IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 20 January 2023, accepted 9 February 2023, date of publication 13 February 2023, date of current version 16 February 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3244327

RESEARCH ARTICLE

Event Message Clustering Algorithm for Selection of Majority Message in VANETs

NARAYAN KHATRI[1], SIHYUNG LEE[2], ABDUL MATEEN[3], AND SEUNG YEOB NAM[1], (Senior Member, IEEE)

[1]Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 38541, South Korea
[2]School of Computer Science and Engineering, Kyungpook National University, Daegu 41566, South Korea
[3]Department of Computer Science, Federal Urdu University of Arts, Science and Technology, Islamabad 45570, Pakistan

Corresponding author: Seung Yeob Nam (synam@ynu.ac.kr)

This work was supported in part by the National Research Foundation of Korea (NRF) through the Korean Government [Ministry of Science and ICT (MSIT)] under Grant 2020R1A2C1010366, and in part by the Basic Science Research Program through the NRF funded by the Ministry of Education under Grant 2021R1A6A1A03039493.

ABSTRACT The trustworthiness of nodes in Vehicular Ad-Hoc Networks (VANETs) is essential for disseminating truthful event messages. False messages may cause vehicles to behave in unintended ways, creating an unreliable transportation system. The efficiency and reliability of the transportation system can be obtained through trustworthy vehicular nodes providing correct event messages. In a VANET, the consensus issue can be resolved by employing blockchain. Even if we employ blockchain in a VANET, the trustworthiness of each message recorded needs to be verified separately since the blockchain itself does not guarantee the trust level of each event message. For instance, when there are multiple conflicting messages associated with a single accident on the road, a vote based on majority opinion can be considered one option for making a decision regarding the accident. In this work, we design the VANET event message clustering algorithm (VEMCA) to resolve the conflicting message problem. Furthermore, we develop a simulator for the VANET environment that demonstrates how the clustering algorithm can be used for event message validation. Experimental results show that our algorithm outperforms state-of-the-art clustering algorithms in terms of accuracy, precision, recall, f1-score, and computational time.

INDEX TERMS VANET, clustering algorithm, trustworthiness, blockchain, simulator.

I. INTRODUCTION

A. BACKGROUND

The Vehicular Ad-hoc Network (VANET) is a special type of network to provide communications among vehicles and roadside units (RSUs) in a specific region. The connection to the vehicle is established through an on-board unit (OBU). The technology for wireless communications among the nodes in the VANET has evolved from IEEE 802.11p Dedicated Short Range Communication (DSRC) to cellular 5G and New Radio (NR) Vehicle-to-Everything (V2X) (i.e., cellular 5G NR V2X) [1]. This technology shift is required to achieve low latency, and high reliability, and to meet high-bandwidth requirements for V2X applications. There are two types of communication in a VANET; vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I). In V2V communication, vehicles interact with other vehicles through OBUs to exchange their own traffic-related information, such as speed, location, and direction. This helps to reduce traffic accidents and avoid congestion on the road. A safety application installed in the vehicles uses messages from other vehicles to determine potential threats, like accidents, traffic jams, slippery roads, etc. This information is delivered to the driver through the safety application in the form of warning messages in screen alerts or audible alerts. On the other hand, V2I communication is between vehicles and roadside infrastructure like traffic lights, radio frequency identification (RFID) readers, cameras, radar (radio detection and ranging), lane markers, parking meters, etc., in order to provide road information to the vehicles. Different sensing technologies provide both safety and mobility benefits to autonomous

The associate editor coordinating the review of this manuscript and approving it for publication was Chao Tong.

vehicles, allowing the infrastructure to warn of potential hazards and optimize traffic flow. The combination of V2V and V2I communications is believed to be promising technology, enhancing efficient road transportation and reducing the number of deaths due to traffic accidents. Through these technologies, vehicles can share real-time data with each other and with road-side infrastructure, such as RSUs, that help to prevent accidents.

Vehicular Ad-hoc Networks (VANETs) have special characteristics, like high mobility and a rapidly fluctuating network topology, which distinguish them from the conventional Mobile Ad-hoc Network (MANET) or other ad hoc networks. Any vehicle can join a VANET based on its communication range. VANET nodes are highly dynamic, and thus, establishing trust among nodes in this dynamic environment is challenging, because the vehicles interact with each other for a short time, disappear suddenly, and then may reappear [2].

B. MOTIVATION

In VANETs, vehicles interact with each other by transmitting safety messages or event messages as well as non-safety messages or beacon messages. The trustworthiness of a node in a VANET is similar to the trustworthiness of safety messages and non-safety messages that are transmitted by the node. Safety messages such as accident information need to be delivered to other nodes in the VANET without delay. There have been cases of attacks on such networks, as discussed in [2]. Some vehicles may provide fake event information to surrounding vehicles to gain road privileges or for malicious purposes (e.g., message modification and false message generation) [3]. Thus, in order to avoid possible malicious or false messages in the network, we must ensure that nodes participating in VANET communications are trustworthy. The trustworthiness of a vehicular node is directly proportional to the trustworthiness of the messages it generates in the network. Trustworthiness of nodes in a VANET is different than security attacks. Consider a case where a legitimate vehicular node is sending false messages. Here, we cannot apply cryptographic techniques to measure the trustworthiness of this node [2]. Neither can the authentication method be applied in the VANET due to its ephemeral nature. Conventional cryptographic techniques may not be good enough to establish trust in a VANET [3]. Blockchain has been used to address trustworthiness issues in VANETs [4], [5], [6], [7]. Blockchain is a distributed ledger technology (DLT), in which records are shared among nodes in the network. By design, a blockchain is secure and tamper-resistant [8]. However, it cannot guarantee the trustworthiness of each event message. Thus, we developed a novel algorithm for event message clustering: the VANET event message clustering algorithm (VEMCA). This algorithm will cluster similar event messages based on event type, vehicle positions, and event detection timestamps. Miner nodes can deploy this algorithm for event validation, and can determine the trustworthiness of each message. Miner nodes can be RSUs or vehicles with high computational power. The use of this algorithm in conjunction with blockchain technology is expected to contribute to the prevention of false messages.

C. BACKGROUND ON TRUST MANAGEMENT IN A VANET

Trust management can be categorized into several approaches: cryptography-based, fuzzy-logic-based, blockchain-based, machine-learning-based, infrastructure-based, game-theory-based, recommendation-based, etc. [2]. In recommendation-based approaches, a vehicular node calculates the trustworthiness of another node based on trust value recommendations from surrounding nodes [2]. Kerrache et al. [9] proposed the T-VNets architecture for trust management in VANET messages, which is based on the European Telecommunication Standards Institute (ETSI) Intelligent Transportation System (ITS) standard. When a direct neighbor vehicle's behavior changes, the trust establishment process triggers a watchdog module, in which the vehicle generates either a positive or negative recommendation to its neighbors. These recommendations are further used for computing trust values of those neighboring vehicles. The problem with this approach is that for the same vehicle, two or more neighboring vehicles may have different recommendation values. This discrepancy causes data synchronization problems and a trust value selection dilemma. The trust establishment models can be entity-based, data-centric, or hybrid trust models [3]. Entity-based trust models are concerned with the trustworthiness of the nodes themselves, whereas data-centric trust models are concerned with the trustworthiness of messages sent from a given node. The entity-based and data-centric trust models are combined in the hybrid trust model. Many researchers have developed node trustworthiness models based on adjacent peer node interaction history information [3]. The lightweight self-organized trust (LSOT) model for VANETs was suggested by Liu et al. [10]. For calculation of a node's trust value, the model additionally considers trust recommendations from trustworthy neighbors. The disadvantage of these models is that two or more nodes may have different trust values for the same node. When a vehicle receives an event message from a neighboring vehicle, it wants to know its trust value. For this, it asks for the message sender's trust value from neighboring peer vehicles. If two or more nodes provide conflicting trust values for the same vehicle, it is hard for the vehicle to determine which information is correct. As a result of this inconsistency, a trustless vehicular network environment may emerge, and we can-not rely on safety messages sent by a particular vehicle. Fig. 1 illustrates the data synchronization issues in VANET trust management. As shown in Fig. 1, when vehicle V_Z receives an event message from vehicle V_X, it will ask for the trustworthiness values of V_X maintained by neighboring peer vehicles (V_A and V_B in this example). We can see how various vehicles can hold contradictory trust values for the same vehicle (0.8 from V_A and 0.3 from V_B). As a result, V_Z can-not know whether the message is trustworthy or not.

