UNIVERSITY CARLOS III OF MADRID

Department of Telematics Engineering

Control-Theoretic Adaptive Mechanisms for Performance Optimization of IEEE 802.11 WLANs: Design, Implementation and Experimental Evaluation

Author: Paul Horaţiu Pătraş, M.Sc.
Advisor: Albert Banchs Roca, Ph.D.

Leganés, Madrid, Spain
March 2011

A thesis submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

The most beautiful thing we can experience is the mysterious.
It is the source of all true art and all science.
He to whom this emotion is a stranger,
who can no longer pause to wonder and stand rapt in awe,
is as good as dead: his eyes are closed.

—Albert Einstein

Acknowledgments

First and foremost, I would like to thank my advisor, Albert Banchs, for his invaluable guidance and insightful suggestions, which contributed to the success of my work. He has been truly a model, teaching me how to tackle the most difficult problems, develop self-confidence and always be determined to overcome the obstacles in my research.

I am extremely grateful to my colleague and dear friend Pablo, who always had his door open for me, constantly challenged me over the past years, was always enthusiastic about working on new ideas and with whom I also spent unforgettable moments outside the campus.

I thank Arturo Azcorra for his valuable feedback on my work and for giving me the opportunity to collaborate with other research groups. I acknowledge Institute IMDEA Networks for funding my Ph.D. and all the institute's staff, but especially Rebeca and José Félix, for providing me support in administrative matters, which helped me devote more time to my research. I would like to thank my colleagues from the NETCOM group for the great collaboration within the DAIDALOS and CARMEN research projects, and in particular Carlos Jesús, Andrés and Antonio, who stood by me with friendship and support in many practical issues. I also appreciate the collaboration with Vincenzo, Andrea and Marco, who helped me with the testbed deployment, measurements and important comments.

I am grateful to Edward Knightly for hosting me at Rice University, for involving me in fruitful debates and focusing my research. I also thank Tasos for his collaboration on wide-spectrum networks research, as well as Cen, Naren, Eugenio, Ryan and Misko for the useful discussions and the good time spent together while working with the Rice Networks Group.

For countless reasons I thank my friends Vali, Sorin, Ionut, Cristi and Cosmin. And in particular, I thank my loving girlfriend, Mariana, whose encouragements, patience and joy have been very important to me in the final stages of this Ph.D.

Finally, but most of all, I thank my family for their endless love and support throughout the years, for believing in me and encouraging my pursuits.

Abstract

The media access control (MAC) layer of the IEEE 802.11 standard specifies a set of parameters that regulate the behavior of the wireless stations when accessing the channel. Although the standard defines a set of recommended values for these parameters, they are statically set and do not take into account the current conditions in the wireless local area network (WLAN) in terms of, e.g., number of contending stations and the traffic they generate, which results in suboptimal performance. In this thesis we propose two novel control theoretic approaches to optimally configure the WLAN parameters based on the dynamically observed network conditions: a *Centralized Adaptive Control (CAC)* algorithm, whereby the access point (AP) computes the configuration that maximizes performance and signals it to the active stations, and a *Distributed Adaptive Control (DAC)* algorithm, which is independently employed by each station with the same goal.

In contrast to previous proposals, which are mostly based on heuristics, our approaches build upon *(i)* analytical models of the WLAN performance, used to derive the optimal point of operation of the IEEE 802.11 protocol, and *(ii)* mathematical foundations from single- and multi-variable control theory, used to design the mechanisms that drive the WLAN to this point of operation. Another key advantage of the proposed algorithms over existing approaches is that they are compliant with the IEEE 802.11 standard and can be implemented with current wireless cards without introducing any modifications into their hardware and/or firmware. We show by means of an exhaustive performance evaluation study that our algorithms maximize the WLAN performance in terms of throughput and delay under a wide set of network conditions, substantially outperforming the standard recommended configuration as well as previous adaptive proposals.

Finally, we present our experiences with implementing the proposed adaptive algorithms in a real IEEE 802.11 testbed and discuss the implementation details of the building blocks that comprise these mechanisms. We evaluate their performance by conducting extensive measurements, considering different network conditions in terms of number of nodes, transmission power employed and traffic generated. Based on the obtained results, we provide valuable insights on the performance of the distributed and centralized algorithms and discuss the suitability of these schemes for real deployments.

Contents

Acknowledgments **i**

Abstract **iii**

Contents **vii**

List of Figures **xi**

1 Introduction **1**

1.1 Summary of Thesis Contributions 3

1.2 Thesis Overview 4

2 Background **7**

2.1 IEEE 802.11 EDCA 7

2.2 Related Work 10

3 Centralized Adaptive Control Algorithm **13**

3.1 Data Traffic Scenario 13

3.1.1 Throughput Analysis and Optimization 14

3.1.2 CAC Algorithm 17

3.1.3 Performance Evaluation 23

3.2 Real-Time Traffic Scenario 30

3.2.1 Analytical Model 31

3.2.2 CAC–VI Algorithm 36

3.2.3 Performance Evaluation 41

3.3 Summary 56

4 Distributed Adaptive Control Algorithm 59

4.1 DAC Algorithm . . . 60

4.2 Steady State Analysis . . . 63

4.3 Stability Analysis . . . 65

4.4 Performance Evaluation . . . 69

4.4.1 Saturated Scenario . . . 70

4.4.2 Non-saturated Scenario . . . 70

4.4.3 Mixed Scenario . . . 71

4.4.4 Convergence . . . 72

4.4.5 Stability and Speed of Reaction to Changes . . . 74

4.4.6 Fairness . . . 76

4.5 Summary . . . 77

5 Experimental Evaluation 79

5.1 Implementation Details . . . 79

5.1.1 CAC Algorithm . . . 80

5.1.2 DAC Algorithm . . . 81

5.1.3 Hardware & Software Platform . . . 82

5.1.4 Implementation Overview . . . 83

5.1.5 Estimation of p_{obs} . . . 84

5.1.6 Estimation of p_{own} . . . 84

5.1.7 Contention Window Update . . . 85

5.2 Performance Evaluation . . . 86

5.2.1 Testbed & Evaluation Methodology . . . 86

5.2.2 Practical Validation of the Algorithms' Operation . . . 89

5.2.3 Impact of Link Quality on Throughput Distribution . . . 91

5.2.4 Impact of Hidden Nodes . . . 95

5.2.5 Impact of Network Size . . . 96

5.2.6 Impact of Dynamic Traffic Conditions . . . 98

5.3 Summary . . . 100

6 Conclusions and Future Work **103**

References **113**

Appendix **115**

List of Figures

2.1 Example of EDCA operation with 2 stations. 8

2.2 Retry flag marking upon collisions. 9

3.1 Control system. 18

3.2 Linearized system. 21

3.3 Throughput performance. 23

3.4 Stable configuration. 25

3.5 Unstable configuration. 25

3.6 Speed of reaction to changes. 26

3.7 Instantaneous throughput. 26

3.8 Non-saturated stations. 27

3.9 Bursty traffic. 28

3.10 Comparison against other approaches. 28

3.11 Impact of channel errors. 29

3.12 Markov chain model of the WLAN. 33

3.13 Control system. 37

3.14 Linearized system. 40

3.15 Validation of the delay model. 42

3.16 Optimal collision probability. 43

3.17 CW configuration. 43

3.18 Delay performance of the proposed algorithm 44

3.19 Stability evaluation. 45

3.20 Speed of reaction to changes. 46
3.21 Time evolution of the error signal. 46
3.22 Delay performance for different TXOP. 47
3.23 Average, 90^{th} and 95^{th} percentiles of the access delay. 48
3.24 Total delay vs. access delay. 49
3.25 Burstiness vs average delay. 49
3.26 Support for graceful degradation of video flows. 50
3.27 Comparison against other approaches: H.264 video. 51
3.28 Comparison against other approaches: MPEG-4 video. 51
3.29 Comparison against other approaches: H.263 video. 52
3.30 MOS evaluation. 53
3.31 Delay performance under channel errors. 54
3.32 Delay performance under mixed traffic. 55
3.33 MOS evaluation with real users. 55
3.34 Sample video frame with the default EDCA configuration. 56
3.35 Sample video frame with the proposed CAC–VI algorithm. 56

4.1 DAC Algorithm 60
4.2 Control system 65
4.3 Saturated scenario 70
4.4 Non-saturation scenario 71
4.5 Throughput performance under mixed traffic conditions 72
4.6 Average delay under mixed traffic conditions 73
4.7 Throughput performance of the mixed unbalanced scenario 73
4.8 Average delay of the mixed unbalanced scenario 74
4.9 Convergence 74
4.10 Stability validation 75
4.11 Speed of reaction to changes 75
4.12 Fairness 76

5.1 CAC algorithm. 80
5.2 DAC algorithm 81
5.3 Hardware platform 82
5.4 CAC and DAC implementations 83
5.5 Deployed testbed. 87
5.6 CAC: Announced CW_{min} 88
5.7 CAC: Observed collision probability 88
5.8 DAC: CW_{min} used by four nodes 90
5.9 DAC: Estimated p_{obs} and p_{own} 90
5.10 Total throughput with heterogeneous link qualities 91
5.11 Throughput per station with heterogeneous link qualities 92
5.12 Jain's fairness index with heterogeneous link qualities 92
5.13 Throughput per station with quasi-homogeneous link qualities 93
5.14 Jain's fairness index with quasi-homogeneous link qualities 94
5.15 Total throughput with quasi-homogeneous link qualities 94
5.16 Throughput performance with hidden nodes 96
5.17 Total throughput for different number of stations 97
5.18 Jain's fairness index for different number of stations 98
5.19 CAC: CW_{min} behavior under increasing number of stations 99
5.20 DAC: CW_{min} behavior under increasing number of stations 99
5.21 Delay performance under dynamic load conditions 100

Chapter 1

Introduction

The IEEE 802.11 standard for Wireless LANs [1] has become one of the most widely deployed technologies for providing broadband connectivity to the Internet in the recent years. The reduced investment costs (facilitated by the use of unlicensed spectrum and the availability of low cost devices), the deployment flexibility and unpretentious management have lead to the emergence of a substantial number of WiFi Access Points, used not only in office environments or as public hot-spots but also to connect residential users and their multimedia devices to the Internet. As a consequence, today's wireless access deployments based on IEEE 802.11 vary from small scale networks installed in airports, cafés and universities, to larger scale public and commercial ones, such as Google WiFi[1] or TFA Wireless.[2]

The IEEE 802.11 standard [1] defines two different channel access mechanisms, a centralized one, known as the Point Coordination Function (PCF), and a distributed one, the Distributed Coordination Function (DCF). However, most of the current WLANs are based on the latter, i.e., a CSMA/CA protocol that only provides a best effort service, while the PCF mechanism has received relatively little attention from manufacturers.

To satisfy the increasing bandwidth demands, the basic physical layer specification of 2 Mbps capacity [2] has been extended, to provide up to 11 Mbps nominal throughput with IEEE 802.11b [3] and up to 54 Mbps with IEEE 802.11a [4] and IEEE 802.11g [5]. This rate increase has enabled the use of WLANs also for real-time applications, such as, e.g., Voice over IP (VoIP), video streaming or video conferencing. (Note that today's laptops already have an integrated webcam.) However, these bandwidth and delay sensitive applications are properly supported only in over-provisioned scenarios, where the best-effort based scheme of DCF is enough to fulfill the QoS requirements.

In order to overcome this limitation, the revised version of the standard specifies

[1] `http://wifi.google.com/`
[2] `http://tfa.rice.edu/`

an improved channel access scheme, the Hybrid Coordination Function (HCF), which consists of two access mechanisms, the HCF Controlled Channel Access (HCCA) and the Enhanced Distributed Coordination Access (EDCA) [6]. The former is based, like PCF, on a centralized controller that schedules the transmissions in the WLAN, while the latter is an extension of DCF that supports service differentiation through four different Access Categories (namely voice, video, best-effort and background). These Access Categories can be configured with different values of the contention parameters, leading to statistical service differentiation. However, the configuration of both mechanisms is left open, as the standard only specifies a simple scheduler to provide constant bit rate (CBR) services for the case of HCCA, and a set of recommended values of the contention parameters for the case of EDCA.

The fixed set of recommended values for the EDCA parameters employed by the standard is statically set, which results in poor throughput and delay performance for most scenarios, as the optimal configuration of the channel access parameters depends on the WLAN conditions, these including the number of stations and their load [7–9]. In particular, if too many stations contend with overly small Contention Window (CW) values,[3] the collision rate will be very high, which yields a degraded performance. Similarly, if few stations contend with too large CW's, the attempt rate will be low and the channel will be underutilized most of the time, leading to poor performance also in this case.

In order to avoid this undesirable behavior, many schemes have been proposed in the literature to dynamically adapt the CW to the current WLAN conditions. This approaches can be classified as either centralized or distributed mechanisms. On one hand, centralized approaches [10–14] are based on a single node (the Access Point) that periodically computes the set of MAC layer parameters to be used and signals this configuration to all stations. On the other hand, with distributed approaches [15–21] each station independently computes its own configuration.

However, these previous works[4] suffer from at least one of the following key limitations: *(i)* they are based on heuristics and therefore lack the mathematical foundations to guarantee optimal performance; *(ii)* they rely on functionality that is not available with existing wireless devices, requiring modifications of their hardware and/or firmware; *(iii)* their performance has not been assessed with real deployments, and therefore lack experimental evidences gathered from scenarios with channel impairments and implementation constraints.

In contrast to the previous proposals, in this thesis we develop analytical models of the WLAN performance, derive the optimal point of operation of the IEEE 802.11 protocol

[3] The description of the IEEE 802.11 EDCA mechanism is provided in Sec. 2.1

[4] Details about these proposals are provided in Sec. 2.2.

in terms of throughput and delay, and propose a centralized and a distributed adaptive algorithm, which are sustained by mathematical foundations from single-/multi-variable control theory. We show that these algorithms are able to drive the WLAN to its optimal point of operation and have the additional key advantage over existing approaches of being compliant with the IEEE 802.11 standard, as they can be implemented by current devices without introducing any modifications into their hardware and/or firmware.

1.1 Summary of Thesis Contributions

The contributions of this thesis are summarized as follows. First, we conduct an analysis of the WLAN saturation throughput and we derive the collision probability of an optimally configured WLAN. Based on this analysis, we propose a novel adaptive algorithm, the *Centralized Adaptive Control (CAC)* [22, 23], which dynamically adjusts the *CW* configuration of IEEE 802.11 stations with the goal of maximizing the overall throughput performance of the wireless network. Compared to the existing schemes, our proposal is fully compatible with the IEEE 802.11 standard, since the dynamic adjustment is based only on observing successfully received frames at the Access Point (AP). *CAC* is based on a well established scheme from discrete-time control theory, namely the *Proportional Integrator* (PI) controller. By conducting a control theoretic analysis of the system we tune the parameters of the PI controller to achieve a good tradeoff between stability and speed of reaction to changes.

Second, we propose an analytical model of the WLAN performance under video traffic, used to derive the optimal point of operation of EDCA with real-time sources. Based on this analysis we extend *CAC* to dynamically adjusts the *CW* configuration of the WLAN with the goal of minimizing the average delay, which results in a better quality of experience (QoE) of the video traffic [24]. In addition to being standard complaint and having mathematical foundations that guarantee optimal performance, the algorithm supports graceful degradation of video flows by implementing a priority based dropping policy, in line with the efforts of IEEE 802.11aa Task Group for robust streaming of audio-video transport streams.

Third, we propose a distributed approach to the optimal configuration of 802.11 WLANs, which shares the same goal of maximizing the overall performance as the centralized scheme [25]. The key novelty of the proposed *Distributed Adaptive Control (DAC)* algorithm is that it is sustained by foundations from the multivariable control theory field. In particular, the proposed algorithm implements a standard PI controller at each station, which uses only locally available information to drive the collision probability in the WLAN to the optimal value. The main advantages of the proposed algorithm over existing distributed approaches are the following: *(i)* its analytical foundations guar-

antee convergence and stability while ensuring a quick reaction to changes, *(ii)* it is standard-compliant as it only relies on functionality available with existing cards, and *(iii)* in contrast to existing schemes, which modify the contention parameters of all stations upon congestion, our algorithm only acts on those stations that are contributing to congestion, providing stations that are not contributing to congestion with a better delay performance.

We undertake a thorough simulation study to evaluate the proposed algorithms and compare their performance against the standard IEEE 802.11 mechanism, as well as previous adaptive approaches. As a benchmark for assessing the performance of our algorithms we also consider the static optimal configuration obtained with the algorithm we proposed in [26]. Note that, although the solution in [26] considers a more general scenario with different traffic types, its limitation lies within the fact that it requires a priori knowledge of the number of stations and the specific requirements of the applications, which involves additional signaling between the stations and the AP, while the algorithms proposed in this thesis do not rely on such information. The results of the evaluation show that *(i)* the proposed schemes outperform substantially both the standard 802.11 mechanism and existing proposals in terms of throughput, *(ii)* they provide a better delay performance than the previous adaptive schemes, and *(iii)* the configuration of the parameters of the PI controllers employed is adequate, as with other settings, the system either becomes unstable or reacts too slowly to changes.

Finally, we present our experiences gained with implementing the two adaptive algorithms and demonstrate that they can be easily deployed with unmodified existing hardware. We provide a detailed description of the implementation of the proposed mechanisms, which run as user space applications, relying on standardized system calls to estimate the contention level in the WLAN and to dynamically adjust the CW. We also give insights on the differences between the theoretical and practical implementations of the algorithms, which arose with the inherent limitations of the real devices, and prove the feasibility of utilizing these algorithms with existing commercial off-the-shelf (COTS) hardware and open-source device drivers. By conducting exhaustive experiments in a medium-scale testbed, we evaluate the performance of our proposals under non-ideal channel effects and different traffic conditions. Additionally, we compare the performance of our algorithms against the default 802.11 mechanism to identify those scenarios where a network deployment can benefit from using such adaptive algorithms [27].

1.2 Thesis Overview

The thesis is structured as follows. In Chapter 2, we summarize the operation of the IEEE 802.11 protocol and discuss the related work. In Chapter 3, we first analyze the

throughput of a WLAN operating with data stations and present the designed centralized algorithm (CAC), which maximizes the total throughput by dynamically adapting the EDCA configuration of the stations. Next, we conduct an analysis of the average delay and extend the centralized solution with the goal of maximizing the WLAN performance under real-time traffic. In Chapter 4, we undertake a distributed approach to the optimal configuration of WLANs and present the design of the DAC algorithm. Chapter 5 presents the results of the extensive set of experiments conducted in a real IEEE 802.11 testbed with prototype implementations of the designed algorithms. Finally, in Chapter 6, we conclude by discussing the implications that result from this thesis and future research directions.

Chapter 2

Background

IEEE 802.11 is the *de facto* standard currently used for providing users with wireless access to private networks and the Internet. In this chapter we first summarize the Enhanced Distributed Channel Access (EDCA) mechanism as specified by the revised version of the IEEE 802.11 standard [1] and then discuss the most relevant related works on WLAN performance modeling, adaptive MAC mechanisms and experimental evaluation studies, highlighting the key differences between previous research efforts and the contributions we present in this thesis.

2.1 IEEE 802.11 EDCA

EDCA regulates the access to the wireless channel on the basis of the *channel access functions* (CAFs). A station may run up to 4 CAFs, and each of the frames generated by the station is mapped to one of them. Once a station becomes active, each CAF executes an independent backoff process to transmit its frames.

A station with a new frame to transmit monitors the channel activity. If the medium is idle for a period of time equal to the arbitration interframe space parameter ($AIFS$), the CAF transmits. Otherwise, if the channel is sensed busy (either immediately or during the $AIFS$ period), the CAF continues to monitor the channel until it is measured idle for an $AIFS$ time, and, at this point, the backoff process starts. The arbitration interframe space takes a value of the form $DIFS + kT_e$, where $DIFS$ (the distributed interframe space) and T_e are constants dependent on the physical layer and k is a non-negative integer.

Upon starting the backoff process, a random value uniformly distributed in the range $[0, CW-1]$ is chosen and the backoff time counter is initialized with this number. The CW value is called the contention window, and depends on the number of failed transmissions

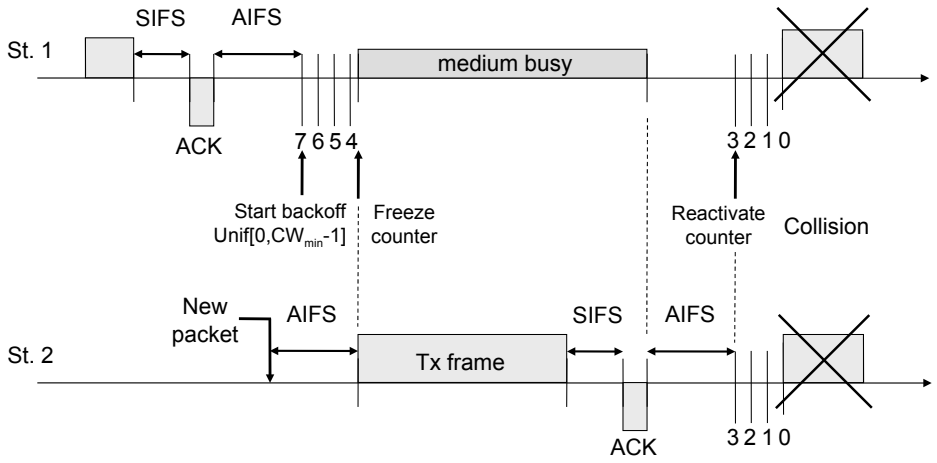

Figure 2.1: Example of EDCA operation with 2 stations.

of a frame. At the first transmission attempt, CW is set equal to the minimum contention window parameter (CW_{min}).

As long as the channel is sensed idle, the backoff time counter is decremented once every empty slot time T_e. When a transmission is detected on the channel the backoff time counter is "frozen", and reactivated again after the channel is sensed idle for a certain period. This period is equal to $AIFS$ if the transmission is received with a correct Frame Check Sequence (FCS), and $EIFS-DIFS+AIFS$ otherwise, where $EIFS$ (the extended interframe space) is a physical layer constant.

As soon as the backoff time counter reaches zero, the CAF transmits its frame. A collision occurs when two or more CAFs start transmitting simultaneously. An acknowledgment (ACK) frame is used to notify the transmitting station that the frame has been successfully received. The ACK is immediately sent upon the reception of the frame, after a period of time equal to the physical layer constant SIFS (the short interframe space). The operation of EDCA is illustrated in Fig. 2.1.

If the ACK is not received within a time interval given by the $ACK\_Timeout$ physical layer constant, the CAF assumes that the frame was not received successfully. The transmission is then rescheduled by reentering the backoff process, which starts at an $AIFS$ time following the timeout expiry. After each unsuccessful transmission, CW is doubled, up to a maximum value given by the CW_{max} parameter. If the number of failed attempts reaches a predetermined retry limit R, the frame is discarded. In order to prevent duplicates, the IEEE 802.11 standard uses a retry bit R to mark those frames that are being retransmitted, i.e., the flag R is set to 0 on the first transmission attempt, and set to 1 on every other transmission (see Fig. 2.2).

In the case of a single station running more than one channel access function, if the

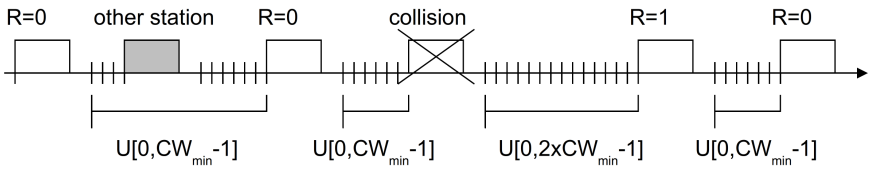

Figure 2.2: Retry flag marking upon collisions.

backoff time counters of two or more CAFs reach zero at the same time, a scheduler inside the station avoids the *internal collision* by granting the access to the channel to the highest priority CAF. The other CAFs of the station involved in the internal collision react as if there had been a collision on the channel, doubling their CW and restarting the backoff process.

After a (successful or unsuccessful) frame transmission, before sending the next frame, the CAF must execute a new backoff process. As an exception to this rule, the protocol allows the continuation of an EDCA transmission opportunity (TXOP). A continuation of an EDCA TXOP occurs when a CAF retains the right to access the channel following the completion of a transmission. In this situation, the station is allowed to send a new frame a SIFS period after the ACK corresponding to the completion of the previously transmitted frame. The period of time a CAF is allowed to retain the right to access the channel is limited by the transmission opportunity limit parameter ($TXOP\_limit$).

Hence, the behavior of a CAF depends on a number of parameters, namely CW_{min}, CW_{max}, $AIFS$ and $TXOP\_limit$. These are configurable parameters that can be set to different values for different CAFs. The CAFs are grouped by Access Categories (ACs), all the CAFs of an AC having the same configuration. In order to provide service differentiation the IEEE 802.11 standard recommends different values for the channel access parameters, listed in Table 2.1 for the case of IEEE 802.11b [3] physical (PHY) layer.[1] Apart from this recommended set of values, the standard also specifies that the Access Point (AP) can periodically broadcast through beacon frames (every 100 ms) the

Access category	$AIFS$	CW_{min}	CW_{max}	$TXOP$
voice	$DIFS$	8	16	3.264ms
video	$DIFS$	16	32	6.016ms
best-effort	$DIFS + Te$	32	1024	0
background	$DIFS + 5Te$	32	1024	0

Table 2.1: Default EDCA configuration for 802.11b PHY.

[1]Note that, with $TXOP = 0$ a station is only allowed to send one frame upon accessing the channel.

EDCA parameters to be used by all stations.

Following the above, when deploying an EDCA WLAN, the main challenge is the configuration of the contention parameters, as the standard set of *recommended* values remains the same for every scenario, regardless of, e.g., the number of stations or their traffic patterns, which leads to suboptimal performance in most circumstances. Next we discuss the previous research efforts in the literature that address the aforementioned challenge by proposing analytical models for the WLAN performance, mechanisms for the configuration of the EDCA parameters to improve performance, adaptive MAC schemes and experimental studies.

2.2 Related Work

Analytical models. Several analytical models of DCF/EDCA performance have been proposed in the literature [7, 9, 28–42]. Most of them [9, 28–35] are based on the assumption that all stations always have packets ready for transmission (commonly referred to as *saturation conditions*). While this assumption may be reasonable for data traffic, it does not hold for real-time traffic. On the other hand, previous models assuming non-saturated conditions have also been developed, considering different types of scenarios including Poisson arrival processes, voice traffic, video sources, etc. [36–42].

In contrast to the above proposals, our recent work of [26] does not make any assumption about the arrival process and allows for variable packet lengths, providing more comprehensive analyses of EDCA, which include generic traffic sources as well as the relevant metrics for data and real-time traffic (namely throughput, average and standard deviation of the delay). In this thesis we leverage our data analysis in [26] and we build on the analytical model presented in [41] to develop control-theoretic mechanisms that optimize the total throughput and the average delay, respectively.

EDCA configuration proposals. As the 802.11 standard allows for the default MAC configuration to be changed, the challenge of tuning the EDCA parameters when the network conditions are foreknown has been addressed recently in the literature [9, 43–48]. The works of [9] and [45] are restricted to data traffic, while the proposals of [43] and [44] are restricted to voice traffic. On the other hand, the approaches developed in [46] and [47] consider two traffic types, voice and data, but do not account for other types. In contrast, the configuration recommended in [48] considers all types of traffic, but it is based on a heuristic and therefore does not guarantee optimal performance.

Centralized approaches. There has been a number of approaches that rely on a single node to compute the set of MAC parameters to be used in the WLAN [10–14]. The main drawbacks of these approaches are that they either are based on heuristics, thereby

lacking analytical support for providing performance guarantees [10–12], or they do not consider the dynamics of the WLAN under realistic scenarios [13, 14]. Moreover, some of these approaches [13,14] require to estimate the number of stations, which adds additional complexity to the APs that have limited computational resources, thus challenging their practical use.

Distributed approaches. Several works have proposed mechanisms that independently adjust the backoff operation of each stations in the WLAN [15–21,49]. A significant drawback of most of these algorithms is that they require substantial modifications to the hardware and/or firmware of the existing wireless cards. The approaches of [15,16] use as input low level data, which is currently not available with existing cards, and require modifying the CW on a per-packet basis, which is not possible with current interfaces, thus bringing substantial complexity. The work of [17] is based on control theory, but models the WLAN as a single variable system, and therefore assumes a simplistic scenario where all stations simultaneously join the WLAN. Furthermore, the proposals of [16,18,49] modify the contention algorithm of IEEE 802.11, which is not supported by current devices.

Implementation experiences and experimental studies. Very few schemes that address the optimization of the WLAN performance have been developed in practice [14, 50, 51]. While the idea behind Idle Sense [16] is fairly simple, its implementation [50] entails a significant level of complexity, introducing tight timing constrains that require programming at the firmware level. The work of [51] prototypes the approach of [20] in a small testbed with four stations. The main weaknesses are that the performance evaluation is only limited to simple network conditions and the solution modifies the IEEE 802.11 state machine. The work of [14] presents an experimental study on a medium-sized testbed to obtain the CW_{min} that achieves proportional fairness. However, similar to [51], the experiments are only performed under static conditions.

Key advantages of the proposed work. In contrast to the above mentioned approaches, the algorithms proposed in this thesis hold the following assets:

- CAC and DAC utilize input data readily available from existing cards and rely on standardized primitives for the CW configuration,
- The algorithms that compute the CW have relaxed timing constraints[2] and do not require any firmware level programming. Indeed, as reported in Chapter 5, our implementations have been realized entirely at the user space level and we have been able to deploy them with a relatively low effort,

[2]While the functionality of previous works impose tight constraints, being typically executed on a per-packet basis and hence handling timescales of hundreds of μs, the proposed CAC and DAC have relaxed timing constraints, as they are only executed with beacon frequency (i.e., every 100 ms).

- The configuration of the algorithms' parameters has been obtained analytically, which guarantees optimal performance. In contrast, previous approaches have obtained the configuration of some of their parameters either heuristically or empirically. The major drawback of such a parameter settings is that they cannot provide any guarantees on the performance of the algorithm for general scenarios; for instance, stability is not guaranteed by any of these approaches,
- In contrast to the previous works, we investigate the performance of the proposed CAC and DAC algorithms under a wide set of network conditions and provide valuable insights on their suitability for deployment in practical environments.

In the following chapters, we present in detail the proposed centralized and distributed adaptive algorithms, the analytical foundations upon which their design is based, and we thoroughly evaluate their performance by means of simulations, as well as by implementing them with COTS devices in a real IEEE 802.11 testbed, to illustrate their performance gains as compared to the previous works.

Chapter 3

Centralized Adaptive Control Algorithm

In this chapter we propose a novel centralized algorithm, which relies on analytical models of the WLAN operation and foundations from control theory to guarantee optimal performance for general scenarios. In the first part, we address the challenge of maximizing of the total throughput of the WLAN, when stations transmit data traffic and propose the *Centralized Adaptive Control (CAC)* algorithm, which dynamically adjusts the CW configuration of IEEE 802.11-based Wireless LANs to achieve this goal. For this purpose, we provide an analytical model of the IEEE 802.11 EDCA behavior, which we use to design the mechanism that tunes the CW_{min} with which stations contend to achieve the optimal operation. Second, we investigate the case in which stations transmit real-time traffic, and extend CAC with the goal of minimizing the average delay, and therefore provide end users with a better Quality of Experience (QoE) of video traffic. To this aim, we model the WLAN behavior under video traffic and compute its optimal point of operation in this scenario. Based on this analysis, the extended CAC-VI algorithm tunes the CW of the video stations to drive the wireless network to this optimal point and thereby minimizes the access delay.

3.1 Data Traffic Scenario

As discussed in Sec. 2.1 the contention window configuration recommended by the IEEE 802.11 standard [1] is statically set, independently of the number of contending stations, thus yielding poor performance in most scenarios. In particular, when there are many stations in the WLAN, it would be desirable to use large CW values, in order to avoid too frequent collisions, while with few stations smaller CWs would reduce the

channel idle time.

Following the above observation, many authors have proposed centralized approaches [10–14] that dynamically adapt the CW by estimating the number of active stations in the WLAN in order to improve the throughput performance. These mechanisms are based on a single node, the Access Point, that periodically computes and distributes the set of MAC layer parameters to be used by every station. Since these approaches are executed on the AP, they do not require any modifications at the stations, therefore have the advantage of being compatible with the IEEE 802.11 standard. However, because they are based on heuristics and lack analytical support, they do not guarantee optimal performance.

The novel adaptive algorithm that we propose shares the same goal of maximizing the overall throughput performance of the wireless network by adjusting the CW, but, in contrast to previous approaches, it benefits from the following key improvements:

1. It does not require estimating the number of active stations, as the dynamic adjustment is solely based on observing successfully received frames at the AP.

2. It is based on a well established scheme from discrete-time control theory, namely the *Proportional Integrator* (PI) controller [52], whose parameters we compute by conducting a control theoretic analysis of the system, to achieve a proper tradeoff between stability and speed of reaction to changes.

3.1.1 Throughput Analysis and Optimization

In this subsection we present a throughput analysis of an EDCA WLAN. Based on this analysis, we find the collision probability of an optimally configured WLAN, which is the basis of the CAC algorithm. We start by analyzing the case when all stations are saturated and consider later the case when some stations are not saturated.

Let us define τ as the probability that a saturated station transmits in a randomly chosen slot time. This can be computed according to [7] as follows:

$$\tau = \frac{2}{1 + W + pW \sum_{i=0}^{m-1} (2p)^i} \tag{3.1}$$

where W is the CW_{min}, m is the maximum backoff stage ($CW_{max} = 2^m CW_{min}$) and p is the probability that a transmission collides. In a WLAN with n stations, this is given by

$$p = 1 - (1 - \tau)^{n-1} \tag{3.2}$$

The throughput obtained by a station can be computed as follows

$$r = \frac{P_s l}{P_s T_s + P_c T_c + P_e T_e} \tag{3.3}$$

where l is the packet length, P_s, P_c and P_e are the probabilities of a success, a collision and an empty slot time, respectively, and T_s, T_c and T_e are the respective slot time durations.

The probabilities P_s, P_c and P_e are computed as

$$P_s = n\tau(1-\tau)^{n-1} \tag{3.4}$$

$$P_e = (1-\tau)^n \tag{3.5}$$

$$P_c = 1 - n\tau(1-\tau)^{n-1} - (1-\tau)^n \tag{3.6}$$

and the slot time durations T_s and T_c as

$$T_s = T_{PLCP} + \frac{H}{C} + \frac{l}{C} + SIFS + T_{PLCP} + T_{ack} + AIFS \tag{3.7}$$

$$T_c = T_{PLCP} + \frac{H}{C} + \frac{l}{C} + EIFS \tag{3.8}$$

where T_{PLCP} is the PLCP (Physical Layer Convergence Protocol) preamble and header transmission time, H is the MAC overhead (header and FCS), T_{ack} is the duration of the acknowledgment frame and C is the channel bit rate.

