TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

ASIF SARDAR

Improving Performance of IEEE 802.11p MAC Layer for Emergency Message Dissemination

Master of Science Thesis

Examiners:
Senior Research Scientist Dmitri Moltchanov
Professor Yevgeni Koucheryavy
Examiner and topic approved by the Faculty Council of the Faculty of Computing and Electrical Engineering on 08 May 2013.

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Information Technology
SARDAR ASIF: Improving Performance of IEEE 802.11p MAC Layer for Emergency Message Dissemination.
Master of Science Thesis, 62 pages
May, 2013
Major Subject: Communications Engineering
Examiners: Senior Research Scientist Dmitri Moltchanov
Professor Yevgeni Koucheryavy
Keywords: VANET, IEEE 802.11p MAC, WAVE, EDCA, emergency message dissemination, one-hop broadcasting

Vehicular ad-hoc networking is the most promising subfield of mobile ad-hoc networks, which may become the ad-hoc networking technology in near future for vehicles communicating amongst themselves on road. It uses IEEE 802.11p MAC protocol as wireless networking technology. The IEEE 802.11p MAC protocol has inherent problems in wireless ad-hoc networking environment due its heterogeneous, infrastructureless and highly dynamic nature. The performance of IEEE 802.11p MAC layer for vehicular ad-hoc networking is based on performance of one-hop broadcasting.

The performance of IEEE 802.11p one-hop broadcasting is of major concern regarding emergency message dissemination. The CSMA/CA protocol used in IEEE 802.11p is far from optimal solution for emergency message dissemination due to inherent properties of random access, higher delivery delays and retransmissions. Techniques to improve emergency message dissemination delivery rate and minimize time latency of message dissemination, such as, disabling backoff and synchronous transmission, have been mentioned in this thesis out of which one technique such as disabling backoff is being evaluated through simulation results.

The goal of this thesis work is to evaluate a technique, modifying the IEEE 802.11p MAC layer protocol using Network Simulator 3 (NS3). The technique is based on introducing a separate EDCA queue and a separate EDCAF function for emergency messages in QoS EDCA priority queues, disabling backoff for emergency messages and giving highest priority to emergency messages in a station having different AC queues seeking for transmission opportunity. Disabling backoff for emergency messages may reduce time latency arising from exponential backoff algorithm. As the backoff is disabled, more than one station may start transmitting emergency message at the same time. So, it can be deduced that such technique could be beneficial for simple emergency applications. The simulation results show that this technique could be useful for emergency applications utilizing a buzz signal for hazardous warnings on road.

PREFACE

This Master of Science Thesis, Improving Performance of IEEE 802.11p MAC Layer for Emergency Message Dissemination, has been carried out in the Department of Communications Engineering at Tampere University of Technology, Finland. The works has been done during the year 2012-2013 for Department of Communication Engineering at Tampere University of Technology, Tampere, Finland.

I would like to say thank you to Dmitri Moltchanov for guiding me with this thesis work. I am thankful to my friends in Finland for sharing their experiences of research work and thesis writing, encouraging me in difficult times and helping me out in trouble. I am thankful to my family, brothers and sisters for encouraging me to live and get higher education abroad. Finally, I am grateful to my parents, who have supported me all my life time.

Tampere, Finland, May 2013.
Asif Sardar.

Finninmaenkatu 4 C 23,
33710, Tampere,
Finland.
asif.sardar@tut.fi
Tel. +358 43 8265795

CONTENT

List of Abbreviations and Definitions

MANET	Mobile Ad-hoc Networks
DARPA	Defense Advanced Research Project Agency
PRNet	Packet Radio Networks
SURAN	Survivable Radio Networks
GloMo	DARPA Global Mobile Information Systems Program
TI	Tactical Internet
WLAN	Wireless Local Area Network
Wi-Fi	Wireless Fidelity
BAN	Body Area Network
PAN	Personal Area Network
LAN	Local Area Network
MAN	Metropolitan Area Network
WAN	Wide Area Network
VANET	Vehicular Ad-hoc Network
ITS	Intelligent Transportation System
DSRC	Dedicated Short Radio Communication
WAVE	Wireless Access in Vehicular Environment
QoS	Quality of Service
V2V	Vehicle to Vehicle
V2I	Vehicle to Infrastructure
VRC	Vehicle to Roadside Communication
ITS	Intelligent Transportation Systems
OBU	OnBoard Unit
RSU	RoadSide Unit
MAC	Medium Access Control
EDCA	Enhanced Distributed Channel Access
CSMA/CA	Carrier Sense Multiple Access / Collision Avoidance
AC	Access Category
VI	Video
VO	Voice
BE	Best Effort
BK	Background
EM	Emergency Message
PCF	Point Coordination Function
DCF	Distributed Coordination Function
IFS	Inter-Frame Spacing
SIFS	Short Inter-Frame Spacing
PIFS	PCF Inter-Frame Spacing

DIFS	DCF Inter-Frame Spacing
AIFS	Arbitrary Inter-Frame Spacing
NS3	Network simulator 3
IDM	Intelligent Driver Model

1 Introduction

1.1 Background and motivation

Research of MANET on its usage in non-military or commercial area has grown substantially during last few decades. A decade ago, a subtype of MANET known as Vehicular Ad-Hoc Network (VANET) gain a lot of attention and popularity amongst researchers, several agencies at government level in different countries, traffic and transportation agencies and car manufacturing companies for enhancing safety mechanisms in future cars using car-to-car and car-to-roadside communication systems. Such communication systems are speculated to have influence on reducing traffic accidents, provide traffic efficiency, low fuel costs, efficient route finding and to reduce travel time and cost.

Out of many interests in VANET network system, one particular field of research is dedicated to emergency services provided by VANETs to introduce safety mechanisms in real-time on road traffic in future. For such safety mechanism in real-time amongst vehicles on road, a communication system between vehicles is needed to keep vehicles in the vehicular network environment informed of emergency situations on road and provide warnings ahead of time for the driver e.g. speed reduction, collision warning, route change ahead due to traffic accident happened on road, traffic congestion and many other real life scenarios happening on highway or road. With introduction of IEEE 802.11 technology, new ad-hoc networking applications appear on horizon mainly in specialized fields such as emergency services. The growth in automotive market and increased demand of traffic/car safety, ad-hoc networking opens new challenges in car safety mechanisms on road or highway. Such car-to-car communication would present a new way of ad-hoc networking in which cars would communicate with each other through wireless networking amongst them.

But, before the VANET network environment could be adopted in real life traffic scenarios, research and field tests are necessary to evaluate performance and feasibility of such systems in real life traffic systems. Due to dynamic, heterogeneous and rapidly changing network environment, ad-hoc networking has inherent problems for wireless communication and networking amongst vehicles. Moreover, the IEEE 802.11p MAC protocol that has been adopted to be the wireless communication technology for future vehicular ad-hoc networks has short comings based on one-hop broadcasting performance for emergency message dissemination. The network gets congested in case of emergency message dissemination due to flooding of emergency messages broadcasted by many cars simultaneously.

The random channel access property in IEEE 802.11p MAC protocol based on CSMA/CA makes it non-optimal solution for wireless channel access technique. In IEEE 802.11p MAC protocol, the traffic is prioritized for a single station to compete for channel access. The prioritization scheme is characterized by CW size and IFS. Smaller CW size for higher priority traffic and increase in number of emergency messages in a network degrades the performance of IEEE 802.11p MAC protocol for emergency messages. The performance of IEEE 802.11p MAC protocol is highly dependent on the performance of one-hop broadcasting. Since, the network is congested and flooded with broadcast storm of emergency messages in an accident or emergency situation, the random access of wireless channel with prioritized traffic characterized by CW size and IFS; it initiates the backoff mechanism in QoS EDCA, which affects the emergency message dissemination for lower delivery delays and higher delivery probabilities. The denseness of vehicle traffic on road in different scenarios e.g. city centers, traffic jams and slow moving traffic cannot be avoided. We need improved techniques and optimal solutions for wireless networking in vehicular traffic environment for emergency message dissemination with improvements in delivery delays and delivery probabilities. We need either new MAC network protocol or improvements to current IEEE 802.11p protocol.

A simple solution that may improve the performance of emergency message dissemination to some extent is to disable the backoff procedure for emergency messages. For such technique, a new AC queue and EDCAF for emergency messages may be introduced in IEEE 802.11p MAC QoS EDCA mechanism. When the emergency message needs access for transmission, the corresponding EDCAF for emergency messages may sense the medium for current transmission. If the current transmission is ongoing, the emergency message may wait in the queue. The emergency message may be given opportunity after current transmission is over and when medium is free. The backoff procedure is disabled for emergency messages, so its queue will not get affected by the backoff counter value growing exponentially. The emergency messages would have lower delivery delay and higher delivery probability in time.

Another approach is to have cooperated access mechanism by many nodes or vehicles for emergency message dissemination in the wireless network proposed in [16]. In such technique, certain nodes or vehicles in certain coverage area in the wireless network transmits the emergency messages synchronously in a certain time slot in real time without competing to access the wireless medium. This technique indeed, exploits the denseness of vehicles in traffic in a way that, all the vehicles in certain coverage area transmits the same emergency message synchronously in a coordinated way instead of competing for transmission opportunity and the signal for same message is superimposed for higher transmission power and higher transmission range. Decrease in number of nodes competing for medium access and increase in number of nodes

transmitting the same emergency information increases the probability of successful reception and speeds up the dissemination process.

1.2 Thesis objectives

We discuss improvements to IEEE 802.11p MAC protocol for improving time latency in case of emergency message dissemination. The details are described in chapter three and four. We discuss a technique proposed in [16], synchronous relaying in vehicular ad-hoc networks, which may improve emergency message dissemination and time latency through improved control of medium access for emergency message dissemination. This technique require tight synchronization at wireless communication physical layer to cope with propagation delay and multipath signals, because in such technique, all the nodes or vehicles on road at particular distance from each other broadcasts same message at same time. The other technique is quite simple, but effective for emergency message applications that may only require informing vehicles about emergency ahead e.g. a buzz signal that warns the driver to reduce speed or be careful ahead due to emergency situation on road. Such technique is based on introduction of separate emergency message AC queue, disabling the backoff mechanism for emergency messages and EDCAF for this queue in IEEE 802.11p MAC QoS EDCA.

1.3 Thesis contributions

The main contributions of this thesis are enumerated below:

- Modifications to NS3 Wi-Fi module for simulation of IEEE 802.11p MAC based vehicular ad-hoc network.
- Modifying NS3 Wi-Fi ad-hoc mode for one-hop broadcasting.
- Adding new AC priority queue for emergency messages in IEEE 802.11p MAC QoS EDCA using modified NS3 Wi-Fi module.
- Modifying IEEE 802.11p MAC QoS EDCA backoff mechanism and medium access for emergency messages.
- Improvement to emergency message dissemination time latency by disabling backoff in IEEE 802.11p MAC QoS EDCA for emergency messages.

1.4 Thesis outline

This thesis is structured in following order:

Chapter 2 discusses the concept of vehicular ad-hoc networks, its history and roots from mobile ad-hoc networks, its wireless access technology, such as IEEE 802.11p

MAC and its short comings for emergency or safety applications in vehicular ad-hoc network environment.

Chapter 3 discusses the techniques for improving performance of IEEE 802.11p MAC wireless access for applications regarding vehicle safety on road.

Chapter 4 describes the implementation details, the network simulator 3 (NS3), open source code for traffic mobility based on mobility and lane change model utilizing NS3 code, modifications done to NS3 Wi-Fi module and performance results based on simulation in NS3.

Chapter 5 concludes the thesis work based on simulation results for emergency message dissemination using modified IEEE 802.11p MAC protocol and presents suggestions for future work.

2 Vehicular ad-hoc networks

2.1 Introduction

Vehicular ad-hoc network VANET is a sub-type of mobile ad-hoc network (MANET) that is used for communication among vehicles and between vehicles and roadside equipments. Characteristics of VANET are much similar to MANET, but details differ, for example, rather moving at random, vehicles move in an organized fashion on road or highway and their range of motion is restricted by being constrained to follow a paved highway. VANET has its roots in MANET and it is perhaps a most promising area of MANET application.

A brief overview of MANET would be beneficial for understanding the details of VANET. A MANET comprise of two or more devices or nodes equipped with wireless communication and networking capability. The nodes can move freely and are able to dynamically create a temporary self organizing network in a peer to peer network topology with no centralized server or gateway, maintaining the routes and network topology among themselves. These nodes primarily communicate with each other directly within their radio range or can communicate with nodes outside their radio range through store-and-forward mechanism, deploying an intermediate router node between source and destination. Such network topology allows for creating a network for devices to seamlessly internetwork with each other without any requirement for communication infrastructure.

MANET has its history that can be traced back for its use in tactical networks such as battlefield communications. Research has been ongoing for its use in military applications and we can see such effort has been made in early 1970's by the US military Defense Advanced Research Project Agency (DARPA) packet radio networks (PRNet) [3]. The project was inspired by advancements in packet switching technology, such as bandwidth sharing and store-and-forward routing mechanism. Broadcast radio channel was used as wireless medium with minimum central control. To support dynamic sharing of radio resource between nodes, a combination of Aloha and CSMA channel access protocols were used. With store and forward routing and multi-hop communication, it anticipated a capability of multi-user communication within a large geographic area. Until 1990's, with internet and microcomputer revolution, different research projects evolved by the course of time which aimed at improving the MANET design capabilities: SURAN, GloMo and TI [3], [4].

Since 1970's, mobile wireless networking also gain its popularity in communication industry for commercial use. Such networks could provide a capability for ubiquitous

computing and information access through wireless networks, infrastructured or infrastructureless. Infrastructureless wireless networks as mentioned earlier aims at providing a MANET with no fixed router and an arbitrary self-organizing network could be created with multi-hop communication and store-and-forward mechanism. Whereas, infrastructured wireless network with fixed and wired routers, gateways and base stations could provide seamless communication among base stations and nodes within range of these base stations through "Hand-offs". If a node travels outside the range of one base station to another, Hand-off occurs and other base station takes control of the node communication. Examples of such wireless networks are WLAN and wireless cellular networks. Wireless networks such as MANET have no commercial applications yet and it is still under research since decades.

Due to inherent challenges in MANET as compared to its wired network counterpart, we do not see MANET used in practice. Some researchers also terms MANET as completely flawed architecture due to its characteristics. Such challenges may arise fundamentally from two key aspects of MANET: self-organization and wireless transport of information [1].

In MANET, the node may move freely at any time and the topology may change randomly and rapidly which makes the routing more difficult. This enforces MANET to implement routing techniques that could tolerate rapid changes in connectivity among nodes. Such routing protocols are different from conventional routing protocols which we use in fixed infrastructure networks. The MANET is characterized by its mobility, large network size, bandwidth, energy constraints and device heterogeneity which makes designing of routing protocols for MANET a major challenge. Researchers have proposed many routing protocols for MANET, but all these protocols have inherent drawbacks and could not be generalized for MANET characteristics because these protocols are designed with fixed parameters such as node density, network coverage area and transceiver power characteristics. A stable routing protocol is essential for MANET, but currently we could not find a stable routing protocol that could withstand MANET characteristics [1].

The MANET inherits the traditional problems of wireless communication and wireless networking [5]. The transmission range of wireless links could not be determined absolutely due to time varying and asymmetric propagation properties. The wireless medium is less reliable in terms of security from outside signals and it may have hidden terminal problem and exposed terminal problem. The wireless medium is less reliable as compared to wired medium [2]. Bandwidth resource is limited in wireless links as compared to fixed hardwired links. Due to multiple access, fading, noise and interference in wireless channel, throughput is low [1].

