A Proposal to Analyse the Advantage of Disseminating Data on Beacons in VANETs

C. Blommendaal
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands

c.blommendaal@student.utwente.nl

ABSTRACT

Vehicular Ad Hoc Networks (VANETs) are an extreme form of mobile ad-hoc networks (MANETs) and can be considered as one of the enabling technologies for Intelligent Transportation Systems (ITS). With the use of VANETs efficiency applications such as cooperative adaptive cruise control and safety applications such as an emergency braking system are designed. These applications use data provided by surrounding cars to create a Cooperative Awareness (CA). This CA is created by updates periodically sent by every car containing status information and details regarding throttle and break. This process is called beaconing. However due to the broad scale of applications capacity problems raise ahead. In this paper we focus on designing an experiment in which less time critical information is piggybacked onto scheduled beacons. This seems to be a way to relieve the amount of stress on the medium.

Keywords

VANETs, vehicular networks, piggybacking, beaconing, data dissemination

1. INTRODUCTION

Since traffic jams cost Europe €50 billion per year and 40,000 people die every year in traffic accidents [5] the time has come to make traffic more efficient and safe. To achieve these goals Intelligent Transportation Systems (ITS) can be used. By putting technology in a car that has awareness of it’s surroundings the car will be able to respond faster than a human will ever can.

These intelligent transportation systems are designed to make transportation more efficient, more reliable and safer. To achieve these goals ITS rely heavily on communication between vehicles. Together these vehicles form so-called Vehicular Ad Hoc Networks (VANETs), a special kind of Mobile Ad Hoc Networks (MANETs).

The network topology of VANETs differs from more familiar networks such as Local Area Networks (LAN) and Wireless Local Area Networks (WLAN) in terms of mobility and coordination. Within VANETs nodes are highly mobile and there is no Access Point with the authority to grant access, assign IP-addresses and control bandwidth [8]. This open structure allows the nodes to generate the Cooperative Awareness needed by the traffic applications.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Copyright 2012, University of Twente, Faculty of Electrical Engineering, Mathematics and Computer Science.

Looking at the communication requirements of these traffic applications we can distinguish two kinds of applications: the ones that are relying on the rapid transmission of data and the more delay tolerant kinds. This first type is usually used in safety applications such as automatic braking or sending a distress signal to all the surrounding cars. This second type of applications operates on a larger geographical scale and has as goal to make sure that the flow of cars trough the highway is as efficient as possible. Due to the informational characteristics of those applications the time between the moment of sending and receiving is becoming less relevant as long as the information actually reaches its destination within a given timeframe.

Within a vehicular network all the nodes within the range of the sender will try to forward the received message as far as possible into the direction of the destination of the packet. Different strategies exist for this so called forwarding; these are discussed in section 4.

Beaconing is a communication pattern in which every car regularly sends an update to all surrounding vehicles. These updates usually contain information about the status, such as the speed of the vehicle and the amount of throttle and brake used at the moment. Surrounding cars can use this information for safety applications such as an emergency braking system.

When it comes to forwarding this information piggybacking comes in play. Piggybacking is a form of multi-hop forwarding in which nodes piggyback application data on top of their beacons, i.e., they transmit the data together with the already scheduled beacons. By having multiple nodes repeat the process, the data can be routed to a specific location (georouting).

Combing all of the above brings us to the following research question.

"In what kind of experiment can piggybacking be compared with other forwarding protocols?"

Hereby arising question would be: what other forwarding strategies exist? And how can the effect of piggybacking be measured in an experimental set-up?

Most likely will piggybacking application data onto beacons lead to a lower busy time on the medium in comparison to the separate dissemination of data. In this last scenario the node has to access the medium an extra time to send the application data, doubling the chance of not being able to get on the medium and waiting another back-off period. By the use of piggybacking all the data (the application data plus the beacon data) is send once the node has access to the medium. Furthermore this method saves on the overhead of packets headers. This saving will be discussed in section 4.