The above terminates our throughput analysis. We next address, based on this analysis, the issue of optimizing the throughput performance of the WLAN. To this aim, we can rearrange Eq. (3.3) to obtain

$$r = \frac{l}{T_s - T_c + \frac{P_e(T_e - T_c) + T_c}{P_s}} \tag{3.9}$$

As l, T_s, and T_c are constants, maximizing the following expression will result in the maximization of r,

$$\hat{r} = \frac{P_s}{P_e(T_e - T_c) + T_c} \tag{3.10}$$

Given $\tau \ll 1$, $\hat{r}$ can be approximated by

$$\hat{r} = \frac{n\tau - n(n-1)\tau^2}{T_e - n(T_e - Tc)\tau + \frac{n(n-1)}{2}(T_e - T_c)\tau^2} \tag{3.11}$$

The optimal value of τ, τ_{opt}, that maximizes $\hat{r}$ can then be obtained by

$$\left.\frac{d\,\hat{r}}{d\,\tau}\right|_{\tau=\tau_{opt}} = 0 \tag{3.12}$$

which, neglecting the terms of higher order than 2, yields

$$a\tau^2 + b\tau + c = 0 \tag{3.13}$$

with

$$a = -\frac{n^2(n-1)}{2}(T_c - T_e) \tag{3.14}$$

$$b = -2n(n-1)T_e \tag{3.15}$$

$$c = nT_e \tag{3.16}$$

Isolating τ_{opt} from the above yields

$$\tau_{opt} = \sqrt{\left(\frac{2T_e}{n(T_c - T_e)}\right)^2 + \frac{2T_e}{n(n-1)(T_c - T_e)}} - \frac{2T_e}{n(T_c - T_e)} \tag{3.17}$$

Given $T_e \ll T_c$, we finally obtain the next approximate solution for the optimal τ,

$$\tau_{opt} \approx \frac{1}{n}\sqrt{\frac{2T_e}{T_c}} \tag{3.18}$$

With the above τ_{opt}, the corresponding optimal collision probability is equal to

$$p_{opt} = 1 - (1 - \tau_{opt})^{n-1} = 1 - \left(1 - \frac{1}{n}\sqrt{\frac{2T_e}{T_c}}\right)^{n-1} \tag{3.19}$$

which can be approximated by

$$p_{opt} \approx 1 - e^{-\sqrt{\frac{2T_e}{T_c}}} \tag{3.20}$$

This implies that, under optimal operation with saturated stations, the collision probability in the WLAN is a *constant independent of the number of stations*. The key approximation in the design of our algorithm is to assume that, for all the cases where some of the stations are saturated and some are not, the optimal collision probability in the WLAN takes this same constant value.

With the above, we design *CAC* with the goal of driving the collision probability to this optimal value, by adjusting the WLAN configuration. Note that, since this a constant value, our algorithm does not need to know the number of stations in the WLAN, which constitutes a major advantage over existing proposals.

3.1.2 CAC Algorithm

We next present *CAC*, our adaptive algorithm; this algorithm runs at the AP and consists of the following two steps which are executed iteratively:

- During the period between two beacon frames (which lasts 100 ms), the AP measures the collision probability of the WLAN resulting from the current CW configuration.
- At the end of this period, the AP computes the new CW configuration based on the measured collision probability and distributes it to the stations in a new beacon frame.

Our algorithm uses a PI controller[1] to drive the WLAN to its optimal point of operation. The key advantage of using a PI controller is that it is simple to design, configure and implement with existing hardware. In the following, we explain how the CW configuration is adjusted using a control signal. We then analyze our system from a control theoretical standpoint, which requires linearizing the behavior of the WLAN. Finally, we use this analysis to adequately configure the parameters of the PI controller.

3.1.2.1 CW Configuration

Following the previous subsection, our goal is to adjust the CW parameters of EDCA (CW_{min} and CW_{max}) in order to force that the collision probability in the WLAN is driven to the value given by Eq. (3.20). Since the default CW values given by the IEEE 802.11 standard ($CW_{min}^{default}$ and $CW_{max}^{default}$) are typically too small, yielding a too aggressive behavior, in order to achieve optimal operation these CW parameters should be increased.

Following the above reasoning, our algorithm tunes the CW_{min} value, while keeping the default value for the maximum backoff stage, i.e.

$$CW_{max} = 2^m CW_{min} \tag{3.21}$$

where m is the maximum backoff stage of the default configuration.

In order to ensure that our algorithm never underperforms the standard default configuration by using overly small CW values, we force that the CW_{min} cannot take values smaller than the standard's default setting $CW_{min}^{default}$. In addition, we also force that CW_{min} cannot exceed $CW_{max}^{default}$. In the rest of the paper we assume that CW_{min} always takes values within these bounds and do not further consider this effect.

[1] We note that previous works have successfully employed a PI controller to address performance issues in communication networks [53, 54].

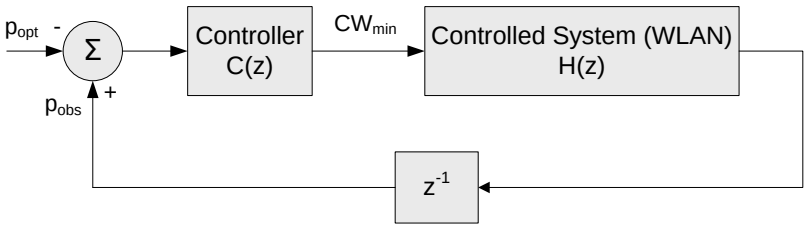

Figure 3.1: Control system.

3.1.2.2 Control System

From a control theoretic standpoint, our system can be seen as the composition of the two modules depicted in Fig. 3.1: the *controller* $C(z)$, which is the adaptive algorithm that controls the WLAN, and the *controlled system* $H(z)$, which is the WLAN itself.

Following the above, our control system consists of the following two modules:

- The PI controller module located at the AP, which takes as input an error signal e, which is the difference between the observed collision probability in the network p_{obs} and its desired value as given by Eq. (3.20), and computes the CW_{min}.
- The controlled module, which is the IEEE 802.11 EDCA WLAN system. As specified by the standard, the AP distributes the new CW configuration to the stations with every beacon frame. This configuration is obtained from the CW_{min} value given by the controller and Eq. (3.21).

The transfer function of the PI controller is given by [52]

$$C(z) = K_p + \frac{K_i}{z-1} \tag{3.22}$$

With the above transfer function, at every beacon interval t, the controller will take as input the estimated error signal $e = p_{obs} - p_{opt}$ and give as output the new CW value to be used by the contending stations.

$$CW_{min}[t] = K_p \cdot e[t] + K_i \sum_{k=0}^{t-1} e[k] \tag{3.23}$$

Note that implementing the above equation would be highly inefficient as it would require storing all the error samples from the past. A much more efficient implementation that only requires storing the previous values of CW_{min} and e is the following:

$$CW_{min}[t] = CW_{min}[t-1] + K_p \cdot e[t] + (K_i - K_p) \cdot e[t-1] \tag{3.24}$$

The estimation of the collision probability over a 100 ms period is performed at the AP as follows. Let R_0 be the number of frames received by the AP during this period with the retry bit unset, and R_1 be the number of frames received with the retry bit set. Then, if we assume that no frames are discarded due to reaching the retry limit, the collision probability p_{obs} can be computed as

$$p_{obs} = \frac{R_1}{R_1 + R_0} \tag{3.25}$$

The above expression is precisely the probability that the first transmission attempt of a frame from any station collides. The reasoning behind the equation is explained as follows. Let us consider that during a given observation period, N packets are transmitted in the WLAN. Assuming that no packets are dropped due to reaching the retry limit,[2] all these packets will eventually be successfully transmitted, either with the retry flag set (R_1) or unset (R_0). Hence a number of packets $N = R_0 + R_1$ will be observed. Assuming that transmission attempts collide with a constant and independent probability,[3] out of these N packets, in average Np_{obs} will collide in the first attempt. These packets will eventually be observed at a later attempt with the retry flag set, which yields $E(R_1) = Np_{obs}$. Then, if we divide the number of packets with the retry flag set by the total number of packets, we obtain (in average) the collision probability,

$$E\left(\frac{R_1}{R_0 + R_1}\right) = \frac{Np_{obs}}{N} = p_{obs} \tag{3.26}$$

which shows that Eq. (3.25) is accurate.

Note that with the above method, the AP can compute the probability p_{obs} by simply analyzing the header of the frames successfully received, which can be easily done with no modifications to the AP's hardware and driver.

3.1.2.3 Transfer Function Characterization

In order to analyze our system from a control theoretic standpoint, we need to characterize the Wireless LAN system with a transfer function that takes CW_{min} as input and gives the collision probability p_{obs} as output. Since the collision probability is measured every 100 ms interval, we can safely assume that the obtained measurement corresponds to stationary conditions and therefore the system does not have any memory. With this

[2]Note that the assumption that no packets are dropped due to reaching the retry limit is accurate. Indeed, the collision probability in an optimally configured WLAN is very low, which makes the probability of dropping a packet due to reaching the maximum allowed number of retransmissions (typically $R = 7$) negligible.

[3]The assumption that transmission attempts collide with a constant and independent probability has been widely used and shown to be accurate in the literature (see e.g. [7, 9, 28]).

assumption,

$$p_{obs} = 1 - (1 - \tau)^{n-1} \tag{3.27}$$

where τ is a function of CW_{min} as given by Eq. (3.1),

$$\tau = \frac{2}{1 + (CW_{min})(1 + p\sum_{i=0}^{m-1}(2p_{obs})^i)} \tag{3.28}$$

The above equations give a nonlinear relationship between p_{obs} and CW_{min}. In order to express this relationship as a transfer function, we linearize this relationship when the system is perturbed around its stable point of operation,[4] i.e.,

$$CW_{min} = CW_{min,opt} + \delta CW_{min} \tag{3.29}$$

where $CW_{min,opt}$ is the CW_{min} value that yields the optimal collision probability p_{opt} computed in Eq. (3.20).

With the above, the oscillations of the collision probability around its point of operation p_{opt} can be approximated by

$$p_{obs} \approx p_{opt} + \frac{\partial p_{obs}}{\partial CW_{min}} \delta CW_{min} \tag{3.30}$$

The above partial derivative can be computed as

$$\frac{\partial p_{obs}}{\partial CW_{min}} = \frac{\partial p_{obs}}{\partial \tau} \frac{\partial \tau}{\partial CW_{min}} \tag{3.31}$$

where

$$\frac{\partial p_{obs}}{\partial \tau} \approx n - 1 \tag{3.32}$$

and

$$\frac{\partial \tau}{\partial CW_{min}} = -\frac{2(1 + p_{obs}\sum_{i=0}^{m-1}(2p_{obs})^i)}{\left(1 + CW_{min}(1 + p_{obs}\sum_{i=0}^{m-1}(2p_{obs})^i)\right)^2} \tag{3.33}$$

Evaluating the partial derivative at the stable point of operation $p_{obs} = p_{opt}$, making the approximation $p_{opt} \approx (n-1)\tau_{opt}$ in Eq. (3.19) and using the expression for τ_{opt} given by Eq. (3.1), we obtain

$$\frac{\partial p_{obs}}{\partial CW_{min}} \approx -p_{opt}\tau_{opt} \frac{1 + p_{opt}\sum_{i=0}^{m-1}(2p_{opt})^i}{2} \tag{3.34}$$

[4] By linearizing the WLAN behavior around its stable point of operation, we accurately model the behavior of the transfer function around the point of operation, but we may not be accurate in regions far from this point. As a result, our analysis guarantees only local stability. A similar approach was used in [53] to analyze RED from a control theoretical standpoint.

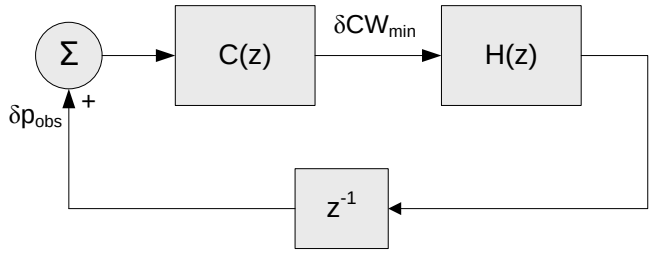

Figure 3.2: Linearized system.

If we now consider the transfer function that allows us to characterize the perturbations of p_{obs} around its stable point of operation as a function of the perturbations in CW_{min},

$$\delta P(z) = H(z)\, \delta CW_{min}(z) \tag{3.35}$$

we obtain from Eqs. (3.30) and (3.34) the following expression for the transfer function,

$$H(z) = -p_{opt}\tau_{opt}\frac{1 + p_{opt}\sum_{i=0}^{m-1}(2p_{opt})^i}{2} \tag{3.36}$$

Fig. 3.2 illustrates the above linearized model when working around its stable operation point, with:

$$\begin{cases} p_{obs} = p_{opt} + \delta p_{obs} \\ CW_{min} = CW_{min,opt} + \delta CW_{min} \end{cases} \tag{3.37}$$

Note that, as compared to the model of Fig. 3.1, in Fig. 3.2 only the perturbations around the stable operation point are considered.

3.1.2.4 Controller Configuration

We next address the issue of configuring the PI controller. We observe from Eq. (3.22) that the PI controller depends on the following two parameters to be configured: K_p and K_i. Our goal in the configuration of these parameters is to find the right tradeoff between speed of reaction to changes and stability, since bounded oscillation and fast response to disturbances are basic requirements in the design of closed-loop systems. To this aim, we use the *Ziegler–Nichols* rules [55] which have been designed for this purpose. These rules are applied as follows. First, we compute the parameter K_u, defined as the K_p value that leads to instability when $K_i = 0$, and the parameter T_i, defined as the oscillation period under these conditions. Then, K_p and K_i are configured as follows:

$$K_p = 0.4K_u \tag{3.38}$$

and

$$K_i = \frac{K_p}{0.85T_i} \tag{3.39}$$

In order to compute K_u we proceed as follows. The system is stable as long as the absolute value of the closed-loop gain is smaller than 1,

$$|H(z)C(z)| = K_p p_{opt} \tau_{opt} \frac{1 + p_{opt} \sum_{i=0}^{m-1} (2p_{opt})^i}{2} < 1 \tag{3.40}$$

which yields the following upper bound for K_p,

$$K_p < \frac{2}{p_{opt}\tau_{opt}(1 + p_{opt} \sum_{i=0}^{m-1} (2p_{opt})^i)} \tag{3.41}$$

Since the above is a function of n (note that τ_{opt} depends on n) and we want to find an upper bound that is independent of n, we proceed as follows. From Eq. (3.19), we observe that τ_{opt} is never larger than p_{opt} for $n > 1$ (note that for $n = 1$ the system is stable for any K_p). With this observation, we obtain the following constant upper bound (independent of n):

$$K_p < \frac{2}{p_{opt}^2(1 + p_{opt} \sum_{i=0}^{m-1} (2p_{opt})^i)} \tag{3.42}$$

Following the above, we take K_u as the value where the system may turn unstable (given by the previous equation),

$$K_u = \frac{2}{p_{opt}^2(1 + p_{opt} \sum_{i=0}^{m-1} (2p_{opt})^i)} \tag{3.43}$$

and set K_p according to Eq. (3.38),

$$K_p = \frac{0.4 \cdot 2}{p_{opt}^2(1 + p_{opt} \sum_{i=0}^{m-1} (2p_{opt})^i)} \tag{3.44}$$

The K_p value that makes the system become unstable yields $H(z)C(z) = -1$. With such a closed-loop transfer function, a given input value changes its sign at every time slot, yielding an oscillation period of two slots ($T_i = 2$). Thus, from Eq. (3.39),

$$K_i = \frac{0.4}{0.85p_{opt}^2(1 + p_{opt} \sum_{i=0}^{m-1} (2p_{opt})^i)} \tag{3.45}$$

which completes the configuration of the PI controller. The stability of this configuration is guaranteed by the following theorem.[5]

Theorem 1. *The system is stable with the proposed K_p and K_i configuration.*

[5]The proofs of the theorems are included in the Appendix.

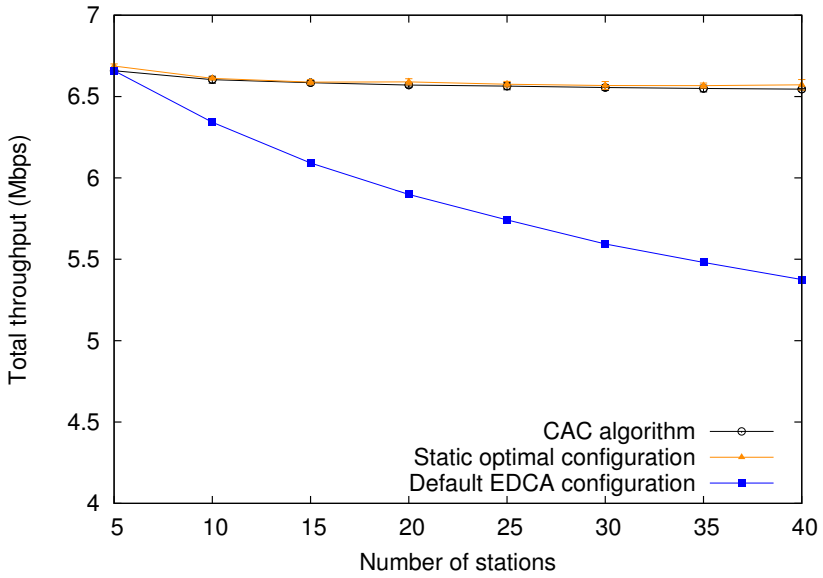

Figure 3.3: Throughput performance.

3.1.3 Performance Evaluation

In order to evaluate the performance of the proposed algorithm, we performed an exhaustive set of simulation experiments. For this purpose, we have extended the simulator used in [9, 56]. This is an event-driven simulator written in OMNeT++.[6] It implements independently for each station the protocol details and timing of the IEEE 802.11 EDCA MAC, and supports both saturated and non-saturated sources. We integrated into the simulator the proposed approach as well as the centralized solutions of [10,11]. The source code of the simulator and basic use instructions are available online at our OWSiM project page.[7]

For all tests, we used a payload size of 1000 bytes and the system parameters of the IEEE 802.11b physical layer [3]. For the simulation results, average and 95% confidence interval values are given (note that in many cases confidence intervals are too small to be appreciated in the graphs). Unless otherwise stated, we assume that all stations are saturated.

3.1.3.1 Throughput Performance

The main objective of the proposed algorithm is to maximize the throughput performance of the WLAN. To verify if the proposed algorithm meets this objective, we evaluated the total throughput obtained for different numbers of stations n. As benchmarks against which to assess the performance of our approach, we use the static optimal

[6] http://www.omnetpp.org
[7] http://enjambre.it.uc3m.es/~ppatras/owsim/

configuration given by [26] and the default EDCA configuration given in the IEEE 802.11e standard [6]. Note that the static optimal configuration method requires the knowledge of the number of active stations, which challenges its practical use.

The results of the experiment described above are given in Fig. 3.3. We can observe from the figure that the performance of the proposed algorithm follows very closely the static optimal configuration in terms of total throughput. In contrast, the default configuration performs well for a small number of stations but sees its performance substantially degraded as the number of stations increases. From these results, we conclude that the proposed algorithm maximizes the throughput performance.

3.1.3.2 Stability

One of the objectives of the configuration of the PI controller presented in Sec. 3.1.2.4 is guaranteeing a stable behavior of the system. In order to assess this objective, we plot in Fig. 3.4 the value of the system's control signal (CW_{min}) every beacon interval, for our $\{K_p, K_i\}$ setting with $n = 20$ stations. We can observe that with the proposed setting, CW_{min} performs stably with minor deviations around its point of operation. Had a larger setting for $\{K_p, K_i\}$ been used to improve the speed of reaction to changes, we would have experienced the situation of Fig. 3.5. For this case, with values of $\{K_p, K_i\}$ 20 times larger, the CW_{min} shows a strong unstable behavior with drastic oscillations. We conclude that the proposed configuration achieves the objective of guaranteeing a stable behavior.

3.1.3.3 Speed of Reaction to Changes

In addition to a stable behavior, we also require the PI controller to quickly react to changes in the WLAN. To assess whether this objective is fulfilled, we ran the following experiment. For a WLAN with 15 saturated stations, at $t = 80$ we added 15 more stations. We plot the behavior of the CW_{min} for our $\{K_p, K_i\}$ setting in Fig. 3.6 (label "K_p, K_i"). The system reacts fast to the changes in the WLAN, as the CW_{min} reaches the new value almost immediately. We have already shown in the Sec. 3.1.3.2 that large values for the parameters of the controller lead to unstable behavior. To analyze the impact of small values for these parameters, we plot on the same figure the CW_{min} evolution for a $\{K_p, K_i\}$ setting 20 times smaller (label "$K_p/20, K_i/20$"). With such setting, the system reacts too slow to changes of the conditions in the WLAN.

We conclude that, by means of the Ziegler–Nichols rules, we achieve a proper tradeoff between stability and speed of reaction to changes. To further validate this, in Fig. 3.7 we illustrate the time plot of the instantaneous throughput of one station, averaged over 1

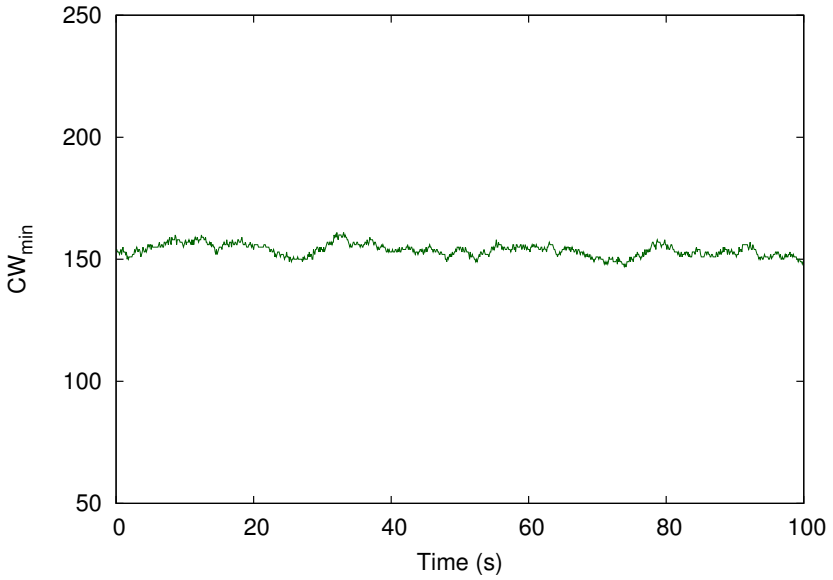

Figure 3.4: Stable configuration.

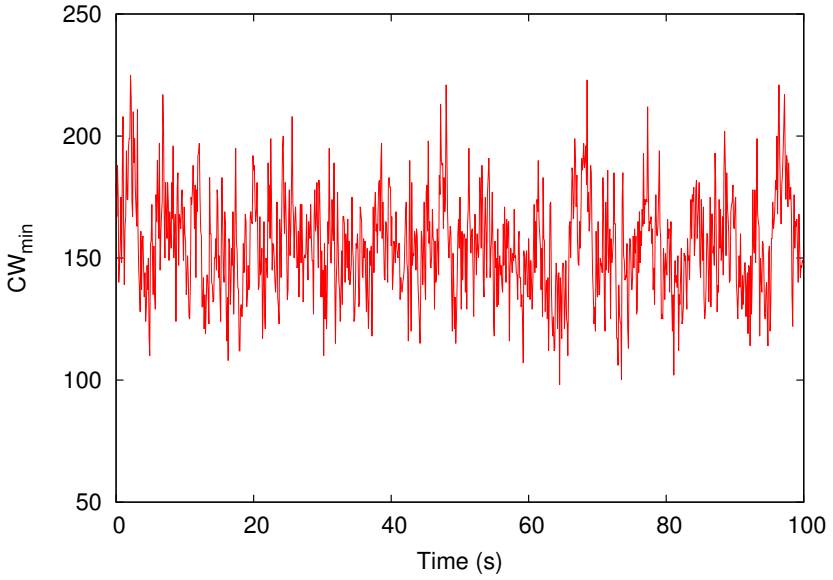

Figure 3.5: Unstable configuration.

second intervals, for the same experiment of Fig. 3.6. We can see from the figure that the system is able to provide stations with constant throughput (apart from minor oscillations due to the use of CSMA/CA), reacting almost immediately to changes.

3.1.3.4 Non-saturated Stations

Our approach has been designed to optimize performance both under saturation and non-saturation conditions, in contrast to the static optimal configuration shown previ-

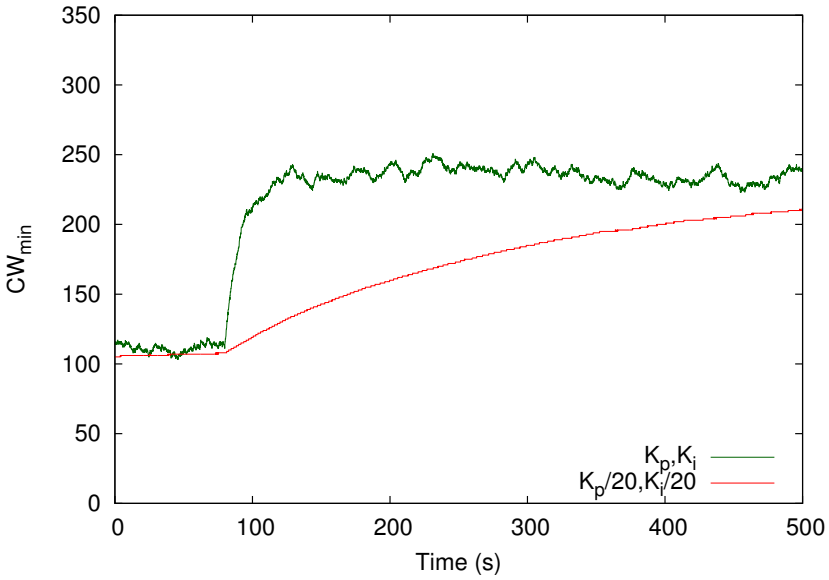

Figure 3.6: Speed of reaction to changes.

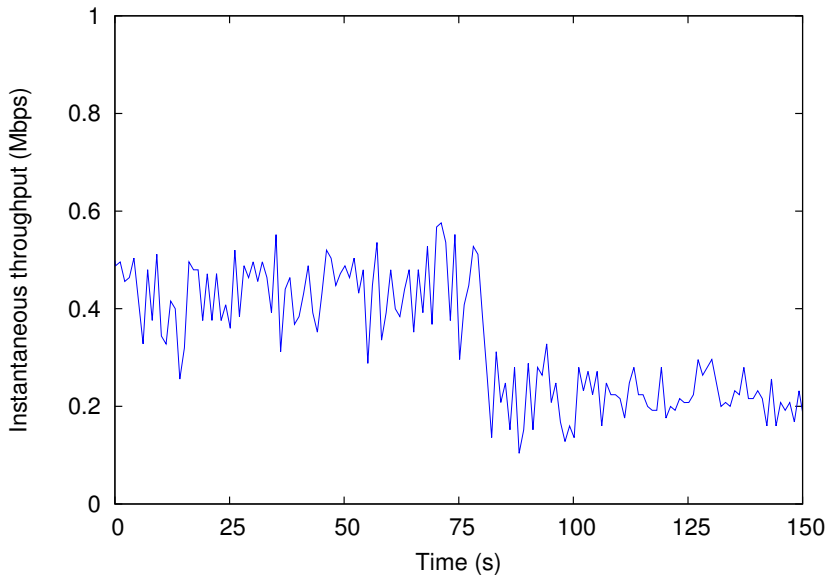

Figure 3.7: Instantaneous throughput.

ously, which is based on the assumption that all stations are saturated. In order to evaluate and compare the performance of the two algorithms when there are non-saturated stations in addition to saturated stations, we performed the following experiment. We had 5 saturated stations and a variable number of non-saturated stations in the WLAN. The non-saturated stations generated CBR traffic at rate of 100 Kbps. The total throughput resulting from this experiment is illustrated in Fig. 3.8. In this figure, we compare the performance of our approach against the static optimal configuration for data traffic given by [26], taking as input the total number of stations n present in the WLAN, regardless

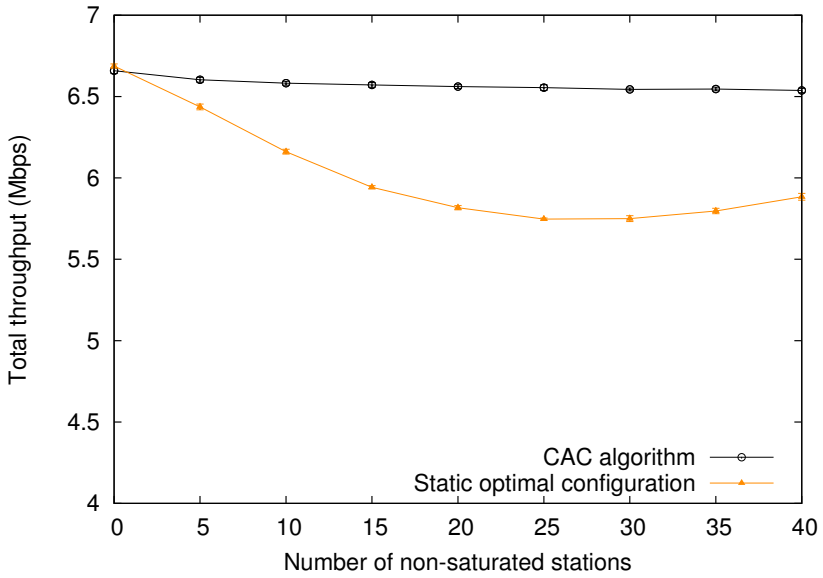

Figure 3.8: Non-saturated stations.

of whether they are saturated or not.

We observe from Fig. 3.8 that, with our approach, the total throughput remains approximately constant with values similar to the ones obtained for saturation conditions (Fig. 3.3), independently of the number of non-saturated stations. In contrast, the performance of the static optimal configuration decreases substantially as the number of non-saturated stations increases. This is due to the fact that the static optimal configuration considers that all stations are continuously sending packets and therefore uses too conservative CW values.

From the above results, we conclude that our algorithm achieves optimal performance also when non-saturated stations are present in the WLAN, in contrast to the static optimal configuration which sees its performance severely degraded as the number of non-saturated stations increases.

3.1.3.5 Bursty Traffic

In order to understand whether bursty traffic can harm the performance of the proposed algorithm, we repeated the experiment reported in Sec. 3.1.3.4 but with the non-saturated stations sending highly bursty traffic instead of CBR. In particular, in our experiment we used ON/OFF sources with exponentially distributed active and idle periods of an average duration of 100 ms each. The results of this experiment are depicted in Fig. 3.9.

We can see from these results that, similarly to Fig. 3.8, the proposed algorithm performs optimally, independent of the number of bursty stations, and substantially out-

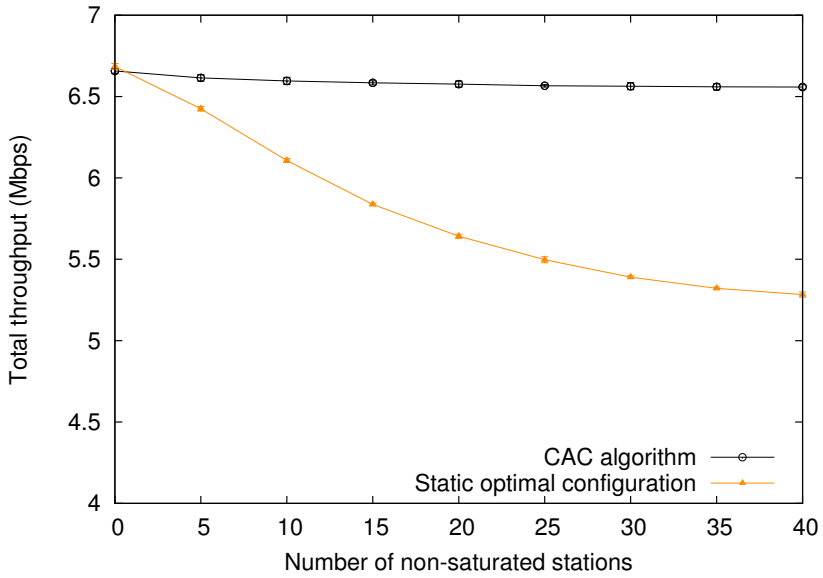

Figure 3.9: Bursty traffic.

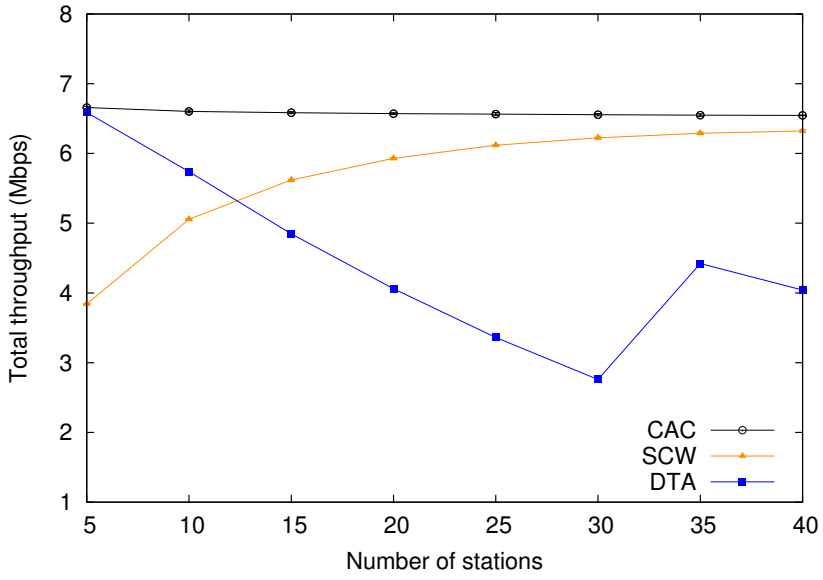

Figure 3.10: Comparison against other approaches.

performs the static optimal configuration. We conclude that our approach does not only work well under constant traffic but also under highly variable sources.

3.1.3.6 Comparison Against Other Approaches

The Sliding Contention Window (SCW) [10] and the dynamic tuning algorithm of [11] (hereafter referred to as DTA) are, like ours, centralized solutions compatible with the IEEE 802.11 standard that do not require hardware modifications. In what follows, we

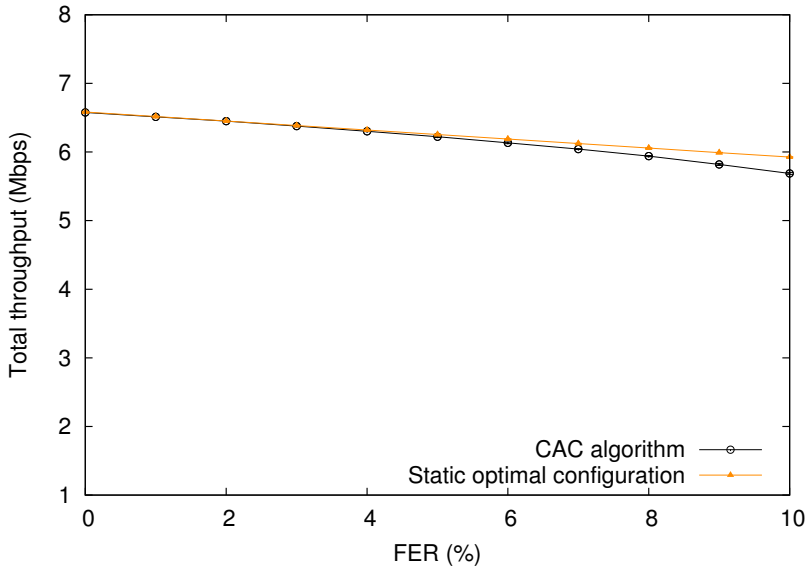

Figure 3.11: Impact of channel errors.

compare our solution against these centralized mechanisms.