The Infrastructure-based trust models have been established for VANETs [11], [12]. The problem with these

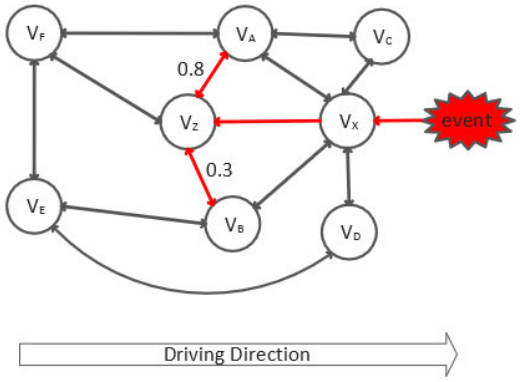

FIGURE 1. Trust value synchronization issue in entity-based trust management mechanisms.

models, however, is the issue of centralization. The entire trust establishment method will be impeded if the central entity fails.

D. CONTRIBUTIONS

The main contributions of this paper can be summarized as follows:

- We propose a novel clustering algorithm for event messages in VANETs. Miner nodes can use this algorithm for event message validation and distinction based on the majority of reports from the vehicles near the event spot. This algorithm is unique and helps to overcome the drawbacks of the K-Means clustering algorithm for K-value selection as well as high computational costs.
- We build a simulator that emulates the VANET environment along with the proposed clustering capability. The simulator can mimic various road scenarios like traffic accidents and traffic jams. According to these events, it can also generate reports and cluster them. This simulator demonstrates the proposed algorithm and creates our own dataset for clustering performance evaluation.
- Experimental results show that our algorithm outperformed K-Means and state-of-the-art clustering algorithms in terms of accuracy, precision, recall, f1-score, and computational cost.

The abbreviations used in this paper are in Table 1.

E. PAPER ORGANIZATION

The rest of the paper is organized as follows. Section II provides related work on the topic of trust management in VANETs using blockchain technology. Furthermore, we survey clustering algorithms that are developed for VANETs in Section II. Section III describes how blockchain can be applied to handling of event messages in VANETs. The detailed explanation of our proposed VANET event message clustering algorithm is given in Section IV. The description of the simulator is provided in Section V. Section VI provides the experimentation process and evaluation of the results. The paper ends with conclusions and future work in Section VII.

II. RELATED WORK

Machine-learning-based clustering techniques have been developed for VANETs. However, they are focused on cluster formation of vehicles, and selection of appropriate cluster heads [13]. The main focus is on efficiency and the stability of the clusters. Taherkhani and Pierre proposed a data congestion control strategy in VANETs using a machine learning clustering algorithm [14]. The messages are clustered using a K-Means clustering algorithm based on features such as message size, validity, and the type of message. The initial centroid, k, needs to be determined in the beginning and set to the number of messages received by the RSUs. Hussain and Chen proposed a clustering technique for VANETs based on hybrid K-Means and Floyd-Warshall algorithms [15]. The main goal of Hussain and Chen's work is cluster formation and cluster head selection. In [16], a trust mechanism for secure message exchange in VANETs was proposed. Clustering of beacon messages is performed based on the coverage range of RSUs. The RSU maintains a table consisting of vehicle IDs, positions, directions, velocities, and restricted neighborhoods. This RSU table can be referred to by vehicles determining the trustworthiness of messages. However, maintaining this table is difficult. Ardakani et al. proposed a cluster-based routing protocol for VANETs [17]. That protocol partitions the network based on node mobility by using a distributed clustering algorithm. The nodes moving in the same region with the same road ID and lane number are clustered. The mobility parameters are node movement region, road ID, and direction. Clustering reduces network traffic by allowing packet aggregation at each CH.

Fuzzy logic has been used to maintain the stability of a clustering algorithm through appropriate cluster head selection in the VANET [18]. A stabilization factor is calculated by the fuzzy logic system based on the relative speed and distance between vehicles in a region. The vehicular node having the higher weighted stabilization factor will be elected as the cluster head. This work was further enhanced to form more stable clusters in VANETs [19]. Here, the authors combined previous metrics for relative speed and distance between vehicles with vehicle acceleration for the cluster formation procedure.

In VANETs, trust management techniques might rely on a central server [20], [21]. A reputation-based announcement scheme for VANETs was proposed by Li et al. [20]. Here, the message receivers report on the credibility of messages in the form of a feedback report. The central trusted party will collect, update, and certify the report score. With the increased flow of communications in the VANET environment, centralized servers may be unable to handle the increased demand. To deal with trust management difficulties, we need decentralized systems. Recently, blockchain-based data-centric trust management solutions for VANETs have been presented [6], [7], [22], [23]. Lu and colleagues developed BARS, a blockchain-based trust management system for VANETs [4], [5]. The authors designed a hybrid reputation management system based on message authentication and

TABLE 1. List of abbreviations.

Abbreviation	Full Form
DBSCAN	Density-Based Spectral Clustering of Applications with Noise
DENM	Decentralized Environmental Notification Message
DLT	Distributed Ledger Technology
DSRC	Dedicated Short Range Communications
ETSI	European Telecommunication Standards Institute
FC-Means	Fuzzy C-Means
GMM	Gaussian Mixture Model
ITS	Intelligent Transportation System
MANET	Mobile Ad-Hoc Network
NR	New Radio
OBU	On-Board Unit
PoW	Proof-of-Work
RFID	Radio Frequency Identification
RSU	Roadside Unit
SHA-256	Secure Hash Algorithm 256-bit
SVM	Support Vector Machine
VANET	Vehicular Ad-Hoc Network
VEMCA	VANET Event Message Clustering Algorithm
V2I	Vehicle-to-Infrastructure communications
V2V	Vehicle-to-Vehicle communications
V2X	Vehicle-to-Everything Communications

recommendations from neighboring vehicles. A blockchain-based decentralized trust management system for VANETs was proposed by Yang et al. [7]. The authors used the distance measure between message sender and event location for rating generation and credibility analysis of event messages. Bayesian inference was used to calculate the credibility of any event.

Entity-based trust models and hybrid trust models are also being developed [3], [24], [25], [26], [27]. Kudva et al. [24] proposed scalable blockchain-based trust management in the VANET routing protocol. The trust value is calculated in two stages. First, is obtaining direct trust scores from neighboring nodes. Secondly, a consortium blockchain-based system is developed in which RSUs serve as trust score validators. Shrestha and Nam proposed a hybrid trust model for trustworthy event information dissemination in VANETs [3]. Trust is calculated based on the trust level opinions of the vehicles near the vehicle that transmits the message. However, they do not address the trust value synchronization issue in a distributed VANET environment. In [26] and [27], a new sort of blockchain that is ideal for VANETs was proposed, which holds vehicle event data for a specific geographic region. The trustworthiness of nodes and messages is determined after each message receiver calculates a trust value and uploads it to the RSUs.

Table 2 provides a comparison of related work for trust management in VANETs. Based on the above literature review, we conclude that the majority of the work for VANET trustworthiness is based on recommendations from neighboring vehicles, which is an untrustworthy approach. When malicious vehicles attempt to act selfishly in order to increase their own trust values, these systems may produce contradictory results. As a result, we require a different approach to addressing the trust management issues in VANET systems. Furthermore, clustering algorithms that are developed for VANETs focus mainly on stability and clustering efficiency through appropriate cluster head selection [13], [28]. There is a limited number of clustering techniques for maintaining the trustworthiness of the VANET event message itself. Thus, the proposed clustering algorithm has a different goal: determining the trustworthiness of conflicting event messages.

III. APPLICATION OF BLOCKCHAIN FOR HANDLING OF EVENT MESSAGES IN VANET

Blockchain is a distributed ledger technology (DLT) that records transactions in blocks that are connected using cryptographic hashes. It has features such as immutability, decentralization, enhanced security, use of consensus mechanisms, etc. Each block consists of a block header and a list of transactions. The block header consists of the hash of the previous block, a timestamp, merkle root, nonce, network difficulty target, etc. The list of transactions in a block is stored as a Merkle tree, with leaf nodes representing the hash of the transactions, and non-leaf nodes representing the hash of the child nodes. Fig. 4 illustrates this mechanism in more detail. When a miner node gets transactions, it tries to group them together into a block and add them to the blockchain. However, numerous miner nodes might attempt to add blocks to the blockchain. In this situation, we need a means to keep the block generation rate consistent. As a result, the network establishes a difficulty target. In order to add a block to the blockchain, the hash of the block must meet the network's difficulty target. Miners change the nonce value, which is essentially a number, to see if the block hash meets the difficulty target. Increasing the number of miners will also increase the difficulty level. Bitcoin uses the

TABLE 2. Comparison of trust management approaches in VANETs.

