AG Vernetzte Systeme
Fachbereich Informatik
Technische Universität Kaiserslautern

Master thesis

Network sensing using WLAN

Kiran Mathews

February 13, 2017

Network sensing using WLAN

Master thesis

AG Vernetzte Systeme
Fachbereich Informatik
Technische Universität Kaiserslautern

Kiran Mathews

Tag der Ausgabe	:	05.09.2016
Tag der Abgabe	:	13.02.2017

Erstgutachter	:	Prof. Dr. Reinhard Gotzhein
Zweitgutachter	:	MSc. Christopher Kramer

I affirm that I have written this master's thesis on my own.
Kaiserslautern, February 13, 2017

(Kiran Mathews)

Abstract

In recent years, interest in wireless communication for industrial automation increases, but reliability is a key requirement in these applications. Reliability of wireless communication depends on factors such as medium access delay, frame loss, and latency. Traditionally, Time Division Multiple Access (TDMA) based medium access methods are used for deterministic access delay. But, TDMA based medium access is not a favorable method in a congested network with external devices operating in the same frequency domain. The option to use a reserved frequency domain for deterministic applications is limited and costly. Moreover, the emergence of IoT, VANETS, and other wireless based technologies will create extra traffic in near future. Apart from reliability, a cheap and hardware independent automation network benefits different industrial applications.

The idea is to introduce a statistically reliable, and cheap closed-loop control wireless automation system. A communication network on top of IEEE 802.11 (WiFi) provides a cheaper solution. IEEE 802.11 by default use Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) based medium access method. Medium access delay using CSMA/CA based method is non-deterministic and depends on the medium conditions, i.e. contention in the medium. In order to provide statistical reliability, a clear picture about the traffic on the medium is required. In this thesis, the possible solutions to sense the medium are investigated. Also proposed a "Token Bucket" based bandwidth distribution, and network sensing mechanism to distribute available bandwidth among the nodes and sense the local medium contention.

Zusammenfassung

In den letzten Jahren steigt das Interesse an der drahtlosen Kommunikation für die industrielle Automatisierung, aber die Zuverlässigkeit ist eine wichtige Anforderung in diesen Anwendungen. Die Zuverlässigkeit der drahtlosen Kommunikation hängt von Faktoren wie der mittleren Zugriffsverzögerung, dem Rahmenverlust und der Latenz ab. Traditionell werden für die deterministische Zugriffsverzögerung TDMA (Time Division Multiple Access) Mediumzugriffsverfahren verwendet. TDMA-basierter Medienzugang ist jedoch keine gute Wahl in einem überlasteten Netzwerk mit externen Geräten, die in demselben Frequenzbereich arbeiten. Die Möglichkeit, einen reservierten Frequenzbereich für deterministische Anwendungen zu verwenden, ist begrenzt und teuer. Darüber hinaus wird die Entstehung von IoT, VANETS und anderen drahtlosen Technologien zusätzlichen Verkehr in naher Zukunft schaffen. Abgesehen von der Zuverlässigkeit profitieren industrielle Anwendungen von einem preiswerten und hardwareunabhängigen Automatisierungsnetzwerk.

Die Idee ist, ein statistisch zuverlässiges und kostengünstiges Automatisierungssystem für drahtlose Regelung einzuführen. Ein Kommunikationsnetzwerk auf Basis von IEEE 802.11 (WiFi) bietet eine kostengünstigere Lösung. IEEE 802.11 verwendet standardmäßig das Carrier Sense Multiple Access mit Collision Avoidance (CSMA / CA) Zugriffsverfahren. Die mittlere Zugriffsverzögerung nach dem CSMA / CA-basierten Verfahren ist nicht deterministisch und hängt von den Medienbedingungen, also der Konkurrenz im Medium, ab. Um statistische Zuverlässigkeit zu gewährleisten, ist ein klares Bild über den Verkehr auf dem Medium erforderlich. In dieser Arbeit werden die möglichen Lösungen zur Erfassung des Medium-Zustands (Medium Sensing) untersucht. Außerdem wurde eine "Token Bucket" basierte Bandbreitenverteilung und ein Netzwerkmessmechanismus vorgeschlagen, um die verfügbare Bandbreite zwischen den Knoten zu verteilen und die lokale Medienkonkurrenz zu erfassen.

Contents

List of Figures

List of Tables

CHAPTER. 1

Introduction

Back in the 1880s, a new revolution in communication systems started with the invention of wireless technologies. Now, wireless is an essential technology for every consumer devices. One of the commonly used radio band is the Industrial, Scientific and Medical (ISM) band, and one of the widespread, and cheap technologies is IEEE 802.11 or WiFi. WiFi's are mainly used for internet connectivity, media streaming, IoTs (Internet of Things), and in a lot of other applications. Low priced hardware and high data transfer makes WiFi suitable for commercial applications. WiFi (IEEE 802.11 b/g/n) is operating mainly on 2.4 GHz, which is also utilized by other devices such as microwave ovens, Bluetooth devices, and cordless phones. This restricts the available medium bandwidth. An alternative is to use the 5 GHz channel (IEEE 802.11 n/ac) which currently is less congested compared to 2.4 GHz, but this might change soon.

Cost benefit, reliability, safety, and management easiness of wireless technologies motivates its use in industrial control systems or industrial automation applications. In the initial stages, Zigbee and Bluetooth were technologies used. Later in 2007, WirelessHART was introduced. WirelessHART [SHM$^+$08] is one of the predominant technologies in control systems for industrial applications operating in 2.4 GHz channel using IEEE 802.15.4e standard radio. It uses Time Synchronized Channel Hopping (TSCH), a combination of Time division multiple access (TDMA) and channel hopping technique to access the medium. TDMA access provides a collision free and deterministic access to the 2.4 GHz medium. WirelessHART provides a channel blacklisting feature, where the network administrator can blacklist congested channels. The Networked Systems group at the Technical University of Kaiserslautern has also developed a system for industrial automation using IEEE 802.15.4 on top of Imote [IMM] platform. This was part of the SINNODIUM project [SIN] used in the "Smart Factory" [SFa] which is an Industry 4.0 project. It also uses TDMA as the medium access method.

As 2.4 GHz ISM band is getting congested, TDMA based approaches require strict control over the operation of other devices on the same channel. Similarly, channel blacklisting also needs to force restrictions over external devices. Talking about medium congestion, the number of Smartphone users and tablets increased drastically in recent years. The number of IoT devices are also increasing day by day. All these technologies are using the same 2.4 GHz channel as the medium for communication. This technological growth will only make the 2.4 GHz ISM band more congested in future. An alternative option to these issues is to use CSMA/CA based network that can only provide statistical guarantees. In [SAB08], author discuss the performance issues and reliability requirements for industrial automation applications. Here, the author also discusses the coexistence issues in 2.4 GHz ISM band as an increasing problem. WiFi technology has already been introduced into this sector [IWL]. IWLAN from Siemens use dedicated hardware and the network use IPCF which is a proprietary extension of standard Point Coordination Function (PCF) [IEE] mechanism for medium access.

The goal is to develop WiFi based control networks for industrial applications independent of dedicated hardware and without modifying the standard IEEE 802.11 MAC layer. Standard WiFi hardware is very cheap priced compared to those dedicated hardware used in WirelessHART or IWLAN. WiFi by default uses CSMA/CA technique to access the medium. As mentioned, TDMA in 2.4 GHz will not provide any advantage in a congested environment and to achieve TDMA in WiFi, changes need to be done in the standard hardware and driver. Another option is to provide statistical guarantees to the network using a centralized bandwidth manager to control the network bandwidth. Bandwidth manager assigns the bandwidth to each node depending on the requirements of each node and available medium bandwidth. The bandwidth requirement varies from node to node depending on the applications it is running. To do this, the bandwidth manager needs information about the medium conditions periodically. Once, the bandwidth manager has information about the medium, it can distribute the resources or bandwidth among the nodes with statistical guarantees.

In this thesis, different methods to sense the medium conditions using a standard WiFi adapter are investigated. The information obtained from the sensing are prioritized based on its quality, and clarity to identify the medium condition. Once the centralized bandwidth controller has information about available bandwidth resource, it is necessary to distribute this resource among the nodes. Here, also proposed a method to distribute the bandwidth among the nodes dynamically depending on the medium condition. The primary focus is to investigate the information that can obtain from a WiFi hardware without creating any patches for the standard driver. The previous project based on Imotes platform was terminated due to the unavailability of the hardware. Therefore, the goal is to create a control system network which is hardware independent, low-priced, and that can cope with coexistence in the channel.

The structure of the thesis report is as following. Chapter 2 discuss about the conceptual motivation behind the work, hardware requirements for the experiments, and its applications. In Chapter 3, related work or similar concepts used in different areas are discussed. In Chapter 4, the communication model, the method used for sharing the bandwidth, and the parameters used for the experiments are defined. Later in Chapter 5, the experiments conducted as part of the thesis are defined, and also the experiment results are discussed along with other findings during the experiments. Finally in Chapter 6, future works based on the findings are discussed followed by the conclusion of the thesis.

CHAPTER. 2

Requirements

In this chapter, the conceptual and technical motivation behind the work, and the requirements to achieve the results are discussed. Initially, discuss the motivation behind the work, followed by the technical requirements for the work and finally, the applications of the work.

2.1 Conceptual Requirements

Reliability of a communication network can be defined as the probability that any desired function can be fulfilled during a specified time period under given working conditions [Sei10]. For a non-deterministic wireless sensor network working on top of WiFi (IEEE 802.11 b/g/n), it is impossible to provide offline guarantees. WLAN operates mainly on license free 2.4 and 5 GHz ISM band. Since the 2.4 GHz is license free, other wireless devices also operate in the same frequency band. WiFi shares the 2.4 GHz frequency band with Bluetooth devices, cordless phones, and microwave oven, which cause interference. To access the medium, standard IEEE 802.11 uses CSMA/CA medium access protocols [C^+99]. Apart from CSMA/CA medium access approach, researchers also proposed TDMA based approaches for Real Time applications [CPVM10] [DM09]. But these TDMA based approaches have strict assumptions about the operation of other devices on the same frequency. As mentioned, 2.4 GHz is used by a lot of other devices which makes the medium access competitive. TDMA based approaches have high probability for frame corruption due to interference from other devices operating on the same channel. In [HXBZ09] [SG05], researchers analyzed the impact on IEEE 802.11 due to other devices operating on the same 2.4 GHz channel. Using TDMA is not a good idea whenever there are other external devices which are uncontrollable.

In this thesis, the focus is on building the fundamentals for improving the reliability of the sensor network running on top of IEEE 802.11 by dynamically sensing the network for packet loss, congestion, and latency. Also, propose a method to share the bandwidth among the nodes in the network depending on the available medium bandwidth and application requirements. Through dynamic sensing, it is possible to identify, and react against congestion in the medium and reduce the workload of overloaded nodes. To achieve these goals, it is necessary to investigate which type of data is useful, available and how to interpret it. It is also necessary to have an idea about the physical layer network parameters that affects the performance of the overall network system.

Using the results, it could be possible to control the non-deterministic network. By controlling the network, it will improve the probability to meet the constraints of an application running on that network. Information such as the number of frame loss, link quality, bandwidth requirements, available bandwidth, the level of medium contention etc. can be aggregated to make decisions for controlling the network. It is necessary to achieve the goals using a standard WLAN adapter, thus resulting building a controlled network system which is independent of hardware, and driver specification.

This thesis focuses on the sensing in single-hop networks, but also lay the foundations for multi-hop sensing. The impact of different parameters in the network layer, MAC layer, and physical layer are also investigated. Next section discusses the technical requirements for the experiments.

2.2 Hardware Requirements

Experiments are planned to conduct on the testbeds from Networked Systems Group of TU Kaiserslautern. Experiments require two testbeds that differs in topology, and external interference. One testbed is located inside university building where there is interference from other wireless networks and the other is a mobile testbed. The mobile testbed is used for conducting experiments in the absence of external traffic. In this document, the testbed inside the university is referred as 'static testbed', while the mobile testbed is referred as 'mobile testbed'. The static testbed is a multi-hop network. The hardware being used as the sensor nodes in the testbeds are Raspberry Pis. The static testbed uses Raspberry Pi 3, while the mobile testbed uses the Raspberry Pi 2 Model B. Raspberry Pi's are single-board computers developed by the British Raspberry Pi Foundation [RPI]. All Raspberry Pi's in both testbeds use class 10 Micro SD cards. The experiments are done on WiFi adapters from TP-LINK (Atheros Chipset) and LogiLink (Ralink Chipset) using Ath9k and rt2800 drivers in Linux kernel version 4.6.5-v7+, respectively. A second WiFi transceiver is connected to the Raspberry Pis of the static testbed, which connects them using infrastructure based networks, e.g to the control the experiment.

Apart from data obtained from standard WiFi transceivers, Software Defines Radios (SDR) can also be used to sense the medium. According to Wireless Innovation Forum, working in collaboration with the IEEE P1900.1 group, SDR can be defined as a "Radio in which some or all of the physical layer functions are software defined" [SDR]. In our case, the SDR is programmed to monitor the IEEE 802.11 spectrum. Data obtained from SDR can be fused with data obtained from the standard WiFi transceiver for more accurate sensing of the network. SDR can also be used to check the correctness of data obtained from standard WiFi transceiver. By analyzing the spectrum, it might also provide some additional information regarding the medium which can be used to sense the network.

Fig 2.1 shows location of nodes in the static testbed. The nodes inside the marked region are used for the experiments. Other WiFi Access Points(AP) are located inside the university building where the static testbed has been deployed. Nodes with node IDs 7, 8, 23, 22, 28 and 29 (in single-hop) are used for the experiments. APs of different research groups are located in the same building. In the same floor, APs are located near to node 7, 23 and 29. Also, the WiFi adapters of node 23 and 28 are placed inside metal bottles and node 7 is located inside a server room with metal shielding, and wall cooling. Thus, the 'static testbed' was created to replicate a factory environment. In mobile testbed, there are six nodes with node ID 38, 39, 40, 41, 42 and 43. Experiments in the absence of external interference using mobile testbed are planned to conduct inside the cellar.

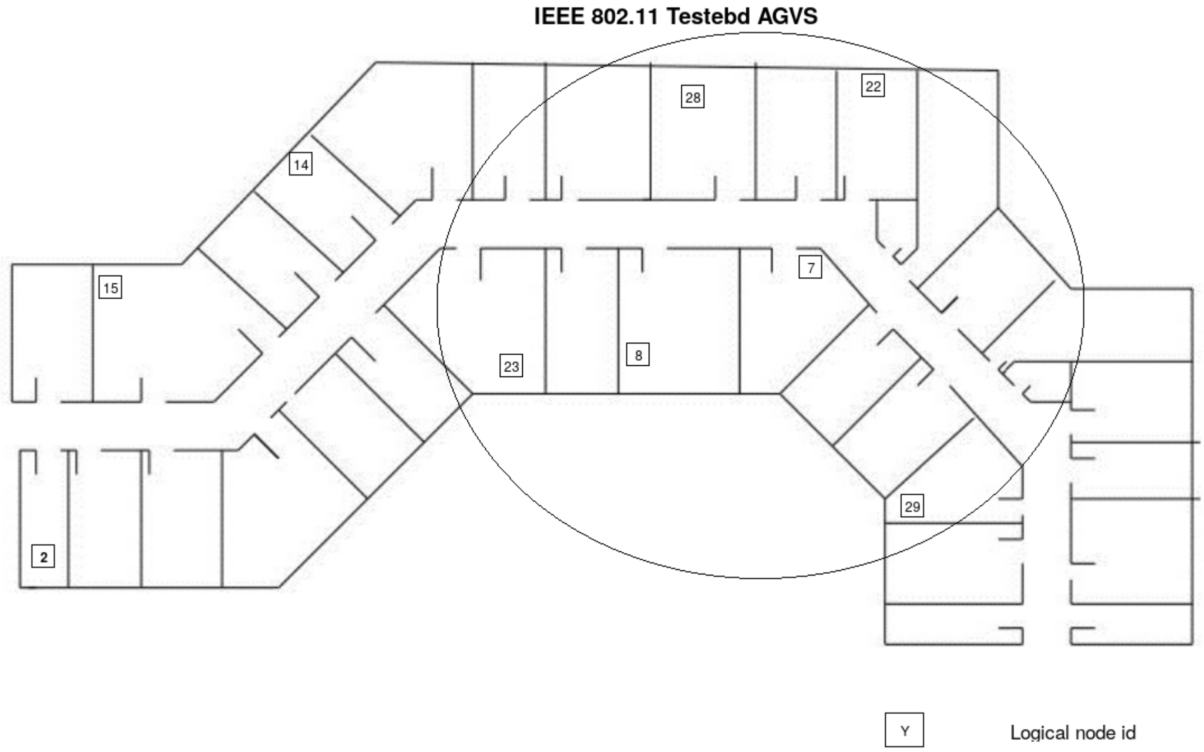


Figure 2.1.: Map showing the location of nodes in the testbed

The experiments are planned to experiment using the framework called *wifiBipsWrapper* developed by Networked Systems Group in TU KL. The WiFi adapters operate in monitor mode and the payload is pushed into the IEEE 802.11 protocol stack along with Radiotap header [Rad] by the framework. Radiotap is the standard header format used for 802.11 frame injection and reception. Theoretically, it can be used for assigning the transmission rate, power, channel and other physical layer parameters for a frame. But, depending on the hardware and driver versions, the features available for frame injection vary. For example, Rt2800 drivers in Linux kernel version 4.6.5-v7+ only supports rate change while Ath9k for USB WiFi adapters (Ath9k_htc) does not have any of those features while Ath9k for PCI Express (Peripheral Component Interconnect Express) WiFi adapters supports every feature. Fig 2.2 shows the architecture of the framework. The framework uses an extensible layered stack based architecture. Next, the different protocol stack layers of the framework are discussed.

As part of the thesis, the idea is to introduce *Network Sensing Layer* and to modify *Bandwidth Management Layer*. The payload from the application enters into the framework through higher layers in the protocol stack. In the experiments, the payload enters directly into the *Bandwidth Management Layer*. Once the frame enters *Bandwidth Management Layer* the frame is pushed into the *Bandwidth Management Layer* queue. This layer verifies whether the node has enough bandwidth for transmitting the payload. If the node has enough bandwidth, the payload is pushed into the *Network Sensing Layer* else it waits inside the queue. It waits until the node gains enough bandwidth for transmitting the payload or, the payload is discarded if it misses its transmission deadline. Inside the queue, the payloads are prioritized based on their deadlines.

The *Network Sensing Layer* monitors or logs the frame transmission and reception of the node. It also senses the medium to analyze the medium condition and external WiFi traffic. Information from medium monitoring could be used to control the network bandwidth allocation. Once an outgoing packet is logged by the layer, it is pushed into the next lower layer, the MAC layer of the framework. MAC layer of the framework creates the MAC Service Data Unit (MSDU) that needed to be transmitted containing the payload and other *wifiBipsWrapper* flags. These flags include sender node ID, sequence number for identifying frame loss, frame type, and other transmission details. It also creates the Radiotap header information for each frame. Then, the MSDU along with Radiotap headers are pushed into the IEEE 802.11 protocol stack for transmission. When there are multiple frames to send, the framework delays pushing the next frame into the IEEE 802.11 protocol stack until it receives an acknowledgment regarding the previous transmission from the kernel, i.e. the frame transmission gets delayed inside the hardware queue while contending for the medium. For the experiments, the hardware retries are not limited.

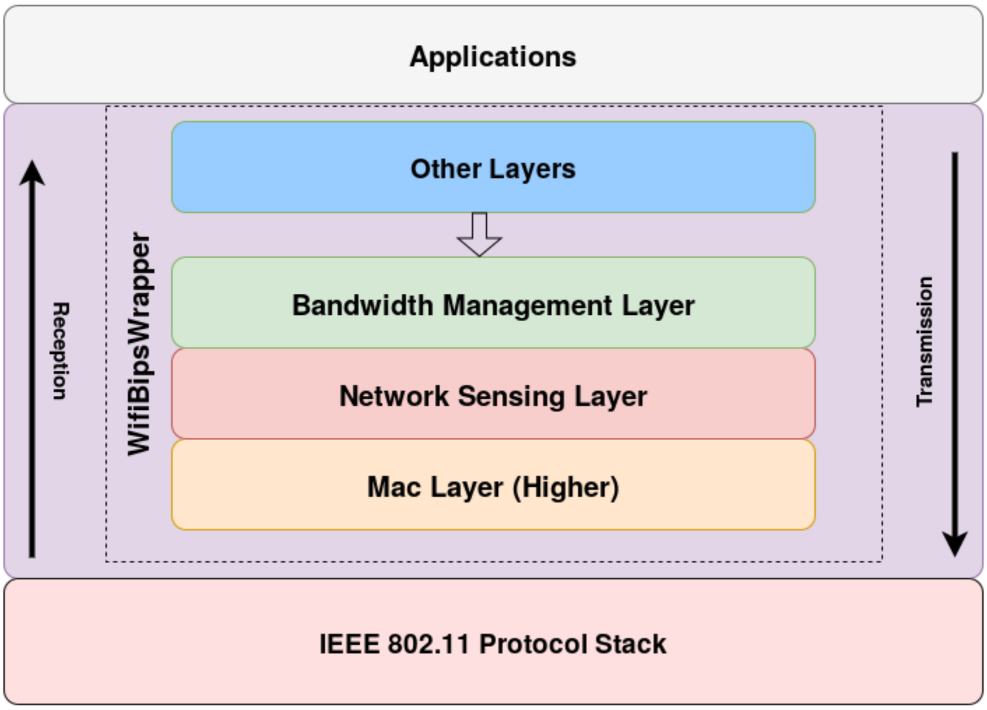


Figure 2.2.: WifiBipsWrapper framework layers

2.3 Applications

The results obtained from sensing a network can be used for different applications. One such application is dynamic topology detection. Topology information is necessary for most of the communication protocols. Some works have already been done on finding dynamic topology information of networks with TDMA [KCG15] and finding static topology information of IEEE 802.11 based multi-hop Ad Hoc networks. To find the dynamic topology information of multi-hop Ad Hoc networks, the data obtained from network sensing could be used. For example, information about the link signal strength, frame loss etc. can be used to observe the link quality. It could be also used to find topologically strong and weak links which could be used for routing.

Bandwidth management is another application where the sensing information could be used for managing the bandwidth requirement of the network or a node. For example, consider three nodes A, B, and C in a single-hop network. An Application running on nodes A and B requires certain extra x bandwidth more than the assigned bandwidth to the nodes. On other hand, applications running on node C have certain unused extra y bandwidth assigned. In such a situation, bandwidth manager can take the extra bandwidth from node C and distribute it among the nodes A and B. For this, the bandwidth manager needs to be aware of the extra bandwidth and required bandwidth of each node in the network. *Network Sensing Layer* provides the bandwidth manager with information about the condition of each node.

CHAPTER. 3

Related Work

In this chapter, similar researches performed by others are discussed. Researchers have studied the impact of frame loss in IEEE 802.11 with respect to data rates, signal to noise ratio (SNR), interference from other sources, and due to other effects. Those results are discussed in the following sections. The token bucket algorithm for traffic shaping, a similar concept used to control the bandwidth management layer is also discussed.

3.1 Effect of Bluetooth and other devices

This section discusses the works conducted to analyze the effects of other wireless devices that operates in 2.4 GHz on IEEE 802.11. The majority of research focuses on understanding the performance degradation of IEEE 802.11 due to interference from Bluetooth devices. In [PTS01] [GVDS$^+$03], researchers analyzed the impact of frame loss, access delays, throughput depending Signal Interference ratio (S/I) and other factors on IEEE 802.11 due to Bluetooth and vice versa. Experiments were conducted to analyze the interference effects on different data rates, transmission power, and frame lengths.

In [PTS01], researchers did experiments on the impact of Bluetooth devices on IEEE 802.11b in different environments. The experiments were conducted in outdoor with less chance for multipath propagation and inside a laboratory. In both setups, the interference from Bluetooth is aimed at the receiver continuously. Experiments were conducted with different transmission power and two different data rates. But the packet length was fixed and to a rather larger value. In both experiments, it was found that frame loss and effective throughput directly depend on S/I ratio. While comparing frame loss and effective throughput with the data rate, it was found that frame loss is high in lower rate which is contrary to intuition. The explanation given by the authors is that sending frames at lower rate takes a lot longer. Even though the encoding scheme of the low rate is less prone to interference, large amount of time required for transmission might increase the chance of packet loss that significantly. To support their argument, experiments with the different frame sizes were not conducted.

In [GVDS$^+$03] [GVDS01], researchers analyzed the impact of frame loss based on WiFi transmission power and offered load, Bluetooth transmission power, hop rate and traffic type on two different stationary topologies in their simulation model. Their experiments on analyzing the frame loss depending transmission power found that increasing the transmission power of transceivers above a certain limit will not improve the loss ratio. On the other hand, increasing the transmission power of IEEE 802.11 had a severe impact on performance of Bluetooth devices. This threshold level for transmission power depends on the distance between the sender and receiver. Regarding performance of IEEE 802.11 based on frame length, it was found that reducing the frame length does not have significant improvement in the performance. Considering the Bluetooth traffic type and hop rate, they found that Bluetooth voice has higher impact on IEEE 802.11 than Bluetooth data traffic and higher Bluetooth hop rates increase the frame loss of IEEE 802.11.

3.2 Medium Sensing and other works

This section discusses the works that use different kinds of medium sensing methods for different goals. To analyze the medium and estimate the available bandwidth, two methods are used. The first method is to estimate the channel load by sniffing the frames using a WiFi adapter in monitor mode. Second is to use the survey information values from the hardware registers of the WiFi adapters. *Medium Utilization Time* defined can be as the estimated time duration where WiFi frames were observed in the medium by a node. This value is calculated by sniffing WiFi packets and estimating the duration for packets. From hardware register, the survey information provides information about the medium busy time. As per [SAB08], this information for hardware registers provides information includes the time spent by the WiFi transceiver for transmission and reception of frames. From the information obtained from hardware driver, *Channel Busy Time* defined as the duration when there is energy in the medium above the noise level. *Channel Active Time* defined as the interface is active[1] duration. In [SAB08], the authors also introduce a control framework that provides QoS to real time applications in static wireless network.

In [JRABR05], the authors use the *Medium Utilization Time* to estimate the channel load and channel congestion in IEEE 802.11b networks. As stated in [ASB$^+$08], a major drawback of this approach is the need to sniff and analyze each and every WiFi packets. In [ASB$^+$08], the authors propose a rate adaption algorithm based on analyzing medium congestion using *Channel Busy Time* value polled from hardware register. As per the authors, *Medium Utilization Time* has a correlation with *Channel Busy Time* with linear correlation coefficient above .9. But, hardware register value *Channel Busy Time* does not differentiate medium energy due to internal and external frames or even due to other devices operating in 2.4 GHz. In [SAB08], the author use *Channel Busy Time* for admission control. In this work, the author utilize the *Channel Busy Time* to calculate the available medium bandwidth. Also, the author proposed a hardware independent (without modification) admission control mechanism to access the medium by sensing the network and analyzing the condition.

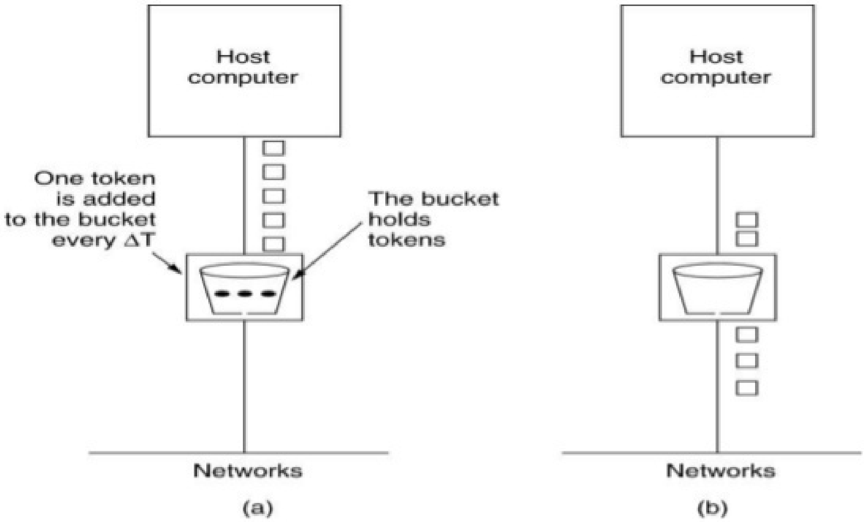
[1] A WiFi adapter is said to be in active mode if it is transmitting or receving any frames

3.3 Token Bucket

Token bucket algorithm [Tan84] is a traffic shaping algorithm to manage the traffic congestion in a packet switched computer network and telephone networks. It is an extension of leaky bucket algorithm [Tan81]. It is used to control the rate of packet transmission.