Fig. 3.10 gives the total throughput performance of the different solutions for various numbers of stations. We observe that the proposed algorithm outperforms significantly both SCW and DTA. The reason is that our algorithm is sustained on the analysis of Sec. 3.1.1, which guarantees optimized performance, in contrast to SCW and DTA which are based on heuristics. In particular, SCW uses an algorithm to adjust CW_{min} that chooses overly large values, thereby degrading the performance. On the other hand, DTA sets the CW_{min} as an heuristic function of the number of stations yielding overly small values, which also results in degraded performance.

3.1.3.7 Impact of Channel Errors

Most of the adaptive mechanisms proposed for IEEE 802.11 WLANs do not consider the impact of channel errors [10–14]. However, channel errors may influence these mechanisms since they are wrongly interpreted as collisions, leading to an unnecessary increase of the CW and therefore to a suboptimal configuration.

In order to asses the impact of channel errors upon our approach we performed the following experiment. We varied the frame error rate (FER) from 0% to 10% for a scenario with $n = 20$ active stations in the WLAN. We compared the performance of our proposal against the static optimal configuration [26], which does not change the configuration upon failed transmissions and therefore uses always the optimal contention window value. The results of this experiment are illustrated in Fig. 3.11. We observe that for a realistic range of error probabilities (from 0% to 5%) the impact on throughput performance is

negligible. Moreover, even for very large error rates (up to 10%) the performance loss is very small. Note that current WLANs use link adaptation mechanisms, which guarantee small error rates by choosing a more robust modulation scheme upon detecting channel quality variations [57]. We conclude that with the proposed scheme errors have a minimal impact on the performance.

With the above, we complete the performance evaluation of the proposed *CAC* algorithm for data traffic. Next, we study the scenario in which stations transmit real-time traffic and extend our algorithm with the goal of improving the delay performance.

3.2 Real-Time Traffic Scenario

The EDCA mechanism is specifically intended to be used for real-time traffic, e.g., video, and, indeed, explicit recommendations for this traffic type are given. However, the use of the fixed set of recommended values for the EDCA parameters results in poor efficiency for most scenarios, as the optimal configuration of the channel access parameters depends on the WLAN conditions. Thus, when the WLAN is heavily loaded, the performance of real-time applications, and in particular the delay experienced by video traffic, is severely degraded. Following this observation, in this section we propose a novel adaptive approach to handle video traffic and optimize its performance. Our proposal embodies an extension to the *CAC* algorithm presented in Sec. 3.1, and is hereafter referred to as *CAC–VI*. *CAC–VI* dynamically adjusts the EDCA configuration of the IEEE 802.11 stations according to the observed conditions in the WLAN, with the goal of minimizing the delay experienced by video traffic. While we tailor our approach specifically to video traffic, we argue that its operation principles can be leveraged to any kind of real-time traffic.

To address the limitations in operating with video traffic, inherent to the standard's fixed configuration of the MAC parameters, previous works proposed different solutions to improve video performance by adapting the channel access protocol or the behavior of the codecs to the network conditions. These works can be classified as follows:

- *Cross-layer approaches* [58–60]. These approaches classify the frames of a layered-encoded video according to their relevance, and map them to different ACs. A major disadvantage of these works is their complexity, as they involve interactions between the application and the MAC layers, and moreover they either require specific video sources, or modifications of the protocol stack.
- *Non standard compliant approaches* [61–63]. These approaches have the key drawback of requiring additional changes to the MAC layer, e.g., modifying the backoff

behavior of the IEEE 802.11 stations, or replacing the MAC layer ARQ mechanism with an application level scheme, and therefore cannot be implemented with current WLAN cards.

- *HCCA compliant approaches* [64–66]. These approaches are compliant with the IEEE 802.11 specifications, but they are based on the centralized mechanism (namely HCCA), which, unlike the EDCA mechanism, has seen lesser deployments. Moreover, some of them [66] rely on feedback information from the clients, which is typically not available with current device drivers.

- *EDCA compliant approaches* [11,67–70]. These approaches rely on the EDCA standard mechanism and dynamically update the EDCA parameters and/or the video codec behavior based on the observed WLAN conditions. Their major drawback is that they are based on heuristics and lack analytical support, and hence do not guarantee optimized performance.

In contrast to the previous proposals mentioned above, our *CAC–VI* algorithm has the following key advantages:

1. It is tailored to video applications, as our goal is to optimize the delay performance, which results in a better QoE of the video traffic,

2. It is based on a well established analytical model of the MAC operation [41], which provides the foundations to guarantee optimal performance,

3. It requires no additional signaling and it is fully standard compliant, since the AP drives the WLAN to the optimal point of operation only by observing the behavior of the WLAN,

4. It guarantees simultaneously quick reaction to the changes in the network and stable operation by means of control theory.

5. It supports graceful degradation of video flows by implementing a priority based dropping policy, in line with the efforts of IEEE 802.11TGaa for robust streaming of audio video transport streams [71].

3.2.1 Analytical Model

In this subsection, we present the analytical model upon which *CAC–VI* is sustained. We first analyze the delay performance of a WLAN under video traffic and then, based on this analysis, we compute the collision probability that provides optimal delay performance. The proposed algorithm aims at driving the collision probability to this value.

3.2.1.1 Parameters Configuration

As discussed in Sec. 2.1, the operation of EDCA depends on four configurable parameters, namely $AIFS$, $TXOP$, CW_{max} and CW_{min}. Based on the following arguments, we fix the first three parameters when there is only video traffic present in the WLAN:

- $AIFS = DIFS$. We set this parameter to its minimum possible value, as otherwise additional time is unnecessarily lost after every transmission. Indeed, this parameter aims at providing differentiation between different traffic types and it is not needed when there is only one traffic type present in the WLAN.
- $CW_{max} = CW_{min}$. When all parameters are statically set, CW_{max} is typically larger than CW_{min}, so that after a collision the CW increases and thus the probability of a new collision is reduced. However, this is not necessary in our case, as our algorithm dynamically adjusts CW_{min}, so that the resulting collision probability corresponds to optimal operation. In addition, if we set CW_{max} larger than CW_{min}, the delay of the packets that suffer one or more collision drastically grows, which harms jitter performance. Experiments conducted with $CW_{max} = 2^6 \cdot CW_{min}$ and with $N = 25$ stations, report jitter values of up to 15 times larger than for a fixed CW setting, inline with this assumption.
- $TXOP = TXOP_{max}$. Considering the strict delay requirements of video traffic, it is desirable that, upon accessing the channel, all the waiting packets in the station's queue are transmitted in order to minimize their delay. To achieve this, we set the TXOP parameter to its maximum allowed value. Our simulation results included in Sec. 3.2.3.5 confirm that the best performance is achieved with this TXOP setting.

The above settings build on previous works [9,43] which show that the optimal operation of the WLAN can be achieved without utilizing the $AIFS$ and CW_{max} differentiation mechanisms, if an appropriate configuration of the CW_{min} is employed. Consequently, we have that the only parameter whose configuration is left open is CW_{min}. The rest of this subsection is devoted to the analysis of performance as a function of this parameter, while in the Sec. 3.2.2 we present the adaptive $CAC\text{–}VI$ algorithm that sets this parameter dynamically. To simplify notation, hereafter we refer to the CW_{min} parameter with CW.

3.2.1.2 Average Delay

In order to maximize video performance, our algorithm aims at finding the CW configuration that minimizes the average delay suffered by video frames. We next analyze the delay as a function of the CW.

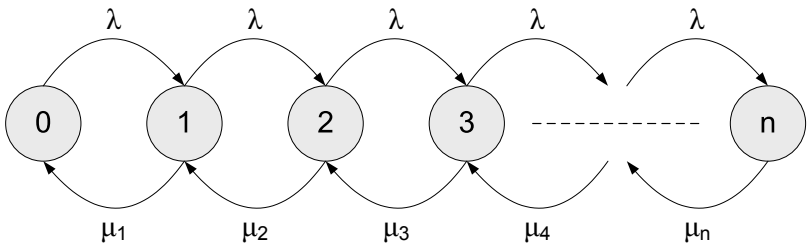

Figure 3.12: Markov chain model of the WLAN.

The key assumptions behind our analysis are:

- Following the findings of [39, 72], we neglect the probability that a station accumulates more than one video frame in its transmission queue.
- We assume that the aggregate arrivals follow a Poisson process. Considering a sufficiently large number of stations, and given their independence, this assumption is sustained by the Palm-Khintchine Theorem [73].
- We consider that access delays are exponentially distributed. This is supported by the observation that delay is mainly dominated by the number of attempts, which follows a geometric distribution, and that such a discrete distribution can be approximated by an exponential one in the continuous domain.

With these assumptions, the WLAN can be analyzed based on the Markov chain of Fig. 3.12, where state i represents the case where there are i backlogged stations with a video frame to transmit, λ is the aggregate arrival rate, computed as the individual arrival rate times the number of stations (denoted by n), and μ_i is the aggregate departure rate at state i.

To compute the μ_i's, we follow the assumption of [41] that the aggregate departure rate when there are i backlogged stations can be approximated by the departure rate of the WLAN with i saturated stations, which yields

$$\mu_i = \frac{r_i^{sat}}{L} \tag{3.46}$$

where L is the average length of a video frame and r_i^{sat} is the total throughput with i saturated stations. r_i^{sat} is computed following Sec. 3.1.1, but considering the now the length of a video frame instead

$$r_i^{sat} = \frac{P_s L}{P_s T_s + P_c T_c + P_e T_e} \tag{3.47}$$

where P_s, P_c and P_e are the probabilities that a slot time contains a successful transmission, a collision and is empty, respectively, and T_s, T_c and T_e are the corresponding average slot time durations. The probabilities are computed similarly to Eqs. (3.4)–(3.6) of Sec. 3.1.1, but considering only i backlogged stations, and the probability τ that a backlogged station transmits in a randomly chosen slot time, computed with Eq. (3.1) in the case of $CW_{min} = CW_{max}$, i.e.,

$$\tau = \frac{2}{CW + 1} \tag{3.48}$$

The average slot time durations T_s and T_c can be computed from the video frame length distribution as follows. Let P_l be the probability that the length of a video frame equals l. Then,

$$T_s = \sum_l P_l T_{s,l} \tag{3.49}$$

where $T_{s,l}$ is the duration of a transmission of a video frame of length l. Note that, since a video frame may be larger than the maximum size of a layer 2 (L2) frame, which we denote by l_{max}, it may need to be transmitted in several back-to-back L2 frames. Thus,

$$\begin{aligned} T_{s,l} &= (N-1)\left(T_{PLCP} + \frac{H + l_{max}}{C} + SIFS + T_{ack} + SIFS\right) \\ &+ T_{PLCP} + \frac{H + l - (N-1)l_{max}}{C} + SIFS + T_{ack} + DIFS \end{aligned} \tag{3.50}$$

where $N = \lceil l/l_{max} \rceil$ is the total number of L2 frames in which the video frame is divided, T_{PLCP} is the Physical Layer Convergence Protocol preamble and header transmission time, H is the L2 overhead (header and FCS), T_{ack} is the duration of the acknowledgment frame and C is the channel bit rate.

To compute T_c, we neglect the probability that more than two stations collide, similar to analysis of [9]. With this assumption, T_c can be computed as

$$T_c = \sum_l \sum_k P_l P_k max(T_{c,l}, T_{c,k}) \tag{3.51}$$

where $T_{c,l}$ is the duration of a slot time that contains a collision in which the largest colliding frame is of size l. Note that in case the video frame is larger than l_{max}, the collision is detected after the first L2 frame transmission and no further L2 frames are sent. Thus,

$$T_{c,l} = T_{PLCP} + \frac{H + min(l, l_{max})}{C} + EIFS \tag{3.52}$$

With the above, we can compute the μ_i values with Eq. (3.46). Once these values have been obtained, the next step is to calculate the state probabilities of the Markov chain. Let P_i be the probability that the Markov chain is in state i. From the balance

equations we have

$$P_i = P_{i-1}\frac{\lambda}{\mu_i} \tag{3.53}$$

and applying this recursively

$$P_i = P_0 \prod_{j=1}^{i} \frac{\lambda}{\mu_j} \tag{3.54}$$

By forcing that all P_i's add to 1, we have

$$P_0 = \frac{1}{1 + \sum_{i=1}^{n} \prod_{j=1}^{i} \frac{\lambda}{\mu_j}} \tag{3.55}$$

From Eqs. (3.54) and (3.55), we can compute all state probabilities P_i, and from the P_i's we then calculate the average number of backlogged stations,

$$n_b = \sum_{i=1}^{n} iP_i \tag{3.56}$$

Finally, by applying Little's formula [74], we obtain the average delay

$$D = \frac{n_b}{\lambda} \tag{3.57}$$

which terminates the delay performance analysis.

3.2.1.3 Optimal Collision Probability

We next compute the optimal collision probability that minimizes the average delay calculated previously. Our optimal collision probability computation is based on the observation that, in order to minimize the average number of backlogged stations (and therefore the delay, since the arrival rates of the Markov chain of Fig. 3.12 are fixed), we need to find the collision probability that maximizes the departure rates μ_i's.

We next compute the collision probabilities that maximize the different μ_i's. We first note that in state $i = 1$, where there is only one backlogged station, the collision probability is necessarily zero, since never more than one station will attempt to transmit in this state.

For $i > 1$ we proceed as follows. According to Eq. (3.46), maximizing μ_i is equivalent to maximizing r_i^{sat}. According to our previous analysis of Sec. 3.1.1 for saturated conditions, this maximization is achieved when the collision probability has the following approximate value:

$$p_{col} = 1 - (1 - \tau_{opt})^{i-1} = 1 - \left(1 - \frac{1}{i}\sqrt{\frac{2T_e}{T_c}}\right)^{i-1} \tag{3.58}$$

which can be approximated by

$$p_{col} \approx 1 - e^{-\sqrt{\frac{2T_c}{T_c}}} \tag{3.59}$$

Note that with the above approximations p_{col} does not depend on the number of backlogged stations i.

Therefore, we conclude that

- When a station transmits in state $i = 1$, the collision probability is always zero.
- When a station transmits in a state $i > 1$, the optimal collision probability is equal to p_{col}, which is a constant independent of i.

The combination of the above two leads to the following collision probability seen by a station in a WLAN under optimal operation:

$$p_{opt} = P(i = 1) \cdot 0 + P(i > 1) \cdot p_{col} = P(i > 1)p_{col} \tag{3.60}$$

where $P(i = 1)$ is the probability that a transmission by a station is attempted in state $i = 1$ and $P(i > 1)$ is the probability that a it is attempted in state $i > 1$.

The remaining challenge to obtain p_{opt} is the computation of $P(i > 1)$. We want to compute this probability by using only data that can be easily measured at the AP. To this aim, we make the following approximations: *(i)* we assume an infinite number of stations, and *(ii)* we neglect the protocol overhead on the μ_i's by taking $\mu_i = 1/T_s \ \forall i$. With these approximations,

$$P_i = \left(\frac{\lambda}{\mu}\right)^{i-1} P_1 \tag{3.61}$$

and

$$P(i > 1) = 1 - \frac{P_1}{\sum_{j=1}^{n} P_j} = \frac{\lambda}{\mu} \tag{3.62}$$

Finally, combining the above equations we obtain

$$p_{opt} = p_{col}\frac{\lambda}{\mu} \tag{3.63}$$

which terminates the analysis of the optimal collision probability. The above expression represents the theoretical optimal at which we would like our system to operate. We note that the expression obtained in Eq. (3.63) depends only on the parameters λ, μ and T_c which can be easily measured at the AP, as explained next.

3.2.2 CAC–VI Algorithm

In this subsection, we present our *CAC–VI* algorithm. This algorithm runs at the AP and, like *CAC*, consists of the following two steps that are executed iteratively:

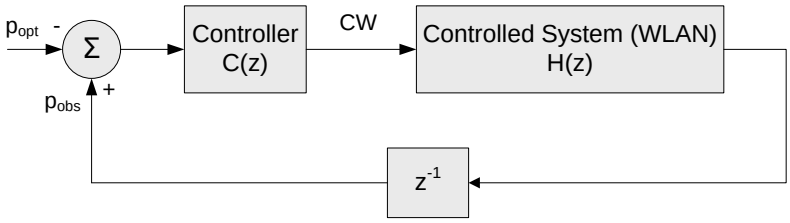

Figure 3.13: Control system.

- During each beacon interval (100 ms), the AP measures the collision probability of the WLAN resulting from the current CW configuration, the arrival rate λ and the departure rate μ.
- At the end of the period, the AP computes the new CW configuration based on the measured collision probability and distributes it to the stations in the new beacon frame.

LikeCAC, $CAC–VI$ relies as well on a PI controller to drive the WLAN to its optimal point of operation. In order to adequately configure the parameters of the PI controller, we proceed as in Sec. 3.1.2.2, first describing our system from a control theoretical standpoint, followed by linearizing the behavior of the WLAN.

3.2.2.1 Control System

Our system can be regarded from a control theoretic perspective as the composition of the two modules depicted in Fig. 3.13:

- The PI controller $C(z)$ is executed at the AP and implements the adaptive algorithm that controls the WLAN. The AP estimates the collision probability according to Eq. (3.25) and provides it to the controller, which takes as input the difference between the estimated collision probability and its desired value that yields the optimal performance as given by Eq. (3.63). With this input, the controller computes the CW value.
- The controlled system $H(z)$ is the WLAN system itself. As specified by the standard, the AP distributes the new CW configuration to the stations with every beacon.

In addition to p_{obs}, the AP also needs to compute the optimal collision probability p_{opt} as given by Eqs. (3.59) and (3.63), which requires the computation of λ, μ and T_c. These

parameters are estimated by the AP over each 100 ms period as follows: λ is measured by counting the number of video frames received during the period, μ is computed from the average length of the frames received during the period, and T_c is calculated by applying Eq. (3.51) to the received frames.

Note that with the above, the AP can measure everything simply analyzing the frames successfully received, which can be easily done with no modifications to the AP's firmware and hardware.

Based on the measurements taken by the AP, the controller adjusts the CW parameter to drive the collision probability to the optimal value. In order to provide a safeguard against too large and too small values of the CW, we force that CW can neither take values below $CW_{lb} = 16$ (which is the minimum standard recommendation for video traffic) nor above $CW_{ub} = 1024$ (which is the maximum CW value for best-effort traffic).

3.2.2.2 Transfer Function Characterization

Like in the case of CAC, in order to analyze our system from a control theoretic standpoint, we need to characterize the WLAN with a transfer function that takes the CW as input and gives the collision probability p_{obs} as output. Since the collision probability is measured every 100 ms interval, we can safely assume that the obtained measurement corresponds to stationary conditions and therefore the system does not have any memory. With this assumption and the analysis of Sec. 3.2.1,

$$p_{obs} = \sum_i P(i) \left(1 - (1-\tau)^{i-1}\right) \tag{3.64}$$

where $P(i)$ is the probability that a transmission is attempted at state i and τ is a function of the CW,

$$\tau = \frac{2}{CW + 1} \tag{3.65}$$

As in Sec. 3.1.2 we express the nonlinear relationship between p_{obs} and CW as a transfer function, by linearizing it when the system is perturbed around its stable point of operation,

$$CW = CW_{opt} + \delta CW \tag{3.66}$$

where CW_{opt} is the CW value that yields the optimal collision probability p_{opt} given by Eq. (3.63).

The oscillations of the collision probability around its point of operation p_{opt} can be approximated by

$$p_{obs} \approx p_{opt} + \frac{\partial p_{obs}}{\partial CW} \delta CW \tag{3.67}$$

The above partial derivative can be computed as

$$\frac{\partial p_{obs}}{\partial CW} = \frac{\partial p_{obs}}{\partial \tau}\frac{\partial \tau}{\partial CW} \tag{3.68}$$

Eq. (3.64) can be approximated by

$$p_{obs} \approx \sum_i P(i)(i-1)\tau \tag{3.69}$$

from which

$$\frac{\partial p_{obs}}{\partial \tau} \approx \sum_i P(i)(i-1) \tag{3.70}$$

Additionally, we have

$$\frac{\partial \tau}{\partial CW} = -\frac{2}{CW^2} \tag{3.71}$$

By taking the above two partial derivatives and using the approximation $\tau \approx 2/CW$, we obtain

$$\frac{\partial p_{obs}}{\partial CW} \approx -\sum_i P(i)(i-1)\frac{\tau^2}{2} \tag{3.72}$$

Since at the stable point of operation $\tau = \tau_{opt}$ we have from Eq. (3.58) that $p_{col} \approx (i-1)\tau_{opt}$ for $i > 1$, the above can be expressed as

$$\frac{\partial p_{obs}}{\partial CW} \approx -P(i>1)p_{col}\frac{\tau_{opt}}{2} \tag{3.73}$$

and combining it with Eq. (3.60) yields

$$\frac{\partial p_{obs}}{\partial CW} \approx -\frac{p_{opt}\tau_{opt}}{2} \tag{3.74}$$

If we now consider the transfer function that allows us to characterize the perturbations of p_{obs} around its stable point of operation as a function of the perturbations in CW,

$$\delta P(z) = H(z)\,\delta CW(z) \tag{3.75}$$

we obtain from Eqs. (3.67) and (3.74) the following expression for the transfer function,

$$H(z) = -\frac{p_{opt}\tau_{opt}}{2} \tag{3.76}$$

The above linearized model is depicted in Fig. 3.14. Note that, as compared to the model of Fig. 3.13, only the perturbations around the stable operation point are considered:

$$\begin{cases} p_{obs} = p_{opt} + \delta p_{obs} \\ CW = CW_{opt} + \delta CW \end{cases} \tag{3.77}$$

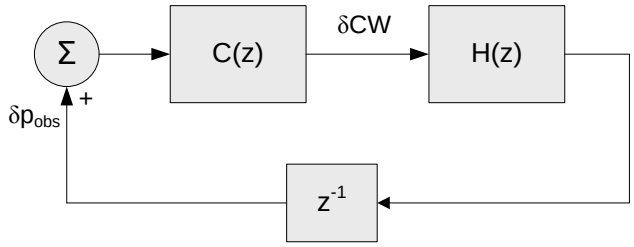

Figure 3.14: Linearized system.

3.2.2.3 Controller Configuration

In what follows we compute the $\{K_p, K_i\}$ configuration of the PI controller, whose transfer function was given in Eq. (3.22). To achieve a proper tradeoff between speed of reaction to changes and stability we use again the Ziegler–Nichols method [55], with K_u defined as the K_p value that leads to instability when $K_i = 0$, and T_i defined as the oscillation period under these conditions:

$$\begin{cases} K_p = 0.4K_u \\ K_i = \frac{K_p}{0.85T_i} \end{cases} \tag{3.78}$$

The system is stable as long as the absolute value of the closed-loop gain is smaller than 1,

$$|H(z)C(z)| = K_p \frac{p_{opt}\tau_{opt}}{2} < 1 \tag{3.79}$$

which yields the following upper bound for K_p,

$$K_p < \frac{2}{p_{opt}\tau_{opt}} \tag{3.80}$$

The above expression depends on τ_{opt}, which is not known by the AP. Since we want to find an upper bound that can be computed at the AP, we proceed as follows. From Eq. (3.58), we have that τ_{opt} is never larger than p_{col}. With this observation, we obtain the following tighter upper bound:

$$K_p < \frac{2}{p_{opt}p_{col}} \tag{3.81}$$

Following the above, we take K_u as the value where the system may turn unstable (given by the previous equation),

$$K_u = \frac{2}{p_{opt}p_{col}} \tag{3.82}$$

and set K_p according to Eq. (3.78). Thus,

$$K_p = \frac{0.4 \cdot 2}{p_{opt} p_{col}} \tag{3.83}$$

For the K_p value that turns the system unstable, the following holds:

$$H(z)C(z) = -1 \tag{3.84}$$

With such a closed-loop transfer function, a given input value changes its sign at every time interval, yielding an oscillation period equal to two intervals ($T_i = 2$). Consequently, from Eq. (3.78) we obtain

$$K_i = \frac{0.4}{0.85 p_{opt} p_{col}} \tag{3.85}$$

which completes the configuration of the PI controller. The stability of this configuration is guaranteed by Theorem 2.

Theorem 2. *The system is stable with the proposed K_p and K_i configuration.*

3.2.3 Performance Evaluation

We validated the *CAC–VI* algorithm by conducting an extensive set of simulations in order to assess the delay performance of the adaptive scheme, the robustness of the underlying analytical model and the configuration of the controller. For this purpose we have extended the simulator used in Sec. 3.1.3.

For all the experiments we have used the physical layer parameters of IEEE 802.11b [3]. In order to evaluate the performance of our adaptive scheme under video traffic we considered three of the most widely used codec types: H.263 [75], MPEG-4 [76] and H.264 [77]. The frame size distribution of the H.263 and MPEG-4 streams were extracted from the video traces of the films *Aladdin* and *Star Wars IV*, respectively, which are available from the Video Traces Library.[8] The H.263 video was a VBR encoded sequence with an unspecified target bitrate and a 20 fps average frame rate. The MPEG-4 trace had a fixed frame rate of 25 fps. We have also analyzed the operation of the adaptive algorithm

Codec	Type	Frame rate	Average bitrate	Average frame
H.263	VBR	20 fps	245 kbps	1535 Bytes
MPEG-4	VBR	25 fps	288 kbps	1440 Bytes
H.264	CBR	30 fps	300 kbps	1237 Bytes

Table 3.1: Characteristics of the video test sequences

[8] `http://trace.eas.asu.edu/`

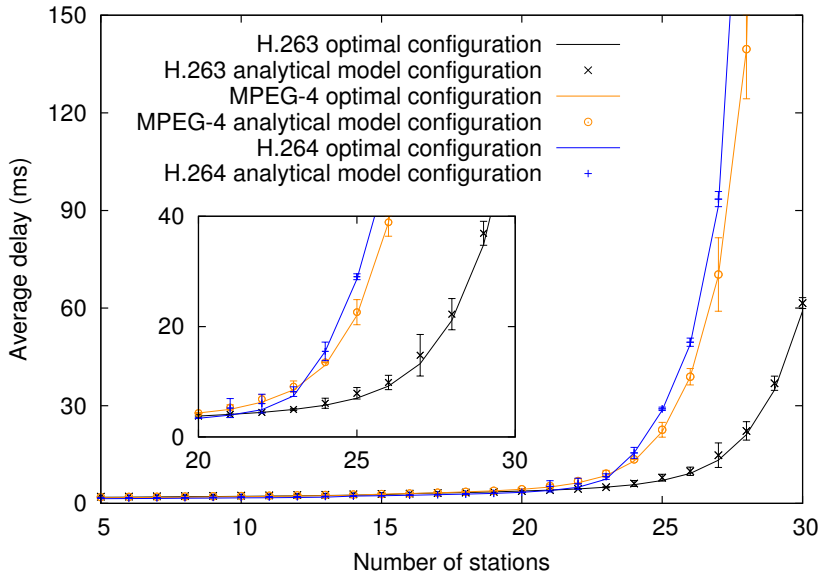

Figure 3.15: Validation of the delay model.

under CBR video, using one of the 30 fps encoded H.264 test sequences of [78]. The properties of these video sequences are summarized in Table 3.1. (Unless otherwise stated, in our simulations we consider that all active stations are transmitting video traffic and no other traffic types are present in the WLAN.) For the obtained results, averages and 95% confidence intervals are given.

3.2.3.1 Validation of the Analytical Model

We first validated the accuracy of the proposed analytical model upon which the adaptive algorithm is based. In particular, we verified that delay is minimized when the collision probability equals the optimal value given by Eq. (3.63), which is the basis of our analysis. To this aim, we simulated the average delay and collision probability from two different CW configurations:

- The CW value that yields a collision probability equal to the optimal collision probability given by our analysis (hereafter we refer to this configuration as the *analytical model configuration*).
- The CW value that gives the smallest average delay, obtained from an exhaustive search on all the possible configurations of the CW parameter (hereafter the *optimal configuration*).

Following our analysis of Sec. 3.2.1, the configuration resulting from the optimal collision probability should minimize the average delay, and therefore the delay resulting

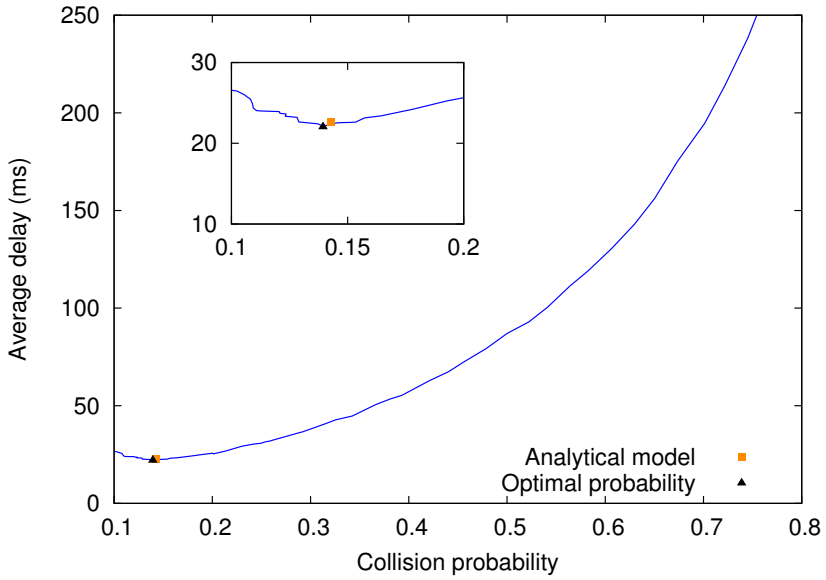

Figure 3.16: Optimal collision probability.

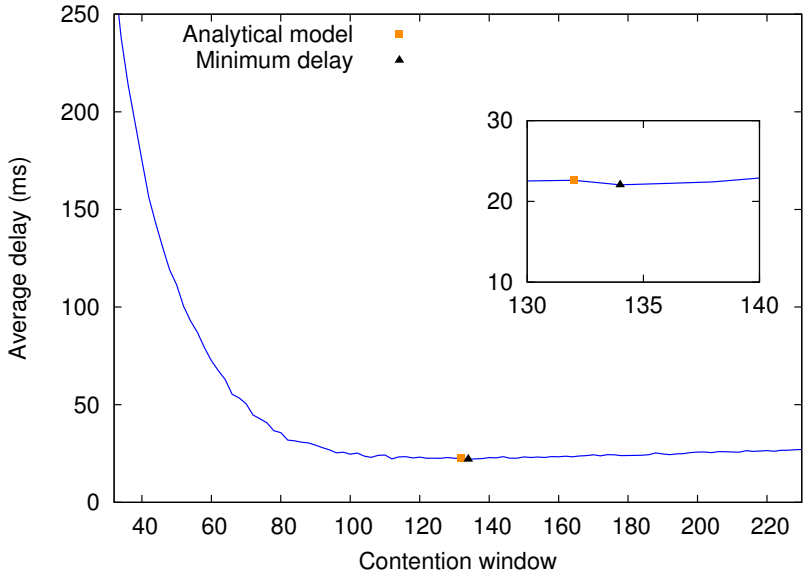

Figure 3.17: CW configuration.

from the two above configurations should be very similar. Fig. 3.15 shows the delay performance resulting from the two configurations for a varying number of stations and the different codecs considered. We observe that in all cases the two configurations provide a very similar delay performance, which validates our analytical model.

To show that the collision probability resulting from the optimal configuration is close to the optimal collision probability computed by our analysis, we plotted in Fig. 3.16 the average delay as a function of the collision probability, for a WLAN with 25 stations

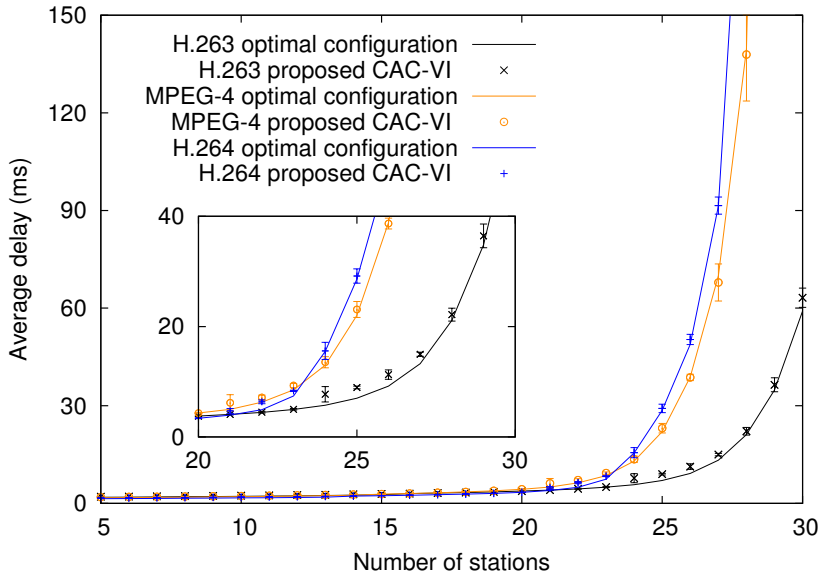

Figure 3.18: Delay performance of the proposed algorithm

sending each of them MPEG-4 video traffic. From the plot, we can see that the optimal collision probability given by our analysis (shown with a square) is very close to the collision probability for which the average delay is minimized (shown with a triangle).

To gain further insight into the CW configuration resulting from our analytical model, we plotted in Fig. 3.17 the average delay as a function of the CW for the same scenario as above with 25 stations and MPEG-4 traffic. We observe that the CW configuration resulting from our analytical model is very close to the optimal one that yields the minimum delay, which further validates our analysis.

3.2.3.2 Adaptive Algorithm Performance

The main objective of our adaptive algorithm is to minimize the average delay of the WLAN. In order to validate that this objective is met, we compared the delay performance of a WLAN which implements our adaptive algorithm against the optimal configuration resulting from the search performed previously. Results are depicted in Fig. 3.18. We observe that the proposed mechanism achieves a delay performance almost identical to the minimum given by the optimal configuration, regardless of the codec used. We conclude that our algorithm fulfills its main objective of minimizing the delay for any video traffic pattern.

