UiO : Department of Informatics
University of Oslo

Toward QoS support for video by packet prioritization in Delay Tolerant Network with ns3

Rizwan Ali Ahmed

Master's Thesis Spring 2014

Toward QoS support for video by packet prioritization in Delay Tolerant Network with ns3

Rizwan Ali Ahmed

14th May 2014

Abstract

QoS support for videostreaming over dynamic wireless network is a research field that has gained a lot popularity for the past decade.

In this thesis we analyze in which extent we are able to provide QoS support for videostreaming in a Delay Tolerant Network, where disruptions occurs frequently. QoS support in this thesis is based on prioritization of videopacket. The results shows that the implemented proposal is at some level able to provide a better QoS.

Contents

List of Figures

List of Tables

Listings

Preface

Due to technical difficulties in Latex in last minutes, citation and references not available in this version of the document.

Chapter 1

Introduction

1.1 Background

In absence of a stable network infrastructure, scenarios might occur where urgent Audio/Visual communication is absolutely necessary. Network can be entirely or partly destroyed, which precludes necessary communication.

With latest technology, possibilities exists to provide audio and visual services in ubiquitous environments by using wireless communicating devices. Having a stable end-to-end communicating path is a big issue, since it may not be possible to determine that there exists a network infrastructure or not. Therefore, a MANET (Mobile Ad-hoc Network) can be established by wireless communicating devices, which might be the only usable network available. MANET has shown promising results, but in order to cover wider areas with same amount of nodes will cause more disruptions in the wireless communication which requires alot more resilience and delay tolerance in the network. Delay and disruption tolerant networking is a new research area that aims to develop protocols for "disconnected" networks, i.e., where there is no direct path from sender to receiver at a given time, but such a path may come into existence later. Most of the protocols in Delay Tolerant Network (DTN) implements a Store Carry Forward (SCF) paradigm, which is based on storing data in buffer at designated nodes before forwarding it when it seems necessary.

The demand of AV-services in a Delayed Tolerant Network (DTN) leads us to a underlying problem of Quality of Service (QoS). Real time-data in a DTN are most likely to be delayed by seconds, minutes or even by hours due to frequent violations in communication paths caused by mobility. In a DTN, mobility will cause short time frame of acceptable link quality so exchanging packets between a sender and a receiver can be possible. With limited time frame of acceptable link quality the matter of prioritizing packets arise in order to provide and maintain a QoS-level required by the application domain.

In addition to prioritization schemes at mobile middleware on these devices, some sort of QoS support can also be provided at the MAC-layer by utilizing IEEE 802.11e which uses a priority-based Medium Access Control mechanism.

1.2 Scenario: Emergency and Rescue

Emergency and Rescue operation in deserted areas might be a possible application domain for DTNs, or other scenarios like battlefields where the situation might be that networking infrastructure is partially or entirely non-existing. In a given ER-scenario we assume that rescue personnel are usually at the scene where immediate help is required, and a Command Control Center (CCC) is placed at a distance supervising those who are on the field. During these circumstances, it is usually important to provide immediate response for all participants. This requires a large amount of communication and coordination among involving parties. Traditional two-way radio transceiver has provided audio services very well for the last half of a century. With rescue personnels in the field equipped with devices (like head mounted cameras, (wrist-)wearable computers, lightweight microphones, and screen projection in glasses e. g.), video-services can be provided to allow "spontaneous" video conferencing between rescue personnels and CCC by using Message Ferries that are travelling static route back and forth between the actual scene and CCC.

1.3 Problem statement

In a DTN there will always be packet loss, but still QoS can always be improved by minimizing the loss of important packets, even through multiple hops. Our main goal is to have an optimal QoS on video stream for end-users, by preventing loss of important packets on the MAC-layer by utilizing IEEE 802.11e in to a existing solution in addition to priority-based queue in the Dts-overlay so packets are arranged in right manner.

1.4 Contribution

Main contribution in this thesis is to add implementation to a existing implementation of DTS-Overlay which is able to prioritize packets and to analyze how utilizing IEEE 802.11e will affect live transmission in different scenarios.

1.5 Outline

In this thesis we present background information on state of the art followed by a discussion about the design and about how QoS support on MAC-layer and priority-based storage should be implemented.

We aim to achieve a concept of prioritization of video packets in a DTN in order to provide QoS. Furthermore, we also want to understand in which extent prioritization will provide a better QoS.

Chapter 2

Delay Tolerant Network

2.1 MANET

A MANET is a self-configuring and infrastructureless network of mobile devices (hereby nodes) connected wireless. In absence of infrastructure, multi-hop routing in a MANET might be the only solution. Each and every node in MANET is responsible for forwarding data to other nodes within its range, and is free to move in any direction. Mobility on each node results frequent changes on its link to other nodes in the network. In order to deal with topology dynamics caused by mobility we need MANET- routing so the spontaneous formation in the network can be supported. Primary challenge here is to continuously maintain information that is required to route traffic properly. Nodes in MANET do either communicate directly with each other or by using other nodes as routers if destination node is outside of senders range. This is because a node can work as well as an end-system beside working as a router.

2.1.1 MANET Routing

Several MANET-routing protocols have been developed to deal with issues in ad hoc network. A routing protocol should be able to present a up-to-date route which has minimal costs (e. g. distance, delay). Routing in MANET is basically performed in two types of protocols, proactive and reactive. The main difference between these two is that proactive routing protocols regularly updates information regarding traffic routing, and exchanges information with nodes nearby to keep an up to date route available when it is needed. In reactive protocols, routes are calculated on demand whenever it seems necessary.

AODV

Ad hoc On Demand Distant Vector is a reactive routing protocol. In a MANET where AODV routing protocol is followed, a broadcast of route request packet is sent when a route is needed. Each node that receives a route request rebroadcasts the packet. The node then updates its local routing table so that this node has a pointer to where the packet came from.

When the request packet reaches the destination, a route reply packet is send from the destination node on the same path. Each node on the path then forwards the packet back towards the source. When the route reply packet reaches the source, each node on the path knows which node it will have to forward it to reach the destination.

DSR

Dynamic Source Routing is also a reactive routing protocol which is similar to AODV routing protocol. A route request packet is broadcasted from the source. Each node receiving such a packet adds its own address to the packet and rebroadcast the message, but it is never done twice for the same source/destination/sequence number combination. When the route request packet reaches the destination, it is returned to the source using addresses which are added to the packet and the route is formed.

OLSR

The Optimized Link State Routing (OLSR) is a proactive protocol and is an enhancement of Link State Routing. Each node uses HELLO messages to discover its 1-hop and 2-hop neighbors. Also to perform a distributed election of a set of multipoint relays (MPR). MPRs are elected so that every 2-hop path between two nodes go through a MPR. These MPR nodes then exchange Topology Control (TC) messages, sharing information about which nodes each MPR covers. Topology information is flooded often enough that routes are not wrong for long periods of time. When a node needs a route, is simply looks in its routing table, which stores the next hop for each destination.

2.2 Delay Tolerant Network

Sparse MANETs are usually sparsely connected with low bandwidth. Disruptions may be caused by few nodes to cover a too large area. There might be variations in landscape that can be physical hindrance of signals. This causes frequent violations in communication path between nodes. At some point there will no be a direct path from one node to another at a given time, but such a path may come into existence later. In order to forward packets in a sparse MANET, a possibility is to let nodes store packets that are forwarded when the connectivity is back. Most of the protocols in Delay Tolerant Network (DTN) implements a Store Carry Forward (SCF) paradigm, which is based on storing data in buffer at designated nodes. Data are stored at these nodes till it seems necessary to forwarding to the next-hop toward destination.

2.2.1 Mobility and density

Terminology that are used in DTN scenario explained in [ ]. A Space Path is described as a multi-hop path where all the links are active at current time. A Space/Time Path is described as a multi-hop path that exists over time. Mobility and density are two factor that are necessary to acquire either a Space Path or a Space/Time path. According to diagram from [] higher mobility in a DTN results higher probability of a Space path or a Space/Time path toward the destination node. The more mobility decreases the more is is the need for higher node density in order to maintaining a Space path or a Space/time path.

2.3 DT-Stream application

Within a relatively short time frame, not all data are able to be transmitted from one node to another. Such violations and delays in transmission of video packets causes variation in the streaming will cause jitter and noise for the end-user application. This results lack of quality, and in worst case loss of valuable information for end-user. Goal for a Delay tolerant streaming (DTS) application is not to break when delay in network occurs, but instead adapt, and start proceeding when connectivity is back . By utilizing a self-adaptive overlay with ability to cache AV data at selected nodes, will increase the resilience of AV services . In order to do so we need autonomic resource management which can discover, monitor and manage resources through distributed admission control and multi-path routing protocols.

2.3.1 Delay Tolerant Stream Overlay

In order to increase the resilience and performance of the AV services in DTN, a self-adaptive overlay that caches AV data needs to be implemented at selected nodes. Delay Tolerant Stream Overlay (DTS-Overlay) is an inspiration from MOMENTUM and is able to cache data at selected nodes to forward it later. The fundamental task of Dts-Overlay is to make forwarding decisions and store packets when it is not any meaning of forward them. These forwarding decisions are based on link status from MAC-layer. Packets that are rejected by the MAC-layer when there is no path to a next-hops for the moment, are stored in cache at the Dts-Overlay. They are are stored in cache until it has been successfully transmitted to the next-hop or has reached the maximum number of retransmissions.

2.3.2 Design of Dts-Overlay

The main task of Dts-Overlay is to conclude whether it is any meaning of forward packet to next-hop or not. In case of denial from lower layers, the Dts-Overlay decides to store packets in its buffer. While the packet is stored in buffer, the routing table tries to measures new route. If no route to destination is given in the routing table produced by routing protocols, Dts-Overlay tries to find recent routes in the routing history. If a recent route suggests a next-hop, packets are sent to this next-hop based on assumption that it is closer to destination. If no next-hop is suggested by the routing table, the current transmission will be temporary suspended. If the next-hop is identified,SS but there is no positive link status from the MAC-layer, in this case transmission is also suspended.

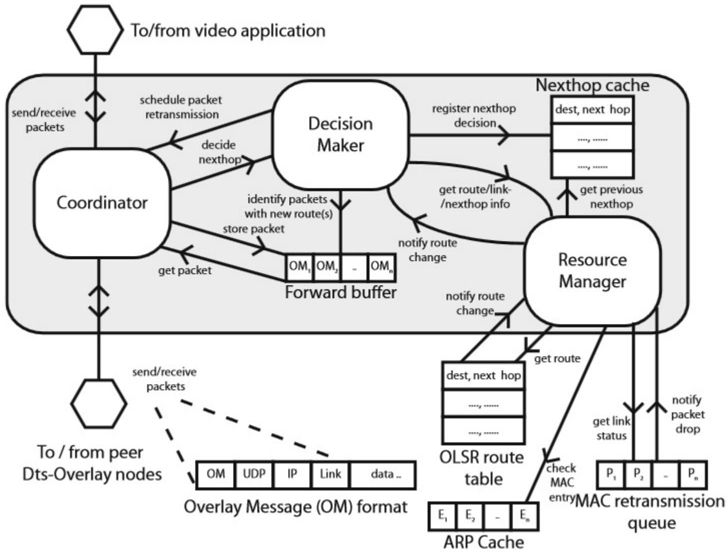

Figure 2.1: Design of Dts-Overlay

Dts-Overlay consist of several parts. Resources Manager collects and monitors state information from the MAC-retransmission queue (link status). It is also responsible for maintaining the routing table and notifies route changes in Decision maker. Main function for Decision Maker is to provide next-hop IP address for a given packet destination. It loops trough the store-forward-carry buffer and matches next-hop to according packet. Store-Carrie-Forward packet buffer is implemented as a simple Drop Tail FIFO queue. Coordinator works as the main interface for video streaming application and is also responsible for forwarding and receiving all communication with other Dts-Overlay peers. Packet exchange between peers is performed through overlay messages. Packets are fetched from the forward buffer.

