QoS in distributed wireless 802.11-based multi-hop networks

vorgelegt von

Emma Maria Elisabeth Carlson
(M.Sc in Electrical Engineering)

von der Fakultät IV - Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
– Dr.-Ing. –
genehmigte Dissertation

Promotionsausschuss:

Vorsitzender: Prof. Dr.-Ing. Radu Popescu-Zeletin
Gutachter: Prof. Dr.-Ing. Adam Wolisz
Prof. Dr. rer. nat. Jens-Peter Redlich

Tag der wissenschaftlichen Aussprache: 12.02.2007

Berlin 2007

D 83

Zusammenfassung

Die Gruppe der IEEE 802.11 Protokolle ist heute die führende Technologie im Wireless Local Area Networks-Bereich (WLAN). Das wachsende Interesse für WLANs in den vergangenen Jahren hat zu Überlegungen, die Single-Hop-Umgebung zu einer Multi-Hop-Umgebung auszubauen. Damit könnte die Netzabdeckung verbessert werden als Knoten auserhalb ihrer Zielreichweite könnten einen dazwischen liegenden Knoten als Relais benutzen.

Heute wird von WLANs nicht nur höchste Qualität in den Anwendungen, z.B. E-Mail oder Web-Browsing, erwartet, zusätzlich wurde auch die Übertragung von Echtzeitanwendungen zu einem wichtigen Thema. Anders als beim best-effort Verkehr erfordern Echtzeitanwendungen eine Übertragung mit niedriger und konstanter Verzögerung oder konstantem Durchsatz. Leider unterstützt das konkurrenzbasierte Distributed Coordination Function (DCF) Zugriffs Protokoll, das in der Technologie von IEEE 802.11 benutzt wird, jedoch nicht solche Quality-of-Service (QoS); unregelmäigkeiten bei der übertragenden Instanz kann zu einem niedrigen Durchsatz und einer hohen und varierenden Paketrate führen. Diese Erscheinungen treten bereits bei einer Single-Hop-Kommunikation auf, verstärken sich aber noch bei Übertragungen über mehrere, nicht-synchronisiert arbeitende Hops.

Es gibt zwei Hauptansätze für QoS im DCF: Echtzeit-Ströme mit höherer Priorität als andere (Ströme) zu übertragen oder Ressourcen für sie zuzuordnen. Das Konzept vom Mechanismus der Priorität wurde in IEEE 802.11e standardisiert, das Enhanced Distributed Channel Access (EDCA) Protokoll. Leider kann QoS nicht garantiert werden; die Durchschnittsübertragung ist noch immer konkurrenzbasiert.

Eine Reservierung basierende Methode beseitigt Unregelmäigkeiten der konkurrenzbasierten Übertragung und stellt so potentiell end-to-end QoS sicher. Der Nachteil dieser Methode ist die Notwendigkeit für Verwaltung der Ressourcen in jedem Knoten und bei Datenübermittlungen zwischen den Knoten. Mein Hauptansatz ist das eine Reservierung basierende Methode ist die tauglichste für die Sicherstellung von QoS. Diese Ansatz war meine Motivation ein Reservierungsprotokoll für 802.11 basierte Multi-Hop-WLANs zu entwickeln. Das Protokoll wirkt in der MAC-Schicht, wo es periodisch auftretende Zeit-Slots in den Knoten dezentral reserviert

und heisst Distributed end-to-end Allocation of time-slots for REal-time traffic (DARE).

Hauptsächlich neuartig am DARE-Protokoll sind die folgenden, völlig verteilten Komponenten: eine Konfiguration mit End-to-End- Reservierungen, Unterstützung der Mobilität, die Verbreitung von ReservierungsInformationen mit niedriger Signalauslastung, die Koordinierung von Mehrfachreservierungen mit unterschiedlichen Zeit-Slots und die Verbreitung von Reservierungsinformationen über zwei Hops entfernte reservierte Übertragungen.

Das DARE-Protokoll ist mit DCF und EDCA vergleicht. Die Simulationsergebnisse zeigen zunächst, dass DARE bezüglich zeitlicher Verzögerungen zuverlässiger ist als DCF oder EDCA; es bleibt absolut konstant bei den Übertragungen. Bei EDCA und DCF treten deutlich grössere Schwankungen und durchschnittliche Zeitverzögerungen bei der End-to-End-Paketübertragung auf. Allerdings hat EDCA in Netzwerken mit niedriger Auslastung einen etwas höheren Durchsatz als DARE.Aber wenn sich die Verkehrsauslastung erhöht, verringert sich bei EDCA der Durchsatz und ist sehr viel geringer als der bei DARE. Die Zusammenfassung der Ergebnisse von Zeitverzögerung und Durchsatz ergibt klar, dass DARE die bessere Methode ist, QoS in verteilten 802.11 basierten drahtlosen Multi-Hop-Netzwerken anzubieten. Sogar in dynamischen Netzwerken, in denen DARE gezwungen ist, einen Grossteil seiner Bandbreite zur Wiederherstellung von Reservierungen und Verschiebung von kollidierenden Reservierungen aufzuwenden, bewährt sich DARE gegenüber EDCA und DCF.

Abstract

Wireless Local Area Networks (WLAN)s are today common for access to Internet; well-known and used is IEEE 802.11 standardized technology. Some drawbacks of the standardized distributed medium access scheme, Distributed Coordination Function (DCF) has resulted in considerations made to extend the DCF from single-hop communications to multi-hop. Multi-hopping has the potential to reduce energy consumption and increase throughput and coverage. The drawback is lack of Quality-of-Service (QoS) support, which already with single-hop communication was bad, but extending a communication between two peers to multi-hop significantly reduces the possibility to offer reliable transmissions. The contention-based medium access performed at each hop with the DCF results in large uncertainties as to when a packet arrives to its end-destination. There are two general approaches for extending the DCF with QoS functionality: 1. Allocate resources for a flow and 2. Assign a flow higher priority than other flows. My hypothesis is that for strict QoS guarantees, a reservation-based approach is the better one. This has been my motivation to design a new medium access protocol; based on the DCF, periodic time slots for QoS-demanding real-time applications are reserved. The *Distributed Allocation of time slots for Real-time traffic* (DARE) is unique as it totally distributed sets up an end-to-end reservation before the transmission of data begins, repairs broken reservations, support many periods and time slot sizes and distribute piggy-backed information even to a two-hop radius from a receiver. This thesis describes the DARE protocol and also presents results from comparisons of DARE, DCF and the IEEE 802.11 E standardized priority-based QoS medium access protocol *Enhanced Distributed Channel Access* (EDCA). Using simulations, this thesis shows that DARE offers constant end-to-end packet delay for a flow, very low average packet delay in the whole system and constant throughput. EDCA and DCF depend strongly on the total network load; average packet delay for a real-time flow increases rapidly with the surrounding load. DARE outperforms both DCF and EDCA when many flows are present, even in networks with frequent topology changes where reservations must be repaired. Thus, the reservation-based DARE protocol is the most suitable approach for extending DCF with QoS.

Acknowledgment

This work has been partly sponsored by Docomo Euro-Labs, Munich. Project leader at Docomo was Dr. Christian Prehofer, now at Nokia in Helsinki. His expertise in the area of communications made all project meetings and discussions very interesting. Prof. Dr. Christian Bettstetter, now at University of Klagenfurt, was part of the Docomo team of this project and was an incredible and enthusiastic person to change ideas with. His large knowledge within the area was very helpful.

Being part of the team at the Institute of telecommunication networks at the TU Berlin was fantastic. I have never had so many nice and incredibly skilled people around me. A special thanks to Martin Kubisch, Daniel Hollos, Marc Löbbers and Heike Klemz. Another great source of inspiration is Prof. Dr. Holger Karl, now at University of Paderborn, whose experience and expertise made daily project issues easy, and also, full of useful discussions. He is a never-emptying source of knowledge. A very special thanks to Prof. Dr. Wolisz, whose knowledge and great interest in the area made this work very interesting, pleasant and fun.

Finally, I want to thank my family and friends for all their support during this work.

Oslo, October 2006

Contents

List of Figures

List of Tables

Acronyms

AC	Access Category
AP	Access Point
ACK	Acknowledgement
AIFS	Arbitrary Interframe Space
AODV	Ad-hoc On Demand Distance Vector routing
AP	Access Point
BER	Bit Error Rate
CDF	Cumulative Distribution Function
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CTR	Clear-To-Reserve
CTS	Clear-To-Send
CW	Contention Window
DARE	Distributed end-to-end Allocation of time slots for REal time traffic
DCF	Distributed Coordination Function
DIFS	Distributed Interframe Space
gcd	greatest common divider
EDCA	Enhanced Distributed Channel Access
EIFS	Extended Distributed Interframe Space
MAC	Medium Access Control
NAV	Network Allocation Vector
PER	Packet Error Rate
PHY	Physical layer
QoS	Quality of Service
RSVP	Resource reSerVation Protocol
RTR	Request-To-Reserve
RTS	Request-To-Send
SIFS	Short Interframe Space

SINR	Signal-To-Interference-and-Noise-Ratio
TDMA	Time Division Multiple Access
UTR	Update-Transmit-Reservation
WLAN	Wireless Local Area Network

Chapter 1

Introduction

With the evolution from fixed networks with stationary desktops to networks with mobile users, moving along with their equipment i.e. lap-tops, the requirement on Internet access has changed. Users demand easy access and stable connections wherever they bring their computer. The access infrastructure technology has therefore also developed; from cabled Internet connections in fixed locations to the popular wireless Internet access infrastructure. Wireless access networks can simpler be deployed in network areas where fixed infrastructure is not possible, or areas where temporary access is required such as exhibition halls.

Further, wireless access networks are popular to use in combination with cellular networks to e.g. increase capacity [1]. Typically, such Wireless Local Area Networks (WLAN) are simple to set up, requires no large administration and are relatively cheap, infrastructure-wise. This has also lead to an increasing interest in using WLANs in home environments.

One popular technology is the IEEE 802.11 standardized, Carrier Sense Multiple Access with Collisions Avoidance (CSMA/CA)-based class of protocols [2]. Today this is the leading technology used in WLANs, in fact, the standardization of this rather simple contention-based wireless access technology has much led to the increased popularity of WLANs. Other approaches that exist, such as HIPERLAN/2 require more planning and a central node that controls all transmissions; such an approach is too complex to gain large interest in homes, temporary locations and hot spots.

This explosive increase in popularity has resulted in continuative work by IEEE 802.11 to develop new functionality for the original standard medium access mechanism. Extensions for higher data rates have been released in IEEE 802.11g [3] and security aspects in IEEE802.11i [4, 5]. Not yet standardized is to extend the single hop environment into a multi-hop one. First of all, this could increase coverage – nodes out of reach of its destination could use an intermediate

node as relay. Also, a node could use another one for relaying even when it can reach its destination, which could lead to a reduced total network energy consumptions and possibly more simultaneous transmissions [6]. A typical multi-hop network is shown in Figure 1.1. A source

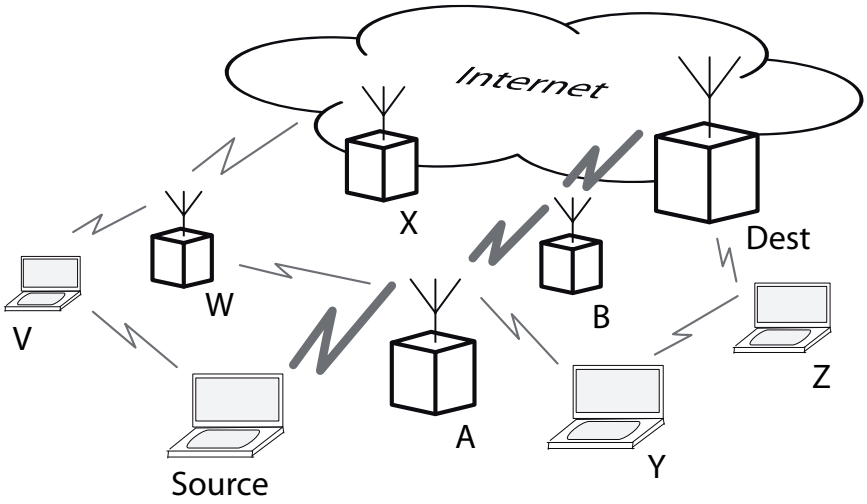


Figure 1.1: Illustration of a wireless multi-hop access network.

node $Source$ transmits over relay nodes A, B to the final destination $Dest$. Other nodes in the area can also use the same relay nodes, e.g. node Y and W.

In parallel to the development of the future multi-hop WLAN is the evolution of application types used in the network. Today, WLANs are not only expected to handle best effort applications, e.g. email or web browsing, but in addition support for transmissions of real-time applications has become an important issue. One example is Voice-over-IP: Due to the availability of easily downloaded software, this service has become very popular. The usage of such *real-time* applications will most probable grow even more; one example is video conferencing and other real-time streaming applications that are getting more and more popular.

Different from best effort traffic, these real-time applications require a transmission with low and non-varying packet delay or constant throughput, i.e. they require Quality-of-Service (QoS). Unfortunately, the basic contention-based Distributed Coordination Function (DCF) [7] access protocol used in the IEEE 802.11 technology, while supporting best effort traffic well, fails to support QoS [8]. While this is the case even with a single hop communication, cascading multiple non-synchronized hops results in summing up these effects [9, 10] – at each hop of the end-to-end path, a separate contention-based medium access procedure is performed.

To increase the reliability of transmissions, the IEEE 802.11 standard also specifies a central

MAC scheme, the Point Coordination Function (PCF). One node controls all transmissions in a network and allocate/distributes transmission instances for the nodes in the network. However, the PCF function has not met the expectations and does not perform well [11].

Some considerations have been given to use a mechanism not included in the standard itself; QoS routing or reservations on the network layer are some examples. Such approaches can improve the transmission quality for the real-time applications, but they cannot give strict guarantees. Even if a transmission instance is reserved for a flow on network layer, the medium access is still contention-based and it is fully possible that the reserved transmission is interrupted by another node. Therefore, an extension/modification of the DCF itself is needed.

One possible improvement is to give packets originating from real-time applications a higher *priority* when accessing the shared channel. This concept has been standardized in IEEE 802.11e, the Enhanced Distributed Channel Access (EDCA) [12] protocol. EDCA defines four different traffic categories: voice, video, best effort, and background. A node handles the packets in different queues, such that a packet from a queue with higher priority is sent before a packet from a queue with lower priority. In a single hop environment with moderate network load, EDCA can offer better average delay and throughput than DCF [13]. However, the performance of EDCA decreases rapidly with increasing load [14]. Also IEEE 802.11e specifies a central MAC scheme, the Hybrid Coordination Function (HCF). But, as described with the PCF above, this mechanism fails to improve reliability in multi-hop networks.

The alternative QoS approach, motivated in principle by circuit switching (wired technology), is to perform an end-to-end *reservation* for each real-time flow. While this approach potentially assures end-to-end QoS, some challenges exist in wireless networks that results in drawbacks such as need for resource management in each node and inter-node signaling. First of all, the *bandwidth* of a wireless network is typically rather scarce, hence any solution must be effective in its way of using the resources. Then, a major issue is that a distributed wireless network has no well-defined collision domain. A node not even intending to communicate with another node can cause severe *interference* at this node when starting a transmission. Wireless nodes have no knowledge about exact conditions of the network and it is difficult for them to make accurate decisions. Further, nodes can be *mobile*, or at least leave and enter the network in an unpredictable manner. This can cause multi-hop paths to break and reservations break before an application session is over. Finally, the multi-hop paths can cross, or intervene with each other in the same neighborhood. This can results in *congestion* if the different transmissions are not handled (scheduled) properly. No transmission should be allowed if it reduces the QoS of other transmissions such that their requests are no longer met. Such functionality is difficult in a network where each node must somehow make an intelligent decision based on local information.

A reservation protocol that considers the challenges above is today not available for IEEE 802.11 networks [15]. This lack has been my motivation to develop a protocol in this domain, which is well suited for multi-hop 802.11-based WLANs. The protocol operates in the MAC layer, where it reserves periodically occurring time slots in the nodes in a completely distributed manner and is called *Distributed end-to-end Allocation of time slots for REal-time traffic* (DARE).

To be more specific, before a real-time transmission can begin, DARE reserves time slots in all nodes along an already existing route between the source node of a real-time flow and its final destination node. It then schedules the real-time data packets between the nodes, transmitting them in the reserved time slots. In essence, the protocol extends the spatial reservation concept of 802.11 — achieved by the exchange of Request-to-Send (RTS) and Clear-to-Send (CTS) messages — to a multi-hop, end-to-end perspective.

Further, DARE protects the allocated time slots from interference by informing nodes located near the real-time path, using a piggy-backing technique. The adjacent nodes will thus abstain from transmitting during the reserved time slots. Additionally, an optional feature where a reservation is protection at a wider range around receivers than in the direct neighborhood (two-hop protection) is included.

The DARE protocol is not limited to one period or time slot size; different flows can have different requirements. The required functionality demands some inter-node signaling at reservation set-up, which is kept local and as simple as possible. DARE also handles the repair of broken reservation paths and release of unneeded reservations.

Data packets coming from non-real-time applications use the CSMA/CA approach of DCF, either with or without the exchange of RTS/CTS messages. Data packets from real-time applications, however, use DARE and are transmitted during the reserved time slots. The resulting medium access protocol is a combination of CSMA and Time Division Multiple Access (TDMA).

1.1 Thesis contributions

The basis for this thesis is my hypothesis that a reservation-based medium access protocol for IEEE 802.11 networks is the best option for strict QoS requirements. This is motivated by the first part of this thesis, which covers an analysis over the different possible solutions for QoS, and more importantly, a discussion of why these solutions are not efficient enough. The analysis covers all different protocol levels (according to the OSI-layer model) and in different domains

(frequency or time).

There are several challenges for a distributed reservation protocol, as mentioned above, which should be considered in the QoS mechanism to achieve best results; interference, mobility and bandwidth constraints are some of them. These challenges lead to some requirements for a reservation-based QoS mechanism. To find the best solution, existing and well-known mechanism are compared to the requirements; all of them failing at least one of them, which motivates my main research goal: Design of a new distributed reservation protocol, which supports both best effort and real-time traffic in wireless multi-hop networks. My hypothesis is that this can be achieved by using a combination of carrier sensed based medium access for best effort traffic and periodic time slot reservations for QoS demanding applications.

The contributions of this thesis is the development of a reservation protocol, analysis of its features and performance comparisons with the DCF and EDCA.

The reservation protocol has the following unique properties: 1. End-to-end aspect. The reservation protocol treats a flow from source to destination as one transmission rather than the DCF, which considers each hop as one separate transmission. 2. Repair of broken reservations if nodes that are participating in a reserved multi-hop transmission are leaving the network. 3. Support of different periods and time slot lengths, no pre-defined frame structure is needed and different applications that have different requirements on the period of the packet transmission can co-exist. For this, a special scheduling mechanism is derived.

1.2 Structure of thesis

This thesis is structured as follows: Chapter 2 describes background information of 802.11 networks and relaying, Chapter 3 describes the system model and Chapter 4 describes the available QoS mechanisms and concludes with the best option for a new medium access protocol. Chapter 5 describes constraints and requirements for the new reservation-based protocol, as well as related work. Chapter 6 describes basic features and presents a first simulative investigation of the new reservation-based protocol, DARE, and also compares it to the DCF in a simple deterministic scenario. Chapter 7 describes maintenance of set up reservations, which includes how to acknowledge reservations as alive, repair mechanism. Chapter 8 describes an optional feature of the DARE protocol – extended protection of a reservation two hops around a receiver. This chapter also contains a simulative investigation of how effective the optional feature is. Chapter 9 summarizes the design part for one real-time flow with a simulative comparison of DARE and EDCA. After this, Chapter 10 describes the problem of multiple reservation, deriva-

tion of the scheduling mechanism, protocol implementation and probability that reserved time slots overlap and must be re-scheduled. It also discusses limitation to the number of flows that can be reserved. Chapter 11 describes the main evaluation of the DARE protocol. The DARE protocol is also compared with the DCF and EDCA for many real-time flows and varying network parameters. Finally, Chapter 12 concludes this thesis and makes suggestions for further studies.

Chapter 2

Background

This chapter describes background information needed for the considered problem and discussion (and conclusion) of suitable solution in this thesis. First, the required parts of the IEEE 802.11 technology is described in Section 2.1. Then, Section 2.2 describes multi-hopping in 802.11 based networks, which concludes with why it is problematic to achieve QoS support. Finally, Section 2.3 describes some of the QoS approaches that exist for wired networks and why these are not appropriate for wireless multi-hopping 802.11-based networks.

2.1 IEEE 802.11 System description

2.1.1 System architecture overview

An IEEE 802.11-based WLAN consists of one- or several Basic Service Set(s) (BSS). A BSS consists of a set of nodes that communicate via an Access Point (AP). The nodes are not stationary and they can move around while communicating. IEEE 802.11 also describes Independent Basic Service Sets (IBSS), which in a way is a BSS without an AP. Any node can initiate a network, by transmitting periodic beacon frames [7]. As more nodes join this network, all involved nodes take turn to transmit these beacon frames. Nodes communicate directly with each other, no AP is involved. These network types are not further described here. The geographical area covered by one AP is called the Basic Service Area (BSA). Several BSSs can be connected with any type of Distributed System (DS). Before any node can start a communication, it must be associated with one AP, and it can only be associated with one AP at the time. Within one BSS, one Coordination Function (CF) controls the medium access to the shared wireless channel, i.e. a Medium Access Control (MAC) protocol. The CF can be either centralized or distributed. In a BSS with central CF, the AP schedules all transmissions and decides when all nodes are allowed

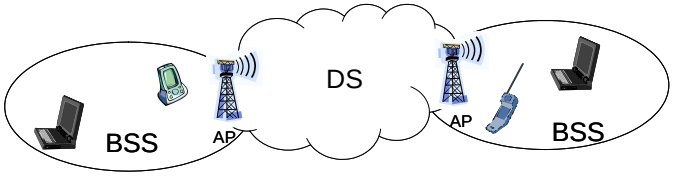


Figure 2.1: Basic WLAN structure.

to transmit. In a BSS with distributed CF, the nodes compete for access – no transmit schedule exists.

Figure 2.1 shows a general WLAN system architecture. The protocols used by all 802 variants, including Ethernet, have a certain common structure. An overview of the protocol stack is given in Figure 2.2. The data link layer is divided into two sub-layers, the Logical Link Control (LLC) and MAC layer. The LLC layer is not unique for 802.11, it is the same for all 802.x standards, originating from Ethernet, 802.3 [16]. This allows for a seamless bridging between wireless IEEE networks to wired IEEE networks. The LLC layer is not further described here. The physical and MAC layer are specific for the 802.11 standard and described in more detail in the following two sections, Section 2.1.2 (physical layer) and Section 2.1.3 (MAC).

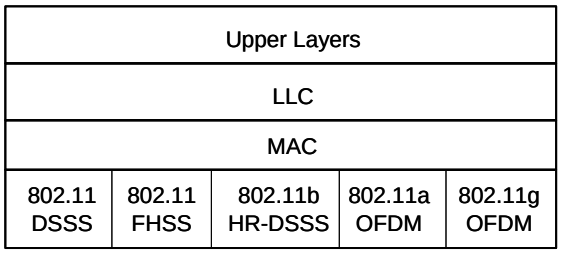


Figure 2.2: Overview of 802.11 protocol stack.

2.1.2 802.11 Physical layer

The 802.11 standard specifies three different transmission techniques: Infrared, Direct Sequence Spread Spectrum (DSSS) and Frequency Hopping Spread Spectrum (FHSS). In addition, two other techniques are specified in 802.11a [17] and 802.11b [18]: Orthogonal Frequency Division Multiplexing (OFDM) and and High Rate Direct Sequence Spread Spectrum (HR-DSSS). Also an OFDM technique for another frequency band than 802.11a was introduced in 802.11g [3]. Each of the five permitted transmission techniques makes it possible to send a MAC frame from one node to another. FHSS uses 79 different frequency channels, each 1 MHz wide in the 2.4 GHz frequency band. A pseudorandom number generator is used to produce the sequence

of frequencies hopped to. As long as nodes are synchronized and use the same seed for the random generator, they will hop to the same frequencies simultaneously. With DSSS each bit is transmitted as 11 chips using a Barker sequence and phase shift keying as modulation. The first of the high speed WLANs, 802.11a uses OFDM to deliver up to 54 Mbps in the wider 5 GHz frequency band. 52 different frequencies are used to split a signal into many narrow bands, simultaneous transmitted. The modulation used are based on both phase shift keying and on QAM. HR-DSSS was introduced in 802.11b and is used to achieve a transmission rate up to 11 Mbps in the 2.4 GHz frequency band. The modulation Complementary Code Keying (CCK) is used. Although this is slower than 802.11a, the range that can be achieved with 802.11b is a large advantage. An enhanced version of 802.11b was approved by IEEE in November 2001, the 802.11g which operates in 2.4 GHz band as 802.11b, but uses OFDM modulation technique. In theory it can operate up to 54 Mbps.

The physical layer can be divided into three sub-layers. The Physical Layer Convergence Procedure (PLCP) adds a PLCP header and a preamble to the MAC frame. They are used at the receiver for demodulation and delivery of the packet. The Physical Medium Dependent (PMD) sub-layer divides the finished PLCP frame into different parts before transmission. This is done as different parts of the packets are transmitted with different transmission rates (modulations). For instance, the MAC header is always transmitted with the basic rate of 1 Mbps. Last, the Layer Management Entity (LME) manages all physical sub-layer functions.

2.1.3 802.11 MAC layer

This section describes the MAC layer of 802.11. In the first subsection, an overview and background of the MAC protocol is given. Then, the following subsections describe the different parts of the distributed CF in more detail. First the carrier sensing mechanism is described, followed by the different waiting times specified in [7]. Following this is a section over a back off mechanism used whenever a node senses the channel busy. Finally, the MAC frame formats are presented.

Overview

The MAC sub-layer of 802.11 standard is responsible for the channel allocation procedures, frame formatting, error checking, segmentation and reassembly. The basic MAC function is the Distributed Coordination Function (DCF), which is a distributed contention-based access scheme. The Point Coordination Function (PCF) [7] can be used on top of the DCF to achieve a centrally controlled transmission schedule. Further, specified in IEEE 802.11e is the Enhanced Distributed Channel Access (EDCA) [12], that allows for service differentiations. Also here,

a central approach is available, the Hybrid Coordination Function (HCF). Another protocol is standardized in IEEE 802.11i [4], which specifies some MAC adjustments for security and in IEEE 802.11h [19] Dynamic Frequency Selection (DFS) and Transmit Power Control (TPC) for the 5 GHz band are specified. In this thesis the DCF is considered as the basic medium access scheme.

When the DCF protocol was designed, the working group for 802.11 chose Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) as the basis for the DCF. Using a CSMA algorithm, all nodes sense the channel before transmission and when busy, they do not transmit. Additionally, the Collision Avoidance algorithm introduces an additional waiting time before transmission, even if the channel at first is sensed idle. The CSMA/CA base as such minimizes costly retransmissions of collision-based packet losses, which in networks with undefined collision domains can be frequent. A CSMA/CA approach as base, this problem is to some extend prevented as the transmitter of any communicating nodes can abstain from transmission if the channel is sensed busy.

However, terminals can be *hidden* and not able to sense an ongoing transmission, which can degrade system performance [20, 21]. This is illustrated in Figure 2.3, where node A is outside the communication range of node C. When node C transmits to node B, A will not sense the

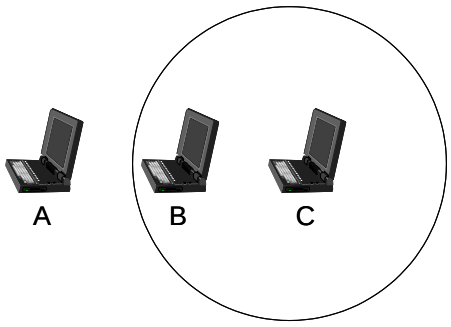


Figure 2.3: Node A is hidden from transmission from node C to B.

channel busy and can with a transmission ruin reception at node B.

To minimize the effect of hidden terminals, the DCF has a specified handshake procedure, first introduced in the MACA protocol [22]. The transmitting node transmits a request for a transmission of a data packet to the intended receiver, which responds only when the requested transmission is ok. The handshake messages are called Request-To-Send (RTS) and Clear-To-Send (CTS). The RTS/CTS exchange serves two purposes: First the exchange enables the receiver to prevent collisions and second the exchange also informs nodes surrounding the communicating pair of nodes about the upcoming transmission so they can abstain transmission. The RTS and CTS are small [7], hence a collision with two RTS messages is not as costly as

when two data packets collide and must be re-transmitted. However, when packets are small, the RTS/CTS exchange introduces unnecessary overhead; The RTS/CTS exchange is optional.

A side effect of the CSMA/CA with an RTS/CTS exchange is the exposed terminal problem, illustrated in Figure 2.4. Here, B wants to send to A, hence it listens to the channel. It overhears

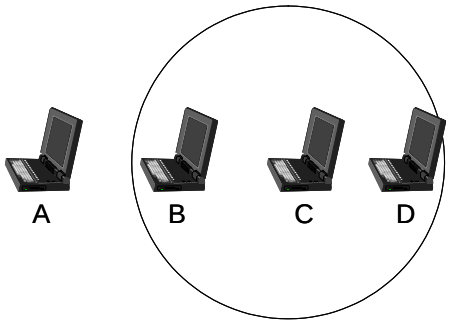


Figure 2.4: Exposed terminal problem: Two possible simultaneous transmissions, B to A and C to D, are prevented.

a transmission from node C to node D and abstains from transmission. Node D and A are so far apart that theoretically the two transmissions could occur simultaneous but this is prevented by the sensing mechanism.

After a successful transmission the receiver acknowledges the reception of the data packet by transmitting an acknowledgment (ACK) back to the transmitter. If a transmitting node does not receive an ACK, it retries after some additional waiting time (explained more below). After x unsuccessful retransmissions, a node considers the transmission unsuccessful and drops the packet. The variable x is in [7] specified to 4 for large packets and 7 for small. However, both the packet size and the retrial count are tunable.

Carrier sensing mechanism

DCF applies both physical and virtual carrier sensing. Physical sensing is when any node actually senses the physical channel for ongoing transmissions. Virtual sensing means decoding overheard messages and reading information from them to get an idea of how long the transmission is ongoing and the node must abstain from transmission. Each node keeps a Network Allocation Vector (NAV) with information about how long they must wait. The advantage with this approach is that a node must not continously sense the channel physically; it retries after the NAV has expired. The algorithm is described using the example shown in Figure 2.5. Here, node A wants to send to node B, node C is a node within range of A and node D is out of range of node A but within range of node B. A senses the channel and when idle it sends an RTS to

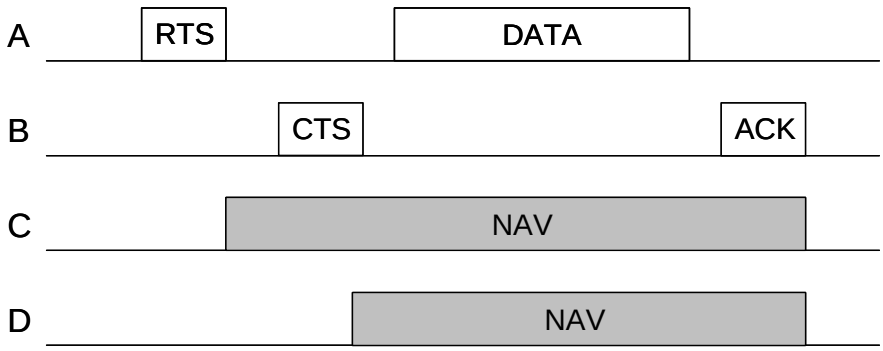


Figure 2.5: Transmission flow and virtual sensing mechanism.

node B requesting permission to transmit. When node B receives this message it here decides to grant the request and replies with a CTS. When A receives the CTS it transmits the data to B, which upon a correct reception answers with an ACK.

Node C overhears the RTS and sets its NAV according to the information specified in the RTS and avoids transmission for as long as the NAV is set; the RTS contains information about the duration of the upcoming data transmission. Also the CTS contains this information (minus the time for transmission of the RTS), hence node D that is only within range of node B can also set its NAV and avoid transmission for the time requested.

Inter Frame Space

DCF defines three different intervals that are used in between frames, each for a specific purpose. The shortest interval is the Short InterFrame Space (SIFS). This is used at any intended receiver in between the reception of RTS and transmission of the CTS, and between data reception and transmission of the ACK. By allowing nodes to use the shortest waiting time in between these actions, the transmission of an ACK and CTS always have higher priority on an idle channel than other actions. Next is the DCF InterFrame Space (DIFS), which is the defined time a node must wait before initiating the data transmission after it has sensed the channel idle. This means before the transmission of the RTS, or when the RTS/CTS exchange is not used, before the transmission of the data packet. The last interval is the Extended InterFrame Space (EIFS), which is used by any nodes that suffer from an unsuccessful transmission before the retransmission. This is the largest interval and gives all involved nodes extra time to learn what is going on in the network.

Back off mechanism

When a node senses the channel busy before transmission, it starts a backoff mechanism. This mechanism works as follows: From a range of natural numbers called the Contention Window (CW), a node picks one. This number defines the number of $aSlotTime$ it must wait before transmission, where $aSlotTime$ is a specified physical layer "time slot" parameter [7]. For each $aSlotTime$ that the node senses the channel idle, the random number is decreased with one. When the back off counter reaches zero, and the channel is still idle, the nodes wait an additional time of DIFS. If the channel is still idle after the DIFS, the node can transmit.

If the channel at any time during the back off is busy, the node freezes its timer and continues the count down only after the channel has been idle for minimum a DIFS.

The CW is defined by a minimum and maximum value, $[CW_{min}, CW_{max}]$. Whenever a node must retransmit a packet the CW_{max} is multiplied with 2. Since a retransmission only occurs after an unsuccessful transmission, i.e. collision, the CW is enlarged; a larger range decreases the probability that two nodes choose the same number, hence decreases the probability of further collisions. There is a maximum number of CW_{max} and when a node reaches this limit, the CW_{max} is no longer increased.

Figure 2.6 gives an example of how the 802.11 DCF works with the back off mechanism. At the beginning of this example node A, node B and node C have their NAVs set due to a

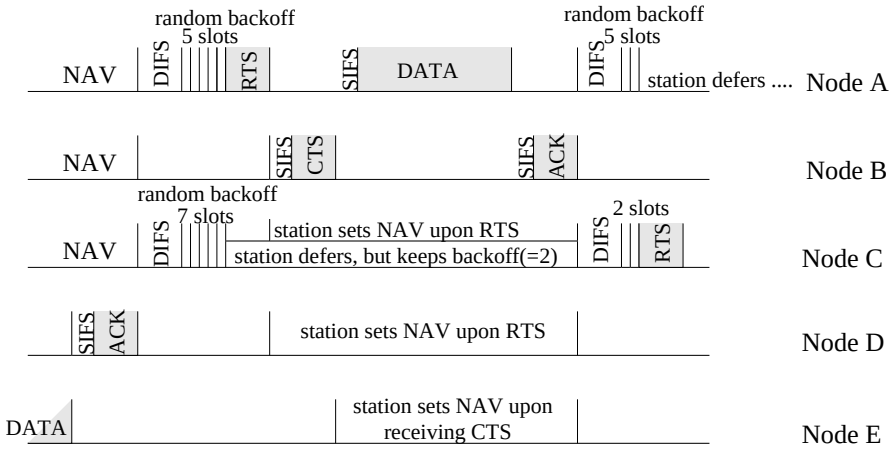


Figure 2.6: Timing of the 802.11 DCF.

transmission between node E and node D. Assume that both node A and node C have a packet to send. After D and E have finished their transmission, node A and node C start to contend for accessing the channel. At first, both nodes have to wait for DIFS, then they start their random backoff. Assume that node A has to wait for 5 slots whereas node C has to wait for 7 slots.

As a conclusion node A sends an RTS to the receiver, e.g. node B, after backing off during 5 $aSlotTime$. Node C detects that the channel has become busy, so it freezes its backoff and defers. Nodes A and B now start the RTS-CTS-DATA-ACK exchange. Upon the reception of the RTS nodes C and D set their NAV until the end of the transmission procedure between nodes A and B, i.e. they can overhear node A. Node E sets its NAV upon the reception of the CTS, i.e. it cannot overhear node A but it can overhear node B. After the transmission procedure is finished node A (assuming node A wants to transmit again) and C start to compete for the channel again. After waiting for DIFS node A generates a new random number of slots it has to wait because it wants to start a new transmission. In contrast to that, node C only has to wait for the remaining two slots; the backoff for this packet was seven slots but node C has already waited for five.

DCF MAC frame format

The different DCF message types are the RTS, CTS, data and ACK. All messages have three frame fields in common: Duration, Frame Control and Frame Check Sequence (FCS). The Duration field contains the duration in time of the intended transmission (including transmissions of RTS/CTS packets). All nodes in the neighborhood capable of decoding the messages will receive information about the transmission length. The duration is of different size in all messages, e. g. the RTS Duration is the longest; it contains the total duration of the transmission with RTS, CTS, data and ACK, whereas the data frame has only the duration of the data frame and the ACK. The frame control and FCS are used to control the frame and to assure delivery of a packet is without errors.

Figure 2.7 shows the RTS frame format. Additionally to the three common fields, the RTS

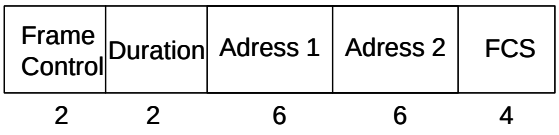


Figure 2.7: RTS frame format with sizes in bytes.

has two address fields which is the MAC addresses to the receiver (Address 1) and the transmitter (Address 2). The CTS illustrated in Figure 2.8 also has the Address 1 field, but no MAC transmitter field; is not necessary in the CTS. The receiver of the CTS is the transmitter of the RTS and the data frame; the Address 2 field is not required.

Figure 2.9 shows the data frame. The data frame have the Address 1 and Address 2 fields as the RTS packet. In addition the data packet has two more address fields, Address 3 and Address 4. These fields are used for identification of the BSS.

The ACK has the same appearance as the CTS and is not shown with an additional picture.

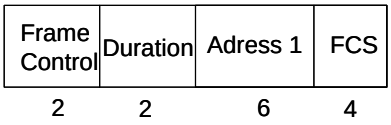


Figure 2.8: CTS frame format with sizes in bytes.

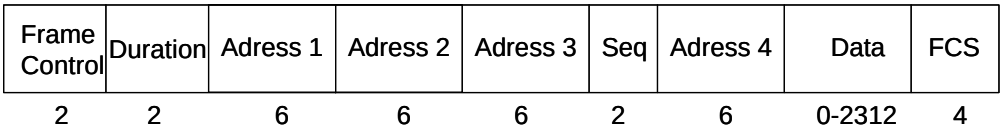


Figure 2.9: DCF data frame format with sizes in bytes.

2.2 Relaying in 802.11-based networks

Relaying, or multi-hopping, is a potential option to increase capacity, extend coverage and improve energy efficiency in wireless networks [6]. It can be conducted in networks with varying degrees of infrastructure such as pure AP-based networks, fixed relaying nodes, or pure ad-hoc networks. Also, relaying can and has to be controlled in the sense of both routing and medium access, either in a distributed or centralized fashion [23]. This section discusses the different options to implement multi-hopping in wireless networks. It covers the following different topics: 1. What to relay, meaning repeat at physical layer or digitally decode and forward. 2. Who should perform the relay, fixed relay nodes or the participants themselves. 3. In which domain should the relaying occur, frequency, time or code.

2.2.1 What is relayed?

Relaying can be done on the basis of either an electro-magnetic impulse (repeated analogously) or a digital entity (store and forward). Using an electro-magnetic impulse, an intermediate node forwards the packet directly on the physical layer. It amplifies the incoming signal with hardly any delay, which is a large advantage. Also, there is no requirement for decoding or any other processing. However, a downside with this method is that packet errors are also amplified.

As an digital entity, an intermediate node decodes the whole packet before forwarding it. Thus, an intermediate node receives, decodes, and possibly alters the packet before forwarding. The packet goes up in the protocol stack to find the next node. This is a more complex solution as processing in intermediate nodes is needed. Some advantages with this solution is that analog errors can be compensated and that e.g. routing decisions for this packet can be taken in intermediate nodes.

Some analysis have been made as to which method is the most suitable one for distributed

wireless networks [24]. The conclusion is that for small path loss coefficients (below 3/2), analogue relaying is the better method. However, in wireless networks this is rarely the case; the path loss coefficient is much higher. Even in Line-of-Sight models where no fading occurs, the path loss coefficient is 2. Thus, the digital "store and forward" method is assumed to be used for relaying.

2.2.2 Who relays?

Relaying can be done by dedicated relay nodes or by the users themselves. Using dedicated nodes makes it possible to have some sort of cell-planning. It is possible to put up more relay nodes where the traffic is heavier. It is also possible to increase the coverage of the network by putting a relay node in side streets, etc. A problem with this is that it is hard to know where to put the relay node; mobile users move around in an unpredictable manner. Research in this area has been done by, e.g., RWTH Aachen, where they call their relaying nodes *mediastations* [25]. The dedicated nodes can also be interconnected by wires, which means that these nodes do not need to use the already limited air interface.

Relaying done by participants is a simple approach; nodes themselves forward packets on behalf of others and no extra processing is needed. Disadvantages with this solution are that the nodes need to use battery power for transmissions other than their own. However, this approach is the simpler one as no fixed infrastructure is required and the only one considered in this thesis.

2.2.3 Relaying in which domain?

Relaying can be performed using three different multiplexing techniques: Time-, Frequency- or Code- Division Multiplex (TDM, FDM, CDM).

- TDM: The multiple packet transmissions that are required for relaying are serialized in time. As an intermediate node receives a packet destined for another at a certain time slot, it forwards it in a following slot. This is the simplest solution and it is supported by current technology, i.e. IEEE 802.11.

  It is possible to give priority for applications with QoS demands, by scheduling traffic and letting high priority packets be transmitted first.

- FDM: Forwarding is done using different frequency channels. One example is to use separate frequency channels for relay packets and for own packets. There are several benefits with this concept, but it is also difficult to implement. Frequency planning is required for the best re-usage and utilization.

- CDM: Forwarding is done by means of using several codes for transmission. Possible ways of implementing different codes in ad hoc networks are: Code per terminal, code per originator or code per flow. A disadvantage of this mechanism is that intermediate nodes must have knowledge about codes not used for their own transmissions, which is very difficult. Another issue is how to distribute codes in best way to avoid interference.

The most simple approach in the DCF is to use a TDM-based forwarding scheme. This fits with the general contention-based access scheme DCF as it operates in the time domain. A relaying node treats a relay packet as any other packet and no major changes in the network are required.

2.2.4 Impact of relaying in DCF

Several possibilities for relaying in wireless 802.11-based networks exist. Here, the nodes themselves relay packets (no fixed relay nodes) and treat the relay packets as any other packet; they decode each packet, let it travel up the protocol stack (layer 3) and forward according to its local routing table. This implies that a relay packet must contend as any other packet at each node, when DCF is in use. For the end-user this means that there are uncertainties as to when a packet arrives; the packet delay can be large and varying. Although this is also the case for single hop environments, the variation of the delay is more severe when communication over multiple hops occur. For some applications that demand a low and stable delay, it is hard to guarantee an end-user reception that has good quality; there is no *Quality of Service (QoS)* support [8, 9, 26]. Examples of applications that could suffer are video-streaming (multimedia services) and normal voice (phone calls), i.e. real-time applications. Thus, some QoS enabling mechanism should be used in conjunction with the DCF. QoS has its background in wired network technology and the known algorithms are discussed in the following section.

2.3 QoS support in wired networks

QoS is known in its principle from wired network technologies. The first technology that could give guarantees to a transmission between two communicating peers is circuit switching, e.g. the telephony system. Here, switches along the multi-hop path were totally reserved for the complete session time. Different is the access mechanism in wired IP-based networks. Here, the MAC protocol was designed for packet oriented data traffic where each packet is treated separately; the IEEE 802.3 standardized Ethernet [16] is still the most used MAC protocol today. To offer QoS in such CSMA/CD-based networks, ideas from the circuit switching techniques were included, resulting in that service differentiation and bandwidth reservation could be supported.

There are many existing approaches how this QoS support was implemented. Here only two are discussed, which are somewhat of de facto standards for QoS in wired networks: Integrated Services, IntServ [27] and Differentiated Services, DiffServ [28].

IntServ [27] is an architecture for QoS support that uses the Resource reservation protocol (RSVP) [29] for signaling to reserve bandwidth for a flow. Three classes of service are defined: guaranteed service, in which delay is limited and zero packet loss is guaranteed; controlled load service, the aim of which is to provide the same services regardless of the network load; and the best-effort service which has no guarantees.

The reservation is maintained *soft state* and needs at each node a periodic refresh. Further, the reservation is receiver-oriented; the actual bandwidth requested is determined by the receiver. The reservation set-up starts with a *path* message that is transmitted from the source node over established unicast- or multicast routes. This includes among other things traffic characteristics of the following application flow. If this message arrives at a router that does not understand the RSVP protocol, it is simply forwarded without any interpretation at this router, hence no resources are reserved. Upon arrival at the destination node this will make a reservation-based on the traffic characteristics described in the path message. The destination node generates a *resv* message that contains a *flowspec*, which states the actual required QoS. The resv message is sent back along the same path to the source node. Each node along the path can reject a request.

The RSVP protocol is open for alternations and many variants exist, e.g. *Mobile RSVP* (MRSVP) [30] that gives additional support for mobile terminals.

DiffServ [28] is different from IntServ as it does not reserve any bandwidth, instead it uses different priorities for different flows. By assigning flows to several service classes whose packets are treated differently at each router, some flows have higher precedence over other flows, with lower priority. The source of a flow marks all packets using the Type-Of-Service (TOS) field of the IP header according to the class of the data. As packets arrive to any router on their way to the destination node, a packet with a higher priority label will be handled before a packet with a lower TOS priority label. One evident advantage with DiffServ is that involved nodes need no resource management functionality. They forward the packets according to the class given in the packet. A major disadvantage is that there are no strict QoS guarantees.

Both approaches can improve the quality of a transmission in wired networks. Typically, a larger wired network consists of several well-defined sub-networks and by implementing any QoS approach in all of the sub-networks, bandwidth guarantees can be met. In all sub-networks, the wired communication between two nodes occur either over a point-to-point link or over a shared medium. Over a point-to-point link, any QoS approach is straight forward; the sending node is in complete control of the channel and decides which packet that should be transmitted at which instant, without interruptions from other users. In the shared medium, it is also possible to

agree on a determined share of bandwidth: All nodes within the domain are able to communicate with each other directly and whenever a node transmits data to another, all other nodes are aware of that. Thus, a wired network with a shared medium is a well defined, closed collision domain – A successful collaboration between all nodes is possible. Therefore, IntServ and DiffServ can be successfully implemented in any wired network.