Note that the optimal configuration against which we compare our approach is the result of an exhaustive search and requires a priori knowledge of the number of active stations and their traffic pattern, which challenges its practical use. In contrast, the

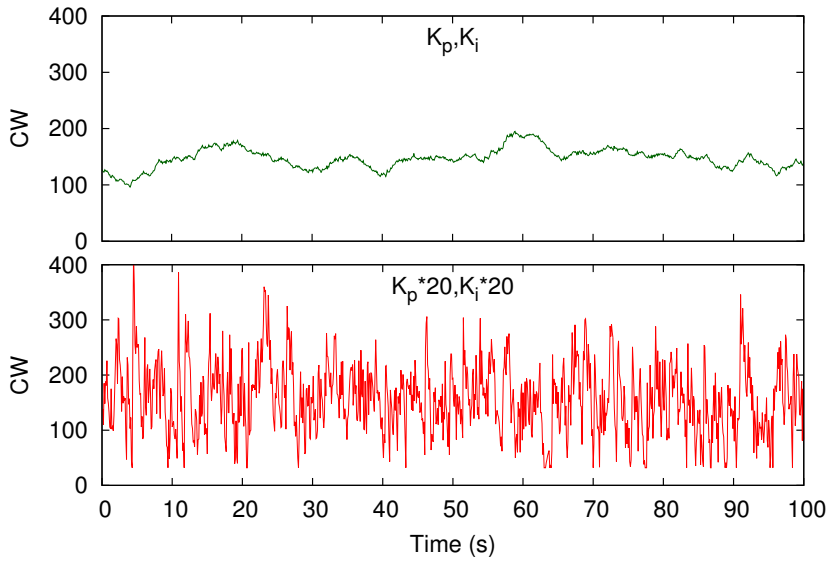

Figure 3.19: Stability evaluation.

adaptive algorithm that we propose does not require any kind of a priori knowledge since it adjusts the WLAN configuration based only on the measurements taken by the AP.

3.2.3.3 Stability

One of the objectives of the configuration of the PI controller presented in Sec. 3.2.2.1 is to guarantee stable behavior of the system. To validate whether this objective is met, we analyzed the evolution of the CW (our control signal) with our $\{K_p, K_i\}$ setting and for a larger configuration of these parameters, in a WLAN with 25 stations, each sending MPEG-4 video traffic. From the results given in Fig. 3.19 we observe form this figure that with the proposed configuration (label "K_p, K_i"), the CW only has minor deviations around its stable point of operation, while if a larger setting is used (label "$K_p * 20, K_i * 20$"), the CW has a strong unstable behavior with drastic oscillations. We conclude that the proposed configuration achieves the objective of guaranteeing stability.

3.2.3.4 Speed of Reaction to Changes

The other objective of the designed PI controller is to react quickly to changes in the WLAN. To verify whether this objective is fulfilled, we ran the following experiment. We had a WLAN with 20 active stations sending MPEG-4 traffic and, at $t = 20$ s, we added 5 more stations. We plot the evolution of the CW for our $\{K_p, K_i\}$ setting in Fig. 3.20 (label "K_p, K_i"). The system reacts fast to the changes on the WLAN, as the CW reaches the new value almost immediately.

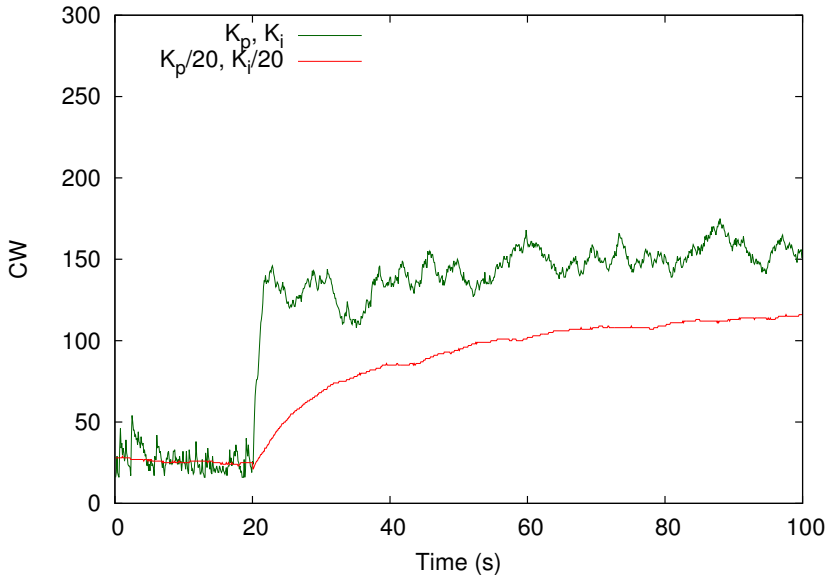

Figure 3.20: Speed of reaction to changes.

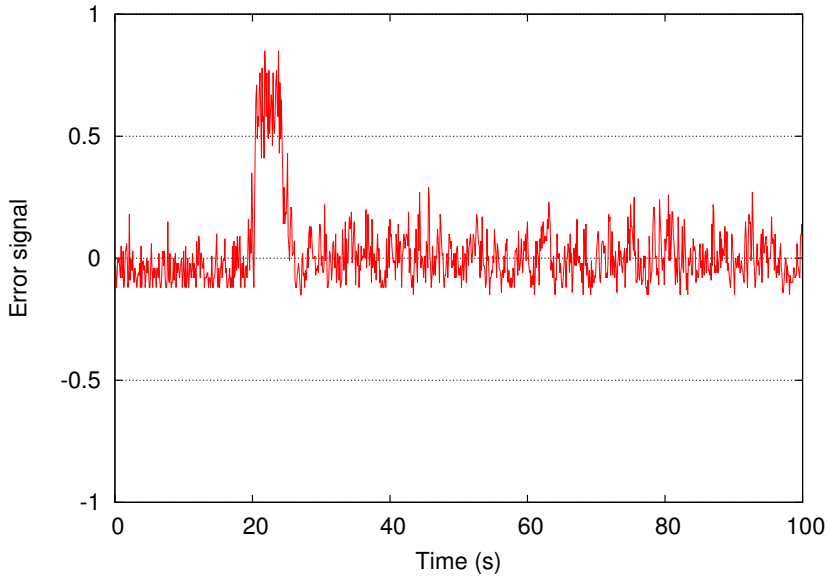

Figure 3.21: Time evolution of the error signal.

We have already shown that large values for the parameters of the controller lead to unstable behavior. To analyze the impact of small values for these parameters, we plot on the same figure the behavior of the CW for a $\{K_p, K_i\}$ setting 20 times smaller (label "$K_p/20, K_i/20$"). We observe that with such setting the system reacts too slow to the changes of the conditions on the WLAN.

In order to validate that the WLAN is operating around its optimal point of operation and our designed system has no steady state error, we plot in Fig. 3.21 the error signal

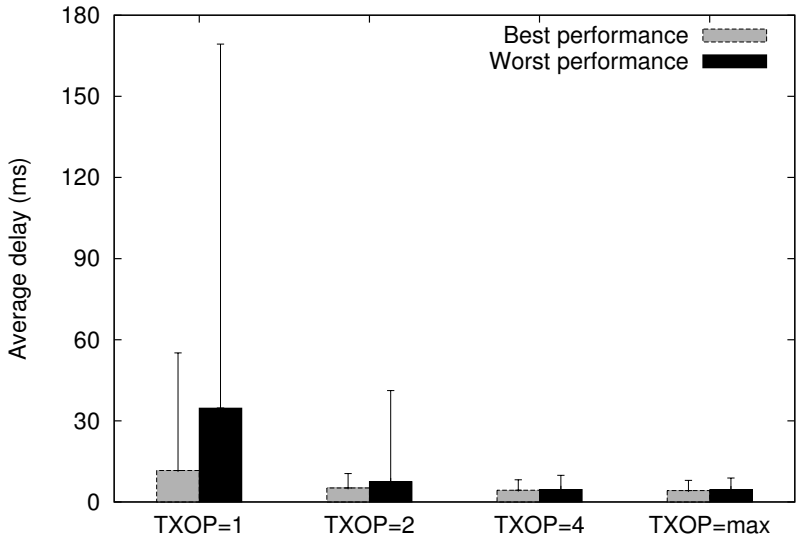

Figure 3.22: Delay performance for different TXOP.

fed to the PI controller for the same experiment. We observe that the error signal exhibits small variations around the zero value and is able to return rapidly to this state upon a change in the network conditions at time $t = 20$ s. We conclude that the designed PI controller indeed yields zero steady state error and is able to drive the collision probability in the WLAN to the optimal value.

3.2.3.5 Impact of TXOP Setting

In order to verify whether the large setting for the TXOP parameter used by our algorithm could introduce fairness issues, potentially due to having stations with buffered frames retaining the access to the medium for a long period of time, we studied the impact of the TXOP on the average and standard deviation of the delay, as experienced by individual stations running *CAC–VI*. For this purpose, we considered a scenario with 20 nodes sending MPEG-4 traffic and plotted in Fig. 3.22 the values of these metrics as experienced by the best and worst performing station for different values of the TXOP parameter. We show that, with our setting ($TXOP = TXOP_{max}$), stations are provided with almost the same average delay performance (plotted with bars), while the standard deviation (depicted with lines) is kept small. In contrast, for smaller TXOP values, the stations are experiencing significantly different average delay values, while the increased delay jitter further harms the performance. More specifically, with a low TXOP setting, some of the stations would experience average delays that are up to 3 times larger than the values experienced by others, thus incurring severe fairness issues. Additionally, the performance of the stations is further degraded by significantly high jitter values provided

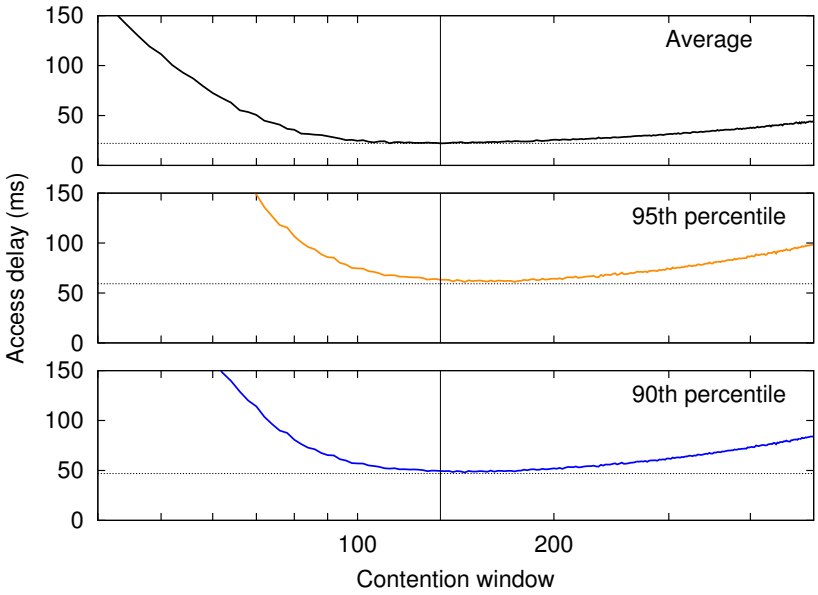

Figure 3.23: Average, 90$^{\text{th}}$ and 95$^{\text{th}}$ percentiles of the access delay.

with lower TXOP values.

3.2.3.6 Delay Distribution

Although we have shown that our algorithm minimizes the average delay, it is also relevant to analyze the probability distribution of the delay samples given by our configuration. For this purpose we evaluated the 90$^{\text{th}}$ and 95$^{\text{th}}$ percentiles of the access delay as a function of the CW. As shown in Fig. 3.23, the CW configuration provided by our algorithm not only minimizes the average delay, but also holds 90$^{\text{th}}$ and 95$^{\text{th}}$ percentiles very close to the minimum values. From this, we conclude that the proposed scheme does not only minimize the average delay but also the distribution of the delay.

3.2.3.7 Total Delay

So far we have evaluated delay performance in terms of access delay, i.e., the time elapsed since a station starts contending until it successfully accesses the channel. This delay coincides with the total delay experienced by video frames if frames do not accumulate in the transmission queue and are transmitted in different accesses, but it is slightly different from the total delay when several video frames are transmitted together. In order to show that achieving the minimum access delay also minimizes the delay experienced by the video frames, in Fig. 3.24 we compare the CW configuration that minimizes the channel access delay with the one that minimizes the per video frame delay. Since the two configurations are identical, we conclude that, by minimizing the access delay, our algorithm also minimizes the delay experienced by video frames.

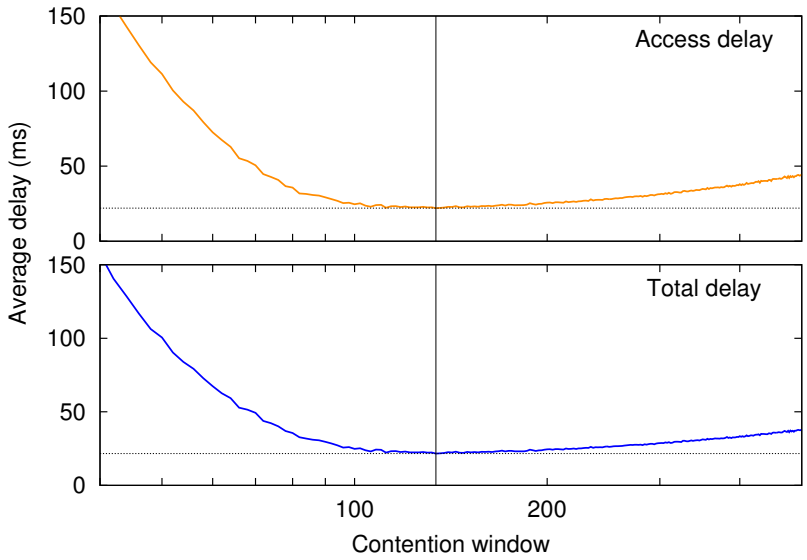

Figure 3.24: Total delay vs. access delay.

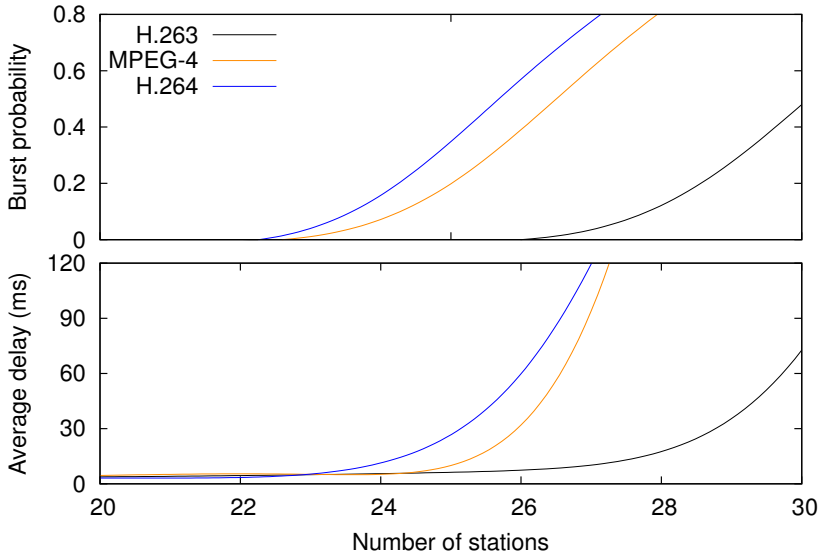

Figure 3.25: Burstiness vs average delay.

3.2.3.8 Support for Admission Control

In order to provide strict delay guarantees for video applications, one must limit the number of stations that join the WLAN when the network load becomes excessive. One possible admission control scheme that could be easily combined with the proposed *CAC–VI* approach is to measure at the AP the probability that a transmission contains more than one video frame (hereafter referred to as *burst probability*) and admit new stations only if this value is below a certain threshold. This is based on the observation that,

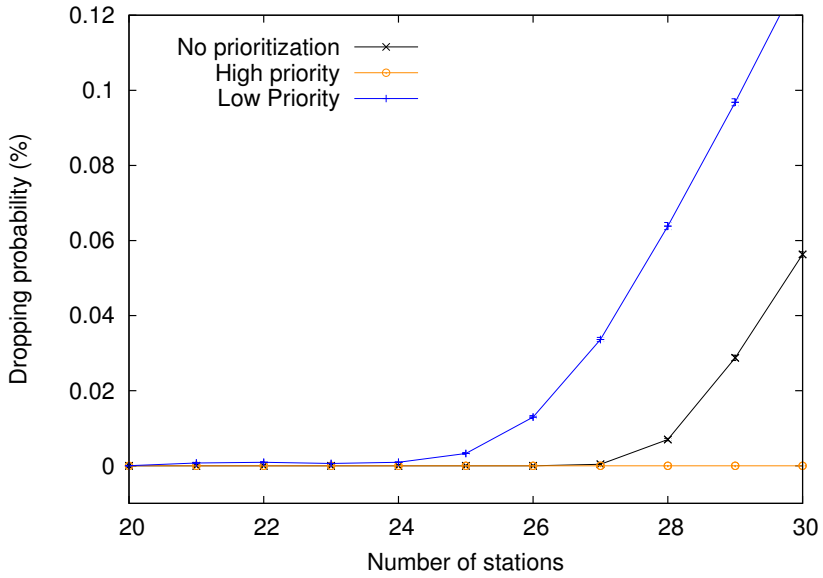

Figure 3.26: Support for graceful degradation of video flows.

the larger the delay, the more likely transmission queues will build up and, as a result, multiple frames are transmitted together by taking advantage of the TXOP parameter.

We next study the relationship between the delay and the burst probability in order to show that the burst probability is indeed a suitable metric to predict delay performance. To this aim, we plot in Fig. 3.25 both the burst probability and the average delay for an increasing number of stations and the three codecs considered. From the results depicted in the figure, it is evident that both variables are quite related.[9] For instance, if a threshold of 10% is used for the burst probability, delays keep well below 30 ms, which ensures appropriate video quality [80].

3.2.3.9 Graceful Degradation of Video Quality

The recently created IEEE 802.11TGaa [71] is standardizing a set of mechanisms to better support video streaming in WLANs. One of these mechanism consists of the so-called graceful degradation of video flows, whose purpose is to first discard less critical frames in case of congestion. Following these lines, in what follows we illustrate how our algorithm can be extended to support this new feature. This extension consists of introducing a queue size threshold Q_{th} which, if reached, triggers the discard of arriving frames marked with low priority. To assess the advantages of this enhancement, in Fig. 3.26 we measure *(i)* the dropping probability when there is no support for graceful degradation, and *(ii)* the dropping probability of high-priority and low-priority frames, respectively,

[9] In fact, the Pearson product-moment correlation coefficient is larger than 0.8 with 95% confidence for all codecs, as obtained by means of the Fisher's r to z transform [79].

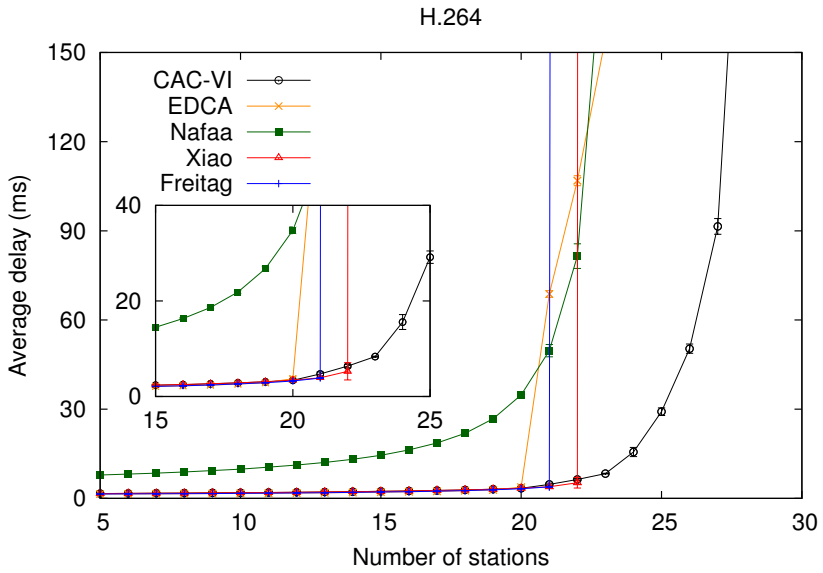

Figure 3.27: Comparison against other approaches: H.264 video.

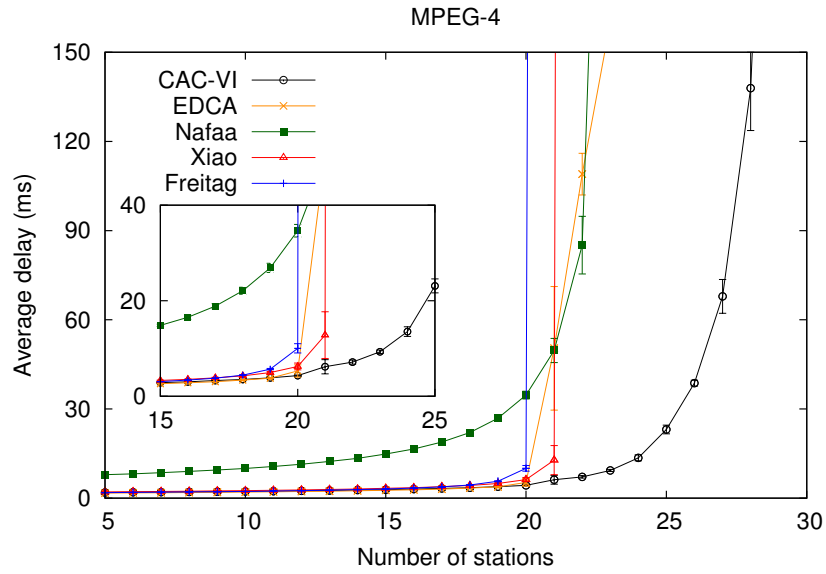

Figure 3.28: Comparison against other approaches: MPEG-4 video.

for the case of $Q_{th} = Q_{max}/2$. As the figure illustrates, this mechanism prevents high priority frames from being discarded even in case of large traffic loads, thereby showing its ability to support a graceful degradation of video traffic.

3.2.3.10 Comparison Against Other Approaches

In order to better assess the advantages of our proposal, we compared it against the following approaches: *(i)* the recommended configuration of IEEE 802.11e [6], *(ii)* the

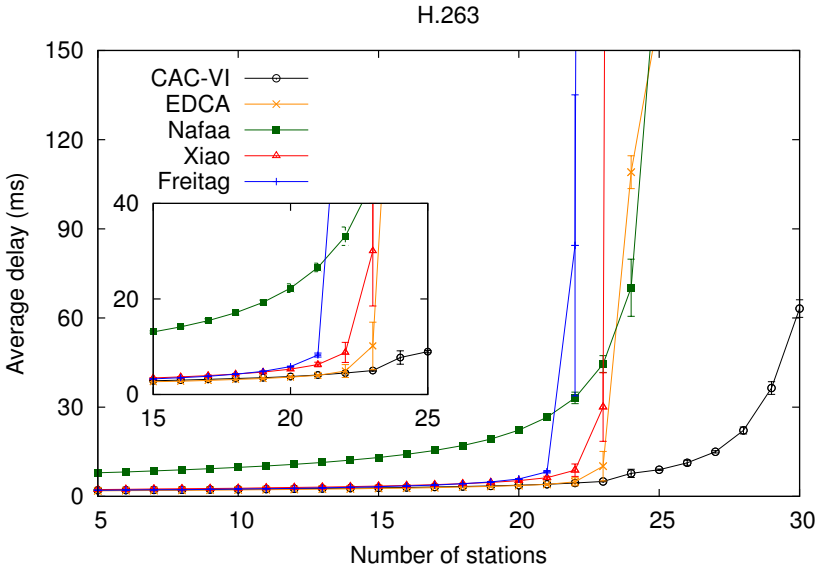

Figure 3.29: Comparison against other approaches: H.263 video.

Codec	Total throughput [Mbps]				
	CAC–VI	EDCA	Nafaa	Xiao	Freitag
H.263	7.365	5.890	5.892	5.636	5.396
MPEG-4	8.062	6.334	6.329	6.050	5.758
H.264	8.095	6.896	6.598	6.595	6.297

Table 3.2: Throughput evaluation.

scheme proposed by [63] (labeled as "Nafaa"), and *(iii)* two other standard compliant proposals, namely the one by [68] (labeled as "Xiao")[10] and the one in [11] (labeled as "Freitag"), respectively.

Figs. 3.27, 3.28 and 3.29 depict the average delay resulting from each of the above approaches as a function of the number of stations in the WLAN, and Table 3.2 shows the average total throughput that can be supported by each of the approaches while guaranteeing an average delay below the playback deadline of 150 ms.[11] We conclude from these results that our algorithm substantially outperforms all other approaches both in terms of delay and throughput.

Additionally, we compared the performance of our algorithm against the EDCA configuration and the other approaches in terms of perceptual quality of the reconstructed video, by evaluating the Mean Opinion Score (MOS) for different number of stations. For this purpose we assumed the same playback deadline of 150 ms. Based on this constraint, we considered that the frames which experience delays above this limit are discarded by

[10]The approach of [67] is very similar to the one of [68] and thus yields comparable performance.
[11]This is the maximum one-way delay as recommended by [80].

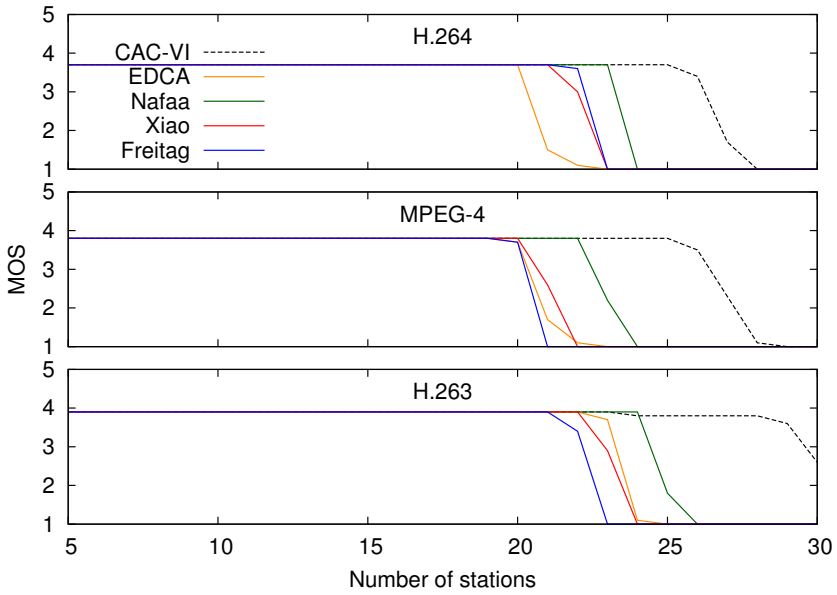

Figure 3.30: MOS evaluation.

the decoder. With the obtained packet loss ratio we computed the MOS of the received sequence according to the method given in [81]. The results are shown in Fig. 3.30. We conclude that our algorithm outperforms the standard recommended configuration as well as the other similar approaches, both in terms of average delay and perceptual video quality, being able to accommodate a substantially larger throughput (approximately 20%).

3.2.3.11 Impact of Channel Errors

Since our algorithm estimates the collision probability by solely relying on the retry flags of the correctly received frames, it is not able to distinguish whether the retransmission were caused by collisions or channel errors. Channel errors may be wrongly interpreted as collisions, leading to an unnecessary increase of the contention window and therefore to a suboptimal configuration. In order to asses the impact of channel errors upon our approach we performed the following experiment. We varied the frame error rate (FER) up to 8%, which is the minimum radio performance imposed by the IEEE 802.11b [3] specification, to ensure satisfactory performance between equipment manufactured by different system vendors for different number of stations in the WLAN. We compared the performance of our proposal against the static optimal configuration obtained through numerical search, which does not change the configuration upon failed transmissions and therefore uses always the optimal contention window value. From the results illustrated in Fig. 3.31 we conclude that, with the proposed scheme, errors have a minimal impact on the performance.

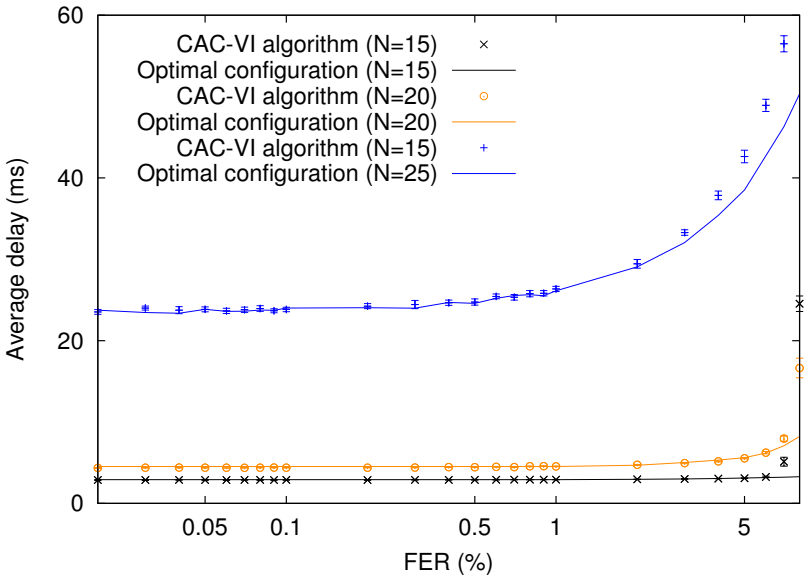

Figure 3.31: Delay performance under channel errors.

3.2.3.12 Mixed Traffic Scenario

Our algorithm focuses specifically on improving the delay when only video traffic is present in the WLAN. However, we argue that our approach can be extended to handle both video and data traffic. In order to show the feasibility of this extension, in the following we consider a scenario where data traffic coexists with the video transmissions. For this purpose, when computing the EDCA configuration for video, our extended algorithm also provides a CW_{BE} setting for the best effort AC, k times larger than the one used for video ($CW_{BE} = k \cdot CW_{VI}$, $k > 1$). To validate our proposal for such a mixed traffic scenario we conducted the following experiment. We considered a WLAN where 5 backlogged nodes send best effort traffic and an increasing number of stations are transmitting MPEG-4 video streams. For the best-effort category, our algorithm uses a TXOP parameter equal to one packet and a CW setting ten times larger than the one used by the video AC (i.e., $k = 10$) to ensure that video traffic is prioritized. As shown in Fig. 3.32, the presence of best effort traffic does not harm the delay performance of the videos significantly, since the throughput of the data traffic (shown in subplot) will be sacrificed in order to better accommodate the video flows as the number of stations increases. We conclude that with the proposed algorithm the video quality can be preserved even when data traffic coexists in the WLAN.

Additionally we aim to quantify the QoE improvements provided by our algorithm by means of a subjective evaluation process conducted with real users. For this purpose we utilize a prototype implementation of *CAC–VI* which we deployed on a 3-node testbed consisting of Debian Linux kernel 2.6.26 laptops equipped with Atheros AR5212 cards

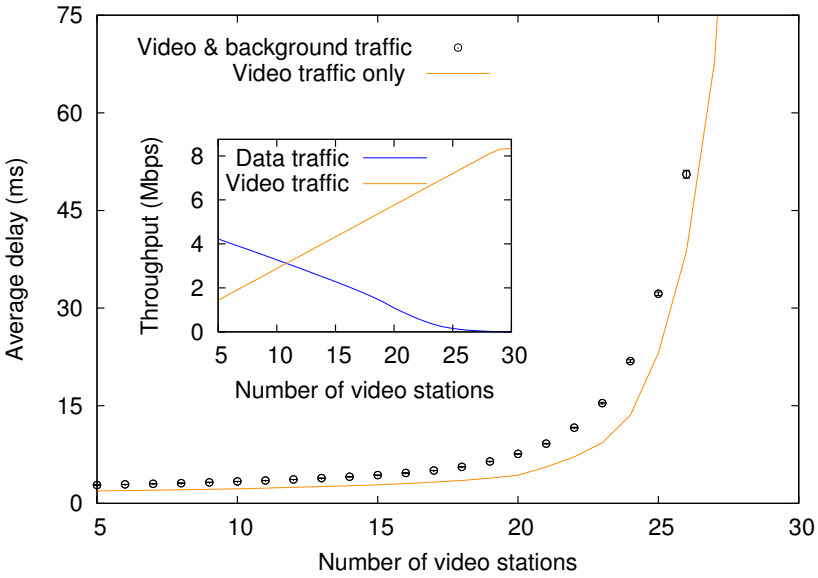

Figure 3.32: Delay performance under mixed traffic.

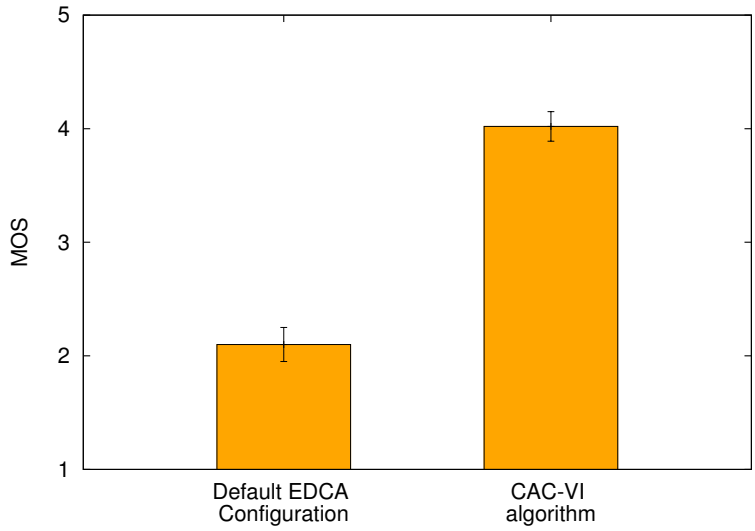

Figure 3.33: MOS evaluation with real users.

operating in 802.11b mode, one acting as an AP and the other two as stations.

Our algorithm runs at the AP, while the two stations stream video towards the AP using the VLC multimedia framework[12] and, simultaneously, transmit saturated UDP flows, thus resembling best effort data transfers. As test video sequences we have utilized a 2 Mbps MPEG-2 [82] encoded fragment of 30 seconds from an Ice Age 3 trailer.[13] We run 2 sets of experiments: on the first set we used our algorithm, while on the second one we

[12] http://www.videolan.org/vlc/
[13] http://www.iceagemovie.com/

Figure 3.34: Sample video frame with the default EDCA configuration.

Figure 3.35: Sample video frame with the proposed CAC–VI algorithm.

used the default EDCA configuration. In each case we launched 10 consecutive streaming sessions, recording the received sequences. A group of 20 people ranked subjectively the perceived quality of this set of sequences watched in random order. With these rankings we computed the MOS for the transmission with and without our algorithm, respectively. The results, shown in Fig. 3.33, prove that our algorithm is able to significantly improve the quality of the received video. We also provide in Figs. 3.34 and 3.35 two visual samples of the received video stream with the default EDCA configuration and the *CAC–VI* algorithm, respectively, which further show the ability of our proposal to protect video.

3.3 Summary

In this chapter, we have proposed a novel centralized adaptive algorithm for optimizing the performance of a WLAN. We first addressed the maximization of the total through-

put when the wireless nodes transmit data traffic and we designed the *CAC* algorithm, which relies on the observation that the collision probability in an optimally configured WLAN is approximately constant, independent of the number of stations. Second, we have analyzed a WLAN under video traffic and computed its optimal point of operation that minimizes delay. Based on this analysis we extended the centralized algorithm and proposed *CAC–VI*, which significantly improves QoE. Our approaches adaptively adjust the EDCA configuration to drive the collision probability to the optimal value as given by the throughput and delay analysis, respectively, thus maximizing performance under these scenarios.

