Department of
Information Engineering
and Computer Science DISI

UNIVERSITY
OF TRENTO - Italy

DISI - Via Sommarive, 14 - 38123 POVO, Trento - Italy
http://disi.unitn.it

Models and Performance of VANET based Emergency Braking

Michele Segata and Renato Lo Cigno

March 2011

Technical Report # DISI-11-354
Version 0.1

Extended Abstract

The network research community is working in the field of automotive to provide VANET based safety applications to reduce the number of accidents, deaths, injuries and loss of money. Several approaches are proposed and investigated in VANET literature, but in a completely network-oriented fashion. Most of them do not take into account application requirements and no one considers the dynamics of the vehicles. Moreover, message repropagation schemes are widely proposed without investigating their benefits and using very complicated approaches.

This technical report, which is derived from the Master Thesis of Michele Segata, focuses on the Emergency Electronic Brake Lights (EEBL) safety application, meant to send warning messages in the case of an emergency brake, in particular performing a joint analysis of network requirements and provided application level benefits. The EEBL application is integrated within a Collaborative Adaptive Cruise Control (CACC) which uses network-provided information to automatically brake the car if the driver does not react to the warning. Moreover, an information aggregation scheme is proposed to analyze the benefits of repropagation together with the consequent increase of network load. This protocol is compared to a protocol without repropagation and to a rebroadcast protocol found in the literature (namely the *weighted p-persistent rebroadcast*).

The scenario is a highway stretch in which a platoon of vehicles brake down to a complete stop. Simulations are performed using the *NS-3* network simulation in which two mobility models have been embedded. The first one, which is called Intelligent Driver Model (IDM) emulates, the behavior of a driver trying to reach a desired speed and braking when approaching vehicles in front. The second one (Minimizing Overall Braking Induced by Lane change (MOBIL)) instead, decides when a vehicle has to change lane in order to perform an overtake or optimize its path. The original simulator has been modified by

- introducing real physical limits to naturally reproduce real crashes;
- implementing a CACC;
- implementing the driver reaction when a warning is received;
- implementing different network protocols.

The tests are performed in different situations, such as different number of lanes (one to five), different average speeds, different network protocols and different market penetration rates and they show that:

- the adoption of this technology considerably decreases car accidents since the overall average maximum deceleration is reduced;
- network load depends on application-level details, such as the implementation of the CACC;
- VANET safety application can improve safety even with a partial market penetration rate;
- message repropagation is important to reduce the risk of accidents when not all vehicles are equipped;
- benefits are gained not only by equipped vehicles but also by unequipped ones.

Keywords: VANET; Vehicular Networks; Emergency Braking Control; Cruise Control; *NS-3* simulation; re-broadcast schemes.

UNIVERSITÀ DEGLI STUDI DI TRENTO

Facoltà di Scienze Matematiche, Fisiche e Naturali

UNIVERSITY OF TRENTO - Italy

Corso di Laurea Magistrale in Informatica

Final Thesis

Models and performance of VANET based emergency braking

Relatore/1st Reader:
Prof. Renato Antonio Lo Cigno

Laureando/Graduant:
Michele Segata

Anno Accademico 2009 - 2010

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

Vehicles are a fundamental part of our life. We use them every day for different purposes, from work to fun. As written in the *National Transportation Statistics* published by the U.S. Department of Transportation [1], the number of cars in the United States in year 2008 was around 137 million, without including motorcycles, trucks and buses. If we consider that the number of inhabitants of the U.S.A. is around 300 million, including non driving people such as infants, we can understand how much road transportation is diffused. In Italy, the number of cars at the beginning of 2010 was 36 million [2], for a population of 60 million people [3] (60 cars every 100 persons).

Safety is one of the most important concerns: according to the *FARS/GES 2008 Data Summary* [4], in 2008 there were 5.8 million crashes in the United States. Roughly 34.000 were fatal, 1.6 million have caused people injuries while the others have caused only property damages. In 2000[1] the economic impact of car accidents was $230.6 billion [5], which means roughly $820 per living person (still in the U.S.). These values includes costs related to lost productivity, medical costs, legal and court costs, emergency service costs, insurance administration costs, travel delay, property damage and workplace losses.

These numbers show how much it is important to work on accident prevention: first of all to reduce deaths and injuries, second to save money and finally to save people time. Indeed when a car crash happens, people can remain stuck in a traffic jam for a long time.

Vehicle-related technologies are giving a great help in security improvement and driving comfort. For security, think of the Antilock Braking System (ABS), which prevents wheel lock in violent braking, or the Electronic Stability Control (ESC), which improves stability by detecting and minimizing skids. For comfort, think of parking sensors or GPS based navigation systems.

[1] At time of writing this was the most recent report available

1.1 THE IDEA OF VANET

In the field of Computer Science and Networking, researchers are actively working to develop WLAN enabled communications between vehicles. This kind of networks has been defined as VANETs, which stands for Vehicular Ad-hoc NETworks: this acronym is inspired by Mobile Ad-hoc NETwork (MANET). A MANET is a kind of network where the position of the nodes is not fixed and they do not rely on a preexisting infrastructure, so every communicating device actively partecipate in routing and data forwarding.

Indeed a vehicular network can be considered as a particular kind of MANET, but with some differences. VANET nodes do not move randomly but in an organized fashion, and their movements are constrained in streets.

Inter-vehicle communications give the basis for any kind of networking software. The *Vehicle Safety Communications Project* [6] identifies a set of applications classified into safety and non-safety, but always related to the pure automotive field.

An example of a safety application described in the document is the *Post-crash warning*: the idea is broadcasting of warning messages by cars that are stuck on a lane due to an accident or a mechanical breakdown. Vehicles receiving these warnings will inform their drivers of the danger, to avoid potential collisions. It can be very useful, for example, when the driver of the crashed vehicle has not yet placed the red warning triangle on the road, or in the case of limited visibility due to fog. Look for an example at Figure 1.1. Classified as non-safety is instead the *Instant messaging* application, which enables drivers to send text messages each other, for example to signal problems such as a flat tire. Another example is the Enhanced Route Guidance and Navigation (ERGN) which can inform the GPS-based navigation system of recent road changes, such as detours, by means of messages received from access points placed on the side of the road.

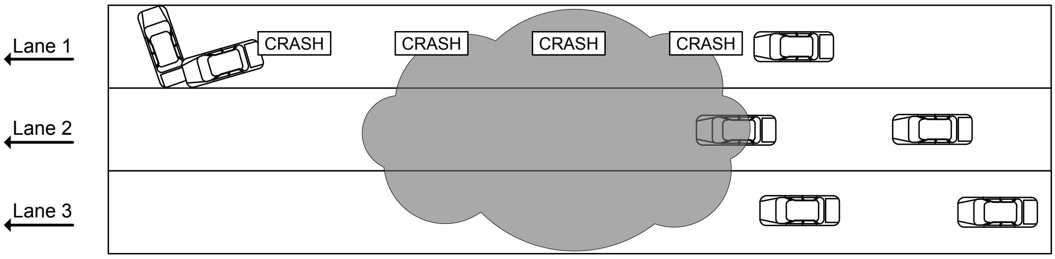


Figure 1.1: The *Post-crash warning* application. Crashed vehicle signals the danger using wireless messages and incoming vehicles can be warned quickly even in low visibility conditions

These are only three examples out of tens described in U.S. DOT [6]. As said before these are pure automotive applications, while others can also be supported by VANETs. For example they can provide Internet connection or inter-vehicle gaming as a novel way of entertainment.

Different applications have clearly different requirements, in terms of bandwidth, connection duration, delay, etc... and it is not possible to analyze them all in a single thesis. The next section will introduce the aim of this work.

1.2 EMERGENCY ELECTRONIC BRAKE LIGHTS

This thesis will focus on a safety application, in particular on the Emergency Electronic Brake Lights (EEBL).

All vehicles are equipped with brake lamps which are meant to warn following vehicle on the activation of brakes by the front driver. However they suffer of two problems. The first one is that the lamp gives no quantification, it does not tell if the driver is pushing lightly or strongly the pedal. The second one is visibility; in the case of fog it is possible that a driver sees the brake lamps of the front vehicle when it is too late to avoid a crash. As another example, when a large vehicle is between two cars, if the first brakes the second will not see its lights. See Figure 1.2 for a better understanding.

Moreover, with usual brake lamps human reaction times cumulate, increasing the risk of chain collisions. Reaction time usally ranges between 0.75 and 1.5s [7] so a vehicle in a queue can start braking several seconds after the leader and, with a speed of 36 m/s (roughly 130 km/h), the distance covered during this period is not negligible.

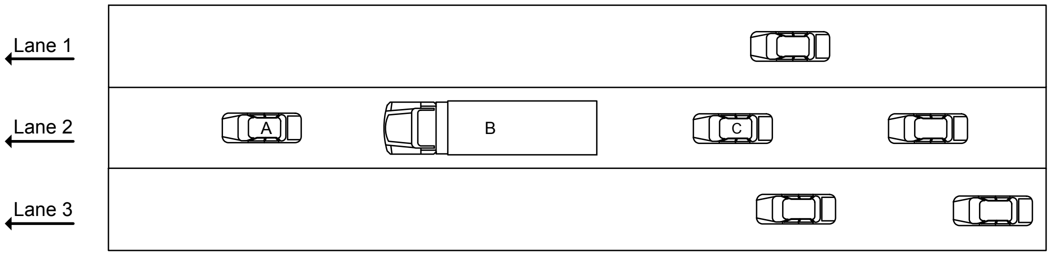


Figure 1.2: Brake lights visibility problem. If vehicle A engages brakes, vehicle C will not be able to see it due to truck B in between

A VANET enabled application could improve safety and reduce the risk of car crashes. As presented in U.S. DOT [6] the EEBL application should broadcast warning messages when the entity of the deceleration is over a certain threshold.

These messages should provide information about the braking vehicle, such as speed, acceleration, position, etc... In this way both problems of quantification and visibility will be solved; a driver could be informed that, five vehicles in front, a car is braking with a deceleration of, for example, 3 m/s^2 and be more focused on driving.

However, the most interesting application is that EEBL-provided information could be passed to an Adaptive Cruise Control (ACC) system. A Cruise Control (CC) is a system which mantains a vehicle on a constant speed selected by the driver. The Adaptive Cruise Control improves the behavior of the CC by automatically decreasing speed if the vehicle is too close to the front one and by maintaing the safety distance. The ACC works using a radar which detects if there is a vehicle in front and determines its speed and distance.

Unfortunately, radar based ACC can give information only about the front vehicle, while a driver could be interested in having a wider perspective, not only about vehicles in the same lane but also in other lanes. Indeed if a car applies the brakes strongly, knowing it is useful for all drivers behind: as an example, imagine the situation in which an animal crosses the street.

Instead of using a radar, the ACC could work using the EEBL messages received from other vehicles: the ones from front car to mantaing safety distance and apply the brakes if needed, the others to give a global perspective to the driver. This will be the concept in this thesis, but in reality this information could be used to improve traffic flow by using the so called Collaborative Adaptive Cruise Control (CACC). The CACC is an improved version of the ACC which uses the information of more vehicles in front to mantain a lower inter-vehicle distance, increasing flow density without a higher risk of crashes, since variations are obtained in real time via wireless messages. More details about the advantages of CACC can be found in van Arem et al. [8].

In this thesis the CACC will be considered as a system which only mantains safety distance and brakes the car in the case of an emergency situation; it will not implement a car following mechanism, so it will not keep a constant distance from front vehicle. If the driver decides to go slower than front vehicle, the system will not accelerate.

An advantage of VANET technology is that once a WLAN card with a processing unit has been installed, all the applications which do not require to actively control the vehicle (e.g., the *Post-Crash warning*) would be simply treated as *Yet Another VANET Application* and could be installed, for example, via a software update.

Obviously some issues have to be considered. VANETs rely on wireless communications so most of the problems (if not more) that we encounter in WLANs will have to be faced.

The focus of this thesis will be a joint analysis of EEBL application and network protocols: indeed as described in Chapter 3 this approach is completely missing in the literature. Either the research completely focuses on network performances or only describes the advantages of a particular application. Second an analysis of the benefits of message repropagation will be performed: is it worth it to replicate EEBL messages more far than a single hop? If so, which consequences has this on the network? Finally, the thesis will analyze the effectiveness of a simple message aggregation protocol.

2 COMMUNICATION TECHNOLOGIES

The Federal Communications Commission (FCC) in 1999 has begun the process of standardization for vehicular communications by allocating the 5.850-5.925 GHz band (75 MHz) for Dedicated Short Range Communications (DSRC)[1]. Then a set of protocols for Wireless Access in Vehicular Environments (WAVE) has been developed by IEEE, in particular by the working groups 802.11p [9] and P1609 [10, 11, 12, 13].

P1609 (P1609.1 to P1609.4) focuses on higher layers: for example P1609.2 addresses security issues, such as encryption and authentication, while P1609.4 regards multi-channel operations. Given the aims of this thesis, these protocols will not be described in details.

802.11p is instead an amendment to IEEE 802.11 [14] which defines the changes for PHY and MAC layer to address communication requirements for vehicular networks, not only Vehicle-to-Vehicle (V2V) but also Vehicle-to-Roadside (V2R) (and viceversa). Indeed some VANET applications cannot work in a pure V2V network, because interaction with the "outside world" is needed. Think as an example to the ERGN application mentioned in Section 1.1: for GPS maps update an access point on the road is needed, at least for initial diffusion. This access point in the VANET terminology is called RoadSide Unit (RSU).

Notice that none of this standards defines application level protocols like re-broadcast metodology or aggregation mechanisms; this aspects are left to the research community.

Understanding 802.11p is fundamental because it has been employed in the simulations described in Chapter 5, but first some notions about 802.11 have to be given.

2.1 802.11 BASICS

The 802.11 standard specifies Physical Layer (PHY) and Medium Access Control (MAC) for WLANs, so it defines the characteristics of the lower layers, such as trans-

[1] The FCC news release can be found at `http://www.fcc.gov/Bureaus/Engineering_Technology/News_Releases/1999/nret9006.html`

mission speeds, transmission techniques, frames format, medium access mechanisms, etc . . . .

Commonly used 802.11 protocols work in the Industrial Scientific and Medical (ISM) band, in particular in the 2.4 GHz (802.11 b and g) and the 5 GHz (802.11a) bands. They use different modulation mechanisms, providing different transmission speeds. For example, 802.11b uses Direct Sequence Spread Spectrum (DSSS) with a maximum transmission speed of 11 Mbps. 802.11a and g instead use Orthogonal Frequency Division Multiplexing (OFDM) which provides up to 54 Mbps.

The DSSS technique is represented (in a very simplified way) in Figure 2.1: basically a random sequence of 1 and -1 values (called chipping sequence) is generated at a frequency much more higher than the one of the original signal. The product of the signal and the sequence results in the spreading of the energy over a wider range of frequencies. The receiver can perform the "de-spreading" by multiplying received signal by the same chipping sequence used by the sender, so the synchronization between sender and received is required for a correct decoding.

The advantage of the DSSS modulation is the resistance to narrow-band interference. Imagine that the original signal in Figure 2.1 is the interference: when the receiver applies the chipping sequence to received signal, the interference is spreaded over the spectrum, so the original signal can be decoded anyhow.

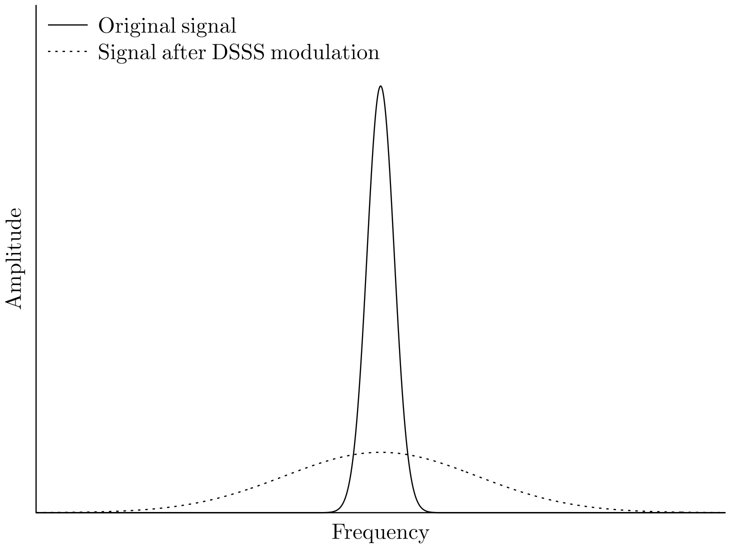


Figure 2.1: DSSS modulation

OFDM uses instead a set of orthogonal subcarriers (Figure 2.2) to perform multiplexing of datastreams. In other words, datastreams are sent in parallel, each of them modulated with a different subcarrier: this mechanism permits to reach higher transmission speeds.

Two signals $\varphi_n(t)$ and $\varphi_m(t)$ are said to be orthogonal in the interval $a < t < b$ if

$$\int_a^b \varphi_n(t)\varphi_m^*(t)dt = 0 \qquad [15]. \tag{2.1}$$

In Equation (2.1), $\varphi_m^*(t)$ indicates the complex conjugate of $\varphi_m^*(t)$, that is the same function but with the imaginary part negated. For example, the complex conjugate of $3 + 2i$ is $3 - 2i$. The orthogonality of the subcarriers is important because it ensures that no inter-carrier interference occurs during transmission.

Notice that DSSS and OFDM are complex modulation techniques, but their description is beyond the scope of this thesis. Two comprehensive reading with background, algorithms, mathematical definitions and examples are the books by Couch [15] and Paulraj et al. [16].

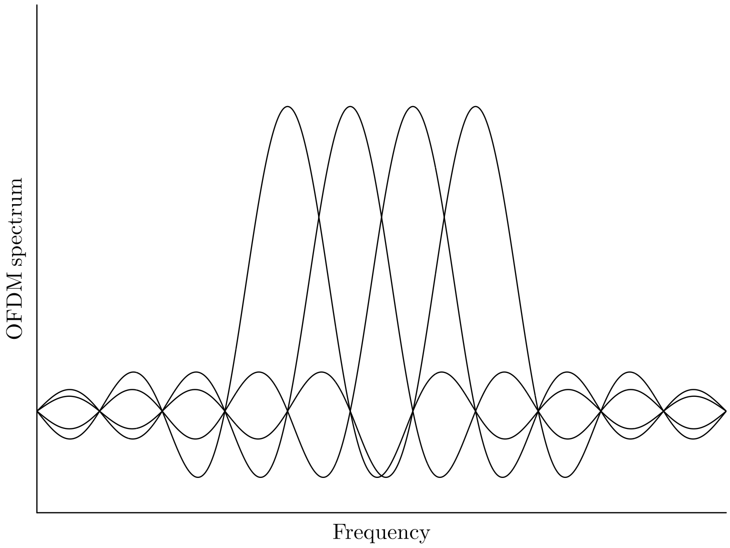


Figure 2.2: OFDM subcarriers representation

All the 802.11 protocols, which are completely different by the point of view of the physical layer, use the same mechanism to obtain access to the medium, that is the so called Distributed Coordination Function (DCF), which follows the Carries Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol. The principle of CSMA is to "listen" if the channel is busy before transmitting data, in order to avoid the disruption of ongoing transmissions. Collision avoidance instead refers to the procedure that the stations obey in order to contend for the channel and minimize the number of collisions. A collision occurs when two nodes (or more) access the channel simultaneously, often causing trasmitted packets to be lost[2].

The DCF basically uses a random defer time (backoff, counted in slots) for transmission if the channel is sensed as busy: before performing the backoff, every station must wait a predefined period of time. Figure 2.3 gives a graphical representation of the mechanism. The purpose of this amount of time is the prioritization of possible subsequent transmissions. To clarify this aspect, it is better to describe the time intervals of Figure 2.3:

SIFS the Short Inter Frame Space is the time elapsed between frames belonging to the same transmission, for example a frame and its acknowledgement. Indeed, for unicast messages, 802.11 uses an ACK, which is the only way to determine if a packet has been correctly received. SIFS is the shortest time because the ACK is the packet with higher priority after a transmission. Another example of usage of SIFS is when fragmentation is adopted: in this case the transmission will be a sequence of frames and ACKs separated by SIFS;

PIFS the PCF Inter Frame Space is used when polling is performed by the access point. The access point can indeed give access to the channel to a particular station in its polling list. This has priority over the distributed channel access, so PIFS is the shortest time after SIFS. It is computed as a SIFS plus one slot time;

DIFS after a Distributed Inter Frame Space, stations can begin the backoff procedure and try to access the channel. It is computed as a SIFS plus two slot times. This time is higher than SIFS and PIFS: if the stations "hear" the channel free for this period of time no higher priority transmission will be performed.

The backoff procedure is simply a "countdown" of a number of slots that is randomly generated in the range [0,CW], where CW is the Contention Window. While counting, a station continuously listens to the channel; if the countdown reaches 0, it can start transmitting, otherwise the value is saved and countdown continues at next contention. If two (or more) stations reach 0 at the same time, a collision occurs; the collision is detected by the missing ACK and, at next contention, stations will perform again the backoff procedure but with a doubled CW. This procedure is repeated until successfull transmission or until a maximum

[2] This is not always true, for example when transmitting stations are very far. In this case the signal of one transmitter could be simply "heard" as non-disrupting noise by a receiver near to the other transmitter

number of attempts has been reached. After a successfull transmission the size of the CW is shrunken back to the minimum value (CW_{min}). On collision instead, the CW can grow up to a maximum value CW_{max}. For example, the values for CW_{min} and CW_{max} for 802.11b and g are 31 and 1023 slots respectively. Notice that in the case of collision, the transmission of the packet is not stopped, because there is no way to detect it, as for CSMA/CD.

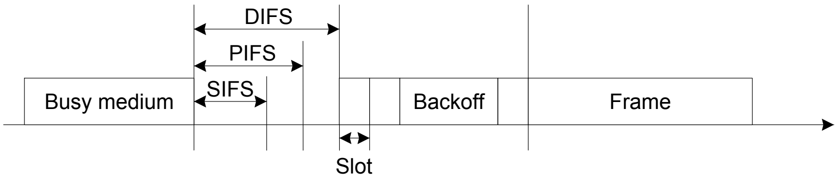


Figure 2.3: The 802.11 DCF access mechanism

In the case of broadcasting this procedure is much more simplified, because messages are directed to all stations in the transmission range of the sender and the ACK cannot be used. If the receivers are twenty, then twenty acknowledgements should be sent, wasting resources. Moreover, what can sender do if no ACK is received? Has a collision occured or are there no receivers in the transmission range? A list of neighbors should be mantained, but then what happens if a node suddenly shuts down? The sender would keep re-sending the same packet waiting for the ACK of the missing station which will never arrive.

So for broadcast messages, after packet transmission, a DIFS elapses and stations start again the contention. Rebroadcast mechanisms should be developed at higher layers. Moreover due to the absence of acknowledgements, the CW never grows.

2.1.1 802.11e and QoS

In 2005 an amendment to the original 802.11 has been developed by the working group e in order to support Quality of Service (QoS)[17]. The main enhancement is the replacement of the DCF with the Enhanced Distributed Channel Access Function (EDCAF), which introduces four access categories AC. The idea is to have four queues for packets with different priorities and let them compete virtually for channel access as with DCF: when two packets of different queues collide, the one with highest priority access the channel, while the other behaves as a real collision has occured. The access categories are:

AC_BK the AC with lower priority, for background traffic;

AC_BE the second AC for best effort traffic;

AC_VI the third AC for video;

AC_VO the AC with higher priority, for voice.

