UNIVERSITATEA POLITEHNICA BUCUREȘTI
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

PROIECT DE DIPLOMĂ

Blockchain în
Sisteme Inteligente de Transport

Dragoș Cocîrlea

Coordonator științific:
Prof. dr. ing. Ciprian Dobre

BUCUREŞTI

2020

UNIVERSITY POLITEHNICA OF BUCHAREST
FACULTY OF AUTOMATIC CONTROL AND COMPUTERS
COMPUTER SCIENCE DEPARTMENT

DIPLOMA PROJECT

Blockchain in
Intelligent Transportation Systems

Dragoș Cocîrlea

Thesis advisor:
Prof. dr. ing. Ciprian Dobre

BUCHAREST

2020

Table of contents

SINOPSIS

Blockchain este o nouă tehnologie în curs de dezvoltare al cărei impact a fost simțit cel mai tare în sectorului financiar si care are potențialul de a avea o influență la fel de mare ca cea a internetul. Un blockchain reprezintă o rețea de noduri interconectate, atât de încredere cât și malițioase, care pot ajunge la un consens si care pot genera un set de date sigure. Informațiile rezultate sunt puse într-un nou block și salvate permanent în rețea într-un mod ce nu permite modificare integrității acestora.

Soluția propusă are două componente interconectate: blockchain-ul de Sisteme Inteligente de Transport, care înmagazinează datele agregate despre trafic, și sistemul de reputație ,care ajută la crearea unui consens asupra datelor de la utilizatori. Arhitectura sistemului este compusă din 3 actori: utilizatorii – care generează date; nodurile regiune – care agregă datele utilizatorilor dintr-o anumită regiune, nodul master – care combină datele generate de nodul precedent și care le pune într-un nou block.

Lucrarea prezintă teste sintetice care validează cazurile de utilizare ale soluției: utilizatori ce trimit date despre viteze si alerte, și un sistem de reputație echitabil. De asemenea demonstrează că sistemul scalează bine cu numărul de date din sistem.

ABSTRACT

Blockchain is an emerging technology that has shaken the financial sector, and which has the potential to become as impactful as the internet. A blockchain is a network of many interconnected nodes, both trustworthy and malicious, which can reach a consensus and generate valid data. The resulting information is packed into a block and permanently saved on the network in a tamper-proof way.

The proposed solution has two interconnected components: the Intelligent Transportation System (ITS) blockchain, which stores the aggregated user traffic data, and the reputation system, which helps nodes reach a consensus on said data. The system architecture has three main actors: the users - which generate the data, the region nodes – which aggregate user data from a specific region, the master node – which combines data generated by the previous nodes and packs it into newly minted blocks.

The paper presents synthetic tests which validate the use cases of the solution: users reporting speeds and alerts, and a fair reputation system. It also demonstrates that it scales well with the amount of data.

1 INTRODUCTION

1.1 Context

The world is in a state of constant digitalization of all industries: from entertainment to education, from finance to agriculture, and one of the faster-changing ones in the past ten years has been the automotive sector. It should also be noted that cars have become a culture this past century, one that has spread over the entire globe. It has shaped both the way people live, by influencing how cities are designed [1] and interconnected, and the economy, by creating new jobs [2] and enabling other industries to flourish.

One of the main directions that drive automotive innovation is safety. Car manufacturers strive to offer their clients safer and safer rides, and technological advancements enable OEMs to provide new levels of protection with every car generation. Government-backed organizations, such as EuroNCAP, oversee how these new systems act in real-world scenarios and assess how effective they are and how well they protect both drivers and pedestrians.

Most of the tests done by these car safety regulation organizations are either for the structural integrity of the car or for the reactive systems which intervene once the driver makes a mistake or when there is an imminent danger. Over time, these safety nets have proven to be indispensable. Still, a proactive way of tackling possible problems can complement the reactive safety features and offer a new level of trust in the car.

As previously stated, the automotive industry has had a significant impact on modern cities. But there is a new shift now to transition into a more connected state where everything is optimized; an idea called smart cities. Such a municipality uses secure, interconnected data sources like homes, crosswalks, street furniture, traffic lights, air quality sensors to create an ecosystem that can react and inform others of relevant changes. By tracking the traffic in a city, users can receive information that they would not have access to otherwise, information that could then be used either by drivers, to make better decisions when driving, or by the transportation system to streamline traffic.

Intelligent Transportation Systems (ITS) are extensions of the current rudimentary implementations that have not seen any significant change since their invention. Most of them are rigid and programmed to act a certain way, but ITS aims to tackle that problem and offer travelers shorter transition times, better fuel economy, and more safety. Such a system relies on consistent, trustworthy user data, and this raises the problem of privacy, transparency, and fair use. Europe's General Data Protection Regulation imposes that the generated data cannot reveal someone's identity to other untrusted entities. These requirements, along with the fact that past data should also be available without anyone having the ability to tamper it, point towards a possible solution: blockchain.

Blockchain is a decentralized peer-to-peer technology where network participants can verify other users' interactions with the network using specialized consensus protocols. While it is still a new technology, different research [3] and even final products [4] have proven its disruptive potential [5].

1.2 Problem

A possible proactive safety system could offer data regarding average speeds on a specific portion of the road, whether there is a traffic jam, and alerts for potential dangers that users might face, such as potholes, roadkills, storms or dangerous bends.

The implementation should also be able to offer reliable data without being influenced too much by malicious intent. This can be achieved through:

a. A hybrid blockchain – this type of blockchain can be accessed by anyone, just like a public blockchain such as Bitcoin. The main difference is that the consensus algorithm is run by pre-assigned nodes which are trusted, thus also acting as a private blockchain. All other nodes are deemed not trustworthy, and their data is judged based on previous interactions with the system
b. A reputation system – every interaction a node has with the proposed solution is assessed, which affects their reputation negatively or positively. This reputation is used to determine how big of weight the user's answers should have; in other words: how trustworthy that user is.

The proposed solution would ideally be agnostic of the car brand, or it could even run on a smartphone if the car does not have the capabilities of supporting it. This way, all drivers would have access to the same data, and they could also offer their data to create a dataset that resembles the real world as closely as possible. The platform independence of the solution could also aid cities that are trying to convert to "smart cities".

1.3 Objectives

This paper covers a proof of concept implementation of a hybrid blockchain that solves the identified problems. It will collect user data, which will be subject to a consensus algorithm that will generate reliable data that both drivers and Intelligent Transportation Systems can use to create a safer, more streamlined and healthier environment.

The main aim of the thesis is to prove that a system like the one previously described is suitable for the identified use-cases. This will be done through testing with both synthetic tests and real-world data. The latter will be using already existing datasets from Sim2Car,

Politehnica University of Bucharest's traffic simulator, which contains car pathways from both San Francisco and Rome.

The analysis includes checking the efficiency of the reputation system, how well the system performs as more and more users are part of the blockchain, how the number of regional nodes in an area affects performance and the percentage of malicious data that seeps through.

1.4 Paper structure

The next section focuses on existing approaches in both blockchain implementations in ITS and reputation systems in ITS. Afterwards, the following two sections present the proposed solution from both a logic point of view and an implementation point of view. The penultimate chapter highlights the findings of this paper, followed by the conclusion and future work.

2 RELATED WORKS

2.1 Blockchain in ITS

Blockchain technology has been in the spotlight for some time due to the popularity of cryptocurrencies such as Bitcoin and Ethereum. This attention attracted both researchers and developers into improving already existing solutions or creating entirely new ones. Nowadays, blockchain has been widely adopted in domains such as the Internet of Things [6], smart cities [7] and Intelligent Transportation Systems.

In [8], the authors propose a new ITS-oriented blockchain model. The paper focuses on the main advantages such a model brings: security, trust and decentralization. Therefore, it has been considered a starting point for many future works involving blockchain and ITS.

In [9], the authors present the impact of blockchain in many domains. When it comes to ITS, data is at the centre point of everything, which means that obtaining and handling it should be a priority. But other test counter this argument and point to the fact that there could be a need for a new way of thinking, a paradigm shift, to create a robust system that works the way it has been envisioned.

[10], [11] and [12] present a similar goal to that of this thesis, a blockchain-based approach whose aim is to be centred around data availability. The third paper presents a very detailed and up to date implementation that uses already popular blockchain products such as the IOTA Tangle, InterPlanetary File System (IPFS), and the Ethereum Virtual Machine to execute smart contracts.

2.2 Reputation systems

The other main component of the proposed solution is the reputation system that aims to create a fair medium where honest actors can have a more significant say in the result when compared to others that tend to report incorrect data. Since the reputation system will also live on the blockchain, the papers that will serve as a reference will also need to have this characteristic.

In [13], the authors propose a reputation system for sending and receiving files where the final reputation is not stored on the blockchain. If a user wants to know the reputation of another node, they need to compute it by themselves. The system works by creating a positive transaction if the user has successfully received the requested file signed with the private key of the sender. If these conditions are not met, a negative transaction is created. Then the transaction is verified with both parties by requesting a signed proof that contains

the file hash and a nonce sent by the miner doing the verification. In order to prevent a user from creating multiple accounts, the authors suggested linking each user to something personal, such as an IP address.

The authors of [14] propose a reputation management system for vehicular ad hoc networks (VANETs) whose aim is to detect dishonest users and their data and offer incentives based on how reliable the provided data is. These incentives influence a user's reputation, which in turn decides how accurate his data can be.

In [15], the authors describe a system that integrates other actors than drivers. These new entities have the responsibility of checking the traffic data reported by drivers and computing their new reputation after parsing the reports. In this solution, vehicles can communicate with one another and check someone's reputation when need be.

The authors of [16] propose an elegant solution that aims to solve three of the main problems such a system should: guaranteed confidentiality, usage of the reputation metric in traffic event validation, and the ease of implementation in a real-life scenario. The protection of a user's data, one of the most talked-about topics in the world right now, is done through a series of security policies published on the blockchain through which a user can decide what data he wants to share with the system. The user's device does this through a specialized module, which guarantees compliance with every policy. Anyone can also verify this because of the open-source nature of the implementation. The other two tackled problems are inherently part of the solution through the way it has been built and structured.

2.3 Conclusion

Research data shows that all reviewed literature have shortcomings when it comes to offering a heterogeneous system that combines a reputation system and a blockchain ITS system. [13] presented a promising candidate but lacked a reputation system, and the number of already established components can induce an unaccounted-for overhead. Thus, it can only represent a reference from a theoretical point of view.