Ref. #	Trust Model	Blockchain Employed?	Trust Computation	Issues	Event Message Validation and Distinction
[20]	Data-Centric	✗	Reputation Server	Risk of centralization	✗
[3]	Hybrid	✗	Recommendation-based	Trust value synchronization issues	✗
[4], [5]	Data-Centric	✓	Reputation score	Involvement of multiple infrastructures	✗
[7]	Data-Centric	✓	Bayesian inference	Trust value synchronization issues	✗
[26], [27]	Hybrid	✓	Direct trust	Trust value synchronization issues	✗
[23]	Data-Centric	✓	Direct and recommendation-based	Trust value synchronization issues	✗
[24]	Entity-Based	✓	Recommendation-based	Trust value synchronization issues	✗
[29]	Hybrid	✓	Direct and recommendation-based	Influence from malicious nodes	✗

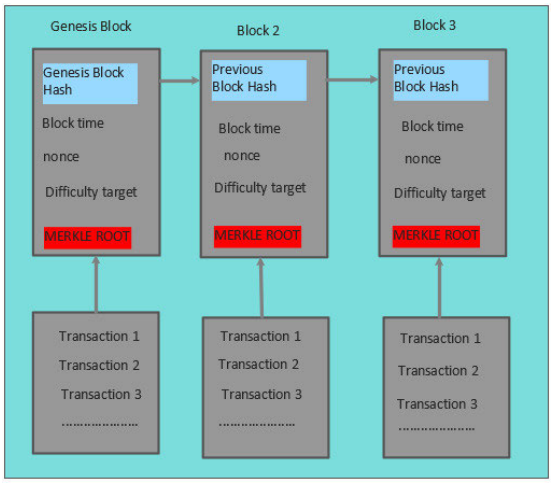

FIGURE 2. The linking of blocks in a blockchain by using hash chains.

SHA-256 hash algorithm to create hashes for transactions. The node first solving the proof of work (PoW) puzzle is the one that gets the chance to add a block to the network. The miner broadcasts the block to other nodes in the network. Other miners check the validity of the mined block and stop mining their current block. In this way, consensus among different nodes is achieved (i.e., miners will agree on the validity of the mined block). The process of mining a new block starts all over again.

Fig. 3 depicts how blockchain can be applied to handling event messages in a VANET. There are two types of nodes in the architecture: event message generation nodes and miner nodes. Vehicle nodes are usually responsible for forwarding messages to other vehicles and RSUs. Miner nodes are those with high computational power and capable of performing more computationally complex tasks like detecting whether an event message is true or false. If a vehicle has high computational power, it can play the role of both event message generation and mining node in the proposed model.

One block in this blockchain can contain the following types of messages:

(A) Type I messages: Statements, i.e., actual messages in VANET transactions such as traffic accidents, traffic jam, ice on the road, etc.
(B) Type II messages: Trust values of nodes

Type I messages contain actual statements of VANET events. These are the messages exchanged between the vehicles and RSUs in the VANET blockchain. They depict various road events, such as traffic accidents, traffic jams, signal violations, adverse weather conditions, etc., throughout the VANET. Type II messages contain trust values of vehicular nodes, which are computed by the miner nodes.

The event message (M_E) consists of the event ID (E_{ID}), vehicle location information (V_{LOC}), and the event detection timestamp (*Timestamp*). The format of an event message is shown in (1):

$$M_E = (E_{ID}, V_{LOC}, Timestamp) \tag{1}$$

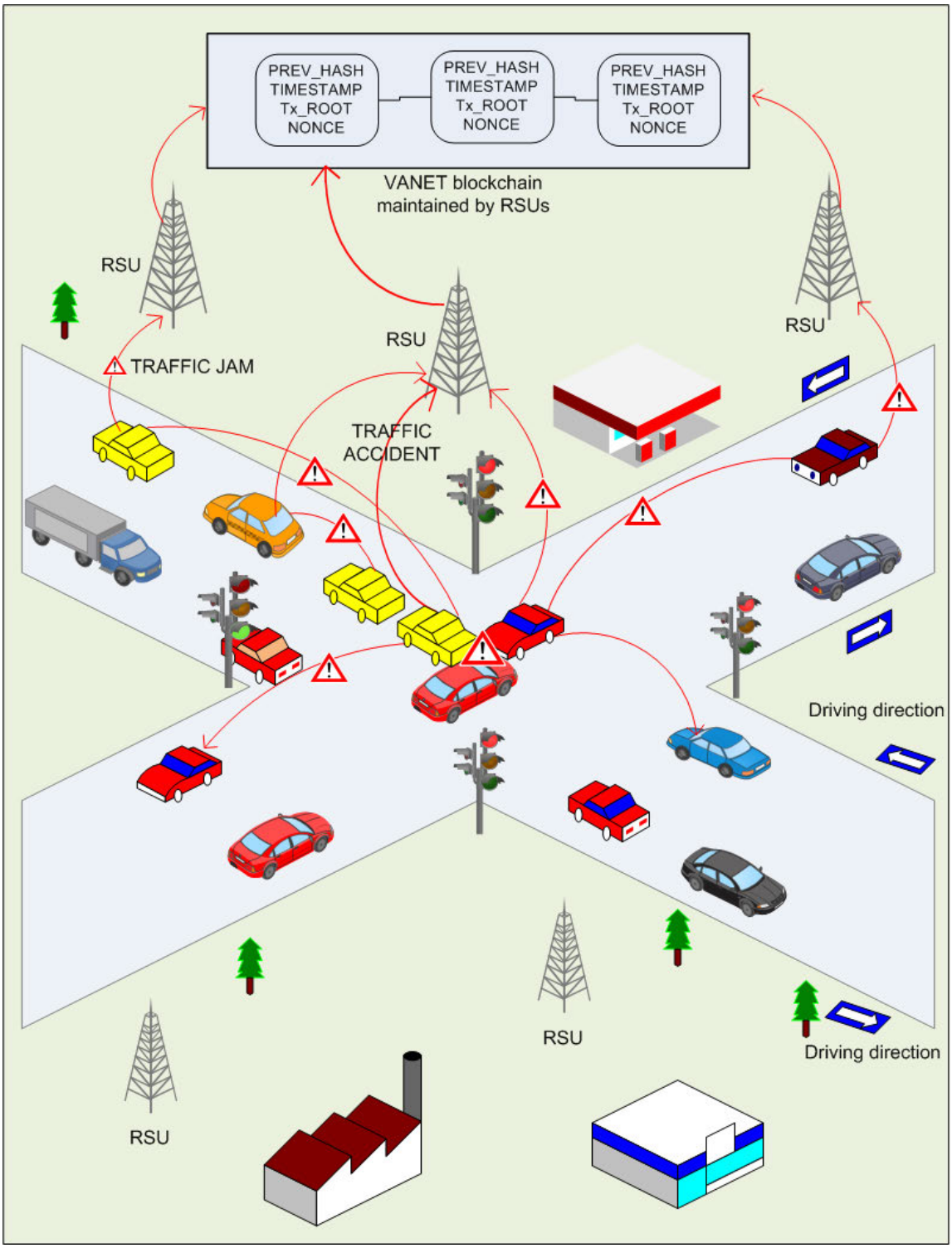

FIGURE 3. Application of blockchain for handling event messages in a VANET.

We assume that each vehicle that detects an event will transmit the information to neighboring nodes. If this information is relayed, it will also reach miner nodes. A miner node's responsibility is to determine whether the event message is true or false. In other words, a miner will calculate the trustworthiness of received messages based on the number of event reports for a particular event type. For this, they run the event message clustering algorithm to determine the number of event reports for distinct event types. A detailed explanation of the clustering algorithm is given in Section IV.