The internal contention mechanism is not enough because packets prioritization does not work among different stations. So different CWs has been used for different access categories. Table 2.1 shows how the values of CW_{min} and CW_{max} are computed in 802.11e. The table could seem a little bit misleading: the actual values of CW_{min} and CW_{max} must be computed depending on the physical layer. Indeed 802.11e describes only the EDCA mechanism, but this must be used on top of another 802.11 PHY standard, for example 802.11a. By looking at IEEE 802.11 [14, p. 626] it is possible to retrieve the values of CW_{min} and CW_{max} for 802.11a OFDM PHY (bandwidth 20 MHz), that are 15 and 1023 respectively. So if we use 802.11e on top of 802.11a PHY, EDCA will use the values shown in Table 2.2.

AC	CW_{min}	CW_{max}
AC_BK	CW_{min}	CW_{max}
AC_BE	CW_{min}	CW_{max}
AC_VI	$(CW_{min}+1)/2 - 1$	CW_{min}
AC_VO	$(CW_{min}+1)/4 - 1$	$(CW_{min}+1)/2 - 1$

Table 2.1: CWs size calculation in 802.11e

AC	CW_{min}	CW_{max}
AC_BK	15	1023
AC_BE	15	1023
AC_VI	7	15
AC_VO	3	7

Table 2.2: CWs size for 802.11e on 802.11a PHY

As shown by Tinnirello et al. [18], the CW differentiation is not enough and must be combined with the usage of different DIFS for different access categories. In particular a different Arbitration Inter Frame Space (AIFS) has been defined for each AC. AIFS[i] is the number of slot after a SIFS that AC i must wait before trying to access the channel. Low priority queues have higher AIFS value: Figure 2.4 shows how AIFS is used in EDCA. Table 2.3 shows the values of AIFS for the different access categories.

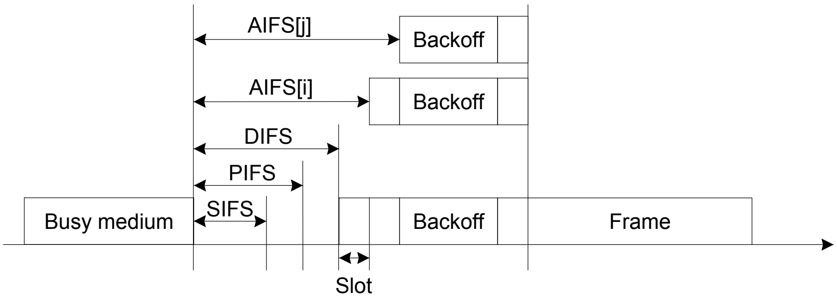


Figure 2.4: AC prioritization through AIFS

AC	AIFS
AC_BK	7
AC_BE	3
AC_VI	2
AC_VO	2

Table 2.3: AIFS values for 802.11e defined in IEEE 802.11e [17]

2.2 802.11p

The 802.11p standard [9] is a variant of 802.11a conceived for vehicular communications. One of the most important features of 802.11p is the ability to communicate in a complete ad-hoc manner, without the usage of the association process typical of 802.11 based communications. The association phase indeed would last too long for a highly mobile network as a VANET. So, as said before, security mechanism must be handled at higher layer.

Regarding the physical layer, the main change from 802.11a is the channel spacing, which as been reduced from 20 MHz to 10 MHz (OFDM is used), supporting 3, 4.5, 6, 9, 12, 18, 24 and 27 Mbps. The 10 MHz bandwidth is used due to robustness issues and the possibility to reuse existing wireless chipsets [19]. With the reduction of the channel spacing other PHY parameters change, such as slot time and SIFS time[3]. The transmission range, according to U.S. DOT [6], should be up to 1 km.

Seven channels have been reserved, in particular one Control CHannel (CCH) and six Service CHannels (SCHs). The aims of the CCH are basically two. The first one is sending safety messages between vehicles while the second is service advertisement: by using the CCH a vehicle can announce a service on a SCH. Since this

[3] So consequently also DIFS

thesis focuses on a safety application, only the CCH will be considered. Further details are given in the book by Hartenstein and Laberteaux [19].

Regarding the MAC layer instead, the EDCA mechanism of 802.11e is adopted in order to give safety messages a higher priority w.r.t. non-safety messages. Table 2.4 and Table 2.5 show the EDCA parameters for CCH and SCH.

AC	CW_{min}	CW_{max}	AIFS
AC_BK	CW_{min} (15)	CW_{max} (1023)	9
AC_BE	$(CW_{min}+1)/2 - 1$ (7)	CW_{min} (15)	6
AC_VI	$(CW_{min}+1)/4 - 1$ (3)	$(CW_{min}+1)/2 - 1$ (7)	3
AC_VO	$(CW_{min}+1)/4 - 1$ (3)	$(CW_{min}+1)/2 - 1$ (7)	2

Table 2.4: MAC parameters for 802.11p CCH

AC	CW_{min}	CW_{max}	AIFS
AC_BK	CW_{min} (15)	CW_{max} (1023)	7
AC_BE	CW_{min} (15)	CW_{max} (1023)	3
AC_VI	$(CW_{min}+1)/2 - 1$ (7)	CW_{min} (15)	2
AC_VO	$(CW_{min}+1)/4 - 1$ (3)	$(CW_{min}+1)/2 - 1$ (7)	2

Table 2.5: MAC parameters for 802.11p SCH

2.3 APPLICATIONS AND OPEN ISSUES

The use of WLAN based communications in road environments (both urban and highway) can be difficult due to different aspects. One of the problems is shadowing, which is the signal attenuation due to large objects between communicating parties, such as buildings or other vehicles. For an example look at Figure 2.5.

Another problem is fading, which is, in a very simplified definition, the deviation of the attenuation due to the overlapping of reflected signal with the original signal. For example imagine two cars running in a urban environment: if one sends a message to the other, the signal will be received directly and reflected by the nearby buildings (Figure 2.6). Clearly the reflected signal will follow a longer path than the original, thus it will be delayed: this delay can cause the reflected signal to amplify or destroy the original basing on the phase.

In VANET application testing these problems can be simulated using mathematical definitions of them. For example the work by Sommer et al. [20] describes

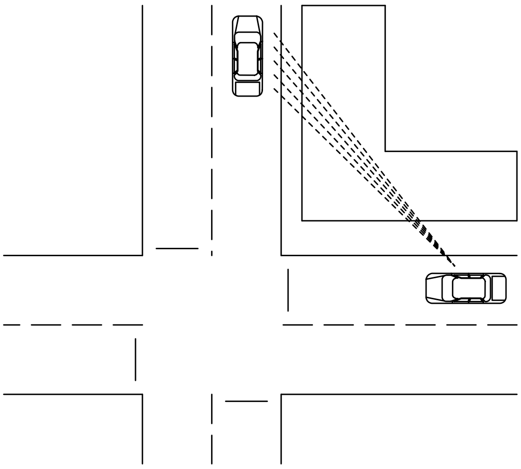


Figure 2.5: Shadowing problem

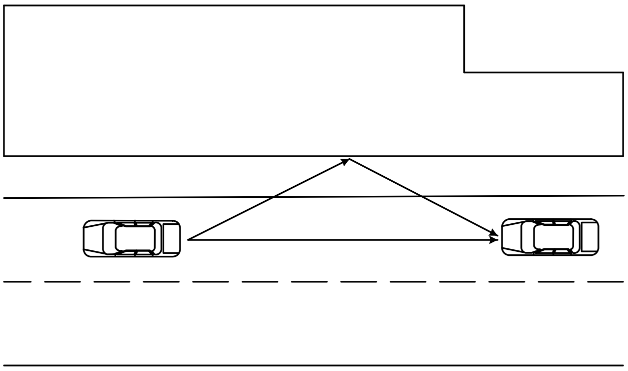


Figure 2.6: Fading problem

how to create shadowing effects in a urban environment, the paper by Mangel et al. [21] models the non-line-of-sight reception due to reflection while the m-distribution (a.k.a. Nakagami distribution [22]) is a popular model for the probability of packet reception under fading.

A recent survey by Boeglen et al. [23] describes a comprehensive list of V2V channel models: in particular the physical layer implementation for *NS*-3[4] by Papanastasiou et al. [24] is described as a very accurate reproduction of a real channel. This implementation has not been used in this thesis for two reasons: the first one is its high computational requirements. The original paper shows a minum effort increment of a factor of 330 w.r.t. the simpler implementations and proposes the usage of GPU computing to solve the problem. This is a scientific open problem in its own, going beyond the scope of this thesis. The second reason is that the level of details of this implementation is not needed to the purposes of

[4] The network simulator used in this thesis, see Chapter 4 for details

the thesis: indeed the implementation includes also the computation of the OFDM symbols, and seems more suitable for developing data-link channel models than studying applications.

This thesis is more concerned on congestion issues, i.e. what happens if a big number of vehicles broadcast ten EEBL messages per second during a violent braking. Figure 2.7 shows what can happen in a highway scenario.

When a driver (of vehicle A in Figure 2.7) senses a dangerous situation, such as an animal on the road, it reacts by applying the brakes. Consequently, the EEBL application will start broadcasting messages to inform the followers at a frequency of 10 Hz (phase 1).

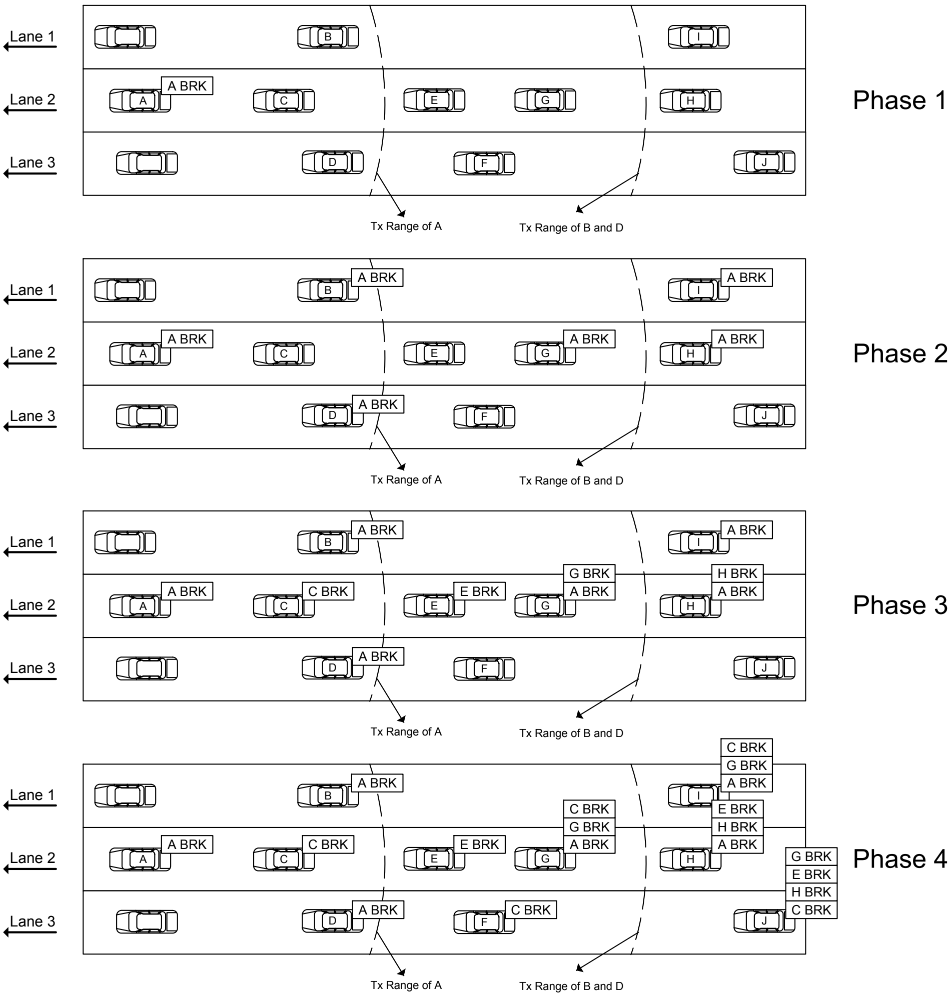


Figure 2.7: EEBL congestion issue. The four phases are described in the text

After a really small amount of time (in the order of a few hundreds of microseconds), since A is not able to reach directly all the vehicles behind it, some of its neighbors (for example B and D) will rebroadcast the message (phase 2). The same will do, for example, vehicle G after hearing B and D messages, and so on.

When drivers behind A become aware of the danger (around a second later, i.e. human reaction time) they also will start braking; as a result, vehicles C, E, G and H will start broadcasting their *"brake message"* (phase 3). Notice that the diffusion of vehicle A *"brake message"* does not stop.

Finally, vehicles behind C, E, G and H will also have to rebroadcast their messages, so network load will suddenly increase (phase 4). Section 2.3.1 attempts to describe a theoretical estimation of the network load.

2.3.1 Theoretical computation of network load

By using the notions about 802.11 given in this chapter, it is easy to compute the duration of the transmission of a single EEBL packet and then derive a theoretical network load as a function of the number of broadcasting vehicles.

To determine transmission duration some parameters must be obtained from IEEE 802.11 [14]: they are listed in Table 2.6. Values for slot time, SIFS, PLCP preamble and header[5] can be retrieved directly from the standard, while the others must be computed. For high priority safety messages the AC_VO access category should be used[6], so the AIFS value is 2 and consequently the duration is 58 μs[7]. For AC_VO the CW_{min} value is 3, so on average the backoff procedure will last 1.5 slots[8] (19.5 μs).

Other durations must be computed depending on the datarate. These quantities are the service field of the PLCP header (16 bits), the MAC and LLC header (30 and 8 bytes respectively), the FCS[9] (4 bytes), the EEBL message (137 bytes (see Section 4.5 for the details of this amount)) and the tail (6 bits). There are also some pad bits of variable length that should be considered[10] but for simplicity they are ignored. Moreover the ACK is not considered because we are dealing with broadcast messages.

[5] PLCP preamble and header have not been mentioned in Section 2.1 for simplicity. The preamble is used to synchonize the receiver with the sender, while the header contains informations such as the length of the packet and the transmission speed

[6] It could seem strange to use the AC for voice for vehicular traffic, but this is the notation used for 802.11e which has been mantained also for 802.11p. The AC for voice is the one with highest priority so it is best suited for safety messages

[7] A SIFS plus two slot times

[8] Remember that in broadcast the CW never grows

[9] The Frame Check Sequence is the checksum appended to the MPDU

[10] The pad bits are added to the packet because OFDM encodes a block of bits in a symbol, so if the size of the packet is not enough to fill the last block, some dummy data must be added

Parameter	Value	Unit
Slot	13	μs
SIFS	32	μs
PLCP preamble	32	μs
PLCP header	8	μs
AIFS	58	μs
Backoff (average)	19.5	μs

Table 2.6: 802.11p CCH PHY and MAC parameters for the AC with highest priority (AC_VO)

If the packet is transmitted at 6 Mbps (this speed has been used in simulations because it has been shown to be (in general) the best choice [25]), its duration will be roughly

$$\begin{aligned} T_{TX} &= AIFS + Backoff + T_{PLCP} + T_{MAC} + T_{LLC} + T_{DATA} + T_{FCS} + T_{TAIL} \\ &= 58\mu s + 19.5\mu s + \left(40\mu s + \frac{16b}{6Mbps}\right) + \frac{(240 + 64 + 32 + 1096 + 6)b}{6Mbps} \simeq 349\mu s. \end{aligned}$$

Now a rough approximation of the channel load for the EEBL application as a function of the number of vehicles n is:

$$Load_{eebl}(n) = \frac{349\mu s \cdot 10 \cdot n}{1s} \cdot 100(\%) \tag{2.2}$$

Clearly Equation (2.2) has a limit in the number of vehicles: it is impossible to have thousands of vehicles running in one kilometer of highway. Moreover, for n it is intended the number of vehicles that can "hear" each other. Anyhow, considering only ten vehicles the network load would be around 3.5%, without using any kind of message repropagation. If we consider fifty vehicles (e.g., in a multi-lane highway) the load would reach roughly 18%, again without message repropagation.

So it is important, if message repropagation is discovered to give benefits in terms of incident reduction, to design an intelligent protocol, in order to avoid a "storm" of messages which could potentially harm the functioning of the CACC.

3 RELATED WORK

In the literature, different methods have been discussed for information dissemination in VANETs, both for unicast and multicast/broadcast transmissions. For safety-related applications the multicast/broadcast approach is best suited; usually when a car sends an alert message, all its neighbors are interested in the transmission and clearly the usage of multiple unicast communications would be inadequate.

The simplest protocol that can be implemented is pure flooding, in which every node that receives a message immediately rebroadcasts it (if that message was never seen before, otherwise it is simply discarded). Obviously this mechanism leads to an unefficient use of the available resources. This problem has been defined by Ni et al. [26] as "*broadcast storm*" and its drawbacks are:

- **Redundancy:** if every node rebroadcasts the same message, this will be received surely more than once by a vehicle;
- **Contention:** in the rebroadcasting phase, nodes will try to access the channel almost at the same time;
- **Collision:** due to high number of contending nodes, the probability of collision will be larger.

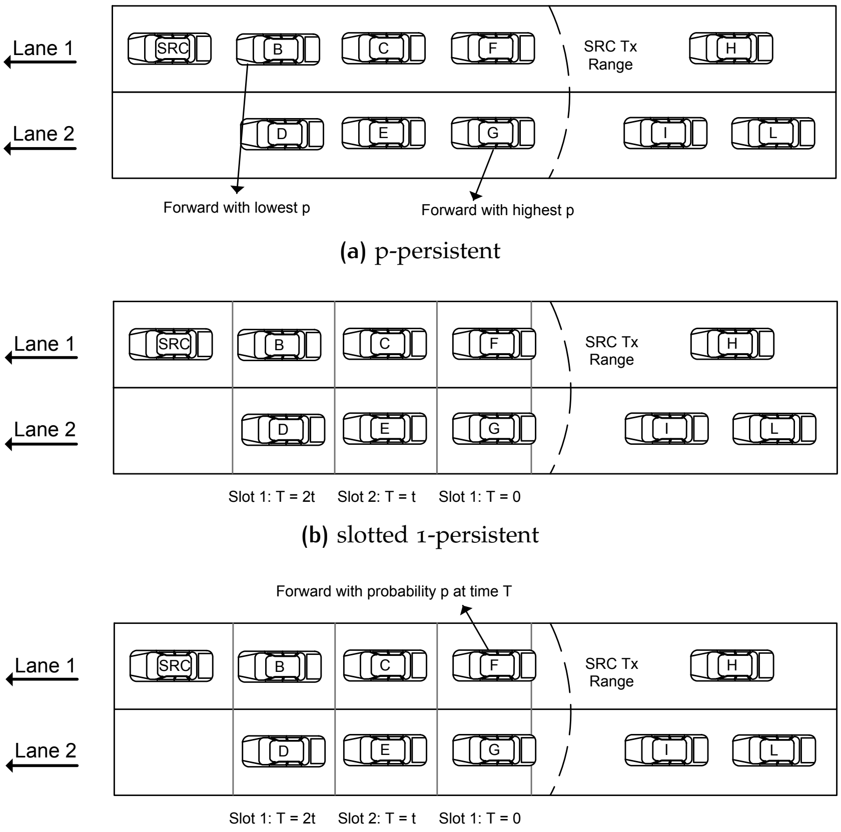
If we consider broadcast wireless transmission with a CSMA/CA access mechanism (as in 802.11), the damages caused by the third point will be even larger, due to the absence of collision detection. Another issue of flooding is the infinite replication of a message, which however can be easily resolved by inserting a Time To Live (TTL) value in the packet and decrementing it at each forwarding.

A simple countermeasure to the problems caused by pure flooding has been proposed by Haas et al. [27] and it is a probabilistic flooding (also referred as "*p-persistent*" flooding [28]); when a node receives a message that it has never seen before, it is rebroadcasted with a certain probability p. Pure flooding is a special case of probabilistic flooding where $p = 1$.

Probabilistic flooding can be combined with other mechanisms to improve node coverage: for example Pleisch et al. [29] defines an algorithm, called *Mistral*, which uses *p-persistent* flooding for initial spreading and then enters a compensation phase which collects a set of messages until a certain threshold (i.e., number of packets) is reached. After that a *compensation packet* is built and broadcasted.

P-persistent flooding has been revisited by Tonguz et al. [28] in three different manners:

- **Weighted p-persistence:** the probability p of rebroadcasting is proportional to the distance from the source node. In this way, farthest nodes will rebroadcast with higher probability, thus increasing dissemination speed;
- **Slotted 1-persistence:** the broadcasting range is divided into regions, called "*slots*" (S_i), and each of them is assigned with a specific transmission time T_i. A node belonging to slot S_i rebroadcasts with probability $p = 1$ at time T_i (retransmission time equal to 0 is assigned to farthest slot);
- **Slotted p-persistence:** in the last proposal a node in a slot S_i rebroadcasts a message at time T_i but with probability p.



(a) p-persistent

(b) slotted 1-persistent

(c) slotted p-persistent

Figure 3.1: Different persistence mechanisms [28].

A graphical representation of these three mechanism is shown in Figure 3.1. To determine distance from sender node it is assumed that each vehicle is equipped with a Global Positioning System (GPS) receiver, so that the location can be communicated to neighbors. This assumption is widely employed in the literature because this system is becoming more and more common and cheaper. Moreover, the presence of an on-board GPS is already considered by the U.S. Department of Transportation [6].

Several other approaches use distance from sender as a measure of priority. One of the simplest is described by Bachir and Benslimane [30]: basically when a node receives a message for the first time it calculates a defer time for retransmission. As for *weighted p-persistence*, higher priority is given to farthest nodes, so a lower defer time will be calculated. If during this period another node sends the same message, the latter is discarded, otherwise when the timer expires it is rebroadcasted. A similar approach is proposed by Briesemeister et al. [31].

Also Li and Lou [32] use a rebroadcast timer which value is calculated as a function of the distance, but with a slightly different algorithm proposal. Nodes are divided into forwarders and makeups; forwarders have the purpose of spreading the message as fast as possible along propagation direction, while makeups try to enhance node coverage by subsequent retransmission in the area covered by a forwarder. Potential forwarders calculate a rebroadcast delay; the one with the smallest value becomes the forwarder and informs other nodes by sending an acknowledgement and then relaies the message. After that, makeups start their rebroadcasting phase to improve the overall probability of reception.

A more sophisticated mechanism has been proposed by Korkmaz et al. [33]. The protocol, named *Urban Multi-hop Broadcast*, divides the portion of the road in the transmission range of the sender into segments along dissemination direction. If the farthest non-empty segment contains more than one node, it is iteratively divided into subsegments until a single node remains in a subsegment. If after a certain number of iterations is not possible to isolate a single vehicle, the nodes in the last segment enter a random phase.