The proposed centralized schemes are based on a PI controller, which has the key advantage of being simple to design, configure and implement with existing hardware. We achieve a proper tradeoff between stability and speed of reaction to changes by applying the Ziegler-Nichols rules to configure the parameters of the PI controllers. The key design features of our centralized approach are: *(i)* we do not require any a priori knowledge about the number of active sources or their traffic patterns, as the AP only needs to examine the successfully received frames, and *(ii)* the solution is fully compatible with the IEEE 802.11 EDCA specification, since it neither requires any modifications at the hardware nor at the firmware level. We have shown that the proposed *CAC* and *CAC–VI* substantially outperform the standard recommended configuration as well as other centralized adaptive solutions.

In the next chapter, we tackle the challenge of maximizing the WLAN performance from a distributed perspective, whereby each station independently adapts its MAC configuration, based on locally observed network conditions, to achieve this goal.

Chapter 4

Distributed Adaptive Control Algorithm

As compared to centralized schemes, distributed algorithms take a different approach to overtake the shortcomings of the standard's proposed EDCA configuration and improve the total throughput of a wireless network. With such mechanisms, each station independently computes its own configuration by observing the current WLAN behavior, thereby eliminating potential single point of failure problems and the need for additional signaling in non-infrastructure based topologies. As compared to the centralized mechanisms, an additional advantage of distributed approaches is that they can operate both under infrastructure and *ad-hoc* mode, which uses no Access Point.

In this chapter, we propose *Distributed Adaptive Control (DAC)*, a novel distributed algorithm that adaptively adjusts the CW configuration of the WLAN with the goal of maximizing the overall performance. The key novelty of the proposed scheme is that it is sustained by foundations from the multivariable control theory field. In particular, the proposed algorithm implements a standard PI controller at each station, that uses only locally available information to drive the collision probability in the WLAN to the optimal value that maximizes performance. The configuration the PI controllers' parameters is obtained by conducting a control theoretic analysis of the distributed system.

The main advantages of the proposed algorithm over existing distributed approaches [15–21], which we discussed in Sec. 2.2, are summarized as follows:

1. In contrast previous works which are mainly based on heuristics, DAC relies on mathematical foundations, which guarantee optimal operation, convergence and stability, while ensuring a quick reaction to changes.

2. As compared to the existing schemes that rely on non-standard capabilities or functionality that is not available with off-the-shelf devices, our mechanism is standard-

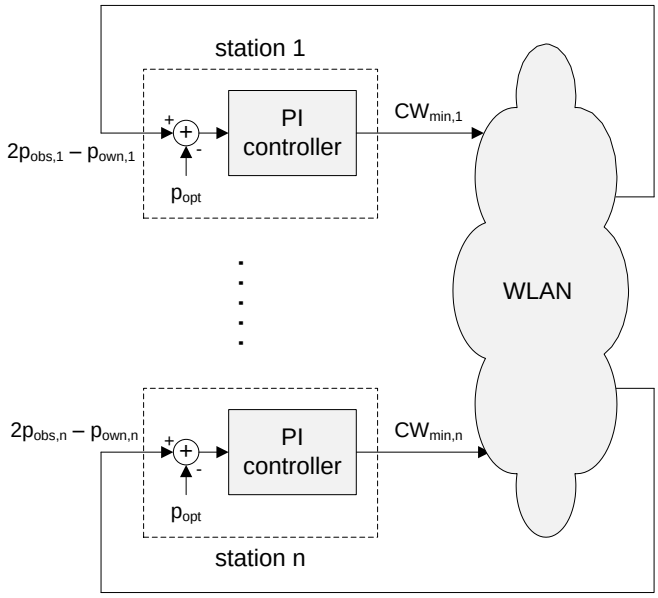

Figure 4.1: DAC Algorithm

compliant and can be implemented with existing cards, without requiring modifications of their hardware and/or firmware.

3. In contrast to all previous proposals, which modify the contention parameters of all stations upon congestion, our algorithm only acts on those that are contributing to congestion, providing stations that are not contributing to congestion with a better delay performance.

4.1 DAC Algorithm

In this section, we present the proposed *DAC* algorithm. *DAC* adjusts the CW_{min} parameter of each station with the goal of driving the WLAN to the optimal point of operation. To achieve the above goal, *DAC* uses a classical system from multivariable control theory [83] which is shown in Fig. 4.1. In this system, each station runs an independent controller that gives the CW_{min} value to be used by the station.

As it can be seen from Fig. 4.1, the PI controller of a station i takes as input the error signal e_i and gives as output the $CW_{min,i}$ configuration of the station. The choice of the error signal e_i is a critical part of the design of the *DAC* algorithm, as it drives the system behavior both under steady and transient conditions.

In steady conditions, a key requirement for the choice of e_i is that there exists a single stable point of operation that yields optimal performance. This requirement is analyzed in Sec. 4.2, where we show that the system reaches the optimal point of operation by driving the collision probability to a desired value.

In transient conditions, we set the following requirements on e_i:

- When the collision probability is far from its desired value, the error signal needs to be large in order to trigger a quick reaction towards the desired value.
- When the collision probability is around its desired value but stations do not share bandwidth fairly, the error should also be large in order to achieve a fair bandwidth sharing.
- In case of congestion, only the saturated stations should increase their $CW_{min,i}$, thus avoiding that the non-saturated stations (which are not contributing to congestion) are unnecessarily penalized.

In order to satisfy the above requirements, we take the error signal as the sum of two terms, such that each term contributes to fulfill some of the requirements described above. These two terms are carefully chosen, so that they do not cancel each other – this is guaranteed by Theorem 3 of Sec. 4.2, which proves that, under steady conditions, the system reaches a state where both components of the error signal are equal to 0.

The first term of the error signal is:

$$e_{collision,i} = p_{obs,i} - p_{opt} \tag{4.1}$$

where $p_{obs,i}$ is the probability that a transmission of a station different from i collides and p_{opt} is the desired value for the collision probability. This term ensures that, if the WLAN is operating at a different collision probability from the desired one, the error is large, achieving thus the first of the three requirements stated above.

The second term of the error signal is:

$$e_{fairness,i} = p_{obs,i} - p_{own,i} \tag{4.2}$$

where $p_{own,i}$ is the probability that a transmission of station i collides. This term ensures that if two stations do not share the bandwidth fairly due to having different $CW_{min,i}$'s, the error will be large. Indeed, a station with a small $CW_{min,i}$ transmits with a large probability, and therefore its $p_{obs,i}$ will be larger than $p_{own,i}$, yielding a large $e_{fairness,i}$. This fulfills the second requirement.

Additionally, the $e_{fairness,i}$ term also ensures that, in case of congestion, only the saturated stations increase their $CW_{min,i}$, which satisfies the last requirement stated

above. This is caused by the fact that saturated stations have a larger transmission probability; as a result, their $p_{obs,i}$ is larger and their $p_{own,i}$ smaller, which makes their $e_{fairness,i}$ larger.

The combination of Eqs. (4.1) and (4.2) yields the following error signal:

$$\begin{aligned} e_i &= e_{collision,i} + e_{fairness,i} \\ &= 2p_{obs,i} - p_{own,i} - p_{opt} \end{aligned} \tag{4.3}$$

where, as depicted in Fig. 4.1, the term $2p_{obs,i} - p_{own,i}$ corresponds to the feedback signal measured from the WLAN and p_{opt} is the reference signal, whose value is given in Sec. 4.2.

Having chosen the error signal as given by the above expression, the remaining key challenge for its computation is the measurement of the values of $p_{own,i}$ and $p_{obs,i}$. In particular, the challenge lies in measuring these values by using only functionality available in current wireless cards. To achieve this, we proceed as follows.

To compute the own collision probability at station i, $p_{own,i}$, we take advantage of the following statistics, which are readily available from wireless cards: the number of successful transmission attempts, denoted by T, and the number of unsuccessful attempts, F. $p_{own,i}$ is then computed by applying the following formula

$$p_{own,i} = \frac{F}{F+T} \tag{4.4}$$

The probability $p_{obs,i}$ cannot be computed following the above procedure, since with current hardware it is not possible to measure the unsuccessful attempts of other stations. Instead, we compute $p_{obs,i}$ using the same strategy of examining the retry flag of the frames successfully transmitted observed by station i, as discussed in Sec. 3.1.2.2.

With the above, each station i periodically measures $p_{obs,i}$ and $p_{own,i}$, and computes the error signal e_i from these measurements. This error signal is then fed into the controller, which triggers an update of $CW_{min,i}$. As a safeguard against too large and too small values of CW_{min}, when updating $CW_{min,i}$ we force that it can neither take values below a given lower bound nor above an upper bound. In particular, the values that we have chosen for the lower and upper bounds in this paper are the default CW_{min} and CW_{max} values used by the standard (for the 802.11g physical layer, these are 16 and 1024, respectively [5]).

Regarding the frequency with which the $CW_{min,i}$ is updated, we take the same approach used in *CAC* and update it every beacon interval. More specifically, we trigger the algorithm upon the reception of a beacon frame. The key advantages of this choice are the following:

- It ensures compatibility with existing hardware, since WLAN cards conforming to the IEEE 802.11 revised standard are able to update the configuration of the CW parameters with the beacon frequency.
- It is a simple way to ensure that all the stations update their configuration with the same pace.

As an exception to the above, if the number of samples used to compute $p_{obs,i}$ or $p_{own,i}$ at the moment of receiving the beacon frame is smaller than 20, the update is not triggered but deferred until the next beacon. The reason is to avoid that a too small number of samples induces a high degree of inaccuracy in the estimation of these parameters. In what follows, we assume that there are always enough samples available and updates are never deferred.

From the above description of DAC, it can be seen that the algorithm relies on p_{opt} as well as the parameters of the PI controller (namely K_p and K_i) [52]. The following two sections address the issue of properly configuring these parameters.

4.2 Steady State Analysis

In the following, we analyze the DAC algorithm under steady conditions and, based on this analysis, we compute the value of the p_{opt} parameter that maximizes the throughput obtained in steady state. The analyses of this and the following section assume saturation conditions, while the simulation results presented in Sec. 4.4 also cover the non-saturated case.

To analyze the system under steady conditions, we proceed as follows. Since the controller includes an integrator, this ensures that there is no steady state error [52]. The steady solution can therefore be obtained from imposing

$$e_i = 0 \ \forall i \tag{4.5}$$

from which

$$2p_{obs,i} - p_{own,i} - p_{opt} = 0 \tag{4.6}$$

Let τ_i be the probability that station i transmits at a given slot time [7]. $p_{own,i}$ and $p_{obs,i}$ can be computed as a function of the τ_i's as follows. $p_{own,i}$ is the probability that a transmission of station i collides

$$p_{own,i} = 1 - \prod_{k \neq i} (1 - \tau_k) \tag{4.7}$$

$p_{obs,i}$ is the average collision probability of the other stations measured by station i, which is computed by adding the individual collision probabilities of the other stations weighted by their transmission probability

$$p_{obs,i} = \sum_{k \neq i} \frac{\tau_k}{\sum_{l \neq i} \tau_l} \left(1 - \prod_{l \neq k} (1 - \tau_l) \right) \tag{4.8}$$

By using the above expressions for $p_{obs,i}$ and $p_{own,i}$, we can express Eq. (4.6) as a system of equations on the τ_i's. Theorem 3, whose proof is included in the Appendix, guarantees the uniqueness of the solution to the system of equations and shows that, with this solution, both terms of the error signal are equal to 0,

Theorem 3. *The system of equations defined by (4.6) has a unique solution that satisfies*

$$e_{collision,i} = e_{fairness,i} = 0 \ \forall i \tag{4.9}$$

and all stations have the same transmission probability,

$$\tau_i = \tau_j \ \forall i, j \tag{4.10}$$

Note that the above result given by Theorem 3 is of particular importance since it guarantees the existence of a unique stable point of operation for the system. Indeed, while the existence of a unique point of operation can be easily guaranteed in a centralized system by imposing the same configuration for all stations, it is much harder to guarantee this in a distributed system in which each station chooses its own configuration.

Substituting $\tau_i = \tau$, given by Eq. (4.10), into Eqs. (4.6), (4.7) and (4.8) yields

$$p_{opt} = 1 - (1 - \tau)^{n-1} \tag{4.11}$$

From the above equation, it follows that by setting the p_{opt} parameter in our control system, we fix the *conditional collision probability under steady conditions*. Therefore, we set this parameter in order to maximize the throughput of the WLAN, according to the analysis of Sec. 3.1.1, from which we have that under optimal operation the conditional collision probability in the WLAN, p_{opt}, is a constant independent of the number of stations. For the sake of completeness, we recall that this optimal values is expressed as

$$p_{opt} \approx 1 - e^{-\sqrt{\frac{2T_e}{T_c}}} \tag{4.12}$$

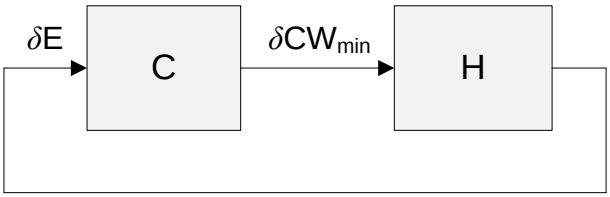

Figure 4.2: Control system

4.3 Stability Analysis

We next conduct a stability analysis of DAC and, based on this analysis, we compute the configuration of the K_p and K_i parameters of the controller. The DAC system presented in Fig. 4.1 can be expressed in the form of Fig. 4.2, where

$$CW_{min} = \begin{pmatrix} CW_{min,1} \\ \vdots \\ CW_{min,n} \end{pmatrix} \tag{4.13}$$

and

$$E = \begin{pmatrix} e_1 \\ \vdots \\ e_n \end{pmatrix} = \begin{pmatrix} 2p_{obs,1} - p_{own,1} - p_{opt} \\ \vdots \\ 2p_{obs,n} - p_{own,n} - p_{opt} \end{pmatrix} \tag{4.14}$$

Our control system consists of one PI controller for each station i, that takes e_i as input and gives $CW_{min,i}$ as output. Following this, we can express the relationship between E and CW_{min} as follows

$$CW_{min}(z) = C \cdot E(z) \tag{4.15}$$

where

$$C = \begin{pmatrix} C_{PI}(z) & 0 & 0 & \dots & 0 \\ 0 & C_{PI}(z) & 0 & \dots & 0 \\ 0 & 0 & C_{PI}(z) & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & C_{PI}(z) \end{pmatrix} \tag{4.16}$$

with $C_{PI}(z)$ being the z transform of a PI controller, whose expression we recall is the following

$$C_{PI}(z) = K_p + \frac{K_i}{z-1} \tag{4.17}$$

In order to analyze our system from a control theoretic standpoint, we need to char-

acterize the Wireless LAN system with a transfer function H that takes CW_{min} as input and gives the E as output.

Since we measure $p_{obs,i}$ and $p_{own,i}$ every 100 ms, we can assume that the measurements are obtained in stationary conditions. This implies that E depends only on the CW_{min} values used in the current interval and not on the previous ones, and hence the system H has no memory. With this, the only component of the delay present in the feedback loop is the one represented by the term z^{-1} of Fig. 4.2, which accounts for the fact that the CW_{min} values used in the current interval are the ones computed with the measurements taken in the previous interval.

Based on the above assumption, E can be computed from the $CW_{min,i}$'s by taking Eq. (4.14) and expressing $p_{own,i}$ and $p_{obs,i}$ as a function of the τ_i's following Eqs. (4.7) and (4.8). Furthermore, the τ_i's can be calculated as a function of the $CW_{min,i}$'s from the following nonlinear equation [7]

$$\tau_i = \frac{2}{1 + CW_{min,i}(1 + p_{own,i} \sum_{k=0}^{m-1} (2p_{own,i})^k)} \tag{4.18}$$

where $p_{own,i}$ is a function of τ_i as given by Eq. (4.7).

The above gives a nonlinear relationship between E and CW_{min}. In order to express this relationship as a transfer function, we linearize it when the system suffers small perturbations around its stable point of operation, taking a similar approach to the one used in Sec. 3.1.2.2, although the analysis of Sec. 3.1.2.2 focused on a single-variable system, while we analyze a multivariable system. In the following, we study the linearized model and force that it is stable.

We express the perturbations around the point of operation as follows:

$$CW_{min,i} = CW_{min,i,opt} + \delta CW_{min,i} \tag{4.19}$$

where $CW_{min,i,opt}$ is the $CW_{min,i}$ value that yields the transmission probability τ_{opt} given by Eq. (3.18).

With the above, the perturbations suffered by E can be approximated by

$$\delta E = H \cdot \delta CW_{min} \tag{4.20}$$

where

$$H = \begin{pmatrix} \frac{\partial e_1}{\partial CW_{min,1}} & \frac{\partial e_1}{\partial CW_{min,2}} & \cdots & \frac{\partial e_1}{\partial CW_{min,n}} \\ \frac{\partial e_2}{\partial CW_{min,1}} & \frac{\partial e_2}{\partial CW_{min,2}} & \cdots & \frac{\partial e_2}{\partial CW_{min,n}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e_n}{\partial CW_{min,1}} & \frac{\partial e_n}{\partial CW_{min,2}} & \cdots & \frac{\partial e_n}{\partial CW_{min,n}} \end{pmatrix} \tag{4.21}$$

The above partial derivatives can be computed as

$$\frac{\partial e_i}{\partial CW_{min,j}} = \frac{\partial e_i}{\partial \tau_j}\frac{\partial \tau_j}{\partial CW_{min,j}} \tag{4.22}$$

where, from Eq. (4.18), we have

$$\frac{\partial \tau_j}{\partial CW_{min,j}} = -\tau_j^2 \frac{\left(1 + p_{own,j}\sum_{k=0}^{m}(2p_{own,j})^k\right)}{2} \tag{4.23}$$

which, evaluated at the stable point of operation, $p_{own,j} = p_{opt}$ and $\tau_j = \tau_{opt}$, yields

$$\frac{\partial \tau_j}{\partial CW_{min,j}} = -\tau_{opt}^2 \frac{\left(1 + p_{opt}\sum_{k=0}^{m}(2p_{opt})^k\right)}{2} \tag{4.24}$$

To compute $\partial e_i/\partial \tau_j$ for $j \neq i$ we proceed as follows

$$\frac{\partial e_i}{\partial \tau_j} = 2\frac{\partial p_{obs,i}}{\partial \tau_j} - \frac{\partial p_{own,i}}{\partial \tau_j} \tag{4.25}$$

By calculating the two partial derivatives of the above equation and evaluating them at $\tau = \tau_{opt}$ we obtain

$$\frac{\partial p_{obs,i}}{\partial \tau_j} = \frac{(n-2)(1-\tau_{opt})^{n-2}}{(n-1)} \tag{4.26}$$

and

$$\frac{\partial p_{own,i}}{\partial \tau_j} = (1-\tau_{opt})^{n-2} \tag{4.27}$$

From the above,

$$\frac{\partial e_i}{\partial \tau_j} = \frac{(n-3)(1-\tau_{opt})^{n-2}}{(n-1)} \tag{4.28}$$

Following a similar procedure, we obtain

$$\frac{\partial e_i}{\partial \tau_i} = 2(1-\tau_{opt})^{n-2} \tag{4.29}$$

Combining all the above, yields

$$H = K_H \begin{pmatrix} 2 & \frac{n-3}{n-1} & \frac{n-3}{n-1} & \cdots & \frac{n-3}{n-1} \\ \frac{n-3}{n-1} & 2 & \frac{n-3}{n-1} & \cdots & \frac{n-3}{n-1} \\ \frac{n-3}{n-1} & \frac{n-3}{n-1} & 2 & \cdots & \frac{n-3}{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{n-3}{n-1} & \frac{n-3}{n-1} & \frac{n-3}{n-1} & \cdots & 2 \end{pmatrix} \tag{4.30}$$

where

$$K_H = -\tau_{opt}^2(1-\tau_{opt})^{n-2}\frac{\left(1+p_{opt}\sum_{k=0}^{m}(2p_{opt})^k\right)}{2} \tag{4.31}$$

With the above, we have our system fully characterized by the matrices C and H. The next step is to configure the K_p and K_i parameters of this system. Following Theorem 4, we have that as long as the $\{K_p, K_i\}$ setting satisfies Eq. (4.32), the system is guaranteed to be stable.

Theorem 4. *The system is guaranteed to be stable as long as K_p and K_i meet the following condition:*

$$-(n-1)K_H(K_p-K_i)-1 < (n-1)K_H(K_p-K_i)+1 \tag{4.32}$$

In addition to guaranteeing stability, our goal in the configuration of the $\{K_p, K_i\}$ parameters is to find the right tradeoff between speed of reaction to changes and oscillations under transient conditions. To this aim, we use again the Ziegler–Nichols rules [55] as in Sec. 3.1.2.4. Therefore, K_p and K_i are configured as follows:

$$K_p = 0.4K_u \tag{4.33}$$

$$K_i = \frac{K_p}{0.85T_i} \tag{4.34}$$

In order to compute K_u we proceed as follows. From Eq. (4.32) with K_i we have

$$K_p < \frac{1}{-(n-1)K_H} \tag{4.35}$$

Combining the above with Eq. (4.31) yields

$$K_p < \frac{2}{(n-1)\tau_{opt}^2(1-\tau_{opt})^{n-2}\left(1+p_{opt}\sum_{k=0}^{m}(2p_{opt})^k\right)} \tag{4.36}$$

Since $p_{opt} = 1-(1-\tau_{opt})^{n-1} \approx (n-1)\tau_{opt}$, the above can be rewritten as

$$K_p < \frac{2}{p_{opt}\tau_{opt}(1-\tau_{opt})^{n-2}\left(1+p_{opt}\sum_{k=0}^{m}(2p_{opt})^k\right)} \tag{4.37}$$

Since the above is a function of n (note that τ_{opt} depends on n) and we want to find an upper bound that is independent of n, we proceed as follows. From $p_{opt} = 1-(1-\tau_{opt})^{n-1}$, we observe that τ_{opt} is never larger than p_{opt} for $n > 1$. (Note that for $n = 1$ the system is stable for any K_p.) Furthermore, we have $(1-\tau_{opt})^{n-2} < 1$. With these observations,

we obtain the following constant upper bound (independent of n):

$$K_p < \frac{2}{p_{opt}^2 \left(1 + p_{opt} \sum_{k=0}^{m} (2p_{opt})^k\right)} \tag{4.38}$$

Following the above, we take K_u as the value where the system may turn unstable (given by the previous equation),

$$K_u = \frac{2}{p_{opt}^2 \left(1 + p_{opt} \sum_{k=0}^{m} (2p_{opt})^k\right)} \tag{4.39}$$

and set K_p according to Eq. (4.33),

$$K_p = \frac{0.4 \cdot 2}{p_{opt}^2 \left(1 + p_{opt} \sum_{k=0}^{m} (2p_{opt}^k\right)} \tag{4.40}$$

With the K_p value that makes the system become unstable, a given set of input values may change their sign up to every time slot, yielding an oscillation period of two slots ($T_i = 2$). Thus, from Eq. (4.34)

$$K_i = \frac{0.4}{0.85 p_{opt}^2 \left(1 + p_{opt} \sum_{k=0}^{m} (2p_{opt})^k\right)} \tag{4.41}$$

which completes the configuration of the PI controller parameters. The stability of this configuration is guaranteed by Corollary 1.

Corollary 1. *The K_p and K_i configuration given by Eqs. (4.40) and (4.41) is stable.*

4.4 Performance Evaluation

In this section we evaluate *DAC* by conducting an extensive set of simulations under different traffic scenarios and compare its performance against the following approaches: the standard default configuration (EDCA) [1], the static optimal configuration obtained with [26] and several other adaptive algorithms, namely the *Enhanced 802.11* [15], *Idle Sense* [16] and the *Dynamic 802.11* [18]. Unlike these previous papers, which assume that all stations are saturated (i.e., they always have a packet ready for transmission), we analyze the saturated and non-saturated scenarios as well as the mixed one.

For the simulations, we have implemented our algorithm as well as the different existing proposals in OMNeT++. In all the experiments, we used the physical layer parameters of IEEE 802.11g [5] and a fixed payload size of 1000 Bytes. For the obtained results, average and 95% confidence intervals are given.

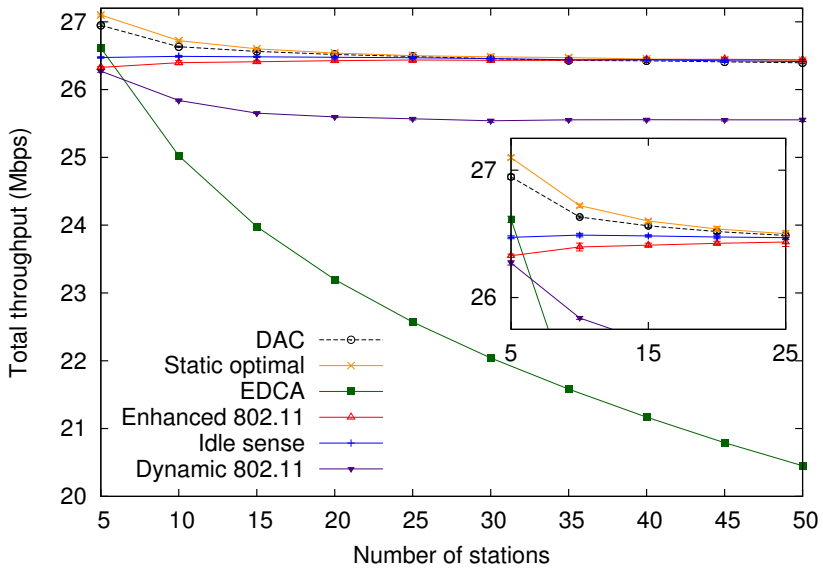

Figure 4.3: Saturated scenario

4.4.1 Saturated Scenario

First, we evaluate the performance of *DAC* in a WLAN operating under saturation conditions. For this purpose, we compare the total throughput achieved by *DAC* for an increasing number of saturated stations against the static optimal configuration, EDCA and the other adaptive schemes.

Results are depicted in Fig. 4.3 (which includes a zoom in subplot). We observe from these results that *(i) DAC* closely follows the static optimal configuration for any n, *(ii)* it slightly outperforms *Enhanced 802.11* and *Idle Sense* for a small number of active stations, and *(iii)* it substantially outperforms *Dynamic 802.11* and EDCA. We recall that the static optimal configuration requires to know a priori the number of stations in the network, which challenges its practical use. Additionally, the other adaptive mechanisms introduce extra complexity and are not standard compliant, which makes them more difficult to deploy.

We conclude from the above that *DAC* achieves the objective of maximizing the total throughput in saturated conditions, without requiring to estimate the number of stations and avoiding complex and non-standard mechanisms.

4.4.2 Non-saturated Scenario

We next analyze the behavior of the proposed algorithm in a non-saturated scenario where all stations send Poisson traffic with an average bit rate of 500 Kbps. Note that,

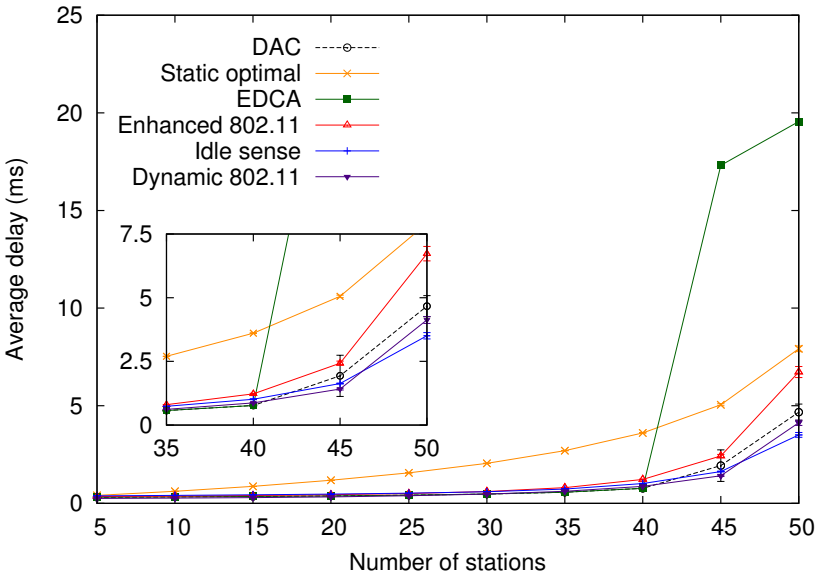

Figure 4.4: Non-saturation scenario

in a non-saturated scenario, all stations see their throughput demands satisfied, and performance is given by delay.

Fig. 4.4 illustrates the average delay in the above scenario as a function of the number of stations. From the results, we observe that our proposal minimizes the average delay. It performs similarly to the other adaptive approaches, and outperforms the static optimal configuration (which is based on the assumption that all stations are saturated and thus enforces an overly large CW) and EDCA (which uses a small fixed value of the CW_{min}, thus degrading performance for large n values).

We conclude that, in addition to maximizing the total throughput under saturation, DAC also minimizes the average delay under non-saturation.

4.4.3 Mixed Scenario

We next address a mixed scenario in which some of the stations are saturated and some are not. In particular, we take half of the stations saturated and the other half sending Poisson traffic at an average bit rate of 500 Kbps.

In Fig. 4.5 we analyze the performance of our algorithm in terms of total throughput. We observe that DAC succeeds in maximizing the throughput also for a mixed scenario, since it outperforms all other approaches and in particular it substantially outperforms the static optimal configuration.

In addition to the throughput evaluation, we also analyze the delay performance of DAC in the same scenario by measuring the average delay experienced by the non-

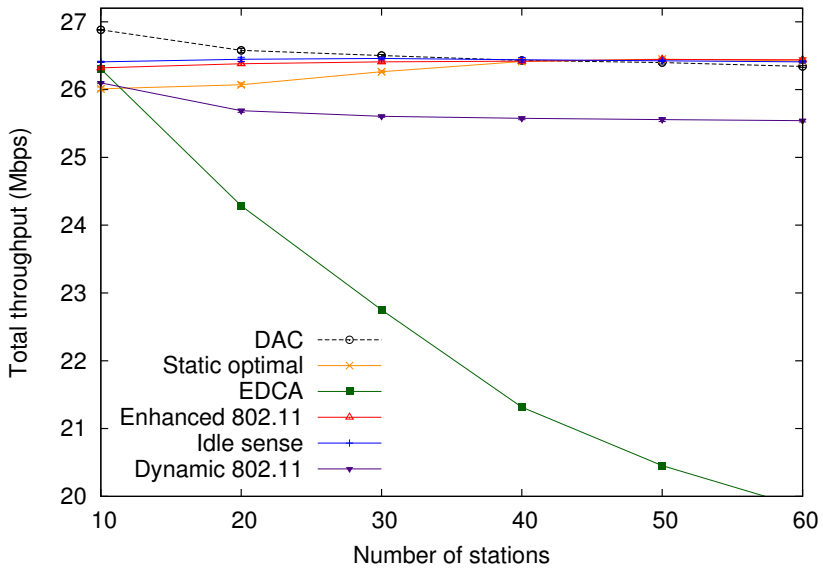

Figure 4.5: Throughput performance under mixed traffic conditions

saturated and saturated stations. Results are depicted in Fig. 4.6 (the delay of the saturated stations is given in a subplot). We can see from the figure that DAC substantially outperforms all the other approaches, since it provides the non-saturated stations with smaller delays without harming the delay performance of the saturated stations. The reason why our approach outperforms the other adaptive approaches is that, upon detecting congestion, the other approaches increase the CW of all stations (the saturated and the non-saturated ones), harming thus the delay performance of the non-saturated stations. In contrast, our algorithm is designed to increase only the CW of the saturated stations, which are the ones contributing to congestion.

In the previous experiment we had the same number of saturated and non-saturated stations. In order to show the impact of having an unbalanced scenario with a different number of saturated and non-saturated stations, we repeat the experiment for 5 non-saturated stations and a variable number of saturated stations. Fig. 4.7 shows the resulting total throughput and Fig. 4.8 the average delay. We conclude from the above that DAC performs better than any other approach when saturated and non-saturated stations coexist in the WLAN, as it minimizes the delay performance of non-saturated station while neither harming the total throughput of the WLAN nor the delay of the saturated stations.

4.4.4 Convergence

Our analysis guarantees that, after some transient, the CW_{min}'s of all stations converge towards a common value. In order to illustrate this behavior, we perform the

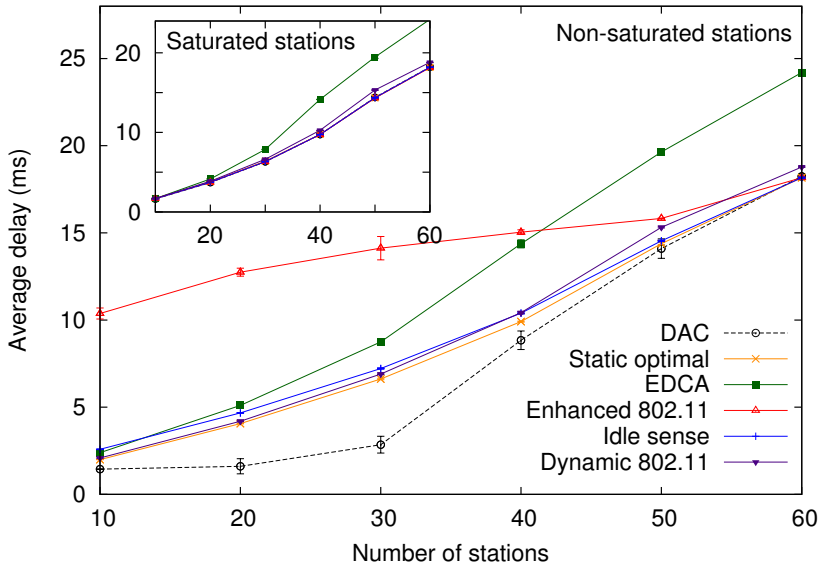

Figure 4.6: Average delay under mixed traffic conditions

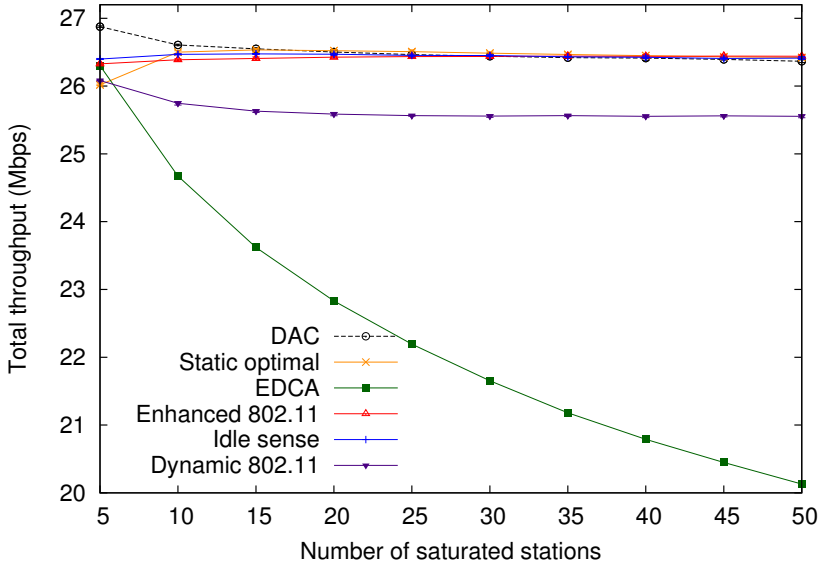

Figure 4.7: Throughput performance of the mixed unbalanced scenario

following experiment. In a WLAN with 5 stations, one new station joins every 20 s until a total of 10 stations is reached. In this experiment, we analyze the CW_{min} of one of the initial stations as well as the CW_{min} of each one of the new stations joining. The results, depicted in Fig. 4.9, show that both the stations already present in the network and the new joining ones converge fast to the same CW_{min} value. Thus, this experiment confirms our theoretical result on the convergence of the proposed distributed algorithm.