When it comes to reputation systems, specialty literature offers a lot more options. Despite all this, most of them had particular use cases, none of which aligned with the purpose of this paper. [16] introduced a solution that best fits this paper's use-case and offers valuable insight into how such a system should work. This makes it a suitable reference model for the solution's implementation.

3 TECHNOLOGIES

In a blockchain project, as in all internet-based services, the communication layer is the foundation upon which everything is built. A reliable, safe, and peer-tested backbone offers developers more freedom to concentrate on the other parts of such a complex project.

3.1 Libp2p

Libp2p is an open-source modular peer-to-peer networking stack created by the team behind InterPlanetary File System (IPFS) and used in popular blockchain projects such as IPFS and Ethereum 2.0.

This project aims to make processes addressable, independently of how many hoops it must pass through, while also guaranteeing the security of the new connection. One of libp2p's main advantages is that it is effortless to implement, which makes it so that other projects can more easily communicate with one that uses this stack. Another main advantage is the fact that libp2p is transport agnostic, meaning that it can work through TCP, UDP, UDT, QUIC, and even secure transport protocols such as TLS and SSH. These represent a great advantage for a blockchain solution whose aim is to communicate with other systems such as a smart city.

As previously stated, libp2p offers a secure connection through a component called *secio*. This creates a secure channel between the two processes involved whose handshake is done through asymmetric cryptographic keys.

3.2 Golang

The rest of the solution is developed exclusively in golang, which is a statically typed, compiled programming language developed at Google. It drew a lot of inspiration from other languages such as Algol, Pascal, and C because of their speed and conciseness. At the same time, it tried to resemble more modern languages such as Java and Ruby for their safety. It is also open-source, just like libp2p, which falls in line with the open-source aspect of the proposed solution.

The reasoning behind this choice is that concurrency has been an integral part of the language since its inception. This makes it easier to create multi-threaded programs where every thread continuously runs its own routine, just like the product presented in this paper. The other main reason for choosing this language is the fact that the golang implementation of libp2p is the most popular and active one, which means that it is also the most thoroughly tested.

4 PROPOSED SOLUTION

The proposed solution is Proof of Concept implementation, whose aim is to emulate real-world test-cases and offer valuable insight into how the system performs, how efficient it is, and how well thought out the concept is.

It runs on any operating system that has the golang package installed. This means that the app can be run on any computer and, with the help of the mobile implementation of Go or the JavaScript implementation of libp2p, the project can be ported to handheld devices running Android and iOS.

The solution is a hybrid blockchain that aims to draw benefits from the two main types of blockchains:

- The public aspect offers others the freedom to connect and disconnect from the network whenever they want. It also involves the users into the governance of the system and incentivizes them to act righteously. It enables other actors, such as third-party companies, to be part of the blockchain and use the resulting data for whatever they want. These actors can represent both entities that deliver another product or regulatory companies that check the data's validity.
- The private aspect ensures that only the developers of the blockchain have access to a user's private data. The fact that the solution is open source offers anyone insight into the handling of data, which reassures users that their data is not used for anything else. A private blockchain offers a lot more safety than a public one because it relies on trusted nodes that are pre-approved by an organization.

One of the main problems of popular cryptocurrencies such as Bitcoin and Ethereum is the electricity wasted in the Proof of Work consensus mechanism. In PoW, all nodes in the network have the possibility of finding the hash of the next block in a process called mining. But this means that all nodes work relentlessly to find a specific nonce, which, when hashed with the rest of the block, creates a required number of leading zeros. While this makes the blockchain tamper-proof by creating a time-constraint to mint or change a block, it also makes the entire network account for up to 80 TWh or 0.3% of yearly global electricity use [17].

The solution proposed in this paper circumvents this problem by introducing an architecture where a single trusted node creates all blocks, thus eliminating the need for such an energy-intensive component.

In conclusion, the choice of a hybrid blockchain dictates the design and the implementation of the system.

4.1 System design

In a standard blockchain implementation, all nodes are expected to be the same and communicate between themselves freely. The proposed solution brings some restrictions to this design due to the nature of the blockchain and introduces three specialized nodes, each with its specific function. Figure 1 depicts the placement of these three actors in the system and how they communicate with one another.

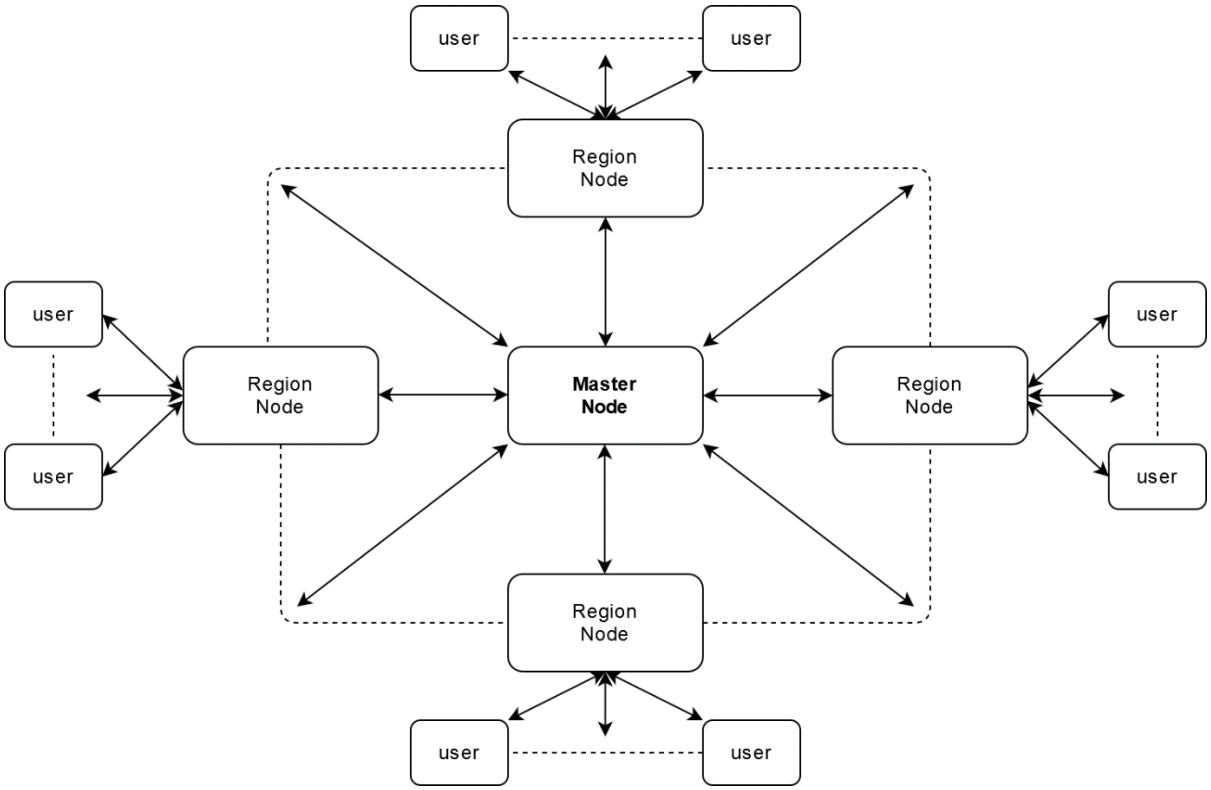

Figure 1. System design of the proposed solution

4.1.1 User nodes

This is an insecure type of node that can be considered malicious. Hence the data report by this node is parsed through the perspective of its reputation. The handling of user-data ensures that no one can access a user's sensitive information and that other external actors cannot trace a user's behaviour. This type of node communicates exclusively with the region node assigned to the location he currently is in and will switch to another region node if he has left the influence area of the first one.

4.1.2 Region nodes

This is a secure node that gathers data from all users located in the region that is under its jurisdiction. Said data is passed through a consensus algorithm that creates new reliable traffic and reputation data. These new values are saved on the blockchain by sending it to the master node.

Note: Every region node has a well-defined area of action such that no two nodes of this type overlap. This ensures the correct parsing of all data, resulting in a trustworthy data source.

4.1.3 Master node

This is also a secure node. It gathers data from all region nodes without checking it because those nodes are trusted, and the security baked into the networking stack, libp2p, ensures the tamper-proof nature of data within transit. The data from all region nodes are packed into a new block, which is added to the internal block of chains and then broadcasted to all subscribed nodes.

4.1.4 Other nodes

Any other node can connect to the master node, thus being on the same level as a region node, which offers other actors insight into the state of the traffic. Such a node could be one that is part of an Intelligent Transportation System or a node that gathers data for a car routing algorithm.

4.2 Solution logic

This subsection provides insight into how the system works, what constraints and dependencies are part of the system and why this is a viable solution for the problem the paper aims to solve.

More details will be presented in the next chapter, which focuses on the particularities of the algorithms and data structures used to create an efficient system, both memory and computationally wise.

As previously stated, the system relies on all three main types of nodes to work as expected. Figure 2 highlight the way these nodes communicate, what data is passed between them and the main actions they have to perform.

The master node is the backbone of the entire system, without which the system would not be able to function. This makes it the most critical node, the first one that needs to be started and the only one that cannot fail.