Imagine to have road segments A, B and C (A is the nearest to the source while C is the farthest). The effective determination of the segments is obtained as follows; first of all source node sends a Request To Broadcast (RTB) packet including its position and dissemination direction. On reception of the RTB, each node sends a *black burst* signal of a length which depends on the distance from the sender. A black burst is simply a jamming signal [34]. Imagine that vehicles belonging to each segment calculate different black burst lengths $T_A = 50\mu s$, $T_B = 100\mu s$, $T_C = 150\mu s$[1]; if a jamming signal of those lengths is sent by each

[1] Note that these values are purely casual and used only to explain the mechanism

vehicle then only nodes in segment C will hear the channel free after their black bursts, so they will know to be in the segment which is more distant from the sender. Then each of them reply with a Clear To Broadcast (CTB): if in C there are more than one node, their CTB will collide and the procedure will be restarted by the source in order subdivide C, otherwise the source will send the data and wait for the ACK of the receiver. In the meanwhile, other nodes will also listen to the transmission.

The length L of the black burst for each node at the first iteration[2] is calculated as

$$L = \left\lfloor \frac{d}{Range} \times N \right\rfloor \times SlotTime$$

where d is the distance of the node from the sender, $Range$ is the transmission range, N is the number of segments and $SlotTime$ is the slot duration of a black burst. An interesting features of *Urban Multi-hop Broadcast* is the automatic adaptation to different traffic conditions: if the traffic is sparse then one iteration will probably be enough to find the farthest node, otherwise the protocol will perform more steps depending on how much traffic there is on the road.

Another protocol using the RTS/CTS mechanism is *Streetcast*, proposed by Yi et al. [35]. In the articles presented till here, the approach of relay selection was receiver-based, in the sense that nodes receiving a broadcast packet (or a RTB) decide by themself who has to act as a relay. *Streetcast* instead uses a sender-based selection algorithm; indeed the sender decides which nodes will rebroadcast the packet and includes this information into a Multicast Request To Send (MRTS). Relay nodes will reply with a CTS packet in the order in which they are listed in the MRTS. After that the source sends the packet and waits for the ACKs of the receivers: by counting the number of ACKs, it can decide whether the transmission is successful or not and in case restart the procedure.

In the paper by [36] is proposed the Distributed Vehicular broadCAST (DV-CAST) protocol which is designed to operate in different traffic conditions, namely dense, sparse and regular traffic regimes. When the traffic is dense, it is proposed to use one of the broadcast suppression mechanism defined by Tonguz et al. [28] (e.g., slotted 1-persistent); this will be enough to diffuse the message since a lot of vehicles are present on the road. If instead the traffic is sparse, we incur in a network which is *fragmented*, in the sense that it is built by a set of *islands* of vehicles which cannot directly communicate, i.e. they are outside their respective

[2] Lengths for successive iterations, as well as other protocol details, are clearly described in the paper by Korkmaz et al. [33]

transmission ranges. In this case the adopted mechanism is called *store-carry-forward* and, as the name suggests, it works by storing the information until it can be forwarded, for example when a new neighbor is discovered. For the third traffic regime, both approaches are considered since some vehicles can have a lot of neighbors while some others only a few.

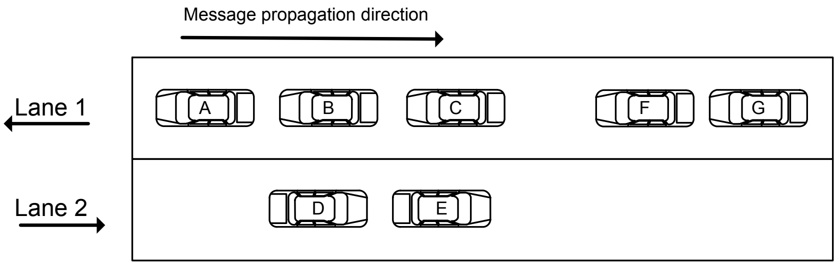


Figure 3.2: Different neighborhood conditions pointed out in the DV-CAST paper. Vehicles A, B, D and F are in a *well-connected neighborhood*, vehicles C and E are in a *sparsely-connected neighborhood* and vehicle G is in a *totally disconnected neighborhood*.

The different traffic regimes are estimated using local connectivity information: in particular, vehicles which have at least one neighbor in the message propagation direction (in the same road lane) are said to be in a *well-connected neighborhood* and the broadcast suppression technique is employed. For example, vehicles A, B, D and F in Figure 3.2 are in this status. If instead a node does not have a neighbor in the message propagation direction, but a neighbor going in its opposite direction, it is said to be in a *sparsely-connected neighborhood* (vehicles C and E in Figure 3.2), otherwise it is said to be in a *totally disconnected neighborhood* (vehicle G in Figure 3.2). In such cases the algorithm uses the *store-carry-forward* mechanism.

Other dissemination algorithms employ clustering techniques, i.e. they group vehicles into clusters. For example, Chang et al. [37] proposes the *TrafficGather* protocol, which however is not meant for broadcasting but for information gathering; a vehicle which wants to ask, for example, for traffic situation, must firstly run a network initiation phase, in which clusters are built. This vehicle takes the role of Cluster head (CV) (i.e., the principal node) and issues a Request Message (RM) to let neighbors know about the creation of the cluster. Each node that receives the RM automatically become a member of the cluster and contend with the other to take the role of Relay Vehicle (RV), which will have the duty of continuing the procedure; they compete by generating a unique waiting time which is inversely proportional to the distance from the CV. Then, node with the lowest waiting time will inform the other that it has become the RV by sending a Winning

Message (WM) when the timer expires. The RV will then forward the RM and its neighbors will restart the same race, but this time to become the CV of the successive cluster. To stop the procedure, the initiator can add into the RM a data collection range, to let vehicles outside this range ignore the message. When the network is initiated, the data collection phase starts; CVs issue a HELLO message to gather information from nodes in their clusters, which synchronize each other by using a slotted access mechanism (i.e., road slot plus lane number). After the reception of the information from all nodes, each CV sends the data toward the initiator using RV nodes. Figure 3.3 gives a graphical representation of how the protocol works.

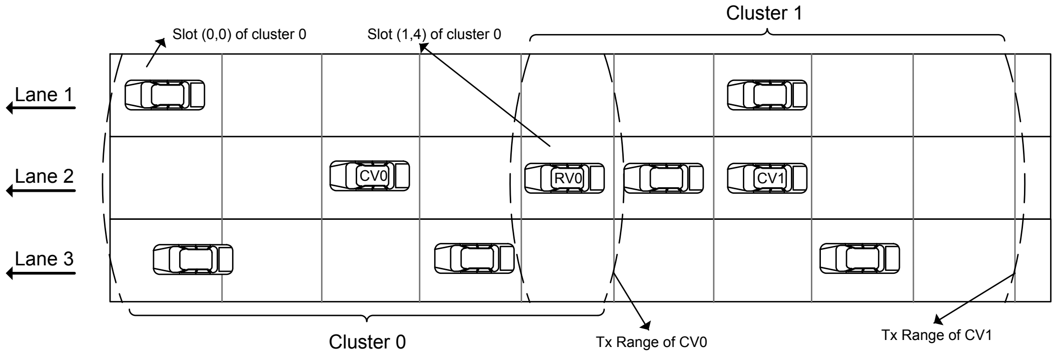


Figure 3.3: The clustering scheme of the *TrafficGather* protocol defined by Chang et al. [37].

There are some approaches that consider also the content of the message, i.e. they determine if received message is relevant (for the higher-level application) and if it should be forwarded. For example [38] defines a context-based broadcasting protocol, where hosts periodically broadcast Basic Safety Messages (BSMs) to inform neighbors of their current status (e.g., position, speed, heading, . . .). A vehicle receiving a BSM can decide whether contained information is relevant or not by evaluating a set of predefined conditions, which clearly are application-dependent.

A similar mechanism is proposed by Ducourthial et al. [39]. It defines the concept of *conditional addressing*, where the intended receiver of a message is not determined by its network address but instead by its condition, for example "*nodes behind the sender*" or "*nodes in a given area*". The protocol evaluates two conditions to decide whether the message should be passed to the upper layer (Upward Condition (CUP)) and whether it should be forwarded (Forward Condition (CFW)); as for Chisalita and Shahmehri [38] the conditions are application-defined. The routing layer, instead of forwarding a message to a precise address, forwards a message to the nodes which satisfy the CUP condition, which will be relayed by nodes which satisfy the CFW condition. The forward conditions can be derived by other works; for example, a condition such as "$\mathtt{rand}() < p$" will make the nodes behave as in the p-persistent flooding.

Another context-aware dissemination protocol has been described by Eichler et al. [40]; they introduce the concept of *message benefit* as the benefit the whole VANET could get from a particular message. Quantification of the benefit is obtained by considering:

- ***Message context*** m**:** this includes parameters like message age, time since last receiption, . . . ;
- ***Vehicle context*** v**:** information relative to the vehicle, like speed, driving direction, number of neighbors, . . . ;
- ***Information context*** i**:** for example, time of the day, travel purpose, ...

The three contexts m, v and i are then used to compute the message benefit as follows:

$$MessageBenefit = \frac{1}{\sum_{i=1}^{N} a_i} \sum_{i=1}^{N} a_i b_i(m, v, i)$$

The formula takes into account N parameters, like source of the message or message age: they can be derived from m, v and i and then evaluated by a set of application dependent subfunctions

$$b_i \colon M \times V \times I \to [0, 1] \qquad i = 1, \ldots, N$$

Clearly a parameter can have more importance than another, so weights a_i are used; the obtained value is then divided by the sum of the weights, so it is a simple weighted average. As a result, benefit value can range from 0 (no benefit) to 1 (maximum benefit). The paper then propose a modification of the MAC layer which takes into account message benefit value. This modification must be performed both for internal and external contention. Internal, because a packet with higher benefit should be processed before a packet with lower benefit, so the traditional FIFO dequeueing must be changed, but this is quite easy to achieve. External because if two vehicles wants to send a message at the same time, the one with the highest benefit should access the media for first, at least with higher probability. So the paper proposes to modify the calculation of the backoff timer of 802.11 in a benefit-based manner, as shown in Equation (3.1). Backoff value is calculated as a random value in a contention window which size depends on the message benefit, as shown by Equation (3.2): contention window can range from CW_{min} (31 slots) to CW_{max} (1023 slots). If the benefit is 0 then $CW = \text{CW}_{max}$, while if the benefit is 1 then $CW = \text{CW}_{min}$.

$$T_{Backoff} = (Rand() \bmod (CW_{bb} + 1)) \times SlotTime \tag{3.1}$$

$$CW_{bb} = ((1 - MsgBenefit) \times (CW_{max} - CW_{min})) + CW_{min} \tag{3.2}$$

It is also proposed and extension to enable a cross-layer communication; indeed, if a message remains in a queue at the MAC layer for a long time, it is possible that its benefit changes. So an Inter-Layer Communication module (ILC) permits the MAC layer to gather the parameters to recompute the benefit of enqueued messages from the application layer. As an alternative, the paper proposes to employ the 802.11e standard which, as described in Section 2.1.1, already includes a prioritization mechanism based on ACs, each of them having a separated queue. For example, packets with benefit from 1 to 0.75 could be inserted into first queue, packets with benefit from 0.75 to 0.5 in the second, and so on. However, the 802.11p standard (Section 2.2) which uses the same EDCA mechanism of 802.11e, is not mentioned by Eichler et al. [40].

The last three proposals only determine whether a message is relevant or not (and in which measure) and decide if it should be discarded or forwarded. No modification are performed on the original message such as integration of other information. Bronsted and Kristensen [41] introduces a data aggregation protocol derived from Wireless Sensor Networks (WSNs). Each node mantains an Environment representation (EM) which describes what a sensor know about sorrounding environment, and it is periodically broadcasted. When an EM is received it is aggregated to the EM of receiving node, according to a combination policy. In the paper, for example, it is presented a simple application for broadcasting road conditions; the street is divided in slots and the vehicles save the condition of each slot like *DRY* or *ICY*. A combination policy could be to set a slot as *ICY* if at least one vehicle announces it as *ICY*.

Another article about aggregation is the one by Ibrahim and Weigle [42]. The paper presents an aggregation scheme based on clustering, where data about a particular cluster is compressed via a delta encoding and compression algorithms. The delta encoding simply computes the average data for a cluster (i.e., average speed) and then computes, for each vehicle, the difference between the average and the actual vehicle data, in order to minimize the size of the packet. In the paper it is stated that the algorithm should provide good performances also for safety application, since the repetition time of safety messages is around 300-400 ms, which is not true by looking at U.S. DOT [6], where EEBL has a repetition time of 100 ms.

Congestion control can be also performed by adjusting transmission power, as described by Torrent-Moreno et al. [43]. However this approach is not so easy as it could seem: indeed a linear programming problem is employed in order to determine the transmission power to be used.

The complexity of most of the algorithms described till now is quite big, and moreover the eventual benefits to applications are not considered at all [33, 35, 36, 37, 43]: they all consider only network related issues. A little bit more of consideration is given to the application requirements when the broadcasting protocol use application-defined conditions to determine relevance of a particular message [38, 39, 40]. Again, no application is analyzed to determine the effectiveness of this protocols.

Opposed to the pure network-oriented literature, there are also pure application-oriented works as in van Arem et al. [8]: here the benefits of CACC on traffic flow characteristics are analyzed, but details about protocols, network load, etc. . . are missing.

Only recently the joint analysis has started being investigated. For example the paper by Zang et al. [44] analyzes the EEBL application similarly as performed in this thesis, but in a different way: no CACC is considered and the EEBL messages are only used to warn the driver, so the analysis is completely centered on humans and not on cooperative and automated driving. Moreover, message repropagation is not considered and the protocol presented stops broadcasting messages when EEBL packets are "heard" from following vehicles, meaning that they are already communicating the danger farther behind. A complete absence of messages could harm the CACC.

In favor of the importance of application requirements a paper appeared in a very recent conference [45]. In it, it is stated the need of taking into account the application when designing network protocols. Anyhow the analysis is not very complete because a bunch of applications has been considered, so the level of details of this thesis are not reached.

This chapter has given, togheter with a state-of-the-art vision about VANETs, the motivations of this work.

4 MOBILITY MODELS AND SIMULATIONS

Testing and performance evaluation in VANETs is a crucial aspect which, however, is not easy to perform in a real environment due to logistical difficulties and economic issues. So simulation is the way to go and to this purpose there is the need to model traffic mobility patterns to reproduce a real traffic scenario inside a simulated environment. A mobility model takes care of different aspects, that are:

- **trip:** it deals with modeling of the motion between points of interest by using an Origin-Destination matrix, which contains the probability of being directed to a particular destination given an origin;
- **path:** it deals with modeling of the path which a vehicle follows, which can be random or based on a trip model;
- **flow:** this aspect considers the interactions between vehicles, for example what happens at an intersection.

Notice that they are not always required togheter: for example, a safety application may require only a flow model, while trip and path modeling could be useful for traffic optimization applications.

Mobility models can be divided into five categories, depending on their scope:

- **random models:** in this category, traffic mobility is random and the parameters such as speed or heading are sampled from random processes. Limited interactions are considered;
- **flow models:** as said before they model traffic interactions, for example in a multi-lane highway following flow theory;
- **traffic models:** they include trip and path models;
- **behavioral models:** in this category human behavior is considered, so vehicles do not follow statically predefined rules but instead they react, for example using artificial intelligence concepts, in different ways depending on the situation;
- **trace-based models:** real traces can be used to simulate traffic scenarios.

For testing a VANET application there is the need of two simulators, one for mobility and one for networking and they clearly have to share information. The simulators can be:

- **isolated:** mobility pattern are statically generated and given to the network simulator, so no real interaction occures;
- **embedded:** a mobility simulator is implemented inside a network simulator, or viceversa. In this way they can easily interact and share information;
- **federated:** mobility simulator and network simulator are separated but they can communicate each other and their communications are managed by a third application.

A very detailed description of mobility models, network and traffic simulator is contained in the book by Hartenstein and Laberteaux [19], but to clarify the generic explanation given, imagine that the aim of a research work is to develop a VANET routing protocol for information diffusion among vehicles (e.g., inter-vehicle gaming). Diffused data do not modify driver behavior so a real traffic trace (i.e., a trace-based model) could be the correct mobility model to adopt: since it is a trace, it does not require computations and so the simulation would be fully focused on networking. Mobility traces could then be used within an isolated network simulator.

In the case of a safety application like EEBL instead, network and traffic simulator are tightly correlated, because network messages can modify the behavior of a vehicle (e.g., automatic application of brakes by the ACC) and conversely the behavior of a vehicle can modify the behavior of the network application (e.g., broadcast of EEBL messages if the deceleration exceeds a certain threshold). So in this case a flow model within an embedded or a federated simulator would fit the situation.

In this thesis the *NS-3*[1] network simulator (v. 3.9) has been used. A flow model[2] to simulate nodes moving in a highway has been embedded in *NS-3* [46].

4.1 NS-3 AND THE MOBILITY SIMULATOR

NS-3 is a descrete event simulator targeted primarily for research. When this thesis began, the available version was v. 3.9, which has been used to perform the simulations. The current version (at the time of writing) is v. 3.10 while v. 3.11

[1] `http://www.nsnam.org`

[2] Indeed a microscopic flow model, see Section 4.2 for details

is planned for release in spring 2011. The simulator is multi-platform, written in C++ and Python, it is free and licensed under the GNU GPLv2 license[3].

It is intended as a replacement of the old *NS-2* which was originally programmable in TCL. Version 3 is only programmable in C++, so instead of writing a description of the scenario and then run the simulation, the scenario is written as a C++ program, compiled and then executed, which makes it very easy to use (if C++ is known by the user).

NS-3 includes a re-implementation of a bunch of network protocols, mainly for data link, network and transport layers. For example, different 802.11 MAC protocols are available, as well as IP and different versions of TCP. All these protocols and other useful functionalities can be used in a object-oriented fashion.

Since the mobility simulator has been embedded in *NS-3*, they do not need to interact using complicated interfaces, but they can "talk" directly. Figure 4.1 shows a simplified schematization of the whole simulator: the mobility simulator moves the nodes and changes their states following the rules given by the mobility model. Then, user-defined application protocols can use nodes information and, if needed, send a message. For example, the EEBL protocol will send a packet if the deceleration of a vehicle is greater than a certain threshold. Then the network simulator will emulate lower layer protocols behavior (e.g., 802.11 MAC and packet loss) and will deliver the data to receiving nodes. Finally, the application layer protocol will modify the state of the node which will in turn modify the behavior of the mobility simulator. For example, received packet can update CACC data which could make the car decelerate to avoid a collision.

So the user defined protocol acts as "glue" between network and mobility simulator but in a very simple way, because the interaction is performed using normal C++ method invocation. The following sections describe in details the mobility model which makes nodes move as vehicles in a real highway.

4.2 INTELLIGENT DRIVER MODEL

The Intelligent Driver Model (IDM) is one of the most popular vehicle mobility models that can be found in the literature. It has been presented in the paper by Treiber et al. [47] and it is a microscopic flow model. Microscopic means that the behavior of every single vehicle is modelled, so for every car in the simulation the model controls position, speed and acceleration. Conversely, a macroscopic flow model controls entire flows of vehicles: the idea is, for every road segment x, to keep track of the density $\rho(x,t)$, the velocity $\upsilon(x,t)$ and the flow $m(x,t)$ during

[3] http://www.gnu.org/copyleft/gpl.html

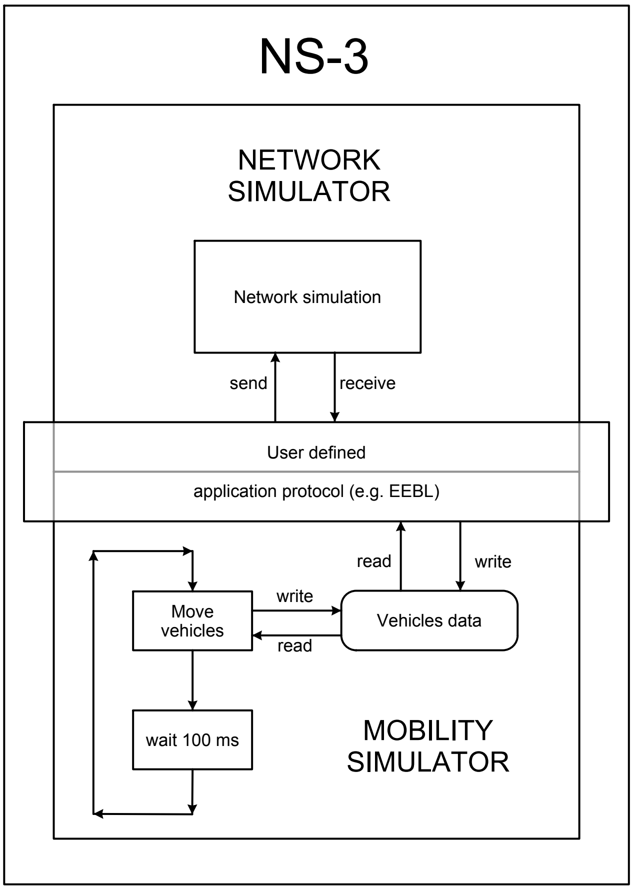


Figure 4.1: Structure of the simulator

the evolution of time (t). Clearly a model like this would be unsuitable to the purposes of this thesis, since there is the need to control the behavior of every single vehicle. Macroscopic models are best suited for testing, for example, if adding a road on a city can improve the overall road system, where potentially thousands of vehicles must be taken into account; the macroscopic characterization of the traffic results indeed in a much lower computational cost. Further details are given in Hartenstein and Laberteaux [19].

IDM belongs to the family of so called *Car Following Models*, where the idea is to model the behavior of a driver by means of a set of rules developed in order to avoid any collision with leading vehicle; so IDM in its original formulation is unsuitable to test the effectiveness of an EEBL system, since crashes will never occur even if physics laws are violated.

The idea of IDM is to take into account two aspects of a driver: the tendency to accelerate in order to reach a desired speed and the tendency to decelerate due to the interaction with leading vehicle. The part of the acceleration describing tendency to reach desired speed is defined as the *free road term* α^{free}:

$$\alpha^{free} = a \left[1 - \left(\frac{v_i}{v_i^{des}} \right)^{\delta} \right]$$

where a is the maximum acceleration, v_i is the current vehicle speed, v_i^{des} is the desired speed and δ is called the acceleration exponent, which models the "slope" of the curve as v_i approaches v_i^{des}. Look at Figure 4.2 to understand what happens for different values of δ. Usually the δ parameter is set to 4.

