Peter Dely

Cross-Layer Optimization of Voice over IP in Wireless Mesh Networks

Karlstad University Studies
2010:25
Wireless Mesh Networks (WMNs) have emerged as a promising network technology, which combines the benefits of cellular networks and Wireless Local Area Networks (WLANs). In a WMN mesh routers wirelessly relay traffic on behalf of other mesh routers or clients and thereby provide coverage areas comparable to cellular networks, while having the low complexity and low costs of WLANs.

As Voice over IP (VoIP) is a very important Internet service, it is critical for the success of WMNs to support high quality VoIP. However, current WMNs are not adapted well for VoIP. The capacity and scalability of single-radio WMNs is low, especially for small packet transmissions of VoIP calls, because the MAC and PHY layer overhead for small packets is high. The scalability of multi-radio/multi-channel WMNs is usually higher, since fewer nodes contend for a channel. However channel scheduling might be required, which can lead to excessive delay and jitter and result in poor VoIP quality. In this thesis we investigate how to deliver high quality VoIP in single radio and multi-radio networks by using cross-layer optimization.

For single radio WMNs, we consider the use of IP packet aggregation and IEEE 802.11e transmission opportunities. We conclude that IP packet aggregation greatly improves the capacity and thereby the scalability of WMNs. We show that the key for providing good quality is to artificially delay packets prior to aggregation. We propose a distributed cross-layer optimization system, which, based on Fuzzy Logic Inference, derives an aggregation delay that enhances the capacity and quality.

For multi-radio/multi-channel WMNs, we demonstrate the importance of quality-of-service-aware channel scheduling. We develop a quality-of-service-aware channel scheduler that compared to a basic round-robin scheme significantly reduces jitter and in that way increases VoIP quality. Our analysis shows that there is a trade-off between the jitter of high priority VoIP traffic and the throughput of background TCP traffic.

The proposed optimizations significantly increase the capacity of single-radio and multi-radio WMNs. This allows network operators to serve more users with an existing mesh infrastructure or provide better service delivery to existing users.
Acknowledgments

First of all, I would like to take this opportunity thank my advisor Andreas Kassler who, like a good sports coach, helped to improve my work through his advice, challenging questions and inspirational ideas. Without his continuous support all this would have not been possible. Thank you.

Furthermore, I would like to express my gratitude to my colleagues and friends from the Computer Science department at Karlstad University, in particular the Distributed Systems and Communications Research (DISCO) group, for their support and the enlightening (and sometimes funny or even absurd) discussions at the coffee table. Also, I am grateful to Dirk Staehle (University of Wuerzburg, Germany), for reviewing my Licentiate proposal and accepting the role of opponent in my Licentiate thesis defense.

I would like to thank the European Commission (well, the European tax payers) for their financial support through the Interreg IVB North See Region project E-CLIC, the FP7 project NEWCOM++ and COST Action for Traffic Monitoring and Analysis. Furthermore, I am grateful for the financial and technical support from Deutsche Telekom Research Labs, in particular Hans Einsiedler and his group.

Huge thanks go to my parents and my family for their support and encouragement throughout all the long years of study. Last but not least, I am indebted to my sweet girlfriend Yao Qin, whose love and understanding keeps me motivated from morning to evening.
List of Appended Papers

This thesis is comprised of the following four peer-reviewed papers. References to the papers will be made using the Roman numbers associated with the papers such as Paper I. The paper reprints are subject to small editorial changes.


Comments on my Participation

For Papers I-III, I am responsible for carrying out the experimental evaluation, and for most of the written material and ideas. For Paper IV, I am responsible for implementing the scheduler and parts of the experiments and the written material.

Other Papers

Apart from the papers included in the thesis, I have co-authored the following papers:


## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acknowledgements</td>
<td>iii</td>
</tr>
<tr>
<td>List of Appended Papers</td>
<td>iv</td>
</tr>
<tr>
<td>Introductory Summary</td>
<td>1</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>3</td>
</tr>
<tr>
<td>2 Background</td>
<td>4</td>
</tr>
<tr>
<td>2.1 Wireless Mesh Networks</td>
<td>4</td>
</tr>
<tr>
<td>2.2 Voice over IP</td>
<td>10</td>
</tr>
<tr>
<td>2.3 Cross-Layer Design and Optimization</td>
<td>13</td>
</tr>
<tr>
<td>3 Challenges, Solutions and Research Questions</td>
<td>16</td>
</tr>
<tr>
<td>3.1 Challenges</td>
<td>16</td>
</tr>
<tr>
<td>3.2 Solutions and Research Questions</td>
<td>17</td>
</tr>
<tr>
<td>4 Research Method</td>
<td>19</td>
</tr>
<tr>
<td>5 Summary of Papers and Contributions</td>
<td>21</td>
</tr>
<tr>
<td>5.1 Paper I - Theoretical and Experimental Analysis of the Channel Busy Fraction in IEEE 802.11</td>
<td>21</td>
</tr>
<tr>
<td>5.2 Paper II - An Experimental Comparison of Burst Packet Transmission Schemes in IEEE 802.11-based Wireless Mesh Networks</td>
<td>23</td>
</tr>
<tr>
<td>5.3 Paper III - FUZPAG: A Fuzzy-Controlled Packet Aggregation Scheme for Wireless Mesh Networks</td>
<td>24</td>
</tr>
<tr>
<td>5.4 Paper IV - QoS-Aware Channel Scheduling for Multi-Radio/Multi-Channel Wireless Mesh Networks</td>
<td>25</td>
</tr>
<tr>
<td>6 Conclusions and Outlook</td>
<td>27</td>
</tr>
<tr>
<td>Paper I: Theoretical and Experimental Analysis of the Channel Busy Fraction in IEEE 802.11</td>
<td>35</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>37</td>
</tr>
<tr>
<td>2 Analytical Model</td>
<td>39</td>
</tr>
<tr>
<td>2.1 IEEE 802.11 DCF under Saturation Conditions</td>
<td>39</td>
</tr>
<tr>
<td>2.2 IEEE 802.11 DCF under Non-Saturation Conditions</td>
<td>40</td>
</tr>
<tr>
<td>2.3 Modeling the Channel Busy Fraction</td>
<td>41</td>
</tr>
<tr>
<td>2.4 Discussion</td>
<td>42</td>
</tr>
<tr>
<td>2.5 Limitations of the Model</td>
<td>42</td>
</tr>
<tr>
<td>CONTENTS</td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td><strong>3 Validation of the Model</strong></td>
<td>44</td>
</tr>
<tr>
<td>3.1 Experimental Setup</td>
<td>44</td>
</tr>
<tr>
<td>3.2 Channel Busy Fraction and Traffic Injection Rate</td>
<td>45</td>
</tr>
<tr>
<td><strong>4 Application: Available Bandwidth Estimation</strong></td>
<td>46</td>
</tr>
<tr>
<td><strong>5 Summary and Conclusion</strong></td>
<td>48</td>
</tr>
<tr>
<td><strong>Paper II: An Experimental Comparison of Burst Packet Transmission Schemes in IEEE 802.11-based Wireless Mesh Networks</strong></td>
<td>51</td>
</tr>
<tr>
<td><strong>1 Introduction</strong></td>
<td>54</td>
</tr>
<tr>
<td><strong>2 Background</strong></td>
<td>55</td>
</tr>
<tr>
<td>2.1 IEEE 802.11, RTS/CTS and Transmission Opportunities</td>
<td>55</td>
</tr>
<tr>
<td>2.2 IEEE 802.11 A-MSDU/A-MPDU</td>
<td>57</td>
</tr>
<tr>
<td>2.3 IP Packet Aggregation</td>
<td>57</td>
</tr>
<tr>
<td><strong>3 Performance Evaluation</strong></td>
<td>58</td>
</tr>
<tr>
<td>3.1 Experimental Setup</td>
<td>58</td>
</tr>
<tr>
<td>3.2 Single-Hop Performance</td>
<td>59</td>
</tr>
<tr>
<td>3.3 Multi-Hop Performance</td>
<td>62</td>
</tr>
<tr>
<td><strong>4 Conclusions</strong></td>
<td>66</td>
</tr>
<tr>
<td><strong>Paper III: FUZPAG: A Fuzzy-Controlled Packet Aggregation Scheme for Wireless Mesh Networks</strong></td>
<td>69</td>
</tr>
<tr>
<td><strong>1 Introduction</strong></td>
<td>71</td>
</tr>
<tr>
<td><strong>2 System Description</strong></td>
<td>73</td>
</tr>
<tr>
<td>2.1 Performance Analysis</td>
<td>74</td>
</tr>
<tr>
<td>2.2 Impact of Packet Aggregation</td>
<td>74</td>
</tr>
<tr>
<td><strong>3 Fuzzy Controlled Packet Aggregation</strong></td>
<td>76</td>
</tr>
<tr>
<td>3.1 Input Variables</td>
<td>76</td>
</tr>
<tr>
<td>3.2 Output</td>
<td>76</td>
</tr>
<tr>
<td>3.3 Fuzzy Rules</td>
<td>77</td>
</tr>
<tr>
<td><strong>4 Implementation and Evaluation</strong></td>
<td>79</td>
</tr>
<tr>
<td>4.1 Implementation</td>
<td>79</td>
</tr>
<tr>
<td>4.2 Evaluation Environment</td>
<td>80</td>
</tr>
<tr>
<td>4.3 Controller Stability and Settling Time</td>
<td>80</td>
</tr>
<tr>
<td>4.4 Single-Hop Scenario</td>
<td>82</td>
</tr>
</tbody>
</table>
4.5 Multi-Hop Scenario ............................................. 83

5 Conclusions ..................................................... 84

Paper IV: QoS-Aware Channel Scheduling for Multi-Radio/Multi-
Channel Wireless Mesh Networks ............................ 87

1 Introduction .................................................. 89

2 Background and Related Work .............................. 91
   2.1 Multi-Channel Mesh Networks ............................ 91
   2.2 IEEE 802.11e EDCA ....................................... 92

3 QoS-Aware Channel Scheduler .............................. 92
   3.1 Design Goals and Motivation ............................ 92
   3.2 Scheduling Algorithm ..................................... 92
   3.3 Analysis .................................................. 93
   3.4 Implementation .......................................... 97

4 Performance Evaluation ..................................... 98
   4.1 Evaluation Environment .................................. 98
   4.2 Results ................................................ 100

5 Conclusion .................................................. 105
Introductory Summary
1 Introduction

Wireless Mesh Networks (WMNs) have gained notable attention by the research community and industry in recent years. WMNs are wireless multi-hop networks, which are typically based on cheap Wireless Local Area Network (WLAN) technology, but exceed the coverage area of WLANs by using multi-hop transmissions. Thereby WMNs have emerged as an alternative to other types of wireless networks, as they share the low costs, good performance and ease of deployment known from classical WLANs, while potentially providing large network access coverage for whole cities comparable to cellular networks.

The combination of low cost, high speeds and large coverage made WMNs popular for application scenarios where WLANs or cellular networks could not be deployed for economic or technical reasons, for example in rural areas of developing countries. Despite first commercial successes, WMNs remain a very active research area. As user numbers grow and new services are introduced, it gets more and more clear that current WMNs cannot fulfill the requirements of future networks in terms of scalability and performance [1]. In particular multi-media services such as Voice over IP (VoIP), i.e. telephony over IP-based networks, or video conferencing put a high burden on networks, since they demand low packet loss rates and delay.

VoIP is an integral service of today's Internet and is likely to be the basis for many future Internet services such as E-learning or E-health. Thus it is crucial for the further success of WMNs to efficiently support high quality VoIP. However, today's WMNs lack the scalability required to provide high quality VoIP to large user groups. Increasing the scalability would be advantageous both for network operators and end-users. Network operators can achieve higher revenues if their networks support more users. End-users benefit, since lower operational expenditures might lead to lower prices and more ubiquitous availability.

The main theme of this thesis is the question of how to advance current WMNs to support high quality VoIP. The thesis contains an introductory summary, followed by re-prints of four peer-reviewed papers on this topic (subject to small editorial changes), which were co-authored by the author of this thesis. To facilitate a better understanding of the questions related to the paper re-prints, Section 2 briefly introduces important background material. In Section 3 specific research questions are posed and their relevance to the research community and industry is elaborated. Section 4 discusses the used research method. In Section 5 we summarize the included papers and comment on their main contributions. Section 6 concludes the introductory summary and provides an outlook to future work.
2 Background

In this section we first explain in detail what WMNs are and how they are used. Then, we proceed with describing the operation and requirements of a typical VoIP system. Finally, we present different approaches for cross-layer performance optimization and relate them to the ideas used in this thesis.

2.1 Wireless Mesh Networks

Following, we introduce some basic terminology related to wireless mesh networks, present typical usage scenarios, discuss different types of WMNs and list characteristics of WMNs and challenges arising from them.

2.1.1 Terminology

According to the IEEE 802.11s draft standard [2], a WMN can comprise four types of nodes: Mesh Stations, Mesh Access Points, Mesh Portals and Stations. A Mesh Station (MSTA) is a node, which supports mesh services i.e. it implements the protocols for the management and operation of a WMN. In particular, MSTAs can wirelessly forward traffic. If a node in addition provides access services to legacy client stations (STA), it is called Mesh Access Point (MAP). Since the association procedure is identical to the association with a normal access point, accessing the mesh via a MAP is transparent for STAs. A Mesh Portal (MPP) is a mesh node, which is connected to the mesh and a second network, for example the Internet. It serves as an entry point for MAC Service Data Units (MSDUs).

As not all WMNs are based on IEEE 802.11s, other terms can be found in literature (an in this thesis) as well. For example, mesh stations or mesh access points are sometimes called mesh routers or mesh relay nodes [3].

2.1.2 Usage Scenarios

Due to their flexible structure WMNs have a wide range of application scenarios, which include:

- Community networks: Local authorities, such as cities or communities, operate mesh networks to provide Internet access to their citizens or tourists. Access to community networks can be free of charge or at very low costs. Well known examples of mesh community networks are Frei-Funk [4] in Germany or AirJaldi in the Himalaya region [5].
2. Background

Figure 1: Types of wireless mesh networks

- Hot-spot extension: Hot-spot operators can extend existing hot-spot infrastructures e.g. on airports or train stations to increase coverage and capacity (see [6]).

- Home networks: WMNs are used in private homes for the distribution of Internet access and multi-media content. This is in particular useful when Internet access has to be distributed over several rooms or floors or garden areas [7], as it can solve the access point positioning problem easily.

- Public security: Closed-circuit television (CCTV) systems are connected to control rooms via a WMN. Because of the ease of deployment, temporary installations are possible too, for example to monitor large events such as football championships or Olympic Games (e.g. [8]).

- Building automation: WMNs are a suitable technology for connecting sensors and actuators for building automation, especially for buildings where no cable infrastructure is present or deploying cables is impossible due to a preservation order (e.g. [9]).

- Disaster recovery: After natural disasters such as earthquakes or flooding, wireless mesh networks can be quickly deployed to replace damaged voice or data networks and to help coordinating rescue teams (e.g. [10]).
2.1.3 Classification of Wireless Mesh Networks

Two types of WMNs are common (depicted in Figure 1) [3]:

- **Infrastructure/Backbone WMN**: MSTAs and MAPs form a meshed wireless network that serves as a backbone for legacy clients. The clients connect to the MAPs via some other standard, such as IEEE 802.11, but do not implement any mesh services. Mesh portals can act as gateways to wired networks and other wireless technologies such as IEEE 802.16 or LTE.

- **Client WMN**: MSTAs form a mesh network and no MAPs are involved. In such a scenario, MSTA are typically mobile and subject to energy constraints, which is normally not the case for MAPs in a backbone WMN. Therefore, the requirements of client WMNs are different from infrastructure WMNs, for example to handle node mobility.

Several other types of wireless networks exist, which have some commonalities with WMNs. Mobile Ad-hoc Networks (MANETs) are wireless multi-hop networks, which are typically formed by mobile clients. They can be seen as a kind of client WMN. The term ad-hoc network is sometimes also used for legacy IEEE 802.11 ad-hoc networks, in which STAs do not forward traffic wirelessly over multiple hops. Wireless Sensor Networks (WSNs) might also use multi-hop transmissions similar to WMNs, but are usually much more restricted in terms of power consumption and processing power. They are therefore considered to be a separate class of networks.

In this thesis WMNs based on IEEE 802.11 are considered. It should however be noted, that other radio technologies are also used to build mesh networks. For example, IEEE 802.16 [11] also provides mesh functionality.

WMNs can also be classified by the number of radios and channels used, as depicted in Figure 2. Radio refers to the wireless network interface card and channel refers to the wireless communication channel. In WMNs of the first generation nodes only have one radio (single-radio mesh). Second generation WMNs use one dedicated wireless radio for client access and one for forwarding data and third generation WMNs forward data on multiple radios (multi-radio mesh).

Using multiple radios instead of only one requires different protocols for medium access, channel scheduling and channel assignment. For both single-radio and multi-radio WMNs, we present a few protocols in detail, which are used in Papers I-IV.
2. Background

2.1.4 Single-Radio Mesh Networks

Current WMNs predominately use Media Access Control (MAC) protocols based on Carrier Sense Multiple Access Collision Avoidance (CSMA/CA). Although there are a few examples of single-radio/multi-channel MAC protocols (e.g. [12] or [13]), they are far less popular than CSMA/CA single-radio/single-channel MAC protocols. Therefore, we focus our discussion on CSMA/CA-based protocols. CSMA/CA belongs to the class of listen-before-talking protocols. Before a node transmits a packet, it listens to the wireless channel (carrier sense) to detect ongoing transmissions. Only if the medium is idle, a node transmits.