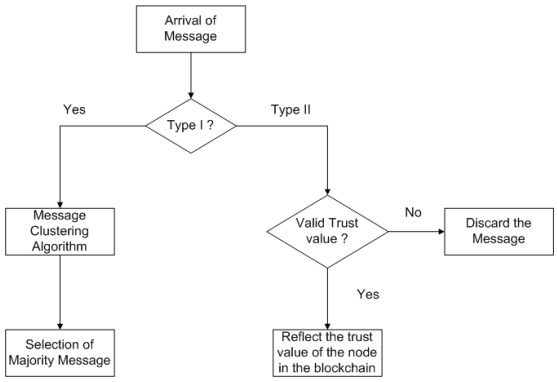

FIGURE 4. Processing of event messages at the miner node.

If an incident occurs on a road section, a large number of vehicles will report the incident to miners. Thus, the trustworthiness of a message can be inferred from voting, because there is a large number of event reports from vehicles around the event location. If there is a sufficient number of event reports, the miner will determine that the event has occurred and the transmitted message is legitimate. Malicious nodes can occasionally send false event reports. If a node receives two or more conflicting messages in the presence of a single event, this is referred to as the *conflicting message problem* in this paper.

In this case, the number of event reports can help miners distinguish between true and false information. When a malicious node tries to influence miners by sending false event messages, legitimate vehicular nodes will not send the same event report. As a result, with far fewer reports, miner nodes can distinguish between malicious and non-malicious vehicles. This majority voting scheme is preferable to recommendation-based trust management because malicious vehicles can launch a man-in-the-middle attack and provide false trust values for their neighbors in the latter scheme. It can also send false trust values in order to gain different road privileges.

The trust value of a vehicular node is calculated based on direct interactions between vehicular nodes and RSUs. The value is calculated as the ratio of true event reports from a vehicle to the total number of event reports from that vehicle in a given period of time [3], [29]. If x_i is the number of true messages from vehicle v_i, and y_i is the total number of messages it has generated up to a specific time, the trust value is calculated with (2).

$$Trust_i = \frac{x_i}{y_i} \tag{2}$$

Using this equation, a miner node can calculate the trust values of vehicular nodes and update the trust values in the blockchain for future reference. Most of the previous work calculated trust based on recommendations from neighboring vehicles and historical trust values stored in the blockchain [2], [3]. However, there is no mechanism for determining the validity of the event message itself. The primary goal of our work is to prevent registration of fake messages in the VANET blockchain. Storage of an event message and a trust value in the blockchain cannot ensure complete trustworthiness. As a result, a mechanism to validate event messages is required. The proposed clustering algorithm based on a majority vote scheme can be used to determine whether a vehicle's event message is true or false. The miner node can easily determine this by looking at clusters of similar messages based on event types. The processing of event messages at the miner node is summarized in Fig. 4. When the miner node receives messages from the neighbor nodes, it classifies the message into either Type I and Type II. When the received message is Type II message, if the trust value of the selected node is considered to be valid, the new trust value will be registered in the blockchain. When the received message is Type I message, the proposed clustering algorithm is applied and the majority message will be selected.

The issue of reflecting the voting result on the trustworthiness of a specific node or a message and design of a new blockchain, especially focusing on the newly added voting-decision role of miner nodes, will be investigated further in our future work, and we will focus on the issue of event message clustering in this paper.

IV. CLUSTERING ALGORITHM FOR VANET EVENT MESSAGES

This section explains the clustering algorithm for similarity analysis of event messages in a VANET blockchain. For clustering event messages based on event type, we referred to ETSI's Decentralized Environmental Notification Message (DENM) Basic Safety Messages [22], [30], [31]. From these sources, we categorized event messages into seven classes, along with their event IDs. Table 3 lists event messages possible in a VANET network with their IDs. From examination of these sources, we developed our own algorithm for event message clustering in the VANET environment. For each event that occurs, vehicles forward an event ID, vehicle positions, and a timestamp to neighboring vehicles and miner nodes. Using the proposed clustering algorithm, miner nodes can obtain the clustering results showing the number of event reports for each event type. Furthermore, miner nodes determine the trustworthiness of message reports based on the clustering results. If there is a large number of reports for an event, they assume the messages are true, and update vehicle's trust values. Otherwise, they will reject those messages, reducing vehicles' trustworthiness.

For two or more event messages to be similar, their distances should be sufficiently short; i.e., event reports should be similar to each other. In other words, vehicle positions, event IDs, and timestamps should be similar. We use Euclidean distance as a metric for calculating similarity between event messages.

A. THE VANET EVENT MESSAGE CLUSTERING ALGORITHM

In this section, we investigate a new clustering algorithm that is suitable for VANETs. Because VANET event messages

TABLE 3. DENM basic safety messages along with their classes [22], [31].

Event Class	Event ID	Event Message
Traffic Accident (TA)	TA_1	multi-vehicleAccident
	TA_2	heavyAccident
	TA_3	accidentInvolvingLorry
	TA_4	accidentInvolvingBus
	TA_5	accidentInvolvingHazardousMaterials
	TA_6	accidentOnOppositeLane
	TA_7	unsecuredAccident
	TA_8	assistanceRequested
Traffic Jam (TJ)	TJ_1	trafficJamAhead
	TJ_2	dangerousEndOfQueue
	TJ_3	suddenEndOfQueue
	TJ_4	queueOverHill
	TJ_5	queueAroundBend
	TJ_6	queueOnTunnel
Vehicle Warning (VW)	VW_1	stoppedVehicle
	VW_2	vehicleBreakdown
	VW_3	postCrash
	VW_4	publicTransportStop
	VW_5	carryingDangerousGoods
	VW_6	emergencyVehicleInOperation
	VW_7	stationarySafeguardingEmergencyVehicle
	VW_8	stationaryWreckingServiceWarning
	VW_9	prioritizedVehicleApproaching
Signal Violation (SV)	SV_1	stopSignViolation
	SV_2	trafficLightViolation
	SV_3	turningRegulationViolation
Adverse Weather Condition (AWC)	AWC_1	slipperyRoad
	AWC_2	precipitation
	AWC_3	fog
	AWC_4	tractionLoss
	AWC_5	fuelOnRoad
	AWC_6	iceOnRoad
	AWC_7	oilOnRoad
	AWC_8	heavyFrostOnRoad
	AWC_9	mudOnRoad
	AWC_{10}	blackIceOnRoad
	AWC_{11}	roadSalted
	AWC_{12}	strongWinds
	AWC_{13}	damagingHail
	AWC_{14}	Hurricane
	AWC_{15}	Thunderstorm
	AWC_{16}	Tornado
	AWC_{17}	Blizzard
	AWC_{18}	heavyRain
	AWC_{19}	heavySnowfall
	AWC_{20}	Smoke
	AWC_{21}	swarmOfInsects
Human Presence on the Road (HR)	HR_1	childOnRoadway
	HR_2	cyclistOnRoadway
	HR_3	motorcyclistOnRoadway
Wrong Way Driving (WD)	WD_1	wrongLane
	WD_2	wrongDirection

are generated sequentially over time, we need an algorithm that can cluster messages sequentially upon arrival, instead of processing a large number of messages collected over time. The disadvantage of the K-Means algorithm is that it requires the user to pre-determine and supply the number of clusters, i.e., the K-value. In some cases, determining the initial value of K is difficult. Because of the number of iterations and the distance calculation, computation time is

longer. Furthermore, it is computationally expensive for large datasets as the K-value becomes large. The disadvantages of the K-Means algorithm are listed below.

- K-Means is slow, and computation time scales poorly with large datasets.
- The user must pre-determine and supply the number of clusters, K.
- It can be difficult to choose a good initial center point for each cluster.
- It does not guarantee convergence to a global minimum. It is affected by initialization of centroids. Different setups may produce different outcomes.
- It has strong sensitivity to outliers.

Details on the proposed clustering algorithm is explained in the paragraph below.

