UNIVERSITY OF SCIENCE AND TECHNOLOGY OF HANOI

UNDERGRADUATE SCHOOL

Research and Development

BACHELOR THESIS

By

Doan Lien Huong

Information and Communication Technology

Implementation of Cluster Establishing algorithm to improve delay in Vehicular Networks

---

Supervisor:

Dr. Nguyen Minh Huong

ICT Lab - USTH

Hanoi, October 2020

Acknowledgements

I would like to express my deepest appreciation to my supervisor, Dr. Nguyen Minh Huong, who is the most caring, thoughtful and patient professor I have known in my entire life. Throughout the internship period, she always inspired me with her profound insights and motivation. I sincerely appreciate her valuable comments and advice on my project during each progress meeting.

I would also like to extend my sincere thanks to all my professors in the ICT Department for their conscientiousness and all the invaluable knowledge they conveyed to me during the 3 years I studied at USTH.

I am extremely grateful to have so many sincere, supportive friends. I can not thank my friend Ngoc Mai enough. She is always willing to help me and answer any of my questions. I believe she is the most diligent, modest and admirable student. I also want to thank all my friends Le Hiep, Khanh Huyen, Bach, Do Ngoc, Quang Huy, and my seniors Do Hoang, Quoc Viet for all the encouragement and happiness they brought to me over the last 3 years.

Last but not least, I would like to express my gratitude to my family members who always support me unconditionally.

List of Acronyms

USTH	**U**niversity of **S**cience and **T**echnology of **H**anoi
DSRC	**D**edicated **S**hort **R**ange **C**ommunication
WAVE	**W**ireless **A**ccess in **V**ehicular **E**nvironment
VANET	**V**ehicular **A**d-hoc **NET**work
MANET	**M**obile **A**d-hoc **NET**work
QoS	**Q**uality **o**f **S**ervice
RSU	**R**oad **S**ide **U**nit
CH	**C**luster **H**ead
CM	**C**luster **M**ember
PDR	**P**acket **D**elivery **R**atio
CCH	**C**ontrol **CH**annel
SCH	**S**ervice **CH**annel
V2V	**V**ehicle-to-**V**ehicle
V2I	**V**ehicle-to-**I**nfrastructure

List of Figures

1 The time cycle diagram 7
2 Design of Classes 12
3 Design of Headers 16
4 Delay Index and PDR in case: 1 node broadcasts 1 packet 20
5 Delay Index and PDR in case: 1 node broadcasts 10 packets 21
6 Delay Index and PDR in case: 2 nodes broadcast 1 packet 22
7 Delay Index and PDR in case: 2 nodes broadcast 10 packets 23

List of Tables

1 Simulation parameters 19

Table of Contents

1 Introduction **1**
1.1 Context and Motivation 1
1.2 Objectives 3
1.3 Thesis structure 3

2 State of the art **4**

3 Methodologies and Tools **6**
3.1 Methodologies 6
3.1.1 System overview and assumptions 6
3.1.2 BEACON_EXCHANGE process 8
3.1.3 CLUSTER_FORMATION process 8
3.1.4 DATA_EXCHANGE process 10
3.2 Tools 11

4 Implementation **12**
4.1 Design of classes 12
4.2 Design of headers 15

5 Performance Evaluation **18**
5.1 Scenario Descriptions 18
5.2 Simulation Setups 18
5.3 Results 20
5.4 Discussions 24

6 Conclusion and Future Work **25**
6.1 Conclusion 25
6.2 Future Work 25

Appendix A T_{wait} calculation **26**

Appendix B Delay Index calculation **27**

Appendix C PDR calculation **28**

Bibliography **29**

Chapter 1

Introduction

1.1 Context and Motivation

Road accidents, traffic congestions and air pollution due to a large number of vehicles have become serious global issues. Traffic incidents are persistent problems in every country, which results in a high fatality rate and poverty. Hence, it is arising the demand for communication among vehicles on the road so that they can respond quickly to hazardous situations. In addition, wireless technology is developing rapidly with time, which leads to the introduction of a new type of networks that can handle this issue, which is called vehicular ad-hoc networks (VANETs).

VANET is a vital part of the Intelligent Transportation System (ITS), which aims to improve road safety and information transmission efficiency on the road. VANET is a form of MANET (Mobile Ad-hoc Network), in which nodes are moving vehicles. In VANET, vehicles can communicate with surrounding vehicles (V2V) via On-board units (OBUs), and communicate with Road Side Unit (V2I) as well.

There are two main types of application in VANET: safety application and non-safety application. The first aims to avoid the risk of accidents and make safer driving by providing information about incidents and hazards on the road to the vehicle occupants. The safety concerns are also the main objectives of VANET in real life.

Dedicated Short-Range Communication (DSRC) was developed to support VANET [1]. DSRC operates in the 5.9 GHz band on a total bandwidth of 75 MHz (from 5.850 GHz to 5.925 GHz), which is divided into 7 channels of 10 MHz. There are 1 control channel (CCH), which is to transmit network management messages, and six service channels (SCH) which are for data transmission for different services. WAVE which is short for Wireless Access in Vehicular Environments, a vehicular communication system, is used to operate in the DSRC band to support data exchange.

Transmitting messages to notify other vehicles immediately when an incident happens is crucial in safety applications. Hence, low latency is one of the most important requirements. However, it is challenging to provide an efficient safety messages scheme due to high mobility and dynamic topology of VANET. Furthermore, message broadcasting in this network often leads to communication failure, overhead and congested channels. One solution is establishing a hierarchical clustering structure in the network. This system helps reduce overhead by cutting down the data volumes exchanged. The main idea is to group the similar nodes in the same cluster and there is a CH node (Cluster Head) which is in charge of managing and controlling the communication of other nodes (Cluster members - CM) in the cluster.