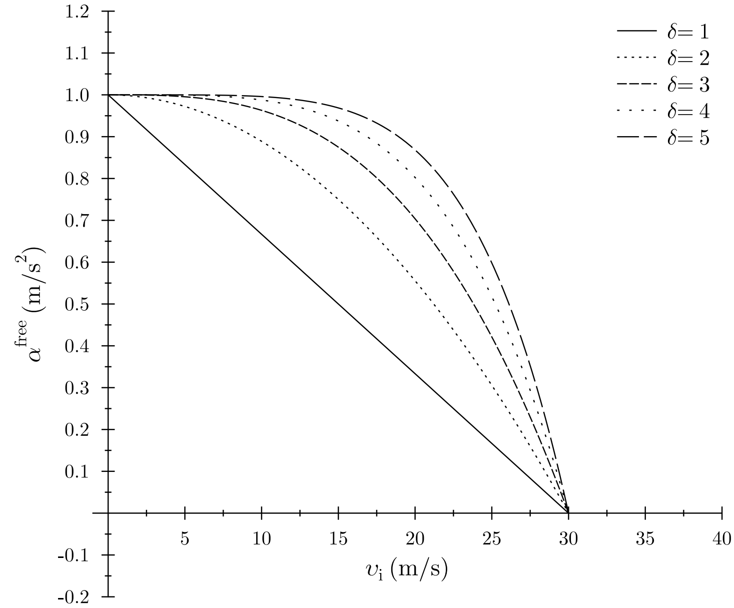


Figure 4.2: Plot of the α^{free} term of the IDM model. Parameters of this plot are $a = 1m/s^2$ and $v_i^{des} = 30m/s$

The part of the acceleration describing the deceleration caused by the interaction with leading vehicle is defined as the *interaction term* α^{int}:

$$\alpha^{int} = -a \left(\frac{s^{des}(v_i, \Delta v_i)}{s_i} \right)^2$$

where Δv_i is the difference of speed between vehicle i and leading one (called also approaching rate), $s^{des}(v_i, \Delta v_i)$ is the desired gap between the two vehicles and s_i the actual gap. Clearly s^{des} is a function of current speed and approaching rate: the higher the speed, the higher must be the safety distance. Moreover, if the approaching rate is high, the model should compute a high deceleration. s^{des} is defined as

$$s^{des}(v_i, \Delta v_i) = s_0 + T \cdot v_i + \frac{v_i \Delta v_i}{2\sqrt{a \cdot b}}$$

where s_0 is the gap when vehicles are stopped, T is the so called *safe time headway* which is simply the time which must elapse between leading vehicle and current one and b is the desired deceleration. The product $T \cdot v_i$ determines the space that is needed in order to have T seconds of time gap between the vehicles at a speed v_i. So in some sense T characterizes the driving style: the lower T, the more the driver will be "aggressive". Clearly there are other parameters which characterize the style, like desired speed or desired deceleration.

So to finally compute the acceleration of a vehicle at a certain time t we combine the free road term and the interaction term to obtain

$$a(t) = a\left[1 - \left(\frac{v_i(t)}{v_i^{des}}\right)^{\delta} - \left(\frac{s^{des}(v_i(t), \Delta v_i(t))}{s_i(t)}\right)^2\right]. \tag{4.1}$$

The model can be analyzed in four cases to understand how it works:

- **equilibrium behavior**: with equilibrium traffic (i.e., when $a(t) = 0$ and $\Delta v_i = 0$) drivers maintain a constant gap $s_i(t)$ which depends on $v_i(t)$. This gap can be obtained by substituting $a(t) = 0$ and $\Delta v_i = 0$ in Equation (4.1):

$$s_i(t) = \frac{s^{des} + T \cdot v_i(t)}{\sqrt{1 - \left(\frac{v_i(t)}{v_i^{des}}\right)^{\delta}}}. \tag{4.2}$$

  The constant gap is "associated" with an equilibrium speed which can be obtained from Equation (4.2) by choosing a particular value of δ. A simple solution can be obtained by setting $\delta = 2$ and $s_0 = 0$:

$$v_i(t) = \sqrt{(v_i^{des})^2 + \left(\frac{s_i(t)}{T}\right)^2};$$

- **free road behavior**: in this case, the distance s_i between vehicles is very high, so the interaction term α^{int} becomes negligible and the car freely accelerates to reach desired speed;

- **high approaching rate behavior**: in this case, Δv_i is high, causing the $\frac{v_i \Delta v_i}{2\sqrt{a \cdot b}}$ part to dominate in the α^{int} term, which becomes $-\frac{(v_i \Delta v_i)^2}{4 \cdot b \cdot s_i^2}$. To understand what changes by varying value of b, Figure 4.3 shows some examples: when $|b|$ is low (i.e., smoother deceleration), the model reacts earlier with a stronger deceleration at a higher gap. Conversely, when $|b|$ is high, the model tends to brake later, resulting in stronger decelerations when the gap becomes small;
- **small net distance behavior**: when inter-vehicle distances are small and the difference of speed is negligible (i.e., $\Delta v_i \simeq 0$) the α^{int} term reduces to $-a\frac{(s_0 + T \cdot v_i)^2}{s_i^2}$ which emulates a Coulomb-like repulsion which results in a oscillatory behavior of speed and inter-vehicle gap, until an equilibrium is reached. The amplitude of the oscillations depends on the value of b: the stronger the deceleration, the greater would be the oscillation amplitude.

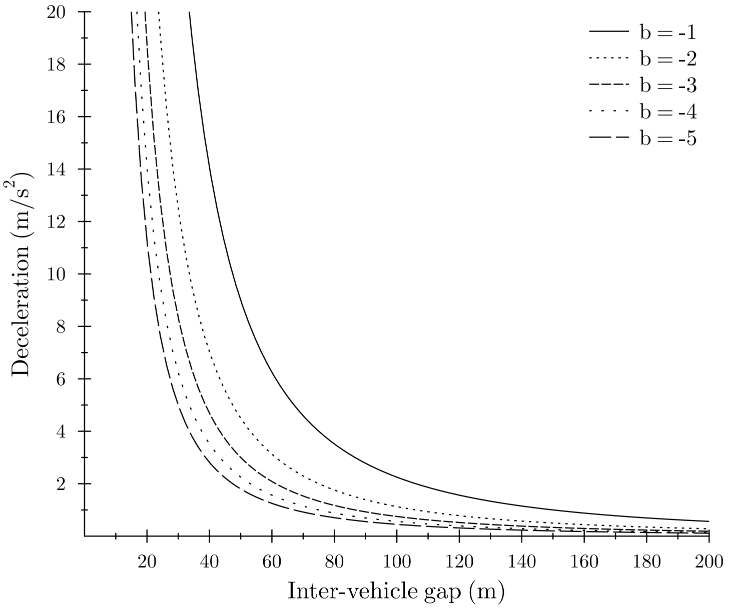


Figure 4.3: Graphical representation of the α^{int} term in the case of a high approach rate for different values of b. In this plot $\Delta v_i = 10m/s$ and $v_i = 30m/s$. Notice that the value of α^{int} is expressed as a deceleration, so it is positive (i.e., a negative acceleration)

Another important thing that it is possible to observe from Figure 4.3 is the violent deceleration that the model can provide if the inter-vehicle gap becomes small. Indeed, as mentioned before, the parameter b is the "comfortable deceleration" and not the "maximum deceleration", which means that vehicles can brake harder than b m/s^2. The model results in crash-free scenarios but sometimes with very strong decelerations, which are unrealistic. During preliminar simulations to test how the IDM model behaves, decelerations much larger than 1 g (i.e., 9.81 m/s^2) were detected, which are not reachable by "normal" production cars. By looking at tests performed in Michigan State Police [48] (on police cars) and in Schultz and Babinchak [49] (on production cars), the average maximum deceleration obtained (on dry surface) is 9.48 and 8.4 m/s^2 respectively.

Moreover in this model the driver should not be called "intelligent", but instead "perfect", since it uses the exact values of distance and approaching rate. So the same inventors of IDM have developed the Human Driver Model (HDM) [50], which takes into account reaction times, estimation errors and anticipation (i.e., a driver which does not look only at leading car but also several vehicles ahead). Unfortunately it is not developed inside the simulator but anyhow it is complicated and unnecessary for the purposes of this thesis.

To simulate crashes as they indeed occur, the only parameter that needs to be introduced is vehicle maximum deceleration b_i^{max} using as a reference the average values obtained in Schultz and Babinchak [49] and so modifying the IDM formula as follows:

$$a(t) = MAX(-b_i^{max}, a^{idm}(t)). \tag{4.3}$$

Equation (4.3) has been used in the simulations to produce crashes; the formula mimics the situation in which a driver reacts too late to a dangerous situation and the inter-vehicle space is not enough to avoid a collision.

4.3 MOBIL LANE CHANGE MODEL

The IDM model determines the acceleration needed in order to reach a desired speed without coming too close to the vehicle in front. This is not enough if we want to emulate a real highway because vehicles would remain always in the same lane in which they have ben placed at the beginning of the simulation. So there is the need to model lane change and let cars compute overtakes when front vehicle is too slow.

To this purpose, the MOBIL microscopic lane change model has been developed: its acronym stands for Minimizing Overall Braking Induced by Lane change and it is presented in the paper by Treiber and Helbing [51][4]. Basically, the model computes the advantage that a vehicle gains by changing lane and compares it

[4] The original paper is in German, but an English version has been written by Kesting et al. [52]

with the gain loss caused to vehicles in the destination lane. If the own advantage is much higher than the disadvantage caused to other vehicles, the lane change is performed.

More formally, lane change can take place when the following disequation holds:

$$\underbrace{\tilde{a}_i + p \cdot (\tilde{a}_f^{old} + \tilde{a}_f^{new})}_{\text{situation after lane change}} > \underbrace{a_i + p \cdot (a_f^{old} + a_f^{new})}_{\text{current situation}} + \delta_{thr} \tag{4.4}$$

where $\tilde{a}_i$ and a_i are the accelerations (computed with the IDM formula) of the vehicle which evaluates the need of changing lane, after and before the change respectively. Similarly, these values are computed for the current following vehicle ($\tilde{a}_f^{old}$ and a_f^{old}) and for the new ($\tilde{a}_f^{new}$ and a_f^{new}). Figure 4.4 gives a graphical representation of such quantities. The politeness factor p is introduced to model the aggressiveness of the driver. If $p = 0$ Disequation (4.4) reduces to $\tilde{a}_i > a_i + \delta_{thr}$, so driver considers only its own advantage. Conversely, the higher p becomes, the more the driver will consider the disandvantage caused to other drivers. By setting p as a negative value we would obtain a "crazy" driver which wants to damage the others even at its own disadvantage. Finally the δ_{thr} parameter is a threshold used to ensure that there is a minimum gain, to avoid drivers "jumping" from a lane to another too frequently, which is unrealistic.

The model considers also two safety conditions to ensure that the incoming driver must not brake too hard due to the vehicle which moves to its lane. The first condition is

$$\tilde{a}_f^{new} > -b_{safe}$$

where $\tilde{a}_f^{new}$ has the same meaning as in Disequation (4.4) and b_{safe} is the maximum deceleration considered as safe. The second condition simply checks that the new inter-vehicle gaps after the lane change are greater than a certain minimum:

$$s_i^{new} > s_{min} \quad \wedge \quad s_{i-1}^{new} > s_{min}.$$

Here s_i^{new} and s_{i-1}^{new} are the gaps between the vehicle and its new leader and its new follower respectively while s_{min} is clearly the minium gap to be respected.

Notice that, till now, the model makes no differences between the lane change to the right and the lane change to the left. If the highway uses a "keep-right" rule then, after an overtake, a vehicle should move back to the lane it was before.

This behavior can be easily obtained by differentiating δ_{thr} for the left-to-right and the right-to-left lane change, in particular by introducing a parameter δ_{bias} and modifying Disequation (4.4) as follows:

$$\tilde{a}_i + p \cdot (\tilde{a}_f^{old} + \tilde{a}_f^{new}) > a_i + p \cdot (a_f^{old} + a_f^{new}) + \delta_{L \to R} \tag{4.5a}$$

$$\tilde{a}_i + p \cdot (\tilde{a}_f^{old} + \tilde{a}_f^{new}) > a_i + p \cdot (a_f^{old} + a_f^{new}) + \delta_{R \to L}. \tag{4.5b}$$

Disequation (4.5a) models the lane change from left to right while Disequation (4.5b) the change from right to left. The thresholds $\delta_{L \to R}$ and $\delta_{R \to L}$ are computed as $\delta_{thr} - \delta_{bias}$ and $\delta_{thr} + \delta_{bias}$ respectively.

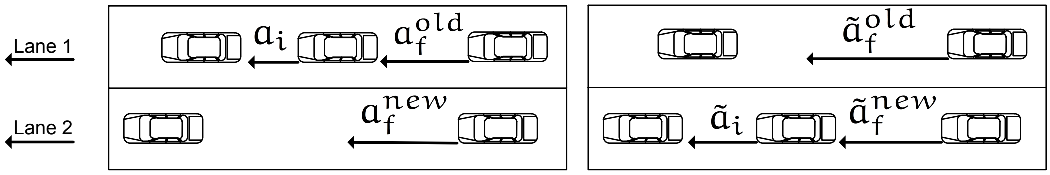


Figure 4.4: The MOBIL lane change model. It compares the gain a vehicle has on its lane (situation on the left) with the gain it earns by changing lane (situation on the right)

Different modifications to the original model can be found in order to reproduce different highway rules or drivers behavior [52]. In particular, the implementation by Arbabi and Weigle [46] ignores the "current follower" part of Disequation (4.5) resulting in

$$\tilde{a}_i - a_i + p \cdot (\tilde{a}_f^{new} - a_f^{new}) > \delta_{L \to R} \tag{4.6a}$$

$$\tilde{a}_i - a_i + p \cdot (\tilde{a}_f^{new} - a_f^{new}) > \delta_{R \to L}. \tag{4.6b}$$

The "semantic" of this modification is different for Disequation (4.6a) and Disequation (4.6b). In particular when changing from right to left, ignoring current follower is quite correct because it will always gain an advantage while, when changing from left to right, ignoring current follower means not moving to right lane to let fast and pushy drivers compute their overtake[5]. This could be considered correct, for example, for simulating U.S. highways where overtakes can be performed on both sides. For simulation of European highways instead, it could be better to consider also the current follower.

In conclusion, the MOBIL implementation by Arbabi and Weigle [46] considers a highway with a "keep right" rule where overtakes can be performed on both sides.

[5] Remember that Disequation (4.6) models an highway with a "keep right" rule

4.4 CHANGES IN THE MOBILITY SIMULATOR

For the purposes of this thesis the original mobility simulator developed by Arbabi and Weigle [46] had to be modified and extended. For example, the standard used in the original simulator was 802.11a, while for VANET simulation 802.11p must be employed. The following sections describe the changes.

4.4.1 Crash simulation and management

First of all the simulator had to take into account car crashes. Collisions are allowed by introducing the more realistic Equation (4.3) in the computation of vehicles acceleration. Then there was the need to manage the collisions, which are detected by computing the distance between each pair of consecutives vehicles: if the distance is negative an accident has occured.

Listing 4.1 shows how the behavior of the vehicles changes after the crash has been detected. First of all the front vehicle is moved ahead of a quantity equal to their distance: in this way the cars are no more "overlapped". Then the speed is set for both as the average of their speed. Finally there could be two cases, i.e. the front vehicle has or has not a stronger deceleration w.r.t. the following vehicle. If the front vehicle has not a stronger deceleration than the other, then they will continue with the same deceleration they had before the crash. In the other case instead, the following vehicle will push the leader and will remain "glued" to it: so their acceleration is set as the average acceleration. For simplicity Listing 4.1 shows the procedure only for two cars but in the simulator the procedure is iterative and it is able to manage a pile-up.

4.4.2 Accelerometer and imperfect clock

Each vehicle has been equipped with an accelerometer, which basically measures the change of speed every 100 ms. This is more realistic than taking the perfect acceleration computed with the IDM formula. Moreover with the accelerometer it is possible to see in the log traces the peaks of acceleration caused by crashes, as shown in Chapter 5.

Another equipment that has been implemented is an imperfect clock, used to avoid the occurrence of events at the same exact time, which is unrealistic and causes the simulation to generate synchronization phenomena. The clock drift has been set as a very small value, i.e. 1 minute in a year. Moreover, *NS-3* does not consider processing delays, and again this could lead to unrealistic

```
distance = DISTANCE(vehicle, front)

if (distance< 0) then

  front.SETPOSITION(front.GETPOSITION() − distance)

  avgSpeed = (vehicle.GETSPEED() + front.GETSPEED()) / 2

  front.SETSPEED(avgSpeed)
  vehicle.SETSPEED(avgSpeed)

  if (front.GETACCELERATION() < vehicle.GETACCELERATION()) then

    avgAcc = (vehicle.GETACCELERATION() + front.GETACCELERATION()) / 2

    front.SETACCELERATION(avgAcc)
    vehicle.SETACCELERATION(avgAcc)

  end if

end if
```

Listing 4.1: Crash management

synchonizations. So every send and receive event has been delayed by a random time between 0 and 10 µs.

4.4.3 CACC

Another fundamental component to implement was a CACC, which has the duty of processing data received from other vehicles to keep safety distance and brake if the driver does not react in time to a dangerous situation. As described in the introduction, the CACC considered in this thesis does not implement a car following procedure. The CACC is very simple: to develop a realistic CACC is not the aim of the thesis.

First of all, if the approaching rate is negative (i.e., leading vehicle is travelling faster than its follower), then the vehicles are automatically increasing their gap, so the CACC remains disabled[6].

If instead the approaching rate is positive, the CACC must compute the safety gap to determine if a vehicle is too close to its leader. Safety gap clearly depends on the speed of the vehicle, so it is computed as

$$s_{safe} = T_{cacc} \cdot v_i + \varepsilon_{cacc}$$

where T_{cacc} is the time headway for the CACC, v_i is the current vehicle speed and ε_{cacc} is a small quantity to account for errors. T_{cacc} has been set to 1 s which, as mentioned in the paper by Treiber et al. [50], is an average time measured on German freeways which is considered safe. ε_{cacc} has been set to 1 m.

The actual gap between the car and leading vehicle can be greater or lower than the safety gap. If it is lower then the follower must brake in order to move away from leader. The applied deceleration can be computed from leader acceleration minus a small quantity. If instead the actual gap is greater than the safety gap the CACC computes a deceleration using the following formula:

$$a^{cacc} = \frac{v_{i+1}^2 - v_i^2}{2 \cdot (s_{actual} - s_{safe})}$$

which computes the acceleration needed to bring the speed of following vehicle (v_i) to the speed of its leader (v_{i+1}) in a space equal to the difference between the actual gap (s_{actual}) and the safety gap s_{safe}).

Notice that if the driver (i.e., the IDM model) decides to brake harder than the CACC, the latter will let the car decelerate as the driver chooses. As a final and important feature, the CACC will not perform any operation if data about front vehicle is outdated more than three seconds.

4.4.4 Air resistance induced deceleration

CACC has been designed to work only with messages regarding front vehicle, but a way to evaluate the benefits of repropagated EEBL messages was needed. The idea here was to use EEBL messages coming from ahead vehicles as warnings directed to the driver: when received, the simulator reacts as if the driver removes

[6] In fact even in this situation if the gap is very small due, for example, to a sudden change of lane, the CACC should brake a little bit in order to increase the gap as soon as possible, but again the development of a perfect CACC is not the purpose of this thesis

his foot from the accelerator pedal, resulting in a deceleration induced by the air resistance. If no EEBL messages are received within two seconds, this mechanism is disabled and the vehicle returns to obey the IDM formula. The drag force can be computed using

$$F_{drag} = \frac{1}{2}\rho \upsilon^2 C_D A. \tag{4.7}$$

Equation (4.7) is known as the drag equation [53] and computes the force a body with a section A and a drag coefficient C_D is subjected if it runs at a speed υ in a fluid of density ρ. Final deceleration has been computed as a function of speed

$$b_{air} = \frac{F_{drag}(\upsilon)}{1500\text{kg}}$$

using $\rho = 1.20$ kg/m^3 (air at 20°C), for a vehicle weight of 1500 kg. The $C_D A$ product is randomly set for each vehicle using values for common production cars found on an unofficial site [54]. As a result the air drag deceleration can result in a maximum of 1 m/s^2 for a speed of 40 m/s (roughly 150 km/h).

This is a simplification of a complex human behavior which could be enhanced giving a weight to the warning according, for example, to the distance. Anyhow this is left as a future work.

4.5 NETWORK PROTOCOLS

The EEBL application clearly requires the definition of some simple network protocols to diffuse information among vehicles. The first one is a beaconing protocol: this is very simple but fundamental to give the CACC "an idea" about nearby vehicles, in particular about the front car. Without beacon messages there could be the possibility that a driver gets so close to its leader that, if the latter starts braking, even the CACC would not be able to avoid a crash. With beacons instead, the CACC can make the car respect the safety distance and easily avoid a crash even in the case of a violent braking. Notice that every car which is equipped with the EEBL application is also equipped with the CACC.

The beaconing protocol simply broadcast messages at a frequency of 1 Hz: messages must contain informations about sender vehicle, such as speed, acceleration, position, etc.... The same packet definition has been used for beacons and EEBL messages, following the indications of U.S. DOT [6]: Table 4.1 shows the fields.

Moreover, a header has been defined to meet other application requirements, for example sender identification. Table 4.2 lists the fields:

TYPE determines if the packet is a beacon, an EEBL message or an aggregated EEBL message;

PACKET ID is used as a unique identificator for the packet for statistical purposes. In a real scenario it could be thought as a sequence number;

ORIGINATOR ID is the id of the vehicle which has created the message, or better, the vehicle that the packet describes;

TTL is the time-to-live of the packet in hops. Initially it is set to 0 for beacons and for EEBL messages which must not be rebroadcasted, otherwise it is set as 5;

SENDER ID is the id of the vehicle which has actually sent the message. In the case of a rebroadcasted EEBL message, it is different from the originator id;

COUNT tells, in the case of an aggregated EEBL message, how many packets have been inserted;

CERTIFICATE is the digital certificate of the originator, for authentication purposes[7]. In the simulation it is not used, but it must be included in order to properly analyze network load. The size of 58 bytes is taken from Ibrahim and Weigle [42];

DIGITAL SIGNATURE is the signature of the originator. As for the certificate, it is not actually used in the simulation and its size is taken from Ibrahim and Weigle [42].

The EEBL protocol is very simple. It works exactly as the beacon protocol, but message frequency is 10 Hz as indicated by U.S. DOT [6]. Switching between beaconing and EEBL protocol happens when the deceleration of the vehicle is stronger than 1 m/s^2 (i.e., an acceleration of -1 m/s^2). Moreover, packets coming from behind vehicles or from vehicles running in the opposite direction[8] are ignored.

Slightly more complicated is instead the EEBLR protocol, in which more application logic must be used as shown in Listing 4.2. First of all, when a packet is received, the protocol must determine if it the first time it sees the packet: to

[7] In the case of rebroadcast a single certificate (with a single signature) is not enough because the receiver is interested in the "full chain", from the originator to the last rebroadcaster. However, security in VANETs is a problem in its own and the development of a secure protocol goes beyond the scope of this thesis

[8] The two-directional highway has not been considered but this simple control has anyhow been implemented

Description	Size (bit)	Size (byte)
GPS coordinates	96	12
Time stamp	64	8
Vehicle speed	16	2
Vehicle acceleration/deceleration	16	2
Vehicle heading	16	2
Vehicle size (length, width, height)	48	6
GPS antenna offset (relative XYZ)	32	4
Total	288	36

Table 4.1: Required message data set for EEBL application, as suggested by U.S. DOT [6]

Description	Size (bit)	Size (byte)
Type	8	1
Packet id	32	4
Originator id	32	4
TTL	8	1
Sender id	32	4
Count	8	1
Certificate	464	58
Digital signature	224	28
Total	808	101

Table 4.2: Additional header for beacons and EEBL packets

this purpose a list of known packets must be mantained. If the protocol determines the packet is not "new" then it must be ignored, since the application has already processed and potentially rebroadcasted it. If the packet is instead "new" it is inserted in the list of known, passed to the application for processing and, if needed, rebroadcasted. Clearly, if the TTL value is 0, it is not rebroadcasted, otherwise the rebroadcast criterion (i.e., the probability p) is computed using a *weighted p-persistence* mechanism [28].

