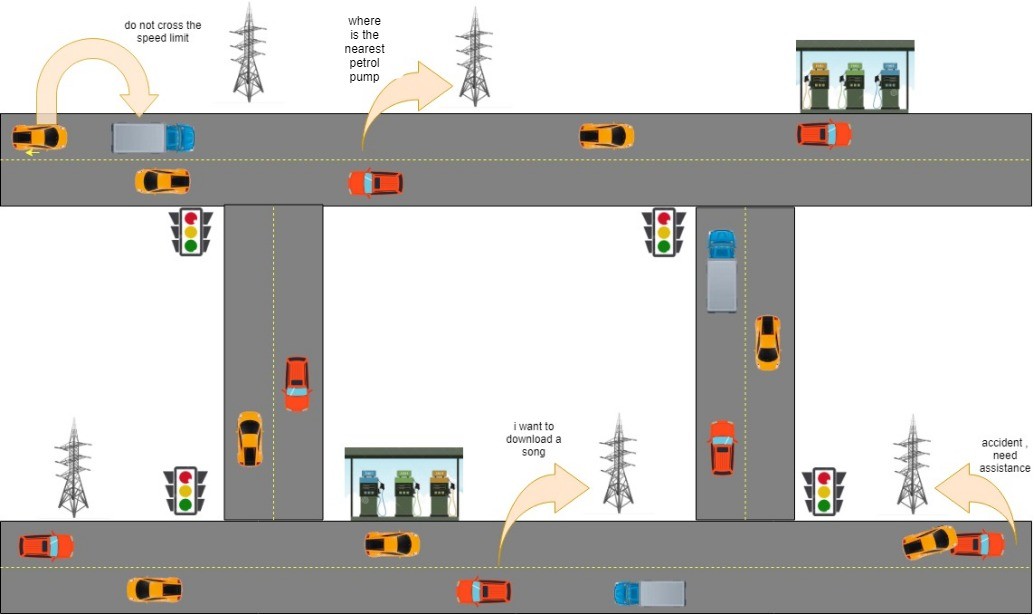


Figure 1: Network of vehicles communicating among each other and Road Side Units (RSU).

of number of vehicles entering and exiting. Using Hashgraph will improve Quality-of-service (QoS) by making it a decentralized network where vehicles can communicate with each other with or without involvement of third party and aided by 5G technology for high speed data exchanges between them. Therefore IoV proves to be a thing of utility in daily life or a perfect example of realization of emerging technologies for practical needs ensuring sophisticated traffic management, proactive vehicles and dynamic and diligent processing of requests. Further opening possibilities for working with driver-less cars too.

1.1. Limitations of Blockchain

Previous works in smart vehicular network has been accomplished using blockchain. But blockchain has its own limitations.

1. Large amount of energy and computing power is spent by miners to validate and add the transactions into the blockchain.

2. The blockchain ledgers are immutable which means data cannot be deleted, modified or transferred. The storage of blockchain is very less. Only a limited number of transactions can be added to blockchain.

3. Due to storage issues, users pay a transaction fee to get their transactions included in the block by the miners. Larger transaction will cost larger fee as it takes up more space in the block.

4. Another limiting factor of blockchain is its low throughput and high latency. This is mainly due to slow transaction speed and minuscule amount of storage space.

5. There is no appropriate ordering of transactions in blockchain.

6. Blockchain’s consensus algorithm offers some limitations too. The consensus mecha- nism followed by blockchain is divided into two categories based on its type i.e Private or Public Blockchain.

7. Private Blockchain is dependent on leader-based consensus mechanism which limits the usage only to trusted partners. But the loosened up security benchmarks make these system potential targets for DDoS attacks.

8. Public Blockchain depends on consensus mechanism such as ’Proof Of Stake’ and

’Proof Of Work’. For these every node must concur with the order of the transactions wherein they have taken place, which limits down the number of applications where these technologies can be utilized.

9. When two unique miners mine a block at the same time, the chain can part into two distinct forks. In this situation, the two forks will keep on approving blocks and include new blocks. When one of the chains approves a block before the other chain finishes it turns into the longest chain. The longest chain is the one with the longest most legitimate blocks joined and becomes the accepted chain. The transactions mined in the shorter chains are rejected.