2.3.3 RTP

Real-time Transport Protocol is a standardized packet format for delivery of real-time data for multimedia applications over IP-network. RTP is basically designed for end-to-end, real-time, transfer of stream data. It is expected for timely delivery of information and it allows to tolerate some packet loss. The protocol is designed to provides facilities regarding compensation of jitter that occurs due to issues in the underlying network. It also have mechanism to detection of out of order received data, which are very common during transmissions over an IP network. This protocol also allow data transfer to multiple destinations through multi casting. One of the features in the design of RTP is able to carry a different types of multimedia formats (such as H.264, MPEG-4, MPEG, etc.) and also allows new formats to be added. Therefore multimedia applications are most likely to utilize RTP for streaming purposes. Multimedia data are then framed, embedded and transmitted as RTP-packets.

RTP packet header

The RTP header has a minimum size of 12 bytes. In the first offset of the RTP headers contains various information regarding version of RTP, indication of padding byte in the packet and an indication of extension of the header. This is followed by CSRC Count, a marker used by the application layer and indication of type of payload. Field of Sequence Number occupies 16 bits of the header. The sequence number increased by one for every RTP packet that is sent for a given stream. The receiver uses this field to detect any packet that is out of range or lost and tries to recover if it is possible. The RTP itself does not specify how to handle packet loss and therefore it does not provide any guarantee of delivery. This is usually handled by the application by detecting missing packets by recognizing sequence numbers based on the information from RTP. Timestamps is also a field that is essential for real-time purpose. Timestamps are generated by the multimedia application, or the source of the real-time data. Use of timestamps is important to reduce jitter that occurs when data arrives to the receiver after a journey through IP-network. For synchronization purposes in case of multiple sources synchronization source identifier (SSRC) and Contributing source ID (CSRC), are used for identify and contribute and to contribute sources to a stream which has been generated from multiple sources. RTP-packets are embedded with and transported as UDP-packets (User Datagram Protocol), because real-time application can not require connection oriented transmission like TCP without effecting QoS of the real-time multimedia stream. When RTP-packets are captured by the receiver, frames are decoded depending on data reference in the payload and forwards it is handed to the application layer.

2.3.4 UDP

The User Datagram Protocol (UDP) is used by end-users to send messages (datagrams) to other hosts on the IP network. Transmission of datagrams is done without any communications to set up transmission channels or data paths establishment in advance. (three way handshake) Transmission model that is used by UDP is quiet simple with a minimum of mechanism that is used by the protocol. For instance, it has no handshaking dialogues prior to transmission. Therefore it is at some level exposed unreliability from the underlying network. Normally a simple IP-packet over unreliable media can neither guarantee proper delivery, order or duplication. But UDP does provide checksum to verify the integrity of the data, along with port number for addressing different functions at the source and destination of the datagram.

UDP is normally suitable for purposes where checking or correction of error may not be necessary for the application or maybe it is handled by the application. It have the advantage of avoiding the overhead on the network. And since 3-way handshake sometimes is not necessary, time-sensitive applications often prefer UDP because is might be a better solution to drop packets than packets to be delayed, which is considered in a real-time system. A host may choose another protocol if error correction facilities are needed in the network, an for example Transmission Control Protocol (TCP) which is designed specially for more reliable transmission. Unlike UDP it does feature mechanism to avoid and handle congestion.

UDP vs TCP

Unlike UDP, TCP is more connection-oriented protocol, which means that a handshake between sender and receiver (end-to-end communicators) is done prior to transmission. When the connection is set up hosts are able to send data bi-directional. TCP is considerate as a reliable protocol where messages are responded by acknowledgements. And if not acknowledgement is received by the sender, retransmission is made on time-out. Sender host may try several attempts to deliver the message, there are usually no missing data. In TCP, messages are usually received in orderly manner, where the first message sent is received first. When segments arrive at receiving host unordered, TCP can buffer the out-of-order data till it is received properly and then present to the application. Even if TCP provides reliability, orderness, and congestion control, it requires a lot more information in the header comparison to UDP, which will give us overhead in the network.

UDP in other hand is a simple which does not set up end-to-end connection before the transmission takes places. Communication is basically one-way from one host to another. When a message is sent, we cannot be sure that it reaches its destination since there are no concept of acknowledgement, retransmission, or timeout. It does not have possibility to manage messages that arrives in incorrect order. UDP itself does not have possibility to avoid congestion in the network, and is depended from the application layer to control a congestion. Overall, UDP is a lightweight protocol compared to TCP and requires less overhead in the network, which is suitable for real-time multimedia.

MTU and packetization

A larger MTU also means processing of fewer packets for the same amount of data. During transmitting a frame, an RTP packet should carry data for only one frame. In case of loss of one RTP which contains data for only one frame, will not have a great impact on the video sequence. The amount of video data in an RTP-packet should be adjusted such that the size of complete RTP-packet (including headers) should not exceed the maximum transfer unit (MTU) to avoid fragmenting packets. IP-layer will usually segment UDP packets exceeding the MTU of underlying layers and will

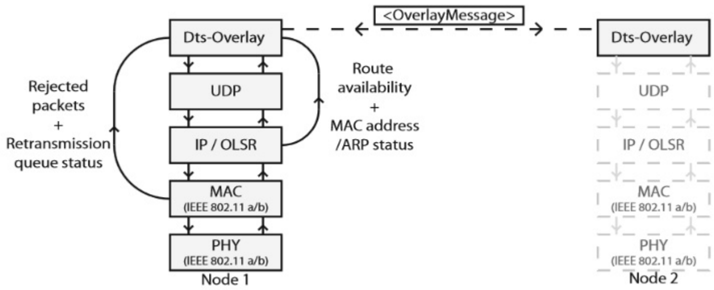

Figure 2.2: Exchange of Overlay messages between Dts-Over peers

try to reassemble them at receiving side. If one segment is missing, the whole packet would be considered lost.

2.4 Network Simulation

Network Simulator 3 (NS-3) is an open source simulation framework written i C++ and is targeted primary for research and educational use. In this framework, tools and libraries are provided to simulate different scenarios that can occur in different networks topologies with desired protocols.

2.4.1 ER-Scenario in NS-3

As explained in section **??**, rescue team personnels equipped with wireless communicating devices will be out in the field, and a CCC on a distance from the scene in occasion of an ER-scenario. Design of this scenario is already implemented in NS3 where Dts-Overlay are installed in selected nodes in order to measure throughput in the wireless networks. Nodes that are representing rescue team members have random mobility in a limited area. Multiple message ferries are designed to follow a static route path from where accident has been occurred to CCC. These message ferries are required to gather as much data as they can and bring it back to CCC.

Chapter 3

Video transmission

3.1 Video compression

A digital video is basically sequence of raw images. Normally such raw images are two-dimensional array of pixels, where each pixel is given by three values. These values are representing RGB color of pixel.

3.1.1 Video hierarchy

In order to understand video compression, we need to understand video hierarchy. Unlike images, digital video is processed and compressed in hierarchy. At the top of the hierarchy we have video sequences, which consist of scenes. These scenes are usually sets of shots. Below this level we will find that each shot is consisting of a Group of pictures (GOP), which in turn consists of multiple video frames. A single slice is divided into slice, which represents independent coding units that can be decoded without referencing to other slices in same frame. Macroblock is again a divided part of a slice, which contains 4 by 4 block, where a block typically is build of 8 x 8 pixels.

3.1.2 Video encoding and decoding

Main purpose for encoding video is to compress video so it requires minimum resources while storing or transmitting video data. Raw video data without compression demands a lot more resources, and needs to be compressed in order to not acquire too much resources. A compression basically requires two algorithms, one for encoding and one for decoding. There are two types of compression techniques for raw multimedia data:

- Lossy encoding usually results data loss during compression which means that some data is lost but still not be visible human eye.

- Lossless encoding will not result such data loss, because data is able to be decoded without any data loss.

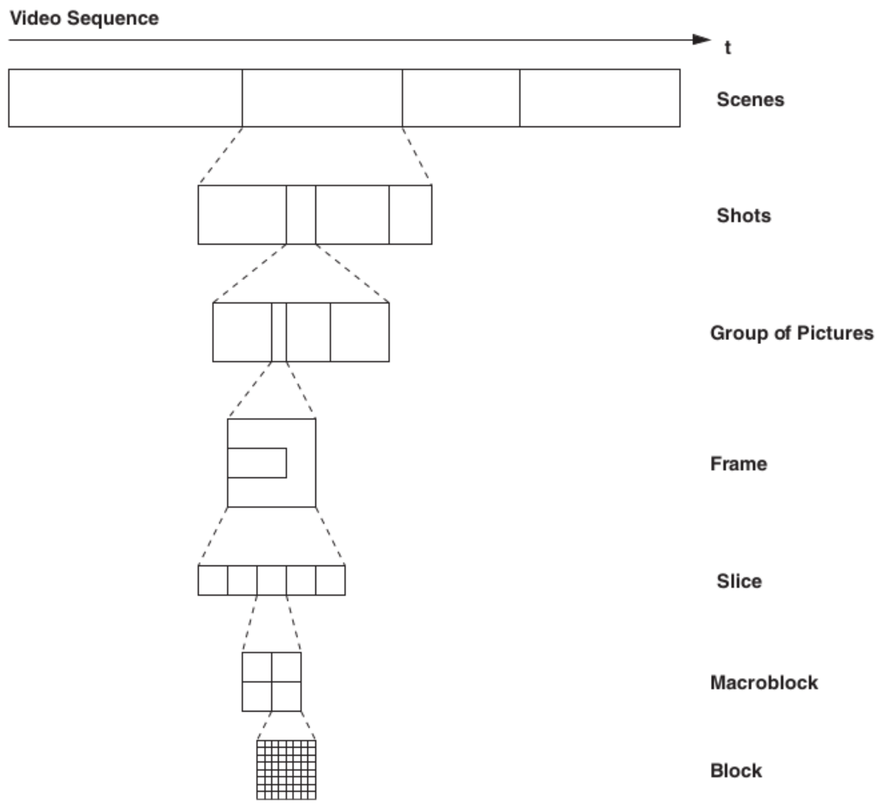

Figure 3.1: Composition of a video sequence

MPEG compression

An MPEG-4/AVC (MPEG-4 Part 10 or Advanced Video Coding) is currently one of the most commonly used compression standard. The acronym MPEG stands for Moving Picture Expert Group, which worked to generate the specifications under ISO, the International Organization for Standardization and IEC, the International Electrotechnical Commission. MPEG-4 encoding is some sort of lossy encoding resulting sequences which are basically consisting of three types of frames. These frames are I-frames, P-frames and B-frames, which would be explained.