In this work, we present a clustering protocol which aims to minimize the delay and ensure the high connectivity in the VANET at the same time. My supervisor is the person who came up with the idea about building this clustering protocol and I am the one who implemented it in the ns-3 simulator and evaluated performance for safety application.

1.2 Objectives

To implement the algorithm and verify its effectiveness in improving delay in Vehicle Ad-hoc Network. Specific goals are:

- Implementing the protocol using ns3 simulator
- Designing simulation scenarios for safety application
- Investigating the protocol in terms of improving the connectivity of the network while optimizing the delay.

1.3 Thesis structure

This thesis is organized as follows: Section 2 contains the literature review about the clustering techniques proposed for VANET. Section 3 explains clearly the methods used in this protocol and introduces the materials used to implement the algorithm. In section 4, we introduce the implementation details. Section 5 contains the evaluation of the protocol we proposed. The last section concludes our work and gives suggestions for further research.

Chapter 2

State of the art

Safety applications are dependent on exchanging safety messages. Such messages are delay-sensitive to ensure that vehicles in the network can take proper action on time. Nevertheless, data message dissemination in VANET can cause communication failure and congested channels due to large data volumes exchanged. To increase the emergency packet coverage over the network, blind rebroadcasts have been applied [2]. This method helps a packet have more chances to reach the destination node through being rebroadcast multiple times, thus increasing connectivity in the network. However, this method also comes with a trade-off that is the high end-to-end delay time. To overcome this, clustering is a solution to facilitate resource reuse and improve system capacity.

Various clustering algorithms have been proposed in the literature.

One of them is a speed difference based clustering technique [3]. The main objective is to enhance the stability of the network by categorizing the neighbour nodes as stable and unstable depending on their velocity and direction. Only nodes which are stable neighbours can participate in the cluster formation process.

Taking the QoS (Quality of Service) into consideration, the 2-layer clustering method is proposed in [4] to retain the stability of the vehicular network while obtaining the QoS demands. These 2 layers are a combination of static and dynamic clustering. In the first layer, CH is the fixed RSU, while in the second layer, the more suitable vehicle becomes CH. Vehicles can communicate with RSU through

dynamic CHs. The QoS metric is defined by appending quality of the link to the RSU, bandwidth and neighbourhood degree.

The authors of [5] presented a cluster-based location routing (CBLR), which mainly focus on enhancing routing efficiency in vehicular networks. In CBLR, nodes exchange their states through HELLO messages. A node relies on whether it receives that message within a limited amount of time or not to change its state to CH or CM of a cluster. Due to the dynamic topology of VANET, nodes in this system store their neighbour information inside a table so that they know their destination position to forward the packets directly.

Another clustering protocol is [6], which aims to form stable clusters and maintain stability during communications and link failures while satisfying the Quality of Service requirements. First, the protocol elects a set of optimal cluster-heads. Next, the elected cluster-heads select a set of optimal Multi Points Relay (MPR) nodes in charge of transmitting the packets and connecting the clusters according to a cheating prevention mechanism. Finally, the MPR recovery algorithm is presented to cope with link failures by choosing alternative MPRs.

Our clustering protocol aims to support safety application. Hence, the node with the most number of neighbours and closest to the RSU (in case RSU available) is elected to be the Cluster Head. CH is in charge of rebroadcasting the safety messages to other nodes.

In this section, we have presented several clustering techniques in the literature and also introduced our protocol. In the next section, we will describe our clustering method, which has the main goal is to reduce delay and enhance the connectivity of the network.

Chapter 3

Methodologies and Tools

3.1 Methodologies

3.1.1 System overview and assumptions

As we mentioned before, the main criteria to choose the CH in our algorithm is depending on the number of neighbor nodes. Firstly, to monitor the list of neighbor of each node, BEACON_EXCHANGE process is created to be the first process to happen. In VANET, nodes are treated equally for data delivery. That is the reason we need to elect a centre node (we call it Cluster Head) to manage data transmitting between nodes in the network. CLUSTER_FORMATION process is built for that reason. In the DATA_EXCHANGE process, our network now has the hierarchical structure due to the cluster formation in the previous process. Thus, data packets will be transmitted from the CH to other nodes in a cluster.

In our system, vehicle nodes operate in cycles. A cycle consists of 3 above processes. Each process is described as the following:

- BEACON_EXCHANGE process: At the start of a cycle, nodes are assigned as UN (unknown state). In this process, nodes exchange their mobility information (beacons) periodically.
- CLUSTER_FORMATION process: the CH election and formation of clusters happen in this process.

- DATA_EXCHANGE process: nodes exchange data packets in this process by broadcasting and rebroadcasting by CH.

Each process happens within a limited amount of time. When all the 3 processes are finished, we refresh all the cluster information and a new cycle begins. Figure 1 illustrates the cycle processing in our system.

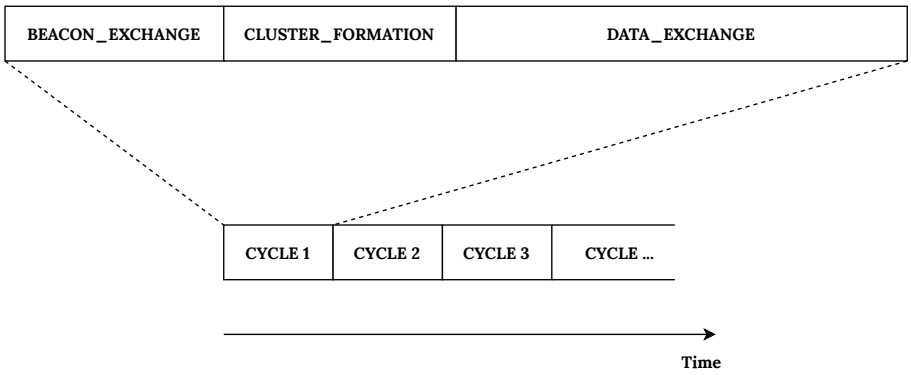

Figure 1: Processes in a cycle in the clustering protocol

The time period for each process should be chosen properly and carefully in order to ensure the quality of packet transmission. In our system, we set the largest duration time for the DATA_EXCHANGE process as the main goal of our protocol is to improve data exchange in the network. The detailed description for each process is presented in the next parts of this thesis.