The proposed event message clustering algorithm for VANETs is depicted in algorithm 1. This algorithm is distinct in that it does not require the number of clusters, i.e., the K-value, like the K-Means clustering algorithm. As the reports are generated, the algorithm will sequentially cluster different event messages based on event type. The algorithm is fed reports sequentially in ascending order of time. The algorithm's output is a list of clusters with their associated event type. The two event reports must occur on the same street in order to be considered for the same cluster. As seen from the algorithm, the first data point will be the centroid of the first cluster (steps 3-4). We establish the initial cluster boundary for each cluster, which is a constant value, δ (Step 5). The default value of the cluster boundary will be set at twice the length of a vehicle, and its optimal value for cluster formation will be determined later, based on experimentation. The cluster boundary is used to determine the similarity of the location from two event messages to avoid formation of multiple clusters for a single event. Whenever a new event report arrives, the algorithm first calculates its proximity to the nearby cluster centroid (i.e., means) using a Euclidean distance formula. If it is near a cluster's centroid, it will be added in the corresponding cluster. We then update the mean and boundary of the same cluster (steps 10-14). The new mean is calculated as the average of all reports in the cluster. The new cluster boundary will be the farthest distance point in the cluster plus cluster boundary constant δ. If the new event report is too far from the available clusters, the algorithm will create a new cluster with the new mean and cluster boundary as the initial settings (steps 15-20). We then push this new cluster to the $Cluster_{list}$ (Step 19). It is possible that a single event can trigger multiple clusters. In this case, a merging algorithm (Algorithm 2) will be invoked to combine two clusters that represent a single event type (steps 22-28). The detail steps of the merging algorithm are provided in algorithm 2. We have to set the conditions for merging two clusters. Two clusters will be merged if they intersect with each other. For two clusters to intersect, the sum of their radii should be less than the sum of their boundaries [32]. The cluster boundary inequality, as depicted by the formula in Step 24 of algorithm 1, represents this condition.

Algorithm 1 VANET Event Message Clustering Algorithm (VEMCA)

1: *Input*: reports sorted in increasing order of time, number of lanes L
2: *Output*: list of clusters with event types differentiated
3: Acquire the first data point, x_1
4: Make the first point (x_1) the first cluster with mean $m_1 = Co - ordinate(x_1)$
5: Set Initial Cluster Boundary, $B_1 = \delta$ //δ is a constant with the optimal value determined based on experiment
6: $Counter, k = 1$ // for updating cluster
7: $Cluster_{list} = [c_1, c_2, c_3, \ldots\ldots c_k]$ //c_i is the i-th cluster list
8: Set the counts $n_1, n_2, \ldots..n_k$ to zero // n_i counter for the i-th cluster
9: **for** each new data point x_i **do**
10: **if** x_i is closest to m_i **then**
11: Add x_i to same cluster
12: $n_i = n_i + 1$ //Update counter
13: $m_i = m_i + 1/n_i(x - m_i)$ //Update mean
14: $B_i = max(d(m_i, x_i)) + \delta$ //Update Boundary
15: **else**
16: $k = k + 1$ //Increment cluster counter
17: $m_i = Co - ordinate(x_i)$
18: $B_i = \delta$
19: Push k into $Cluster_{list}$
20: **end if**
21: **end for**
22: **for** each cluster k_i in $Cluster_{list}$ **do**
23: **for** each cluster k_j in $Cluster_{list}$ where $k_j \neq k_i$ **do**
24: **if** $d(m_i, m_j) \leq (B_i + B_j)$ **then**
25: $Merge(k_i, k_j)$
26: **end if**
27: **end for**
28: **end for**

The steps for merging two clusters is provided in algorithm 2. If we want to merge C_2 and C_1, we copy all reports from C_2 to C_1. Then, we delete cluster C_2 from the $Cluster_{list}$. We then update the mean and boundary of the merged cluster (steps 8-10). The algorithm will return merged cluster C_1 along with its centroid.

V. VANET SIMULATOR FOR EVENT MESSAGE CLUSTERING

The VANET event message clustering simulator was created using the Pygame module, which is used to create multimedia applications such as video games using the Python programming language. The simulation was the intersection of two eight-lane streets, each with two-way traffic. A snapshot of our simulator is shown in Fig. 5. The simulator can mimic two types of events: an accident or a traffic jam. To simulate an accident, mouse-click the car(s) in the road to cause the

Algorithm 2 Merge Two Clusters

Merge($C_1[N]$, $C_2[M]$)
for $i = 0; i < M; i++$ **do**
    $C_1[N + i] = C_2[i]$
end for
Delete ($C_2[]$)
Update mean and boundary of merged cluster
m = $Co - ordinates(C_1[0])$
for each data x in $C_1[]$ **do**
    $m = m + (1/(N + M)) * (x - m)$ //Update mean
    $B = max(d(m, x)) + \delta$ //Update Boundary
end for
return ($C_1[]$, m)

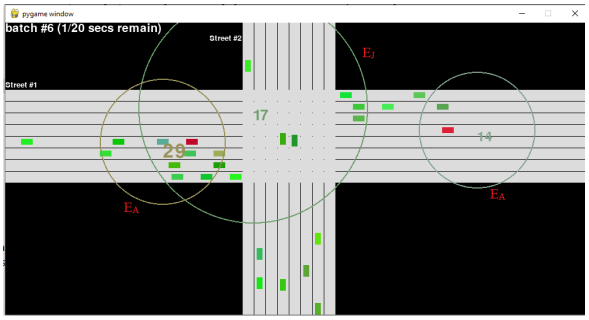

FIGURE 5. Graphical output of the simulator.

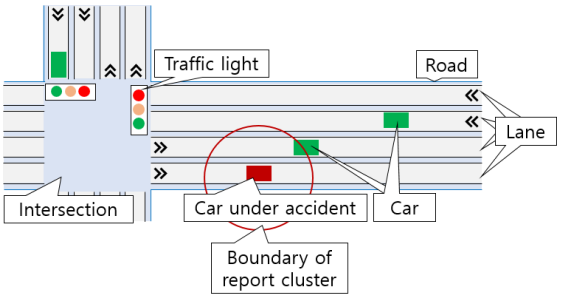

FIGURE 6. Overview of objects in traffic and clustering simulator.

event. Cars that have to stop on a regular basis, either due to congestion or an accident, will cause a traffic jam event.

The simulator will model car movement, report generation, and clustering. It creates five different types of object, (road, lane, car, report, and cluster), as shown in Fig. 6. A road object contains one or more lanes, and each lane contains cars. A car begins to move in its initially assigned lane but can change lanes to avoid traffic jams or accidents. Cars also generate reports based on specified conditions, and these reports are grouped according to the propsed clustering algorithm. In addition to the above five objects, traffic lights are created at each intersection of the roads, such that cars from different lanes pass the intersection without interrupting one another.

The simulation proceeds according to algorithm 3. It begins by creating road and lane objects as specified by

Algorithm 3 Event-Processing Loop of the Simulator

Create roads and lanes as specified by the user
Create traffic lights at intersections of the roads
Configure events {add_{car}, $move_{car}$, $change_{light}$} to fire at their specified intervals
Configure event $mouse_{click}$ to fire at user's mouse click
Configure event *exit* to fire when the window is closed
$running = True$
while *running* **do**
    for each event e **in** event queue **do**
        if e is add_{car} **then**
            choose a lane l at random
            add a new car to l
        else if e is $move_{car}$ **then**
            move each car (as per Algorithm 4)
        else if e is $change_{light}$ **then**
            switch traffic lights among {*green*, *amber*, *red*}
        else if e is $mouse_{click}$ **then**
            find the car c at mouse position
            toggle $c's$ state between {*accident*, *normal*}
        else if e is *exit* **then**
            $running = False$
        end if
    end for
end while

Algorithm 4 Moving Car Objects

procedure move_car(c) // c is a car object to move
    if c is under *accident* **then**
        stop and generate reports
    else if a red light is on in $c's$ lane **then**
        stop
    else if the car in front of c blocks the way **then**
        if a neighbor lane l has space **then**
            switch into l
        else
            reduce speed and generate reports
        end if
    else
        go forward at $c's$ full speed
    end if
end procedure

the user (steps 1–2). It also initiates all the events so they occur at regular intervals (steps 3–5). Based on these events, the simulator performs a set of operations on a regular basis, such as adding a new car (steps 9–11), moving existing cars (steps 12–13), and switching traffic lights (steps 14–15). It also creates an accident upon a mouse click on a car (steps 16–18), so that the car stops and generates reports.

Each car moves according to algorithm 4. When a new car is created, it is placed at the entrance of a randomly selected lane, and its speed is assigned from a predefined range. Then,

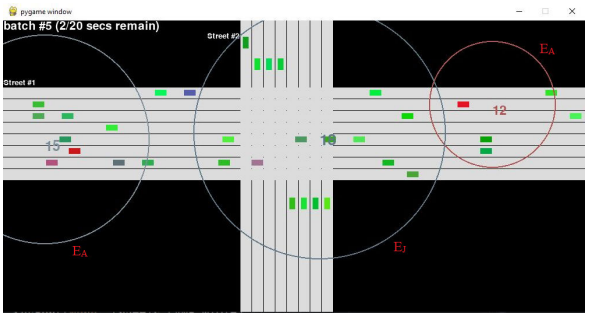

FIGURE 7. Clustering based on event types.