The goal of this paper is to design an experiment in which the impact on the busy time of the network can be measured when piggybacking application data onto scheduled beacons,
compared to the situation in which application data is forwarded independently of the beaconing process.

The outline of this paper is as follows. In section two related work will be discussed and the actual problem is stated in section three. In the subsequent section different forwarding protocols are shown and in section five there is a comparison between these protocols. Section six covers the experimental set-up and finally a conclusion is being drawn and future work is addressed in section seven.

2. RELATED WORK

In this section we will discuss some of the related work in the vehicular networking sector. Vehicular networking is based on the 802.11p Wireless Access in Vehicular Environments (WAVE) standard [22], which is an enhancement of the more familiar 802.11, or better known as the Wi-Fi, standard.

Lots and lots of methods have been developed to optimize applications using the 802.11p protocol (i.e. Adaptive Cruise Control [2]). Disturbing factors such as high relative speed between the sender and receiver can be countered by the proper usage of sending priorities [1].

Furthermore there is lot of discussion about security in 802.11p applications. In what way can information be exchanged without being compromised by address spoofing and man-in-the-middle attacks? After the 802.11-2007 revision it is fair to say that the 802.11 protocol is pretty safe when the TKIP and the CCMP protocols are being used. [8].

Despite all the success stories [2], [6] beaconing is still a quite new field of science. In crowded situations it’s not always performing as desired [10]. Because the intention of VANETs is to provide additional safety many say that there should be a separate channel for safety messages only. This to reduce the waiting times involved when sending (emergency) messages.

As said earlier, almost all the VANET applications require some kind of CA to work properly. Creating this CA for each single node puts tremendous pressure on the network. Due to the rapidly changing network topology [6] the beaconing rate must be high (=10Hz) to create an adequate awareness.

Alternatives for creating this awareness exists i.e. a Shared Context Aware system [14]. In such a system a link is established between V2V and V2I network. By putting the knowledge of the current situation in the off-road infrastructure the number of beacons required can be reduced massively.

However, regardless of its efficiency, this a costly alternative compared to V2V VANETS.

Other problem is that VANETs are not able to approach the designed speed. A node can never be aware of the precise network state at that particular moment. There will always be capacity loss in idle time and collisions. These collisions may be a result of the hidden node problem, cause VANETs don’t use the Ready To Send/Clear To Send extension. [4]

By not being able to sense the exact network state congestion is unavoidable in VANETs. When the medium is congested the actual throughput rate lies above the designed throughput rate. In this state delays increase significantly while the throughput gradually decreases.

To avoid this many congestion avoidance protocols have been developed. Problem is that most of these protocols just tackle one of the many problems causing the network to congest. Studies [4], [6] show that these protocols [17] don’t work good enough. The chances of a safety message is lost in a crowded situation is still enormous (>0.8) [17].

Despite the high packet loss ratings messages still reach their destinations relatively fast (within 55ms over 100nodes/1km) [17]. But since the expectation is that more and more applications will become available in the future the network load has to be reduced to prevent the medium from being congested even more in the future.

3. PROBLEM STATEMENT

In this section we clearly define our problem statement. Where other papers focus on eliminating data redundancy on the medium; our focus goes to the combining of data, the so-called piggybacking. We want to design an experiment in which the effect of piggybacking on the medium can be measured.

This experiment will not actually be carried out in this paper but after carrying out our designed experiment the following research question can be answered:

"Is piggybacking application data onto scheduled beacons a way to reduce the busy time on a VANET compared to other forwarding techniques?"

To make the idea of piggybacking more clear see the functional analysis below (Fig. 1).

White squares represent the moment when the to be sent data is generated, and the grey squares represent the moment of actual sending. Note that with separated dissemination these moments are identical, given the fact that the medium is ready at the moment and one DIFS has passed.