Consider a bucket that holds the tokens for transmitting a packet. The tokens are periodically refilled into the bucket. The packets are pushed into the queue and waits until enough tokens are refilled into the bucket. When the host collects enough tokens in the bucket, it transmits the packets by consuming respective amount of tokens required for transmission.

The size of the tokens are constant and represent permission to send a certain number of bytes. For transmitting a packet of size L bytes, the bucket needs to collect tokens of atleast total size L. The bucket can only hold some certain number of tokens. Thus, the maximum packet size that a host can transmit is the size of the token bucket. Therefore, the size of the token bucket defines the maximum burst that a node can produce. Fig 3.1 shows the process of the token bucket algorithm. In Fig 3.1:a, the tokens are filled into the bucket every ΔT times. In Fig 3.1:b, three tokens are consumed for transmitting a packet of length three times the token size. Thus by controlling the token size and refilling interval, it is possible to control traffic inside a network.



(a) Before (b) After

Figure 3.1.: Working of a token bucket[Tan84]

Comparing to the leaky bucket, the leaky bucket has a constant transmission rate while using token packets can be send at different rate. In the case of the leaky bucket, the burst size is the size of the leak while in the token bucket it the size of the token bucket. I.e. a packet can consume the complete token inside the bucket for transmitting at higher rate.

CHAPTER. 4

Communication Model

In this chapter, the communication model is defined, which consists of communication topology, frame format, dynamic communication behavior, frame scheduling and the parameters to sense the network.

4.1 Communication topology

Let $G = (V, E)$ be the communication network, consisting of a set of nodes V and a relation (set of links) $E \subseteq V \times V$. Set $V = V^i \cup V^e$, $V^i \cap V^e = \emptyset$, where V^i is the set of internal nodes and V^e is the set of external nodes. $linkQual : E \rightarrow \mathbb{R}^{0,1}$ is a function defining, for each link, its communication quality. Then $E_c \subseteq E$, where E_c is the set of communication links such that $E_c = \{e \in E \mid linkQual(e) \geq linkQual_{min}\}$. $linkQual_{min}$ denotes the quality threshold for a communication link.

4.2 Frame format

Communication between nodes is by the exchange of messages called *frames*. The frame format identifies the information carried by a frame, i.e. the control fields. Furthermore, specialized data fields determined by the receiving node are associated with received frames. This information is the basis to obtain processed data expressing the current network status.

Formally, a frame is a tuple, consisting of a number of fields. Two sets of frames, to distinguish between sent and received frames are introduced. The set F_{tx} defines the format of frames. In addition, F_{rx} contains values that are added/sensed by a node receiving the frame. We also introduce two sets F^e_{tx} and F^i_{tx} that define the formats of external frames and internal frames, respectively. Similarly, two sets F^e_{rx} and F^i_{rx} define values that are added/sensed by a node receiving the external frames and internal frames, respectively.

– $F_{tx} = FrameType \times SndAddr \times DestAddr \times TxRate \times FrameSize$

- $FrameType$: set of frame types, where *FrameType*={ DATA, ACK }
- $SndAddr$: set of sender addresses, where $SndAddr = V$
- $DestAddr$: set of destination addresses, where $DestAddr = V$
- $TxRate$: set of transmission rates, where *TxRate* = {1, 2, 5.5, 9, 11, 12, 18, 24, 36, 48, 54} on physical layer in Mbps
- $FrameSize$: size of the frame in bytes

– $F^e_{tx} = F_{tx}$

– $F^i_{tx} = F_{tx} \times TxPower \times TxTime \times SndSeqNo \times AckSeqNo$

- $TxPower$: transmission power at which the frame was sent in dBm
- $TxTime$: transmission time, the local timestamp of the sender
- $SndSeqNo$: sequence number that uniquely identifies the frame sent by each sender
- $AckSeqNo$: used by Acknowledgment frames to denote acknowledgment to certain Data frame

– $F_{rx} = RcvAddr \times RcvSigStrength \times RxTime \times CorrInd$

- $RcvAddr$: receiver address
- $RcvSigStrength$: received signal strength (RSS) in dBm
- $RxTime$: reception time
- $CorrInd$: corruption indication - true iff frame is corrupted

– $F^e_{rx} = F_{tx} \times F_{rx}$

– $F^i_{rx} = F^i_{tx} \times F_{rx}$

4.3 Dynamic communication behavior

A dynamic communication behavior is a tuple(G, F^d_{tx}, F^d_{rx}), where G denotes the network, $F^d_{tx} = F^{dInt}_{tx}$ and $F^d_{rx} = F^{dInt}_{rx} \cup F^{dExt}_{rx}$ denote the sets of transmitted and received frames, respectively. For the internal nodes, $F^{dInt}_{tx} \subseteq F^i_{tx}$ and $F^{dInt}_{rx} \subseteq F^i_{rx}$, represent the sets of transmitted and received frames, respectively. Similarly, for external network $F^{dExt}_{rx} \subseteq F^e_{rx}$. For a given communication behavior, the resource constraints have to be observed (see Section 4.4). Also, the following consistency constraints have to be maintained:

- For all successfully received internal frames, there must be a corresponding sent frame with the same field values:

$$\forall f \in F^{dInt}_{rx}.(\neg f.corrInd \Rightarrow \exists f' \in F^{dInt}_{tx}.f \mid_{F^i_{tx}} = f')$$

- For a successfully received acknowledgment frame, previously there must be a successful transmission of internal unicast data frame:

$$\begin{aligned}&\forall f \in F_{rx}^{dInt}.(\neg f.corrInd \wedge f.frameType = ACK)\\ &\quad\Rightarrow \exists f' \in F_{rx}^{dInt}.((f'.destAddr = f.sndAddr) \wedge (f'.frameType = DATA)\\ &\qquad \wedge (f'.sndSeqNo = f.ackSeqNo) \wedge (f.rxTime > f'.txTime))\end{aligned}$$

The following notations are introduced:

- $F_{rx}^{d}(v,ws,wn) = \{f \in F_{rx}^{d} \mid f.rcvAddr = v \wedge f.rxTime < ws \cdot wn \wedge f.rxTime > (wn-1) \cdot ws\}$: is the set of frames received by node v during the window wn with window length ws.
- $\forall v \in V^i$, $F_{tx}^{dInt}(v) = \{f \in F_{tx}^{dInt} \mid f.sndAddr = v\}$: set of frames sent by internal node v.
- $\forall v \in V^i$, $F_{tx}^{dInt}(v,t) = \{f \in F_{tx}^{dInt} \mid f.sndAddr = v \wedge f.txTime < t\}$: set of frames sent by internal node v untill time t from the start.
- $F_{tx}^{dInt}(v,ws,wn) = \{f \in F_{tx}^{dInt} \mid f.txTime < ws \cdot wn \wedge f.txTime > (wn-1) \cdot ws\}$: is the set of frames transmitted by node v during the window wn with window length ws.
- $F_{rx,s}^{dInt} = \{f \in F_{rx}^{dInt} \mid \neg f.corrInd\}$: is the set of frames sent and successfully received by internal nodes
- $F_{rx,c}^{dInt} = \{f \in F_{rx}^{dInt} \mid f.corrInd\}$: is the set of internal frames sent and received corruptly by internal nodes
- $F_{rx,s}^{dInt}(v) = \{f \in F_{rx}^{dInt} \mid f.rcvAddr = v\}$: is the set of internal frames successfully received by internal node v
- $F_{rx,s}^{dInt}(v,ws,wn) = \{f \in F_{rx,s}^{dInt}(v) \mid f.rxTime < ws \cdot wn \wedge f.rxTime \geq (wn-1) \cdot ws\}$: is the set of internal frames received by node v during the window wn with window length ws
- $\forall v,v' \in V^i$, $F_{rx,s}^{dInt}(v,v') = \{f \in F_{rx,s}^{dInt} \mid f.sndAddr = v \wedge f.rcvAddr = v'\}$: set of successfully received internal frames on link *e = (v,v')*.
- $\forall v,v' \in V^i$, $F_{rx,c}^{dInt}(v,v') = \{f \in F_{rx,c}^{dInt} \mid f.sndAddr = v \wedge f.rcvAddr = v'\}$: set of corrupted internal frames on link *e = (v,v')*.
- $\forall v,v' \in V^i$, $F_{rx,s}^{dInt}(v,v',t,t') = \{f \in F_{rx,s}^{dInt}(v,v') \mid f.rxTime \geq t \wedge f.rxTime < t'\}$: set of successfully received internal frames on link *e = (v,v')* between the time t and t'.
- $\forall v \in V^i$, $F_{rx}^{dExt}(v) = \{f \in F_{rx}^{dExt} \mid f.rcvAddr = v\}$: set of external frames observed by internal node v
- $F_{rx}^{dExt}(v,ws,wn) = \{f \in F_{rx}^{dExt}(v) \mid f.rxTime < ws \cdot wn \wedge f.rxTime \geq (wn-1) \cdot ws\}$: is the set of external frames received by node v during the window wn with window length ws

4.4 Frame scheduling

Each node has a bandwidth profile and application profile that defines the bandwidth settings and frame arrival timings depending on the application. The bandwidth mentioned before refers to the time allocated for sending to each node by the framework[2]. It includes maximum medium occupancy duration, including MAC frame send duration, physical layer (PHY) overhead, Inter-Frame spacing (IFS), and back-off delay[3]. Application profile defines frame transmission intervals and it varies for each experiment. Next, the details of bandwidth profile and working of the bandwidth allocation by the framework are explained.

To schedule the frames, a variant of the token bucket algorithm (see Section 3.3) is applied. The token bucket algorithm performs the local rate control and thus controls the pace at which a node can transmit frames. The amount of time a node is permitted to use medium depends on its bandwidth profile. For a network G, $assignedBaseBw_G$ denotes the bandwidth assigned (a static value) and $usableBaseBw_G$ denotes the available bandwidth (a dynamic value), that depends on the medium conditions. Similarly, a node v has $assignedBaseBw_v$, that depends on the network settings. While assigning the bandwidth to the nodes, it is necessary to make sure that the sum of assigned bandwidth's to each node must be less than or equal to the available bandwidth of the network. i.e. $\sum_{v \in V} assignedBaseBw_v \leq assignedBaseBw_G$ and also $assignedBaseBw_G \leq usableBaseBw_G$.

As mentioned in Section 2.2, *wifiBipsWrapper* handles the creation, filtering, queuing and sending of frames. It also manages the bandwidth allocation for each node. Bandwidth allocation in the framework uses the modified concept of the token bucket, which controls the available bandwidth for each node. In this case, tokens denote the time allocated to a node to use the medium for sending frames and each node has a token bucket. So, the bucket contains microseconds of transmission time. Next, the bandwidth profile, which contains the parameters that specify the token bucket of a node v is defined.

The bandwidth profile BP_v of a node $v \in V^i$ is a tuple

$$(maxTxBw_v, refilIntvlBw_v, assignedBaseBw_v)$$

where,

- $maxTxBw_v \in \mathbb{N}_0$ [μs] is the maximum medium occupancy duration for a node v for a single transmission, including MAC frame transmission duration, PHY overhead, IFS and back-off delay
- $refilIntvlBw_v \in \mathbb{N}_0$ [μs] is the refill interval in which the medium occupancy duration collected in the bucket of node v is incremented, and
- $assignedBaseBw_v \in \mathbb{R}^{0,1}$ is the base bandwidth assigned to node v relative to the network bandwidth.

Comparing to the generic token bucket, $maxTxBw_v$, and $refilIntvlBw_v$ denote the bucket size and the token refill intervals, respectively. From the above bandwidth profile for a node v, the remaining bucket parameters are calculated.

Example 4.4.1 *For node 7, BP_7 is :*

$$maxTxBw_7 = 1000\,\mu s$$
$$refilIntvlBw_7 = 50\,\mu s$$
$$assignedBaseBw_7 = 5\%$$

[2]Network load offered from application layer
[3]communication bus delay and other delays

For a node v, the size of the token is denoted as $sizeOfFilling_v$, the total number of increments required for completely filling the bucket is denoted as $noOfFillings_v$, and the time required to completely fill the bucket is denoted as $fillDuration_v$. $availableBw_v(t)$ and $wastedTokens_v(t)$ denote the available bandwidth and the number of tokens wasted by a node v until time t. Tokens are wasted due to bucket overflow and it occurs due to the following conditions:

1. when there are not enough frames to transmit or,
2. when the bandwidth manager allocates more bandwidth to a node than the available bandwidth. As mentioned in Section 2.2, when there are multiple frames to send, the framework only push the latter into IEEE 802.11 protocol stack, once it receives acknowledgment for transmission of the former frame. The bandwidth manager estimates this delay, but it is not possible to predict the delay due to channel sensing (CSMA) and communication bus latency[4]. When bandwidth manager assign more bandwidth than available bandwidth, the frames get delayed due to medium contention, thus resulting in wasted tokens.

$sizeOfFilling_v$, $noOfFillings_v$, and $fillDuration_v$ are calculated using the following equations:

$$sizeOfFilling_v = refilIntvlBw_v \cdot assignedBaseBw_v \tag{4.1}$$

$$noOfFillings_v = \lceil \frac{maxTxBw_v}{sizeOfFilling_v} \rceil \tag{4.2}$$

$$fillDuration_v = noOfFillings_v \cdot refilIntvlBw_v \tag{4.3}$$

Example 4.4.1 (cont) *The remaining bucket parameters for node 7 are,*

$$sizeOfFilling_7 = refilIntvlBw_7 \cdot assignedBaseBw_7 = 50\,\mu\text{s} \ \cdot 5\% = 2.5\,\mu\text{s}$$
$$noOfFillings_7 = \lceil \frac{maxTxBw_7}{sizeOfFilling_7} \rceil = \lceil \frac{1000\,\mu\text{s}}{2.5\,\mu\text{s}} \rceil = 400$$
$$fillDuration_7 = noOfFillings_7 \cdot refilIntvlBw_7 = 50\,\mu\text{s} \ \cdot 400 = 20000\,\mu\text{s}$$

Fig 4.2 and Fig 4.1 shows the working of bandwidth allocation. Fig 4.1 shows the idea of bandwidth manager. Bandwidth manager assigns the bandwidth, $assignedBaseBw$ to each node depending on their requirements and available bandwidth of the network. As you can see in the Fig 4.2, each frame requires certain bandwidth for transmission. Once the node earns the required bandwidth for sending the frame, frames are pushed into the sending queue.

Bandwidth profile of a node depends on applications running on top of it. An application can be periodic or aperiodic. For all periodic applications, the bandwidth requirement of a node is calculated depending on the application profile. The application profile AP_x of an application x is a tuple :

$$(payloadSize_x, applPeriod_x, noOfMsgsInPeriod_x)$$

where,

- $payloadSize_x \in \mathbb{N}_0$ is the payload size in bytes.
- $applPeriod_x \in \mathbb{N}_0$ [μs] is period length for periodic applications. Set to zero for aperiodic applications.

[4]experiments are conducted using USB WiFi adapters

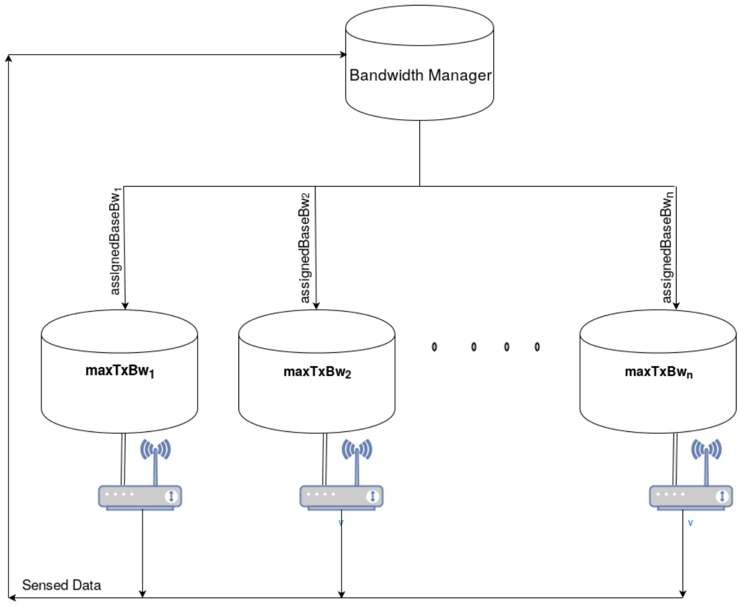


Figure 4.1.: Bandwidth manager assigns the bandwidth to each node depending on the information sensed by each node. The node transmit frame when their bucket collects enough bandwidth for sending the frame

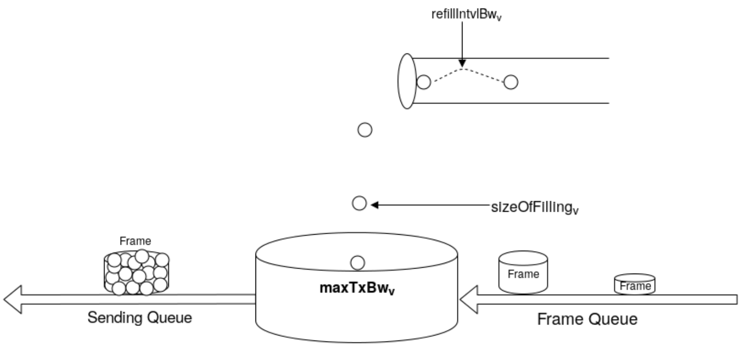


Figure 4.2.: Token bucket of a node

- $noOfMsgsInPeriod_x \in \mathbb{N}_0$ number of messages to send inside a period for a periodic application.

$maxTxBw_v$ of a node v depends on the *payloadSize* of an application and the transmission rate, $txRate$. $maxTxBw_v$ is calculated using the following equation:

$$maxTxBw_v = \frac{payloadSize + MAC\_OVERHEAD}{txRate}[\mu s] + PLO_{avg} \tag{4.4}$$

where $MAC\_\ OVERHEAD$ is the MAC header length and in this case, its a constant size of 66 bytes. PLO_{avg} denotes the average physical layer overheads[5] and calculated as $288.5\mu s$. Since the testbed use Ralink adapters, the average backoff is considered as $263.5\mu s$ [GLMC14]. Further, $assignedBaseBw_v$ is calculated as:

$$assignedBaseBw_v = \frac{maxTxBw_v \cdot noOfMsgsInPeriod_x}{applPeriod_x} \tag{4.5}$$

$refilIntvlBw_v$ parameter is independent of the application.

[5]average of (DIFS (distributed interframe space)+ BO(variable back-off))

Node id	$maxTxBw_v$	$refilIntvlBw_v$	assigned $BaseBw_v(\%)$	$sizeOfFilling_v$
7	1000 μs	50 μs	5 %	2.5 μs
8	1000 μs	50 μs	5 %	2.5 μs
12	1000 μs	50 μs	5 %	2.5 μs
21	1000 μs	50 μs	5 %	2.5 μs
23	1000 μs	50 μs	5 %	2.5 μs
29	1000 μs	50 μs	5 %	2.5 μs

Table 4.1.: A sample bandwidth profile, where $usableBaseBw_G$ = 30%

Example 4.4.2 *Consider a periodic application appX with following values and let us assume that the transmission rate is 1 Mb/s and $refilIntvlBw$ = 1000 µs:*

$$payloadSize_{appX} = 300\,bytes$$

$$applPeriod_{appX} = 10^7\,\mu s$$

$$noOfMsgsInPeriod_{appX} = 5$$

$$\begin{aligned} maxTxBw_v =& \frac{300B + 52B}{1Mbps} + 288.5\mu s \\ =& \frac{2816b}{1Mbps} + 288.5\mu s \\ =& 2816\mu s + 288.5\mu s = 3104.5\mu s \end{aligned}$$

$$assignedBaseBw_v = \frac{maxTxBw_v \cdot noOfMsgsInPeriod_x}{applPeriod_x} = \frac{3104.5\mu s \cdot 5}{10^7\mu s} = 0.00155225\ or\ 0.15\%$$

The remaining parameters can be calculated using Equation 4.1, Equation 4.2 and Equation 4.3.

As mentioned in Section 2.1, in this thesis, the focus is on a single-hop network. For this, 6 nodes from the testbed, which are in single-hop distance are considered. From the testbed Fig 2.1, the nodes with node id 23, 8, 7, 12, 21 and 29 are selected for the experiments i.e. set $V = \{23, 8, 7, 12, 21, 29\}$. Table 4.1 shows a sample bandwidth profile for six nodes in single hop. In this case, available bandwidth for the network G, $usableBaseBw_G$ = 30%.

4.5 Processed Data

In this section, the sensing parameters are defined. Processed data are obtained from aggregating the raw data. The parameter t_{start} is used to denote the time at which a node starts.

- $noFramesSent(v)$: total number of frame sent by node v

$$noFramesSent(v) = |F_{tx}^{dInt}(v)| \tag{4.6}$$

- $totalNoFrames_{v,v'}$: total number of frames successfully sent on link v, v'.

$$totalNoFrames_{v,v'} = |F^{dInt}_{rx,s}(v, v')| \tag{4.7}$$

- $noCorruptedFrames_{v,v'}$: the number of frames corrupted on link v, v'.

$$noCorruptedFrames_{v,v'} = |F^{dInt}_{rx,c}(v, v')| \tag{4.8}$$

- $noLostFrames_{v,v'}$: the total number of lost frames sent on link v, v'.

$$noLostFrames_{v,v'} = |F^{dInt}_{tx}(v)| - |F^{dInt}_{rx,s}(v, v')| \tag{4.9}$$

- $maxContinuousLoss_{v,v'}$: maximum number of continuous frame loss occurred on a link v,v'.
- $lossRatio_{v,v'}$: the loss ratio on the link v, v'. It is calculated as,

$$lossRatio_{v,v'} = \frac{noLostFrames_{v,v'}}{noFramesSent(v)} \tag{4.10}$$

- $corruptRatio_{v,v'}$: the ratio of corrupted frames on the link v, v'. It is calculated as,

$$corruptRatio_{v,v'} = \frac{noCorruptedFrames_{v,v'}}{noFramesSent(v)} \tag{4.11}$$

- $noInteFrameInWindow^{ws}_{v}(wn)$: defines the number of internal frames transmitted by an internal node during the window wn with a window length of ws.

$$noInteFrameInWindow^{ws}_{v}(wn) = |F^{dInt}_{tx}(v, ws, wn)| \tag{4.12}$$

- $noAvgInteFrameInWindow^{ws}_{v}(t)$: defines the average number of internal frames received during each window of size ws for an experiment duration of t from the start time.

$$noAvgInteFrameInWindow^{ws}_{v}(t) = \frac{|F^{dInt}_{rx,s}(v)|}{n} \tag{4.13}$$

where $n = \dfrac{t - t_{start}}{ws}$.

- $avgRSS^{t,t'}_{v,v'}$: holds the RSS average in dBm for a link v, v' in the network between the time t and t'.

$$avgRSS^{t,t'}_{v,v'} = \frac{\sum_{\forall f \in F^{dInt}_{rx,s}(v,v',t,t')} f.rcvSigStrength}{|F^{dInt}_{rx,s}(v, v', t, t')|} \tag{4.14}$$

- $channelBusyTime_{v}(t)$: holds the value *Channel Busy Time* of the node v at time t.
- $channelBusyTimePer_{v}(t)$: the percentage of the time, the medium was observed busy by Node v until time t.

$$channelBusyTimePer_{v}(t) = \frac{channelBusyTime_{v}(t) - channelBusyTime_{v}(t_{start})}{t - t_{start}} \tag{4.15}$$

- $channelBusyTimeInWindow_v^{ws}(wn)$: the amount of time channel was busy during the window wn with window size ws.

$$\begin{aligned} channelBusyTimeInWindow_v^{ws}(wn) = channelBusyTime_v(t_{start} + (n * ws)) - \\ channelBusyTime_v(t_{start} + ((n-1) * ws)) \end{aligned} \tag{4.16}$$

- $channelBusyTimeInWindowPer_v^{ws}(wn)$: percentage of the time, the medium was busy at window wn with the window size of ws.

$$channelBusyTimeInWindowPer_v^{ws}(wn) = \frac{channelBusyTimeInWindow_v^{ws}(wn)}{ws} \tag{4.17}$$

- $timeTxNode_v(t)$: is the amount of time a node v used the medium for transmitting frames until time t from the start of an experiment.

$$timeTxNode_v(t) = \sum_{\forall f \in F_{tx}^{dInt}(v,t)} \frac{f.frameSize}{f.txRate} + PLO_{avg} \tag{4.18}$$

- $mediumEnergySensed_v(t)$: For node v, $channelBusyTime_v(t)$ includes the duration where node sensed energy in the medium plus the time taken for transmitting the frames [nl8][cfg]. $mediumEnergySensed_v(t)$ defines the amount of time the node sensed medium energy excluding its transmission duration until time t from the start.

$$\begin{aligned} mediumEnergySensed_v(t) = (channelBusyTime_v(t) - channelBusyTime_v(t_{start})) - \\ timeTxNode_v(t) \end{aligned} \tag{4.19}$$

- $mediumEnergySensedPer_v(t)$: the percentage of the time, the node sensed medium energy excluding its transmission duration until time t from the start.

$$channelBusyTimePer_v(t) = \frac{mediumEnergySensed_v(t)}{t - t_{start}} \tag{4.20}$$

- $mediumUtilizedInWindow_v^{ws}(wn)$: is the *Medium Utilization Time* observed by the Node v during the window wn. It is calculated by using the information from internal and external frames captured by the node.

$$mediumUtilizedInWindow_v^{ws}(wn) = \sum_{\forall f \in F_{rx}^{d}(v,ws,wn)} \frac{f.frameSize}{f.txRate} + PLO_{avg} \tag{4.21}$$

- $mediumUtilizedInWindowPer_v^{ws}(wn)$: percentage of the time, the medium was utilized by the WLAN frames.

$$mediumUtilizedInWindowPer_v^{ws}(wn) = \frac{mediumUtilizedInWindow_v^{ws}(wn)}{ws} \tag{4.22}$$

- $mediumUtilizedIntFramWindow_v^{ws}(wn)$: defined as the time duration where internal frames were observed on the medium in the window wn with window size ws.

$$mediumUtilizedIntFramWindow_v^{ws}(wn) = \sum_{\forall f \in F_{rx,s}^{dInt}(v,ws,wn)} \frac{f.frameSize}{f.txRate} + PLO_{avg} \tag{4.23}$$

- $mediumUtilizedIntFramWindowPer_v^{ws}(wn)$: percentage of the time, the medium was utilized by internal frames during the window wn.

$$mediumUtilizedIntFramWindowPer_v^{ws}(wn) = \frac{mediumUtilizedIntFramWindow_v^{ws}(wn)}{ws} \tag{4.24}$$

- $mediumUtilizedExtFramWindow_v^{ws}(wn)$: defined as the time duration where external frames were observed in the medium during the window wn.

$$mediumUtilizedExtFramWindow_v^{ws}(wn) = \sum_{\forall f \in F_{rx}^{dExt}(v,ws,wn)} \frac{f.frameSize}{f.txRate} \tag{4.25}$$

- $mediumUtilizedExtFramWindowPer_v^{ws}(wn)$: percentage of the time, the medium was utilized by external frames during the window wn.

$$mediumUtilizedIntFramWindowPer_v^{ws}(wn) = \frac{mediumUtilizedExtFramWindow_v^{ws}(wn)}{ws} \tag{4.26}$$

- $usedBwPer_v$: It calculates percentage of used assigned bandwidth by a node v until time t from the node start time. It is calculated as,

$$usedBwPer_v(t) = \frac{timeTxNode_v(t)}{t \cdot assignedBaseBw_v} \tag{4.27}$$

- $wastedBwPer_v(t)$: the percentage of wasted assigned bandwidth by the node v at time t from the start.

$$wastedBwPer_v(t) = \frac{wastedTokens_v(t) \cdot sizeOfFilling_v}{(t - t_{start}) \cdot assignedBaseBw_v} \tag{4.28}$$

- $wastedNetworkBwPer_G(t)$: the percentage of wasted assigned network bandwidth at time t from the start.

$$wastedNetworkBwPer_G(t) = \frac{\sum_{\forall v \in V^i} wastedBwPer_v^t}{\mid V^i \mid} \tag{4.29}$$

- $wastedTokensInWindow_v^{ws}(n)$: the number of wasted tokens on node v during the window n with window size ws

$$\begin{split} wastedTokensInWindow_v^{ws}(n) = wastedTokens_v(t_{start} + (n \cdot ws)) - wastedTokens_v(\\ t_{start} + ((n-1) \cdot ws)) \end{split} \tag{4.30}$$

- $wastedBwInWindowPer_v^{ws}(n)$: denotes the percentage of wasted assigned bandwidth during the window n with window size ws

$$wastedBwInWindowPer_v^{ws}(n) = \frac{wastedTokensInWindow_v^{ws}(n) \cdot maxTxBw_v}{ws \cdot assignedBaseBw_v} \tag{4.31}$$

- $wastedNetworkBwInWindowPer_G^{ws}(n)$: the percentage of wasted assigned network bandwidth during the window n with window size ws.

$$wastedNetworkBwInWindowPer_G^{ws} = \frac{\sum_{\forall v \in V^i} wastedBwInWindowPer_v^{ws}(n)}{\mid V^i \mid} \tag{4.32}$$

CHAPTER. 5

Experiments

This chapter defines the experiments and their evaluations. First, the experiments defined, with their motivations, frame header values, and physical layer transmission parameters, and the bandwidth configuration used for the experiment. Then, the experiment results are discussed, followed by other findings during the experiment.