Furthermore, lack of fixed infrastructure may impose additional challenges, complexities and design constraints for MANET. The nodes share information with each other through store-and-forward mechanism in peer-to-peer mode, no default router is available and nodes acts as independent router and end system at same time and generates independent data. Due to multi-hop routing, network management is distributed across nodes which bring added difficulty in network management and fault detection. Mobility and fragility of nodes makes it difficult to be available every time in a network which makes it reasonably difficult for fault diagnosis and security protection in MANET. The network changes rapidly and unpredictably which results in route changes, network partitions and packet losses. It is difficult to predict when a node will join or leave a network or may cause route changes due to mobility. It is difficult to assume a node to have persistent data storage. We could see that MANET is inherently vulnerable to security attacks.

Each node in a network may have varying transmission and/or receiving capabilities and may operate in different frequency bands. The heterogeneous nature of nodes with different hardware and software configuration may result in asymmetric radio links and variability of processing capabilities which makes designing of network protocols and algorithms difficult for MANET requiring adaptability to changing wireless network environment. Also, each mobile node may have limited battery power supply which limits services and applications in MANET due to limited processing power of a node in network. Energy consumption directly relates to operational lifetime of the network. Currently, self-organization protocols may work well enough for only small wireless networks keeping many system parameters fixed such as node density, network coverage area and transceiver power characteristics. But, still we do not see much reliable practical applications of such small scale MANET.

Many MANET applications may require large scale wireless infrastructureless network, so scalability of these networks is essential to take into account in design parameters. Designing of such network of nodes with limited resources is not straight forward and require many challenges to be solved such as addressing, routing, location management, configuration management, interoperability, security, high capacity wireless technologies and network management [2].

To address the above challenges and constraints in MANET, research activities had been carried out on ad-hoc networking. Ad-hoc networks can be classified into body (BAN), Personal (PAN), Local (LAN), Metropolitan (MAN) and Wide (WAN) area networks. These types of networks are characterized by their network coverage area. WANs and MANs are multi-hop wireless networks that cover a large area e.g. 10 – 1000 km. The challenges that these networks present are not straight forward e.g. addressing, routing, location management, security etc. Thus, we do not see these types of networks in practice in near future. On the other hand, BAN, PAN and LAN

coverage area is approximately 1 – 500 m that are already common in the market [6]. Small multi-hop ad-hoc networks can be created using BAN, PAN and LAN, in fact, these can be considered as enabling technologies for ad-hoc networks in the market, because these can be the building blocks for large area networks such as WAN and MAN [7].

Considering BAN, PAN and LAN as enabling technologies for MANET, two main standards emerged in the market for ad-hoc networking. Bluetooth for short range wireless communications such as in BAN and PAN the range of which span from 1 – 10 m and IEEE 802.11 for wireless LAN (WLAN) the range of which span from 10 – 500 m. It has been speculated that IEEE 802.11 standard may potentially be exploited to build large area networks covering several square kilometers such as WAN and MAN through multi-hop radio. The MANET can also be classified by the number of active nodes in the ad-hoc network. Small to moderate scale ad-hoc networks may be composed of 2–100 active nodes. While large to very large ad-hoc networks composed of nodes larger than 1000 nodes. Experimental study in [9] shows that the per node throughput decays exponentially with large number of nodes in an ad-hoc network. With current technologies, only small to moderate scale ad-hoc networks can be implemented efficiently [2]. Furthermore, to study more on IEEE 802.11 based ad-hoc networking and its challenges in terms of performance and evaluation, [8] presents some open issues.

While the MANET research evolved through military oriented applications, research on its usage in non-military or commercial area has also grown substantially. With introduction of IEEE 802.11 technology, new ad-hoc networking applications appear on horizon mainly in specialized fields such as emergency and safety services. The growth in automotive market and increased demand of traffic/car safety, ad-hoc networking opens new challenges in car safety mechanisms on road or highway. Car-to-car communication would present a new way of ad-hoc networking in which cars could communicate with each other through wireless networking among them. Such ad-hoc networks are known as VANETs.

2.2 Brief overview

The VANET is a type of wireless ad-hoc network that is mainly under research from last couple of years. Recent advances in communication technologies, hardware and software are enabling the design and implementation of ubiquitous computing and we see different types of wireless networks in coming future. One such type of network is VANET which has received a lot of interest to improve vehicle and road safety, traffic efficiency and comfort to both drivers and passengers. With advancements in wireless devices and mobile networks, the need to support infotainment and digital wireless products in vehicles has also grown. The demand for V2V, V2I and VRC

communication in the form of Intelligent Transportation System (ITS) has grown to enable broad range of safety and non-safety applications such as vehicle safety, automated toll payment, traffic management, enhanced navigation, location-based services finding the fuel station, restaurant or travel lodge and infotainment applications providing access to internet [10], [11].

In ITS applications, the vehicles form a short-range wireless ad-hoc networks in which each vehicle send, receive or route information in the network. The vehicles are equipped with radio interface or OnBoard Unit (OBU) for communication among vehicles and RoadSide Units (RSUs). The OBUs consists of wireless networking technologies such as DSRC or WAVE (IEEE 802.11p) for vehicular ad-hoc networking. The RSUs are connected to backbone network and are fixed across the road side to facilitate wireless communication and internet for vehicles on road. The possible communication configuration in ITS applications includes inter-vehicle, vehicle-to-roadside and routing-based communication which rely on accurate and up-to-date information from positioning systems and require smart communication protocols for exchanging information [10].

The inter-vehicle communication uses multi-hop broadcasting to transmit traffic related information to vehicles. In ITS applications, the inter-vehicle communication is responsible for message forwarding for the activity on the road ahead in forward direction of vehicles traveling on road. This is logical in the sense that for emergency message dissemination about collision or route scheduling, the ITS should take care of traffic ahead on road instead of traffic coming from behind and forward such messages to traffic coming from behind for safety related consequences. The inter-vehicle communication constitutes two types of message forwarding: naïve broadcasting and intelligent broadcasting [10]. In naïve broadcasting, the messages are periodically broadcasted at regular intervals in time from the vehicles. The message is ignored by the vehicle if it is coming from vehicle behind it and forwards the message to vehicles behind it if the message comes from the vehicle in front of it. The naïve broadcasting has limitations in terms of message delivery in time because large broadcasting messages are generated which lowers the message dissemination efficiency by lowering delivery rate and increased delivery time due to message collision which is inherent problem in shared medium in wireless networks. The intelligent broadcasting limits the number of broadcasted messages for the emergency event through implicit acknowledgement. If the event-detecting vehicle receives the same message from behind, it stops broadcasting assuming that the vehicle behind it has received the message and would broadcast the messages to other vehicles coming from behind, thus, decreasing message collision probability and increasing the message delivery time. This fact highlights that the efficiency of wireless networking technology is crucial for success of emergency related applications on road in terms of number of vehicles broadcasting emergency messages in dense traffic.

In vehicle-to-roadside (RSU) communication, RSU sends a one-hop broadcast message to all equipped vehicles for services related to traffic efficiency and safety. For example, broadcasting dynamic speed limits or speed limit warnings to the vehicles in the vicinity of RSU if a vehicle violated the desired speed limit. The fixed RSUs are connected to backbone internetwork which also facilitates internet access to vehicles providing telecommunication and infotainment services, such as messaging, social networking, social media, chatting, music, video, whether information etc.

In routing-based communication, a multi-hop unicast message is delivered through multi-hop to the target vehicle. The query received by the vehicle containing the desired service or application sends a multi-hop unicast message containing the desired data to the vehicle it received request from. The unicast message is routed through multi-hop mechanism by the vehicles in ad-hoc wireless network until the target vehicle receives the desired data.

The DSRC standard was developed to support short to medium range communications in ITS applications such as vehicle-to-vehicle and vehicle-to-roadside communications in a short to medium range distance providing high data transfers with low communication latency. The DSRC standard is based on the IEEE 802.11a physical layer and 802.11 MAC layer. The DSRC spectrum is organized into 7 channels of 10 MHz bandwidth. One channel is used for safety communications (such as warning messages for drivers) and two other are used for other special purposes (such as life and public safety). The remaining are service channels used for safety or non-safety applications. Safety applications have higher priority over non-safety applications [10]. The traditional IEEE 802.11 MAC used in fixed wireless network or Wi-Fi suffers from significant overhead not suitable for vehicular wireless access environment to ensure fast data exchanges for safety related communications with variable driving speed, changing traffic patterns and varying driving environments. Thus, vehicular wireless network have greater challenges then fixed infrastructured wireless networks. To address such challenges, the group working on DSRC migrated to IEEE 802.11 standard group which renamed the DSRC to IEEE 802.11p WAVE. The WAVE is limited by the scope of IEEE 802.11, working strictly at the MAC and physical layers. The layers above the MAC and physical layers in DSRC or WAVE are handled by the upper layers of the IEEE 1609 standards [10], [12].

The VANET has been the active field of research and development nowadays and the recent research results that have been achieved by the VANET research community with the advancement in communications and computing technology includes routing, broadcasting, QoS and security [10].

Developing routing protocols could be challenging in VANET as compared to traditional wireless networks or wireless infrastructured networks. The VANET is a special class of MANET, so most of testing and research work for ad-hoc routing protocols in VANET has been used from MANET. The VANET issues such as network environment, mobility, different traffic density, vehicles arriving and leaving the network etc among others makes conventional ad-hoc routing protocols complex and inefficient [10]. Proactive routing protocols such as Destination-Sequenced Distance-Vector (DSDV) and Optimized Link State Routing protocol (OLSR) maintain and update route information among all nodes in a network at all times even if the paths are not in use. Maintenance of unused routes in the network uses significant part of the available bandwidth regardless of network load, size and frequently changing network topology which makes it inefficient for VANET. Reactive routing protocols such as Dynamic Source Routing (DSR) and Ad-hoc On-Demand Distance Vector (AODV) are suitable in dynamic network environment e.g. ad-hoc networks, VANETs, MANETs and reduces the computational complexity of maintaining routes in the network through algorithm that implement route determination on demand. Reactive routing protocols are particularly suitable for VANETs [10]. But, the above mentioned topology-based routing protocols such as Proactive and Reactive routing protocols can lead to broken routes and high overhead to repair these routes.

For dynamic vehicular network environment with short connectivity time and positioning systems, we need dedicated routing solutions for wireless multi-hop communications based on vehicles geographic positions. The position of the vehicle instead of route can be used with the help of Global Positioning System (GPS) by decoupling forwarding mechanism from the vehicles identity which makes it more scalable and efficient for highly volatile VANET environment. The beaconing and location service uses algorithm such as periodic broadcast of short packets with the vehicle identifier and its geographic position. For communication among vehicles to be established, the requesting vehicle issues a location query message with identifier and ‘hop limit’ to know the position of a required vehicle. The message is rebroadcasted to nearby vehicles until it reaches the required vehicle or ‘hop limit’. The required vehicle upon reception replies with a location reply message, answering its position and timestamp. The forwarding based on geographic position employs a geographic unicast or a geographic broadcast. In geographic unicast, the packet is transported between two vehicles via wireless hops using the location of the vehicle. The sending vehicle determines the location or position of the destination vehicle and forwards the unicast packet through multi-hop using neighboring vehicles. While in geographic broadcast, the data packets are flooded and rebroadcasted by vehicles if they are located in the geographic area determined by the packet [10].

For delivering warning and emergency messages in VANET, broadcasting is used for forwarding messages to the vehicles. As mentioned earlier, the volume of forwarded

messages in broadcasting can significantly increase message-collisions in IEEE 802.11 MAC which in turn lowers the message delivery rate and increases the delivery time. This happens when a vehicle trying to send a broadcasting message such as emergency message fails and it periodically retransmits it, thus, increasing the network congestion of shared wireless medium. Also, flooding due to broadcasted messages can decrease the performance of IEEE 802.11 MAC due to large number of messages trying to access the shared wireless medium or resource. As mentioned previously, the inter-vehicle communication in ITS uses new approach such as intelligent broadcasting protocol which can significantly reduce message collision and retransmission overhead [10].

To express performance of VANET in terms of QoS, robustness of route and time required to establish a route or reestablish a broken route are significant metrics to take into account. However, due to dynamic network environment and lack of consistent infrastructure, high levels of QoS cannot be guaranteed. This lack of performance arises from factors such as vehicles velocity, position and distance between vehicles, reliability of and delay between radio links which seriously affect the stability of particular route [10]. Further research is needed to improve QoS metrics in ad-hoc network environments such as VANET.

The security in VANET is of great concern and ignorance regarding security of drivers and vehicles in VANET environment may lead to critical life threatening situations. The information is gathered and shared among vehicles in VANET, therefore, there must be a system to determine malicious person in VANET trying to insert or modify information and raising concerns about data authenticity. The system must be reliable in maintaining privacy of drivers and vehicles. The security problems in VANET might be challenging to solve because of factors such as: network size, speed of vehicles, their relative geographic position and randomness in connectivity between them. Security threats in VANET could be reduced by determining such threats and identifying them in VANET environment would be a challenging task to take into account for trusted network [10]. According to [10] attacks in VANET could be threat to availability, threat to authenticity and threat to confidentiality.

The threat to availability raises concerns about the vehicle-to-vehicle and vehicle-to-roadside communications. The threat to availability as identified in [10] are: Denial of service attack, where attackers in a network as insiders or outsiders could make network unavailable for vehicles through flooding or jamming with artificially generated messages. Broadcast tempering, where attacker as insider in a network could inject false safety messages into the network such as false traffic warnings or false route information causing life threatening catastrophic results. Malware, such as viruses or worms could cause potential threat in operation of the VANET. Spamming, increasing

the transmission latency and black hole attack where a black hole is formed, nodes refuse to join the network or when drops from the network.

The threat to authenticity arises when an insider or outsider attacker fabricate, alter or suppress and introduce misinformation in the vehicular network using false identity. The threat to authenticity as indentified in [10] are: Masquerading, such as an attacker joining the network as a legitimate vehicle and conducts attacks such as black holes and false messages or misinformation. Replay attack, where an attacker re-injects the previously received packets into the network and corrupts the vehicles location table. Global Positioning System (GPS) spoofing, an attacker can fool other vehicles by altering their geographic position, producing false readings through a GPS satellite simulator. Tunneling, an attacker injects false data into vehicles onboard unit after the vehicles get lost the positioning information while entering into tunnel and before vehicles receives authentic GPS information. Position faking, an attacker can impersonate using other vehicles position information in unsecured VANET communication posing threat to vehicles authenticity. Message tempering, can result in modifying the messages exchanged in V2V and VRC communication. Message Suppression/Fabrication/Alteration, an attacker can physically disable inter-vehicle communication or modify VANET application in vehicles to refrain from sending to or receiving from other vehicles, the application beacons. Key and/or Certificate Replication, an attacker could undermine the system by tampering key management and/or certificate replication through broadcasting duplicate vehicles identity across several other vehicles, thus confusing authentication process and preventing identification of vehicles. Sybil Attack, an attacker can potentially partition the network and make delivery of safety messages impossible, since the safety messages are normally periodic one-hop broadcasts.

The threat to confidentiality as mentioned in [10] poses problems such as collection of messages through eavesdropping and gathering of location information available through broadcasted messages. Location privacy and anonymity must be achieved for secure VANET environment. Unsecured confidentiality and privacy in a network may pose serious threats by an attacker collecting information about vehicles on road without their knowledge and use them for malicious purposes.

The above mentioned types of threat in VANET would make the vehicular environment vulnerable to attackers in terms of availability, security and confidentiality of the network. Due to heterogeneous, dynamic and rapidly changing network environment, providing security system or mechanism in VANET could be a challenging task to be dealt with. Research and development needs to be done particularly for safety mechanisms for VANET.