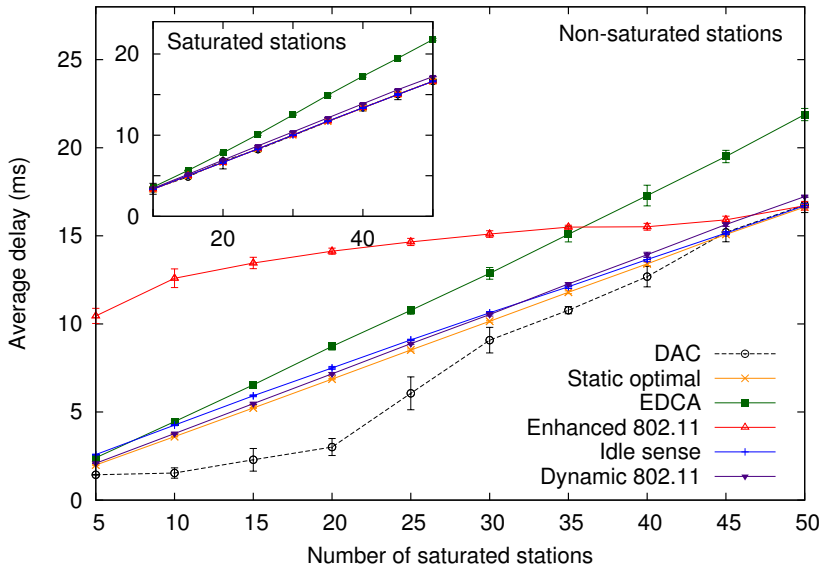

Figure 4.8: Average delay of the mixed unbalanced scenario

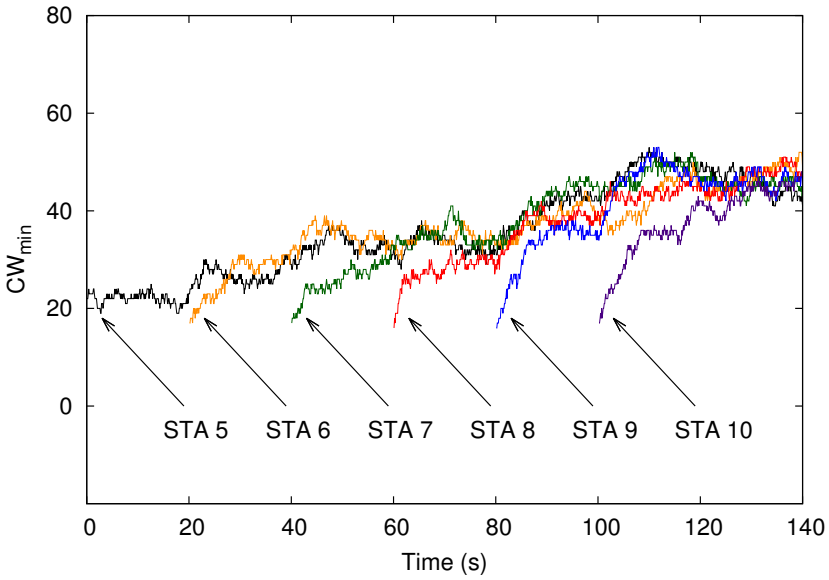

Figure 4.9: Convergence

4.4.5 Stability and Speed of Reaction to Changes

The main objective in the configuration of the K_p and K_i parameters proposed in Sec. 4.3 is to achieve a proper tradeoff between stability and speed of reaction to changes. This objective is verified by the results presented in this subsection.

To validate that our system guarantees a stable behavior, we analyze the evolution in time of the control signal (CW_{min}) for our $\{K_p, K_i\}$ setting and a configuration with values of these parameters 20 times larger, in a network with 10 saturated stations. We

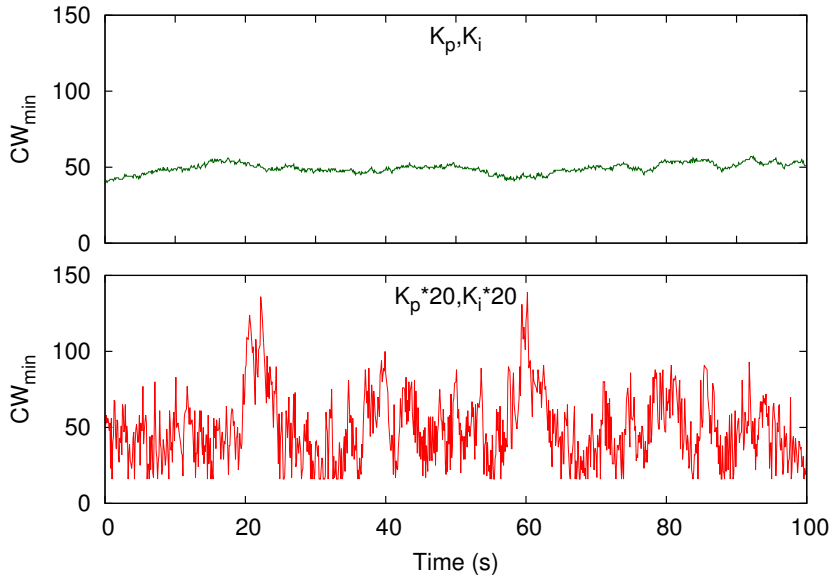

Figure 4.10: Stability validation

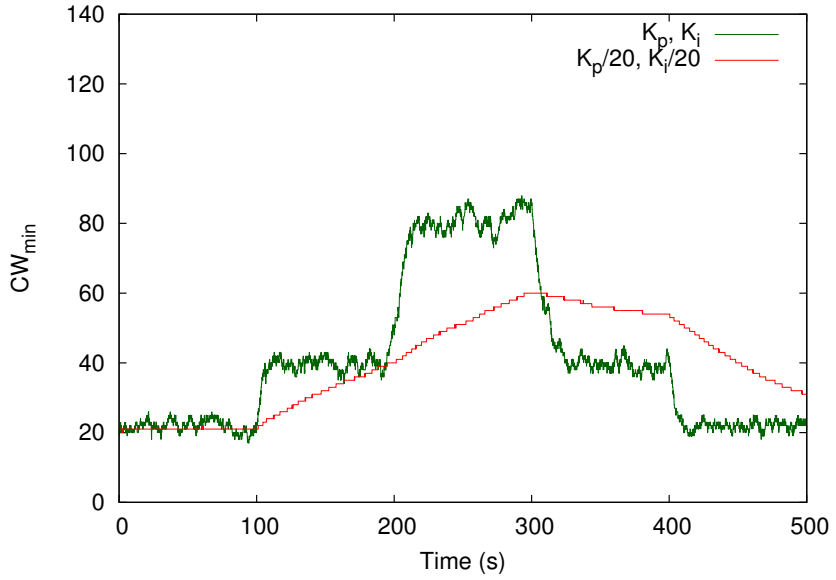

Figure 4.11: Speed of reaction to changes

observe form Fig. 4.10 that with the proposed configuration (label "K_p, K_i"), the CW_{min} only presents minor deviations around its stable point of operation, while if a larger setting is used (label "$K_p * 20, K_i * 20$"), the CW_{min} has a strong unstable behavior with drastic oscillations. We conclude that the proposed configuration achieves the objective of guaranteeing stability.

In order to verify that our system has the ability to rapidly react to changes in the network, we conduct the following experiment. In a WLAN initially with 5 stations, 5

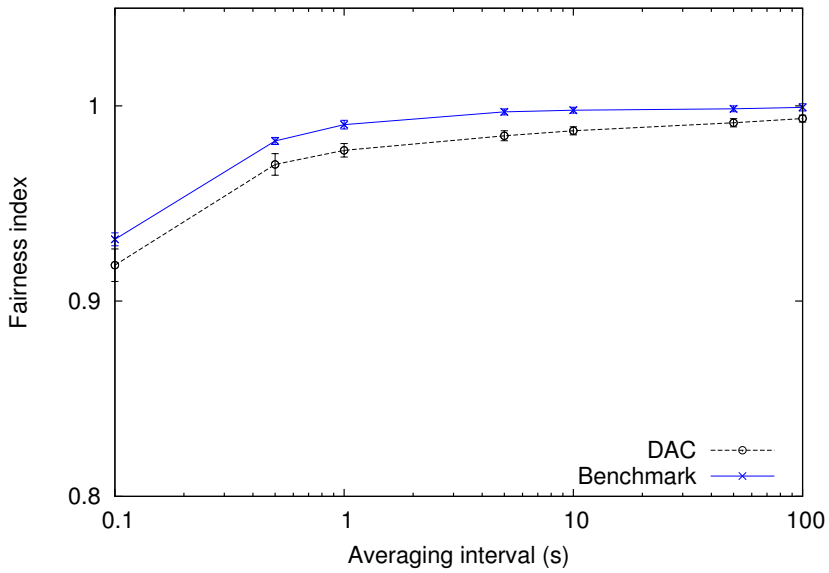

Figure 4.12: Fairness

additional stations join the WLAN at time 100 s, and 5 more stations (yielding a total of 15) join 100 s afterwards. After additional 100 s, 5 stations leave the WLAN, and again 5 more stations leave, returning to the initial state with 5 stations. For this experiment, we examine the evolution over time of the CW_{min} used by one station of the initial group for our $\{K_p, K_i\}$ setting, as well as for a smaller value of these parameters. From Fig. 4.11, we observe that, with our configuration (label "K_p, K_i"), the system reacts fast to the changes on the WLAN, as the CW_{min} reaches the new value almost immediately. In contrast, for a setting of these parameters 20 times smaller (label "$K_p/20, K_i/20$"), the system cannot keep up with the changes as CW_{min} reacts too slowly.

From the above results, we conclude that the proposed setting of $\{K_p, K_i\}$ provides a good tradeoff between stability and speed of reaction, since with a larger setting the system suffers from instability and with a smaller one it reacts too slowly to changes.

4.4.6 Fairness

In Sec. 4.4.1, we have evaluated the total throughput performance of our approach, but it is also relevant to analyze whether the total throughput is fairly shared among stations over short time scales and understand the impact of varying CW_{min} on fairness. Although our algorithm provides the same average CW_{min} to all stations over long time periods, at a given instant two stations may have slightly different CW_{min} values. In order to understand if this has any significant impact on short-term fairness we compare our approach against benchmark values. More specifically, we evaluate Jain's fairness index [84] over different averaging intervals for our approach and a configuration in which

all stations use the same CW_{min}, whose value is equal to the average CW_{min} used by the adaptive algorithm.

The scenario consists of 10 stations always having a packet ready for transmission. The result of this experiment is depicted in Fig. 4.12. We conclude that our approach performs close to the benchmark configuration in terms of short-term fairness and the fairness index of DAC is close to 1 for reasonable periods of time.

4.5 Summary

In this chapter, we have proposed a distributed adaptive algorithm to optimally configure IEEE 802.11 networks. The key advantages of the proposed algorithm over existing approaches are: *(i)* the DAC algorithm is sustained by mathematical foundations that guarantee optimal performance, convergence and stability, *(ii)* the mechanism is standard-compliant and can be implemented with existing hardware, and *(iii)* it outperforms previous approaches in terms of throughput and delay.

The proposed algorithm executes an independent PI controller at each station, that takes as input the measured error signal and gives as output the station's configuration. The error signal has been carefully chosen to ensure that *(i)* the stable point of operation gives optimal throughput performance, and *(ii)* when the WLAN operates at any other point, the error signal is large thus forcing the WLAN to quickly converge to the stable point.

The performance of the proposed algorithm has been extensively evaluated by means of simulations. Results have shown that *(i)* our scheme substantially outperforms EDCA in terms of throughput, *(ii)* it performs better than the static optimal configuration when not all stations are saturated, and *(iii)* it outperforms other distributed adaptive approaches in terms of delay.

While we have demonstrated that the proposed mechanisms significantly outperform previous schemes, in the next chapter we further illustrate the advantages of the CAC and DAC algorithms by showing that they are indeed compatible with the IEEE 802.11 standard and can be implemented with COTS devices without any modifications. We also investigate their performance in a real-life testbed under a multitude of network conditions and provide valuable insights on their suitability for practical deployments.

Chapter 5

Experimental Evaluation

Although a significant effort has been devoted in the literature to the design of centralized and distributed algorithm that aim to enhance the WLAN performance, very few of the proposed schemes have been developed in practice [14, 50, 51]. Additionally, the existing implementations involve increased complexity and impose tight timing constraints [50], while the conducted experimental studies are limited to very basic scenarios [14, 51]. In this chapter, we present our experiences implementing the *CAC* and *DAC* algorithms proposed in Chapters 3 and 4, respectively. We show that, in contrast to previous approaches, our algorithms can be implemented with existing devices without requiring any modifications of their hardware or firmware.

We discuss the differences between the theoretical and practical implementations of the algorithms, inherent to the limitations of the real devices, and we evaluate the performance of our proposals by conducting a wide set of experiments in a medium-scale testbed under and different network conditions. From the results obtained, we identify possible scenarios where a network deployment can benefit from using the *CAC* and *DAC* algorithms over the default IEEE 802.11 mechanism.

5.1 Implementation Details

In this section we present our implementation experiences with *CAC* and *DAC*: we describe the design principles, the hardware platform used in our deployment, the modifications introduced in the software drivers to realize the key building blocks, and the main differences between the theoretical design of the algorithms and their implementation, which arise due to the constraints imposed by the devices.

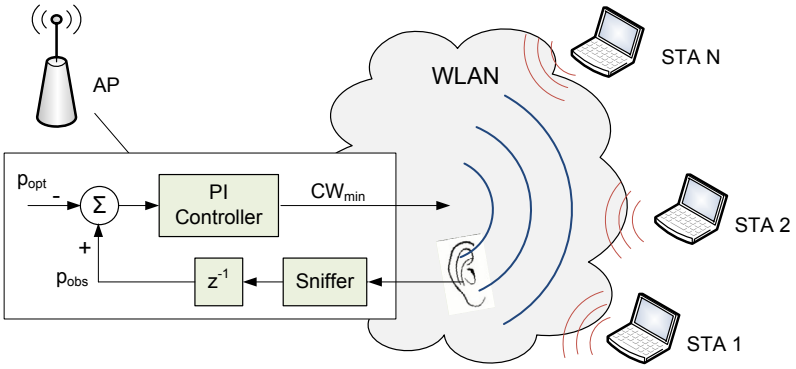

Figure 5.1: CAC algorithm.

5.1.1 CAC Algorithm

The *CAC* algorithm presented in Chapter 3 relies on a PI controller residing at the AP, which performs two tasks every beacon interval (approx. 100 ms): *(i)* an estimation of the current point of operation of the whole WLAN as given by the observed collision probability p_{obs}, and *(ii)* based on this estimation and p_{opt}, the computation and broadcast (in a standard beacon frame) of the CW configuration that stations will use (Fig. 5.1).

The computation of p_{obs} is based on the examination of the retry flag of the correctly received frames and is estimated using Eq. (3.25), which can be regarded as the probability that the first transmission attempt from a station collides.

We recall that the error signal e fed into the PI controller to compute the new CW_{min} consists of the difference between the observed collision probability p_{obs} and the target

Algorithm 1 Centralized Adaptive Control algorithm

```
1:  while CAC on do
2:      while next beacon interval do
3:          if new frame sniffed then
4:              retrieve retry flag
5:              if retry flag is set then
6:                  Increment R_1
7:              else
8:                  Increment R_0
9:              end if
10:         end if
11:     end while
12:     compute p_obs using Eq. (3.25)
13:     compute e[t] = p_obs[t] − p_opt
14:     CW_min[t] = CW_min[t − 1] + K_p · e[t] + (K_i − K_p) · e[t − 1]
15:     schedule new CW update with next beacon
16: end while
```

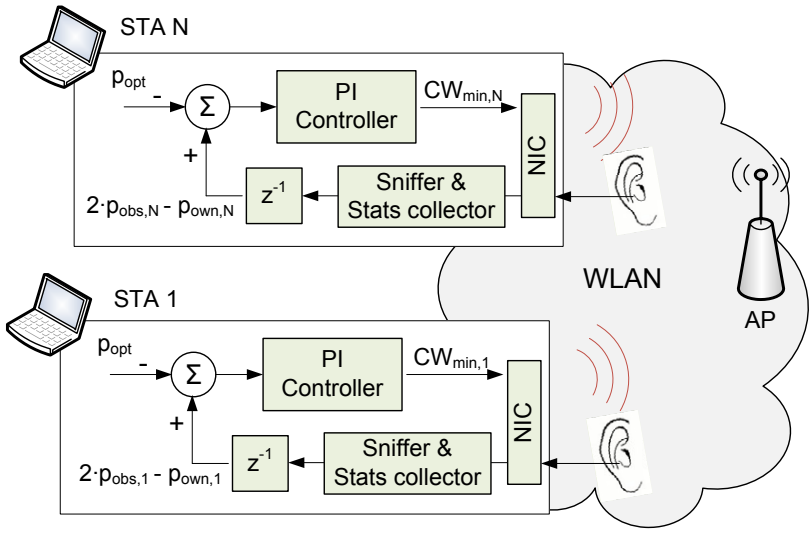

Figure 5.2: DAC algorithm

value p_{opt}.

The newly computed configuration is distributed by the AP to all the associated stations with the next scheduled beacon frame, a feature which is available with the current standard [1].

5.1.2 DAC Algorithm

The *DAC* algorithm, presented in Chapter 4, employs a PI controller at each station to drive the overall collision probability to the target p_{opt}. As illustrated in Fig. 5.2, each controller independently computes the CW_{min} value to be used by its Network Interface Card (NIC), based on the locally observed network conditions.

Algorithm 2 Distributed Adaptive Control algorithm

while DAC on **do**
  while next received beacon **do**
    if new frame sniffed **then**
      retrieve retry flag and increment R_0 or R_1 accordingly
    end if
  end while
  estimate $p_{obs,i}$ using Eq. (3.25)
  query device for statistics
  compute $p_{own,i}$ by applying Eq. (4.4)
  $e[t] = 2 \cdot p_{obs,i}[t] - p_{own,i}[t] - p_{opt}$
  $CW_{min}[t] = CW_{min}[t-1] + K_p \cdot e[t] + (K_i - K_p) \cdot e[t-1]$
  update the local CW configuration
end while

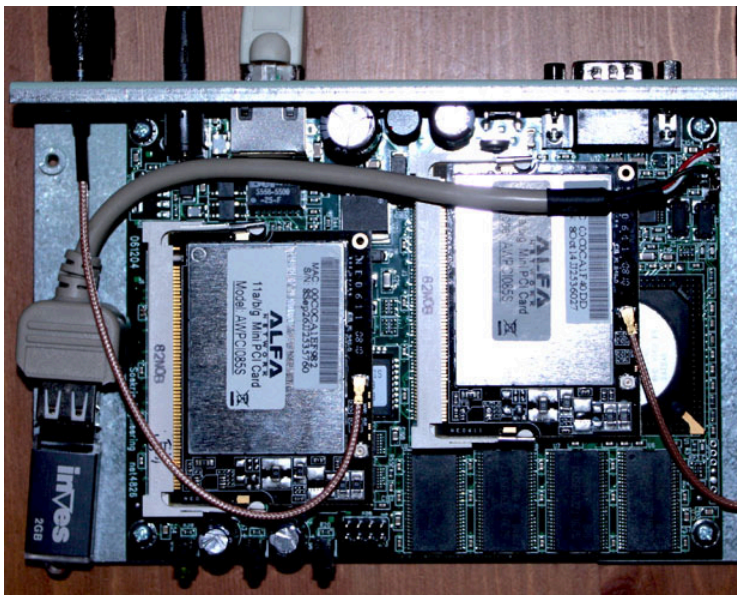

Figure 5.3: Hardware platform

The error signal used by DAC is composed of two terms: one that drives the WLAN to the desired point of operation, and another one that aims to achieve fairness among stations (See. Eq. (4.3)). In order to compute this above error signal, DAC needs to measure $p_{obs,i}$ and $p_{own,i}$. Similar to CAC, the former term is computed using Eq. (3.25). For the computation of $p_{own,i}$, we rely on the following statistics which are readily available from wireless cards: the number of successful transmission attempts and the number of failed attempts. Thus, p_{own} is computed according to Eq. (4.4).

The controller employed by each station is characterized by the transfer function of Eq. (4.17), with the $\{K_p, K_i\}$ parameters as given in Sec. 4.3.

5.1.3 Hardware & Software Platform

We have implemented our algorithms with the Soekris net4826-48 advanced embedded communication computers.[1] These are low-power, low-costs PC-based devices equipped with 233 MHz AMD Geode SC1100 CPU, 2 Mini-PCI sockets, 128 Mbyte SDRAM and 256 Mbyte compact flash circuits for data storage. To accommodate the installation of current Linux distributions, we have extended the storage capacity of the boards with 2 Gbyte USB drives.

We have utilized as wireless interfaces Atheros AR5414 based 802.11a/b/g devices, manufactured by Alfa Network. Atheros chipset cards have been widely used by the research community due to the availability of open-source drivers developed for them

[1] `http://www.soekris.com/`

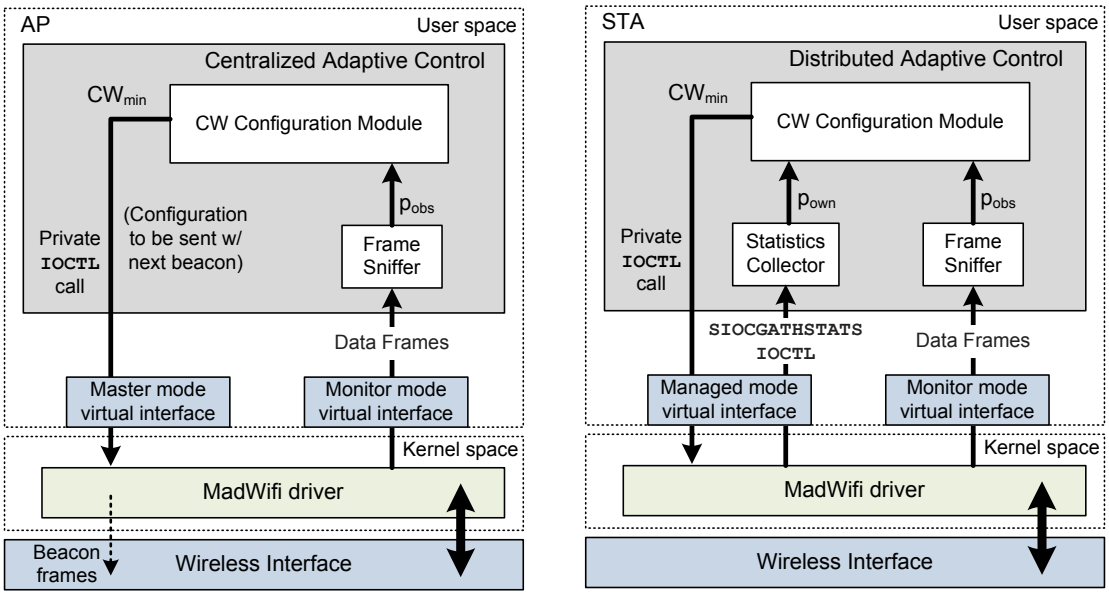

Figure 5.4: CAC and DAC implementations

and their flexibility in accessing low layer functionalities [85–87]. The hardware platform utilized for the evaluation of our algorithms is depicted in Fig. 5.3 (note that, although the board shown above is equipped with two wireless adapters, we only use one of them).

As software platform, we have installed Gentoo Linux OS (with kernel 2.6.24) and the popular MadWifi[2] open-source WLAN driver (version v0.9.4), which we have modified as follows: *(i)* we enabled the dynamic setting of the EDCA parameters for the best effort access category, which is in line with the standard specifications but not supported by the default driver, *(ii)* we overwrote the drivers' EDCA values for the best effort traffic with the standard recommended values [1], and *(iii)* for the case of *DAC*, we modified the driver such that the stations employ the locally computed EDCA configuration using standardized system calls (Sec. 5.1.7). The source code of our modified drivers is available online.[3]

5.1.4 Implementation Overview

A major advantage of *CAC* and *DAC* is that they do not require modifications to the hardware and/or firmware of the wireless interface, neither introducing tight timing constraints nor violating the channel access mechanism as defined by the standard. As illustrated in Fig. 5.4, both algorithms run as user-space applications and communicate with the driver by means of `IOCTL` implementing the desired functionality. We also take

[2] `http://madwifi-project.org/`
[3] `http://enjambre.it.uc3m.es/~ppatras/research.php`

advantage of the ability of the MadWifi driver to support multiple virtual devices using different operation modes (master/managed/monitor) with a single physical interface.

5.1.5 Estimation of p_{obs}

Both algorithms require to estimate the collision probability observed in the WLAN. For the case of *CAC*, this is performed only at the AP and results in p_{obs}, while for the case of *DAC* this is performed independently at each station i and results in $p_{obs,i}$. The estimators are computed with Eq. (3.25), which relies on the observation of the retry flag of the overheard frames. We next explain how these values are obtained from a practical perspective.

To overhear frames, we utilize a virtual device operating in the so called `monitor` mode with the promiscuous configuration. This determines the device to pass all traffic to user-space applications, including frames not addressed to the station. We also configure the device to pass the received frames with full IEEE 802.11 link layer headers, such that Frame Control field of the frames (where the retry flag resides) can be examined.

With this set-up, the algorithms open a `raw` socket to the driver, which enables the reception of Layer 2 frames. Through this socket, the algorithms listen for transmitted frames and process their headers in an independent thread (this is the "Frame Sniffer" module of Fig. 5.4). For every observed frame, one of the counters used in the estimation of the collision probability is incremented: R_0 if the retry flag was unset, R_1 if the retry flag was set. For every beacon interval, the computation of p_{obs} (for the case of *CAC*) or $p_{obs,i}$ (for the case of *DAC*) is triggered, and then the counters are reset to zero.

5.1.6 Estimation of p_{own}

In addition to the observed collision probability $p_{obs,i}$, the *DAC* algorithm requires to estimate the experienced collision probability $p_{own,i}$. We perform this computation in the "Statistics Collector" module of Fig. 5.4, using information recorded by the wireless driver. More specifically, at the end of a beacon interval we open a communication channel with the driver instance, configured as a `managed` interface, and perform a `SIOCGATHSTATS IOCTL` request. Upon this request, the driver will populate an `ath_stats` data structure, which contains detailed information about the transmitted and received frames since the Linux kernel has loaded the driver module. Out of the statistics retrieved, the records that are of particular interest for our implementation are:

- `ast_tx_packets`: number of unique frames sent to the transmission interface,
- `ast_tx_noack`: number of transmitted frames that do not require ACK,

- `ast_tx_longretry`: number of transmission retries of frames larger than the RTS threshold. Note that we do not use the RTS/CTS mechanisms, so we can interpret this number as the total number of retransmissions,
- `ast_tx_xretries`: number of frames not transmitted due to exceeding the retry limit, which is set by the `MAX_RETRY` parameter.

Our estimation of $p_{own,i}$ is based on data frames actually received. In order to compute the total number of frames delivered, we subtract from the number of unique frames those that are not acknowledged (e.g., management frames) and those that were not delivered, i.e.,

$$Successes = \texttt{ast\_tx\_packets} - \texttt{ast\_tx\_xretries} - \texttt{ast\_tx\_noack}$$

Similarly, out of the total number of retransmissions we do not count those retransmissions caused by frames that were eventually discarded because the `MAX_RETRY` limit had been reached. Therefore,

$$Failures = \texttt{ast\_tx\_longretry} - \texttt{ast\_tx\_xretries} \cdot \texttt{MAX\_RETRY}$$

With the above, the terms F and T of Eq. (4.4) used to estimate $p_{own,i}$ are computed as follows

$$F[t] = Failures[t] - Failures[t-1]$$

$$T[t] = Successes[t] - Successes[t-1]$$

where t denotes the current beacon interval and $t-1$ the previous one.

5.1.7 Contention Window Update

With the estimated collision probabilities, CAC and DAC compute the error signal at the end of an update interval. Depending on this value the controller will trigger an increase or decrease of the CW_{min} to be used in the next beacon interval t, according to the following expression for the PI controller operation:

$$CW_{\min}[t] = CW_{\min}[t-1] + K_P \cdot e[t] + (K_I - K_P) \cdot e[t-1] \tag{5.1}$$

To ensure a safeguard against too large and too small CW_{min} values we impose lower and upper bounds for the CW_{min}. We set these bounds to the default CW_{min} and CW_{max} used by the standard, which are 16 and 1024, respectively, for the case of IEEE 802.11a [4].

The algorithm assumes that the CW_{min} can take any real value in the $[16, 1024]$ range. However, with our devices only integer powers of 2 are supported (i.e., $CW_{min} \in$

$\{16, 32, \ldots, 1024\}$). Therefore, while the CW_{min} is computed according to the above equation and its value is kept for the next computation, the value actually used (at the local NIC or broadcasted in the beacon frame) is obtained as:

$$\texttt{CW[t] = (int) rint(log}_2\texttt{(}CW_{min}[t]\texttt{))}$$

The above gives exactly the power 2 exponent of the CW_{min}, as required by the driver.

In order to commit the computed `CW` configuration, first we retrieve the list of private `IOCTL`s supported by the device to search for the call that sets the CW_{min}. Once this call has been identified, we prepare a data structure, namely `iwreq`, with the following information: the interface name, the `CW` computed as above, the access category index as defined by the standard (e.g., 0 for Best Effort) and an additional parameter that identifies if the value is intended to be used locally or propagated. For the case of *DAC* this value is set to 0, as the `CW` is intended to the local NIC only, while for the case of *CAC* is set to 1, thereby requesting the driver to broadcast the new `CW` within the EDCA Parameter Set element of the next scheduled beacon frame. As an exception to this, as explained in Sec. 4.1, the update is differed if the number of samples used to compute p_{own} is smaller than 20, to avoid inaccurate estimations caused by a low number of samples.

5.2 Performance Evaluation

In this section, we evaluate the performance of *CAC* and *DAC* in our medium-scale testbed. We first describe the deployed testbed, then we evaluate the behavior of the proposed algorithms under different channel conditions and diverse number of active nodes. To this aim, we investigate the effect of link heterogeneity in the network, the capture effect and the occurrence of hidden nodes. We also show the impact of the dynamic network conditions in terms of numbers of active stations and on-off traffic patterns. We provide insights about the scenarios which best suit the use of each algorithm, while contrasting their performance with the standard EDCA operation.

5.2.1 Testbed & Evaluation Methodology

Our testbed consists of 12 Soekris net4826-48 devices deployed inside a laboratory located in the Torres Quevedo building at University Carlos III of Madrid. Our deployment extends our previously established FloorNet testbed[4] and utilizes the already existing infrastructure, which enabled us to better control and automate the experiments from a

[4] `http://floornet.org/`

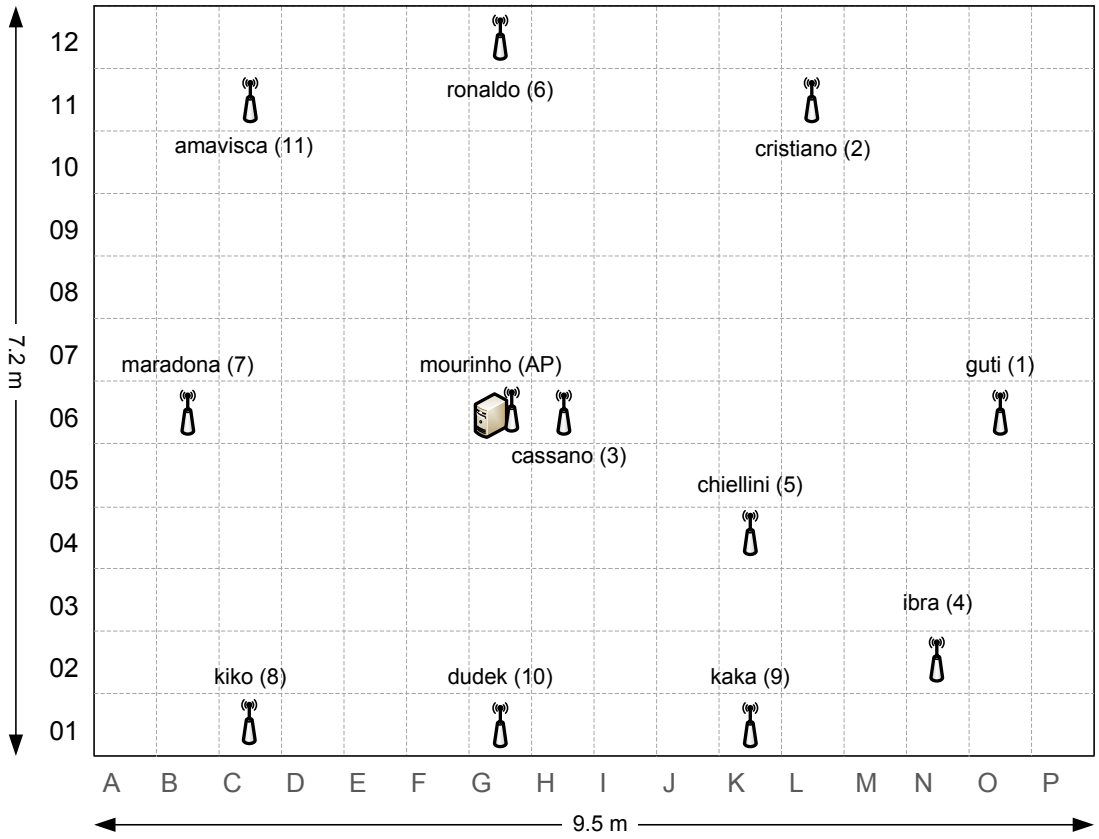

Figure 5.5: Deployed testbed.

single console. The nodes were deployed under the false floor of the laboratory, which provides physical protection that prevents disconnections or misplacements, as well as radio shielding thanks to the false floor panels [88].

Fig. 5.5 illustrates the locations of the nodes. We placed one node towards the center of the testbed, thus following the placement of an AP in a realistic deployment, while the other stations are distributed at different distances from this AP. All nodes are equipped with 5 dBi omni-directional antennas and are configured to operate on channel 64 (5.32 GHz), as specified by the IEEE 802.11a standard [4], where no other WLAN were detected. All nodes use the 16-QAM modulation and coding scheme, which provides 24 Mbps channel bit rate, as calibration measurements showed that this was the highest rate achievable by the node with the worst link to the AP (node 4). As we show in the next subsections, this placement supports the emulation of realistic channel conditions, as the channel between the nodes and the AP is not equalized but instead the so called *capture effect* is present (as we discuss in Sec. 5.2.3), and we also succeeded in replicating *hidden nodes* scenarios (Sec. 5.2.4).