![Figure 1, functional analysis between straight on forwarding and piggybacking](image)

Putting this question to test can be done be carrying out a large number of experiments, these are discussed in section 6. The piggybacking forwarding technique has to be tested against other protocols to determine which leads to the lowest busy time on the network. These other protocols will be introduced throughout section 4.

To draw conclusions out of these experiments average has to be taken from large sets of simulations. This because there is some randomness involved that we want to eliminate. This randomness will be explained in the next section. A set-up for such an experiment will be discussed in section 6.

4. FORWARDING PROTOCOLS

In this section the piggybacking protocol will be explained further and some other protocols, that forward data in another way, will be introduced. All of these protocols use their GPS signal to determine their relative position to the receiver. In this way the node can sense whether its closer to the receiver than the node that just sent the packet to him. When there is no node available to transmit the message to then the store-carry-forward mechanism [12] is used. Basically this means that the node will carry the message with him until another node that is closer to the destination becomes in range. Then the message is
forwarded to the node lying closer to the destination. This destination can either be a node or an area. This last case is called geographical routing.

4.1 PIGGYBACKED FORWARDING
The first to be introduced forwarding protocol is the one that uses piggybacking. If an efficiency message has to be forwarded using this protocol it waits until the next beacon kicks in. In this case the to be send message is piggybacked onto the beacon and sent along. This means that the time till transmission will be something between zero and the beaconing frequency \( f^{-1} \) (see Fig. 2).

![Graphical overview of timing within piggybacking](image)

**Figure 2.** Graphical overview of timing within piggybacking

Because of the randomness involved in this forwarding method it is necessary to conduct a large number of experiments to obtain averages. Then the formula to calculate the average expected number of hops required to reach the destination is:

\[
\text{\( n_{\text{hops}} = \frac{d}{0.5R} = \frac{2d}{R} \)}
\]

**Equation 1**

In which:

\( n_{\text{hops}} \) = the amount of hops required to propagate a message over a certain distance \( d \)

\( d \) = the distance between sender and receiver

\( R \) = the transmission radius (equal for each node)

4.2 STRAIGHT ON FORWARDING
This second protocol will just straight on forward information without the use of piggybacking. The message with application data will be broadcasted between beacons. A node within range of the sender will receive the message and start a countdown timer. The value of this timer will be a random number between the smallest time slot and the time between beacons \( f^{-1} \).

The timer interval is set to these values to make a good comparison between this method and the piggyback method. Using two different timers would result in an unfair comparison.

Because beacons can be considered asynchronous this will most likely cause all nodes to pick a different moment to retransmit the received message. Note that it’s still possible that they pick the same moment due to the random nature of both the timer and the next beaconing moment.

Assume node S sends a message M over a distance \( d \) to node R in a certain direction. All nodes that receive within the range of S who successfully receive M will start a timer. The node with the first expiring timer (hereafter noted as \( F_1 \)) will attempt to retransmit this message. If this retransmission is successful all nodes behind \( F_1 \) will discard their time and remove the message. Because the message is now closer to the destination they are. The responsibility to propagate the message further into the direction of the destination isn’t theirs anymore.

In contradiction to the nodes that lie behind \( F_1 \) the nodes lying before \( F_1 \) will not discard the message. This because there is no confirmation that the message has been propagated further along the road. So in the event of a lost message, the nodes lying after \( F_1 \) will still have a timer running. If they don’t receive a retransmission by a certain \( F_2 \) they will retransmit the message to prevent it from dying out.

After all, assuming that there has been no loss the average number of hops will be the distance to be travelled divided by half of the transmission range. This division by 2 is because of the random nature of the timers. Further propagation of the message will in some cases be handled by a node lying close to the sender, other times by a node lying further from the sender.

4.3 DISTANCE BASED ROUTING
Distance based routing is one of the more intelligent forwarding protocols. It uses the relative position of a node towards the sender to adjust forwarding chances. In this section we will illustrate this principle with an example.