In recent years, several intelligent transportation system initiatives and trials have been going on to experiment various aspects of VANET systems and its architecture. In "Phase 1", research and development of vehicular communications network is being done using simulations of real-life industrial trials. Development of these trials focused on the underlying wireless protocol infrastructure and included physical and MAC protocol standardization such as IEEE 802.11p, WAVE [10]. In "Phase 2", various projects involve standardization and field trials for verification of the protocols and architectures developed in "Phase 1", several consortia involving organizations such as automotive industry, highway control authorities, toll service providers and safety organizations are now involved to demonstrate real life VANET implementations in this regard and are being funded by governments of USA, Japan and the European Union as described in [10].

In USA, several projects and field trials have been or are in progress. These include: Wireless Access in Vehicular Environments (WAVE) (2004), which enabled practical trials of V2V and V2I to be demonstrated and its performance measured. Intelligent Vehicle Initiative (1998-2004), the objectives of the program were to develop technology to avoid driver distraction and facilitate deployment of crash avoidance systems. Vehicle Safety Communications (VSC) (2002-2004), (VSC-2) (2006-2009) consortium, it may be considered as "work in progress" and involves improvement of critical safety scenario through use of DSRC/WAVE IEEE 802.11p along with positioning systems, determining system requirements and performance measures for vehicular safety applications and deployment models for communication-based vehicle safety systems. And, Vehicle Infrastructure Integration (VII) (2004-2009) consortia, provides coordination between key automobile manufacturers (Ford, General Motors, Daimler-Chrysler, Toyota, Nissan, Honda, Volkswagen, BMW), IT suppliers, U.S. Federal and state transportation departments and professional associations [10]. The applications under development so far, include: collisions and unsafe driving conditions for drivers, warn drivers if they are about to run off the road, providing real-time information such as congestion, whether conditions and hazardous incidents, to system operators.

The European Union (EU) have been involved in projects and trials that include: Car-to-Car Communications Consortium (C2C-CC), comprising of European vehicle manufacturers started trials in 2001 and demonstrated IEEE 802.11 WLAN technologies to be used in vehicular wireless communication within range of few hundred of meters and has been actively involved in contributing to the European standardization bodies, particularly, ETSI TC ITS (European Telecommunications standard Institute: Technical Committee: Intelligent Transportation Systems) and a key contributor to the V2V and V2I validation trial process. FleetNet (2000-2003), a trial was built on the results of simulation experiments and software prototype called FleetNet Demonstrator to identify and evaluate problems in inter-vehicle

communication in realistic VANET environment such as highways and cities. Network on Wheels (NoW) (2004-2008), founded by the automobile manufacturers (Daimler, BMW, Volkswagen), the Fraunhofer Institute for Open Communication Systems, NEC Deutschland GmbH and Siemens AG in 2004 and now supported by the Federal Ministry of Education and Research Germany contributed to solve key issues related to communication protocols and data security for C2C communications in both safety related and infotainment applications, providing open communication platform for a broad spectrum of applications. PReVENT (2004-2008), demonstrated safety application using sensors, maps, and communication systems for evaluating safety including: safe speed and distance, collision and intersections, vehicles lane-change warning, development of Advanced Driver Assistance System (ADAS) with mapping and GPS location systems. Cooperative Vehicles and Infrastructure Systems (CVIS) (2006-2010), tested technologies for V2V and V2I communications by managing traffic control systems and implemented driver routing systems to avoid hazardous conditions. Car Talk 2000 (2000-2003), has been active in developing reliable components for Advanced Driver Assistance System (ADAS) such as Advanced Cruise Control (ACC).

Japan has been involved in development projects that include: Advanced Safety Vehicle Program (ASV-2) (1996-2000), (ASV-3) (2001-2005), (ASV-4) (2005-2007), supported by Japanese Ministry of Transport, automobile manufacturers (Honda, Mitsubishi, Suzuki and Toyota in particular) and academic and research organizations focused on driver inattention and errors such as drowsiness warning systems, vision enhancement systems, navigation systems, automatic collision avoidance systems, lane departure systems, impact absorption systems, pedestrian protection systems and door lock sensing systems. Demo 2000 and JARI (Japan Automobile Research Institute) demonstrated cooperative driver assistance system and evaluated the feasibilities and technologies necessary for inter-vehicle communications.

Moreover, the trials and outdoor experiments could be used for evaluating VANET protocols and applications, but it may be expensive and difficult to implement due to inherently distributed and complex VANET environment and topology. Therefore, VANET simulation tools are widely used before actual field trials. For good VANET simulation results, we need mobility model that is realistic as VANET network to higher degree. A role based mobility model to differentiate nodes based on their mobility roles, incapable of simulating complex traffic elements such as overpasses, bridges and tunnels as developed in [13]. A VANET simulation tool known as VGSim, accurately model traffic mobility and fulfils most of the requirements of accurate simulation, highly flexible and resource efficient in terms of adopting different mobility models, proposed by [14]. [15] describe mobility model separated at two levels: Macroscopic and Microscopic. The node mobility that includes streets, lights, roads, buildings etc are defined under macroscopic level and movement of vehicles and their behavior are defined under microscopic level. Also, mobility model can be created

through patterns from mobility traces by using available measurements that characterizes same statistical properties of the real VANET scenarios. Many simulators may exist, but they have limitations for simulating complete real VANET environment. These limitations can be categorized into traffic simulations that are used for transportation and traffic engineering and network simulation that are used to evaluate network protocols and applications. These two simulators work independently with different formats and a solution is needed to integrate them in order to inter-operate and evolve as VANET simulator [10].

As mentioned above, explaining characteristics of VANET as future networking platform, broadcasting techniques play an important part for safety applications. For emergency message dissemination, broadcasting emergency messages using WAVE IEEE 802.11p MAC protocol to access shared wireless medium may not be efficient with QoS Enhanced Distributed Channel Access (EDCA), because the probability of message collision increases with increased number of vehicles broadcasting the emergency message. But, further research on WAVE IEEE 802.11p MAC protocol modification may result in performance improvement such as low latency data dissemination and decrease in message collision as mentioned in [16].

2.3 Wireless access technology (IEEE 802.11p)

2.3.1 IEEE 802.11p description

In vehicular environment, the IEEE 802.11p aims to provide wireless communication in ranges up to 1000 m in (urban, suburban, rural, and motorway) for vehicles having velocity of up to 30 m/s. Therefore, frequent handshakes and authorization have to be limited to reduce link disconnections and low-latency and high reliability are required for safety-related applications in VANETs on the MAC sublayer, considering the fast movement and frequent trajectory changes.

The IEEE 802.11p uses an EDCA MAC sublayer protocol designed based on that of the IEEE 802.11e WLAN. However, the IEEE 802.11p provides some modifications to the transmission parameters based on characteristics of propagation environment in VANET systems. The physical layer of the IEEE 802.11p is similar to that of the IEEE 802.11a, operating at 5.9 GHz frequency band that is closer to 5 GHz frequency band of the IEEE 802.11a and adopts an orthogonal frequency-division multiplexing transmission technique. Moreover, based on the fact that in a vehicular communication environment, the relative velocity of vehicles could be significantly higher as compared to the node velocity in traditional WLAN, the delay spread of multiple paths could be significantly higher, hence the transmitted signal could suffer from intersymbol interference when the signal bandwidth is high, so, the bandwidth of a single channel in 802.11p is scaled down to 10 MHz from that of the IEEE 802.11a which appears to be a

reasonable choice for vehicular environments, supporting transmission rates ranging from 3 to 27 Mb/s over a bandwidth of 10 MHz [17].

The EDCA proposed in IEEE 802.11e provides priority based QoS support for four different access categories (ACs) for data traffic with four priorities. These ACs with prioritized data traffic has queues, each working as an independent DCF station with enhanced distributed channel access function (EDCAF) to contend for transmission opportunities with its own EDCA parameters. Prioritization with four different transmission queues for prioritized data traffic with four independent EDCAFs for different ACs is shown in figure 2.1. The EDCA based prioritized transmission mechanism for four different types of data traffic uses new interframe space, i.e. AIFS, denoted by AIFS[AC]. Each AC queue has its own values for AIFS, CWmin and CWmax.

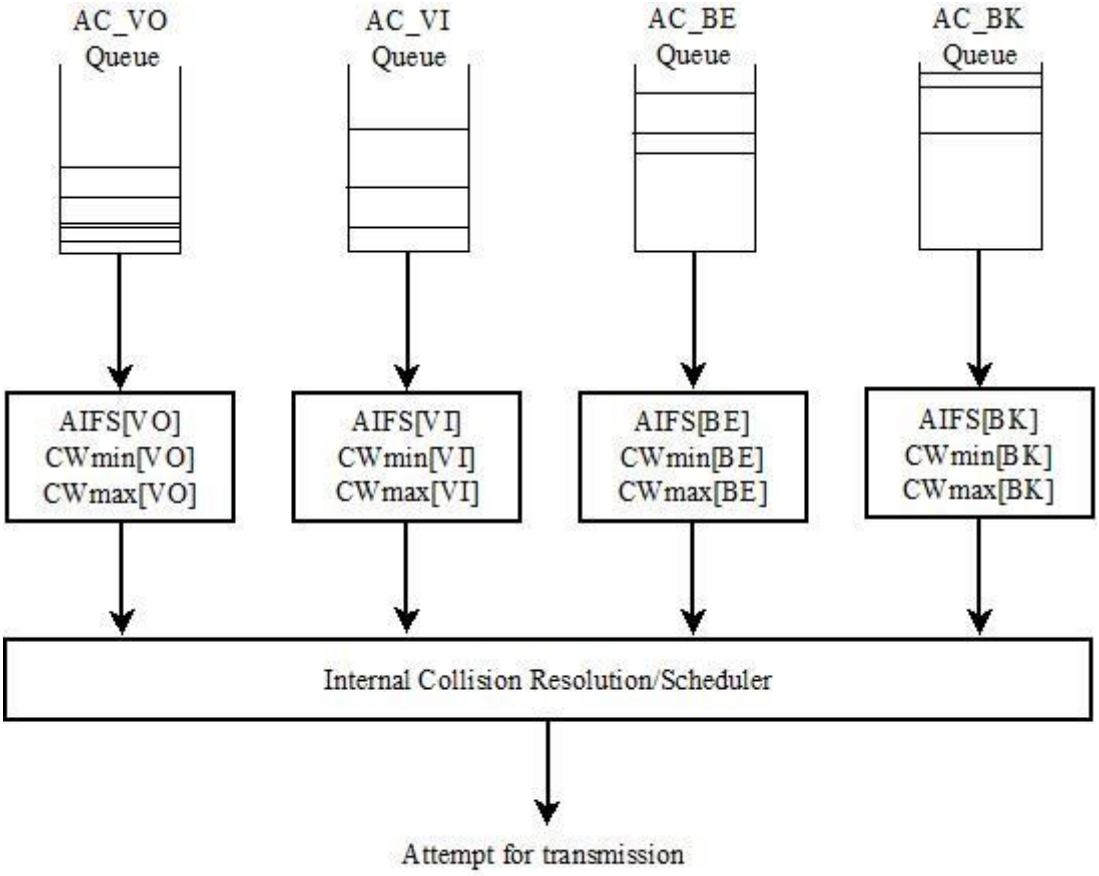

Figure 2.1. Enhanced Distributed Channel Access (EDCA) Categories

The new interframe space, i.e. AIFS, is an extension of the backoff procedure in DCF. The interframe spaces, such as, SIFS, PIFS, DIFS, AIFS and the prioritization based backoff procedure are shown in figure 2.2. The duration AIFS[AC] is derived from the following relation

AIFS[AC] = AIFSN[AC] x *aSlotTime* + *aSIFSTime*

Where AIFSN[AC] is the value for different ACs which is set by each MAC protocol in the EDCA parameter table, *aSlotTime* is the duration of a slot time and *aSIFSTime* is the length of SIFS. The smaller the value of AIFS, the higher the priority AC has to access the channel and vice versa. Similarly, the shorter the CW size for AC is, the higher the chance of AC to access the channel and vice versa.

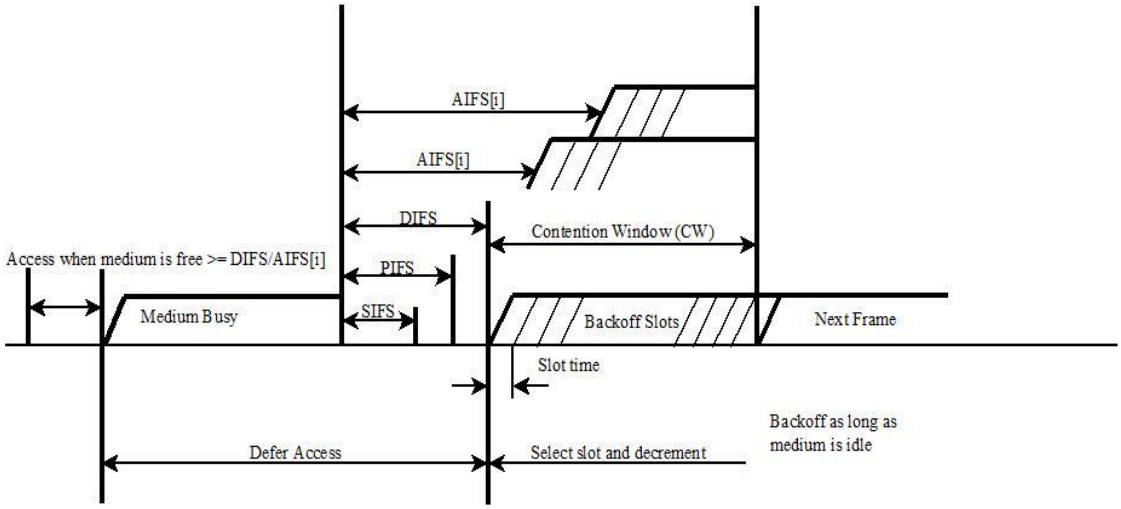

Figure 2.2. Inter-frame Spacing Relationships in EDCA categories

Table 2.1 shows the default EDCA parameters for IEEE 802.11p WAVE. The size for CWmin is 15 and the size for CWmax is 1023.

Table 2.1 IEEE 802.11p WAVE (Draft 8.0) Default EDCA Parameters

AC	CWmin[AC]	CWmax[AC]	AIFSN[AC]
AC_VO	(CWmin+1)/4 - 1	(CWmin+1)/2 - 1	2
AC_VI	(CWmin+1)/4 - 1	CWmin	3
AC_BE	(CWmin+1)/2 - 1	CWmax	6
AC_BK	CWmin	CWmax	9

2.3.2 Short comings in IEEE 802.11p

According to enhanced distributed channel access EDCA in 802.11p CSMA/CA protocol, the station competes randomly for channel access and the critical or emergency messages are prioritized according to different IFS intervals and different CW sizes for lower delivery delays and higher delivery probabilities. The critical or emergency messages shall be disseminated as quickly as possible and distributed to all

vehicles surrounded by a car detecting hazard, but CSMA/CA protocol used in IEEE 802.11p is far from the optimal solution due to the inherent properties of random access. In an arbitrary road environment which is crowded with vehicles equipped with the IEEE 802.11p unit, if a hazard is detected, the efficiency of dissemination of emergency information based on different probabilistic approaches to prevent the broadcast storm depends on the performance of one-hop IEEE 802.11p broadcasting. The one-hop IEEE 802.11p broadcasting performance has two unwanted dependencies for the distribution of emergency information: Smaller contention window size for higher priority messages and increase in number of nodes due to traffic jam badly degrades the packet reception probability [16]. Secondly, increase in number of nodes having same emergency data to be distributed does not positively affect the probability of packet delivery and competing nodes may degrade performance of different forms of forwarding e.g. two or more nodes with same IFS start retransmit immediately after receiving the message.

These problems are inherent for city centers, traffic jams and high density slow moving traffic and should be avoided by providing improvements in the protocol such as synchronous relaying to exploit denseness of networking nodes distributing the same information and the system still operates according to the current version of IEEE 802.11p [16].