The EEBLR has the problem of high redundancy of the same message (i.e., same packet id and same TTL), due to the fact that, once a packet has been scheduled for rebroadcast, it cannot be removed from the MAC queue without performing a layering violation. So if a rebroadcaster receives the packet from another one, it will unnecessarily send it, wasting resources. The EEBLA protocol is similar to EEBLR but addresses its problems. The logic of the protocol is shown in Listing 4.3.

```
list KnownPackets

on InitProgram():
  KnownPackets ← $\emptyset$

on ReceiveEEBLPacket(eebl):
  if (KnownPackets.Contains(eebl)) then
    return
  else
    KnownPackets.Insert(eebl)
    ProcessPacket(eebl)
    if (eebl.ttl $\neq$ 0)
      p ← ComputeRebroadcastProbability(eebl)
      if (Random() < p)
        eebl.ttl ← eebl.ttl − 1
        Broadcast(eebl)
      end if
    end if
  end if
```

Listing 4.2: The EEBLR protocol

The idea is to use a queue for packets which is emptied every 100 ms. With this mechanism, since the packets are queued at the application layer, useless rebroadcast can be avoided. Moreover, if more messages are contained in the queue, a single packet can be sent, reducing overhead.

The pseudocode of EEBLA is similar to the one in Listing 4.2 with some changes. First of all, upon reception of an aggregated EEBL message, every single packet is processed by the ReceiveEEBLPacket procedure. If the packet is known, before being ignored, it is removed from the send queue if present, since someone else has already rebroadcasted it. If instead the packet is not known and the rebroadcast criterion is satisfied (the same of EEBLR) it is inserted into the send queue, which is emptied every 100 ms by the SendPackets procedure.

```
list KnownPackets
list SendQueue

on InitProgram():
   KnownPackets $\leftarrow \emptyset$
   SendQueue $\leftarrow \emptyset$
   ScheduleEvent(SendPackets, 100ms)

on ReceiveAggregatedEEBLPacket(aggregatedeebl):
   foreach (eebl in aggregatedeebl)
      ReceiveEEBLPacket(eebl):

on ReceiveEEBLPacket(eebl):
   if (KnownPackets.Contains(eebl)) then
      if (SendQueue.Contains(eebl))
         SendQueue.Remove(eebl)
      end if
      return
   else
      KnownPackets.Insert(eebl)
      ProcessPacket(eebl)
      if ($eebl.ttl \neq 0$)
         p $\leftarrow$ ComputeRebroadcastProbability(eebl)
         if (Random() $<$ p)
            $eebl.ttl \leftarrow eebl.ttl - 1$
            SendQueue.Insert(eebl)
         end if
      end if
   end if

on SendPackets():
   if (SendQueue.Size() $\neq 0$) then
      if (SendQueue.Size() $= 1$) then
         packet $\leftarrow$ SendQueue.Get(0)
      else
         packet $\leftarrow$ CreateAggregatedPacket(SendQueue)
      end if
      Broadcast(packet)
      SendQueue.Empty()
   end if
   ScheduleEvent(SendPackets, 100ms)
```

Listing 4.3: The EEBLA protocol

5 PERFORMED TESTS

The tests that have been performed for EEBL protocol and application analysis reproduce the scenario in which the leader of a platoon of vehicles performs a complete stop (e.g., when an accident occures on the highway). Basically the leader (or the leaders, in the multi-lane scenario) performs a brake with a constant deceleration of 4 m/s^2. Different kind of simulations have been done in order to analyze different aspects of the EEBL application, for example network load and the reduction of the number of crashes.

Tests can be divided into five big classes, that are the models that have been tested:

PURE IDM which reproduces the scenario with the IDM model as originally described by Treiber et al. [47]: cars use no VANET technologies. This is important to show how violent and unrealistic the deceleration of a vehicle under the IDM model can be;

LIMITED IDM which uses the IDM model with the maximum deceleration, as described in Section 4.2, Equation (4.3). In this way it is possible to have an idea of the number of crashes that occure. As for pure IDM, VANET technologies are not employed;

EEBL test uses beaconing and EEBL protocol without any kind of rebroadcast. The maximum deceleration for IDM has been used;

EEBL WITH REBROADCAST test uses beaconing and EEBLR protocol;

EEBL WITH AGGREGATION test uses beaconing and EEBLA protocol.

What have been investigated are first of all the benefits an EEBL system can provide if all vehicles are equipped with this VANET technology. This test has been performed with a single lane, fifty vehicles and six different average speeds that are 13.88, 19.44, 25, 30.55, 36.11, 41.66 m/s (50, 70, 90, 110, 130, 150 km/h). Each test has been repeated twenty times with different initial conditions. What is analyzed here are the percentages of cars crashed, the average maximum decelerations, the traces in time of the accelerations and the network load to initially identify the differences between the EEBL protocols.

The first test has then been extended to a multi-lane scenario and it has been investigated by the point of the network, analyzing the differences between EEBLR and EEBLA. Tests have been performed with 1, 2, 3, 4 and 5 lanes, fifty vehicles per lane at an average speed of 36.11 m/s (130 km/h). Each test has been repeated twenty times.

Second a market penetration rate analysis has been performed. The penetration rate indicates the fraction of vehicles that are equipped with the EEBL application. The aim is to determine if it is possible to gain benefits even if not all vehicles use this technology because, clearly, it will not be deployed in a single day worldwide. As before, tests are performed on a single lane with fifty cars at six different average speeds. Penetration rates used are 0%, 10% to 40% in steps of 2%, and 50%. This time each single test have been repeated thirty times to have a more precise indication of the percentage of crashes. Accidents have been investigated by comparing the results obtained with EEBL, EEBLR and EEBLA and by analyzing the differences between equipped and unequipped vehicles.

Finally the market penetration rate analysis has been done in the multi-lane scenario, with 2, 3, 4 and 5 lanes at an average speed of 36.11 m/s (130 km/h). Penetration rates have been reduced to 10, 20, 25, 30, 35, 40 and 50% due to the high computational requirements of the multi-lane simulation. For the same reason, each test has been repeated twenty times instead of thirty.

5.1 SIMULATION PARAMETERS

To perform all the simulations the same set of parameters has been used. Table 5.1 lists the values for IDM parameters. Cars maximum deceleration (when used) has been set randomly in the interval $[5.9, 8.4]$ obtained by U.S. DOT [6] in dry surface conditions. The desired speed has also been set randomly as the average speed of the simulation $\pm$ 15%. Desired deceleration has been set to -4 m/s^2, which is quite a big value but the aim is to verify what happens in particularly dangerous situation, and so also T ranges randomly between 0.1 (very aggressive driver) and 1.1 s (safe driver). Finally, s_0 and δ has been taken from the original IDM paper [47].

The second set of parameters (shown in Table 5.2) is for MOBIL. The politeness factor p ranges randomly between 0 (totally impolite) and 0.5 (very polite). The b_{safe} parameter has been set so that 7 m/s^2 is the maximum deceleration a driver can cause to incoming vehicles by changing lane. Minimum gap s_{min} has been set to 2 m, δ_{thr} to 0.3 m/s^2 and δ_{bias} to 0.2 m/s^2. So $\delta_{R \to L}$ and $\delta_{L \to R}$ were 0.5 and 0.1 m/s^2 respectively.

Parameter	Value	Unit
b_{max}	$[5.9, 8.4]$	m/s^2
v^{des}	$\bar{v} \cdot [0.85, 1.15]$ ($\pm 15\%$)	m/s
a	1.7	m/s^2
b	-4	m/s^2
T	$[0.1, 1.1]$	s
s_0	2	m
δ	4	#

Table 5.1: IDM parameters used in simulations

Parameter	Value	Unit
p	$[0, 0.5]$	#
b_{safe}	7	m/s^2
s_{min}	2	m
δ_{thr}	0.3	m/s^2
δ_{bias}	0.2	m/s^2

Table 5.2: MOBIL parameters used in simulations

Final parameters needed are the ones related to the wireless communications, which are listed in Table 5.3. Clearly, the 802.11p standard has been selected and two different ACs have been used for beacons and EEBL messages, i.e. AC_BK and AC_VO respectively. Selected datarate is 6 Mbps with a bandwidth of 10 MHz (as 802.11p mandates) and a transmission power of 20 dBm. To model the propagation loss, the "three log distance" of *NS-3* with default parameters has been used[1]. This model computes the loss of power as a function of the distance: in particular the loss is logarithmic w.r.t. the distance, computed as

$$L = L_0 + 10 \cdot n_0 \log_{10}\left(\frac{d}{d_0}\right) (dB)$$

where L_0 (dB) is the path loss at reference distance d_0 (m), d (m) is the distance and n_0 (unitless) is the path loss exponent.

[1] http://www.nsnam.org/doxygen-release/classns3_1_1_three_log_distance_propagation_loss_model.html

The three log model uses three different distance fields resulting in the following final formulation:

$$L = \begin{cases} 0 & d < d_0 \\ L_0 + 10 \cdot n_0 \log_{10}\left(\frac{d}{d_0}\right) & d_0 \leqslant d < d_1 \\ L_0 + 10 \cdot n_0 \log_{10}\left(\frac{d_1}{d_0}\right) + 10 \cdot n_1 \log_{10}\left(\frac{d}{d_1}\right) & d_1 \leqslant d < d_2 \\ L_0 + 10 \cdot n_0 \log_{10}\left(\frac{d_1}{d_0}\right) + 10 \cdot n_1 \log_{10}\left(\frac{d_2}{d_1}\right) + 10 \cdot n_2 \log_{10}\left(\frac{d}{d_2}\right) & d_2 \leqslant d \end{cases}$$

where d_0, d_1 and d_2 (m) are the three distance fields, d (m) is the distance, L_0 (dB) is the path loss at reference distance and n_0, n_1 and n_2 (unitless) are the path loss exponents for the fields.

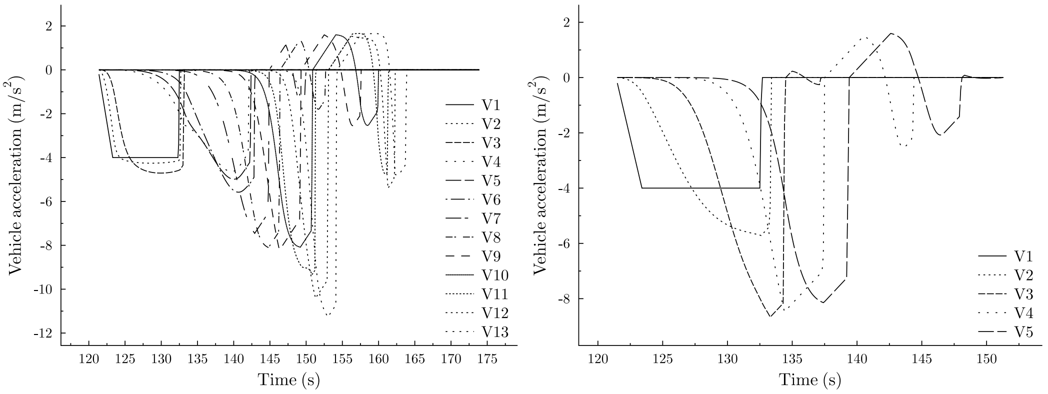
Parameter	Value	Unit
IEEE standard	802.11p CCH	
AC (beacons)	AC_BK	
AC (EEBL)	AC_VO	
Data rate	6	Mbps
Bandwidth	10	MHz
Tx power	20	dBm
Propagation loss	Three log distance	

Table 5.3: Network parameters

5.2 SINGLE LANE TESTS

Figure 5.1a shows a trace in time of the acceleration of the first thirteen vehicles, from the moment in which the first vehicle starts braking. The deceleration of the first vehicle is fixed at 4 m/s^2, while the followers use the acceleration provided by the model. In particular, vehicles number 12 and 13 reach a deceleration larger than 1 G, which is not possible with common production cars [49, 48]. Figure 5.1b instead is focused on first five vehicles for a better understanding: it is possible to see that the model can result in different "braking forces" depending on the aggressiveness of the driver. Moreover, after having stopped, drivers re-accelerate to come closer to their leader, which is a common situation.

The modification done to IDM (Section 4.2, Equation (4.3)) results in crashes. It is possible to see in Figure 5.2 the peaks of acceleration and deceleration caused by rear end collisions. Figure 5.2a shows a collision between two vehicles, in particu-

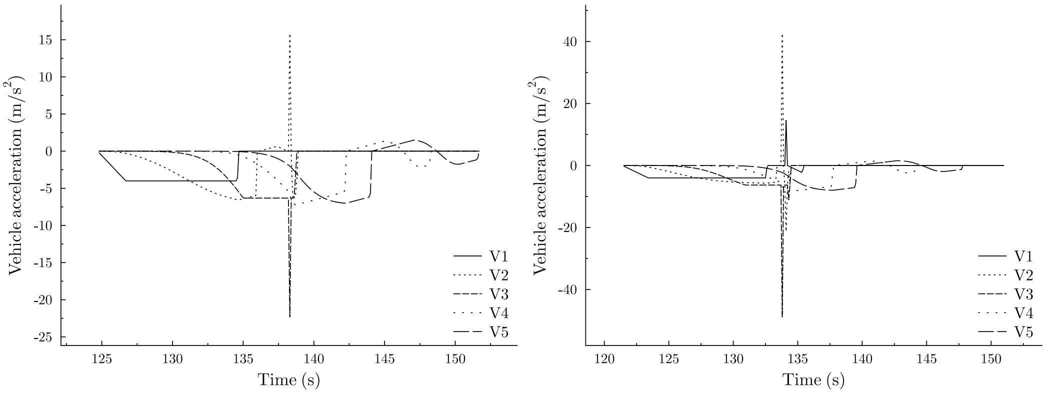


(a) First thirteen vehicles

(b) First five vehicles

Figure 5.1: Trace of the accelerations in time for pure idm test, 41.66 m/s

lar V3 which collides with V2 (notice the deceleration of V3 and the acceleration of V2) while Figure 5.2b shows a pile-up of three cars (V3, V2 and V1).

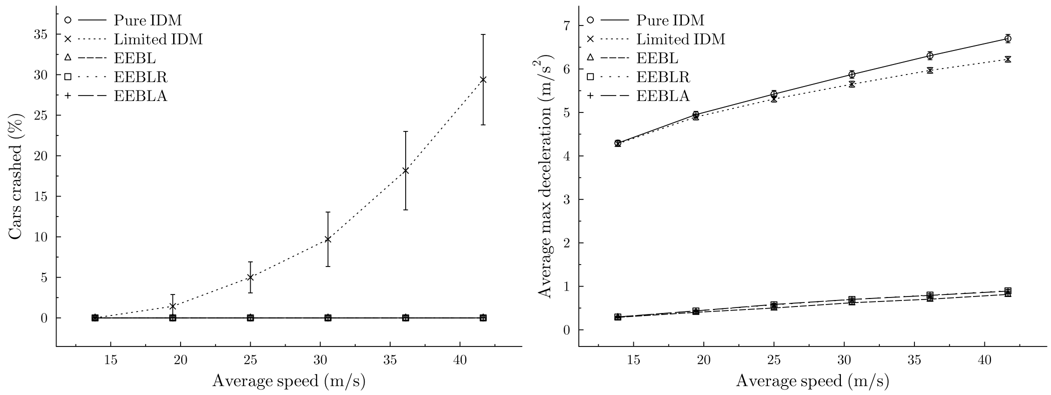


(a) 36.11 m/s

(b) 41.66 m/s

Figure 5.2: Trace of the accelerations in time of the first five vehicles for the limited idm test

So the first question to be answered is: can an eebl system improve the safety of a highway? The answer is straightforward, but it must be justified and quantified. In particular Figure 5.3a shows the percentage of vehicles involved in a crash, with a 95% confidence interval, as a function of the speed. Just to clarify, if vehicles A, B, C and D are involved in a pile-up, then four vehicles are counted. Obviously, pure idm results in a crash free scenario, since it is a car following model, while



(a) Crash percentages

(b) Average maximum deceleration

Figure 5.3: Quantification of the benefits of an EEBL system, for a 100% market penetration rate

limited idm results in an increase of crashes with a non-linear growth. This result obtained by the simulation reflects what happens in reality [55][2].

For a 100% market penetration rate, the EEBL system integrated with the CACC would completely avoid crashes, whatever the protocol. Clearly we are assuming that drivers never perform a sharp steering to avoid a collision: in such case no system can avoid a disaster, unless the on-board computer completely controls the car, which is unlikely. The result could seem straightforward: if all vehicles are equipped then the situation is always under control. However this assumption cannot be given a priori: if, for example, the EEBLR protocol is so "aggressive" that the number of collisions (in the network) causes a lack of information to the CACC, possible dangerous situations could happen.

Figure 5.3b shows the average maximum decelerations for the various tests. To obtain the results, the acceleration of all vehicles have been monitored during simulations: at the end the maximum values of deceleration have been averaged and plotted with a 95% confidence interval. The experiments with pure idm and limited idm show an average maximum deceleration which is always higher than 4 m/s^2 (the deceleration of the first vehicle). limited idm is "softer", but only because vehicles cannot brake stronger than a certain limit. For the EEBL enabled experiments instead the deceleration is very smooth. This is important

[2] Indeed in the cited report the authors found an exponenential relationship between speed and crash risk. In this thesis it is not possible to analyze such relationship because only six speeds have been considered. Moreover, it is not in the aims of this work to reproduce a real crash model. The important thing here is not having a linear relationship, because real world is not like that (as far as Kloeden et al. have discovered)

for safety: if an automated system performs a violent braking maneuver on a wet road, consequences can be devastating.

From the point of view of the automotive field, the deceleration could be considered too weak, with possible negative influences on traffic flow. However this result is due to the "naïveness" of the implemented CACC: desired deceleration could be parameterized (as for IDM) and increased, without harming the overall safety. A deceleration of 2-3 m/s^2 can be considered safe even on wet floors, where decelerations of 6 m/s^2 can be easily reached [49]. On icy roads the problem would come up again, but in that case a specialized study should be done.

At this stage of the results it could seem that EEBLR and EEBLA give no benefits and are only a waste of resources. In Figure 5.3 EEBL, EEBLR and EEBLA are indistinguishable. But it is important to remember that till now a full market penetration rate is considered, and we have not yet analyzed the network load.

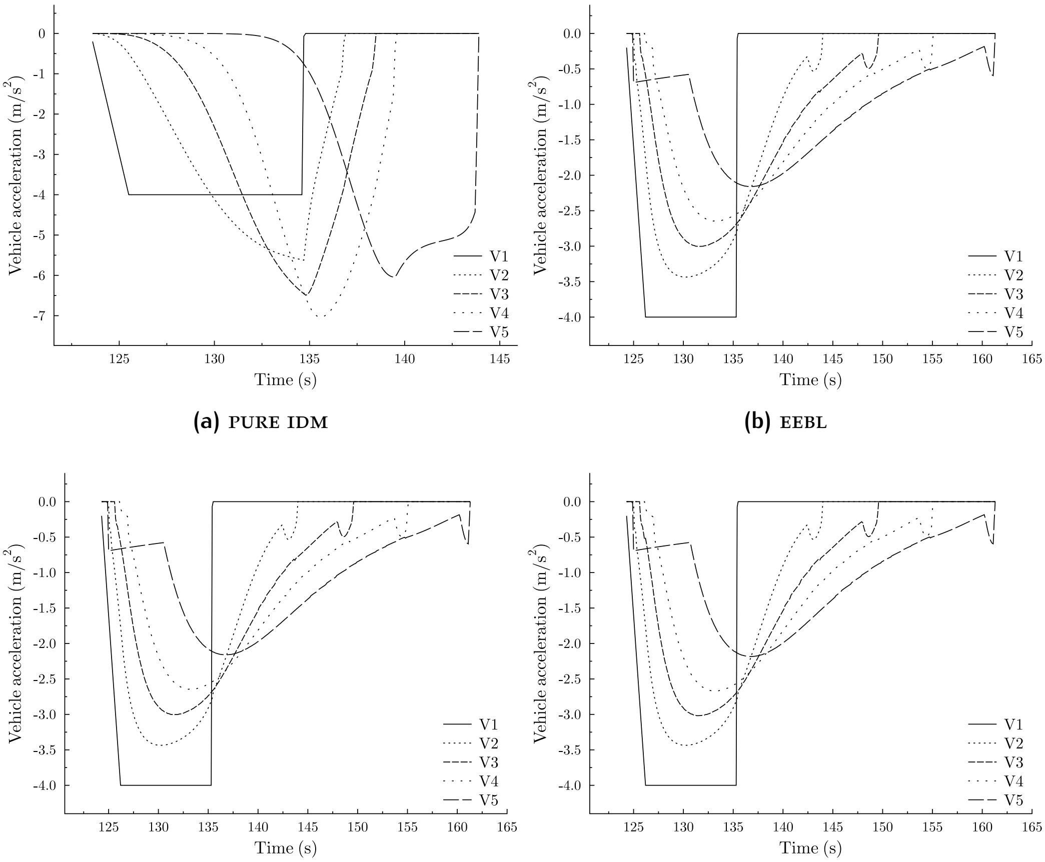
The reason why the average maximum deceleration is so low, is shown in Figure 5.4: in the PURE IDM test, vehicles behind the leader brake harder than it, while with the CACC the deceleration becomes more and more smooth. This is for sure a benefit due to the safety distance mantained by the CACC. This result could be the basis for the study of a CACC which minimizes the inter-vehicle distance, to improve traffic flow. A lower safety distance, would result in a stronger deceleration, which could be calibrated in order to maximize traffic flow while keeping the risk of accidents under control: again this is not the aim of the thesis.

In Figure 5.4 moreover, it is possible to notice again the similarity between EEBL, EEBLR and EEBLA from the point of view of the automotive application. It is important to give some more details about those pictures (e.g., Figure 5.4b): first of all it is possible to notice the deceleration caused by the air drag force for vehicle 5. When it receives the EEBL message from first vehicles, the driver removes its foot from the accelerator pedal and the car begins to decelerate due to air resistance. Then when vehicle 4 starts braking harder and sending EEBL messages, the CACC of vehicle 5 starts reacting.

When vehicle 2 reaches 1 m/s^2 of deceleration, it switches from EEBL protocol to normal beaconing: however the system still performs well[3]. This is due to the implementation of the CACC: in the case of missing packets it predicts position and speed of front vehicle using previously collected data and applying simple physics formulas (e.g., $x(t) = v(t) \cdot \Delta t + x_0$). Finally a little peak of deceleration occurs due to the IDM model: basically "the driver" wants to apply a brake force stronger than the CACC, so the latter leaves the control to IDM.

Given the results of the application-oriented analysis, it is possible to start with network load characterization. Figure 5.6 shows a three-dimensional representa-

[3] In fact it is possible to see small imperfections, but they are very difficult to notice



(a) PURE IDM

(b) EEBL

(c) EEBLR

(d) EEBLA

Figure 5.4: Traces of decelerations for different experiments. The average speed is 41.66 m/s

tion of the averaged network load perceived by vehicles, as a function of time and vehicle position. Before describing the graph it is better to clarify how it has been obtained.