I-frame (intra coded frame): This frame works as static picture and carries the richest reference for data (critical). Only information that is found within this frame is used for compression of current frame.

P-frames (predicted frame): Compression of this frames is based on the information from previous I-frame or P-frame. In order words P-frames can only be completely decoded if the previous I-frame and P-frame is available.

B-frames (Bi-predicted frame): Compression of this frames is based on a forward prediction from a previous I- or P-frame, as well as a backward

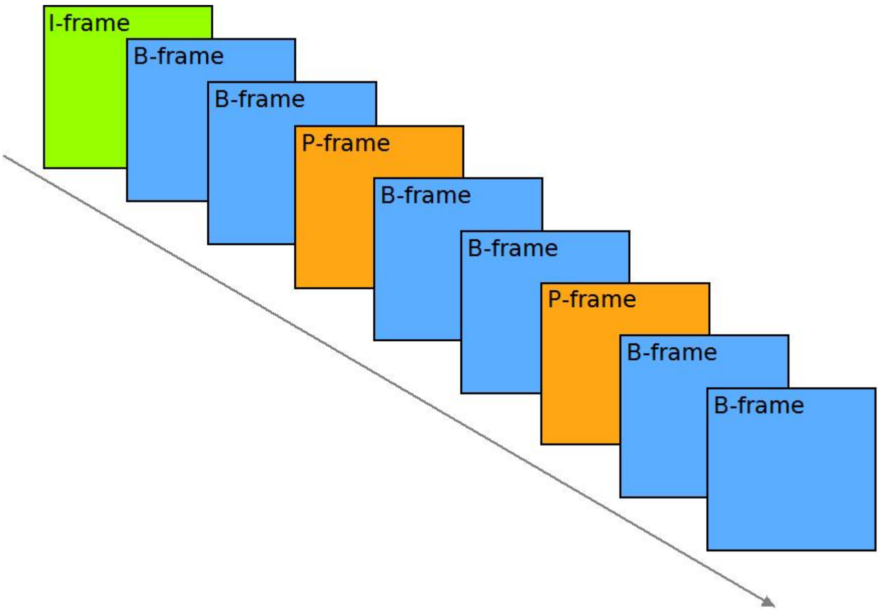

Figure 3.2: MPEG

prediction from a succeeding I- or P-frame. This means that B-frames can only be completely decoded if previous and successive I- or P-frame is available.

Frames are usually segmented into macro blocks. The way this work is by having individual prediction types can be selected on a macro block instead of having the same prediction for the entire picture, as follows:

- I-frames can contain only intra macro blocks
- P-frames can contain either intra macro blocks or predicted macro blocks
- B-frames can contain intra, predicted, or bi-predicted macro blocks

3.1.3 Decoding and display order

On suppose these frames are sent during a live transmission, these frames need to be decoded immediately on arrival at the destination. Decoding of a video frames is depended on how it was encoded in first place. Decode of a P frame is dependent on previous (I or P) frame. This means that the previous frame already needs to be decoded in order to be decode current frame. Same procedure applice for decoding of B frame since is dependent on existence of previous and successive (I or P) frames. This is the reason why MPEG reorders the frames before transmission, so receiver can decode frame "on the fly" when it arrives.

Display order	Frame type	Decode order
1	I	2
2	B	3
3	B	1
4	P	5
5	B	6
6	B	4
...		

3.2 QoS on video

QoS is a definition that is commonly used to define a granted level of quality on a audio/visual services. For instance, in the field of computer networking Quality of service is the ability to provide different priority to different data flows, and to guarantee a certain level of performance to the end-user.

For example in our scenarios, multimedia data flow in a network, requires bit rate, delay, jitter, packet dropping probability and/or bit error rate which needs to be guaranteed. These guarantees are important if the network capacity is insufficient, especially for real-time streaming multimedia applications, since these often require fixed bit rate and are delay sensitive, and in networks where the capacity is a limited resource, for example in cellular data communication.

End-to-end QoS on video is achievable but depended on various factor in network traffic.

Throughput When many users are sharing the same network resources, it results variations of load. This causes the provided bit rate to a certain data stream may be too low for real time multimedia services, if all data streams get the same scheduling priority.

Packet loss There can be many reasons for a packet drop. The routers might fail to deliver some packets if their data somehow is corrupted or they arrive where its not possible to cache data. The receiver should be able to detect packets that are dropped and ask for data retransmission if it is possible.

Errors Sometimes packets are corrupted due to bit errors caused by noise and interference, especially in wireless communications. The receiver should be able to detect any corruption in bits received, and ask data to be retransmitted.

Delay It might take some time for a packet to reach its destination. Packets may get held up in long queues, or the protocol decides to choose a another less direct route to avoid congestion. Even if the throughput is almost normal, as the delay can build up over time. Such latency can sometimes be handled by an application.

Jitter Packets from the source will reach the destination with different delays. A packet's delay varies with its position in the queues of the routers along the path between source and destination and this position can vary unpredictably. Such variation in delay is known as jitter and can seriously affect the quality of streaming multimedia.

Out-of-order delivery When a collection of related packets is routed through a network, different packets may take different routes, each resulting in a different delay. The result is that the packets arrive in a different order than they were sent. This problem requires special additional protocols responsible for rearranging out-of-order packets once they reach their destination. This is especially important for multimedia streams where quality is dramatically affected by both latency and sequence that are out of order.

3.2.1 Evaluation of Video Quality

QoS can either be measured based on measurements from the underlying network characteristics or by how data are presented for the user. Many task in research require a method that is automated to evaluate video quality. QoS on video can also be measured from the end-users perspective, there are two methods that are commonly used for this matter:

PSNR Peak signal-to-noise ratio, (PSNR) is a term for the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the integrity of its representation. PSNR is a widespread method calculation and is usually expressed in logarithmic decibel scale, since most signals have dynamic range. PSNR is mostly used to measure the quality of reconstruction of lossy compressions. The signal in our case is the original data, and the noise is the error introduced by compression. When comparing compression codecs, PSNR is an approximation to human perception of reconstruction quality. Although a higher PSNR generally indicates that the reconstruction is of higher quality, in some cases it may not. One has to be extremely careful with the range of validity of this metric; it is only conclusively valid when it is used to compare results from the same codec (or codec type) and same content.

$$PSNR = 20 \cdot log_{10}\left(\frac{MAX_I}{\sqrt{\frac{1}{mn}\sum_{i=0}^{m-1}\sum_{i=1}^{n-1}[I(i,j) - K(i,j)]}}\right)$$

MOS PSNR alone is not enough to measure QoS of the video, since its time series are not very concise. Mean opinion score (MOS) is a test that has been used to obtain the human user's view of the quality of the network. The human quality impression usually is given on a scare from 5 (best) to 1 (worst), which MOS is mostly based on. PSNR for every single frame is mapped to the MOS scale.

MOS	**Quality**	**Impairment**
5	Excellent	imperceptible
4	Good	Perceptible but not annoying
3	Fair	Slightly annoying
2	Poor	Annoying
1	Bad	Very annoying

Chapter 4

QoS on MAC-layer

4.1 IEEE 802.11 legacy

IEEE 802.11 Wireless Local Area Network (WLAN) is one of the most widely used wireless network technology to transfer data. It defines two types of architectures, BSS (Basic Service Set) and IBBB (Independent Basic Service Set). In a BSS, a number of wireless station (STA) are communicating through Access Point (AP) they are associated with. In IBSS, STAs can communicate directly with each other if they are within each others transmission range. Therefore IBSS allows STAs to form a wireless ad hoc network. IEEE 802.11 standard are implemented with two fundamental channel access mechanisms, Distributed Coordination Function (DCF) that allows distributed channel access, and Point Coordination Function (PCF) which provides centrally controlled polling.

4.1.1 IEEE 802.11 standards

There are many IEEE 802.11 standards that are used in wireless networking. The main differences between these standards are bandwidth, frequency and wave modulation during propagation. For instance IEEE 802.11a used the OFDM modulation technique on the 5 GHz band, and allows 54 Mbit/s.

IEEE 802.11b on the other hand used a DSSS modulation technique and is an amendment to the IEEE 802.11 wireless networking specification that extends throughput up to 11 Mbit/s using the same 2.4GHz band. This specification is marketed as Wi-Fi and has been implemented all over the world. A related amendment was incorporated into the IEEE 802.11-2007 standard.

4.1.2 Distributed Coordination Function

The Medium Access Control (MAC) mechanism for IEEE 802.11 is called Distributed Coordination Function (DFC), which is an access method for medium access on a channel. Basically DCF works as "listen-before-talk", where STAs accesses the medium in a distributive manner. When a station is ready to transmit packets on the wireless medium, it senses the channel to determine that it is idle. The protocol used by DCF is called

Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA). The collision avoidance portion of CSMA/CA is performed with a random back off procedure. If the station senses that the channel is busy, it waits for a random time frame initiated by a back-off-algorithm.

Collision avoidance

The reason why IEEE 802.11 uses Collision Avoidance instead of Collision Detection (CSMA/CD) which is used wired network, is the lack of capability of detecting collision in wireless network. Unlike wired network, where STAs are able to receive and transceive simultaneously, STAs in wireless network are not able to detect other signals when transceiving/receiving. This is because of characteristic of wireless communication, which requires signal strength to be stable above an acceptable level, and experience a greater degree of interference compared to wired network.

The backoff-timer performed by CSMA/CA is is a uniformly distributed random number between 0 and the maximum number of slots in the contention window (CW). At very first transmission attempt, CW is set to be minimum. Once the station realizes that the channel has been idle for a duration of DCF inter frame space (DIFS), it starts the timer. If the timer countdown to 0 has been succeeded without any interruptions, the station starts transmitting. If the medium becomes busy in the middle of the countdown, the station pauses the timer and continues after a short period defined by the station that requested the medium first.

For each unsuccessful transmission (not acknowledged), size of CW is expanded exponentially (2x(CW+1)-1), unitil CWmax is reached, which is the maximum size of the Contention Window. DCF also specifies a maximum number of retransmissions for a single frame. When the number of unsuccessful retransmissions exceeds the limit of maximum retransmissions, the frame is dropped. When transmission of frame is successful, CW is set til CWmin.

4.1.3 Point Coordination Function

IEEE 802.11 also defines a an optional access mechanism Point Coordination Function (PCF), which is connection-oriented and provides contention-free frame transfer. It is required for PCF to co-exist with the DCF and logically it sits on top of the DCF. This function allows different access rules based on polling by point coordinator (PC) operation at an access point (AP). Due to the priority of PCF over DCF, stations that only use DCF might not gain access to the medium. To prevent this, a repetition interval called super frame which includes a Contention Period (CP) where only DCF is allowed to gain access to the medium and a Contention Free Period (CFP) where PCF uses itself. During the CFP, the AP maintains control of the medium, by polling stations that have requested a to be in a polling list. There is no backoff or contention during this period and frames are exchanged only when stations are polled.