A prominent example of a CSMA/CA-based protocol is the Distributed Coordination Function (DCF), which is the default MAC protocol for IEEE 802.11. DCF implements two modes of operation: In the basic mode, a station transmits after a backoff. If the transmission is successful, the receiver waits for Short Interframe Space (SIFS) and answers with an acknowledgement (ACK). If the frame (or the ACK) has not been received correctly, a timer expires at the sender and triggers a retry after a backoff procedure. In the other mode, each transmission starts with a request-to-send (RTS) and clear-to-send (CTS) handshake to virtually reserve the medium. Such control packets can improve the performance in the single-hop case as the data packets are usually larger compared to RTS-CTS control packets and more affected by collisions [14].

The backoff procedure is identical for both modes of operation and works as follows: Initially, a station chooses a backoff counter randomly and uniformly from \([0, W_0 - 1]\), where \(W_0 = CW_{\text{min}}\). After the channel was idle during a slot of length \(\sigma\), the backoff counter is decremented by 1. When the channel is busy,
the countdown is frozen, until the channel becomes idle again for a period of DCF Interframe Space (DIFS). When the countdown reaches 0, the station attempts to transmit. Each time a transmission fails, the station selects a new counter value, this time from \([0, \min(W_i - 1, CW_{max})]\), where \(i\) denotes the transmission attempt counter and \(W_i = 2^i W_0\).

\(CW_{min}\), \(CW_{max}\) and DIFS are configurable parameters. The Enhanced Distributed Channel Access (EDCA) is an improved variant of DCF that allows different \(CW_{min}\), \(CW_{max}\) and DIFS values, depending on the traffic type. Thereby the channel access of one traffic class (a so-called access class) can be prioritized over a second class. In addition, the EDCA introduces the concept of Transmission Opportunities (TXOPs). A TXOP is a time interval (specified by its length TXOPlimit), in which a node might transmit several packets separated only by SIFS, without contending for the medium. TXOPs can also be configured for each access class individually. EDCA is mandatory in IEEE 802.11s compliant devices. Optionally, the Mesh Coordinated Channel Access (MCCA) can be used in IEEE 802.11s [15].

CSMA/CA-based protocols like EDCA suffer from the hidden and the exposed node problem. A hidden node is a node which cannot sense the transmission of a neighbor and therefore transmits, which results in a collision with the neighbor's transmission. A node is called exposed, when it does not transmit because of an ongoing neighboring transmission, although it could transmit without causing a collision. Both hidden and exposed nodes reduce the performance. Multi-radio mesh networks can alleviate the problem to a certain extent, since fewer nodes contend for a channel and therefore the probability of a hidden or exposed nodes is lower.

In Papers I and III DCF is used, Paper II also uses features from EDCA.

### 2.1.5 Multi-Radio Mesh Networks

Multi-radio mesh networks naturally create the possibility of concurrent use of multiple channels to increase performance. This inherently poses the question of how to assign and schedule channels. According to [16], the classification of channel assignment protocols can be based on how frequently the channel assignments are performed, therefore the protocols and architectures for multi-radio/multi-channel networks can be classified as static, dynamic, semi-dynamic and hybrid.

In the static approach channels are assigned to radios for permanent use. With a dynamic channel assignment scheme, a node switches from one to another channel between two consecutive data transmissions. In contrast, semi-dynamic approaches assign or reassign channels at a larger time scale, for
2. Background

Hybrid schemes are a combination between static or semi-dynamic and dynamic channel assignment. Typically, one channel is assigned statically or semi-dynamically to one radio, whereas a second radio switches from channel to channel in a dynamic way.

The aim of channel assignment algorithms is to maximize one or several performance metrics such as throughput, while ensuring the connectedness of the network [17]. A network is called connected, if any node can reach any other node (possibly over multiple hops). A pair of nodes can only communicate if both have one radio tuned to the same channel. To achieve connectedness, a channel assignment algorithm either needs to synchronize channel switches (for dynamic schemes), or assign channels fixed over a longer period of time (static, semi-dynamic or hybrid). With static schemes, a node can at maximum use as many channels has it has radios. With dynamic or hybrid schemes it is possible to efficiently utilize a large number of channels, even though each node is equipped with only two radios.

The Net-X system [18], which is used in Paper IV, is an example of a hybrid approach, as it applies a semi-dynamic assignment to the fixed radio (used primarily for receiving data from neighbors) and a dynamic assignment to the switchable radio (used to transmit data to its neighbors). The semi-dynamic reassignments of the fixed radios are based on the current number of nodes using the same fixed channel. Therefore, if a node notices that the number of nodes using the same fixed channel as itself is larger, it can reassign its radio to a less used channel and inform its neighbors.

The channel used by the switchable radio may be changed at any time, without having to inform the neighbors. Thus, the switchable radio can be used to transmit to neighbors whose fixed radios may potentially be on different channels. This is illustrated in Figure 3. Node A tunes its switchable radio to Channel 2 to communicate with node B. Similarly, Node B, uses its switchable radio at Channel 3 to transmit data to Node C.
2.1.6 Characteristics, Challenges and Solutions

WMNs differ from normal LANs or WLANs in many aspects. This brings along new challenges for the design of protocols and requires new solutions, of which we discuss a few in this subsection.

Usually, WMNs operate in unlicensed bands, for example the ISM band at 2.4 GHz or the U-NII band at 5 GHz. In those frequency bands strong fluctuations in link quality due to external interference are common [19]. In addition fading causes variation of the channel quality. This necessitates specialized channel assignment protocols such as [20], which take into account self-inflicted and external interference.

Another feature of WMNs, which requires special attention, is the multi-hop communication paradigm. To enable it, efficient routing protocols have to be deployed. Compared to wired networks, links in mesh networks are characterized through a higher dynamicity and heterogeneity. Protocols like Optimized Link State Routing (OLSR) [21] and routing metrics like the Expected Transmission Count (ETX) [22] are tailored to multi-hop wireless networks and therefore achieve better performance than protocols from the wired domain.

An important requirement for WMN protocols is scalability, such that they operate efficiently in networks of a few nodes as well as in networks of hundreds of nodes. Distributed algorithms usually have better scaling properties than centralized ones and thus are preferable used in WMNs. Distributed protocols further improve resilience, since they do not have a single point of failure.

An important challenge in the design of WMNs is to ensure interoperability. A WMN can comprise different radio technologies, hardware platforms, protocols etc. This challenge can be met with architectures as for example proposed in the CARMEN project [23]. This architecture specifies technology independent interfaces, which enable interoperability.

Compared to wired networks, the throughput of WMNs is typically lower. Providing multi-media services such as streaming video or VoIP, which have strict Quality of Service (QoS) requirements, is hard [1]. Using admission control schemes like [24] or dynamic bandwidth control like [25], it is possible to improve the quality of service. Another approach is to increase the efficiency of multi-media service transmissions, which is one main theme of this thesis.

2.2 Voice over IP

The International Telecom Union (ITU) defines Voice over IP (VoIP) as the transmission of voice, fax and related services over packet-switched IP-based networks [26]. Services that were previously provided by Public Switched Tele-
phone Networks systems (PSTN) are run over IP-based networks such as LANs or the Internet. The terminals, i.e. the phones, are IP-enabled devices such as personal computers, smart phones or fixed IP phones.

In the following, an overview on the processing of audio signals in a VoIP system is provided, relevant standards for transport, coding and signalling are introduced and a description of the quality evaluation of VoIP is given.

### 2.2.1 Transmission of Audio Signals with Voice over IP

The processing chain of a VoIP system consists of several steps, which are illustrated in Figure 4. At the sender, the analog voice is recorded by a microphone and converted into a digital stream of data. The digital voice is encoded by a speech encoder, which outputs speech frames, each containing the encoded speech for a small time interval (e.g. 10 ms). One or more speech frames are then packed together and encapsulated in transport protocol headers. Finally, an IP header is added and the packet can be sent through an IP-based network. Optionally, the Voice Activity Detection (VAD) recognizes silent periods and suppresses the generation of packets to save bandwidth.

At the receiver side, the incoming packets are depacketized. Since the packets might not arrive in a constant flow, they are collected in a de-jitter buffer, that adjusts arrival time differences and the arrival order. If a packet is lost e.g. due to errors in the network transmission or excessive jitter, it can be substituted for by example duplicating the packet prior to it or by interpolation of received packets. This can conceal the loss to a certain extent. Finally, the receiver decodes the packets and outputs them via the sound card.

![Processing chain of a VoIP system](image)

**Figure 4: Processing chain of a VoIP system**
2.2.2 Relevant Standards

A VoIP system comprises components for audio processing, signalling and data transport. We briefly introduce standards for those tasks now. The system converting analogue speech to a digital format and back to an analogue waveform is called codec. The most important characteristics of a codec are the bit rate, the coding delay, the computational complexity and the perceptual quality. Prominent examples of audio codecs are Speex [27], G.711 [28], G.726 [29], G.729 [30].

Transport protocols for VoIP provide packet sequence numbers and time stamps to allow the detection of packet re-ordering and the synchronization of streams. Furthermore, they can give feedback from the network, for example about congestion events or buffer states, to the VoIP application. Commonly, no congestion control and packet retransmissions are done in VoIP transport protocols. Packet retransmissions are normally not useful for VoIP, because packet retransmissions would result in too high delay. By far the most used VoIP transport protocol is the Real Time Protocol (RTP) [31], which is often used along with the Real Time Control Protocol (RTCP). RTCP provides the application feedback about the network status. Both protocols are mostly encapsulated in the User Datagram Protocol (UDP), sometimes also the Stream Control Transmission Protocol (SCTP). Some applications, such as Skype implement proprietary transport protocols, which also include features such as encryption and firewall traversal [32].

A VoIP call consists of two uni-directional streams of compressed audio over an IP network. Signalling protocols are used to determine how the caller and the callee communicate with each other and how to setup those streams. Besides that, signalling protocols carry out a number of other tasks, such as terminating calls or registering a terminal at a central address register. In the currently most popular signalling protocols, H.323 and Session Initiation Protocol (SIP) [33], signalling is independent from the media flow (out-of-band signalling).

Coding, signalling and transport protocols are largely interchangeable: for example, one can use RTP as transport protocol for G.711 or G.723 and set up the session with SIP or H.323. This modularization makes VoIP flexible and new protocols can be easily deployed when new requirements and applications emerge.

2.2.3 Measuring Voice over IP Quality

Many components influence the quality of a VoIP transmission. In order to optimize the quality it is essential to have a precise definition of what quality is
and to define how to measure it. Quality measurement refers to the process of obtaining a quality measure or quality metric. A quality metric is a numerical value describing the perceived quality of a VoIP call or speech sample. The measurement process is called active or intrusive, when additional probe traffic is inserted into the system. In contrast passive or non-intrusive measurements do not use probe traffic.

There are two major approaches for measuring speech quality: subjective and objective methods. Subjective methods require humans to listen to audio samples and to judge their quality. The Mean Opinion Score (MOS) is the most widely used subjective quality metric. It is obtained as follows: A group of human testers listens to a set of speech samples. Each person puts each sample into a category according to her/his quality perception. The categories range from bad to excellent, or 1 to 5. The MOS of a sample is the average of all scores for this sample. ITU P.800 [34] describes in detail how a test for obtaining an MOS has to be set up. A clear advantage of subjective methods is that the judgment of test persons will reflect how real people perceive the quality. On the downside, subjective methods do not allow real-time operation, they are expensive and the test team needs to be large enough and skilled to provide reliable results.

In contrast, objective measurement methods infer how the quality will be perceived by humans through algorithmic means. The benefits of these methods are that no human interaction is necessary, real-time operation is possible, the operation is cheap and the results are reproducible. However, the outcome of the algorithms might not correlate with the human perception. The most popular objective quality model is the E-model, which is defined in ITU G.107 [35]. As it does not require a reference signal but only impairment parameters such delay, packet loss and codec distortion it is also called a parametric model. The E-model calculates the R-factor and does not require intrusive measurements. The core assumption of the E-model is that the impairments are independent and additive. Models were proposed (e.g. [36] or [37]) to convert an R-factor into a MOS. Thereby it is possible to predict the quality experienced by humans using a parametric model.

In Paper II, the quality of G.711 VoIP calls is analyzed using the E-Model under consideration of a de-jitter buffer. The R-factor values are converted into MOS values using [37].

2.3 Cross-Layer Design and Optimization

In a classical layered architecture, such as the Open Systems Interconnection model (OSI model) shown in Figure 5a, a protocol layer just makes use of the
services of adjacent layers. According to [38], cross-layer design is a method of designing network protocols by deliberately violating the rules of a layered reference architecture. Cross-layer design is interesting in particular for wireless networks, since many higher layer protocols have initially been designed for wired networks and typically only the MAC layer and the Physical Layer (PHY layer) are designed for wireless networks. As wireless networks commonly have different packet loss probabilities or other medium access delay characteristics than wired networks, higher level protocols might perform poorly on wireless networks (e.g. TCP). Tuning higher layer protocols, so that they are better suited for wireless networks, or tuning wireless networks so that they better support specific higher layer protocols can yield high performance gains. For example, in [39], a distributed power control algorithm is presented that jointly optimizes the throughput of existing TCP protocols and the energy efficiency of a wireless multi-hop network.

As stated in [40], two optimization approaches can be found: loosely or tightly coupled. In a loosely coupled optimization, one layer knows the parameters of an other layer and optimizes its operation according to it. In the tightly coupled approach, the parameters of two or more layers are jointly optimized.

### 2.3.1 Cross-Layer Design and Architectures

Cross-layer design can result in different cross-layer architectures [38]. In Figure 5b new interfaces are defined to exchange information between non-adjacent layers in a uni-directional or bi-directional way. This architecture is well suited for a loosely coupled cross-layer optimization. Figure 5c illustrates...
a vertical calibration architecture, which uses a shared database for information exchange. This architecture is in particular suitable for a tightly coupled optimization that spans across layers. Another approach in cross-layer design is the creation of completely new abstractions, which is shown in Figure 5d. Here, a new protocol layering is defined, for example by optimization decomposition. If the decomposition is done successfully, the maximum overall network utility can be achieved [40].

Each architecture has its strong and weak points. Direct communication between layers is a lightweight approach, which does not require many modifications to the existing layered reference architecture. However, its extensibility is limited. A shared database for the vertical calibration of layers provides a clean structure for extensions, but is slightly more complex than direct communication between layers. Creating completely new abstractions certainly brings along the highest degree of freedom, but also requires a completely new way of thinking. This design paradigm has gained popularity in the research community as part of the clean-slate design movement [41] in recent years.

In this thesis, two different cross-layer architectures are used to increase the quality of voice traffic over mesh networks. Paper III implements the shared database approach, while in Paper IV the application layer and the medium access layer directly communicate with each other.

2.3.2 Challenges and Solutions

Cross-layer design has been a very active research area in the past few years and has been applied successfully to different problems (e.g. [42], [43] and [44]). Nevertheless, major challenges remain:

First, the coexistence of different cross-layer solutions needs to be studied. If several cross-layer solutions are deployed simultaneously, they might unintentionally have negative effects on the overall system performance [45]. Second, when the principles of the standardized layered architectures are violated, it is then desirable to have new rules and standards for how to violate the reference architecture. Third, most cross-layer proposals are designed for specific network conditions. If the network does not run under those conditions, the optimizations will not perform well.

Deploying the solutions presented in Papers III and IV can hence lead to conflicts with other, already installed cross-layer solutions or deliver low performance, if the network is operated under conditions, which were not anticipated in the design of the solutions.

One promising concept to address those issues are so-called cognitive networks. Cognitive networks have the ability to learn from past experiences and
take into account end-to-end goals [46], while cross-layer optimizations are typically just local, memory-less adaptations, that perform the same optimization again even if result was poor in the past. We leave it as future work to perform the cross-layer optimizations of Papers III and IV in a cognitive way.

3 Challenges, Solutions and Research Questions

3.1 Challenges

VoIP is an integral service of today’s and the future Internet. Hence, it is critical for the success of WMNs to support VoIP. However, using current technology poses the following problem: The capacity and scalability of VoIP over WMNs is low because of the low transmission efficiency of small VoIP packets and the high requirements VoIP has on the network in terms of delay, packet loss and jitter.

To emphasize the low efficiency of small packet transmissions, Figure 6 depicts the various time periods spent in a packet transmission using the IEEE 802.11 DCF by comparing 160 and 2304 byte packets. A transmission consists of a backoff, followed by waiting DIFS, the transmission of PHY, MAC and IP headers and the data payload. The receiver waits for SIFS and answers with ACK, which also requires a PHY header.

For large packets (2304 bytes), the fraction of time spent for transmitting the payload in relation to the whole transmission time (= efficiency) is higher than for small packets. Also, the efficiency decreases when the PHY rate is increased (Figure 7), since several protocol overheads (such as SIFS and DIFS) have a fixed length that does not shrink with a higher PHY rate. This highlights the importance of transmitting large packets to achieve high efficiency, in particular in the wake of ever increasing PHY speeds.