First of all it must be noticed that describing the network load in a VANET is not trivial. Considering the spatial extension of the scenario, different network loads could be measured: if vehicles in a zone of the highway are communicating each other, they will "feel" a certain value of channel load which will not "be felt" five kilometers away. Measuring the channel load by the point of view of the sender is not correct or, at least, it gives only a partial vision of the congestion. So the load has been measured as seen by receivers: in particular for each packet sent, send time, receive time and the list of receivers have been logged. Then the load has been obtained by summing the duration of all the packets sent in a second, for all seconds of the simulation.

To be more formal, if Tx_i and Rx_i are the transmission and reception time (in microseconds)[4] respectively and v the vehicle that has received the packet then the load can be computed as shown in Listing 5.1. In particular the first thing to do is determine the "slot" of seconds in which packet must be counted: if $Tx_i = 10123456\mu s$ then the packet must be counted in the range between 10 and 11 seconds of simulation. If $secondsTx$ is different from $secondsRx$ then the transmission began in a second and ended in the successive. For example, if $Tx_i = 10999900$ and $Rx_i = 11000200$ then 100 µs must be counted in the tenth seconds slot, while the other 200 in the eleventh seconds slot.

```
$secondsTx = \lfloor Tx_i/10^6 \rfloor$
$secondsRx = \lfloor Rx_i/10^6 \rfloor$

if ($secondsTx = secondsRx$) then
  $load[v][secondsTx] += (Rx_i - Tx_i)/10^6$
else
  $firstAmount = (secondsTx + 1) \times 10^6 - Tx_i$
  $load[v][secondsTx] += firstAmount/10^6$
  $load[v][secondsRx] += (Rx_i - Tx_i - firstAmount)/10^6$
end if
```

Listing 5.1: Channel load computation

This procedure is not accurate enough because it does not account for situations like the one in Figure 5.5. In such case the algorithm in Listing 5.1 would count $Rx_1 - Tx_1 + Rx_2 - Tx_2$ which is not correct. So an algorithm to merge such situations (so that the load is computed as $Rx_2 - Tx_1$) have been developed and used for channel load measurement.

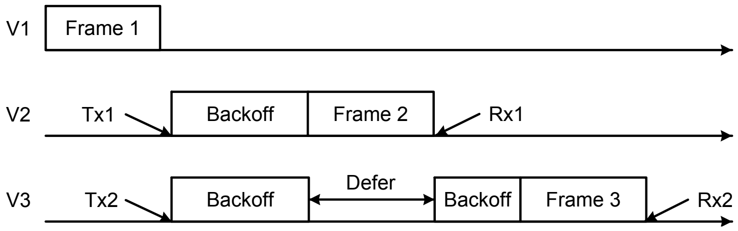


Figure 5.5: Contention for the channel

At the end of the computation the matrix $load$ contains the perceived load for each vehicle for each second of simulation. Then the vehicles are sorted as

[4] N.B.: Tx_i indicates the time at which the application passes the packet to the MAC layer, while Rx_i the time at which the packet is delivered to the application by the MAC layer

a function of their distance from the first vehicle of the simulation, obtaining a matrix pos such that

$$pos[i][t] = \text{id of the } i\text{-th vehicle in distance from first vehicle at time } t.$$

Then elements are re-arranged so that a new load matrix $sortedLoad$ is obtained:

$$sortedLoad[i][t] = load[pos[i][t]][t]. \tag{5.1}$$

The matrix obtained by Equation (5.1) is identical to the matrix of Listing 5.1 in the case of a single lane, while this is not true in the multi-lane scenario, where overtakes occur. This sorted matrix is important because otherwise the resulting graph would mix the loads of vehicles which are in complete different locations of the scenario. For example, imagine that a vehicle at the end of the platoon[5] is so fast that it overtakes all vehicles up to the leaders: with the matrix of Listing 5.1, its load value would be plotted in the rightmost part of the graph, which is not correct, since the vehicle is together with the firsts in the platoon.

If a test is repeated twenty times, twenty matrices are obtained. These matrices are averaged and the result is used to finally plot the colormap.

Even with such solution the load would not tell the true story, because collided packets do not have receivers, so their load cannot be computed. In the presence of collisions, the load would be underestimated. So an analysis of packet collisions has been performed (described later).

As a final remark, notice that the computation of the load takes into account also AIFS and backoff procedure[6], which is correct. The channel is considered busy in those periods because it cannot be accessed.

Now it is possible to come back to Figure 5.6: on the abscissa there are vehicles ordered by position, while on the ordinate the time. The first thing to notice is the black triangle on the bottom, which is due to the fact that vehicles are injected while the simulation runs, so at the beginning not all vehicles are present. Then the purple stripe indicates a low load zone due to the beaconing: one packet per second for fifty vehicles does not create any problem. Moreover the distance makes the load even lower, because packets of first vehicles are not received by vehicles at the end of the platoon, and viceversa. To give a theoretical quantification of the load it is possible to adapt Equation (2.2) for beacons. For the AC_BK (the

[5] If a vehicle is one of the last that has "joined" the simulation it has a high id, since id are given to the vehicles in the order they are injected in the highway

[6] Because send time considers the moment in which the packet is passed to the MAC layer

AC used for beacons) the AIFS value is 9 slots (so 117 µs) while the backoff on average should be 97.5 µs (because CW_{min} is 15 slots). Then the load for beaconing protocol is:

$$\mathrm{Load}_{beacon}(n) = \frac{486\mu s \cdot n}{1s} \cdot 100(\%). \tag{5.2}$$

However Equation (5.2) is not very correct in this particular case. Due to the low load, all nodes will very probably find the channel free when trying to send a packet. Following the CSMA rules, they will access the channel immediately, without waiting for AIFS and without performing the backoff procedure and so a more correct formulation is

$$\mathrm{Load}_{beacon}(n) = \frac{272\mu s \cdot n}{1s} \cdot 100(\%). \tag{5.3}$$

Later the correctness of Equation (5.3) will be empirically proved.

At roughly 150 seconds the first vehicle starts braking and sending EEBL messages. Consequently, its followers do the same, so the network load rapidly increases up to 2%, as shown by the light blue stain. Here it must be noticed that the load increases only for the first fifteen vehicles, which is a consequence of how the CACC works: by looking back at Figure 5.4b, the maximum deceleration decreases more and more while going far away from first vehicle. As a result, some vehicles never decelerate stronger than 1 m/s^2 so they never send an EEBL message. This has clearly positive effects on the network: some vehicles (between 10 and 15) perceive an increase of network load for a smaller amount of time. Now consider again the statement written to comment Figure 5.3: by redesigning the CACC in order to have a stronger deceleration and improve traffic flow, the network load would increase. This is an important result which emphasizes the need to take into accont the dependency between the application and the network protocol.

After the severe braking phase, the platoon starts to “compact”. This produces an increase of the network load because more vehicles move in the transmission range of the others and more and more beacons are received as time elapses. By looking at first vehicles (1 to 10) it is possible to se a shade from purple to blue. Central vehicles instead feel the increase before the others, because they can receive packets from the front and the end of the platoon. Final vehicles instead are clearly the last to perceive the increment. A further analysis of this dynamic is given in Section 5.3.

Now it is possible to show empirically that Equation (5.3) is correct. In the case of fifty vehicles, the equation would result in a load of 1.36% which agrees with the blue zone at the top of the graph.

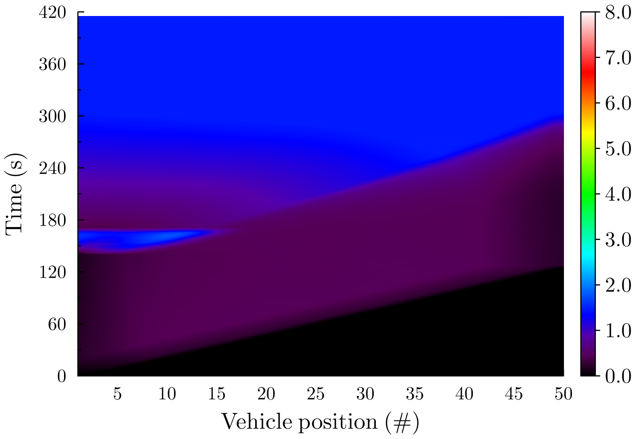
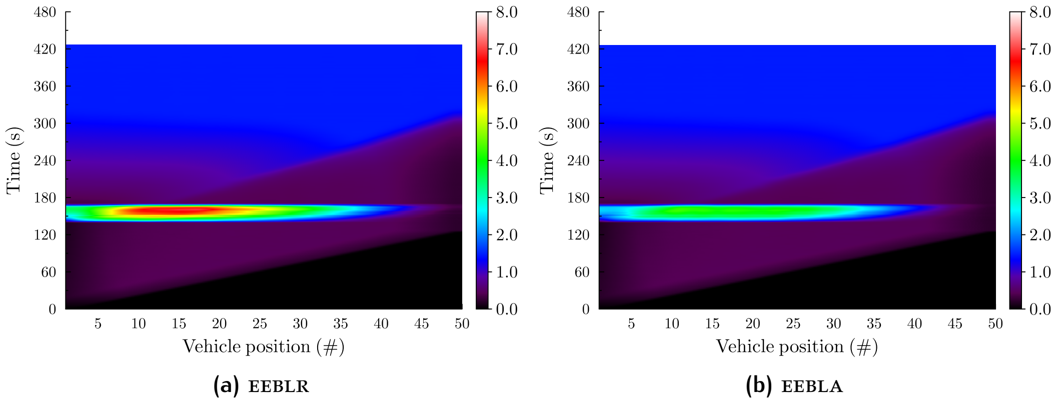


Figure 5.6: Network load for EEBL experiment, one lane 36.11 m/s

Figure 5.7 shows the loads produced by the EEBLR and EEBLA protocols. The figures are similar to the situation of Figure 5.6, because the dynamics of the vehicles are the same. What changes is clearly the network load: in particular, EEBLR reaches an average maximum of 7%, while EEBLA roughly 4%. So the load of EEBLR is more than tripled w.r.t. normal EEBL, while EEBLA is doubled. Notice that doubling the load of the network with a repropagation protocol is somehow natural. Imagine that vehicle A in Figure 5.8 broadcasts a packet and vehicle B repropagates it: vehicles in their transmission range will clearly receive the packet twice, there is no way to avoid it.

It is possible to see that the color stain caused by the braking is spread in time and space, because the information disseminates farther thanks to repropagation.



(a) EEBLR

(b) EEBLA

Figure 5.7: Network load for EEBLR and EEBLA experiments, one lane 36.11 m/s

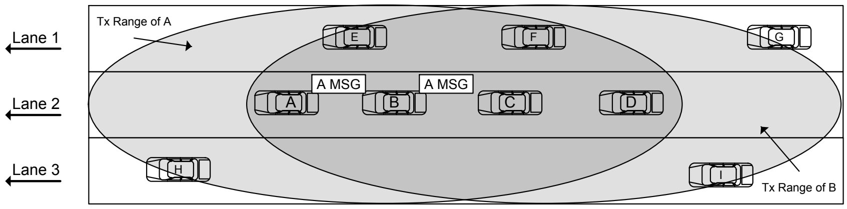


Figure 5.8: Natural doubling of network load due to message repropagation

To conclude this section one more graph is presented. Given the load matrix of Equation (5.1), the set of maximum load values have been extracted. More formally:

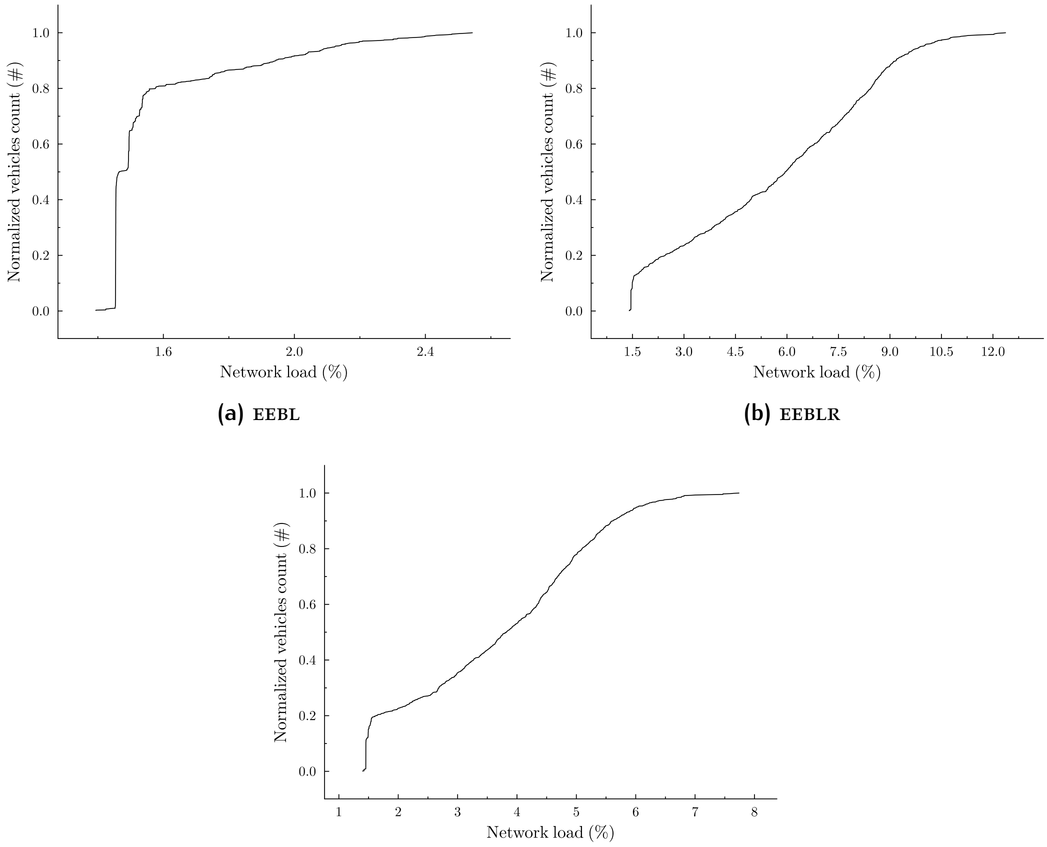
$$maxValues[exp][i] = \max_{t \in T} \mathrm{sortedLoad}_{exp}[i][t] \qquad exp \in [1, \ldots, N_{exp}].$$

These values have been used to plot a Cumulative Distribution Function (CDF), so to determine the number of vehicles having "noticed" a maximum load lower than a certain value, for all experiments together. The results are shown in Figure 5.9: the ordinate axis has been normalized.

The EEBL experiment shows a complete different trend w.r.t. the other two. This is probably given by the fact that the EEBL protocol during the braking phase has more or less the same load as the beaconing at the end of the simulation, so almost all vehicles have "felt" the same maximum value. For EEBLR and EEBLA instead, the load during the braking phase is much higher, so the CDF changes. Comparing the EEBLR experiment with EEBLA it is possible to see that EEBLR protocol has an overall maximum slightly more than 12%, while EEBLA reaches 8%. Those values tell that loads of Figure 5.6 and Figure 5.7 are not the real maximum, but this is clearly due to averaging. In Figure 5.9a indeed the maximum obtained is 2.6% and not 2%.

5.3 MULTI LANE TESTS

In this section the multi-lane scenario will be analyzed by the point of view of the network, since no crashes occured in any tests with EEBL. To continue with the analysis given in the last part of previous section, Figure 5.10 shows the network load for the five lane test, with 250 vehicles. The "graphical aspect" is the same as in Figure 5.6 and Figure 5.7, but clearly the scale changes. In particular the EEBL test reaches 10% during the braking phase, so there is a linear increase w.r.t. the number of vehicles, which is quite obvious.



(a) EEBL

(b) EEBLR

(c) EEBLA

Figure 5.9: CDFs of maximum loads of all single-lane experiments, for average speed 36.11 m/s

The maximum load of the EEBLR test is again more than tripled, reaching up to 36%. This may seem a safe limit: still two thirds of the channel time are free and the simulation has a large number of vehicles. However some observations are in order:

1. as said before, collided packets are not counted in the network load. Later on the collisions will be analyzed and it will be shown that the situation is much worse;
2. as in one lane test, the design of the CACC makes the majority of the vehicles brake with a deceleration lower than 1 m/s^2, so for the majority of the vehicles no EEBL message is ever sent;
3. lanes in the opposite direction are not considered. Having other five lanes can further worsen the situation. Probably in most cases the opposite lanes

would not broadcast other EEBL messages but this cannot be given for sure, for example when accidents regards both directions[7].

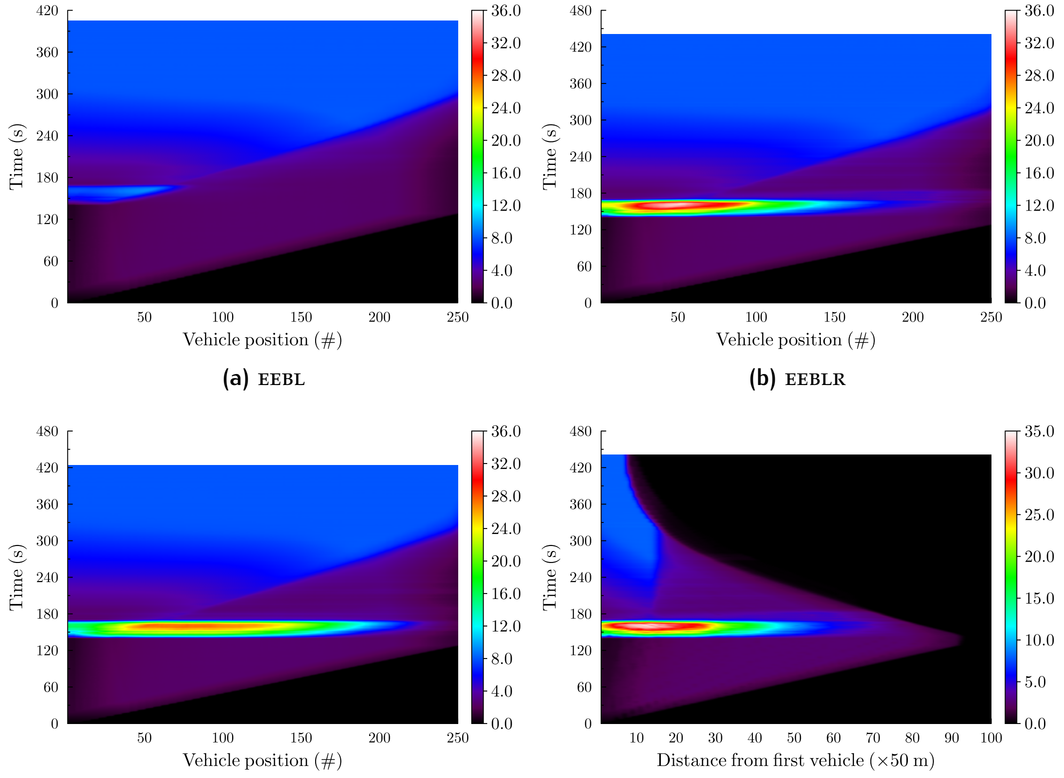
The EEBLA experiment instead shows a maximum of roughly 26%. It is 10% less than EEBLR and 2.6 times more than in the EEBL experiment.

Finally, Figure 5.10d shows the load for the EEBLR test with five lanes but this time the colormap is drawn as a function of the distance from first vehicle. The portion of highway behind the leader has been divided into blocks of 50 meters each. Then the load of each block has been computed as the average load of the vehicles in that block. This representation gives a better idea of the dynamics of the platoon and their consequences on the network load. The black triangle at the bottom of the image remains as before. The platoon reaches roughly 4.5 km of extension and then leaders begin to brake. It is possible to see how the size of the platoon decreases down to roughly 500 meters: as the group of vehicles decreases in size, the load of the network increases, as shown by the light blue zone at the top of the figure.

To further analyze the consequences of the protocols on network load two more graphs have been plotted. The first one measures the percentage of packets that have not been received by any vehicle during the braking phase, with a confidence interval of 95%. This is not the same as measuring the number of collisions: indeed if a vehicle is alone in the highway its messages will not be received by anybody, but this does not mean that they had a collision. However, given the density of vehicles in the scenario, the measure should give a very good approximation. The reason why this measure has been taken into account, instead of counting the number of collisions, is that *NS-3* does not have any way to measure them. Indeed, since the simulator reproduces quite faithfully the reality, situations as in Figure 5.11 are ambiguous. If vehicles A and D send a message at the same time, vehicle C will most probably not be able to receive them (so a collision is counted), while vehicle B will probably get the message, since the packet of vehicle D will only be heard as background noise due to signal attenuation. The second graph instead counts the number of packets that vehicles try to send for each second of the simulation, including received and unreceived.

Figure 5.12 shows the first graph. As said in the previous paragraph, the percentage of unreceived packets has been analyzed during the braking phase. This period starts when leaders begin to brake and ends when the deceleration of all

[7] Remember the accident on the Italian highway A4 in august 2008 when a truck suddenly steered and invaded the opposite lanes (article (in Italian) http://archiviostorico.corriere.it/2008/agosto/09/Tir_salta_guardrai_Strage_dell_co_9_080809024.shtml, video http://www.youtube.com/watch?v=kjajyJEzKGU, last visited 16/02/2011)



(a) EEBL

(b) EEBLR

(c) EEBLA

(d) EEBLR as a function of the distance from first vehicle

Figure 5.10: Average loads for test with five lanes, average speed 36.11 m/s

vehicles is less than 1 m/s^2 and their speed is lower than 8.33 m/s (30 km/h)[8]. The analysis has been limited in this period because first of all it is the "critical moment" for the network and secondly, as it is possible to see from Figure 5.10, the portion of the simulation where the load is very high is only a small fraction, so averaging the values of the whole simulation would lead to an underestimation of the real problem.

By looking at Figure 5.12 it is possible to see the huge amount of unreceived packets for the EEBLR protocol. For the one lane scenario, the percentage is 20%, while for five lane the percentage reaches 60%. Notice that this does not mean that the maximum load of Figure 5.10b must be increased of those percentages. Indeed if, for example, four packets collide, the collision period will last from the beginning of the first till the end of the last and the frames are overlapped for

[8] This is a heuristic choice to try to grab the most critical part of the dynamic in absence of a formal definition

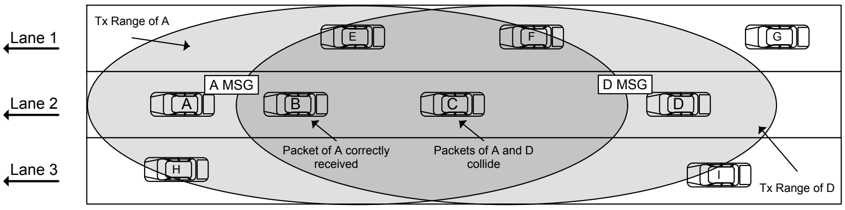


Figure 5.11: Collision detection problem

the majority of the time, so the load increases of a quantity which is roughly the duration of the biggest frame.

The "enqueue and defer transmission" mechanism of EEBLA instead performs very well. The percentage of unreceived packet is not zero, neither for EEBLA nor for EEBL, but it is under 1%. This is because the retransmission of a particular packet is delayed of an amount of time between 0 and 100 ms. This delay permits the nodes to detect prior retransmissions and remove relative packet from the send queue. With EEBLR instead, once a packet has been scheduled for retransmission it is sent to the MAC queue, and then it cannot be removed. Notice that a delay of (maximum) 100 ms is good for the application requirements.