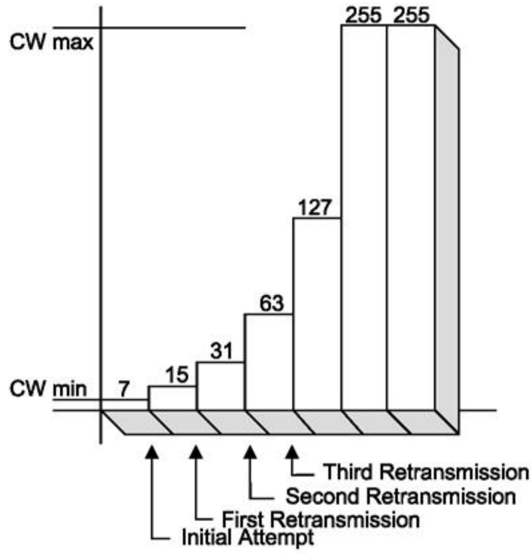

Figure 4.1: Exponential increase

Although the PCF appears to have the potential to deliver QoS, it still have some limitations. For example during a CP, where current superframe is not finished and is temporary blocked by DCF. This may delay the next superframe as well. And since implementation of PCF is optional, it is not widely implemented.

4.1.4 Inter Frame Spaces in IEEE 802.11

In order to avoid multiple stations transmitting simultaneously and pollute the medium with "collisions", there must be a controlled access. Access to the wireless medium is controlled by use of inter frame space (IFS), which is the time interval between transmission of frames. A station waits and senses for a IFS on the channel before it starts transmitting data if the channel is idle. The collision avoidance portion of CSMA/CA is performed with a random back off procedure. When a STA is ready to transmit frames on the wireless medium, it senses the channel to assure that it is idle. If the channel is busy, then the sending node waits until channel becomes idle.

There are basically three IFS defined in IEEE 802.11: Short Inter Frame Space (SIFS), PCF Inter Frame Space (PIFS) and DCF Inter Frame Space (DIFS). SIFS has the shortest time interval and therefore has the highest priority to the channel. It is mostly used to to transmit acknowledgements on the channel. PIFS and DIFS are in other hand, meant to transmit regular MAC service data unit (MSDU) on the channel. As mentioned above PIFS is acquired by PCF and has a shorter time interval than DIFS, which is acquired by DCF.

DCF starts its backoff procedure after the channel is idle for a period of DIFS:

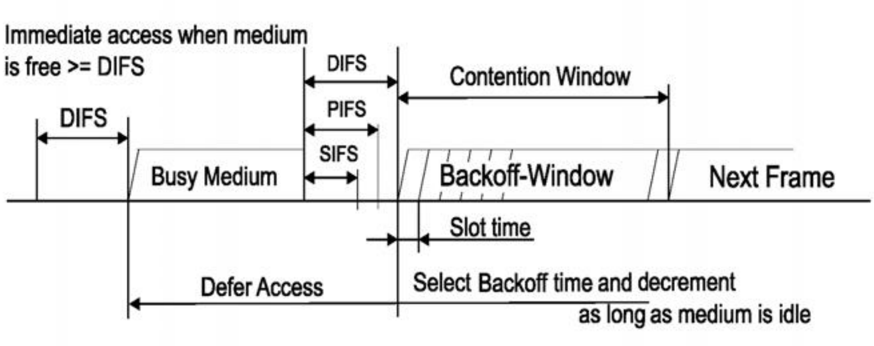

Figure 4.2: Inter Frame Spaces in IEEE 802.11

$$DIFS = 2 \times aSlotTime + aSIFSTime$$

DIFS should be equal to double time slots plus time period of a SIFS.

4.2 QoS with IEEE 802.11e

IEEE 802.11e is a standard on its final stage of development by IEEE 802.11 working group, and is developed to provide QoS features and multimedia support to the existing IEEE 802.11 wireless standards on the MAC layer. This standard has the ability to prioritize different source access to the physical layer and to support requirements demanded by the application layer.

In IEEE 802.11e, APs and STAs that provides QoS are referred to as QSTA (QoS Station) and QAP (QoS Station). BSS and IBSS that is operating QSTAs and QAPs, is known as QBSS and QIBSS.

4.2.1 Hybrid Coordination Function

In IEEE 802.11e, DCF and PCF are replaced by Hybrid Coordination (HCF), a new coordination function which is mandatory on in all QoS stations. Within HCF there are two access mechanism, the enhanced distributed channel access (EDCA) and HCF controlled channel access. Unlike PCF, HCF defines a uniform set of exchange sequences that are usable at anytime.

Enhanced Distributed Access Control

Medium Access Control mechanism that is used in IEEE 802.11e is called EDCA (Enhanced Distributed Access Control). In EDCA, the QoS support is realized by introducing multiple access categories (AC) on each station.

Access Categories

IEEE 802.11e defines four Access Categories, and they all have different priorities and they all are defined for different type of traffic.

Each AC works an enhanced version of DCF which achieves a transmission opportunity (TXOP) using specified channel access parameters. With EDCA, high-priority traffic has a higher chance of being sent than low-priority traffic. A station utilized with IEEE 802.11e waits a little longer before it sends packets that don't require a higher priority. If the back-off counters of two or more ACs on same station elapse at the same time, a scheduler inside the station treats the event as a virtual collision. The TXOP is given to the AC with the highest priority among the "colliding" ACs. Other ACs try tries again later as if the collision occurred in the real medium.

Frames from different type of data traffic are mapped into different ACs, depending on requirement from the application-layer. The four ACs are names AC_BK, AC_BE, AC_VI and AC_VO.

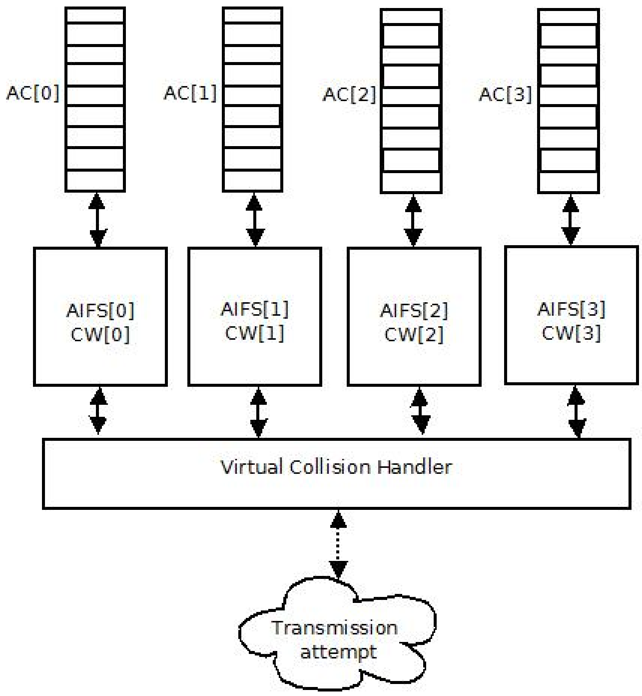

Figure 4.3: Access Categories in IEEE 802.11e

Enhanced Distributed Channel Access Function

EDAC maintains four independent EDCAFs, one for each AC. EDCAF is an enhancement of DCF, which gains access to the medium similarly with CSMACA. Basically the main difference between DCF and EDCAF is that EDCAF uses specific CWmin[AC], CWmax[AC] and AIFSN[AC] for each AC.

Access Category	Name of AC	Designation	User Priority	Priority
AC[0]	AC_BK	Background	1	Lowest
AC[0]	AC_BK	Background	2	×
AC[1]	AC_BE	Best effort	0	×
AC[1]	AC_VI	Video	3	×
AC[2]	AC_VI	Video	4	×
AC[2]	AC_VI	Video	5	×
AC[3]	AC_VO	Voice	6	×
AC[3]	AC_VO	Voice	7	Highest

Table 4.1: Access categories (AC), their designations and priorities

4.2.2 Arbitrary Inter Frame Spaces

High-priority traffic gets a shorter arbitration inter-frame space (AIFS) and a shorter contention window (CW). In addition, EDCA provides contention-free access to the channel for a period called a Transmit Opportunity (TXOP). A TXOP is a bounded time interval during which a station can send as many frames as possible (as long as the duration of the transmissions does not extend beyond the maximum duration of the TXOP). If a frame is too large to be transmitted in a single TXOP, it is usually fragmented into smaller frames. The use of TXOPs reduces the problem of low rate stations gaining an inordinate amount of channel time in the legacy 802.11 DCF MAC.

The levels of priority in EDCA are called access categories (ACs). The CWmin and CWmax values are calculated from aCWmin and aCWmax values, respectively, that are defined for each physical layer supported by 802.11e. In table all Access Categories are shown with their designation and priority. An AC starts the backoff procedure after the channel is idle for a period of AIFS[AC].

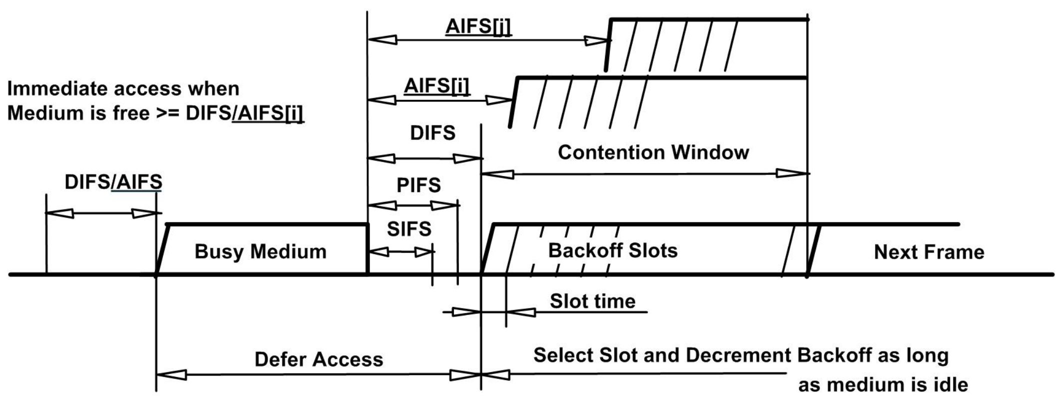

Figure 4.4: Arbitrary Inter Frame Spaces in IEEE 802.11e

AIFS[AC] is equal to multiple time slots (depending on AIFSN[AC]) plus a period of SIFS. AIFSN[AC] are usually not set to less than 2 which means that the shortest waiting for a AIFS is equal to DIFS.

Access Category	CWmin	CWmax	AIFSN	TXOP limit
AC_BK	CWmin	CWmax	7	
AC_BE	CWmin	CWmax	3	
AC_VI	(CWmin+1)/2-1	CWmax	2	
AC_VO	(CWmin+1)/4-1	(CWmin+1)/2-1	2	

4.2.3 Transmission Opportunity

Transmission Opportunity (TXOP) is time period where EDCAF transmit after gaining accesss to medium.

4.2.4 Virtual collisions

The four EDCAF behave like virtual station inside the real station where each EDCAF access the medium independently of other EDCAFs. This result two types of contentions which may occur, internal contention among different EDCAF inside the same station and external contention among different stations. When two EDCAF from same station acquires access at the same time, a situation will occur which will lead to virtual collision. In case of a virtual collision, access will be granted to AC with highest priority among colliding EDCAF.