Furthermore, VoIP only tolerates a small amount of packet loss and low one-way delay. As shown in Figure 8, one-way delays exceeding 200 ms result in a severe quality degradation. In addition, packet loss has a great impact on the perceived quality. For G.729, a packet loss of only 4% leads to a large number of dissatisfied users. While the delay requirements between different codecs only differ slightly, the packet loss requirements largely depend on the codec design. Theoretically it is possible to design a codec which tolerate as large loss fraction. However, this comes at the cost of high bandwidth requirements, which would render such a codec ill-suited for WMNs.
3. Challenges, Solutions and Research Questions

Figure 6: Transmission times (in µs) for packet length 160 bytes (left) and 2304 bytes (right) at 54 Mbit/s PHY rate

3.2 Solutions and Research Questions

Two possibilities to increase the VoIP capacity of WMNs are to enhance the transmission efficiency of small packets or use WMNs of the third generation, which allow multiple concurrent transmissions and thereby have higher capacities. Either approach needs to take into account the packet loss and delay requirements of VoIP to achieve high quality.

Solving those problems would have benefits for network operators and users. Network operators gain from a higher scalability of their networks, since it allows them to serve more users and thereby achieving higher revenues. End-users benefit from a better VoIP quality. In addition, an increased scalability makes mesh networks economically more viable, thereby giving benefits to both operators and users.

This thesis addresses those problems by investigating the following questions:

• **Question 1:** How to increase the transmission efficiency for small packets in IEEE 802.11-based WMNs?
  
Due to the high MAC and PHY overhead, the transmission of small packets in IEEE 802.11 is inefficient and thus the capacity is low. In Paper I we formulate an analytical model to investigate the relationship between the wireless channel utilization and the mean packet size and arrival rate in a single-cell wireless network. In Paper II we experimentally compare two burst transmission schemes, IEEE 802.11e TXOPs and IP packet aggregation. We show that both schemes can improve the transmission efficiency of small packets in IEEE 802.11-based WMNs.

• **Question 2:** How to use cross-layer optimization and IP packet aggre-
Figure 7: Transmission efficiency of a 160 byte packet

Figure 8: R-factor as a function of packet loss and one-way delay for G.711 and G.729. Source: [47]
4. Research Method

The method used in this thesis follows the common practice [49] of the engineering sciences and comprises the following steps: literature review, problem statement, hypothesis formulation, hypothesis testing and analysis. This process is iterative, meaning that after the analysis phase the hypothesis is refined and tested again, until the hypothesis can be accepted with a high confidence.

Literature review is done in order to identify the state-of-the-art and relevant problems. Subsequently, a research problem is stated and how a solution
to this problem advances the state-of-the-art. Based on the knowledge gained from the literature review, a hypothesis is formulated. The hypothesis delivers a potential explanation of some aspect of the system under consideration and allows making predictions.

In the next step the hypothesis is tested. In the performance analysis of computer systems the most common methods for hypothesis testing are analytical modeling, computer simulation or real-world experiments. An analytical model is a mathematical description of a system. In the process of formulating an analytical model, one needs to find a balance between complexity of the model and level of detail. Usually, a higher level of detail leads to more complexity, but makes the model more predictive. Computer simulations can include more details than analytical models, but still exhibit the same problem of finding a balance between complexity of the simulator and level of abstraction. Complex simulators are more likely to contain software bugs than simple ones. Thus, a higher level of complexity does not necessarily lead to more accuracy [50]. Also, the simulation run-time increases with the complexity of the simulator. Real-world experiments have the lowest level of abstraction, but the system under investigation needs to exist and environmental factors are hard to control. Also, for cost reasons real-world experiments are usually only possible for small number of scenarios, which makes it hard to draw general conclusions from them.

Each of the hypothesis testing methods has its advantages and disadvantages, which need to be considered when selecting the method. However, it is important to understand, that neither of the methods should be solely used to test a hypothesis. As a best practice [50], all three methods should be used to validate each others results. Only when one has confidence that the model, the implementation of the simulator or real system are correct, one should use them for hypothesis testing.

In the following, we describe by the example of Paper I how the research method was applied: In this paper we analyzed the use of the channel busy fraction as indicator of available bandwidth. The literature review showed that current approaches mainly are probe-based, which induces additional overhead. Therefore, the problem we stated was how to obtain the available bandwidth using passive measurements. Based on the study of related work, we formulated the hypothesis, that the channel busy fraction is a good indicator of the available bandwidth. To test this hypothesis, we formulated an analytical model of the channel busy fraction based on an embedded time Markov chain and implemented a measurement system for the channel busy fraction in the KAUMesh testbed. We chose not to use computer simulations here, since the testbed was readily available and the implications of different experimental designs were well understood. We validated the model against the implemen-
5. Summary of Papers and Contributions

In this section we summarize the related work and background material, highlight the most important outcomes and contributions and discuss shortcomings of the presented papers.

5.1 Paper I - Theoretical and Experimental Analysis of the Channel Busy Fraction in IEEE 802.11

The congestion level on the wireless channel is an important information for the operation and optimization of IEEE 802.11 networks, for example to perform admission control. Traditionally, the congestion level has been estimated with probe packets, for example in the ETX routing metric [51]. However, probe packets create additional traffic and there is an inherent trade-off between accuracy and probe frequency. More recently, it has been proposed to use passive measurements, such as capturing all packets in RF-monitor mode (e.g. [52]) or using the Clear Channel Assessment (CCA) (e.g. [53]) instead. The CCA indicates whether there is an ongoing transmission. Using this information, one can calculate the channel busy fraction, i.e. the fraction of time the channel is sensed busy. Previous research has studied the channel busy fraction in context of a specific application only (e.g. [53] or [54]).

In Paper I we present a thorough evaluation of the relationship between the busy fraction and other important characteristics such as the collision probability and throughput. Our main contributions are:

• An analytical model, that is capable of predicting the channel busy fraction as a function of traffic arrival rates, packet size and network size.

• A validation of the model with measurements in the KAUMesh testbed.
As shown in Figure 9 the predictions from the model match measurements well.

- A simple, but accurate method of estimating the available bandwidth. We show that the channel busy fraction allows an accurate prediction of the available bandwidth with an error smaller than 70 kbit/s.

The main limitation of our analytical model is the focus on single cell networks and the inability to handle hidden terminals. Our model is based on an embedded time Markov chain [55], which requires well defined slot-boundaries in the state transitions. Unfortunately, this cannot be guaranteed in the presence of hidden terminals. Alternative formulations of the problem are possible, for example using the approach in [56]. However the resulting model is by far more complex.

Estimating the available link bandwidth is useful in certain scenarios, for example in single-cell WLANs. However, predicting the available bandwidth of a path that traverses multiple wireless hops such as found in WMNs is more challenging. Due to intra-path interference in multi-hop transmissions, the estimation of the available end-to-end bandwidth is more complex and requires a different modeling approach.
5. Summary of Papers and Contributions

5.2 Paper II - An Experimental Comparison of Burst Packet Transmission Schemes in IEEE 802.11-based Wireless Mesh Networks

Using the IEEE 802.11 distributed coordination function (DCF) as MAC layer, a node needs to contend for the medium each time it wants to transmit a packet. This creates high overhead in particular for small packets and leads to poor performance for real-time applications such as Voice over IP (VoIP) or online gaming.

Burst packet transmission can increase the efficiency. For example, with the Transmission Opportunity limit (TXOPlimit) in IEEE 802.11e, a station may transfer several packets without contending for the channel in between. Similarly, IP packet aggregation combines several IP packets together and sends them as one MAC Service Data Unit. Originally, both schemes have been developed for single-hop networks only. Thus the impact on WMNs is unclear if the packets need to contend over multiple hops.

As the main contribution of Paper II we present measurements from a 9-node WMN testbed to compare TXOPs and IP packet aggregation for VoIP in terms of fairness, network capacity and quality of user experience. Our most important insights are:

![Figure 10: Cumulative distribution of MOS with 12 VoIP calls with TXOPs, RTS/CTS and IP packet aggregation (values from 50 different scenarios)](chart.png)
• For low networks loads, both TXOPs and IP packet aggregation increase the VoIP quality compared to IEEE 802.11 DCF.

• In multi-hop transmissions traffic is typically not backlogged. While IP packet aggregation artificially delays packets prior to aggregation, IEEE 802.11e just uses the medium access delay to buffer packets to be sent within one TXOP. Therefore IP packet aggregation can create larger burst sizes and yields a higher efficiency than TXOPs.

• For highly loaded networks, represented by Figure 10, the VoIP quality for standard IEEE 802.11 is poor. Only a small fraction of the calls receive an acceptable MOS (> 3.5). TXOPs and IP packet aggregation significantly increase the number of high quality calls. Interestingly, the use of a RTS/CTS handshake is counterproductive when using TXOPs or IP packet aggregation, although it should help to remedy the impact of collisions and hidden nodes when long packets are transferred. A potential explanation for this behavior is the creation of exposed nodes by RTS/CTS, which are received by far distant nodes and thereby unnecessarily block distant transmissions.

As one of the main shortcomings of Paper II, it does not include the A-MSDU and A-MPDU schemes of IEEE 802.11n in the performance evaluation. Also, the interaction with other network functions such as routing and the impact of hidden nodes should be studied in future work.

5.3 Paper III - FUZPAG: A Fuzzy-Controlled Packet Aggregation Scheme for Wireless Mesh Networks

Packet aggregation increases the capacity of IEEE 802.11-based WMNs by aggregating small packets into larger ones and thereby reducing overhead. In order to have enough packets to aggregate, packets need to be delayed in a buffer. Current aggregation mechanisms use fixed buffer delays or do not take into account the delay characteristics of the saturated IEEE 802.11 MAC layer.

By varying the buffer delay it is possible to increase or decrease the aggregation efficiency and thereby the load on the network. For a given traffic input rate (e.g. 5 Mbit/s), larger packets are transmitted more efficiently and thus use fewer channel resources (less overhead, fewer collisions) than smaller packets. However, too large buffer delays lead to large end-to-end latency, which is disadvantageous for VoIP. For low network loads it is not necessary at all to artificially delay packets in the buffer. As shown in Paper I, the channel busy fraction is a good indicator for the network load and therefore helps to find a good buffer delay.
5. Summary of Papers and Contributions

In Paper III, we present FUZPAG, a novel packet aggregation architecture for IEEE 802.11-based wireless mesh networks. FUZPAG uses Fuzzy Control to determine a reasonable aggregation buffer delay under the current channel utilization. FUZPAG selects the minimum buffer delay which is required to transmit packet sizes large enough to keep the network right before saturation state. In this state the collision probability and thus medium access delay is low. By cooperation among neighboring nodes FUZPAG distributes the buffer delay in a fair way.

We have implemented the system on Linux and evaluated it in KAUMesh testbed. For different network topologies we show that FUZPAG outperforms standard aggregation in terms of end-to-end latency under a wide range of traffic. Figure 11 shows the end-to-end latency of UDP flows when no aggregation is used (NOAGG), static buffer sizes are configured (AGG-bufferdelay) and FUZPAG selects the buffer delay (FUZPAG). The result show that FUZPAG chooses a buffer delay which results in a low end-to-end latency, while static schemes might add too little or too much delay, depending on the network load. The low end-to-end latency of FUZPAG is important for VoIP.

The main contributions of Paper III are the definition and implementation of a modular cross-layer optimization system that implements the shared database approach and an algorithm for dynamically adapting the aggregation delay based on the network load. As major improvement to existing works (e.g. [57]), we estimate the channel load from the clear channel assessment (CCA) data of IEEE 802.11 (as described in Paper I) and through cooperation distribute the buffer delay among nodes in a fair way.

As potential future improvements of FUZPAG, the convergence time of the controller should be reduced to make it more suited when traffic rates vary fast.

5.4 Paper IV - QoS-Aware Channel Scheduling for Multi-Radio/Multi-Channel Wireless Mesh Networks

In non-static multi-radio-multi-channel wireless mesh networks architectures such as Net-X [18], mesh nodes need to switch channels in order to communicate with different neighbors. If the channel scheduler does not consider the requirements of real time traffic such as VoIP, this can lead to excessive delay or jitter and low VoIP quality.

In Paper IV we propose a channel scheduler for the Net-X platform that takes into account the priority of the currently used channel and the priority of all other channels, which have packets to send. The scheduler first serves channels with high priority traffic and afterwards channels with low priority.
traffic. The scheduling pattern is chosen in a way to minimize delay and jitter for high priority traffic, but still giving good throughput to low priority traffic. A configurable parameter allows reducing jitter on cost of throughput or vice versa.

The evaluation of the algorithm in the KAUMesh testbed shows that it outperforms the standard round-robin scheduler both in terms of average delay and jitter. The 90-percentile of end-to-end packet delay is around 30 ms lower with the QoS-aware scheduler (Figure 12).

The main contributions of Paper IV are the definition and analysis of a QoS-aware scheduling algorithm, its implementation in the Net-X platform and its evaluation.

The proposed algorithm schedules channels on local knowledge only. Including neighbor information to coordinate channel switches could further decrease the end-to-end delay and jitter. Also in our performance evaluation we make use of static traffic priorities among flows. Dynamically assigning packet priorities based on the already experienced delay or jitter promises further improvements.
6 Conclusions and Outlook

In this thesis we have investigated the feasibility of voice over single-radio and multi-radio WMNs. We have shown that through the optimizations proposed in this thesis the capacity can be significantly improved compared to current protocols. Yet, many questions are still open and need to be investigated. In particular, future work has to address two major shortcomings of this thesis:

First, this thesis uses the unrealistic assumption that all traffic originates from mesh nodes. In a real WMN, most traffic originates from end-user devices which connect to the mesh network through some access network, such as WLAN. This leads to different traffic arrival patterns and network loads. Also, other types of traffic found in currently popular applications, such as video could be used. Second, the proposed cross-layer optimizations are domain specific and do not make use of potential benefits from learning the network behavior. To deal with the first issue, the interaction between WLAN access to a WMN and the WMN backbone network need to be studied. The second issue can be addressed by applying the cognitive network paradigm to the discussed cross-layer optimization problems.

On the methodological side, it is required to give more comprehensive mathematical descriptions of the system and perform experimental evaluations that incorporate data from real user traces and operational networks. A more sophisticated mathematical model should lead to a deeper understanding of the
processes affecting the performance of WMNs. Using real user data helps to avoid wrong assumptions about traffic patterns, network loads and deployment scenarios.

References


REFERENCES


REFERENCES


Theoretical and Experimental Analysis of the Channel Busy Fraction in IEEE 802.11

Reprinted from

Proceedings of Future Network & Mobile Summit 2010, Florence, Italy, June 2010
Theoretical and Experimental Analysis of the Channel Busy Fraction in IEEE 802.11

Peter Dely, Andreas Kassler and Dmitry Sivchenko
{peter.dely | andreas.kassler}@kau.se, dmitry.sivchenko@telekom.de

Abstract

Optimizing the operation of IEEE 802.11 networks requires estimating the load of the wireless channel. The channel busy fraction, which is the fraction of time in which the wireless channel is sensed busy due to successful or unsuccessful transmissions, can be used as indicator for this purpose. It can be obtained from e.g. the IEEE 802.11k channel load report or hardware-specific interfaces. Previously, the channel busy fraction has been used as a metric for different purposes such as routing and admission control. However, a thorough evaluation of the relationship between the busy fraction and other important characteristics such as the collision probability and throughput is missing. In this paper, we present an analytical model to study the channel busy fraction in non-saturated IEEE 802.11 networks. We validate the model with measurements in a testbed. The predictions from the model match measurements well. Furthermore, we demonstrate how to apply the model to estimate the available link bandwidth. Using measurements obtained from a testbed operated at 6 Mbit/s, we show that the channel busy fraction allows an accurate prediction of the available bandwidth with an error smaller than 70 kbit/s.

1 Introduction

In IEEE 802.11-based WLANs and wireless mesh networks (WMNs) mechanisms such as routing, admission control and channel assignment require an accurate estimation of the local congestion level on a wireless channel. Congested networks manifest themselves in high latency, jitter and packet loss, which should be avoided in order to provide predictable performance. Traditionally, the congestion level estimation has been performed using special probe packets. For example, the widespread ETX metric [1] is calculated by sending probe packets to neighbors and calculating the loss ratio of the probe
packets. The ETX is used as an indicator for the congestion level and the collision probability. Probing mechanisms are easy to implement and work on different hardware platforms. However, they create additional traffic and the probing frequency is a trade-off between accuracy and overhead.

To alleviate those problems some researchers proposed to estimate the channel load by using the monitor mode of the network card [2]. However, this creates a non-negligible processing overhead, since all packets on the channel need to be decoded and processed by the operating system. The need for interfaces to directly estimate the channel load has been recognized by hardware manufacturers and standardization bodies. For example, newer Atheros WLAN-chipsets offer direct access to channel load information. Also, the IEEE 802.11k standard [3] requires the implementation a Channel Load Report. Both the proprietary Atheros interface and IEEE 802.11k expose the channel busy fraction, which is the fraction of time in which the wireless channel is sensed busy due to own or neighbor activity. It has been used as a metric for different purposes.