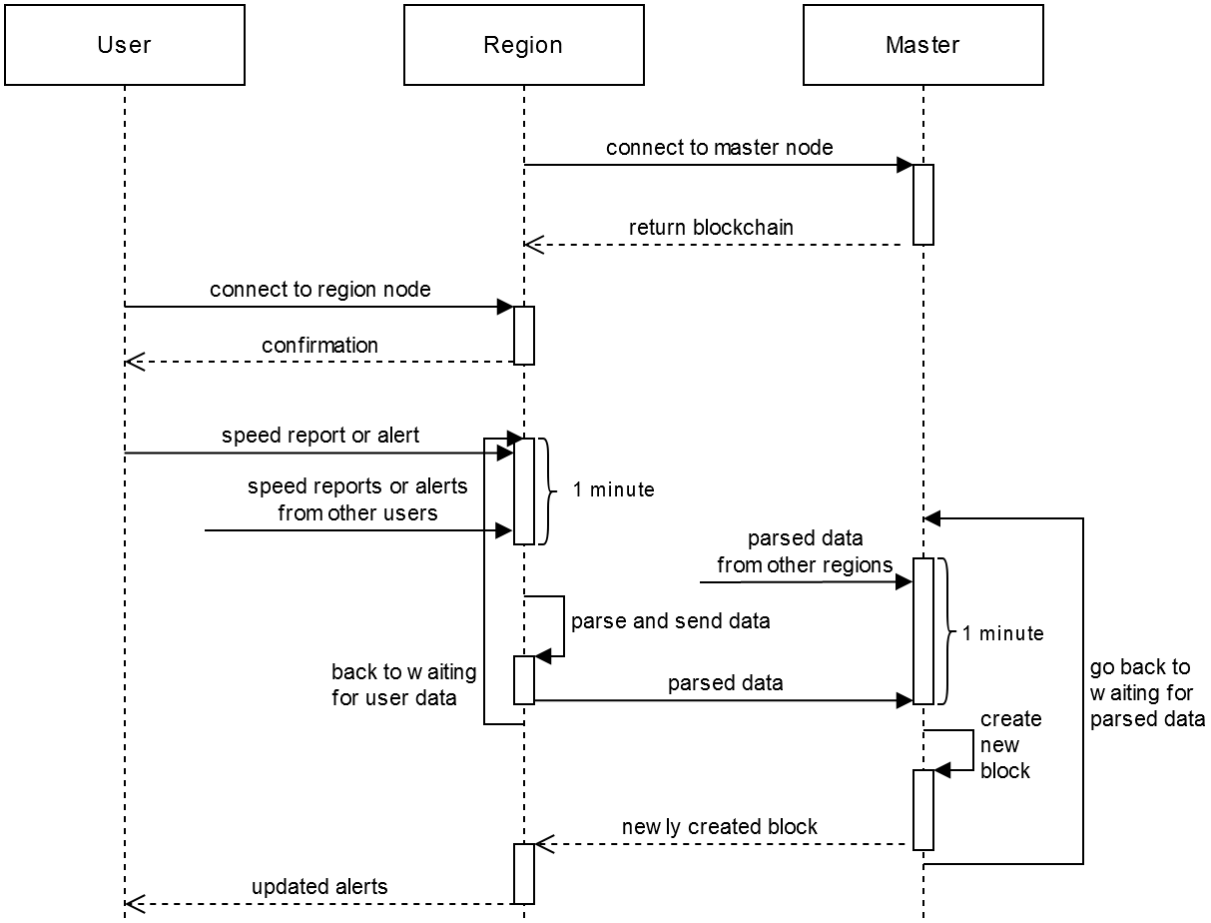

Figure 2. Solution logic

Region nodes can be created at any time, the only constraint being that the supervised area does not overlap with that of any other region node. After such a node connects to the master, it receives the entire blockchain. This enables region nodes to have an overview of all data in the system, which offers better management of user data.

Once a region node has been spawned, it parses the blockchain and offers user nodes the ability to connect to it. After a confirmation has been received, users can start sending data.

The region node collects user data for one minute after which it applies the consensus algorithm. This way, new trustworthy speed reports, alerts and user reputations are created from the aggregated data and sent to the master node.

The master node also collects data for one minute. The fact that trusted nodes created said data and the safe nature of the connection mean that the master node does not need to check what it receives. Everything is then packed into a new block that gets added to the internal chain of blocks and is then broadcasted to all other connected nodes (region nodes, third party nodes).

When a region node receives a new block, it appends the block to its internal chain of blocks, updates specific values such as the users' reputation and sends users new alerts to inform them about possible dangers.

5 IMPLEMENTATION DETAILS

5.1 Chain of blocks design

One of the main attributes of a blockchain is the way data is stored. In this respect, the current State of the Art implementations use two variants:

- A block-lattice data structure, which is a unique implementation of a Directed Acyclic Graph. As stated in the Nano Whitepaper [18], a block is just one transaction. In this case, every account has its own blockchain that contains inbound and outbound transactions. This ensures a higher number of transactions per second.
- A single monolithic block of chains. This is a classic implementation that is present in the two most popular public blockchains: Bitcoin and Ethereum. A node packs an arbitrary number of transactions into a block which is then added to the block of chains and broadcasted to all other nodes.

For the proposed solution, the latter option has been chosen due to:

a. Storage space
   A block contains both data and metadata. While the amount of data is variable, the size of the metadata is always the same because it always consists of:
   - index – the blocks order number in the chain of blocks (the first block is called the genesis block)
   - timestamp – the exact time at which the block has been created
   - hash – the hash of the entire block, not just de data
   - previous hash – the hash of the previous block, which helps in checking data consistency.

   As Figure 3 shows, this means that the same data takes less space if it is packed into a single block than if it is divided into more blocks because of the metadata.

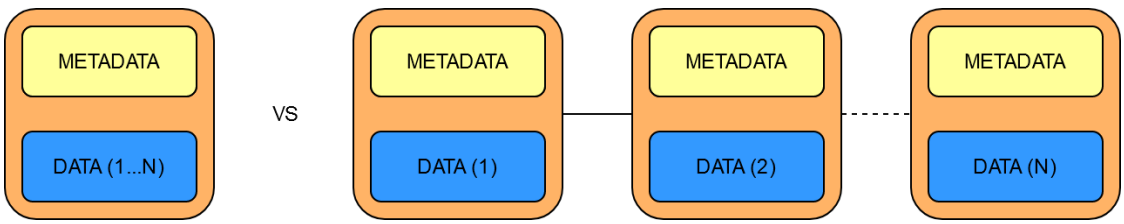

Figure 3. One block vs more blocks – same amount of data

b. The way users or other nodes will interact with the data
   Having a single block at a regular interval makes it easier and more intuitive to query traffic status. This way, all blocks in a specific timeframe will be parsed, without checking whether they hold the requested data.

Figure 4 depicts the design of a block used in the blockchain. The relevant data is comprised of three lists:

- The list of speed reports holds data about the average velocity for a specific location and direction.
- The list of alerts contains relevant data such as an identifier, when it was created, and the votes it has received from users.
- The users' reputation represents the last piece of user-relevant data, and it comes in the form of differences between the new and the old values. This way, if any node wants to know a user's reputation, it must parse the entire blockchain and compute it.

A block contains new data that was not available when the previous block was created.

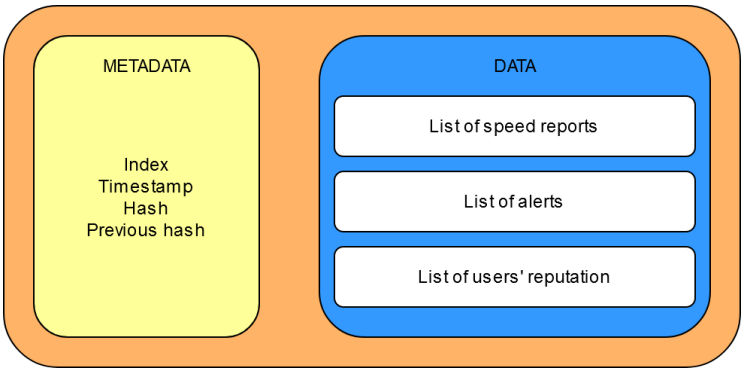

Figure 4. Block structure

5.2 Nodes communication

Nodes communicate through a read-write stream built with the help of libp2p. Every node spawns several goroutines specialized in executing a specific task like sending or receiving a speed report, parsing all received speed reports, or creating a new block. Each job uses a different stream for every type of message: speed reports, alerts, users' reputation and blocks.

Messages only transfer the needed data, without containing any redundancies. This way, the network load is kept at a minimum, resulting in a more responsive system overall.

Users send two types of messages to region nodes:

- Speed reports, which consist of position latitude and longitude and their speed
- Alerts, which consist of the position of the alert, what kind of alert it is and whether it is still active

Every minute, region nodes send the master node only one type of message which contains the three components that make up the data of a block: computed speed reports, alerts and users' reputation.

When a region node receives a new block from the master node, it extracts new alerts and sends them to users that are in that region.

5.3 Computing directions of travel

Every time a user sends its location and speed, the region node computes the bearing (or direction) based on the previous report. If there is no prior location sent by the user, which happens if a user has just connected to the region node, the speed report is ignored.

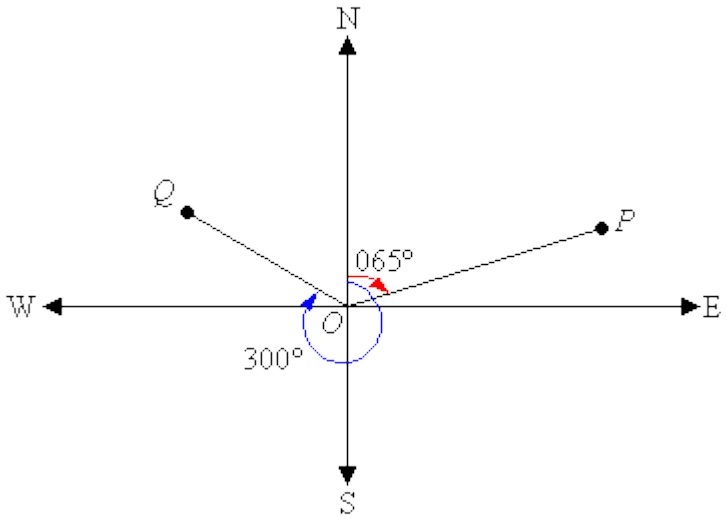

Figure 5. Bearing example

A bearing is the clockwise angle, measured in degrees, between the straight line from a user's location to the north pole and the direction said user is facing. Figure 5 shows two examples of this. In the first one someone has reached point O and faces towards point P, the final bearing being 65°. The second one depicts someone in point O pointing towards Q, with the final bearing equal to 300°.

These bearings are used to compute road direction in a certain spot through a clustering process that outputs the general directions that cars in that location are going. Instead of

using a general-purpose clustering algorithm, such as k-means, the solution uses a more direct approach that is faster and more efficient.

The clustering process uses data about the maximum difference in direction angle between cars moving in the same direction. Said data is dependent on the speed of the vehicle, the distance it moved horizontally, such as when changing lanes, and the time between sending two speed reports.