This result indicates that the load plotted in Figure 5.10b is underestimated, while the loads of Figure 5.10a and Figure 5.10c are correct.

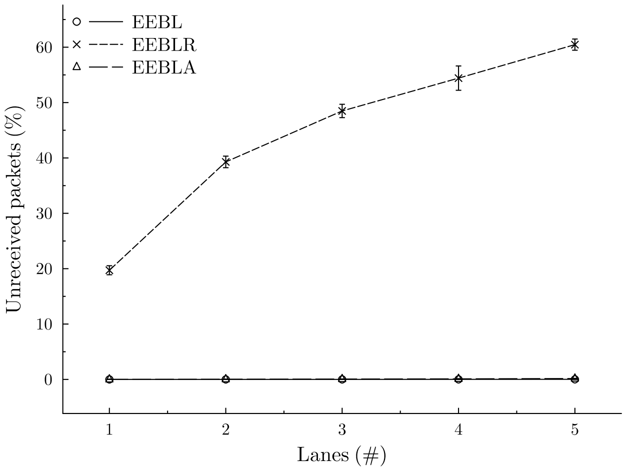
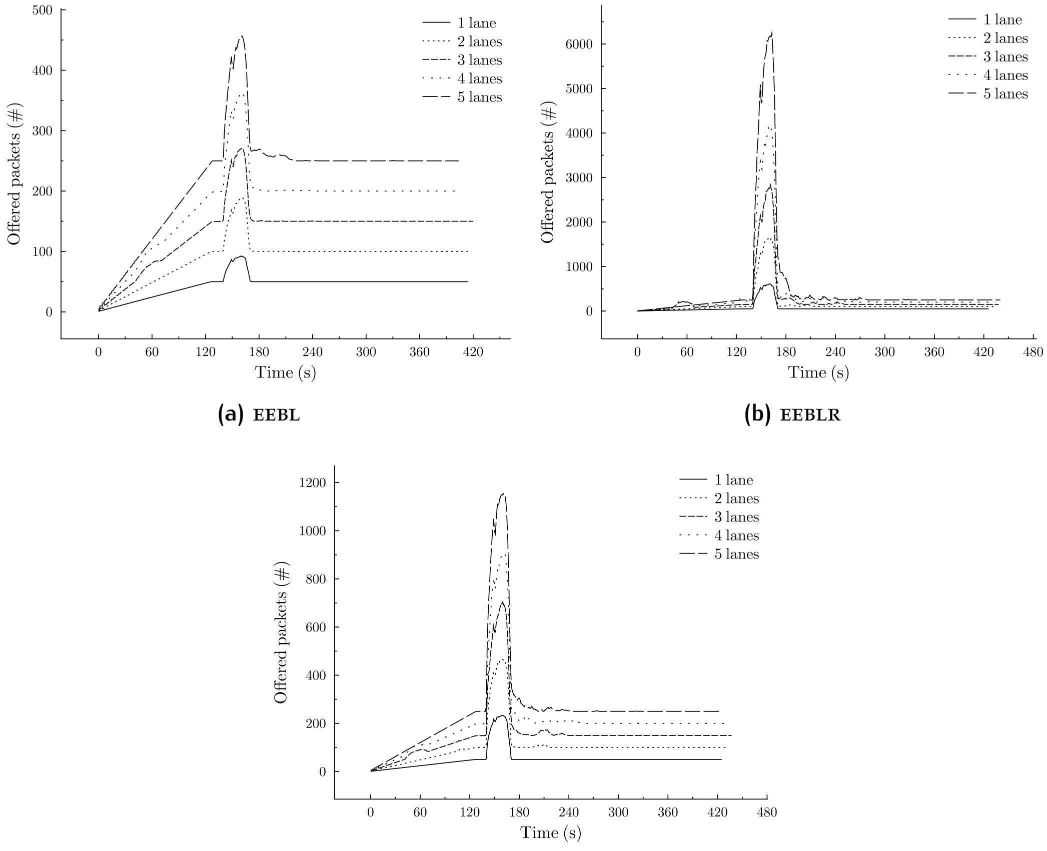


Figure 5.12: Percentage of packets that have been sent but not received by any node. Average speed 36.11 m/s, 100% market penetration rate

Figure 5.13 shows the second set of graphs. Their "aspect" is similar: they have an initial linear increase due to the injection of vehicles in the simulation. Then during the braking phase the number of offered packets suddenly increases and finally decreases down to a constant value which is equal to the number of vehicles, since all nodes are sending one beacon per second.

Now by looking at maximum peaks it is possible to see that they reach, for the EEBL experiment, roughly 100, 200, 300, 400 and 500. For the EEBLA experiment, those peaks reach roughly 200, 450, 700, 900 and 1150 so, as already said before, there is an increase of a factor of slightly more than two. The EEBLR experiment instead shows the aggressiveness of the EEBLR protocol. The peaks here reach around 500, 1500, 3000, 4000, and 6000, which is definitely too much. Notice that the EEBLR protocol already implements a broadcast suppression mechanism (i.e., *p-persistent* rebroadcast [28].), so imagine how devastating could be a pure flooding protocol.



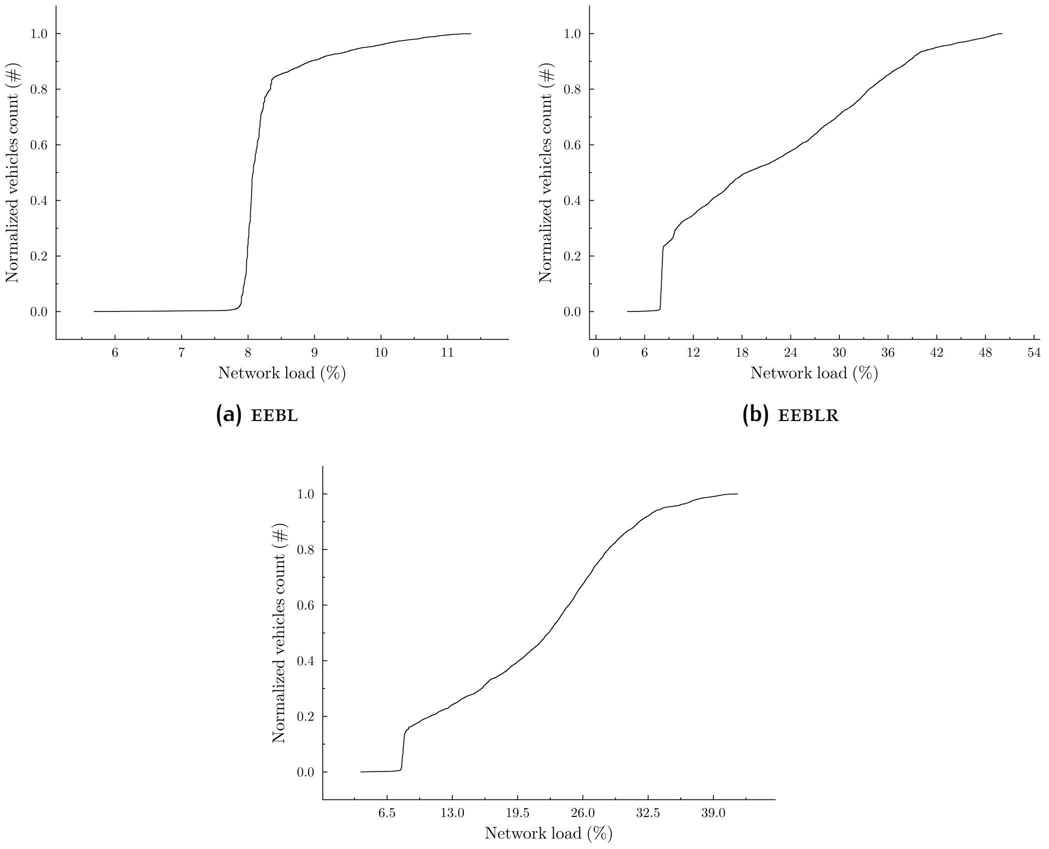
(a) EEBL

(b) EEBLR

(c) EEBLA

Figure 5.13: Average number of offered packets during the simulation for the different protocols. Average speed 36.11 m/s

To further understand the congestion problem, the same CDFs as in Section 5.2 have been plotted (Figure 5.14). The trends are basically the same as before: what changes are clearly the maximum values reached. EEBL goes up to 11.5%, EEBLA up to 40% while EEBLR up to 50%. Notice that these are absolute maximums and, by considering the 90% of the vehicles, they decrease down to 9% (EEBL), 29% (EEBLA) and 39% (EEBLR). These values show again that EEBLR is aggressive and that also EEBLA could be improved, which should be somehow easy, given its simple logic.



(a) EEBL

(b) EEBLR

(c) EEBLA

Figure 5.14: CDFs of maximum loads of all five-lane experiments, for average speed 36.11 m/s

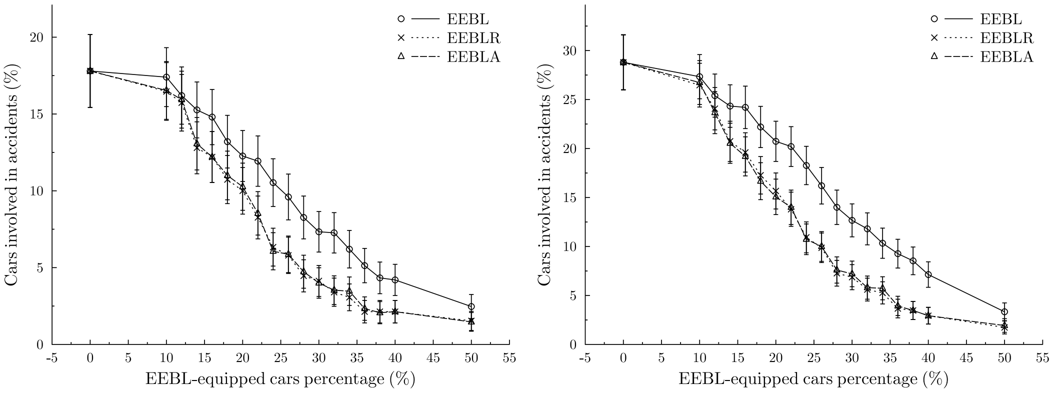
Till now, the EEBLA protocol has shown to be a good compromise between network load and information spread, without using any complex mechanism found in the literature. From the point of view of the application, the CACC works the same as with EEBL protocol, so what is the profit in message repropagation?

5.4 SINGLE LANE MARKET PENETRATION RATE TESTS

The question left open in the previous section finds its answer here. As described at the beginning of the chapter, the market penetration rate test gives an idea of the benefits of VANET safety applications when not all vehicles are equipped.

Figure 5.15 shows the plot of the percentage of cars crashed (with 95% confidence interval) as a function of the market penetration rate, for EEBL, EEBLR and EEBLA tests at 36.11 and 41.66 m/s. The graph shows the clear advantages of message repropagation. Both the EEBLR and EEBLA tests have a lower percentage of accidents, due to the fact that drivers are informed of the dangerous situation with higher probability and in advance, and the simple action of removing the foot from the accelerator pedal and let the car gently decrease speed reduces the risk of impact. The higher gain is noticeable in the central part of the graph, between 15 and 40%. For market penetratrion rates lower than 15% the difference is smaller because the low fraction of equipped vehicles limits message repropagation. With a 50% rate it is possble to see that the EEBL test starts to converge to the results of the others. The difference between the protocols at 41.66 m/s is even more evident.

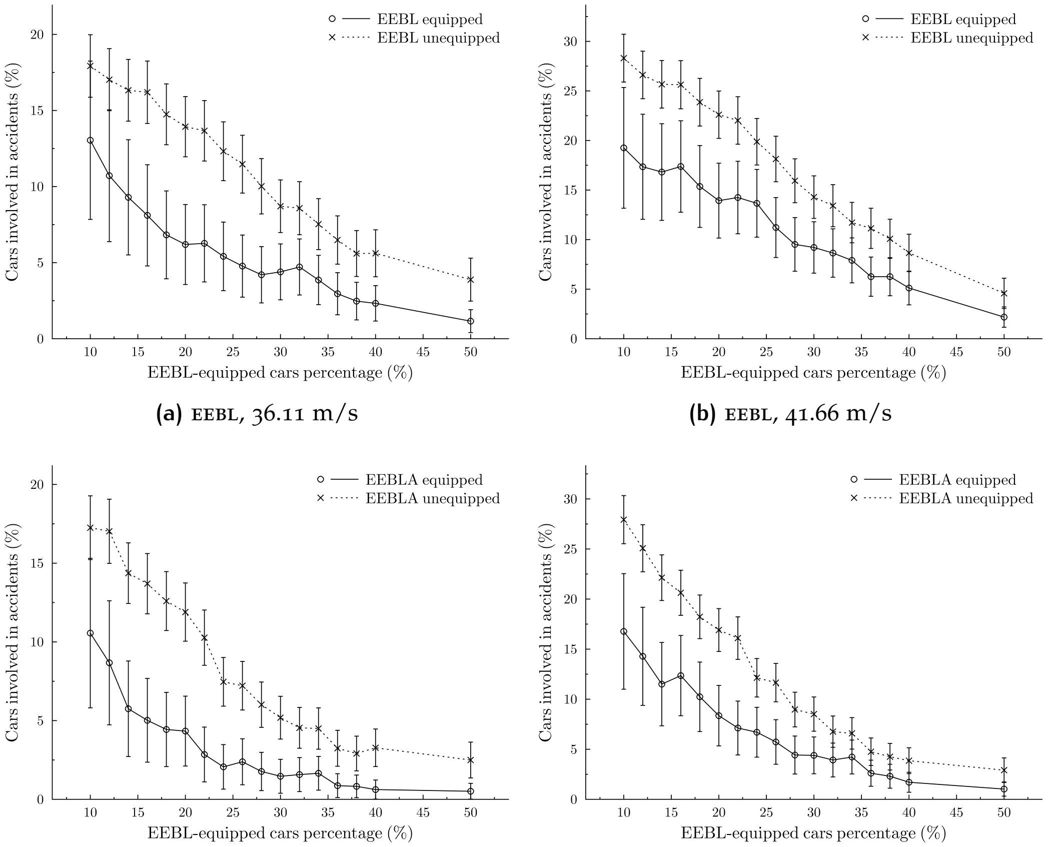
Notice that this result is most probably underestimated because the action of removing the foot from the pedal is an oversimplification of the real human behavior. If a driver receives an EEBL message sent a few vehicles in front it would most probably brake a little bit, thus increasing inter vehicle distance earlier.



(a) 36.11 m/s

(b) 41.66 m/s

Figure 5.15: Percentage of cars involved in accidents for single-lane market penetration rate tests



(a) EEBL, 36.11 m/s

(b) EEBL, 41.66 m/s

(c) EEBLA, 36.11 m/s

(d) EEBLA, 41.66 m/s

Figure 5.16: Percentage of cars crashed divided for equipped and unequipped for the single-lane test

A question that could be arisen is whether the EEBL combined with the CACC gives benefits only to the equipped vehicles. To address this problem the percentage of cars crashed has been divided for equipped and unequipped and plotted for all the experiments, with a 95% confidence interval.

The results for the EEBL experiment are shown in Figure 5.16a and Figure 5.16b: clearly on the abscissa the 0% market penetration rate has been removed. The graph depicts the same trend of Figure 5.15 and shows that the percentage of crashed vehicles with the EEBL system is lower than for unequipped, which is quite obvious. The important thing here is that the probability of crash for unequipped vehicles decresases in the same way as for equipped. The reason is simple: if an EEBL enabled car receives a message and starts decelerating smoothly, its follower will also do the same. When vehicles then arrive into the "braking zone" their speed will be lower, thus they will not need to perform a violent deceleration and they will most probably avoid a read-end collision.

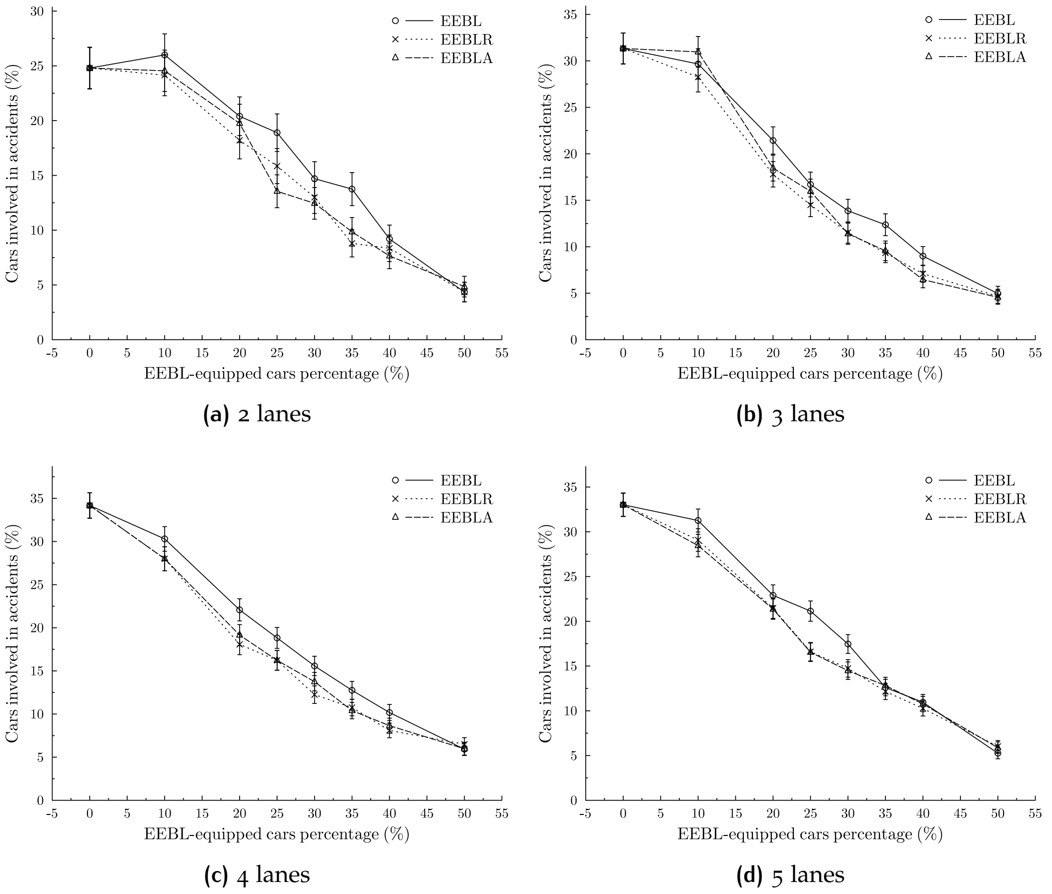
For the EEBLA experiment (Figure 5.16c and Figure 5.16d) the result is similar (in the sense that both equipped and unequipped gain advantages), but the overall percentage is clearly lower than for the EEBL experiment. Moreover in Figure 5.16c the percentage for equipped vehicles reaches low values much more earlier: with a market penetration rate of 24% the probability is lower than 2.5%, while in Figure 5.16a that probability is reached when the 38-40% of the cars are equipped. The results for the EEBLR tests have been omitted, since they are almost identical to EEBLA ones.

5.5 MULTI LANE MARKET PENETRATION RATE TESTS

To conclude the experiments, this section shows the market penetration rate results for the multi-lane scenario. The comparison between the three tests for two, three, four and five lanes is shown in Figure 5.17. As for the single lane test, there is a visible difference between the basic protocol and the ones with repropagation. This distinction tends to become smaller as the number of lanes increase and, for the five lane test, the results for the rates 35, 40 and 50% are almost identical. The reason is explained in Figure 5.18[9]: in the single-lane case, if a vehicle is not able to reach directly one of its equipped followers (A and B in Figure 5.18a) then nobody behind will know about the dangerous situation in advance. With more lanes instead, the message of a braking vehicle could be received by a driver on another lane (like B in Figure 5.18b) which will smoothly decelerate, reducing its risk of collision. After a while, it will have anyhow the need to brake so it will start sending messages informing other vehicles (such as C). C will then do the same as B, providing information about the situation to the driver of vehicle D. So now, vehicle D is aware of the danger even if it is too far from vehicle A to receive any of its messages.

As for the one lane tests, both equipped and unequipped vehicles can gain advantages by the presence of an EEBL system as depicted by Figure 5.19. For unequipped vehicles, the trends of Figure 5.19a and Figure 5.19b are roughly the same, with EEBLA which has a slightly lower probability of crash. Equipped vehicles instead show a completely different trend, emphasizing the results discussed in the previous section: the difference here is even more evident. However this

[9] N.B.: the arrows in the picture must not be intended as repropagation, but just represent the fact that a higher density of equipped vehicles increases the probability that "gentle" behaviors propagate in the platoon



(a) 2 lanes

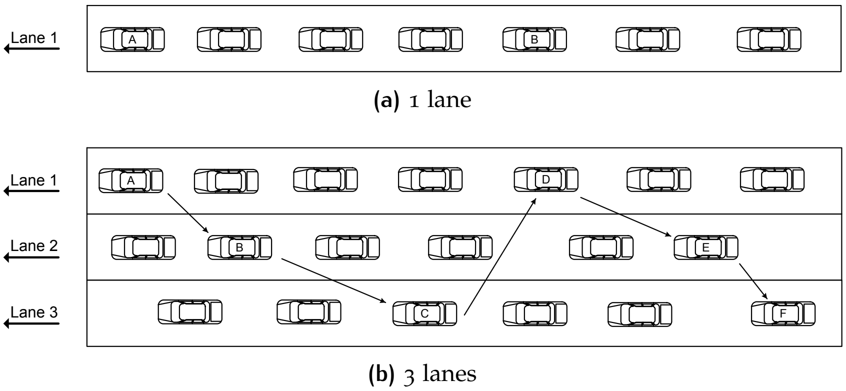
(b) 3 lanes

(c) 4 lanes

(d) 5 lanes

Figure 5.17: Percentage of cars involved in accidents for multi-lane market penetration rate tests, average speed 36.11 m/s

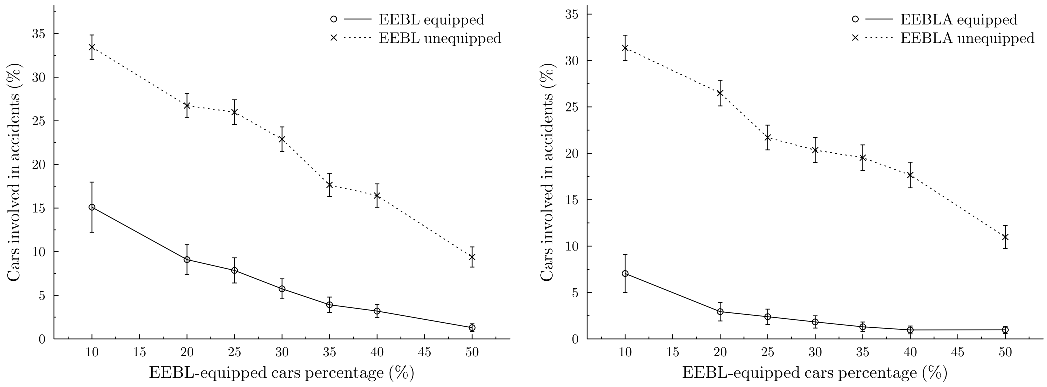
is not due to a lower number of crashes for the equipped vehicles, indeed the probability of crash is almost the same in Figure 5.16c and in Figure 5.19b. What changes is the probability of crash for unequipped vehicles, which is higher for the multi-lane scenario. As depicted by Figure 5.15a and Figure 5.17, the percentage of crashes (for a 0% market penetration rate, so it is not due the the presence of the CACC) increases with the number of lanes. This could be reasonable in reality where, after a crash, a vehicle could unintentionally move to the adjacent lane causing another accident. In the simulator however this is somehow strange because such realism is not present. This is a consequence of the lane change: in the single-lane scenario a slow vehicle cannot be overtaken, so faster drivers must keep a lower speed. Consequently, the average speed of the platoon is reduced, decreasing the risk of accidents. Anyhow this result does not "nullify" any of the advantages of the system described in the whole chapter.