For instance, Zhai et al. [4] derive analytical model which predicts packet loss, delay and jitter for the IEEE 802.11 distributed coordination function (DCF). They observe that the channel busy fraction is a good indicator for the collision probability. However, their model does not allow correlating the channel busy fraction with the traffic pattern and the MAC layer configuration.

Acharya et al. [5] use the channel busy fraction in a rate-adaptation algorithm. The channel busy fraction helps to distinguish between bit-error related losses and losses due to collisions. Sheriff et al. [6] use the channel busy time for admission control in wireless mesh networks. Based on a model which relates new traffic flows with the channel busy fraction, they deduce rules for an admission control system, which they evaluate in a testbed. However, their model needs as input measured quantities such as the ETX and thus has limited use for the general study of the channel busy fraction.

Salonidis et al. [7] employ the channel busy fraction for identifying high throughput paths in wireless mesh networks. They use the local channel busy fraction as a measure for the intra-flow and inter-flow interference. A new routing metric is derived, which according to simulation results outperforms ETX.

Previous research has studied the channel busy fraction in context of a specific application. What is missing is a generic model that allows relating the channel busy fraction with key parameters such as the network size, the MAC layer configuration and the offered load. This is desirable, since this would enable the use of the channel busy fraction in other contexts. We thus, as the key contribution of this paper, address this problem by extending Malone’s model
for IEEE 802.11 DCF under non-saturation conditions [8] to incorporate the channel busy fraction. By using the new model we can study the channel busy fraction as a function of different key parameters. The results obtained from experiments in a real testbed reveal a close relationship between the measurements and the model predictions.

The remainder of the paper is organized as follows: Section 2 introduces an analytical model of the IEEE 802.11 DCF under non-saturation conditions, extends it to incorporate the channel busy fraction and discusses the model. In Section 3, we validate this model using testbed measurements. Section 4 demonstrates how to apply the model for available bandwidth estimation. Finally, Section 5 concludes the paper.

2 Analytical Model

In this section we first review Bianchi’s model of IEEE 802.11 DCF. We then present Malone’s model for non-saturated networks, in which we subsequently incorporate the channel busy fraction.

2.1 IEEE 802.11 DCF under Saturation Conditions

Bianchi’s seminal paper [9] presents an analytical model of IEEE 802.11 DCF for a homogeneous network with \( n \) stations under saturation conditions. The back-off process is described by a Markov-chain with states \((i, k)\). \( i \) (from 0 to \( m \)) denotes the backoff stage, which is increased after each failed transmission. After a successful transmission \( i \) is reset to 0. \( k \) is a counter for the backoff-window, which is initially chosen randomly and uniformly from \([0, W_i - 1]\), where \( W_i = 2^i W_0 \) and \( W_0 = CW_{\text{min}} \). When the channel is idle during a slot of length \( \sigma \), \( k \) is decremented by 1. When \( k = 0 \) the station attempts to transmit. Bianchi shows how to calculate the steady-state probabilities \( b(i, k) \) of the Markov chain, in particular the probability of the ready-to-send state \((0, 0)\), which is given as

\[
b(0, 0) = \frac{2(1 - 2p)(1 - p)}{(1 - 2p)(W_0 + 1) + pW_0(1 - (2p)^m)}.
\]  

The probability \( p \), that a packet experiences a collision when being transmitted, is given by

\[
p = 1 - (1 - \tau)^{n-1}.
\]
\[
P_s = \frac{a_{\text{PHY}} - \tau}{1 - \tau},
\]
\[
P_{tr} = 1 - (1 - \tau)^n
\]
\[
T_s = T_{\text{DIFS}} + T_{\text{PHY}} + T_{\text{MAC}} + T_{\text{DATA}} + \gamma + T_{\text{SIFS}} + T_{\text{ACK}} + \gamma
\]
\[
T_c = T_{\text{DIFS}} + T_{\text{PHY}} + T_{\text{MAC}} + T_{\text{DATA}} + \gamma + T_{\text{SIFS}} + T_{\text{ACKtimeout}}
\]
\[
E = T_{\text{DATA}}
\]
\[
T_{\text{PHY}}, T_{\text{MAC}}, T_{\text{ACK}} \text{ and } T_{\text{DATA}}
\]
\[
\sigma
\]
\[
\gamma
\]

Table 1: Used symbols and description

The probability \( \tau \) denotes that a node transmits in a given slot and is calculated as

\[
\tau = \frac{b(0, 0)}{1 - p}.
\]

Finally, equations 3 and 2 form a system of two equations, which can be solved for \( \tau \) and \( p \).

The system throughput \( S \) is defined as the ratio of time spent on payload transmissions and time spent by successful transmissions, collisions or in an idle channel (see also Table 1). It can be written as

\[
S = \frac{P_s P_{tr} E}{(1 - P_{tr})(\sigma + P_{tr}P_s T_s + P_{tr}(1 - P_s)T_c)}.
\]

### 2.2 IEEE 802.11 DCF under Non-Saturation Conditions

Malone et al. [8, 10] extended Bianchi’s model to also account for non-saturated and heterogeneous networks. For simplicity, we only present here how to apply the model to the homogeneous networks. For details on the model and an extensive discussion the reader is referred to [8].

The key idea is to introduce additional states \((0, k)\), which represent the case where a node has transmitted a packet and no further packet waiting to be transmitted. In particular, the probability \( b(0, 0) \) is of interest. In the stage
2. Analytical Model

(0, 0), a node has completed the counter decrement, but no packet to transmit. The inverse of the steady-state probability of the state (0, 0), is

\[
1/b_{0,0} = (1 - q) + \frac{q^2 W_0 (W_0 + 1)}{2(1 - (1 - q)W_0)} + \frac{q (W_0 + 1)}{2(1 - q)} \left( \frac{q^2 W_0}{1 - (1 - q)W_0} + \frac{W_0}{1 - (1 - q)W_0} \right)
\]

\[
+ p (1 - q) - q (1 - p) \right) + \frac{pq^2}{2(1 - q)(1 - p)} \left( \frac{W_0}{1 - (1 - q)W_0} - (1 - p)^2 \right)(2W_0 - \frac{1 - p - p(2p)^{m-1}}{1 - 2p}) + 1).
\]

In analogy to eq. 1 the transmission probability \( \tau \) is

\[
\tau = b_{0,0} \frac{q^2}{1 - q} \left( \frac{W_0}{(1 - p)(1 - (1 - q)W_0)} - (1 - p) \right).
\]

\( q \) denotes the constant probability, that at least one packet arrives within the expected duration of a state in the Markov chain. It is related with the input rate and the traffic pattern. For a Poisson arrival pattern and small buffers, \( q \) can be approximated by \( q = 1 - e^{-\lambda T} \), with \( T \) being the expected duration of a state computed by the denominator of eq. 4 and \( \lambda \) being the packet input rate (pkts/s). [8] also provides a more precise definition of \( q \), which does not utilize mean state durations, but requires a complicated calculation of mean MAC delay. Since simulation results show that the approximation is satisfactorily precise, we subsequently use the approximation. By substituting \( \tau \) in eq. 2 with eq. 6 the collision probability \( p \) is obtained. Finally, eq. 4 gives the system throughput.

2.3 Modeling the Channel Busy Fraction

Based on Bianchi’s and Malone’s model we define the channel busy fraction. A channel is busy either due to successful transmissions for a period of \( P_s T_s \) or due to collisions for a period of \( P_c (1 - P_s) T_c \). The channel busy fraction is the ratio of the busy periods and all channel states (busy or idle), which can be expressed as

\[
 cbf = \frac{P_s P_c T_s + P_s (1 - P_s) T_c}{(1 - P_s) \tau + P_s P_c T_s + P_s (1 - P_s) T_c}.
\]
successful transmissions as busy, eq. 7 also takes into account medium usage by collisions. By that, eq. 7 is more accurate, in particular for higher loaded networks.

2.4 Discussion

Next, we apply the model (parameters see Table 2) to study the channel busy fraction in different settings. Figure 1a shows the aggregate throughput (solid lines, primary y-axis) and the busy fraction (dashed lines, secondary y-axis) for 4, 8 and 12 nodes and 1400 byte UDP packets. In the rest of the paper packet sizes denote the UDP payload length (without UDP/IP header) and the aggregate throughput is the sum of the UDP throughputs for all nodes. As already known from [8], the peak performance is reached before saturation. Going beyond this point towards a saturated network, will result in more collisions, but not more throughput. Rather the throughput decreases, especially for larger networks.

While the channel busy fraction at which the peak performance is achieved is fairly insensitive to the network size, the packet size plays a more important role. Figure 1b displays the busy fraction which gives the maximum throughput for 200, 500 and 1400 byte UDP packets. For 1400 byte packets the peak performance is reached at the channel busy fraction of about 95%, almost irrespective of the network size. For 200 byte packets the performance peak is already reached at 92% busy fraction. With smaller packets the stations access the wireless channel more frequently and thus have a higher chance of creating a collision.

2.5 Limitations of the Model

The model is based on two assumptions: an error-free channel and perfect carrier sensing. The first assumption implies that packets are only lost due to collisions, not due to bit errors induced by the channel noise. Following the argumentation in [11], a crude approximation of the effect is possible by introducing a new probability $p_e$, which denotes that a data packet is corrupted by bit errors. For space limitation we do not present this model extension here. The second assumption is only valid if there are no hidden or exposed nodes. The use of RTS/CTS, transmission power control and tuning of the carrier sensing threshold alleviates the problem to some extend, but will still lead to inaccurate predictions by the model. Yet, even the imprecise predictions can be useful. One could apply the model and use information about a significant difference between the model prediction and real measurements for example...
2. Analytical Model

(a) Channel busy fraction and throughput as function of aggregate injection rate

(b) Channel busy fraction which gives peak performance

Figure 1: Model results for 4, 8 and 12 nodes
as an indicator of hidden nodes.

3 Validation of the Model

In this section we validate the model against testbed measurements.

3.1 Experimental Setup

WLAN cards that implement IEEE 802.11k [3] present the channel busy fraction in the channel load report. In our testbed we used cards which are based on the wide-spread Atheros 5212 802.11a/b/g chipset and do not support IEEE 802.11k. However, in this chipset the registers PROFCNT_RXCLR (0x80f4) and PROFCNT_CYCLE (0x80f8) contain information about the number of timeslots that were sensed busy due to the clear-channel-assessment (CCA) and the total number of timeslots that have passed. The length of a timeslot depends on the card clock and is approximately $2 \times 10^{-8}$ seconds for Atheros 5212.

According to IEEE 802.11a [12] the CCA reports the channel idle if the energy level is below a threshold (-82 dBm if a preamble was detected, -62 dBm otherwise). Optionally, the network allocation vector (NAV) can be an indicator if the medium is busy. We extended the driver to export the content of the two registers to the /proc-filesystem. A userspace application reads the values every 500 ms and calculates the channel busy fraction as PROFCNT_RXCLR/PROFCNT_CYCLE. The model assumes that the medium is also busy during periods of SIFS or DIFS. Since the CCA detects those periods as idle we correct the measured PROFCNT_RXCLR by a $T_{SIFS}$ and $T_{DIFS}$ for each overheard or sent packet (using antenna statistics from the athstats utility). In contrast to IEEE 802.11k, the NAV from RTS/CTS packets is not considered in the proprietary Atheros interface for determining the channel busy fraction.

We have validated the model with five Cambria GW2358-4 network computers placed on a desk in a lab room. Each node is equipped with three network cards. The nodes run Linux 2.6.22 and MadWIFI 0.9.4. One node acted as receiver, four nodes generate UDP traffic with mgen [13]. Since the backoff processes run independently on all cards, four nodes are sufficient to emulate 12 senders. We sent UDP packets with 1400 bytes payload according to a Poisson arrival process. In Figure 2 each datapoint represents the average of a 4 minute testrun. Further settings are listed in Table 2.
3. Validation of the Model

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_{SIFS}$</td>
<td>$16 \times 10^{-6}$ sec.</td>
</tr>
<tr>
<td>$T_{DIFS}$</td>
<td>$34 \times 10^{-6}$ sec.</td>
</tr>
<tr>
<td>Backoff slot time $\sigma$</td>
<td>$9 \times 10^{-5}$ sec.</td>
</tr>
<tr>
<td>Data rate/Basic rate</td>
<td>6 Mbit/s</td>
</tr>
<tr>
<td>ACK timeout $T_{ACKTIMEOUT}$</td>
<td>$25 \times 10^{-6}$ sec.</td>
</tr>
<tr>
<td>$\text{CWMIN}/W_c$</td>
<td>15</td>
</tr>
<tr>
<td>$\text{CWMAX}$</td>
<td>1023</td>
</tr>
<tr>
<td>Max. number of retransmissions</td>
<td>5</td>
</tr>
<tr>
<td>Payload length (incl. UDP/IP header)</td>
<td>1428 bytes</td>
</tr>
<tr>
<td>RTS/CTS</td>
<td>Disabled</td>
</tr>
</tbody>
</table>

Table 2: Parameters of the model and the experiments

3.2 Channel Busy Fraction and Traffic Injection Rate

We now compare the predictions of the model with the measurements from our testbed. Figure 2a depicts the channel busy fraction as a function of the injected traffic. For aggregate injection rates smaller than 5 Mbit/s the channel busy fraction grows roughly linearly with the traffic injection rate. In these low loaded conditions the impact of collisions is low and the numerator of eq. 7 is governed by $P_{tr}P_sT_s$, while $P_{tr}(1 - P_s)T_c$ is small. What follows from Figure 2a is that $cbf \approx c \times S$, where $c$ is a constant depending on the data rate and packet size. After 5 Mbit/s the busy fraction increases faster than the injection rate, since collisions are more likely then. The model slightly underestimates this disproportionate growth and the maximum achievable channel busy fraction.

Next, in Figure 2b shows the average number of transmissions as a function of the busy fraction. From `/proc/net/madwifi/athX/ratestats.1600` we read the average number of packet transmissions. In the model it is obtained using eq. 2 as input to $\text{avg}(p) = \sum_{m=1}^{\infty} p^{m-1}(1 - p)i$. We use $\text{avg}(p)$ to calculate the average number of transmissions for a packet and compare it with the value from the `/proc`-file. As in [9] we assume the collision probability $p$ to be the same for each of the at maximum $m$ transmission attempts and that the attempts are independent of each other.

As mentioned above, for low loaded networks the average number of transmissions is close to one. Only when the load and thus the busy fraction increases, the average number of transmissions rises sharply at a busy fraction of about 90%. The model underestimates the average number of collisions and is more accurate for small networks (4 nodes). While in the model the transmission number is insensitive to the network size, in the experiments there is
a difference between small and large networks.

Despite great efforts, we were not able to find the cause why the quantitative predictions of the model are inaccurate in the transition from a non-saturated to a saturated network. We assume one or a combination of the following reasons: queuing effects, traffic generation and a non-standard back-off process. The modeled relation between $q$ and $\lambda$ is an approximation for very short queues. In the experiment we used a small hardware buffer (5 packets) to reduce effects caused by variations in the time from packet generation until delivery to the hardware. Using simulations [14] has shown that the buffering process has a significant impact on the throughput and collisions. Also, due to randomly occurring interrupts the arrival process of packets at the hardware is not exactly the Poisson one (although generated in Poisson manner with mgen [13]). Finally, according to [15], many IEEE 802.11 cards do not back-off according to the IEEE 802.11 standard on which the model is based.

4 Application: Available Bandwidth Estimation

The available bandwidth $AVBW$ of a wireless link denotes the bandwidth a flow can use, without disturbing ongoing flows. Previously, the channel busy fraction has been used as indicator for the available bandwidth in [4] and [5]. Here we present the estimation of the available bandwidth as one application of our model.

Following the observation in Section 3.2 that the channel busy fraction scales almost linearly with the traffic injection rate, we define the available bandwidth as

$$AVBW = (0.96 - cbf) \times C$$

where $C = \frac{\text{payloadlength}}{T_s}$ and 0.96 means that we only utilize the network to 96%. While not utilizing the network to 100% seems to be a waste of resources at first, it gets clear from Figures 1a and 1b that a higher utilization of the network will just result in more collisions and subsequently high delay and jitter. The aggregate network throughput would then not increase. A negative $AVBW$ (for $cbf > 0.96$) indicates that the network is overloaded and that the sending rate should be reduced.

We have evaluated this approach in the testbed using the following setting: node A sends 1400 byte UDP datagrams at 0.5, 1, 2, 3 and 4 Mbit/s to node B. Node C senses the initial busy fraction and estimates the available bandwidth $AVBW$ using eq. 8. Node C then sends a UDP stream to node D. It starts sending the UDP stream at a rate of $0.9 \times AVBW$ and increases up to $1.1 \times AVBW$ in steps of 1% every 30 seconds. Thereby, we measure the actual available
Figure 2: Comparison of model prediction and experimental results for 4, 8 and 12 sender
Theoretical and Experimental Analysis of the Channel Busy Fraction in IEEE 802.11

<table>
<thead>
<tr>
<th>Traffic Injection Rate (Mbit/s)</th>
<th>Initial Channel Busy Fraction</th>
<th>Estimated Available Bandwidth (Mbit/s)</th>
<th>Actual Available Bandwidth (Mbit/s)</th>
<th>Estimation Error (Mbit/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.09</td>
<td>4.81</td>
<td>4.80</td>
<td>0.01</td>
</tr>
<tr>
<td>1.0</td>
<td>0.18</td>
<td>4.31</td>
<td>4.38</td>
<td>-0.07</td>
</tr>
<tr>
<td>2.0</td>
<td>0.36</td>
<td>3.30</td>
<td>3.25</td>
<td>0.06</td>
</tr>
<tr>
<td>3.0</td>
<td>0.55</td>
<td>2.26</td>
<td>2.31</td>
<td>-0.05</td>
</tr>
<tr>
<td>4.0</td>
<td>0.73</td>
<td>1.25</td>
<td>1.29</td>
<td>-0.04</td>
</tr>
</tbody>
</table>

Table 3: Available bandwidth estimation using the channel busy fraction

bandwidth as the rate at which node C can send, such that either stream has less than 0.1% packet loss (after MAC layer retransmissions). We accept 0.1% packet loss since even in low loaded network a small number of packets can get lost due to periodic card re-calibrations.