Applying IntServ in wireless ad hoc networks is difficult; the wired and wireless networks have not the same characteristics (no closed collision domains, mobile nodes, fading, ...) and assumptions made for QoS support in wired networks do not hold in the wireless ones [8]. Further, the RSVP has a large signaling load, which is not suited for wireless ad-hoc networks where resources are scarce. Also, reservations made above the network layer cannot be guaranteed as no predefined transmission schedule with repeating time slots exists; nodes cannot identify collision domains such that a transmission schedule can be maintained.

Applying DiffServ in wireless ad hoc networks is possible and packets with higher TOS field could be sent to the MAC layer before other packets within one node. However, two nodes contending for access have the same probability of channel access; no precedence is given to a packet with higher TOS number at one node towards a packet with lower TOS number at another node. In highly loaded networks, the DiffServ approach will therefore not be able to give higher priority for some traffic types.

To conclude, new approaches for both bandwidth reservation and service differentiation is needed, designed particularly for distributed wireless multi-hop ad hoc networks.

Chapter 3

System description

This chapter describes the system for which QoS should be supported. The system consist of nodes that are connected to an outside network (not considered here). Some nodes function as gateways out and are called Access Points (AP)s. These are described more in Section 3.1. Other nodes transmit using the APs to an outside network and they can all relay transmission for each other and are described more in Section 3.2. Further Section 3.3 describe protocols used for transmissions and Section 3.4 describe the different traffic types that exist. The transmissions occur over a channel with characteristics described in Section 3.5. The performance of these transmissions are determined using a set of system metrics, described in Section 3.6. In the system, Quality-of-Service (QoS) is required for some traffic types. These requirements and also available QoS mechanisms are described in Section 3.7. Finally, Section 3.8 presents a definition of a time slot, which is crucial for QoS in this system.

3.1 Access Points

In the network, one or several AP(s) exist. These AP(s) serve as gateway(s) to external networks, such as the Internet. Each node in the network is at one point in time associated with one AP only. The APs are fixed in location and have no other intelligence apart from serving as gateways; they cannot decide upon transmissions of other nodes, e.g. schedule transmissions as in a centrally controlled network. The AP functions as any other node; they must contend for access as every other node. There is no priority given to the links closer to the AP. No other fixed infrastructure exist, i.e. no fixed relay nodes.

3.2 Nodes

The nodes in the network are equipped with the IEEE 802.11 DCF [7] as basic MAC functionality. Apart from the DCF, each node is capable of using IEEE 802.11e standardized EDCA [12]. The nodes are at a point of time associated with one AP only. A set of nodes generate traffic out of the network and are called source nodes. Their destination node in this network is the AP with which the source nodes are associated. Some source nodes will have a direct communication to the AP and some are over multiple hops, via other nodes, thus each node is able to relay transmission for other users. Routes over multiple hops are called chains or paths.

After a node has finished its session, it switches off. A switched off node is no longer available for relaying other nodes' transmissions, hence paths can break.

Further, all nodes are assumed to have non-drifting internal clocks.

3.3 Communication protocols

A routing protocol that finds a suitable route between the source and destination node (over multiple hops when needed) is available. When a path breaks, the routing protocol can initiate a route repair locally or from the source node.

The intermediate nodes do not differentiate between a relay packet or its own packets. All packets are stored in a queue (or several when EDCA is used, see below) and treated in a first come first serve method.

The basic medium access protocol used by default by each node is the DCF. The DCF functionality is described in more detail in Section 2.1.3. Further, all nodes are capable of using the standardized priority mechanism EDCA as one mechanism for QoS. The EDCA is more described in Section 4.3.1. All packets belonging to the same priority class is within each node handled the same; intermediate nodes do not differentiate between its own packets and packets generated at other source nodes.

3.4 Traffic model

The source nodes generate both real-time and best effort (background) traffic. Both these traffic types cause flows from a source node to its destination node (AP). A real-time flow consists of periodic transmissions of fixed sized packet, i.e. real-time packet. For one flow, the period and packet size is fixed. Different flows can have different sized packets and periods. Each flow i transmits a packet of fixed size b_i every p_i second during a session time of t_i seconds.

A data flow consists of bursty traffic, here modeled by letting the packets be generated at a data source node according to a Poisson process. This means that each data flow j generates packets with size b_j with exponential inter-arrival times δ_j, also here during t_j seconds. The probability density function for an exponential arrival time t is defined according to:

$$f_r(t, \lambda) = \lambda \cdot e^{-\lambda \cdot t}, t \geq 0 \tag{3.1}$$

where $\lambda > 0$ is the parameter of the distribution called the *rate parameter*.

The traffic is only between cells; intra-cell traffic is assumed non-existing. All communication is therefore to the AP, which forwards messages out of the cell to another network or cell.

3.5 Channel model

The channel is defined as the radio interface between two communicating nodes. The channel is assumed symmetric, hence has the same characteristics in both directions between the two communicating nodes. There are three different power levels defined for a channel: 1. Communication level, 2. Sensitivity level and 3. Interference level. The communication level is the received power level required for a node to be able to receive and decode a packet properly. The sensitivity level is the power level a node must sense to identify that the channel is busy. The interference level is the level at which the interference is too high for a successful reception. If the received power level is above the communication level, but the ratio between the received power level and interference power level is lower than a threshold, the reception is unsuccessful and a packet considered lost. This threshold is the Signal-to-Interference (and Noise) (SINR) value. These packet errors can also be modeled analytically with an error probability, described more below.

All transmissions suffer from path losses, which determines the actual signal strength at a node. Many simultaneous transmissions in the network can sum up to a value corresponding to the sensitivity- or interference level, measured at one node. The communication level is always measured from one transmission only.

The received signal strength of any transmission can be predicted using a path loss model. Two models are used here: 1. Free space model or 2. Two-ray ground reflection model. The free space propagation model assumes the ideal propagation condition where there is only one clear line-of-sight path between the receiver and transmitter. The received signal power can be calculated according to:

$$P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2 L} \tag{3.2}$$

where P_t is the transmitted signal power, G_t and G_r are the antenna gains of the transmitter and the receiver respectively. L is the system loss and λ is the wavelength [1].

The two-ray ground reflection model considers both the direct path and also a ground reflection path. It is shown that this model gives more accurate prediction at a long distance than the free space model [31]. The received power at distance d is given by:

$$P_r(d) = \frac{P_t G_t G_r {h_t}^2 {h_r}^2}{d^4 L} \tag{3.3}$$

where h_t and h_r are the heights of the transmit and receive antennas respectively.

As mentioned above, a packet is considered erroneous, or lost, when the SINR is below a certain threshold. However, packet losses can also be modeled analytically where the packet error model of the channel follows the 802.11 physical layer standard. For a channel with 1 Mbps data rate, Differential Binary Phase Shift Keying (DBPSK) is used as modulation which has an bit error probability of $BER = \frac{1}{2}e^{-\frac{E_b}{N_0}}$, where $\frac{E_b}{N_0}$ is the SINR. Using the equations for received signal strength above, BER and from this, Packet Error Rates (PER) can be analyzed. For other error models for the physical layer of 802.11, see [7, 32].

3.6 System metrics

The metrics for the evaluation is delay, variation of delay (jitter), throughput and packet error rates. These are described in more detail in the following paragraphs.

Delay This thesis considers delay as the end-to-end delay of one packet; from source node to its destination. This means the time between of when a packet is received at the MAC layer at the source node from higher layers, until the whole packet is received at the MAC layer at the destination node and it is ready to be sent up in the protocol stack towards the application layer.

Jitter Jitter is the statistical variation of the packet inter arrival time. It has been defined in RFC 1889 [33] as the mean deviation of the packet spacing change between the sender and the receiver. In this thesis, however, the Probability Density Function (PDF), or histograms, and the Cumulative Distribution Function (CDF) of packet delays are used. These will give the probability that a packet arrives with a certain delay or the probability that a packet has a delay that is lower than a certain time. From the PDF and CDF, it is possible to distinguish how the packet delay varies, and is sufficient for the purposes at investigations here performed.

[1] L, G_t and G_r are commonly set to 1 in simulations

Throughput Throughput is defined as bits per second received at the application layer of the end-user(s) in the network. The real-time and background traffic throughput is considered separately. First, real-time throughput is defined. Assume $n = N$ real-time flows exist, each generating traffic according to parameters defined in Section 3.4: One packet with b_n bits is transmitted every p_n second. Each flow is active for t_n time. This generates a system throughput Thr in bits per second as:

$$Thr = \sum_{n=1}^{n=N} \frac{b_n \cdot \frac{t_n}{p_n}}{t_n}$$

Similar, throughput for data traffic can be calculated. Assume same parameters as above, but p_n is not used. Instead assume that k_n packets are successfully received for each flow during their individual session time t_n. Thus, the total network throughput for data traffic is

$$Thr = \sum_{n=1}^{n=N} \frac{b_n \cdot k_n}{t_n}$$

Packet error/loss rate The packet loss rate in the system is defined as the percentage of the transmitted packets that are lost. This covers packets that are lost due to collisions or too high interference, i.e. Signal-to-Interference-and-Noise Ratio (SINR). Packets dropped at a node due to route failure, queue overflow or any other losses before transmitted are not considered.

3.7 QoS in the system

The definition of Quality-of-Service is in this system divided into two parts, where the first part is *per flow* and the second one is for the *whole network*. QoS per flow means that all packets belonging to one flow arrive with minimum variation in transmission time at the destination of each flow. In fact, here totally *strict* QoS-guarantees is required, hence each real-time flow should have a totally non-varying packet delay. The variations of packet arrival is here investigated as described in the previous section, under jitter. Secondly, the throughput per flow should be constant for the whole time a flow is active. It is not possible to state an exact limit of the delay and minimum allowed throughput for a flow; obviously this depends on the application type. However, if number of hops between source and destination node, and the traffic model (packet size and period) are known, it should be possible to quite accurately predict the end-to-end delay and throughput.

The second aspect of QoS covers the whole system performance. First, the average system packet delay should be low. Different flows can have different end-to-end packet delays as different flows might be transmitted over paths with different number of hops. Also, different flows

can have different fixed sized packets. The average delay covers all successful packet arrivals at all destinations in the system. Secondly, system throughput should be as close to offered load as possible. The system throughput is defined in the previous section. As with per flow QoS, also for network QoS it is impossible to state exact allowed limits on average delay and system throughput – it depends on the applications (packet size and periods) available in the system.

In this system, as defined according to previous sections, there are some possible mechanisms for QoS-support that are available. First of all, as the IEEE 802.11 standard is only defined from LLC layer and below, it is possible to use a QoS-mechanism above these layers. For this, no changes to the 802.11 standard itself are needed. Secondly, the system supports usage of mechanism that require changes to the MAC layer and the physical layer. All nodes in the network are able to support alterations to the 802.11 protocol suite itself. The possible solutions are investigated in more detail in the next chapter.

3.8 Time slot definition

A time slot is defined as the transmission time or reception time of one packet only. A time slot can therefore be of varying sizes, depending on which type of packet that is transmitted (data, ACK,...) and also depending on which size the data packet has for different real-time flows. For one flow, the time slot belonging to a data packet is always of one size only since for one real-time flow, the data packet size is non-varying.

The transmission time of a packet is noted as S_t throughout this thesis. Obviously the transmission and reception slots of one packet have the same length. Further, also used is S_a, which is the transmission/reception time of an acknowledgment.

When an intermediate node reserved a time slot for a flow in the DCF with no RTS/CTS handshake, this really means that it reserves $2S_t$+$2S_a$, one S_t for receiving, one S_a for acknowledgment back to the transmitter, one S_t for forwarding the data packet and finally one S_a for receiving an acknowledgment.

Chapter 4

QoS support in 802.11-based networks

This chapter investigates the different options that exist for QoS, and evaluates which approach is the most suitable one for the QoS definition in Section 3.7 (strictly reliable and predictable transmissions). The mechanisms are classified according to the OSI protocol layer model as follows:

- Solutions on network layer or above. This covers reservation-based mechanisms, where periodic transmission instances are reserved for a flow, and QoS-routing mechanism, where the goal is to find a path between source and destination nodes where the requirements for the actual application have the largest possibility to be met.
- Solutions in the 802.11 standard itself. This is further classified in two categories:
  - Solutions on physical layer. This basically covers rate adaptation mechanisms, where the rate can be adopted according to different criterias, e.g. transmission status (successful or unsuccessful).
  - Solutions on MAC layer. The DCF can be modified to include either service differentiations, where different types of application flows have different priorities, or reservation mechanisms, where resources are allocated for a flow. The latter is classified according to the domain in which they operate; frequency or time can be reserved to achieve QoS.

Using this classification, the advantages and drawbacks of each mechanism are here discussed. First, mechanisms above the actual IEEE 802.11 standard are presented; routing, network layer and above in Section 4.1. Then the modifications to the 802.11 standard are presented. This covers the physical layer in Section 4.2 and MAC (LLC) layer in Section 4.3. This

chapter is concluded with a suggestion of which type of mechanism that is the most suitable one for QoS as it is defined in this thesis (Section 3.7).

4.1 Network layer and above

The first QoS approaches for wireless networks came from wired technology and was modified for the characteristics of wireless networks; a common approach was to introduce some QoS support above the MAC layer (see Section 2.3). One idea is QoS routing for mobile ad hoc networks. Mainly, the idea is to find appropriate routes for QoS demanding applications by "routing around" congestion or nodes which cannot fulfill a certain QoS requirement [34, 35, 36] . Typically, well-known ad hoc routing protocols (see Appendix B) are modified to serve as a QoS mechanism. The CEDAR protocol [34] can recognize unicast routes that are able to support a certain amount of bandwidth for a flow. The protocol consists of three different key components: 1. *Core Extraction*, where some distributed nodes are elected to form a "core". These core nodes maintain local topology information of all nodes and perform all route calculations for nodes in their direct neighborhood. 2. *Link State Propagation*, where link state information is propagated by the core nodes throughout the network. Information about links which have large amount of bandwidth available is propagated far away in the network whereas information about links which has low or no bandwidth available is kept local. 3. *Route Computation*, which is the last component where the actual route is computed using information retrieved from the link state propagation. First a core path between any source and destination pair is found, which gives the *direction*. The source then uses this direction to find the node that is furthest away and defines a partial route to this node. The source node transmits to this intermediate node. Upon reception, this intermediate node will function as a source for the next interaction and finds a new partial route in the direction of the core path. This repeats until the destination is reached. Using only information about links where the actual QoS requirements can be met (see 2.), the most optimum path can be found.

With a QoS routing approach it is possible to provide routes that are more likely than others to satisfy a bandwidth demand. But, as also the authors of state [34], it is not possible with such an approach to guarantee any bandwidth; ad hoc networks are highly dynamic and transmissions are susceptible to interference and hidden/exposed terminals (presented in Section 2.1.3), which prevents any guarantees.

The other options are resource reservations and service differentiation on network/IP layer (or above), which are well-known from wired network technology. But, as described in Section 2.3, mechanisms such as DiffServ or IntServ will not work well in a wireless ad hoc network. Some

mechanisms have been proposed that are based on the above mentioned mechanisms, but in addition consider aspects specific for wireless ad hoc networks. In [37, 38, 39], mapping of the QoS parameters used in DiffServ (TOS field in IP header) or RSVP (required bandwidth reservation on application layer) to the MAC layer is investigated. Integrating DiffServ-defined priorities into the MAC layer is possible, but an appropriate service differentiation mechanism on the MAC layer is required to which the priorities of DiffServ can be mapped. For RSVP signaling it is more difficult – a cross layer design is needed from the MAC layer up to the application layer. This is a very complex procedure and limits the number of possible usable protocols in the network.

Some modifications of the DiffServ, IntServ (RSVP) for better support in wireless networks have been suggested such as the *INSIGNIA* protocol [40] or the *ASAP* [41] protocol. The INSIGNIA protocol include additional fields for QoS requests in the IP header. Any node can request a predefined bandwidth, either a MIN or a MAX level. At each hop of the path between the requesting source node and its destination, a bandwidth is reserved according to the level given in the IP header at the source node. If a node cannot grant the MAX value, it decreases the requested bandwidth to the MIN value and sends the request on. If a node cannot accept the MIN value, the request is denied. All other nodes that already have reserved the MAX amount for this flow change the reservation to the minimum level accepted through out the whole chain.

The Adaptive reServation And Pre-allocation protocol (ASAP) [41] uses the basic functionality of INSIGNIA but offers soft state reservations and not only hard state. It is a two phase set-up of reservation; a Soft Reservation (SR) is first made and this bandwidth can be used by other traffic but cannot be reserved by another flow. The SR is followed by a Hard Reservation (HR) state; no other traffic is allowed anymore. The source transmits a SR message and each node creates a flow entry in a reservation table according to what bandwidth is requested, if the request can be fulfilled. The request can be any value within the range of MIN and MAX. If any node participating in multi-hop transmission path cannot fulfill the request, it updates the soft bandwidth field of the SR message with the amount of bandwidth it can reserve. It then forwards the message. This is repeated at each intermediate node. At the final receiver a HR message is generated with bandwidth reservation information equal to the soft bandwidth field of the latest SR. All nodes along the path adopt their entries in their reservation tables according to this value. When the source receives the HR message it can start the transmission with the reserved bandwidth.

The SR messages are in-band signaling, inserted in the IP header as the INSIGNIA protocol. The HR messages are out-of-band signaling, thus a separate message is sent along path from destination towards the source node. The HR message is also used whenever there are changes in the network as to the available resources; they are transmitted in the opposite direction of the

flow to update the reservation. After set-up, SRs are also periodically inserted in the IP header to collect QoS information. Using this information, the source node can also scale down its transmission rate if the allocated resources can no longer be maintained.

One evident drawback with the INSIGNIA and ASAP protocol is that if a request cannot be met, it makes little sense to reserve anything at all; reception at an end-user might not be with fully acceptable quality. Another option would be to block the request, tell the source node to try later (busy tone like approaches) and use the resources for another flow, requesting less bandwidth. The most critical drawback, though, is that the CSMA/CA-based MAC scheme in 802.11-based networks cannot guarantee a reservation made on higher layers; the actual access mechanism is still contention-based. This leads to that the DCF medium access mechanism itself should be modified.

4.2 802.11 Physical layer – link adaptation

The physical layer of all 802.11 standards supports several transmission rates, one example is 802.11b, which can transmit data with rates 1, 2, 5.5 and 11 Mbps (see Section 2.1.2). These different possible transmission rates can be used adaptive, i.e. alternating the transmission rate when needed to keep the throughput for a flow over a certain threshold. Also, the mechanisms strive to give the maximum possible throughput for a flow, meanwhile keeping the packet error rate below a certain threshold; typically a higher transmission rate leads to higher probability of packet errors. One mechanism is *PER-prediction* [42], by which the transmission rate is alternated according to a predicted Packet Error Rate (PER). When a source node predicts that the PER is increasing, the transmission rate is decreased. Another mechanism introduced by Pavon et al. in [43] uses Received Signal Strength (RSS) along with the number of retransmissions, in order to predict the channel and receiver conditions. A node adapts the transmission rate based on received signal strength from packets sent by an AP. All nodes have fixed transmission power, hence the condition at the node can be predicted. Any node chooses a transmission rate for a packet based upon this RSS level, frame size and number of retransmission for this packet. All nodes start with the slowest transmission rate and whenever the RSS is over one of 12 different thresholds, it switches to the transmission rate associated with that threshold. Upon an unsuccessful transmission, a node decreases the rate for the retransmission. A similar mechanism is Code Adapts To Enhance Reliability (CATER) [44], which uses Bit Error Rate (BER) for rate estimates. Another option is to use the transmission acknowledgments [45] and alter the transmission rate according to the success or failure of a packet transmission – upon successful packet transmissions, i.e. reception of acknowledgments, the transmission rate can be increased.

Predicting the channel state, and a direct reaction to a received signal strength or acknowl-

edgment requires that the receiver informs the transmitter about the channel state. Using the acknowledgment is a simple option since this packet is anyhow transmitted. Also, since this requires no new medium access for a separate channel state message, signaling overhead is minimized.

Generally, all these mechanism can increase the throughput performance in a single hop environment. Once the packet is transmitted, a higher rate results in a higher throughput. But, there are some drawbacks. First, the possible distances between communication pairs for a successful transmission can decrease with increasing transmission rate, hence decreasing the possible network coverage. Also, a higher modulation rate is more sensitive to packet errors.

A major drawback is that rate adaptation is not effective in multi-hop communications; each node of a multi-hop path performs a randomized medium access, which reduces the gain from higher transmission rate. The end-to-end delay can still be largely varying. Further, the header of a data packet must be transmitted using the basic data rate of 1 Mbps [7]. Real-time packets are typically rather small, hence the gain of using a higher transmission rate on the payload can be small. Also, when RTS/CTS is used, these messages must also be transmitted with this basic rate. Therefore, the link adaptation is not the best solution for strict QoS guarantees, or for applications sensitive to delay variations. For this, a modification of the MAC layer is needed.

4.3 802.11 MAC layer – extending the DCF

There are two basic methods to extend the DCF: 1. Service differentiation, which gives priority to some type of applications over others and 2. Reservation mechanism, which allocates bandwidth for a transmission flow. These two methods are described and analyzed in the following two sections.

4.3.1 Service differentiation – priority mechanisms

A simple method to implement QoS support on the MAC layer is to include service differentiation in the DCF. By defining the channel access parameters of the DCF differently for different traffic types, priority separation of the traffic types is achieved. The parameters that can be altered are: CW size, backoff algorithm and interframe space (see Section 2.1.3). Tuning these parameters enables high priority packets to have higher probability of winning access to the channel than low priority packets. This is true for both the contention within a node and for the contention between two nodes. Typically, within a node, the different priority classes have different virtual queues and a packet from a high priority queue is handled before a packet in a queue with lower priority. Contention against packets at other nodes are differentiated using

Access Category	voice	video	best effort	background	DCF
CW_{min}	7	15	31	31	31
CW_{max}	15	31	1023	1023	1023
AIFSN	2	2	3	7	2

Table 4.1: Back off and AIFSN values for EDCA and DCF.

shorter backoff and interframe space, which results in that the high priority packet has shorter waiting time than the lower prioritized packet, thus the probability of access is larger.

The IEEE 802.11 standard E, Enhanced Distributed Channel Access (EDCA) [12] is the most known priority mechanism. The EDCA is based on the DCF and the medium access is performed with sensing and back off as described in Section 2.1.3. Different is that EDCA separates between four Access Categories (AC) by alternating both the interframe space and the CW size: Voice, video, best effort, and background traffic. Voice has the highest and background traffic the lowest priority. Each category has its own queue within a node. Packets from the queue with the highest priority are transmitted first. When this queue is empty, packets from the second highest priority queue are transmitted, and so on. Furthermore, each category has a different contention window and backoff times. Table 4.1 shows the minimum and maximum values of the contention window (CW_{min}, CW_{max}). Voice packets have the lowest backoff interval (from 7 to 15); best effort and background packets have the same backoff interval as DCF (from 31 to 1023).

The time period that a node has to sense a channel to be idle before it is allowed to transmit, is called *Arbitrary Inter Frame Space* (AIFS) in EDCA. It is determined according to

$$AIFS = AIFSN \cdot aSlotTime + SIFS$$

where the number $AIFSN$ is defined by the access category (Table 4.1), $SIFS$ and $aSlotTime$ is defined in Section 2.1.3. By assigning packets with high priority a small AIFSN, the waiting time before transmission becomes smaller.

Other mechanisms are Distributed Fair Scheduling (DFS) [46] and Distributed Weighted Fair Queue (DWFQ) [47]. DFS differentiates the backoff length according to the packet size and traffic class. Generally a larger packet has longer backoff time, but the length of a packet is weighted with a factor according to the traffic class. DWFQ alternates the CW size according to the actual throughput and expected (requested) throughput. If the requested throughput is higher than the actual one, the CW size is decreased. Such an adaptive approach is also used in QPART [48], which alters the CW and waiting time according to the throughput, as described

above, and also the delay; if the requested packet delay is lower than the actual delay, the CW and waiting times are decreased. Other algorithms that deal with scheduling and priority-based medium access are described in [49, 50, 51, 52, 53]. Typically, they all have the same basic idea, but differ in the way a packet is weighted.

A priority/scheduling mechanism does not need any explicit signaling, the service differentiation is handled separately within each node. Although they have the potential to perform better than the DCF and give some real-time QoS-requiring applications better support than the best effort class, the medium access is still contention-based; the uncertainties of access can still result in largely varying packet delays and throughput. This is especially true when network load increases, or when single hop communication is extended to a multi-hop one [13, 54, 55]. Thus, another mechanism is needed, which removes the medium access scheme itself and allows nodes to keep a non varying transmission schedule; allocation of resources for flows are required.

4.3.2 Reservation mechanisms

For strict QoS guarantees that are independent of the surrounding traffic load, a reservation-based MAC layer is needed. In CSMA/CA-based networks, there are generally two options for resource reservation: reservation of frequency or time. The following sections describe these two options in more detail.

Reservation of frequency channel

One common approach to enable reservation of frequency channels for a certain transmission is to use a dedicated signaling channel to distribute channels for transmission of data. Typically, all nodes listen to this signaling channel and when a node wants to initiate a transmission it agrees with the intended recipient upon another frequency channel to use for the actual data transmission, hereby allocating it for their transmission only [56, 57, 58]. Common is to use the RTS/CTS exchange for the agreement on the dedicated signaling channel [59]. Although not a reservation mechanism, another option is to dedicate one frequency for busy tones. Any node that is busy with reception/transmission simultaneously transmits a busy tone on the dedicated channel, hereby informing nodes in the neighborhood not to disturb [60, 61].

The IEEE 802.11 standard a and b have multiple frequency channels available [7], but unfortunately, most user equipment today are not fully equipped to implement such methods. Typically they have only one half-duplex transceiver, hence a node can only transmit or receive at one time and only at one frequency. Using multiple frequencies is therefore today not possible.

Reservation of time

The second possibility is to reserve a certain amount of time for a future packet transmission. This can be done either centrally or distributed. With the central approach, one node (AP) controls the whole network and assigns time to other nodes for their respective transmissions. For this, all nodes must be synchronized and pre-defined time slots, i.e. when the transmission should start and when it must end, must exist. If not, one node cannot control all other nodes time slots. With such *global synchronization* the transmissions in the network follow a fixed schedule; typically a fixed air frame divided into predefined time slots [62, 63, 64, 65]. Such approaches are pure TDMA systems, like in existing cellular networks. The obvious advantage is their simplicity to allocate resources for all the participants. But, it is difficult for the central controlling node to distribute time slot information over multiple hops.

In a distributed network with no central control, it is difficult to maintain global synchronization; maintaining a fixed structure of time slots which start and end at globally defined time instances is tricky. One possible method to achieve the global synchronization is to use Global Positioning Systems (GPS). This is used in the Distributed Packet Reservation Multiple Access [66] (D-PRMA) protocol. Each node has GPS equipment that enables a fixed time divided air interface with frames and time slots. These time slots are then divided in smaller time slots, so called mini slots. To gain access to the whole slot, a station has to win the contention in the first mini slot. It can then use the rest of the time slot for its transmission. The same time slot is also reserved for this node's transmission and can be used in all following frames. This is only true for transmissions that consist of real-time packets. A transmitter with data can only transmit in the time slot where it won the access contention. If the first mini slot does not lead to a winner, the contention continues in the next mini slot. If a node wins contention in one of the mini slots *except* the first one, it can only start its transmission in the next frame. The contention is performed in the same way as the DCF with RTS and CTS messages.

Although the authors state that due to a growing and cost decreasing development of GPS, slotted-channel-based MAC schemes become available and most interesting for mobile ad hoc networks, it is not a realistic assumptions that all nodes carry it. Although the approach is interesting and very promising a reservation mechanism cannot rely on globally synchronized network. A reservation mechanism must function in distributed networks where the participants themselves manage and schedule transmissions; it must function distributed. For clarification, also here nodes must be synchronized in that sense that no clocks are drifting, but time slots are not defined globally. Thus, the suggestion of this thesis is that a distributed time slot reservation protocol is the best option for QoS-support in DCF-based networks.

4.4 Summary

This chapter has described different existing mechanism for QoS support and also analyzed which type of approach is the best one for strict QoS requirements. First of all, a QoS mechanism for wireless 802.11-based networks should be implemented in the 802.11 standard itself. Approaches above (network layer and above) are here not satisfying. A solution on the physical layer, e.g. link adaptation, can increase the throughput, but it cannot offer strict QoS guarantees; the transmission rate is enhanced, but the actual transmission instance is still uncertain as the medium access is still contention-based. Therefore, the MAC layer must be extended. There are two basic options to enhance the quality of a transmissions using the DCF: 1. Give the QoS-requiring transmission higher priority or 2. Allocate resources. A priority mechanism is still contention-based; no transmission instances are guaranteed. This results in that packet delay can still be varying and large. A reservation mechanism is based on the nodes ability to reserve, schedule and co-operate when time slots are reserved. Thus, one drawback with this approach is that inter-node signaling and resource management at each node are needed. However, the advantage is overwhelming: a reservation mechanism can offer strict QoS guarantees. If signaling load can be kept low, it is the most suitable solution. Thus, my suggestion for extending DCF based networks with QoS support is to use a distributed time slot reservation protocol.

Chapter 5

Distributed time slot reservations in 802.11-based networks

The previous chapter looked at the different possible solutions that exist for a general distributed wireless multi-hop network. The conclusion was that a distributed periodic time slot reservation protocol is the most suitable one for these networks. Although many such approaches exist, it is still a challenging research issue, especially in multi-hop networks [15, 54], still lacking is e.g. a full end-to-end aspect of the reservation set up and lack of mobility support.

The goal of this chapter is to define requirements for the design of a new reservation-based MAC protocol for 802.11-based networks. To complete the design in an effective way, some constraints to the general system description is needed. The *system under study* is defined in Section 5.1. Then, the requirements (or constraints) for a reservation protocol in this system under study are described in Section 5.2. These requirements are compared to some of the already existing mechanisms in Section 5.3, concluding with that no existing mechanism can cover them all.

5.1 System under study

Before discussing the requirements for the reservation protocol itself, some assumptions for further narrowing of the system is here presented. Although a solution for the whole universe would be nice, this is hardly possible. Thus, I have chosen to bound the system described in Chapter 3 for my design of the reservation protocol. The *system under study* consists of nodes and APs using a set of communication protocols, all described in Chapter 3. The first basic assumption is that the DCF is possible to modify; it is fully feasible to extend it with a QoS mechanism.

The number of APs and nodes are limited. Further, the area is bounded and it is not possible to communicate outside the border of it. The APs are not directly reached by all nodes; some nodes must communicate over multiple hops. The number of hops between source node and AP is limited [67]. Further, the communication ends at the AP. No considerations are taken to what happens with packets after this node. Thus, packet deliveries are measured after their arrival at the AP. Further, interference range is assumed not much larger than twice the communication range and the bandwidth is limited; no over-provisioning is available.

Then, one constraint is included for mobility in the system. Mobility is modeled with nodes switching on and off in one location. No nodes move around while still communicating.

The channel is assumed not to suffer from any fading.

Further constraints are for the traffic: The periodic real-time transmission has no silent periods. Typically, voice traffic, e.g. Voice-over-IP, consists of talk bursts where the communicating pair of nodes take turn in transmitting period packets. This is not considered here – A source node of a real-time application transmits one packet at each periodic interval until the session is over. As mentioned in Section 3.4, the periods that different application flows use can be different, but each flow has only one period. Here, this is further limited; the periods used are always from a certain set such that a common divider of all periods is available.

At last, no considerations are taken to other types of traffic; although background (or best effort) traffic is existing in the network, the major goal here is to give real-time traffic sufficient support.

5.2 Reservation requirements

This section describes some requirements for a distributed reservation mechanism in the system under study, as defined above. These include limited bandwidth, interference, mobility, multi-hop paths, multiple paths crossing in a node, reservation reject and release and traffic requirements. They are described in that order in the following subsections.

5.2.1 Limited bandwidth

When two nodes communicate, the CSMA/CA-based access scheme force all nodes in the neighborhood of these two nodes to abstain transmission during the whole communication. Even if another pair, intending only to communicate with each other, can communicate, they are not allowed to; the broadcasting nature of the wireless channel prevents such simultaneous transmissions. Therefore, all extra signaling that is meant for two nodes only, agreeing upon a time slot reservation will affect all nodes in the direct neighborhood, preventing others from trans-

missions. If the signaling load is high, it might result in low system throughput. Thus, signaling load must be kept as low.

5.2.2 Interference

After a successful time slot reservation is set up, the contention-based medium access is removed; a node transmits directly during a reserved time slot without sensing the channel first. If another node that is in such location that it can cause severe interference would transmit at any time during this time slot, there would be a collision. The region for which such interference can occur around a receiver is called the neighborhood. Consider Figure 5.1 where one reserved path between source node A and destination node D and another path from source node E to destination node H exist. Node F could possibly interfere with both paths, when transmitting. Thus, a suitable spread of reservation information to nodes in the local environment is required.

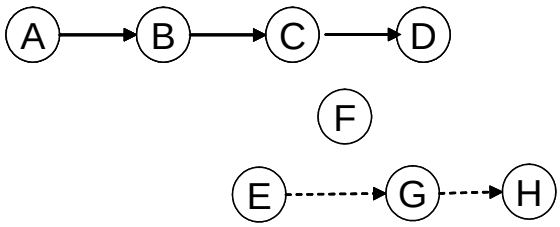


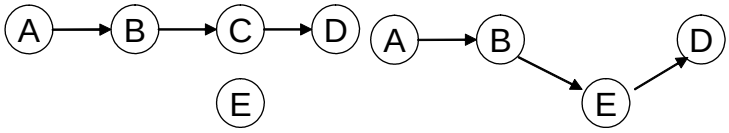
Figure 5.1: Node F must be aware of two reserved chains and avoid transmissions during all reserved time slots.

5.2.3 Mobility

Nodes that are mobile can introduce additional problems for a reservation protocol; a reserved path can break as any node participating in the chain moves out, or switches off. Illustrated in Figure 5.2 is a case where a path is broken due to a node leaving the network (node C). If possible, a new node takes over (node E) and forms a new path between the source and destination. Such functionality must be included in the reservation protocol.

5.2.4 Multi-hop wireless paths

Different from a single-hop environment, the multi-hop communication paths involve nodes that do not actually participate in the data exchange (on application level). The more nodes involved in a communication path, the more problematic is a successful reservation. Some parts of a chain might have more communications occurring in the direct neighborhood than others, hence it is



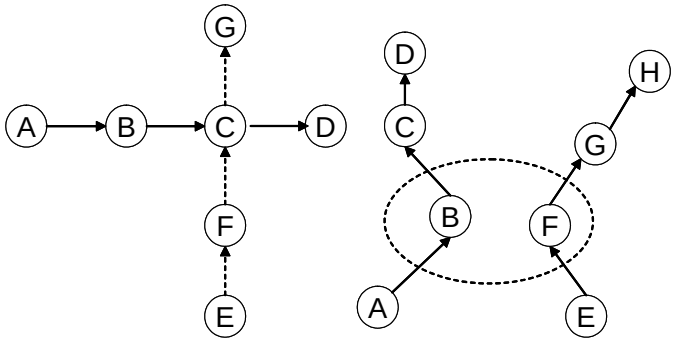
(a) original (b) new

Figure 5.2: The original path with node C (a), and the path when node C has left and E taken over (b).

possible that the reservation is only successful at some hops of the path. Thus, the reservation protocol must consider the whole path at the set-up.

5.2.5 Multiple crossing reservations

When multiple reservations exist, these might cross in one node, or be set up in the direct neighborhood of each other. Thus, the reservation protocol must include a suitable scheduling mechanism that enables successful reservations, managed by each node locally. Figure 5.3 illustrates two examples where such functionality is required. In (a), two paths cross in a node C, whereas



(a) (b)

Figure 5.3: Reservations (a) crossing and (b) in the neighborhood.

in (b) the paths are only in the neighborhood, but node F and B can interfere with each others transmissions. This is an especially tricky functionality as the scheduling mechanism is required to support transmissions with different periods and packet sizes.

5.2.6 Reservation reject and release

If a reservation cannot be set up it must be rejected. This includes all cases where a suitable time slot cannot be found, e.g. at initial set-up and repair of broken reservations. For cases where

packet losses occur due to overlapping time slots, or unexpected interference, a reservation must also be released (failure handling).

5.2.7 Traffic requirements

Different applications can have different transmission characteristics. For instance, voice and video might require period reservations with different time slot sizes and periods. Another example is Voice-over-IP, which uses a CODEC to generate periodic fixed sized packets at the application level. This packet size and period can vary with different CODEC [68, 69]. To not limit the new protocol to one CODEC, the reservation protocol should be able to handle different sized time slots with several possible periods. Further, non-reserved traffic must be able to co-exist with reserved traffic in the distributed network.

5.3 Related work

Several reservation-based extensions to the DCF exist. This section describes some of these mechanisms and compare the mechanisms with the requirements described in the previous section, and most importantly describe why these mechanisms are not enough.

QPART The QoS Protocol for Ad hoc Real-time Traffic (QPART) protocol is presented in [48]. This protocol defines two different possible QoS requirements for an application flow: either a minimum throughput or maximum end-to-end packet delay. QPART is a mix of a priority- and reservation-based protocol; applications that require some QoS for their transmissions have priority over other applications that are not as sensitive to either the delay or throughput and at the same time, new requests are blocked or some flows are released if the required QoS cannot be met within a node. If a flow must be released, the flow that has been established last is released first; the longer a flow has been active in the network, the higher is its priority.

The tunable parameter within each node is the contention window size, which is adapted constantly with the actual load in the network; The requested delay/throughput is compared with the actual achieved one.

QPART consists of two components: the QoS scheduler and the QoS manager. The QoS scheduler differentiates between the different flows in a node where each flow has its own queue on the network layer. The QoS Manager handles the release and rejection of flows and also monitoring the level of congestion. This information is used in both the QoS manager and the contention window adaptor in the QoS aware scheduler. Therefore, the flows are controlled on both network and MAC level, hence a cross-layer designed protocol.

This approach is successful in the sense that it is possible to fast adapt to the traffic of the network. Yang and Kravets show that they can keep the delay under a certain level and the throughput over a level. However, the packet delay is varying – jitter could be a problem. The QPART protocol is not a full reservation protocol in such a sense that a periodic time slot is reserved; it is still contention-based and it fulfills most of the requirements described in the previous section. The only one missing is maintenance of broken paths. But, since it cannot guarantee a strict packet delay, this approach is not suitable here.

MACA/PR In [70], Gerla and Lin describe a periodical packet reservation technique, Multiple Access Collision Avoidance with Piggyback Reservations (MACA/PR). This is an asynchronous network solution based on DCF.

MACA/PR is equipped with a bandwidth reservation technique and a QoS routing protocol. The QoS routing algorithm is responsible of finding the most optimum existing and available path it can before a periodic reservation for packet transmissions is performed. More detailed information about the QoS routing algorithm can be found in [70]. After the route is found, the first packet in a stream initiates the reservation of a transmission window for following packets with a predefined global period. The reservation is set up using the RTS/CTS exchange. Before transmission of the first packet, or rather the RTS packet, the source node checks in a reservation table for a free periodic slot. In the first available one, it imitates the RTS/CTS exchange; it starts listening to the channel. If the channel is idle, it transmits its RTS to the receiver. The receiver answers with a CTS, which, when reaching the source node, completes the reservation phase. The source node will from now on use the same slot for all its transmissions, until it has no more to send. If the receiving node is an intermediate node, after it has received the packet, it performs the same steps to reserve a periodic time slot as the source node did.

To inform surrounding nodes about the reservation, all data packets and acknowledgment carry reservation information in their headers. Using this information, nodes that overhear the transmissions make entries in their respective reservation tables. Whenever a new node enters the network, it listens to the channel for a whole period to hear all ongoing transmissions. Further, to make completely sure that the new node will not interfere with an ongoing, reserved real-time transmission, all reservation tables in the systems are exchanged, initiated by the new node. MACA/PR as a MAC protocol sets up a real-time connection only over a single hop. Each hop must be reserved separately.

Gerla and Lin first compare their scheme with a totally asynchronous network, where not unexpectedly MACA/PR performs very well (throughput and delay is compared). Then the scheme is compared with a totally synchronized TDM network, where MACA/PR is outperformed. However, the conclusion is that MACA/PR has a good overall performance and is a

good, cost effective compromise between the two above mentioned single-hop networks.

Compared to the constraints in Section 5.2, MACA/PR is not a good alternative. The first constraints of limited bandwidth is not considered; the exchange of reservation tables demands high signaling load. Mobility and reservation repair is not considered. The traffic requirements, different periods and time slots, is not fulfilled. Also, the flow reservation is done on a single-hop basis and a packet is not guaranteed a reservation at all hops between the source and destination.

Blackburst Blackburst [71, 72] is a technique by Sobrinho and Krishnakumar, which minimizes and bounds delay of real time traffic. First of all, best effort traffic is still transmitted using the DCF. A real-time station that wants to transmit starts its access procedure by first jamming the channel with a energy burst, so called Blackburst. The length of the jamming burst is determined by the time that the station has waited for its transmission. After transmission of the Blackburst, the station listens to the medium to see if some other station is sending a longer Blackburst. This implies that this station has waited longer and therefore that it transmits first. Whenever a packet is transmitted, the node schedules the transmission for the next packet. This is done using a predefined period, which is the same for all nodes and all packets.

Their investigation shows that the transmission of Blackburst before transmitting is an efficient way of allocating the network periodically. However, it produces a lot of unnecessary signaling overhead. In [72], a method for increasing the efficiency and decreasing overhead of the Blackburst algorithm is described. A node that won the Blackburst contention and is transmitting include an invitation for the node that has waited longest after the transmitting one. Therefore, stations only transmit a Blackburst for their first packet. Then they use, as described above, the same fixed interval for all packet transmissions. This works well when only real-time traffic is present. But, if a node leaves an empty “time slot” exists in the transmission order. If a data transmission with normal DCF occurs in this time slot, the scheme no longer works as all following transmissions must be moved forward in time. The authors have investigated the behavior of the Blackburst mechanism under the assumption that all stations hear each other.

Using the enhancement, the constraint of limited bandwidth is fulfilled, as long as the scheme works and no node leaves the network. Missing is the interference aspect; they have assumed that all nodes hear each other, hence the black burst prevent collisions. This restriction is not acceptable. Further, no special considerations is done to multi-hop paths, the traffic requirements or path repair. In the basic black burst mechanism, each packet of a flow contend for access with transmissions of black burst. This means that the delay can be very varying and high. This mechanism, and other similar ones, is not the optimum one.

DBRP, DBASE Papers [73, 74] describe MAC protocols Distributed Bandwidth Reservation Protocol (DBRP) and Distributed Bandwidth Allocation/Sharing/Extension Protocol (DBASE), that use broadcast messages, so called Reservation Frames, to allocate resources. The first voice station that enters the network gets the responsibility of broadcasting these RFs. Each station keeps a reservation table with information about ongoing reserved transmissions. A voice station that wants to transmit, first listens to the channel during a predefined time period. If this new station hears a RF, it knows that there are other active real-time nodes in the network. It must then wait and listen until the periodic cycle is over (until next RF) in order not to disturb any ongoing reserved transmissions. It also listens to hear if there is an empty slot. If so, and if it has something to transmit, it starts an access procedure for this time slot using traditional DCF-based back off (in case other nodes wants this time slot as well) and RTS/CTS, where the RTS/CTS set up the reservation between the transmitter and the intended receiver. An allocated time slot that is unused for a certain time period is considered vacant and others can use it for transmissions. Best effort data is transmitted via IEEE 802.11 DCF.

With this scheme a node reserves periodic time slots, i.e. it reserves for its whole flow. But, the reservation is on a hop-per-hop basis, no considerations are taken for the whole path. The most obvious problem is how to handle the broadcasting of RFs in a larger multi-hop network; it is difficult to assign which stations should forward the RF. If possible at all, the overhead is large and also, there will be some time synchronization problem – a node cannot receive an RF and forward it simultaneously. The traffic constraints in Section 5.2 is not fulfilled as only one period exist. Neither is repair of broken paths (mobility) hence this approach is not the best one in distributed wireless multi-hop networks.

DRRP In [75], the authors describe the Distributed Reservation Request Protocol (DRRP). This protocol uses the basic mechanism of IEEE 802.11e but further enhances the performance for QoS-demanding traffic with a reservation of future resources (IEEE 802.11e standardized transmission opportunity (TXOP)). The authors piggy-backs the request for a periodic reservation on a data packet as it is transmitted. They include information as to when the next instance is and how it repeats periodically. Further, the reservation request includes a priority. This priority is used when two transmissions ask to reserve the same time slot. The one with the highest priority wins. Surrounding nodes retrieves information regarding the transmission from the overheard data packet. They also include information in acknowledgments for some nodes that cannot overhear the transmitted data packet.

This protocol does not reserve end-to-end and there is no repair mechanism if a reservation is broken (mobility caused). The protocol do not consider protection in a wider range than the direct communication neighborhood. However, the reservation information is handled in

the optimum way – signaling overhead is minimized with the piggy back technique. Thus, the DRRP protocol is not the optimum one.

The above described protocol all lack some, or several of the requirements. Thus these are not satisfying approaches for the system considered here. Also other mechanisms [76, 77, 78, 79, 80, 81] that are not discussed more in detail here *all* lack some of the requirements described in the previous section. Thus, a new protocol is needed, which fulfills all the requirements and constraints in Section 5.2. This protocol is described in the following section.

5.4 Summary

This chapter has presented the requirements for a distributed reservation protocol in 802.11-based networks. These requirements were compared with some well known existing DCF-integrated reservation mechanisms. To minimize the probability that a path might not be fully reserved a reservation should be performed *end-to-end* before transmission of data begins. Existing mechanisms however, tend to reserve each hop separately – most common here is to use the RTS/CTS exchange. Further, the reservation should be performed for the *whole flow*, which minimizes the signaling load and guarantees a non-varying quality during the whole application transmission. Some existing approaches tend to reserve a transmission instance for an upcoming packet as one packet is transmitted.

Reservation information should be spread with minimum signaling load; using a piggy backing technique is one successful method. Some approaches use less efficient (signaling load) methods such as a complete exchange of a reservation table or transmissions of energy burst that indicate which node should transmit. Further, a wider range than the direct neighborhood is needed, at least as an extension or additional feature. This is not approached by any related reservation protocol.

What is largely missing is some support of mobility. In the system under study, mobility is modeled with node switching on/off, resulting in that reserved paths can break. For best user satisfaction, a broken reservation should be *repaired*, hence a user must not do anything to re-initiate the communication. Also, *failure handling* such as overlapping time slots are lacking. Most protocols assume a network where every node can hear each other.