5.1 Experiment Definitions

In this section, the experiments are defined. Experiments are divided into three sections based on their purpose. First, to check effects depending on the different transmission settings. Second, experiments conducted to analyze the bandwidth management and the parameters to measure the available bandwidth and medium sensing. Third, sensing the medium with help of SDR and aggregating information with standard hardware.

5.1.1 Cross Layer experiments

Motivation: In IEEE 802.11, the assessments in higher layer dependents heavily on lower layer parameters. For example, the packet loss depends on the medium conditions, data rate, transmitted power and other such factors. So understanding and configuring the lower layer settings can result in achieving better results in the higher layer. The motivation behind the experiment was to analyze the impact of parameters in different layers. As mentioned in Section 3, many works have been done focusing on those aspects. In following experiments, the goal is to reproduce and analyze those results using the standard driver.

Frame header values:
These are the frame configuration parameters used for the following experiments. $\forall f \in F^i_{tx}$ of each node v are :

$$f.frameType = DATA, f.sndAddr = v, f.destAddr = 65535,$$
$$f.frameSize = 1467 \text{ bytes}^{6}, f.txPower = 16\ dBm$$

As mentioned in Section 4.3, the parameters *f.txTime, f.seqNo* are dynamic values. Transmission rate, $f.txRate$ for each frame depends on the sub-experiments. Destination address 65535 is the broadcast address used by the framework.

Bucket parameters:
Depending on the experiment motivation, bandwidth profiles were defined. For the experiment Exp 5.1.1.1, the *Bandwidth Management Layer* is not used. For experiment Exp 5.1.1.2, each node v choose a bandwidth profile, BP_v (11736 μs , 100 μs , 2%). The bucket size, 11736 μs chosen as the maximum transmission time required to sent the frame at a 1 Mbps.

Experiment 5.1.1.1: *Frame loss depending on data rate*

This experiment conducted to analyze the frame loss depending on the transmission rates. Theoretically, lower transmission rates are less prone to get corrupted than higher transmission rates due to the difference in their encoding scheme. In [Bic05], the author proposed a rate selection algorithm and also analyzed about different existing bit-rate selection algorithms and their weakness. The author also points out that using lower rates is not always the best solution. Through this experiment, the motivation is to lay the foundation for choosing transmission rates for the bandwidth manager. The experiments were conducted both in the presence of external traffic and in the absence of external traffic.

As mentioned above, this experiment does not use the *Bandwidth Management Layer*. The frames are sent with a random interval between them to find the frame loss depending on the transmission rate. For each frame, the rates are chosen randomly, i.e $f.txRate = randomRate()$. Function *randomRate()* chooses a random rate from the set *TxRate* for each frame. This reduces the probability of results depending on the medium condition. For example, suppose there is some interference for a certain interval during the experiment, and if frames of one particular transmission rate are sent during this interval, then it could affect the end result. Choosing a random rate for each frame could mitigate these kinds of errors. 10000 frames at each data rate were randomly sent for the experiment in the presence of external traffic and 1000 frames for the experiment in the absence of external traffic. Since the number of transmitted frames is a static value, for each link the number of lost frames can be calculated by comparing with the number of successfully received frames.
Things to observe:
In this experiments, the goals are:

- for all link $v, v' \in V^i$ observe the $lossRatio_{v,v'}$ and $corruptRatio_{v,v'}$ depending on transmission rate.

Experiment 5.1.1.2: *To analyze the correlation between Channel Busy Time and Medium Utilization Time*

As mentioned in Section 3, *Channel Busy Time* can be used to estimate the amount of time the medium was busy. Since this value is polled from hardware, it depends on transceiver's cali-

[6] including the IEEE 802.11 MAC overhead

bration for noise level. Since it strongly depends on the hardware, it is necessary to check the precision of the value for the hardware used in the testbeds. As mentioned, in [ASB+08] the authors stated a correlation between *Channel Busy Time* and Medium Utilization with correlation coefficient of .92 for Atheros chipset. In our testbeds, the experiment is conducted on WiFi adapters using Ralink chipset. In this experiment, the goal is to analyze the correlation for Ralink chipset. For the experiment, a transmission rate, $f.txRate = 1\,Mbps$ is chosen for calculation simplicity.

Things to observe:

In this experiments, the goals are to:

- $\forall v \in V^i$ observe the relation between $channelBusyTimeInWindowPer_v^{ws}(wn)$ and $mediumUtilizedInWindowPer_v^{ws}(wn)$ for each window wn.
- Observe the difference in *Channel Busy Time* register value for two nodes with one node transmitting and other node sensing, i.e. for each window, check the difference in $channelBusyTimeInWindowPer_v^{ws}(wn)$ for those two nodes.

5.1.2 Bandwidth Management

Motivation: The motivation behind the experiment was to analyze and study the values of parameters that are used for controlling the bandwidth of the network. Bandwidth manager using the token bucket method is responsible for allocating the required bandwidth for sending frames depending on the requirements of the application run on the node. In this experiment, the goal is to investigate the parameters that can be used by the bandwidth manager for controlling the network. In these experiments, it is assumed that there is always a frame in bandwidth management queue waiting for the bandwidth bucket to be refilled. So, once the bucket is full[7], the frame is pushed into IEEE 802.11 protocol stack. Six nodes are used for the following experiments and frames were sent as broadcast frames. Next, the frame configuration parameters and the bandwidth profile used for the set of experiments are defined, followed by defining sub-experiments.

Frame header values:

These are the frame configuration parameters used for the following experiments. $\forall f \in F_{tx}^i$ of each node v are :

$$f.frameType = DATA,\ f.sndAddr = v,\ f.destAddr = 65535,\ f.txRate = 1\ Mbps$$
$$f.frameSize = 480 \text{ bytes}^8,\ f.txPower = 16\ dBm$$

Parameters *f.txTime*, *f.seqNo* are dynamic values, given by the *wifiBipsWrapper* framework.

Bucket parameters:

For the following experiments, a frame length of 480 bytes and data rate of 1 Mbps is chosen. Therefore maximum medium occupancy duration for a frame, $maxTxBw$ is:

$$maxTxBw_v = \frac{f.frameSize}{f.txRate} = \frac{480\,bytes + MAC\_OVERHEAD}{1\,MBps} + PLO_{avg} = 4545\,\mu\text{s}$$

[7] it gains enough bandwidth for transmission

[8] excluding the IEEE 802.11 MAC overhead

The remaining bucket parameters of each node differs depending on the bandwidth profile BP_v. For example, for BP_v = (4545 μs ,100 μs ,5%):

- $sizeOfFilling_v$ = 100 * .05 = 5 μs
- $noOfFillings = \lceil \frac{4545}{5} \rceil = 909$
- $fillDuration_v$ = 909 · 100 = 90900 μs

Profile No	*Bandwidth Profile*	$sizeOfFilling$	$noOfFillings$	$fillDuration_v$
P.5.1.a	(4545 μs ,100 μs ,.5%)	0.5 μs	9090	909000 μs
P.5.1.b	(4545 μs ,100 μs ,1%)	1 μs	4545	454500 μs
P.5.1.c	(4545 μs ,100 μs ,2%)	2 μs	2273	227300 μs
P.5.1.d	(4545 μs ,100 μs ,5%)	5 μs	909	90900 μs
P.5.1.e	(4545 μs ,100 μs ,6%)	6 μs	758	75800 μs
P.5.1.f	(4545 μs ,100 μs ,9%)	9 μs	505	50500 μs
P.5.1.g	(4545 μs ,100 μs ,10%)	10 μs	455	45500 μs
P.5.1.h	(4545 μs ,100 μs ,11%)	11 μs	414	41400 μs
P.5.1.i	(4545 μs ,100 μs ,12%)	12 μs	379	37900 μs
P.5.1.j	(4545 μs ,100 μs ,13%)	13 μs	350	35000 μs
P.5.1.k	(4545 μs ,100 μs ,14%)	14 μs	325	32500 μs
P.5.1.l	(4545 μs ,100 μs ,15%)	15 μs	303	30300 μs
P.5.1.m	(4545 μs ,100 μs ,16%)	16 μs	303	30300 μs

Table 5.1.: Bandwidth profiles used for the experiments

Similarly, Table 5.1 shows the values of remaining bandwidth bucket parameters for certain bandwidth profile. Even in the absence of external APs, maximum $usableBaseBw_G$ is less than 100% [CVB02], but let us assume the maximum $usableBaseBw_G$ is 100%. For the experiments, in a single run, each node uses the same bandwidth profile. I.e. during an experiment with bandwidth profile *5.1.e* and *5.1.l*, the 6 nodes utilize 36% and 90% of the total medium (network load), respectively.

Experiment 5.1.2.3: *Network sensing using wasted tokens*

This experiment conducted to check the idea of wasted tokens depending on the network load. This experiment mainly focus on observing the relation between the number of wasted tokens and assigned base bandwidth. Experiment consist of several runs and in each run, nodes uses a certain bandwidth. If the medium is busy, nodes will start losing the tokens for transmitting each frame due to medium contention, which calculated as the wasted assigned base bandwidth ($wastedBwPer^t_v$) for the node. It occurs either due to the traffic from the external nodes or an internal node.

The experiment is conducted in the static testbed (Fig 2.1) with 6 nodes on channel 1 (2.412 GHz) and 6 (2.437 GHz) in presence of external traffic, and on a mobile testbed in absence of external traffic. Different APs are operating on different channels inside the university building. Running the experiment on different channels helps to observe the relation between wasted tokens and external traffic.

Things to observe:
In this experiments, the goals are to:

- $\forall v \in V^i$, observe $usedBwPer_v^t$ and $wastedBandwidthPer_G^t$ depending on $assignedBaseBw_v$ and $assignedBaseBw_G$, respectively. This provides the information about the medium condition observed by each node. i.e higher $wastedBandwidthPer_v^t$ implies high contention in the medium, which results in delaying or dropping frames. The time period t is the duration of the experiments.
- For each bandwidth profile, analyze the $wastedNetworkBwPer_G^t$ for a similar time period t with respect to $assignedBaseBw_G$.

Experiment 5.1.2.4: *Sensing medium congestion at low and high node base bandwidth profiles*

The medium could get congested either due to the traffic from internal nodes or due to external traffic. In this experiment, the wasted bandwidth in each node is analyzed for low and high bandwidth profiles.

Even though the nodes selected in the static testbed are a single hop, the link quality between each node depends on their topology. This affects the number of internal frames received by each node. Thus, contention between the internal nodes can result in wasted tokens.
Things to observe:
In this experiments, the goals are to:

- $\forall v \in V^i$ compare $channelBusyTimePer_v(t)$ and $mediumEnergySensedPer_v(t)$ to $wastedBwPer_v^t$, for each experiment run at lower and higher assigned base bandwidth profiles.
- When the nodes start to lose tokens, check whether it is due to the internal nodes or external nodes. I.e. relation between $wastedBwPer_v^t$ for the experiment duration t with $noAvgInteFrameInWindow_v^{ws}$. Relating to the number of internal frames received is similar to estimating the medium usage by internal frames since length and transmission rate of internal frames are same for the experiment.

Experiment 5.1.2.5: *Frame loss depending on offered network load*[9]

This experiment is conducted to analyze the frame lost in each link, medium energy observed by each node depending on the assigned bandwidth, and total network load. It was conducted in single hop topology with six nodes.

Nodes were assigned a base bandwidth, $assignedBaseBw_v$ of .5%, 1%, 2%, 5%, 10%, and 15% resulting in a network load of 3%, 6%, 12%, 30%, and 90%, respectively. These higher and lower bandwidth configurations are selected to check whether there is a noticeable difference in the frame loss ratio (in the medium) at higher network loads and channel busy time. Bandwidth profiles are defined in Table 5.1
Things to observe:
In this experiments, the goals are to:

[9]The network load refers to the load from the application layer, not the load in physical layer

- observe the frame loss depending on each bandwidth profile and total network load. i.e. $\forall v, v' \in V^i$ observe the $lossRatio_{v,v'}$ and $corruptRatio_{v,v'}$ with respect to $assignedBaseBw_v$. Also, compare with the $usedBwPer_v^t$ where time period t is for the total duration of experiment run time.

5.1.3 Network Sensing using SDR

Motivation: In this experiment, the idea is to use an SDR/USRP to improve the accuracy of measurement. The IEEE 802.11 operates on 2.4 GHz and 5 GHz frequency. The experiments are done on 2.4 GHz channel. Apart from IEEE 802.11, there are a lot of other wireless devices which uses 2.4 GHz and it could create unpredictable interference and frame loss in the network. This could be sensed and measured precisely with the help of an SDR. Even though medium can be sensed using most of the standard WIFI hardware, it is important to check its precision. The motivation behind the experiment is to check the impact of other devices on the same channel, and also the impact of other IEEE 802.11 devices which are not part of the network.

Things to observe:
In this experiments, the goals are to:

- compare the $mediumEnergySensedPer_v(t)$ and $channelBusyTimeInWindowPer_v^{ws}(wn)$ for a node v and energy sensed by the SDR. Filter the samples obtained from SDR for different samples.

5.2 Evaluation

In this section, experiment results and other findings are discussed. First, discusses the main objectives of the experiments and their results. Later, discusses additional findings observed during the experiments, which were not the primary objectives.

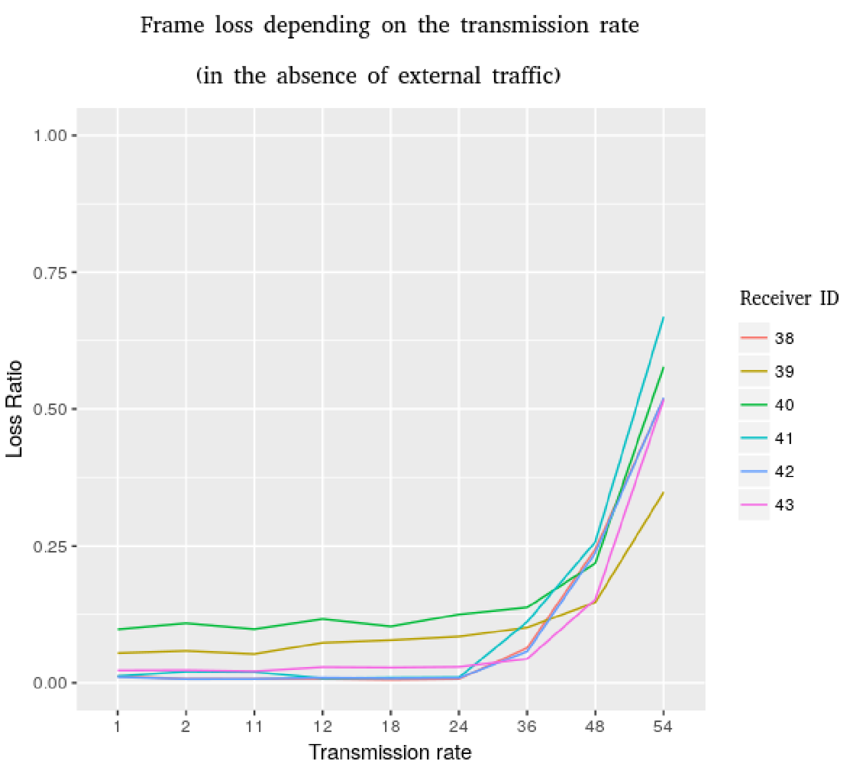
5.2.1 Experiment Results

This section discusses the main result of each experiment and how could these results be used for controlling the network.

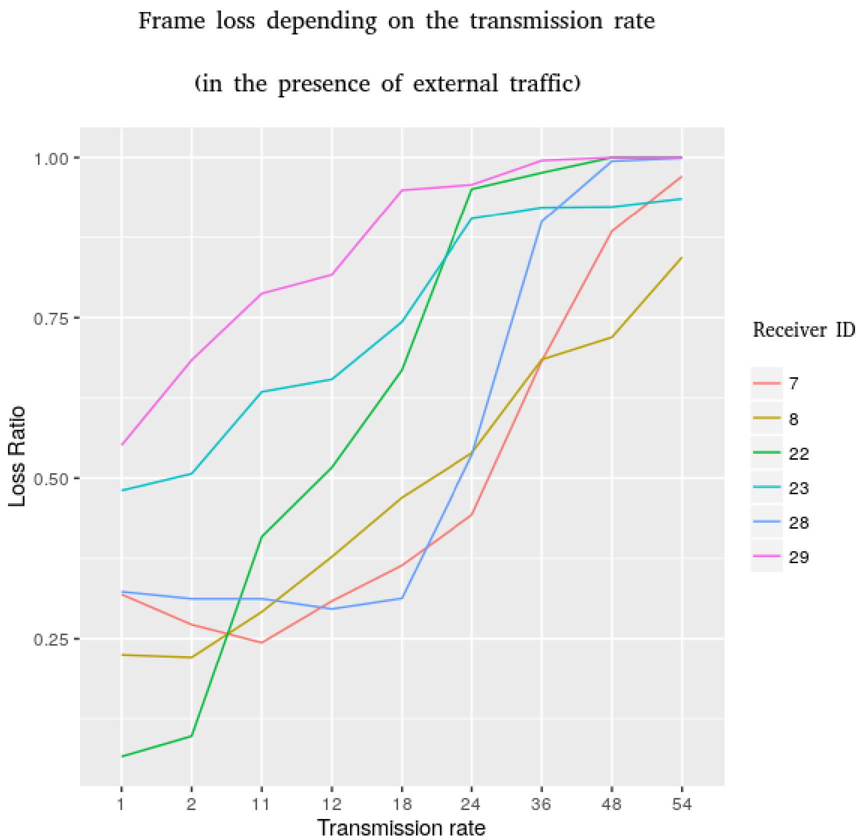
Experiment 5.1.1:

Here, the result of the experiments that depend on the physical layer parameters is discussed.

Experiment 5.1.1.1: *Frame loss dependent on data rate*



(a) Frame loss ratio observed by each node depending on the transmission rate in the absence of external



(b) Frame loss ratio observed by each node depending on the transmission rate in the presence of external

Figure 5.1.: Experiment conducted to check the frame loss ratio depending on the transmission rate in the absence of external traffic

In this experiment, frame loss in each link is analyzed depending on the transmission rate. The experiments were conducted on channel 6 (Central frequency of 2437 MHz). First, let us discuss the results of the experiment conducted in the absence of external traffic and then, in the presence of external traffic

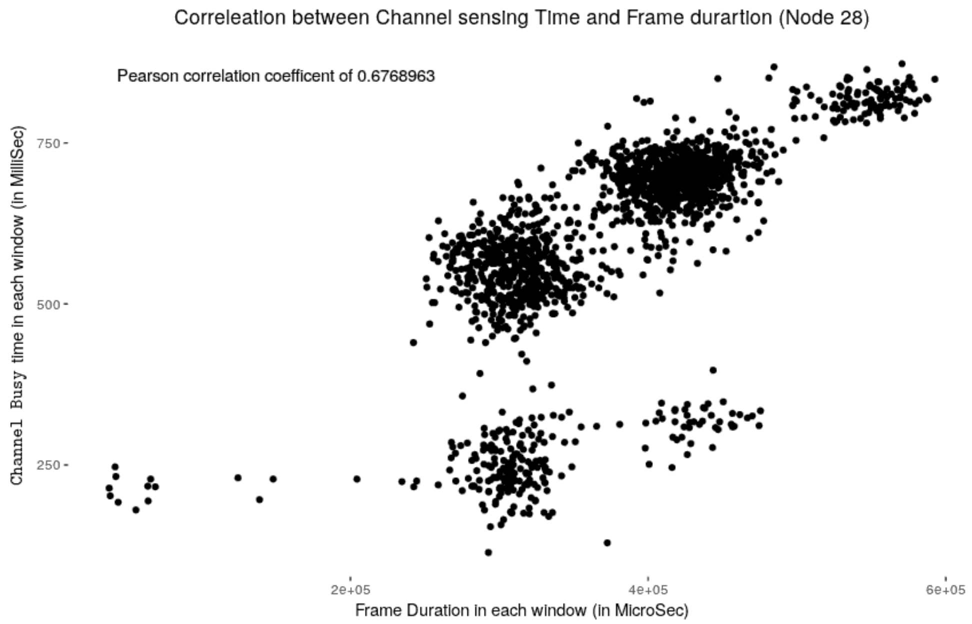
Fig 5.1a shows the results of the experiment conducted in the absence of external traffic. Fig A.1 shows the results for each link. The results show the expected behavior that, the lower the transmission rate, less chance for the frame to get corrupted or lost. The nodes were in single-hop with a maximum distance of 1m between them. In general, a higher frame loss ratio observed for transmission rate at 36 Mbps or above. But below 36 Mbps, the frame loss ratio is irrespective of the transmission rate. For certain links, even at a transmission rate of 54 Mbps, the frame loss ratio is less than .01 %. This shows that the frame loss ratio for a link may or may not depend on the transmission rate. But in most cases, it depends on the transmission rate.

Fig 5.1b shows the results of the experiment conducted in the presence of external traffic. Fig A.2 shows the results for each link. The distance between the nodes in the testbed is longer compare to the distance between nodes for the experiment in the absence of external traffic. Fig A.2g shows the result of the whole network, i.e. frame loss in each node as the receiver depending on the transmission rate. Fig A.2a to Fig A.2f shows the frame loss ratio for each link with the receptive node as the sender. For example, Fig A.2a shows the frame loss ratio for the links with Node 7 as the sender. As you can see from the Fig A.2g, from the network point of view, nodes 23 and 29 have almost 50% frame loss ratio in receiving at 1 Mb/s. This due to the poor quality of links between nodes 22 $\rightarrow$ 23, 29 $\rightarrow$ 23, and 23 $\rightarrow$ 29, 28 $\leftrightarrow$ 29. In this experiment run, there is no link between 28 $\leftrightarrow$ 29 (even though it receives some corrupted frames), but while repeating the experiment, link with poor link quality was detected. But for later experiments, with one constant transmission rate (Fig A.5f and Fig A.5e), link 28 $\leftrightarrow$ 29 has better frame loss ratio of 22 - 30%. Those experiments use the bandwidth manager to control the traffic of internal nodes. Details about those results are explained in later sections. Overall, lower rates show less frame loss ratio except in some conditions. Fig A.2g shows that node 7 has best reception quality for receiving frames transmitted at a rate of 11 or 12 Mbps, not at 1 Mbps. This might be due to the topological location of node 7 that it is placed inside a server room with servers and metal shield for room cooling. Further looking into each link, links 8 $\rightarrow$ 7, 22 $\rightarrow$ 7, 22 $\rightarrow$ 8, 23 $\rightarrow$ 7, 28 $\rightarrow$ 7, 29 $\rightarrow$ 7, 7 $\rightarrow$ 23, 7 $\rightarrow$ 28 has better reception quality at 11 Mbps. This shows that it is better to choose transmission rate for each link rather than sticking with lower transmission rates. It is also better to use higher transmission rates to meet tight bandwidth requirements until a tolerable frame lose ratio.

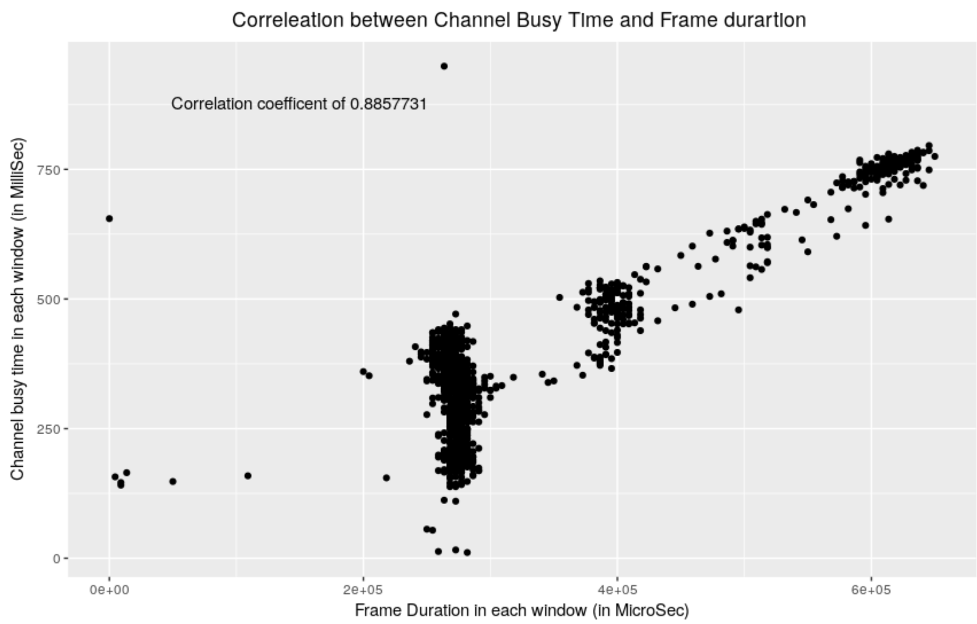
The experiments were repeated and produced similar results. Considering the $avgRSS_{v,v'}^{t,t'}$ of each link for the duration of the experiment where as expected and the average received signal strength of each link depends on their topology. The maximum $corruptRatio_{v,v'}$ for a link observed in the experiment was below .35%.

Experiment 5.1.1.2: *To analyze the correlation between Channel Busy Time and Medium Utilization Time*

The experiment was conducted in the presence and absence of external traffic. Fig 5.2 shows the results of different experiments.