Chapter 5

Design and implementation

In this chapter we introduce our scheme of prioritization, on the DTS-Overlay and by utilizing IEEE 802.11e. Description of metrics and workload that are used to evaluate is presented afterwards, which is followed by description of systematics and how procedure toward the evaluation is done.

5.1 Assumptions

Before we go further into discussing details in our scheme of prioritization, we have some assumptions.

- Initially nodes do not have pre-knowledge of their whereabouts
- Nodes in the scenario are together able to form a MANET by wireless connection to neighbour nodes within their range.
- Sender and receiver are "connected" to each other through a Delay Tolerant Network consisting of intermediate node with the possibility of cache packets.
- DTN should provide resilience in the network. This means that there might not be any path (directly or indirectly) from a sender to receiver at a given time, but a path will hopefully come to existence later.
- A disruptive network cannot be considerate to provide resilience unless there is an overlay which handles challenges regarding adaptiveness and heterogeneous.
- Senders streams a MPEG4 formatted videostream.
- Unorderness provided by DTN is handled by the receiver by utilizing a playout buffer.
- DT-video stream through a DTN should provide best possible QoS given underlying challenges in the network.

- Data that are sent trough the network are encoded video frames, fragmented and packeted in to UDP datagram.
- Priority is flagged in the header of OverlayMessage.

5.2 Prioritization

As explained in **??**, the encoding-process of a video causes unequal data distribution to different frames. It is therefore likely that during transmission some frames contains more valuable data than others. Frequent disruptions are very common in DTN due to mobility, which causes packet loss. In order to prevent loss of most valuable data during transmission and to provide end-users with as much relevant data as possible, it is necessary to consider prioritization of "important packets". Specially in case of an emergency scenario explained in 1.2, it is desired that packets with most important data are prioritized during transmission and successfully delivered to destination.

In this thesis we present two approaches for prioritization. One way of achieving prioritization in a DTN is by implementing our scheme in the DTS-Overlay. And the second way is by utilizing QoS support at MAC-layer to prioritize frames.

5.2.1 Characteristics of importance packets

In order to implement the concept of prioritization of packets in our solution we first define characterization of importance.

Frame type	**Importance**
I-frame	1
P-frame	2
B-frame	3
Meta data	4

Table 5.1: Frame designation

Packet types and their importance are shown in table 5.1. Since I-frames neither are dependent on successive or preceding frames to be decoded and contains data that represents whole static picture. Therefore it is natural to assign highest priority to I-frames. P-frames can only be completely decoded if the previous I-frame and P-frame are available, but still P-frames contains valuable information regarding motions in the picture. It should therefore be considered as packet of higher importance. Therefore it is designated priority 2 in our scheme. Compression of B-frames is based on a forward prediction from a previous I- or P-frame, as well as a backward prediction from a succeeding I- or P-frame. This means that B-frames can only be completely decoded if previous and successive I-or P-frame is available. Therefore B-frames are designated priority 3. In addition to these video frame type we also have a packet called meta packet (M-frane) which is a dummy packet sent every 0,6 second. Main reason for usage of a fourth packet type is to utilize all access channels that are provided in QoS MAC.

5.2.2 Prioritization on Overlay

As mentioned in section 2.3.2, forwarding buffer in Dts-Overlay is implemented as FIFO droptail-queue, where every frame type is equally treated. As an MPEG4 videostream usually contains few I-frames compared to B-frames, it is desired minimal loss of packets.

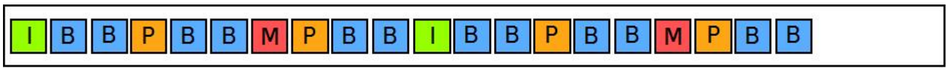

Figure 5.1: Buffer with FIFO cache

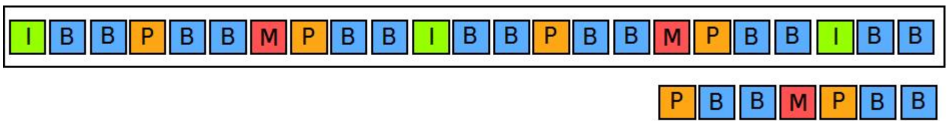

Figure 5.2: Overloaded FIFO cache

In order to utilize prioritization of packets in the DTS-Overlay, we must implement priority rules for the forward buffer. An approach for prioritization in Dts-Overlay is by replacing droptail-queue with a priority-queue, where packets are ordered according to their priority as showed in table 5.1, i.e. the highest priority packet first and the lowest priority last.

Figure 5.3: Buffer with QoS cache

Priority queue

As shown in the illustration I-frames will always appear first in the priority queue followed by all P-frames, B-frames and M-frames respectively.

In case of full storage in the forward buffer, there are a some strategies which needs to be enlightened.

1. All new arriving packets are dropped until there is enough space in the queue.

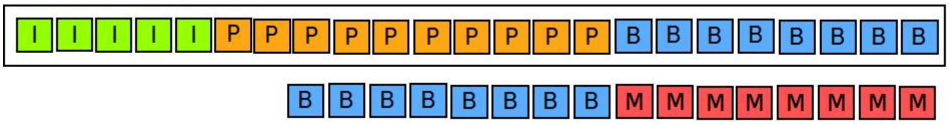

Figure 5.4: Overloaded QoS cache

2. If the new incoming packet have a lower priority than packets in the queue, it will be rejected.
3. If the incoming packet have have a higher priority than packets in the queue, packet with the lowest priority is dropped.
4. If the incoming packet have the same priority as packets in the queue,

In a disruptive MANET it is likely that packets arrives to the destination unordered. Orderness of packets that arrives are not relevant in our case, since we are already assuming this will be handled by receiver. By providing sorting of packet based on timestamps besides prioritization, some of QoS challenges in real-time streaming in a Delay-Tolerant-Network might be solved for end-users. As we are aiming for priority in a DTN, prioritization within same packet time is not not adopted since timestamps of packets are not considered in our prioritization.

Implementation of QoS overlay

Unlike elements in standard droptail-queues, elements in priority-queues need to be sorted based on their priority all the time. Due to lack of support for build-in sorting for queues in ns3-framework, the QoS Overlay is not implemented as a standard queue, but as deque. Deque is similar to standard queues but is double ended, which also supports buildt-in sorting on elements based on defined criteria and rules. A downside for using deque instead of droptail-queue is that it will not inherit all properties from the super class.

As shown in **??** prioritization function which return a boolean based on comparison of two priorities.

Listing 5.1: Criteria for sorting

```
template<typename T>
bool prioritization (const Ptr<T> p, const Ptr<T> q)
{
        OverlayMessage om;
        SeqTsHeader seqTs;

        p->PeekHeader(om);
        p->PeekHeader(seqTs);
        uint16_t pri_a = om.GetPriority();
        //uint64_t time_a = seqTs.GetTs().GetNanoSeconds();

        q->PeekHeader(om);
```

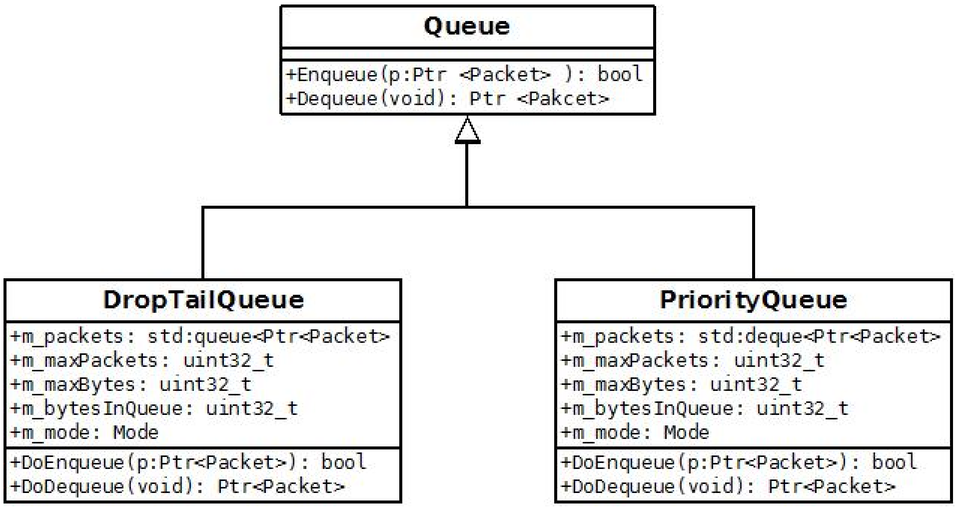

Figure 5.5: Class hierarchy of Queue in ns-3

```
        q->PeekHeader(seqTs);
        uint16_t pri_b = om.GetPriority();
        //uint64_t time_b = seqTs.GetTs().GetNanoSeconds();

        return /*((time_a<time_b) && (pri_a == pri_b)) ||*/ (pri_a < pri_b);
}
```

Challenges in simulations There are some challenges regarding maintaining the structure of priority queue. In order to do so, it is desired that elements sorted continuously. There are two ways to do so:

- Additional to a queue the new incoming packet could be compared to other elements in the queue, so every element is placed on its rightful place.
- Deletion from a queue where every element is randomly placed and seek the first element to push out.

Even if structure of cache should not have any effect on out coming result of our experiments, both of these strategies proved to be costly in our simulations due to processing time on the computer.

Instead of sorting the queue for each addition or deletion, we choose to sort the queue within constant interval. This method will constraint our experiments by not providing optimal prioritization, but this way we will also prevent memory leaks in longer run.

Listing 5.2: Insertion of element in QoS Overlay

```
bool
  PriorityQueue::DoEnqueue (Ptr<Packet> p)
  {
        if (m_mode == BYTES
```

```
            && (m_bytesInQueue + p->GetSize () >= m_maxBytes))
    {
        Drop (p);
        return false;
    }
    if (m_mode == PACKETS && (m_packets.size () >= m_maxPackets))
    {
        Drop (p);
        return false;
    }
    m_packets.push_back(p);
    sort(m_packets.begin(),m_packets.end(),ns3::prioritization<ns3::Packet>);
    return true;
}
```

Listing 5.3: Fetching of element in QoS Overlay

```
Ptr<Packet>
PriorityQueue::DoDequeue (void)
{
  NS_LOG_FUNCTION (this);

  if (m_packets.empty ())
    {
      NS_LOG_LOGIC ("Queue empty");
      return 0;
    }

  //Ptr<Packet> p = m_packets.front();
  //RemovePacketProcedure(p);
  //m_packets.erase(m_packets.begin());
  //m_packets.pop ();
  //m_bytesInQueue -= p->GetSize ();
  Ptr<Packet> p = m_packets.front();
  m_packets.pop_front();
  NS_LOG_LOGIC ("Popped " << p);

  NS_LOG_LOGIC ("Number packets " << m_packets.size ());
  NS_LOG_LOGIC ("Number bytes " << m_bytesInQueue);

  return p;
}
```

Listing 5.4: Peeking in to the queue

```
Ptr<const Packet>
PriorityQueue::DoPeek (void) const
{
  NS_LOG_FUNCTION (this);
  if (m_packets.empty())
    {
      NS_LOG_LOGIC ("Queue empty");
      return 0;
    }
```

```
    Ptr<Packet> p = m_packets.front ();
    NS_LOG_LOGIC ("Number packets " << m_packets.size ());
    NS_LOG_LOGIC ("Number bytes " << m_bytesInQueue);

    return p;
  }
```