There are a few assumptions that we use to build this algorithm:

- All vehicles in the network have the same communication range R
- vehicles move with a minor speed different
- simulations using the IEEE 802.11p standard

3.1.2 BEACON_EXCHANGE process

In this process, vehicle nodes in the network exchange their own mobility information by broadcasting beacon messages. A beacon message contains:

- The identification number of that vehicle
- The timestamp recorded when a beacon packet is sent
- Position vector of that vehicle
- Velocity vector of that vehicle

When node i receives a beacon message from node j ($beacon_j$), it will extract the information from that beacon and check whether the distance between it and node j $d(i,j)$ within the communication range R or not. If satisfied, node i appends all the received information of node j to its neighbour list N_i.

Algorithm 1: Beacon exchange process

```
1 Node i receives beacon_j
2 if d(i,j) < R then
3 |   N_i.append(j)
4 end
```

3.1.3 CLUSTER_FORMATION process

This process happens right after the BEACON_EXCHANGE process. Firstly, from the neighbour list N_i obtained in the first process, node i calculates the waiting time T_{wait} (Appendix A). That time period is the maximum time for node i to wait to receive the Form Cluster message from another node (node j). If the waiting time is up, node i changes its state $state_i$ from UN (unknown) to CH and sends Cluster Formation message to announce other nodes that it becomes CH and calls for CMs. Else, if node i receives Cluster Formation Message from node j, it stops waiting and changes its state to CM and its cluster ID (CH_i) to j.

T_{wait} is calculated according to this formula:

$$T_{wait} = (1 - connectivity\_index).T_{waitmax}$$

where

- $connectivity\_index = \frac{centrality\_index + distance\_index}{2}$
- $centrality\_index = \frac{sizeof(N_i)}{total\_nodes - 1}$
- $distance\_index = \frac{R - d_{RSU}}{R}$
- $T_{waitmax}$: the maximum waiting time. In this simulation, we set $T_{waitmax}$ = 1s.

We consider the scenarios without the RSU so we assume $distance\_index = 0$. Therefore, $connectivity\_index = centrality\_index$.

The Cluster Formation message contains the following information:

- The ID of the cluster which the sender belongs to. In this case, cluster ID is equal to the ID of CH.
- The identification number of the sender node (CH).
- The timestamp recorded when a packet is sent.

Algorithm 2: Cluster formation process

```
1  Calculate T_wait
2  while T_wait > 0 do
3      if i receives FormClusterMsg from j then
4          state_i ← CM
5          CH_i ←j
6      else
7          Decrement T_wait
8      end
9  end
10 state_i ← CH
11 CH_i ← i
12 SendFormationClusterMsg()
```

3.1.4 DATA_EXCHANGE process

This process happens after the cluster is formed in the second process. In this process, vehicle nodes start broadcasting safety messages (we call them data packets in this case). In this work, we only focus on intra-communication in the cluster, which is about exchanging data inside a cluster. When this process begins, a node *i* broadcasts safety messages in the form of data packets to other nodes. If the receiver is the CH, it will rebroadcast that packets to other nodes.

A data message consists of the following information:

- The timestamp recorded when a packet is sent. In this process, the time stamp plays an important role to calculate the delay time.
- The ID number of the source node
- The ID of the packet.

Algorithm 3: Data exchange process for a CH node

1 *CH receives a data packet (p_i) from source node i*

2 *broadcast p_i*

3.2 Tools

We use the ns-3 simulator to implement the proposed algorithm and run the simulation. ns-3 is a discrete-event network simulator and has been developed to provide an open, extensible network simulation platform, for networking research and education [7]. Several models of how packet data networks work and perform are provided in ns-3. Users can conduct simulation experiments from writing programs in C++ or Python. In ns-3, nodes are the basic computing device abstraction which represents computers in real life. We can install devices, internet stacks, application to our nodes using several Helper classes. Using ns-3, we can create multiple communication types such as PointToPoint, Wireless, CSMA.

In this project, we worked on the version 3.31 - the latest version of the ns-3 simulator which maintains an implementation for WAVE (Wireless Access in Vehicular Network). We used several built-in functions from WAVE module to implement the proposed protocol.

According to the methodologies proposed and introduced tools in this section, we implement a program for this clustering protocol. Our detailed implementation is described in the next section.

Chapter 4

Implementation

4.1 Design of classes

Figure 2 lists out the most important attributes and methods in our implementation.