Figure 6 depicts two cars going on the same road, one of them going straight from point 1 to point 2, the other one changing lanes from point 1' to point 2. There are four variables at play here:

- X – the distance the first car covers
- Y – the distance the second car covers
- Z – the distance the second car moved horizontally
- α – the angle between the direction of the two cars

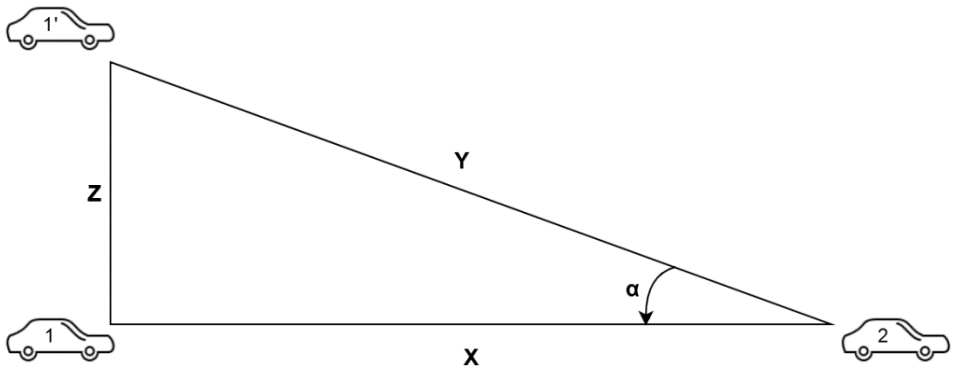

Figure 6. Example of cars going the same way but having different directions

Based on this example, the angle can be computed as:

$$\boldsymbol{\alpha = \arcsin(Z/Y)}$$

The distances the two cars move (X and Y) relies on the velocity of the vehicle and the time between the speed reports. An ideal scenario is considered, where a speed report is sent when starting to change lanes and the other one when the driver has merged onto the desired lane. The relation between them is directly proportional, which means that worst-case scenarios emerge when these values are low.

$$\boldsymbol{Y = \Delta t * speed}$$

Z depends on how many lanes the second driver changes and how wide a lane is. According to data from both Europe and the United States, the average width is 3.5 meters, which means that Z will be considered as a multiple of this value.

$$\boldsymbol{Z = no.\ lanes * 3.5m}$$

Table 1 shows the angle between the directions of two cars going the same way (one changing lanes and the other going straight) by variating the three values it depends on:

- speed ∈ {20 km/h, 50 km/h}
- time between speed reports ∈ {5 seconds, 10 seconds, 15 seconds}
- number of lanes changed ∈ {1, 2, 3} → Z ∈ {3.5 meters, 7 meters, 10 meters}

Table 1. Possible cases for the situation depicted in Figure 6

Speed (km/h)	Δt (s)	No. lanes changed	Z (m)	Y (m)	α (degrees)
50	15	1	3.5	208.3	**1.0**
50	15	2	7	208.3	**1.9**
50	15	3	10.5	208.3	**2.9**
50	10	1	3.5	138.9	**1.4**
50	10	2	7	138.9	**2.9**
50	10	3	10.5	138.9	**4.3**
50	5	1	3.5	69.4	**2.9**
50	5	2	7	69.4	**5.8**
50	5	3	10.5	69.4	**8.6**
20	15	1	3.5	83.3	**2.4**
20	15	2	7	83.3	**4.8**
20	15	3	10.5	83.3	**7.2**
20	10	1	3.5	55.5	**3.6**
20	10	2	7	55.5	**7.2**
20	10	3	10.5	55.5	**10.7**
20	5	1	3.5	27.8	**7.2**
20	5	2	7	27.8	**14.1**
20	5	3	10.5	27.8	**20.7**

This data highlights that a worst-case scenario, where a car is going 20 km/h, changes three lanes in 5 seconds, would have a final angle of 20.7° when compared to a vehicle going straight on the final lane.

Most roads in the world do not have more than four lanes, highways being the only cases in which there might be more. But these are high-speed roads where a car cannot change between the first and the last lane at a speed of 20 km/h. This means that a multi-lane change

incurs a higher velocity than 50 km/h, which results is a smaller angle due to the travelled distance (Y), as shown in Table 1.

Figure 7 exemplifies another scenario in which a car changes two lanes and the other one just a lane, ultimately reaching the same one. Based on the data in Table 1, α could be as big as 14.1°, while β could be equal to 7.2°, which points towards a total direction difference of 21.3°.

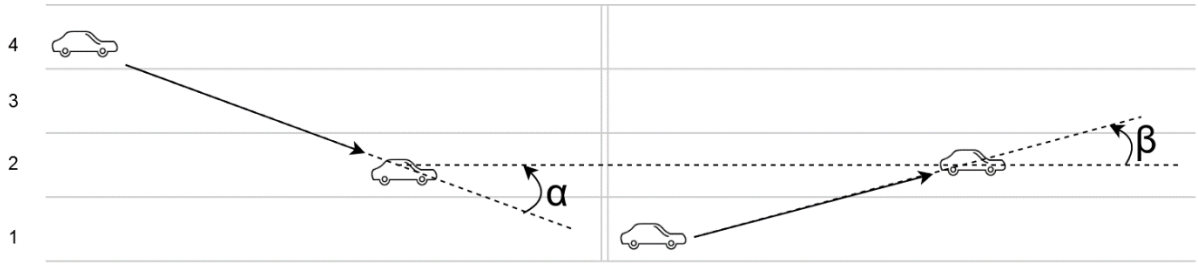

Figure 7. Example of two cars changing to the same lane

Based on the presented data and considering an extra precaution in case of unaccounted for events, the value of 25° has been chosen to be the maximum circle sector in which the direction of all cars going in a direction can be found. This value is used in the clustering process.

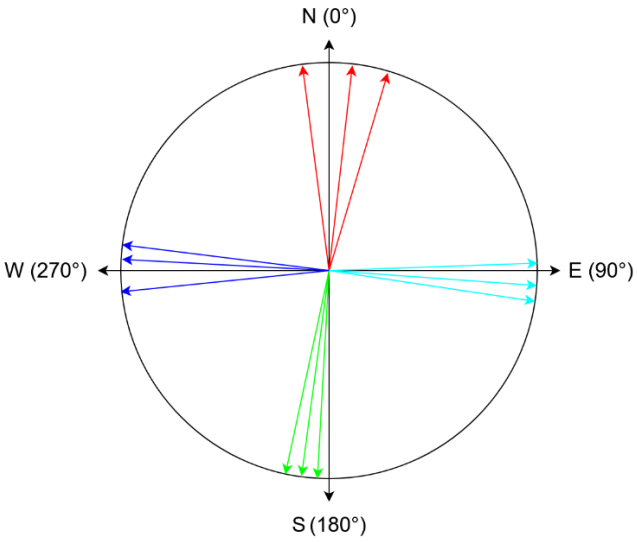

Figure 8. Example of user directions used in the clustering process

The clustering algorithm works by going through the sorted user directions and consists of four steps:

1. Find the first gap greater than 25°. In Figure 8, this would translate to iterating until reaching the first cyan direction, which would constitute the start of a direction sector.
2. Go through direction values until the difference between the current direction and the start of the direction sector is bigger than 25°. The current direction will be considered the start of the next direction sector. In Figure 8, this would coincide with finding the first green direction, which will become the start of the green direction sector.
3. Compute the mean value of the directions in the sector that has just ended. In Figure 8, this translates into calculating the mean value of the cyan directions, which would result in the final cyan direction.
4. If the start of the next sector is different from that start of the first sector computed, go back to step 2. In the example from Figure 8, the algorithm would compute the final green and blue directions. The calculation of the final red direction would coincide with finding the start of the first cyan direction again, which would end the algorithm.

5.4 Speed alerts consensus algorithm

The speed consensus algorithm is one of the most critical parts of the system because it has the responsibility of creating a reliable snapshot of a system in which both fair and malicious users participate.

The algorithm computes the actual speed in a specific place, based both the users' reputation and on the data collected from users that have passed through that point in the one-minute time interval.

After the one-minute mark has passed, the system goes through all geographic points where a speed report has been made and applies the following steps:

1. Compute the main directions using the clustering process previously described.
2. For each direction, calculate the mean speed with respect to the reputation of the user that has created the report.

$$average\ speed = \frac{\Sigma(\text{reported speed} * \text{user reputation})}{\Sigma(\text{user reputation})}$$

3. Check every user's answer against the computed average speed. The value computed at step 2 has a tolerance of 20%, thus, only if a user has reported a velocity that is in the interval [80% * average speed, 120% * average speed], will the answer be counted as correct.

At the end of this algorithm, the region node will have an overview of the traffic under its jurisdiction that will be sent to the master node.

5.5 The alert system

The alert system is used for reporting and interacting with special events such as accidents, potholes, road-side constructions, or any other type of alert that might be of use to drivers. It is an indispensable component of the proposed solution because it informs drivers about events that they should pay attention to, thus encouraging them to act more proactively.

When a user creates an alert, it is sent to the region node, which in turn sends it to the master node to be added to the blockchain. When the new block is created and broadcasted, the receiving region nodes inform their users about new alerts.

This also offers users an opportunity to interact with that alert and give feedback about whether it is valid. Interacting with an already existing alert is conditioned by the proximity between the user and the event. If a user is further than 500 meters from the alert, his vote will be ignored.

When an alert is first created, it is in a voting state that lasts 10 minutes, in which alerts stay active for users to vote. At the end of the 10 minutes, a decision about the validity of the alert is taken, based on the number of votes. At this point, users are also rewarded reputation points based on their votes and on the consensus answer.

If the alert is deemed active, it remains that way until the number of negative votes surpasses the positive ones or until an hour has passed until the last positive interaction. The latter condition tackles the case in which an alert has amassed a lot of positive votes, and it would take a considerable amount of time for the vote balance to tip the other way.

It should be noted that alert votes are represented by a user's reputation — this way, the system is guaranteed to be fair by tackling the impact of malicious interactions.

5.6 Reputation system

The reputation system is a central point of the proposed solution which implements a consensus algorithm where a user's weight is directly proportional to his reputation, and the user's answer directly influences a user's reputation when compared to the final solution of the topic. This creates a feedback loop that makes the system self-stabilize over time and output reliable data for drivers and third parties to use.

The reputation of a user is represented by a number between 0 and 1, and it is used in the two previously highlighted cases. The following subsections present how a user's reputation is affected in both events.

5.6.1 Speed reports

At the end of the speed consensus algorithm, all the reports that a user has made in that one-minute interval are categorized into correct and incorrect answers. Using the cardinal of these two categories and the current reputation that a user has, the system computes a change in reputation.

The main idea of the algorithm is presented below. The speedCoefficient is a number that lowers the impact the algorithm has on the user's reputation. This is a desired effect because users should not be able to easily attain a high reputation because it could be used by malicious users to play the system.

```
answerRatio = (noCorrectAns – noWrongAns) / noAns => ∈ [-1, 1]
reputationChange = answerRatio * speedCoefficient => ∈ [-speedCoef, +speedCoef]
if reputationChange > 0:
        consideredReputation = min(userReputation, 1 – userReputation)
        deltaReputation = reputationChange * consideredReputation
else:
        deltaReputation = reputationChange * userReputation
```

The change in reputation is calculated based on the correct and incorrect answers, and the speedCoefficient previously presented.