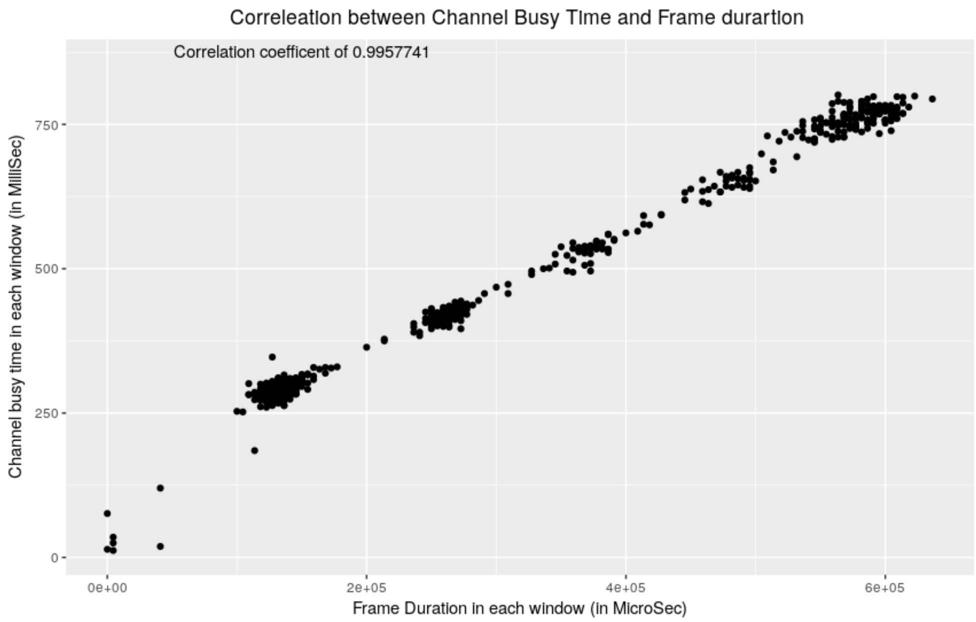


(a) Result observed in the presence of external traffic



(b) Result observed in the absence of external traffic

Figure 5.2.: Graph showing the correlation *Channel Busy Time* and *Medium Utilization Time*



(c) Best result observed in the absence of external traffic

Figure 5.2.: Graph showing the correlation *Channel Busy Time* and *Medium Utilization Time*

Fig 5.2c shows the best result obtained in the absence of any wireless interference. In this result, *Channel Busy Time* has a correlation with *Medium Utilization Time*, with a correlation coefficient of 0.995. This shows that, first, *Channel Busy Time* can be used to sense the medium energy and second, the calculated estimate (Equation 4.4) for the time taken by a frame to transmit is accurate enough. Fig 5.2a shows the result conducted in presence of external traffic and other interference. The correlation is weak with a correlation coefficient of 0.67. This can be either due to the presence of other interference (apart from IEEE802.11 frames) in the same channel.

Node Id	*noFramesSent* (v)	ChannelActiveTime/ t	*channelBusy TimePer*$_v(t)$(ms)	*assigned BaseBw*$_v$(%)
38	8576	299850	39800	15
39	0	299851	47238	0

Table 5.2.: Table showing the results of the Experiment 5.1.1.2

This result shows that it is better to rely on *Channel Busy Time* than on *Medium Utilization Time*. Calculating *Medium Utilization Time* brings extra overhead since it needs to capture and process the each every external frame. So it is better to rely on *Channel Busy Time* which can polled from the hardware register using Netlink Protocol Library Suite [Net]. But it was observed that *Channel Busy Time* underestimate the channel usage in certain situations. In Fig 5.2a the WiFi adapter sensed less medium energy than expected for certain windows. The number of windows that underestimated the medium energy is less than 5 % of the total number of the windows during the experiment.

It is also observed during the experiments that different methods are required to poll values from the hardware depending on the hardware. For a Loglink Adapter with Ralink chipset, when it is polled to receive the register values such as *Channel Busy Time* or *Channel Active Time*, the register value get reset to zero. But in the case of TPlink adapters with Atheros chipset, it register values won't reset to zero during each polling. Apart from that, for a Loglink Adapter with Ralink chipset, the register has an upper limit of 3.72e+9 Microseconds, i.e the values should be polled at least once in every 3.72e+9 Microseconds.

Another motivation behind the experiment was to compare the register values of two nodes with one node transmitting and another node sensing. Table 5.2 shows the results of the experiment in absence of external traffic with node 38 transmitting and node 39 sensing. The experiment was conducted for 299850 ms and a total of 8576 frames were transmitted by node 38. As you see in Table 5.2, node 39 sensed more energy comparing to node 38, i.e. a difference of 7438 ms. As per our calculation mentioned in Section 5.1.1, time required to transmit one frame is 4545 μs . Here, node 38 transmitted 8576 similar frames. i.e. total time required as per the calculation is $8576 \cdot 4545 = 38977920$ μs or 38978 ms. Node 38 observed a *Channel Busy Time* for 39800 ms. This also shows that estimation Equation 4.4 for calculating frame transmission time is accurate for estimating the duration. On the other hand, node 39 showed more time with a difference of 7438 ms. This difference can be either due to the difference in calibration of noise level by node 39. As mentioned, the experiment was conducted in the absence of external traffic where medium energy is comparatively too low. Thus, the node might have calibrated to much lower noise level resulting in sensing more energy.

Experiment 5.1.2:

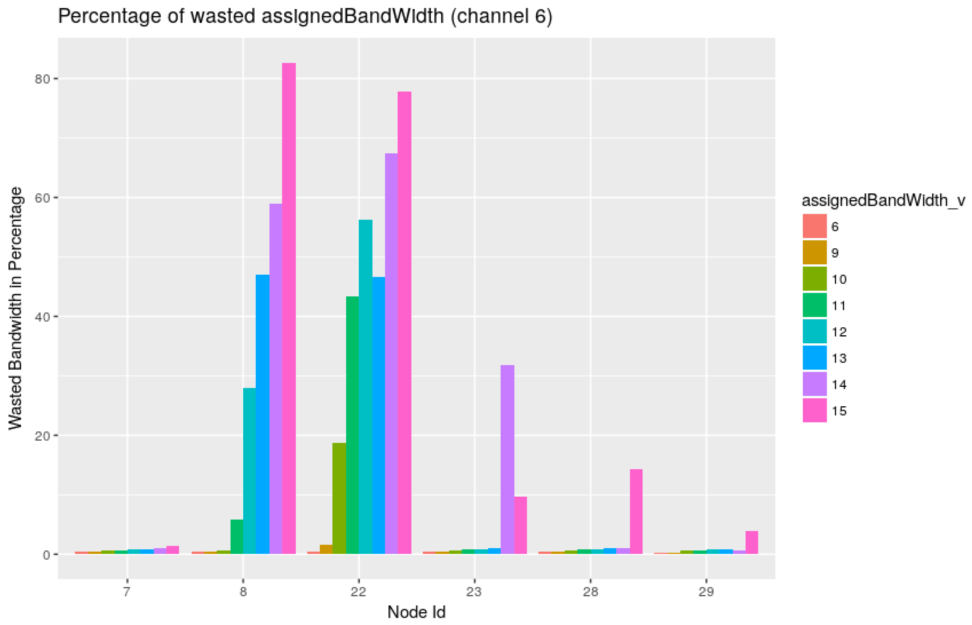
Here, the results of the experiments for controlling the bandwidth of the network are discussed.

Experiment 5.1.2.3: *Network sensing using wasted tokens*

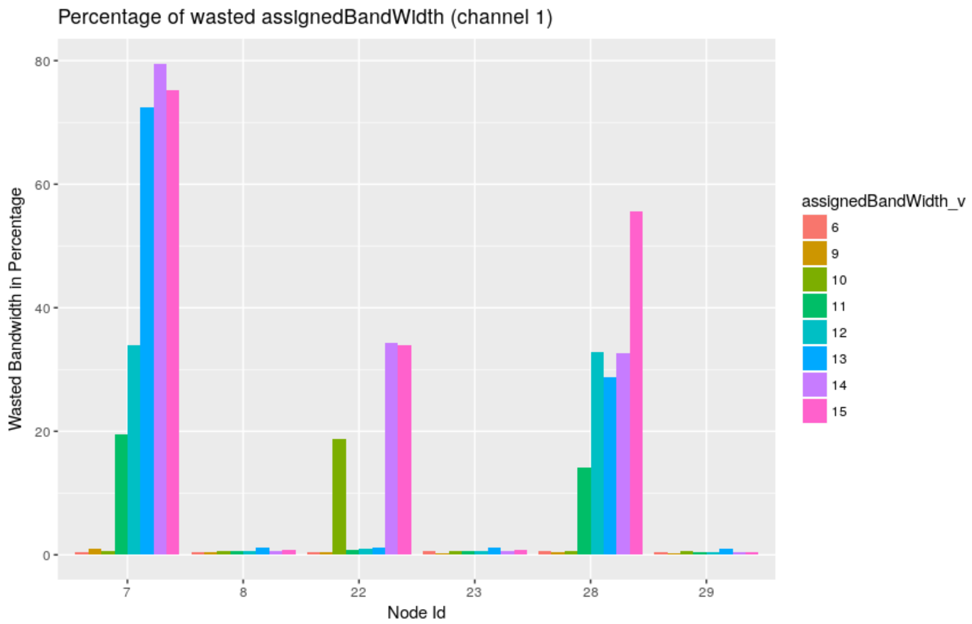
The experiments conducted in two different conditions, in the presence and absence of external APs. Experiments on the static testbed were conducted in the presence of other APs. For the experiment in the absence of external interference, the mobile testbed was used. On the static testbed in presence of external traffic, the experiments conducted on two different channels. The motivation was to observe the results depending on external traffic and transmission frequency. Next, discusses the results from experiments.

First, the wasted bandwidth by each node depending on the total network bandwidth is measured. Initially, the nodes were assigned with a base bandwidth of 6%, i.e the six nodes produce total network load of 36%. Later, the bandwidth of each node increases to 9%, 10%, 11%, 12%, 13%, 14%, 15% which results in total network load of 54%, 60%, 66%, 72%, 78%, 84%, 90%. i.e for an experiment in the presence of external traffic, bandwidth profiles *P.5.1.e*, *P.5.1.f*, *P.5.1.g*, *P.5.1.h*, *P.5.1.i*, *P.5.1.j*, *P.5.1.k*, and *P.5.1.l* were used.

Fig 5.3a and Fig 5.3b are the results of the experiment in the static testbed on channel 6 and 1, respectively. Fig 5.3c shows results in the absence of external traffic. As you can see in the Fig 5.3a that the node 22 starts to lose 19% of the assigned base bandwidth for a network load of 60%. This gets worse as the network load increases further. Next, the node 8 started to lose 5% of the assigned base bandwidth for a network load of 66% and the further load increases, the more bandwidth is lost. Altogether, the increased network load started to initially affect the nodes 8 and 22. But on channel 1 as seen in the Fig 5.3b, the increased network load affects mainly node 7, followed by nodes 28 and 23.

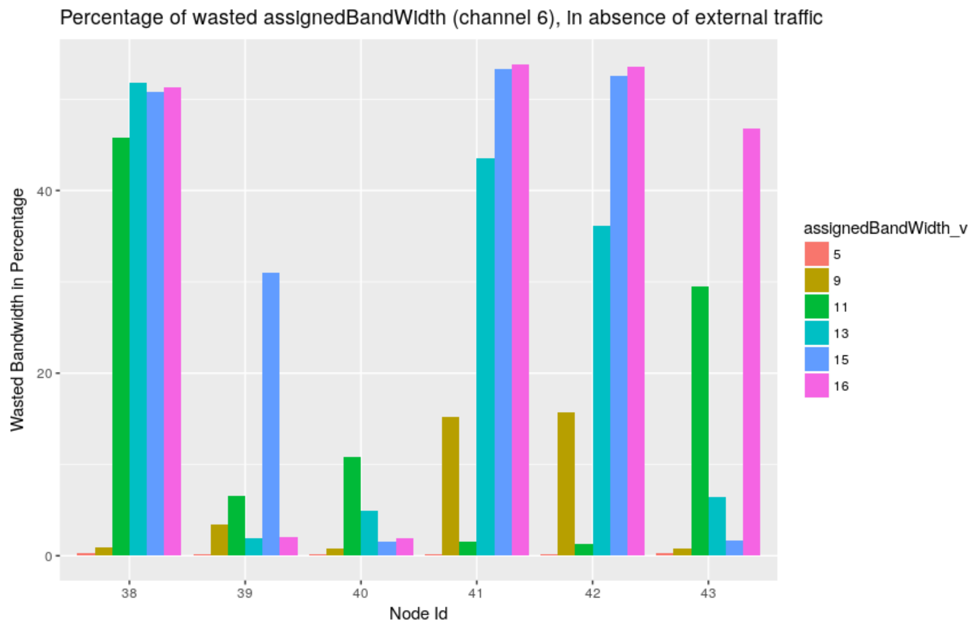


(a) Experiment on the static testbed running in channel 6



(b) Experiment on the static testbed running in channel 1

Figure 5.3.: Wasted assigned bandwidth for nodes in three different experiments

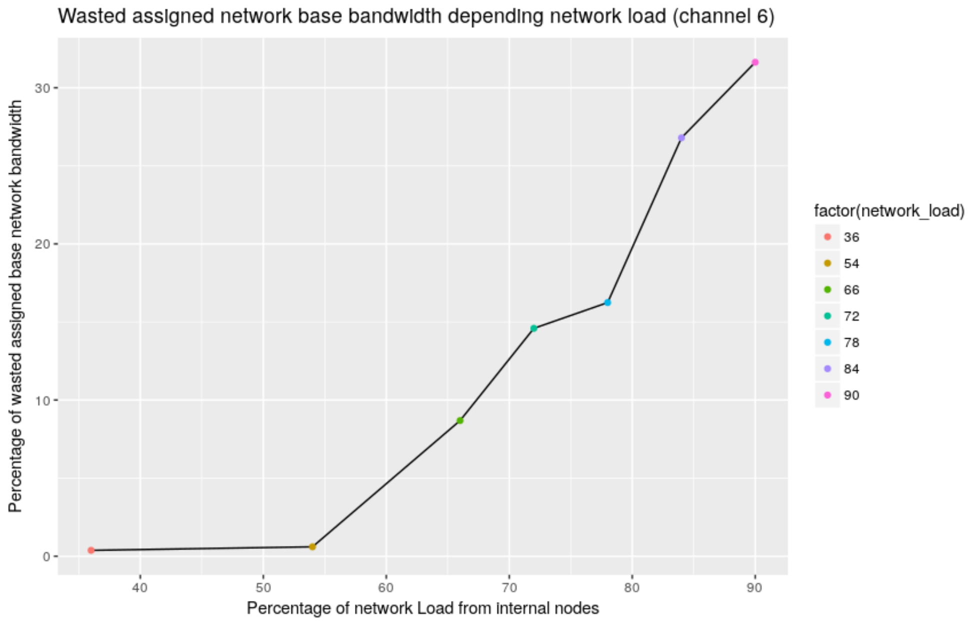


(c) Experiment conducted in the absence of external traffic

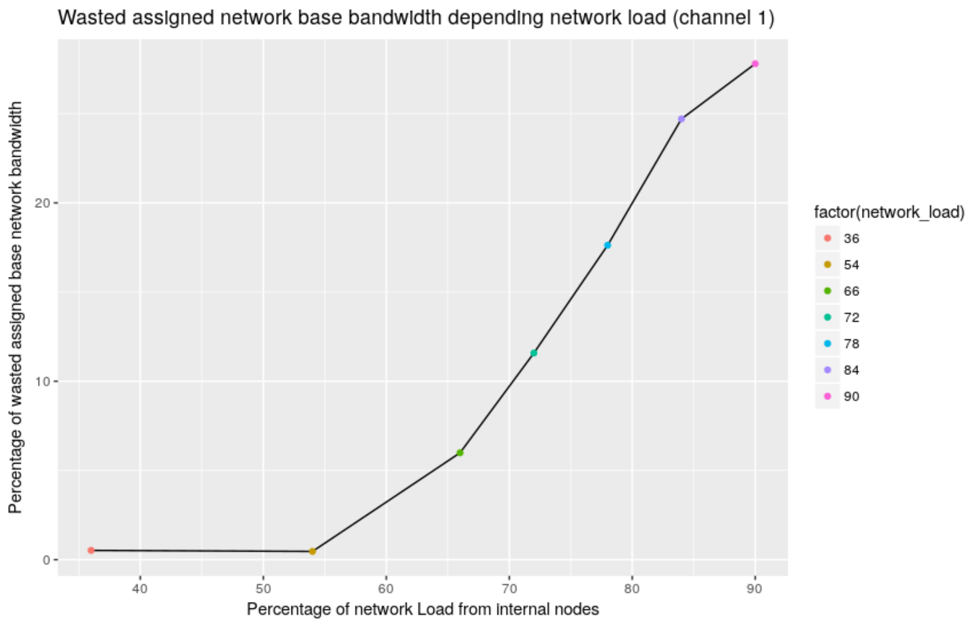
Figure 5.3.: Wasted assigned bandwidth for nodes in three different experiments

As stated before, experiments were conducted in different channels. The reason behind the results can be either due to the change in the link qualities between the node depending on the channel frequency or due to the impact of external APs. The next experiment analyzes the impact of internal frames on wasted bandwidth. Apart from that, the WLAN AP of our group was operating in channel 1 and located near to the node 7. Indeed, as you can see that node 7 lost a high percentage of bandwidth compared to other nodes. While nodes 22 and 28 are topologically close to this AP (see Fig 2.1). It is also found that in channel 6, APs from other research group located nearby node 22 and 8. This might be one reason for token loss on those particular nodes. This affects the medium energy observed and contention faced by each node. Since AP management frames are short, the medium energy sensed by the node will be less. Even though the observed energy is low, it affects contention depending on their number and interval length. It could be also due to the change in link qualities depending on the channel selected. In that case, the traffic due to internal frames results in wasted tokens. This impact will be analyzed in coming experiments.

Fig 5.3c shows the result of the experiment conducted in absence of external traffic. For the experiment, bandwidth profiles *P.5.1.e*, *P.5.1.f*, *P.5.1.h*, *P.5.1.j*, *P.5.1.l*, and *P.5.1.m* were used. Although node 38 lost most of the assigned base bandwidth, at different runs, different nodes lost most of the assigned bandwidth. For example, with assigned base bandwidth of 9%, nodes 41 and 42 lost most of the assigned bandwidth while in the run with 10% node 38 and 43 lost the most. Node 39 lost most of its assigned base bandwidth for the run with assigned base bandwidth 15% than 16%. This is the case for other nodes too. It shows that since there is no external traffic and nodes are very close to each other, the increased network load affects the nodes randomly which depend on medium contention and random backoff. Also, the maximum percentage of wasted bandwidth for a node observed is around 50% (for total network load of 96%), while nodes in presence of external traffic lost around 80% (for total network load of 84%) of the assigned due to external traffic.

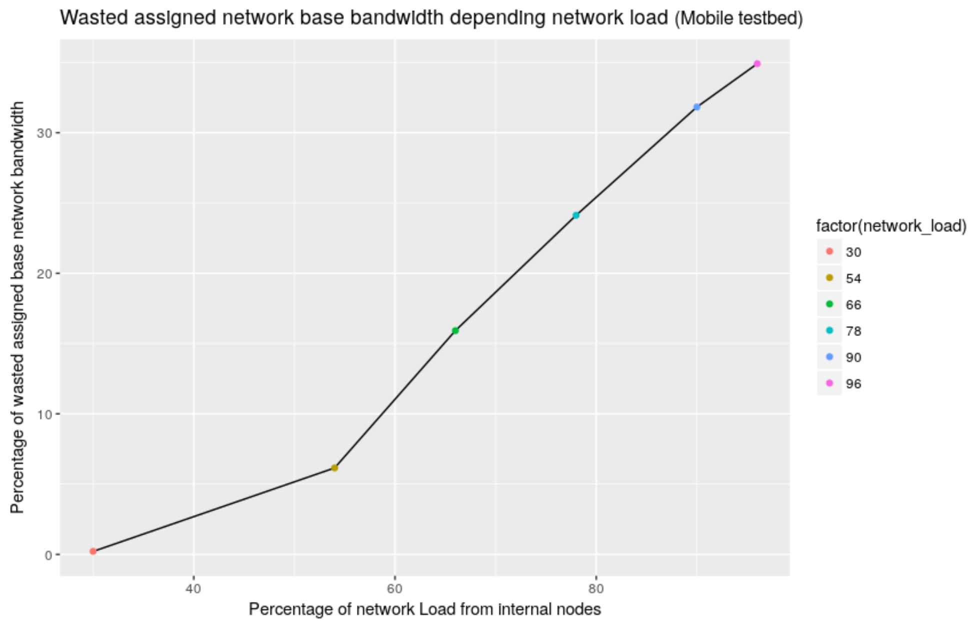


(a) Experiment on the static testbed running in channel 6



(b) Experiment on the static testbed running in channel 1

Figure 5.4.: Wasted assigned bandwidth for the network in three different experiments



(c) Experiment conducted in the absence of external traffic

Figure 5.4.: Wasted assigned bandwidth for the network in three different experiments

Next, the results of analyzing the relation between the $wastedBandwidthPer^t_G$ and $assignedBaseBw_G$ (network load) are discussed. Fig 5.4 shows the result of the three different experiments. It shows that network load above a certain threshold results in bandwidth wastage. This implies that the frames will get delayed or dropped due to medium contention. As you can see from the result, the network starts loose to tokens above 50% of total network load. A similar threshold is observed for both experiments in the presence and absence of external traffic. Fig 5.4a and Fig 5.4b shows the result $wastedBandwidthPer^t_G$ depending on network load for the experiments Fig 5.3a and Fig 5.3b, respectively. As you see in Fig 5.4a and Fig 5.4b that in both experiments on different channel in the presence of external traffic, nodes lost assigned bandwidth in similar relation depending on the network load. But Fig 5.3a and Fig 5.3b shows that in those experiments different nodes get affected while the overall impact on the network remains same.

These results show that the number of *wastedTokens* by a node gives a passive overview about the network contention in the nodes sensing range. If the node started to loose tokens, it shows that node has assigned more bandwidth than the available bandwidth.

Experiment 5.1.2.4: *Sensing medium congestion at low and high node base bandwidth profiles*

In this experiment, the bandwidth utilization at lower and higher bandwidth profiles are analyzed. Fig 5.5 and Fig A.3 show the results of lower and higher bandwidth profiles, respectively. First, the results of the experiments at low bandwidth, then the results of the experiment at high bandwidth are discussed. Later, we conclude our findings.

With lower bandwidth profiles, the probability to loose tokens due to competition between the internal nodes are low. Table 5.3 shows the result of the experiment conducted in the absence of external traffic. The nodes lost zero tokens with bandwidth profile (*P.5.1.b*) and network load of 6%. So, if the nodes loose tokens with same or less base bandwidth profile in the presence of external traffic, it could be due to the contention with external traffic.

src	$wastedTokens_{src}(t)$	token_size (μs)	$usedBwPer_{src}(t)$	$channelBusyTimePer_{src}(t)$	$mediumEnergySensedPer_{src}(t)$	$wastedBwPer_v(t)$
38	0	1	0.90	5.70	4.79	0.00
39	0	1	0.92	7.80	6.88	0.00
40	0	1	0.93	8.14	7.21	0.00
41	0	1	0.92	5.80	4.87	0.00
42	0	1	0.93	5.74	4.81	0.00
43	0	1	0.93	6.36	5.44	0.00

Table 5.3.: Experiment with 1% node base bandwidth and network load of 6% in the absence of external traffic (bandwidth profile *P.5.1.a*)

src	$wastedTokens_{src}(t)$	token_size (μs)	$usedBwPer_{src}(t)$	$channelBusyTimePer_{src}(t)$	$mediumEnergySensedPer_{src}(t)$	$wastedBwPer_v(t)$
7	293358	1	0.98	24.80	23.83	0.89
8	232686	1	0.98	19.87	18.89	0.71
22	13686	1	0.99	16.90	15.91	0.04
23	249952	1	0.98	24.57	23.59	0.76
28	42648	1	0.98	17.90	16.92	0.13
29	11152	1	0.99	14.32	13.32	0.03

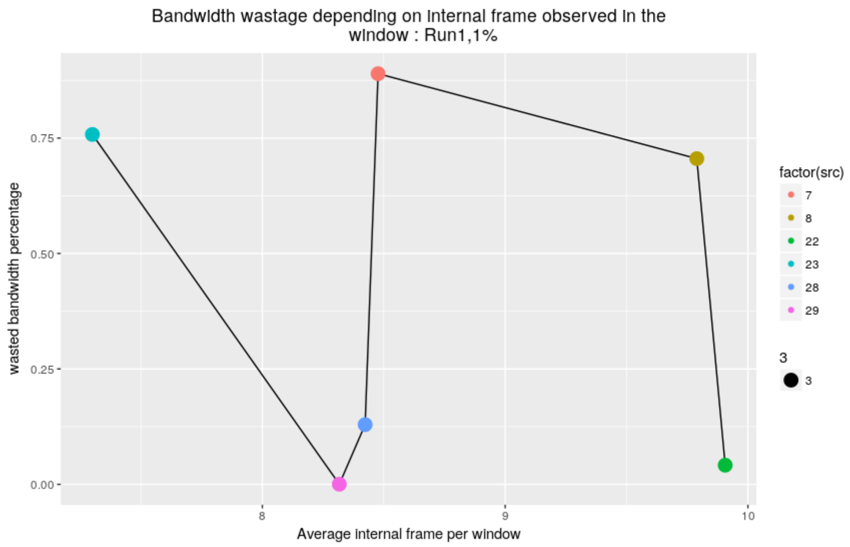
Table 5.4.: Run 1 with 1% assigned Base bandwidth and 6% network load in the presence of external traffic (bandwidth profile *P.5.1.b*)

src	$wastedTokens_{src}(t)$	token_size (μs)	$usedBwPer_{src}(t)$	$channelBusyTimePer_{src}(t)$	$mediumEnergySensedPer_{src}(t)$	$wastedBwPer_v(t)$
7	8164	0.50	0.49	14.80	14.31	0.02
8	4726	0.50	0.49	10.25	9.76	0.01
22	0	0.50	0.50	11.86	11.37	0.00
23	8527	0.50	0.49	13.73	13.24	0.03
28	0	0.50	0.50	12.74	12.24	0.00
29	0	0.50	0.50	8.39	7.89	0.00

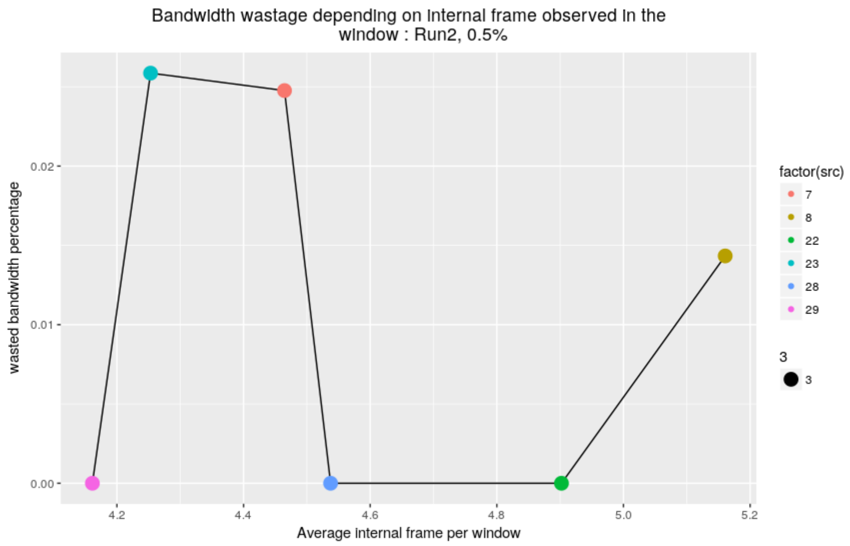
Table 5.5.: Run 2 with .5% assigned Base bandwidth and 3% network load in the presence of external traffic (bandwidth profile *P.5.1.a*)

Fig 5.5 shows the result of an experiment with low bandwidth profiles. Fig 5.5a and Fig 5.5c, and Fig 5.5b and Fig 5.5d show the results of an experiment using bandwidth profile number, *P.5.1.b* and *P.5.1.a*, respectively. As you can observe from Fig 5.5a and Fig 5.5b, the percentage of wasted assigned bandwidth seems independent of the internal frames received. I.e. the medium contention by the internal nodes were independent on internal frames received or due to the presence of internal traffic. But Fig A.3a, Fig A.3b, and Fig A.3e shows that at higher bandwidth profile wasted assigned bandwidth dependents on the number of internal frames received. I.e the medium contention is between the internal nodes.