Overlay message

Prioritization of a packet is initially taking place on the source node, by adding a priority-tag in the overlay-message. Overlay messages are messages that are exchanged between Dts-Overlay peers. These messages contain necessary information that are required to perform a transmission. The header of overlay message is implemented as followed:

Listing 5.5: Overlay message header

```
  Ipv4Address m_destinationAddress;   // Destination address
  Ipv4Address m_nextHopAddress;    // Next hop address
  uint16_t m_priority;        // Priority field added
  uint16_t m_sequence;        // Sequence number field added
  uint16_t m_hops;         // Number of hops field added
```

Listing 5.6: Priority of packet is based on frame type

```
void
DtsTraceClient::SendPacket (uint32_t size, char frametype)
{
  Ptr<Packet> p;
  uint32_t packetSize;
  uint32_t qos_flag;

  if (size>12)
  {
     packetSize = size - 14;
  }
  else
  {
    packetSize = 0;
  }

  p = Create<Packet> (packetSize);
  SeqTsHeader seqTs;
  seqTs.SetSeq (m_sent);

  switch (frametype)
  {
  case 'I': qos_flag = 0; seqTs.SetPri(0); break;
  case 'P': qos_flag = 1; seqTs.SetPri(1); break;
  case 'B': qos_flag = 2; seqTs.SetPri(2); break;
  case 'M': qos_flag = 3; seqTs.SetPri(3); break;
  }
  p->AddHeader (seqTs);
```

```
  GetObject<DtsOverlay> ()->SendTo (p, qos_flag, m_peerAddress)
}
```

Listing 5.7: Meta-packet

```
void
DtsTraceClient::SendMeta (void)
{
  NS_LOG_FUNCTION_NOARGS ();

  //NS_ASSERT (m_sendEvent.IsExpired ());

  //SendPacket (m_maxPacketSize,'M');
  Simulator::Schedule (MilliSeconds (600), &DtsTraceClient::SendMeta, this);
}
```

5.2.3 Prioritization on MAC-layer

IEEE 802.11e can also be leveraged to achieve QoS support at MAC-layer. In this thesis we suggest the mapping to ACs as similarly to the prioritization scheme.

Mapping to Access Category on MAC-layer

Since I-frames are assigned highest priority in our prioritization scheme, it should also acquire highest AC (AC0). P-frames and B-frames should acquire transmission opportunity (TXOP), with AC1 and AC2 accordingly. For the Meta data we have assigned the lowest priority so there is mapped to AC3. The Overlay decides which Access Category packet belongs to so it can achieve a transmission opportunity (TXOP).

Access Channel	Frame type	Importance
AC_VO	I-frame	1
AC_VI	P-frame	2
AC_BE	B-frame	3
AC_BK	Meta data	4

Table 5.2: Frames are designated to different Access Channel

Listing 5.8: Priority of packet is set by the Dts-Overlay

```
\label{overlay-init}
DtsOverlay::SendTo (Ptr<Packet> p, uint32_t flags, Ipv4Address addr)
{

  OverlayMessage om;
  om.SetDestinationAddress (addr);
```

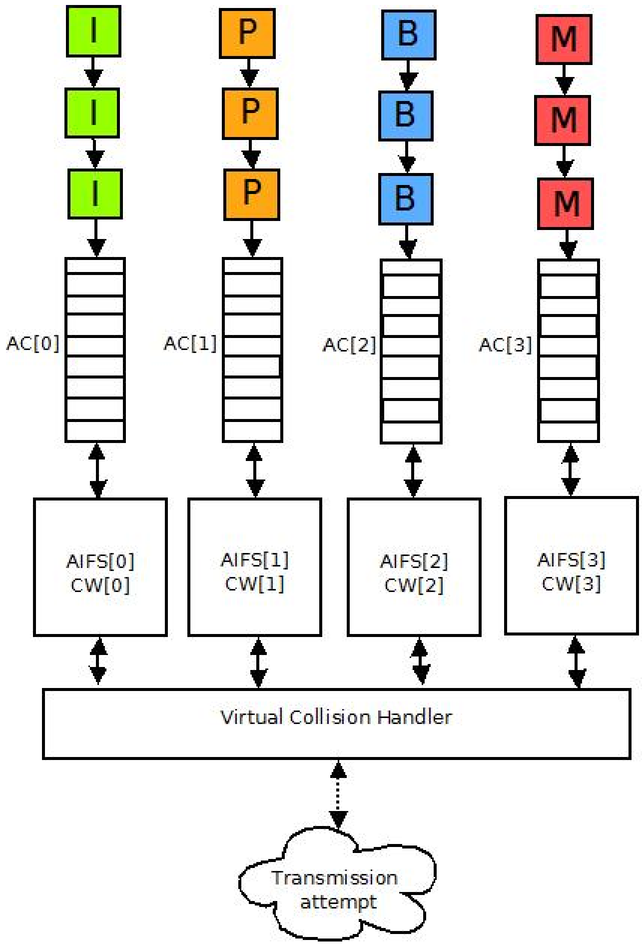

Figure 5.6: Caption fix

```
switch(flags)
{
case 0: om.SetPriority(1); break;
case 1: om.SetPriority(2); break;
case 2: om.SetPriority(3); break;
case 3: om.SetPriority(4); break;
}

QosTag qostag;
switch(flags)
{
        case 1: qostag.SetUserPriority(UP_VO); break;
        case 2: qostag.SetUserPriority(UP_VI); break;
        case 3: qostag.SetUserPriority(UP_BE); break;
        case 4: qostag.SetUserPriority(UP_BK); break;
}

p->AddPacketTag(qostag);
p->AddHeader(om);
p->PeekPacketTag(qostag);
```

```
  HandlePacket (p);
  return 0;
}
```

5.2.4 Performance metrics

To evaluate performance of a simulation, it is necessary to define metrics that are used. Since there are two types of priority methods (prioritization in DTS-Overlay and QoS on MAC-layer) that we initially have intended to evaluate, we also need to define metrics for each of them.

As explained in 3.1, video QoS is usually measured with PSNR and MOS. But there are many challenges is regarding evaluation of a video stream through a Delay Tolerant Network. Since we have not managed to implement an actual

For simplicity, metrics we use in our evaluation are:

- Data delivery
- End-to-end transmission delay
- Data delivery over time
- End-to-end transmission delay
- Effective throughput

Data delivery over time (packets/bytes)

Data delivery itself tells us about the total amount of successfully delivered data. Data delivery over time also tells us about variations in data delivery. Counting data that is successfully delivered to the receiver over time, is able to tell us how performance is affected. Received data can be counted by packet count and by packet size.

End-to-end transmission delay

End-to-end transmission delay is a factor that is very essential in this thesis. Even if the network itself is meant to be delay tolerant, we aim for smaller delay for packets with higher priority. It may sound contradictory since delay is already accepted in such network, but in a certain cases, ER-scenario for instance, it might be necessary to provide important information as soon as possible. Transmission delay could be measured for each packet received or an average for each packet type. End-to-end delay is measured:

$$E2ED = ArrivalTime - DepartureTime \tag{5.1}$$

What we expect in with QoS on MAC-layer that transmission through channel AC0, AC1, AC2 will provide a significantly lower delay than MAC-layer without QoS.

Throughput

Throughput is the average rate of successfully packet delivery over a communication channel. This data may be delivered over a physical or logical link, or pass through a certain network node. The throughput is usually measured in bits per second (bit/s or bps). One could measure throughput by

$$Throughput = \frac{\sum ReceivedData}{Time} \tag{5.2}$$

This requires all experiments to run in exact time interval in order to see differences for each experiment. Since this calculation of throughput only let us know average successfully delivered data over time period, but does not tell about successfully delivery rate for a given time in that period. An alternative way would be to measure throughput for smaller intervals.

Jitter

Jitter is a significant, and usually undesired, factor in the design of almost all communications links and is defined by variations in end-to-end delay between each packets. There are suggested calculations to measure jitter. One could measure jitter by combined with end-to-end delay. Be calculating the difference between maximum and minimum end-to-end delay.

5.3 Workload

The workload used in our simulation consist of video trace files provided by TKN research group. Source node continuously generates a data stream that is meant to be a delay tolerant video stream. This stream partly consists of video stream of I-frame, P-frames, B-frames. The video is 14 minutes long and it is repeated continuously throughout duration time. Format of the video stream is CIF where resolution is 352x288 with a 30 fps frame rate. GOP format of the video stream is (M=9 N=2, I-frame once every ninth frame and two B-frames between every I-frame/P-frame) IBBPBBPBB.... The reason why this GOP is chosen, is to send enough I-frames. In addition to these video stream we also have a packet called meta packet which is a dummy packet sent every 0,6 second. Main reason for usage of a fourth packet type is to utilize all access channels that are provided in QoS MAC.

Listing 5.9: Tracefile generated from Highway video without Meta packet

```
H 12086
P 13252
B 5389
B 5564
P 12205
B 5102
B 5760
P 12803
B 6081
```

10	B	4008
11	H	20409
12	B	5520
13	B	5353
14	P	4033
15	B	1680
16	B	892
17	P	1719
18	B	440
19	B	431

Level of workload does have an affection on the performance. We can expect to see larger workload causing more congestion not only for the source node but in entire DTN, but this is dependent on how often packets are sent. As packets are generated at a constant rate, they are remained in the cache until Decision maker component in DTS-Overlay finds a path toward destination and forwards the packet to MAC-layer for further handling. In case buffer is full, further storage of packets is not possible and they are therefore dropped.

Buffersize

Cache in DTS-Overlay helps us avoiding congestion in the underlying network in the first place, by holding on to a packet for as long it is necessary. The larger cache a node has, the more data is the node able to pass on to the next node. In the longer run is anticipated that more data is received by the destination node. In case of full buffer, no

Buffer Empty Rate

The Decision maker component in DTS-Overlay continuously pushes down contents of cache to the MAC-layer at a constant rate (Packets per second). We have defined this rate as buffer empty rate.

MAC-layer retransmission

MAC-layer tries to retransmit a packet number of times before it decides to drop the packet. The number of MAC-layer retransmissions could have an impact on the forwarding a packet. An advantage of decreasing number of retransmission attempts would relieve DTS-Overlay from storing some packets unnecessarily longer than it should and be able to make room for new packets a lot faster than having a higher retransmission attempts. But this will also increase probability of losing a valuable packet in the network.

Bandwidth

In computer networks, bandwidth is often used as a synonym for data transfer rate - the amount of data that can be carried from one point to another in a given time. It is usually measured in bps (bits per second). For QoS MAC bandwidth is distributed between ACs which tries gain TXOP.

5.4 Systematics for evaluation

Systematics for the evaluation process is divided in two parts. In the first part we will be aiming for evaluate efficiently by adding prioritization of video packets in the Dts-Overlay.

5.4.1 Semantics for Dts-Overlay

For the DTS-Overlay we are aiming for a comparison of simulations where the buffer on nodes are implemented as priority queue and Drop-Tail-Queue. For MAC-layer, we are aiming for a comparison between MAC with QOS and MAC with NQOS.