If the reputation of the user should grow, then 0.5 represents a critical point where the incentive is at a maximum. This is because, past a certain point, the reward starts to decrease in order to keep the maximum reputation capped at 1.

If the user's reputation should decrease, then the change in reputation is always done with respect to the current reputation of a user. This penalizes trustworthy users more and makes sure that the minimum reputation cannot go below 0.

For example, if a user has a reputation of 0.9 and only makes invalid reports in a one-minute interval in which the region node collects data, his reputation would decrease in relation to 0.9. For this user to get back to approximately the reputation, he would need to have nine one-minute intervals in which his reputation increases at a maximum rate because the growth would happen with respect to 0.1. This creates a balanced system where users cannot easily attain reputation and where high reputation can be easily lost and hard to win back.

5.6.2 Alerts

In the case of alerts, the change in reputation occurs ten minutes after an alert is created, when the voting period ends, and the alert consensus algorithm outputs the validity of the alert.

Users are then rewarded based on how they voted. Computing the reputation change is similar to the speed reputation case presented in the previous subsection. The main difference is that there is always only one answer which is either correct or incorrect. This means that when it comes to alerts, the reputation change will always be the maximum possible.

Because a user automatically sends speed reports, but he has to interact with alerts manually, there are bound to be a lot fewer reputation rewards for alerts than for speed reports. In order to make alerts interaction impactful, the coefficient used for calculating alert reputation changes is bigger than the one used in the case of speed reports.

5.6.3 Initial reputation and coefficients

The initial reputation of a new user plays a critical role in balancing the system. First and foremost, it signifies the weight new users have when deciding the state of traffic. The value should also offer room for the reputation of malicious users to fall.

It also sets a standard of how important reports made by high reputation users are when compared to regular users. This is because ordinary users are not expected to drive too much, which has a direct impact on someone's reputation. Such users are bound to manifest more false positives and false negatives, which would keep their reputation in the middle, at 0.5.

Based on this analysis, the initial reputation of a user has been set to 0.3.

The speedCoefficient dictates how much a user's reputation can change every time a block is created. Three numbers have been chosen as possible values: 1e-6, 5e-5, and 1e-5. Figures 9, 10, and 11 show the impact these values have in the case of a user with an initial reputation of 0.3. The figures also show when the reputation has reached 0.9 and 0.99.

The test that generated data from the figures has been made for 1 million blocks after a user has started using the system. With a block created every minute, this equates to 694 days of continuous use. It is supposed that the user always makes accurate reports.

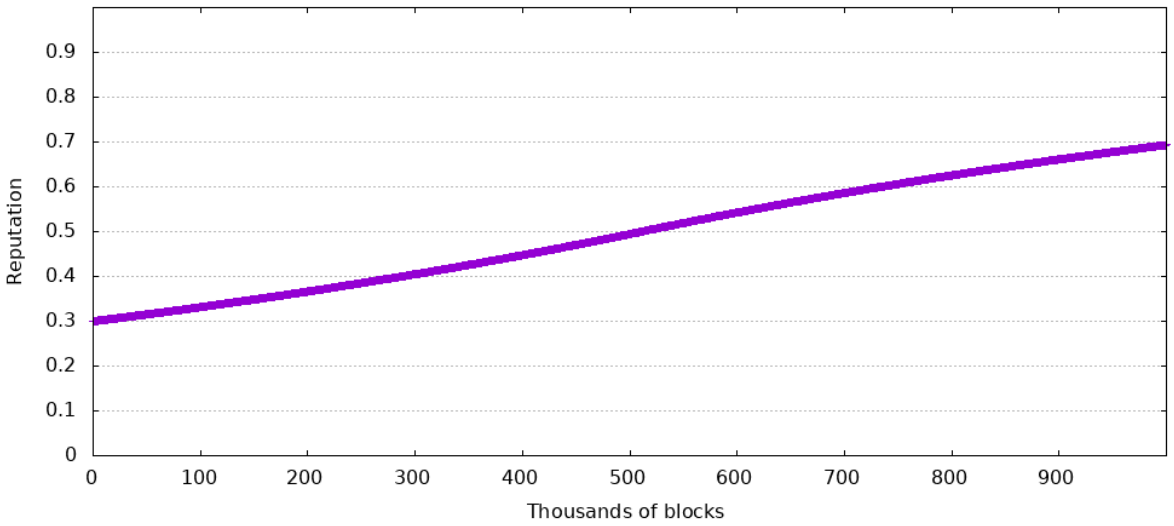

Figure 9. Reputation change over time (speedCoefficient = 1e-6)

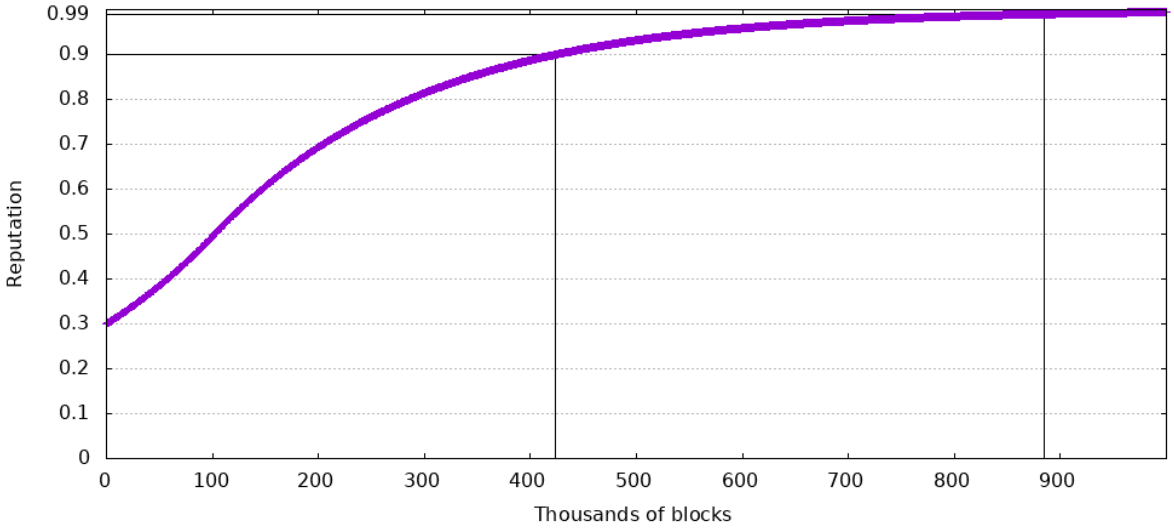

Figure 10. Reputation change over time (speedCoefficient = 1e-5)

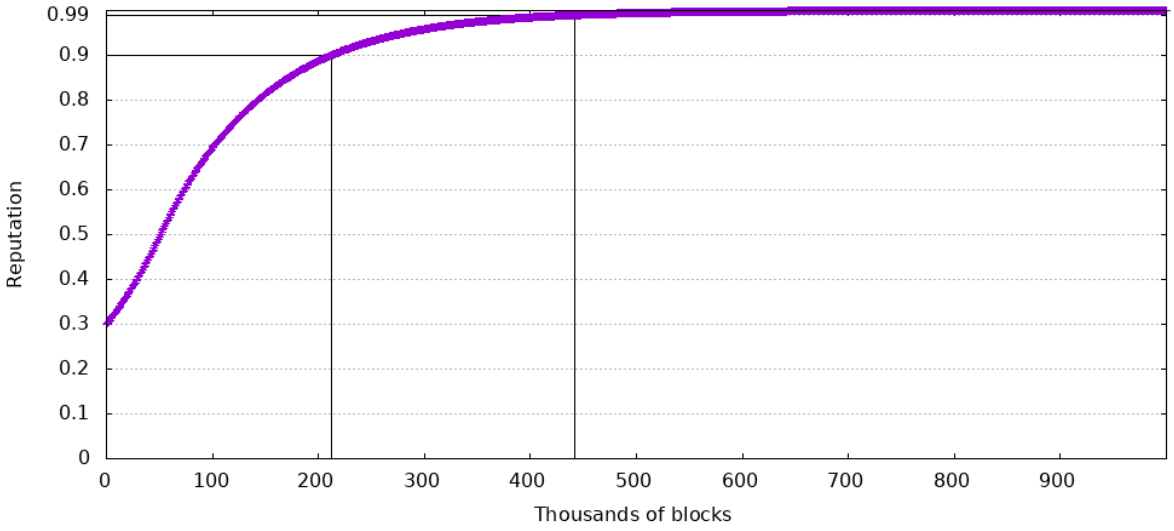

Figure 11. Reputation change over time (speedCoefficient = 5e-5)

In the case of a maximum reputation change of 1e-6, the user only reaches a maximum reputation of 0.6934, which is not a desirable outcome because the change happens too slowly. This spawns another problem: the ineffectiveness of the reputation system. If the change does not happen fast enough, the reputation will not affect the consensus for a very long time.

If the maximum reputation change is 1e-5, ten times greater than the previous values, a user's reputation will reach 0.9 in 212,025 blocks and 0.99 in 442,283 blocks. This means that a user could have a huge influence, of 90% of the possible maximum, in about 920 hours of using the system. At the same time, 99% of the maximum reputation could be reached in 1842 hours of constant use of the system. Even though these values are a lot better than the previous ones, malicious agents might exploit them.

Based on the previous figures, the most balanced value for this variable is 5e-6. In this case, a new user reaches a reputation of 0.9 in 442,052 blocks or 1766 hours, and 0.99 in 884,568 blocks or 3685 hours.

Out of all three cases, the last one manifests the most balanced slope for the considered time interval. It should also be noted that in a real-world case, it will be almost impossible for someone to only make accurate reports. This means that the time it takes a driver to reach 90% of the maximum reputation is prolonged.

6 RESULTS

This chapter focuses on analyzing metrics about how efficient the implementation is, from both memory usage and computational intensity perspectives. It also highlights why some choices have been made when it comes to values used throughout the solution, such as the one-minute timeframe both the region node and the master node gather data for.

Finally, it will present tests using an already existing dataset from Sim2Car, the Politehnica University of Bucharest's traffic simulator, which contains real-world car pathways from both San Francisco and Rome. This will serve as an overview of the feasibility of the proposed solution and of the impact it could have when implemented.

The following data is the result of running the proposed solution on a system equipped with an Intel Core i5 8350u and 16 GB of dual-channel DDR4 RAM running at 2400 MHz.

6.1 Block size

A block is comprised of metadata, speed reports, alerts, and reputation changes that have happened in the one-minute time interval. The size of the metadata is always the same, which means that the size of a block is entirely dependent on the other three components. The following three figures highlight the impact each type of data has on the size of a block.