Table 5.4 and Table 5.5 shows the details of the experiments with low bandwidth profile conducted in the presence of external traffic. The value of the parameter t is the end time of the experiment. While comparing the energy sensed in the medium by each node, the experiments conducted in the presence of external traffic (see Table 5.4) have higher values for *channelBusyTimePer* and *energy_sensed_avg* than in the absence of external traffic (see Table 5.3). Another aspect is the relation between the $wastedBwPer_v$, and *channelBusyTimePer_v* and *mediumEnergySensedPer_v* for each node v. The results show that the nodes which sensed high medium energy lost the higher amount of tokens comparing to nodes which sensed less amount of energy, i.e. the higher the medium energy, the higher the medium contention. Even though nodes lost at most .90% of the assigned bandwidth, the results show that the concept of *wasted tokens* can be used to identify medium contention.

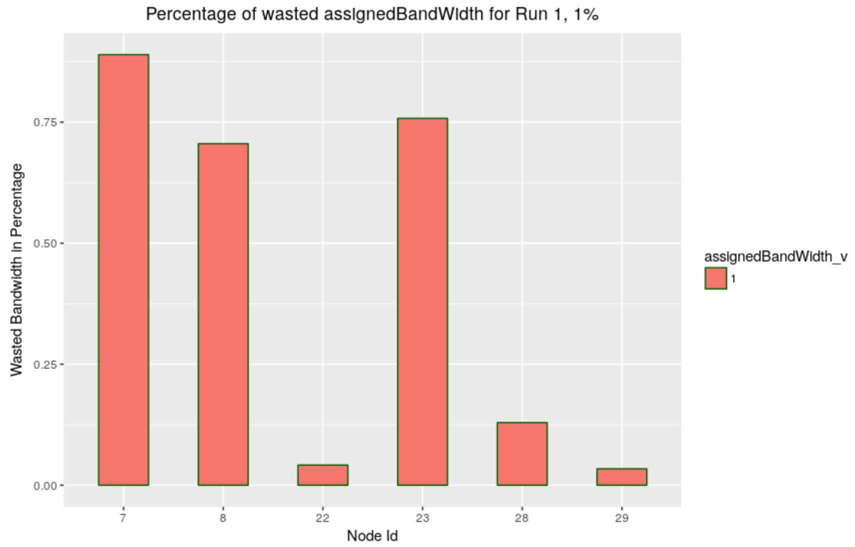


(a) Wasted assigned bandwidth for nodes depending on the internal frames received during experiment with bandwidth profile number *P.5.1.b*

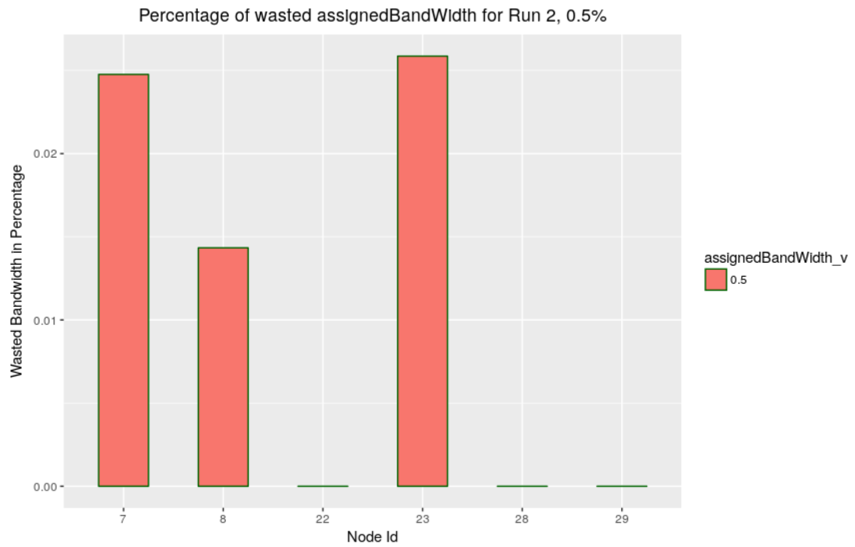


(b) Wasted assigned bandwidth for nodes depending on the internal frames received during experiment with bandwidth profile number *P.5.1.a*

Figure 5.5.: Wasted assigned bandwidth for the network depending on the external traffic



(c) Wasted assigned bandwidth for nodes during experiment with bandwidth profile number *P.5.1.b*



(d) Wasted assigned bandwidth for nodes during experiment with bandwidth profile number *P.5.1.a*

Figure 5.5.: Wasted assigned bandwidth for the network depending on the external traffic

Fig A.3, Table A.1, Table A.2 and Table A.3 shows the result of higher node base bandwidth profiles. The value of the parameter t is the end time of the experiment. The results again show that the nodes that received higher amount internal frames lost high amount of assigned base bandwidth. Similarly, those node's also have a higher value for *mediumEnergySensedPer*. This shows that medium contention was due to the high amount of internal traffic, i.e. $assignedBaseBw_G \geq usableBaseBw_G$. This condition can be controlled by reducing $assignedBaseBw_G$ to the network. By analyzing the internal frames received by the nodes in the network, it is possible to identify the medium congestion due to internal nodes, which is controllable.

It is important to identify whether the tokens are lost due to contention between internal nodes or external nodes. In the case of token loss due to external, either reduce the $assignedBaseBw_G$ by the reducing $assignedBaseBw_v$ of the respective node. Another possibility is to increase the transmission rate to reduce the $maxTxBw_v$, if it is possible to achieve a tolerable frame loss ratio at higher rates. In the case of token loss due to internal traffic, identify the node causing the contention, and control its traffic.

Experiment 5.1.2.3: *Frame loss depending on offered network load*
In this experiment, frame loss depending on the network load is analyzed. Fig A.4 and Fig A.5 show the results of experiments in absence and presence of external traffic, respectively. First, the results of the experiments in absence of external traffic, then the results of the experiment in presence of external traffic are discussed. Later, we conclude our findings.

Fig A.4 shows the result of the experiment in the absence of external traffic. In the figure, $assingedBaseBw_v$ denotes the node bandwidth. For example, an experiment run with $assingedBaseBw_v = 9\%$ generated a network load of 9 % · 6 = 54%. As mentioned earlier, the experiment consists of 6 nodes. Looking into the results, the link with worst frame loss ratio or the worst link is 40 $\leftrightarrow$ 41 with frame loss ratio of 14% for a network load of 96% (Note: not the $usedBwPer_v^t$). As you can observe, some links show strict pattern for frame loss depending on the network load while other does not, i.e. the frame loss ratio depends on the network load. The higher the network load, the higher the frame loss ratio and vice versa. Consider the links with node ID 38, 41, 42 or 43 as the source shows this pattern, but considering the links with node 39 or 40 as the source does not show this pattern. Fig A.4g shows the wasted bandwidth for the experiment run. The result shows that the node 38 and 40 lost the least amount of assigned bandwidth, which implies the nodes 39 and 40 are topological in a good position to have strong links and to win the medium contention.

Fig 5.3c shows the wasted bandwidth for the same nodes with similar topology with bandwidth profiles during a different experiment run. These results show that node 40 lost the least amount of tokens at different runs followed by nodes 39 and 43. The results show that the nodes which lost the least amount of tokens have less medium contention are in a good topological position to win the contention, which results in strong links. While the nodes that lost the highest amount of tokens which have the most medium contention are in a bad topological position, resulting in the weak link show this pattern.

Fig A.5 shows the result in the presence of external traffic. In these results, the pattern observed previously is rare to find. Also, the frame loss ratio for certain links are relatively higher compared to the results of the experiment conducted in absence of external traffic. For certain links, the link quality is very weak (about 40% to 50% frame loss ratio). But certain links have much better frame loss ratio at lower network load. For example, consider the link between nodes

28 $\leftrightarrow$ 29. In the Experiment 5.1.1.1, which use a random interval between frames and random rate for each frame show relatively no link between nodes 28 $\leftrightarrow$ 29. But in this experiment, Fig A.5e, and Fig A.5f shows that the link 28 $\leftrightarrow$ 29 has better quality at lower network load. Its link quality gets worse when the network load is more than 12%. Links 28 $\rightarrow$ 22, 28 $\rightarrow$ 8, 29 $\rightarrow$ 8, 29 $\rightarrow$ 22, 7 $\rightarrow$ 8, 7 $\rightarrow$ 22, and 7 $\rightarrow$ 23 show relatively higher value for $lossRatio_{v,v'}$ at higher network loads. Apart from that, the link 23 $\leftrightarrow$ 28 observed a higher frame loss ratio for low network loads (for 3% and 6%). Analyzing closely, the link observed a continuous frame loss ($maxContinuousLoss_{v,v'}$) of 457 frames out of transmitted 7128 frames in one direction. Continuous frame loss occurred in other direction too, and more than once. This can be due to some device operating on the same channel which does not perform channel sensing to access the medium, like a microwave oven.

Overall, considering the results from both experiments, sticking to lower node base bandwidth or network load can improve the link quality for most weak links. Strong links always show better result irrespective of the network load. This might be due to their topological position. Finding those strong and weak links, and weak links that can perform in lower network loads is an important step towards controlling the network and in routing.

Experiment 5.1.3: *Using SDR/USRP to improve the results*

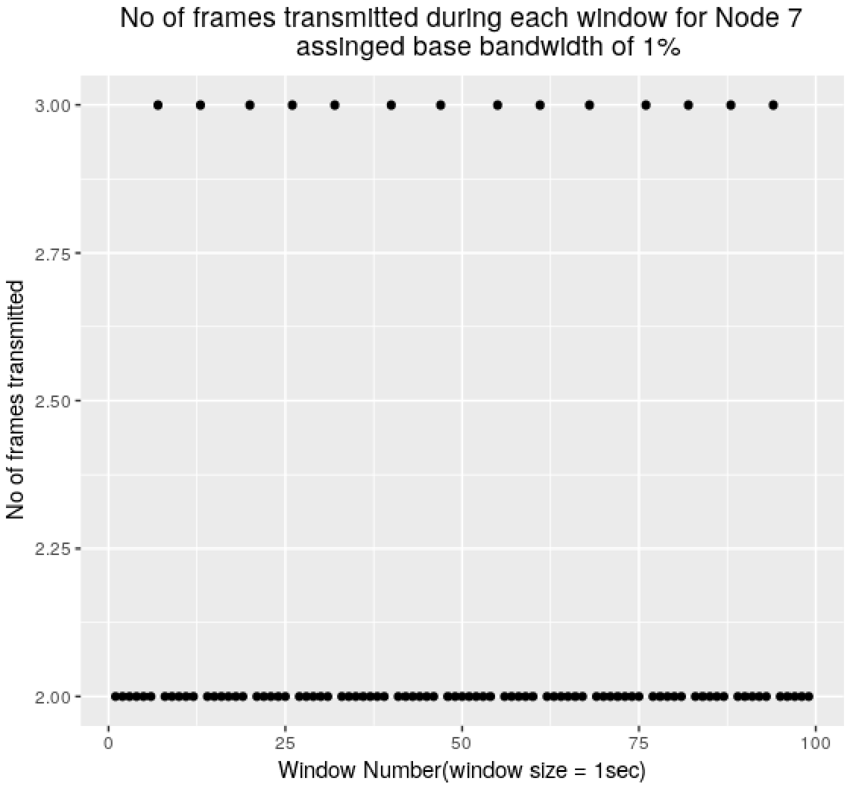
The motivation behind the experiment was to use a USRP to improve the measurement. Due to technical reasons, this experiment shifted to future work. This section explains the work done up to now. Using USRP, it possible to sense medium efficiently, but it is a complex process. The initial idea was to collect the raw samples to analyze the medium energy. The sample was collected with a sample rate of 20 Million samples per second on 2.4 GHz frequency. The data collected during the experiments is so large, so processing within a reasonable amount of time is diffcult. Later, the idea was to use existing open source projects such as [BSSD13] for medium sensing using USRP and GNU Radio [GNU] for IEEE 802.11 frames.

5.2.2 Other Findings

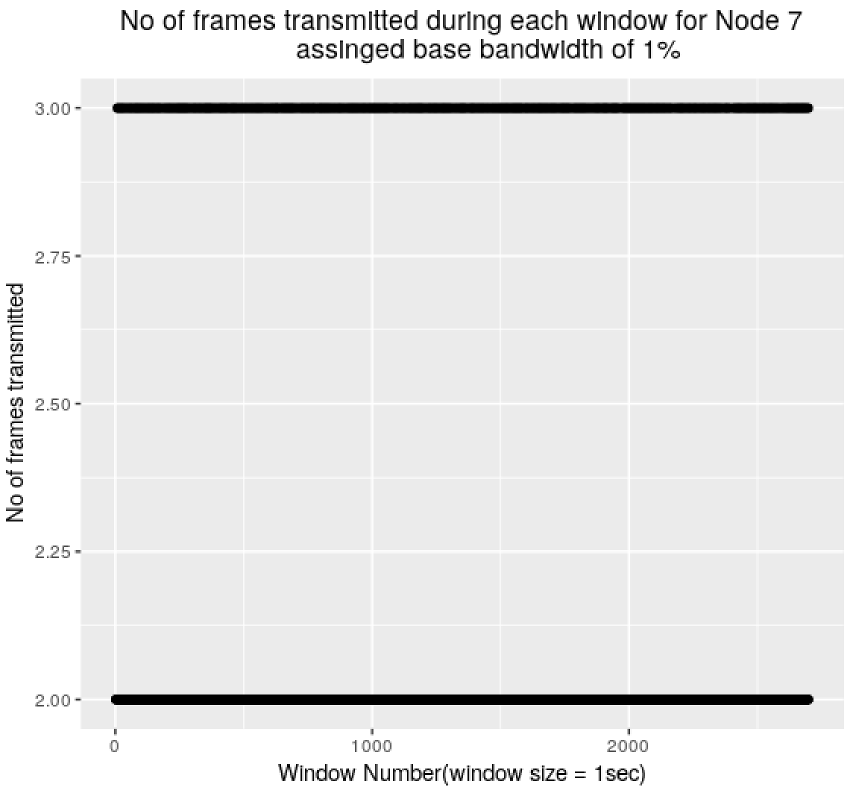
This section discuss findings during the experiments which was not part of the experiment. Throughout the experiment, window length of 1 sec is used. The results discussed below is also using a window length of 1 sec.

First, let us observe the number of internal frames sent $noInteFrameInWindow_v^{ws}(wn)$ during each window. Depending on the bandwidth configuration, the number varies. For lower base bandwidth configurations, less number of frames are sent comparing to higher bandwidth configurations. Fig 5.6 shows the $noInteFrameInWindow_v^{ws}(wn)$ for node 7 for an experiment run with *assignedBaseBw* of 1% and network load of 6%. Fig 5.6a shows the detailed view of a short duration of time. The constantly transmitted 2 or 3 during each window without wasting any bandwidth. Now, let us analyze the plot for higher *assignedBaseBw* configuration.

Fig 5.7 shows the result for a higher *assignedBaseBw* configurations. Figure shows the result for nodes 7, 28 and 8 for an experiment run with *assignedBaseBw* of 13% and network load of 78%. In this experiment run, node 7 lost the highest amount of token followed by node 28. Node 8 has a value for used bandwidth percentage, $usedBwPer_v^t$ = 12.41% of the 13% *assignedBaseBw*.

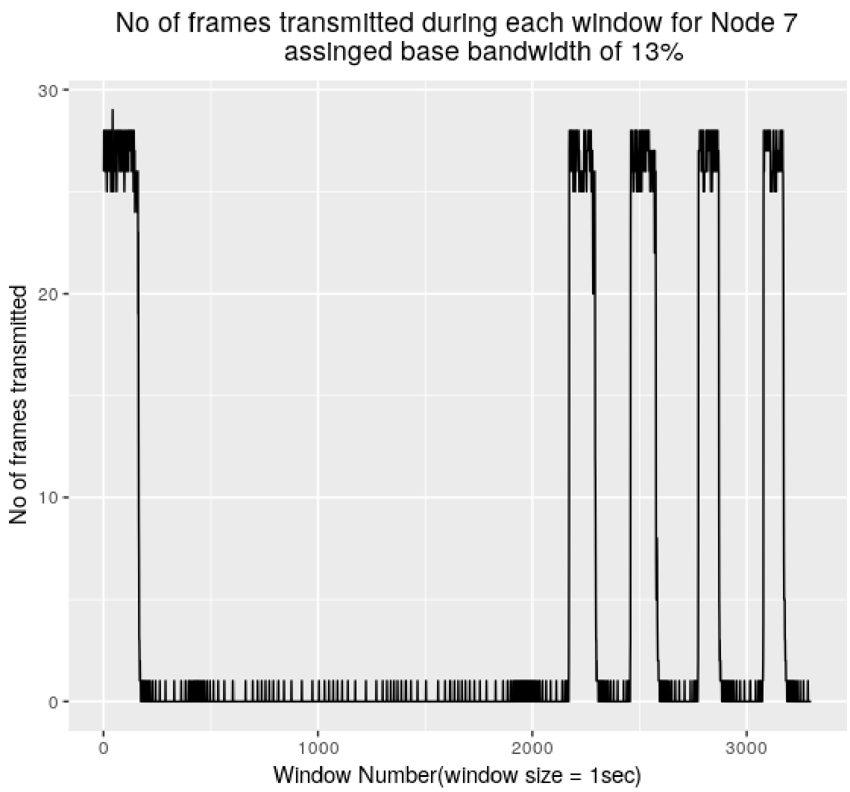


(a) For windows wn from 0 - 1000 of node 7, graph showing relation between $noInteFrameInWindow_v^{ws}(wn)$ in each wn. Experiment conducted with bandwidth profile *P.5.1.b*

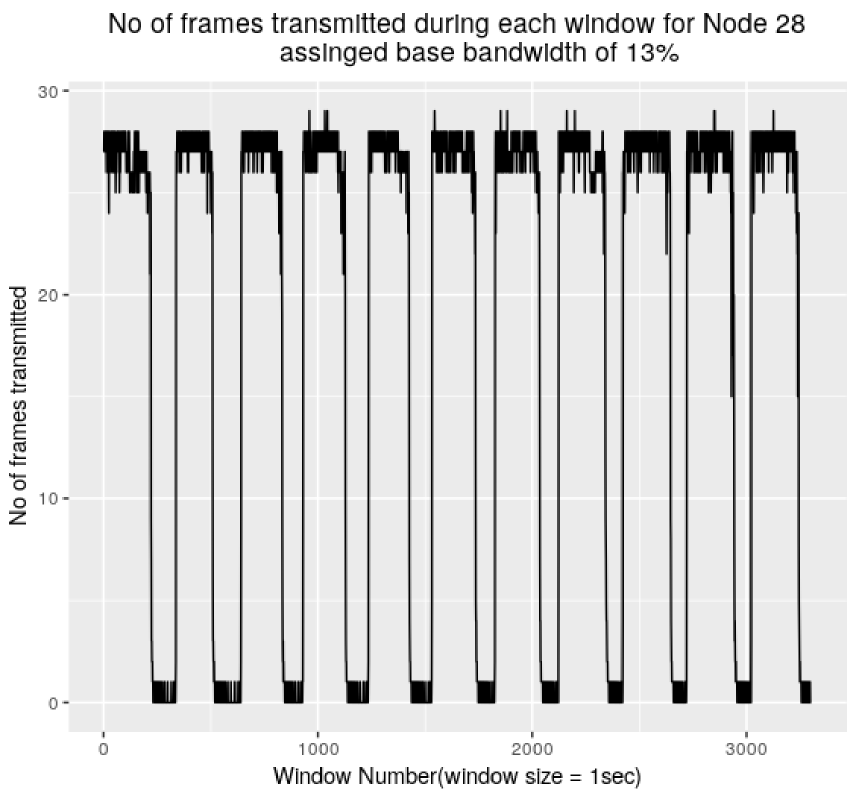


(b) For windows wn from 0 - 2900 of node 7, graph showing relation between $noInteFrameInWindow_v^{ws}(wn)$ in each wn. Experiment conducted with bandwidth profile *P.5.1.b*

Figure 5.6.: Number of frames transmitted by node 7 in each window for the bandwidth profile *p.5.1.a*

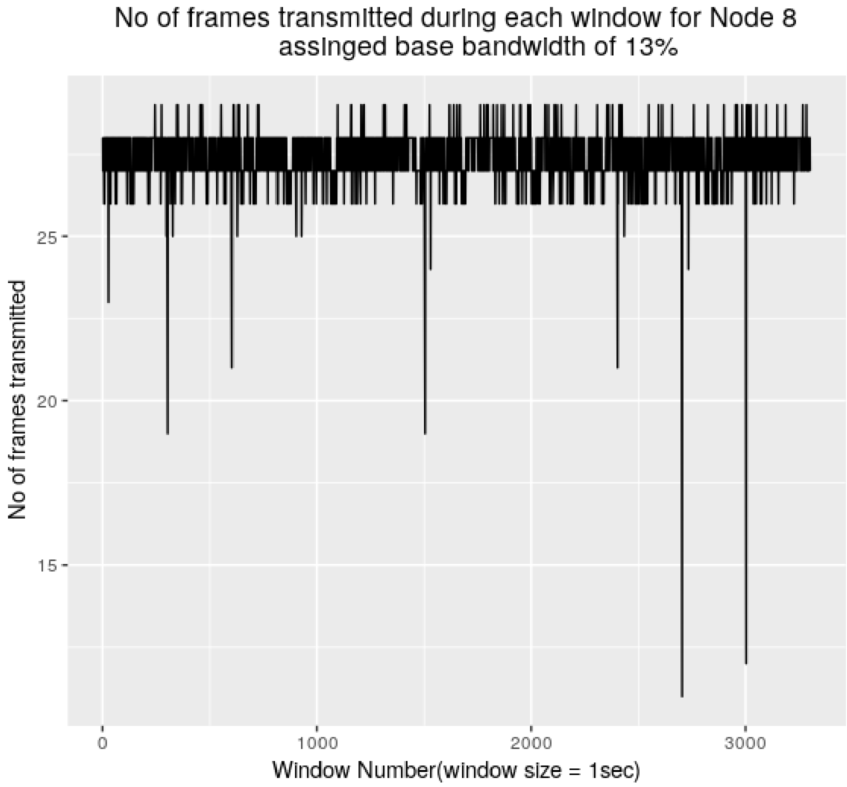


(a) For windows wn from 0 - 3500 of node 7, graph showing relation between $noInteFrameInWindow_v^{ws}(wn)$ in each wn. Experiment conducted with bandwidth profile *P.5.1.j*



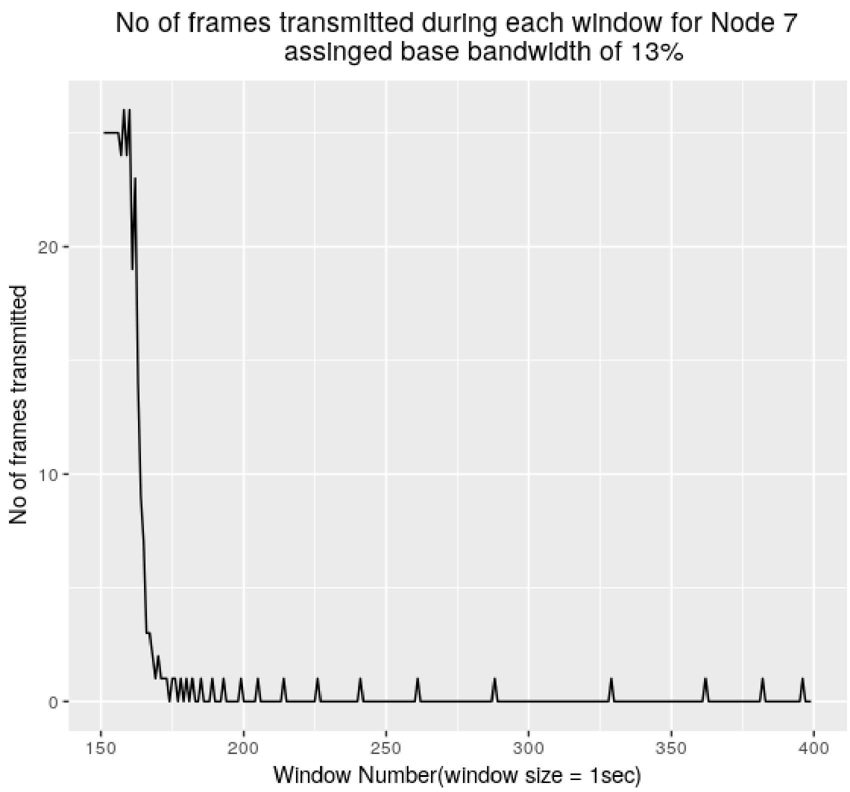
(b) For windows wn from 0 - 3500 of node 28, graph showing relation between $noInteFrameInWindow_v^{ws}(wn)$ in each wn. Experiment conducted with bandwidth profile *P.5.1.j*

Figure 5.7.: Number of frames transmitted by nodes 7 and 28 in each window for the bandwidth profile *p.5.1.j*

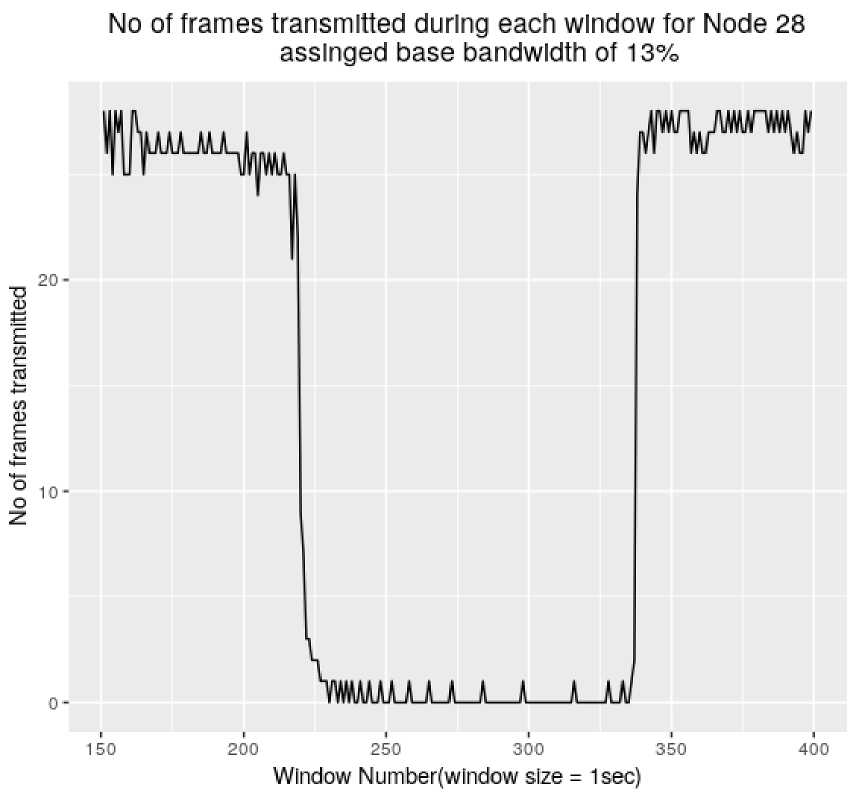


(c) For windows wn from 0 - 3500 of node 8, graph showing relation between $noInteFrameInWindow_v^{ws}(wn)$ in each wn. Experiment conducted with bandwidth profile *P.5.1.j*

Figure 5.7.: Number of frames transmitted by node 8 in each window for the bandwidth profile *p.5.1.j*



(a) For windows wn from 150 - 400 of node 7, graph showing relation between $noInteFrameInWindow_v^{ws}(wn)$ in each wn. Experiment conducted with bandwidth profile *P.5.1.j*



(b) For windows wn from 150 - 400 of node 28, graph showing relation between $noInteFrameInWindow_v^{ws}(wn)$ in each wn. Experiment conducted with bandwidth profile *P.5.1.j*

Figure 5.8.: Closer look (windows where contention occurred) at number of frames transmitted by node 7 and 28 in each window for the bandwidth profile *p.5.1.j*

As you can see in the Fig 5.7a node 7 lost the medium for large amount time or high amount $wastedBwInWindowPer_v^{ws}(n)$ between window 150 and 2200.