The reservation protocol should be able to support different reservation requirements, i.e. *different periods* and *time slots*. The existing periodic reservation protocols do only have one pre-defined period and time slot size.

The rest of this thesis presents the new reservation-based MAC protocol that takes all the above mentioned aspects into account. This QoS enabling protocol is called *Distributed end-to-*

end Allocations of time slots for REal-time traffic (DARE) [82, 83, 84, 85, 86].

Chapter 6

DARE Basic functionality

This chapter describes first of all the concept of the new DARE protocol with a qualitative protocol description in Section 6.1. Then, the basic functionality of the DARE protocol is presented. This includes such functionality that one periodic time slot reservation can be set up over a multi-hop path, and that the reserved periodic transmissions are successful. First, all nodes need a reservation table, which is described in Section 6.2. Then, the set-up messages are defined and the mechanism itself is described in Section 6.3. Further, nodes surrounding the reserved chain must avoid transmission during the reserved time slots. The basic protection of the reserved chain is described in Section 6.4. After the reservation is set-up a periodic transmission schedule is generated. This transmission flow is described in Section 6.5. Nodes involved in the reservation can also participate in non-reserved transmissions, i.e. non-real-time, or background traffic. These transmissions follow the DCF medium access scheme. For the different traffic types to co-exist, some modifications to the DCF are needed. These are described in Section 6.6. Then, the difference in signaling load for the DARE and DCF is described in Section 6.7. At the end, a simple simulative comparison of DARE and DCF for one real-time transmission flow is described in Section 6.8.

6.1 Concept of DARE – qualitative description

The DARE protocol is a totally distributed access protocol. Nodes schedule time slots based on locally collected information. For this, each node have clocks which are non-drifting and are able to reserve periodic time slots for the real-time flows. A time slot is defined in Chapter 3 as the length of time a node needs for either transmitting or receiving a packet. In case of multi-hop communication, each node of the path will reserve time slots for one real-time flow. Any relaying node therefore reserves minimum one time slot for receiving and one time slot for

transmitting, both with same size. The slot allocation is illustrated with an example where three nodes are part of a multi-hop chain with a reservation for packet transmission every p period. Figure 6.1 shows allocated slots for sending s and receiving r at nodes $n-1$, n and $n+1$. For simplicity, a reference point 0 when node $n-1$ transmits to node n is set.

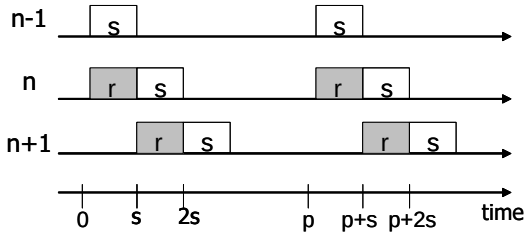


Figure 6.1: Reserved slots without shift.

Relay nodes, or hosts in the direct neighborhood of a reserved path that are not participating, abstain from transmission during the time slots where they could interfere if transmitting. This is, in contrast to wired networks, necessary as they could cause severe damage even if they do not intend to communicate with any node part of the reserved path.

Further, all relay nodes are able to reserve time slots for several flows. In case of conflicting reservations, a relay node can shift its own transmission. Consider again the example given above and assume further that node n has other obligations part of another flow, see Figure 6.2. The packet from node $n-1$ arrives at a) and a direct transmission would overlap with the already

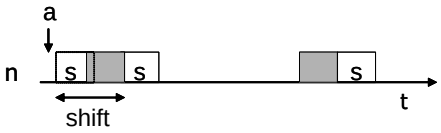


Figure 6.2: Shift at node n.

existing reservation (grey). Node n shifts the transmission until after the already reserved time slot. As a result, all packets for this flow are delayed with $shift$.

Typically, protection in the direct receiver neighborhood is not sufficient [87]. Thus, all nodes are able to support minimum up to a two hop protection; all participating nodes reserve time slots for receive or transmit at other nodes part of the same chain, when necessary. Similar, all non-participating nodes that are in the neighborhood are informed about possible slots they must abstain two hops back in the reserved path.

When a communication is finished, all nodes released their reservations for the real-time flow which is no longer in use. If a relay node leaves the network and this results in a broken path,

all reservations no longer in use on this path are also released. A new reservation is established along the new path. To detect if a path is broken, all transmissions of reserved data packets must be acknowledged.

Since bandwidth is limited, the reservation protocol should be simple with as low signaling overhead as possible; the reservation signaling is as much as possible integrated in the DCF standard.

6.2 Reservation table

Each node has a reservation table where it keeps all its active reserved time slots for transmissions and receptions. This is used to schedule the different reserved time slots such that the correct action is taken during them, e.g. the correct packet is transmitted during the time slot reserved for the flow it belongs to. A reservation entry can be of three different types: Transmitting, receiving and avoiding. During a transmit slot, a node transmits a packet. During a receive slot, a node receives a packet. During an avoid slot, a node abstains from transmissions so a nearby reservation is not disturbed.

Each reservation entry can have two different status, *preliminary* and *fixed*. When a request is initiated within a node it makes an entry with preliminary status. When the request has been accepted throughout the whole path and a node can acknowledge that the reservation is ok, it changes the entry to fixed status.

The preliminary status does not give any strict guarantees of a reserved transmission instance, it is used if a node receives a second request before it has fixed the first one. The node then considers the preliminary entry with higher priority. Further information kept in the reservation table are: Time slot length, period, receiving and transmitting nodes' MAC addresses (due to that a node can overhear several nodes that are participating in the reservation), and source and destination addresses.

6.3 Reservation set-up

DARE protocol sets up a periodic time slot reservation end-to-end before the transmission of data begins. In a sense, the RTS/CTS exchange of DCF is extended to cover the whole multihop path. This section describes the set-up of the periodic time slot reservation. Section 6.3.1 describes the set-up messages followed by Section 6.3.2, which describes the set-up function.

6.3.1 Set-up messages

Before explaining how the set-up works in detail, the set-up messages are introduced. These messages are:

Request-To-Reserve (RTR) : Generated by source to initiate the reservation set-up.

Clear-To-Reserve (CTR) : Generated by final destination to confirm the reservation request.

Update-Transmit-Reservation (UTR) : Generated by any node upon receiving an RTR with a conflicting reservation request which the node itself cannot change. This is sent to a node that must make adjustments to a preliminary reserved time slot.

Figure 6.3 shows the frame structure of the RTR. The UTR and CTR have the same structure (but in the UTR, the fields have different purposes explained later on in this section). The RTR

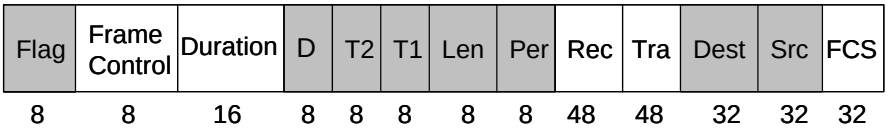


Figure 6.3: DARE RTR message format. Field sizes in bits.

is extended from the RTS and some fields are re-used in DARE. These are the ones needed for control information and are shown as white fields (described in Section 2.1.3). The additional information fields needed for the reservation set-up are shown in grey and contain the following information (from right):

Src Source node address of the application flow.

Dest Destination node address of the application flow.

Per Period of the reservation.

Len Length of the requested periodic reservation time slot.

T1 Relative time stamp for slots reserved at the nodes transmitting the RTR.

T2 Relative time stamp for slot reserved at node preceding the node transmitting the RTR.

D Delay introduced due to DCF.

Flag Indicator of which type of message it is. Value range: 0 to 6.

The **Src** and **Dest** address fields have several purposes. First of all, they are required to find the next hop of the path. Different from the RTS/CTS exchange of DCF, where the exchange occurs single-hop wise, the RTR is forwarded all the way to the final destination. Further, these fields are needed for all nodes participating in the reservation to identify to which flow a certain reserved time slot belongs. The **Per** and **Len** information fields contain the numerical values of the time slot and the period for the requested time slot. **T1** and **T2** contain information about slots reserved at preceding nodes of the chain. These are needed for two-hop protection issues. When a node receives an RTR, it uses T1 to identify the time slot reserved for reception at the node one hop back. Similar, it uses T2 for the time slot reserved for reception two hops back. This is more described in Section 6.4. The **D** field contains the extra delay that the DCF introduces at the transmission of the RTR, CTR and UTR, see Section 6.3.2. The **Flag** field is used to identify which type of packet is it. It has separate values for the RTR ($Flag = 2$), CTR ($Flag = 3$) and UTR (Flag range 4-6), and is also included in the data packets to separate real-time ($Flag = 1$) from non-real-time ($Flag = 0$) traffic.

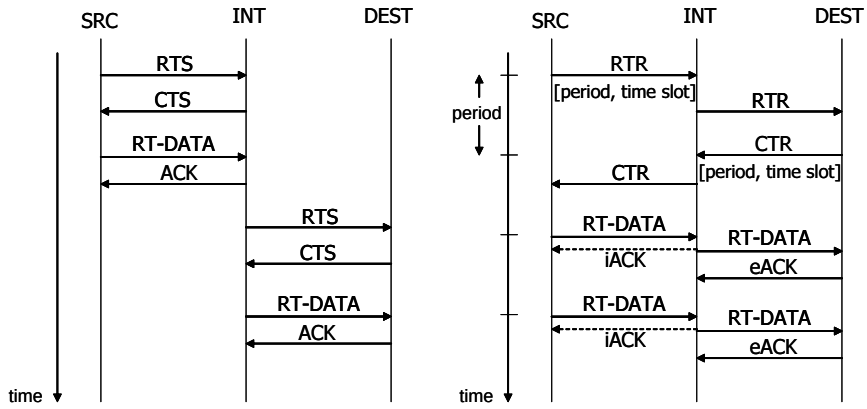
The CTR has the same information fields as the RTR and is not further explained here. Compared to the CTS of the DCF, which carries less information than the RTS, here all fields are needed in both the RTR and the CTR. The source and destination addresses are needed to identify to which reservation the CTR belongs, and to find the next hop of the path. All time slot information fields are also required as a time slot might have changed from the initial requested values, see Chapter 10. Also, this information is used by nodes overhearing the CTR (or the RTR) to abstain transmission during the reserved instances.

The UTR also contains the same information fields as the RTR/CTR, but some have a different purpose. The UTR is used at a node that cannot accept a reservation for a receive slot; it transmits this message back in the chain to the node that must alter its transmission slot so no conflicting time slots are reserved. **Src** and **Dest** are the same. The time information fields **T1** and **T2** contains new suggested transmission slots for the receiver of the UTR. The **Flag** can have three different values, depending on which node that must make alterations to its reservation. This is described in more detail in Section 10.7.

6.3.2 Set-up mechanism

The RTR/CTR end-to-end exchange implies that no hop is reserved before the whole path has agreed to the reservation requirements. This is different from the DCF, where the RTS/CTS exchange only “reserves” a transmission instance per hop wise. This is illustrated in Figure 6.4, which shows the set-up of DCF (left) and DARE (right).

The reservation set-up starts as the source node receives the first real-time data packet of



(a) DCF (b) DARE

Figure 6.4: Basic set-up concept for DCF (left) and DARE with the periodic time slots shown on the time axis (right).

a flow at the MAC layer. The source node generates an RTR and fills the fields with the parameters described in Section 6.3.1. The flag has the value 2 for an RTR. All parameters are known to the MAC layer such as the final destination address, next hop MAC address and requested period and time slot length. The source node checks for a suitable transmission time slot, which is not conflicting with other entries in its reservation table and transmits the RTR accordingly. This RTR is transmitted using the DCF. Therefore, directly before the transmission, the additional delay introduced by carrier sensing is included in the packet so next node knows the actual transmission instance for the upcoming data packets (the *D* field of the RTR packet). If the receiving node can fulfill the reservation request, it makes an entry for a receive slot in its reservation table with preliminary status (see Section 6.2). If it is non-conflicting with another reserved time slot, it forwards the RTR. If it is conflicting, the transmission of the RTR follows the same procedure as at the source node; the node finds another suitable transmission instance and delays the transmission of the RTR. This is described more in Section 10.7. This forwarding is comparable to the transmission of the CTS in the RTS/CTS exchange of DCF; the node forwards the RTR after waiting SIFS. Thus, the RTR is directly forwarded and has higher priority that any other non-reserved transmission (after channel is sensed idle, a node must wait DIFS, which is longer than SIFS).

When a node hears the next node of the chain forwarding the RTR, its own RTR transmission is acknowledged. If a transmission of an RTR is unsuccessful, it is retransmitted according to the retransmission procedure of the DCF: If an RTR is not acknowledged after a certain time, the transmission is considered unsuccessful and retransmitted.

If all nodes of the chain can fulfill the requested reservation, the RTR reaches the final destination and all nodes have a preliminary entry in their reservation tables. The final destination generates a CTR, which has the Flag value 3. This message is forwarded via the same intermediate nodes back to the source. The routes between source and destination must be symmetric, i.e. the routing protocol must provide such symmetric routes. To assure this, mechanisms like symmetric route pinning [88] can be used. However, this is an issue for the routing protocol; the DARE protocol assumes that this is provided. The arrival of the CTR at each node changes the status of the reservation table entry to *fixed*. Also the CTR is transmitted using DCF. The RTR/CTR exchange leads to reserved receive and transmit slots as described in Figure 6.5.

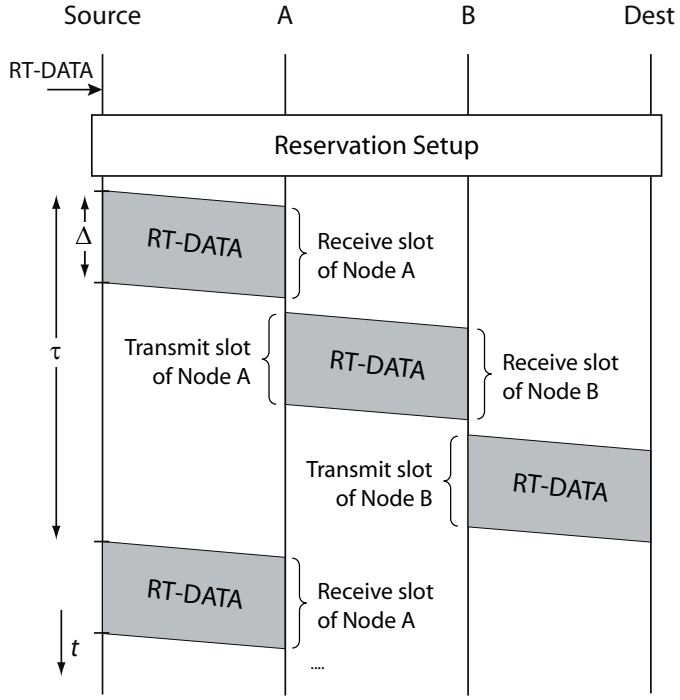


Figure 6.5: The end-to-end RTR message generates preliminary reserved receive and transmit slots.

If a node cannot fulfill the request, the RTR is not forwarded. This means that preceding nodes will not receive a CTR and the reservation status will never be fixed. Although the preliminary status does not mean that a time slot is fully reserved, it can affect other reservation requests. Therefore, the reservation must be released; a time out function is used. When a node sends the RTR to the next node in the chain, it starts an RTR timer. If no CTR is received before this times out, the reservation is released. The correct time out value for the RTR is challenging as any RTR is transmitted with the DCF and it is difficult to predict the number of hops of a

path. This is a parameter that can be changed according to traffic requirements (network planning). Another option would be to dynamically adopt the time out value according to the result of the transmission. The value can also be determined after the time it takes for the RTR to be acknowledged. Assume that a node overhears the next node forwarding after time t. If t is large, this would indicate that the medium is rather congested and a longer time out value is needed. In this thesis, this value is set to 12 periods, assuming paths is not much longer than 6 hops and that no node must wait longer than 1 period for its transmission.

The full set-up mechanism with scheduling of several reservations is described and illustrated with state diagrams in Section 10.7.

6.4 Basic reservation protection

After a reservation is set up, it must be protected from other nodes in the neighborhood, whose transmissions could interfere and cause collision-based packet losses. There are two basic methods how this can be done: Explicitly inform nodes with a separate reservation information message or use existing messages that are anyhow transmitted and piggy back the information (implicit information). To explicitly inform nodes e.g. with an exchange of reservation tables or reservation information is difficult. A node must know when it enters into an area where it needs to be re-informed and initiate this information exchange. Another drawback is the high signaling load that the separate medium access with DCF causes for the transmissions of these messages. An implicit reservation information dissemination requires no additional medium access as the messages used are anyhow transmitted. Since all nodes are equipped with basic DCF functionality, they always listens to the channel and can retrieve information from messages they can decode. Therefore, the DARE protocol uses piggy backing for dissemination of reservation information.

The RTR, CTR and UTR all have reservation information in their headers, used for set-up of a reservation. This information can easily be used by the surrounding nodes that overhear these messages and anyhow decode them, to retrieve sufficient information. Unfortunately, these messages are only transmitted once for each flow; if a node enters the area of the reservation after it is set-up, it must retrieve information elsewhere. Therefore, other packets used in the reserved flow must piggy-back reservation information as well. An obvious choice here is the real-time packets, which can distribute reservation information periodically, thus all real-time data packets carry reservation information in their headers. Also, an explicit acknowledgment used at the last hop keeps reservation information (see Chapter 7). The information in the data packets is shown in Section 6.5.

As stated in the requirements and constraints in Chapter 5, there is a need in a CSMA/CA-

based network to spread information to a wider extent than in the direct neighborhood of two communicating nodes. In [89] it is shown that spreading the reservation in two hops around a receiver is sufficient protection. Two hop protection is with DARE divided into two parts: 1. Within the reserved chain and 2. To nodes surrounding a chain. Consider Figure 6.6. Protection part 1 is illustrated with node F, which is within the communication range of the reservation (circle around node C) and avoids reserved slots up to two hops away (all nodes participating in the flow also avoid up to two hops away). Protection part 2 is illustrated with node G, which has no information but is two hops away from the reservation and could interfere.

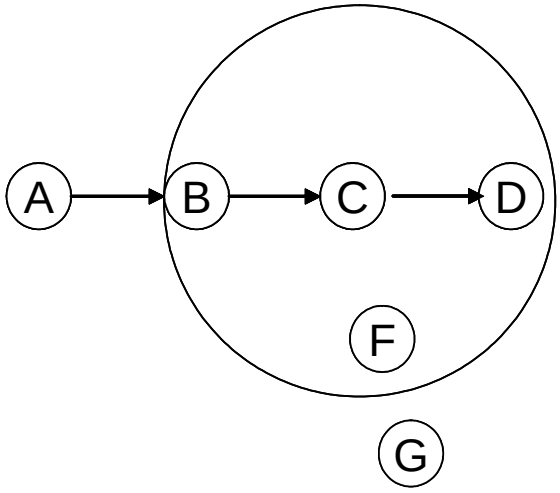


Figure 6.6: Two cases of protection: Node F avoids time slots up to two nodes away. Node G is two hops away from the reservation and could interfere but does not overhear reserved transmissions.

The second protection is an optional feature of DARE and more described in Chapter 8. Two hops within a chain is straight forward to implement. All participating nodes are receivers of set-up messages and data packets, which can easily be used to spread information about time slots reserved at nodes two-hops away. As described in Section 6.3.1, the set up messages contain two parameters **T1** and **T2**. T1 is the relative starting point of the receive slot at the node transmitting the RTR. This means, that any node that overhears or receives the RTR can distinguish not only its own receive (avoid) slot, but also the one reserved at the node transmitting the RTR. This value is sufficient information as both the time slot length and period is given in Len and Per fields and have not changed. Similar is true for the T2 field, with the difference that any receiving (overhearing) node can distinguish the receive slot that the node preceding the one actually transmitting has reserved. Possibly three preliminary reserved time slots entries in the reservation table can be an result when an intermediate node receives and RTR.

When forwarding an RTR, any node enters the information from the reservation table in the T1 and T2 fields. This guarantees that all nodes abstain reserved time slots at nodes up to two hops back in the reserved chain. Also, nodes that are not participating in the reservation but are in the direct neighborhood avoid all slots. The reservation must also be protected in the other direction. This is enabled by letting nodes learn from overhearing the next node in the chain forward the RTR and/or using the information in the CTR.

6.5 Transmission flow

After the source node has received the CTR, it starts the transmission of real-time data packets in the next upcoming reserved time slot. The data packets are transmitted directly in the reserved slot, with no contention-based medium access. This results in that the end-to-end delay of all packets belonging to one flow is the same; all variations that can be critical for real-time traffic is removed. Consider again Figure 6.4(b), which illustrates the transmission flow of a two hop path. The reserved transmission instances are shown on the time scale to the left. As can be seen, the source waits for the next reserved time slot after it has received the CTR. The acknowledgments shown in the figure (iACK and eACK) are described in Chapter 7.

DARE data packets that are lost are not retransmitted; a lost packet is typically less severe for the reception quality of a real-time application than a delayed one is. Also, another rationale behind this is that collisions are rare because the reservation information spreads to possibly interfering nodes “quickly enough” (see Section 6.4).

The data packet is shown in Figure 6.7. Many of the information fields are identical to those

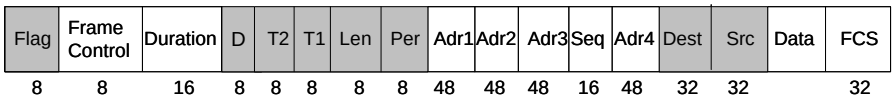


Figure 6.7: DARE data packet format. Field sizes in bits.

of the RTR, CTR and UTR with one exception: the D field is not in use since no additional DCF caused delay exist for reserved real-time data packet transmissions. The white fields of the DARE real-time data packets are identical to those of a DCF transmitted data packet, which are described in Section 2.1.3. The Flag has the value 1.

6.6 Modification to DCF for non-reserved traffic

Nodes participating in real-time reservations can also receive and transmit non-real-time traffic. Figure 6.8 describes a non-real-time traffic transmission using the DCF functionality in between

two reserved packet transmissions. The intermediate node must know how to separate and handle non-real-time traffic. Therefore, the structure of the DCF MAC frames must be slightly

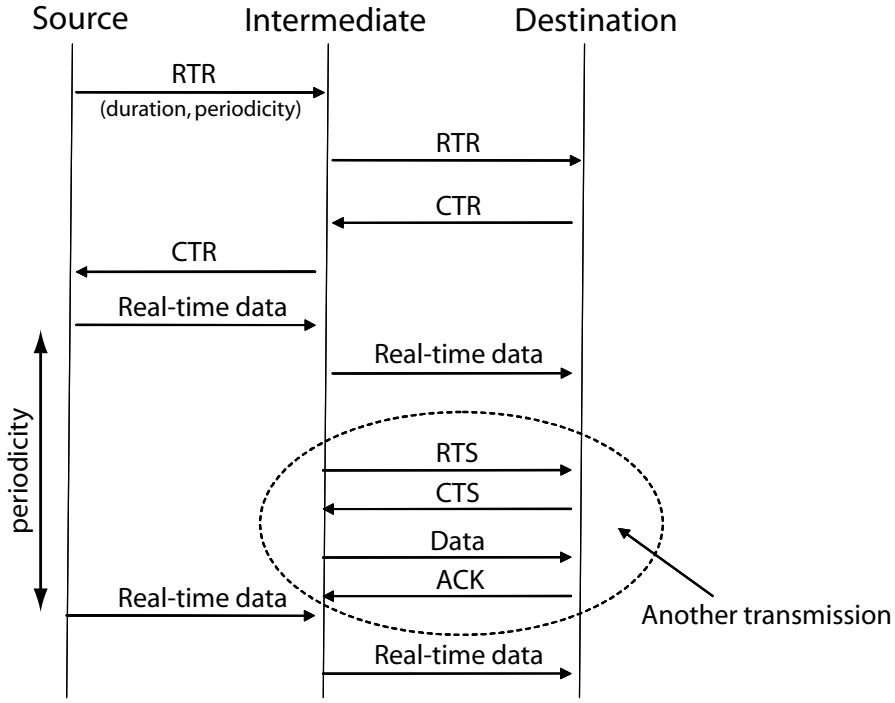


Figure 6.8: Non-real-time traffic transmission in between DARE reserved packet transmissions.

altered for non-real-time traffic, to achieve a similar appearance and common understanding between all nodes. All nodes must be able to separate between the different data frames that occur in the system.

First of all, the DARE non-real-time data frame is extended from its normal DCF appearance with the Flag field from the real-time data packet. This flag is set to 0 for non-real-time packets. With this flag, any node can distinguish which type of message that it has received/overheard. If the flag is 0 it knows that the MAC frame is a DCF type and can read it as that. If the flag is set to 1, the node knows that the MAC frame is DARE real-time type and can read it and separate all the different fields accordingly. Hereby, the DCF frame can keep its original format (accept for the Flag field) and signaling overhead is kept low.

Additionally, the acknowledgment of a DCF transmitted data packet must have this Flag field as well. This is because the last hop of a reserved path, a real-time packet is acknowledged explicitly, and these acknowledgments contain reservation information as the real-time data packet. Further, DARE offers an optional feature, where reservation information is spread even further around the reserved chain, described in Chapter 8. Here the RTS/CTS exchange

preceding a transmission of a non-real-time packet can be used for dissemination of reservation information. If this feature is requested, the RTS and CTSs of all non-real-time packet transmissions are modified to keep the same reservation information as the real-time data packet shown in Section 6.5.

6.7 Signaling overhead

This section compares the signaling load with DARE and DCF. The DARE reservation set up introduces some additional signaling compared with DCF. A packet transmitted using the RTS/CTS exchange, has lower signaling load than the RTR/CTR exchange. The overhead per set-up is:

$$hops \cdot (size(RTR) - size(RTS)) + hops \cdot (size(CTR) - size(CTR))$$

The RTR is 232 bits large and the RTS is 160 bits, resulting in a difference in size of 72 bits. The difference between the CTR and the CTS is larger as the CTS is only 112 bits – 120 bits. This results in a signaling overhead of 192 bits for each hop of the path. If the UTR is used for re-scheduling of some time slots, the signaling overhead is much larger; a corresponding message type is not used at all in the DCF. Each UTR results in minimum 232 extra bits plus the DCF transmission overhead. When the reservation is set up, each real-time data packet contains an additional 112 bits of information in the header. For a 1 Mbps channel, this results in approximately 0.1 ms of extra transmission time per hop.

However, when more than one packet is transmitted in a flow, *each* data packet transmitted with DCF is typically preceded by an RTS/CTS exchange for each hop, which results in minimum 160 bits of signaling overhead for every packet compared with DARE. Additionally, the contention-based access results in a much larger signaling overhead than that of the DARE protocol. Also, if the RTS/CTS exchange is not used, typically for small packets, the DIFS and SIFS (and probably some back off) alone are typically larger than the signaling introduced by each data packet, at each hop by DARE.

6.8 Basic features – Simulation study

This section describes a simple simulative investigation, where DARE is compared with the DCF for one chain with some background traffic. The goal is to get an idea of how the basic functionality compares to DCF and if the approach is at all useful. If DARE cannot even support one real-time transmission better than DCF, it will not be a good option in larger dynamic

network where real-time load is high.

6.8.1 Simulation model

The simulator used is NS-2, shortly described in Appendix A or in the manual [90], which also defines all other physical parameters not mentioned here.

One three-hop real-time transmission path to an AP, reserved by DARE, exists. The S_t matches a transmission of a 512 bytes packet using the basic data rate with 1 Mbps. This results in approximately 4.8 ms transmission time on the channel. The packet transmission is periodic with period 100 ms. No other traffic is going through the stations involved in this transmission. The reservation neither breaks nor is it released. Further, some non-real-time nodes exist transmitting packets also with 512 bytes size, with an exponentially distributed inter-arrival time directly to the AP. Parameters for the distribution (i.e. rate parameter) is chosen according to NS-2 default values and can be found in the manual. In model 1, one up to eight non-real-time stationary stations, and in model 2, one non-real-time station that moves in from a long distance at a speed of 2.5 m/s to the area with the reserved real-time transmission (Figure 6.9). For model

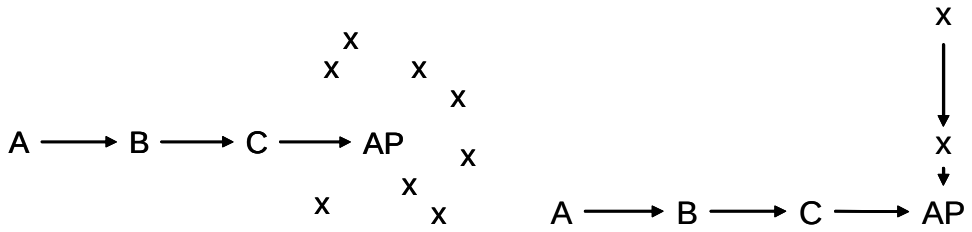


Figure 6.9: One three hop path from source A to the AP. In a) Model 1 with up to eight non-real-time stations (NRT) and in b) Model 2 with one non-real-time station, that moves in on reserved transmission with a speed of 2.5 m/s.

1, each non-real-time station has a load of 100 kbps. For model 2, the load of the non-real-time stations is varied, 100 kbps–800 kbps.

Each parameter combination is simulated 5 times, each time for 1000 seconds simulated time. The comparison case is DCF with RTS/CTS exchange. AODV is used as a routing protocol.

6.8.2 Simulation results

The DARE and DCF are compared with throughput, end-to-end packet delay and packet loss rates, described in Section 3.6. Also, the performance results of non-reserved traffic is shown.

The throughput results in Figure 6.10 and Figure 6.11 show that DARE offers the same real-time throughput, irrespective of the non-real-time traffic load. DCF achieves for real-time traffic under high loads in model 1 only one tenth of the DARE throughput. For the highest non-real-time load case, the system in model 1 is saturated. For model 2, the value is better. The high load case of model 2 is not saturated as only one node performs medium access according to the DCF.

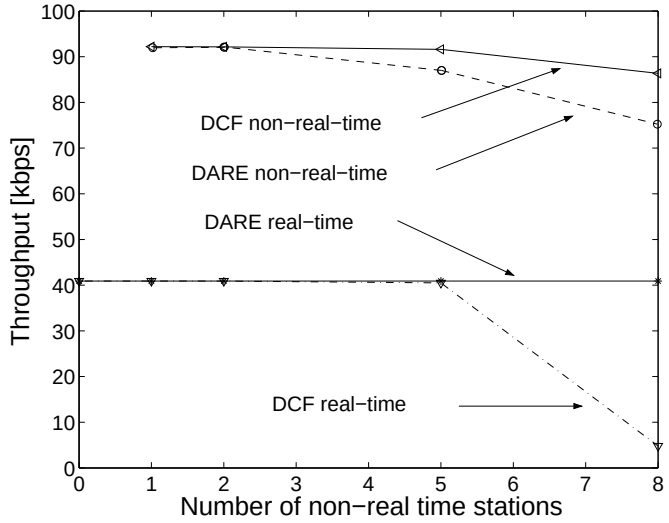


Figure 6.10: Throughput per node for real-time and non-real-time traffic, model 1.

For non-real-time traffic, DARE provides, at a larger number of nodes, a reduced throughput compared to standard DCF. There is a trade-off for the non-real-time traffic with reservation mechanisms for real-time traffic.

But more important for real-time traffic are the delay results. Figure 6.12 shows histograms of packet delays for both DCF and DARE from model 1 (for 5 non-real-time stations). The distribution of these packet delays is considerably narrower – constant 0.0145 seconds (three hops, each approximately 0.0048 seconds transmission time) for DARE than DCF, hence less jitter, which better supports real-time transmissions.

The average real-time delay for real-time packets is shown in Figure 6.13 and Figure 6.14. The average real-time packet delay is constant for DARE in both models, 0.0145 seconds. The average real-time delay for DCF is larger. In model 1, for 8 non-real-time stations, the average real-time delay is 1000 times larger than with DARE, 1.65 seconds. For model 2 and 800 kbps non-real-time load, the average real-time delay for DCF is 10 times larger. The difference between these two models is the number of access procedures, which in model 1 introduces a larger packet delay. This is also described earlier for the throughput results.

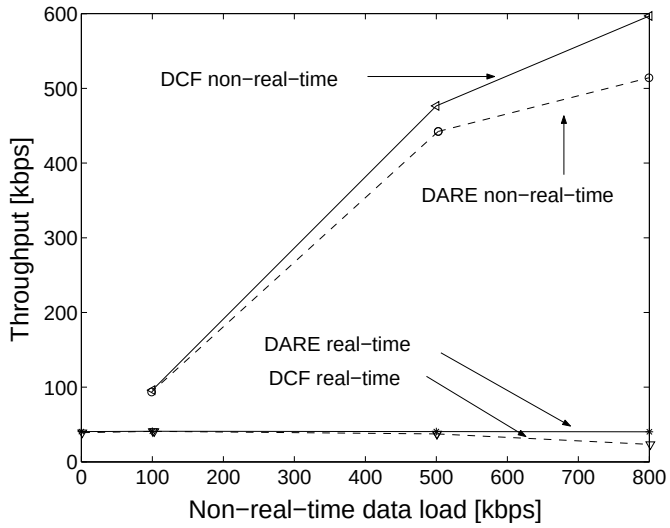


Figure 6.11: Throughput per node for real-time and non-real-time traffic, model 2.

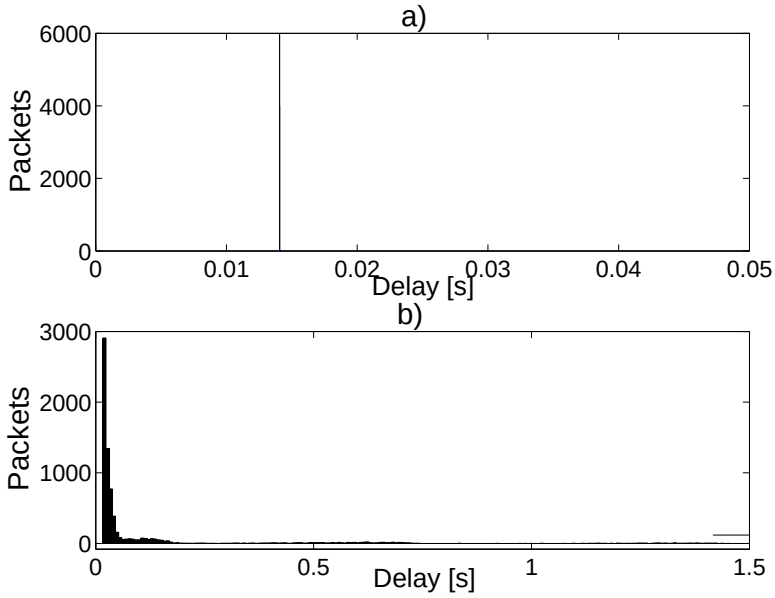


Figure 6.12: Real-time packet time delay distribution for model 1, 5 non-real-time stations in a) DARE and b) DCF(note the different scale of the x and y axis).

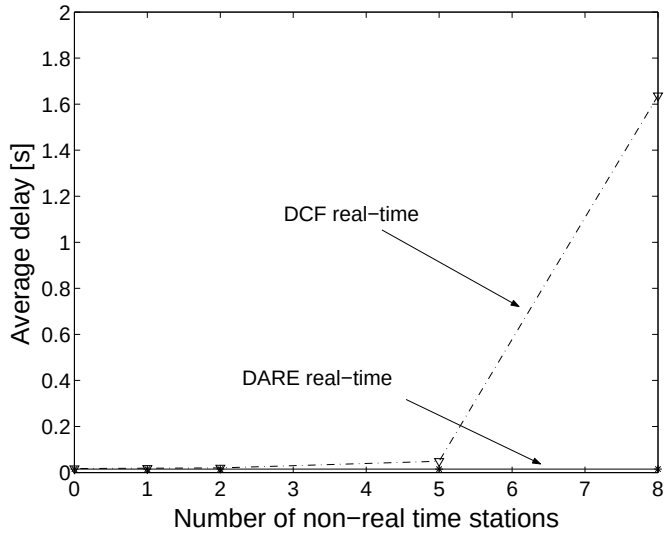


Figure 6.13: Average real-time packet time delay vs. non-real-time station load in model 1.

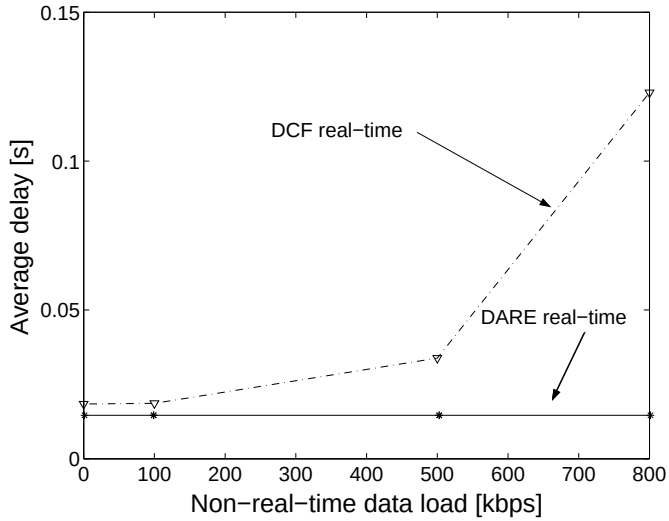


Figure 6.14: Average real-time packet time delay vs. number of non-real-time stations in model 2.

The delay results have to be considered together with the packet loss rates, as one main claim was that oblivious nodes can be informed efficiently about existing reservations. The packet loss rates for model 1 in Figure 6.15 show that the real-time packet loss rate using the DARE is indeed 0% for all numbers of non-real-time stations. It is higher for DCF, which has more collisions and MAC time-outs that generate packet losses. For the 8 non-real-time station scenario, the real-time packet loss rate for the DCF is approximately 11%.

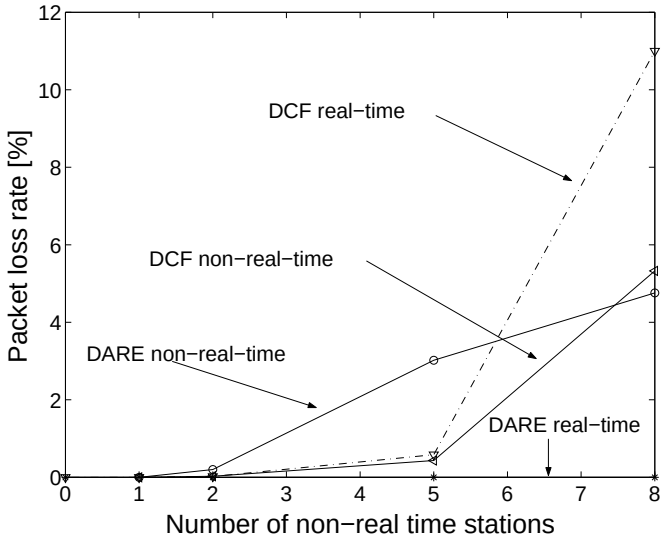


Figure 6.15: MAC packet loss rate vs. number of non-real-time nodes in model 1.

For model 2, the packet loss rate can be seen in Figure 6.16. The packet loss rate for DARE is 0% for both non-real-time and real-time transmissions. For the DCF, this rate is for real-time traffic up to 2%. This figure shows the loss rates after the non-real-time station has overheard a real-time packet and set its timer for avoiding the reservation. Before this happens, some real-time packet losses occur, which is up to 1.5% for the maximum non-real-time load.

6.9 Summary

This chapter has described the basic functionality of the DARE protocol, which enables a periodic time slot reservation to be set up over a multi-hop path. Using end-to-end set-up messages RTR/CTR, the source requests a reservation using the RTR at all nodes up to the destination. The reservation is confirmed with the CTR, traveling the same way back to the source. Further, surrounding nodes that are not participating in the reservation learn about it by overhearing the DATA or ACK packets. The signaling overhead is low, in fact the DCF using the RTS/CTS

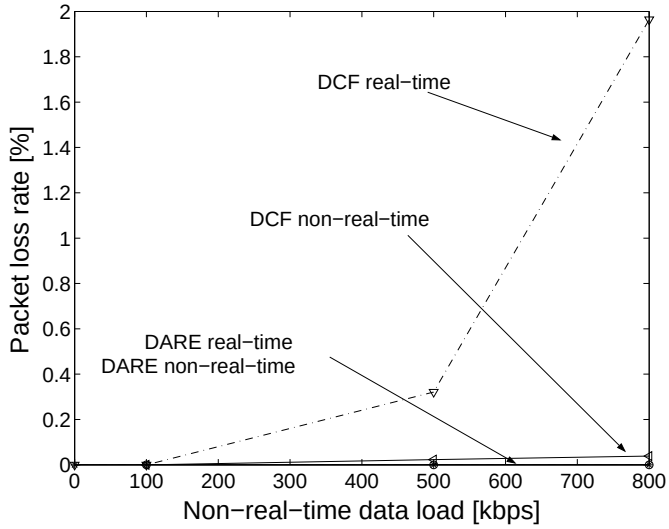


Figure 6.16: MAC packet loss rate vs. non-real-time load for model 2.

handshake has higher signaling load than the DARE protocol for each real-time data packet. A simulation-based study of one real-time flow shows that the DARE protocol serves its purpose: The end-to-end delay is low and stable and the real-time throughput is higher than that of DCF. This study has shown that the basic functions of the DARE protocol is a good foundation for a reservation-based QoS-enabling approach. In the following chapters, the more advanced functions of the DARE protocol are described.

Chapter 7

Reservation maintenance

When a session between source and destination is finished, the reserved time slots are no longer in use. Clearly, such unused allocated bandwidth should be released for other active sessions and transmissions. Hence, any reservation must be acknowledged by all involved nodes as active/inactive. If the network experiences topology changes, e.g. due to nodes switching on and off or move, the transmission path between source and destination may break. Such breaks must be repaired. If a node can no longer reach its subsequent node in the path, its data link layer will indicate the link break to the network layer. While the transmission route itself is then repaired by the routing protocol, i.e., an alternative route is found, also a distributed repair procedure for the reservations is needed. For this, nodes need to detect a break of the path, repair the reservation in a distributed manner and release outdated reservations. The reservation acknowledgment is discussed and described in Section 7.1. This feature is then used to detect a broken path and initiate a reservation repair, described in Section 7.2, and to release old/unused reservations, described in Section 7.3

7.1 Reservation acknowledgment

The basic requirement to initiate a path repair is that a node must somehow notice that the link to the subsequent node in the path is broken. When a session is over, all nodes involved in a reservation for this flow must acknowledge that the session is over. Therefore, the reservation must be acknowledged. One method is to let nodes transmit periodic alive messages letting all nodes in the direct neighborhood know that the reservation is still active [63]. However, since real-time packets are transmitted periodically, it is simpler to use these periodic transmissions for alive purpose. Also, this results in less signaling as no new message needs to be transmitted. So, if the periodic packet transmission/reception is acknowledged at each hop the nodes know

that the reservation is active – also the nodes in the direct neighborhood that are avoiding the reservation can by periodically overhearing a packet transmission acknowledge the reservation as active.

For the packet transmission acknowledgment there are several options. One option is to use *explicit acknowledgments* (eACKs) that notify a node whether its message did or did not reach the next hop. Each time a node receives real-time traffic in the corresponding time slot, it returns an eACK to the preceding node. This is used in the DCF, and is also the most common approach to acknowledge a transmission. The eACK could be included in the allocated time slot or follow the standard DCF procedure. As each hop is acknowledged, the signaling overhead increases (even if cumulative eACKs are used).

Another option is to use *negative acknowledgments* (nACKs), which are sent by nodes that do not receive any data in the allocated time slots [91]. The advantage of this approach is its lower signaling overhead. The major drawback is that the information must reach the node preceding the "hole", which calls for higher transmission power, if possible at all.

A more elegant solution is provided by *implicit acknowledgments* (iACKs) [92]. If a node A has sent real-time traffic to a node B, node A can overhear node B's transmission to its successor node C in the next time slot, thus can be ascertain that its transmission has been received by node B. There is no signaling overhead for this approach. However, this solution may be unsuccessful if power control is used, e.g., if node B uses such a small power that node A cannot overhear the transmission. Furthermore, as the final destination node does not forward any message, the last hop of the path cannot be implicitly acknowledged.

In conclusion, as long as there is no power control, the best option seems to be a combination of iACKs and eACK: For each intermediate hop, up to the last hop, the transmission is acknowledged by overhearing the subsequent node forwarding the packet; the channel is assumed bidirectional, hence has the same characteristics in both directions between two nodes (see Chapter 3). In the last hop, an eACK is used. This eACK on the last hop also takes the function of informing potential interferers located adjacent to the destination node; information regarding the time slot duration and periodicity is included. This is shown in Figure 7.1 where a real-time packet (RT-DATA) is acknowledged by overhearing at each hop, except the last where an explicit acknowledgment is used.

7.2 Repair of broken reservation path

When an intermediate node of a reserved transmissions path leaves the network a hole in the path results. The node preceding this hole will by not receiving acknowledgment for its packet transmission notice that the path is broken. First, the route itself must be repaired. Depending

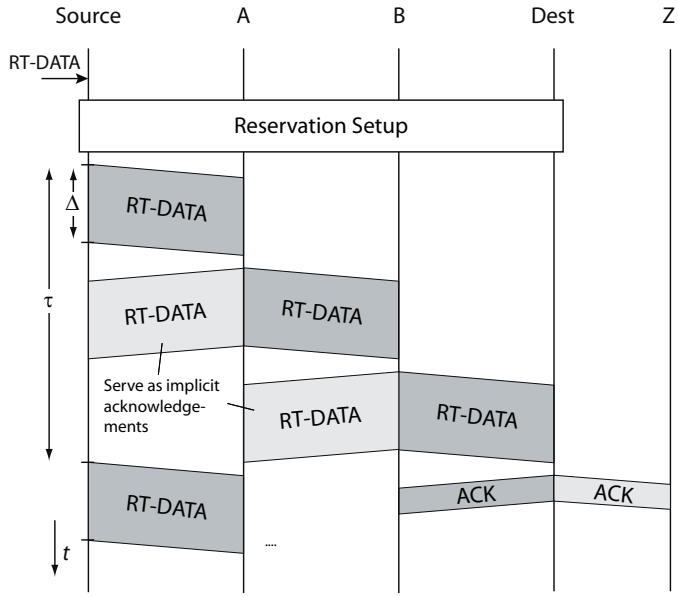


Figure 7.1: The transmission of real-time data packets (RT-DATA) is acknowledged implicitly and explicitly.

on the routing protocol this can be done either locally or source-initiated. When a new route is established, the reservation can be repaired or a new set-up along the new path. To get an idea of whether or not it makes sense to repair a reservation, or if the route repair results in too many lost/delayed packets, first both local and source-initiated route repair is investigated and described in Section 7.2.1. Depending on these results, the options for reservation repair is described in Section 7.2.2.