These highlighted deficiencies can be overcome either by providing coordinated access mechanisms and get rid of random MAC which is not compatible with IEEE 802.11p CSMA/CA protocol or modify the random access mechanism such that the nodes cooperate with each other instead of competing in case of distribution of emergency information to speed up the emergency message propagation process by increasing one-hop transmission range [16].

3 Improving performance of IEEE 802.11p MAC

3.1 Back-off disabled for emergency messages

In QoS EDCA, each station has four AC queues with four EDCAFs which logically act as four independent stations. The channel is sensed for the duration of AIFS[i] as shown in figure 4.2. If a channel is sensed as idle for this duration and the AC(i) queue has backlogged data for transmission, the EDCAF tries to initiate a transmission sequence after checking its backoff timer. If the backoff timer has nonzero value, the EDCAF will decrease the backoff timer until the medium is sensed idle. The EDCAF initiates the transmission sequence when medium is sensed free and the backoff timer has zero value [17].

However, there is a probability that more than one EDCAF initiates the transmission sequence at the same time. If this is the case, a collision may occur inside a single station which is termed as internal collision. To avoid such internal collision inside a single station, the EDCA prioritize the access to wireless channel based on priorities defined in IEEE 802.11p for different AC queues inside each station e.g. granting the transmission opportunity to the highest priority access queue. In the meanwhile, the other AC queues invoke the backoff procedure due to the internal collision and behave as if there were an external collision on the wireless channel. In this way, the station content for wireless channel access based on priority for different type of data, hence probability of wireless channel access for different type of data is improved based on priority mechanism. It is worth to be noted here that, there is no priority mechanism for different stations competing for wireless channel access and they compete for medium access with equal opportunity. When more than one AC queue is granted transmission opportunity by different stations at same time, external collision happens, the collided frames are deferred, and the backoff procedure is invoked [17].

The operation of QoS EDCA in IEEE 802.11p MAC mentioned above shows that the performance will be degraded for one-hop broadcasting of emergency messages by many vehicles due to network congestion and deferred access due to backoff procedure in internal and/or external collisions. This problem is well explained in [16], which shows that the probability of message collision increases considerably when the number of nodes or vehicles broadcasting emergency message through one-hop broadcasting also increases e.g. traffic is blocked on road and many vehicles tries broadcasting emergency messages.

A simple solution to this problem could be to introduce the AC queue and EDCAF for emergency message dissemination in IEEE 802.11p MAC as highest priority queue and

also disable the backoff procedure for this queue i.e. AIFS, CW and backoff counter with zero value. When the emergency AC queue has message to send, it will sense the channel for current transmission. When the current transmission is over and medium is free, the emergency AC queue is granted the transmission opportunity. In this way, the emergency AC queue access will not be deferred with backoff counter value exponentially growing when the network is flooded with emergency messages. Hence, time latency of emergency message in IEEE 802.11p may improve reasonably. This technique is experimented in NS3 by modifying NS3 Wi-Fi module for IEEE 802.11p MAC protocol and incorporating the emergency message AC queue and EDCAF in QoS EDCA. The simulation results are described in chapter four.

3.2 Synchronous transmission

According to proposed cooperation mechanism in [16], nodes having the same emergency information compete to get access to the medium and a node transmits the message upon access to notify all the other nodes in the coverage area, but for retransmission they do not compete for the medium anymore and start transmitting the received message synchronously in a certain time slot in real time avoiding collisions. Adopting synchronous transmission, the increase in the number of nodes having same emergency information increases the probability of successful reception by other nodes by amplifying property of received signal and decrease in the number of competing nodes speeds up the dissemination process.

Several implementation approaches have been proposed in [16]. The nodes retransmitting the emergency information distribute it just after the current transmission is over. The nodes do not encounter any backoff delays, competition from the lower priority traffic and additional synchronization at MAC layer accessing the channel immediately. But this approach might block other hazard messages detected later and also requires restrictions on speed of information processing at network nodes.

Another approach is to sense the media for fixed time intervals, expressed in a certain number of time slots between redistribution of emergency information, and begin synchronized relaying when media is free. There is non-zero probability that low priority traffic or other emergency messages may occupy the medium first in a local contention environment, but the nodes waiting for relaying freeze their counters for transmission of these messages. More than one flow of emergency messages may result in permanent collisions, but the collisions could be minimized by maximizing the rate of information distribution by lowering the probability of channel access for nodes having low priority traffic. This could be achieved by nodes, choosing lower fixed time interval for high density nodes for sensing the medium to retransmit around the hazard. A certain node may participate in disseminating emergency information associated with

different hazards by storing backoff counters for different emergency packets and performing the mentioned procedure for each packet independently [16].

Finally, the node originating the hazard information may choose the backoff counter value for accessing the medium probabilistically and advertise it in the sent message which better illustrate the backward compatibility of the proposed scheme with CSMA/CA based IEEE 802.11p mechanism. With the proposed scheme, the contention environment with nodes having different traffic degenerates to the contention environment having lesser number of nodes comparatively [16].

Synchronization between nodes at both MAC and physical layer is required. The synchronization at MAC is achieved by counting the number of slot since the last received message. Ensuring synchronization of the beginning of slots at the physical layer is partially alleviated by usage of OFDM in 802.11p, but still requires novel techniques to address this issue [16].

The concept behind synchronous transmission technique can be well described with the help of figure 3.1.

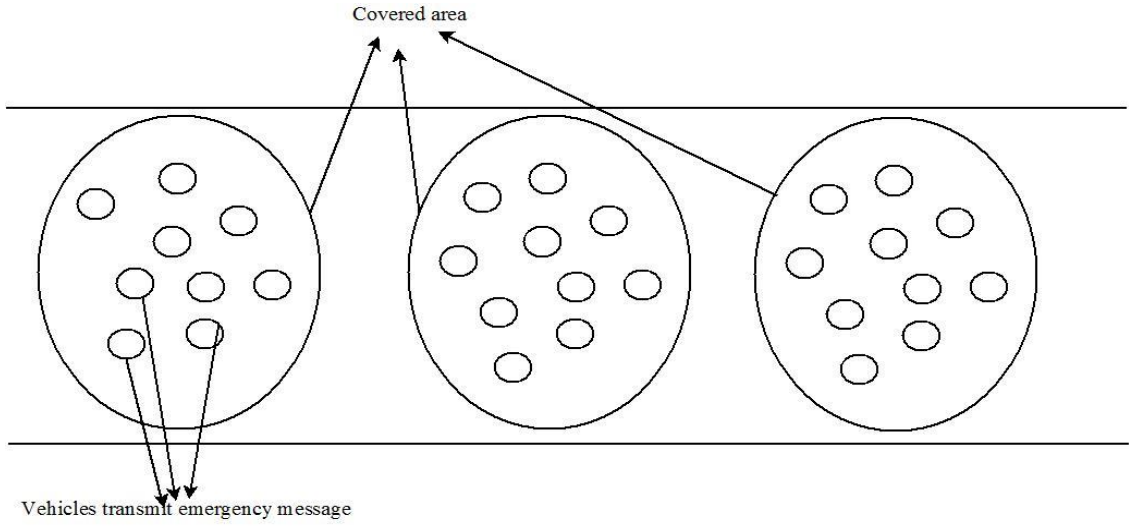

Figure 3.1 Vehicles in certain covered area transmit emergency message in same time slot

From the figure 3.1, it can be seen that vehicles in particular covered area in meters receives the emergency message. One of the vehicles in that particular area competes for transmission opportunity. When it gets access to the medium, the vehicles in that area start transmitting emergency messages in the same time slot synchronously. Thus, the number of nodes or vehicles competing for access to the wireless medium would decreases in number and instead of competing for the medium, all the nodes in the covered area start transmitting the same emergency message in a coordinated way that affects the dissemination of information in terms of speed and lower delay.

4 Implementation and performance results

4.1 Network simulator 3 (NS3)

The proposed techniques for improved performance in 802.11p MAC protocol have been evaluated using NS3. NS3 is a discrete-event network simulator, an open source environment meant primarily for research and educational purposes. NS3 is a new simulator, different from NS2; both are written in C++ and meant for studying and research on communication networks. There are two main differences between NS2 and NS3: The choice of scripting language and enhanced network capabilities.

The NS2 is dependent on Otcl scripting for writing simulation scripts and results of animation could be visualized in network animator nam. The NS3 simulator is written entirely in C++ with optional python bindings and simulations could be run entirely from C++, unlike NS2. Some NS3 simulations could still be visualized in nam, but new animators are under development.

NS3 has improved capabilities as compared to NS2 e.g. improved multiple interfaces on nodes, use of IP addressing, internet protocols, more detailed and realistic IEEE 802.11 models with several rate adaptation algorithms and support for MAC and physical layers. Some NS2 models could be ported to NS3 and also the NS3 is on active developments from multiple fronts. The NS3 simulator is being recommended for new simulator projects due to more realistic and enhanced network modules. Moreover, scientific or research community could actively participate to develop new models, debug or maintain existing ones and share the results for development of NS3.

The NS3 source code is being changed and modified by the contributing community very often and to handle such complex software system NS3 uses Mercurial for its source code management through repository. The make software build system is well known but difficult to use in a large and highly configurable system and alternatively NS3 uses waf build system which is a new generation Python-based build system.

A good knowledge of C++, object oriented programming, socket programming and to some extent Python programming language is required for NS3 simulations scripting either in C++ or Python. The NS3 uses most of the components from GNU tool chain such as gcc, binutils and gdb, however, NS3 uses Python-based waf build system instead of using GNU based make and autotools.

It is recommended to work in Linux or a Linux like environment for using NS3 simulator for building the source code for application development purposes. For

those using windows, it is recommended to install a virtual machine environment such as VMware in windows and install Linux there. NS3 supports development in the Cygwin environment and also provides popular Linux system commands, but it has been reported as problematic in emulation and its interaction with other Windows softwares and products.

It is worth noted that the mobility models provided by NS3 are not suitable for mobility of vehicles yet. In NS3, the node mobility depends on the node itself, whereas, for mobility of vehicles the mobility model of the node must depend on surrounding nodes and the conditions on the road e.g. the messages in the network may be able to affect the mobility of the nodes on the roads for emergency messages such as reduction of speed [18]. According to [18], the car-following models, such as the Intelligent Driver Model (IDM) provides realistic movement.

4.2 Open source code for traffic mobility

As mentioned above in chapter 3 in overview of Vehicular Ad-hoc Networks, a good VANET simulation results, we need mobility model that is realistic as VANET network to a higher degree. We use the mobility model IDM and MOBIL lane change model source code, an open source code implemented in NS3 by Arbabi and Weigle as mentioned in [18], to evaluate the performance of IEEE 802.11p modifications in a VANET using NS3 source code. In an open source code implemented in NS3 by Arbabi and Weigle, a Highway class represents a straight multi-lane and bi-directional roadway that manages the behavior and mobility of vehicles on the road.

The source code implemented by Arbabi and Weigle consists of five main classes as mentioned in [18]. The five classes are Vehicle, Obstacle, Model, LaneChange and Highway. All these classes are being implemented using C++ objected oriented programming in NS3 environment.

Vehicle is a mobile node that contains a wireless communication device. The Highway class is responsible for managing the position, direction and the lane numbers of vehicles on road. The acceleration and velocity of vehicles can be set manually or it can be calculated using the IDM mobility rules. The lane change is done using the MOBIL lane change model. The vehicles can be manually created or automatically injected onto the highway in the source code. Moreover, the vehicles have wireless communication devices such as Wi-Fi and they are able to send and/or receive messages to communicate with each other through the NS3 Wi-Fi channels. By setting the appropriate event handlers to the implemented callbacks in NS3, the wireless network environment is able to capture sent and received packets and all related events in the network. Vehicles can unicast a packet or broadcast messages, and scheduling the sending process and receiving the event in callback can be manually controlled.

Different callbacks can be used for the purpose of tracing different network layers and mobility of vehicles which helps in creating and tracing simulation scenarios [18].

Obstacle is similar to vehicle except that it has no mobility. It also contains a wireless communication device and is inherited from the Vehicle class. It contains all the features that are present in the Vehicle class except that it cannot be mobile. Its main usage in the traffic mobility code is to use it as barrier to close a lane or to temporarily create stoppages that result in congestion on the highway. But, it can also be used as roadside unit (RSU) along, but outside of, the highway. Like in Vehicle, it must have direction and lane number except for mobility features [18].

Model class implements the car-following mobility model such as IDM for mobility of vehicles on the road. In such model, the vehicle's acceleration and deceleration depends upon its own velocity, its desired velocity and the position and velocity of the vehicle immediately in front in the same lane. The Model uses the IDM equations to calculate and return new acceleration at each time step, which in turn calculates the vehicle's new position and velocity based on its current state and the state of front vehicle on the road using the parameters defined in IDM equations. The Model class is customizable, such that, each vehicle on road has its own set of IDM parameters for specific experiments e.g. sports cars, police cars, truck, buses and emergency vehicles could have different mobility characteristics [18].

LaneChange class implements the MOBIL lane changing model for a vehicle. The lane changing model in MOBIL takes care of the safety criterion and the incentive criterion. The safety criterion takes care of the safe deceleration of the vehicle at back of the front vehicle when changing lanes. The incentive criterion is satisfied if the vehicle changing the lane has greater advantage then the other vehicles' disadvantages. The vehicle advantage is simply the difference between the vehicle's current acceleration and the vehicle's new acceleration after the lane change has occurred. The vehicle disadvantage is simply the difference between the vehicle's current acceleration before lane change and the vehicle's new acceleration after the lane change. In vehicle disadvantage, the back vehicles in both the current lane and new lane are taken into consideration. In addition to lane change model parameters, the IDM rules also apply to the mobility of vehicles while changing the lanes [18].

Highway is the class essential for correct behavior of vehicles mobility and lane change as it holds all the vehicles on road. Highway class represents the straight bidirectional highway with physical properties as length, number of lanes, lane width and median gap and implements several vehicle management functions to automatically create vehicle objects with certain parameters and insert them into lanes, for bidirectional use vehicles are added to both sides of highway with equal rate. For automatic vehicle injection onto road, the Highway class creates default mobility and lane change models with

parameters set appropriately for the car and truck. Also, the automatically created vehicles are provided with default Wi-Fi Phy/Mac settings appropriate for vehicles in vehicular ad-hoc network. Besides automatic creation of vehicles, the vehicles can also be created and injected onto road or lane in Highway manually. Each lane in Highway is a list structure as lists in C++ and when vehicle is added to Highway, it is inserted in its proper place according to lane, direction and its position on Highway. When simulation is run in NS3 environment, the Highway calls its step function instantaneously, which updates the position, velocity and acceleration of the vehicles according to the mobility model. Also, the vehicle's lane changing occurs according to the MOBIL lane change model and when a lane change is allowed, it occurs before mobility is updated, so that vehicles only change positions instantaneously one time in one simulation step. Highway provides user access to vehicles through its vehicle ID to change vehicles' parameters and allow feedback between the network and the mobility model. Several events provided by Highway can be triggered and bound to the event handlers created by the user. The communication channel, the Phy/Mac layer and the behavior of the network devices can also be traced through appropriate trace events in Highway [18].

The three main events in Highway class are InitVehicle, ControlVehicle and ReceiveData. The InitVehicle event is triggered at Highway initialization time. This provides ability for customization of simulation and modifications to initial settings for classes used for traffic mobility in NS3 e.g. Vehicle, Obstacle, Highway. This event handler is the ideal place for creating and placing initial objects on the highway e.g. adding vehicles on lanes at Highway manually, changing the Highway or Vehicle parameters and incrementing the vehicle ID. The ControlVehicle event is triggered by the step function of Highway instantaneously. This event provides the user to control VANET simulation scenarios by having access to the mobility, lane change, velocity, acceleration and position of vehicles at each time step during simulation. ReceiveData event is triggered when any vehicle on Highway with a wireless communication device receives data from the network [18]. Similarly, this event provides flexibility for a user to add its own functionality for simulation scenarios e.g. tracing the data packets, forwarding received messages to other vehicles in a network, broadcasting of emergency messages to vehicles.