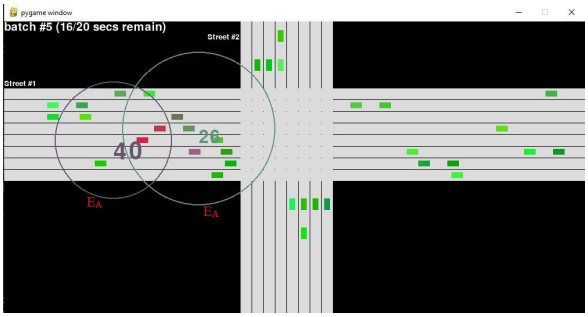

FIGURE 8. The need to merge two clusters.

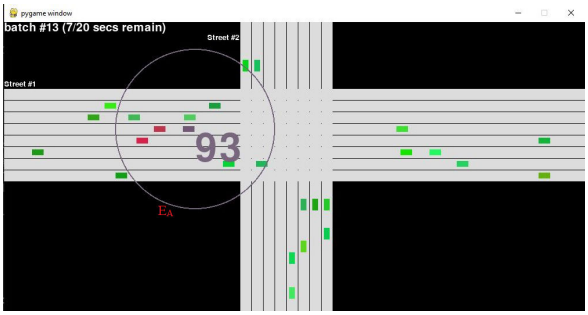

FIGURE 9. Two clusters merged.

the simulator moves the car on a regular basis. If the car is under an accident or if it hits a red light, it does not move (steps 2–5). If the car needs to slow down due to a traffic jam or an accident in front, it attempts to switch into a nearby lane with sufficient space, or it reduces speed and generates reports (steps 6–11). Otherwise, the car continues on the current lane with its full speed (steps 12–14). The reports are generated and clustered according to the rules described in Section IV-A. On a regular basis, the simulator removes existing cluster instances, and begins to create a new set of clusters. This renewal corresponds to the generation of a new block in the blockchain system.

At GitHub (https://github.com/sihyunglee26/Clustering-Simulation), we posted the code for the simulator. We also posted a demonstration of the clustering simulator for the three different scenarios described in subsections V-A, V-B, and V-C below.

The following subsections explain the workings of our simulator by considering several scenarios. We go over in detail how the clustering simulator distinguishes between two different event messages.

A. SCENARIO 1: CLUSTERING EVENT REPORTS BASED ON EVENT TYPE

We consider two types of events in this simulation: an accident (E_A) and a traffic jam (E_J). This scenario, looks at how event reports are grouped based on the type of event detected. Fig. 7 depicts two scenarios in which event reports are clustered according to two distinct event types. A red rectangle represents a car in an accident. The algorithm clusters the reports for this event and displays the total number of reports received from neighboring vehicles. The other clusters are formed by vehicles that are forced to stop due to a traffic jam or an accident. In this case, as shown in Fig. 7, a separate cluster is formed from a traffic jam. We can see from the simulation that the algorithm produces distinct types of clusters for each event. Clustering is done in an online manner as new reports arrive based on time.

B. SCENARIO 2: DISTINGUISHING EVENT CLUSTERS

The distinction between two or more clusters is made first by event type. Second, if events are of the same type, position information will serve as a criterion for cluster distinction, as shown in Fig. 7. There might be multiple accidents on the highway, and the algorithm will produce clustering results based on accident position. For example, as shown in Fig. 7, the simulator displays three distinct event clusters at different positions. On Street #1, two accidents have occured at different locations. The proposed algorithm creates two separate clusters. Although they are both accidents, they will never be the same event because they occurred in different places. There might be cases when it is hard to distinguish between two or more clusters that are formed due to the same type of event. We discuss this case in Scenario 3.

C. SCENARIO 3: THE NEED TO MERGE TWO CLUSTERS

A single accident can result in multiple clusters. In such a case, it is difficult to know if two adjacent clusters correspond to the same event or not, as illustrated in Fig. 8. Accidents can cause traffic jams, so we need a way to differentiate between these clusters. We developed an algorithm to merge two or more clusters if they represent the same event type. Two clusters that intersects each other will merge to form one cluster, as shown in Fig. 8 and Fig. 9. From this type of clustering based on event type, blockchain miners can easily determine the validity of the reports, and can maintain vehicle trustworthiness in a blockchain network. Thus, the proposed algorithm is important in order to create a trustworthy VANET environment.

VI. EXPERIMENT SETUP AND PERFORMANCE EVALUATION

This section describes the experiment environment and the evaluation results for our clustering algorithm. In addition, dataset generation and performance metrics are described.

	X	Y	TIME	EVENT_ID
0	258.0	244.0	1653544822	1
1	258.0	244.0	1653544822	1
2	258.0	244.0	1653544822	1
3	258.0	244.0	1653544822	1
4	258.0	244.0	1653544822	1
...	...	...	...	...
6234	872.0	184.0	1653544954	2
6235	536.0	100.0	1653544954	2
6236	496.0	414.0	1653544956	2
6237	208.0	244.0	1653544956	2
6238	436.0	25.0	1653544956	2

6239 rows × 4 columns

FIGURE 10. The description of the dataset.

A. SIMULATION PARAMETERS AND DATASET GENERATION

The proposed algorithm was evaluated using the VANET event message dataset generated by our simulator. Table 4 shows the simulation setup parameters for our experiment. The simulation lasted about three minutes, and a dataset consisting of 6239 event reports was generated. As shown in Fig. 10, each row contains the x-position, y-position, timestamp, and ID of the event. In our simulator, we limited the number of event types to two for simplicity. The primary goal of dataset generation is to compare clustering results from our proposed algorithms with other algorithms, such as K-Means [33], K-Medoids [34], Fuzzy C-Means (FC-Means) [35], the Gaussian mixture model (GMM) [36], spectral clustering [37], and density-based spectral clustering(DBSCAN) [38]. The dataset was preprocessed before passing it to the clustering algorithm. Data preprocessing refers to the steps required to transform or encode data so they can easily be parsed by the clustering application. Preliminary steps include filling in the missing values and removing duplicates from the dataset.

B. PERFORMANCE METRICS

The performance of the proposed algorithm was evaluated in terms of accuracy, precision, recall, f1-score, and computation time. Accuracy is defined as the ratio of correctly predicted observations to the total number of observations. It is calculated with (3).

$$accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{3}$$

TABLE 4. Simulation setup parameters.

Parameter	Value
Number of streets	2
Number of lanes	8 (two-way traffic)
Lane width	4 m
Number of events	2
Cluster boundary (δ)	10 m
Time to add car	200 m
Time to move car	80 m
Simulation time	3 min
Time to change traffic signal from red to green	5000 m
Amber signal time (yellow light)	1000 m
Velocity of vehicles	70 km/h

where TP, TN, FP, and FN represents True Positive, True Negative, False Positive, and False Negative, respectively. TP represents the predictions where the model correctly predicts the positive class. Similarly, TN represents the predictions where the model correctly predicts the negative class. FP is the model predictions where it incorrectly predicts the positive class. FN is the result of model predictions where it incorrectly predicts the negative class. Precision is defined as the proportion of correctly predicted positive observations to all positive predictions. It is calculated with (4). Recall represents the total number of correct positive predictions out of all the positive cases in the dataset. It is calculated with (5). The f1-score is the harmonic mean of precision and recall. It is calculated with (6).

$$precision = \frac{TP}{TP + FP} \tag{4}$$

$$recall = \frac{TP}{TP + FN} \tag{5}$$

$$f1 - score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{6}$$

Computation time is the amount of time required by the algorithm to cluster the given dataset. It is related to the algorithm's complexity.

C. PERFORMANCE EVALUATION

In this section, we examine the performance of our VANET event message clustering algorithm and compare it to state-of-the-art clustering algorithms such as K-Means, K-Medoids, FC-Means, the GMM, spectral clustering, and DBSCAN. The Python scikit-learn library used for this comparison is an open source, machine learning library that includes support vector machine (SVM), K-Means, random forest, and DBSCAN algorithms for classification, clustering, and regression problems [39]. The evaluation results in Table 5 show that our model outperformed state-of-the-art algorithms. The accuracy, precision, recall, and f1-score of our proposed algorithm were 91.28%, 90.41%, 91.29%, and 90.24%, respectively. Accuracy, precision, recall, and f1-score for the K-Means algorithm were 85.65%, 73.36%, 85.65%, and 79.03%, respectively. These results show the

TABLE 5. Performance result comparison between the proposed algorithm and state-of-the-art clustering algorithms.