Table 3 shows the actual available bandwidth, the estimated available bandwidth using the channel busy fraction and the estimation error. A negative estimation error means that the estimated available bandwidth is lower than the actual available bandwidth. For all cases the error is small, within 0.07 Mbit/s. Furthermore, increasing the rate of the UDP stream from C at steps smaller than 1% might improve the result further.

5 Summary and Conclusion

In this paper, we have explored the channel busy fraction by an analytical model and by testbed experiments. We have shown that our model predicts the testbed results qualitatively and quantitatively reasonable well in low loaded network. In higher loaded networks the model provides correct qualitative predictions. Using the extended model we have derived a simple and accurate method for estimating the available bandwidth of a WLAN link. In future work we plan to apply the proposed model to tackle different problems such as admission control and channel assignment.

References


An Experimental Comparison of Burst Packet Transmission Schemes in IEEE 802.11-based Wireless Mesh Networks

Reprinted from

Proceedings of IEEE Global Telecommunications Conference (GLOBECOM) 2010, Miami, Florida, December 2010
An Experimental Comparison of Burst Packet Transmission Schemes in IEEE 802.11-based Wireless Mesh Networks

Peter Dely, Andreas Kassler, Nico Bayer and Dimitry Sivchenko

{peter.dely | andreas.kassler}@kau.se, {nico.bayer | dimitry.sivchenko}@telekom.de

Abstract

Wireless Mesh Networks (WMNs) are wireless multi-hop networks comprised of mesh routers, which relay traffic on behalf of clients and other nodes. Using the standard IEEE 802.11 distributed coordination function (DCF) as MAC layer, a node needs to contend for the medium each time it wants to transmit a packet. This creates high overhead in particular for small packets and leads to poor performance for real-time applications such as Voice over IP (VoIP) or online gaming. Burst packet transmission can increase the efficiency. For example, with the Transmission Opportunity limit (TXOPlimit) in IEEE 802.11e, a station may transfer several packets without contending for the channel in between. Similarly, IP packet aggregation combines several IP packets together and sends them as one MAC Service Data Unit. Originally, both schemes have been developed for single-hop networks only. Thus the impact on WMNs is unclear if the packets need to contend over multiple hops. In this paper, we use measurements from a 9-node WMN testbed to compare TXOPs and IP packet aggregation for VoIP in terms of fairness, network capacity and quality of user experience. We show that for low networks loads, both TXOPs and IP packet aggregation increase the VoIP quality compared to IEEE 802.11 DCF. However, in highly loaded networks IP packet aggregation outperforms the other schemes.
1 Introduction

Wireless Mesh Networks (WMNs) are a promising technology for providing cost-efficient wireless Internet access to e.g. rural or urban areas. In a WMN, mesh routers relay traffic on behalf of clients or other mesh routers and thus form a wireless multi-hop network. Most WMNs are based IEEE 802.11 standards, e.g. 802.11s is standardising mesh mode operation. With the ever increasing transmission speed of 802.11-based technology due to the introduction of MIMO, OFDM, etc. transmission times for user payloads decrease rapidly. However, the performance of WMNs based on 802.11 can still be low, as the time spent on overhead (such as backoff, MAC and PHY layer headers) dominates for small packets. As an example, the transmission time of a 100 byte packet sent at 54 Mbit/s consists to 95% of overhead created by the IEEE 802.11 MAC layer. This problem becomes more severe as the data rate increases, because most of the headers are transmitted at a lower data rate. Also, this effect is more prominent for short frames, such as those typically used for VoIP. Furthermore, as John et al. [1] report, 50% of the packets on the Internet are smaller than 700 bytes. Since VoIP is an important service to be considered for WMNc operators, it is important to transmit small packets in an efficient way.

One possibility to increase transmission efficiency is to aggregate multiple smaller frames together into a larger one for transmission in one burst. This approach has multiple benefits because it reduces the PHY and MAC header overhead. Also, it reduces the total number of transmissions, which reduces contention and collision probabilities. This is especially important in a multi-hop setting under presence of hidden nodes as collisions lead to low throughput.

Burst transmission schemes have been investigated at different layers. MAC layer frame aggregation is a key mechanism to achieve higher throughput in IEEE 802.11n. Using the concept of Transmission Opportunity Limits (TX-OPLimit) in IEEE 802.11e allows a sender to send multiple packets once it has gained access to the medium. While those schemes work very efficient, a drawback is that they require a specific MAC layer. On the other hand, there have been several packet aggregation schemes developed that work with any 802.11-based MAC layer. In such approaches, a layer 2.5 module aggregates multiple IP packets together and forms an aggregated IP packet which is then transmitted at the MAC layer.

To have enough packets to aggregate, it is possible to delay packets to achieve a higher aggregation ratio. Such delay might seem counterproductive but can increase aggregation ratio, especially in low traffic scenarios. The collision probability will thus be further reduced, leading to lower a overall end-to-end delay as the average backoff delay will decrease. In a single-hop
2. Background

2.1 IEEE 802.11, RTS/CTS and Transmission Opportunities

There are two main operations defined for 802.11 distributed coordination function (DCF) MAC (see Figure 1). In the basic mode, a station transmits after a backoff. If the transmission is successful, the receiver sends back an ACK. If the frame (or the ACK) has not been received correctly, the sender has a timeout and retries after a backoff procedure. In the other mode, each transmission starts with a request-to-send (RTS) and clear-to-send (CTS) handshake to virtually reserve the medium. Such control packets can improve the performance in the single-hop case as the data packets are usually larger compared to RTS-CTS control packets and more affected by collisions. However, RTS-CTS packets are transmitted without any further protection and still may collide. Especially in the multi-hop case under the presence of hidden terminals, frequent collisions among RTS or CTS packets may severely degrade the performance. Also, the usage of RTS-CTS packets should be avoided for small packets due to the increased overhead.

The concept of transmission opportunities (TXOPs) in 802.11e [2] can be used to increase performance and enable Quality of Service provisioning. Once a sender has successfully contended for the medium, it can send several frames
IEEE 802.11 DCF

IEEE 802.11 DCF with RTS/CTS

IEEE 802.11e with TXOPs

IEEE 802.11 with IP Packet Aggregation

Figure 1: Burst packet transmission schemes for IEEE 802.11

separated by SIFS without contending for the medium in-between (see Figure 1). A TXOP is defined by its starting time and duration during which a station may transfer data of a particular traffic class. TXOPs can be either obtained via contention-based medium access (EDCA-TXOPs) or via controlled medium access (HCCA-TXOP or polled TXOP). The maximum duration of an EDCA-TXOP is limited by the parameter \( TXOPlimit \), which is distributed periodically through beacon messages. The \( TXOPlimit \) allows controlling the maximum time a station can allocate the medium for the delivery of MAC Service Data Units. As different service classes can define different \( TXOPlimits \), this mechanism enables an effective control of the throughput. IEEE 802.11e allows the use of block-ACKs which enables the receiver to acknowledge the successful reception of multiple frames using a single ACK packet.
2.2 IEEE 802.11 A-MSDU/A-MPDU

IEEE 802.11n [3] introduces MAC frame aggregation, where the sender either aggregates MAC Protocol Data Units (A-MPDU) or MAC Service Data Units (A-MSDU). In the A-MSDU mode, the MAC layer aggregates multiple packets from the upper layer by adding a single MAC header and check-sum. In contrast, the A-MPDU mode concatenates multiple 802.11 MAC frames each having its own MAC header and check-sum. By introducing a MAC delimiter, a receiver is able to separate each subframe, even if some of the sub-frames are corrupted. It also supports a block ACK scheme which allows the sender to retransmit only erroneous subframes. This can improve performance for channels having high bit error rates. The standard does not specify when packets should be aggregated, but normally this is done when there is more than one frame available in the sender queue. Hence, under low load, most packets will be sent unaggregated. Skordoulis et al. [3] show that frame aggregation in 802.11n can lead to performance improvements in single hop cases, if both modes are combined effectively. Kim [4] evaluates the performance of an early version of 802.11n frame aggregation as a function of payload size and data rate also in the single hop case.

2.3 IP Packet Aggregation

With IP packet aggregation packets destined for the same next-hop are concatenated before passing them to the MAC layer. An extra IP-header is added, which enables the next hop to de-aggregate the packet (see Figure 1). This mechanism is transparent to the MAC layer and thus no partial MAC-layer retransmission of erroneous segments is possible. While in theory limiting the maximum burst length to a value smaller than the MTU on weak links can reduce the packet error rate, this is not very applicable in practice. The transition region from a good link (that allows to fill the whole MTU) to a bad link (which requires bursts smaller than the MTU) is about 1-2 dB in SNR [5]. Due to the coarse quantization of SNR measurements on current cards and due to small scale fading, tuning the maximum burst size not very effective. Instead, the rate adaptation scheme should select a PHY rate that supports large frames.

Since IP packet aggregation is decoupled from the MAC layer, it cannot utilize the inherent delay for access the medium in the MAC. Artificially delaying packets by the right amount of time is thus crucial. Kyungtae and Ganguly [6] propose to let ingress mesh router probe the path to the destination to determine the end-to-end latency. The aggregation delay is set so that end-to-end latency plus the buffer delay does not exceed a pre-configured threshold. Intermediate nodes are not allowed to artificially delay packets further, but can...
aggregate additional packets whenever available. Riggio et al. [7] use a combination of probe messages, channel monitoring and an analytical model to derive an optimum packet size for a given network condition. Packets are delayed to create packet bursts of the optimum packet size.

3 Performance Evaluation

3.1 Experimental Setup

We compared the performance of multiple burst transmission schemes in the KAUMesh testbed, which consists of 20 Cambria GW2358-4 based mesh routers attached to the ceiling of the engineering building of Karlstad University. The nodes are equipped with Atheros 5212-based IEEE 802.11a/b/g wireless cards. A wired Ethernet card was used for time synchronization and to transfer traffic log-files. The mesh nodes run Linux 2.6.22 and MadWIFI 0.9.4, which we modified to support IEEE 802.11e in ad-hoc mode. In order to circumvent CPU bottlenecks and unwanted effects of the rate adaptation scheme, we disabled Auto-Rate and fixed the PHY data rate to 6 Mbit/s. The cards are operated in the 5 GHz frequency range to avoid interference from the campus WLAN. A subset of 9 nodes was used for our evaluation topology (Figure 2).

![Figure 2: Evaluation topology in the KAUMesh testbed](image-url)
We have implemented IP packet aggregation as a module for the Linux traffic control subsystem. The module contains a virtual FIFO-queue for each neighbor. When an IP-packet is sent from the user space or forwarded, it is marked with an expiration timestamp and enqueued in a virtual queue. After a packet is enqueued, the network card requests packets from the operating system or a timer expires, the aggregation module selects a virtual queue and concatenates all packets up to a size of MTU. Virtual queues are only dequeued, if packets have surpassed their expiration times (“aggregation delay”) or enough packets are available to fill up the whole MTU. An extra IP header indicating an aggregated packet is prepended and the aggregation packet is sent. A new timer is set to trigger a dequeue of the virtual queue when the next packet expires. The aggregation delay is configured statically. On the receiving node, aggregated incoming packets are identified by the extra IP header, de-aggregated in a netfilter-module and inserted into the normal Linux IP-stack. As this scheme is implemented in the OS-kernel, it imposes additional computations (e.g. for IP checksums) and memory accesses when creating the aggregation packet.

For each IEEE 802.11e access category, the network card (in our case based on the Atheros 5212 chipset) has a hardware FIFO queue, in which MAC frames are stored before transmission. As soon as the station has successfully contended for the medium, it can transmit frames of one access category that are available in the MAC frame buffer for a maximum time of TXOPlimit. To the best of our knowledge, support for block-ACKs cannot be configured with the Atheros 5212 chipset. As our WLAN NICs do not support the IEEE 802.11n standard, we could not evaluate the A-MSDU scheme.

3.2 Single-Hop Performance

We compare the IEEE 802.11 DCF, IEEE 802.11e with TXOPlimits of 1, 2, 3 and 8 ms and IP packet aggregation with aggregation delays of 1, 2, 3 and 8 ms in a single-hop scenario. We transmitted parallel UDP flows (200 bytes payload) at rates of 300 up to 650 packets/s (in steps of 25 packets) from nodes 7, 10, 13, 22 and 23 to node 21, using mgen [8]. Each test was executed for 60 seconds and repeated 5 times.

3.2.1 Maximum Achievable Rate

In Figure 3 the average end-to-end packet loss (top) and delay (bottom) (error bars are the standard deviation of individual test runs) show the well known behavior [9] of the IEEE 802.11 DCF. For lightly loaded networks (e.g. load \( \leq 3.0 \text{ Mbit/s} \)), the packet loss ratio and delay are low. In the transition from
a non-saturated to a saturated network, the packet loss ratio and delay rise quickly. While the network throughput in saturated IEEE 802.11 networks is at its or close to its peak, delay and packet loss are usually unacceptable for VoIP, which mandates a delay below 150 ms at a loss below 3% [6]. Thus, the optimum operation point for a network is just before the saturation. For
3. Performance Evaluation

3.1 Performance Evaluation

Using TXOPs, the number of channel access attempts is reduced, less time is spent in backoff-phases and fewer collisions occur. IP packet aggregation in addition reduces the number of ACKs and inter-frame waiting times. The efficiency depends on the number of packets sent within one burst. For IEEE 802.11e, we measured the average length of a burst by capturing all traffic with a wireless NIC in RF-monitor mode. This allows to determine when a packet arrived at the network card using the MAC-timestamp field in the Radiotap header. Measurements show that transferring one packet requires around 440 µs (including all headers and the ACK, but not the channel access). If difference in arrival times of two subsequent packets is smaller than 460 µs (440 µs transfer + 20 µs error margin), we conclude that both were sent within the same TXOP. For the IP packet aggregation we obtained the average burst size directly from the aggregation module statistics. Using the method described in [10] we also measured channel busy fraction (fraction of time the channel is sensed busy due to transmission or collisions). This is a good indicator for network congestion but also can be used to measure transmission efficiency.

From Figure 4 we observe that for a low offered load (3.2 Mbit/s) IEEE 802.11e with TXOPs hardly sends more than one packet within one burst. At 3.2 Mbit/s data injection rate the channel busy fraction is 74% (TXOP case), which indicates that collisions are rare and MAC layer queues are short. Therefore, almost no packets are available in the queue for sending within one TXOP. In contrast, IP packet aggregation artificially delays packets and thereby can on average send more than two packets at once. The measured channel busy fraction here is only 60% for an aggregation delay of 8 ms (67% for 1 ms), which shows the higher efficiency of the IP packet aggregation leading also to lower MAC layer utilization. Higher aggregation delay leads to larger burst length as more packets are available to be aggregated.

For higher rates (4.1 Mbit/s) the channel busy fraction increased to 89% (TXOP case) and we observed a considerable amount of collisions. As a consequence, the queue builds up in the MAC hardware buffers and therefore several
Figure 4: Average burst length

Queuing in the MAC layer only has a minor effect on the burst size of IP packet aggregation. The aggregation module cannot utilize packets waiting in the MAC layer queue to create longer bursts. Only if the MAC layer queue is full (maximum length 50 by default), packets queue up in the aggregation module. Otherwise only artificially delayed packets are available for aggregation. Thus the average burst size is lower for IP packet aggregation. Due to the reduction in inter-frame waiting times and ACKs, aggregation is still more efficient than TXOPs. Reducing the length of the MAC layer queue could increase the aggregation burst length. However, a short MAC layer queue requires very fast packet processing in the higher layers, since the MAC layer needs to have packets available as soon as the medium becomes idle.

3.3 Multi-Hop Performance

Next, we compare IEEE 802.11, IP packet aggregation and IEEE 802.11e with TXOPlimit=8 ms (best performance under high load in single-hop) in a multi-
3. Performance Evaluation

3.3.1 Traffic Generation and Quality Evaluation

We assume that one VoIP call consists of two G.711 audio flows: one from the mesh gateway to the mesh router and one in the reverse direction. We emulate a flow using a stream of 200 byte UDP datagrams with a constant arrival rate of 50 packets per second. We evaluated 50 different scenarios with 8 concurrent calls and 50 different scenarios with 12 concurrent calls. We created one scenario by randomly selecting one node as gateway. Among the remaining mesh nodes, we then randomly chose sources/destinations (one node can be source/destination for several calls), which communicate with the gateway node. Since our evaluation mainly focuses on MAC layer issues, we used static routes.