10. Finality of blockchain is probabilistic. Rejection of shorter chains due to creation of forks doesn’t guarantee that the transaction is irreversible.

11. Creation of forks can cause selfish mining. This means miners can create a separate fork and hide newly-generated blocks from main blockchain and earn more bitcoins.

To overcome these limitations there was an utmost need of a new promising approach. Hashgraph is appropriate to turn into the world’s first mass-embraced public distributed ledger, supporting a huge range of utilization’s.

1.2. Advantages of hashgraph

Just like Blockchain, Hashgraph is also based on Distributed Ledger technology. Perfor- mance, reliability, security are the pre-eminent variables that make a Hashgraph a remark- able innovation fit for contending with blockchain.

1. Performance - The consensus algorithm of Hashgraph can process thousands of trans- actions per second. Consensus latency is very low while it provides very high through- put. Hashgraph technology is quick as far as utilizing the gossip protocol which spreads messages between network users. These messages are selectively optimized to diminish the communication overhead.

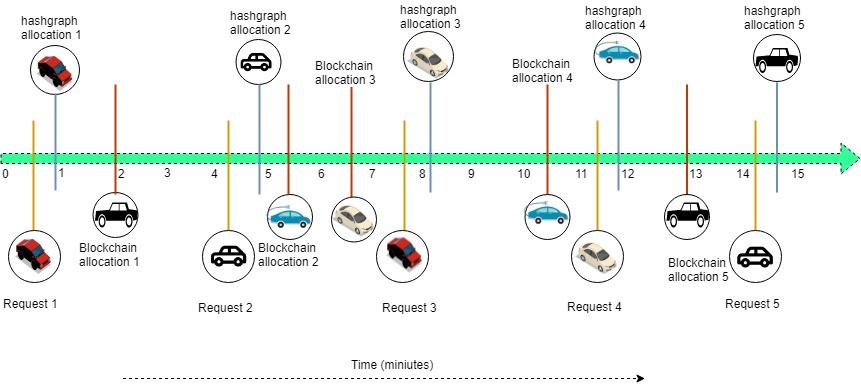


Figure 2: Hashgraph Vs Blockchain

2. Fairness - Hashgraph provides fairness through consensus timestamp. This implies in the event that one transaction achieves two-thirds of the network ahead of the other transactions, is considered to be the first. If there exists a danger of malicious conduct, nodes are approved to stop the transaction and report the illicit or malignant conduct, in this way keeping it from achieving consensus. Hashgraph ensures both Fair Access and Fair Ordering.

3. Security - A few aspects furnish Hashgraph users with improved security and safety.

Among them are:

(a) Cryptography - Cryptographic hashing of all communications and interactions is done and transactions are signed digitally. The algorithms used, cling to security guidelines to ensure personal and private information.

(b) BFT - Hashgraph is asynchronous Byzantine Fault Tolerant. This is a special term implying that no single user can keep the network from achieving a consen- sus. A faulty node may prevent other nodes from reaching consensus but BFT counteracts this, guaranteeing flexibility against a wide range of attacks.

(c) Acid Compliance - All together for a transaction to be finished, all nodes must agree to atomicity, consistency, isolation, durability.

(d) Distributed Denial of Service Attack resilience - One type of Denial of Service (DoS) attack happens when an attacker can flood a legit node on a system with unimportant messages, keeping that node from performing other (legitimate) obli- gations and jobs. A Distributed Denial of Service (DDoS) utilizes open admin- istrations or devices to accidentally intensify that DoS attack—making them a significantly more prominent threat. In a DLT arrange, a DDoS attack could focus on the nodes that add to the meaning of consensus and, possibly, keep that

consensus from being established. The hashgraph is DDoS resilient as it engages no single node or modest number of nodes with uncommon rights or duties in setting up consensus.

4. Governance - The governance model defines a set of rules to conform to the technolog- ical software policy, coin issuance, and the incentive model. Open consensus provides trust between the nodes while carrying out transaction processing. Additionally, all members can be chosen and elected within the governance model, with the opportunity to vote and settle upon policy rules and codebase changes.