4.3 NS3 Wi-Fi modification

The technique described in section 3.1 is based on introduction of emergency AC queue and EDCAF for emergency messages with AIFS, CW and backoff counter with zero value. The modification can be visualized with the help of figure 4.1.

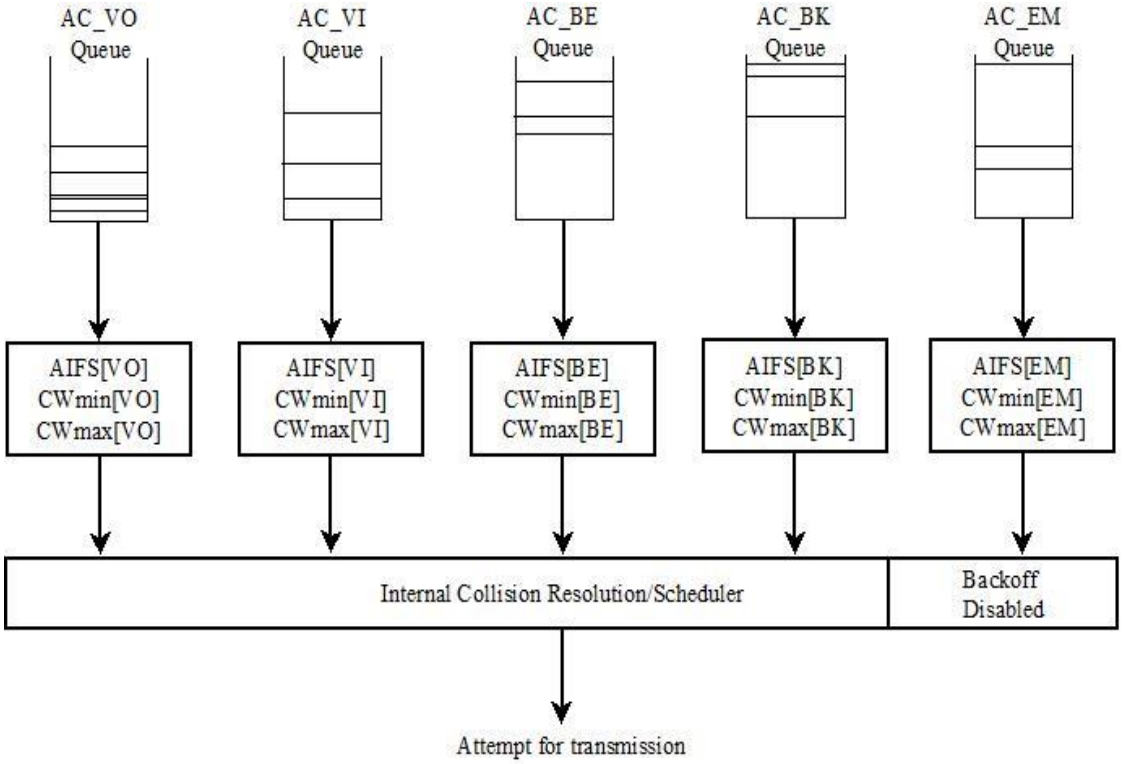

Figure 4.1. Introduction of EDCAF and Emergency Message Queue in IEEE 802.11p based QoS EDCA Categories

In figure 4.1, the emergency message (EM) queue is introduced into IEEE 802.11p based QoS EDCA. The backoff procedure is disabled for the AC_EM queue and the emergency message is granted access by its EDCAF when current transmission in the medium is over. If there is no ongoing transmission in the medium, the channel is sensed by corresponding EDCAF until DIFS interval and then granted the transmission opportunity.

The EDCA parameters for emergency message can be introduced to the table 2.1 as shown in table 4.1.

Table 4.1 IEEE 802.11p WAVE (Draft 8.0) Default EDCA Parameters

AC	**CWmin[AC]**	**CWmax[AC]**	**AIFSN[AC]**
AC_VO	(CWmin+1)/4 - 1	(CWmin+1)/2 - 1	2
AC_VI	(CWmin+1)/4 - 1	CWmin	3
AC_BE	(CWmin+1)/2 - 1	CWmax	6
AC_BK	CWmin	CWmax	9
AC_EM	0	0	0

The overall QoS EDCA mechanism with introduction of emergency message queue and its EDCAF can be seen in figure 4.2. It also shows the emergency message dissemination process.

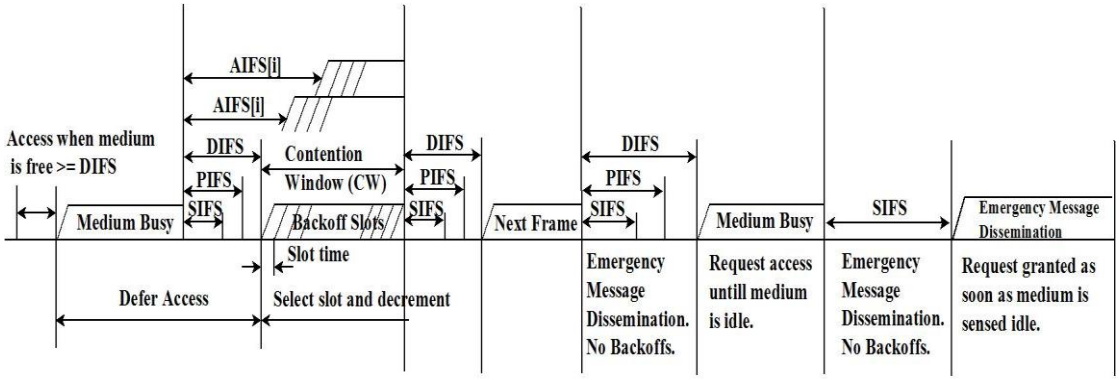

Figure 4.2. No backoffs for Emergency Messages in QoS EDCA

4.4 Simulation and performance results

A simple VANET network environment is simulated in a discrete-event network simulator e.g. NS3. The simulation can be well explained with the help of figure 4.3. The two lane one-directional highway is represented by Highway class in Mobility and Lane change model explained in section 4.2. The length of highway is 1000 meters, with two lanes. Each lane contains three vehicles with some distance from each other. The vehicles only move in their own lanes with some speed, velocity and acceleration. Each vehicle is equipped with a wireless device e.g. IEEE 802.11p for wireless communication with transmission range of 250 – 300 meters. There is an obstacle in a mid-highway in lane 2 at 500 meters, which represents blockage due to an accident. It periodically broadcasts message containing emergency warning at intervals of every 1.0 seconds for vehicles approaching it. The vehicles upon receiving the emergency warning message decelerate in lane 2 and broadcast this message to other vehicles. The QoS tag for this broadcasted message is AC_EM. The other vehicles upon receiving such message also broadcast it to other nearby vehicles.

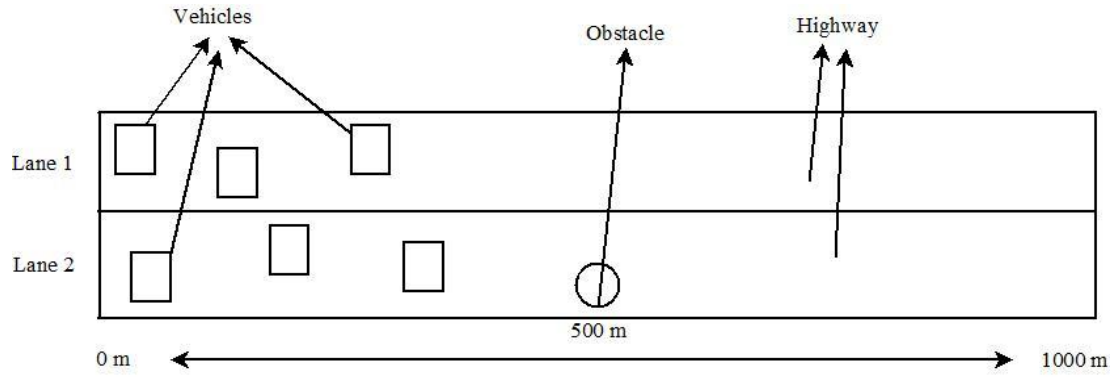

Figure 4.3 One directional highway with two lanes

The VANET network environment shown in figure 4.3 is simulated for time 0 – 60 seconds, with discrete step size of 0.1 seconds. The ControlVehicle event in Highway class mentioned in section 4.2 is triggered every 0.1 seconds, which controls the

movement of vehicles on highway e.g. acceleration, speed, velocity and lane change. The ReceiveData event in Highway class mentioned in section 4.2 is triggered when a vehicle receives a message in VANET network shown in figure 4.3. These two events are modified to simulate with respect to the above mentioned network scenario on highway. The other event InitVehicle in Highway class mentioned in section 4.2 is used for initializing the settings for classes Vehicle, Obstacle and Highway.

With such a simulation scenario the performance of IEEE 802.11p MAC QoS EDCA is evaluated with modifications described in section 3.1. The performance results does not guarantee the improvements in IEEE 802.11p MAC QoS EDCA by applying technique mentioned in section 3.1. The section 4.4.1, 4.4.2, 4.4.3 and 4.4.4 only evaluates this technique and concludes its outcomes based on simulation results.

4.4.1 Emergency message assigned AC_BK

The emergency message is assigned AC as AC_BK in IEEE 802.11p. The average access grant time for AC_BK as compared to AC_EM for emergency messages is shown in table 4.2 and table 4.3.

Table 4.2 Average Access Grant Time for Emergency Message (AC_BK) in Classic IEEE 802.11p

AC Access Category	Classic IEEE 802.11p
AC_EM Average EDCA Access Grant Time	45.46
AC_BE Average EDCA Access Grant Time	29.5
AC_VI Average EDCA Access Grant Time	29.7504
AC_VO Average EDCA Access Grant Time	29.9750

Table 4.3 Average Access Grant Time for Emergency Message (AC_BK) in Improved IEEE 802.11p

AC Access Category	Improved IEEE 802.11p
AC_EM Average EDCA Access Grant Time	46.1179
AC_BK Average EDCA Access Grant Time	28.5001
AC_VI Average EDCA Access Grant Time	29.7504
AC_VO Average EDCA Access Grant Time	29.9750

The technique mentioned in section 3.1 slightly improves the average access grant time for emergency messages in terms of transmission opportunity in the wireless medium. The results would have been even more deductive if there would have been backoff

activated for other AC as compared to AC_EM in dense or congested network simulation scenario.

4.4.2 Emergency message assigned AC_BE

The emergency message is assigned access category as AC_BE in IEEE 802.11p. The average access grant time for AC_BE as compared to AC_EM for emergency messages is shown in table 4.4 and table 4.5.

Table 4.4 Average Access Grant Time for Emergency Message (AC_ BE) in Classic IEEE 802.11p

AC Access Category	Classic IEEE 802.11p
AC_EM Average EDCA Access Grant Time	45.4599
AC_BK Average EDCA Access Grant Time	29.500
AC_VI Average EDCA Access Grant Time	29.7504
AC_VO Average EDCA Access Grant Time	29.9750

Table 4.5 Average Access Grant Time for Emergency Message (AC_ BE) in Improved IEEE 802.11p

AC Access Category	Improved IEEE 802.11p
AC_EM Average EDCA Access Grant Time	46.1179
AC_BK Average EDCA Access Grant Time	28.5001
AC_VI Average EDCA Access Grant Time	29.7504
AC_VO Average EDCA Access Grant Time	29.9750

Again, the technique mentioned in section 3.1 slightly improves the average access grant time for emergency messages in terms of transmission opportunity in the wireless medium.

4.4.3 Emergency message assigned AC_EM

The emergency message is assigned access category as AC_EM in IEEE 802.11p. The average access grant time for emergency messages is shown in table 4.6.

Table 4.6 Average Access Grant Time for Emergency Message (AC_ EM) in Improved IEEE 802.11p

AC Access Category	Classic IEEE 802.11p
AC_EM Average EDCA Access Grant Time	46.1179
AC_BK Average EDCA Access Grant Time	28.5001
AC_VI Average EDCA Access Grant Time	29.7504
AC_VO Average EDCA Access Grant Time	29.9750

It may be noted that according to results obtained in section 4.4.1, 4.4.2 and 4.4.3, the average access grant time for emergency message dissemination over a period of time may not be improved significantly. In section 4.4.4, we evaluate the improvements to IEEE 802.11 MAC QoS EDCA by experiment the medium access mechanism more closely in terms of access grant time and backoff procedure.

4.4.4 Emergency messages not affected by backoff

The results are evaluated not based on vehicle mobility, but stationary vehicle broadcasting different types of traffic depending on AC in a network. The access grant times in the wireless medium through improved IEEE 802.11p MAC QoS EDCA are then closely observed for the traffic by the output log file obtained through NS3.

Output from NS3 log file has been taken and shown below to show the results for emergency messages dissemination. Actual simulation runs for 60 seconds, but for demonstration we show only few output text lines. Different types of traffic with different AC e.g. voice, video, background, best effort and emergency messages are being broadcasted by a stationary vehicle on a highway. The experiment is being done to demonstrate that the new AC such as AC_EM for emergency messages does not affect by backoff and gets access to the wireless medium when channel is free or current transmission is over. In this simulation, the video message is broadcasted every 0.5 seconds, the voice message every 0.05 seconds, the background traffic every 0.11 seconds and the emergency message also every 0.11 seconds intentionally. We intentionally make intervals for emergency message traffic and background traffic same, so that they may compete for accessing the medium at same time. We can see from the following output that emergency message gets timely access to the medium, while background messages may backoff due to the shared wireless medium.

```
Emergency access granted at t = 0
Voice access granted at t = 0.000297
Video access granted at t = 0.000573
Best effort access granted at t = 0.000919
Background access granted at t = 0.001648
Voice access granted at t = 0.05
Voice access granted at t = 0.1
```

Background access granted at t = 0.11
Emergency access granted at t = 0.110232
Voice access granted at t = 0.15
Voice access granted at t = 0.2
Background access granted at t = 0.22
Emergency access granted at t = 0.220232
Voice access granted at t = 0.25
Voice access granted at t = 0.3
Background access granted at t = 0.33
Emergency access granted at t = 0.330232
Voice access granted at t = 0.35
Voice access granted at t = 0.4
Background access granted at t = 0.44
Emergency access granted at t = 0.440232
Voice access granted at t = 0.45
Video access granted at t = 0.5
Voice access granted at t = 0.500281
Background access granted at t = 0.55
Emergency access granted at t = 0.550232
Voice access granted at t = 0.66
Emergency access granted at t = 0.660224
Emergency access granted at t = 0.77
Emergency access granted at t = 0.88
Emergency access granted at t = 0.99
Voice access granted at t = 1
Voice access granted at t = 1.00028
Voice access granted at t = 1.00056
Voice access granted at t = 1.00083
Video access granted at t = 1.0011
Voice access granted at t = 1.00136
Voice access granted at t = 1.00162
Voice access granted at t = 1.00191
Voice access granted at t = 1.00219
Voice access granted at t = 1.00246
Best effort access granted at t = 1.00278
Background access granted at t = 1.00335
Background access granted at t = 1.00388
Background access granted at t = 1.00435
Background access granted at t = 1.00474
Voice access granted at t = 1.05
Background access granted at t = 1.1
Emergency access granted at t = 1.10023
Voice access granted at t = 1.21
Emergency access granted at t = 1.21022
Emergency access granted at t = 1.32
Emergency access granted at t = 1.43
Voice access granted at t = 1.5
Video access granted at t = 1.50026
Voice access granted at t = 1.50053
Voice access granted at t = 1.50079
Voice access granted at t = 1.50107

```
Voice access granted at t = 1.50135
Voice access granted at t = 1.50162
Voice access granted at t = 1.5019
Voice access granted at t = 1.50216
Background access granted at t = 1.5025
Background access granted at t = 1.50302
Background access granted at t = 1.50349
Background access granted at t = 1.54
Emergency access granted at t = 1.54023
Voice access granted at t = 1.55
Voice access granted at t = 1.6
Background access granted at t = 1.65
Emergency access granted at t = 1.65023
Voice access granted at t = 1.76
Emergency access granted at t = 1.76022
Emergency access granted at t = 1.87
Emergency access granted at t = 1.98
Voice access granted at t = 2
Voice access granted at t = 2.00026
Voice access granted at t = 2.00051
Video access granted at t = 2.00078
Voice access granted at t = 2.00104
Voice access granted at t = 2.00131
Voice access granted at t = 2.00157
Voice access granted at t = 2.00186
Best effort access granted at t = 2.00219
Background access granted at t = 2.00284
Background access granted at t = 2.00323
Background access granted at t = 2.00372
Voice access granted at t = 2.05
Background access granted at t = 2.09
Emergency access granted at t = 2.09023
Voice access granted at t = 2.1
Voice access granted at t = 2.15
Background access granted at t = 2.2
Emergency access granted at t = 2.20023
Voice access granted at t = 2.31
Emergency access granted at t = 2.31022
Emergency access granted at t = 2.42
Voice access granted at t = 2.5
Voice access granted at t = 2.50029
Video access granted at t = 2.50056
Voice access granted at t = 2.50082
Voice access granted at t = 2.50111
Voice access granted at t = 2.50138
Voice access granted at t = 2.50165
Background access granted at t = 2.50211
Background access granted at t = 2.50258
Background access granted at t = 2.53
Emergency access granted at t = 2.53023
Voice access granted at t = 2.55
```