Fig 5.8a shows closer look into window regions where node 7 lost the medium. As you can observe, the node sends one frame then delays the next transmission for a certain time. This time increases from time to time. I.e. node sends one frame, delays the transmission for one second, then sends the next frame, delays the transmission for one second, then sends the next frame, delays the transmission for two second and so on. Similarly, Fig 5.8b shows a closer look into the window regions where node 28 lost and retain the medium. Node 28 also shows a similar pattern in frame transmission once it lost contention. The delay between frame transmission also increases in a similar pattern. But node 28 retains the medium after some period. During other experiments, similar patterns were observed for nodes which lost most of the tokens. An experiment was conducted using TP-Link adapters with Atheros chipset, and similar pattern were observed during higher loads. Thus, this seems to be some logic in Linux driver level independent of the hardware.

CHAPTER. 6

Summary

In this chapter, the work of this thesis is summarized along with future works. Section 6.1 discusses the future work to be done on the conceptual side, and experiments to verify further improvements in networking sensing.

6.1 Future Work

In this thesis, the experiment and bandwidth sharing mechanism were experimented in a single-hop network. The first step is to extend the Token Bucket based bandwidth sharing mechanism into a multi-hop network. The medium is shared with the nodes in interference range, so the nodes in a particular area contend for the medium. The concept of bandwidth assigned to the network to be distributed among each hop or cluster in the network. During the experiments, the nodes received a maximum of .35% of the total received frames as corrupted. Even though the percentage of corruptly received frames is very low, adding separate CRC checks for the meta-data inserted by the framework might help to get some valuable information. For examples, a corrupted frame with correct meta-data but corrupted data could be negative acknowledged.

In Experiment 5.1.2.4, the medium congestion at higher network bandwidth depending on the number of internal frames was analyzed, i.e. by monitoring the number of internal frames received with respect to the number of wasted tokens. In this experiment, the node which is responsible for the contention is not analyzed. By finding the node responsible for contention, it possible to control the node. The controlling part is done by bandwidth manager. Bandwidth manager prioritize the sensed information and control the network based on the information obtained from the sensing layer. This experiment along with other experiments mentioned in Section 5.1.2 need to be experimented in multi-hop.

Currently, the experiments have been done using Linux kernel version 4.6.5. Ralink drivers in the previous version of Linux kernel ignored the rate value provided in Radiotap headers. So, it is worth to check updates made in the latest versions for new features. While extending the experiment results to multi-hop network, controlling the transmission power could provide extra control over the network. As mentioned in Section 3.1, increasing transmission power above a certain limit will not improve the loss ratio. Instead, the interference range of node will increase. By controlling the transmission power of each node, it is possible to control the interference range of a node, and save the energy.

While conducting the experiment in multi-hop topology, USRPs give extra and unique information about the medium. Channel Sensing Time observed by each node varies depending on the noise calibration level of each node. This measurement could be improved with the help of USRPs. Using GNU Radio along with USRPs will give a real-time information about the medium energy. As mentioned, there existed some open-source projects with this motivation. It is necessary to understand existing projects and how they could be used in this case.

Finally, the experiment results obtained from this work along with results obtained from further experiments need to be implemented into the centralized bandwidth manager.

6.2 Conclusion

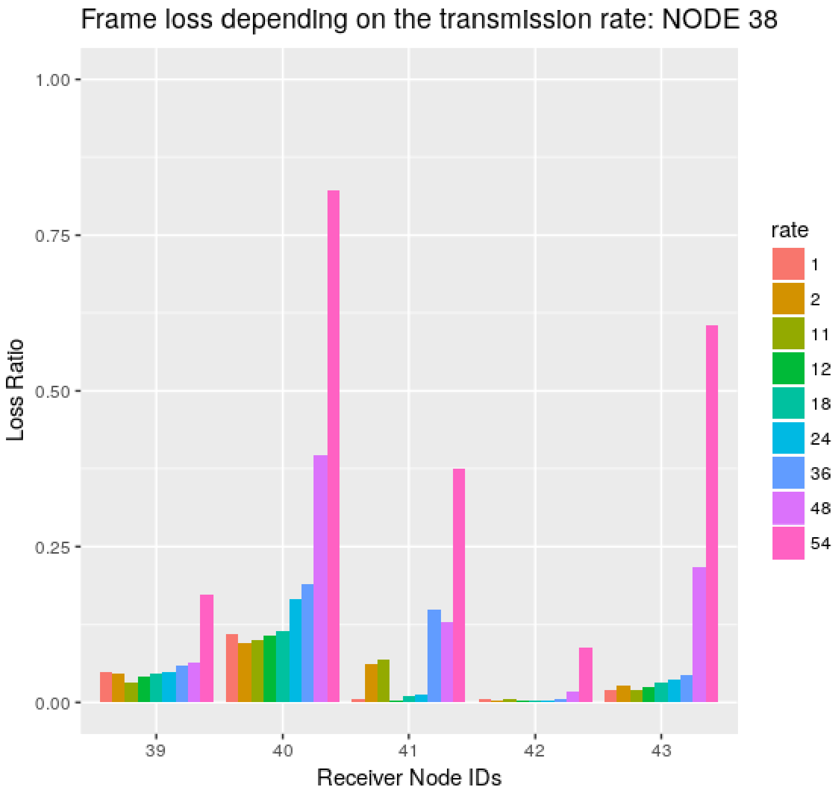
In this section, the thesis work is summarized. The motivation behind the work is to lay the foundation for the network sensing layer used by the bandwidth manager. As part of the thesis, a thorough research about different possibilities to sense the medium without tuning the standard MAC of IEEE 802.11 stack. This thesis work also propose a mechanism to share the medium bandwidth among the nodes in a single hop. The design of token bucket based bandwidth sharing mechanism and controlled refilling technique restricts the occurrences of burst transmission by a node. Using this mechanism, it is also possible to indirectly sense the medium congestion in the form of *wasted tokens*. Nodes that face too much medium contention start losing tokens that are refilled into their local bandwidth bucket. By comparing the number of lost tokens in each node, it is possible to analyze the medium contention of each node.

The proposed concept was tested in two different medium conditions, in the presence and absence of external traffic. In the experiments, quality of parameters such *Channel Busy Time*, *Medium Utilization Time* and number of wasted tokens were analyzed. The experiment results show that *Channel Busy Time* along with number of wasted tokens can be used to sense the medium contention at each node. Experiments were conducted to analyze the frame loss on each link depending on the transmission rate and network load. The results show that it necessary to choose transmission rate for each link depending on link quality. Choosing an proper transmission rate for each link will improve the frame loss and reduce the required bandwidth for a transmission node. Apart from losing tokens at higher network loads, higher network load only affect certain links in the network. These links lose high amount of frames as the network loads increases. But certain links have high link quality even at higher network load. Based on these results, it is possible to categorize the links as strong and weak links, which could be used by the routing algorithm.

Apart from these, certain specific patterns were observed during medium contention, i.e. the way frames in transmission queue get delayed by IEEE 802.11 MAC layer. When the MAC layer senses the medium as busy, it will back off transmission for a long duration of time. Aggregating the results obtained from this work can be used by the bandwidth manager to monitor the medium and to control the network.

Appendix. A

Appendix 1

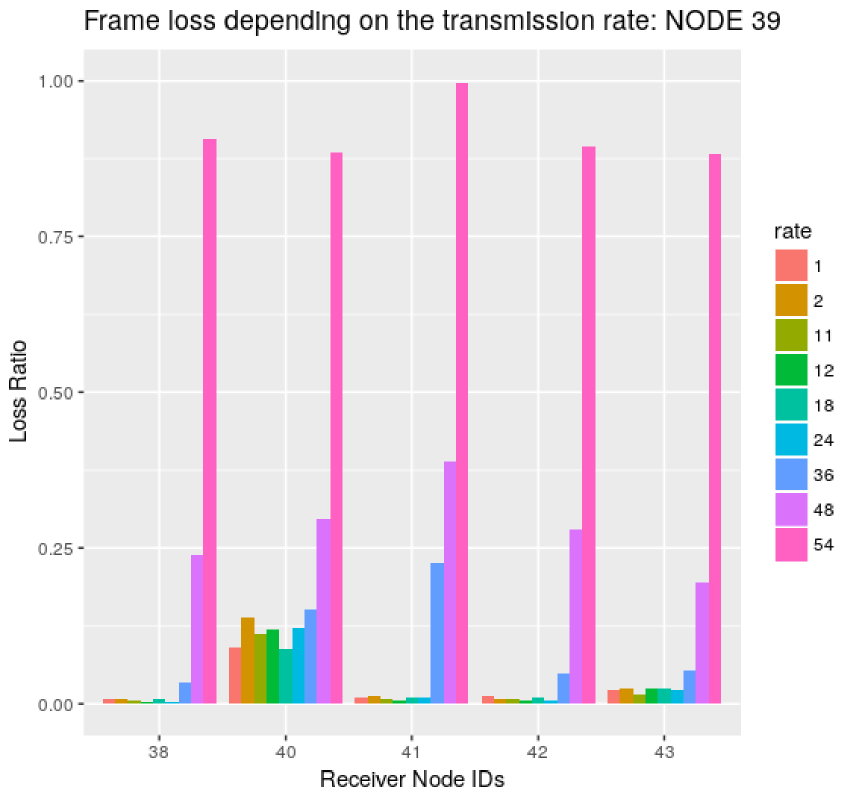


(a) Frame lose ratio ($lossRatio_{v,v'}$) with respect to transmission rate on links with Node 38 as the source

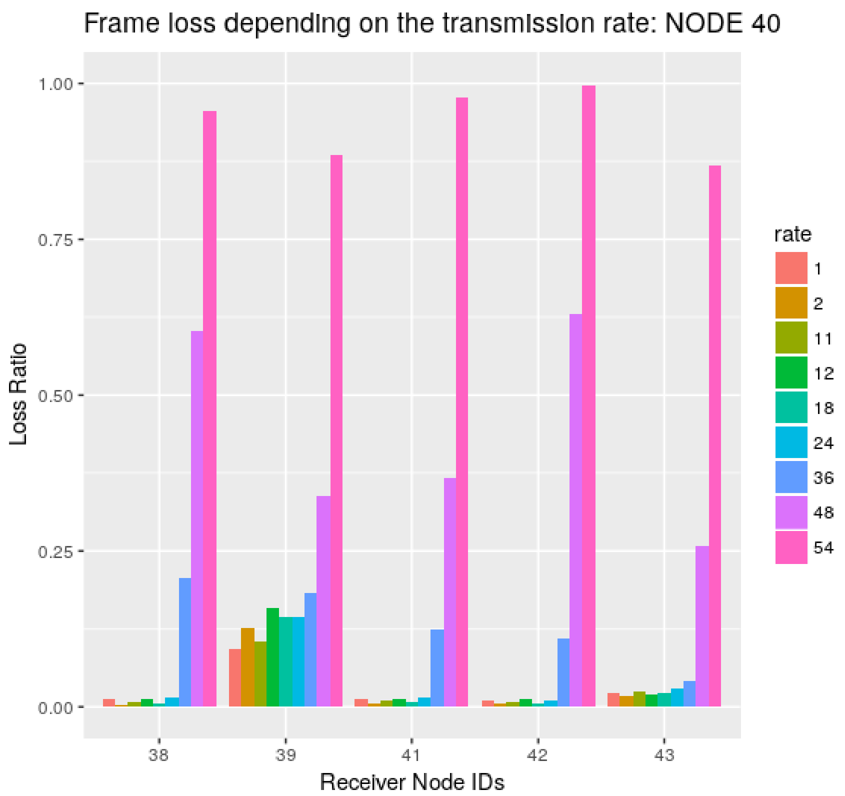
Figure A.1.: Experiment conducted to check the frame loss ratio depending on the transmission rate in the absence of external traffic

src	$wastedTokens_{src}(t)$	token_size (μs)	$usedBwPer_{src}(t)$	$channelBusyTimePer_{src}(t)$	$mediumEnergySensedPer_{src}(t)$	$wastedBwPer_{v}(t)$
38	3295833	13	1.99	41.33	39.34	48.14
39	117072	13	10.84	42.99	32.15	1.82
40	295679	13	10.27	43.19	32.92	4.55
41	2742212	13	3.41	41.46	38.05	40.39
42	2263827	13	4.72	41.49	36.77	33.50
43	388266	13	10.05	42.49	32.44	6.02

Table A.1.: Experiment with 13% node base bandwidth and network load of 78% in the absence of external traffic (bandwidth profile *P.5.1.j*)

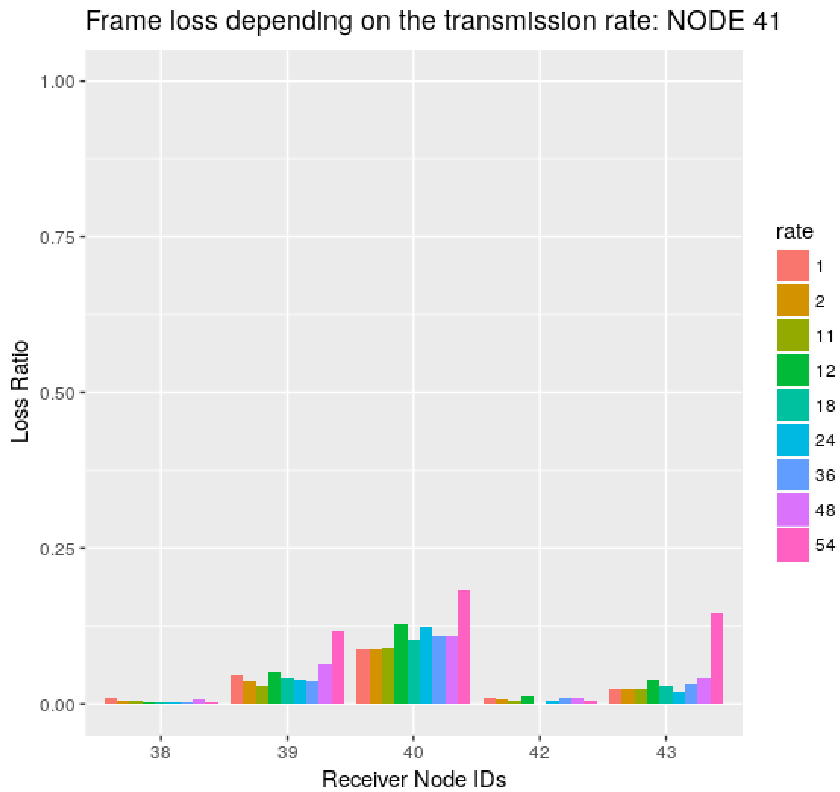


(b) Frame lose ratio ($lossRatio_{v,v'}$) with respect to transmission rate on links with Node 39 as the source

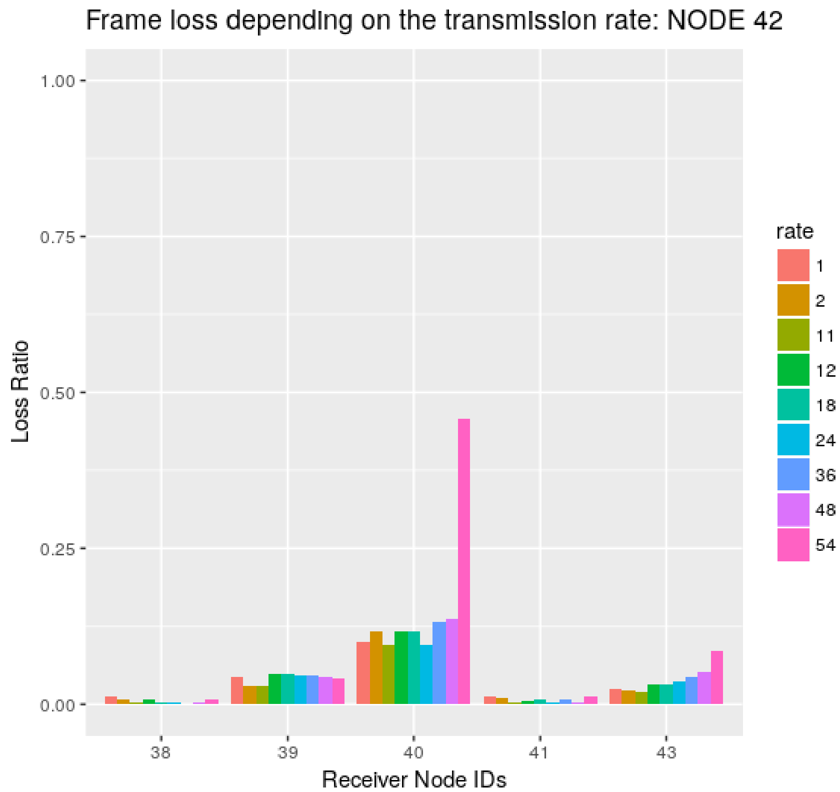


(c) Frame lose ratio ($lossRatio_{v,v'}$) with respect to transmission rate on links with Node 40 as the source

Figure A.1.: Experiment conducted to check the frame loss ratio depending on the transmission rate in the absence of external traffic

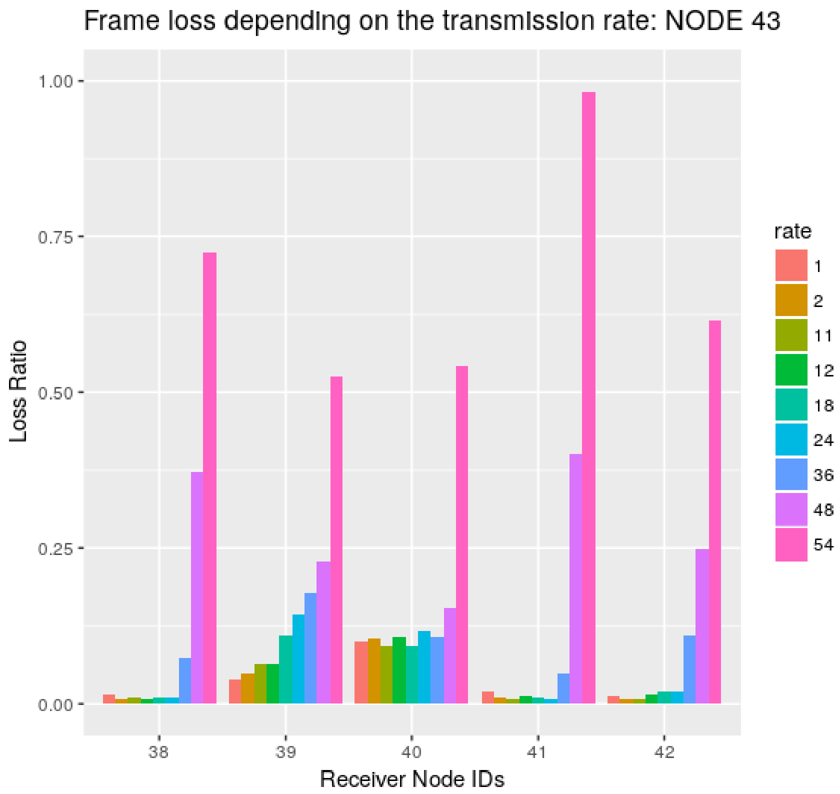


(d) Frame lose ratio ($lossRatio_{v,v'}$) with respect to transmission rate on links with Node 41 as the source

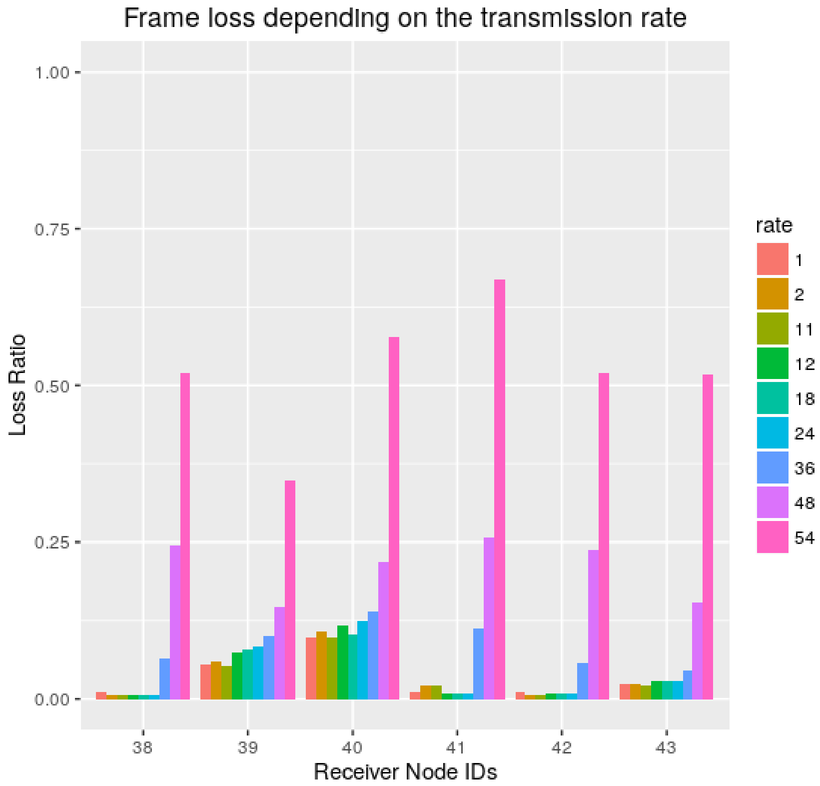


(e) Frame lose ratio ($lossRatio_{v,v'}$) with respect to transmission rate on links with Node 42 as the source

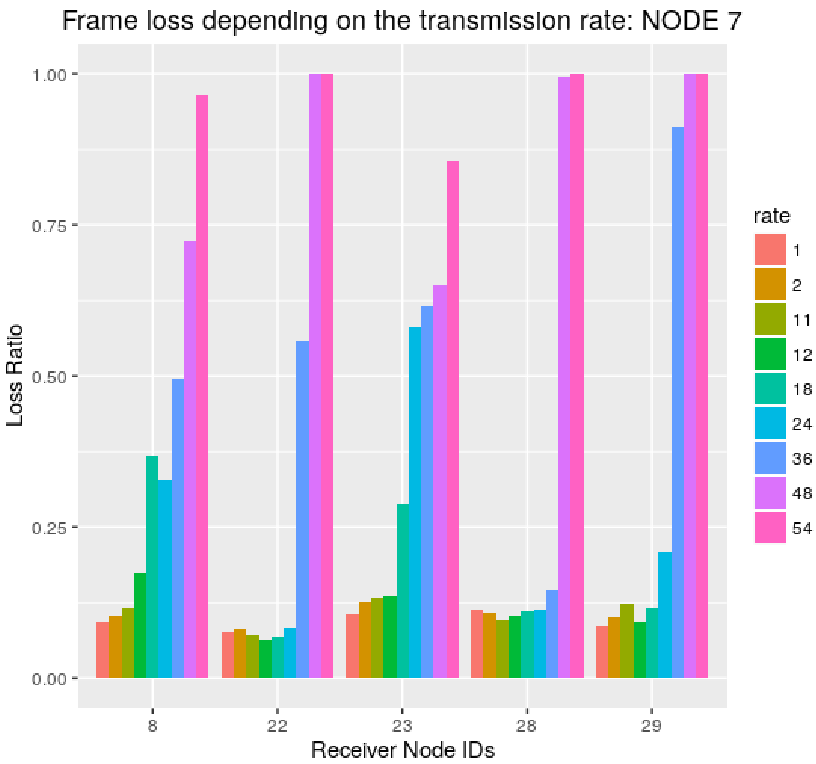
Figure A.1.: Experiment conducted to check the frame loss ratio depending on the transmission rate in the absence of external traffic



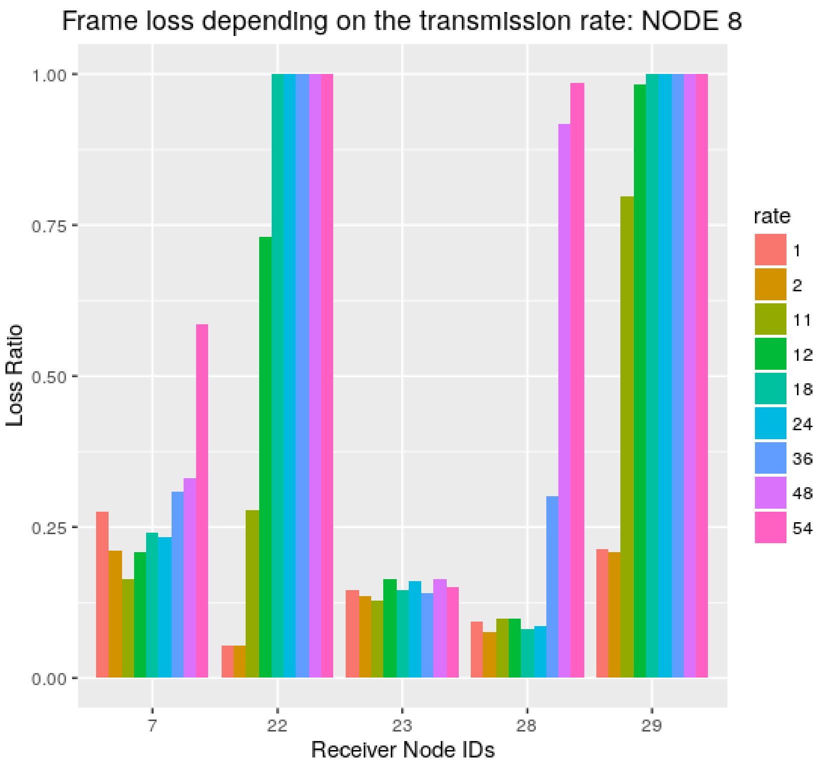
(f) Frame lose ratio ($lossRatio_{v,v'}$) with respect to transmission rate on links with Node 43 as the source



(g) Frame lose ratio ($lossRatio_{v,v'}$) observed in each node as receiver during each experiment

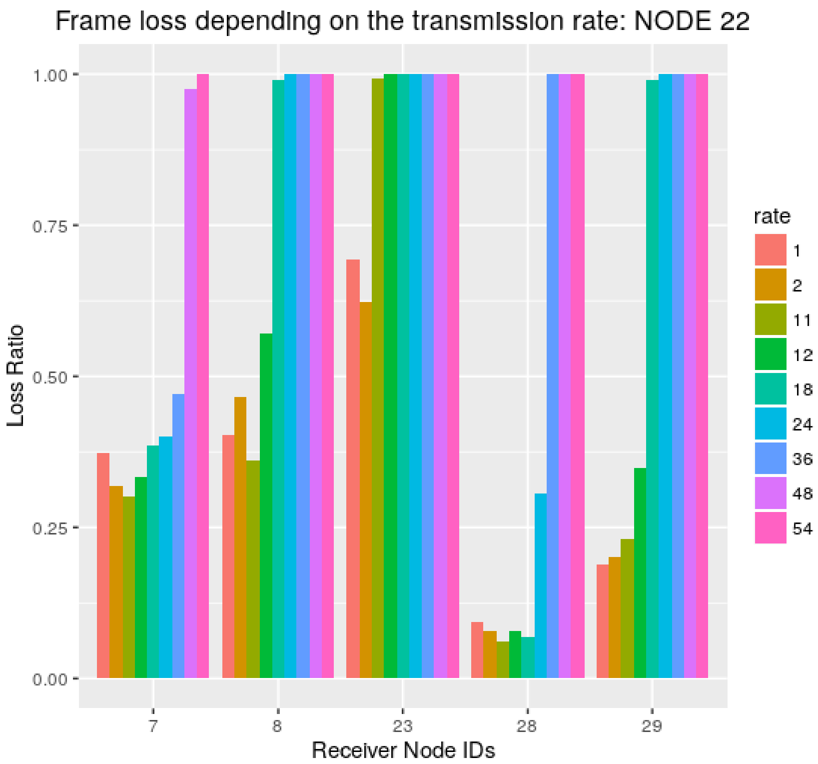


(a) Frame lose ratio ($lossRatio_{v,v'}$) with respect to transmission rate on links with Node 7 as the source