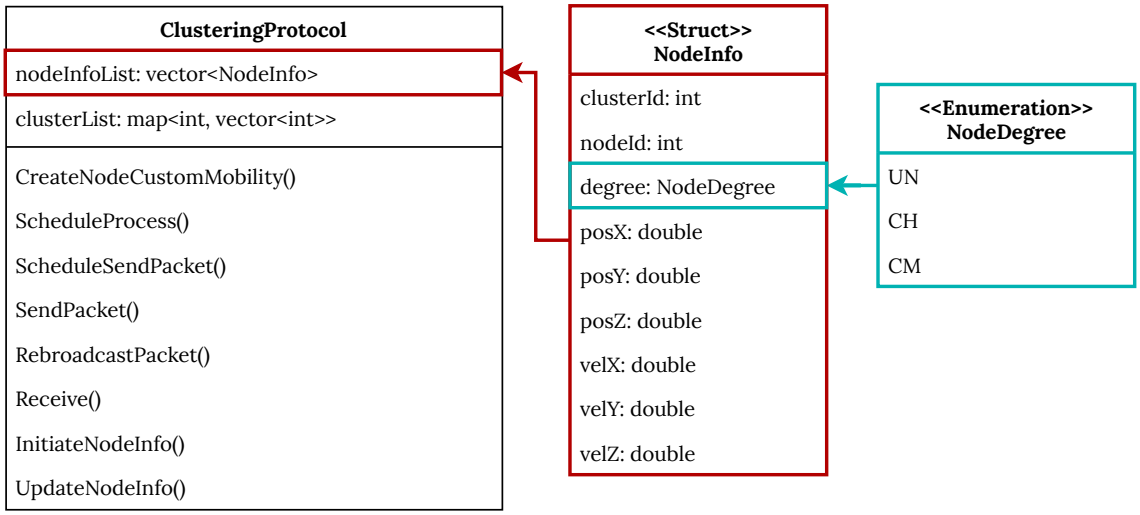

Figure 2: Design of Classes

We defined the main class ClusteringProtocol which contains several important attributes and methods we used in simulation scenarios. Two important attributes are nodeInfoList and clusterList as shown in Figure 2.

- nodeInfoList contains all the information of nodes in the network including ID, state, position and velocity. We created a structure object NodeInfo to

contain these node information and an enumeration NodeDegree to contain the state of a node (UN, CH or CM)

- clusterList is a list of cluster IDs and their cluster member ID.

There are some main functions of our clustering protocol shown in Figure 2.

- CreateNodeCustomMobility() which is responsible for creating vehicle nodes, installing the Mobility Model into nodes. This method also contains the InitiateNodeInfo() method.
- ScheduleProcess() takes the process number as an input (eg: in this implementation, we take 1, 2, 3 as presenting 3 processes: BEACON_EXCHANGE, CLUSTER_FORMATION and DATA_EXCHANGE respectively). This method is in charge of determining the starting time of each process and then calls the SendPacket() function.
- SendPacket() depends on the process to send appropriate packets. It adds the exact header information according to the process to the packet and broadcasts it. This function can call the ScheduleSendPacket() function for a node to send other packets after a fixed time. Take the BEACON_EXCHANGE process as an example, in our simulation we set nodes to broadcast a beacon message every 0.1 seconds by calling the ScheduleSendPacket() function.
- ScheduleSendPacket() schedules the send event in SendPacket() method after a period delay.
- Receive() defines the different behaviour of nodes after receiving packets. It extracts the TypeID of the packet from the metadata to obtain the Header name so that a node can acknowledge which type of packet is received. Depend on packet type, the receiver node will have different actions:
  - Beacon packet: receiver will store sender information into its neighbour list.
  - Cluster formation packet: if the receiver state is UN, it changes its state to CM. Else, no actions happen.

  - Data packet: if the receiver is a CH, it will rebroadcast that packet. In this case, the method RebroadcastPacket() is called. Else, the receiver node only prints out the packet information.

- RebroadcastPacket() is called in the DATA_EXCHANGE process when a CH receives a data packet and needs to rebroadcast it. It keeps the original header of the packet and broadcasts it again. Thus, the timestamp of that packet is unchanged and we can use it to calculate the Delay Index later.

- InitiateNodeInfo() initiates node information when the algorithm begins. For more details, all the node information, including state, position and velocity, are stored in a list called nodeInfoList. When the simulation starts, node states are assigned as UN, their position and velocity are obtained through the Mobility Model.

- UpdateNodeInfo() which is called whenever a node changes its state. There are two times, this function is called: one is in the Receive() function when a node receives the cluster formation message of the other node and changes its state to CM; the other one is from SendPacket() function, it happens when the waiting time of a node run out, that node will set its state as CH and send formation message to others.

In addition, there are also some functions which are important to our algorithm and support when we evaluate the performance of this protocol.

- CalculateWaitingTime() is called when the CLUSTER_FORMATION process starts. The method ScheduleProcess() then knows when the node will send the Cluster Formation packet; hence, it schedules the precise time the sender sends packets based on the T_{wait} of that node. This calculation function takes the node id as the input and returns the waiting time corresponding to that node.

- GetTotalAverageDelayIndex() which calculates the *Average Delay Index* for the whole network. (The detail of calculation is shown in Appendix B)

- GetTotalPDR() which calculates the *PDR* for the whole network. (The detail of calculation is shown in Appendix C)

4.2 Design of headers

Figure 3 presents the design for each header.

In the clustering protocol, as packets are sent in 3 different processes for different purposes, they need to be embedded with different kinds of information. That is the reason we implemented new header classes by extending the built-in Header class in ns-3. Each overrides the original methods and provides necessary setter and getter for its data as well. 5 main methods each header overrides are:

- Serialize() stores a header into the byte buffer of a packet
- Deserialize() re-creates a header from the byte buffer of a packet
- GetTypeId() gets the type ID of the header
- Print() prints the content of a header as ascii data to a c++ output stream.
- GetSerializedSize() returns the number of bytes which are needed to store the full header data by Serialize().