We want to observe in which extent we can gain efficiency by utilizing QOS on MAC and on Overlay in different scenarios. For each scenario we will run simulations with different velocity and density. For selected simulations we will rerun with different workload and bandwidth. Metrics that are used are delivery over time and delay for each packet.

5.4.2 Scenario

Performances can vary from one scenario to another. There are many reasons for disruptions in a wireless network. Nodes may be out of each others range or there might be hindrance in the terrain between two connecting nodes. There are basically two factors such as mobility and density which can cause differences in out coming result.

ER-Scenario

In this scenario, we have a source node within a MANET defined in a quadratic area 500m x 500m (incident area). Unfortunately destination node (CCC) is placed 2000m away from the incident area, which is beyond wireless range. In order to maintain a Space/time-path between the incident area and CCC, we have a number of ferry nodes, called carriers, which follow a static path between incident area and CCC.

5.4.3 Semantics for evaluation of QoS MAC-layer

Buffer Empty Rate

DTS-Overlay continuously pushes down contents of cache to the MAC-layer at a constant rate (Packets per second). By increasing this number we can push down more packets to MAC-queue, we may cause contention on the MAC-layer.

MAC-layer retransmission

MAC-layer tries to retransmit frames seven times before it decides to drop the packet. to we might be able to relieve some of the cache. Increasing the rate of Packet per seconds can be counterproductive since the purpose of the DTS-Overlay is to relieve the MAC-layer with that it can not handle.

Bandwidth

Bandwidth that are allowed use for IEEE 802.11 direct-sequence spread spectrum (DSSS) in ns3 are following:

- 11 Mbps
- 5.5 Mbps
- 2 Mbps
- 1 Mbps

Lower bandwidth than 1 Mbps for our implementation does not seems to be possible in ns3. What we expect to see that for higher bandwidth, we may not be able to see efficiency for QoS on MAC compared to NQoS MAC, but it is dependent on the workload. Unlike MAC-layer without QoS extension which utilized one single access channel, bandwidth is distributed between all access channels in QoS MAC-layer.

Chapter 6

Evaluation

In this chapter evaluation of our proimplementation proposed in chaper. Outcome of this chapter is the simulation setup used in our experiments, followed by configuration that we have decided to use in simulations.

6.1 Introduction

What we want to evaluate is in which extent is it possible to provide QoS by replacing a FIFO-cache with a priority-based cache. Parameters that have been measured in our simulations is basically packet delivery and delay of delivered packets.

6.1.1 Simulation setup

Simulation of our scenario is done in NS3 which provides a powerful and complex framework to create simulations and scenarios of our desire. The simulation framework is written is C++. Simulations in ns3 normally consists of following steps.

1. Create nodes that will participating in the simulation.
2. Make connections between nodes, either they are suppose to be wires or connected wireless.
3. Set the network parameters.
4. Install network protocols, applications, mobility and initiating positions on nodes.
5. Start the simulation.

Since our implementation of scenario does not support multiple source and destination nodes, we have only one source node and one destination node in our experiments. These nodes have a static position in our scenarios. We also have a variable number of intermediate nodes which will be mobile within a defined area. Number of intermediate node, their positions and mobility are configured for each experiment.

ER-Scenario and topology

Our simulations are based on a hypothetical Emergency & Rescue Scenario, where an incident has occurred in an area. A Command Control Center (CCC) is placed on a distance. Data from the incident area is delivered to CCC through a number of nodes in the incident area and carriers. Carriers follows a static route between the incident area and CCC.

Topology used in our simulation is based on description of ER-scenario. Like any other simulation, we have a sourcenode which represent the node capturing incident and a receiver, which in our case is CCC.

Listing 6.1: Creating nodes in ns3

```
NodeContainer nodes_sender.Create(nr_onloc_vidsources);
NodeContainer nodes_area1.Create(nr_onloc_nodes);
NodeContainer nodes_carriers.Create(nr_carriers);
NodeContainer nodes_ccc.Create(nr_receiver);
```

DTS-Overlay is implemented in each and every nodes participating in our experiments.

Listing 6.2: Installing DTS-Overlay on every node.

```
int i;
for (i = 0; i < numNodes; i++) {
        Ptr<DtsOverlay> dtsOverlay = CreateObject <DtsOverlay> ();
        all_nodes.Get (i)->AggregateObject (dtsOverlay);
}
```

The addresses we assigned for using in in wireless network is IP version 4, where source node and destination node is designated address 10.0.0.1 and 10.0.3.1 respectfully. Intermediate nodes are designated addresses between 10.0.2.1 - 10.0.2.X.

Listing 6.3: Set up an installing of source and destination node

```
// Set up and install video client (sender)
Ptr<DtsTraceClient> dtsclient;
dtsclient = CreateObject<DtsTraceClient> ();
dtsclient->SetAttribute ("RemoteAddress", destination);
dtsclient->SetAttribute ("RemotePort", UintegerValue (port));
dtsclient->SetAttribute ("TraceFilename",StringValue (trace_onlocvideo));
dtsclient->SetStartTime (Seconds (10));
dtsclient->SetStopTime (Seconds (duration-2));
all_nodes.Get (0)->AddApplication (dtsclient);

// Set up and install video server (receiver)
Ptr<DtsServer> dtsserver;
dtsserver = CreateObject<DtsServer> ();
dtsserver->SetAttribute ("Port", UintegerValue(port));
dtsserver->SetStartTime (Seconds (0));
dtsserver->SetStopTime (Seconds (duration));
all_nodes.Get (numNodes-1)->AddApplication (dtsserver);
```

Listing 6.4: Create trace file for visualization of the simulation

```
VisualizerTraceHelper traceHelper (duration*1000, all_nodes);
traceHelper.StartWritingFile (testname+".txt");
traceHelper.StaticPosition (numNodes−1, x_ccc, y_ccc);
traceHelper.StaticPosition (0, x_locsite, y_locsite);
```

6.1.2 Configuration

To compare analyze usage of priority queue with droptail-queue it is necessary to run simulation with different configuration of buffersize. In order to see the effect of the prioritization in the buffer we must make sure that size of the buffer is small enough that packet drop will occur.

Configurations that are used in our simulations for size of the buffer are set between 1 Megabyte and 10 Megabyte as shown in **??**. Buffer size larger than 10 Megabyte results that no packet are dropped from the Overlay will occur. Packets are still dropped, but caused by of other reasons, like maximum retransmission attempts by the MAC-layer.

Config	BufferSize
1	1 MB
2	3 MB
3	5 MB
4	7 MB
5	10 MB

Table 6.1: Table of configurations for BufferSize

Simulations are run on a computer with following parameters:

- 8 GB RAM
- Operative system: Ubuntu 10.04 LTS, Linux kernel 2.6.32-58.
- 4 cores CPU, 2,93 GHz each.
- ns3 version: 13

These paramenter does not have any affection on outcoming results of our simulations, but it has affection on usage of resources in our simulations. Some simulations takes longer time to complete than others, for instance simulations where larger queue of packets which needs to be sorted will require some more time to complete.

Nodes in Area	4
Area node mobilitymodel	RandomWalkMobility
Carriers	2
Carrier speed	10 m/s
Routing Protocol	OSLR
Duration	3600 seconds

Table 6.2: Parameters used in simulation

6.1.3 Simulation parameters

6.2 Results

6.2.1 Data sent from the source

	I-frame	P-frame	B-frame	M-frame
In packets	13328	46646	79964	11996
In Bytes	7117152	42781074	32558288	16794400

6.2.2 Data delivery

Config	I-frame	P-frame	B-frame	M-frame
1 PRI	1177799	5152097	2752066	1381149
1 FIFO	785203	4121678	3669422	2762298
2 PRI	2896146	14119238	6713496	2130978
2 FIFO	1930764	11295391	8951329	4261956
3 PRI	4613700	4613700	10529482.5	3805956
3 FIFO	3075800	18225775	14039310	7611912
4 PRI	5369520	25608247	12310188	12310188
4 FIFO	3579680	20486598	16413584	8054046
5 PRI	7648680	37094513	15035654	5096322
5 FIFO	5099120	29675611	23369132	10192644

6.2.3 Delivery percentage

Config	I-frame	P-frame	B-frame	M-frame
1 PRI	16.6%	12%	8.4%	8.2%
1 FIFO	11%	9.6%	11.3%	16.4%
2 PRI	40.7%	33%	20%	12.6%
2 FIFO	27%	26.4%	27.5%	25%
3 PRI	64.8%	56.9%	32.3%	22.6%
3 FIFO	43.2%	37.9%	43%	45%
4 PRI	75.4%	59%	37%	23.9%
4 FIFO	50.3%	47.8%	50.4%	47.9%
5 PRI	76.3%	86.7%	46%	30%
5 FIFO	71.6%	69%	71%	60.7%

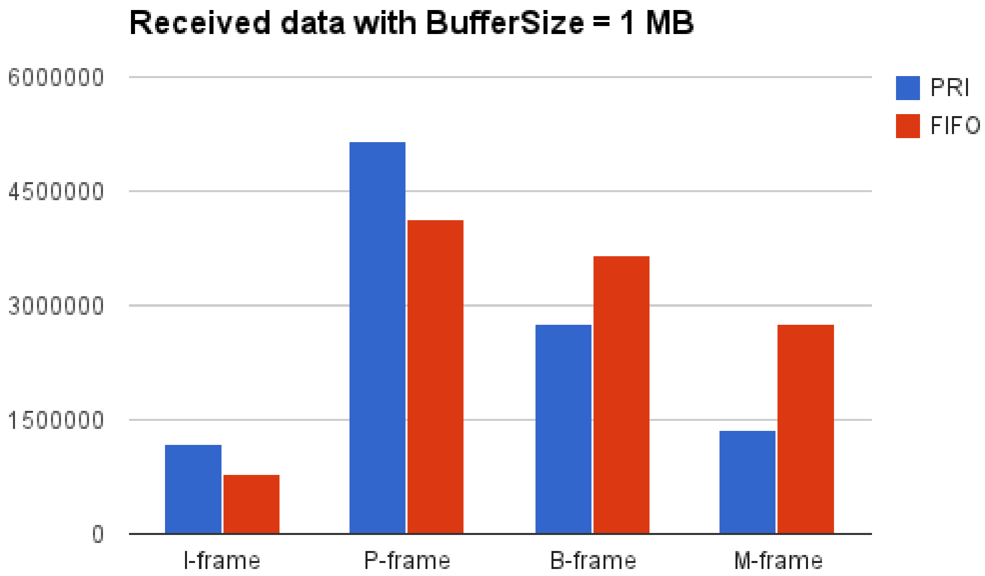

Figure 6.1: Received data with BufferSize = 1 MB

6.3 Evalation summary

We are interested in delivering most of I-frames and P-frames since they have a higher importance in our experiments. With the outcoming results from simulations we have managed to show that we are capable of providing come better QoS by replacing droptail-queue with a priority queue in a DTS-Overlay.