7.2.1 Route repair

Upon classifying the path as broken, a node can initiate a route repair mechanism. Typically, a responsible routing protocol can either repair the path, or imitate a totally new path from source to destination as long as there are other nodes in the neighborhood or network such that possible new paths exist. The procedure starts with that routing messages are exchanged in the network, which leads to finding a new route. Some available routing protocols for distributed wireless ad hoc networks are described in Appendix B. If the route repair is time consuming, a reservation repair might not be suitable; a totally new reservation set-up might be more efficient. Here, a simulation-based investigation of both local and source initiated route repair gives an idea of how the route update procedure impacts the packet transmissions of a flow. Using NS-2 [90],

the deterministic scenario shown in Figure 7.2 is simulated. Initially, there is a real-time path

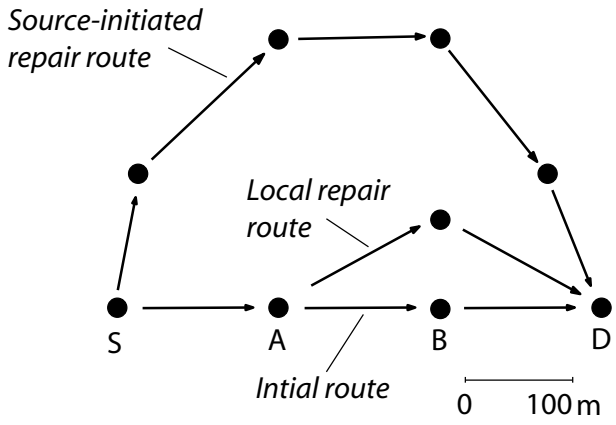


Figure 7.2: Simulation scenario with a local or source-initiated repair.

from node S via node A and node B to node D. Every $\Delta = 0.1$ s, node S sends a packet of size $s = 512$ bytes. An on-demand routing protocol, such as AODV [93], is assumed employed in the ad hoc network. Thus, both local and source-initiated repair is possible.

First, a local route repair is forced; after 2.1 seconds node B leaves the network. The node in the direct neighborhood of node B takes over. Then, a source initiated route repair is simulated; after 2.1 seconds node A leaves the network and a totally new path from source node S to destination node D is found. Figure 7.3 shows the sequence of the packets received by D. Upon

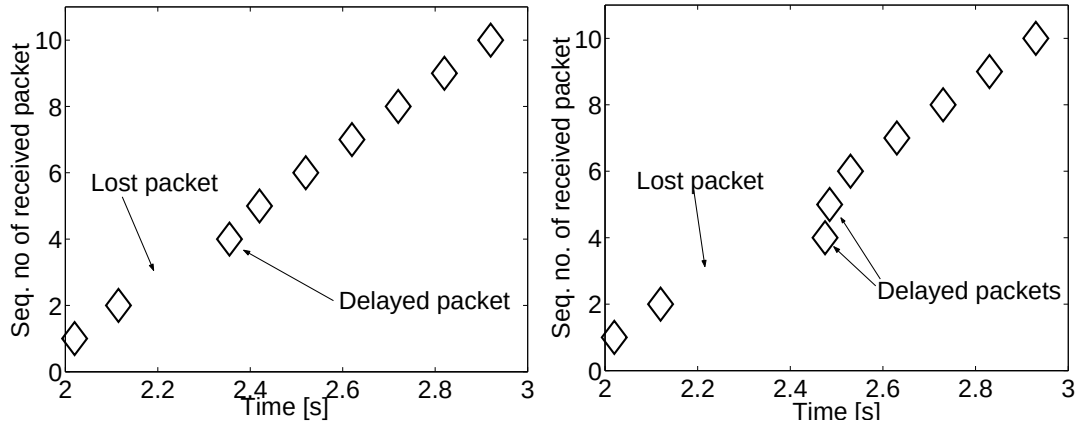


Figure 7.3: Sequence of received packets using either local (left) or source-initiated (right) repair.

outage of node B, the route is repaired locally. Here, a single packet is lost, a single packet

is delayed, and the path is reestablished after $t_r = 0.2\,\mathrm{s}$ (packets are buffered during the route update). Upon outage of node A, the route is repaired from the source. Here, it takes a little longer to repair the path, resulting in a slight delay of an additional packet. These results assume that the AODV routing messages are given precedence over the actual data packets and are sent in the reserved time slots. This priority explains the delay of real-time packets (which are sent as soon as possible after the routing packets); the lost packet is due to a timeout, not due to a collision. With only one packet lost, one (two for source-initiated route repair) packet delayed and an average time of 0.2 seconds, a repair of the reservation makes sense. Obviously, when the period of the real-time transmission is smaller, more packets are delayed (or dropped) during the repair time. However, an average of 0.2 seconds silent time is more acceptable than when a user must re-imitate a communication.

7.2.2 Reservation repair

If the routing protocol has performed a *local repair*, the best option for the DARE protocol is that reservations are repaired locally as well; some nodes that were part of the old route are still part of the new route, and neither the periodicity nor the time slot length have changed. Each node that is located in the communication range of the node initializing the repair is a potential candidate to be employed as a new relay node in the new reservation path. Each of these nodes has overheard the real-time transmission and is thus already avoiding the reserved time slots. Thus, it is very likely that these nodes can allocate resources in the hitherto avoided time slots. If so, there is no need for a new reservation setup. Any node with time slots reserved for overhearing/avoiding the reserved transmission (which has no other reservations with which these time slots would cause a conflict) can simply "take over" from the leaving node. If there would be a conflict with other reservations, the time slots must be shifted. The shifting algorithm is more described in Section 10.7.

If the routing protocol has performed a *source-initiated repair*, a completely different route could have been established. In this case, the only option for the DARE protocol is to release the old reservations and set up a completely new reservation from the source. On the one hand, such a source-initiated repair always yields an optimized path, compared with local repair. On the other hand, it might cause problems if nodes along the new route might not be able to fulfill the requested reservation. The reservation along the old path is released according to the method described in Section 7.3.

During the repair process, higher priority to the messages of the routing protocol are given, by transmitting them in the reserved time slots. This should accelerate the route repair process.

7.3 Release of unused reservation

There are two possible main methods to release any reservation: 1. Use explicit release messages and 2. Use a time out function. On the one hand, an explicit release message is more time efficient as the involved nodes that should release the reservations are directly told to do so. On the other hand, it is not guaranteed that all nodes can be reached. Consider a scenario where a node not directly involved in a reservation, but avoiding time slots in order not to interfere, moves away from the direct neighborhood of the reservation. This is illustrated in Figure 7.4 where node x has reserved time slots for avoiding the reservation from overhearing transmission from node C. Node x then moves away from the communication range of node C and can no

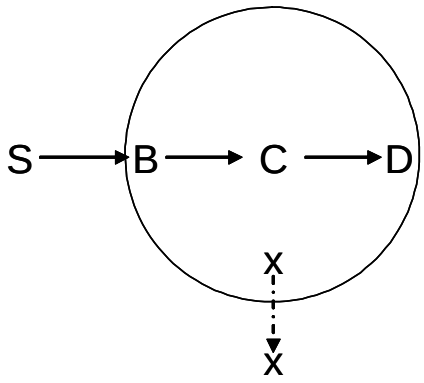


Figure 7.4: Node E moves out of reach from reserved path S-B-C-D.

longer be reached by any of the nodes participating in the reservation, hence cannot receive a potential release message. Node x must release its reservation by a time out function. Therefore, this is the method used in DARE. When a reservation has not been used for a certain time length it is considered inactive and released and the time slot can be used for other transmissions. The time out function in DARE is based on the transmission, reception or overhearing of a reserved transmission. When a node is missing reception during a reserved receive slot for n consecutive slots, the reservation is considered old and inactive. This generates a release of all existing reserved slots (transmission, reception or avoiding) in the reservation table for this flow. Similar is true for lack of successful transmission in n consecutive reserved transmission slots and for surrounding nodes, when they do not overhear during their reserved overhearing slots for n consecutive slots. In some applications where silent periods occur reservations might be falsely released. Therefore, the source node has an option to transmit dummy packets. However, this is not studied further in this thesis.

7.4 Reservation maintenance – Simulation study

This section describes a simulation-based investigation in a dynamic network where reserved paths break and are re-established. The idea is to investigate the repair mechanism in detail, and how fast changing networks affect the performance. DARE is compared to DCF with one real-time chain in a network with many nodes; the goal is to analyze if the repair mechanism is efficient enough.

7.4.1 Simulation model

In this simulation model, n = 100 nodes are placed randomly with the positions sampled from a uniform distribution on a square area with side $s = 700$ m. The simulation is performed using NS-2, which has a 802.11 DCF module implemented. All nodes use the maximum transmission power of 100 mW when transmitting. The resulting communication range is about $r_0 = 230$ m (see Appendix A), this setup guarantees that the resulting network is connected with high probability [94], i.e., each node can communicate with each other node either via a direct link or via multihop routing.

Two nodes are randomly chosen to act as the source and destination for the real-time traffic. All remaining 98 nodes generate background traffic, destined to the same destination as with real-time traffic. All routes are found using AODV as routing protocol. Both traffic types have a packet size of 512 bytes, in real-time flows transmitted every 100 ms and in background traffic with exponentially distributed inter-arrival times (exponential distribution parameters used are NS-2 default ones, see manual [90] for more information). The total load of the background traffic is at most 500 kbit/s. This rather low total load is chosen because it is well-known that DCF has poor performance for high loads [9]. Further, my investigation in Section 6.8 also results in the same conclusion.

To study the impact of topology dynamics, each non-real-time node switches off after some random time, which is sampled from a negative exponential distribution with a given expected value $E\{T_{\text{on}}\}$. It switches on again after another random time, sampled from the same distribution with the expected value $E\{T_{\text{off}}\}$, and so on. For simplicity, $E\{T_{\text{on}}\} = E\{T_{\text{off}}\} = \mu$. The total simulation time of one scenario is 3600 s, 50 random scenarios are investigated from which the performance values are averaged. The same experiments are repeated using conventional DCF.

7.4.2 Simulation results

The simulation results are shown with real-time delay, throughput and packet losses (defined in Section 3.6). Figure 7.5 shows the average delay. The use of DARE reduces the average delay compared to DCF. For both protocols, the average delay increases, as μ decreases. Small μ causes frequent topology changes which induce more route update procedures (and here more packet buffering).

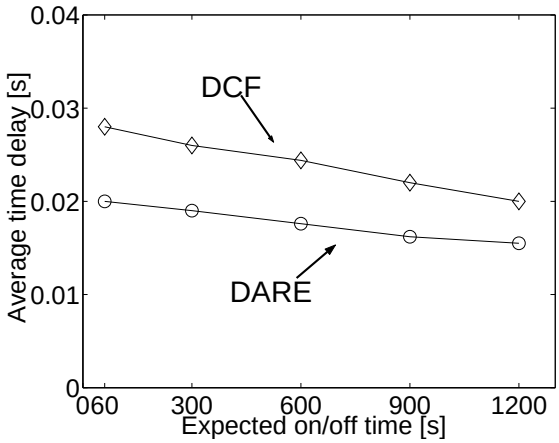
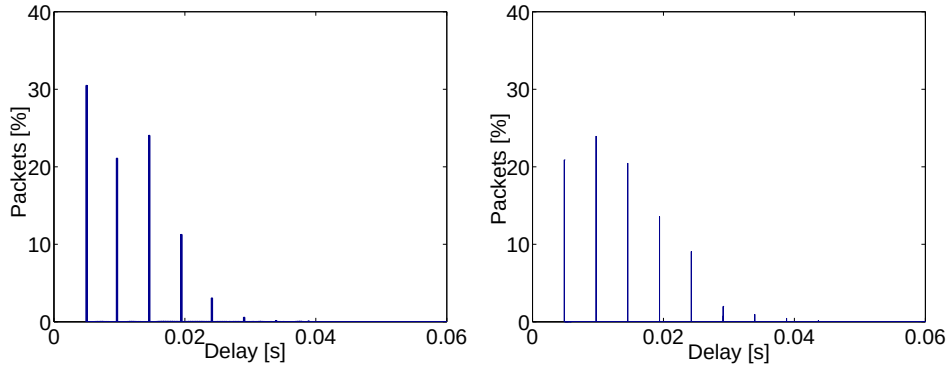


Figure 7.5: Average end-to-end delay of real-time packets.

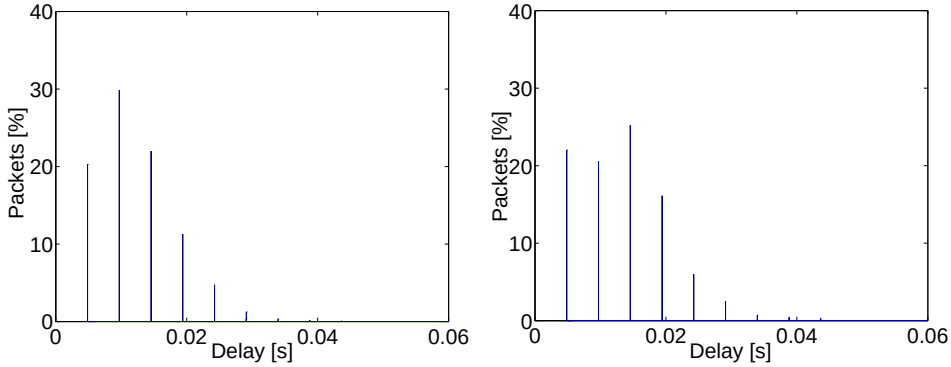
A deeper insight into the delay behavior can be obtained by studying the percentage at which certain end-to-end delay values occur, see histograms in Figure 7.6. DARE produces a discrete, sharply separated set of equidistant delay values. The different values correspond to different path lengths between the source and destination node. If the source and destination nodes can communicate within one hop, the communication takes 4.8 ms; if they are two hops apart, it takes 9.6 ms, and so on. It seems that longer on/off periods cause higher delays to occur with higher probability.

Figure 7.7 shows the corresponding CDF of the delay, defined as the percentage of received packets with a delay lower than a certain value. For example, more than $80\,\%$ of the packets experience a delay lower than 0.025 s (for all μ).

The delay histograms using DCF are shown in Figure 7.8. The histogram is not discrete, but the delay values are distributed around equidistant peaks. Again, the time value at which a peak occurs corresponds to the number of hops between source and destination. A one-hop communication takes about 5.5 ms. The delay variation increases with the number of hops. For example, the peak for a four hop communication (at about 0.02 s) is much wider than the peak for a two hop communication (at about 0.01 s). The corresponding CDF of the delay is depicted

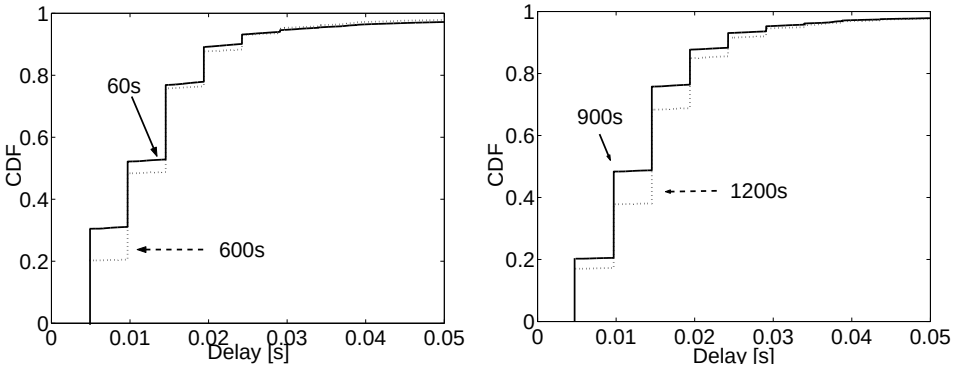


(a) $\mu = 60\,\mathrm{s}$ (b) $\mu = 600\,\mathrm{s}$



(c) $\mu = 900\,\mathrm{s}$ (d) $\mu = 1200\,\mathrm{s}$

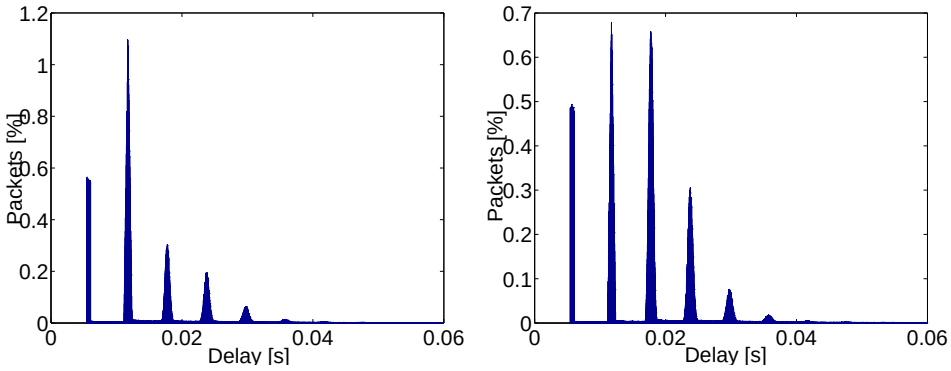
Figure 7.6: Histogram of end-to-end delay for real-time packets using DARE.



(a) $\mu = 60\,\mathrm{s}, 600\,\mathrm{s}$ (b) $\mu = 900, 1200\,\mathrm{s}$

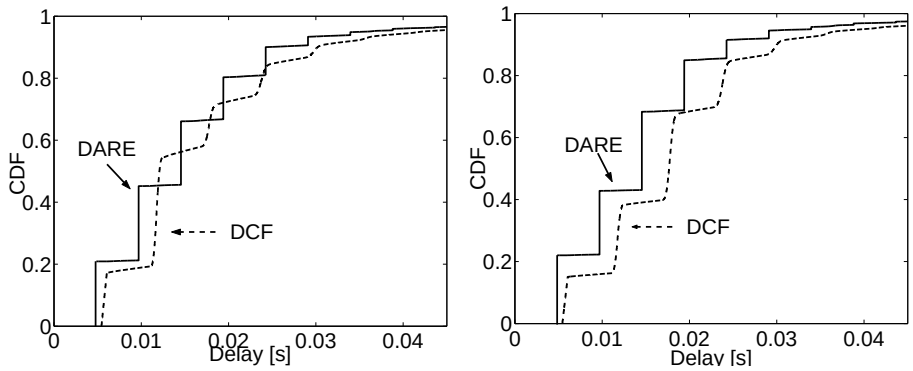
Figure 7.7: CDF of end-to-end delay of real-time packets using DARE.

in Figure 7.9, in comparison to that of DARE.



(a) $\mu = 600\,\mathrm{s}$ (b) $\mu = 1200\,\mathrm{s}$

Figure 7.8: Histogram of end-to-end delay of real-time packets using DCF.



(a) $\mu = 600\,\mathrm{s}$ (b) $\mu = 1200\,\mathrm{s}$

Figure 7.9: CDF of end-to-end delay of real-time packets.

In summary, the use of DARE for real-time traffic leads to better delay characteristics than the use of DCF: the end-to-end delay is on average lower and more stable.

The packet loss rate and throughput of DARE and DCF are shown in Figure 7.10. The average packet loss rate (left) using the DARE protocol is higher than that using DCF. The difference is about $30\,\%$ at small μ and about $25\,\%$ at larger μ. The reason for this difference is as follows: Using DARE, the nodes of the reservation path perform no channel sensing but transmit immediately during the time slots. Some of the non-real-time nodes are located at such a distance to the reservation path that they are too far away to successfully receive the reservation information, but still close enough to interfere the real-time transmission. This is due to the

fact that the transmission range between two nodes is in general lower than the interference range between them. The NRT nodes cause losses of real-time packets whenever they transmit only during a reserved time slot. These types of losses do not occur using DCF for real-time transmission, since here each node senses the channel before trying to transmit and backs off in case the channel is busy. In addition, if a collision would occur with DCF (if nodes start a transmission exactly at the same time), the packet is retransmitted, which improves the packet loss rate but increases the end-to-end delay as discussed above. One possibility to reduce the packet loss rate for DARE is to spread the reservation information to an increased distance around each sender (see Chapter 8).

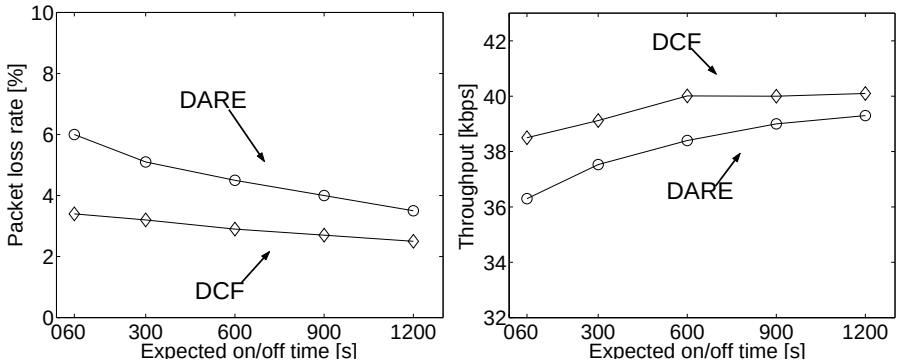


Figure 7.10: Average packet loss rate (left) and throughput (right).

In both protocols, the impact of μ on the packet loss rate is as follows: As μ increases, i.e., nodes switch on and off less frequently, less real-time packets get lost. One reason for this is that high μ causes less topology changes. Another reason is that the nodes perform a route search procedure every time they switch on. As the routing messages have priority over the reserved real-time transmission, the NRT nodes transmit during the allocated time slots, causing interference-based packet losses.

Figure 7.10 (right side) shows the throughput of both protocols. DARE has a slightly lower throughput than DCF, approximately $5\,\%$ lower for short on/off periods, and $2.5\,\%$ lower for long on/off periods.

7.5 Summary

To acknowledge that the reservation is still active, each packet transmission is acknowledged via iACKs at every relay node and an eACK from the final destination node. A node declares the path broken after n successive unsuccessful transmissions, receptions or overhearing. The

routing protocol handles the route repair. For a source-initiated route repair, DARE performs source-initiated reservation repair on the MAC layer. For a local route repair, it performs a local reservation repair. During the repair procedure, the messages of the routing protocol has highest priority and the real-time packets can be deleted or buffered. Reservations that are no longer needed are released, using a time out function. After n unused time slots (initial simulations have shown that $n = 4$ is a reasonable value), the reservation is released. All different time slot types can be used for this purpose; lack of successful transmission, lack of overhearing and lack of receiving generates a release of all different slots belonging to the actual reservation. Further, this can be used for any node to force a reservation to be broken so that a route update procedure is initiated (used for failure handling).

A simulation-based study shows that the repair mechanism of DARE is effective and well working in highly dynamic networks; even in networks where nodes switch on and off and reservations must be repaired due to broken paths, the DARE protocol has a lower averaged and more stable end-to-end delay. However, DCF has slightly higher throughput due to DARE's higher packet error rate. The reason for this is some lack of two hop protection around the reserved chain. This is an optional feature in the DARE protocol and described further in the next chapter.

Chapter 8

Two hop neighborhood protection

For any time slot based reservation to function successfully, surrounding nodes must abstain transmission during the reserved time slots. As described in Section 6.4, DARE uses a piggy back technique where information of reserved time slots are included in the set-up messages, data packets and acknowledgments. All nodes include not only the time slots where they actually receive or transmit, but also time slots reserved at nodes preceding in the chain, due to a larger interference range than communication range [87]. In the basic protection scheme all nodes participating in a reservation, and the nodes in the direct neighborhood of a reserved path, avoid time slots reserved up to two hops away. What is missing are nodes that cannot overhear reservation information, but can still interfere. Figure 8.1 illustrates a situation where a reserved transmission is ongoing between node A and node C. The area where it is possible to read reservation information from data packets and acknowledgments is grey. Node D is within this area and has overheard transmissions from node C; node D knows and abstains the reserved time slots. Node E is outside of this area, hence has no knowledge about the reservation. The interference range resulting from node E's transmission is larger than the receiving range from the reserved chain. As a result, node E could interfere with reserved reception at node C.

This chapter discusses extended protection around the multi-hop real-time transmission chain. The obvious advantage of such wider reservation protection is lower packet losses at a reserved transmission. However, increasing the range also results in less concurrent transmissions, hence the range of protection must not be too large; an over-protected network results in too low utilization of the available bandwidth. In [89], an analysis is made where it is shown that spreading the information *two hops around any receiver* gives the optimal result. In Section 8.1, an analysis of the gain resulted from such two-hop protection for a given packet error rate is analyzed. Section 8.2 presents a method for such spread in the DARE protocol and Section 8.3 describes a simulation of this optional feature.

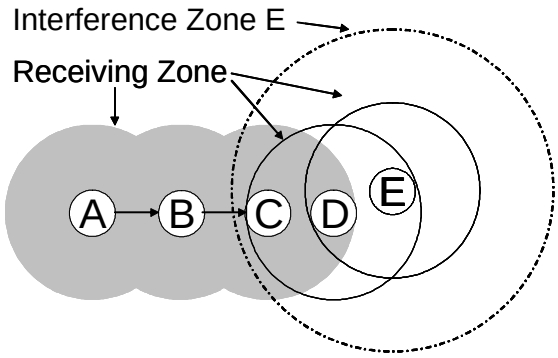


Figure 8.1: Reserved transmission from node A to node C. Node D has knowledge about the reservation whereas node E has not.

8.1 Analysis

This section presents an analysis of the gain from extending the dissemination of reservation information to two hops around a receiver [63].

8.1.1 Assumptions

For this analysis, some assumptions regarding channel, transmission rate and power levels are required. Some are defined in Chapter 3 and here repeated and some additional assumptions are in this section defined. The basic 1 MBps modulation modus of 802.11 with differential binary phase shift keying (DBPSK) is assumed. The receiver sensitivity is set to $-87\ \mathrm{dBm}$, which is common for currently deployed transceivers [95]. All nodes transmit with maximum power of $100\ \mathrm{mW}$. The *radio range* n is defined as the maximal interference range of a node; its value is implicitly given by the range where the received power equals, or is greater than, the background noise, chosen as $-111\ \mathrm{dBm}$ [87]. Further, a Line-of-Sight pathloss model is assumed. The *communication range* r is defined as the distance in which a node can successfully receive a packet in case of no interference. All nodes have uniform communication range and use the frequency of $2.4\ \mathrm{GHz}$. The path loss coefficient is 3 and maximum acceptable Packet Error Rate (PER) is set to $5\ \%$, commonly used as limit for a successful real-time transmission [96]. The packet size is varied: 64, 512 and 2300 bytes are investigated.

8.1.2 Scenario

The scenario which is used for the analysis is shown in Figure 8.2. Assume that node B has successfully scheduled a data transmission to node A, distance d between them. Message sent

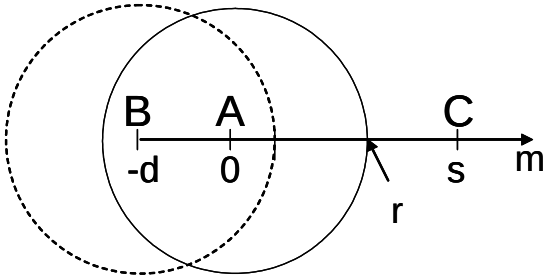


Figure 8.2: Basic scenario for protection analysis, distance on axis given in meters (m).

by node A can be correctly received within the solid line and messages sent by node B within the dotted line. Therefore, all nodes within these regions are aware of the reservation and can successfully avoid it; they cause no interference. The distance d can be any value in the range $(0, r)$. One possibly interfering node C exists at a distance s from node A, which is within the range (r, n). Outside n, which is the noise level, no interference will be generated at node A from node C.

Using this rather simple scenario, I will show that *even with only one interferer* the receiver suffers from unacceptable packet losses when node C is located outside of the communication range and starts a transmission during the time slot. The interference from node C at some distances s outside of the communication range leads directly to a packet loss.

8.1.3 Calculations

The goal is to calculate the possible gain two hop protection introduces. The steps towards this analysis is: 1. Calculate communication range r, 2. Calculate PER at node A for all d and s and 3. Show the gain that protection within $2r$ from a receiver gives. For all individual calculations in this section, formulas from link budget calculations are used [32].

The communication range r for our model is calculated using:

$$P_r(d) = P_t \frac{\lambda^2}{(4\pi)^2 \cdot d^\alpha}$$

where P_r and P_t are the receive and transmit power respectively, λ is the wavelength and α is the path loss coefficient. The values given in Section 8.1.1 results in $r = 45$ m.

Then the PER at node A with varied interfering distance $s = 45, 75, 90, 120$ and 145 m to node C is calculated. These values are chosen to be larger than the communication radius, thus nodes at these distances are not aware of the reservation. For this, the following equation is used:

$$\text{PER} = 1 - (1 - \text{BER})^{size}$$

where $\mathrm{BER} = \frac{1}{2}e^{-\frac{E_\mathrm{b}}{N_0}}$ is the Bit Error Rate for DBPSK modulation. $\frac{E_\mathrm{b}}{N_0}$ is calculated from:

$$\mathrm{SINR} = \frac{E_\mathrm{b}}{N_0} \cdot \frac{R}{\mathrm{BW}} = \frac{P_\mathrm{RBA}}{P_\mathrm{RCA} + N_0}$$

where P_RBA is the received power at A by node B, P_RCA is the received power at A by the interferer node C, BW is the bandwidth and R is the transmission rate. This PER was calculated for all possible $d = (0, r)$, increased with 1 m for each calculation. The resulting PER curves are shown in Figure 8.3. The results for $s > 90\ \mathrm{m}$ are omitted because the PER was $0\ \%$ in these cases.

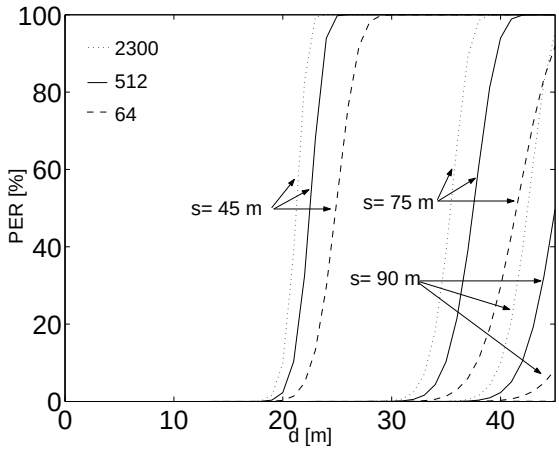


Figure 8.3: PER versus distance d for s = 45 - 90 meters and packet sizes of 64, 512 and 2300 bytes.

A more closer look of the 5% border is shown in Figure 8.4. These curves show that for all packet sizes, the critical s where the PER is larger than 5 % is within $2r$ for almost all distances d between the two communicating nodes. Using these results, for each $d = (0, r)$ the s that corresponds to a 5% PER is extracted. This is the minimum allowed distance that the interferer C can have from node A. Figure 8.5 shows the dependency of the minimal allowed interfering distance s versus the distance of the communicating peers d. The two straight dotted lines mark the distances where $s = r$ and $2r$ respectively. a and b are described in the next section. For all ds where node C is closer to node B (distance s is too small), the PER is larger than 5%, hence unacceptable. This result shows that the minimum required s falls between r and $2r$ for the majority of the cases, which suggests that it is sufficient to protect the receiver in a two-hops range. There is only a tiny interval $42\ \mathrm{m} < d < r$ which requires three-hop protection.

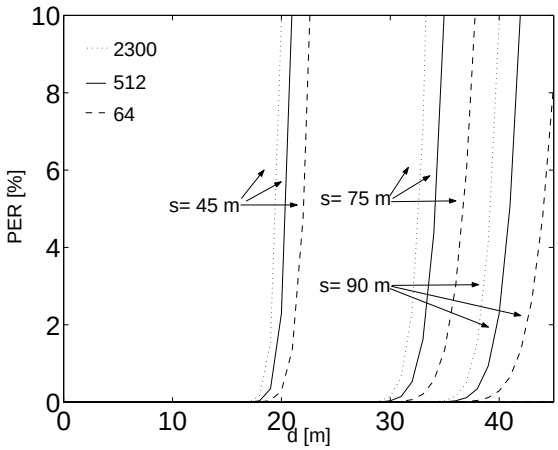


Figure 8.4: 5 % zoom of PER with and packet sizes of 64, 512 and 2300 bytes.

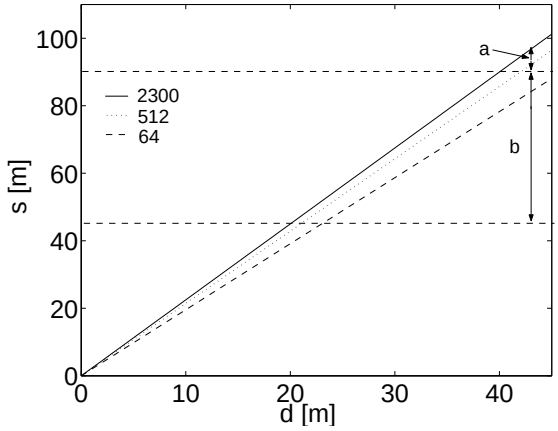


Figure 8.5: Acceptable s for a PER of 5 % and packet sizes of 64, 512 and 2300 bytes.

8.1.4 Probability of interference within two hop range

In this section, the probability of Cs location is investigated. The goal is to look at the possible gain with an introduced two hop protection around the receiver A. For this, two distances are introduced: 1) a is the part of the unacceptable s outside $2r$. 2) b is the part of s that is between r and $2r$. The parts a and b of the unacceptable s distances for all packet sizes result in problematic annulus areas, each spanned by either a ($2r > s > n$) or b ($r < s < 2r$). For these annulus the interfering node does not retrieve information about the transmission that will take place.

Assume that the nodes are randomly distributed uniformly. The probability that a node is within a certain region is simply the area of that region divided by the total area. Figure 8.6 shows the probabilities that the interferer C is within each annulus. It implies that the probabilities of a packet error is larger than 5 % for any node in this region, i.e., the node is located outside the communication range of node A, while still in the two-hop radius or even outside of the two-hop radius. Three lines for the probability are shown: First the PER is larger than 5 % and the interferer is outside the communication range r, shown in the figure as $Prob[PER > 5\%|s > r]$ (total probability). Second the PER is larger than 5 % and the interferer is outside r but within $2r$ ($Prob[PER > 5\%|r > s > 2r]$). Third the PER is larger than 5 % and the interferer is outside $2r$ ($Prob[PER > 5\%|s > 2r]$). The acceptable s is within $2r$ as long as the distance

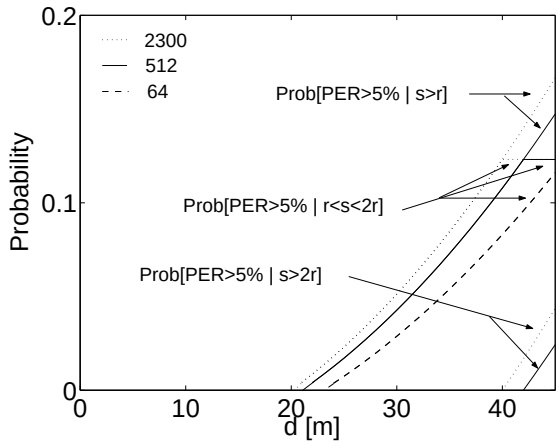


Figure 8.6: Probabilities that an interfering node is located within one of the annulus' and causing a PER larger than 5 %.

between the receiver and the transmitter is not larger than 42 meters. These results remain valid, independently from the packet size. Furthermore, the probability that a node at this position has an s which is within a two-hop communication radius is much larger than the probability that the node is located outside $2r$. Hence, by introducing a two-hop protection the number of possible interferer (which could cause a PER larger than 5 %) can be significantly reduced, see

Figure 8.6. For ds larger than 42 meters, the probability that one interferer is located such that $PER < 5\%$ is reduced with up to 80%.

8.2 Protocol implementation

There are two major options for implementing two-hop protection in the DARE protocol: implicitly inform using a piggy back technique as with the basic protection or explicitly spread information using new reservation information messages. As with the basic protection, signaling load should be kept as small as possible, hence a piggy-backing is a good technique. Since the nodes that should be informed do not overhear any transmission from the reserved real-time chain itself, nodes that are aware must forward this information. Assume a node not aware of the reservation starts a transmission of non-real-time data to a node that is aware of a reserved transmission in its direct neighborhood. Here, one simple method would be to use the RTS/CTS messages that these nodes exchange before transmission of the data packet. The receiver of the RTS with reservation information include this in the CTS sent back to a node that is non aware of the reservation. If the data packet is small, so no RTS/CTS exchange occurs, the information can be included in the acknowledgment. This is not further considered here.

The information that is carried by the CTSs includes the time slots reserved up to one hop away, which results in two hop protection at the receiver of the CTS. Consider the scenario shown in Figure 8.1. Node D includes information about time slots that are reserved at node C. Clearly, node E must not avoid all the time slots that D needs to avoid.

More problematic is a scenario where node E starts a transmission to another node that has no reservation information. This is at present not a feature of the DARE protocol, but an idea for how this could be realized is described in Appendix E.

8.2.1 Adding information onto CTS packets

Including the information in the CTS packet introduces overhead compared to the standard DCF CTS packet format [7]. If information about one reservation should be included the overhead consists of six new fields: Flag, Period (how often reservation repeats), Length (how long time each instance), Next instance (when the next reserved time-slot is relative to the station transmitting the CTS) and Source and Destination address for identification. The sizes of these extra fields are similar to those explained in Section 6.3.1. This results in that a DARE CTS is 96 bits larger than a CTS. Since these messages are rarely transmitted, this overhead is acceptable and it does pay off for the increased performance of real-time flows, which is investigated in Section 8.3.

8.3 Enhanced protection – Simulation study

For the evaluation of the extended protection, the same model as in Section 7.4.1 is used. Here, a short summary of this simulation model is given: 100 nodes are uniformly located in a square area with a side of 700 meters. One real-time path between a source and an AP exists, other nodes transmit background traffic to the same AP. 512 bytes packets are considered for both traffic types, in real-time flows transmitted every 100 ms and in background traffic generated with exponentially distributed inter-arrival times. The transmission rate is 1 Mbps and 50 NS-2 simulations each with 3600 s simulation time is run. Background nodes switch on/off to simulate a dynamic network with exponentially distributed expected on/off times: 60, 300, 600, 900 and 1200 seconds.

The results are presented with throughput and packet loss rate. The delay results are not affected by the increased protection and are the same as in the investigation performed in Section 7.4.1. These are here not again presented. Throughput and packet loss rates are shown in Figure 8.7 and Figure 8.8 where results for DARE with CTS information (DARE CTS), DARE without CTS information (DARE) and DCF are shown. The packet loss rates with CTS information is a lot lower than the DARE without it. For fast dynamic network, i. e. $E\{on\} = E\{off\} = 60$ s, the packet loss rate is decreased with approximately 45 %, from 6 % to approximately 3.5 %. For all values, DARE with the CTS information is lower than the DCF. This is an important feature as some applications have upper limits to the PER for good quality

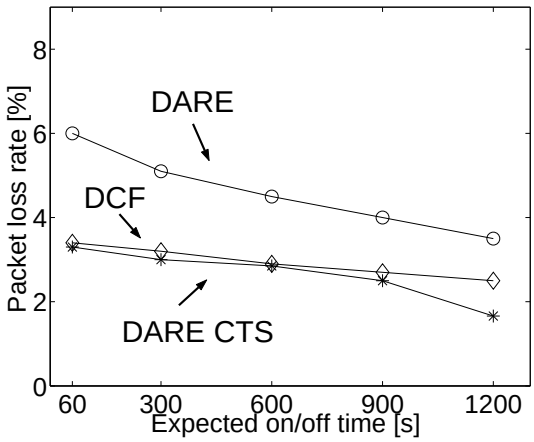


Figure 8.7: Average packet loss rate versus Eon/Eoff.

receptions. The analysis described in previous chapters require a PER smaller than 5%. With the information in CTS feature, this can be achieved. Average throughput in Figure 8.8 follows the packet loss rates and is increased when including information in the CTS, about the same as

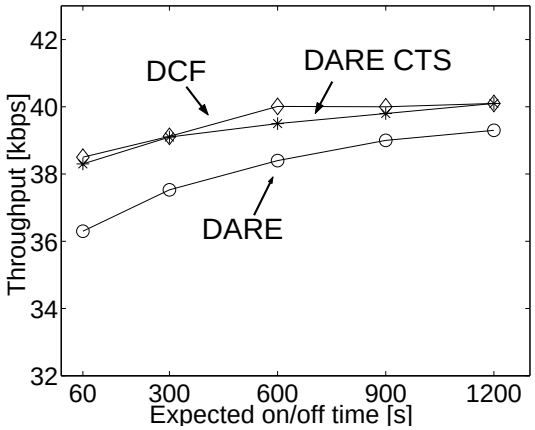


Figure 8.8: Average real-time throughput versus Eon/Eoff.

the throughput for DCF.

8.4 Summary

Protecting a reservation further than in the direct neighborhood can decrease packet loss rates, thus increase the throughput. In the DARE protocol, one simple method to implement such additional protection is to use the RTS/CTS exchange preceding a non-real-time data transmission and piggy back reservation information on the CTSs. This has proven to function well; in a large random simulative investigation, the packet loss rate can be reduced with up to approximately 45 %. For some QoS-demanding applications the packet loss rate is, aside from the end-to-end delay and jitter, an important metric. For these applications, including information in CTSs for surrounding transmissions is a successful method of protecting the real-time transmissions and improving the end-user's quality of reception. However, the approach is complex and requires extra signaling and can drain the network capacity if not used with care. The DARE protocol functions well without is as well, but for some application types with requirements on the packet loss rates it can be very useful. Therefore, this is an optional feature of the DARE protocol.

Chapter 9

Evaluation – one real-time flow comparison with EDCA

The last chapters have described the design of the different DARE components, also with simulative investigations for each component comparing DARE with the DCF. These results have been useful to identify the efficiency of the DARE protocol and its ability to better support QoS than the DCF for *one real-time flow*. However, since the motivation and hypothesis of the DARE protocol is that a reservation-based approach is better than the other main category of QoS enhancing mechanisms in DCF, the *priority mechanisms*, a comparison of them is of large interest before moving into design issues for scheduling of multiple flows. The priority mechanism EDCA is a good reference to use for comparison since it is standardized by IEEE 802.11 working group E. The crucial issue of a comparison with EDCA is simply: If EDCA performs better than DARE for one real-time flow, the DARE protocol is not the better approach, hence no purpose of a further design of the protocol with multiple flows.

This chapter describes a simulative comparison of a network with one real-time flow, transmitted with DARE and the EDCA. Section 9.1 summarizes the simulation set up, Section 9.2 describes the simulation results and Section 9.3 summarizes this section.

9.1 Simulation model

The simulation model used is the same as the investigation in the previous two chapters, see Section 7.4.1. Worth repeating is the network dynamics: Each node apart from source and destination switch on and off with exponentially distributed expected mean values, $E\{on\} = E\{off\} = \mu$. Further, the two-hop protection option described in the last chapter is not used. There is one difference in background load: Three different load values for the background

traffic is used: $\lambda = 5$ kbps, 10 kbps, and 100 kbps. These values are enough to show the impact background traffic has on the real-time traffic performance. The simulations of EDCA are performed with the highest priority (AC = voice) for the real-time flow and lowest priority (AC = background) for the background traffic (see Table 4.1). For more parameters of the EDCA simulation, see [97].

All above simulation choices are made to give EDCA the largest advantage in the comparison: The dynamic network (high signaling load for reservation repair), highest EDCA priority for the real-time flow, low background traffic and the lack of two-hop protection really puts DARE to the test.

9.2 Simulation results

9.2.1 Average packet delay and packet loss rates

Figure 9.1 shows the average end-to-end delay of real-time packets over μ for three different values of the background traffic ($\lambda = 5$ kbps, 10 kbps, and 100 kbps per NRT node). Using

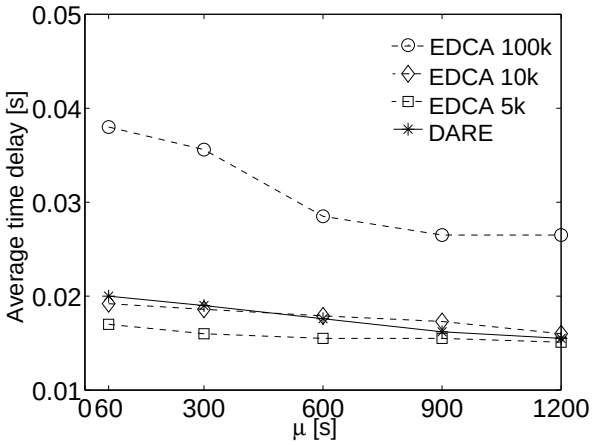


Figure 9.1: Average end-to-end delay of real-time packets versus μ.

EDCA, the background traffic has a huge impact on the delay. With 100 kbps load, the average delay is almost three times as large as with 5 kbps. The delay using DARE is independent of the background traffic. For both protocols, if nodes switch on and off more frequently (low μ) more delay is introduced, which is caused by more frequently performed re-routing procedures. If much background traffic is transmitted, EDCA suffers from a much higher delay than DARE. For medium background traffic load, both protocols show similar delay values. Only if the

background load is very low, EDCA is superior to DARE. The reason for the latter result is that, in the simulations, EDCA does not employ any exchange of RTS/CTS messages. This lack of coordination, however, can lead to high packet losses if small CW intervals are used, as it is then very likely that the backoff timers of two or more nodes expire at the same moment.

Thus, also the packet loss rate is analyzed, which is shown in Figure 9.2. As can be seen,

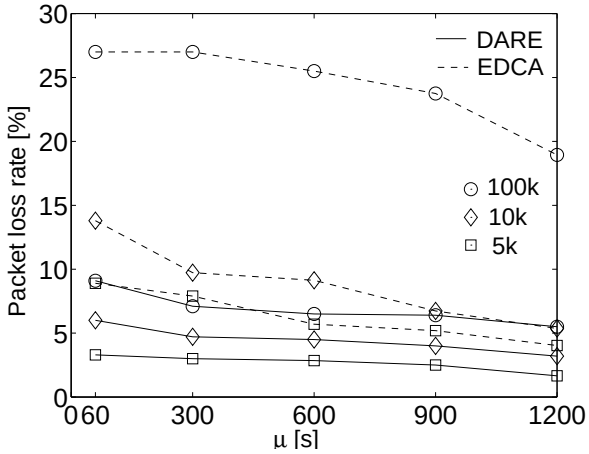


Figure 9.2: Packet loss rate versus μ.