ALGORITHM	ACCURACY (%)	PRECISION (%)	RECALL (%)	F1-SCORE (%)
K-Means	85.65	73.36	85.65	79.03
K-Medoids	85.65	73.36	85.65	79.03
FC-Means	85.65	73.36	85.65	79.03
GMM	85.65	73.36	85.65	79.03
Spectra clustering	87.27	88.91	87.27	82.64
DBSCAN	87.27	88.91	87.27	82.64
VEMCA (proposed)	**91.28**	**90.41**	**91.29**	**90.24**

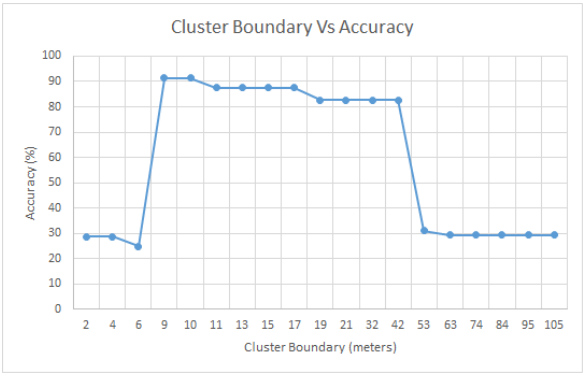

FIGURE 11. Impact of cluster boundary δ on algorithm accuracy.

efficacy of our algorithm and its usefulness in clustering VANET event messages.

Experiment results show that cluster boundaries have a significant effect on algorithm accuracy, as shown in Fig. 11. If the cluster boundary is too small, there will be too few event reports, resulting in poor cluster formation and reduced accuracy. On the other hand, if the cluster boundary is too high, two clusters that belong to different event types might be merged, reducing the algorithm's accuracy. Experiment results show that a cluster boundary in the range of 9-10 m is optimal for cluster formation. Thus, as a rule of thumb, we find that twice the vehicle length can be a good candidate for the value of cluster boundary δ. We assume the average vehicle length is about 5 m.

D. COMPLEXITY ANALYSIS

We ran our algorithm and the K-Means algorithm in order to measure the execution times for several datasets of varying sizes. We implemented both algorithms in C++ for the analysis. Fig. 12 compares the execution times (in milliseconds) of these algorithms based on varying data sizes. Table 6 provides the execution times for the K-Means and the proposed algorithms. As shown in Fig. 12, the execution time of the K-Means algorithm for a dataset with one million rows was 12,270 m. In contrast, the proposed VEMCA took just 370 m to cluster a dataset of one million records. These results demonstrate that the proposed algorithm is faster, regardless of the size of the dataset.

TABLE 6. Execution times of the K-Means and the proposed algorithms for various datasets.

Data Size	K-Means (ms)	VEMCA (ms)
1000 rows	10	3
10,000 rows	140	6
100,000 rows	1050	30
1,000,000 rows	12270	370

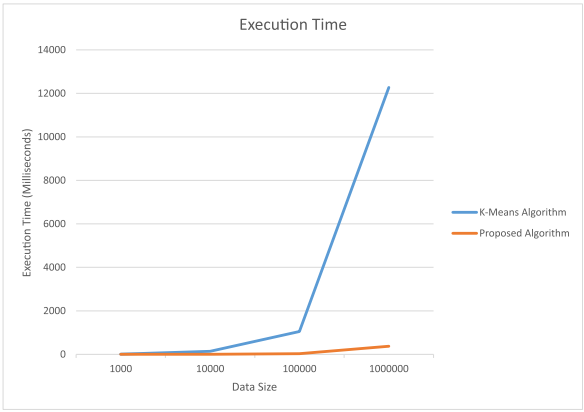

FIGURE 12. Execution time comparison of the K-Means and the proposed algorithms.

It is well known that the K-Means algorithm has $O(n^2)$ time complexity, where n is the size of the input data [40]. Due to its quadratic complexity, the algorithm may not be efficient for big and critical applications where time is limited. Our algorithm greatly shortened the execution time by avoiding the initial cluster size problem of the K-Means algorithm by using a sequential approach. Fig. 12 shows that, in comparison to the K-Means method, the number of operations increased slowly as input increased. Thus, we expect our algorithm will be more efficient, compared to the K-Means algorithm, for a VANET, where the number of vehicles can be very large.

VII. CONCLUSION AND FUTURE WORKS

This paper proposes the VANET event message clustering algorithm for VANETs. Blockchain miners can use this algorithm to determine the validity of event messages

and maintain trustworthiness scores for vehicles in the blockchain. VEMCA is unique and does not require guessing the initial number of clusters. It can perform clustering as new event reports arrive. Furthermore, we developed a new simulator to model event message generation in the presence of a traffic event, including accidents, and we evaluated the proposed event message clustering algorithm through simulations based on our simulator. Our proposed algorithm was evaluated in terms of accuracy, precision, recall, f1-score, and computation time. Results show that our algorithm outperformed various clustering algorithms, such as K-Means, K-Medoids, FC-Means, the GMM, spectral clustering, and DBSCAN.

We will investigate how to determine the trust level of each node and that of each message based on VANET blockchain and our clustering algorithm in our future work.