Consider the image below (Fig. 3) with 9 nodes. The eclipse around \( n_2 \) illustrates the transmission range of that node. In this case \( n_2 \) (S) wants to send a message to a certain area (D). S will piggyback the message onto the next scheduled beacon and transmit it.

![Diagram of a piggybacked message](image)

**Figure 3.** Example flow of a piggybacked message (1/2)

Nodes \( n_1, n_3, n_4 \) and \( n_5 \) will receive this beacon. Node \( n_1 \) will sense that he is further away from D than S is so will not take any action. This sensing is done with the cars GPS signal. In absence of this signal, i.e. in tunnels, the Received Signal Strength (RSS) of the received packet can be used to determine whether to forward the packet or not as discussed in [19].

Nodes \( n_2, n_4 \) and \( n_5 \) are all candidates for forwarding the received message. Ideal would be if \( n_2 \) would forward the message since this node is the closest to the destination D. This can be achieved by using a 1-persistent protocol in which every node forwards every incoming packet with a probability of 1 [18]. In other words, every incoming packet is being forwarded. In this way large packet penetration can be achieved; drawback in this method is that a lot of redundancy comes in play and that this method floods the network [19]. Therefore 1-persistence is not the way to go in this scenario. A promising method is distance based routing as introduced in [19]. In this method the chance of further propagation along the line depends on the distance between the sender and the receiver. A higher distance leads to a higher probability and vice versa.
5. COMPARISON

To make a fair comparison between two forwarding methods it is necessary that both methods use roughly the same amount of hops and that all other parameters are identical in both test setups. This to ensure that these won’t influence the results.

Therefore we compare the piggybacking way of forwarding with the straight on forwarding protocol first. This because of the same kind of randomness they share and therefore they will on average use the same amount of hops. Therefore they can be equally compared to measure the impact on the medium. By analyzing both protocols it is clear that the number of hops will be roughly the same in both cases. The time that it will take for a message to reach the destination relies heavily on the random nature of the involved retransmissions in both schemes.

### Table 1: Differences between piggybacking protocol and the straight on forwarding protocol as discussed in section 5.1

<table>
<thead>
<tr>
<th>Method</th>
<th>Time to transmission</th>
<th>#attempts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Straight on</td>
<td>random(0,Bf)</td>
<td>2</td>
</tr>
<tr>
<td>piggyback</td>
<td>random(0,Bf)</td>
<td>1</td>
</tr>
</tbody>
</table>

In table 1 differences between the two methods are being shown given the fact that the number of requires hops is equal. The biggest difference is the (minimal) number of attempts required to transmit. By the usage of piggybacking the number of required attempts on the network will be identical to the amount of beacons sent.

Note that there is also a difference in the amount of headers required to send the data. In the regular way of disseminating data two blocks are required where in the piggybacked case on one is needed.

When transmitting data in a VANET the chances of collisions are pretty high [2], especially when the inter node distance is
low. Therefore reducing the number of attempts required is a valuable thing. Since reducing collisions on the network will lead to a lower busy time it is likely that reducing the number of attempts made on the network will also lead to a lower busy time in comparison to the straight on forwarding protocol.

When comparing this straight on forwarding protocol with the piggybacking method it is obvious that, in a scenario where both methods require an equal amount of hops, the piggybacking method saves on transmission attempts and headers. Maximum header size is 32 bytes. Compared to the maximum payload on a MAC packet (4,095 bytes) it is only under the 1% in size. But in practice most packets only contain around the 1,500 bytes. [15]

This means more than 2 percent saving on packet overhead per piggyback. Of course the piggybacking of multiple packets onto one beacon is possible and the savings on overhead will just continue to rise. Certainly if we scale up to a traffic jam over 4 lanes with 100 cars/km/lane.