5. Stability - Stability is another hashgraph advantage. Technical and legitimate controls have been created to ensure stable working conditions that are not compromised by the likelihood of forking. In this manner, technical controls provides signed state proofs, ledger ID, and also taking care of forks.

2. Working of hashgraph

Since a hashgraph exploits directed acyclic graphs for recording the operations. Every node in the network can be said to have a graph of its own and this graph is with every other node. The same thing is done using chain in blockchain technology. So how does this takes place? This takes place using a comprehensive process. This process leads to various concepts such as gossip protocol, virtual voting, etc. Keeping an honest consensus in the network ensures resistance from attacks.

1. Gossip Protocol - Gossip protocol consists of the sharing of information rapidly between the nodes. One node to other to another and keeps on happening and the event spreads in the network. Every event must be reached to all nodes. It is the responsibility of the network to spread the information (events) and not sender’s responsibility. This frequent sharing of information gradually builds the hashgraph and the graph keeps on increasing. It is noteworthy that the memory does not actually store such a graph, it is more of an idea. however, events are recorded in the memory. Security of the hashgraph data structure so built is cryptographically reliable. The network so formed accommodates the communication history of the process.Further, it is not just enough to ensure that all the events know about each other. It is equally necessary that events agree on a certain linear order of transactions that are recorded inside the events. It seems like once gossiping takes place every event will be aware about every other event and hence the consensus is reached but this is not true. There may be moments when some recent events are yet to be communicated, solving this issue leads us to Virtual Voting.

2. Virtual Voting - Some of the key terms to understand virtual voting are- transactions, consensus, nodes. Virtual voting is quite like actual voting, it is reasonably democratic in technical terms.It calculates vote of a witness based on the existing knowledge of that witness/event without actually getting them to vote. In technologies like bitcoin,

a block gets to make the consensus decisions which is fast enough to solve the hash first but in virtual voting, such a node is selected by an elaborate method, consisting of three steps-

(a) Divide Round

(b) Decide Frame

(c) Order Transactions

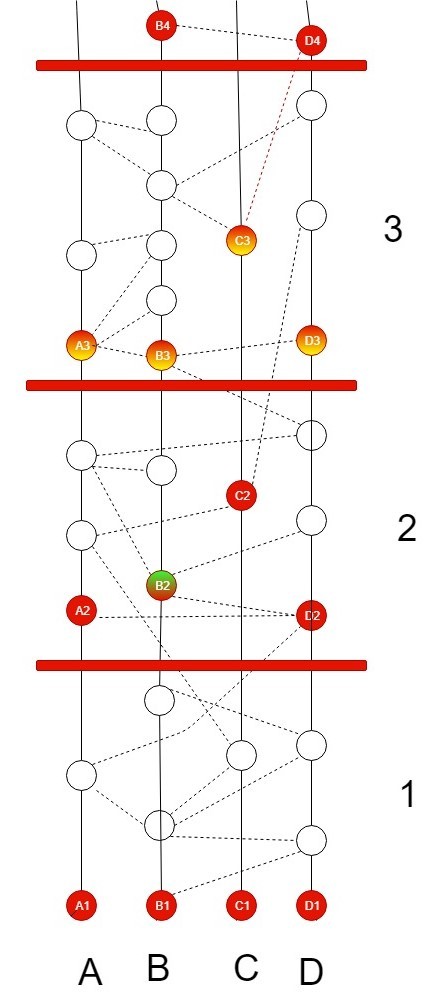


Figure 3: Round creation in hashgraph

3. Divide Round - Rounds are created on the basis of strongly seeing events. If an even can see 2/3rd events of round p then it is assigned round p+1. In a round, all the first events are called witnesses and only the witness events are entitled to send or receive virtual votes. In the given figure, A1, B1, C1, D1, A2, B2, C2, D2, A3, B3, etc. all are

witnesses as they are first events of rounds 1, 2 , 3 respectively. This process occurs in Decide Frame.