For each scenario, end-to-end delay, packet loss ratio and jitter of each flow was measured for 60 seconds. We emulated a fixed play-out buffer, which drops packets if their jitter according to [11] is greater than 30 ms. We estimated the perceived user experience of the VoIP call by (see also [12]) first calculating the R-factor (considering the impairment by packet loss including drops by the play-out buffer and the delay) and then converting it to the Mean Opinion Score (MOS), as described by ITU-T E-model using eq. b-4 of [13]. The MOS describes the average user satisfaction, where 5 is “Excellent”, 4 is “Good”, 3 is “Fair”, 2 “Poor” and 1 is “Bad”.

3.3.2 Average Quality

Figure 5 depicts the cumulative distribution of the MOS for VoIP flows with 8 (top) and 12 (bottom) concurrent calls (combined over all scenarios and flows). For low loaded networks (8 calls), the standard IEEE 802.11 MAC layer provides good quality (MOS ≥ 4) to 92% of the flows. The remaining 8% is constituted mainly by flows that need to be relayed over 3 or 4 hops. Using TXOPs or IP packet aggregation reduces the overall network load and gives good quality to all flows. For highly loaded networks (12 calls), none of the compared modes provides good quality to all flows. The best choice here is IP packet aggregation, where approx. 73% of the flows have a MOS greater than 4. Using TXOPs, the number of flows having MOS smaller than 4 increased to 46%. Interestingly, enabling RTS/CTS never improved performance.

A larger hop-count creates more opportunities for packet loss and increases end-to-end delay, which negatively impacts VoIP quality. Figure 6 shows this
relation where we group the results according to the hop-counts that flows traversed. The large deviations for flows of a given hop count can be explained as follows: in some scenarios there are only a few flows with high hop counts (2 or 3 hops). In this case the resulting network load is moderate and thus even the flows traversing more hops have good quality. In other scenarios however,
many flows have high hop counts. The resulting network load is high and the quality is poor. Using IEEE 802.11e with a TXOP limit of 8 ms the average burst length was 1.52 packets. In contrast, with IP packet aggregation (delay=8 ms) the average burst length was 1.96. As IP packet aggregation delays packets artificially, it can send more packets at once. In particular in multi-hop networks, where traffic is forwarded and thus not always back-logged, artificially delaying packets may lead to better performance, especially in higher loaded networks.

The general trend is clear: The average MOS decreases when number of hops increases. RTS/CTS does not improve the performance in average. Due to the higher overhead of RTS/CTS the medium is saturated earlier and hence the VoIP is degraded. The potential benefit of using RTS/CTS by reserving the medium so that fewer collisions occur is counteracted by the lower efficiency and other problems created by RTS/CTS (such as increased number of exposed nodes when RTS/CTS collide). Aggregation outperforms the use of TXOPs, which again is superior to the standard mode. Therefore, the key to improve performance for VoIP under high load is to enable an efficient transmission mode (e.g. by using packet aggregation).
3.3.3 Fairness

To study the impact of RTS/CTS on fairness, we calculated Jain’s fairness index [14] for each scenario, where a value of 1 implies perfect fairness (all flows have same MOS). Figure 7 displays the average over all scenarios. If the network load is low (8 concurrent calls) and thus no congestion occurs also flows traversing three or four hops receive good quality, which leads to a high fairness index. Compared to the standard IEEE 802.11e, the use of TXOPs or IP packet aggregation reduces overhead and consequently network load. Therefore fairness is increased by those burst transmission schemes. Even with 12 concurrent calls the overhead reduction due to burst packet transmission improves fairness.

4 Conclusions

We have evaluated the VoIP performance of IP packet aggregation and IEEE 802.11e TXOPs in a wireless mesh network. Compared to the standard IEEE 802.11 DCF, both schemes significantly increase the average VoIP quality and fairness. RTS/CTS does not contribute to a better average quality or more fairness. In multi-hop scenarios artificially delaying packets can be beneficial, since it creates longer bursts and enhances efficiency. As future work we plan to study the frame aggregation of IEEE 802.11n in a multi-hop setting and the
adaptive control of the aggregation delay for IP packet aggregation to cope with different traffic load.

References


FUZPAG: A Fuzzy-Controlled Packet Aggregation Scheme for Wireless Mesh Networks

Reprinted from

Proceedings of the 7th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD’10), Yantai, China, August 2010
FUZPAG: A Fuzzy-Controlled Packet Aggregation Scheme for Wireless Mesh Networks

Peter Dely, Andreas J. Kassler, Nico Bayer, Hans-Joachim Einsiedler and Dmitry Sivchenko

{peter.dely | andreas.kassler}@kau.se, {nico.bayer | hans.einsiedler | dimitry.sivchenko}@telekom.de

Abstract

Wireless Mesh Networks (WMNs) are wireless multi-hop backhaul networks in which mesh routers relay traffic on behalf of clients or other routers. Due to large MAC layer overhead, applications such as Voice over IP, which send many small packets, show poor performance in WMNs. Packet aggregation increases the capacity of IEEE 802.11-based WMNs by aggregating small packets into larger ones and thereby reducing overhead. In order to have enough packets to aggregate, packets need to be delayed and buffered. Current aggregation mechanisms use fixed buffer delays or do not take into account the delay characteristics of the saturated IEEE 802.11 MAC layer. In this work, we present FUZPAG, a novel packet aggregation architecture for IEEE 802.11-based wireless mesh networks. It uses fuzzy control to determine the optimum aggregation buffer delay under the current channel utilization. By cooperation among neighboring nodes FUZPAG distributes the buffer delay in a fair way. We implemented and evaluated the system in a wireless mesh testbed. For different network topologies we show that FUZPAG outperforms standard aggregation in terms of end-to-end latency under a wide range of traffic patterns.

1 Introduction

Wireless Mesh Networks (WMNs) are considered to be a promising technology for cost-efficient proliferation of wireless Internet access in both sparse rural areas as well as in densely populated urban areas. In a WMN, mesh routers relay traffic on behalf of clients or other routers and by this form a wireless
multi-hop network. Most WMNs are based on IEEE 802.11 commodity hardware. However, the performance of such WMNs can be low. One reason is the inefficient handling of small packets by the IEEE 802.11 MAC layer. The transmission time of a 100 byte packet sent at 54 Mbit/s consists of 95% of overhead created by the IEEE 802.11 MAC layer. Furthermore, as reported by [1] about 50% of the packets on the Internet are smaller than 700 bytes. For WMNs it is consequently important to transmit small packets in an efficient way. One possibility to increase the efficiency is packet aggregation.

Consider the example in Figure 1, in which three packets are transmitted towards the common next hop without and with aggregation. A normal IEEE 802.11 MAC layer data transmission consists of backoff, DIFS, the data transmission, SIFS and ACK. Using packet aggregation, the packets are concatenated and prepended with an aggregation header that informs the receiver how to deaggregate the packets. Instead of three DIFS, SIFS, ACKs and backoff periods, only one is thus needed, significantly reducing overhead. To have enough packets to aggregate, they need to be buffered first which adds extra delay. This can lead to quality degradation in latency-sensitive applications such as Voice over IP (VoIP) or video conferencing. Thus, it is important to find the best aggregation delay. If this delay is too low, the reduction in overhead is low and the total system throughput is low as fewer aggregation opportunities exist. If the buffer delay is too large, quality degradations in delay-sensitive applications are inevitable.

Existing solutions tackle the problem of finding the buffer delay in different ways. Recent IEEE 802.11-standards, such as IEEE 802.11e [2] or IEEE 802.11n [3] provide static aggregation mechanisms such as block-ACKs or MAC-frame aggregation. In IEEE 802.11e aggregation is only performed...
if there is more than one frame available in the sender queue, without artificially delaying frames. [3] shows that frame aggregation in 802.11n can lead to great performance improvements, however leaves unanswered how to set the aggregation parameters.

In [4] the ingress mesh router probes the path to the destination to determine the end-to-end latency. The aggregation delay is set so that end-to-end latency plus the buffer delay does not exceed a pre-configured threshold. Intermediate nodes are not allowed to artificially delay packets further, but can aggregate additional packets whenever available. Since [4] does not take into account the actual network load, it delays packets even if not necessary.

[5] uses the expected transmission count (ETX) to estimate the packet loss probability due to collisions and due to bit errors. Furthermore [5] utilizes the processing intensive RFMON mode of the wireless card to estimate the channel load. For a given ETX and channel load, the frame size which maximizes the saturation throughput is derived analytically.

The key problem for providing an efficient aggregation mechanism is to find the best aggregation delay. As the main contribution of this paper we present a novel mechanism to derive this value based on a fuzzy control system. As improvement to existing solutions, it estimates the channel load from the clear channel assessment (CCA) data of IEEE 802.11 and through cooperation distributes the buffer delay among nodes in a fair way. This allows both low aggregation delay under low load and high efficiency. The rest of this paper is outlined as follows: In Section 2 we describe the system model. In Section 3 we derive a control strategy for the buffer delay. Section 4 presents the implementation and evaluates the performance of the proposal. Finally, Section 5 concludes the paper.

2 System Description

The aim of the aggregation mechanism is to increase network efficiency and scalability by tuning the network to a state, where it provides a high throughput while keeping the latency low. We use Malone's model [6] of the IEEE 802.11 distributed coordination function for non-saturated networks to study throughput and MAC layer service delay. The system throughput is defined as the ratio of time spent on payload transmissions and time spent in an idle channel, by successful transmissions or collisions. The MAC layer service delay is the time span from when the packet arrives as head-of-line at the MAC layer and is successfully sent and is estimated according to [7]. Furthermore, we incorporate a new metric, the channel busy fraction (CBF). The CBF is the ratio of the periods in which the wireless channel is sensed busy and all chan-
nel states (busy or idle). The CBF can be measured directly by the network card [8].

2.1 Performance Analysis

Using the model above we study the system throughput, the channel busy fraction and the MAC layer service delay. Figure 2a depicts normalized system throughput for 4, 8 and 12 sender with IEEE 802.11a and 6 Mbit/s PHY rate. For packet sizes of 200, 800 and 2000 bytes the throughput increases almost linearly up to 3, 4.5 and 5 Mbit/s total offered load. If more traffic is inserted into the network, more and more collisions occur and the throughput grows slower or even decreases. For larger networks (12 nodes) the saturation throughput is smaller than the maximum throughput. Due to the lower overhead and fewer collision opportunities, larger packets (2000 bytes) allow a higher throughput (approx. 0.85) than smaller packets (approx. 0.6 for 200 bytes).

In Figure 2b we consider the MAC layer service delay for different traffic injection rates. For low traffic rates, the collision probability of two packets is low. Consequently, the MAC layer service delay is low as well. When the network goes from a non-saturated into a saturated state, the collision probability and delay increase sharply. For example, for 2000 byte packets, the MAC layer service delay starts to increase at about 5.1 Mbit/s.

Figure 2c shows the channel busy fraction for increasing traffic. Due to higher overhead, the channel busy fraction increases faster with small packets than with large ones. For example, at 3 Mbit/s injection rate, 200 byte packets cause a more than 80% channel busy fraction, while 2000 byte packets cause less than 60%.

Finally, in Figure 2d the MAC layer service delay is displayed as a function of the channel busy fraction. Almost independent of packet size or network size, up to 90% channel busy fraction the MAC layer service delay is low and then increases sharply.

2.2 Impact of Packet Aggregation

Figure 2d shows that channel busy fractions greater than 90% will cause high MAC layer service delay. Furthermore, Figure 2c shows that for the same traffic injection rate different packet sizes cause different MAC layer service delays. Due to the different channel utilization, if one sends larger packets instead of smaller ones (but with the same aggregate injection rate), one can reduce the channel busy fraction and the MAC layer service delay. Packet ag-
2. System Description

Figure 2: Analysis of IEEE 802.11 DCF under non-saturation

Aggregation exploits this relation by sending larger packets and reducing thereby the MAC layer service delay. By artificially delaying and then aggregating packets, it is possible to reduce the CBF, the MAC layer service delay and thus reducing the end-to-end latency. Since the packet arrival pattern and packet size distribution is not known in general, one cannot know in beforehand how much buffer delay is necessary to obtain a desired aggregation rate. However, a control system can measure the CBF and infer the network load. If the load is too high, the buffer delay is increased, which results in higher aggregation rates and lowers the network load. If the load is low, no aggregation is needed and the buffer delay should be low accordingly.
3 Fuzzy Controlled Packet Aggregation

The structure of the problem (making decisions about the buffer delay using vague and fuzzy information) motivates the use of fuzzy logic control. We transform the knowledge from the previous section into a fuzzy control system, which controls $delay_{MAX}$, the time a packet is artificially delayed at maximum. As soon as enough packets are available (MTU size) or $delay_{MAX}$ expires, all packets are aggregated and sent as one MAC layer transmission.

3.1 Input Variables

$busy\ fraction$  We use the channel busy fraction as indicator for the system throughput and MAC layer service delay of the network. In Figures 2c and 2d we observe three operating regions for the $busy\ fraction$: low, medium and high. With a high $busy\ fraction$ the MAC layer service delay is high. With a medium $busy\ fraction$ the throughput is high and the delay is low. With a low $busy\ fraction$ throughput and delay are low. Figure 3a depicts the membership functions.

$\Delta busy\ fraction$  denotes the change in $busy\ fraction$ between current and previous execution of the fuzzy controller. The universe of discourse ranges from -100% to 100% and is split into a negative, a neutral and a positive region (Figure 3b). Values outside this range are ped to the border of the universe by the input pre-processing.

$ratio\ difference$  represents fairness. Nodes that insert more traffic than average into the network should be punished by using a higher $delay_{MAX}$. A node estimates the busyfraction it causes by the packets ($own\ busy\ fraction$) it sends. It then calculates $busyratio_{OWN} = \frac{own\ busy\ fraction}{delay_{MAX}}$. Furthermore, it computes the average busyratio of its two-hop neighbors $busyratio_{NB}$. Each node computes the difference in his own busyratio and the average neighbors’ ratio as $ratio\ difference = (1 - \frac{busyratio_{OWN}}{busyratio_{NB}})$.

3.2 Output

$\Delta delay_{MAX}$ is the requested change of $delay_{MAX}$. By altering the aggregation delay, the number of aggregated packets, the overhead and channel busy fraction change. The output is divided into fuzzy sets for large and small positive and negative changes, as well as one fuzzy set for changes around zero. The crisp output value of $\Delta delay_{MAX}$ is calculated using center of gravity defuzzification [9].
3.3 Fuzzy Rules

As outlined in Section 2.2 increasing $delay_{MAX}$ reduces $busy\ fraction$, decreasing $delay_{MAX}$ increases $busy\ fraction$. At the system setpoint ($busy\ fraction$ is medium, centered at 0.7) the throughput is high and the MAC layer service delay is low.

The dynamic system behavior can be characterized by nine zones, as displayed in Figure 4. To achieve convergence, a node considers the following rules: If $busy\ fraction$ is low and not changing towards medium (zones 7/8), a node decreases $delay_{MAX}$. If $busy\ fraction$ is high and not changing towards medium (zones 3/4), a node increases $\Delta delay_{MAX}$. If $busy\ fraction$ is medium but increasing or decreasing (zones 2/6), a node decreases or increases $delay_{MAX}$ accordingly. Only if $busy\ fraction$ is changing towards medium (zones 1/5) or $busy\ fraction$ is already medium and not changing (zone 9), $delay_{MAX}$ remains unchanged. Those rules only apply if all neighboring nodes aggregate similarly much ($ratiodifference$ is neutral). If $ratiodifference$ is negative, i.e. the node aggregates little compared to its neighbors, $\Delta delay_{MAX}$ is increased a little despite...
of the busy fraction or $\Delta$busy fraction. On the opposite, if a node aggregates more than the average, it reduces $\Delta$delay$_{MAX}$ a bit.

![Figure 4: Dynamic system behavior](image)

Table 1 lists the complete rule-set. It consists of 27 rules, one for each combination of the three input variables and the associated three fuzzy sets. The rules have the form “IF $\text{ratio difference}$ IS $\text{input}_1$ AND $\text{busy fraction}$ IS $\text{input}_2$ AND $\Delta$busy fraction IS $\text{input}_3$ THEN $\Delta$delay$_{MAX}$ IS $\text{output}$”. For example, the output of “$\text{ratio difference}$ IS Neutral AND $\text{busy fraction}$ IS Medium AND $\Delta$busy fraction IS Neutral” is “Neutral”.