6. EXPERIMENTAL SET-UP
In this Section we describe our experimental set-up to evaluate the performance of piggybacking. The goal is to conduct an experiment in which the effect of piggybacking will be examined. Idea is to run at least two simulations on the same scenario, one simulation will use the straight on data dissemination strategy (no piggybacking involved) and the other one uses the piggyback strategy. The only difference between the two should only be the forwarding strategy. All other parameters should be identical.

The to be simulated network will be a V2V VANET. Because communication between nodes happens on a much smaller timescale than the physical movement of nodes the nodes can be considered to be static. The broadcast range of one node can differ very much due to different circumstances. Killat et al have developed an empirical model for broadcast range using a Nakagami-m distribution [9] because deterministic models use a fixed broadcast range. In practice the probability of successful transmission decreases gradually. Studies [16] show that such a model can be used to model ideal circumstances on highways.

A widely available cross-platform software tool for simulations in general is OMNET++ [13]. Combining this with the MiXiM plugin [11] that is especially designed for simulating vehicular networks gives a solid base to run the simulations on.

Expectations in such an experiment are that without the use of piggybacking the data will be faster at the destination. This because the to be forwarded message will not have to wait for the next scheduled beacon to be piggybacked on. With piggybacking the data will travel slower but the impact on the medium will probably be reduced.

However there are a lot of parameters involved such an experiment we can define some parameter ranges prior to the experiment. The most important parameters are (derived from [2]):

- **Number of nodes (n)**
  The number of nodes should be varied throughout the experiment. This can be done by altering the number of lanes \( l \), or changing the vehicles/km/lane \( \rho \).
- **Beacon rate \((\lambda_b)\)**
  Small changes are likely to have a huge impact on the busy time of the medium. The beacon rate can be altered, but safety applications may malfunction in such a case.

* **Distance between sender and receiver**
  A message has to be sent from a certain node to an area over a distance large enough (>1000m) that multiple hops are required to reach the destination. Distance should be varied throughout the experiment.

As previously stated, traffic can be considered static in such environments due to the small timescale on which the transmissions take place. Most likely will piggybacking lead to a lower network load if the car density has a certain value. Further details about parameters (such as transmission power and time slots) can be found in [2] and in the 802.11p standard [18].

6.1 MEASURING
Lots of methods are available to measure the amount of stress on the medium, all using other parameter settings. A few basic parameters that can be used are:

- **Busy Time**
  Percentage of how much time the medium is busy
- **Number of hops**
  The amount of hops needed in a certain case to propagate the message
- **Number of Collisions**
  The amount of occurred collisions while transmitting the message from sender to the receiving area
- **Travel time**
  The time it takes for a message to travel from sender to receiving area.

All of the parameters above serve different needs. For our purpose the busy time will be the most important parameters since we are examining the impact on the whole network. This busy time is easy to measure. Just by dividing the time that the medium is busy by the total time observed this busy time ratio is obtained.

7. CONCLUSION
In this paper we have designed an experiment in which the impact on the busy time of the network can be measured when piggybacking application data onto scheduled beacons, compared to the situation in which application data is forwarded independently of the beaconing process. We also gave some insight in which parameters are important and how they should be used in the experiment.

Based on the theory on the 802.11p protocol the assumption raises that piggybacking application data onto scheduled beacons will lead to less stress on the medium in certain situations. It’s likely that these situations involve a medium traffic density. To make sure that this assumption is right the designed experiment should be carried out.

When it comes to measuring the impact of piggybacking on the medium multiple possibilities exist, each focusing on a different aspect. When it comes to measuring the overall stress level on the network, the network load approach seems most suitable.
7.1 FUTURE WORK
Most important is to actually carry out the experiment as discussed in this paper. In this way it is possible to determine whether piggybacking is able to reduce the busy time on the medium. It is likely that when comparing the piggybacking forwarding method with the straight on forwarding method introduced earlier, piggybacking looks more favourable.

Therefore should piggybacking also be compared with more intelligent protocols such as distance-based vector routing, or (micro) slotting as introduced in section 4.

8. REFERENCES