DARE has a lower packet loss rate for all values of the on/off time and all background loads. Packet losses with DARE are mainly caused by interference from nodes that are more than one hop away from the real-time transmission path. Some of this interference is avoided, but not all (two-hop protection feature not used). Since retransmission of lost packets often makes little sense in real-time applications, the high packet loss rates of EDCA imply that data do not reach the final destination node.

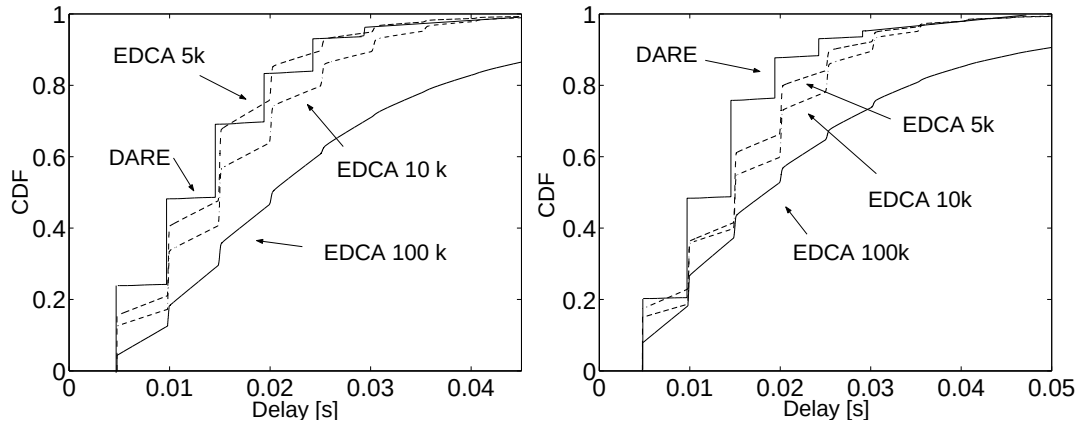
In summary, if the background traffic load is small, EDCA has slightly lower average time delay but higher packet loss rate. For high background loads, however, DARE has a much lower average time delay and lower packet loss rate. EDCA has a tradeoff between the packet loss rates and the time delay, which does not exist for DARE.

9.2.2 Distribution of packet delay

The end-to-end delay of a packet depends on the number of hops h on the path between the source and destination node. In this scenario, the source and destination node are randomly picked on the given area. Hence, the number of hops is a random variable. Simulations with

the given parameters have shown that the value range of h is between 1 and 6, where most source-destination pairs have $h = 2$ and 3.

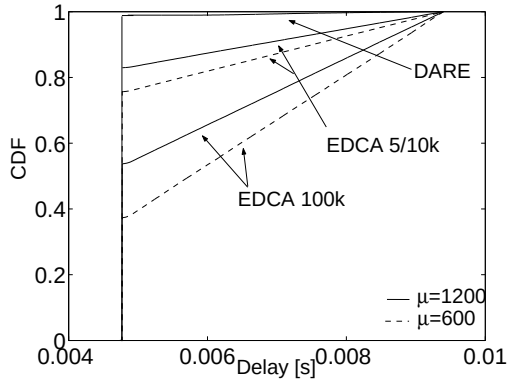
Using DARE, the transmission duration is $S_t = 4.8$ ms. Since no random access is needed, the transmission via one hop takes about this time. The total end-to-end delay is the sum of the delays of each hop. If the path between source and destination remains unbroken during transmission, the end-to-end delay will be hS_t. If the path breaks, the delay can also be higher than hS_t because packets are here buffered during the repair process. This behavior is well reflected in the DARE simulations results shown in Figure 9.3.



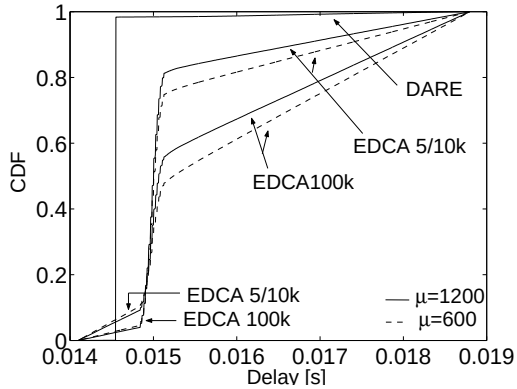
(a) $\mu = 600$ s (b) $\mu = 900$ s

Figure 9.3: End-to-end delay of real-time packets.

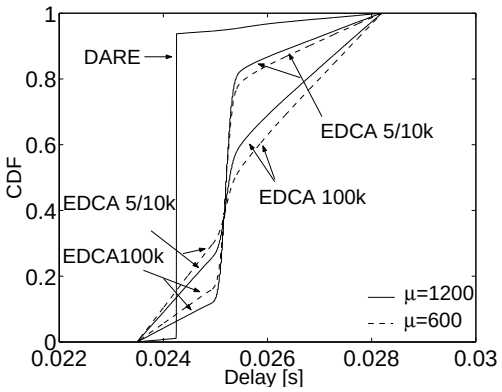
The CDF of the end-to-end delay is a step function with step width S_t. The delay CDF for $\mu = 600$ s can be interpreted as follows: About $20\,\%$ of all real-time packets are transmitted via only one hop ($h = 1$), hence their delay is 5 ms. About $50\,\%$ of all real-time packets experience a delay that is at most 10 ms, about $70\,\%$ have at most 15 ms, and so on. The result is similar for $\mu = 900$ s with slightly increased occurrence of lower delays caused by less frequent path breaks. If the number of hops is known, the packet delay is quite predictable; a sudden change of the packet delay only occurs for the first packets upon repair of a broken path. In total, the experienced jitter is very low. As the background load is decreased to $\lambda = 10$ kbps and finally $\lambda = 5$ kbps, the performance of EDCA improves but is still inferior to DARE (except for some particular delay bounds around 20 ms if $\mu = 600$ s). The delay behavior of packets using EDCA is completely different. Due to the random nature of the medium access in EDCA, the delay can now take values from a continuous value range. Hence, the CDF is no longer a step function. For much background traffic, EDCA achieves a much lower performance than DARE. In the simulation with $\lambda = 100$ kbps and $\mu = 600$ s, for instance, a delay below 20 ms is achieved by



(a) Packets after $h = 1$ hop



(b) Packets after $h = 3$ hops



(c) Packets after $h = 5$ hops

Figure 9.4: Delay of real-time packets after h hops. The results for $\lambda = 5$ kbps and 10 kbps are very similar.

about $50\,\%$ of all packets, compared to more than $80\,\%$ using DARE. Also the jitter in the delay using EDCA is higher and increases with increasing number of hops.

In a final simulation, the delay for all real-time packets at a given hop distance h from the sender is investigated. Figure 9.4 shows results for $h = 1$, 3, and 5. Using DARE, almost $100\,\%$ of the real-time packets arrive at the first hop within 4.8 ms and at the third hop within 14.4 ms. About $95\,\%$ of the packets arrive at the fifth hop within 24.2 ms. Hence, almost all packets arrive at a well-defined time, thus no jitter occurs. Using EDCA, there are some few packets that arrive faster than with DARE. The majority of packets, however, arrive later. For example, with $\lambda = 100$ kbps background load and $\mu = 600$ s, the delay of $80\,\%$ of the packets is still higher than 24.2 ms at the fifth hop.

9.3 Summary

The simulation study performed in this chapter compared for one real-time flow the two main approaches for QoS in DCF-based networks: reservation-based approach versus service differentiation. In a sense, this is an evaluation of how the inter-node signaling needed for reservations compares to the signaling overhead introduced by the contention-based priority mechanism. To make it even more interesting, the comparison is performed such that the priority mechanism used, the IEEE 802.11 E standardized EDCA, is given the most advantages compared to DARE; this comparison is performed in a scenario where reservations are repaired frequently, no two hop protection of DARE is used and EDCA's highest priority class is used for real-time traffic.

To summarize the results, DARE has the capability to give real-time flows a more stable end-to-end delay with low average. A major advantage is that DARE can guarantee a fixed delay per real-time flow even if the background traffic is very high. EDCA can bound the average end-to-end delay, but the delay is dependent on the surrounding background traffic load.

Even in scenarios where paths break often and the reservation repeatedly has to be re-established, DARE outperforms the less complex priority mechanism EDCA both in delay and packet losses (thus also throughput). DARE is an effective QoS approach for one real-time flow; it makes sense to continue the design with scheduling and support for multiple flows.

Chapter 10

Scheduling of multiple reservations

Multiple reservations in a network with a non-defined collision domain and no central control that globally schedules the transmissions is challenging. The past chapters have described all components of the DARE protocol apart from how to handle multiple reservations. As described, the functionality of the DARE protocol is controlled locally; only nodes participating in the transmission and the direct neighbors are involved. At the set up, this can result in that nodes in the same neighborhood reserve time slots that might overlap, which can cause collisions and unsuccessful reserved transmissions. If this cannot be controlled and avoided, the benefit of a reservation for a transmission is lost. The goal of this chapter is to find a simple distributed solution where time slots are locally scheduled so the packet loss rate is kept low. First of all, the problem and possible solutions are introduced in Section 10.1 and Section 10.2. This section concludes with a suggestion for a time shift function where overlapping slots are shifted in time, locally at a node. For this node to be able to find the correct shift, an expression for when time slots overlap is derived in Section 10.3. This is used in Section 10.4 for an assertion, which defines how a possible time shift can be computed. Section 10.5 describes how the shift function should be used and Section 10.6 investigates the probability that time slots directly overlap, i.e. how probable a direct shift is. Section 10.7 describes the DARE protocol implementation of the shift function, which is evaluated with simulations in Section 10.8. Section 10.9 describes some limitations to the number of accepted reserved flows and finally, Section 10.10 summarizes this chapter.

10.1 Problem description

When no central controlling unit exists and the users themselves allocate periodic time slots for several reserved paths, the time slots of different paths might overlap. This could lead to a

collision and has to be avoided. Depending on the periods of different flows, the impact that overlapping time slots have is different. If the periods of two paths with overlapping time slots are equal, the collisions will take place in all slots. If the periods are different, which can be possible if e.g. different CODECs (G. 729 [68] and G.711 [98] are some examples) are used in the nodes' equipment, only some of the time slots might overlap. An example for this is shown in Figure 10.1, where two paths have overlapping time slots.

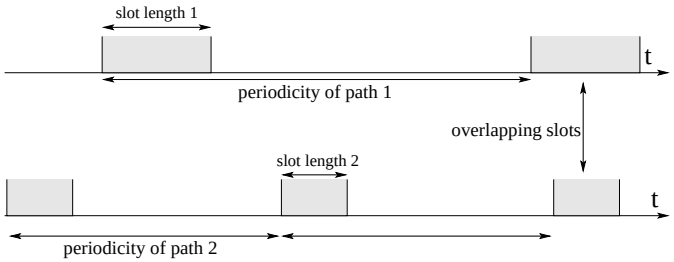


Figure 10.1: Example where time slots overlap.

When a node already has one, or several co-existing, reservation(s) and it receives a request for a new one, it must decide based on local information whether or not to accept the request. The possible solutions that these nodes have are described in the next section.

10.2 Possible solutions

Nodes that receive a request for a time slot reservation that is conflicting with an already existing one have the following options for handling this situation:

Fixed transmission schedule The most common solution for scheduling of time slots is to use a repeating transmission schedule that is divided into pre-defined time slots in a TDMA-based manner. All nodes can assign one (or several) time slot (s) for a transmission flow. This enables the nodes to keep a schedule of periodic reservations which is quite simple to maintain. Several time slot assignments methods exist [62, 64, 65]. The disadvantage of such an approach is first of all that all nodes must be centrally synchronized; all time-slots in the system must start and end at the same time. This is not an option here as the network is totally distributed. But, even if this would be possible, such an approach is not the optimum when different transmission flows require different time slot sizes and periods. The different time slot sizes could lead to overhead when a requested time slot is smaller than the fixed time slots in the transmission schedule; a full time slot has to be reserved although the actual time used for transmission is smaller. The different periods are problematic as a flow might not need a time slot in every frame; the unused

time slots might not be possible to assign to another flow. A different approach is needed that is totally distributed and that can handle different time slots and periods.

Allow overlapping time slots The easiest solution for all nodes is to allow overlapping time slots. This is only suitable if the packet loss rates are acceptable for the application. Furthermore this solution only makes sense if the periods are different; equal periods with time slot overlapping results in a packet loss rate of 100%. As the solution should be independent of the periods, this is not a solution which will be considered.

One path abandons If a slot is known to overlap with a slot of another path, one packet that was supposed to be sent in the overlapping time slots is discarded. If the time slots repeatedly overlap, there are two options: 1) One flow discards all the packets meant to be sent in these time slots or 2) the transmission flows takes turn, e.g. in a round robin scheme, to discard a packet. Both options can only be used when the periods are very different so the resulting rate of dropped packets is not too high. Therefore, this solution is not further considered here; the solution should be independent of both period and time slot sizes.

Time shift Upon receiving a conflicting time slot request, a node shifts one periodic reservation in time so that all reservations successfully co-exists and no overlapping time slots exist at all. The transmission flows that are shifted will suffer from some extra end-to-end delay for all the packets, but all packets arrive at the receiver at the same time (no variations/jitter) . All nodes must be able to calculate a suitable transmission instance for conflicting time slot request and suggest a new reserved instance. This implies that some extra signaling is needed. Further, when a node cannot find a suitable time shift for a flow, the request must be rejected. This solution accepts all time slot sizes and periods. No central synchronization is needed. Therefore this is the suggested solution to be implemented in the DARE protocol which will be further investigated in this chapter.

10.3 Foundations – time slot overlapping

To be able to define the proper shift function, some pre-requisite information is needed. First of all, a mathematical expression of when time slots of two paths with different periods overlap is needed. These formulas are then used for a simple simulation-based investigation; the goal is to find a pattern for how the time shift time should be chosen such that no time slots at all overlap. The two pre-requisite parts are useful for an assertion of the time shift function and its proof.

For these investigations the notations visualized in Figure 10.2 are used. Assume that path

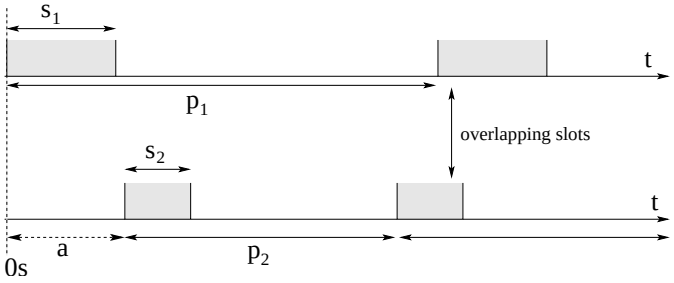


Figure 10.2: Two paths with period p_1 and p_2, and their respective time slot s_1 and s_2 should co-exist.

1 (p_1,s_1) exists and path 2 (p_2, s_2) should be set up and that $p_1 > p_2$ (which period is the larger one does not influence the results). The first slot of path 2 is requested at time a, which can be any point in time when the first reservation is active. For simplicity, a is assumed to occur between any instance [0,p_1]. As a conclusion, path 1 transmits its second time slot at p_1 and path 2 at $a + p_2$.

10.3.1 Overlap of time slots of two paths

To be able to avoid high occurrence of overlapping time slots, which leads to packet errors, the impact of the starting point a on the packet error rate must be investigated. Using the notations in Figure 10.2, the goal in section is to derive an expression for when time slots of two paths, regardless of period, overlap that is depending on the starting point of the second path, a. This will then be used to find a function to avoid such starting points. The first slot of path 2 overlaps with the first or second slot of path 1 if:

$$a \leq s_1 \tag{10.1}$$

$$\vee \quad a \geq p_1 - s_2 \tag{10.2}$$

If Equation 10.1 is true the first slot of path 2 overlaps with the first slot of path 1. If Equation 10.2 is true, the first slot of path 2 overlaps with the second slot of path 1. This means that a time displacement a that avoids the collision of the first slots can only be found if $p_1 > s_1 + s_2$.

The second slot of path 2 overlaps with the second slot of path 1 if the second slot of path 2 is between $p_1 - s_2$ and $p_1 + s_1$:

$$p_1 - s_2 \leq a + p_2 \leq p_1 + s_1 \tag{10.3}$$

The different ways the second time slots of path 2 may overlap are shown in Figure 10.3. In

a) s_2 starts before s_1 is ended. In b) s_2 starts before s_1 has started, but also ends after s_1 has started. In c) s_2 both starts and ends with s_1 and in d) s_2 starts before s_1 and ends after s_1. All these cases are covered by Equation 10.3.

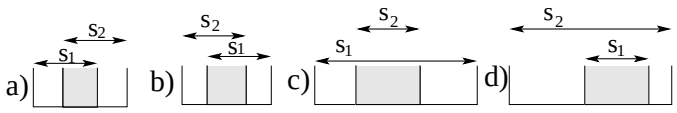


Figure 10.3: The different cases when time slots can overlap. Grey areas indicate cases where slot s_1 and s_2 overlap.

For any other slot than the first slot of path 2, Equation 10.3 can be extended to:

$$n_1 p_1 - s_2 \leq a + n_2 p_2 \leq n_1 p_1 + s_1 \tag{10.4}$$

where $n_1, n_2 \in \mathbb{N}$. $n_1 p_1$ describes the point of time path 1 starts to transmit its $(n_1+1)^{\text{th}}$ slot and at $a + n_2 p_2$, path 2 starts to transmit its $(n_2+1)^{\text{th}}$ slot. If Equation 10.4 is true for any combination of $n_1, n_2 \in \mathbb{N}$ the $(n_1+1)^{\text{th}}$ slot of path 1 overlaps with the $(n_2+1)^{\text{th}}$ slot of path 2. Equation 10.4 is for simplicity expressed as a system of two equations:

$$n_1 p_1 - s_2 \leq a + n_2 p_2 \tag{10.5}$$

$$\wedge \quad n_1 p_1 + s_1 \geq a + n_2 p_2 \tag{10.6}$$

which can be transformed into:

$$n_1 \tfrac{p_1}{p_2} - \tfrac{s_2}{p_2} - \tfrac{a}{p_2} \leq n_2 \tag{10.7}$$

$$\wedge \quad n_1 \tfrac{p_1}{p_2} + \tfrac{s_1}{p_2} - \tfrac{a}{p_2} \geq n_2 \tag{10.8}$$

Both Equations 10.7 and 10.8 can only be true if n_2 is the smallest natural number greater than $n_1 \frac{p_1}{p_2} - \frac{s_2}{p_2} - \frac{a}{p_2}$ and at the same time the greatest natural number smaller than $n_1 \frac{p_1}{p_2} + \frac{s_1}{p_2} - \frac{a}{p_2}$ respectively. To conclude, Equation 10.7 and Equation 10.8 lead to the following:

If

$$\lceil n_1 \frac{p_1}{p_2} - \frac{s_2}{p_2} - \frac{a}{p_2} \rceil = \lfloor n_1 \frac{p_1}{p_2} + \frac{s_1}{p_2} - \frac{a}{p_2} \rfloor \tag{10.9}$$

then

$$n_2 = \lceil n_1 \frac{p_1}{p_2} - \frac{s_2}{p_2} - \frac{a}{p_2} \rceil \tag{10.10}$$

or

$$n_2 = \lfloor n_1 \frac{p_1}{p_2} + \frac{s_1}{p_2} - \frac{a}{p_2} \rfloor \tag{10.11}$$

respectively. If Equation 10.9 is true both Equations 10.10 and 10.11 lead to the same result and as a conclusion the $(n_1 + 1)^{\text{th}}$ slot of path 1 and the $(n_2 + 1)^{\text{th}}$ slot of path 2 do overlap. These equations are used in the next section, where a simulative investigation is performed to look more into the behavior of the packet error rates versus the starting point a.

10.3.2 Frequency of overlapping time slots

To further look into how the time shift a should be chosen, a simulation study of different combinations of p_1, p_2, s_1, s_2 and a impact the packet error rate is performed. Path 1 is set up and has again its first time slot starting at a reference time 0. Path 2 is also requested at time 0 but shifted with an a within $[0, p_1]$. For each combination of (p_1, s_1), (p_2, s_2), several simulations are performed, each simulation has a different time shift a. Table 10.1 shows the p_1, p_2, s_1, s_2 used. These values include typical VoIP parameters, e.g. for ITU-T's G.729 [68] and G.711 [98], and also extend these values to get a general idea of the time shift characteristics. The a alters for each simulation with 1 ms, starting at 0 and finishing at p_1. No larger value on a is investigated as this will lead to the same results as for the first period $[0, p_1]$. The number of overlapping time slots is measured for $5000 \cdot p_1$, using Equation 10.9.

p_1, p_2:	[10, 11,. . . ,40]
s_1, s_1:	[1, 2,. . . ,11]

Table 10.1: All employed values for the parameters.

For most combinations it was not possible to avoid overlapping time slots completely. Some of these combinations are shown in Table 10.2; a checkmark ($\surd$) indicates that an a was found such that no time slots overlapped.

	p=20 ms s=2 ms	p=30 ms s=2 ms	p=40 ms s=2 ms	p=20 ms s=4 ms	p=30 ms s=7 ms	p=40 ms s=9 ms
p=20 ms, s=2 ms	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$
p=30 ms, s=2 ms	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	
p=40 ms, s=2 ms	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$
p=20 ms, s=4 ms	$\surd$	$\surd$	$\surd$	$\surd$		$\surd$
p=30 ms, s=7 ms	$\surd$	$\surd$	$\surd$		$\surd$	
p=40 ms, s=9 ms	$\surd$		$\surd$	$\surd$		$\surd$

Table 10.2: Combinations of periods and slot durations for which an a was possible to find with no time slots overlapping.

To get an indication of what the rate of overlapping time slots actually mean, assume that one packet is sent in each time slot and both packets transmitted during s_1 and s_2 are lost if they overlap. First, a case where the time shift was successful is shown with packet loss rate versus time shift a from 0 up to p_1 (30 ms) in Figure 10.4. Path 1 has a packet loss rate of 50% when a is chosen so s_1 and s_2 overlap and path 2 ($p_2 = 20$ ms) has 33%.

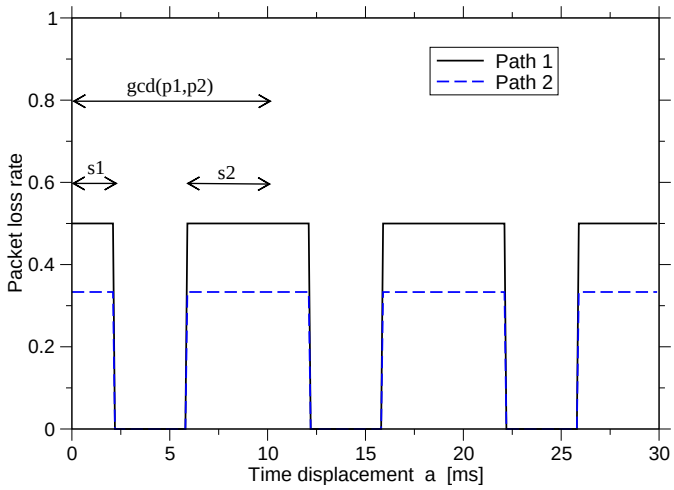


Figure 10.4: PER versus time shift a for two paths, path 1: $p_1 = 30ms, s_1 = 2ms$, path 2: $p_2 = 20ms, s_2 = 4ms$. A successful time shift is possible.

Figure 10.5 shows an example of a set of parameters for which it is not possible to find such displacements, again with the packet loss rate versus the time shift a. Here it is possible to distinguish that path 1 has up to 66% packet loss rate and path 2 up to 50%.

First of all, in both figures it is possible to distinguish that if a time shift a is falsly chosen, the resulting packet loss rate is unacceptably high. Further, the packet loss rate follows a certain periodic behavior. As shown in both figures, the periodicity is the *greatest common divider* (gcd) of the two periods. Hence it makes little sense to choose a time shift larger than the gcd; it only repeats itself. This is more described in Section 10.5.1. Also shown with arrows in these figures are the two slot lengths s_1 and s_2. As illustrated (and also valid from all results in Table 10.2), if the sum of the two slot lengths is smaller that the gcd, it is possible to shift path 2 with a so that no time slots overlap.

Finally Figure 10.6 shows a case where no gcd exist for the two different periods. First of all, no time shift can be found so that no time slots overlap. The resulting packet loss rate however, is much lower than for the cases where the periods have a gcd. Other combinations of Table 10.2 are shown in Appendix D. From these results, it is now possible to state an assertion for the time shift function.

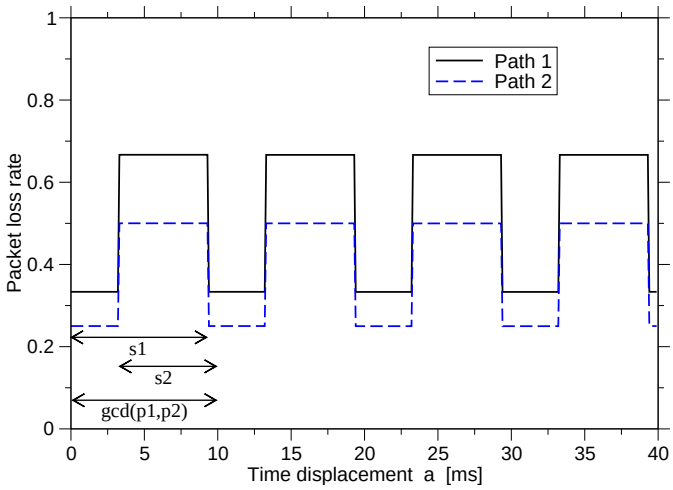


Figure 10.5: PER versus time shift a for two paths, path 1: $p_1 = 40ms, s_1 = 9ms$, path 2: $p_2 = 30ms, s_2 = 7ms$. A successful time shift is not possible.

10.4 Assertion for the shift functionality

Overlapping time slots can be avoided accordingly: If the periodicities p_1, p_2 have a *greatest common divider* (gcd) which is larger than the sum of the slot lengths, a time displacement a can be found so that the time slots do not overlap. Mathematically: If

$$\gcd(p_1, p_2) > s_1 + s_2 \tag{10.12}$$

then a displacement a can be found so that time slots of two different paths do not overlap.

Explanation Again let the first slot of both path 1 and path 2 start at reference time 0. The next time the two paths have slots directly overlapping is at time $xp_1 = yp_2$. The time between a slot of path 1 and a slot of path 2 is m. The minimum m that always exists is the gcd. Therefore, for two slots to successfully co-exist, the sum of these two slots must be smaller than the gcd. This can be seen in Figure 10.4 – the packet loss rate is non zero at periodic instances with period gcd. The following paragraph is a proof that the smallest m is always the gcd.

Proof Assume two natural numbers $z_1, z_2 \in \mathbb{N}$ that have a gcd d, i.e. these numbers can be expressed in the following way:

$$z_1 = n_1 d$$
$$z_2 = n_2 d$$

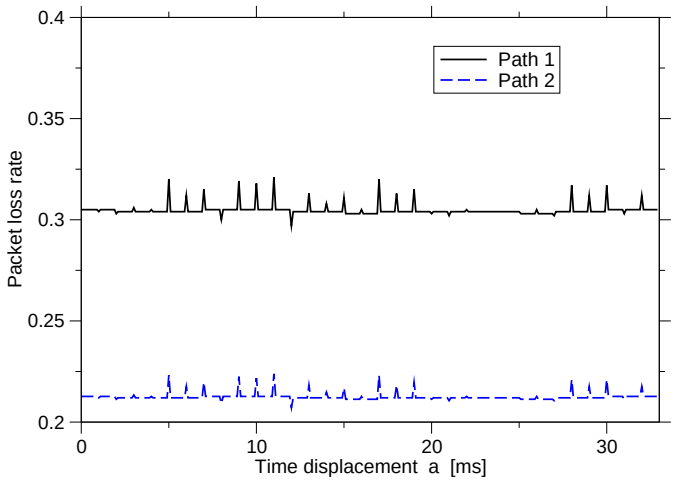


Figure 10.6: PER versus time shift a for two paths, path 1: $p_1 = 33ms, s_1 = 4ms$, path 2: $p_2 = 23ms, s_2 = 3ms$. No gcd exists for the two periods.

where $n_1, n_2 \in \mathbb{N}$. The difference between these two numbers has to be multiples of d:

$$|z_1 - z_2| = |n_1 d - n_2 d| = |n_1 - n_2| \cdot d \tag{10.13}$$

Since both n_1 and n_2 are natural numbers, also $|n_1 - n_2|$ must be a natural number, which in the smallest case larger than zero is one. Therefore the smallest *possible* difference larger than 0 is $1 \cdot d$. However, the fact that the smallest possible difference is the gcd does not mean that it always exists. To prove that it always exists it has to be shown that for every two periodicities p_1 and p_2 one can find n_1 and n_2 so that

$$|n_1 p_1 - n_2 p_2| = d \text{ with } n_1, n_2, p_1, p_2 \in \mathbb{N}, \tag{10.14}$$

with d being the gcd of p_1 and p_2. This can be shown with the help of theorems from the elementary number theory, which are explained in the following paragraphs. Theorem 1 is needed for Theorem 2 and here not described in full.

Theorem 1 For every two integers a and b with $a \neq 0$ there are unique integers q and r with

$$b = qa + r \text{ and } 0 \leq r < a.$$

The proof of this theorem can be found in [99] and in Appendix C.1

Theorem 2 If $a, b \in \mathbb{Z}$ both not 0 then the set

$$M := \{xa + yb : x, y \in \mathbb{Z}\} \tag{10.15}$$

contains positive integers. The smallest among them is the *greatest common divider* d of a and b.

For every gcd d of a and b there is $x, y \in \mathbb{Z}$ so that

$$d = xa + yb. \tag{10.16}$$

This theorem is proven as follows: It is obvious that the set M (Equation 10.15) contains positive integers. Let d be the smallest among them and let $x, y \in \mathbb{Z}$ be chosen so that $d = xa + yb$. Obviously every common divider smaller than the gcd d of a and b is also a divider of d.

If the smallest number of M is not the gcd, i.e. d is *not* a divider of a nor b, Theorem 2 does not hold. So, assume that this is the case, i.e. $d \not| a$. Theorem 1 gives that there is $q, r \in \mathbb{Z}$ such that $a = qd + r$ and $0 < r < d$. This can be transformed to:

$$r = a - qd = a - q(xa + yb) = (1 - qx)a + (-qy)b$$

$(1 - qx)$ and $(-qy)$ are integers which can be expressed as $\tilde{x}$ and $\tilde{y}$ giving that the number r is an element of M by definition:

$$r = \tilde{x}a + \tilde{y}b \in M$$

However, since $0 < r < d$ and d stated above is the *smallest positive integer*, this is a contradiction. The assumption $d \not| b$ can be led to a contradiction in an analog way. Conclusively d is the greatest common divider of a and b.

Transforming the above results to positive periodic integers p_1 and p_2, i.e. a and b are positive integers, it is obvious that either x or y from Equation 10.16 has to be a negative integer so that d is the gcd of a and b. If one instance of a slot of path 1 e.g. s_1 occurs at a reference time t_0, there is one slot of path 2 at a time $t_1 > t_0$ and another time slot of path 2 at $t_2 < t_0$. As a conclusion the propositions of Equation 10.14 and Theorem 1 are equal. To conclude, the smallest inter slot time between any arbitrary slots belonging to two different paths is always the gcd of the two periods.

10.5 The scheduling algorithm – gcdShift

After the sections analyzing the problem and finding the requirements for the time shift method, this section describes the shift function *gcdShift* in more detail. Illustrated by two examples, a rule for the time shift a is defined, which also explains the periodic behavior of the packet loss rates of the figures in Section 10.3.2. Then the computation of the gcd is presented, followed by some requirements on how to shift the individual slots. Further, the time shift function for more than 2 paths is described. The gcdShift algorithm itself is described in detail in Appendix C.3.

10.5.1 Time displacement

Figure 10.7 shows path 1 and path 2 where $gcd(p_1, p_2) > s_1 + s_2$. Three shifted versions of path 2 are shown: case *a*, *b* and *c*. Case *a* shows that path 2 has to be shifted at least s_1 in order not to overlap with the first slot of path 1. Case *b* shows that path 2 cannot be shifted more

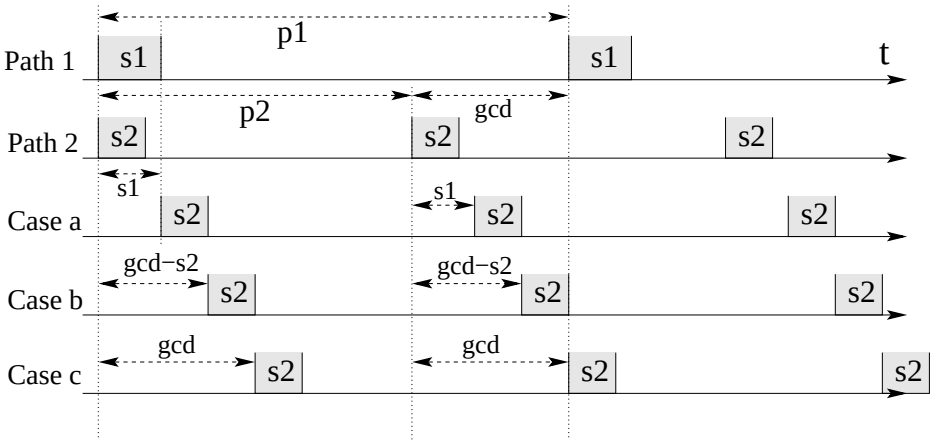


Figure 10.7: Shifting path 2 in time when $gcd(p_1, p_2) > s_1 + s_2$.

than $\gcd(p_1, p_2) - s_2$ so that the second slot of path 2 does not overlap with the second slot of path 1. Therefore the time shift must be $s_1 < a < \gcd(p_1, p_2) - s_2$. Case *c* shows when the time displacement is exactly $gcd(p_1, p_2)$. The two time slots that had been $\gcd(p_1, p_2)$ away from each other with no time displacement ($a = 0$) now exactly overlap. Further shifting of path two will lead to the same results as when it was $0 \leq a < \gcd(p_1, p_2)$. As a conclusion the repeating pattern of the packet error rates in Figure 10.4 is $\gcd(p_1, p_2)$.

Figure 10.8 shows an example where $gcd(p_1, p_2) < s_1 + s_2$. Two shifted versions of path 2 are shown: case *a* and *b*. Case *A* shows that the second slot of path 2 already overlaps with the second slot of path 1 although the first slots of both paths still overlap; more slots overlap with such a shift than without any shift. This explains why the curves in Figure 10.5 jump to a higher

packet loss rate level after a time displacement $a = \gcd(p_1, p_2) - s_2$. In case *B* the second slots of path 2 starts at the exact same time as the second slot of path 1; both paths overlap the same way as the first slots of both paths overlapped without any shift. Thus, the packet loss rate follows a repeating pattern also in this case and the curves in Figure 10.5 have a periodicity of $\gcd(p_1, p_2)$.

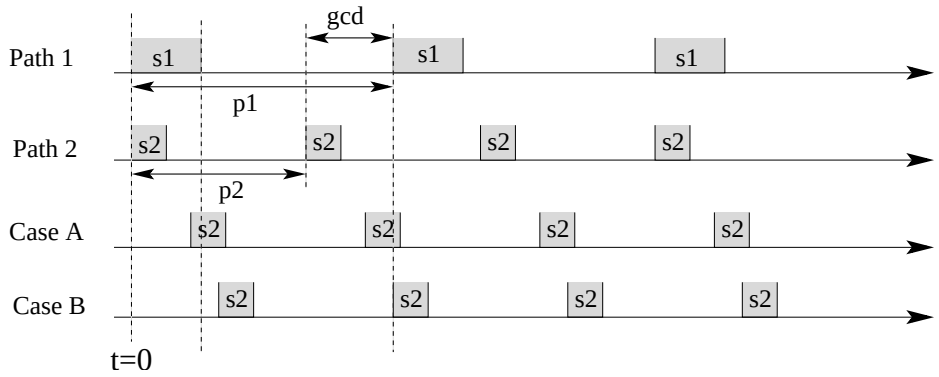


Figure 10.8: Shifting path 2 when $s_1 + s_2 > \gcd(p_1, p_2)$.

To conclude, for two paths to successfully co-exist $gcd(p_1, p_2) < s_1 + s_2$ must be fulfilled and a time shift can only be found during the time $t = \gcd$. If no suitable shift can be found in this time it can never be found.

10.5.2 Computing the gcd

The gcd of two numbers can be computed with the algorithm of Euklid [99]. It is described in pseudo code in Algorithm 1.

Algorithm: Algorithm of Euklid
Input: $a, b \in \mathbb{N}$
Result: Greatest Common Divider of two numbers a and b

```
1 r = a%b
2 while r ≠ 0 do
3 |   a = b
4 |   b = r
5 |   r = a%b
  end
6 return b
```

Algorithm 1: The Algorithm of Euklid

10.5.3 More than two reservation paths

Assume a node already has reservations for more than one path ($n-1$) and receives a request for another one (p_n, s_n). First the gcd of all periodicities, including the new one, has to be computed.

$$\text{gcd}_{\text{new}} = \text{gcd}(\text{gcd}_{\text{old}}, p_n) \tag{10.17}$$

The precondition that the new path can be established is:

$$s_n < \text{gcd}(p_1, \ldots, p_n) - \sum_{i=1}^{n-1} s_i \tag{10.18}$$

However, if Equation 10.18 is true, it does not necessarily mean that all paths fit together without overlapping time slots, as with the case of only two paths. Consider Figure 10.9, where 2 reserved paths (p_1, s_1) and (p_2, s_2) exist and a third (p_3, s_3) should be set up. In case A the

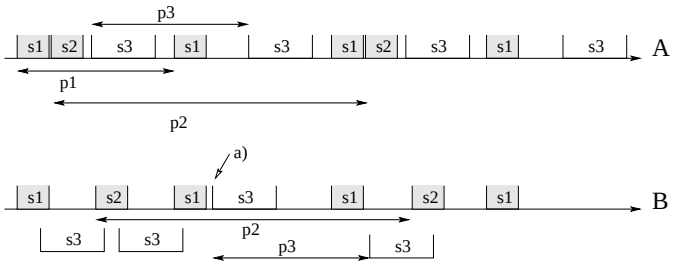


Figure 10.9: Two paths are reserved, a third one is requested. In case A, slots of the third path can fit whereas in case B, slots of third path cannot.

slots of the third path (s_3, p_3) fit together with the already existing ones, in case B they do not. If the request is for the point which is labeled *a* (see arrow), the first slot of path 3 can be accepted, but if the path is set up, the next periodic slot would collide with a slot of path 2.

To further look into this problem, consider first an example with two paths as illustrated in Figure 10.10. The advantage of periods that have a gcd is that the time instances of all slots reserved at one node periodically repeat; the first slots of both paths exactly overlap and also the third slot of path 1 and the fourth slot of path 2, as well as the sixth slot of path 1 and the eighth slot of path 2. Therefore, it would be sufficient to look if slots overlap until the next overlapping instance, that is until the third slot of path 1 and fourth slot of path 2, or the *smallest common multiple* P_{scm}.

Assume that the xth slot of path 1 starts at the same point of time as the yth slot of path 2.

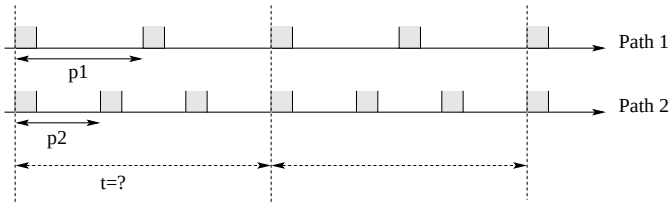


Figure 10.10: Repeating schemes of periodicities with a gcd. Time slots overlap periodically with a time t.

This means that $P_{scm} = xp_1 = yp_2$. The periods p_1 and p_2 that have a gcd can be expressed as:

$$p_1 = k_1 gcd(p_1, p_2),\ p_2 = k_2 gcd(p_1, p_2) \text{ with } k_1, k_2 \in \mathbb{N}. \tag{10.19}$$

This leads to the following Equations:

$$\begin{aligned} P_{scm} = xp_1 &= yp_2 \\ \Rightarrow xk_1 gcd(p_1, p_2) &= yk_2 gcd(p_1, p_2) && (10.20) \\ \Rightarrow x = k_2 &\ \wedge\ y = k_1 && (10.21) \end{aligned}$$

Equation 10.20 can therefore be expressed as $k_1 k_2 gcd(p_1, p_2) = \frac{k_2 k_1 gcd(p_1,p_2) gcd(p_1,p_2)}{gcd(p_1,p_2)} = \frac{p_1 p_2}{gcd(p_1,p_2)}$, leading to the period P_{scm}:

$$P_{scm} = \frac{p_1 p_2}{gcd(p_1, p_2)}$$

Now assume a node has reserved slots $s_1, \ldots, s_n$ for n paths with periodicities $p_1, \ldots, p_n$ and a new path is requested with periodicity p_{n+1} which is *different* from any $p_1, ..., p_n$. First the basic requirement i Equation (10.18) has to be fulfilled. Then, P_{scm_k} has to be computed for every tuple (p_{n+1}, p_k), where $k = 1, ..., n$:

$$P_{scm_1} = \frac{p_{n+1} p_1}{gcd(p_{n+1}, p_1)} \ldots P_{scm_n} = \frac{p_{n+1} p_n}{gcd(p_{n+1}, p_n)}$$

All possible slot collisions is covered when no slots overlap in the maximum of these n periods, hence for $T = \max(P_{scm_1} \ldots P_{scm_n})$ a shift of the new slot s_{n+1} must not overlap with any of the slots $s_1 \ldots s_n$.

When p_{n+1} is equal to any $p_1, ..., p_n$, these periods can alternate, hence more paths can be supported. One example of this is when two paths exist with periods 20 ms and 40 ms. If a third reservation with period 40 ms is requested, the two paths with periodicity 40 ms can alternate

even if Equation (10.18) would not be true.

10.5.4 How to shift the individual slots

Until now, all individual slots have been considered as one large time slot. In reality one node reserves several time slots for each path; individual slots for receiving, transmitting and acknowledging as described in Section 3.8. When a node receives a request that is conflicting with another reservation, it can either divide the different slots or it can shift all together. Consider a simple scenario where two chains, A-B-C and D-B-E cross in one node B, A-B-C (path 1) exists and the path D-B-E (path 2) is requested, see Figure 10.11. Assume that the request for

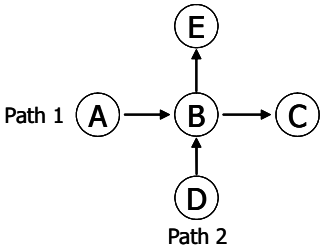


Figure 10.11: Path 1 (node A, B and C) is set up and path 2 (node D, B and E) is requested.

the acknowledgment of the path D-B-E is conflicting with slots of the path A-B-C in the cross node B. This is illustrated in Figure 10.12, where the slot assignment in node B is shown. On top, the reserved slots from path 1 are shown: Rba where node B receives from node A, Sbc where node B forwards the real-time data packet to node C and Abc where node B receives an acknowledgment from node C. The second request is shown at the bottom, where Rbd is the receive slot at node B from node D, Sbe is the send slot where node B *should* send to node E and Abe is the acknowledgment slot where node B should receive an acknowledgment from node E. As illustrated, the acknowledgment slot conflicts with the receive slot at node B from the sender node A.

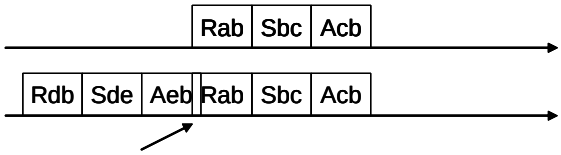


Figure 10.12: Example of overlapping slots. The acknowledgment of path 2 is conflicting with slots assigned for path 1 in the cross node B.

The main possibilities to shift time slots are: 1. Shift only the conflicting slot, in the example shift only the Abe slot. 2. Somehow group the slots and shift them as a unit. The first option has

the advantage of a small additional delay introduced by the shifting mechanism. It is a suitable option when not many paths exist. This is illustrated in Figure 10.13 case 1. As seen, shifting the acknowledgment only results in a small available time slot in between the two reservations, which is hard to use for another flow. This can result in that a new reservation is rejected. For option 2, there are some possibilities as to how to group the slots. The most obvious option is to treat all individual slots together and shift them as one, illustrated in Figure 10.13 case 2. However, when many nodes are involved in the reservation, that is, the paths are long, it is difficult to shift all slots, if possible at all.

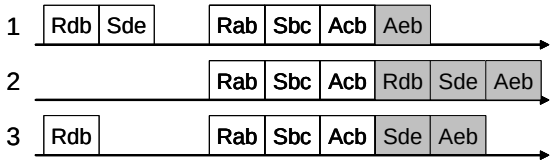


Figure 10.13: Possible solutions for shift of time slots.

Therefore, another option is introduced where individual slots at any node are categorized in two different groups: *receiving* and *transmitting* slots. For node B this means that for the path A-B-C, the slot where node A transmits (or when node B receives from node A) comes into the receive group. The transmission to node C and the acknowledgment sent from node C to node B belong to the transmit group of slots. This categorization is done as node B, or any node, can control the transmitting slots, but not the receiving slots. When node B discovers that the acknowledgment cannot fit, it shifts the send and acknowledgment slot together according to the gcdShift function, see Figure 10.13 case 3. This method of shifting time slots removes very small time slots that are reserved separately and also minimizes the signaling as not all nodes in the chain must change the reservation. The scheduling is kept local and simple.

10.6 Probability of needed time shift

To see how often a reservation must be shifted, this section investigates the probability that the requested time slots directly overlap with existing ones. The probability that two paths overlap is analytically derived and then compared with a simulative investigation. Also, the probability that a shift is needed with more paths is addressed.