Voice access granted at t = 2.6
Background access granted at t = 2.64
Emergency access granted at t = 2.64023
Voice access granted at t = 2.65
Voice access granted at t = 2.7
Background access granted at t = 2.75
Emergency access granted at t = 2.75023
Voice access granted at t = 2.86
Emergency access granted at t = 2.86022
Emergency access granted at t = 2.97
Voice access granted at t = 3
Voice access granted at t = 3.00028
Voice access granted at t = 3.00054
Voice access granted at t = 3.00081
Voice access granted at t = 3.00108
Video access granted at t = 3.00138
Background access granted at t = 3.00174
Best effort access granted at t = 3.00208
Background access granted at t = 3.00282
Voice access granted at t = 3.05
Background access granted at t = 3.08
Emergency access granted at t = 3.08023
Voice access granted at t = 3.1
Voice access granted at t = 3.15
Background access granted at t = 3.19
Emergency access granted at t = 3.19023
Voice access granted at t = 3.2
Voice access granted at t = 3.25
Background access granted at t = 3.3
Emergency access granted at t = 3.30023
Voice access granted at t = 3.41
Emergency access granted at t = 3.41022
Voice access granted at t = 3.5
Voice access granted at t = 3.50026
Video access granted at t = 3.50054
Voice access granted at t = 3.50079
Voice access granted at t = 3.50108
Background access granted at t = 3.50148
Background access granted at t = 3.52
Emergency access granted at t = 3.52023
Voice access granted at t = 3.55
Voice access granted at t = 3.6
Background access granted at t = 3.63
Emergency access granted at t = 3.63023
Voice access granted at t = 3.65
Voice access granted at t = 3.7
Background access granted at t = 3.74
Emergency access granted at t = 3.74023

4.4.5 Emergency messages blocks other traffic

The snapshot shown below is being taken from NS3 simulation output log file. The simulation scenario is same as mentioned in section 4.4.4.1, except that the background messages are broadcasted every 3 seconds and emergency messages are broadcasted every 0.1 seconds. It is being observed that when the access time of emergency messages coincides with access times of other messages, the emergency messages completely blocks other messages with different AC types.

```
Emergency access granted at t = 0
Voice access granted at t = 0.000297
Video access granted at t = 0.000573
Best effort access granted at t = 0.000919
Background access granted at t = 0.001648
Voice access granted at t = 0.05
Emergency access granted at t = 0.1
Voice access granted at t = 0.100297
Voice access granted at t = 0.15
Emergency access granted at t = 0.2
Voice access granted at t = 0.200297
Voice access granted at t = 0.25
Emergency access granted at t = 0.3
Voice access granted at t = 0.300271
Voice access granted at t = 0.35
Emergency access granted at t = 0.4
Voice access granted at t = 0.400297
Voice access granted at t = 0.45
Video access granted at t = 0.5
Emergency access granted at t = 0.500216
Emergency access granted at t = 0.6
Emergency access granted at t = 0.7
Emergency access granted at t = 0.8
Emergency access granted at t = 0.9
Voice access granted at t = 1
Emergency access granted at t = 1.00022
Emergency access granted at t = 1.1
Emergency access granted at t = 1.2
Emergency access granted at t = 1.3
Emergency access granted at t = 1.4
Emergency access granted at t = 1.5
Emergency access granted at t = 1.6
Emergency access granted at t = 1.7
Emergency access granted at t = 1.8
Emergency access granted at t = 1.9
Emergency access granted at t = 2
Emergency access granted at t = 2.1
Emergency access granted at t = 2.2
Emergency access granted at t = 2.3
Emergency access granted at t = 2.4
Emergency access granted at t = 2.5
```

Emergency access granted at t = 2.6
Emergency access granted at t = 2.7
Emergency access granted at t = 2.8
Emergency access granted at t = 2.9
Voice access granted at t = 3
Emergency access granted at t = 3.00022
Emergency access granted at t = 3.1
Emergency access granted at t = 3.2
Emergency access granted at t = 3.3
Emergency access granted at t = 3.4
Emergency access granted at t = 3.5
Emergency access granted at t = 3.6
Emergency access granted at t = 3.7
Emergency access granted at t = 3.8
Emergency access granted at t = 3.9
Emergency access granted at t = 4
Emergency access granted at t = 4.1
Emergency access granted at t = 4.2
Emergency access granted at t = 4.3
Emergency access granted at t = 4.4
Emergency access granted at t = 4.5
Emergency access granted at t = 4.6
Emergency access granted at t = 4.7
Emergency access granted at t = 4.8
Emergency access granted at t = 4.9
Emergency access granted at t = 5
Emergency access granted at t = 5.1
Emergency access granted at t = 5.2
Emergency access granted at t = 5.3
Emergency access granted at t = 5.4
Emergency access granted at t = 5.5
Emergency access granted at t = 5.6
Emergency access granted at t = 5.7
Emergency access granted at t = 5.8
Emergency access granted at t = 5.9
Emergency access granted at t = 6
Emergency access granted at t = 6.1
Emergency access granted at t = 6.2
Emergency access granted at t = 6.3
Emergency access granted at t = 6.4
Emergency access granted at t = 6.5
Emergency access granted at t = 6.6
Emergency access granted at t = 6.7
Emergency access granted at t = 6.8
Emergency access granted at t = 6.9
Emergency access granted at t = 7
Emergency access granted at t = 7.1
Emergency access granted at t = 7.2
Emergency access granted at t = 7.3
Emergency access granted at t = 7.4
Emergency access granted at t = 7.5

```
Emergency access granted at t = 7.6
Emergency access granted at t = 7.7
Emergency access granted at t = 7.8
Emergency access granted at t = 7.9
Emergency access granted at t = 8
Emergency access granted at t = 8.1
Emergency access granted at t = 8.2
Emergency access granted at t = 8.3
Emergency access granted at t = 8.4
Emergency access granted at t = 8.5
Emergency access granted at t = 8.6
Emergency access granted at t = 8.7
Emergency access granted at t = 8.8
Emergency access granted at t = 8.9
Emergency access granted at t = 9
Emergency access granted at t = 9.1
Emergency access granted at t = 9.2
Emergency access granted at t = 9.3
Emergency access granted at t = 9.4
Emergency access granted at t = 9.5
Emergency access granted at t = 9.6
Emergency access granted at t = 9.7
Emergency access granted at t = 9.8
Emergency access granted at t = 9.9
Emergency access granted at t = 10
Emergency access granted at t = 10.1
Emergency access granted at t = 10.2
Emergency access granted at t = 10.3
Emergency access granted at t = 10.4
Emergency access granted at t = 10.5
Emergency access granted at t = 10.6
Emergency access granted at t = 10.7
Emergency access granted at t = 10.8
Emergency access granted at t = 10.9
Emergency access granted at t = 11
Emergency access granted at t = 11.1
Emergency access granted at t = 11.2
Emergency access granted at t = 11.3
Emergency access granted at t = 11.4
Emergency access granted at t = 11.5
Emergency access granted at t = 11.6
Emergency access granted at t = 11.7
Emergency access granted at t = 11.8
Emergency access granted at t = 11.9
Emergency access granted at t = 12
Emergency access granted at t = 12.1
Emergency access granted at t = 12.2
Emergency access granted at t = 12.3
Emergency access granted at t = 12.4
Emergency access granted at t = 12.5
Emergency access granted at t = 12.6
```

```
Emergency access granted at t = 12.7
Emergency access granted at t = 12.8
Emergency access granted at t = 12.9
Emergency access granted at t = 13
Emergency access granted at t = 13.1
Emergency access granted at t = 13.2
Emergency access granted at t = 13.3
Emergency access granted at t = 13.4
Emergency access granted at t = 13.5
Emergency access granted at t = 13.6
Emergency access granted at t = 13.7
Emergency access granted at t = 13.8
Emergency access granted at t = 13.9
Emergency access granted at t = 14
Emergency access granted at t = 14.1
Emergency access granted at t = 14.2
Emergency access granted at t = 14.3
Emergency access granted at t = 14.4
Emergency access granted at t = 14.5
Emergency access granted at t = 14.6
Emergency access granted at t = 14.7
Emergency access granted at t = 14.8
Emergency access granted at t = 14.9
Emergency access granted at t = 15
Emergency access granted at t = 15.1
```

5 Conclusion and future work

Disabling backoff for emergency message dissemination as mentioned in section 3.1 may be beneficial for simple emergency applications. Such emergency applications could utilize the technique mentioned in section 3.1 for notifying other vehicles on road about the warning or accident ahead as a buzz signal. Such technique could not be used for proper message communication amongst vehicles, because the vehicles broadcasting the emergency messages with improved IEEE 802.11p MAC QoS EDCA devices may transmit the same emergency message at different times with different distances from each other. Such differences in time and distance between cars may result in intersymbol interference in the original message. The original message will be corrupted by intersymbol interference.

On the other way, synchronous transmission technique mentioned in section 3.2 is a promising technique. The technique seems to be promising for improvement in emergency message dissemination process, whereas, with novel physical synchronization technique at physical layer, the message may not be get corrupted by wireless propagation delay. The synchronous transmission, relaying emergency messages in vehicular ad-hoc network in a coordinated or synchronized way may positively affect the message dissemination speedup and higher reception probability in dense vehicular traffic environment.

Simulation of synchronous transmission of vehicles or nodes transmitting emergency messages with backward compatibility to IEEE 802.11p protocol is left for future work.

6 Appendix

Vanet-adhoc-802_11p.cc file contains C++ code for starting VANET simulation in NS3.

```
/* -*- Mode: C++; c-file-style: "gnu"; indent-tabs-mode: nil; -*- */

/*
 * Copyright (c) 2005-2009 Old Dominion University [ARBABI]
 *
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License version 2 as
 * published by the Free Software Foundation;
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307 USA
 *
 * Author: Hadi Arbabi <marbabi@cs.odu.edu>
 */

/*
This the starting point of the simulation and experiments. The main function will parse
the input and parameter settings. Creates a highway and set the highway parameters.
Then bind the events (callbacks) to the created controller and designed handlers. Sets
the highway start and end time, and eventually runs the simulation which is basically
running a highway with a controller. You can add your functions to controller to create
various scenarios.
*/

#include <fstream>
#include <iostream>
#include <iomanip>
#include "ns3/core-module.h"
#include "ns3/common-module.h"
#include "ns3/node-module.h"
#include "ns3/helper-module.h"
#include "ns3/mobility-module.h"
#include "ns3/contrib-module.h"
#include "ns3/wifi-module.h"
#include "Highway.h"
#include "Controller.h"
```

```
NS_LOG_COMPONENT_DEFINE ("HADI");

using namespace ns3;
using namespace std;

static void Start(Ptr<Highway> highway)
{
        highway->Start();
}

static void Stop(Ptr<Highway> highway)
{
         highway->Stop();
}

int main (int argc, char *argv[])
{
        float simTime = 0.0;
         bool plot = false;

        // Process command-line args
        CommandLine cmd;
        cmd.AddValue ("time", "simulation time", simTime);
        cmd.AddValue ("plot", "generate output fot gnuplot", plot);
        cmd.Parse(argc, argv);

        Ptr<Highway>    highway    = CreateObject<Highway>();
        Ptr<Controller> controller = CreateObject<Controller>();

        controller->SetHighway(highway);
        controller->Plot = plot;

        highway->SetNumberOfLanes(2);
        highway->SetTwoDirectional(false);
        highway->SetHighwayLength(1000.0);
        highway->SetLaneWidth(5.0);
        highway->SetAutoInject(false);
        highway->SetInjectionGap(10.0);
        highway->SetChangeLane(false);
        highway->SetDeltaT(0.1);

        // Change the transmission range of wifi shared in the Highway.
        // 250-300 meter transmission range
        highway->GetYansWifiPhyHelper().Set("TxPowerStart", DoubleValue(21.5));

        // 250-300 meter transmission range
        highway->GetYansWifiPhyHelper().Set("TxPowerEnd", DoubleValue(21.5));

        // Bind the Highway/Vehicle events to the event handlers. Controller's will catch
        // them.
```

```
        highway>SetControlVehicleCallback(MakeCallback(&Controller::ControlVehic
        le, controller));

        highway->SetInitVehicleCallback(MakeCallback(&Controller::InitVehicle,
        controller));

        highway->SetReceiveDataCallback(MakeCallback(&Controller::ReceiveData,
        controller));

        ns3::PacketMetadata::Enable();
        Config::SetDefault("ns3::WifiRemoteStationManager::FragmentationThreshol",
        StringValue ("2200"));
        Config::SetDefault ("ns3::WifiRemoteStationManager::RtsCtsThreshold",
        StringValue("2200"));

        Simulator::Schedule(Seconds(0.0), &Start, highway);
        Simulator::Schedule(Seconds(simTime), &Stop, highway);
        Simulator::Stop(Seconds(simTime));
        Simulator::Run();
        Simulator::Destroy();

        return 0;
}
```