(a) 1 lane

(b) 3 lanes

Figure 5.18: Higher probability of "natural" propagation for a multi-lane highway. Vehicles with letters are the equipped ones



(a) EEBL

(b) EEBLA

Figure 5.19: Percentage of cars crashed divided for equipped and unequipped for the five lanes test, average speed 36.11 m/s

6 FUTURE WORK

This thesis gives the basis for a deeper analysis in the field of VANETs having analyzed for the first time with a joint mobility-network simulator real applications like EEBL. The dependency between the application and the network, however, need further investigation. In U.S. DOT [6] tens of safety and non-safety applications are listed: they clearly have to coexist but, from the point of view of the network, they have completely different requirements, in terms of information spreading and update rate. Both of these requirements should be checked: indeed this thesis has shown that a greater "spread range" can improve safety and that with a frequency update lower than mandated systems can perform well[1]. As an example, an application dependent protocol for EEBL could use a high update rate when deceleration increases and a lower update rate when it decreases. This has shown to perform good for a CACC but EEBL is thought also for humans where the reaction time plays a fundamental role, and so it should be tested. Another modification could be to increase the deceleration threshold used to determine when EEBL messages should be sent, again analyzing the consequences on the overall dynamics.

Regarding human behavior modeling, a more sophisticated "emulation" should be implemented because, as already said before, the action of removing the foot from the accelerator pedal upon reception of an EEBL warning is an oversimplification. Without taking into account complex AI algorithms, a severity value could be assigned to an EEBL message as a function of the distance from the original sender (i.e., decrease severity as distance increases) and as a function of the lane (i.e., if the originator is on the same lane the severity is maximal, otherwise it is set depending on the lane the receiver is located). Given the severity, the "driver" could perform different operations, from braking smoothly to simply ignore the message. Once a better human reaction is developed, an extension to the market penetration rate test can be done by considering completely unequipped vehicles, vehicles equipped only with a 802.11p network card only (so to simulate cars with acoustic warning) and vehicles equipped with a CACC.

By the point of view of the automotive field it could be interesting to improve the behavior of the CACC in order to provide a stronger deceleration reducing the time the platoon needs to compact, while finding a way to keep the network

[1] Remember in Section 5.2 where the prediction mechanism of the CACC is described

traffic under control. Another improvement could be to take into account a real CACC, so to enable an automatic car following mechanism and analyzing again network requirements and the benefits on traffic flow. For a better analysis of traffic flow properties in general, scenarios different from the simple emergency brake should be done, for example by emulating a sort of stop and start and observe if with such systems the "accordion effect" reduces.

The EEBLA algorithm has shown to be a good compromise between application and network requirements, without the need of complex methodologies, opposed to the approaches found in the literature. However, given the simplicity of EEBLA, most probably its results on network load could be further improved by some simple modifications, for example using other broadcast suppression mechanisms as the basis of the algorithm, such as *slotted p-persistence* [28] and/or data compression. Moreover, the differences on load given by different transmission ranges could be analyzed, without the need of taking into account complex mathematical model as in Torrent-Moreno et al. [43]. To further reduce the network load lower values of TTL can be tested, but always keeping under control application-level benefits. Reducing the value of TTL reduces the spread range of information, thus decreases the "global knowledge" of the drivers. Finally, a more accurate simulation of the physical layer must be taken into account. Since this thesis was focused on congestion issues a simple propagation loss model had been used; in future more realistic models can be used, for example considering also fading.

The simulations have been performed using a bunch of parameters derived from the literature. For example, the values for maximum decelerations have been obtained from Schultz and Babinchak [49], which is quite an old document. A more recent report is Michigan State Police [48] but it takes into account cars for the police and it is not known whether those vehicles are more performant than normal production cars. Recent documents describing this parameter must be found in order to use values of vehicles of nowadays. This could give a more precise indication of the number of car crashes. Another parameter that could be modified is the desired deceleration (b in IDM): for this thesis it has been fixed to a value of -4 m/s^2, in order to have quite critical situations. For high-fidelity modeling, values of deceleration used by real drivers could be searched: if no statistics on this parameter exists, it could be at least randomized to reproduce different braking "methodologies".

7 CONCLUSIONS

This thesis has faced different aspects of VANETs. From the implementative point of view, Chapter 4 has shown how to modify the original mobility simulator by Arbabi and Weigle [46] to introduce and manage crashes, drifting clocks to avoid unreal synchronization effects, a basic CACC and the deceleration caused by air resistance, in order to emulate a naturally decelerating car. Furthermore, the implementation of network protocols devoted to the application-aware aggregation and consolidation of messages has been presented.

These modifications to the original simulator permitted to perform the tests described in Chapter 5, which have shown the benefits of a VANET safety application and its relative network analysis in a coordinated and joint mobility and network simulation tool which enables more realistic analysis of the vehicular traffic evaluation.

First of all, Section 5.2, has shown how the Intelligent Driver Model model behaves in the case of emergency braking and how the more realistic model results in crashes. Then it has been shown that with a full market penetration an automatized EEBL plus CACC system would completely avoid crashes and reduce the deceleration peaks further, thus improving safety by avoiding skids: all the versions of the EEBL protocol performs the same. The analysis of the network has shown that the way the CACC is implemented can influence the load, thus emphasizing the dependency between application and network protocols. A further element proving this relationship is given by the increase of network load caused by the compacting of platoon extension during the braking.

Comparing EEBLA, the novel retransmission and aggregation protocol proposed in this thesis, with pure EEBL without retransmissions, the network load increase is slightly more than double. This must be confronted with the load generated by EEBLR (based on the *weighted p-persistence* broadcast suppression mechanism [28]) which is more than triple. Increasing the number of lanes, and hence the base offered load, this diversity leads to significative differences at the application level, which can, under specific circumstances, mean safety threats to vehicles.

The tests on market penetration rate have shown the benefits of message repropagation which cannot be observed with a full penetration rate. In particular, there is a clear distinction between the probability of crash with and without repropagation. Rebroadcast improves safety, since even drivers far away from the danger are aware of the situation. The benefits given by the EEBL application are "earned" not only by equipped vehicles, but also by unequipped ones because drivers following an EEBL enabled car can decelerate with a smoother jerk because their leader are doing the same. The same distinction has been shown for the multi-lane tests but with a slightly lower difference in some cases, probably caused by the increment of density of equipped vehicles per road kilometer due to the higher number of lanes.

Results obtained in this thesis are only a little contribution in understanding the dynamics of VANETs and CACC systems but we hope they can contribute to deploy this systems as soon as possible and reduce car accidents, fatalities, injuries and the consequent money loss.

BIBLIOGRAPHY

[1] U.S. Department of Transportation. National Transportation Statistics, 2010. URL http://www.bts.gov/publications/national_transportation_statistics/pdf/entire.pdf. Last visited: 21/12/2010.

[2] Automobile Club Italia. Annuario Statistico (Statistical Yearbook), 2010. URL http://www.aci.it/sezione-istituzionale/studi-e-ricerche/dati-e-statistiche/annuario-statistico-2010.html. Last visited: 04/02/2011.

[3] Istituto Nazionale di Statistica. Demographic Balance, 2010. URL http://demo.istat.it/index_e.html. Last visited: 04/02/2011.

[4] National Highway Traffic Safety Administration. Fatality Analysis Reporting System - General Estimates System - Data Summary 2008, 2008. URL http://www-nrd.nhtsa.dot.gov/Pubs/811171.PDF. Last visited: 09/02/2011.

[5] L. Blincoe, A. Seay, E. Zaloshnja, T. Miller, E. Romano, S. Luchter, and R. Spicer. The Economic Impact of Motor Vehicle Crashes 2000. U.S. Department of Transportation - National Highway Traffic Safety Administration, May. 2002.

[6] Vehicle safety communications project task 3 final report. Technical report, U.S. Department of Transportation, National Highway Traffic Safety Administration, Mar. 2005.

[7] S. Biswas, R. Tatchikou, and F. Dion. Vehicle-to-vehicle wireless communication protocols for enhancing highway traffic safety. *Communications Magazine, IEEE*, 44(1):74 – 82, Jan. 2006.

[8] B. van Arem, C.J.G. van Driel, and R. Visser. The Impact of Cooperative Adaptive Cruise Control on Traffic-Flow Characteristics. *Intelligent Transportation Systems, IEEE Transactions on*, 7(4):429 –436, Dec. 2006.

[9] IEEE Standard for Information technology–Telecommunications and information exchange between systems–Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments. *IEEE Std 802.11p-2010 (Amendment to IEEE Std*

802.11-2007 as amended by IEEE Std 802.11k-2008, IEEE Std 802.11r-2008, IEEE Std 802.11y-2008, IEEE Std 802.11n-2009, and IEEE Std 802.11w-2009), pages 1 –51, Jul. 2010.

[10] IEEE Trial-Use Standard for Wireless Access in Vehicular Environments (WAVE) - Resource Manager. *IEEE Std 1609.1-2006*, pages c1 –63, 2006.

[11] IEEE Trial-Use Standard for Wireless Access in Vehicular Environments - Security Services for Applications and Management Messages. *IEEE Std 1609.2-2006*, pages c1 –105, 2006.

[12] IEEE Trial-Use Standard for Wireless Access in Vehicular Environments (WAVE) - Multi-Channel Operation. *IEEE Std 1609.4-2006*, pages c1 –74, 2006.

[13] IEEE Trial-Use Standard for Wireless Access in Vehicular Environments (WAVE) - Networking Services. *IEEE Std 1609.3-2007*, pages c1 –87, 2007.

[14] IEEE Standard for Information Technology-Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)*, pages C1 –1184, Jun. 2007.

[15] Leon W. Couch, II. *Digital and Analog Communication Systems*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 6th edition, 2000.

[16] Arogyaswami Paulraj, Rohit Nabar, and Dhananjay Gore. *Introduction to Space-Time Wireless Communications*. Cambridge University Press, New York, NY, USA, 1st edition, 2008.

[17] IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements. *IEEE Std 802.11e-2005 (Amendment to IEEE Std 802.11, 1999 Edition (Reaff 2003)*, pages 1 –189, 2005.

[18] I. Tinnirello, G. Bianchi, and L. Scalia. Performance evaluation of differentiated access mechanisms effectiveness in 802.11 networks. In *Global Telecommunications Conference, 2004. GLOBECOM '04. IEEE*, volume 5, pages 3007 – 3011, Nov. 2004.

[19] Hannes Hartenstein and Kenneth Laberteaux, editors. *VANET Vehicular Applications and Inter-Networking Technologies*. John Wiley & Sons, Jan. 2010.

[20] Christoph Sommer, David Eckhoff, Reinhard German, and Falko Dressler. A Computationally Inexpensive Empirical Model of IEEE 802.11p Radio Shadowing in Urban Environments. In *Eighth International Conference on Wireless On-Demand Network Systems and Services*, 2011.

[21] Thomas Mangel, Friedrich Schweizer, Timo Kosch, and Hannes Hartenstein. Vehicular Safety Communication at Intersections: Buildings, Non-Line-Of-Sight and Representative Scenarios. In *Eighth International Conference on Wireless On-Demand Network Systems and Services*, 2011.

[22] M. Nakagami. The m-distribution, a general formula for intensity distribution of rapid fading. In W. G. Hoffman, editor, *Statistical Methods in Radio Wave Propagation*. Oxford, England: Pergamon, 1960.

[23] Hervé Boeglen, Benoît Hilt, Pascal Lorenz, Jonathan Ledy, Anne-Marie Poussard, and Rodolphe Vauzelle. A survey of V2V channel modeling for VANET simulations. In *Eighth International Conference on Wireless On-Demand Network Systems and Services*, 2011.

[24] Stylianos Papanastasiou, Jens Mittag, Erik G. Strom, and Hannes Hartenstein. Bridging the Gap between Physical Layer Emulation and Network Simulation. In *Wireless Communications and Networking Conference (WCNC), 2010 IEEE*, pages 1 –6, Apr. 2010.

[25] Daniel Jiang, Qi Chen, and Luca Delgrossi. Optimal data rate selection for vehicle safety communications. In *VANET '08: Fifth ACM international workshop on VehiculAr Inter-NETworking*, pages 30–38, 2008.

[26] Sze-Yao Ni, Yu-Chee Tseng, Yuh-Shyan Chen, and Jang-Ping Sheu. The broadcast storm problem in a mobile ad hoc network. In *MobiCom '99: 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 151–162, 1999.

[27] Zygmunt J. Haas, Joseph Y. Halpern, and Li Li. Gossip-based ad hoc routing. In *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1707 – 1716, 2002.

[28] O.K. Tonguz, N. Wisitpongphan, J.S. Parikh, Fan Bai, P. Mudalige, and V.K. Sadekar. On the Broadcast Storm Problem in Ad hoc Wireless Networks.

In *Broadband Communications, Networks and Systems, 2006. BROADNETS 2006. 3rd International Conference on*, pages 1 –11, Oct. 2006.

[29] Stefan Pleisch, Mahesh Balakrishnan, Ken Birman, and Robbert van Renesse. MISTRAL: efficient flooding in mobile ad-hoc networks. In *MobiHoc '06: 7th ACM international symposium on Mobile ad hoc networking and computing*, pages 1–12, 2006.

[30] A. Bachir and A. Benslimane. A multicast protocol in ad hoc networks intervehicle geocast. In *Vehicular Technology Conference, 2003. VTC 2003-Spring. The 57th IEEE Semiannual*, volume 4, pages 2456 – 2460, Apr. 2003.

[31] L. Briesemeister, L. Schafers, and G. Hommel. Disseminating messages among highly mobile hosts based on inter-vehicle communication. In *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE*, pages 522 –527, 2000.

[32] Ming Li and Wenjing Lou. Opportunistic broadcast of emergency messages in vehicular ad hoc networks with unreliable links. In *QShine '08: 5th International ICST Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness*, pages 1–7, 2008.

[33] Gökhan Korkmaz, Eylem Ekici, Füsun Özgüner, and Ümit Özgüner. Urban multi-hop broadcast protocol for inter-vehicle communication systems. In *VANET '04: 1st ACM international workshop on Vehicular ad hoc networks*, pages 76–85, 2004.

[34] J.L. Sobrinho and A.S. Krishnakumar. Distributed multiple access procedures to provide voice communications over IEEE 802.11 wireless networks. In *Global Telecommunications Conference, 1996. GLOBECOM '96. 'Communications: The Key to Global Prosperity*, volume 3, pages 1689 –1694, Nov. 1996.

[35] Chih-Wei Yi, Yi-Ta Chuang, Hou-Heng Yeh, Yu-Chee Tseng, and Pin-Chuan Liu. Streetcast: An Urban Broadcast Protocol for Vehicular Ad-Hoc Networks. In *Vehicular Technology Conference (VTC 2010-Spring), 2010 IEEE 71st*, pages 1 –5, May. 2010.

[36] O. Tonguz, N. Wisitpongphan, F. Bai, P. Mudalige, and V. Sadekar. Broadcasting in VANET. In *2007 Mobile Networking for Vehicular Environments*, pages 7 –12, May. 2007.

[37] Wang-Rong Chang, Hui-Tang Lin, and Bo-Xuan Chen. TrafficGather: An Efficient and Scalable Data Collection Protocol for Vehicular Ad Hoc Networks.

In *Consumer Communications and Networking Conference, 2008. CCNC 2008. 5th IEEE*, pages 365 –369, Jan. 2008.

[38] I. Chisalita and N. Shahmehri. A context-based vehicular communication protocol. In *Personal, Indoor and Mobile Radio Communications, 2004. PIMRC 2004. 15th IEEE International Symposium on*, volume 4, pages 2820 – 2824, Sep. 2004.

[39] B. Ducourthial, Y. Khaled, and M. Shawky. Conditional Transmissions: Performance Study of a New Communication Strategy in VANET. *Vehicular Technology, IEEE Transactions on*, 56(6):3348 –3357, Nov. 2007.

[40] S. Eichler, C. Schroth, T. Kosch, and M. Strassberger. Strategies for Context-Adaptive Message Dissemination in Vehicular Ad Hoc Networks. In *Mobile and Ubiquitous Systems - Workshops, 2006. 3rd Annual International Conference on*, pages 1 –9, Jul. 2006.

[41] J. Bronsted and L.M. Kristensen. Specification and performance evaluation of two zone dissemination protocols for vehicular ad-hoc networks. In *Simulation Symposium, 2006. 39th Annual*, page 12, Apr. 2006.

[42] K. Ibrahim and M.C. Weigle. CASCADE: Cluster-Based Accurate Syntactic Compression of Aggregated Data in VANETs. In *GLOBECOM Workshops, 2008 IEEE*, pages 1 –10, Dec. 2008.

[43] M. Torrent-Moreno, P. Santi, and H. Hartenstein. Distributed Fair Transmit Power Adjustment for Vehicular Ad Hoc Networks. In *Sensor and Ad Hoc Communications and Networks, 2006. SECON '06. 2006 3rd Annual IEEE Communications Society on*, volume 2, pages 479 –488, Sep. 2006.

[44] Yunpeng Zang, L. Stibor, H.-J. Reumerman, and Hiu Chen. Wireless local danger warning using inter-vehicle communications in highway scenarios. In *Wireless Conference, 2008. EW 2008. 14th European*, pages 1 –7, Jun. 2008.

[45] M. Sepulcre and J. Gozalvez. On the Importance of Application Requirements in Cooperative Vehicular Communications. In *Eighth International Conference on Wireless On-Demand Network Systems and Services*, 2011.

[46] Hadi Arbabi and Michele C. Weigle. Highway Mobility and Vehicular Ad-Hoc Networks in ns-3. In *Winter Simulation Conference (WSC)*, Dec. 2010.

[47] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested Traffic States in Empirical Observations and Microscopic Simulations. *PHYSICAL REVIEW E*, 62:1805, 2000.

[48] 2010 Aftermarket Brake Pad Evaluation. Technical report, Michigan State Police Precision Driving Unit, Nov. 2010.

[49] Gregory Schultz and Michael Babinchak. Final Report For The Methodology Study Of The Consumer Braking Information Initiative. Technical report, U.S. Department of Transortation, Mar. 1999.

[50] Martin Treiber, Arne Kesting, and Dirk Helbing. Delays, Inaccuracies and Anticipation in Microscopic Traffic Models. *PHYSICA A*, 360:71, 2006.

[51] Martin Treiber and Dirk Helbing. Realistische Mikrosimulation von Strassenverkehr mit einem einfachen Modell. *16th Symposium Simulationstechnik ASIM*, 2002.

[52] Arne Kesting, Martin Treiber, and Dirk Helbing. General Lane-Changing Model MOBIL for Car-Following Models. *Transportation Research Record: Journal of the Transportation Research Board*, 1999:86–94, Jan. 2007.

[53] G.K. Batchelor. *An introduction to fluid dynamics*. Cambridge mathematical library. Cambridge University Press, 2000.

[54] Rüdiger Cordes. Große C_w-Werte-Sammlung von Autos (Large collection of cars C_ws), 2010. URL `http://rc.opelgt.org/indexcw.php`. Last visited: 12/02/2011.

[55] C. N. Kloeden, A. J. McLean, and G. Glonek. Reanalysis of Travelling Speed and the Risk of Crash Involvement in Adelaide South Australia, 2002. URL `http://casr.adelaide.edu.au/speed/RESPEED.PDF`. Last visited: 14/02/2011.

LIST OF ACRONYMS

ABS	Antilock Braking System
AC	Access Category
AC_BE	AC for best effort traffic
AC_BK	AC for background traffic
AC_VI	AC for video traffic
AC_VO	AC for voice traffic
ACC	Adaptive Cruise Control
ACK	ACKnowledgement
AI	Artificial Intelligence
AIFS	Arbitration Inter Frame Space
BSM	Basic Safety Message
CACC	Collaborative Adaptive Cruise Control
CC	Cruise Control
CCH	Control CHannel
CDF	Cumulative Distribution Function
CFW	Forward Condition
CSMA	Carrier Sense Multiple Access
CSMA/CA	Carries Sense Multiple Access with Collision Avoidance
CSMA/CD	Carries Sense Multiple Access with Collision Detection
CTB	Clear To Broadcast
CTS	Clear To Send
CUP	Upward Condition

CV	Cluster head
CW	Contention Window
DCF	Distributed Coordination Function
DIFS	Distributed Inter Frame Space
DSRC	Dedicated Short Range Communications
DSSS	Direct Sequence Spread Spectrum
DV-CAST	Distributed Vehicular broadCAST
EDCA	Enhanced Distributed Channel Access
EDCAF	Enhanced Distributed Channel Access Function
EEBL	Emergency Electronic Brake Lights
EEBLA	EEBL with Aggregation
EEBLR	EEBL with Rebroadcast
EM	Environment representation
ERGN	Enhanced Route Guidance and Navigation
ESC	Electronic Stability Control
FCC	Federal Communications Commission
FCS	Frame Check Sequence
FIFO	First In First Out
GNU	GNU is Not Unix
GPL	General Public License
GPS	Global Positioning System
GPU	Graphics Processing Unit
HDM	Human Driver Model
IDM	Intelligent Driver Model
IEEE	Institute of Electrical and Electronics Engineers

ILC	Inter-Layer Communication module
IP	Internet Protocol
ISM	Industrial Scientific and Medical
LAN	Local Area Network
LLC	Logical Link Control
MAC	Medium Access Control
MANET	Mobile Ad-hoc NETwork
MOBIL	Minimizing Overall Braking Induced by Lane change
MPDU	MAC Protocol Data Unit
MRTS	Multicast Request To Send
OFDM	Orthogonal Frequency Division Multiplexing
PCF	Point Coordination Function
PHY	Physical Layer
PIFS	PCF Inter Frame Space
PLCP	Physical Layer Convergence Procedure
QoS	Quality of Service
RM	Request Message
RSU	RoadSide Unit
RTB	Request To Broadcast
RTS	Request To Send
RV	Relay Vehicle
SCH	Service CHannel
SIFS	Short Inter Frame Space
TCL	Tool Command Language
TCP	Transmission Control Protocol

TTL	Time To Live
V2R	Vehicle-to-Roadside
V2V	Vehicle-to-Vehicle
VANET	Vehicular Ad-hoc NETwork
WAVE	Wireless Access in Vehicular Environments
WLAN	Wireless LAN
WM	Winning Message
WSN	Wireless Sensor Network