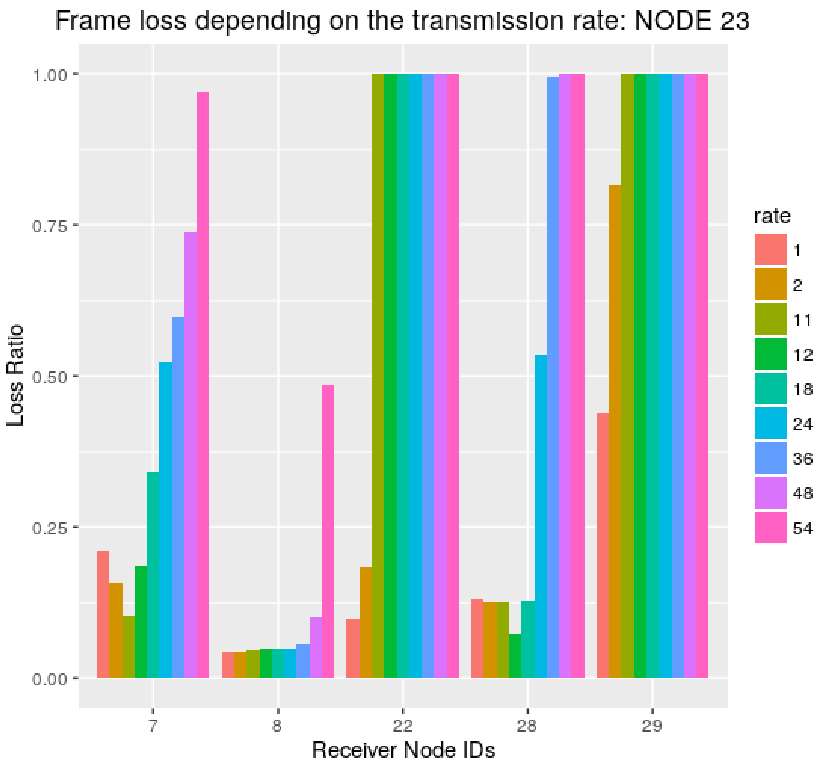


(b) Frame lose ratio ($lossRatio_{v,v'}$) with respect to transmission rate on links with Node 8 as the source

Figure A.2.: Experiment conducted to check the frame loss ratio depending on the transmission rate in the presence of external traffic

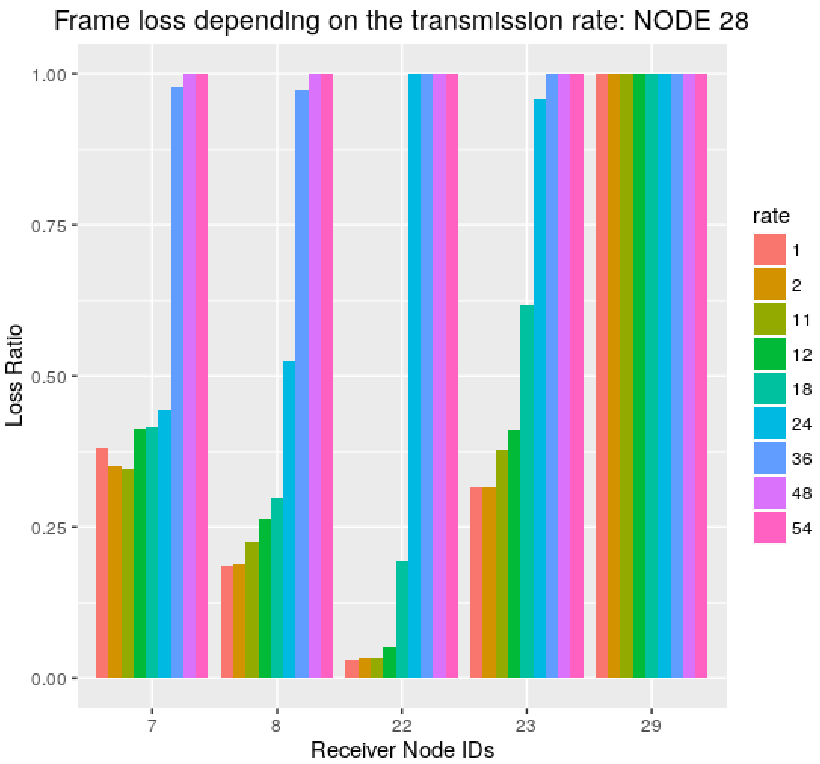


(c) Frame lose ratio ($lossRatio_{v,v'}$) with respect to transmission rate on links with Node 22 as the source

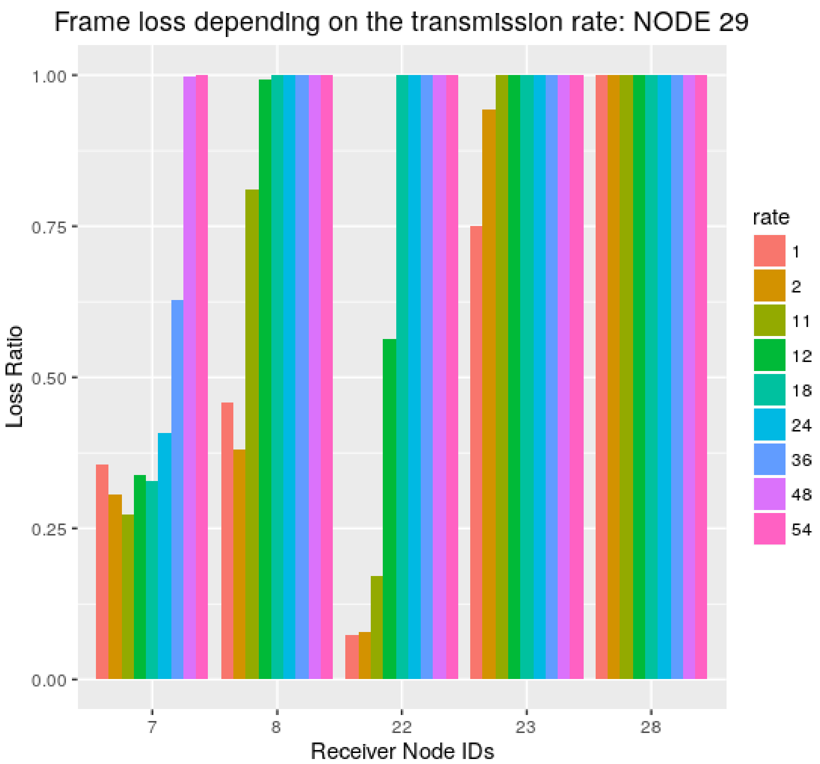


(d) Frame lose ratio ($lossRatio_{v,v'}$) with respect to transmission rate on links with Node 23 as the source

Figure A.2.: Experiment conducted to check the frame loss ratio depending on the transmission rate in the presence of external traffic

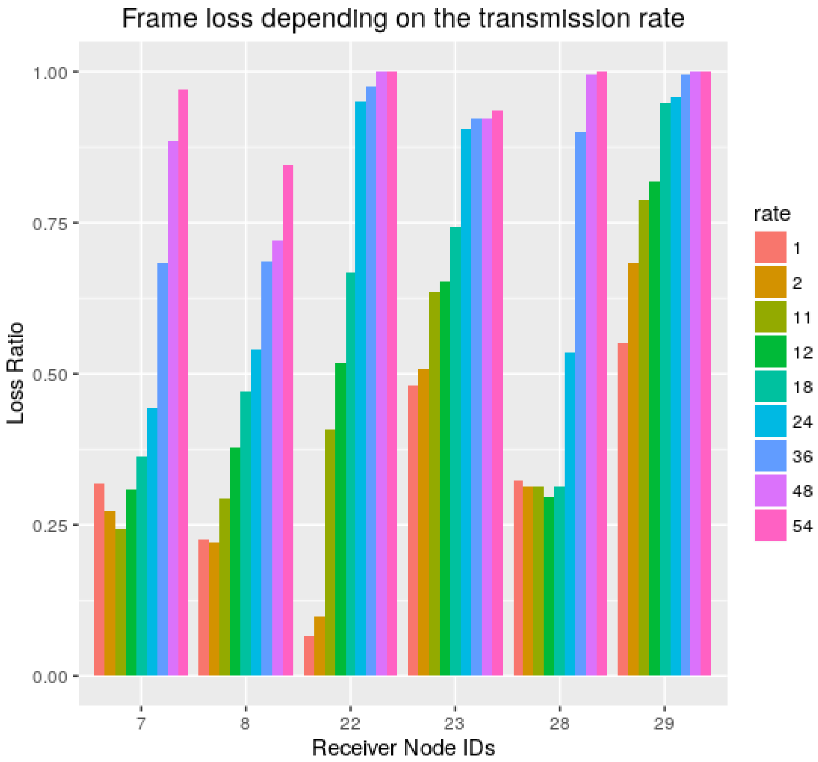


(e) Frame lose ratio ($lossRatio_{v,v'}$) with respect to transmission rate on links with Node 28 as the source



(f) Frame lose ratio ($lossRatio_{v,v'}$) with respect to transmission rate on links with Node 29 as the source

Figure A.2.: Experiment conducted to check the frame loss ratio depending on the transmission rate in the presence of external traffic



(g) Frame lose ratio ($lossRatio_{v,v'}$) observed in each node as receiver during each experiment

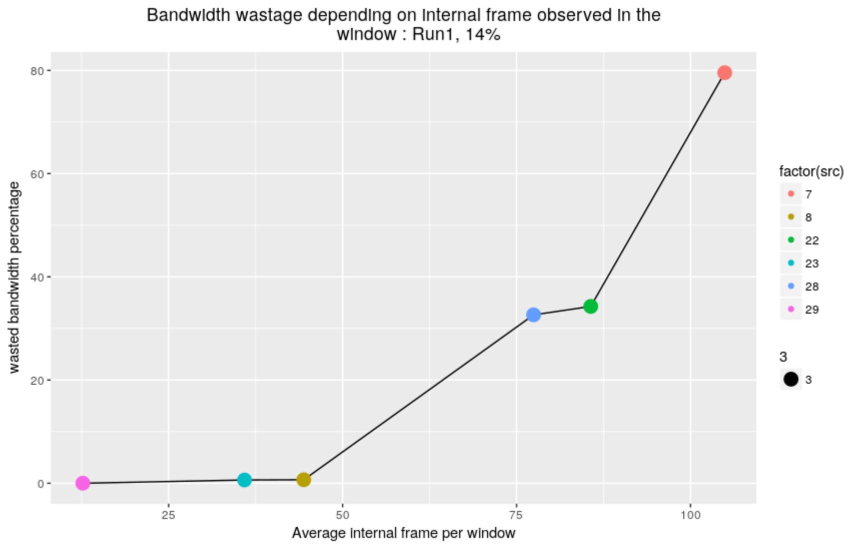
Figure A.2.: Experiment conducted to check the frame loss ratio depending on the transmission rate in the presence of external traffic

src	$wastedTokens_{src}(t)$	token_size (μs)	$usedBwPer_{src}(t)$	$channelBusyTimePer_{src}(t)$	$mediumEnergySensedPer_{src}(t)$	$wastedBwPer_v(t)$
7	14949733	14	1.08	64.77	63.68	79.56
8	140337	14	13.45	64.64	51.20	0.67
22	7161257	14	8.39	60.40	52.02	34.27
23	131446	14	13.63	67.93	54.31	0.63
28	6880844	14	8.64	58.51	49.86	32.63
29	102198	14	13.70	56.94	43.23	0.49

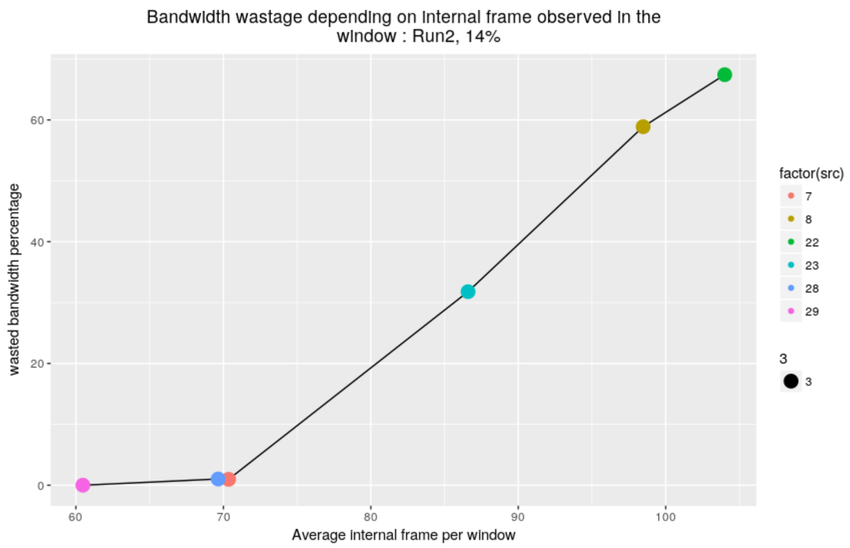
Table A.2.: Run 1 with 14% node base bandwidth and 84% network load in the presence of external traffic (bandwidth profile *P.5.1.k*)

src	$wastedTokens_{src}(t)$	token_size (μs)	$usedBwPer_{src}(t)$	$channelBusyTimePer_{src}(t)$	$mediumEnergySensedPer_{src}(t)$	$wastedBwPer_v(t)$
7	57471	14	13.57	59.23	45.66	0.97
8	3653458	14	4.30	56.65	52.35	58.88
22	4298137	14	2.63	57.19	54.55	67.42
23	1924793	14	8.74	58.19	49.44	31.79
28	60390	14	13.65	58.49	44.84	1.02
29	38153	14	13.67	54.16	40.48	0.65

Table A.3.: Run 2 with 14% node base bandwidth and 84% network load in the presence of external traffic (bandwidth profile *P.5.1.k*)

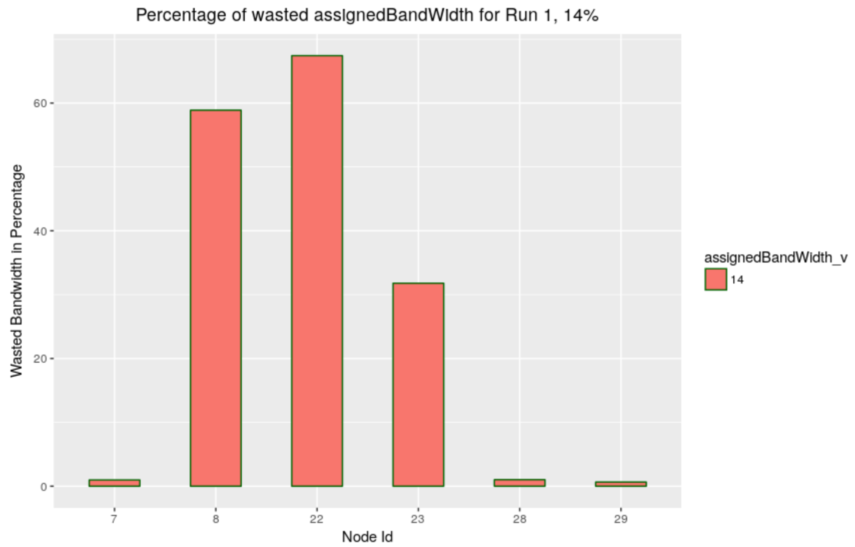


(a) Wasted assigned bandwidth for nodes depedning on the internal frames received during experiment with bandwidth profile number *P.5.1.k* for Run 1

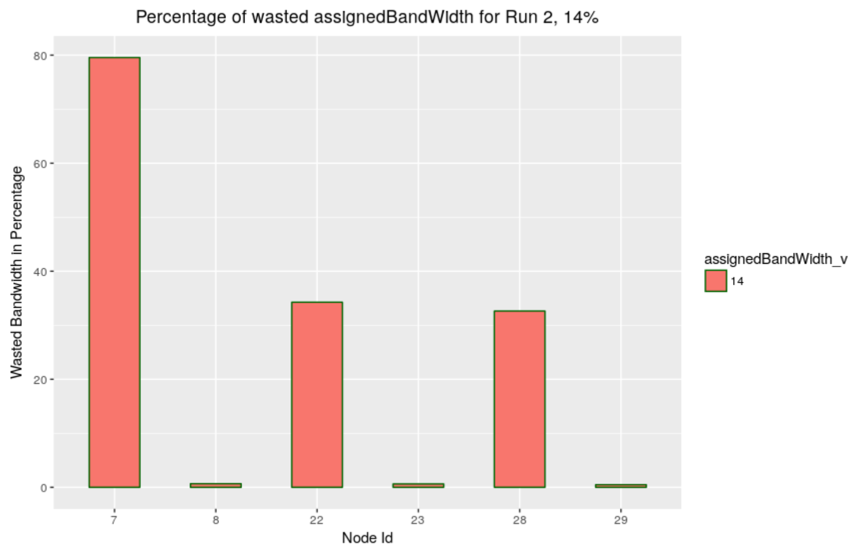


(b) Wasted assigned bandwidth for nodes depedning on the internal frames received during experiment with bandwidth profile number *P.5.1.k* for Run 2

Figure A.3.: Wasted assigned bandwidth for the network that dependent, and independent on the external traffic

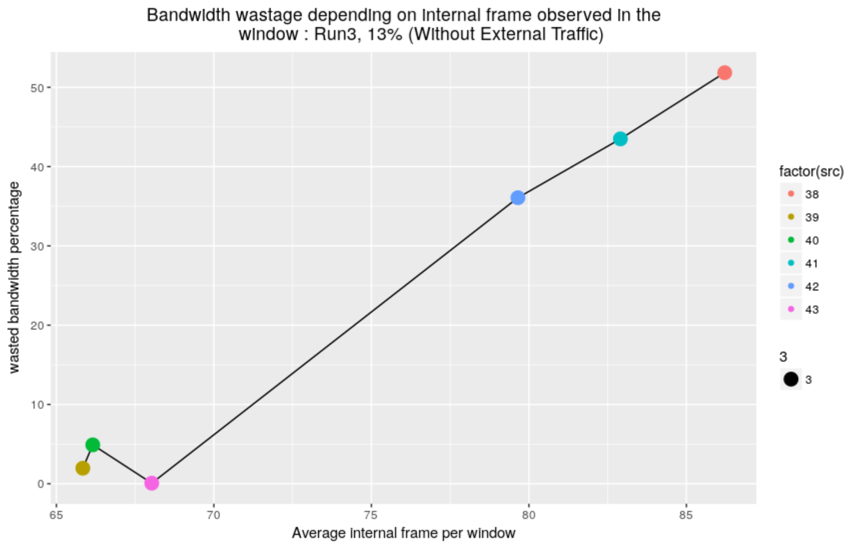


(c) Wasted assigned bandwidth for nodes during experiment with bandwidth profile number *P.5.1.k* for Run 1



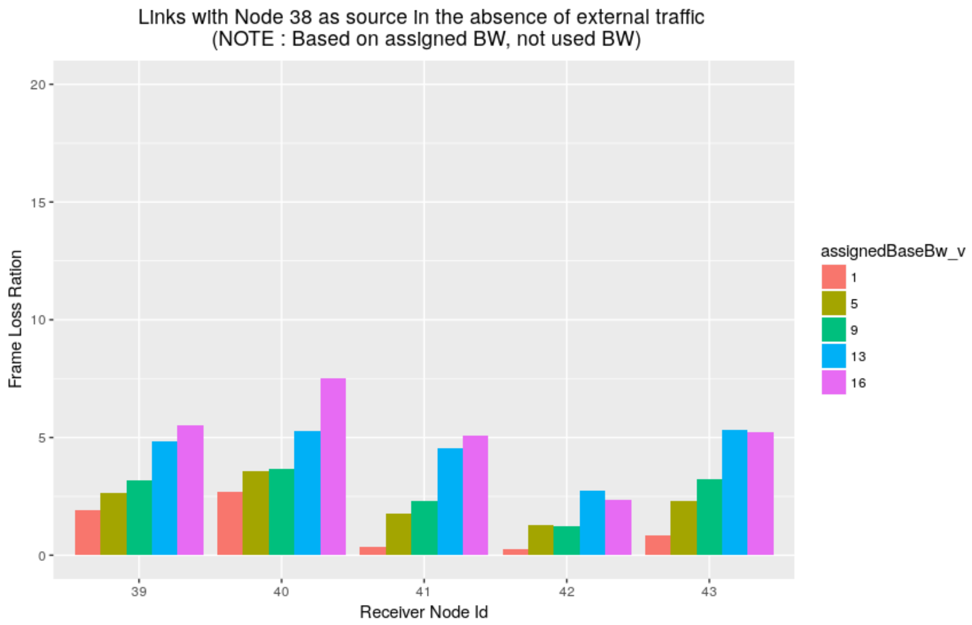
(d) Wasted assigned bandwidth for nodes during experiment with bandwidth profile number *P.5.1.k* for Run 2

Figure A.3.: Wasted assigned bandwidth for the network that dependent, and independent on the external traffic

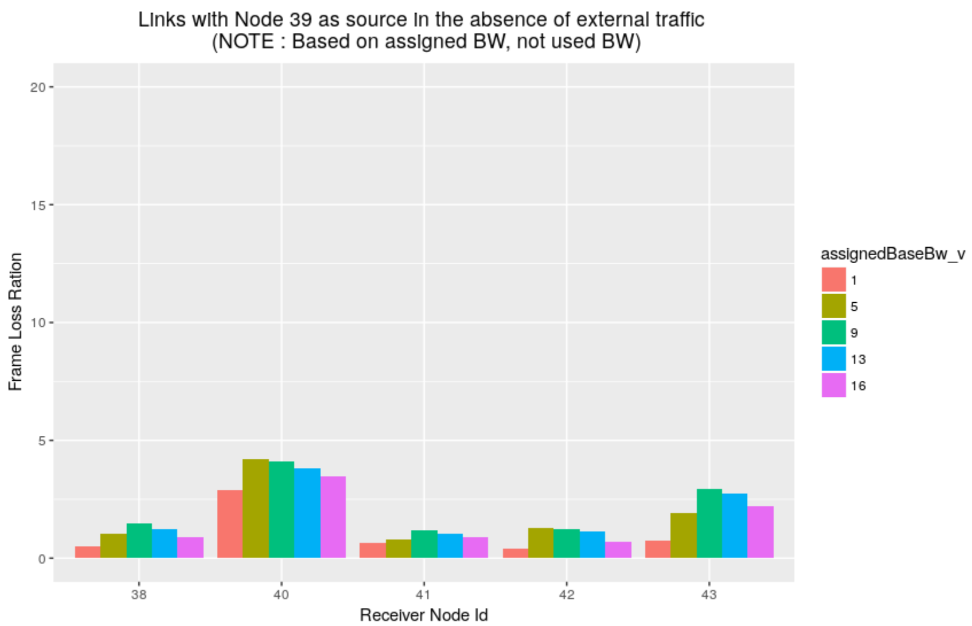


(e) Wasted assigned bandwidth for the network that dependent, and independent on the external traffic

Figure A.3.: Wasted assigned bandwidth for the network that dependent, and independent on the external traffic

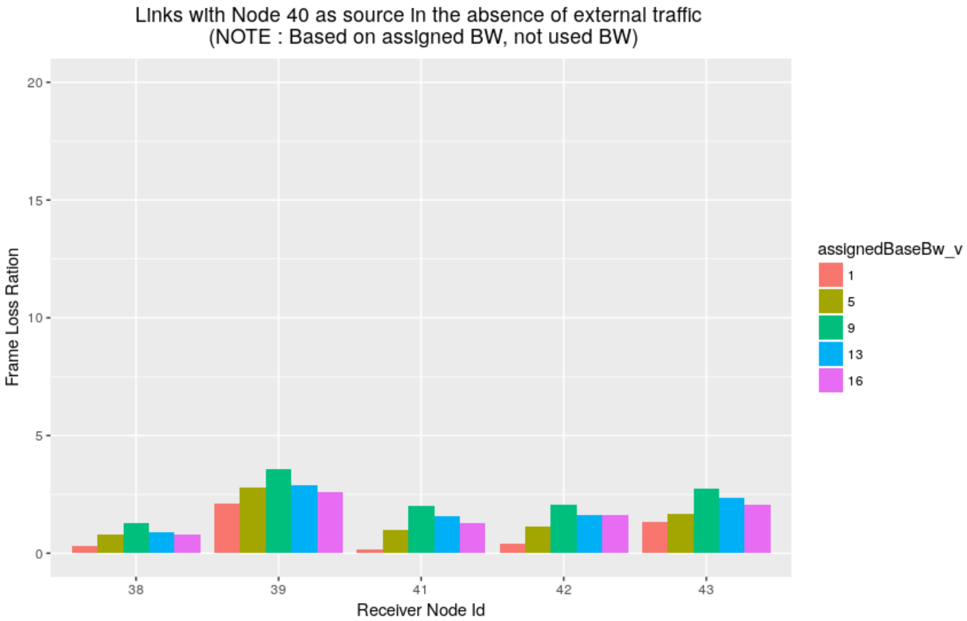


(a) Frame lose ratio ($lossRatio_{v,v'}$) for links with Node 38 as the source

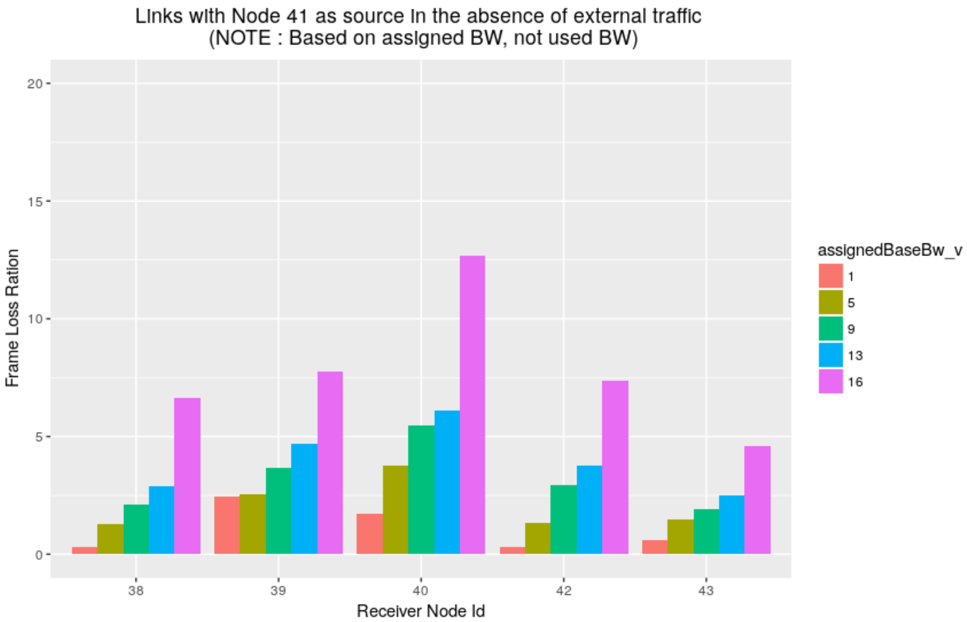


(b) Frame lose ratio ($lossRatio_{v,v'}$) for links with Node 39 as the source

Figure A.4.: Frame lose ratio ($lossRatio_{v,v'}$) depending on network load in the absence of external traffic