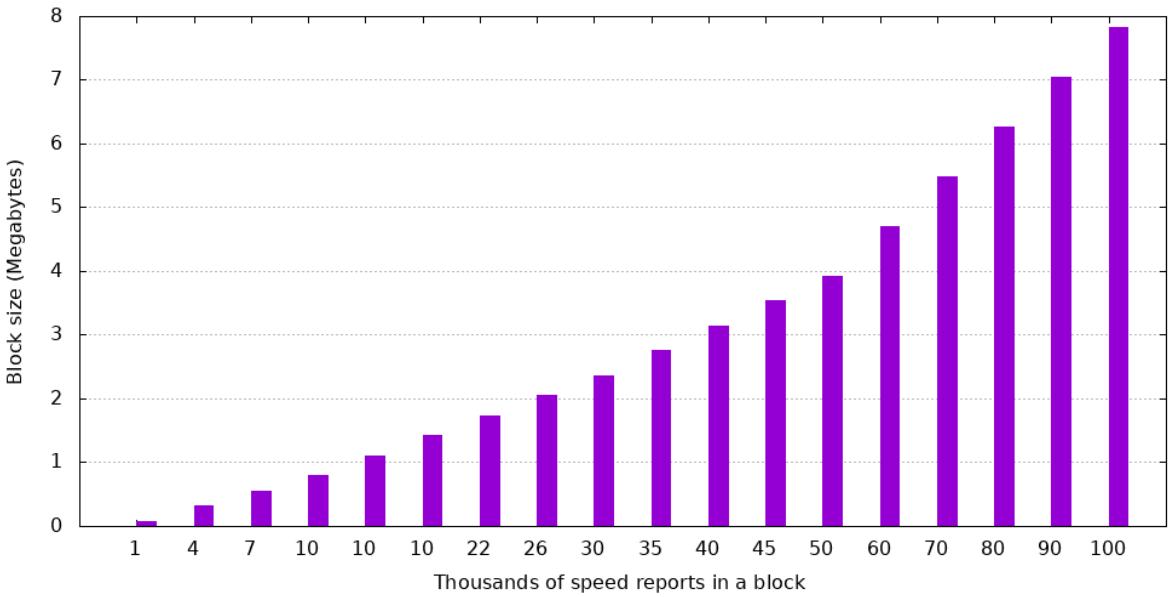

Figure 12. Impact of speed reports on the size of a block

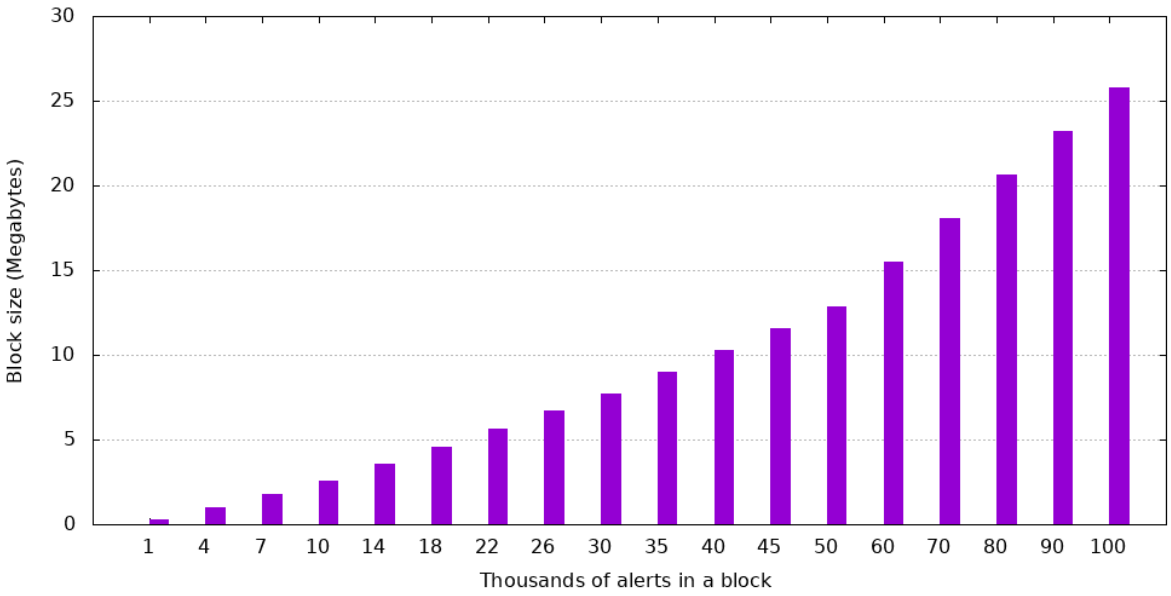

Figure 13. Impact of alerts on the size of a block

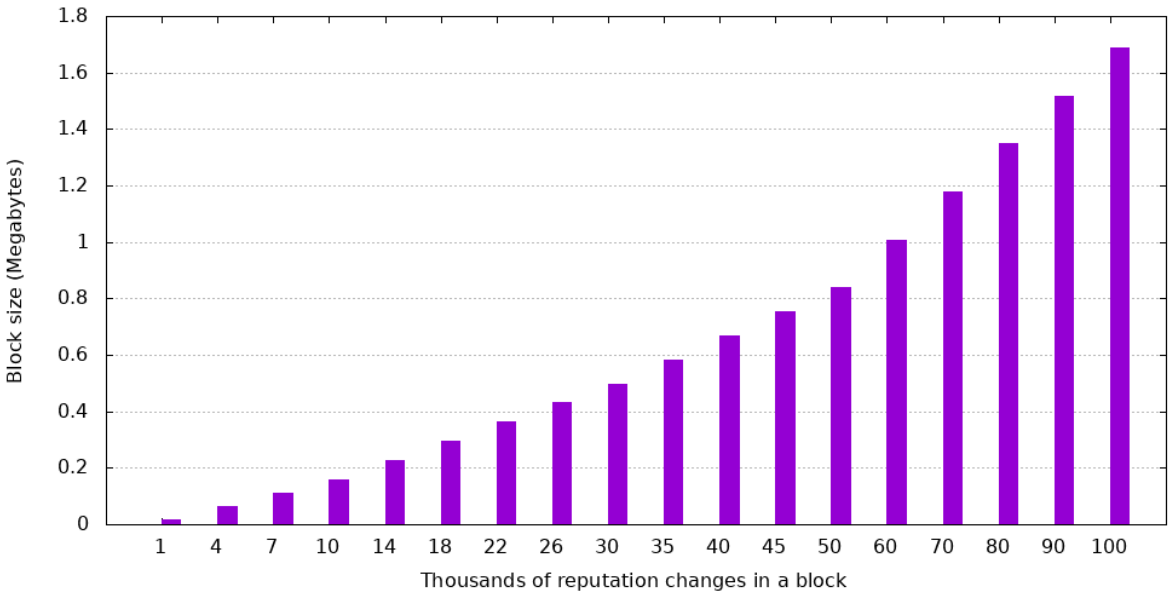

Figure 14. Impact of reputation changes on the size of a block

As the data shows, alerts have the highest impact on the size of a block, with 100,000 of them taking up 25.8 megabytes. This is due to the amount of data needed to store such a piece of information. Because alerts are bound to be created only for exceptional cases, they are not expected to be as prevalent as reputation changes or speed alerts, which mitigates their contribution to the final size.

Reputation changes have the lowest impact due to the simple composition: user ID and the change in reputation. This behavior shows that the proposed solution can handle a high number of users from a memory usage perspective.

Lastly, the number of speed reports in the case of two-way streets has an impact that sits between the other two. Clustering reports that are close can lower the number of speed reports in a block. This way, for example, two speed reports that are 2 meters apart can be combined to save space. This function is not part of the proposed solution, but it could be implemented in a future enhancement.

Data presented in figures 12, 13, and 14 are cumulative, which means the size of a block containing all three types of data can be computed by adding up values from said figures. This means that a block containing 100,000 speed reports, 50,000 alerts, and 100,000 reputation changes is going to be 22.4 MB.

For reference, the most famous cryptocurrency, Bitcoin, has a maximum block size of 1.4 MB, which is a lot lower than a generic block from the proposed solution. When considering that the system is expected to handle even more data, the maximum size of a block is bound to rise even higher. One counterexample is Bitcoin SV, a popular fork of Satoshi Nakamoto's project, which initially had a maximum block size of 128 MB. This figure was later upgraded by the Quasar upgrade, which lifter the maximum block size to 2 GB.

6.2 Alert propagation time

The time it takes a new alert to reach other drivers is critical, especially in the case of an accident. This should be one of the main focuses of the system because it could make a huge difference in conditions such as fog, heavy rain, or snow, where the ability to observe the event and react to it is hindered.

As previously stated, alerts are created by users that send them to the region node. Both the region node and the master node have one-minute time intervals to collect data.

In a best-case scenario, the user would send an alert just before the region node creates the message containing validated data, and this message would reach the master node just before a new block was created. This scenario can be observed in Figure 15.

In this case, the propagation time is equal to:

$$\begin{aligned} &\text{packetTime}(\text{user}, \text{regionNode}) + \\ &+ \text{packetTime}(\text{regionNode}, \text{masterNode}) + \\ &+ \text{blockCreationTime}() + \text{packetTime}(\text{masterNode}, \text{regionNode}) + \\ &+ \text{packetTime}(\text{regionNode}, \text{user}) \end{aligned}$$

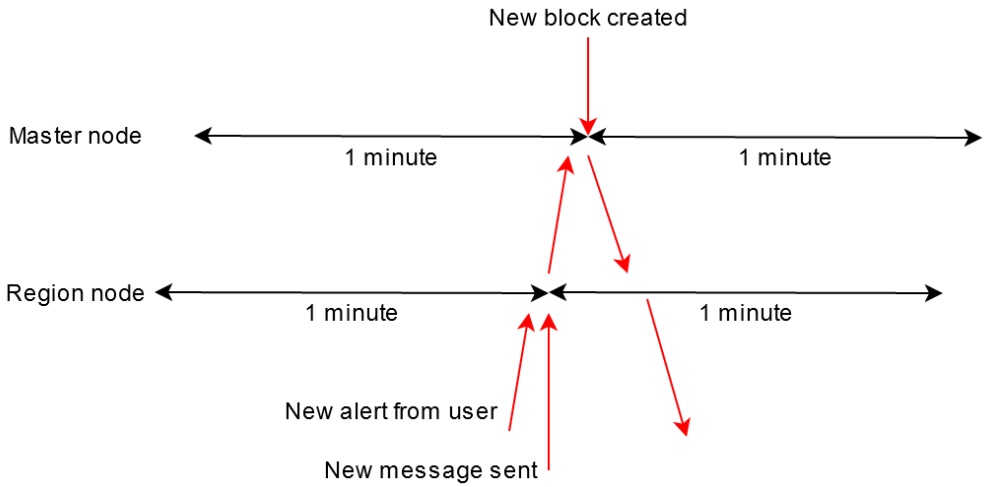

Figure 15. Propagation time - best-case scenario

In a worst-case scenario, the user would send an alert when a region node begins its one-minute data collection period, and the message containing validated data would reach the master node of its own gathering period. Figure 16 depicts this scenario.