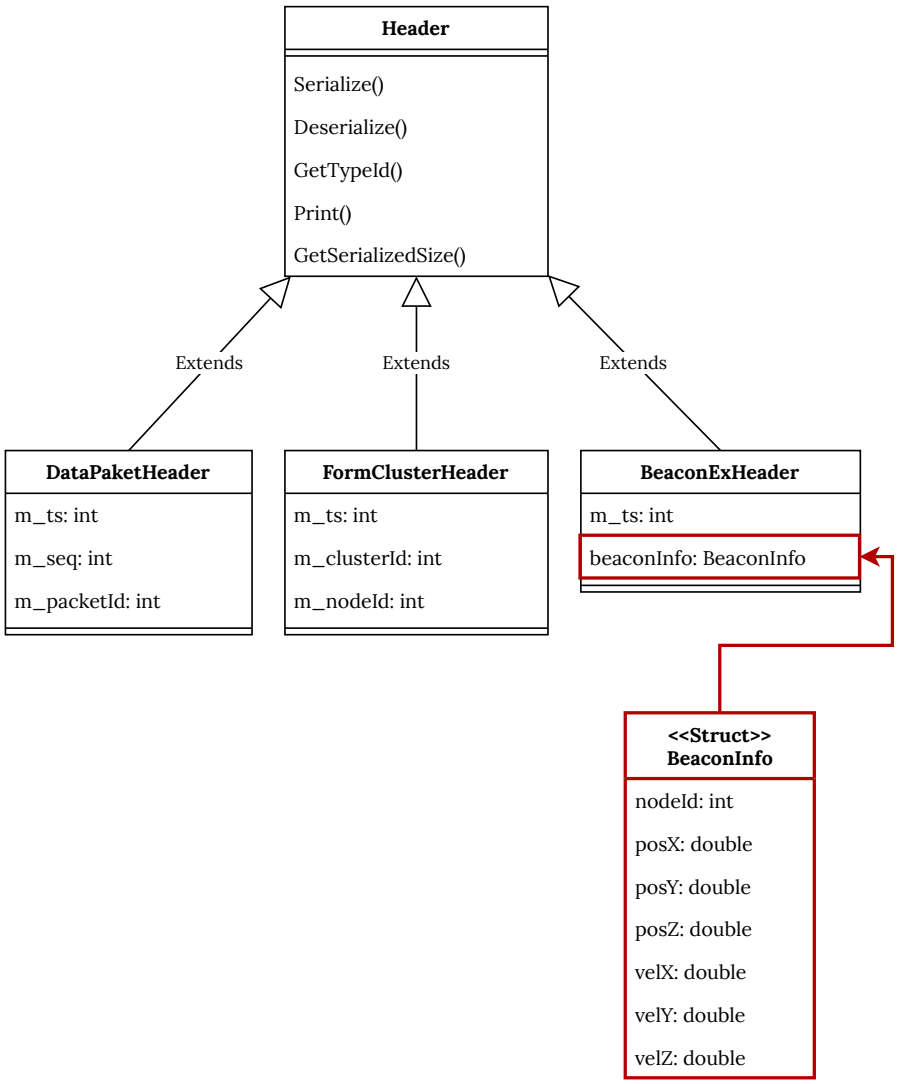

Figure 3: Design of Headers

3 types of header class are:

- BeaconExHeader: for packet broadcast in the BEACON_EXCHANGE process. Each node embeds its mobility information (position, velocity) in the packet to broadcast to other nodes. We created a struct object BeaconInfo to contain all those information.
- FormClusterHeader: for packet broadcast in the CLUSTER_FORMATION process. Those packets will be broadcast by the CH to notify other nodes that it is a CH and the receiver nodes will be its CMs (Cluster member). It contains:

  - ts: the timestamp recorded when a packet is sent.
  - Cluster ID: the identification number of the cluster which CH belongs to. Cluster ID is equal to the ID of CH
  - Node ID: the identification number of CH.

- DataPacketHeader: for packet broadcast in the DATA_EXCHANGE process. Vehicles broadcast safety messages in the form of data packets to other vehicles. A data packet contains:
  - ts: The timestamp recorded when a data packet is sent. This attribute is crucial when calculating the delay time of a packet.
  - seq: contains the ID of the source node.
  - Packet ID: the ID of a packet. This attribute is helpful when calculating the Delay Index in case a node broadcasts multiple packets. Because we have to calculate the Delay Index of the sender node by taking the average of the delay time of packets which that node sent.

In this section, we presented our implementation for this protocol by explaining our design for classes and packet headers. From this implementation, we run the simulations to obtain the results and evaluate our protocol performance, which is described in the next section.

Chapter 5

Performance Evaluation

This section presents the evaluation of our proposed clustering protocol in improving the delay in VANET.

5.1 Scenario Descriptions

For supporting safety application, it is essential to manage the packet coverage and packet delivery time in the network when a hazardous situation happens. Therefore, we consider the scenario in which a node or more than a node broadcasts safety messages to other nodes in the network. We focus on analyzing the number of receiver nodes and delay time in four simulation cases which will be described carefully in the following section.

5.2 Simulation Setups

In the simulation, we monitor a maximum of 50 vehicles on a 3-lane one-direction highway of 100 m length and 2 m width each lane. Vehicles are uniformly distributed in lanes with predefined velocities.

The common configuration of some basic parameter used in our simulation is shown in Table 1:

Table 1: Simulation parameters

Parameters	**Value**
Communication range	100m
Number of lanes	3
Lane distance	2m
Number of vehicles	10/20/30/40/50
Simulation time	40s

We conducted 4 different simulation cases which are 1 node broadcasts 1 packet, 1 node broadcasts 10 packets, 2 nodes broadcast 1 packet and 2 nodes broadcast 10 packets. In each case, we run 5 experiments at different times ($t_0 = 10.06s, 10.2s, 10.31s, 10.55s, 10.7s$).

To evaluate the performance, we use the “Delay Index" metric which measures the mean of delay time in V2V communications and PDR (Packet Delivery Ratio) metric which is the successful arrival rate of packets.