10.6.1 Two paths overlapping probability

To find the probability that two paths overlap, the same parameters as in Figure 10.2 are used: Path 1 (p_1, s_1) exists and path 2 (p_2, s_2) is requested with equal probability within any period of

path 1; the first slot of path 2 is requested at a uniformly distributed time t within $[kp_1, (k+1)p_1]$. Again, for simplicity $k = 0$. This leads to a probability density function (pdf) for t: $f(t) = \frac{1}{p_1}$. As shown in Figure 10.14, the requested t for slot 1 of path 2 can directly overlap with the slots

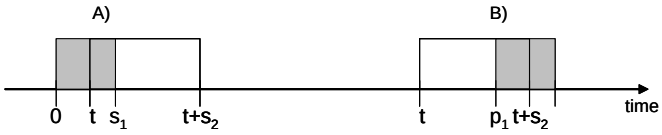


Figure 10.14: Slot 1 of path 2 (white slot) overlap with A) slot 1 and B) slot 2 of path 1 (grey slots).

of path 1 (path 1 has grey slots). Case A) shows when slot 1 of path 2 overlaps with slot 1 of path 1 and case B) shows when slot 1 of path 2 overlaps with slot 2 of path 1. Therefore slot 1 of path 2 overlaps with slot 1 of path 2 if t occurs within the interval $[0, s_1]$. Hence $P[0 \leq t \leq s_1] = \int_0^{s_1} \frac{1}{p_1} dt = \frac{s_1}{p_1}$. Similar, slot 1 of path 2 overlaps with slot 2 of path 1 if t is within the interval $[p_1 - s_2, p_1]$, giving the probability of overlap as: $P[s_2 \leq t \leq p_1] = \int_{p_1-s_2}^{p_1} \frac{1}{p_1} dt = \frac{s_2}{p_1}$. The total probability that slot 1 of path 2 overlaps with path 1 is:

$$P[shift] = \frac{s_2}{p_1} + \frac{s_1}{p_1} = \frac{s_1 + s_2}{p_1}$$

When $p_1 = p_2$, this is the total probability that a shift is needed. When the periods are different, additional slots can overlap. First, assume that p_2 is a fraction of p_1, i.e. $xp_2 = p_1$. Here p_2 is also the gcd of the two periods. Slot 1 of path 2 overlaps with slot 1 or slot 2 of path 1 with the probability given in Equation 10.6.1. Additional to this, there are $x - 1$ slots of path 2 that can overlap. This occurs when slot 1 of path 2 is requested at a t within $[n\frac{1}{x}p_1 - s_2, n\frac{1}{x}p_1 + s_1]$, where $n = 1, ..., x - 1$. Following the same calculations as for the case for slot 1 of path 2 above, the probability that any other slot but slot 1 of path 2 overlaps can be expressed as: $P[n\frac{1}{x}p_1 - s_2 \leq t \leq n\frac{1}{x}p_1 + s_1] = \frac{(x-1)s_1+(x-1)s_2}{p_1} = \frac{(x-1)(s_1+s_2)}{p_1}$. Adding slot 1 of path 2 gives the total probability of a needed shift as:

$$P[shift(xp2 = p_1)] = \frac{xs_1 + xs_2}{p_1} = \frac{s_1 + s_2}{p_2}$$

If the second periodicity would be larger than the first, $p_2 \geq p_1$ the same formula holds, but instead of p_2 in the divider it is p_1. As the period which is smallest also is the gcd, the general formula for probability of overlap when the periods are fractions can be formulated as follows:

$$P[shift] = \frac{s_1 + s_2}{min(p_1, p_2)} = \frac{s_1 + s_2}{gcd(p_1, p_2)} \tag{10.22}$$

The last case that must be covered is when the periods have a gcd but are not fractions, that is $xp_1 = yp_2$. Still, the first slot of path 2 arrives at time t within $[0, p_1]$. Different is that not only can the slots of path 2 overlap with slot 1 and slot 2 of path 1 as in the previous cases, they can also overlap with later slots of path 1. This occurs when slot 1 of path 2 is within the intervals $[n\frac{p_1}{y} - s_2, n\frac{p_1}{y} + s_1]$, where $n = 1, ..(y-1)$. This together with the probability of overlap of slot 1 of path 2 results in the same formula as above, see Equation (10.22).

This is illustrated with an example where $2p_1 = 3p_2$. Consider Figure 10.15, where first of all path 1 is shown at the line marked *Path 1* with three slots (grey) marked with their respective number. Slot 1 of path 1 again occurs at reference time 0. Further, Figure 10.15 illustrates three different cases of the requested t for slot 1 of path 2. *Path 2* shows when the requested t, hence slot 1 of path 2 (also grey), is also at the reference time point 0. Here, slot 1 of path 2 overlaps with slot 1 of path 1 and slot 4 of path 2 overlaps with slot 3 of path 1. *Path 2a* shows in white where slot 1 of path 2 *cannot* start (i.e. t) for slot 2 of path 2 not to overlap with slot 2 of path 1. *Path 2b* shows where slot 1 *cannot* start for slot 3 of path 2 not to overlap with slot 3 of path 1. The probability of a time slot overlap for the above described cases are: *Path 2* as discussed in

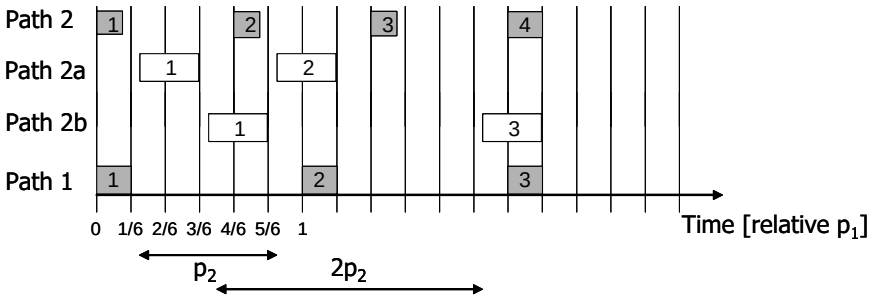


Figure 10.15: Three cases for overlap when $2p_1 = 3p_2$.

previous cases above has a probability of $\frac{s_1+s_2}{p_1}$. The second case, *Path 2a* has an overlap of time slots when slot 1 of path 2 starts within the interval $[1/3p_1 - s_2, 1/3p_1 + s_1]$. Finally, the third case, *Path 2b* has an overlap of time slots when slot 1 of path 2 is within $[2/3p_1 - s_2, 2/3p_1 + s_1]$. This results in the total probability of overlap as $\frac{s_1+s_2}{p_1}$ + $P[2/3p_1 - s_2 \leq t \leq 2/3p_1 + s_1]$ and $P[1/3p_1 - s_2 \leq t \leq 1/3p_1 + s_1]$, which calculated as above gives the total probability of a needed shift as:

$$P[shift(3p_2 = 2p_1)] = 3\frac{s_1 + s_2}{p_1} = \frac{3(s_1 + s_2)}{p_1}$$

This example illustrates that the fraction $\frac{p_1}{3}$ is the gcd between the two periods ($x = 3$), hence Equation (10.22) holds for all cases with two paths.

10.6.2 More than one path

Now, assume that n reserved chains exist and a request for a $n+1$ paths is arriving, all with periodicity p_1. The probability that this reservation must be shifted is then:

$$P[shift] = \frac{s_{n+1}}{p_1} + \frac{\sum_{k=1}^{n} s_k}{p_1} = \frac{s_{n+1} + \sum_{k=1}^{n} s_k}{p_1} \tag{10.23}$$

When the periods are different, the same discussion as for two paths lead to a general formula as:

$$P[shift(path_{n+1})] = \frac{s_{n+1}}{gcd_{n+1}} + \frac{\sum_{k=1}^{n} s_k}{gcd_{n+1}} = \frac{s_{n+1} + \sum_{k=1}^{n} s_k}{gcd_{n+1}} \tag{10.24}$$

where $gcd_{n+1} = gcd(p_1, .., p_{n+1})$. Unfortunately, this does not include the shifting that needs to be done due to bad slot location which is described in Section 10.5.3. This is not possible to calculate analytical, hence only an estimate of a lower limit of the probability of overlapping slots can be achieved with these formulas.

10.6.3 Probability of shift – cross scenario

For the numerical investigation, the scenarios in Figure 10.16 are used. Three different scenarios are shown with two paths crossing each other in one node B. *Cross I* scenario has three nodes in each chain, source node A1 sends to destination node C1 and for the second path source node D1 sends to destination node E1. *Cross II* scenario has two more nodes in each chain; node A2 sends over A1, B, and C1 to destination node C2 and node D2 sends to destination node E2 over D1, B and E1. Similar, *Cross III* scenario has 7 nodes in each chain: Node A3 sends over A2, A1, B, C1, C2 to node C3 and node D3 sends to destination E3.

The probability that cross node B must shift one reservation is analyzed using combinations of periods 20, 30 and 40 ms and a packet size of 64 bytes. Transmission rates used are both 1 Mbps and 11 Mbps, resulting in a transmit/receive time slot size of $S_t = 1.33$ ms and $S_t = 0.86$ ms respectively. In Cross II scenario, the cross node B has for the first reservation allocated approximately 6 ms for 1 Mbps and 4 ms for 11 Mbps (4 S_t+ 1S_a[1] for slots: receive node A1, receive, send, implicit ack when C1 sends to C2 and explicit ack at C1). In Cross III scenario approximately 8.66 ms for 1 Mbps and 5.7 ms is allocated in the cross node (2 more S_ts is allocated for 2 hop protection: receive at nodes A2 and A1, receive, send, implicit ack when C1 send to C2, implicit ack at C1 when C2 sends to C3 and explicit ack at C2). The probability that slots overlap for different combinations of periods are calculated with Equation 10.22 and shown in Table 10.3.

[1] Approximate transmission time of one MAC acknowledgment 0.6 ms

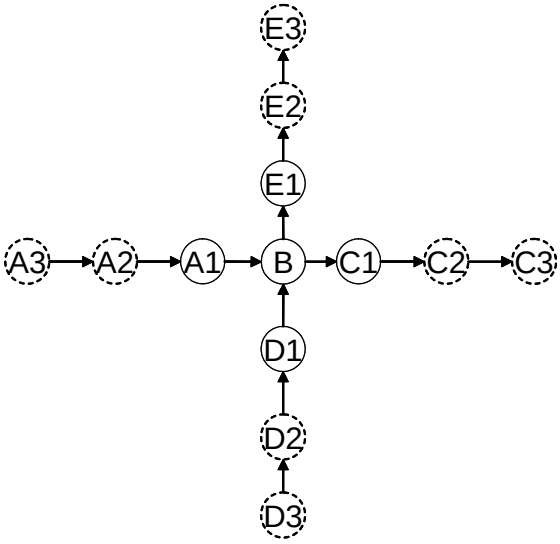


Figure 10.16: Cross configurations I with 3 nodes, II with 5 nodes and III with 7 nodes in each chain crossing at the middle node B. The path from node Ax to node Cx exists and the path from node Dx to node Ex is requested.

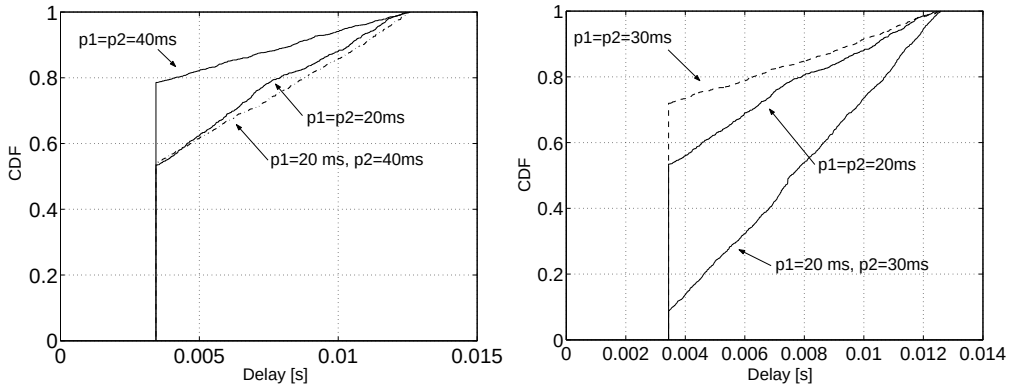
Period p_1,p_2 [ms]	20,20		30,30		40,40		20,40		20,30	
Transmission Rate [Mbps]	1	11	1	11	1	11	1	11	1	11
Cross II probability of shift	0.6	0.4	0.4	0.27	0.3	0.2	0.6	0.4	>1	0.84
Cross III probability of shift	0.86	0.6	0.57	0.4	0.43	0.3	0.86	0.6	>1	>1

Table 10.3: Probability that path 2 directly overlaps with any slot of path 1 for packet size 64 bytes over 1 and 11 Mbps channel.

The probability that a direct shift of the second path is needed, ranges from 0.2 - >1. First of all, a probability larger than one means that only one path can be accepted as $s_1 + s_2 > gcd(p_1, p_2)$. A lower probability directly tells us if more than two paths can fit at all. However, a lower probability leads to an increased probability of unusable inter slot space. If the time slots of the second path must be shifted, the inter slot space will be smaller and the probability that more paths can be accepted is higher. Hence, both these aspects must be considered when looking at the shifting probability. Finally, the probability gives an idea of the final end-to-end delay. The higher the probability of a needed shift of a reservation is, the higher is the probability that the end-to-end packet delay is higher.

To compare the analytical results in Table 10.3 a simulation-based study of the Cross scenarios with same parameters as above is performed. 1000 simulations for each combination of the two periods p_1 and p_2 are performed, using NS-2 [90]. The second path is requested at a

uniformly random t within $[0, p_1]$. The resulting end-to-end delay of path 2 is measured (path 1 has always the same delay). After the shift, every packet of path 2 has the same delay, thus the simulation could be interrupted as soon as the second path is set up. Figure 10.17 shows the simulation results for the Cross II scenario with the CDF over the end-to-end packet delay of path 2. The delay with no shift is the first step of the function. First of all, the calculations of the probability of a needed shift are the same as with the calculations, e.g. $p_1 = 20$ ms, $p_2 = 40$ ms, 11 Mbps channel has a 0.58 probability of no shift and the approximate calculated value is 0.6. Also, the probability of shift increases with smaller gcd, see Figure 10.17(b) where combination $p_1 = 20$, $p_2 = 30$ ms has a much larger shift probability than other combinations due to the small gcd of 10 ms. Further, the maximum shift that the gcdShift function introduces is approximately 3 times the original allocated time in the node, which is due to how the slots are shifted as described in Figure 10.13. As these values are anyhow under the ITU-T recommendation (also the case for Cross III not shown here) of one-way transmission time [69], the increased end-to-end delay is not an issue here.



(a) 20 ms, 40 ms (b) 20 ms, 30 ms

Figure 10.17: CDF for cross scenario II of path 2 for different combinations of periods.

To conclude, the probability of a direct shift gives an idea of how many paths that can directly fit in a network. It also indicates how the theoretical maximum number of established reservations can differ from the actual number; a lower probability of a shift result in a higher probability of unused inter slot space. This will also show in simulation results of a larger simulation, which is described after the protocol implementation description in the next section.

10.7 Protocol implementation

This section describes how the gcdShift function is implemented in the DARE protocol. For this, a general knowledge about the set-up mechanism and the MAC layer frames is needed (Section 6.3). The description here starts when a node in the path receives the RTR.

Upon arrival of the RTR at any node, this node first checks the receive slot(s). If this slot is non-conflicting, the node continues by checking for a suitable transmit slot, possibly shifting the transmit slot in time if necessary. Using the gcdShift function for all periods already accepted and the requested one, the node can find a suitable shift. Periods that do not have any common divider, e.g. prime number based periods, are not considered. If a node detects a receive slot to be conflicting, it transmits an UTR message back to the node that generated the receive slot, suggesting one, or more, new transmission slot(s) that would be suitable. For this, the time information fields of the UTR are used, see Section 6.3.1. If the new transmission instance can be fulfilled, old preliminary reservations are deleted, a new RTR is generated and also new preliminary reservations.

Since two hop protection is applied in the reserved path, see Section 6.4, receive slots at preceding nodes must also be scheduled. If a conflict of a receive slot two hops back is discovered, the UTR is sent three hops back. The **Flag** field is used for this purpose. This field information can have the value 4, 5 or 6. If a node receives an UTR with Flag equal 4, it knows that the UTR is intended for itself. This node must then re-schedule its transmission slot belonging to the reservation. It knows which reservation it is by reading the source and destination address fields in the UTR and compare these with entries in its reservation table. When a node receives an UTR with a value higher than 4, it reduces the Flag value by one and then sends it back to the node preceding it in the chain. If no new receive or transmit slots can be scheduled, the reservation request is rejected and the flow is blocked. The approach may not be globally optimal, but the decisions are kept local and simple.

The following three figures describes state diagrams for the full DARE set-up mechanism. In Figure 10.18, a node receives an RTR from a preceding node and checks whether or not it can fulfill the request for a receive slot. The MAC layer receives a reservation request for a receive slot (Res Recv Request). If this is ok, it makes an entry in the reservation table and forwards the RTR to a queue. This queue follows the standard DCF, which specifies a packet queue used to store packets after routing decision has been made and the LLC layer indicates its availability. If the request is not ok, it generates an UTR and sends it back in the chain to the node which must change its transmit slot. Figure 10.19 describes when a node after accepting the request for the receive slot, receives the RTR from the internal queue above the MAC layer and checks for a transmit slot. First, the MAC layer receives the request from the queue (Res Send Request). At

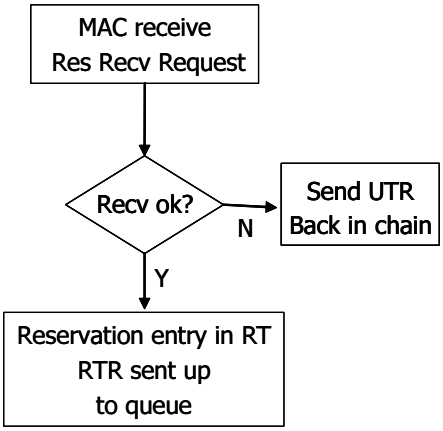


Figure 10.18: Shift function algorithm; node MAC layer receiving the RTR from preceding node and checking the request for the receive slot.

the MAC layer, the RTR is initiated by alternating the fields with new MAC addresses. If the RTR can directly be transmitted (after sifs) it is transmitted. If the direct send slot is conflicting, the node checks the reservation table for a suitable shift. If found, it starts a shift timer and sends the RTR back to the queue. When the timer expires, the RTR is retrieved from the queue and is again initiated; new information is entered in the time information fields. A control check is performed to see that the new send slot is ok and the node makes an entry with preliminary status and sends the RTR to the next node in the path. If a suitable time shift cannot be found, the RTR is deleted and no reservation is performed.

The last state diagram is shown in Figure 10.20, which is the state diagram for a node that receives an UTR. Upon receiving the UTR, a node releases all reservations it has for the particular reservation (based on source and destination addresses). It then checks if the Flag fields has the value 4. If not, it reduces the Flag value by 1 and sends it back in the chain towards the source of the application flow. If the Flag is 4, it checks the time information fields for new suggested transmit time slots. If none is ok, no new RTR is initiated and no time slots are reserved. If the suggested time slot is ok, the node stores the UTR in the queue (with Flag value 2, i.e. it will be read as an RTR) and starts the shift timer. When the shift timer times out the node goes into state Res Send Request shown in Figure 10.19.

10.8 Multiple reservations – Simulation study

This section describes a simulation performed to investigate the impact different packet sizes and periods have on the number of possible established paths at one AP. The simulation model

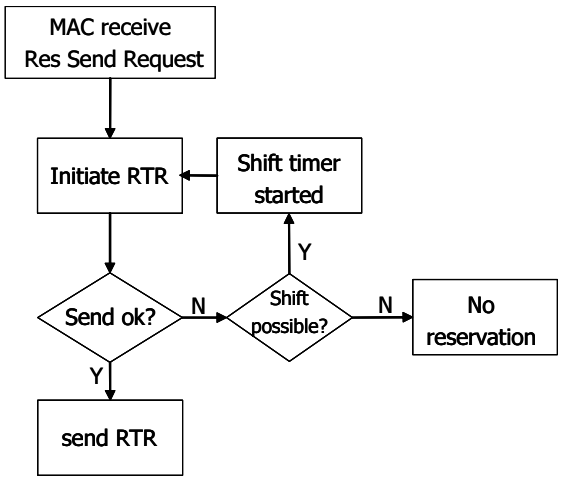


Figure 10.19: Shift function algorithm; node MAC layer receiving RTR from queue and checking the request for the send slot.

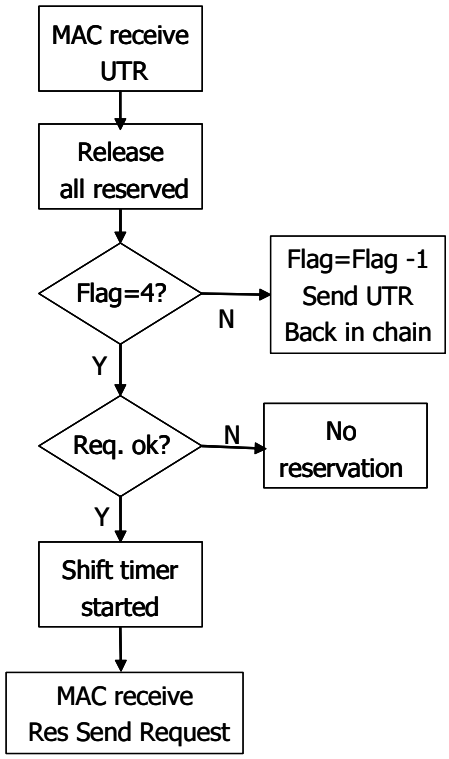


Figure 10.20: Shift function algorithm for shifting receive slot; UTR transmission scheme.

is described in Section 10.8.1, followed by simulation results, covering the number of established paths when packet sizes and periods are varying in Section 10.8.2.

10.8.1 Simulation model

In this scenario, the maximum number of established reservations at one AP is investigated for different periods and time slot sizes. Source nodes transmitting to the AP are added until the maximum number of flows is reached at the AP. 400 simulations using NS-2 [90] are performed for each combination of periods and time slot sizes. The source nodes transmit over either a 2 or 3 hop path to the AP. They are located such that no intermediate node is involved in more than one real-time flow. No longer paths are investigated as the AP will due to basic protection scheme not reserve any more time slots than up to two hops away. Values of packet size and periods are given directly under each result section. One scenario with 12 source nodes and paths of 2 hops is illustrated in Figure 10.21. The scenario where all paths have 2 hops is called Star I and the scenario where all paths are with 3 hops is called Star II.

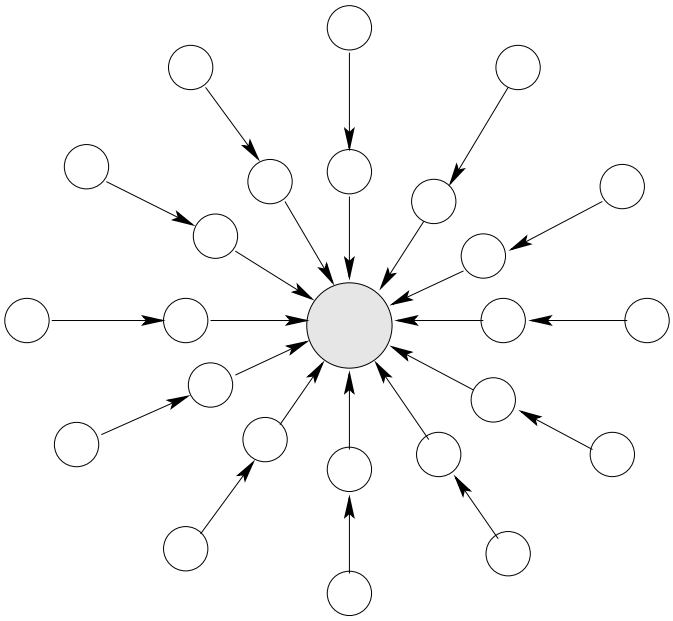


Figure 10.21: Star I: Sending to an access point via two hops.

10.8.2 Simulation results

First, the packet size is varied, which results in varying time slot size. After these results are presented, the period is varied.

Varying packet Size

The real-time traffic's period is 100 ms and packet sizes 144, 320, 512, and 1024 bytes is investigated. The outcome is the average number of paths that can be accepted, given in Table 10.4.

Packet size	Hops per path	
	2	3
144	8.4	6.2
320	7.7	5.9
512	6.4	4.7
1024	2.3	1.3

Table 10.4: Number of established reservation paths, different packet sizes.

The average decreases with increasing hop count. Obviously, the AP must reserve more time with a longer path since all nodes reserve slots up to two hops away. Not surprisingly, larger packets, thus larger time slots, also decreases the average number of established paths. But, interesting is that, although not shown here, the maximum number of successfully reserved paths can vary a lot, but the average number of successfully reserved paths does not. The maximum number of paths for packet sizes 144 and 320 bytes were actually larger than 12 for both two and three hop paths. The maximum number of possible paths for 512 bytes is 8 for the three hop scenario. Due to inter-slot space between accepted reservations that could not be used, the average number of paths do not differ that much (except for 1024 bytes).

Varying period

The packet sizes and periods are chosen from the G. 729 CODEC [68]. The combinations are (period, packet size): (20 ms, 64 bytes), (30 ms, 74 bytes) and (40 ms, 84 bytes). The results are given in two instances, the range of the numbers of established paths and the average of the number of established paths, see Table 10.5. If only one periodicity is given in the first column this means that all paths use the same periodicity. If two periodicities are given this means that every second source node uses p_x and every second uses p_y. The most obvious result from the table above is the impact of the gcd. The combination of 20 ms and 30 ms periods result in the smallest gcd, hence the least number of paths can be reserved at the AP. Generally these results confirm that the smaller the relation $\frac{\mathrm{gcd}}{\mathrm{slotlength}}$ is, the less reservation paths can be established. Interesting is that the transmission rate has not as large impact as expected. Although the rate is increased with 11, the range do not increase that much. Consider the case where periods 20 and 40 are used. The average number of paths for the Star I scenario only increases with 1.5 path.

Periodicities [ms] **and transmission speed**	**Star I** Nr. of paths		**Star II** Nr. of paths	
	range	average	range	average
p=20, 1 Mbit/s	4-6	4.93	3-4	3.27
p=20, 11 Mbit/s	6-8	6.99	4-6	4.93
p=40, 1 Mbit/s	7-11	8.77	4-7	5.67
p=40, 11 Mbit/s	9-12	10.58	6-8	7.63
p_x=20, p_y=40, 1 Mbit/s	4-6	5.26	3-4	3.8
p_x=20, p_y=40, 11 Mbit/s	6-8	6.7	4-6	5.24
p_x=20, p_y=30, 1 Mbit/s	2-3	2.99	1-2	1.98
p_x=20, p_y=30, 11 Mbit/s	3-4	3.99	2-3	2.98

Table 10.5: Number of established reservation paths, different periods and packet sizes.

The same scenarios were simulated again but this time the source nodes randomly chose their periodicity. Table 10.6 shows the results for these simulations. In the first column the periodicities are given from which a source node could choose from.

It can be seen that the ranges are bigger than in Table 10.5. If for example the periodicities

Periodicities and transmission speed[ms]	**Star 1** Nr. of paths		**Star 2** Nr. of paths	
	range	average	range	average
p_x=20 p_y=30, 1 Mbit/s	2-7	3.47	2-5	2.55
p_x=20 p_y=30, 11 Mbit/s	4-7	4.24	2-6	3.19
p_x=20 p_y=40, 1 Mbit/s	4-8	5.72	3-6	3.96
p_x=20 p_y=40, 11 Mbit/s	6-8	7.33	3-6	5.44
p_x=20 p_y=30 p_z=40, 1 Mbit/s	3-6	3.5	2-5	2.8
p_x=20 p_y=30 p_z=40, 11 Mbit/s	4-8	4.55	2-6	3.57

Table 10.6: Nr. of established reservation paths with randomly chooses periodicities.

20 ms and 30 ms can be chosen the number of established reservation paths ranges between 2 and 7 (Star 1, 1 Mbit/s). This is due to that possibly all paths choose one period, thus the gcd is higher, allowing more paths to be accepted. The average values from Table 10.6 are visualized as graph in Figure 10.22 and Figure 10.23. The values for the cases where only one periodicity can be chosen are taken from Table 10.5. In conclusion, to get as many set up paths as possible, it is an advantage to only allow some periods and packet sizes in the network. The relation $\frac{\text{gcd}}{\text{slotlength}}$ should be as big as possible to achieve the best scalability results.

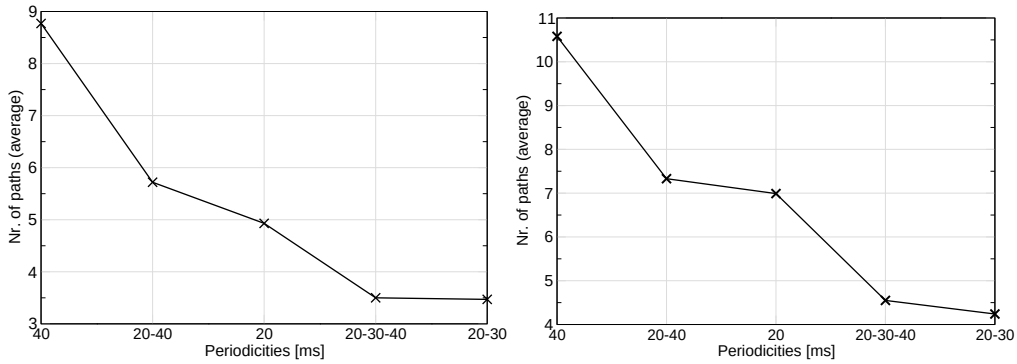


Figure 10.22: Star I, 1 Mbps (left) and 11 Mbps (right): Established paths for random periodicity combinations.

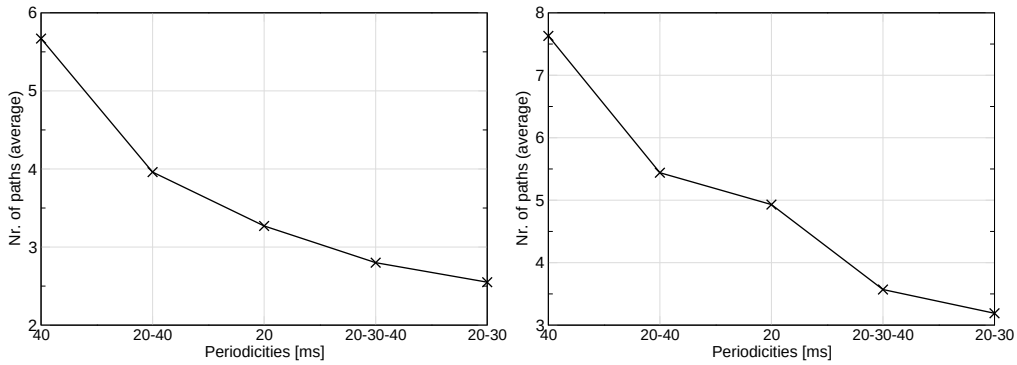


Figure 10.23: Star II, 1 Mbps (left) and 11 Mbps (right): Established paths for random periodicity combinations.

10.9 Limits for multiple reservations

The greatest common divider The precondition for the possibility to have multiple coexisting reservation paths is that the periodicities of these paths must have a gcd that is greater than the sum of the slot lengths. The question is what happens if two periodicities are used that are relatively prime, e.g. 20 ms and 33 ms. The answer is very simple; they exclude each other. The path which is set up first wins and the other path does not have a chance to be built up before the first path is released. To ensure that multiple reservation paths can co-exist it makes sense to allow only periodicities that have a certain common divider.

Slot lengths The gcd gives an upper limit for the theoretical maximum number of possible reservation paths in a node that depends on the different slot lengths. The introduced shifting mechanism only makes sense if comparable slot lengths are used. Consider the following exam-

ple. Assume one path with periodicity $p_1 = 20$ ms sending 64 byte packets and a second path with periodicity $p_2 = 100$ ms sending 512 byte packets. The gcd of the periodicities is 20 ms. For a transmission speed of 1 Mbit/s the transmission time of a 512 byte packet results in 4.8 ms. When all individual time slots are reserved and 2 hop protection is used, the slot sizes is much larger than the gcd and the other path cannot be reserved. Since one time slot is much larger than the other, it might make sense to force a fragmentation of the larger packet. Hereby, more flows might be accepted.

Randomness of slot start Another limit is the randomness of the time for which the new path is requested. Remember that if the requested path fits together with the existing ones everything is left as it is and no shift is done. This can leave gaps between the slots. The ideal case is that the newly requested slot starts right after an existing slot ends. In a disadvantageous case this gap is much bigger, but just not long enough for another slot to be able to fit between them, i.e. the time between these slots cannot be used for a newly requested path. This is shown is Figure 10.24. As a conclusion the theoretically possible maximum number of paths might not necessarily be reached.

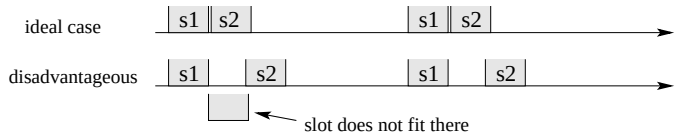


Figure 10.24: Slots placed ideally and disadvantageous.

Nodes without knowledge Assume there is no intermediate node between such two nodes so that they cannot communicate with each other. Such a situation is shown in Figure 10.25. Node B and F cannot communicate with each other but they are in each other's interference range. As a conclusion they do not have any information about each other's reserved time slots. If reserved time slots at nodes B and F overlap, collisions and packet losses can occur frequently (depending on the periods at which the time slots overlap). There is no way to solve this problem. The only option is to release one of the two paths.

Impossibility of shifting due to other reservations It can be possible that the actual shifting of a *send* slot has to be done up to three hops before the node where the conflict is discovered. If this node cannot shift the slot as requested, possibly due to other reservations, the reservation is not set up.

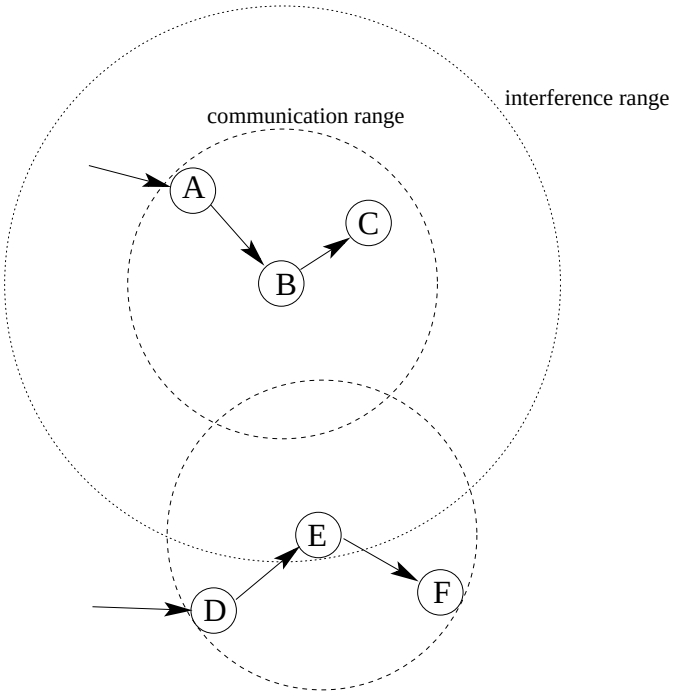


Figure 10.25: Node B cannot communicate with node E, but can interfere with it.

10.10 Summary

Applications in one network might have different flow characteristics; requested periods and time slot sizes for reservations can differ. If one time slot can co-exist with others already reserved, it is not guaranteed that future slots do not overlap. Such a requested time slot must be shifted in time so all slots are non-overlapping. Since many different periods can exist within one node, this node must be able to schedule all time slot allocations accordingly. In this chapter, first an expression for when time slots with different periods overlap was derived and used in a simple simulative investigation. As a result, a requirement for all nodes to use when deciding on accepting a new flow or not was derived. If the greatest common divider of the periods is larger than the sum of all time slots, it can be accepted. Further, when more than two flows cross in one node, inter slot space can result in less paths accepted than a theoretical maximum. However, if a new requested periodic time slot does not cause any overlapping time slots within the greatest common period of all periods, no time slots will ever overlap, hence the reservation can be accepted. The time slots for a flow is divided into transmitting and receiving slots When a shift of any slot within one of these two groups is needed, they are all shifted together. This chapter also describe the probability that a shift is needed and the protocol implementation of the gcdShift function. Finally, a simulation-based study shows how the allowed periods and packet

sizes affect the possible number of established reservations. Some combination of periods can result in that only 10 % of the offered flows are accepted, whereas other combinations can have almost all accepted.

Chapter 11

Evaluation – multiple real-time flows

After the last chapter, which concluded the design of the DARE protocol, this chapter describes the main evaluation of this thesis. First, the DARE protocol alone is investigated with a simulation study of how different network parameters affect the DARE performance. The results are presented in Section 11.2. After this, DARE is compared with DCF and EDCA in Section 11.3. In both investigations, many real-time flows are present in the network that all request a reservation. The goal is to get a clear image of how DARE performs compared to DCF and if DARE is a better QoS-approach than EDCA.

For both investigations one simulation model is used as a base; each individual simulation then use different tunable parameters. This basic simulation model is presented in Section 11.1.

11.1 Simulation model

All investigations of the DARE protocol are performed with NS-2 and use the same simulation model as base. Each separate investigation has then a set of parameters that are tunable. These are described under the corresponding investigations. In addition, the comparison of DARE, EDCA and DCF also use an additional simulation model, described in the result section itself.

In the basic simulation model, the system consist of n nodes, uniformly distributed on a square area with side s. In the system, there are a APs, which are pre-defined located in a grid fashion with equal distance between them and the area borders. These APs are assumed to function as gateways, e.g. connect the wireless access network to a wired network or to other wireless ones. The network is illustrated in Figure 11.1, here with $a = 4$ and $n = 400$. From the n nodes, n_r are randomly chosen to be sources of real-time traffic. For these flows, a reservation will be requested. The end-destination of these transmissions is an AP. When several APs exist, it is the AP closest to them. A real-time flow sends fixed-size packets of 512 bytes every 100

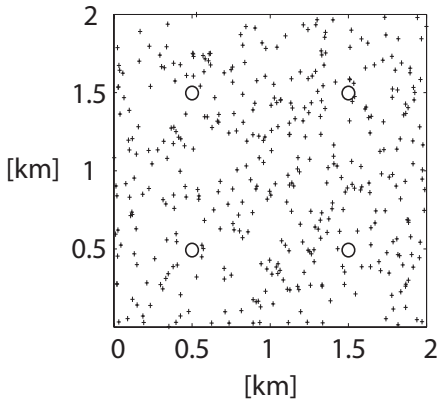


Figure 11.1: Simulation setup: 4 gateways (o), 400 randomly located nodes (+).

ms. When the packet is transmitted on the channel (including UDP, IP, MAC, and physical-layer header and preambles), it has a transmission (or reception) slot $S_t = 4.8$ ms, using the 1 Mbps channel .

The MAC protocols used in the investigations are: DCF as provided by NS-2, EDCA with NS-2 code provided according to [97] and DARE, own implementation. For DARE, the optional feature of two-hop protection is here not considered. The rationale is to investigate the reservation protocol in its simplest modus, to get a fair comparison with EDCA and DCF or even to test DARE the most. Therefore, also in simulations with EDCA the parameters are set to give EDCA advantage: The real-time flows are given the highest priority (AC = voice) and in case of background traffic, this has the lowest priority (AC = background). Parameters used for these classes are presented in Table 4.1.

The radio model is according to the NS-2 simulator, which has implemented the physical characteristics from the Lucent WaveLan interface card. The range of reception is approximately 230 meters and the nodes use full transmit power of 100 mW. As a routing protocol, NS-2's implementation of AODV is used. All parameters for DCF and radio model can be found in the NS-2 manual [90]. 200 repetitions for each selected combination of parameters is ran; each repetition for 2000 s simulated time, unless other simulation time stated.

Investigated *metrics* are: 1. the delay of packets from source to access point, 2. the throughput for individual real-time flows, and 3. the amount of slot shifts and blocked flows. These are defined in Section 3.6. Only successfully reserved real-time flows are considered in these metrics (except for blocked flows, of course).

In this simulation, only one period and packet size is used. The previous chapter has shown simulation results for many different periods and packet sizes and how the combination of them

impact the DARE performance. The reason for only choosing one value here is to get an idea of the maximum capacity of DARE, at the same as EDCA is given an advantage at the choice of other simulation parameters (AC = voice, no two-hop protection).

11.2 DARE evaluation

This section describes results of a simulation-based study where the impact of DARE performance for different parameter values is investigated. To achieve this, the number of real-time flows is here fixed, $n_r = 10$. The parameters that are varied are: 1. Area side, $s = 700, 1300, 2000$ m and for some cases 3000 m and 4000 m. 2. AP number, $a = 1, 2, 4$. 3. Nodes $n = 100, 200, 400$. 4. Order of reserved chain, e.g. if a flow is reserved as number 1 or number 9 and its impact on the end-to-end delay.

11.2.1 Distribution of end-to-end packet delay

The end-to-end delay has a distribution (CDF) which follows a step function – each step corresponds to either a shift in time or that the path, or chain, length is different. One example is shown in Figure 11.2, which shows results from one simulation of 10 chains where all chains were successfully reserved.

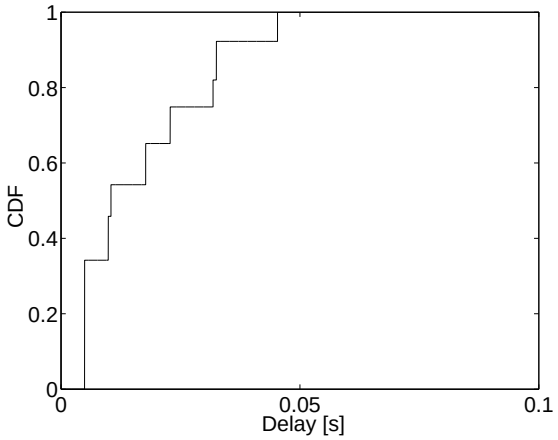


Figure 11.2: CDF of end-to-end delay for 1 simulation where all 10 real-time flows were successfully reserved.

When delays from all simulations are included, the CDF will typically not have such sharp steps; the end-to-end delay of one path can have any value due to a shift. Nevertheless, one flow will *always* have the exact same delay for all its packets after the reservation is set up. This

section continues with a presentation of how the different parameters affect the distribution of the packet delay.

Impact of area side The end-to-end delay of all chains vary with the cell side. Figure 11.3 shows the CDF of end-to-end packet delay in a scenario where there are 4 APs and 400 nodes. In

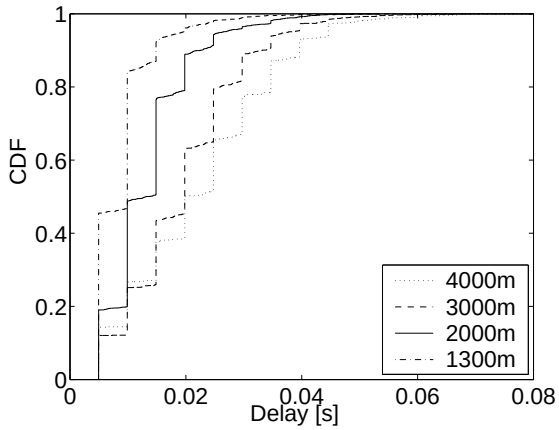


Figure 11.3: CDF of end-to-end delay for different area sides $s = 1300, 2000, 3000, 4000$ m. Number of APs $a = 4$ and number of nodes $n = 400$.

this scenario, the network is always connected; no source node is out of reach from its destination. The end-to-end delay increases with the area side; longer paths are required for successful connections in the network. Consider a traffic requirement of end-to-end delay less than 0.02 s. If $s = 2000$ m, approximately 90% of all packets are below the limit, whereas if $s = 4000$ m, only 50% is below. Another distinct difference is that larger s has more varying path delay; smaller s has not so many varying paths lengths.

Impact of active chains When the number of set-up and active reservation paths grows in the network, the probability that a new one can be accepted directly without a time shift decreases. Figure 11.4 shows the delays for different chains when $n = 400$ nodes, $s = 2000$ m and $a = 1$ AP. Chain 1 means that this is the first reserved flow in the network, chain 5 that 4 others are active and chain 10 that 9 others are active. The chain that is first set up has a less varying end-to-end delay than the chain that is set up last.

Impact of AP number With increasing a, the number of nodes that are involved in several reservations decreases. Figure 11.5 compares Figure 11.4 (shown again in (a)) with the same configuration but where $a = 4$ APs, shown in (b). The individual chains in (b) compared

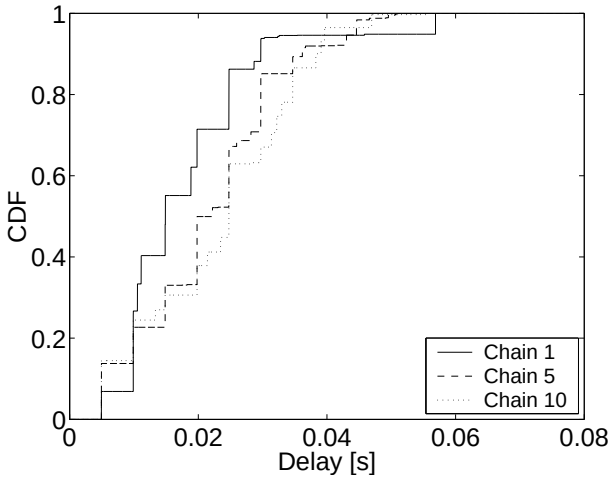
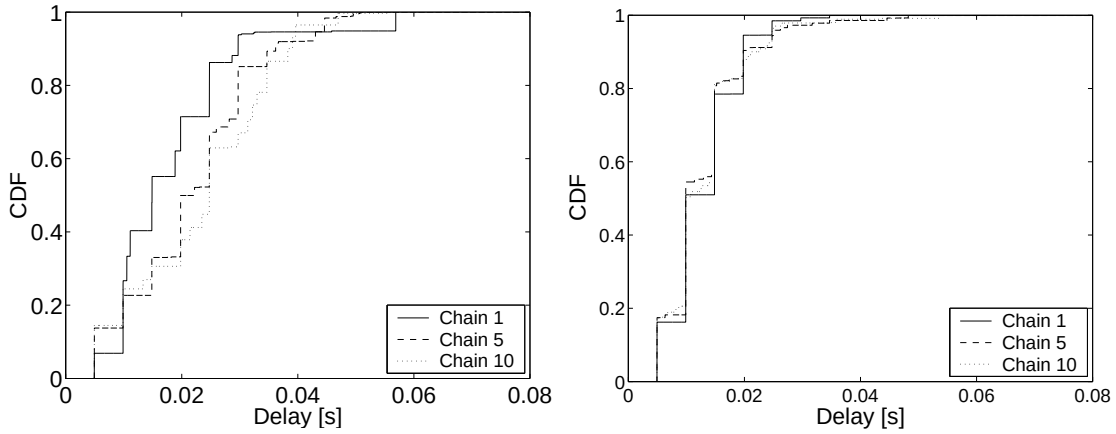


Figure 11.4: CDF of packet delay for chain 1, 5 and 10 with $n = 400$, cell side $s = 2000$ m and $a = 1$ AP.