In this case the propagation time is equal to:

$$\begin{aligned} &\text{packetTime(user, regionNode)} + \\ &+ \ 1 \text{ minute} + \text{packetTime(regionNode, masterNode)} + \\ &+ \ 1 \text{ minute} + \text{blockCreationTime()} + \text{packetTime(masterNode, regionNode)} + \\ &+ \ \text{packetTime(regionNode, user)} \end{aligned}$$

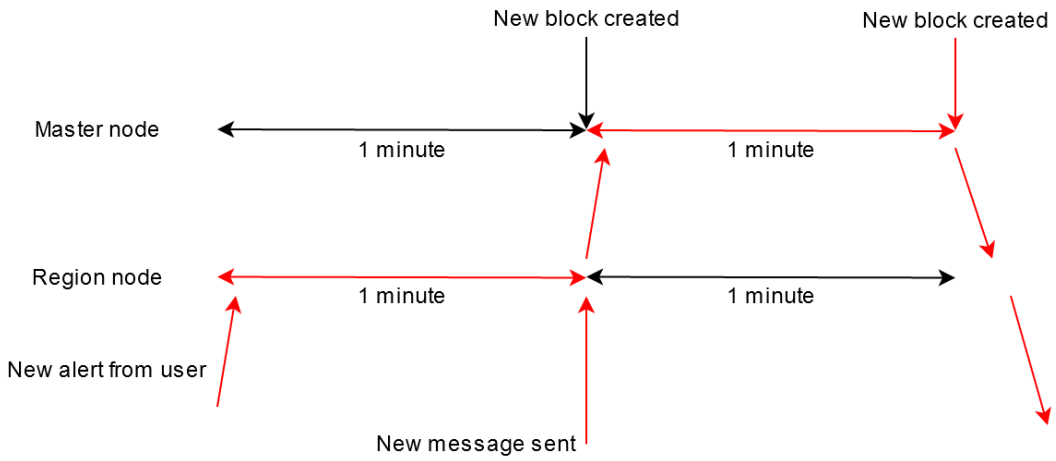

Figure 16. Propagation time - worst-case scenario

Based on the two presented cases, it can be deduced that the alert propagation time is dependent on two factors:

- When the user sends the alert with respect to when the region message will be sent. This factor cannot be influenced, and it is entirely dependent on the user.

- When the region node sends its message in relation to when the master node creates a new block. This factor is baked into the way the system is designed. If a region node connects to the master node at the wrong time, this could make it “forever be at a disadvantage” when compared to the other region nodes. This could possibly be circumvented by making the region waiting interval be dynamic.

A future enhancement could tackle critical events such as an accident. In this case, the region node could send the alert to all users without waiting for it to be added to the blockchain. This action would enable drivers to receive the warning as soon as possible.

6.3 Identifying events from a time interval

One of the main use-cases that the proposed system is going to serve is being a data source for third parties. These can either be a city’s Intelligent Transportation System or another platform that specializes in offering traffic data and predictions. In both cases, the solution proposed in this paper should provide the possibility of finding past data from a specific time interval, as fast as possible.

The block IDs of the interval can be calculated by using three pieces of data: the creation time of the latest block in the chain of blocks, the desired timeframe, and the fact that the master node creates a block every minute.

$$\begin{aligned}
&\text{startOffset} = (\text{latestBlock.CreationTime} - \text{queryStartTime}).\text{ToMinutes}() \\
&\text{endOffset} = (\text{startOffset} - (\text{queryEndTime} - \text{queryStartTime})).\text{ToMinutes}() \\
&\text{startBlockID} = \text{latestBlock.ID} - \text{startOffset} \\
&\text{endBlockID} = \text{latestBlock.ID} - \text{endOffset}
\end{aligned}$$

Since the action only involves four simple mathematical calculations, identifying the blocks associated with a time interval is instantaneous due to the computing power of modern-day computers.

If the querier is a type of node that holds the blockchain, it can immediately start parsing the needed data. There is also a possibility that it relies on other nodes to deliver its query, in which case the querier will have to wait for the delivery of its requested data.

6.4 Real-world tests

This section aims to test the proposed solution in a real-world environment and investigate how the system works when confronted with a situation that it was designed for. It will examine how effective the system is in calculating traffic conditions and the impact the reputation system has on the outcome.

The data used in these tests has been extracted from Sim2Car, Politehnica University of Bucharest's traffic simulator. It contains real-world car traces from San Francisco, Rome, and Beijing, which were obtained from cabs and interpolated to create a multitude of traces that resemble those of real cars. The simulator offers developers a way of creating and testing different implementations for Intelligent Transportation Systems such as smart traffic lights or dynamic traffic signs.

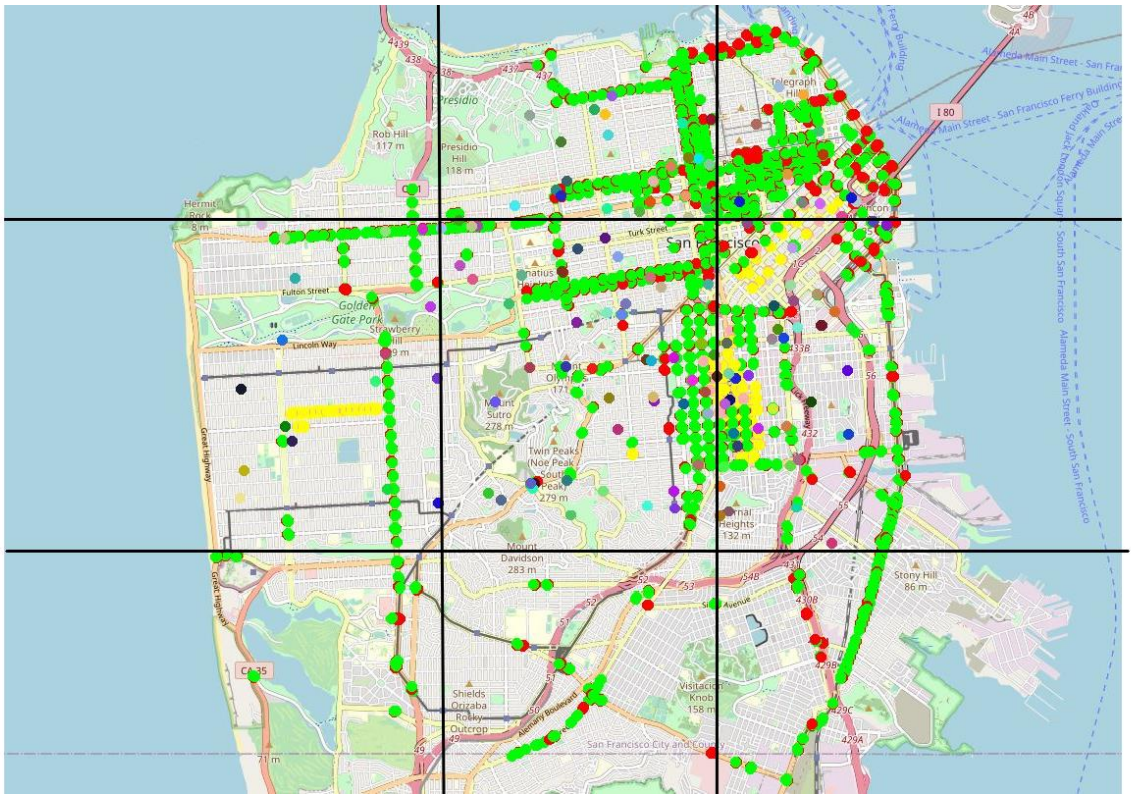

Figure 17. Capture of San Francisco from Sim2Car with proposed region separation

The number of region nodes in a given area is one of the main variables that influence the proposed solution. The higher this figure, the less work each region node has to do when running the consensus algorithm. But if the figure is too high (e. g.: a node for every 500 m^2), the communication overhead could outweigh the benefits of distributing the work. The impact of this variable is dependent on factors such as the hardware and network connecting the nodes; thus, testing could not be reliably conducted.

When the system is first deployed, every user will have the same initial reputation, which means that the system will act as if the reputation system does not exist. As the system continues to adapt, it will learn which users are to be trusted and which ones report malicious data. This way, the proposed solution will act better as time goes on.

The following two figures depict a standard case where the median velocity is 45 km/h, as reported by trusted users, where the impact of the reputation system can be observed. In Figure 18, the malicious users report a velocity of 2 km/h to make the road seem congested and make other drivers choose alternate routes. In Figure 19, malicious users report a speed of 102 km/h, thus giving the impression that the road is free, which will result in more drivers choosing to go that way.