Unless otherwise specified, all nodes use the same transmission power level of 15 dBm,

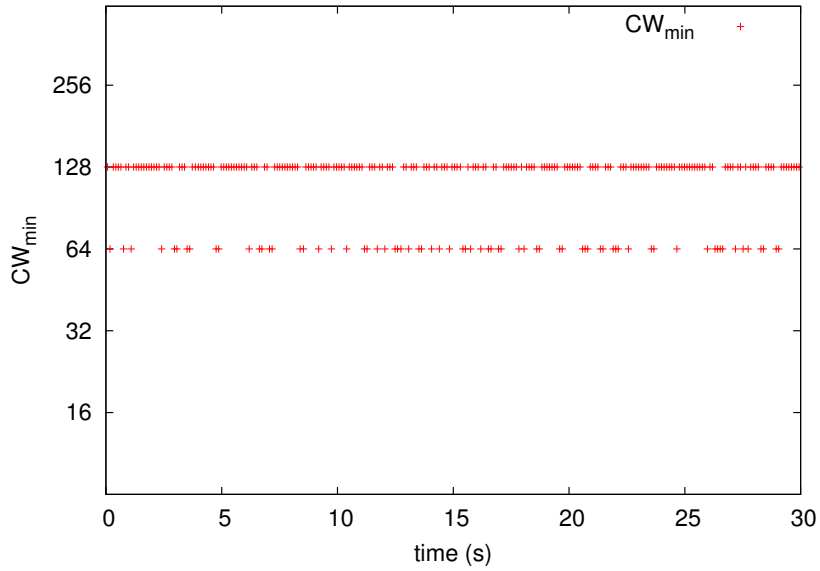

Figure 5.6: CAC: Announced CW_{min}

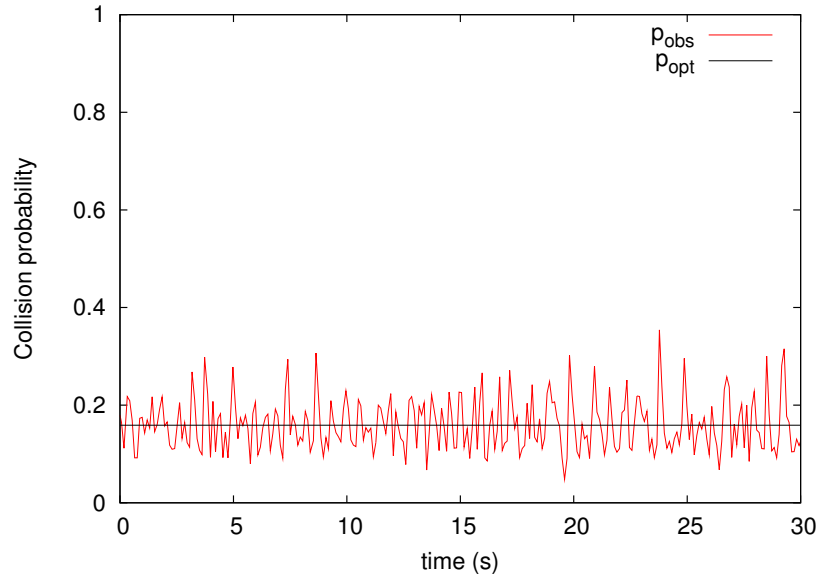

Figure 5.7: CAC: Observed collision probability

and generate UDP traffic towards the AP utilizing the `iperf`[5] tool. We fixed the size of L2 frames to 1500 bytes, and the duration of each measurement to 2 minutes, which were repeated 5 times to obtain average values of the measured metrics with good statistical meaning. Additionally, prior to any measurement we used the `iwpriv` command to disable the RTS/CTS, turbo, fast frame, bursting and unscheduled automatic power save delivery functionality, as well as the antenna diversity scheme for transmission/reception.

[5] `http://sourceforge.net/projects/iperf/`

5.2.2 Practical Validation of the Algorithms' Operation

Our first set of experiments aims at confirming that the good operation properties of *CAC* and *DAC*, previously demonstrated analytically and via simulations, are also achieved in a real testbed. Specifically, we want to assess if the use of the algorithms results in stable behavior, despite the described hardware/software limitations and the impairments introduced by the varying channel conditions. Our tests involve the $N = 11$ stations in the testbed transmitting at the same time.

Operation of *CAC*. First, we focus on the performance of our centralized algorithm. We set all 11 stations to send traffic to the AP, and log the key variables of the algorithm, namely, the `CW` announced with the beacon frames and the observed collision probability p_{obs}. Both are obtained every 100 ms and depicted in Fig. 5.6 and Fig. 5.7, respectively. In Fig. 5.7 we also plot the target collision probability p_{opt} as given by Eq. (3.20).

The figures show that the *CAC* algorithm drives the WLAN to the desired point of operation. Indeed, the announced `CW` oscillates between two values, i.e., the power of two closest to the optimal CW_{min}, whilst p_{obs} fluctuates stably around the desired p_{opt}. We conclude that, despite the hardware limitations imposed on the values of `CW` and the channel impairments, *CAC* is able to drive the WLAN to the desired point of operation.

Operation of *DAC*. Next, we track the operation of the distributed algorithm. We proceed as before, logging the key parameters of the algorithm, namely $CW_{min,i}$, $p_{own,i}$ and $p_{obs,i}$. In Fig. 5.8 we depict the evolution of the `CW` used by four representative nodes (namely 2, 3, 8 and 9), while in Fig. 5.9 we plot the collision probabilities (p_{obs} and p_{own}) estimated by node 2.

From the figures we can see that, similarly to *CAC*, *DAC* also drives the average collision probability in the WLAN to the desired value. However, there is a key difference as compared to the previous case: while with *CAC* all stations operate at the same *CW* value, with *DAC* they operate at different average *CW*s. Indeed, the four stations considered in the experiment use average *CW*s values of 92, 300, 92 and 64, respectively. This behavior is caused by the fact that the stations that are closer to the AP and benefit from the capture effect observe a smaller collision rate, which triggers an increase in their *CW*.[6] This behavior, along with its impact on the resulting throughput distribution, is discussed in the next subsection with greater detail.

Resource consumption. In addition to analyzing the performance of *CAC*, it is also important to asses their demand for computational resources. For this purpose, we analyzed the CPU and memory usage of the algorithms at the AP and stations, respectively, utilizing the `top` Linux application, which provides a dynamic real-time view

[6]It is worthwhile to note that this behavior is not specific to our algorithm, but would be present in any distributed algorithm that uses similar statistics.

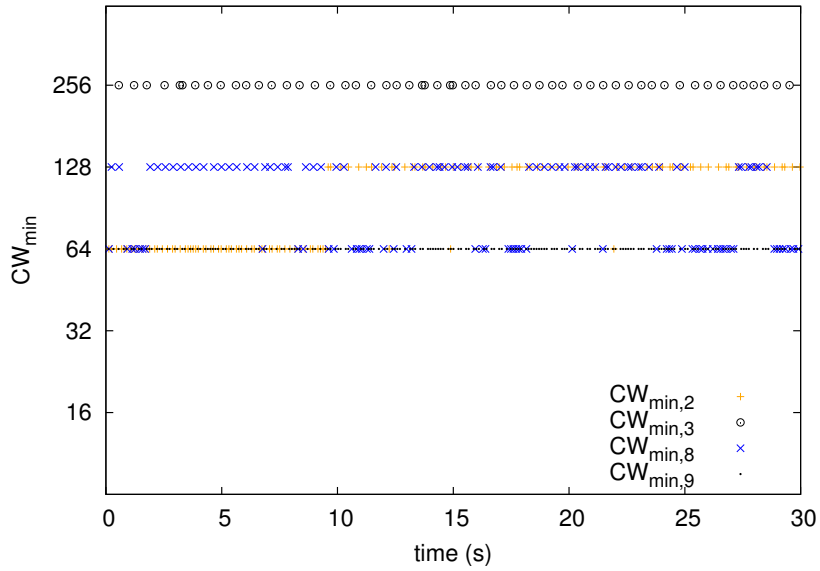

Figure 5.8: DAC: CW_{min} used by four nodes

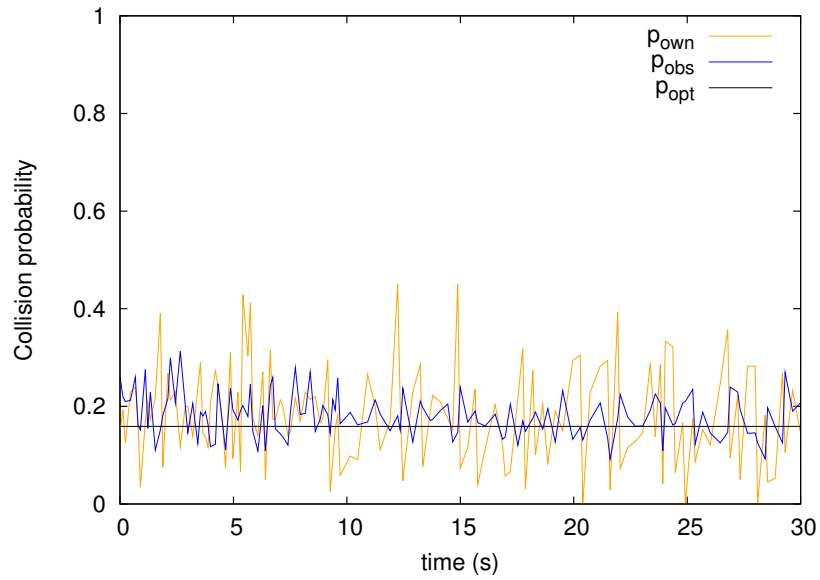

Figure 5.9: DAC: Estimated p_{obs} and p_{own}

of a running system. With this tool, we recorded the used shares of the total CPU time and available physical memory with a frequency of 1 second and computed the average usage. *CAC* demands on average 39% of the CPU time and only 1.6% of the physical memory. Conversely, *DAC* consumes in average 28% of the total CPU time, while utilizing 4.3% of the available memory. Given the low speed of the nodes' CPU (233 MHz) and their reduced physical memory (128 MB), these results show that *CAC* is suitable for commercial APs, while *DAC* will not affect the performance of current portable devices which employ faster CPUs.

5.2.3 Impact of Link Quality on Throughput Distribution

We carried out comparative tests of the throughput performance achieved using three mechanisms: *CAC*, *DAC* and the recommended standard configuration (denoted as EDCA). The latter corresponds to the normal operation of current WLANs and we use it as a benchmark against which to assess the improvements provided by our algorithms.

First, we considered the case where all stations use the transmission power (15 dBm); given the node placement of Fig. 5.5, we expect this to result in very dissimilar link qualities between each station and the AP (e.g., node 3 is extremely close, both absolutely and relatively). Next, we performed exhaustive measurements in order to equalize the link qualities, these resulting in quasi-homogeneous channel conditions.

Heterogeneous link qualities. Using the same setting of 15 dBm for the transmission power of all nodes, we measured the `iperf` throughput between each station and the AP when all stations are transmitting at the same time. The total throughput obtained for each mechanism is depicted in Fig. 5.10, where we observe that the use of the EDCA default configuration achieves below 14 Mbps, while the use of *DAC* and *CAC* improves performance by approximately 19% and 21%, respectively. Therefore, we confirm that the use of the adaptive algorithms results in higher efficiency, as they are able to adapt to the (relatively large) number of contending stations.

To better examine the obtained performance, we plot the per-station throughput in Fig. 5.11. Here, the results provide a deeper understanding of the resulting performance, which can be summarized as follows:

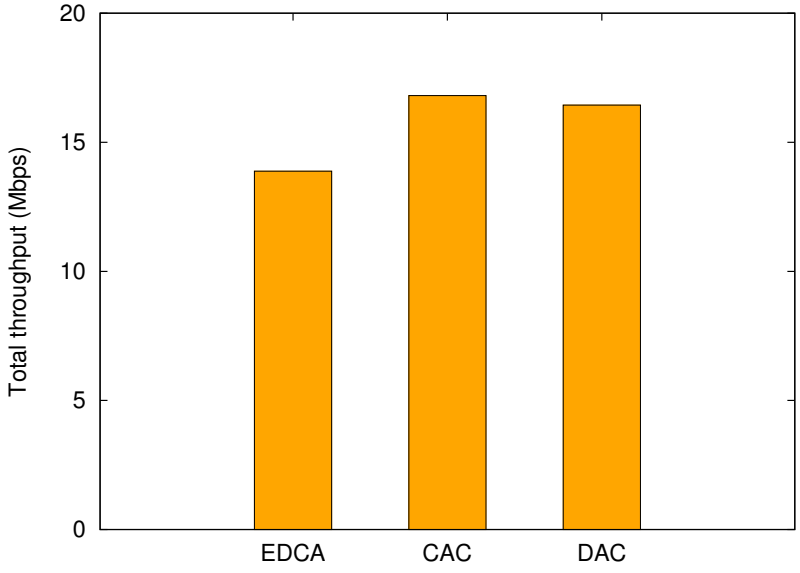

Figure 5.10: Total throughput with heterogeneous link qualities

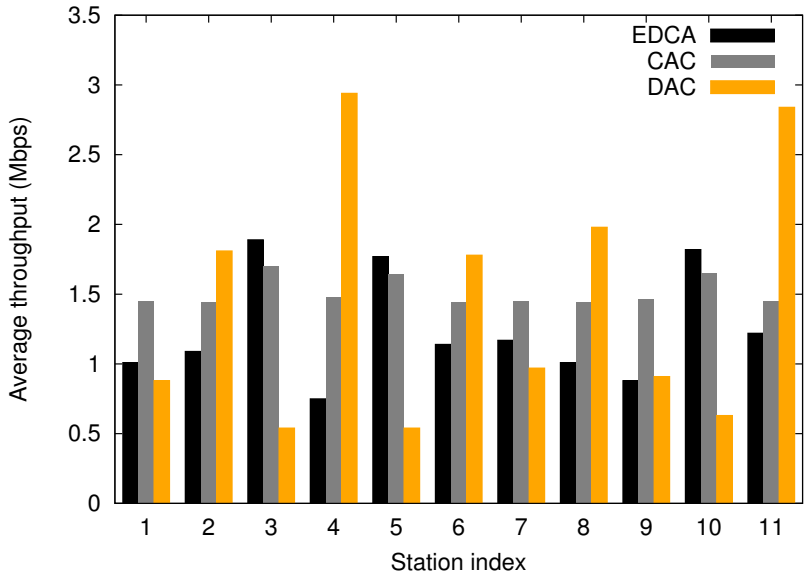

Figure 5.11: Throughput per station with heterogeneous link qualities

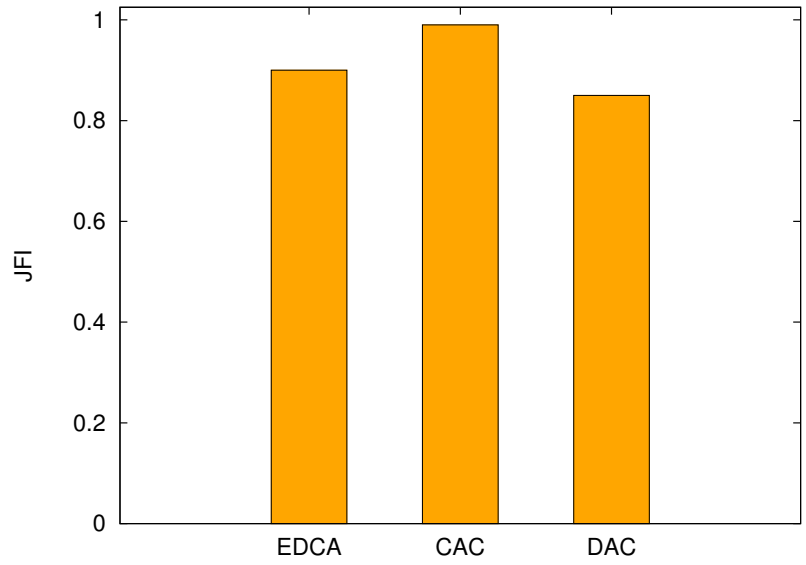

Figure 5.12: Jain's fairness index with heterogeneous link qualities

- The use of the EDCA recommended values not only provides the lowest overall throughput figures, but also fails to provide a fair bandwidth share. Indeed, it can be seen that station 3, the closest to the AP, achieves the largest throughput, this being more than twice the throughput obtained by station 4 (which is farther form the AP).

- The use of *DAC*, despite providing a larger overall figure than EDCA, also fails to achieve fairness. Actually, it results in a somehow *opposite* performance as compared

to EDCA: stations that obtained a relatively large bandwidth with EDCA, obtain a relatively small bandwidth with *DAC*.

- The use of *CAC* provides the best performance in terms of fairness. Despite the binary exponential backoff is still active (like in the case of EDCA), the announced CW_{min} values are able to significantly lessen the impact of the heterogeneous channel conditions.

The above throughput fairness results are quantified using Jains fairness index (JFI) [84] in Fig. 5.12. This figure confirms the good fairness properties of *CAC*, which achieves a value of 0.99. The figure also shows that *DAC* suffers a significant level of unfairness (even higher than EDCA). The reason for this behavior is the inability of the MAC layer to distinguish between collision-free transmissions and those that benefit from the capture effect. In particular, with *DAC* a station that is close to the AP and always captures the channel will (wrongly) assume that its lower collision rate is due to the fact that it uses a smaller CW, and it will react by increasing its CW, this resulting in a lower throughput for this station.

To confirm that the heterogeneous conditions are the reason for the observed unfairness, we next try to equalize the channel between each station and the AP and repeat the experiments, in order to restore fairness.

Quasi-homogeneous link qualities. To equalize the link qualities between each station and the AP we used the following approach. We fixed the transmission power of the station with the poorest link (node 4) to the maximum value (i.e., 17 dBm). Then,

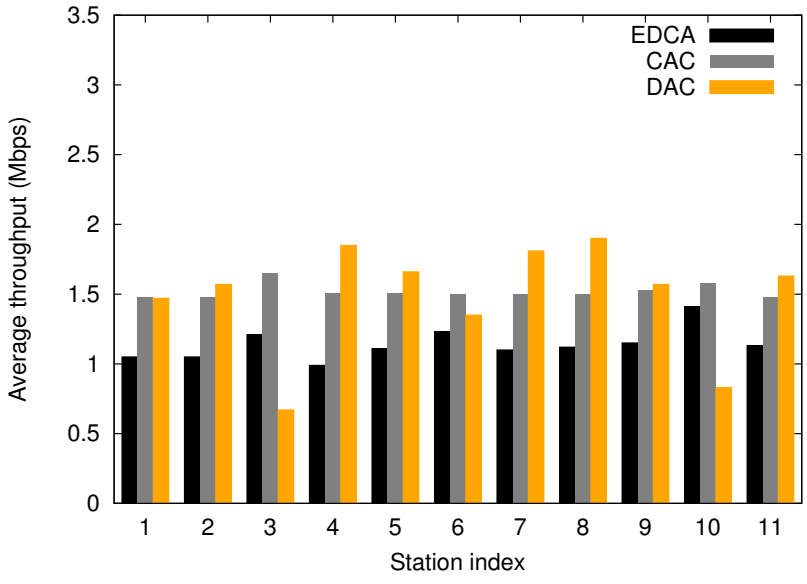

Figure 5.13: Throughput per station with quasi-homogeneous link qualities

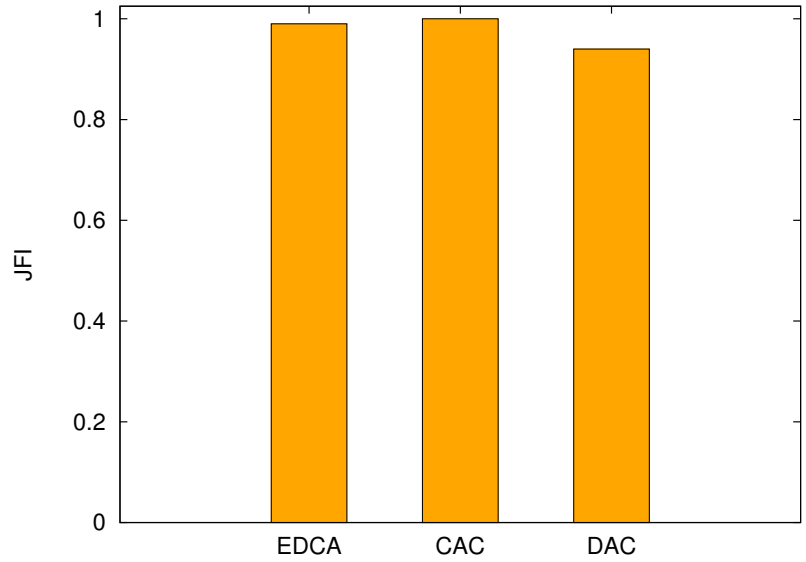

Figure 5.14: Jain's fairness index with quasi-homogeneous link qualities

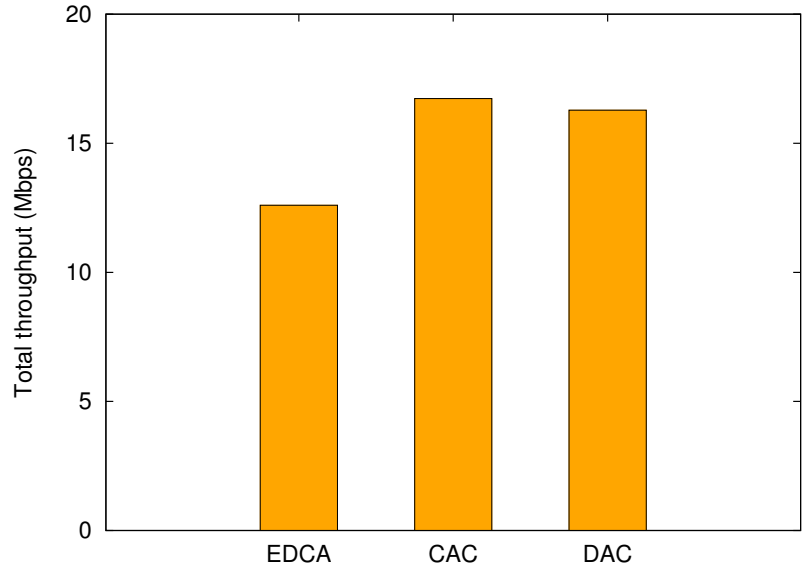

Figure 5.15: Total throughput with quasi-homogeneous link qualities

we carefully tuned the transmission power of the rest of the stations, using the following algorithm: (1) we set the power to the minimum value; (2) we compute the throughput obtained when both nodes (the station being configured and node 4) are the only one transmitting; (3) we increase the transmission power used at step 1 and repeat step 2, until both throughputs are similar (less than 10% of difference) or the maximum power is reached.

The resulting values of the throughput per station for the three different approaches

are illustrated in Fig. 5.13. The figure shows that the obtained throughputs are much more equalized, with the absence of "spikes" above 2 Mbps. Indeed, while for the case of *CAC* the improvement is not very significant, for both the cases of EDCA and *DAC* the discrepancies between rates are much smaller. We also observe the same asymmetric behavior between these two approaches: those nodes achieving a relatively better performance with EDCA (e.g., nodes 3, 10) are *penalized* by *DAC*, while nodes with the worse performance under EDCA (e.g., node 4, 5) achieve a better performance with *DAC*.

The fairness figures, shown in Fig. 5.14, confirm that a careful tuning of the transmission power can significantly reduce channel impairments due to heterogeneous link qualities. It is interesting to observe that the improvement is practically negligible for the case of *CAC*, while the other two approaches greatly benefit from the careful power setting. It should be noted, though, that the algorithm used to achieve this quasi-homogeneous link qualities is very time-consuming, and practically unfeasible in a real-life scenario (i.e., a hotspot).

Finally, we compute the total throughput obtained with the three approaches in the quasi-homogeneous scenario. Results are given in Fig. 5.15. As compared with the case of heterogeneous channel conditions, it is interesting to observe that while both *DAC* and *CAC* seem oblivious to the link qualities, EDCA exhibits *a reduction in the total throughput* (approximately 10 %). The reason for this behavior is that with the careful tuning of the power settings, stations are less likely to benefit from the capture effect, and therefore collisions will have a larger impact on the channel efficiency. This effect is less noticeable with *CAC* and *DAC*, as they both try to reduce the number of collisions in the WLAN.

5.2.4 Impact of Hidden Nodes

The proposed adaptive mechanisms were designed with the assumption that all stations are in radio range and can coordinate their transmission attempts by using the carrier sense mechanism. However, this may not always be true in real deployments, as in the case of hidden nodes, whereby stations are not in range and cannot coordinate their transmissions, which leads to additional collisions on the channel. In order to test the behavior of *CAC* and *DAC* under hidden nodes, we designed an experiment with three stations: the AP and two stations, whose transmission power was set such that they had a good channel to the AP, but were hidden from each other.

We used node 3 as AP, and nodes 2 and 8 as stations, and set the transmission power of nodes 2 and 8 to 5 dBm. We observed that each station, when transmitting in isolation, i.e., the other station being silent, achieves 17.4 Mbps of throughput without experiencing any channel error and hence any MAC retransmission. When transmitting together, the

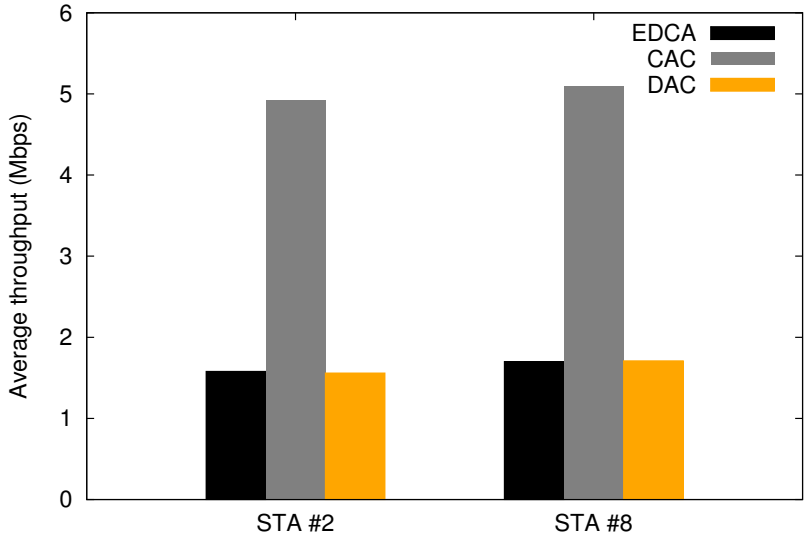

Figure 5.16: Throughput performance with hidden nodes

two stations receive about 1.5 Mbps each (see Fig. 5.16), the channel quality of the two stations being practically the same. Therefore, we concluded that we managed to replicate hidden node conditions.

We then repeated the experiment with *CAC* and *DAC*. The results are depicted in Fig. 5.16. (Apart from minor channel quality fluctuation, the throughput received by the two stations was practically the same in all the experiments, which were repeated 5 times.) Noticeably, while using *DAC* does not give appreciable advantages over EDCA, when employing *CAC*, a dramatic throughput increase is achieved, i.e., more than three times the throughput attained with EDCA.

The centralized approach is able to detect excessive collisions and command hidden nodes to be less aggressive, i.e., to use a higher CW_{min}, which will leverage, not eliminate, the hidden node problem. On the other hand, a station running *DAC* is not able to overhear MAC (re-)transmissions from hidden nodes, and hence cannot correctly estimate p_{obs}.

We conclude that *CAC* is able to alleviate the hidden node problem, while *DAC* does not provide any enhancement is this scenario, but has no negative impact on the WLAN performance.

5.2.5 Impact of Network Size

Next, we evaluated the performance of *CAC* and *DAC* as a function of the number of active stations. To this aim, we measured throughput and fairness for an increasing

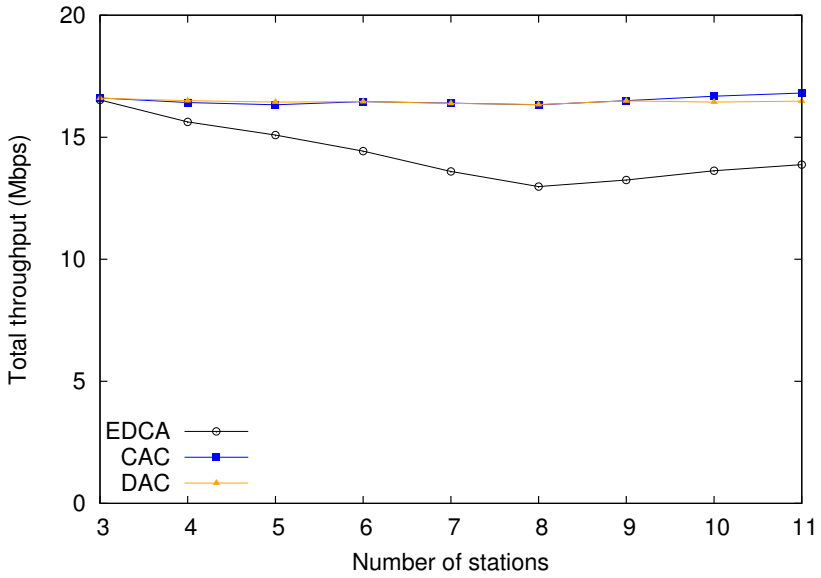

Figure 5.17: Total throughput for different number of stations

number of stations. When increasing the number of stations, we started with the ones with the poorest link quality and added new stations in order of increasing link quality.

Fig. 5.17 plots the total throughput obtained vs. the number of contending stations when they are added in the aforementioned order. We draw the following main observations:

- For both the case of *DAC* and *CAC* the total throughput performance is relatively flat, regardless of the number of stations. This result confirms that the approaches are able to optimize throughput performance by adjusting the CW to the number of stations present in the WLAN.
- For the case of EDCA, initially performance degrades with the number of stations, which is the expected result from the use of a fixed set of (relatively small) contention parameters. However, for $N \geq 8$ stations, the total throughput performance starts to grow again.

Note that, the larger the number of stations, the more nodes benefit form uneven channel conditions. In particular, EDCA throughput starts to grow when node 10 is added to the experiment. According to Fig. 5.11, this node is among the ones that mostly benefit from the heterogeneous link qualities. Based on this observation, we argue that the improved performance is caused by the dissimilar channel conditions.

To gain insight on the throughput distribution, we also compute the JFI, which we illustrate in Fig. 5.18. The results confirm our conjecture: EDCA is fair for $N \leq 8$ nodes.

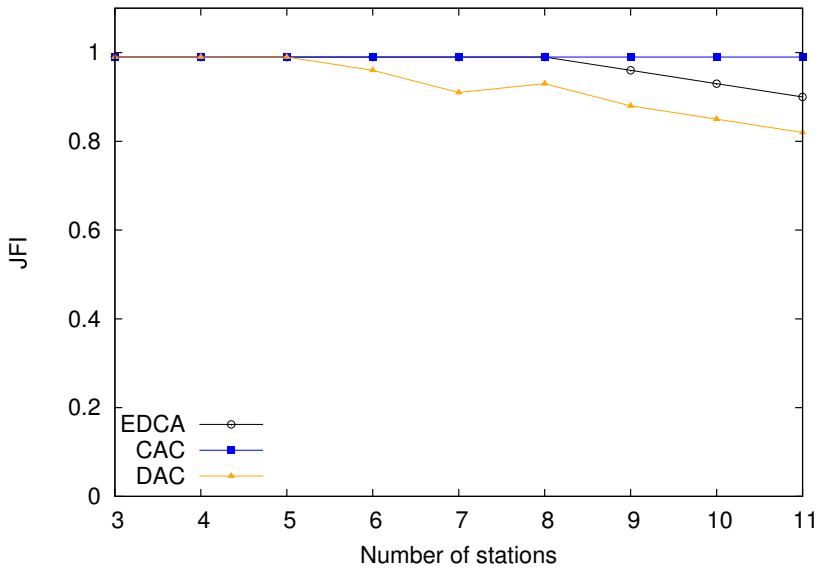

Figure 5.18: Jain's fairness index for different number of stations

From this point on, i.e., with the addition of the nodes benefiting from the capture effect, the throughput distribution becomes more unfair. It is also worth mentioning that *DAC*, as seen in Sec. 5.2.3, exacerbates the problems that arise from heterogeneous link conditions, as the fairness index decreases for $N \geq 6$. Finally, the experiments confirm the good properties of *CAC*, as the fairness index is practically constant for all N values.

5.2.6 Impact of Dynamic Traffic Conditions

In the previous subsections we have only considered fixed traffic conditions, whereby a given experiment was defined with a static number N of constantly–backlogged (i.e., saturated) stations. Here, we are interested in analyzing the performance of *CAC* and *DAC* under dynamic conditions, where the number of contending stations changes over time, in order to assess their performance and to confirm the validity of the proposed configuration of the underlying PI controllers. We analyze two cases: *(i)* first we evaluate the two schemes when the number of active stations in the WLAN changes, and *(ii)* next we relax the saturation conditions, considering stations that have active and silent periods.

Varying the number of stations. A key property of a properly configured PI controller is its ability to react fast to changes. To validate that *CAC* and *DAC* achieve this property with the designed $\{K_p, K_i\}$ configuration, we conducted two experiments, in which network conditions changed significantly over time. More specifically, we considered a scenario with 5 active stations sending traffic and 5 additional stations joining the WLAN after 15 seconds.