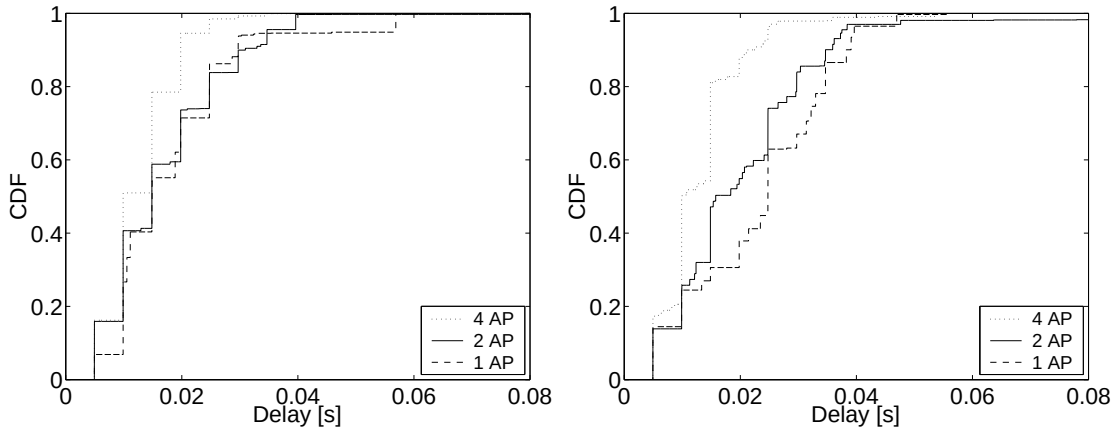


(a) 1 AP (b) 4 APs

Figure 11.5: CDF of packet delay for chain 1, 5 and 10, $n = 400$ nodes, cell side $s = 2000$ m, a) $a = 1$ AP and b) $a = 4$ APs.

with their respective chain in (a) have less varying end-to-end delay. Again, consider a traffic requirement of packet delay under 0.02 s. In (b), chain 1 has in 95% of the cases a end-to-end delay under 0.02 s whereas in (a) this value is 70%. It is even more evident at chains set up later. Further, the difference between the chains is much smaller in (b) than in (a).

Figure 11.6 shows the impact of the AP number more clearly. The CDF of chain 1 (shown in (a)) and chain 10 (shown in (b)) are shown with varying a, other parameters as above. The



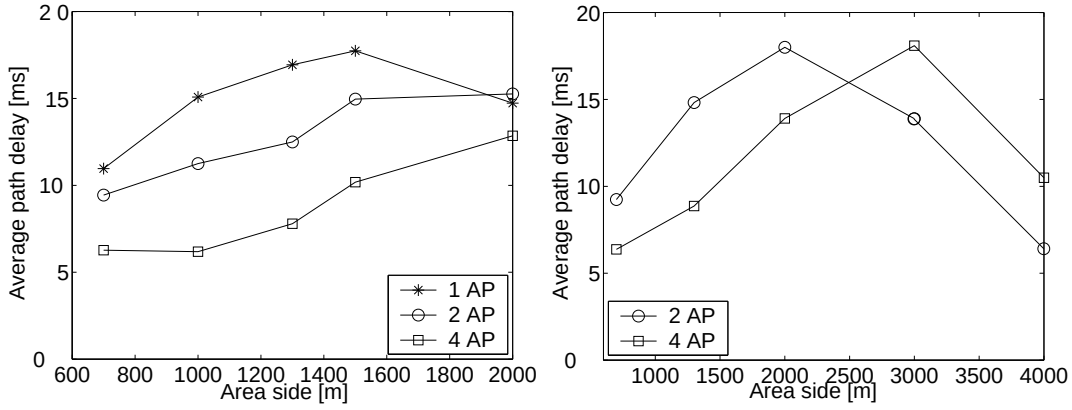
(a) Chains 1 (b) Chain 10

Figure 11.6: CDF of end-to-end packet delay for a) chain 1 and b) chain 10, $n = 400$, $s = 2000$ m and a is varying: 1, 2 or 4.

variation of the end-to-end delay decreases when the number of APs increases for all chain types. Once again, the variation is only between different paths, the delay of all packets transmitted over the same path is constant.

11.2.2 Average path delay

The average end-to-end delay can be seen in Figure 11.7(a). Here, the delay is shown for $n = 100$ and 200 nodes for different area sides s and AP numbers a (Note: different scales on x-axis). With $n = 100$, there is no point in presenting the results for s larger than 2000 m; not enough flows are set up for reasonable delay results. Increasing n to 200 results in a larger coverage area; more nodes can relay transmissions. The behavior of both cases are similar: At first the delay increases with area side, but after a peak decreases rapidly. Only sources that are relatively close to an AP finds a route. Increasing the number of APs can extend coverage. At a fixed area side, e.g.700 m higher AP results in lower average delay. More paths consist of single hop, or rather few hops to reach the destination. The average throughput results in the next section



(a) $n = 100$ (b) $n = 200$

Figure 11.7: Average path delay for (a) $n = 100$ and (b) $n = 200$, AP number $a = 1, 2, 4$ versus area sides s .

confirms these results.

11.2.3 Throughput and unsuccessful reservations

When 10 chains are accepted fully, the total network throughput is 10*$\frac{512*8}{0.1}$ bits/s = 409 kbps. The average throughput over all 200 simulations will not reach this value as some reservations will not be set up. In Figure 11.8 the average network throughput ($n = 200$, $a = 1, 2, 4$) versus the area side s. The curves follow the average delay results; after a certain s, all sources

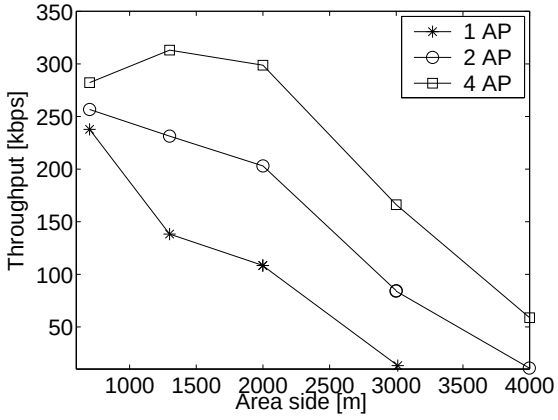


Figure 11.8: Average real-time throughput for $n = 200$, $a = 1, 2, 4$ versus area side s.

do not reach their final destination – the network is not fully connected. When $a = 4$ the

network is connected up to approximately $s = 2000$ m. The maximum average throughput for this connected network is around 320 kbps. This means that approximately one forth of all chains are not successfully set up. Figure 11.9 shows the corresponding average number of unsuccessful reservations. Comparing the maximum throughput with 4 APs when $s = 1300$ m, 20 % of the paths are on average unsuccessful, which corresponds to the throughput results presented above.

The number of unsuccessful paths consist of both the number of *blocked* paths – the paths that could not be allowed due to that the resources were already allocated and the *overlapping* paths – paths that due to the lack of two hop protection are unsuccessful. The sources that do not have coverage are not included in this number. As can be seen, for all AP numbers the number

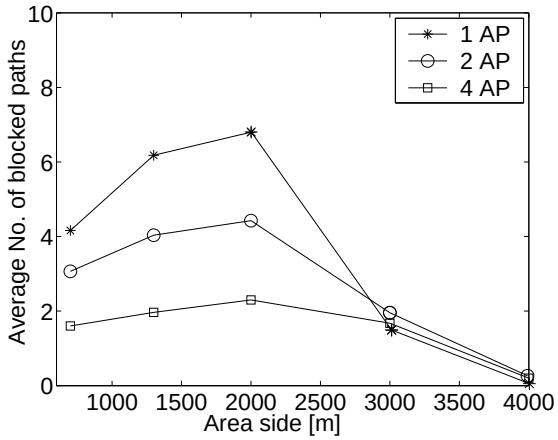


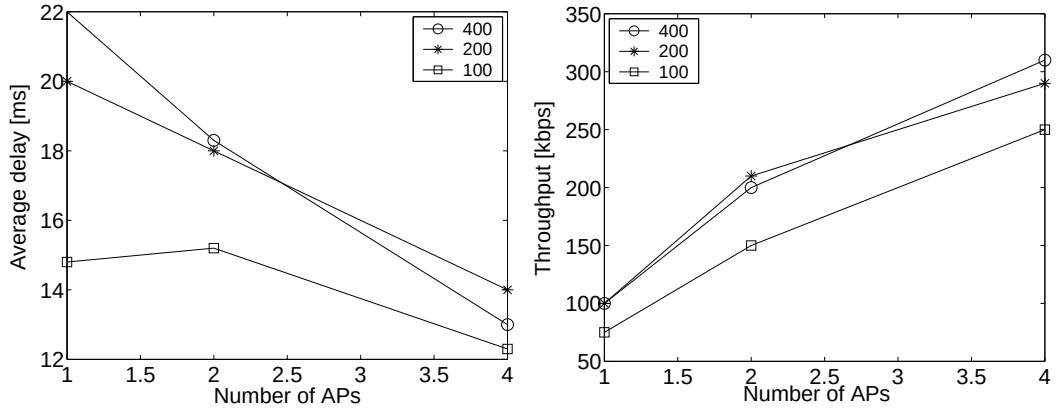
Figure 11.9: Unsuccessful reservations, $n = 200$ and $a = 1, 2, 4$ versus area side s.

of unsuccessful reservations increases until the area side is 2000 m. Thereafter, it decreases rapidly; the network is no longer connected.

11.2.4 Summary – fixed area

The last sections have described how area side, number of APs and nodes affect the achievable delay and throughput with DARE. The outcome is that depending on traffic requirements, some set ups are more suitable than others. Depending on which parameter that is tunable, the optimum network configuration can be reached for each traffic requirement. To summarize all parameters above, here a fixed area side is assumed, $s = 2000$ m; n and a are varied.

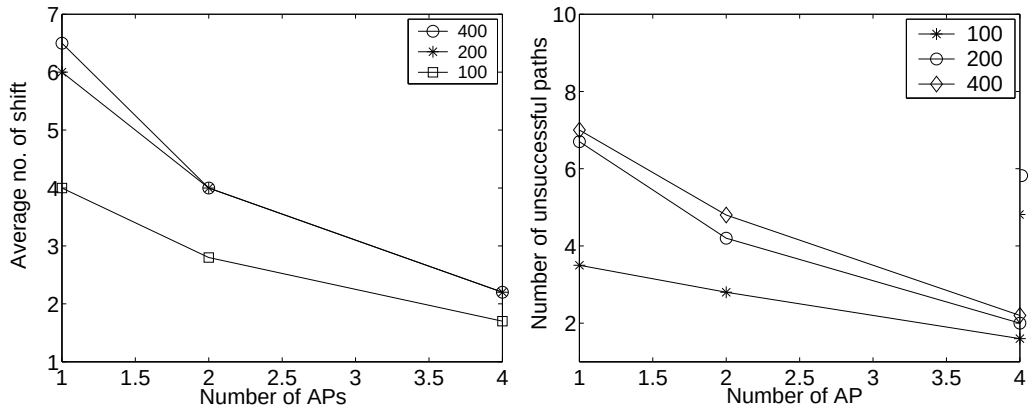
The average delay and throughput for different n versus a can be seen in Figure 11.10. The average delay decreases with increasing a for all n. Further, a higher number of n results in higher delay. Again, this is average path delay over 200 simulations and for low n, not



(a) Average delay (b) Average network throughput

Figure 11.10: Average end-to-end delay and throughput, $n = 100, 200, 400$ versus a.

all sources find a path. This is visible in the throughput results in (b); $n = 400$ has highest throughput. Figure 11.11 (b) shows the number of unsuccessful paths. Here it becomes clear



(a) Shift (b) Unsuccessful paths

Figure 11.11: Average no of shifts in set-up phase and average number of rejected paths for AP number $a = 1, 2, 4$ and number of nodes $n = 100, 200, 400$.

that for $n = 100$ and $n = 200$, the network is not totally connected. At the same time as $n = 100$ has high number of unsuccessful paths, the number of shifts is lower, shown in Figure 11.11 (a). Thus, as not as many shifts is performed for $n = 100$ as with $n = 400$, but $n = 100$ has higher number unsuccessful paths, these must be due to a none connected network. From this configuration, it can be distinguished that in a scenario where $s = 2000$ m, a suitable a is 4 and

n is 400. These are the parameters that will be used for the comparison with DCF and EDCA in the next section.

11.3 Comparison of DARE, DCF and EDCA

The parameters used for the comparison are derived in the last section: a fixed cell side $s = 2000$ m, fixed number of APs, $a = 4$, and fixed number of nodes, $n = 400$. Further parameters used in this comparison are (apart from the basic model described in Section 11.1): Number of real-time source nodes is varied, $n_r = 5, 10, 15, 20$. Background traffic exist, a number of sources is chosen randomly, each with load 20 or 50 kbps that sums up to a certain total background load (total load is given as reference in figures). Background traffic packets arrive with exponentially distributed inter arrival times; rate parameter used is the NS-2 default one [90]. Also for these flows, the destination node is the closest AP. No considerations are taken to the performance of background traffic (Section 6.8.2 shows some results of best effort traffic performance). The $n - n_r$ nodes that are not a source of a real-time flow can switch on or off. Both the on and off periods are modeled by exponentially distributed random variables with a the same mean value 600s. No other value is used as the impact of different on/off times is investigated in Section 7.4. Nodes originating a flow or AP never switch off.

The varying *factors* in this performance evaluation are 1. the number of real-time flows (default: 10), 2. the non-real-time traffic load (default: 0), 4. the packet size of real-time flows (default: 512 bytes), and, obviously, 5. the MAC protocols.

Two simulation studies are made with completely different simulation modes, impact of packet size and impact of number of hops in a chain. The models are described under the corresponding paragraph.

11.3.1 End-to-end delay

As a first performance metric, the end-to-end delay of packets in a real-time flow is investigated. Figure 11.12 shows the CDF of the delay, comparing DCF, EDCA, and DARE using default factor values. DARE manages to deliver *all* packets of reserved flows to their destination within less than 0.05 s. DCF and EDCA, on the other hand, deliver a substantially smaller fraction of packets within this time; DCF needs up to 3 s to deliver a packet in this setup.

Increasing the number of real-time flows from 10 to 20, makes the differences between these protocols become more pronounced (Figure 11.13).

The fact that the CDF of DARE is not a perfect step function (with one delay value for each number of hops between source and destination) is due to slot shifting taking place in the

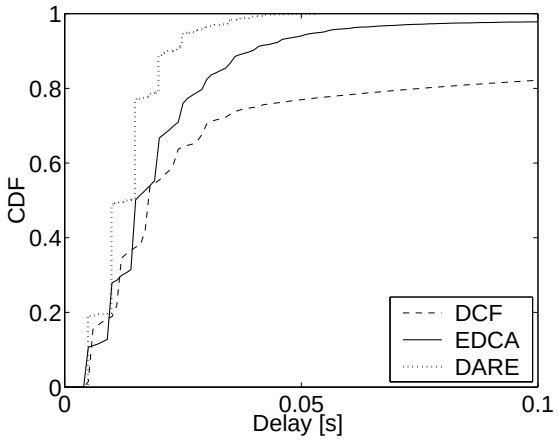


Figure 11.12: CDF of end-to-end delay for 10 real-time flows.

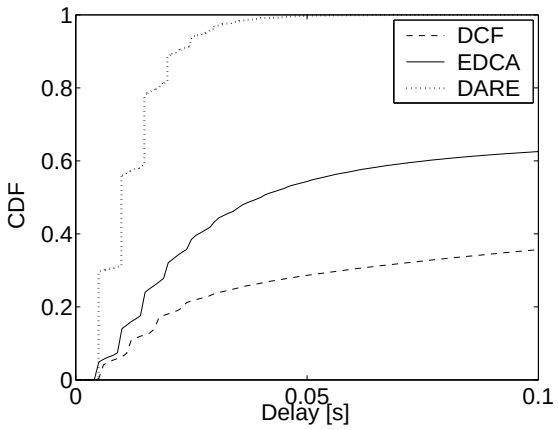


Figure 11.13: CDF of end-to-end delay for 20 real-time flows.

network. Worth repeating is that still, all packets belonging to an established flow arrive with the same delay to the destination; the difference only exists between different flows, not between packets of the same flow. Overall, the delay is *predictable* when knowing the number of hops that a packet has to travel. DCF and EDCA, on the other hand, have a much more spread out CDF, representing their unpredictability of the random access delay.

Not only is DARE's packet delay more predictable, it is also a lot smaller. Figure 11.14 compares average packet delays of the three MAC protocols as a function of the number of real-time flows. Averaged over different flows with different hop count, DARE achieves a practically constant delay; DCF and EDCA increase rapidly at higher offered load.

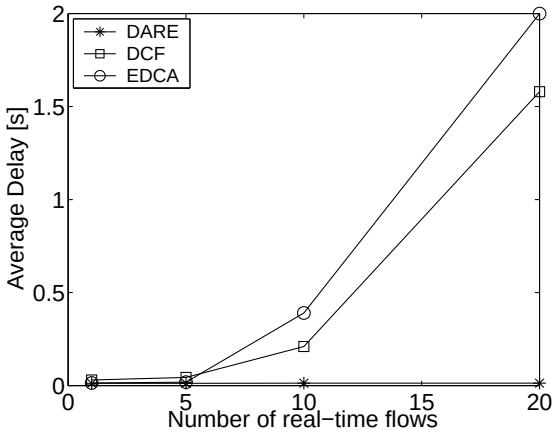


Figure 11.14: Average delay versus the number of real-time flows.

Comparing Figures 11.12 and 11.13 also indicates that with an increased number of offered flows, the step characteristic of DARE's delay CDF is less pronounced. This observation complies with the percentage of flows that experience a slot shift, shown in Figure 11.15. Slot shifting becomes necessary more frequently if the network fills up with reservations, and new flows can only be admitted if they "squeeze in" between existing flows. This explains the somewhat increased variability of DARE's delay at higher offered load. Again, all packets belonging to a shifted flow have the same delay with no variation.

11.3.2 Throughput and blocking

Figure 11.16 shows the average throughput for each protocol as a function of the number of attempted real-time flows. While DCF shows the lowest throughput, EDCA outperforms DARE if the number of attempted flows is low (here: 10). For more real-time flows (here: 20), DARE performs better than EDCA. DARE can actually only support about 7 out of 10 offered flows

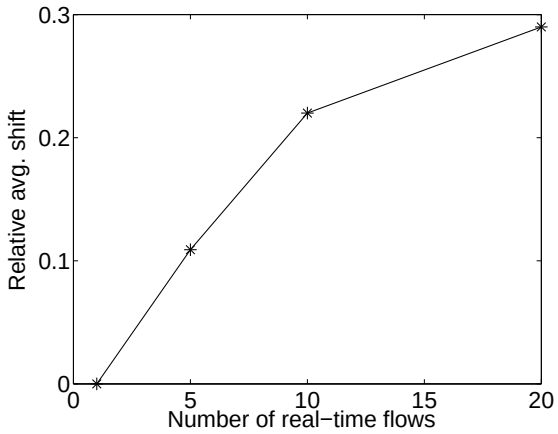


Figure 11.15: Percentage of flows experiencing slot shifts.

(resulting in about 280 kbps offered load). Since a successfully reserved flow should be able to transport all its packets, this lack in throughput could be explained by rejected flow reservations. This result is supported by Figure 11.17, showing the ratio of flows that are requested but not

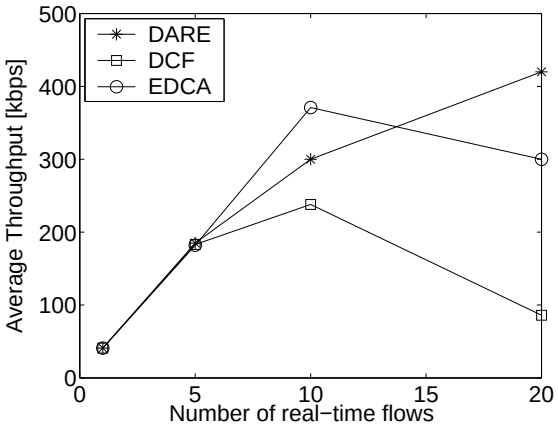


Figure 11.16: Average throughput over the number of real-time flows.

accepted by DARE. These numbers explain the smaller throughput of DARE compared to EDCA and are in accordance with DARE's design philosophy only to admit a flow when it can be supported at high throughput and low delay. Accordingly, the number of blocked flows also increases at higher offered load since the network is less likely to be able to support it.

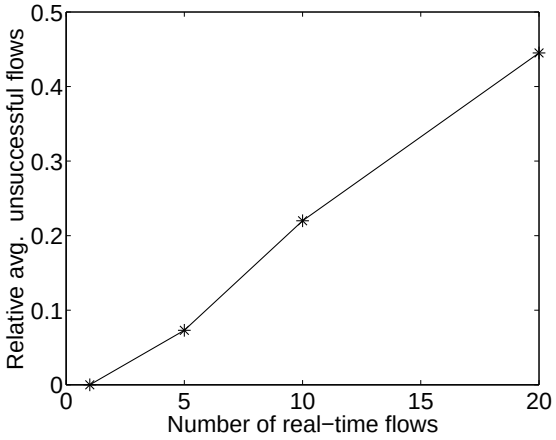


Figure 11.17: Percentage of blocked flows as function of number of offered flows.

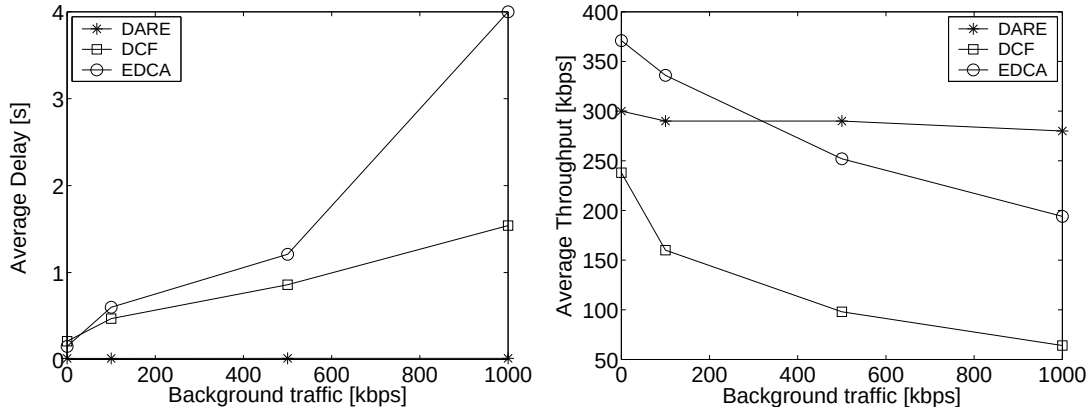
11.3.3 Impact of background traffic

So far, no background traffic is included in the scenarios. This subsection looks at the consequences of such background traffic. Again, all factors are fixed to their default value except background load, which is varied between 0 and 1000 kbps total load generated by all sources. This corresponds to one fourth of the total available network capacity (four access points operating at 1 Mbps each can at maximum drain 4 Mbps from the mesh) and is sufficient to demonstrate crucial differences between different protocols. Larger values of the background traffic are analyzed in [84] for different scenarios.

Figure 11.18 shows the impact of the background traffic on the average delay and throughput of the real-time traffic. DARE's delay and throughput do not significantly vary with increased background traffic, confirming the hypothesis of a reservation-based QoS approach. Even without the optional feature of two-hop protection, real-time traffic is protected by means of reservations from interference. DCF and EDCA, on the other hand, suffer considerably from increased background load. Even at modest background load, the performance of DCF or EDCA is unacceptable, e.g., for real-time applications.

11.3.4 Impact of number of hops

The number of hops of a path has huge impact on the end-to-end delay. Naturally, increasing a path with another hop means that one more node must receive and transmit the packet. For EDCA and DCF, which have contention-based access, an increased hop number has bigger impact on delay and throughput than for DARE. This is illustrated with a simulation of a network

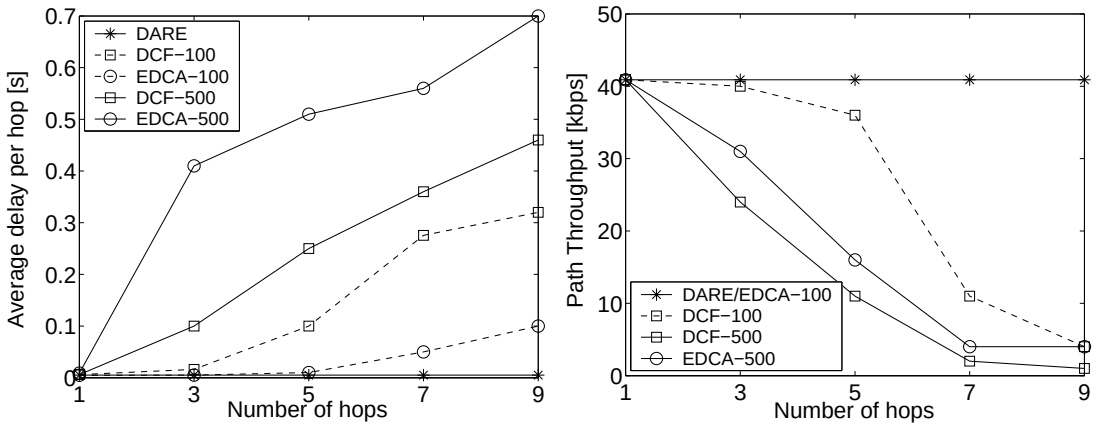


(a) Average real-time delay. (b) Average real-time throughput.

Figure 11.18: Impact of background traffic load on real-time delay and throughput.

with only one real-time flow where the number of hops is varied. Two background traffic types, 100 kbps and 500 kbps, are transmitted from nodes all within the direct neighborhood of the reserved path.

Fig. 11.19 shows the impact of the number of hops on the delay and the throughput. The delay is shown as average delay per hop, where DARE has a constant per hop delay of approximately 0.005 s. EDCA and DCF have a large increase of the per-hop delay. The average path throughput decreases drastically for EDCA and DCF as the number of hops increase, especially when the network load is higher.



(a) Average real-time delay. (b) Average real-time throughput.

Figure 11.19: Impact of path length on real-time delay and throughput.

11.4 Summary

Network parameters affects the performance of real-time traffic transmitted with DARE. When the network area grows, so does the end-to-end delay; more multi-hop paths are required to reach an AP than in smaller areas. Further, when number of APs increase, so does the network throughput. The delay decreases with increasing number of APs; shorter paths are more common to reach the destination. Another interesting aspect is the order of which flows are reserved, the shifting of time slots affect the end-to-end delay. The end-to-end delay for a new flow increases with the number of already active flows. However, when reserved, the end-to-end per packet delay is constant.

DARE offers the same average delay for accepted flows regardless of the number of requested real-time flows. EDCA and DCF has a growing delay with the number of requested flows. CDFs of the delay show that DARE offers all the reserved flows a constant delay for all its packets, whereas EDCA and DCF has a large packet delay variation. With medium load (10 real-time flows) DARE cannot grant them all access at every simulation (randomness of slot start); EDCA has slightly higher throughput. However, as the load grows, the throughput of EDCA decreases whereas DARE increases. The contention-based access of EDCA results in lower throughput with growing number of contending nodes. Still, for all loads DARE outperforms EDCA and DCF in delay performance. For 20 real-time flows, all the packets with DARE arrive at the end destinations within 0.05 seconds whereas EDCA has 50% of the packets and DCF approximately 25%. Background traffic affects the results of DCF and EDCA tremendously, the delay values are unacceptable and especially EDCA – the priority-based QoS approach – is not as good of an option for reliable transmissions as the reservation-based DARE protocol.

Chapter 12

Conclusions

Coverage in IEEE 802.11-based Wireless Local Area Networks (WLAN) can be increased by introducing multi-hopping. Nodes relay transmissions on behalf of others that cannot reach their final destination. Unfortunately, this has an impact on the reliability of the transmissions; contention-based access is performed at each hop, resulting in possibly large and varying packet delays. This lack of Quality-of-Service (QoS) is especially crucial for real-time applications, which require stable transmissions.

The goal of this thesis is to find a QoS enabling mechanism for 802.11-based networks, especially considering multi-hop environments. Many different types of QoS enabling mechanism/extensions to the 802.11 standard exists, thus in the first part of this thesis I analyze which type of QoS mechanism is the best one. Some existing mechanisms use QoS routing or reservations on network layer. However, these do not fulfill any strict guarantees for throughput or delay; the medium access is still contention-based. Therefore, QoS support should be implemented in the MAC protocol, extending the Distributed Coordination Function (DCF) of 802.11. Here, there are two basic options: 1. Assign higher priority or 2. Allocate resources, to the applications with QoS requirements. The major trade off between these two approaches is inter-node signaling required for strict reservation-based QoS and contention-based access that cannot actually guarantee any strict guarantees. If signaling load for a reservation can be kept low, this is by far the best option for strict QoS guarantees.

The major part of this thesis describes a new reservation-based MAC protocol, designed within the borders as they were defined (system under study). The key differentiators to other already existing reservation-based protocols are: End-to-end reservation set up, repair of broken reservations (mobility support), dissemination of reservation information with low signaling load (using piggy-backing), scheduling of multiple reservation with support of different periods and time slots, and dissemination of reservation information two hop around a reserved transmissions

(optional feature). This protocol is called Distributed Allocations of time slots for REal-time traffic (DARE).

The DARE protocol is first of all compared with the DCF using simulations. Then, as my hypothesis is that a reservation-based approach is a better option for both delay and throughput than a priority mechanism, the DARE protocol is also compared with the IEEE 802.11 E standardized priority mechanism Enhanced Distributed Channel Access (EDCA). The results show first of all that the delay of DARE outperforms both EDCA and DCF; it is totally non-varying for a flow. EDCA and DCF have significantly larger variations and average end-to-end packet delay. For throughput, however, EDCA has in networks with low load slightly higher throughput than DARE. DARE does not accept a reservation if the full requirement of it cannot be reached; the flow is blocked. Nevertheless, for each flow the throughput is constant, whereas with EDCA the throughput per flow is lower. When background traffic is increased and also the number of offered real-time flows, the throughput of EDCA decreases and is much smaller than that of DARE. Combining the delay and throughput results gives a clear picture; the DARE protocol is a better method to offer strict QoS in distributed 802.11-based wireless multi-hop networks. Even in dynamic networks where paths break and must be repaired, and where DARE must reschedule reserved transmissions due to reservation conflicts, EDCA and DCF are outperformed.

To conclude, the outcome of this thesis is the foundations for a medium access protocol that especially considers WLAN environments where multi-hopping is used. Its simple and locally bounded functionality for time slot reservations is an effective mean to keep signaling load low and I believe that the DARE protocol has in this thesis proven to be a good means to provide stricter QoS guarantees than a priority based mechanism or the DCF itself.

12.1 Open issues and further studies

One open issue in the DARE protocol is a problematic situation that could occur when nodes are mobile; two already set up chains move in on each other. This is illustrated in Figure 12.1, where one reservation along the path between node A and D, and one reservation along the path between node E and H exist. Assume node B and node F have overlapping time slots, which were no conflict at the set up. If they move closer to each other, the overlapping time slots result in two unsuccessful reserved paths where all packets collide. The reservation protocol must release one of the paths and set up a new one along this path. This is not included in the DARE design today as DARE anyhow has some failure handling that forces a reservation to be released after x successive periodic transmissions have been unsuccessful. However, it is an interesting suggestion for further studies.

Another issue that has not been approached is networks that suffer from fading. However,

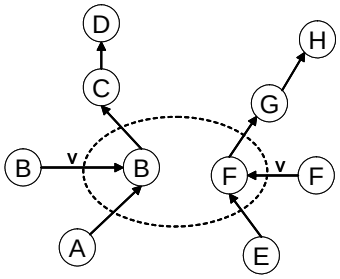


Figure 12.1: Two chains moving in on each other

due to the failure handling above, if a wireless channel suffers from fading during a long instance, the reservation is simply released. Also, no new reservation set up will be successful until fading has dropped.

Publication list

1. E. Carlson, H. Karl and A. Wolisz, *Medium Access Issues for QoS Support, Power Control and Multiple Frequency Channels in Distributed Multi-Hopping Wireless Networks*,Technical Report TKN-04-010, Telecommunication Networks Group, Technische Universität Berln, sep 2004.

2. E. Carlson, H. Karl, A. Wolisz and C. Prehofer, *Distributed Allocation of Time Slots for Real-time Traffic in a Wireless Multi-hop Network*, In Proc. European Wireless, Barcelona, Spain, Feb 2004.

3. E. Carlson, C. Bettstetter, H. Karl, C. Prehofer and A. Wolisz, *Distributed Maintenance of Resource Reservation Paths in Multihop 802.11 Networks*, In Proc. IEEE Vehicular Technology Conference (VTC), Los Angeles, USA, Sep. 2004.

4. E. Carlson, C. Bettstetter, H. Karl, C. Prehofer and A. Wolisz, *POSTER: Distributed MAC for Real-Time Traffic in Multi-Hop Wireless Networks*, in Proc. IEEE Conf. on Sensor and Ad Hoc Communications and Networks, Santa Clara, CA, USA, Oct, 2004.

5. E. Carlson, C. Bettstetter, C. Prehofer and A. Wolisz, *A Performance Comparison of QoS Approaches for Ad Hoc Networks: 802.11e versus Distributed Resource Allocation*, in Proc. European Wireless, Nicosia, Cyprus, Apr. 2005.

6. E. Carlson, M. Kubisch and D. Hollos, *A Receiver Based Protecting Protocol for Wireless Multi-hop Networks*, in Proc. ACM Intl. Workshop on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks (PE-WASUN 2005), Montreal, Canada, Oct. 2005.

7. F. Rindler, M. Kubisch, E. Carlson and D. Hollos, *On the roper Interference Protection in Wireless Multi-hop Networks*, Wireless Communications and Networking Conf. (WCNC), Hong Kong, Hong Kong, Mar. 2007.

8. E. Carlson, C. Bettstetter, C. Prehofer, H. Karl and A. Wolisz, *A Distributed End-to-end Reservation Protocol for Wireless IEEE 802.11 based Mesh Networks*,Journal of Selected Areas of Communication (JSAC): Special Edition on MESH networks, 2006.

Bibliography

[1] M. Frodigh, S. Parkwall, C. Roobol, P. Johansson, and P. Larsson. Future-generation wireless networks. *IEEE Personal Communications Magazine*, pages 10–17, October 2001.

[2] M. S. Gast. *802.11 Wireless Networks: The Definite Guide*, chapter A peek ahead at 802.11n: MIMO-OFDM. O'Reilly, 2 edition, 2005.

[3] IEEE Std 802.11g: Specific Requirements Part II: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 2003.

[4] IEEE Std 802.11i: Specific Requirements Part 11: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications amendement 6: MAC Security Enhancements, 2004.

[5] J.-C. Chen, M.-C. Jiang, and Y. Liu. Wireless LAN security and IEEE 802.11i. *IEEE Wireless Communications*, February 2005.

[6] M. Kubisch, S. Mengesha, D. Hollos, H. Karl, and A. Wolisz. Applying ad-hoc relaying to improve capacity, energy efficiency, and immission in infrastructure-based WLANs. In *Proc. of Kommunikation in Verteilten Systemen (KiVS)*. Leipzig, Germany, February 2003.

[7] IEEE Std 802.11: Wireless LAN Medium Access Control (MAC) and Physical layer (PHY) specifications, 1997.

[8] M. Gerharz, C. de Waal, M. Frank, and P. James. A Practical View on Quality-of-Service Support in Wireless Ad Hoc Networks. In *Proc. IEEE Workshop on Applications and Services in Wireless Networks (ASWN)*. Bern, Switzerland, July 2003.

[9] S. Xu and T. Saadawi. Does the IEEE 802.11 MAC protocol work well in Multihop Ad Hoc Networks? *IEEE Commun. Mag.*, June 2001.

[10] K. Sundaresan, H.-Y. Hsieh, and R. Sivakumar. IEEE 802.11 over Multi-hop Wireless Networks: Problems and new Perspectives. *Elsevier Ad Hoc Networks*, April 2004.

[11] A. Lindgren, A. Almquist, and O. Schelén. Quality of Service Schemes for IEEE 802.11 - A Simulation Study. In *Proc. Intl. Workshop on Quality of Service (IWQoS)*. Karlsruhe, Germany, June 2001.

[12] IEEE Std 802.11e: Specific Requirements Part 11: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications amendment 8: MAC Quality of Service (QoS), 2005.

[13] S. Mangold, S. Choi, P. May, O. Klein, G. Hiertz, and L. Stibor. IEEE802.11e Wireless LAN for Quality of Service. In *Proc. European Wireless*. Florence, Italy, February 2002.

[14] C. T. Calafate, P. Manzoni, and M. P. Malumbres. Assessing the effectiveness of IEEE 802.11e in multi-hop mobile network environments. In *Proc. IEEE Intl. Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS)*. Volendam, The Netherlands, October 2004.

[15] R. Ramanathan. Changelles: A Radically New Architecture for Next Generation Mobile Ad Hoc Networks. In *Proc. ACM Intl. Conf. on Mobile computing and newtorking (MOBICOM)*. Cologne, Germany, September 2005.

[16] IEEE Std 802.3: Standards for Local Area Networks: Carrier Sense Multiplexing with Collision Detection (CSMA/CD) Access method and Physical layer Specifications, 1985.

[17] IEEE Std 802.11a: Supplement to Specific Requirements Part 11: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications: High Speed Physical Layer in the 5 GHz band, 1999.

[18] IEEE Std 802.11b: Supplement to Specific Requirements Part 11: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Higher-speed Physical Layer Extension in the 2.4 GHz band, 1999.

[19] IEEE Std 802.11h: Specific Requirements Part II: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications – Amendement for Dynamic Frequency Selection (DFS) and Transmit Power Control (TPC) in the 5 GHz band, 2004.

[20] S. Khurana, A. Kahol, and A. Jayasumana. Effect of Hidden Terminals on the Performance of the IEEE 802.11 MAC Protocol. In *Proc. of Local Computer Networks Conf. (LCN)*. Boston, Massachusetts, USA, October 1998.

[21] W. M. Moh, D. Yao, and K. Makki. Wireless LAN: Study of Hidden-terminal Effect and Multimedia Support. In *Proc. Intl. Conf. on Computer Communications and Networks (ICCCN)*. Lafayette, LA, USA, October 1998.

[22] P. Karn. MACA – A New Channel Access Method for Packet Radio. In *ARRL/CRRL Amateur Radio 9th Computer Networking Conf.* Ontario, Canada, September 1990.

[23] E. Carlson, H. Karl, and A. Wolisz. Medium Access Issues for QoS Support, Power Control and Multiple Frequency Channels in Distributed Multi-Hopping Wireless Networks. Technical report, Technical Report TKN-04-010, Telecommunication Networks Group, Technische Universität Berlin, September 2004.

[24] L.-L. Xie and P. R. Kumar. A Network Information Theory for Wireless Communication: Scaling Laws and Optimal Operation. In *Submitted to IEEE Transactions on Information Theory*, Apr 12, 2002. Revised November 9, 2003.

[25] Media stations project, University of Aachen, http://www.comnets.rwth-aachen.de/ ftp-mul/.

[26] K. Sundaresan, H.-Y. Hsieh, and R. Sivakumar. IEEE 802.11 over Multi-hop Wireless Networks: Problems and New Perspectives. *Elsevier Ad Hoc Networks*, April 2004.

[27] R. Braden, D. Clark, and S. Shenker. Integrated Services in the Internet Architecture: An Overview. IETF, RFC 1633, June 1994.

[28] S. Black, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for Differentiated Services. IETF, RFC 2475, December 1998.

[29] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. Resource ReSerVation Protocol (RSVP) – Functional Specification. IETF, RFC 2205, September 1997.

[30] A. Talukdar, B. Badrinath, and A. Acharya. MRSVP: A Resource Reservation Protocol for an Integrated Services Network with Mobile Hosts. *The Journal of Wireless Networks*, 7(1), 1997.

[31] T. S. Rappaport. *Wireless Communications, principles and practise*. Prentice Hall, 1996.

[32] J. Zander and L. Ahlin. *Principles of Wireless Communications*. Studentlitteratur, Sweden, 1998.

[33] H Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: ATransport Protocol for Real-Time Applications. IETF, RFC 1889, January 1996.

[34] R. Sivakumar, P. Sihna, and V. Bharghavan. CEDAR: a Core-Extraction Distributed Ad hoc Routing algorithm. *IEEE Journal on Selected Areas in Communications*, pages 1454–1465, 1999.

[35] H.-K. Wu and P.-H. Chuang. Dynamic QoS Allocation for Multimedia Ad Hoch Wireless Networks. *Mobile Networks and Applications*, 6:377 – 384, August 2001.

[36] M. Kazantzidis, M. Gerla, and S.-J. Lee. Permissible Throughput Network Feedback for Adaptive Multimedia in AODV MANETs. In *Proc. Intl. Conf. on Communications (ICC)*. Amsterdam, Netherlands, June 2001.

[37] P. J. M. Havinga and G. Wu. Wireless Internet on Heterogenous Networks. In *Proc. Workshop Mobile Commun. in Perspective*. Enschede, The Netherlands, February 2001.

[38] S.-Y. Park, K. Kim, D.C Kim, S. Choi, and S. Hong. Collaborative QoS Architecture between DiffServ and 802.11e Wireless LAN. In *Proc. Vehicular Technology Conference (VTC)*. Jeju, Korea, April 2003.

[39] M. Li, H. Zhu, S. Sathyamurthy, I. Chlamtac, and B. Prabhakaran. End-to-End Framework for QoS Guarantee in Heterogeneous Wired-cum-Wireless Networks. In *Proc. Conf. on Quality of Service in Heterogeneous Wired/Wireless Networks (QSHINE)*. Dallas, Texas, USA, October 2004.

[40] S. Lee and A. Campell. INSIGNIA: In-band Signalling Support for QoS in Mobile Ad Hoc Networks. In *Proc. Intl. Workshop on Mobile Mulimedia Communication (MoMuC)*. Berlin, Germany, October 1998.

[41] J. Xue, P. Stuedi, and G. Alonso. ASAP: An Adaptive QoS Protocol for Mobile Ad Hoc Networks. In *IEEE Intl. Symposium on Personal Indoor and Mobile Radio Communications (PIMRC)*. Beijing, China, September 2003.

[42] M. Lampe, H. Rohling, and J. Eichinger. PER-Prediction for Link Adaptation in OFDM Systems. In *Proc. OFDM Workshop*. Hamburg, Germany, 2002.

[43] J. D. P. Pavon and S. Choi. Link Adaptation Strategy for IEEE 802.11 WLAN via Received Signal Strength Measurement. In *Proc. Intl. Conf. on Communications (ICC)*. Anchorage, Alaska, USA, May 2003.

[44] B. E. Mullins, N. J. Davis IV, and S. F. Midkiff. An Adaptive Wireless Local Area Network Protocol that Improves Throughput via Adaptive Control of Direct Sequence Spectrum Parameters.

ACM SIGMOBILE Mobile Comp. and Communications, 1997.

[45] P. Chevillat, J Jelitto, A. N. Barreto, and H. L. Truong. A Dynamic Link Adaptation Algorithm for IEEE 802.11s Wireless LANs. In *Proc. Intl. Conf. on Communications (ICC)*. Anchorage, Alaska, USA, May 2003.

[46] N. H. Vaidya, P. Bahl, and S. Gupta. Distributed Fair Scheduling in Wireless LAN. In *Proc. ACM Intl. Conf. on Mobile Computing and Networking (MOBICOM)*. Boston, USA, August 2000.

[47] A. Banchs and X. Perez. Distributed Weighted Fair Queuing in 802.11 Wireless LAN. In *Proc. Intl. Conf. on Communications (ICC)*. Copenhagen, Denmark, May 2002.

[48] Y. Yang and R. Kravets. Distributed QoS Guarantees for Realtime Traffic in Ad Hoc Networks. In *IEEE Conf. on Sensor and Ad Hoc Communications and Networks (SECON)*. Santa Clara, CA, USA, October 2004.

[49] Q. Qiang, L. Jacob, R. Radhakrishna Pillai, and B. Prabhakaran. MAC Protocol Enhancements for QoS Guarantee and Fairness over the IEEE 802.11 Wireless LAN. In *Proc. Intl. Conf. on Computer Communications and Networks (ICCNC)*. Miami, USA, October 2002.

[50] M. A. Visser and M. El Zarki. Voice and Data Transmission over an 802.11 Wireless Network. In *Proc. IEEE Intl. Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. Toronto, Canada, September 1995.

[51] L. Jacob, R. Radhakrishna Pillai, and B. Prabhakaran. MAC Protocol Enhancements and a Distributed Scheduler for QoS Guarantees over the IEEE 802.11 Wireless LAN. In *Proc. IEEE Vehicular Technology Conference (VTC)*. Boston, USA, September 2002.

[52] D. Qiao and K.G. Shin. Achieving Efficient Channel Utilization and Weighted fairness for Data Communications in IEEE 802.11 WLAN under the DCF. In *Proc. IEEE Intl. Workshop on Quality of Service*. Miami Beach, USA, May 2002.

[53] V. Kanodia, C. Li, A. Sabharwal, B. Sadeghi, and E. Knightly. Distributed Multi-Hop Scheduling and Medium Access with Delay and Throughput Constraints. In *Proc. ACM Intl. Conf. on Mobile Computing and Networking (MOBICOM)*. Rome, Italy, July 2001.

[54] H. Zhu, M. Li, I. Chlamtac, and B. Prabhakaran. A Survey of Quality of Service in IEEE 802.11 Networks. *IEEE WIreless Communications*, August 2004.

[55] A. Lindgren, A. Almquist, and O. Schelén. Evaluation of Quality of Service Schemes for IEEE 802.11 wireless LANs. In *Proc. IEEE Conf. on Local Computer Networks (LCN)*. Tampa, Florida, USA, November 2001.

[56] N. Jain, S.R. Das, and A. Nasipuri. A Multichannel CSMA MAC Protocol with Receiver-based Channel Selection for Multihop Wireless Networks. In *Proc. Intl. Conf. on Computer Communications and Networks (ICCCN)*. Scottsdale, Arizona, USA, October 2001.

[57] J. So and N. H. Vaidya. A Multichannel MAC Protocol for Ad Hoc Wireless Networks. In *Technical Report, Dept. of Electrical and Computer Engineering, University of Illinois, jso1,nhv@uiuc.edu*, January 2003.

[58] A. Tzamaloukas and J.J Garcia-Luna-Aceves. A Channel-Hopping Protocol for Ad-Hoc Networks.

In *Proc. Intl. Conf. on Computer Communications and Networks (ICCNC)*. Las Vegas, USA, April 2000.

[59] S.-L. Wu, C.-Y. Lin, Y.-C. Tseng, and J.P.-Sheu. A New Multi-Channel MAC Protocol with On-demand Channel Assignement for Multi-hop Mobile Ad Hoc Networks. In *Proc. IEEE Wireless Communications and Networking Conference (WCNC)*. Chicago, USA, September 2000.