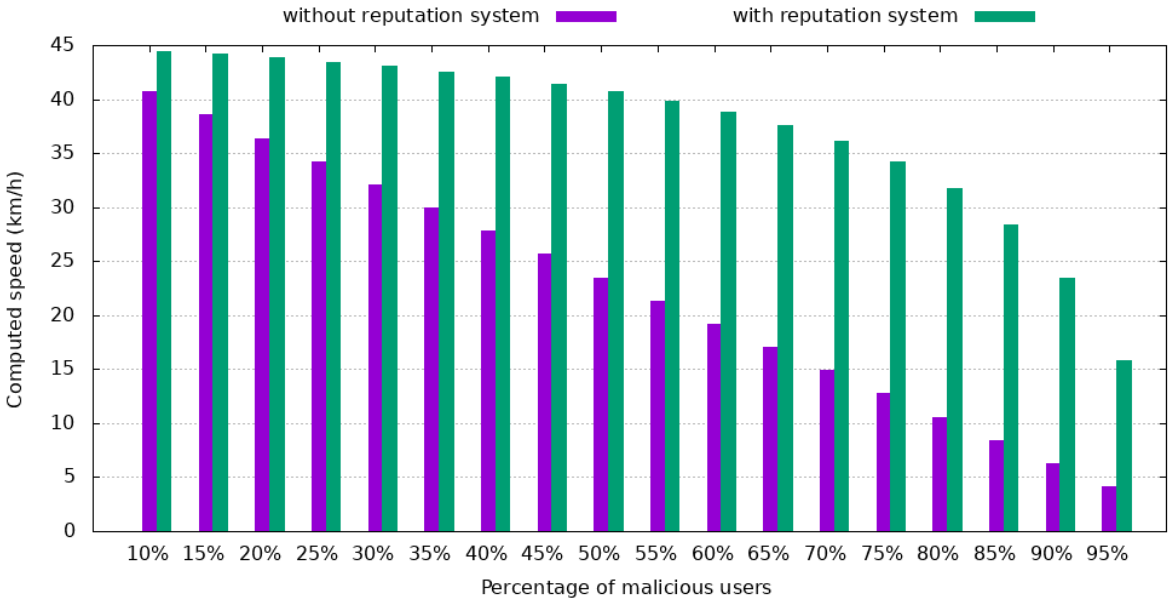

Figure 18. Impact of malicious users reporting lower speeds

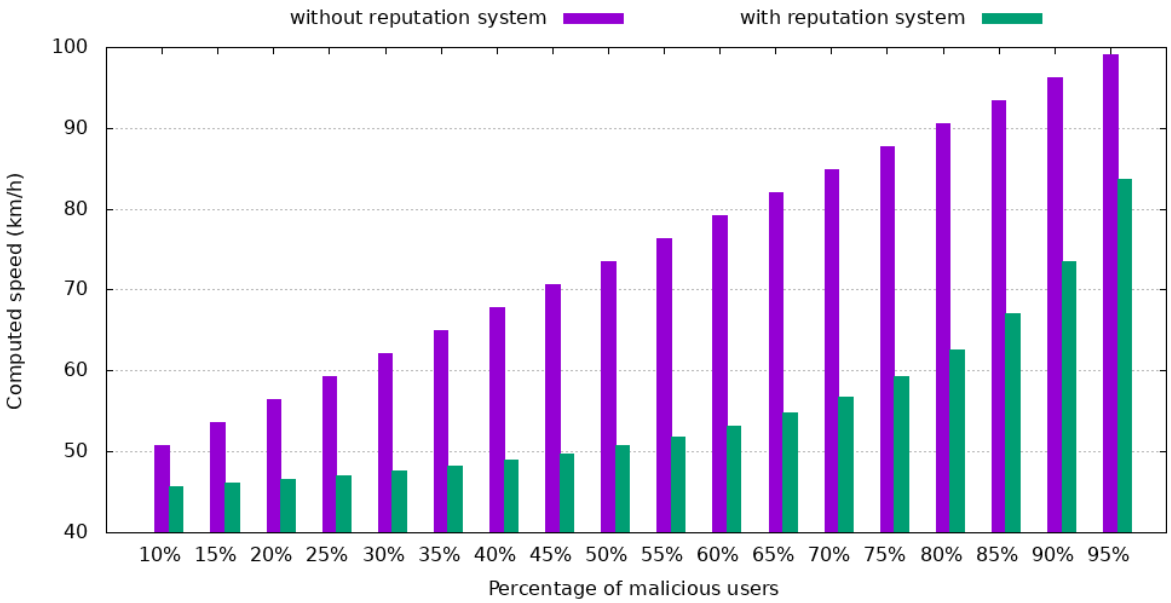

Figure 19. Impact of malicious users reporting higher speeds

Both tests used a reputation of 0.9 for trustworthy users and 0.1 for malicious users.

As Figure 18 shows, even with 55% of drivers intentionally reporting unreliable data, the reputation system ensures that the computed median velocity is 39.85 km/h, which equates to a 12.4% deviation from the correct answer. For less than 40% of users being malicious, the variance is less than 6.6%.

Figure 19 presents the same trend as the previous case. When 55% of users are malicious and report a speed that is more than double of the real one, the final speed only increases by 15%. When less than 40% of users interacting with a report are malicious, a value that resembles a real-world case more closely, the computed velocity has an error of less than 8.7%.

Considering that the process of awarding reputation for speed reports uses a maximum deviation of 20% from the final answer, it means that even if 60% of users are malicious, all users will receive the correct reputation incentive.

These two cases show that the reputation system manages to counteract the impact of malicious users in a typical situation where the proposed solution receives multiple reports in a specific location. Despite this, there are other cases where a malicious user is alone on the street and, since there is no trustworthy user to tip the scale, he has full control over the state of that portion of the road. A future improvement of the proposed solution could take into account the reputation of the reporting user and the usual median velocity at that moment in time, based on previous data.

The authors of [16] have conducted a test that focuses on the impact the data from a similar system has on an Intelligent Transportation System. For this test, it is supposed that malicious users managed to introduce data so that 20% of the traffic light junctions are congested. Table 2 highlights the impact this has on the average speed and on the fuel economy of cars. Testing has been done using the intelligent routing algorithm from Sim2Car in both San Francisco and Rome.

City	Normal traffic		20% of intersections blocked	
	Avg. speed (km/h)	Fuel economy (l/h)	Avg. speed (km/h)	Fuel economy (l/h)
Rome	41.24	11.49	40.76	11.38
San Francisco	41.09	11.4	38.23	11.16

Table 2. Impact of malicious data on average speed and fuel economy

The data shows that for Rome, the average velocity has decreased by 1.16%, and the quantity of fuel used has also reduced by 0.96%. San Francisco shows the same downwards trend: -6.96% for average speed and -2.1% for fuel consumed.

Based on these results, it can be concluded that the average velocity has fallen because cars are routed away from what the system thinks are busy streets. This, in turn, makes it so that more cars are going through the same streets, thus resulting in a lower median speed. Because cars move less, the amount of fuel they use per hour is smaller. But because getting

from point A to point B takes longer, the amount of fuel a driver uses for the same journey is higher than usual.

The proposed system aims to mitigate such attacks from malicious users. Even if unreliable data finds its way onto the blockchain, reports from other users will soon restore the real state of traffic and enable an intelligent routing algorithm to guide the cars as efficiently as possible.

7 CONCLUSION

This paper focused on creating a blockchain system for storing reliable traffic data that can be used in various situations by both drivers and other Intelligent Transportation Systems. Due to the nature of a blockchain, data is challenging to tamper, which ensures its integrity.

The thesis also presented a reputation system that aims to help reach a consensus within a decentralized network of unreliable nodes, represented by the drivers. The network contains two other types of trusted nodes: region nodes – they gather data from users in their region and run the consensus algorithm on said data, and the master node – it gathers reliable data from region nodes and creates new blocks. Libp2p encrypts the communication between all nodes, thus ensuring the integrity of messages.

Future research for the proposed solution will mainly focus on data:

- faster delivery in the case of critical events such as an accident
- more efficient packaging through clustering speed reports that are close together
- better reliability by taking into account metrics based on past data

Another future objective could be integrating the proposed solution with Sim2Car. This would enhance the testing capabilities of both platforms and offer researchers better insight into this field.

8 BIBLIOGRAPHY

[1] Sheller, Mimi & Urry, John. (2000). The City and the Car. International Journal of Urban and Regional Research. 24. 737 - 757. 10.1111/1468-2427.00276.

[2] "Employment trends in the EU automotive industry", https://www.acea.be/statistics/article/employment [Accessed 11 04 2020]

[3] Reyna, A.; Martín, C.; Chen, J.; Soler, E.; Diaz, M. On blockchain and its integration with IoT. Challenges and opportunities. Future Gener. Comput. Syst. 2018, 88, 173–190.

[4] Chen, Y. Blockchain tokens and the potential democratization of entrepreneurship and innovation. Bus. Horiz. 2018, 61, 567–575.

[5] Casino, F.; Dasaklis, T.K.; Patsakis, C. A systematic literature review of blockchain-based applications: Current status, classification and open issues. Telematics Inf. 2019, 36, 55–81.

[6] X. Wang, X. Zha, W. Ni, R. P. Liu, Y. J. Guo, X. Niu, and K. Zheng, "Survey on blockchain for Internet of Things," Computer Communications, 2019.

[7] S. Ibba, A. Pinna, M. Seu, and F. E. Pani, "City sense: blockchain-oriented smart cities," in Proc. of the XP2017 Scientific Workshops. ACM, 2017.

[8] Y. Yuan and F.-Y. Wang, "Towards blockchain-based intelligent transportation systems," in19th International Conference on Intelligent Transportation Systems (ITSC). IEEE, 2016.

[9] P. K. Sharma, S. Y. Moon, and J. H. Park, "Block-VN: A distributed blockchain based vehicular network architecture in smart city," JIPS, vol. 13, no. 1,2017.

[10] V. Ortega, F. Bouchmal, and J. F. Monserrat, "Trusted 5G vehicular networks: Blockchains and content-centric networking," IEEE Vehicular Technology Magazine, vol. 13, no. 2, 2018.

[11] H. Khelifi, S. Luo, B. Nour, H. Moungla, and S. H. Ahmed, "Reputation-based blockchain for secure NDN caching in vehicular networks," in Proc. of Conference on Standards for Communications and Networking (CSCN). IEEE, 2018.

[12] Mirko Zichichi, Stefano Ferretti, Gabriele D'Angelo, "A Distributed Ledger Based Infrastructure for Smart Transportation System and Social Good"

[13] Dennis, R.; Owen, G. Rep on the block: A next generation reputation system based on the blockchain. In Proceedings of the 2015 10th International Conference for Internet Technology and Secured Transactions (ICITST), London, UK, 14–16 December 2015; pp. 131–138.

[14] Zhang, J. A Survey on Trust Management for VANETs. In Proceedings of the 2011 IEEE International Conference on Advanced Information Networking and Applications, Singapore, 22–25 March 2011; pp.105–112.

[15] Mühlbauer, R.; Kleinschmidt, J. Bring Your Own Reputation: A Feasible Trust System for Vehicular Ad Hoc Networks. J. Sens. Actuator Netw. 2018, 7, 37.

[16] Liviu-Adrian Hîrțan, Ciprian Dobre, Horacio González-Vélez - Blockchain-based Reputation for Intelligent Transportation Systems, Sensors article, 2020

[17] IEA (2019), *Bitcoin energy use - mined the gap*, IEA, Paris https://www.iea.org/commentaries/bitcoin-energy-use-mined-the-gap [Accessed

[18] https://content.nano.org/whitepaper/Nano_Whitepaper_en.pdf [Accessed 13.04.2020