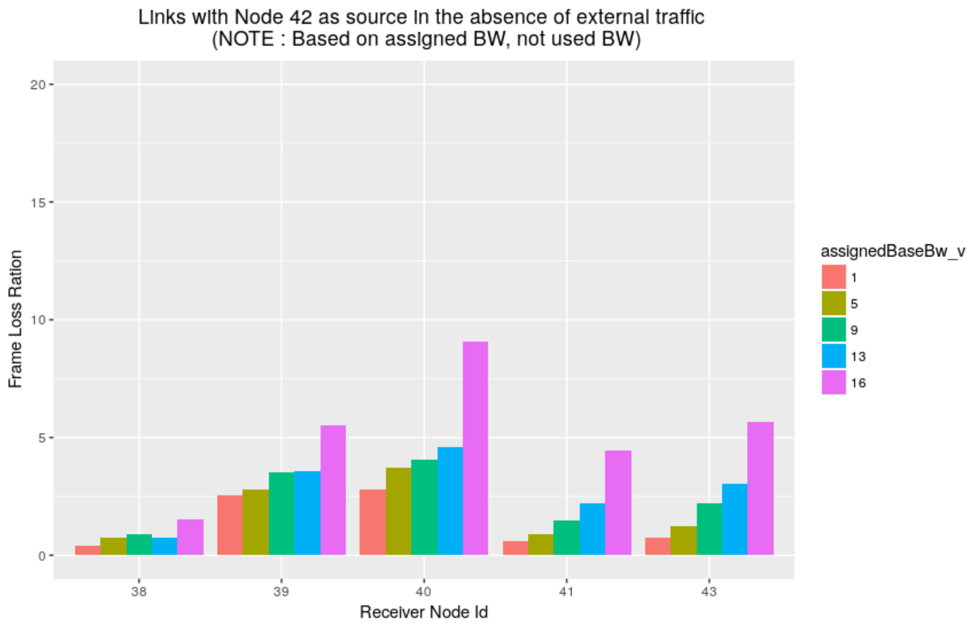


(c) Frame lose ratio ($lossRatio_{v,v'}$) for links with Node 40 as the source

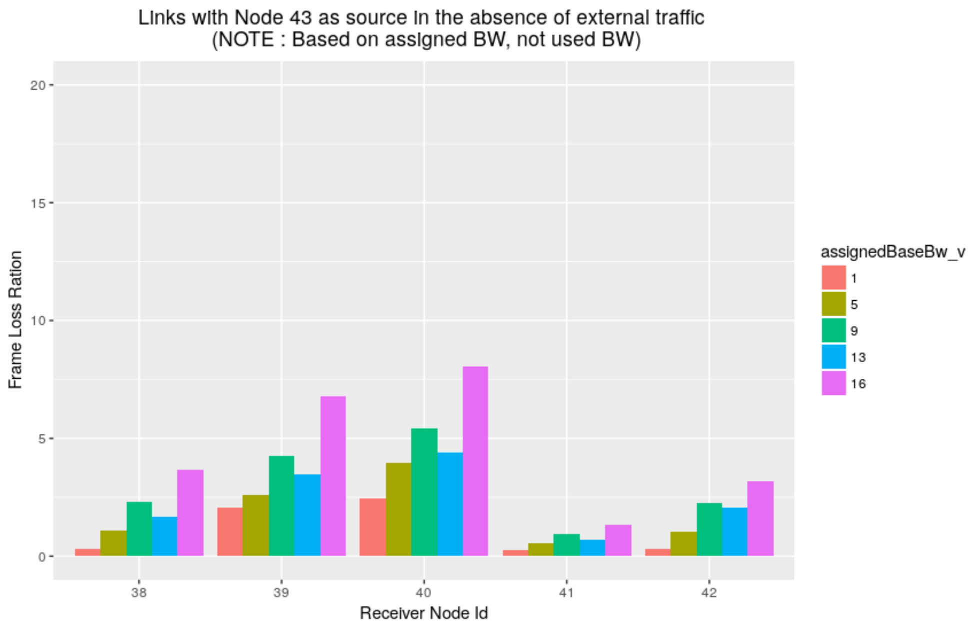


(d) Frame lose ratio ($lossRatio_{v,v'}$) for links with Node 41 as the source

Figure A.4.: Frame lose ratio ($lossRatio_{v,v'}$) depending on network load in the absence of external traffic

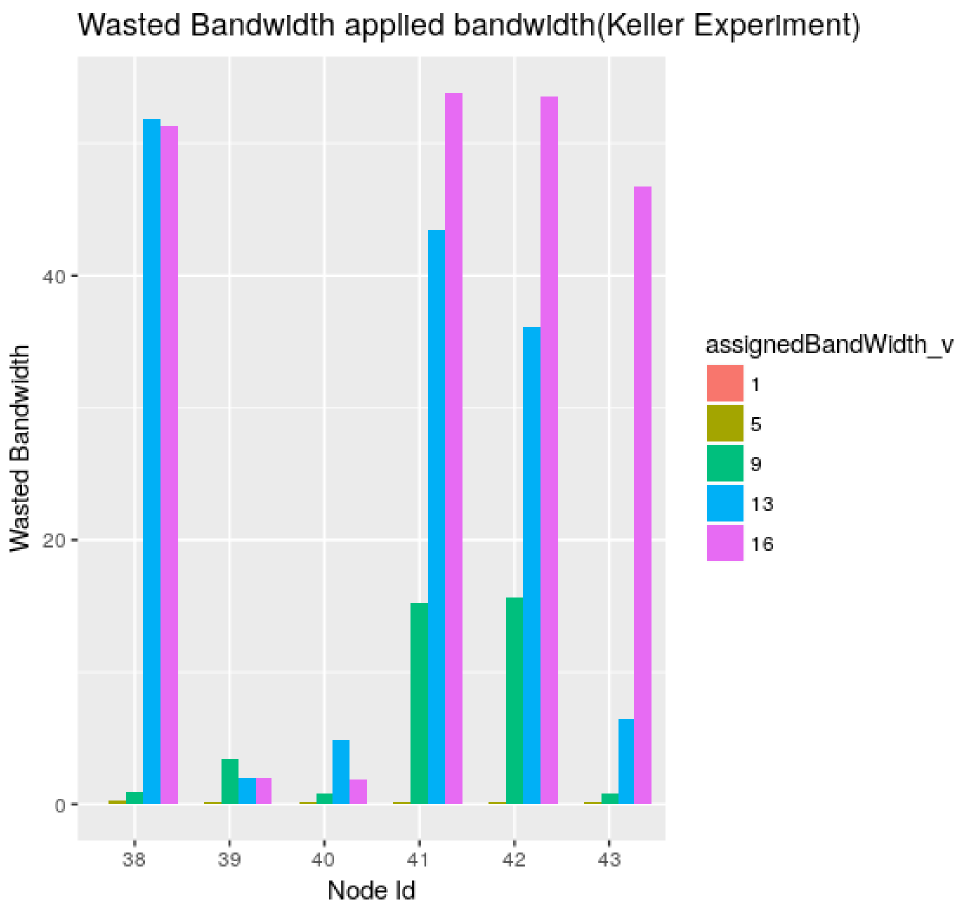


(e) Frame lose ratio ($lossRatio_{v,v'}$) for links with Node 42 as the source



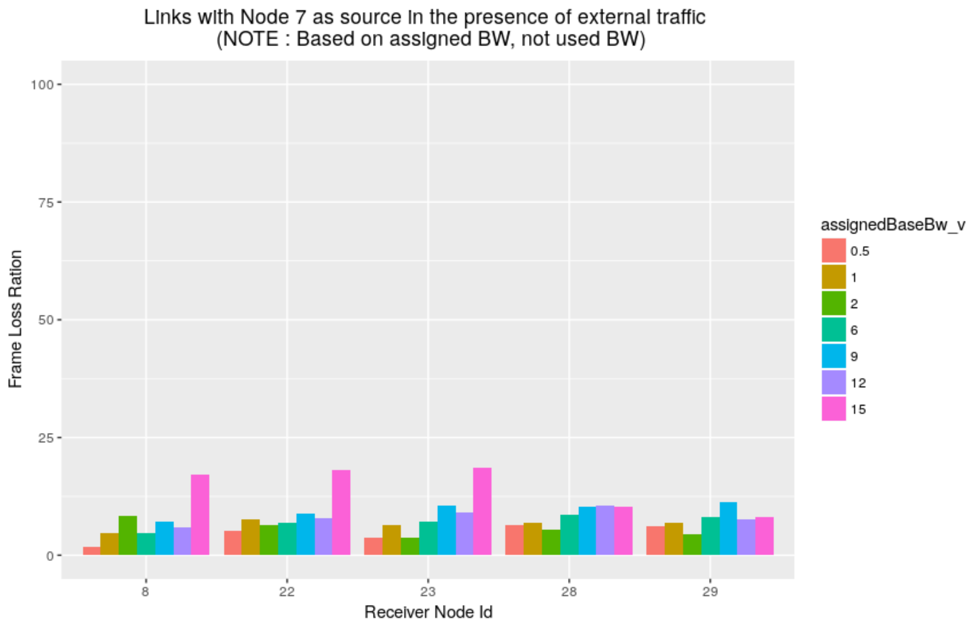
(f) Frame lose ratio ($lossRatio_{v,v'}$) for links with Node 43 as the source

Figure A.4.: Frame lose ratio ($lossRatio_{v,v'}$) depending on network load in the absence of external traffic

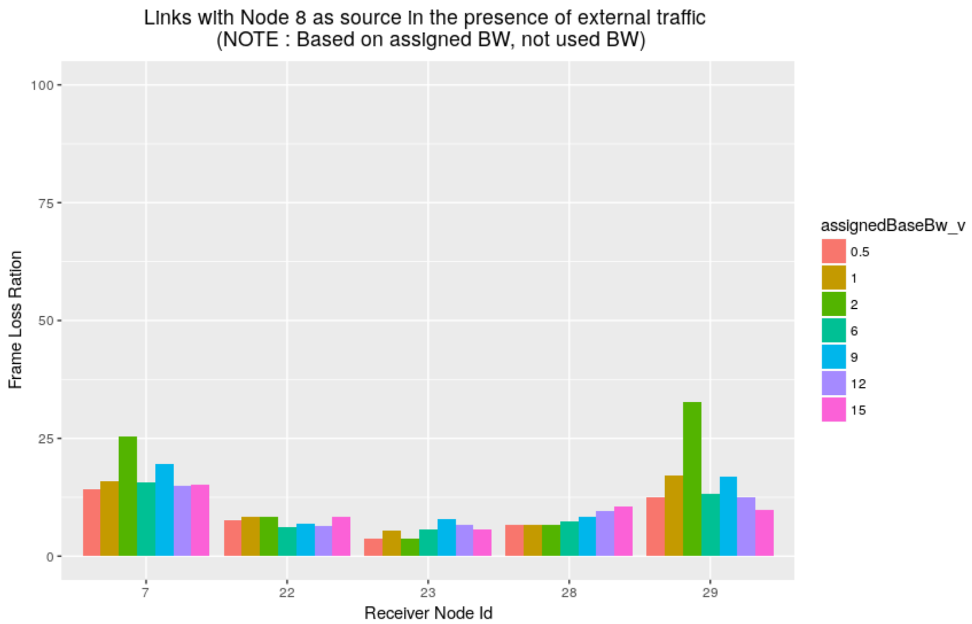


(g) Frame lose ratio ($lossRatio_{v,v'}$) observed in each node as the receiver during each experiment

Figure A.4.: Frame lose ratio ($lossRatio_{v,v'}$) depending on network load in the absence of external traffic

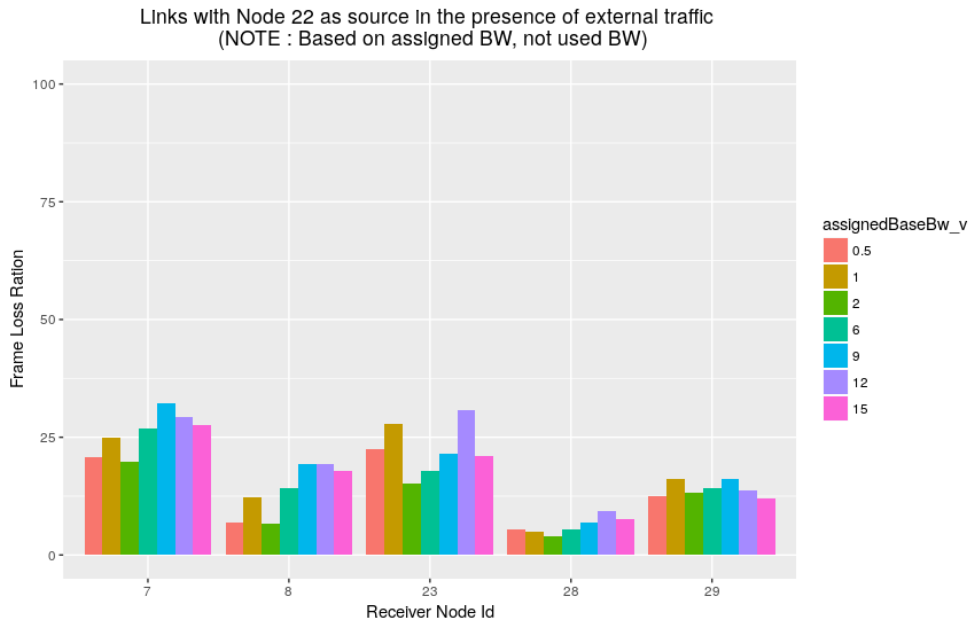


(a) Frame lose ratio ($lossRatio_{v,v'}$) for links with Node 7 as the source

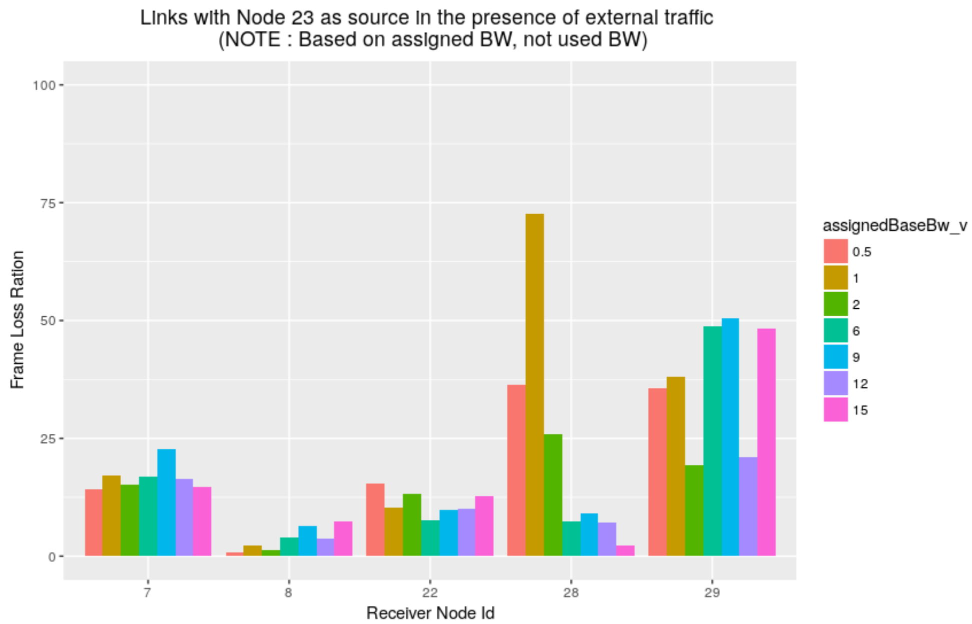


(b) Frame lose ratio ($lossRatio_{v,v'}$) for links with Node 8 as the source

Figure A.5.: Frame lose ratio ($lossRatio_{v,v'}$) depending on network load in the presence of external traffic

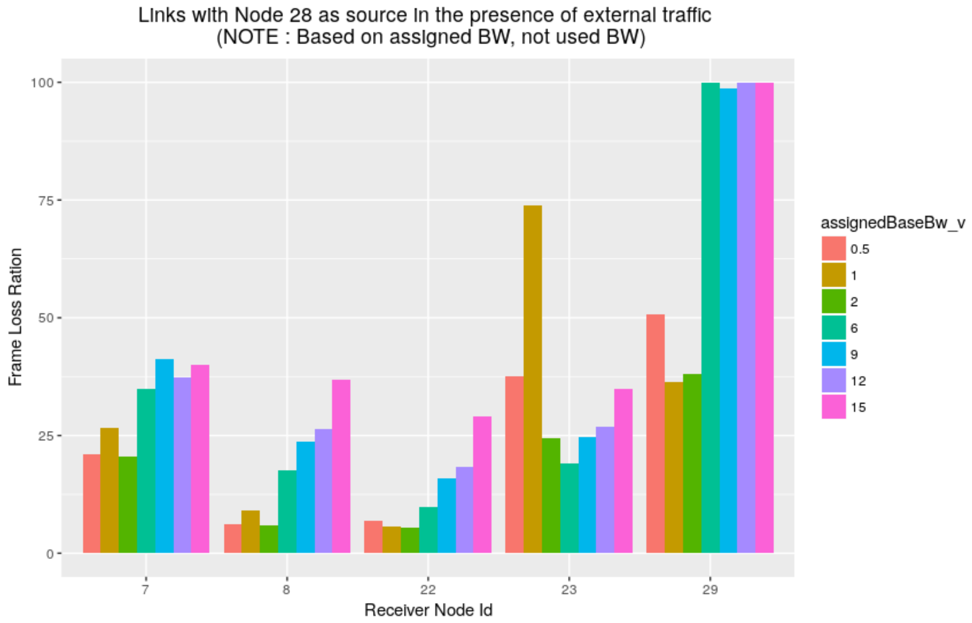


(c) Frame lose ratio ($lossRatio_{v,v'}$) for links with Node 22 as the source

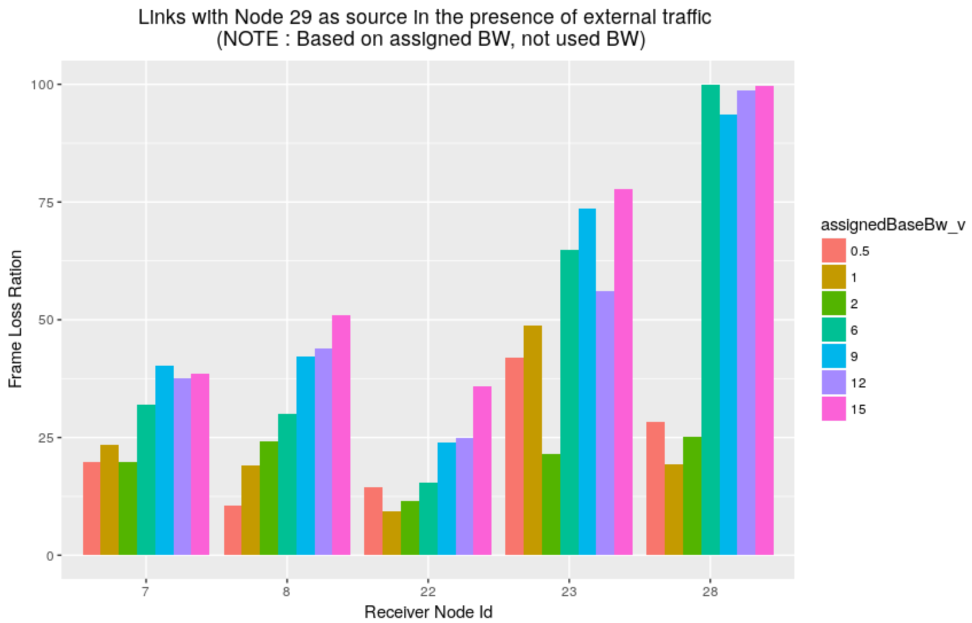


(d) Frame lose ratio ($lossRatio_{v,v'}$) for links with Node 23 as the source

Figure A.5.: Frame lose ratio ($lossRatio_{v,v'}$) depending on network load in the presence of external traffic

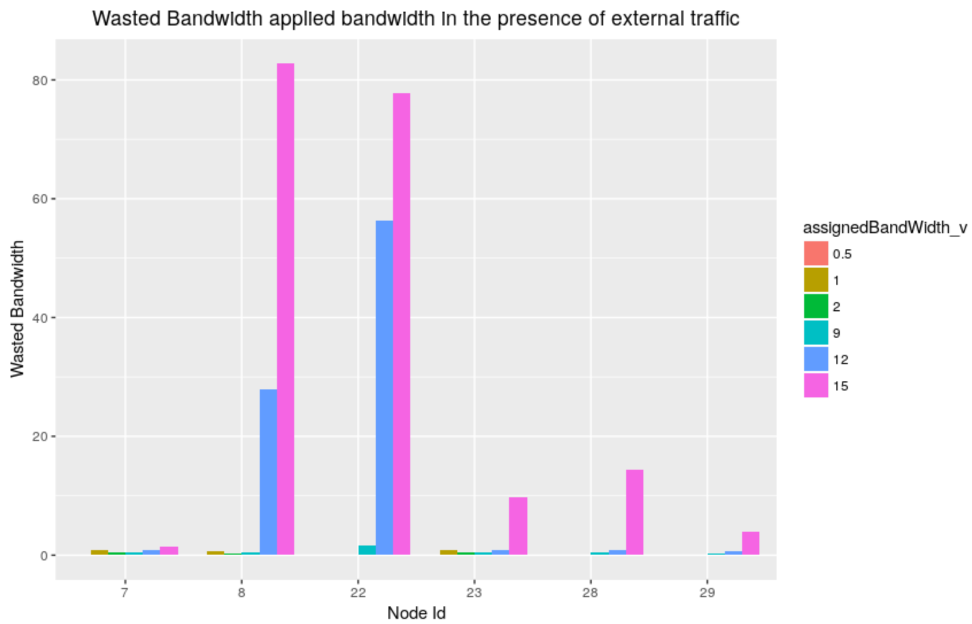


(e) Frame lose ratio ($lossRatio_{v,v'}$) for links with Node 28 as the source



(f) Frame lose ratio ($lossRatio_{v,v'}$) for links with Node 29 as the source

Figure A.5.: Frame lose ratio ($lossRatio_{v,v'}$) depending on network load in the presence of external traffic



(g) Frame lose ratio ($lossRatio_{v,v'}$) observed in each node as the receiver during each experiment

Figure A.5.: Frame lose ratio ($lossRatio_{v,v'}$) depending on network load in the presence of external traffic

Bibliography

[ASB+08] ACHARYA, PRASHANTH ARAVINDA KUMAR, ASHISH SHARMA, ELIZABETH M BELDING, KEVIN C ALMEROTH and KONSTANTINA PAPAGIANNAKI: *Congestion-aware rate adaptation in wireless networks: A measurement-driven approach*. In *2008 5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, pages 1–9. IEEE, 2008.

[Bic05] BICKET, JOHN CHARLES: *Bit-rate selection in wireless networks*. 2005.

[BSSD13] BLOESSL, BASTIAN, MICHELE SEGATA, CHRISTOPH SOMMER and FALKO DRESSLER: *An IEEE 802.11 a/g/p OFDM Receiver for GNU Radio*. In *Proceedings of the second workshop on Software radio implementation forum*, pages 9–16. ACM, 2013.

[C+99] COMMITTEE, IEEE COMPUTER SOCIETY LAN MAN STANDARDS et al.: *IEEE Standard for Information Technology: Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, 1999.

[cfg] *http://lxr.free-electrons.com/source/include/net/cfg80211.h?v=3.9 lines 460 - 476*.

[CPVM10] COSTA, ROBSON, PAULO PORTUGAL, FRANCISCO VASQUES and RICARDO MORAES: *A TDMA-based mechanism for real-time communication in IEEE 802.11 e networks*. In *Emerging Technologies and Factory Automation (ETFA), 2010 IEEE Conference on*, pages 1–9. IEEE, 2010.

[CVB02] CHATZIMISIOS, PERIKLIS, VASILEIOS VITSAS and ANTHONY C BOUCOUVALAS: *Throughput and delay analysis of IEEE 802.11 protocol*. In *Networked Appliances, 2002. Liverpool. Proceedings. 2002 IEEE 5th International Workshop on*, pages 168–174. IEEE, 2002.

[DM09] DJUKIC, PETAR and PRASANT MOHAPATRA: *Soft-TDMAC: A software TDMA-based MAC over commodity 802.11 hardware*. In *INFOCOM 2009, IEEE*, pages 1836–1844. IEEE, 2009.

[GLMC14] GÜNTHER, STEPHAN M, MAURICE LECLAIRE, JULIUS MICHAELIS and GEORG CARLE: *Analysis of injection capabilities and media access of ieee 802.11 hardware in monitor mode*. In *2014 IEEE Network Operations and Management Symposium (NOMS)*, pages 1–9. IEEE, 2014.

[GNU] *http://gnuradio.org/*.

[GVDS01] GOLMIE, NADA, ROBERT E VAN DYCK and A SOLTANIAN: *Interference of Bluetooth and IEEE 802.11: Simulation modeling and performance evaluation*. In *Proceedings of the*

4th ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems, pages 11–18. ACM, 2001.

[GVDS+03] GOLMIE, NADA, ROBERT E VAN DYCK, AMIR SOLTANIAN, ARNAUD TONNERRE and OLIVIER REBALA: *Interference evaluation of Bluetooth and IEEE 802.11 b systems*. Wireless Networks, 9(3):201–211, 2003.

[HXBZ09] HUO, HONGWEI, YOUZHI XU, CELAL CAN BILEN and HONGKE ZHANG: *Coexistence issues of 2.4 GHz sensor networks with other RF devices at home*. In *Sensor Technologies and Applications, 2009. SENSORCOMM'09. Third International Conference on*, pages 200–205. IEEE, 2009.

[IEE] *IEEE 802.11™-2012*.

[IMM] *http://crawdad.org/cambridge/haggle/20060915/imote/*.

[IWL] *Industrial WLAN from siemens*.

[JRABR05] JARDOSH, AMIT P, KRISHNA N RAMACHANDRAN, KEVIN C ALMEROTH and ELIZABETH M BELDING-ROYER: *Understanding congestion in IEEE 802.11 b wireless networks*. In *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*, pages 25–25. USENIX Association, 2005.

[KCG15] KRAMER, CHRISTOPHER, DENNIS CHRISTMANN and REINHARD GOTZHEIN: *Automatic topology discovery in TDMA-based Ad Hoc networks*. In *2015 International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 634–639. IEEE, 2015.

[Net] *Netlink Protocol Library Suite : https://www.infradead.org/ tgr/libnl/*.

[nl8] *http://lxr.free-electrons.com/source/include/linux/nl80211.h?v=3.5 lines 2016 - 2032*.

[PTS01] PUNNOOSE, RATISH J, RICHARD S TSENG and DANIEL D STANCIL: *Experimental results for interference between Bluetooth and IEEE 802.11 b DSSS systems*. In *IEEE Vehicular Technology Conference*, volume 1, pages 67–71. IEEE; 1999, 2001.

[Rad] *http://www.radiotap.org/*.

[RPI] *https://www.raspberrypi.org/*.

[SAB08] SHERIFF, IRFAN, PRASHANTH ARAVINDA KUMAR ACHARYA and ELIZABETH M BELDING: *Measurement-driven admission control on wireless backhaul networks*. Computer Communications, 31(7):1354–1371, 2008.

[SDR] *http://www.wirelessinnovation.org/assets/documents/SDRF-06-R-0011-V1_0_0.pdf*.

[Sei10] SEITZ, JOCHEN: *Modeling and Simulation of Communication Systems and Networks Chapter 1. Reliability*. 2010.

[SFa] *http://www.smartfactory-kl.de/*.

[SG05] SIKORA, AXEL and VOICU F GROZA: *Coexistence of IEEE802. 15.4 with other Systems in the 2.4 GHz-ISM-Band*. In *2005 IEEE Instrumentationand Measurement Technology Conference Proceedings*, volume 3, pages 1786–1791. IEEE, 2005.

[SHM+08] SONG, JIANPING, SONG HAN, AL MOK, DEJI CHEN, MIKE LUCAS, MARK NIXON and WALLY PRATT: *WirelessHART: Applying wireless technology in real-time industrial* *process control*. In *Real-Time and Embedded Technology and Applications Symposium, 2008. RTAS'08. IEEE*, pages 377–386. IEEE, 2008.

[SIN] *http://www.software-cluster.org/en/research/projects/joint-projects/sinnodium*.

[SK96] SOBRINHO, JOAO L and ANJUR SUNDARESAN KRISHNAKUMAR: *Real-time traffic over the IEEE 802.11 medium access control layer*. Bell labs technical Journal, 1(2):172–187, 1996.

[Tan81] TANENBAUM, ANDREW S.: *Computer Networks, Third edition*. Prentice Hall PTR, Reading, Massachusetts, 2003, Pages 380-381.

[Tan84] TANENBAUM, ANDREW S.: *Computer Networks, Third edition*. Prentice Hall PTR, Reading, Massachusetts, 2003, Pages 381-384.