In the case of blind-rebroadcast, we implemented each receiver node will rebroadcast received packet after 0.05s from the time it received the packet. Each node will rebroadcast a packet with a maximum of 3 times.

In the case of the clustering algorithm, in the DATA_EXCHANGE process, we schedule 1 node to broadcast one or more than 1 packet to other nodes in the network. Then we calculate the Delay Index corresponding to each sender node. The average Delay Index of the whole network is calculated by taking the average of the Delay Index of individual sender nodes (please refer to Appendix B for more detailed calculation). The same process happens for calculating PDR (please refer to Appendix C for more detailed calculation). Then, we present a comparison among the proposed algorithm with blind rebroadcast and non-protocol scheme.

5.3 Results

Simulation case 1: 1 node broadcasts 1 packet

In this simulation, we simulated 1 node in the network to broadcast 1 packet to other nodes. Figure 4 shows the results of this simulation.

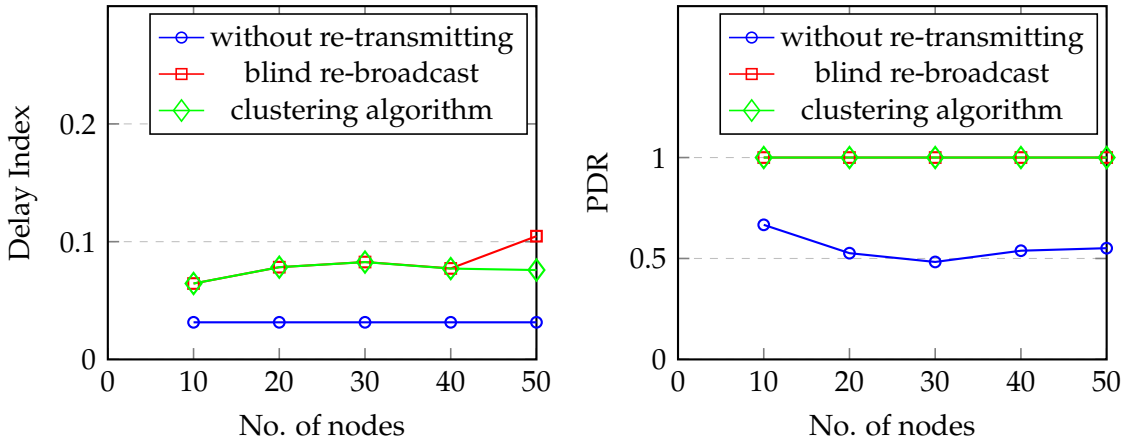

Figure 4: Delay Index and PDR in case: 1 node broadcasts 1 packet

As we can see in Figure 4, the delay indexes measured in the blind rebroadcast and clustering modes are much higher than the one without re-transmitting mode. This is the result of packet rebroadcasting. The delay time when using clustering mode is clearly lower than blind rebroadcast in the more than 40-node network. As in our clustering algorithm, only CH is in charge of rebroadcasting packets, fewer packet collisions occur than in blind rebroadcast (in which any node receive packets can rebroadcast them).

In the PDR graph, both blind rebroadcast and clustering modes reach the highest and also the maximum PDR 1. This result is quite difficult to happen in real life. However, in our simulation, we conducted the simulation in 30 seconds which is not so long and the relative velocity between nodes is small. Furthermore, we assume that losses of packets are only from packet collisions, we did not consider the transmission medium factors. It is possible for the PDR to obtain 1 due to the high stability in the network topology in our simulation.

Simulation case 2: 1 node broadcasts 10 packets

In this simulation, we increased the number of packets that a node broadcasts to 10 with the time interval between sending each packet is 0.1 seconds. Figure 5 presents the results of this simulation.

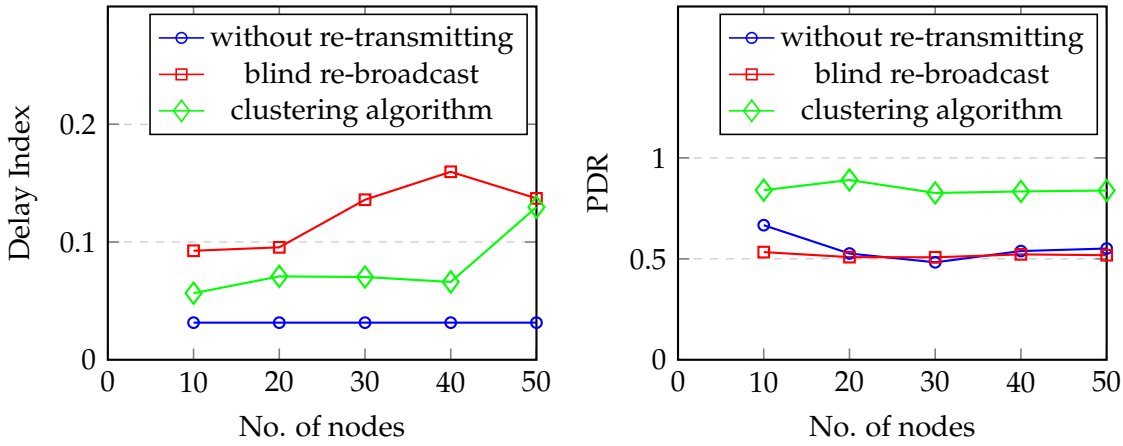

Figure 5: Delay Index and PDR in case: 1 node broadcasts 10 packets

It is clearly to be seen from Figure 5 that using blind rebroadcast technique leads to the highest delay time due to packet collisions. Packet numbers in the network now are much more than the first case. Furthermore, on the same channel, at the same time, there is a limited amount of packets can be transmitted. The more nodes in the network, the more packets wait to be sent. Especially for the blind rebroadcast, any receiver nodes can resend packets, which causes resource scarcity and packet losses. This is shown in the PDR graph. The number of packets received is even lower than the non-re-transmitting mode.