<table>
<thead>
<tr>
<th>Input: $\text{ratio difference}$</th>
<th>Input: $\text{busy fraction}$</th>
<th>Input: $\Delta$busy fraction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Negative</td>
<td>Low</td>
<td>Negative</td>
</tr>
<tr>
<td>Negative</td>
<td>Medium</td>
<td>Little Positive</td>
</tr>
<tr>
<td>Negative</td>
<td>High</td>
<td>Little Positive</td>
</tr>
<tr>
<td>Neutral</td>
<td>Low</td>
<td>Negative</td>
</tr>
<tr>
<td>Neutral</td>
<td>Medium</td>
<td>Negative</td>
</tr>
<tr>
<td>Neutral</td>
<td>High</td>
<td>Neutral</td>
</tr>
<tr>
<td>Positive</td>
<td>Low</td>
<td>Little Negative</td>
</tr>
<tr>
<td>Positive</td>
<td>Medium</td>
<td>Little Negative</td>
</tr>
<tr>
<td>Positive</td>
<td>High</td>
<td>Little Negative</td>
</tr>
</tbody>
</table>

Table 1: Fuzzy Rules
4. Implementation and Evaluation

4.1 Implementation

We have implemented FUZPAG system, which is shown in Figure 5, on Linux 2.6.22. It consists of user-space components (fuzzy controller, olsrd extension [10]) and kernel-space components (aggregation, deaggregation, medium sensing). The aggregation module is integrated into the Linux traffic control subsystem. It concatenates IP-packets with the same next-hop and prepends an extra IP header indicating it is an aggregated packet. On the receiving node, aggregated incoming packets are identified by the extra IP header, de-aggregated in an netfilter-module and inserted into the normal Linux IP-stack. delay\_MAX, the maximum aggregation delay, is set from the fuzzy controller via a netlink interface. The fuzzy controller is implemented using the Free Fuzzy Logic Library [11]. busy fraction is obtained as the fraction of the Atheros chipset registers 0x80f4 (PROFCNT\_RXCLR) and 0x80f8 (PROFCNT\_CYCLE). To cal-

Figure 5: FUZPAG architecture
culate ratio difference, information about the current delay_{MAX} and own busy fraction is exchanged with a new olsrd [10] FUZPAG-message type among 2-hop neighbors.

4.2 Evaluation Environment

We evaluated FUZPAG on the KAUMesh testbed, a testbed consisting of 20 Cambria GW2358-4 network computers installed in ceiling of the engineering building of Karlstad University (partly depicted in Figure 6). The nodes are equipped with Atheros 5212-based wireless cards. For the single-hop tests the PHY data rate was fixed to 6 Mbit/s. For the multi-hop tests the PHY rate was fixed to 18 Mbit/s. Time synchronization for the latency measurements nodes was done with wired Ethernet and NTP. The traffic was generated with mgen [12].

![Figure 6: Evaluation topology](image)

4.3 Controller Stability and Settling Time

The controller is executed every T_{exec} seconds. After the controller changes delay_{MAX} it waits for T_{exec} + rand[0, T_{exec}], to avoid a ping-pong effect and allow the controller to stabilize. If all nodes change delay_{MAX} at the same point of time, they might add too much delay at first, then too little and so on. Also, if T_{exec} is too small, the changes might not be reflected in the controller input variables such as the busy fraction yet. We sample the busy fraction every 250 ms. To avoid distortions by short traffic bursts, we filter busy fraction using
an exponential moving average ($\alpha = 0.1$). For sudden changes in the busyfraction it takes several seconds until filtered value converges towards the real busyfraction. Thus $T_{exec}$ needs to be large enough to allow busyfraction to settle. Since the controller needs to adapt to changes in the network traffic, a too large $T_{exec}$ would also render the controller useless.

In order to obtain a reasonable value for $T_{exec}$, we conducted a series of experiments. Nodes 7, 10, 13, 22 and 23 each send 480 UDP datagrams/s à 200 bytes to n21. For different values of $T_{exec}$ we checked whether the controller converges. The minimum value of $T_{exec}$ to have a reproducible and stable behavior was 5 seconds.

![Figure 7: Convergence of the controller](image)

For $T_{exec} = 5$s Figure 7 displays the settling behavior. The upper part shows busyfraction (measured by all nodes), the lower part the aggregation delay during the first 35 seconds of the experiment. First the nodes detect a high busyfraction and increase $\text{delay}_{\text{MAX}}$. With some time lag, the busyfraction decays. Finally, nodes tune $\text{delay}_{\text{MAX}}$ to around 2000 $\mu$s. Due to the fairness property of FUZPAG and all nodes inserting the same amount of traffic, all nodes have the same $\text{delay}_{\text{MAX}}$ (with a small error).
4.4 Single-Hop Scenario

Next, we studied the impact of the aggregation on end-to-end latency. Again, nodes 7, 10, 13, 22 and 23 sent 200 byte datagrams to node 21. We used a combined traffic injection rate of 2, 3.2, 4.4 and 5.6 Mbit/s. We ran each experiment for 60 seconds and repeated it 5 times. The fuzzy controller got in addition 40 seconds to settle before the measurement of the delay started. Figure 8 displays the average values and the standard deviation for no aggregation, aggregation with a fixed delay of 500 to 3000 µs and the fuzzy controlled aggregation.

With 2 Mbit/s input rate the network is lightly loaded and aggregation is not required. Here the latency is lower with no aggregation than with static aggregation values. The fuzzy controlled aggregation tunes the aggregation delay to 0 in this situation and therefore is as good as no aggregation. As other aggregation schemes use fixed delays, the average packet delay is 2-3 ms higher. With 3.2 Mbit/s injection rate and no aggregation the network is close to saturation. The delay is higher than before and varies a lot. All aggregation modes are about equally good. With 4.4 Mbit/s injection rate, 500-1500 µs aggregation delay is not enough. Too few packets are aggregated, the efficiency is low and thus the network is too highly utilized. The latency is between 220 and 300 ms\(^1\). With 5.6 Mbit/s injection rate, also 2000 µs aggregation delay is not

\(^1\)We cut Y-axis of the the diagram at 40 ms for scaling reasons. The cut-off values are all larger
sufficient. For all injection rates, the fuzzy controlled aggregation has lowest or close to lowest end-to-end latency.

4.5 Multi-Hop Scenario

The next scenario resembles a WMN connected to the Internet. Node 15 is the gateway and nodes 21, 7 and 11 communicate with the gateway. Nodes 7 and 21 send and receive their traffic to/from node 7 via 22 and 13, node 11 via node 13. The routes were set-up using the OLSR protocol and fixed during the experiments. In this experiment we evaluate the impact of aggregation on VoIP traffic. One VoIP call is emulated by a constant stream of 50 UDP datagrams per second (200 bytes length) from and to the source.

Figure 9 depicts the average end-to-end latency for different aggregation strategies. For low network loads (12 and 15 calls) it is better not to use aggregation. Static aggregation adds delay unnecessarily. When the network load gets higher, aggregation should be used. For 18 and 21 concurrent calls using fuzzy controlled aggregation reduces the end-to-end delay by 45 ms and 65 ms compared to not using aggregation. Fuzzy controlled aggregation outperforms static aggregation delay settings by about 5-10 ms. Since in this scenario the load is not uniformly distributed over the network, the nodes should use different aggregation delays. For example, using the fuzzy controlled aggregation

than 200 ms.
and 18 concurrent calls, the highly loaded node 13 uses an average aggregation delay of 955 µs, while the lighter loaded nodes 21 and 7 have 346 µs and 343 µs respectively.

5 Conclusions

In this paper we have demonstrated that, by using fuzzy control, it is possible to tune the aggregation buffer delay $\text{delay}_{\text{MAX}}$ so that extra delay is only added when necessary. FUZPAG outperforms a static configuration of $\text{delay}_{\text{MAX}}$ in a wide range of scenarios. Depending on the network topology and traffic, the reduction in end-to-end latency by using FUZPAG is up to 65 ms in multi-hop and several hundred milliseconds in single-hop scenarios. As future work we plan to dynamically tune the membership functions using a genetic model reference adaptive controller.

References


QoS-Aware Channel Scheduling for Multi-Radio/Multi-Channel Wireless Mesh Networks

Reprinted from

Proceedings of the Fourth ACM International Workshop on Wireless Network Testbeds, Experimental evaluation and Characterization (WiNTECH 09), Beijing, China, September 2010
QoS-Aware Channel Scheduling for Multi-Radio/Multi-Channel Wireless Mesh Networks

Marcel C. Castro, Peter Dely, Andreas J. Kassler and Nitin H. Vaidya

{marcel.cavalcanti | peter.dely | andreas.kassler}@kau.se, nhv@illinois.edu

Abstract

In non-static multi-radio/multi-channel wireless mesh networks architectures such as Net-X, mesh nodes need to switch channels in order to communicate with different neighbors. Present channel schedulers do not consider the requirements of real time traffic such as Voice over IP. Thus the resulting quality is low. We propose a novel channel scheduler for the Net-X platform that takes into account packet priorities. We evaluate the algorithm on the KAUMesh testbed. Our algorithm outperforms the standard round-robin scheduler both in terms of average delay and jitter.

1 Introduction

Wireless Mesh Networks (WMNs) are considered to be promising technology for cost-efficient proliferation of Internet access in both sparse rural areas as well as in densely populated urban areas. In a WMN mesh routers relay traffic on behalf of clients or other routers and by this form a wireless backbone. While in the first generation of mesh networks routers only used one radio and one channel to forward traffic, in newer WMNs multiple radios and channels are used simultaneously. This can increase the network capacity drastically, but also adds extra complexity.

For example, if a node uses more channels than it has cards available, then multi-channel operation imposes the need for channel switching. According to [1] the channel switching time on current hardware is between 200 µs and 20 ms, which causes a very high overhead for per-packet switched channels.
The channel-switching-time is composed of several phases: sending buffered frames in the hardware queue of the NIC, stopping interrupt service routines of the driver, tuning to the new frequency, re-starting the interrupt service routines and sensing the medium.

Some of those phases can be shortened significantly by clever hard- and software-design, others however are more challenging to optimize. For example, per-channel hardware buffers can store frames across channel switches and thus flushing the buffer prior to the switch is not needed. Also an optimized design of the RF-filter mask tuning improves the performance.

In the standard IEEE 802.11 DCF protocol collision avoidance relies to a large part on correctly detecting the medium status by the means of channel clear assessment and/or RTS/CTS. Those functions only work well if a node can sense the medium long enough to hear frame-preambles or RTS/CTS frames [2]. Thus, a node should not immediately transmit data after a channel switch. We believe that this is a fundamental shortcoming of the IEEE 802.11 MAC protocol in channel-switched systems, which needs more investigation. As a result, even good hard- and software design do not allow per-packet channel switching with an acceptable overhead when IEEE 802.11 DCF is applied.

Therefore, multi-radio multi-channel mesh network platforms such as Net-X [3] use per-channel queues, which are serviced for a longer time span. A scheduler then has to decide when and for how long a channel is serviced.

For delay sensitive traffic such as Voice over IP, the right scheduling strategy is crucial for providing good end-user quality. Important information such as traffic priority should be considered as input parameter to the scheduling approach. In this paper we address the issue of a QoS-aware channel scheduling based on traffic priorities. The key-contributions are:

- An analytical evaluation of the algorithm.
- An implementation of the algorithm in the Net-X platform and a performance evaluation.

The remainder of this paper is organized as follows: In Section 2 we introduce the multi-radio/multi-channel framework Net-X and discuss related work. The design and implementation of our proposed scheduler is presented in Section 3. Section 4 compares the performance of our scheduler to a base-line round-robin scheduler. Finally, Section 5 presents our conclusions and suggests future improvements.
2 Background and Related Work

2.1 Multi-Channel Mesh Networks

Multi-channel protocols and architectures are designed to exploit the available channels to enhance the overall throughput. According to [4], the classification of channel assignment protocols can be based on how frequently the channel assignments are performed, therefore the protocols and architectures for multi-radio/multi-channel networks can be classified as dynamic, semi-dynamic, static and hybrid. Net-X [3] is a hybrid approach, as it applies a semi-dynamic assignment to the fixed interface (used primarily for receiving data from neighbors) and a dynamic assignment to the switchable interfaces (used to transmit data to its neighbors). The semi-dynamic reassignments to the fixed interfaces are based on the current number of nodes using the same fixed channel. Therefore, if a node notices that the number of nodes using the same fixed channel as itself is larger, it can reassign its interface to a less used channel and inform its neighbors.

The channel used by the switchable interface may be changed at any time, without having to inform the neighbors. Thus, the switchable interface can be used to transmit to neighbors whose fixed interfaces may potentially be on different channels. Net-X protocol stack includes a channel management module that determines which channel to assign to each fixed interface, and when each switchable interface may switch its channel. By judicious use of channels, it is possible to efficiently utilize a large number of channels in the mesh, even though each node is equipped with only two interfaces. Since the interface channel-switching may incur a non-negligible delay, a queuing algorithm to buffer packets is deployed in Net-X, as well as a round robin scheduling policy to transmit buffered packets in order to reduce frequent switching.

As shown in [5], unnecessary delays are created in Net-X due to the hello packets being transmitted consecutively on all the channels. For example, an interface that services four channels must switch to three other channels and spend at least a minimal amount of time on each channel before returning to the loaded channel. In order to avoid such unnecessary delays [5] proposes to replace the round robin scheduler by the "delay sensitive" channel scheduler. The objective of the proposed scheduler is to service more often on the channels that have a higher average queue length. This is done in [5] by staggering the creation of the “hello” packets due to the overall benefit of servicing queues that have a longer average length. It is important to note that this solution does not make differentiation among service priorities, therefore if two channels are loaded (e.g. one with VoIP traffic and another with TCP traffic), the scheduler will serve them equally.
2.2 IEEE 802.11e EDCA

IEEE 802.11e is a standard for QoS over IEEE 802.11 based networks (see for example [6]). Among other extensions to the original IEEE 802.11 MAC layer, it includes the Enhanced Distributed Channel Access (EDCA). EDCA allows the prioritization of frames inside a node and among nodes. Inside a node, for each service class a queue is created, which holds packets of its service class. Queues are served by a virtual contention resolution mechanism similar to DCF. Furthermore, among nodes the using different inter-frame times can prioritize frames.

Initially, IEEE 802.11e was not designed for multi-channel/multi-radio networks. For example, IEEE 802.11e does not consider channel switching. Also, inside a node, each network card runs its own instance of the MAC protocol, not exploiting information possibly available from other network cards. However, IEEE 802.11e can be used as a complimentary design element in multi-channel networks. If, for example, several nodes compete for the same channel, IEEE 802.11e can still be useful to prioritize the medium access. IEEE 802.11e and the proposed scheduler are orthogonal approaches. In this paper we concentrate on the scheduler design and evaluation, but we consider combining both techniques in a future work.

3 QoS-Aware Channel Scheduler

3.1 Design Goals and Motivation

Existing schedulers do not take into account the priority of packets. Thus it can happen that packets with smaller delay budget are delayed unnecessarily long, because a channel with delay-insensitive packets is served before. For delay sensitive traffic such as VoIP this will reduce the perceived quality. The goals of our scheduler are thus to minimize the delay and waiting time for delay sensitive traffic while at the same time providing reasonable throughput for delay-insensitive traffic at reasonable switching cost.

3.2 Scheduling Algorithm

Our scheduling algorithm selects the next channel based on the priority of the current channel and the priority of all other channels, which have packets to send. We assign the priority to a channel according to the priority of packets which are queued to be sent for this channel. If packets with different priorities are queued, the highest priority among all packets is used. The currently used
3. QoS-Aware Channel Scheduler

channel might not have packets queued. In this case the priorities of packets since the last channel switch are taken into account. A packet’s priority is determined by its DiffServ Code Point [7]. While DiffServ allows the definition of multiple traffic classes, for simplicity we only consider a low and high priority traffic class. The sender or ingress-router marks delay-insensitive TCP-traffic with low priority, real-time traffic such as VoIP with high priority. The concept could be extended to multiple priorities, which are even dynamically assigned (for example based on available delay budget).

The scheduling algorithm is sketched in code listing 1 and is composed of two parts. The first part (lines 1-8) determines which priorities the current (getChannelPrio(Current Channel)) and the next channel have. For each priority, the algorithm maintains a counter served[prio] that represents the number of times that the priority was served. If the counter exceeds the per-priority configurable threshold \( T_{u,prio} \), a new lower target priority (targetprio) is selected and the current counter is reset to zero. If there are no channels of the target priority available, the algorithm looks for the next lower priority (getTargetPrio(prio)). Upon reaching the lowest possible priority, the algorithm starts over with the highest priority. In the second part (lines 9-12), among all channels of the new priority (getChannelsByPrio(targetprio)), the channel which has not been served longest is selected by calling the function getOldestChannel(CandidateSet).

When a channel is selected it is scheduled for a time of \( T_{min} \). Furthermore if after \( T_{min} \) there are still packets available to send, the channel gets an extra time of \( T_{defer,prio} \). After this time the channel scheduler is called again to select a new channel. In the rest of this paper we assume backlogged traffic, i.e. a channel is always scheduled for a service time \( S = T_{min} + T_{defer,prio} \); \( T_{min} \) and \( T_{defer,prio} \) are configurable. Switching from one channel to another requires \( T_s \) delay.

3.3 Analysis

In this section we analyze the scheduling performance when two priorities are used. For example, consider a node has to schedule two low priority channels \( (L_1 \) and \( L_2 \)) and two high priority channels \( (H_1 \) and \( H_2 \)). The channel hopping pattern with our algorithm will be \( H_1 H_2 L_1 H_1 H_2 L_2 \) and so forth (\( T_{u,l} = 1 \) and \( T_{u,h} = 2 \)). In contrast the round robin scheduler produces a pattern like \( H_1 H_2 L_1 H_2 L_2 H_1 H_2 L_1 L_2 \).