4. Decide Frame - In this round, it is decided that whether an event(witness) is famous or not. A witness (shown in red) can be called famous when many witnesses of next round can see it. For example, B2 is seen by A3 through B3, it is also seen by C3 via D3 and B3, therefore it is seen by four witnesses of next round namely- A3, B3, C3, D3 ( witnesses of Round-3). This makes B2 famous. This exercise takes place for all witness events.When two-thirds of a population (which refers to stake, described later) sees an event then is is said to be seen by a super-majority, implying that it is strongly seen. Vote counting is performed by witness events. A3 can be strongly seen from B4 via paths through A, B and D, which is a super-majority. B4 can also strongly see B3 and also C3 through A, B and D. It can also strongly see D3, via A, B, C and D. This proves that B2 is undoubtedly famous after vote-counting.

5. Order Transactions - After the consensus has been reached to determine the fame of certain events, another consensus regarding the order of transactions by sorting out timestamps and consensus total order on older events can be achieved. To understand this step, the concept of ’received round’ must be understood. An event ’e’ has a received round ’p’ if p is the first round where all the unique famous witnesses were descendants of e. For example- e has a received round p, a new unique famous witness x is created by A in round p. however, x has a self-ancestor y which keeps knowledge of e beforehand. Also when it was first created it was give a timestamp t. t is therefore the time when A first learned of e. Therefore, received time of e can be calculated by taking median of all such timestamps, created by multiple members in round p. This way consensus order is received. All events are sorted by their received rounds. If two events have the same rounds then they can be sorted by received times, yet if there is a tie, they are sorted by their signatures. Events are split by sorting with respect to signatures after the XORing is performed on these signatures with other unique famous witness’ signatures in the received round.

3. Final ordering of the requested messages

The ascending order of events is generated based on hashgraph consensus timestamp as explained in the preceding section. However, there can be a required priority set of messages and also a priority set based on deadline of messages. The timestamp ordering might not always be same as the ordering based on priority of the message requests the deadline provided by requesting vehicles. The final order of the events is expected to include three orders - hashgraph timestamp, priority order and deadline time of requested messages or flags. Therefore, it is required that a final order be obtained after taking into consideration all three sets of event-orders. Let us assume that there are n number of events present in network. A particular weight can be set for prioritizing the order through hashgraph consensus algorithm, expected priority order and least deadline time. Let P1, P2 and P3 be

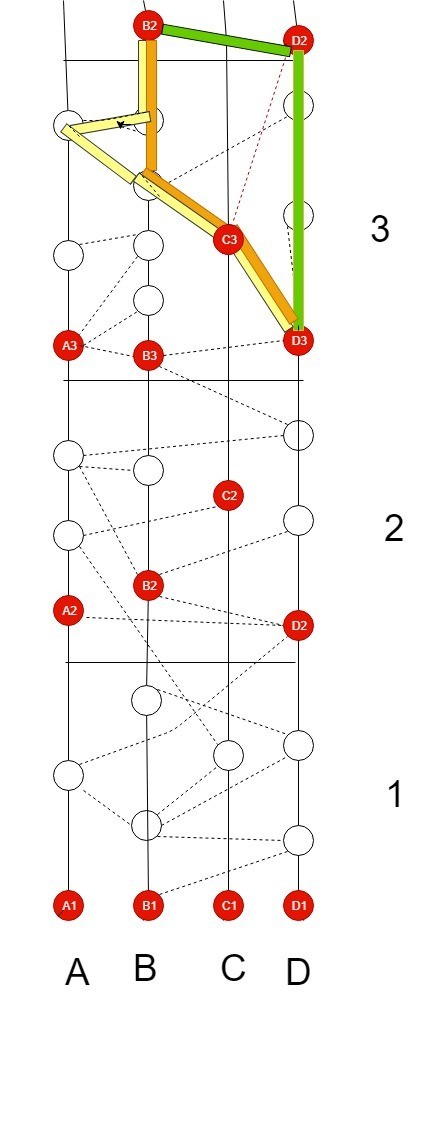


Figure 4: Round creation in hashgraph

the vectors of transaction-orders generated by vehicle-nodes through hashgraph consensus algorithm, expected priority order and least deadline time respectively. Let W1, W2 and W3 be the weight associated with hashgraph consensus order, pre-defined priority-order and least deadline time order, such that W1 + W2 + W3 = 1 The values in vector P1, P2 and P3 are multiplied with the respective weights W1, W2 and W3 and the results are stored in vectors R1, R2 and R3 respectively.