In contrast, clustering algorithm now shows great effectiveness with a lower end-to-end delay and highest packet coverage over the network.

Simulation case 3: 2 nodes broadcast 1 packet

For better observing the effect of our proposed protocol, we set up the simulation with 2 nodes are the senders and each of them broadcast 1 packet. The results are illustrated in Figure 6.

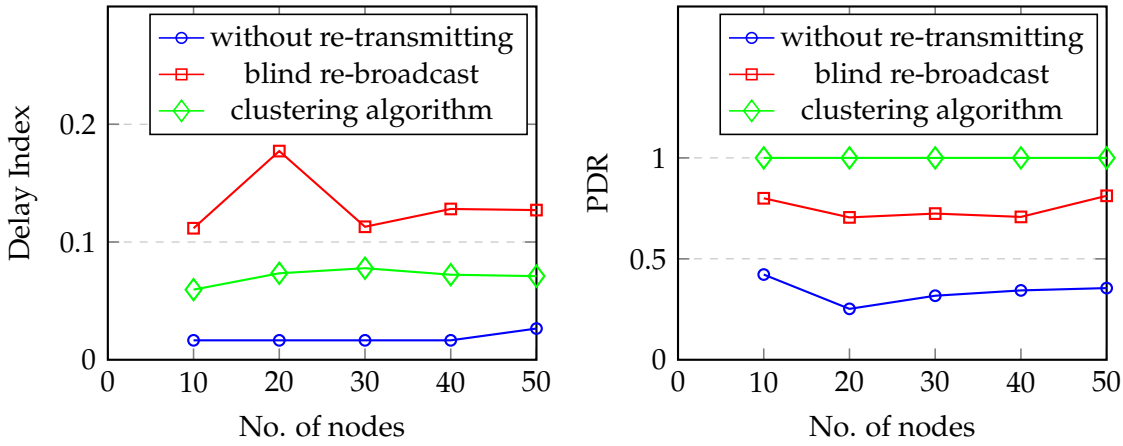

Figure 6: Delay Index and PDR in case: 2 nodes broadcast 1 packet

These results strengthen our previous inferences. Clustering technique shows the best performance with the highest PDR and low latency due to fewer rebroadcasting and fewer packet losses. As we mentioned in the first simulation case, obtaining the maximum PDR as shown in the graph is not possible in practical scenario due to many transmission medium factors which we did not take into account in this work.

On the other hand, a high delay index when the network in the blind rebroadcast mode is the result of packet collisions and rebroadcasting packets multiple times. However, as we can see from the PDR graph, this method still helps more nodes to receive packets when we compare to non-re-transmitting mode.

Simulation case 4: 2 nodes broadcast 10 packets

In the last simulation case, we simulated 2 nodes to broadcast in the network. Each node now broadcasts 10 packets with the time interval between sending each packet is 0.1 seconds. Figure 7 shows the results for this simulation case.

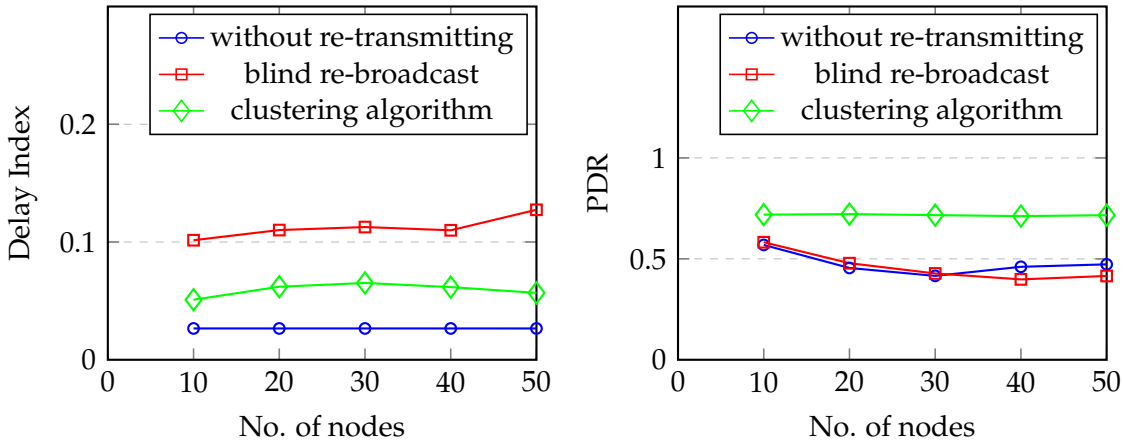

Figure 7: Delay Index and PDR in case: 2 nodes broadcast 10 packets

In this simulation case, because of the increase in the packet numbers lead to more packet collisions, we can see the number of received packets is smaller than other simulation cases in all of three transmit modes.

Nevertheless, it is clear that our protocol still brings the highest PDR among 3 modes. And blind rebroadcast in this case does not make any effect on the PDR performance on account of packet drops and packet collisions.

In addition, delay of packet delivery in blind rebroadcast mode is highest. This proves the ineffectiveness of this technique when we exchange more packets in the network.

5.4 Discussions

In brief, by analyzing the PDR and Delay Index in 4 simulation cases, we can evaluate the performance of our proposed protocol. The simulation results indicate that the blind broadcast technique is not efficient and leads to scarce resource. The more nodes in the network, the more rebroadcast, the more packets wait to be sent. That results in either packet drops or delay increase.