The Controller.cc file contains code related to the Highway class events also mentioned in section 4.2.

```
/* -*- Mode: C++; c-file-style: "gnu"; indent-tabs-mode: nil; -*- */
/*
 * Copyright (c) 2005-2009 Old Dominion University [ARBABI]
 *
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License version 2 as
 * published by the Free Software Foundation;
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307 USA
 *
 * Author: Hadi Arbabi <marbabi@cs.odu.edu>
 */

#include <iostream>
#include <sstream>
```

```
#include <vector>
#include "Controller.h"

using namespace std;
using namespace ns3;

namespace ns3
{

	bool initAccidentMessage = true;

	Controller::Controller()
	{
		 T=-1.0;
		Plot=false;
	}

	Controller::Controller(Ptr<Highway> highway)
	{
		this->highway=highway;
	}

	void Controller::SetHighway(Ptr<Highway> highway)
	{
		this->highway=highway;
	}

	Ptr<Highway> Controller::GetHighway()
	{
		return this->highway;
	}

	bool Controller::InitVehicle(Ptr<Highway> highway, int& VID)
	{
		//cout << "highway initVehicle" <<endl;

		// Block the road with warning, 500 meters away, at the right most lane
		// [lane=0, dir=1]
		Ptr<Obstacle> obstacle=CreateObject<Obstacle>();
		obstacle->SetupWifi( highway->GetWifiHelper(),
		highway->GetYansWifiPhyHelper(),
		highway->GetQosWifiMacHelper());

		obstacle->SetVehicleId(VID++);
		obstacle->SetDirection(1);
		obstacle->SetLane(0);
		obstacle->SetPosition(Vector(500.0, highway->GetYForLane(0,1), 0));
		obstacle->SetLength(8);
		obstacle->SetWidth(2);
		obstacle->SetReceiveCallback(highway->GetReceiveDataCallback());
```

```
highway->AddVehicle(obstacle);

Simulator::Schedule( Seconds(0.0),
&Controller::BroadcastWarningObstacle, this, obstacle);
Simulator::Schedule(Seconds(0.0), &Controller::backGroundTraffic,
this, obstacle);
Simulator::Schedule(Seconds(0.0), &Controller::bestEffortTraffic, this,
obstacle);
Simulator::Schedule(Seconds(0.0), &Controller::videoTraffic, this,
obstacle);
Simulator::Schedule(Seconds(0.0), &Controller::voiceTraffic, this,
obstacle);
Simulator::Schedule(Seconds(0.0),Controller::BroadcastEmergencyObst
acle, this, obstacle);

// Three vehicles in lane 1
int numVehicleLane_1 = 3;
vector< Ptr<Vehicle> > vehicleLane_1;
for (int i = 0; i < numVehicleLane_1; i++) {
        Ptr<Vehicle> temp = CreateObject<Vehicle>();
        temp->SetupWifi(highway->GetWifiHelper(),
        highway->GetYansWifiPhyHelper(),
        highway->GetQosWifiMacHelper());
        temp->SetVehicleId(VID++);
        temp->SetDirection(1);
        temp->SetLane(0);
        temp->SetPosition(Vector(0.0, highway->GetYForLane(0,1), 0));
        temp->SetVelocity(0.0);
        temp->SetAcceleration(0.0);
        temp->SetModel(highway->GetSedanModel());
        temp->SetLaneChange(highway->GetSedanLaneChange());
        temp->SetLength(5);
        temp->SetWidth(2);
        temp>SetReceiveCallback(highway>GetReceiveDataCallback());
        vehicleLane_1.push_back(temp);
}

// Three vehicles in lane 2
int numVehicleLane_2 = 3;
vector< Ptr<Vehicle> > vehicleLane_2;
for (int i = 0; i < numVehicleLane_2; i++) {
        Ptr<Vehicle> temp = CreateObject<Vehicle>();
        temp->SetupWifi(highway->GetWifiHelper(),
        highway->GetYansWifiPhyHelper(),
        highway->GetQosWifiMacHelper());
        temp->SetVehicleId(VID++);
        temp->SetDirection(1);
        temp->SetLane(1);
        temp->SetPosition(Vector(0.0, highway->GetYForLane(0,1), 0));
        temp->SetVelocity(0.0);
```

```
                temp->SetAcceleration(0.0);
                temp->SetModel(highway->GetSedanModel());
                temp->SetLaneChange(highway->GetSedanLaneChange());
                temp->SetLength(5);
                temp->SetWidth(2);
                temp>SetReceiveCallback(highway>GetReceiveDataCallback());
                vehicleLane_2.push_back(temp);
        }

        int numVehiclesLane = 3;
        for (int i = 0; i < numVehiclesLane; i++) {
                highway->AddVehicle(vehicleLane_1.at(i));
                highway->AddVehicle(vehicleLane_2.at(i));
        }

        return true;
}

bool Controller::ControlVehicle(Ptr<Highway> highway, Ptr<Vehicle> vehicle,
double dt)
{

        // we aim to create outputs which are readable by gnuplot for visulization
        // purpose this can be happen at beginning of each simulation step here.
        if(Plot==true)
        {
                bool newStep=false;
                double now=Simulator::Now().GetHighPrecision().GetDouble();
                if(now > T)
                {
                        T = now;
                        newStep=true;
                }

                if(newStep==true)
                {
                        if(T!=0.0)
                        {
                                cout << "e" << endl;
                        }
                        float xrange = highway->GetHighwayLength();
                        float yrange =
                        highway->GetLaneWidth()*highway-
                        >GetNumberOfLanes();

                        if(highway->GetTwoDirectional())
                                yrange=2*yrange + highway->GetMedianGap();
                        cout << "set xrange [0:"<< xrange <<"]" << endl;
                        cout << "set yrange [0:"<< yrange <<"]" << endl;
                        cout << "plot '-' w points" << endl;
```

```
                newStep=false;
            }

            if(newStep==false)
            {
                cout << vehicle->GetPosition().x << " " <<
                vehicle->GetPosition().y << endl;
            }
        }

        // To decelerate and stop the vehicles reaching the obstacle at lane 0
        if( (vehicle->GetVehicleId() == 2 && vehicle->GetPosition().x >=350) ||
        (vehicle->GetVehicleId() == 3 && vehicle->GetPosition().x >=350) ||
        (vehicle->GetVehicleId() == 4 && vehicle->GetPosition().x >=350) )
        {
            vehicle->SetAcceleration(-2.0);

            return true;
        }

        // return false: a signal to highway that lets the vehicle automatically be
        // handled (using IDM/MOBIL rules)
        return false;
}

void Controller::BroadcastWarningObstacle(Ptr<Vehicle> veh)
{
        stringstream msg;
        msg << "Obstacle/accident has happened, lane 0 is blocked. "
        << "Message sent from vehicle "
        << veh->GetVehicleId()
        << " direction = " << veh->GetDirection()
        << " lane = " << veh->GetLane();

        Ptr<Packet> packet = Create<Packet>((uint8_t*) msg.str().c_str(),
        msg.str().length());

        QosTag qos;
        uint8_t tid = 3;
        qos.SetTid(tid);
        packet->AddPacketTag(qos);

        veh->SendTo(veh->GetBroadcastAddress(), packet);


Simulator::Schedule(Seconds(1.0),&Controller::BroadcastWarningObstacle,
this, veh);
}

void Controller::BroadcastEmergencyObstacle(Ptr<Vehicle> veh)
```

```
{
        stringstream msgEm;
        msgEm << "Emergency message";

        Ptr<Packet> packetEm1=Create<Packet>((uint8_t*)msgEm.str().c_str(),
        msgEm.str().length());

        QosTag qosEm;
        uint8_t tidEm = 8;
        qosEm.SetTid(tidEm);

        packetEm1->AddPacketTag(qosEm);

        veh->SendTo(veh->GetBroadcastAddress(), packetEm1);
        Simulator::Schedule(Seconds(0.1),&Controller::BroadcastEmergencyOb
        stacle, this, veh);
}

void Controller::backGroundTraffic(Ptr<Vehicle> veh) {
        stringstream msgEm;
        msgEm << "Background traffic";

        Ptr<Packet> packetEm1=Create<Packet>((uint8_t*)msgEm.str().c_str(),
        msgEm.str().length());

        QosTag qosEm;
        uint8_t tidEm = 1;
        qosEm.SetTid(tidEm);

        packetEm1->AddPacketTag(qosEm);

        veh->SendTo(veh->GetBroadcastAddress(), packetEm1);

        Simulator::Schedule(Seconds(3.0),&Controller::backGroundTraffic,this,
        veh);
}

 void Controller::bestEffortTraffic(Ptr<Vehicle> veh) {
        stringstream msgEm;
        msgEm << "Best effort traffic";

        Ptr<Packet> packetEm1=Create<Packet>((uint8_t*)msgEm.str().c_str(),
        msgEm.str().length());

        QosTag qosEm;
        uint8_t tidEm = 0;
        qosEm.SetTid(tidEm);

        packetEm1->AddPacketTag(qosEm);
```

```
        veh->SendTo(veh->GetBroadcastAddress(), packetEm1);

        Simulator::Schedule(Seconds(5.0),&Controller::bestEffortTraffic, this,
        veh);
}

void Controller::videoTraffic(Ptr<Vehicle> veh) {
        stringstream msgEm;
        msgEm << "Video traffic";

        Ptr<Packet> packetEm1 =Create<Packet>((uint8_t*)msgEm.str().c_str(),
        msgEm.str().length());

        QosTag qosEm;
        uint8_t tidEm = 5;
        qosEm.SetTid(tidEm);

        packetEm1->AddPacketTag(qosEm);

        veh->SendTo(veh->GetBroadcastAddress(), packetEm1);

        Simulator::Schedule(Seconds(0.5),&Controller::videoTraffic, this, veh);
}

void Controller::voiceTraffic(Ptr<Vehicle> veh) {
        stringstream msgEm;
        msgEm << "Voice traffic 1";

        Ptr<Packet> packetEm1 =Create<Packet>((uint8_t*)msgEm.str().c_str(),
        msgEm.str().length());

        QosTag qosEm;
        uint8_t tidEm = 7;
        qosEm.SetTid(tidEm);

        packetEm1->AddPacketTag(qosEm);

        veh->SendTo(veh->GetBroadcastAddress(), packetEm1);

        Simulator::Schedule(Seconds(0.05),&Controller::voiceTraffic1, this,
        veh);
}


void Controller::VehicleBroadcastingMessage(Ptr<Vehicle> veh, Ptr<Packet>
packet)
{
        veh->SendTo(veh->GetBroadcastAddress(), packet);
}
```

```
        void Controller::ReceiveData(Ptr<Vehicle> veh, Ptr<const Packet> packet,
        Address address)
        {
                string data = string((char*)packet->PeekData());
                stringstream ss (stringstream::in | stringstream::out);

                double obs_id, obs_x;
                ss << data;
                ss >> obs_id;
                ss >> obs_x;

                int vid = veh->GetVehicleId();

                QosTag qos;
                bool peek = packet->PeekPacketTag(qos);
                uint8_t tid = qos.GetTid();

                uint8_t tidBe1 = 0;
                uint8_t tidBe2 = 3;

                if ( peek && ( (tid == tidBe1) || (tid == tidBe2) ) )
                {
                        QosTag qosEm;
                        uint8_t tidEm = 8;
                        qosEm.SetTid(tidEm);
                        Ptr<Packet>packetBroadcast=Create<Packet>((uint8_t*)data.c_s
                        tr(), data.length());
                        packetBroadcast->AddPacketTag(qosEm);
                }
        }
}
```

The qos-tag.hh code specifies TID for different traffic flow in Wi-Fi module including modifications for emergency messages.

```
/* -*- Mode: C++; c-file-style: "gnu"; indent-tabs-mode: nil; -*- */
/*
 * Copyright (c) 2009 MIRKO BANCHI
 *
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License version 2 as
 * published by the Free Software Foundation;
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
```

```
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307 USA
 *
 * Author: Mirko Banchi <mk.banchi@gmail.com>
 */
#ifndef QOS_TAG_H
#define QOS_TAG_H

#include "ns3/packet.h"

namespace ns3 {

        class Tag;


        /**
         * As per IEEE Std. 802.11-2007, Section 6.1.1.1.1, when EDCA is used the
         * the Traffic ID (TID) value corresponds to one of the User Priority (UP)
         * values defined by the IEEE Std. 802.1D-2004, Annex G, table G-2.
         *
         * Note that this correspondence does not hold for HCCA, since in that
         * case the mapping between UPs and TIDs should be determined by a
         * TSPEC element as per IEEE Std. 802.11-2007, Section 7.3.2.30
         */
        enum UserPriority {
          UP_BK = 1, /**< background */
          UP_BE = 0, /**< best effort (default) */
          UP_EE = 3, /**< excellent effort */
          UP_CL = 4, /**< controlled load */
          UP_VI = 5, /**< video, < 100ms latency and jitter */
          UP_VO = 6, /**< voice, < 10ms latency and jitter */
          UP_NC = 7, /**< network control */
          UP_EM = 8 /**< Emergency message */
        };

          /**
           * The aim of the QosTag is to provide means for an Application to
           * specify the TID which will be used by a QoS-aware WifiMac for a
           * given traffic flow. Note that the current QosTag class was
           * designed to be completely mac/wifi specific without any attempt
           * at being generic.
           */
        class QosTag : public Tag
        {
                public:
                        static TypeId GetTypeId (void);
                        virtual TypeId GetInstanceTypeId (void) const;

                        /**
                        * Create a QosTag with the default TID = 0 (best effort traffic)
```

```
                        */
                        QosTag ();

                        /**
                        * Create a QosTag with the given TID
                        */
                        QosTag (uint8_t tid);

                        /**
                        * Set the TID to the given value. This assumes that the
                        * application is aware of the QoS support provided by the MAC
                        * layer, and is therefore able to set the correct TID.
                        *
                        * @param tid the value of the TID to set
                        */
                        void SetTid (uint8_t tid);

                        /**
                        * Set the TID to the value that corresponds to the requested
                        * UserPriority. Note that, since the mapping of UserPriority to
                        * TID is defined for EDCA only, you should call this method
                        * only when EDCA is used. When using HDCA, QosTag(uint8_t
                        * tid) should be used instead.
                        *
                        * @param up the requested UserPriority
                        *
                        */
                        void SetUserPriority (UserPriority up);

                        virtual void Serialize (TagBuffer i) const;
                        virtual void Deserialize (TagBuffer i);
                        virtual uint32_t GetSerializedSize () const;
                        virtual void Print (std::ostream &os) const;

                        uint8_t GetTid (void) const;

                private:
                        uint8_t m_tid;
        };

} //namespace ns3

#endif /* QOS_TAG_H */
```

The qos-utils.cc file contains the code to get AC index and TID for the traffic flow in Wi-Fi module including emergency messages.

```
/* -*- Mode: C++; c-file-style: "gnu"; indent-tabs-mode: nil; -*- */
/*
```

```
 * Copyright (c) 2009 MIRKO BANCHI
 *
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License version 2 as
 * published by the Free Software Foundation;
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307 USA
 *
 * Author: Mirko Banchi <mk.banchi@gmail.com>
 * Author: Cecchi NiccolÃ² <insa@igeek.it>
 */
#include "qos-utils.h"
#include "qos-tag.h"

namespace ns3 {

        AcIndex QosUtilsMapTidToAc (uint8_t tid)
        {
                switch (tid) {
                        case 0 :
                                return AC_BE;
                        break;
                        case 1 :
                                return AC_BK;
                        break;
                        case 2 :
                                return AC_BK;
                        break;
                        case 3 :
                                return AC_BE;
                        break;
                        case 4 :
                                return AC_VI;
                        break;
                        case 5 :
                                return AC_VI;
                        break;
                        case 6 :
                                return AC_VO;
                        break;
                        case 7 :
                                return AC_VO;
                        break;
```

```
                case 8 :
                        return AC_EM;
                break;
        }
        return AC_UNDEF;
}

uint8_t QosUtilsGetTidForPacket (Ptr<const Packet> packet)
{
        QosTag qos;
        uint8_t tid = 9;
         if (packet->PeekPacketTag (qos))
        {
                if (qos.GetTid () < 9)
                {
                        tid = qos.GetTid ();
                }
        }
        return tid;
}

uint32_t QosUtilsMapSeqControlToUniqueInteger (uint16_t seqControl,
uint16_t endSequence)
{
        uint32_t integer = 0;
        uint16_t numberSeq = (seqControl>>4) & 0x0fff;
        integer = (4096 - (endSequence + 1) + numberSeq) % 4096;
        integer *= 16;
        integer += (seqControl & 0x000f);
        return integer;
}

bool QosUtilsIsOldPacket (uint16_t startingSeq, uint16_t seqNumber)
{
        NS_ASSERT (startingSeq < 4096);
        NS_ASSERT (seqNumber < 4096);
        uint16_t distance = ((seqNumber - startingSeq) + 4096) % 4096;
        return (distance >= 2048);
}

} //namespace ns3
```