n n

R1(i)i=1 = W1(P1(i)i=1) (1)

n n

R2(j)j=1 = W2(P2(j)j=1) (2)

n n

R3(k)k=1 = W3(P3(j)k=1) (3)

The events corresponding to the three vectors R1, R2 and R3 are same. The values in vector R1 are compared to values in vector R2. If the first value in vector R1 is smaller than all values in vector R2 then that particular event in both the vectors R1 and R2 are eliminated and the event is stored in another vector X . Otherwise, if any value in vector R2 is found smaller than the first value in vector R1, then that particular event is eliminated and the event is stored in another vector X . The process is repeated until all the events in vector R1 and R2 are eliminated and vector X gives the final ordering of the offloading tasks considering both the hashgraph consensus timestamp and pre-defined priority order. The new weight for vector X being the average of W1 and W2, denoted as Wx. Now, the new vector X is multiplied by weight Wx and we obtain vector N .

n n

where,

N (i)i=1 = Wx(X (i)i=1) (4)

W 1 + W 2

W x =

2

Now the same process of elimination is carried out on vectors- N and R3. That is, vector N and vector R3, which was obtained by multiplying deadline order with weight W3 finally give the required order vector- F.

4. Improving Hashgraph Timestamp:

The ascending order of events is generated based on hashgraph consensus timestamp. However, the timestamp ordering might not always be same as the ordering based on deadline of the offloading. The final order of the events is expected to include both hashgraph timestamp and deadline time of data items. Let us assume that there are n number of events present in network. We give more priority to order through hashgraph consensus algorithm compared to order of least deadline time with data items present in mobile devices. Let z1 be be vector of events generated by mobile devices through hashgraph consensus algorithm arranged in ascending order of consensus timestamp. Now w\_1 is the weight multiplied with every event's consensus timestamp in the vector z1 and the results are stored in another vector y1.

z2 is vector having least deadline time of data items with events in the vector z1. Where j lies in the range between 1 to n. w\_{2} is the weight multiplied with every event's least deadline time in the vector z2 and results are stored in another vector y2.

As we see the events corresponding to both vectors y1, y2 are same. Now the values in vector y1 are compared to values in vector y2. The event associated with the less value of vector y1 than any other event's value of vector y2 is eliminated from both vectors y1, y2 and stored in another vector $x$ otherwise event associated with less value in vector y2 is eliminated from both vectors y1, y2 and stored in x\_{k} of vector x. Where k lies in the range between 1 to n.

Finally the vector x\_{k} has n events ordered in efficient manner while importance to order based on consensus timestamp along with least deadline time of data items in mobile devices.

5. Simulation

The idea of the proposed model is presented by a simulation on OmNet++. OmNet++ provides network simulation facility which is modular, extensible, component-based C++ library. Although, here a basic simulation is used to depict gossip protocol of hashgraph between the vehicles. The messages transferred are of four types namely- security message, help message, event message and multimedia message. It has an excellent graphic user interface. The components of simulation such as initialization, handling and forwarding of messages are written in C++. These components are assembled using network definition files, written in NED language where design part of simulation is focused. Some advantages of using OMNet++ over other simulators is that is is well structured, public availability of source code and it is not just limited to network protocol simulation. Its hierarchical nature helps in large-scale simulations as it offers re-usability of components. It helps in analysing results as well. The vehicles which are to be connected are represented as nodes in the simulator. Multiple nodes are connected to each other with lines depicting the connection as shown in the figure. Message requests made by nodes travel in the network in random order in case of blockchain. However, in case of event order obtained by hashgraph, messages travel in the order decided my timestamps produced by hashgraph. In that case, the vehicles depicted as nodes in the simulation are actually events of hashgraph and the messages which travel in the network are transactions taking place in hashgraph. Also, the code can be fixed such as to share messages according to various event orders such as- predefined priority order or deadline-based order.