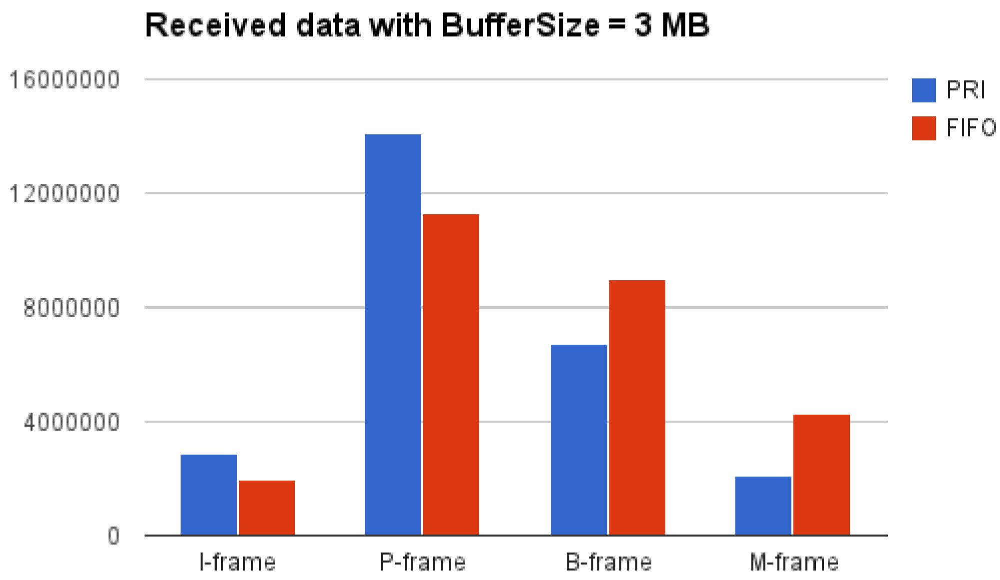

Figure 6.2: Received data with BufferSize = 3 MB

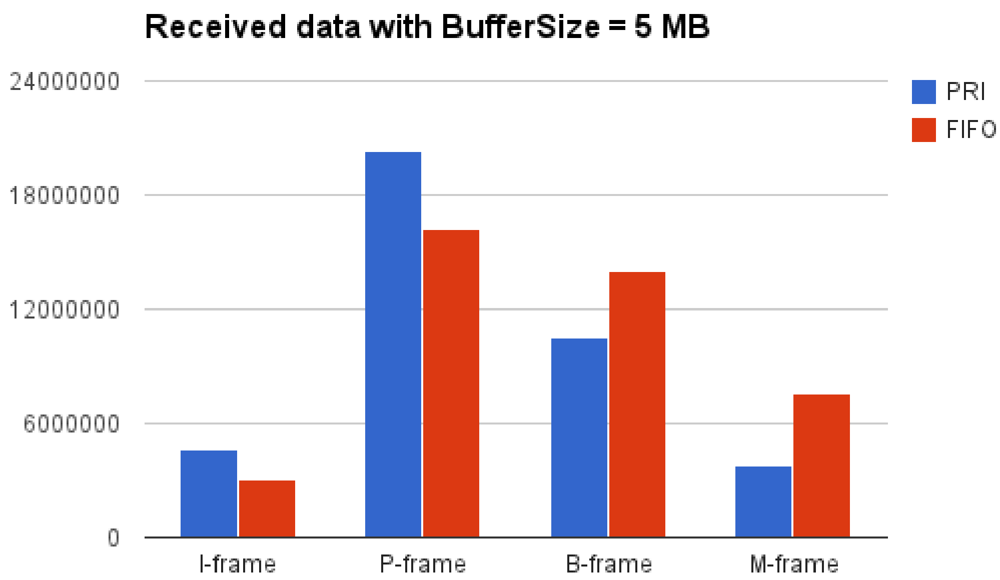

Figure 6.3: Received data with BufferSize = 5 MB

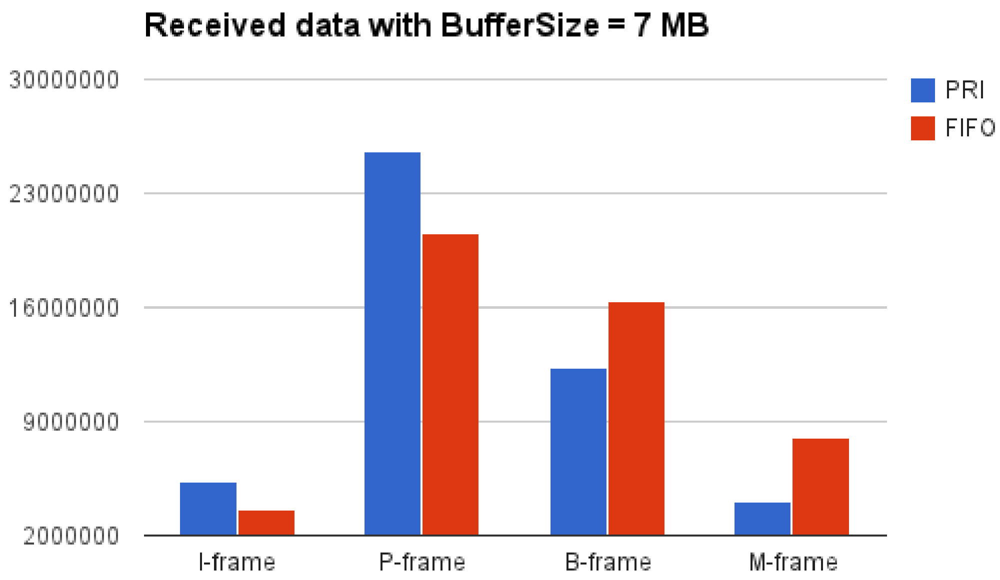

Figure 6.4: Received data with BufferSize = 7 MB

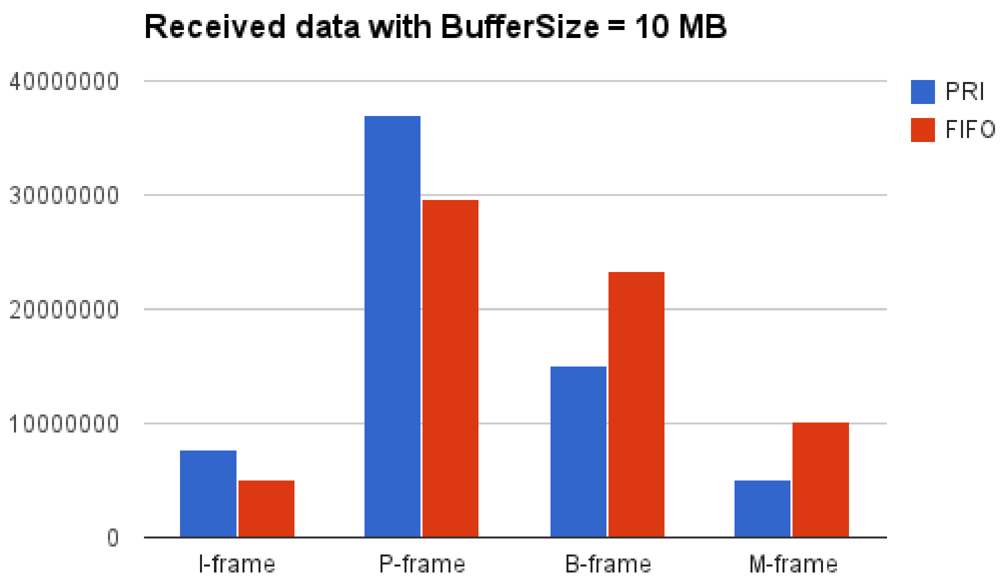

Figure 6.5: Received data with BufferSize = 10 MB

Chapter 7

Challenges faced in ns-3

Not everything went according to plan in this thesis. It took some time to figure out about a failure that was discovered very late during the evaluation process. In this chapter we will go through difficulties that was faced, how and why we did not were able to work further with initially problem statement.

7.1 Introduction

Initially our goal was to compare performance of QOS MAC with NQOS MAC in different scenarios and with different configurations, such as mobility, density, and buffer empty rate. During the evaluation-process abnormal and unnaturally differences were discovered in results. Further investigation indicated unpredictable factors that influenced the out coming results. Main reason for these unpredictable factors was the usage of randomness in ns-3. Before we go any further it is important that explain some functionality in ns-3.

Randomness in ns-3

According to ns-3, random numbers are provided through instances of `ns3::RandomVariable`. By default, ns-3 simulations uses a fixed seed to generate randomness for one simulation. For each run of the simulation should yield identical results if not the seed or run number is changed. In ns-3 version 13, to set a seed, `ns3::SeedManager::SetSeed()` is called at the beginning of a program. Run number, is called by `ns3::SeedManager::SetRun()` at the starting of the program.

Packet capture in ns-3

The ns-3 device helpers can also be used to create trace files in the .pcap format. The acronym pcap stands for packet capture, and is actually an API that includes the definition of a .pcap file format. Packet captures often analysed in wireshark.

7.2 What went wrong

Our coming result from evaluation proved significance differences between QOS MAC and NQOS MAC simulations over time. Delivery-over-time graphs showed unsynchronized increase in the graphs, which indicates differences in underlying parameters.

7.2.1 Challenges with mobility

Mobility model we use in our experiments is RandomWalkMobility, which is a mobility scenario that is increasingly relied on instances from `ns3::RandomVariable`. In ns-3 with help from VisualTrace module in the framework, it is possible to generate mobility trace for each and every node in simulation. To visualize mobility, we have used BIENVISTO, which is a tool that helps us to visualize a network simulation by reading mobility trace generated by ns-3.

Differentiation of mobility tracefiles generated by ns-3 for QOS MAC and NQOS MAC were huge. Visualization in BIENVISTO also showed different mobility. The larger the topology was, the more the simulation was influenced by randomness.

Workaround solution for mobility

As a workaround for this challenge we used a third party software named Bonnmotion to generate mobility. BonnMotion is a Java software which is developed by the Communication Systems group at the University of Bonn. Bonnmotion creates mobility scenarios and is commonly used as a tool for the investigation of mobile ad hoc network characteristics. This software can also be used to generate mobility scenario based on simple parameters from user. Mobility scenarios are generated as tracefiles which is read by several network simulators, such as ns-2 and ns-3.

By creating pre-generated node movements from Bonnmotion we were able to force ns-3 to follow identical movements in simulations with NQOS MAC and QOS MAC.

7.2.2 Challenges with routing

Furthermore in evaluation-process, results still showed differences which indicated irregularities. Observations in BIENVISTO, showed different routing in visual presentation of simulation for NQOS MAC and QOS MAC.

To investigate this we used Wireshark to analyze pcap from devices in simulations. Wireshark is a network packet analyzer, which captures and analyses network packets and displays packetdata as detailed as possible. Individual analysis of pcap files from sender-devices from NQOS MAC and QOS MAC showed variations in broadcasting of HELLO-packet of OSLR protocol.

As shown in , already few seconds in to the simulation, minor differences occurred, like different sending time of packets, orderness off broadcasting Hello message. This gave us reason to believe that these minor differences were led to a differences on a larger scale like "butterfly effect", which caused different route for each scenario.

Chapter 8

Conclusion and Future works

We have managed to show that we are capable of providing a better QoS for video at some level, but still it need improvements. Due to complications in implementation I have not managed to evaluate with other parameters.

8.1 Future works

Appendix A

DVD rom