Modifications in NS3 edca-txop-n.cc file for emergency messages are shown below.

```
void EdcaTxopN::NotifyAccessGranted (void)
{

        NS_LOG_FUNCTION (this);
```

```
if (m_currentPacket == 0)
{
        if (m_queue->IsEmpty () && !m_baManager->HasPackets ())
        {
                NS_LOG_DEBUG ("queue is empty");
                return;
        }

        if (m_baManager->HasBar (m_currentBar))
        {
                SendBlockAckRequest (m_currentBar);
                return;
        }

        /* check if packets need retransmission are stored in BlockAckManager
        */
        m_currentPacket = m_baManager->GetNextPacket (m_currentHdr);
        if (m_currentPacket == 0)
        {
        if (m_queue->PeekFirstAvailable (&m_currentHdr,
        m_currentPacketTimestamp, m_qosBlockedDestinations) == 0)
        {
                NS_LOG_DEBUG ("no available packets in the queue");
                return;
        }
        if (m_currentHdr.IsQosData () && !m_currentHdr.GetAddr1
        ().IsBroadcast () && m_blockAckThreshold > 0 && !m_baManager-
        >ExistsAgreement (m_currentHdr.GetAddr1 (),
        m_currentHdr.GetQosTid ()) && SetupBlockAckIfNeeded ())
        {
                return;
        }
        m_currentPacket = m_queue->DequeueFirstAvailable (&m_currentHdr,
        m_currentPacketTimestamp, m_qosBlockedDestinations);
        NS_ASSERT (m_currentPacket != 0);

        uint16_t sequence = m_txMiddle->GetNextSequenceNumberfor
        (&m_currentHdr);
        m_currentHdr.SetSequenceNumber (sequence);
        m_currentHdr.SetFragmentNumber (0);
        m_currentHdr.SetNoMoreFragments ();
        m_currentHdr.SetNoRetry ();
        m_fragmentNumber = 0;

        NS_LOG_DEBUG ("dequeued size="<<m_currentPacket->GetSize ()<<
        ", to="<<m_currentHdr.GetAddr1 ()<<
        ", seq="<<m_currentHdr.GetSequenceControl ());

        if (m_currentHdr.IsQosData () && !m_currentHdr.GetAddr1
        ().IsBroadcast ())
```

```
                {
                        VerifyBlockAck ();
                }
        }
  }

MacLowTransmissionParameters params;
params.DisableOverrideDurationId ();

if (m_currentHdr.GetAddr1 ().IsGroup ())
{
        params.DisableRts ();
        params.DisableAck ();
        params.DisableNextData ();
        m_low->StartTransmission (m_currentPacket, &m_currentHdr, params,
        m_transmissionListener);
        QosTag qos;
        const uint8_t tidAC_EM  = 8;
        const uint8_t tidAC_BK1 = 1;
        const uint8_t tidAC_BK2 = 2;
        const uint8_t tidAC_BE1 = 0;
        const uint8_t tidAC_BE2 = 3;
        const uint8_t tidAC_VI1 = 4;
        const uint8_t tidAC_VI2 = 5;
        const uint8_t tidAC_VO1 = 6;
        const uint8_t tidAC_VO2 = 7;

        if (m_currentPacket->PeekPacketTag (qos))
        {
                switch (qos.GetTid ())
                {
                        case tidAC_BE1 : {
                                double now = Simulator::Now().GetSeconds();
                                cout << "Best effort access granted at t = " << now <<
                                endl;
                                m_currentPacket = 0;
                                m_dcf->ResetCw ();
                                m_dcf->StartBackoffNow (m_rng->GetNext (0, m_dcf-
                                >GetCw ()));
                                StartAccessIfNeeded ();
                                NS_LOG_DEBUG ("Best effort tx");
                                break;
                        }
                        case tidAC_BE2 : {
                                double now = Simulator::Now().GetSeconds();
                                cout << "Best effort access granted at t = " << now <<
                                endl;
                                m_currentPacket = 0;
                                m_dcf->ResetCw ();
                                m_dcf->StartBackoffNow (m_rng->GetNext (0, m_dcf-
```

```
                >GetCw ()));
                StartAccessIfNeeded ();
                NS_LOG_DEBUG ("Best effort tx");
                break;
        }
        case tidAC_BK1 : {
                double now = Simulator::Now().GetSeconds();
                cout << "Background access granted at t = " << now <<
                endl;
                m_currentPacket = 0;
                m_dcf->ResetCw ();

                m_dcf->StartBackoffNow (m_rng->GetNext (0, m_dcf-
                >GetCw ()));
                StartAccessIfNeeded ();
                NS_LOG_DEBUG ("Background tx");
                break;
        }
        case tidAC_BK2 : {
                double now = Simulator::Now().GetSeconds();
                cout << "Background access granted at t = " << now <<
                endl;
                m_currentPacket = 0;
                m_dcf->ResetCw ();
                m_dcf->StartBackoffNow (m_rng->GetNext (0, m_dcf-
                >GetCw ()));
                StartAccessIfNeeded ();
                NS_LOG_DEBUG ("Background tx");
                break;
        }
        case tidAC_VI1 : {
                double now = Simulator::Now().GetSeconds();
                cout << "Video access granted at t = " << now << endl;
                m_currentPacket = 0;
                m_dcf->ResetCw ();
                m_dcf->StartBackoffNow (m_rng->GetNext (0, m_dcf-
                >GetCw ()));
                StartAccessIfNeeded ();
                NS_LOG_DEBUG ("Video tx");
                break;
        }
        case tidAC_VI2 : {
                double now = Simulator::Now().GetSeconds();
                cout << "Video access granted at t = " << now << endl;
                m_currentPacket = 0;
                m_dcf->ResetCw ();
                m_dcf->StartBackoffNow (m_rng->GetNext (0, m_dcf-
                >GetCw ()));
                StartAccessIfNeeded ();
                NS_LOG_DEBUG ("Video tx");
```

```
                        break;
                    }
                    case tidAC_VO1 : {
                        double now = Simulator::Now().GetSeconds();
                        cout << "Voice access granted at t = " << now << endl;
                        m_currentPacket = 0;
                        m_dcf->ResetCw ();
                        m_dcf->StartBackoffNow (m_rng->GetNext (0, m_dcf-
                        >GetCw ()));
                        StartAccessIfNeeded ();
                        NS_LOG_DEBUG ("Voice tx");
                        break;
                    }
                    case tidAC_VO2 : {
                        double now = Simulator::Now().GetSeconds();
                        cout << "Voice access granted at t = " << now << endl;
                        m_currentPacket = 0;
                        m_dcf->ResetCw ();
                        m_dcf->StartBackoffNow (m_rng->GetNext (0, m_dcf-
                        >GetCw ()));
                        StartAccessIfNeeded ();
                        NS_LOG_DEBUG ("Voice tx");
                        break;
                    }
                    case tidAC_EM : {
                        double now = Simulator::Now().GetSeconds();
                        cout << "Emergency access granted at t = " << now <<
                        endl;
                        m_currentPacket = 0;
                        StartEmergencyAccessIfNeeded ();
                        NS_LOG_DEBUG ("Emergency access tx broadcast");
                        break;
                    }
                }
            }
        }

        else if (m_currentHdr.GetType() == WIFI_MAC_CTL_BACKREQ)
        {
            SendBlockAckRequest (m_currentBar);
        }
        else
        {
            if (m_currentHdr.IsQosData () && m_currentHdr.IsQosBlockAck ())
            {
                params.DisableAck ();
            }
            else
            {
                params.EnableAck ();
```

```
}
if (NeedFragmentation () && ((m_currentHdr.IsQosData () &&
!m_currentHdr.IsQosAmsdu ()) || m_currentHdr.IsData ()) &&
(m_blockAckThreshold == 0 || m_blockAckType == BASIC_BLOCK_ACK))
{
        //With COMPRESSED_BLOCK_ACK fragmentation must be avoided.
        params.DisableRts ();
        WifiMacHeader hdr;
        Ptr<Packet> fragment = GetFragmentPacket (&hdr);
        if (IsLastFragment ()) {
                NS_LOG_DEBUG ("fragmenting last fragment size=" <<
                fragment->GetSize ());
                params.DisableNextData ();
        }
        else
        {
                NS_LOG_DEBUG ("fragmenting size=" << fragment->GetSize
                ());
                params.EnableNextData (GetNextFragmentSize ());
        }
        m_low->StartTransmission (fragment, &hdr, params,
        m_transmissionListener);
}
else
{
        WifiMacHeader peekedHdr;
        if (m_currentHdr.IsQosData () && m_queue->PeekByTidAndAddress
        (&peekedHdr, m_currentHdr.GetQosTid (), WifiMacHeader::ADDR1,
        m_currentHdr.GetAddr1 ()) && !m_currentHdr.GetAddr1 ().IsBroadcast
        () && m_aggregator != 0 && !m_currentHdr.IsRetry ())
        {
                /* here is performed aggregation */
                Ptr<Packet> currentAggregatedPacket = Create<Packet> ();
                m_aggregator->Aggregate (m_currentPacket,
                currentAggregatedPacket, MapSrcAddressForAggregation
                (peekedHdr), MapDestAddressForAggregation (peekedHdr));
                bool aggregated = false;
                bool isAmsdu = false;
                Ptr<const Packet> peekedPacket =
                m_queue>PeekByTidAndAddress
                (&peekedHdr, m_currentHdr.GetQosTid (),
                WifiMacHeader::ADDR1,
                m_currentHdr.GetAddr1 ());
                while (peekedPacket != 0)
                {
                        aggregated = m_aggregator->Aggregate (peekedPacket,
                        currentAggregatedPacket,
                        MapSrcAddressForAggregation(peekedHdr),MapDestAd
                        dressForAggregation(peekedHdr));
                        if (aggregated)
```

```
                                {
                                        isAmsdu = true;
                                        m_queue->Remove (peekedPacket);
                                }
                                else
                                {
                                        break;
                                }
                                peekedPacket = m_queue->PeekByTidAndAddress
                                (&peekedHdr, m_currentHdr.GetQosTid (),
                                WifiMacHeader::ADDR1,
                                m_currentHdr.GetAddr1 ());
                        }

                         if (isAmsdu)
                        {
                                m_currentHdr.SetQosAmsdu ();
                                m_currentHdr.SetAddr3 (m_low->GetBssid ());
                                m_currentPacket = currentAggregatedPacket;
                                currentAggregatedPacket = 0;
                                NS_LOG_DEBUG ("tx unicast A-MSDU");
                        }
                }

                if (NeedRts ())
                {
                        params.EnableRts ();
                        NS_LOG_DEBUG ("tx unicast rts");
                }
                else
                {
                        params.DisableRts ();
                        NS_LOG_DEBUG ("tx unicast");
                }
                params.DisableNextData ();
                m_low->StartTransmission (m_currentPacket, &m_currentHdr, params,
                m_transmissionListener);
                CompleteTx ();
        }
    }
}

void EdcaTxopN::Queue (Ptr<const Packet> packet, const WifiMacHeader &hdr)
{
        NS_LOG_FUNCTION (this << packet << &hdr);
        WifiMacTrailer fcs;
        uint32_t fullPacketSize = hdr.GetSerializedSize () + packet->GetSize () +
        fcs.GetSerializedSize ();
        m_stationManager->PrepareForQueue (hdr.GetAddr1 (), &hdr, packet,
        fullPacketSize);
```

```
        m_queue->Enqueue (packet, hdr);

        QosTag qos;
        uint8_t tid = 8;

        if (packet->PeekPacketTag (qos))
        {
                if (qos.GetTid () == tid)
                {
                        StartEmergencyAccessIfNeeded ();
                }
                else
                {
                        StartAccessIfNeeded ();
                }
        }
}

void EdcaTxopN::StartEmergencyAccessIfNeeded (void)
{
        NS_LOG_FUNCTION (this);
        if(m_currentPacket==0&&(!m_queue>IsEmpty()||m_baManager>HasPackets())
        && !m_dcf->IsAccessRequested ())
        {
                m_manager->RequestEmergencyAccess (m_dcf);
        }
}
```

7 References

[1] Lu Han, Wireless Ad-hoc Networks, October 8, 2004.

[2] Imrich Chlamtac, Marco Conti, Jennifer J.-N. Liu, Mobile ad hoc networking: imperatives and challenges, in: Ad Hoc Networks 1 (2003), pp. 13-64.

[3] James A. Freebersyser, Barry Leiner, A DoD perspective on mobile ad hoc networks, in: Charles E. Perkins (Ed.), Ad Hoc Networking, Addison Wesley, Reading, MA, 2001, pp. 29–51.

[4] B. Leiner, R. Ruth, A.R. Sastry, Goals and challenges of the DARPA GloMo program, IEEE Personal Communications 3 (6) (1996) 34–43.

[5] IEEE P802.11/D10, January 14, 1999.

[6] M. Conti, Body, personal, and local wireless ad hoc networks, in: M. Ilyas (Ed.), Handbook of Ad Hoc Networks, CRC Press, New York, 2003 (Chapter 1).

[7] M.S. Corson, J.P. Maker, J.H. Cernicione, Internet-based mobile ad hoc networking, IEEE Internet Computing 3 (4) (1999) 63–70.

[8] G. Anastasi, M. Conti, E. Gregori, IEEE 802.11 ad hoc networks: protocols, performance and open issues, in: S. Basagni, M. Conti, S. Giordano, I. Stojmenovic (Eds.), Ad hoc Networking, IEEE Press Wiley, New York, 2003.

[9] Piyush Gupta, Robert Gray, P.R. Kumar, An Experimental Scaling Law for Ad HocNetworks, 001. Available from <http://black.csl.uiuc.edu/~prkumar/postscript_files.html>.

[10] Sherali Zeadally, Ray Hunt, Yuh-Shyan Chen, Angela Irwin, Aamir Hassan, Vehicular ad-hoc networks (VANETS): status, results, and challenges, in: Telecommunication Systems, Volume 50, Issue 4, pp. 217-241.

[11] Gerlach, M. (2006). Full paper: assessing and improving privacy in VANETs. www.network-on-wheels.de/downloads/escar2006gerlach.pdf (accessed: May 29, 2010).

[12] Festag, A. (2009). Global standardization of network and transport protocols for ITS with 5 GHz radio technologies. In Proceedings of the ETSI TC ITS workshop, Sophia Antipolis, France, February 2009.

[13] Wang, J., & Yan, W. (2009). RBM: a role based mobility model for VANET. In Proceedings of international conference on communications and mobile computing (CMC'09) (Vol. 2, pp. 437– 443), January 2009.

[14] Liu, B., Khorashadi, B., Du, H., Ghosal, D., Chuah, C., & Zhang, M. (2009). VGSim: an integrated networking and microscopic vehicular mobility simulation platform—[Topics in automotive networking]. IEEE Communications Magazine, 47(5), 134–141.

[15] Fiore, M., Harri, J., Filali, F., & Bonnet, C. (2007). Vehicular mobility simulation for VANETs. In Proceedings of 40th annual simulation, symposium (pp. 301–309), March 2007.

[16] Dmitri Moltchanov, Alexey V. Vinel, Jakub Jakubiak, Yevgeni Kouchervavy: Synchronous Relaying in Vehicular Ad-hoc Networks, IJWNBT 1(2) (2011), pp. 36-41.

[17] Chong Han, Student Member, IEEE, Mehrdad Dianati, Member, IEEE, Rahim Tafazolli, Senior Member, IEEE, Ralf Kernchen, and Xuemin (Sherman) Shen, Fellow, IEEE: Analytical Study of the IEEE 802.11p MAC Sublayer in Vehicular Networks. IEEE Transactions on Intelligent Transportation Systems, Vol. 13, NO. 2, June 2012.

[18] Hadi Arbabi, Michele C. Weigle: Highway Mobility and Vehicular Ad-Hoc Networks in NS-3. Proceedings of the 2010 Winter Simulation Conference.