[60] J. Deng and Z. J. Haas. Dual Busy Tone Multiple Access (DBTMA) : A New Medium Access Control for Packet Radio Networks. In *Proc. IEEE Intl. Conf. on Universal Personal Communications (ICUCP)*. Italy, October 1998.

[61] Z. J. Haas and J. Deng. Dual Busy Tone Multiple Access (DBTMA) Performance Evaluation. In *Proc. IEEE Vehicular Technology Conference (VTC)*. Houston, Texas, USA, May 1999.

[62] J. C. Fang and G. D. Kondylis. A Synchronous, Reservation Based Medium Access Control Protocol for Multihop Wireless Networks. In *Proc. IEEE Wireless Communications and Networking Conf. (WCNC)*. New Orleans, Lousiana, USA, March 2003.

[63] E. Carlson, M. Kubisch, and D. Hollos. A Receiver Based Protecting Protocol for Wireless Multihop Networks. In *Proc. of ACM Intl. Workshop on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks (PE-WASUN 2005)*. Montreal, Canada, October 2005.

[64] Venkatesh Rajendran, Katia Obraczka, and J. J. Garcia-Luna-Aceves. Energy-efficient Collision-free Medium Access Control for Wireless Sensor Networks. In *Proc. Intl. Conf. on Embedded networked sensor systems (SenSys)*. Los Angeles, California, USA, November 2003.

[65] J. Grönkvist. Assignment Methods for Spatial reuse TDMA. In *Proc. ACM Intl. symposium on Mobile ad hoc networking & computing (MOBIHOC)*. Boston, Massachusetts, August 2000.

[66] S. Jiang, J. Rao, D. He, X. Ling, and C. C. Ko. A Simple Distributed PRMA for MANETs. *IEEE Transactions on Vehicular Technology*, March 2002.

[67] C. Bettstetter and J. Eberspächer. Hop Distances in Homogeneous Ad Hoc Networks. In *Proc. IEEE Vehicular Techn. Conf. (VTC)*. Jeju, Korea, April 2003.

[68] *ITU-T Recommendation G.729, Coding of Speech at 8 kbps Using Conjugate-Strucutre Algobraic-Code-Excited Linear-Prediction (CS-ACELP)*, March 1996.

[69] *ITU-T Recommendation G.114, One-way Transmission Time*, March 1993.

[70] C. H. R. Lin and M. Gerla. Asynchronous Multimedia Multihop Wireless Networks. In *Proc. Conf. of the IEEE Computer and Communications Societies (INFOCOM)*. Kobe, Japan, April 1997.

[71] J. L. Sobrinho and A. S. Krishnakumar. Real-time Traffic over the IEEE 802.11 Medium Access Control Layer. *Bell Labs Technical Journal*, pages 172–187, 1996.

[72] J. L. Sobrinho and A. S. Krishnakumar. Quality of Service in Ad hoc Carrier Sense Multiple Access Wireless Networs. *IEEE Journal on Selected Areas in Communications*, pages 17(8):1353–1368, August 1999.

[73] S.-T.Sheu, T.-F. Sheu, C.-C. Wu, and J.-Y. Luo. Desing and Implementation of a Reservation-based MAC Protocol for Voice/Data over IEEE802.11 Ad Hoc Wireless Networks. In *Proc. Intl. Conf. on Communications (ICC)*. Amsterdam, Netherlands, June 2001.

[74] S.-T. Sheu and T.-F. Sheu. DBASE: A Distributed Bandwidth Allocation/Sharing/Extension Protocol for Multimedia over IEEE 802.11 Ad Hoc Wireless LANs. In *Proc. Joint Conf. of the IEEE Computer and Communications Societies (INFOCOM)*. Anchorage, Alaska, USA, April 2001.

[75] G. R. Hiertz, J. Habetha, P. May, E. Wei, R. Bagul, and S. Mangold. A Decentralized Reservation Scheme for IEEE 802.11 Ad Hoc Networks. In *Proc. IEEE Intl. Symposium on Personal Indoor and Mobile Radio Communications (PIMRC)*. Bejing, P.R. China, September 2003.

[76] Z. Ying, A.L. Ananda, and L. Jacob. A QoS Enabled MAC Protocol for Multi-Hop Ad Hoc Wireless Networks. In *Proc. IEEE Intl. Conf. on Performance, Computing, and Communications (IPCCC)*. Phoenix, USA, April 2003.

[77] M. K. Marina, G. D. Kondylis, and U. C. Kozat. RBRP: a Robust Broadcast Reservation Protocol for Mobile Ad Hoc Networks. In *Proc. Intl. Conf. on Communications (ICC)*. Amsterdam, Netherlands, June 2001.

[78] S. Lal and E. Sousa. Distributed resource allocation for DS-CDMA based multi-media wireless LANs. In *Proc. IEEE Military Communications Conference (MILCOM)*. Boston, USA, October 1998.

[79] S. Koutroubinas, T. Antonakopoulos, and V. Makios. A new efficient access protocol for integrating multimedia services in the home environment. *IEEE Transactions on Consumer Electronics*, pages 481–487, August 1999.

[80] A. Muir and J.J. Garcia-Luna-Aceves. Group Allocation Multiple Access with Collision Detection. In *Proc. Joint Conf. of the IEEE Computer and Communications Societies (INFOCOM)*. Kobe, Japan, April 1997.

[81] J. L. Sobrihno and J. M. Brazio. A Multiple Access Protocol for Intergrated Voice and High-Speed Data in Wireless Networks. In *Proc. IEEE Intl. Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. The Hague, Holand, September 1994.

[82] E. Carlson, C. Bettstetter, H. Karl, C. Prehofer, and A. Wolisz. Distributed Maintenance of Resource Reservation Paths in Multihop 802.11 Networks. In *Proc. IEEE Vehicular Technology Conference (VTC)*. Los Angeles, USA, September 2004.

[83] E. Carlson, C. Bettstetter, C. Prehofer, and A. Wolisz. A Performance Comparison of QoS Approaches for Ad Hoc networks: 802.11e versus Distributed Resource Allocation. In *Proc. European Wireless*. Nicosia, Cyprus, April 2005.

[84] E. Carlson, H. Karl, A. Wolisz, and C. Prehofer. Distributed Allocation of Time Slots for Real-time Traffic in a Wireless Multi-hop Network. In *Proc. European Wireless*. Barcelona, Spain, February 2004.

[85] E. Carlson, C. Bettstetter, H. Karl, C. Prehofer, and A. Wolisz. POSTER: Distributed MAC for Real-Time Traffic in Multi-Hop Wireless Networks. In *IEEE Conf. on Sensor and Ad Hoc Communications and Networks*. Santa Clara, CA, USA, October 2004.

[86] E. Carlson, C. Bettstetter, C. Prehofer, H. Karl, and A. Wolisz. A Distributed End-to-end Reservation Protocol for Wireless IEEE 802.11 based Mesh Networks. *Journal of Selected Areas of Communication (JSAC): Special Edition on MESH networks*, 2006.

[87] S. Desilva and R. V. Boppana. On the Impact of Noise Sensitivity on Transport Layer Performance in 802.11 based Ad Hoc Networks. In *Proc. Intl. Conf. on Communication (ICC)*. Paris, France, June 2004.

[88] V. Anantharaman, S.-J. Park, K. Sundaresan, and R. Sivakumar. TCP performance over mobile ad-hoc networks: A quantative study. *Wireless Communications and Mobile Computing*, March 2004.

[89] F. Rindler, M. Kubisch, E. Carlson, and D. Hollos. on the Proper Interference Protection in Wireless Multi-hop Networks. In *Proc. of Wireless Communications and Networking Conf. (WCNC)*. Hong Kong, Hong Kong, March 2007.

[90] The Network Simulator 2. http://www.isi.edu/nsnam/ns/.

[91] G.-S. Poo. A Cumulative Negative Acknowledgement (CNAK) approach for Scalable Reliable Multicast. In *Proc. IEEE Conf on Computer, Communications and Networks (ICCCN)*. Scottsdale, AZ, USA, October 2001.

[92] C. R. Lin and M. Gerla. Adaptive Clustering for Mobile Wireless Networks. *IEEE Journal on Selected Areas in Communications*, 15(7), September 1997.

[93] IETF MANET Working Group. RFC 3561 – Ad hoc On-Demand Distance Vector Routing (AODV). http://www.ietf.org/rfc/rfc3561.txt.

[94] C. Bettstetter. On the Connectivity of Wireless Multihop Networks with Homogeneous and Inhomogeneous Range Assignment. In *Proc. IEEE Vehicular Techn. Conf. (VTC)*, September 2002.

[95] Nordic VLSI ASA. *nRF2401 Single Chip 2.4GHz Radio Transceiver*. Tiller, Norway, March 2003.

[96] I. Marsh, F. Li, and G. Karlsson. Wide Area Measurements of VoIP Quality. In *Cost Intl. Workshop on Quality of Future Internet Services (QoFIS)*. Stockholm, Sweden, October 2003.

[97] S. Wiethoelter and C. Hoene. Design and Verification of an IEEE 802.11e EDCA Simulation Model in ns-2.26. Technical Report TKN-03-019, Telecommunication Networks Group, Technische Universität Berlin, November 2003.

[98] *ITU-T Recommendation G.711, Pulse Code Modulation (PCM) of Voice Frequencies*, November 1998.

[99] G. Fischer and R. Sacher. *Einfürung in die Algebra*, chapter Anhang 1: Elementare Zahlentheorie, pages 212–214. Teubner, 1983.

[100] IETF MANET Working Group, March 2003. http://www.ietf.org/html.charters/manet-charters.html.

[101] IETF MANET Working Group. RFC 2501 – Routing Protocol Performance Issues and Evaluation Considerations. http://www.ietf.org/rfc/rfc2501.txt.

[102] IETF MANET Working Group. Internet Draft – Dynamic MANET On-Demand Routing (DYMO). http://www.ietf.org/internet-drafts/draft-ietf-manet-dymo-04.txt.

[103] IETF MANET Working Group. RFC 3626 – Optimized Link State Routing Protocol (OLSR). http://www.ietf.org/rfc/rfc3626.txt.

[104] IETF MANET Working Group. Internet Draft – Dynamic Source Routing Protocol (DSR).

http://www.ietf.org/internet-draft/draft-ietf-manet-dsr-10.txt.

Appendix A

NS-2 simulator overview

The simulations are performed using the network simulator NS-2 (exact version: ns-2.26). It is a discrete event simulator which supports wired as well as wireless networking protocols. Several communication protocols are included in NS-2, e.g. IEEE 802.11 and AODV routing and it also includes different traffic generation models. The simulator is an open source project, i.e. the whole code is available in the internet [90].

The basic structure of NS-2 and the networking protocols are implemented in the programming language C++. For easy control and assembly of simulations, the script language Tcl is used.

The idea of discrete event simulations is that events may only be initiated as a result of other events or inputs. Therefore NS-2 consists of a scheduler and a scheduling list. Each event has to be inserted into the scheduling list together with its expiration date. The scheduler goes through the scheduling list at runtime and starts the actions which are associated with the expired date.

The physical parameters used for the 802.11 module in NS-2 are taken from the Lucent WaveLan card and can also be found on the web-site. The resulting communication range is about 230 meters and the interference range is about 500 meters.

Appendix B

Ad hoc routing

The IEEE 802.11 standard covers the LLC, MAC and physical layers. Routing is not covered and theoretically, it could be possible to use a known routing protocol from wired network technology in 802.11 based networks. But, routing in wireless multi-hop networks is difficult since there are problems that do not exist in a wired network; fading, shadowing and mobility can cause transmissions over a set up path to be unsuccessful. Further nodes can be mobile and enter, exit and move around in an unpredictable manner, which can cause paths to break. This requires a routing protocol designed for these issues that can effectively repair, or find new paths. Further, when the communication is over multiple hops, an intermediate node can leave and it is up to the routing protocol to find a new path for communication. The routing protocol has to be dynamic and follow the fast changes of the network. Routing protocols from wired networks technology which do not have to deal with such problems are not suitable.

A lot of research for routing protocols in wireless (multi-hop) ad hoc networks is done by IETF Mobile Ad-hoc NETwork (MANET) working group [100]. They basically look at two different sorts of protocols: Reactive and proactive. The main difference is that the proactive protocols are updated at a node whenever a change occurs in the network; no matter whether the change actually affects the node itself or not. Reactive protocol only update route information when the change in topology actually affects the node itself or when the node takes an action that requires a new route, e.g. initiates a transmission. The routes are changed only when a transmission is not successful or after a time-out. This is different from proactive routing protocols where the routes are determined before transmission and constantly updated.

Another classification of routing protocols is centralized and distributed routing [101]. With centralized routing, a central node, i.e. AP, handles all routing decisions. This implies that an AP needs to have information about the radio conditions in the whole network in order to take the correct routing decisions. If this information can be supplied, the approach of centralized

routing is simple. But, the AP needs to inform the users about the routing, demanding additional signaling. The total signaling load from both AP to user (route information) and user to AP (link information) can be high. Thus, this is not very attractive in a network where load is heavy. There is also a question of how often information should be requested by an AP in order to keep routes up to date.

With a distributed routing approach, all nodes take their own routing decision. One big advantage with this solution is that nodes can use information from other layers on the spot and utilize such information to make better routing decisions. Further, there is no need to distribute routing information from/to a centrally controlling node (AP).

How to determine which type of routing to use depends on the network, e.g. where it is put up and what kind of traffic is to be supported. Thorough analysis before such a decision can be made is required. However, for a distributed ad hoc network, a reactive distributed routing protocol is suitable since no central control is required and information is exchanged only when needed. IETF has specified many reactive routing protocols [93, 102, 103, 104]. The optimized state routing protocol (OLSR) [103] uses multipoint relays, which are selected nodes that forward broadcast messages during the flooding state of the routing mechanism. All routing protocols flood the network with routing messages to find and update link- and route states. The approach in OLSR minimizes the signaling overhead towards classical flooding techniques where every node forwards the broadcasting messages. Link state information is generated by the relay nodes only. It is also possible to forward some state information only; such partial link state flooding also minimizes signaling. The link state is used to calculate optimum routes, based on number of hops.

The Dynamic Source Routing (DSR) protocol [104] is simple and specially designed for multi-hop wireless networks. It allows the network to be completely self organized and self configuring, with no infrastructure (as with OLSR) needed. The protocol uses two main mechanisms: 1. Route discovery and 2. Route Maintenance that reacts and cooperate completely on demand to retrieve the best routes between a source and destination node. The protocol allows multiple routes to any destination and allows the sender to select and control the routes. This allows for a node to have routes for different purposes, e.g. at congestion it might use one specific route. The recovery when a route is broken is very rapid. However, the protocol is designed for up to around 200 nodes only. It is designed to work well with very high rates of mobility.

Another protocol is the Ad hoc On-Demand Distance Vector (AODV) [93] algorithm that also is completely dynamic. Routes are established quickly for new destinations and nodes are not required to maintain routes to destinations that are not in active communication. Route breakage is recovered quickly and all broken paths are invalidated by all affected nodes instantly. AODV uses three messages, Route Requests (RREQ), Route Replies (RREP) and Route Error

(RERR), which are received at any node via UDP. When a route to a destination is needed, AODV broadcasts RREQ messages. A route can be determined when the destination itself receives the RREQ or when an intermediate node has a valid route to the destination. The route is available when the destination or the intermediate node unicasts a RREP back to the source node. When a link breaks a RERR message is used. The RERR indicates which nodes are no longer reachable. The AODV algorithm is simple, well working and totally distributed and reactive. It is suitable for many network types and is not limited to any fixed infrastructure, low load or number of nodes. For simulations in this thesis, the AODV protocol is used. However, the functionality designed in this thesis is not limited to any particular routing protocol.

Appendix C

Additional analyses and algorithms

C.1 Proof of Theorem 1 from Section 10.4

The following theorem and its proof are taken from [99], but can be found in any book about elementary number theory.

Theorem 1 For every two integers a and b with $a \neq 0$ there are unique integers q and r with

$$b = qa + r \text{ and } 0 \leq r < a.$$

Proof One only has to consider the case $a > 0$ because from $a < 0$ it follows that $|a| = -a > 0$. So, let a be $a > 0$ and $M := \{b - na : n \in \mathbb{Z}\}$. M contains natural numbers, because in case $b \geq 0$, b is in M and in case $b < 0$ it is $b - ba = b(1 - a) \in \mathbb{N}$. Let $r = b - qa$ be the smallest natural number contained in M. Then it is $b - (q + 1)a < b - qa = r$, i.e. $b - (q + 1)a \notin \mathbb{N}$ and therefore $r < a$, so that altogether it is

$$b = qa + r \text{ and } 0 \leq r < a.$$

If it is

$$b = q'a + r' \text{ and } 0 < r' < a,$$

it follows $q + 1 \leq q'$ from $q < q'$, i.e. $r' = b - q'a \leq b - (q + 1)a = b - qa - a = r - a < 0$. This is a contradiction. The case $q' < q$ can be led to a contradiction in an analog way, so one obtains $q = q'$ and $r = r'$.

C.2 Additional analysis to Section 10.4

In this section it is analyzed if the difference between the beginnings of two slots can be smaller than the gcd of the periodicities. The measures used in this section are the same as in Section 10.4.

What if $\frac{d}{2}$ is also a natural number? Conclusively it is also a divider of both z_1 and z_2.

$$z_1 = m_1 \frac{d}{2}$$
$$z_2 = m_2 \frac{d}{2}$$

If we put this definition into Equation 10.13 we obtain the following.

$$|z_1 - z_2| = |m_1 \frac{d}{2} - m_2 \frac{d}{2}| = |m_1 - m_2| \cdot \frac{d}{2} \tag{C.1}$$

But this does *not* mean that the smallest possible distance is $\frac{d}{2}$! The reason for this is that m_1 and m_2 *cannot* be arbitrary natural numbers. Not every number that can be divided by $\frac{d}{2}$ can also be divided by d. As we assumed that d *is* a divider of both z_1 and z_2, m_1 and m_2 *must* be multiples of 2, which leads to the same result as before.

$$z_1 = m_1 \frac{d}{2} = 2n_1 \frac{d}{2}$$
$$z_2 = m_2 \frac{d}{2} = 2n_2 \frac{d}{2}$$

$$\Rightarrow |z_1 - z_2| = |2n_1 \frac{d}{2} - 2n_2 \frac{d}{2}| = |2n_1 - 2n_2| \cdot \frac{d}{2} = |n_1 - n_2| \cdot d \tag{C.2}$$

In other words the smallest possible difference between z_1 and z_2 larger than 0 must be d because otherwise d would not be a divider of one of the numbers. The next number larger than z_1 that can be divided by d is $z_1 + d$. However, this only shows us that the smallest *possible* difference of z_1 and z_2 is the gcd d.

C.3 The shifting algorithm

Algorithm 2 presents a shifting algorithm that a node should use whenever it receives a request for a new reservation path. If the slots of the newly requested reservation path fit together with other potentially already existing reservations this algorithm returns 0. In case of overlapping time slots this shifting algorithm returns either a suitable time shift so that all slots fit together

without overlapping with one another or it returns -1 if such a shift is not possible. The required information to use this algorithm are the following.

$p_1, \ldots, p_n$**:** The periodicities of the reservation paths this node knows about

p_r**:** The periodicity of the requested reservation path

$s_1, \ldots, s_n$**:** The slot lengths of the reservation paths this node knows about

s_r**:** The slot length of the requested reservation path

$\mathrm{gcd}_{\mathrm{old}}$**:** The *greatest common divider* of the periodicities $p_1, \ldots, p_n$

$\mathrm{startTime}$**:** The point of time when the requested time slots starts

n**:** The number of reservation paths this node knows about

Algorithm: Shifting algorithm

Input: $p_1 \dots p_n, p_r, s_1 \dots s_n, s_r, \mathrm{GCD_{old}}$,startTime, n
Result: Suitable time shift or -1

```
1  gcd_new = gcd(gcd_old, p_r)
2  if s_r + Σ_{i=1}^{n} s_i ≥ gcd_new then
3  |  return -1
   end
4  for i ∈ {1 ... n} do
5  |  P_i = p_i·p_r / gcd(p_i, p_r)
   end
6  checkTime=max(P_i ... P_n)
7  works=false
8  t=startTime
9  while !works && t-startTime¡gcd_new do
10 |  works=true
11 |  for t̃ = t; t̃ <startTime+checkTime;t̃+=p_r do
12 |  |  if slotStart_last + s_last ≥ t̃ then
13 |  |  |  t+=slotStart_last + s_last − t̃
14 |  |  |  works=false
15 |  |  |  break
   |  |  end
16 |  |  if slotStart_next ≤ t + s_r then
17 |  |  |  t+=slotStart_next + s_next − t̃
18 |  |  |  works=false
19 |  |  |  break
   |  |  end
   |  end
   end
20 if works then
21 |  return t−startTime
   else
22 |  return -1
   end
```

Algorithm 2: The shifting algorithm.

First, the gcd of all periodicities including the requested one is computed using Algorithm 1 from Section 10.4. The sum of all slot lengths including the requested one is then compared to the newly computed gcd. The algorithm returns -1 if the sum is greater than the gcd. If the sum is smaller than the gcd the algorithm checks if the requested slot fits together with the others at the time it is requested (startTime). If it does not fit, the algorithm looks for a suitable place for the requested slot during the period of time defined in 10.5.3. This time is called checkTime.

Further the algorithm uses the boolean variable *works* and the time variable t. The boolean variable *works* is used to signal whether a suggested slot fits with the others or not. For entering the *while* loop it has to be set to *false*. The other condition is that the shifted time is smaller than the gcd, because after gcd the slot schemes repeat themselves (see Section 10.4). The expression $t -$ startTime represents the shifted time. In the beginning t is set to *startTime* which results in a time shift of 0. At first it has to be checked whether the suggested slot fits together with the others without any shift at all. After entering the *while* loop *works* is set to true.

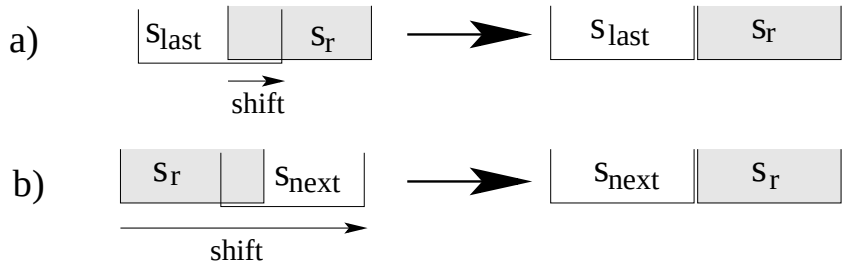


Figure C.1: How the shifting algorithm shifts.

In the *for* loop it is tested whether the requested slot and all future slots of the requested path that occur during *checkTime* fit together. For this another time variable $\tilde{t}$ is set to t and after every execution of the loop body it is increased by the requested periodicity p_r. The condition for the execution is that $\tilde{t}$ is less than *startTime+checkTime*. The first *if* loop tests whether the last slot that occurred before the slot that starts at $\tilde{t}$ overlaps with it, i.e. if the end of that last slot is greater than $\tilde{t}$ (the starting point of the requested slots or one of its future followers during *checkTime*, as shown in Case a in Figure C.1). If these slots do overlap, a new point of time is suggested where the requested slot could start. This point of time is right at the end of the slot it overlaps with. In this case the variable *works* is set to *false* and the *for* loop is exited. The second *if* loop tests whether the suggested slot (or one of its future followers) overlaps with the next slot that occurs after $\tilde{t}$, i.e. the end of the requested slot is bigger than the beginning of the next slot, as shown in Case b in Figure C.1. If this is the case a new starting time for the requested slot is suggested. Here it is set right at the end of the next slot. Again, *works* is set to *false* and the *for* loop is exited.

If the *for* loop is exited without any overlapping slots, *works* is still *true*, so the while loop is exited, too. If the suggested shifted time is greater than $\gcd - s_r$ the *while* loop is exited with *works* being *false*. If after exiting the *while* loop the variable *works* is true, a suitable time shift could be found so that no time slots do overlap. This time shift is returned as result. But if *works* is false, this means than no suitable shift could be found, so -1 is returned as result.

Appendix D

Additional figures to Section 10.3.2

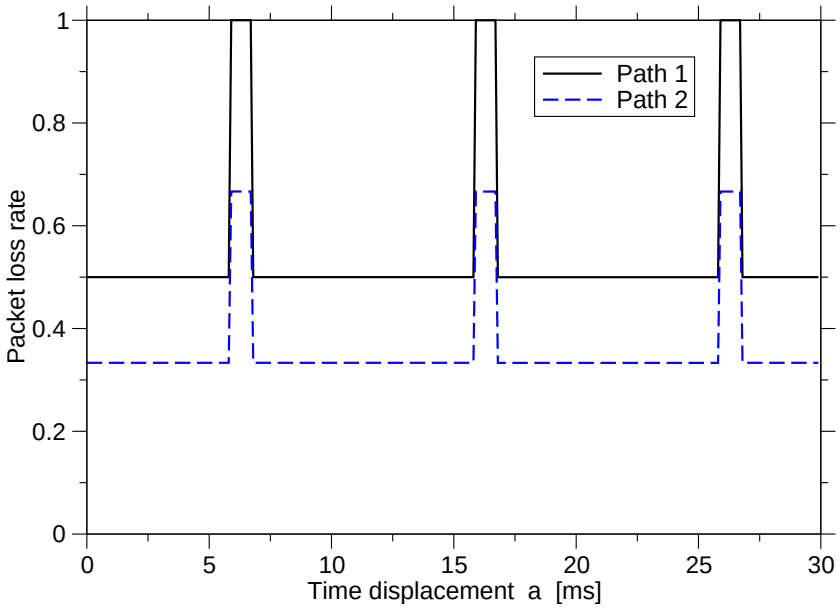


Figure D.1: Path 1: $p_1 = 30\text{ms}, \text{s}_1 = 6.752\text{ms}$, Path 2: $p_2 = 20\text{ms}, \text{s}_2 = 4.192\text{ms}$.

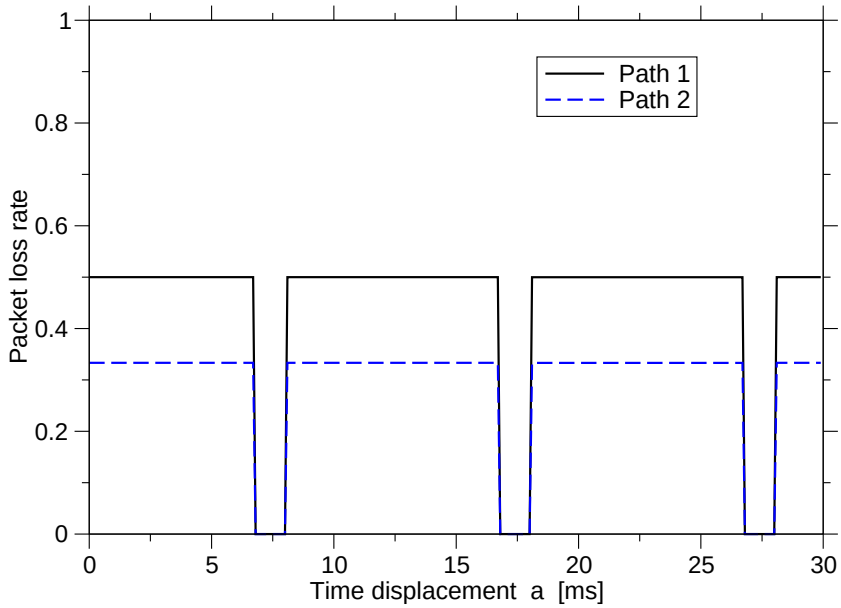


Figure D.2: Path 1: $p_1 = 30\text{ms}, \text{s}_1 = 6.752\text{ms}$, Path 2: $p_2 = 20\text{ms}, \text{s}_2 = 1.952\text{ms}$.

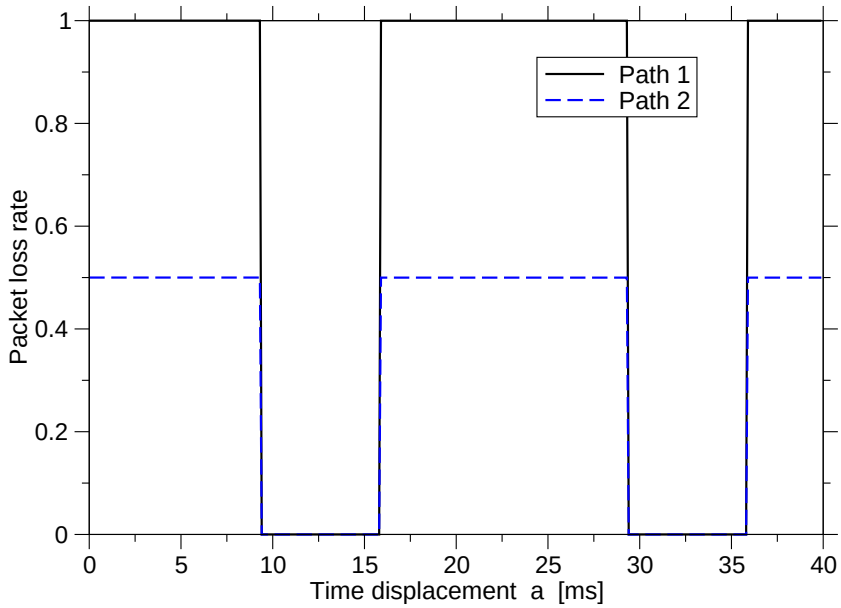


Figure D.3: Path 1: $p_1 = 40\text{ms}, \text{s}_1 = 9\text{ms}$, Path 2: $p_2 = 20\text{ms}, \text{s}_2 = 4\text{ ms}$.

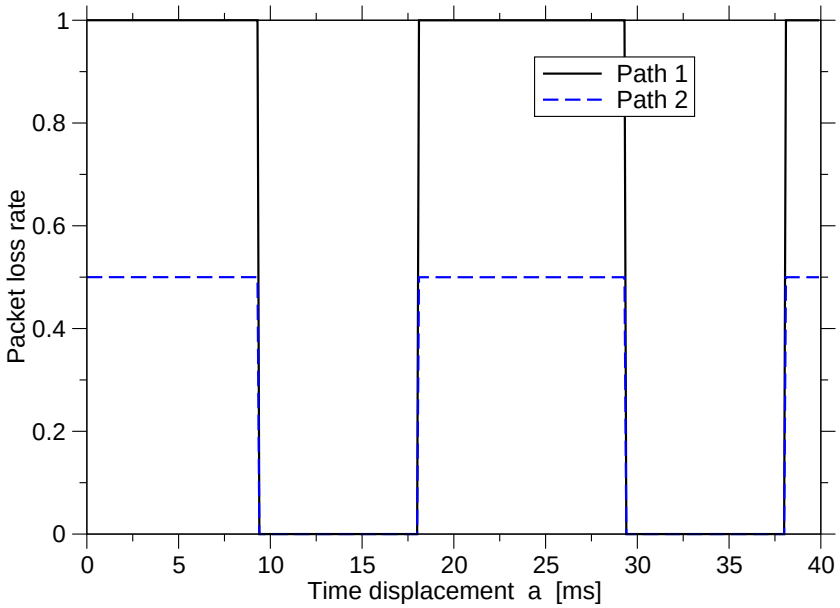


Figure D.4: Path 1: $p_1 = 40\text{ms}, \text{s}_1 = 9.321\text{ms}$, Path 2: $p_2 = 20\text{ms}, \text{s}_2 = 1.952\text{ms}$.

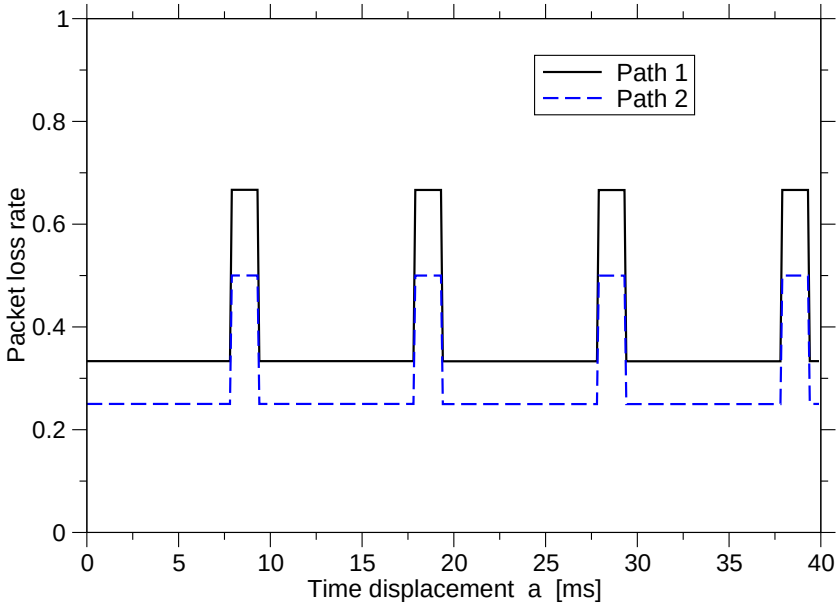


Figure D.5: Path 1: $p_1 = 40\text{ms}, \text{s}_1 = 9.321\text{ms}$, Path 2: $p_2 = 30\text{ms}, \text{s}_2 = 2.112\text{ms}$.

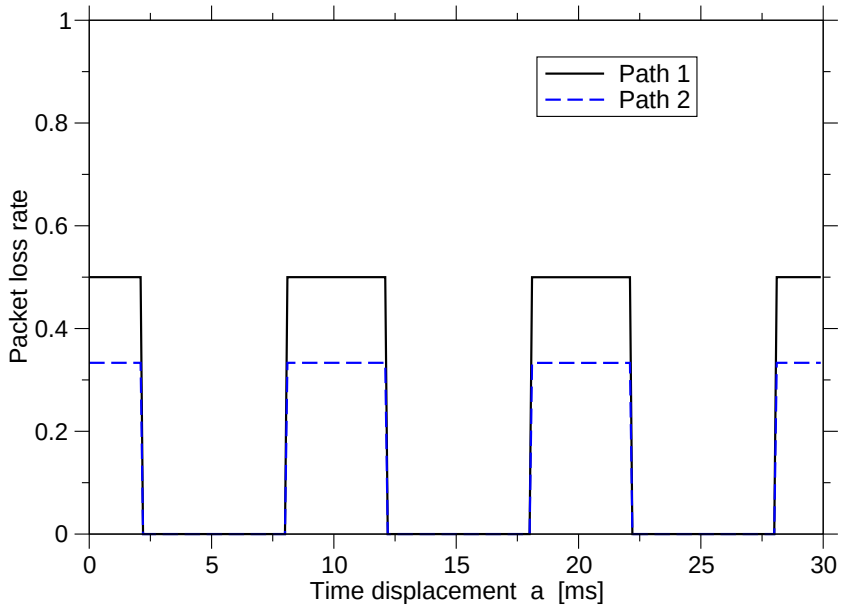


Figure D.6: Path 1: $p_1 = 30\text{ms}, \text{s}_1 = 2.112\text{ms}$, Path 2: $p_2 = 20\text{ms}, \text{s}_2 = 1.952\text{ms}$.

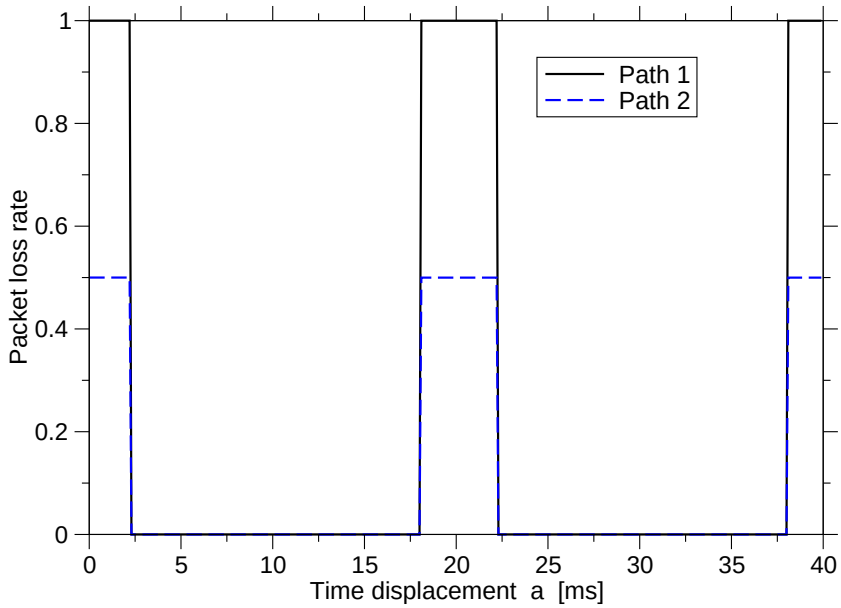


Figure D.7: Path 1: $p_1 = 40\text{ms}, \text{s}_1 = 2\text{ms}$, Path 2: $p_2 = 20\text{ms}, \text{s}_2 = 2\text{ ms}$.

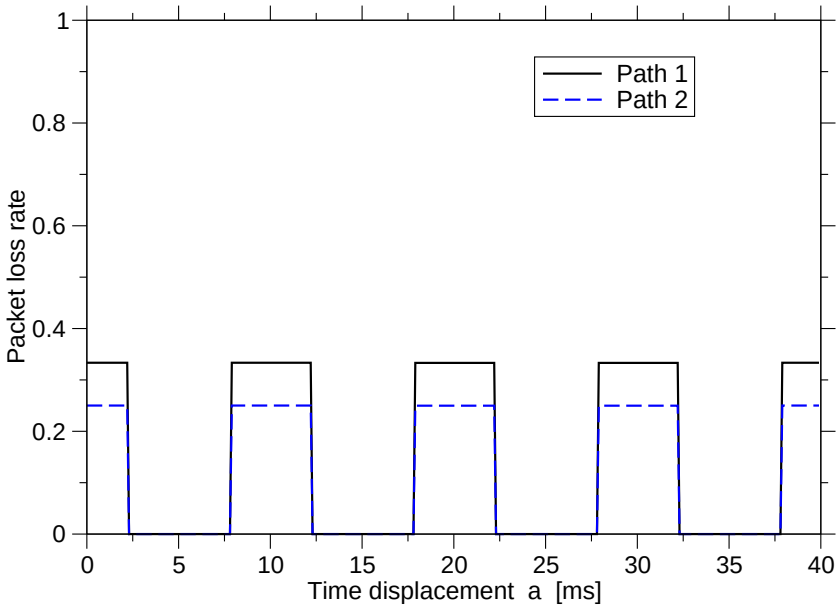


Figure D.8: Path 1: $p_1 = 40\text{ms}, \text{s}_1 = 2.272\text{ms}$, Path 2: $p_2 = 30\text{ms}, \text{s}_2 = 2.112\text{ms}$.

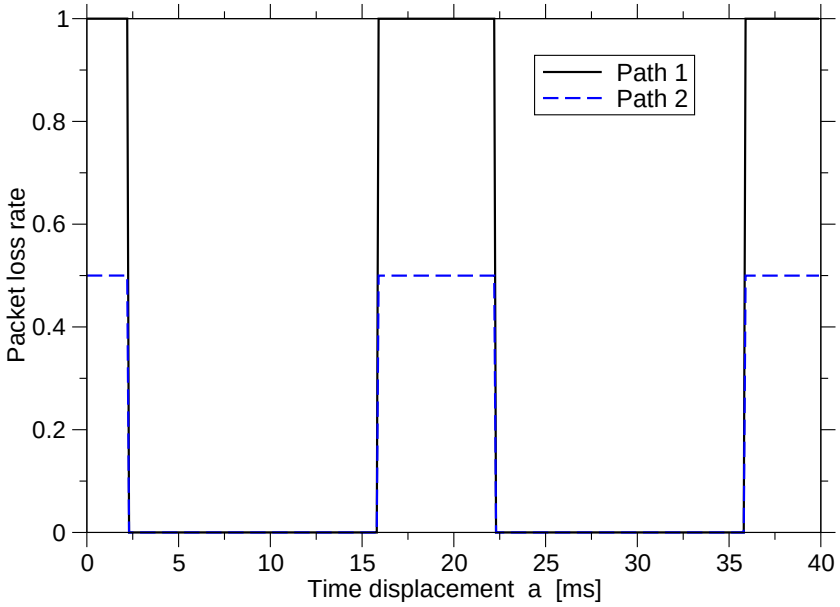


Figure D.9: Path 1: $p_1 = 40\text{ms}, \text{s}_1 = 2.272\text{ms}$, Path 2: $p_2 = 20\text{ms}, \text{s}_2 = 4.192\text{ms}$.

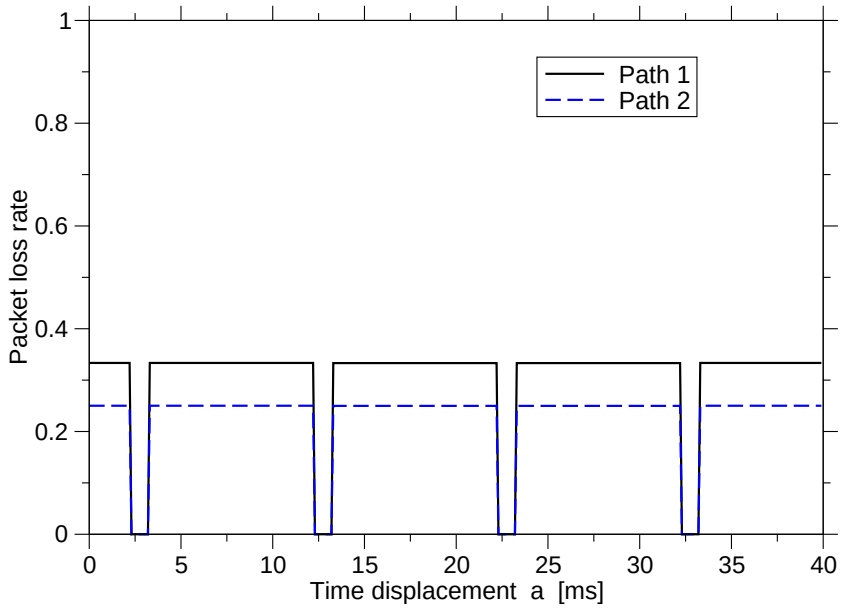


Figure D.10: Path 1: $p_1 = 40\text{ms}, \text{s}_1 = 2.272\text{ms}$, Path 2: $p_2 = 30\text{ms}, \text{s}_2 = 6.752\text{ms}$.

Appendix E

Additional protection features

This chapter presents some additional analysis and ideas for the protection of a reservation.

E.1 Jamming of conflicting transmission requests

DARE offers an additional feature where the RTS/CTS exchange is used to inform nodes non-aware of a reservation that initiates a transmission to a node aware of a reservation. This has proven to function well, see Section 8.3. More problematic is the scenario when a node starts a transmission to a node which has no knowledge about the reservation; there are no means of explicitly inform this node. Here an idea for how to handle other cases is presented.

When a node with information about a reservation overhears a conflicting reservation request or request for a non-real-time transmission, RTS or RTR, it can jam the reception of the CTR/CTS. Figure E.1 shows a jamming scenario. A reserved transmission from node A to node C exist. The grey area indicates where the send packet along this path can be read. Node D can read information and has knowledge about the reservation. Node E has no reservation information and starts a transmission to node F. Node D can jam the CTS that node E should receive.

However, for every started conflicting transmission, a jamming is needed, which leads to overhead; more nodes in the area that normally could transmit during a reserved time slot are stopped as the jamming burst is widely spread. Also, it might not be possible for the nodes aware of the reservation to jam – The jamming burst can interfere with the reservation. An implementation of jamming functionality is presented in [63]

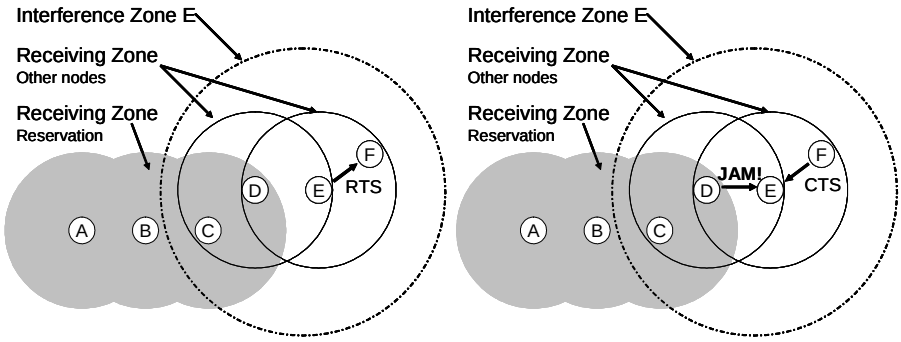


Figure E.1: Jamming scenario.

E.2 Spatial re-usage

There are cases when a full two-hop protection is not needed and some simultaneous transmissions within the two-hop neighborhood can be allowed. The motivation is to increase the a) spatial reuse, hence the overall bandwidth utilization and b) fairness for the non-real-time transmissions that do occur in the network. Consider the simple node chain configuration in Figure E.2 where node A, B and C are part of a reservation, node D has knowledge about the reservation and node E and F have not. The circular lines marks the receive zone of every node. First, let node A be the source and node C be the destination. If node E would be a

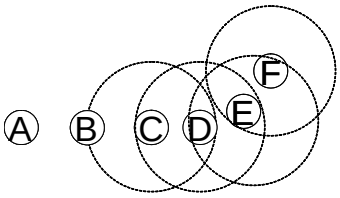


Figure E.2: Node A, B and C are part of a reservation. Node D has knowledge about it, node E and F have not.

receiver, that is node D overhears a CTS only, this transmission is allowed. It will not interfere with the reception at node C. Respectively, If node E is a transmitter, node D must stop this transmission.

Assume the direction of the reservation is switched to the opposite and node C is the source and node A is the destination, node D would know about the reservation from hearing an RTR or one DATA frame only. Here, if node E starts a transmission (node D overhears an RTS or RTR), node D can allow this transmission. Node B which is a receiver is on three hops from the

transmitting node E. It all depends on which message node D has received reservation information from and which new message node D overhears. All cases are shown in Table E.2. When a

Newly received/overheard	Knowledge from existing reservations		
	1 RTR/DATA	2 RTR/DATA	CTR/ACK
RTR	allow	not allow	not allow
2 RTR	not allow	not allow	not allow
CTR	2nd res. suffers	2nd res. suffers	allow
RTS	allow	not allow	not allow
CTS	NRT suffers	NRT suffers	allow

node knows about a reservation from one RTR only, that is it is on the source side, several simultaneous transmissions can be allowed. On the end destination side, all new transmissions where the node overhears a CTR (only) or a CTS can be allowed. All others must be stopped. Using these rules in a jamming mechanism can increase the network performance and also fairness for other non-reserved transmissions.