In contrast, our proposed clustering algorithm presents better performance in comparison to the previous technique. It reduces the delay and increases the packet delivery rate as well.

Our simulations have several limitations. They only focus on single-hop communication. However, in practical scenarios, packets are transmitted through multiple nodes or multiple hops. We should take that into account to develop this algorithm in the future.

We also need to consider the scenario which contains multiple clusters. That leads to the increment in the demand of communication among clusters, between Cluster Heads, which is the inter-communication. In our algorithm for now focuses only on the intra-communication.

To obtain more accuracy in the clustering protocol, several criteria can be considered to choose a proper Cluster Head. For example, modifying the T_{wait} formula by adding the RSU into the network with the aim to deliver packets across clusters. Hence, the T_{wait} depends on the distance from that node to the RSU instead depends only on the neighbor numbers of that node.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

To conclude, we proposed a protocol for clustering establishment in vehicular network with the view of optimizing the delay while increasing the connectivity for safety application. We implemented the protocol in the ns-3 simulator and run the simulation for 4 simulation cases. After conducting the simulations and analysing the results, we observe that our proposed clustering algorithm helps to reduce the delay in comparison to another method which is the blind rebroadcast while maintaining the high packet coverage over the network. Therefore, this algorithm not only responds to the low-latency demand but also assures the efficient warning message dissemination in VANET to support safety application.

6.2 Future Work

Our implementation has several limitations. For better respond to the practical scenarios, we can upgrade the protocol by:

- Including channel parameters such as bit error rate to make the protocol more precise and realistic
- Developing algorithm for inter-communication
- Simulating the scenarios with multiple clusters
- Modifying the calculation formula to get more accurate results.

Appendix A

T_{wait} calculation

Given a vehicle node i which has the neighbor list N_i , distance to closest RSU d_{RSU} in a network contains n nodes and communication range R, the waiting time of node in election process is calculated as the following:

$$T_{wait} = (1 - connectivity\_index).T_{waitmax}$$

where

- $centrality\_index = \frac{sizeof(N_i)}{n-1}$
- $distance\_index = \frac{R-d_{RSU}}{R}$
- $connectivity\_index = \frac{centrality\_index+distance\_index}{2}$
- $T_{waitmax}$: the maximum waiting time. In this simulation, we set $T_{waitmax}$ = 1s.

We consider the scenario without the RSU so we assume $distance\_index = 0$. Therefore, $connectivity\_index = centrality\_index$.

Appendix B

Delay Index calculation

Consider the packet broadcasting in the network.

Assume there are more than 1 node broadcasting more than 1 packets in the network.

Assume the total delay time of a packet when it is received by other nodes in the network is called D.

Delay Index of a packet: $DelayIdxPacket = avg(D)$

Hence, Delay Index of packets sent by a node: $DelayIdxPackets = sum(DelayIdxPacket)$

Delay Index of a sender node: $DelayIdxNode = avg(DelayIdxPackets)$.

(If that node only broadcasts 1 packet: $DelayIdxNode = DelayIdxPacket$)

Average Delay Index of the whole network: $AvgDelayIdx = avg(DelayIdxNodes)$ where $DelayIdxNodes = sum(DelayIndexNode)$.

(If there is only 1 sender node: $AvvgDelayIdx = DelayIdxNode$)

Appendix C

PDR calculation

Consider the packet broadcasting in the network.

Given the network contains n nodes.

Nodes broadcast more than 1 packet to other nodes.

Assume the total nodes receive a packet is called N.

PDR of a packet: $PDRPacket = \frac{N}{n-1}$

Hence, PDR of packets sent by a node: $PDRPackets = sum(PDRPacket)$

Then we can calculate PDR of the sender node:

PDR of a sender node: $PDRNode = avg(PDRPackets)$

(If that node broadcasts only 1 packet: $PDRNode = PDRPacket$)

Therefore, average Delay PDR of the whole network: $AvgPDR = avg(PDRNodes)$ with $PDRNodes = sum(PDRNode)$

(If there is only 1 sender node: $AvgPDR = PDRNode$)

Bibliography

[1] M. Hadded et al. “TDMA-Based MAC Protocols for Vehicular Ad Hoc Networks: A Survey, Qualitative Analysis, and Open Research Issues”. In: *IEEE Communications Surveys Tutorials* 17 (2015), pp. 2461–2492.

[2] Ghazal Farrokhi and Saadan Zokaei. “Improving safety message dissemination in IEEE 802.11e based VANETs using direction oriented controlled repetition technique”. In: *2014 IEEE 21st Symposium on Communications and Vehicular Technology in the Benelux (SCVT)* (2014), pp. 100–104.

[3] Zaydoun Y. Rawashdeh and Syed Mahmud. “A novel algorithm to form stable clusters in vehicular ad hoc networks on highways”. In: *EURASIP Journal on Wireless Communications and Networking* 2012 (2012), pp. 1–13.

[4] H. Arkian et al. “A Stable Clustering Scheme Based on Adaptive Multiple Metric in Vehicular Ad-hoc Networks”. In: *J. Inf. Sci. Eng.* 31 (2015), pp. 361–386.

[5] RA Santos, RM Edwards, and NL Seed. “Inter vehicular data exchange between fast moving road traffic using ad-hoc cluster based location algorithm and 802.11b direct sequence spread spectrum radio”. In: *Post-Graduate Networking Conference* (2003).

[6] O. Wahab, Hadi Otrok, and A. Mourad. “VANET QoS-OLSR: QoS-based clustering protocol for Vehicular Ad hoc Networks”. In: *Comput. Commun.* 36 (2013), pp. 1422–1435.

[7] ns 3 project. “ns-3 TutorialRelease ns-3-dev”. In: (2020).