With regard to those two hopping patterns one can see that the waiting time for high priority traffic and the throughput for background traffic depends on the switch patterns and the values of \( T_{min} \) and \( T_{defer,prio} \). In the following
Algorithm 1: QoS-Aware Channel Scheduling Algorithm

sub-sections we will first derive equations for the channel waiting time and throughput. Then we will apply those models to an example traffic pattern.

The analysis is based on the following assumptions:

- backlogged traffic
- two different packet priorities
- every channel only serves packets of one priority
- arbitrary number of high and low priority channels
- a fixed data rate

3.3.1 Waiting Time

The waiting time is the maximum time that a channel needs to wait until it will be rescheduled again. The waiting time has a direct influence on the packet delay. For a simple case with \( m \) high priority (H) and \( n \) low priority (L) channels the waiting time \( WT(H) \) for a high priority channel is given in equation 1. It is the sum of the service-times of all low and high priority channels that need to be served before the initial channel is serviced again. Please note that the service times \( S(H) \) and \( S(L) \) can be different for the QoS-aware scheduler, since it allows to have per-priority \( T_{\text{def.\,prio}} \). However for the round-robin scheduler \( S(L) \) equals \( S(H) \).

\[
WT_{\text{QoS}}(H) = (T_{u,H} - 1) \ast S(H) + T_{u,L} \ast S(L) + (T_{u,L} + T_{u,H}) \ast T_s
\] (1)
Equation 2 calculates the waiting time for a round-robin scheduler.

$$WT_{RR}(H) = (m - 1) \times S(H) + n \times S(L) + (m + n) \times T_s$$  \hspace{1em} (2)

### 3.3.2 Throughput

Apart from the waiting time for the high priority traffic, the throughput of the background traffic is also important. We assume that the throughput for one channel is proportional to the amount of time scheduled for this channel. We are aware that the throughput is dependent on other factors such as the number of collisions and the transport protocol as well. Yet we believe that channel time is a sufficiently good measure of achievable throughput in our context. Through the use of channel diversity the number of collisions and neighbors competing for the same channel is reduced.

The cycle length of a channel switch pattern is the number of priority changes after which the same pattern repeats again. For the QoS-aware scheduler the cycle length is

$$CY_{QOS} = \text{lcm}\left( \frac{\text{lcm}(m, T_u)}{T_u, H}, \frac{\text{lcm}(n, T_u)}{T_u, L} \right)$$  \hspace{1em} (3)

Here, $\text{lcm}(m, n)$ denotes the least common multiple of $m$ and $n$. It calculates when a pattern is repeated. If $T_{u,H}$ or $T_{u,L}$ are 0, the cycle time is $m$ or $n$ respectively. Using this equation we express the ratio of the service time within a cycle of a specific class (H or L) and the total cycle length as

$$CT_{QOS}(L) = \frac{CY_{QOS} \times T_u \times S(L)}{CY_{QOS} \times T_u \times S(H) + CY_{QOS} \times T_u \times S(L)}$$  \hspace{1em} (4)

and

$$CT_{QOS}(H) = \frac{CY_{QOS} \times T_u \times S(H)}{CY_{QOS} \times T_u \times S(H) + CY_{QOS} \times T_u \times S(L)}$$  \hspace{1em} (5)

For the round-robin scheduler the amount is given by

$$CT_{RR}(L) = \frac{n \times S(L)}{m \times S(H) + n \times S(L) + T_s}$$  \hspace{1em} (6)

and

$$CT_{RR}(H) = \frac{m \times S(H)}{m \times S(H) + n \times S(L) + T_s}$$  \hspace{1em} (7)

### 3.3.3 Example

Using the equations from the previous section we compare the throughput and the waiting time of our QoS-aware scheduler and the round-robin scheduler.
The example makes use of some system and traffic pattern parameters which are used in the real system evaluation described later on in Section 4. The parameters and their values are listed in Table 1.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_s$</td>
<td>4 ms</td>
</tr>
<tr>
<td>$T_{min}$</td>
<td>15 ms</td>
</tr>
<tr>
<td>$T_{u,L}$</td>
<td>1</td>
</tr>
<tr>
<td>$T_{u,H}$</td>
<td>2</td>
</tr>
<tr>
<td>$T_{defer,L}$</td>
<td>10 ms</td>
</tr>
<tr>
<td>$T_{defer,H}$</td>
<td>10 ms for RR, 0 for QoS</td>
</tr>
<tr>
<td>m</td>
<td>2</td>
</tr>
<tr>
<td>n</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 1: Algorithm Parameters

$T_s$ represents the time, which the network card needs to tune to the new channel. It depends on the card chipset and the driver and is around 4 ms for our NIC and driver. As discussed in Section 1, this value is dependent on many factors and cannot be arbitrarily small. The minimal scheduling time in a channel, $T_{min}$, is also limited by the implementation factors (e.g. interrupts, timer accuracy and overhead). The standard Linux timers have an accuracy of about 2 ms [8]. Considering this and the 4 ms overhead involved in every channel switch, we choose a $T_{min}$ of 15 ms as a compromise between switching overhead and delay. $m$ and $n$ indicate the number of low- and high priority channels and are a direct result of the traffic pattern.

In contrast to the previous parameters, that are implementation-dependent, $T_{u,L}$, $T_{u,H}$, $T_{defer,L}$ and $T_{defer,H}$ are configurable parameters. We chose the values such that high priority traffic is served more often ($T_{u,H} > T_{u,L}$) and low priority traffic is served longer ($T_{defer,H} = 10 \text{ ms}$) each time. Those settings are specific for the given traffic pattern. By using these parameters, Table 2 shows that the waiting time $\text{WT}(H)$ for QoS-aware scheduler is around 29 ms lower than the one for the round-robin scheduler. However, the QoS-aware scheduler gives the low priority traffic a lower time share. Thus the throughput of the background traffic should be lower for the QoS-aware scheduler.

There is a trade-off between achieving small waiting time for high priority traffic and guaranteeing low priority traffic throughput. This trade-off can be controlled by $T_{defer,L}$. For the given configuration, the waiting time for high priority traffic equals $T_{defer,L} + 45$. Figure 1 depicts this trade-off and compares it to the default round-robin scheduler. It is important to note that we just vary the configuration parameters of the QoS scheduler (given by the solid line), and compare it to the waiting time for high priority traffic and low priority channel.
3. QoS-Aware Channel Scheduler

<table>
<thead>
<tr>
<th>Value</th>
<th>QoS Scheduler</th>
<th>Round Robin Scheduler</th>
</tr>
</thead>
<tbody>
<tr>
<td>CT(H)</td>
<td>44.8%</td>
<td>43.1%</td>
</tr>
<tr>
<td>CT(L)</td>
<td>37.3%</td>
<td>43.1%</td>
</tr>
<tr>
<td>Switching Overhead</td>
<td>17.9%</td>
<td>13.8%</td>
</tr>
<tr>
<td>WT(H)</td>
<td>52 ms</td>
<td>81 ms</td>
</tr>
</tbody>
</table>

Table 2: Comparison QoS and Round-Robin Scheduler

time share results of the round-robin scheduler presented in Table 2 (WT(H) and CT(L), given by the intersection of the dotted lines).

The graph can be divided into three regions. In the first region (waiting time $< 59$ ms), the background traffic has lower time share and the high priority traffic lower delay as compared to the round-robin scheduler. In the second region (waiting time $> 59$ ms and $< 81$ ms) the background traffic gets more time share and the high priority traffic has a lower waiting time. In the third region the QoS-aware scheduler gives the background traffic more channel time on cost of a higher waiting time for high priority traffic. To summarize, in the second region the QoS-aware scheduler is always better than the round-robin scheduler. In the first and third region, the operator can sacrifice either delay (of high priority traffic) or throughput (of low priority traffics).

For arbitrary traffic patterns, equations (1)-(7) allow to quantify the trade-off between throughput and waiting time similarly to Figure 1 and how it behaves by varying $T_{def, L}$. However not considered in this work, a natural extension to our algorithm is the selection of the parameters based on a higher-level objective, such as the ratio between gain of latency (smaller channel waiting time) versus loss of throughput compared to the standard RR-scheduler.

3.4 Implementation

We implemented our algorithm by extending the bonding driver from the Net-X platform. A hybrid multi-channel protocol is used in order to couple the destination address and the channel used to reach this destination [9]. Figure 2 shows an architectural view of the bonding driver. When packets arrive from the user space or the routing layer they are assigned a priority based on the Diff-Serv field in the IP-header. There is one queue for every channel. If the card is already tuned to the correct channel, packets are directly forwarded to the network card driver. Otherwise packets are placed in their respective queues. The scheduling algorithm is called by a timer when the service time for a channel ends and selects the next channel queue to service based on algorithm 1 or a round-robin scheme.
4 Performance Evaluation

4.1 Evaluation Environment

We evaluated our algorithm on the KAUMesh testbed [10], which is an indoor wireless mesh network with 20 nodes deployed at House 21 at Karlstad University. The nodes are based on the Cambria GW2358-4 platform, Linux 2.6.22, MadWifi 0.9.4 and Net-X. Each node is equipped with three Atheros-based wireless network interface cards. We aim to minimize wireless link variability factors, therefore the PHY-rate and the routes are fixed throughout the evaluation. Two interfaces are used for the mesh-backbone operating in IEEE 802.11a mode and an PHY-rate of 6 Mbit/s. The third interface can be used for client access in 802.11b/g mode. By using the IEEE 802.11a channels on the backbone, which operate in the 5 GHz-band, we avoid interfere with the campus WLAN network operated on the 2.4 GHz-band.

Figure 3 depicts the network topology used for the evaluation. It consists of five nodes each mounted in the ceiling of lab rooms or corridors in House 21. The topology is simple and controlled to avoid problems such as route breaks, re-routing and bad links, which will typically occur as the network grows. Node 7 runs an NTP-server. All other nodes synchronize their hardware clocks against the NTP-server every three seconds. By this we achieve a clock skew smaller than 1 ms. Please note that the clock synchronization is
4. Performance Evaluation

Figure 2: Bonding module

Locally generated or forwarded packets

Bonding Driver

Classifier

ch=1

ch=2

\ldots

ch=n

Select channel

Scheduler

Switch channel

Forward packets

Network Card 1 (fixed)

Network Card 2 (switchable)

not needed for the scheduling, but only for the timestamps used in the delay measurements.

In our experiments the VoIP traffic is generated by mgen [11] using 200 UDP datagrams per second (CBR) of 168 bytes between nodes 7 and 11 in both directions. This traffic pattern simulates four concurrent VoIP calls with the G.711 codec. At the same time, the TCP traffic is generated with the iperf tool [12] between nodes 10 and 21, and nodes 10 and 23, as shown in Figure 3. Therefore, for the given scenario, the number of high and low priority channels is equal to 2 (m = 2 and n = 2, respectively). Each experiment run lasted for 55 seconds.
4.2 Results

Latency, jitter and packet loss are key characteristics to be considered while investigating the performance of voice traffic over multi-radio multi-channel wireless mesh networks. In this section we present the behavior of these metrics while analyzing the scheduling algorithm. The results represent the average over five repetitions, while the error bars represent the respective standard deviations.

Understanding latency characteristics is crucial for delay sensitive applications such as VoIP. According to [13] to obtain a good voice quality the delay imposed by the network should stay below 150 ms, since VoIP does not tolerate excessive delays in the conversation. The results in Figure 4a show the average latency experienced by the two scheduling strategies studied: round robin scheduling (default) and the QoS-aware channel scheduling (QoS). As expected, we observe that the QoS-aware channel scheduler decreases the average latency of the VoIP packets when compared to the round robin scheduler. This is because the proposed scheduler guarantees that high priority channels carrying VoIP traffic are preferred to the lower priority channels carrying TCP flows, which consequently reduces the average TCP throughput as shown in Figure 4b.

Similarly, excessive jitter makes the service unusable by negatively impacting service quality. Since jitter is categorized as the change in latency from packet to packet, we plot in Figure 5 the latency experienced by the VoIP packets along the measurements. The X-axis represents the VoIP packets’ IDs of one UDP flow between nodes 7 and 11 and the Y-axis represents the packet

\[\text{Due to space constrains, we plot a range of 300 packet IDs.}\]
4. Performance Evaluation

Figure 4: Average latency and TCP throughput

delay in seconds. The graphs show the behavior of packets’ delay (Y-axis) along the packets generated (X-axis). The spikes represent the most delayed packets that are waiting on the node’s channel queue to be transmitted. In contrast, the least delayed packets are the packets not queued, as the node currently accessing the right channel. We see that by using the round robin scheme, some VoIP packets can achieve delay of 100 ms or more, in contrast with 60 ms while using the QoS-aware scheme. The network jitter in both schemes is a result of the delay that certain (most delayed) packets experience in the nodes’ queues while waiting for the next channel transmission opportunity. As expected, our scheme reduces VoIP latency, and therefore network jitter, as the VoIP channels have higher priority during the scheduling decision than the TCP channels.

When compared to the waiting time presented in Table 2, the measured waiting time values of 60 and 100 ms obtained are within a small error in accordance with the theoretical analysis from Section 3.3. The remaining difference is present since in the theoretical analysis the time for transmitting the packet over the air and to processing it at the receiver is not considered.

To better explain the previous results, we plot in Figure 6 the channel hopping pattern for the two schedulers studied. The channel hopping patterns are obtained through the analysis of the channel switching dynamics in node 10’s switchable interface. The X-axis illustrates the measurement time in milliseconds and the Y-axis illustrates the 802.11a channels used by the switchable interface. By considering the topology in Figure 3, we can point out that the channels 36 and 64 are the high priority channels (VoIP packets) and the channels 48 and 140 are the low priority channels (TCP packets) used. According to the results, by using the round robin scheme, the average waiting time for the high ($WT_{RR}(36)$ and $WT_{RR}(64)$) and the low ($WT_{RR}(48)$ and $WT_{RR}(140)$) priority channels were 79, 81, 70 and 68 ms, respectively. However, by using the QoS-aware channel scheme, the average waiting times for the high ($WT_{QoS}(36)$ and
Figure 5: Voice packet delay versus voice packet IDs

WT_QoS(64) and the low (WT_QoS(48) and WT_QoS(140)) priority channels were 56, 47, 128 and 101 ms, respectively. In Figure 6 we can also observe the influence of defer-time for the high ($T_{defer,H}$) and low ($T_{defer,L}$) priority channels presented in Table 1. For the round robin scheduler, the high and low priority channels make use of the defer-time and therefore the channel service time can be extended by 10 ms. For the QoS scheduler, the $T_{defer,H}$ is not used and therefore just the low priority channels experience a higher channel service time.
Applications and end-user devices are designed to tolerate a certain amount of jitter. This is achieved through a so-called jitter buffer, by buffering the data flow and designing processing algorithms to compensate for small changes in latency occurring from packet to packet. Depending on the application, the tolerable amount of jitter will vary. For example, a VoIP service should have a jitter buffer of approximately 80 ms or less [13]. By achieving a lower waiting time for high priority traffic using the QoS-aware scheduling, we also guarantee the reduction of the jitter buffer size required by the nodes. In order to verify that, we plot in Figure 7 the histogram and the cumulative distribution function (CDF) of the measured delay of VoIP packets for one test run and both schedulers. The X-axis represents the packet delays and the Y-axis represents the normalized number of packets mapped inside each interval. The Y-axis in
the right hand-side presents the CDF of the packet delays. From the CDF we note that for the round robin scheduler more than 40% of packets experience a delay larger than 50 ms. In contrast, for the QoS scheduler only 8% of the packets experience such delay. The histogram also shows that the majority of the packets experience a maximum delay of 70 ms and 100 ms for the QoS and default scheduler respectively.

![Figure 7: Histogram of voice packet delay](image)

Packet loss can occur due to several factors, such as medium congestion and high traffic load. The use of a jitter buffer also augments the amount of packet loss since packets with a delay difference greater than the selected jitter buffer size are also considered lost. Packet loss, then, can significantly reduce quality of service. We show in Figure 8 the relationship between the loss ratio of voice packets versus jitter buffer size for both schedulers. The packet loss ratio consists of packet loss introduced by the network and the jitter buffer. The packet loss introduced by the network in our measurements is 2.8±1.6% and 1.3±0.5% for the default and the QoS-aware scheduler respectively. We assume a simple, static jitter buffer that drops packets with a packet inter-arrival time difference greater than a certain threshold (shown on the X-axis). It is clear that the introduction of a greater jitter buffer size (e.g. 100 ms) may decrease the packet loss in the system at the cost of further delaying the voice packets. However we need to note that for the same amount of packet loss (e.g. 5%), the QoS-aware scheduler requires a smaller jitter buffer size (70 ms) as compared to the round robin scheduler (100 ms).
In this paper, we have presented a new QoS-aware channel scheduling technique for multi-radio/multi-channel wireless mesh networks. Through the use of traffic priority information carried in the packet header, we propose a new channel scheduler that gives priority to VoIP traffic and still guarantees reasonable throughput of the non-priority TCP traffic. This scheduler has been integrated to the Net-X platform. Through testbed measurements we showed a reduction in average end-to-end delay and network jitter of VoIP flows for the proposed QoS-aware channel scheduler.

As future work, we plan to investigate two issues. First, we plan to carry out an extended study where different scenarios with greater number of nodes and different traffic demands are analyzed. In our current performance evaluation, we make use of static traffic priorities among flows. Our second research idea is to evaluate the assignment of dynamic priorities.
References