REFERENCES

[1] R. Shrestha, S. Y. Nam, R. Bajracharya, and S. Kim, "Evolution of V2X communication and integration of blockchain for security enhancements," *Electronics*, vol. 9, no. 9, p. 1338, Aug. 2020.
[2] R. Hussain, J. Lee, and S. Zeadally, "Trust in VANET: A survey of current solutions and future research opportunities," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 5, pp. 2553–2571, May 2021.
[3] R. Shrestha and S. Y. Nam, "Trustworthy event-information dissemination in vehicular ad hoc networks," *Mobile Inf. Syst.*, vol. 2017, pp. 1–16, Nov. 2017.
[4] Z. Lu, Q. Wang, G. Qu, and Z. Liu, "BARS: A blockchain-based anonymous reputation system for trust management in VANETs," in *Proc. 17th IEEE Int. Conf. Trust, Secur. Privacy Comput. Commun./12th IEEE Int. Conf. Big Data Sci. Eng. (TrustCom/BigDataSE)*, Aug. 2018, pp. 98–103.
[5] Z. Lu, W. Liu, Q. Wang, G. Qu, and Z. Liu, "A privacy-preserving trust model based on blockchain for VANETs," *IEEE Access*, vol. 6, pp. 45655–45664, 2018.
[6] Z. Yang, K. Zheng, K. Yang, and V. C. M. Leung, "A blockchain-based reputation system for data credibility assessment in vehicular networks," in *Proc. IEEE 28th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun. (PIMRC)*, Oct. 2017, pp. 1–5.
[7] Z. Yang, K. Yang, L. Lei, K. Zheng, and V. C. M. Leung, "Blockchain-based decentralized trust management in vehicular networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1495–1505, Apr. 2019.
[8] N. Khatri, R. Shrestha, and S. Y. Nam, "Security issues with in-vehicle networks, and enhanced countermeasures based on blockchain," *Electronics*, vol. 10, no. 8, p. 893, Apr. 2021.
[9] C. A. Kerrache, N. Lagraa, C. T. Calafate, J. C. Cano, and P. Manzoni, "T-VNets: A novel trust architecture for vehicular networks using the standardized messaging services of ETSI ITS," *Comput. Commun.*, vol. 93, pp. 68–83, Nov. 2016.
[10] Z. Liu, J. Ma, Z. Jiang, H. Zhu, and Y. Miao, "LSOT: A lightweight self-organized trust model in VANETs," *Mobile Inf. Syst.*, vol. 2016, pp. 1–15, Dec. 2016.
[11] C. A. Kerrache, C. T. Calafate, N. Lagraa, J.-C. Cano, and P. Manzoni, "Hierarchical adaptive trust establishment solution for vehicular networks," in *Proc. IEEE 27th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun. (PIMRC)*, Sep. 2016, pp. 1–6.
[12] T. Biswas, A. Sanzgiri, and S. Upadhyaya, "Building long term trust in vehicular networks," in *Proc. IEEE 83rd Veh. Technol. Conf. (VTC Spring)*, May 2016, pp. 1–5.
[13] M. Mukhtaruzzaman and M. Atiquzzaman, "Clustering in vehicular ad hoc network: Algorithms and challenges," *Comput. Electr. Eng.*, vol. 88, Dec. 2020, Art. no. 106851.
[14] N. Taherkhani and S. Pierre, "Centralized and localized data congestion control strategy for vehicular ad hoc networks using a machine learning clustering algorithm," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 11, pp. 3275–3285, Nov. 2016.
[15] I. Hussain and C. Bingcai, "Cluster formation and cluster head selection approach for vehicle ad-hoc network (VANETs) using K-means and Floyd-Warshall technique," *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 12, pp. 11–15, 2017.
[16] A. Kchaou, R. Abassi, and S. G. El Fatmi, "Towards a secured clustering mechanism for messages exchange in VANET," in *Proc. 32nd Int. Conf. Adv. Inf. Netw. Appl. Workshops (WAINA)*, May 2018, pp. 88–93.
[17] S. P. Ardakani, C. F. Kwong, P. Kar, Q. Liu, and L. Li, "CNN: A cluster-based named data routing for vehicular networks," *IEEE Access*, vol. 9, pp. 159036–159047, 2021.
[18] K. A. Hafeez, L. Zhao, Z. Liao, and B. N.-W. Ma, "A fuzzy-logic-based cluster head selection algorithm in VANETs," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2012, pp. 203–207.
[19] K. A. Hafeez, L. Zhao, J. W. Mark, X. Shen, and Z. Niu, "Distributed multichannel and mobility-aware cluster-based MAC protocol for vehicular ad hoc networks," *IEEE Trans. Veh. Technol.*, vol. 62, no. 8, pp. 3886–3902, Oct. 2013.
[20] Q. Li, A. Malip, K. M. Martin, S.-L. Ng, and J. Zhang, "A reputation-based announcement scheme for VANETs," *IEEE Trans. Veh. Technol.*, vol. 61, no. 9, pp. 4095–4108, Nov. 2012.
[21] C. Lai, K. Zhang, N. Cheng, H. Li, and X. Shen, "SIRC: A secure incentive scheme for reliable cooperative downloading in highway VANETs," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 6, pp. 1559–1574, Jun. 2017.
[22] Y.-T. Yang, L.-D. Chou, C.-W. Tseng, F.-H. Tseng, and C.-C. Liu, "Blockchain-based traffic event validation and trust verification for VANETs," *IEEE Access*, vol. 7, pp. 30868–30877, 2019.
[23] X. Liu, H. Huang, F. Xiao, and Z. Ma, "A blockchain-based trust management with conditional privacy-preserving announcement scheme for VANETs," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4101–4112, May 2020.
[24] S. Kudva, S. Badsha, S. Sengupta, H. La, I. Khalil, and M. Atiquzzaman, "A scalable blockchain based trust management in VANET routing protocol," *J. Parallel Distrib. Comput.*, vol. 152, pp. 144–156, Jun. 2021.
[25] L. Xie, Y. Ding, H. Yang, and X. Wang, "Blockchain-based secure and trustworthy Internet of Things in SDN-enabled 5G-VANETs," *IEEE Access*, vol. 7, pp. 56656–56666, 2019.
[26] R. Shrestha, R. Bajracharya, A. P. Shrestha, and S. Y. Nam, "A new type of blockchain for secure message exchange in VANET," *Digit. Commun. Netw.*, vol. 6, no. 2, pp. 177–186, May 2020.
[27] R. Shrestha, R. Bajracharya, and S. Y. Nam, "Blockchain-based message dissemination in VANET," in *Proc. IEEE 3rd Int. Conf. Comput., Commun. Secur. (ICCCS)*, Oct. 2018, pp. 161–166.
[28] C. Cooper, D. Franklin, M. Ros, F. Safaei, and M. Abolhasan, "A comparative survey of VANET clustering techniques," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 1, pp. 657–681, 1st Quart., 2017.
[29] W. Ahmed, W. Di, and D. Mukathe, "Privacy-preserving blockchain-based authentication and trust management in VANETs," *IET Netw.*, vol. 11, nos. 3–4, pp. 89–111, May 2022.
[30] J. Santa, F. Pereniguez, A. Moragon, and A. F. Skarmeta, "Vehicle-to-infrastructure messaging proposal based on CAM/DENM specifications," in *Proc. IFIP Wireless Days (WD)*, Nov. 2013, pp. 1–7.
[31] *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Application; Part 3: Specifications of Decentralized Environmental Notification Basic Service*, Standard ETSI 302 637–3, 2014.
[32] E. W. Weisstein. (2003). *Circle-Circle Intersection*. [Online]. Available: https://mathworld.wolfram.com/
[33] K. Kandali, L. Bennis, and H. Bennis, "A new hybrid routing protocol using a modified K-means clustering algorithm and continuous Hopfield network for VANET," *IEEE Access*, vol. 9, pp. 47169–47183, 2021.
[34] S. A. Abbas, A. Aslam, A. U. Rehman, W. A. Abbasi, S. Arif, and S. Z. H. Kazmi, "K-means and K-medoids: Cluster analysis on birth data collected in city Muzaffarabad, Kashmir," *IEEE Access*, vol. 8, pp. 151847–151855, 2020.
[35] P. K. Mishro, S. Agrawal, R. Panda, and A. Abraham, "A novel type-2 fuzzy C-means clustering for brain MR image segmentation," *IEEE Trans. Cybern.*, vol. 51, no. 8, pp. 3901–3912, Aug. 2021.
[36] Y. Li, J. Zhang, Z. Ma, and Y. Zhang, "Clustering analysis in the wireless propagation channel with a variational Gaussian mixture model," *IEEE Trans. Big Data*, vol. 6, no. 2, pp. 223–232, Jun. 2020.
[37] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel K-means: Spectral clustering and normalized cuts," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2004, pp. 551–556.
[38] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowl. Discovery Data Mining*, Aug. 1996, vol. 96, no. 34, pp. 226–231.

[39] O. Kramer, "Scikit-learn," in *Machine Learning for Evolution Strategies*. Berlin, Germany: Springer, 2016, pp. 45–53.
[40] M. K. Pakhira, "A linear time-complexity K-means algorithm using cluster shifting," in *Proc. Int. Conf. Comput. Intell. Commun. Netw.*, Nov. 2014, pp. 1047–1051.

NARAYAN KHATRI received the B.E. degree from Purbanchal University, Nepal, in 2013, and the M.S. degree in computer engineering from Yeungnam University, South Korea, in 2017, where he is currently pursuing the Ph.D. degree in information and communication engineering. He is also working as a Research Assistant with the Computer Network and Security Laboratory. His research interests include network security, blockchain, vehicular ad-hoc networks, and machine learning for network intrusion detection systems.

SIHYUNG LEE received the B.S. (summa cum laude) and M.S. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), South Korea, and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University (CMU). Then, he worked as a Postdoctoral Researcher at the IBM Thomas J. Watson Research Center. He is currently a Professor with the School of Computer Science and Engineering, Kyungpook National University. His research interests include pattern mining from social network traffic and program synthesis for classical and quantum computers.

ABDUL MATEEN received the Ph.D. degree from the Department of Information and Communication Engineering, Yeungnam University, South Korea, and the Ph.D. degree in computer science from the International Islamic University, Islamabad, Pakistan. He is currently an Assistant Professor with the Federal Urdu University of Arts, Science and Technology, Islamabad. Previously, he worked with the COMSATS Institute of Information Technology, Pakistan Space and Upper Atmosphere Research Commission (SUPARCO), and Pakistan Telecommunication Company Ltd. (PTCL). He has teaching and industry experience, spanning more than 20 years. His main research interests include blockchain, data mining, autonomous computing, and agent-oriented software engineering.

SEUNG YEOB NAM (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 1997, 1999, and 2004, respectively. From 2004 to 2006, he was a Postdoctoral Research Fellow with the CyLab, Carnegie Mellon University. From 2006 to 2007, he was a Postdoctoral Researcher with the Department of Electrical Engineering and Computer Science, KAIST. In 2007, he joined the Department of Information and Communication Engineering, Yeungnam University, Gyeongsan, South Korea, where he is currently a Professor. His research interests include network security, blockchain, network management, and wireless networks.