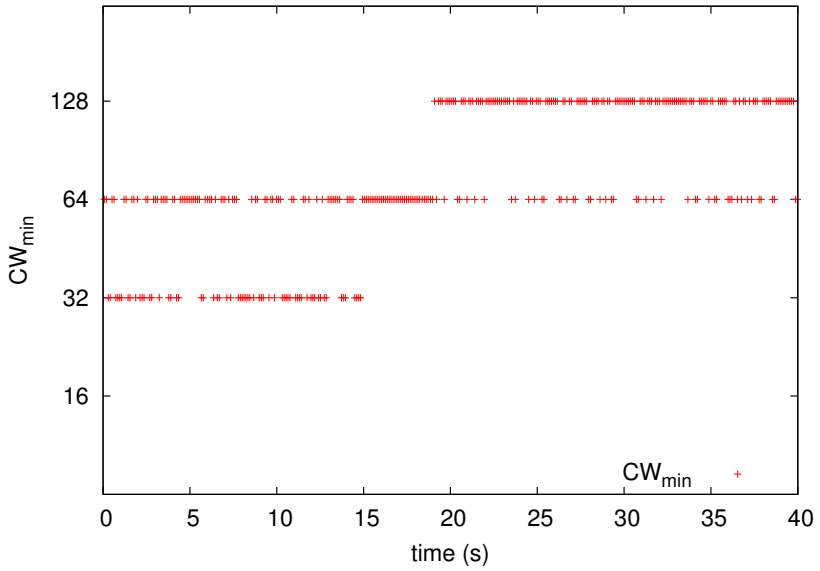

Figure 5.19: CAC: CW_{min} behavior under increasing number of stations

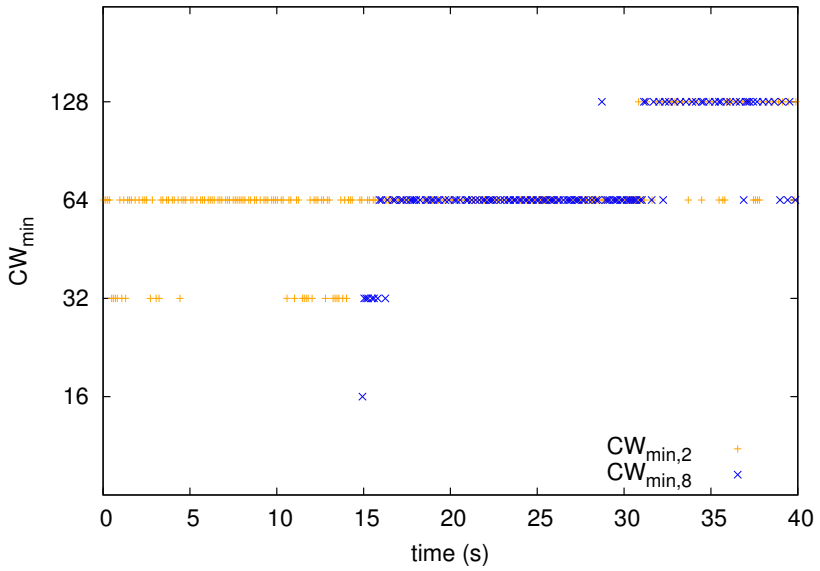

Figure 5.20: DAC: CW_{min} behavior under increasing number of stations

Under these dynamic conditions for the case of *CAC* we monitored the CW_{min} configuration at the output of the controller running at the AP. For the case of *DAC* we tracked the CW_{min} value used by one of the stations initially present in the network (node 2) and the behavior of a station activated later (node 8). As depicted in Fig. 5.19, *CAC* immediately detects the change at $t = 15$ and hinders the oscillation between the 32 and 64 values. Moreover, the system moves to the optimal point of operation and stations are provided with the new *CW* configuration within few seconds. Also, in the case of *DAC*, the reaction to changes is triggered instantly, both the newly added sta-

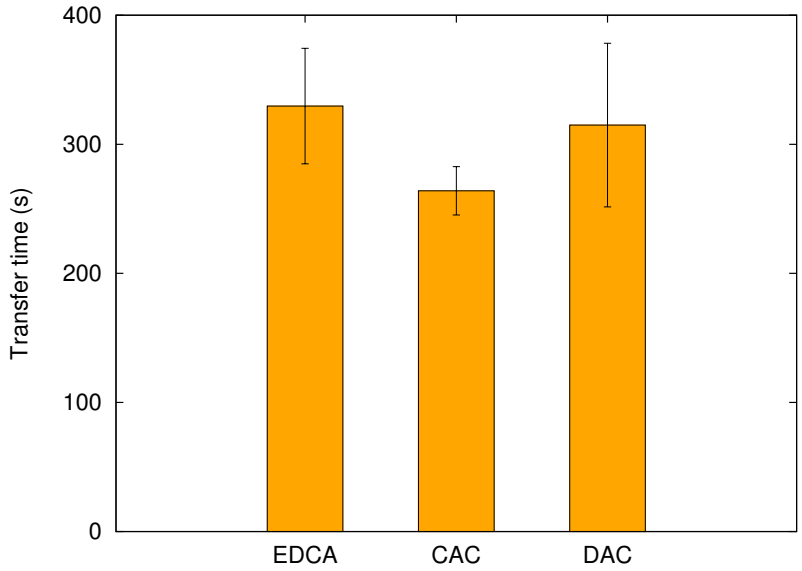

Figure 5.21: Delay performance under dynamic load conditions

tions and the already existing ones detecting the increased collision rate and increasing their CW_{min} values, as shown in Fig. 5.20. However, as compared to centralized scheme, the distributed algorithm requires additional time to reach the desired optimal point of operation, which further highlights the advantages of CAC over the DAC algorithm.

Varying the traffic load. Next we analyze the performance of CAC and DAC under dynamic traffic load conditions. For this purpose, we devised the following experiment. We considered a scenario in which each of the 11 stations initiates 5 successive transfers of a 50 Mbyte file to the AP, with random idle periods uniformly distributed in the [0,20] seconds interval, after each file transfer is completed.

We measured the average and standard deviation of the total duration of a file transfer when CAC, DAC and the default EDCA configuration, respectively, are employed. As shown in Fig. 5.21, CAC significantly improves the average transmission time, while DAC slightly outperforms the default EDCA configuration, but presents a larger standard deviation of the delay. We confirm that the centralized scheme is more agile to variable traffic load and provides users with better delay performance as compared to the standard's configuration.

5.3 Summary

In this chapter, we have addressed the implementation and experimental evaluation of the proposed centralized and distributed algorithms that optimize WLAN performance. These mechanisms stand out due to their solid mathematical foundations that provide

performance guarantees, which makes them good candidates for real-life deployments. We have analyzed their performance in a medium-size testbed built with COTS hardware and open-source drivers. Through extensive experimentation, we (*i*) confirmed their good properties in terms of total throughput, outperforming the standard 802.11 configuration; (*ii*) proved their ability to adapt to dynamic network conditions; and (*iii*) analyzed the impact of link qualities on their performance. We have shown that *CAC* significantly outperforms the IEEE 802.11 standard even in the presence of hidden nodes and we have identified the reasons that jeopardize the performance of *DAC*. Although the experimental study conducted in this chapter has focused on two specific algorithms, we believe that most of the conclusions drawn herein can be generalized to any centralized or distributed algorithm.

Chapter 6

Conclusions and Future Work

In this thesis we have proposed novel centralized and distributed adaptive algorithms based on analytical models of the IEEE 802.11 protocol behavior, that drive the EDCA configuration of the wireless stations to the optimal point of operation that maximizes performance in terms of throughput and delay. The proposed algorithms are sustained by mathematical foundations from single-/multi-variable control theory, which guarantee their stability and convergence.

The proposed *Centralized Adaptive Control (CAC)* algorithm dynamically adjusts the *CW* configuration of the IEEE 802.11 stations, with the goal of maximizing the overall throughput performance of the wireless network. To design our algorithm, we first analyzed the behavior of a WLAN and derived the collision probability that optimizes the throughput. A key finding of our analysis is that the optimal collision probability can be approximated by a constant, independent of number of stations. This allows to steer our system by means of a controller, which takes the value of the optimal collision probability as the reference signal and drives the collision probability of the WLAN to this optimal value.

To configure the parameters of the controller, we linearized the WLAN behavior and we conducted a control theoretic analysis that guarantees a proper tradeoff between stability and speed of reaction to changes. In contrast, the existing adaptive approaches obtain the configuration of the involved parameters either heuristically or empirically, thus lacking such guarantees. Moreover, unlike the existing schemes, *CAC* is fully compatible with the IEEE 802.11 standard and the dynamic tuning of the configuration solely relies on analyzing the headers of the successfully received frames at the Access Point.

Likewise, we modeled the WLAN behavior under video traffic and extended our centralized approach, proposing *CAC–VI*, which dynamically adjusts the *CW* configuration of the WLAN, with the goal of minimizing the average delay. The key insight upon which our algorithm relies is the observation that the collision probability that yields

the minimum delay can be computed using estimates of the arrival rate and packet size distribution, which can be easily obtained at the AP by simply monitoring the correctly received frames. Based on this observation, we applied a controller that drives the collision probability to this reference value and thus optimizes the delay performance of the WLAN.

The proposed *Distributed Adaptive Control (DAC)* algorithm shares the same goal of maximizing the overall performance as the centralized scheme, but is independently executed by each station and uses only locally available information to drive the collision probability in the WLAN to the optimal value. While it is easy to guarantee convergence with centralized schemes, where the behavior of all stations is controlled by a single node, ensuring convergence in a distributed system, whereby each nodes acts independently, is not straightforward. We addressed this challenge by carefully choosing the control signal that determines the configuration of each station, which is composed of two terms that ensure the collision probability in the network is driven to the optimal value, while stations share the bandwidth fairly.

We have conducted a steady-state analysis of our system to guarantee that the two terms comprising the error signal do not cancel each other. As the system relies on a number of independent variables, namely the configuration of each station, we have used techniques from multivariable control theory in order to configure the parameters of the controllers. From this analysis, we have first obtained the stability region of the parameter values, and then we have chosen a configuration within this region that provides a proper tradeoff between stability and speed of reaction to changes.

We have extensively evaluated *CAC* and *DAC* by means of simulations conducted under a wide set of scenarios, including saturation, non-saturation, mixed traffic conditions, etc. The obtained results confirm that the proposed algorithms achieve their goal of optimizing the relevant performance metrics, i.e. throughput and delay, and, moreover, they substantially outperform both the standard IEEE 802.11 mechanism and previous adaptive proposals. We also evaluated the impact of the configuration of the involved parameters and showed that, with our settings, the systems are stable, they converge to the same point of operation and are able to rapidly track changes in the network conditions, while other settings would fail to provide these properties.

In addition to these results, the experimental study we undertook by implementing our algorithms with COTS devices and open-source drivers in a medium scale testbed has proven that our proposals can be executed by current devices without introducing any modifications into their hardware and/or firmware, and has given further insights on their performance. In particular, the experimental results have shown that centralized approaches are more robust to non-ideal channel effects, thereby providing a substantially improved performance over distributed schemes.

In the future, we plan to extend our work and address the performance optimization of real-time traffic in a distributed manner, but also tackle the joint optimization of data and real-time traffic, both with a centralized and distributed approach. Furthermore, we intend to extend our experimental analysis by enlarging our testbed and gain further insights on the behavior of our approaches in real scenarios that involve a larger number of nodes. We aim to extended our proposals to investigate their performance under TCP traffic and lessen the impact of asymmetries, which can cause fairness issues at the AP, since in such scenarios the AP has to simultaneously acknowledge session from different clients [89]. From an experimental perspective, we also plan to further validate our algorithms in a deployment with real users that exchange heterogeneous and more dynamic traffic.

References

[1] IEEE 802.11, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. Revision of IEEE Std 802.11-1999, 2007.

[2] IEEE 802.11, *Wireless LAN Medium Access Control (MAC) And Physical Layer (PHY) Specifications*. 1997.

[3] IEEE 802.11b, *Wireless LAN Medium Access Control (MAC) And Physical Layer (PHY) Specifications: Higher-speed Physical Layer Extension In The 2.4 GHz Band*. 1999.

[4] IEEE 802.11, *Supplement to Wireless LAN Medium Access Control and Physical Layer specifications: high-speed physical layer in the 5 GHz band*. IEEE Std 802.11a, 1999.

[5] IEEE 802.11g, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 4: Further Higher Data Rate Extension in the 2.4 GHz Band*. Supplement to IEEE 802.11 Standard, 2003.

[6] IEEE 802.11 WG, *Amendment to Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: MAC Enhancements for Quality of Service (QoS)*. Supplement to IEEE 802.11 Standard, 2005.

[7] G. Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function," *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 535–547, March 2000.

[8] A. Banchs, X. Pérez-Costa, and D. Qiao, "Providing Throughput Guarantees in IEEE 802.11e Wireless LANs," in *Proc. International Teletraffic Congress (ITC)*, (Berlin, Germany), 2003.

[9] A. Banchs and L. Vollero, "Throughput Analysis and Optimal Configuration of 802.11e EDCA," *Computer Networks*, vol. 50, August 2006.

[10] A. Nafaa and A. Ksentini and A. Ahmed Mehaoua and B. Ishibashi and Y. Iraqi and R. Boutaba, "Sliding Contention Window (SCW): Towards Backoff Range-Based Service Differentiation over IEEE 802.11 Wireless LAN Networks," *IEEE Network*, vol. 19, pp. 45–51, July 2005.

[11] J. Freitag and N. L. S. da Fonseca and J. F. de Rezende, "Tuning of 802.11e Network Parameters," *IEEE Communications Letters*, vol. 10, pp. 611–613, August 2006.

[12] L. Scalia, I. Tinnirello, J. Tantra, and C. H. Foh, "Dynamic MAC Parameters Configuration for Performance Optimization in 802.11e Networks," in *Proc. GLOBECOM*, (San Francisco, CA, USA), pp. 1–6, Dec 2006.

[13] Chen, Wen-Tzu, "An effective medium contention method to improve the performance of IEEE 802.11," *Wireless Networks*, vol. 14, no. 6, pp. 769–776, 2008.

[14] V. A. Siris and G. Stamatakis, "Optimal CWmin selection for achieving proportional fairness in multi-rate 802.11e WLANs: test-bed implementation and evaluation," in *Proc. International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization (WINTECH)*, (Los Angeles, CA, USA), pp. 41–48, 2006.

[15] G. Bianchi, L. L. Fratta, and M. Oliveri, "Performance evaluation and enhancement of the CSMA/CA MAC protocol for 802.11 wireless LANs," in *Proc. IEEE Personal, Indoor, and Mobile Radio Communications Conference (PIMRC)*, (Taipei, Taiwan), pp. 392–396, October 1996.

[16] M. Heusse, F. Rousseau, R. Guillier, and A. Duda, "Idle sense: an optimal access method for high throughput and fairness in rate diverse wireless LANs," in *Proc. ACM SIGCOMM*, (Philadelphia, Pennsylvania, USA), pp. 121–132, 2005.

[17] Q. Xia and M. Hamdi, "Contention Window Adjustment for IEEE 802.11 WLANs: A Control-Theoretic Approach," in *Proc. International Conference on Computer Communications (ICC)*, (Istanbul, Turkey), June 2006.

[18] F. Cali, M. Conti, and E. Gregori, "IEEE 802.11 protocol: design and performance evaluation of an adaptive backoff mechanism," *IEEE Journal on Selected Areas in Communications*, vol. 18, September 2000.

[19] Y. Yang, J. J. Wang, and R. Kravets, "Distributed Optimal Contention Window Control for Elastic Traffic in Single-Cell Wireless LANs ," *IEEE/ACM Transactions on Networking*, vol. 15, pp. 1373–1386, December 2007.

[20] L. Bononi, M. Conti, and E. Gregori, "Runtime optimization of ieee 802.11 wireless lans performance," *IEEE Transactions on Parallel and Distributed Systems*, vol. 15, pp. 66–80, January 2004.

[21] L. Gannoune and S. Robert, "Dynamic Tuning of the Contention Window Minimum (CWmin) for Enhanced Service Differentiation in IEEE 802.11 Wireless Ad-Hoc Networks," in *Proc. IEEE Personal, Indoor, and Mobile Radio Communications Conference (PIMRC)*, (Barcelona, Spain), Sep 2004.

[22] P. Patras, A. Banchs, and P. Serrano, "A Control Theoretic Approach for Throughput Optimization in IEEE 802.11e EDCA WLANs," *Mobile Networks and Applications (MONET)*, vol. 14, pp. 697–708, December 2009.

[23] P. Patras, A. Banchs, and P. Serrano, "A Control Theoretic Framework for Performance Optimization of IEEE 802.11 Networks," in *IEEE INFOCOM Workshops*, (Rio de Janeiro, Brazil), pp. 1–2, April 2009.

[24] P. Patras, A. Banchs, and P. Serrano, "A Control Theoretic Scheme for Efficient Video Transmission over IEEE 802.11e EDCA WLANs," *submitted manuscript*, November 2010.

[25] P. Patras, A. Banchs, P. Serrano, and A. Azcorra, "A Control Theoretic Approach to Distributed Optimal Configuration of 802.11 WLANs," *IEEE Transactions on Mobile Computing*, vol. 99, December 2010.

[26] P. Serrano, A. Banchs, P. Patras, and A. Azcorra, "Optimal Configuration of 802.11e EDCA for Real-Time and Data Traffic," *IEEE Transactions on Vehicular Technology*, vol. 59, pp. 2511–2528, June 2010.

[27] P. Serrano, P. Patras, V. Mancuso, A. Mannocci, and A. Banchs, "Adaptive Mechanisms for Performance Optimization of IEEE 802.11 WLANs: Implementation Experiences and Practical Evaluation ," *submitted manuscript*, December 2010.

[28] J. W. Robinson and T. S. Randhawa, "Saturation Throughput Analysis of IEEE 802.11e Enhanced Distributed Coordination Function," *IEEE Journal on Selected Areas in Communications*, vol. 22, pp. 917–928, June 2004.

[29] Z. Kong, D. H. K. Tsang, B. Bensaou, and D. Gao, "Performance Analysis of IEEE 802.11e Contention-Based Channel Access," *IEEE Journal on Selected Areas in Communications*, vol. 22, pp. 2095–2106, December 2004.

[30] V. Ramaiyan, A. Kumar, and E. Altman, "Fixed point analysis of single cell IEEE 802.11e WLANs: uniqueness, multistability and throughput differentiation," in *Proc. ACM SIGMETRICS*, June 2005.

[31] J. Hui and M. Devetsikiotis, "A unified model for the performance analysis of IEEE 802.11e EDCA," *IEEE Transactions on Communications*, vol. 53, pp. 1498–1510, September 2005.

[32] H. Zhu and I. Chlamtac, "Performance Analysis for IEEE 802.11e EDCF Service Differentiation," *IEEE Transactions on Wireless Communications*, vol. 4, June 2005.

[33] A. Banchs and L. Vollero, "A Delay Model for IEEE 802.11e EDCA," *IEEE Communications Letters*, vol. 9, pp. 508–510, June 2005.

[34] H. Wu, Y. Peng, K. Long, S. Cheng, and J. Ma, "Performance of reliable transport protocol over IEEE 802.11 wireles LAN: Analysis and enhancement," in *Proc. IEEE INFOCOM*, June 2002.

[35] H. Y. Hwang, S. J. Kim, D. K. Sung, and N.-O. Song, "Performance analysis of ieee 802.11e edca with a virtual collision handler," *IEEE Transactions on Vehicular Technology*, vol. 57, pp. 1293–1297, March 2008.

[36] P. E. Engelstad and O. N. Osterbo, "Non-saturation and saturation analysis of IEEE 802.11e EDCA with starvation prediction," in *Proc. ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems (MSWIM)*, October 2005.

[37] G.-R. Cantieni, Q. Ni, C. Barakat, and T. Turletti, "Performance Analysis of Finite Load Sources in 802.11b Multirate Environments," *Computer Communications*, vol. 28, pp. 1095–1109, June 2005.

[38] J. W. Tantra, C. H. Foh, I. Tinnirello, and G. Bianchi, "Analysis of the IEEE 802.11e EDCA under Statistical Traffic," in *Proc. International Conference on Computer Communications (ICC)*, (Istanbul, Turkey), June 2006.

[39] K. Duffy, D. Malone, and D. Leith, "Modeling the 802.11 Distributed Coordination Function in Non-saturated Conditions," *IEEE Communications Letters*, vol. 9, no. 8, 2005.

[40] B. Li and R. Battiti, "Analysis of the 802.11 DCF with Service Differentiation Support in Non-saturation Conditions," in *Proc. International Workshop on Quality of future Internet Services (QofIS)*, September 2005.

[41] C. Foh, M. Zukerman, and J. Tantra, "A Markovian Framework for Performance Evaluation of IEEE 802.11," *IEEE Transactions on Wireless Communications*, vol. 6, no. 4, pp. 1276–1265, 2007.

[42] P. Serrano, A. Banchs, and A. Azcorra, "A throughput and delay model for IEEE 802.11e EDCA under non saturation," *Wireless Personal Communincations*, vol. 43, no. 2, pp. 467–479, 2007.

[43] P. Serrano, A. Banchs, and J. Kukielka, "Optimal Configuration of 802.11e EDCA Under Voice Traffic," in *Proc. GLOBECOM*, pp. 5107–5111, 2007.

[44] D. Gao, J. Cai, C. H. Foh, C.-T. Lau, and K. N. Ngan, "Improving wlan voip capacity through service differentiation," *IEEE Transactions on Vehicular Technology*, vol. 57, pp. 465–474, January 2008.

[45] Y. Ge, J. C. Hou, and S. Choi, "An analytic study of tuning systems parameters in IEEE 802.11e enhanced distributed channel access," *Computer Networks*, vol. 51, no. 8, pp. 1955–1980, 2007.

[46] M. Narbutt and M. Davis, "Experimental tuning of AIFSN and CWmin parameters to prioritize voice over data transmission in 802.11e WLAN networks," in *Proc. International conference on Wireless communications and mobile computing (IWCMC)*, (Honolulu, Hawaii, USA), pp. 140–145, 2007.

[47] C. Cano, B. Bellalta, and M. Oliver, "Adaptive admission control mechanism for ieee 802.11e wlans," *Proc. IEEE Personal, Indoor, and Mobile Radio Communications Conference (PIMRC)*, pp. 1–5, Sept. 2007.

[48] J. Lee, W. Liao, J.-M. Chen, and H.-H. Lee, "A practical QoS solution to voice over IP in IEEE 802.11 WLANs ," *IEEE Communications Magazine*, vol. 47, pp. 111–117, April 2009.

[49] Q. Ni, I. Aad, C. Barakat, and T. Turletti, "Modeling and Analysis of Slow CW Decrease for IEEE 802.1 1 WLAN," in *Proc. IEEE Personal, Indoor, and Mobile Radio Communications Conference (PIMRC)*, (Beijing, China), September 2003.

[50] Y. Grunenberger, M. Heusse, F. Rousseau, and A. Duda, "Experience with an implementation of the Idle Sense wireless access method," in *Proc. ACM CoNEXT*, (New York, New York), pp. 1–12, 2007.

[51] R. Bernasconi, S. Giordano, A. Puiatti, R. Bruno, and E. Gregori, "Design and Implementation of an Enhanced 802.11 MAC Architecture for Single-Hop Wireless Networks," *EURASIP Journal on Wireless Communications and Networking*, 2007.

[52] K. Aström and R. M. Murray, *Feedback Systems.* Princeton University Press, 2008.

[53] C. Hollot, V. Misra, D. Towsley, and W.-B. Gong, "A Control Theoretic Analysis of RED," in *Proc. IEEE INFOCOM*, (Anchorage, Alaska, USA), April 2001.

[54] D. Cavendish, M. Gerla, and S. Mascolo, "A control theoretical approach to congestion control in packet networks," *IEEE/ACM Transactions on Networking*, vol. 12, pp. 893–906, October 2004.

[55] G. F. Franklin, J. D. Powell, and M. L. Workman, *Digital Control of Dynamic Systems*. Prentice Hall, 3rd ed., 1997.

[56] A. Banchs, P. Serrano, and H. Oliver, "Proportional Fair Throughput Allocation in Multirate 802.11e EDCA Wireless LANs," *Wireless Networks*, vol. 13, October 2007.

[57] A. Aguiar and A. Wolisz, "Channel Prediction Heuristics for Adaptive Modulation in WLAN," in *Proc. IEEE Vehicular Technology Conference (VTC)*, (Dublin, Ireland), April 2007.

[58] C. Foh, Y. Zhang, Z. Ni, J. Cai, and K. Ngan, "Optimized Cross-Layer Design for Scalable Video Transmission Over the IEEE 802.11e Networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 12, 2007.

[59] A. Ksentini, M. Naimi, and A. Gueroui, "Toward an improvement of H.264 video transmission over IEEE 802.11e through a cross-layer architecture," *IEEE Communications Magazine*, vol. 44, no. 1, pp. 107–114, 2006.

[60] W. He, K. Nahrstedt, and X. Liu, "End-to-end delay control of multimedia applications over multihop wireless links," *ACM Transactions on Multimedia Computing, Communications and Applications (TOMCCAP)*, vol. 5, pp. 1–16, November 2008.

[61] A. Argyriou, "Error-Resilient Video Encoding and Transmission in Multirate Wireless LANs," *IEEE Transactions on Multimedia*, vol. 10, no. 5, pp. 691–700, 2008.

[62] P. Bucciol, G. Davini, E. Masala, E. Filippi, and J. De Martin, "Cross-layer perceptual ARQ for H.264 video streaming over 802.11 wireless networks," in *Proc. GLOBECOM*, vol. 5, 2004.

[63] A. Nafaa and A. Ksentini, "On Sustained QoS Guarantees in Operated IEEE 802.11 Wireless LANs," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 8, pp. 1020–1033, 2008.

[64] L. Grieco, G. Boggia, S. Mascolo, and P. Camarda, "A control theoretic approach for supporting quality of service in IEEE 802.11e WLANs with HCF," in *Proc. IEEE Conference on Decision and Control*, vol. 2, pp. 1586–1591, 2003.

[65] G. Boggia, P. Camarda, L. Grieco, and S. Mascolo, "Feedback-Based Control for Providing Real-Time Services With the 802.11e MAC," *IEEE/ACM Transactions on Networking*, vol. 15, no. 2, pp. 323–333, 2007.

[66] Y. Yang, B. R. Haverkort, and G. J. Heijenk, "A centralized feedback control model for resource management in wireless networks," in *8th Intl. Workshop on Performance Modeling of Computer and Communications Systems*, September 2007.

[67] Y. Xiao, F. H. Li, and B. Li, "Bandwidth Sharing Schemes for Multimedia Traffic in the IEEE 802.11e Contention-Based WLANs," *IEEE Transactions on Mobile Computing*, vol. 6, pp. 815–831, July 2007.

[68] Y. Xiao, H. Li, and S. Choi, "Protection and guarantee for voice and video traffic in IEEE 802.11e wireless LANs," in *Proc. IEEE INFOCOM*, vol. 3, pp. 2152–2162, 2004.

[69] Y. Zhang, C. Foh, and J. Cai, "An On-Off Queue Control Mechanism for Scalable Video Streaming over the IEEE 802.11e WLAN," in *Proc. International Conference on Computer Communications (ICC)*, pp. 4958–4962, 2008.

[70] C.-L. Chen, "IEEE 802.11e EDCA QoS Provisioning with Dynamic Fuzzy Control and Cross-Layer Interface," in *Proc. International Conference on Computer Communications and Networks (ICCCN)*, (Honolulu, HI, USA), August 2007.

[71] IEEE 802.11TGaa, *Wireless LAN Medium Access Control (MAC) And Physical Layer (PHY) Specifications: Amendment for Robust streaming of Audio Video Transport Streams, Draft 2.0.* 2010.

[72] D. Malone, K. Duffy, and D. Leith, "Modeling the 802.11 Distributed Coordination Function in Non-saturated Heterogeneous Conditions," *IEEE/ACM Transactions on Networking*, vol. 15, no. 1, 2007.

[73] D.P. Heyman and M.J. Sobel, *Stochastic Models in Operations Research, Vol. I: Stochastic Processes and Operating Characteristics.* Dover Publications, 2004.

[74] L. Kleinrock, *Queuing Systems, Vol. 1: Theory.* Wiley-Interscience, 1975.

[75] ITU-T, *Recommendation H.263: Video coding for low bit rate communication.* 2005.

[76] I. 14496-2, *Information technology – Coding of audio-visual objects – Part 2: Visual.* 2004.

[77] ITU-T, *Recommendation H.264: Advanced video coding for generic audiovisual services.* 2003.

[78] P. Lambert, W. De Neve, P. De Neve, I. Moerman, P. Demeester, and R. Van de Walle, "Rate-distortion performance of H.264/AVC compared to state-of-the-art video codecs," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 1, pp. 134–140, 2006.

[79] P. R. Cohen, *Empirical Methods for Artificial Intelligence.* Cambridge, Massachusetts: MIT Press, 1995.

[80] ITU-T, *Recommendation G.1010: End-user Multimedia QoS Categories.* 2001.

[81] ITU-T, *Recommendation G.1070: Opinion model for video-telephony applications.* 2007.

[82] I. 13818-2, *Information technology – Generic coding of moving pictures and associated audio information: Video.* 2000.

[83] T. Glad and L. Ljung, *Control theory: multivariable and nonlinear methods.* Taylor & Francis, 2000.

[84] R. Jain, Chiu, D.M., and W. Hawe, "A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Systems," *DEC Research Report TR-301*, 1984.

[85] D. Giustiniano, E. Goma, A. Lopez Toledo, I. Dangerfield, J. Morillo, and P. Rodriguez, "Fair wlan backhaul aggregation," in *Proc. ACM MobiCom*, (Chicago, Illinois, USA), pp. 269–280, 2010.

[86] D. Malone, P. Clifford, D. Reid, and D. Leith, "Experimental Implementation of Optimal WLAN Channel Selection without Communication," in *Proc. IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pp. 316–319, Apr 2007.

[87] G. Bianchi, F. Formisano, and D. Giustiniano, "802.11b/g Link Level Measurements for an Outdoor Wireless Campus Network," in *Proc. of International Symposium on World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pp. 525–530, 2006.

[88] P. Serrano, C. J. Bernardos, A. de la Oliva, A. Banchs, I. Soto, and M. Zink, "FloorNet: Deployment and Evaluation of a Multihop Wireless 802.11 Testbed," *EURASIP Journal on Wireless Communications and Networking*, 2010.

[89] D. J. Leith and P. Clifford, "Using the 802.11e EDCF to Achieve TCP Upload Fairness over WLAN Links," in *Proc. of International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WIOPT)*, (Washington, DC, USA), pp. 109–118, 2005.

[90] K. Aström and B. Wittenmark, *Computer-Controlled Systems: Theory and Design*. Prentice Hall, 3rd ed., 1996.

Appendix

Theorem 1. *The system of Sec. 3.1.2 is stable with the proposed K_p and K_i configuration.*

Proof. The closed-loop transfer function of our system is

$$S(z) = \frac{-C(z)H(z)}{1 - C(z)H(z)} = \frac{-z(z-1)HK_p - zHK_i}{z^2 + (-HK_p - 1)z + H(K_p - K_i)} \tag{6.1}$$

where

$$H = -\frac{\tau_{opt} p_{opt} (1 + p_{opt} \sum_{i=0}^{m-1} (2p_{opt})^i)}{2} \tag{6.2}$$

A sufficient condition for stability is that the poles of the above polynomial fall within the unit circle $|z| < 1$. This can be ensured by choosing coefficients $\{a_1, a_2\}$ of the characteristic polynomial that belong to the stability triangle [90]:

$$a_2 < 1 \tag{6.3}$$

$$a_1 < a_2 + 1 \tag{6.4}$$

$$a_1 > -1 - a_2 \tag{6.5}$$

In the transfer function of Eq. (6.1) the coefficients of the characteristic polynomial are

$$a_1 = -HK_p - 1 \tag{6.6}$$

$$a_2 = H(K_p - K_i) \tag{6.7}$$

From Eqs. (3.44) and (6.2) we have

$$HK_p = -0.4\frac{\tau_{opt}}{p_{opt}} \tag{6.8}$$

and from Eqs. (3.45) and (6.2) we have

$$HK_i = -\frac{0.4}{0.85 \cdot 2}\frac{\tau_{opt}}{p_{opt}} \tag{6.9}$$

from which

$$a_1 = 0.4\frac{\tau_{opt}}{p_{opt}} - 1 \tag{6.10}$$

$$a_2 = -0.16\frac{\tau_{opt}}{p_{opt}} \tag{6.11}$$

Given $\tau_{opt} \leq p_{opt}$, it can be easily seen that the above $\{a_1, a_2\}$ satisfy the conditions of Eqs. (6.3), (6.4) and (6.5). The proof follows. □

Theorem 2. *The system of Sec. 3.2.2 is stable with the proposed K_p and K_i configuration.*

Proof. The closed-loop transfer function of the system is

$$S(z) = \frac{-C(z)H(z)}{1 - z^{-1}C(z)H(z)} = \frac{z(z-1)\frac{\tau_{opt}p_{opt}}{2}K_p + z\frac{\tau_{opt}p_{opt}}{2}K_i}{z^2 + (\frac{\tau_{opt}p_{opt}}{2}K_p - 1)z + \frac{\tau_{opt}p_{opt}}{2}(K_i - K_p)} \tag{6.12}$$

A sufficient condition for stability is that the poles of the above expression fall within the unit circle $|z| < 1$. This can be ensured by choosing coefficients $\{a_1, a_2\}$ of the characteristic polynomial that belong to the stability triangle [90]:

$$\begin{cases} a_2 < 1 \\ a_1 < a_2 + 1 \\ a_1 > -1 - a_2 \end{cases} \tag{6.13}$$

where

$$a_1 = \frac{\tau_{opt}p_{opt}}{2}K_p - 1 \tag{6.14}$$

$$a_2 = \frac{\tau_{opt}p_{opt}}{2}(K_i - K_p) \tag{6.15}$$

With K_p and K_i given by Eq. (3.78) we obtain

$$a_1 = 0.4\frac{\tau_{opt}}{p_{col}} - 1 \tag{6.16}$$

$$a_2 = -0.16\frac{\tau_{opt}}{p_{col}} \tag{6.17}$$

Given $\tau_{opt} \leq p_{col}$, it can be easily seen that the above $\{a_1, a_2\}$ satisfy the conditions of the system (6.13). The proof follows. □

Theorem 3. *The system of equations defined by (4.6) has a unique solution that satisfies $e_{collision,i} = e_{fairness,i} = 0\ \forall i$, and all stations have the same transmission probability, $\tau_i = \tau_j\ \forall i, j$.*

Proof. From Eq. (4.6) we have

$$2p_{obs,i} - p_{own,i} - p_{opt} = 0 \tag{6.18}$$

which, following Sec. 4.2, can be rewritten as

$$2\sum_{k\neq i} \frac{\tau_k}{\sum_{l\neq i}\tau_l} p_{own,k} - p_{own,i} - p_{opt} = 0 \tag{6.19}$$

From Eq. (6.18), we have

$$2p_{obs,i} - p_{own,i} - p_{opt} - \frac{\sum_{k\neq i}\tau_l}{\sum_{k\neq j}\tau_l}\left(2p_{obs,j} - p_{own,j} - p_{opt}\right) = 0 \tag{6.20}$$

Applying Eq. (6.19) to the above, yields

$$\frac{2\tau_j}{\sum_{k\neq i}\tau_k} p_{own,j} + \frac{\sum_{k\neq j}\tau_k}{\sum_{k\neq i}\tau_k} p_{own,j} - \frac{2\tau_i}{\sum_{k\neq i}\tau_k} p_{own,i} - p_{own,i} - p_{opt} + \frac{\sum_{k\neq j}\tau_k}{\sum_{k\neq i}\tau_k} p_{opt} = 0 \tag{6.21}$$

from where

$$\left(\tau_j + \sum_k \tau_k\right) p_{own,j} - \left(\tau_i + \sum_k \tau_k\right) p_{own,i} + (\tau_j - \tau_i) p_{opt} = 0 \tag{6.22}$$

Substituting the expressions of $p_{own,j}$ and $p_{own,i}$ by Eq. (4.7) and operating on the above, yields

$$(\tau_j - \tau_i)\left(1 - \sum_k \tau_k \prod_{k\neq i,j} 1-\tau_k - \prod_{k\neq i,j} 1-\tau_k - p_{col}\right) = 0 \tag{6.23}$$

Note that Eq. (6.22) can be rewritten as

$$\left(\tau_j + \sum_k \tau_k\right)(p_{own,j} - p_{opt}) - \left(\tau_i + \sum_k \tau_k\right)(p_{own,i} - p_{opt}) = 0 \tag{6.24}$$

from where $p_j \leq p_{opt} \leq p_i$ or $p_i \leq p_{opt} \leq p_j$, which forces that either $p_{opt} \geq 1 - \prod_{k\neq i} 1-\tau_k$ or $p_{opt} \geq 1 - \prod_{k\neq j} 1-\tau_k$. This leads to

$$p_{opt} > 1 - \prod_{k\neq i,j} 1-\tau_k \tag{6.25}$$

Combining the above with Eq. (6.22), we have that the second term of Eq. (6.22) is

surely negative, which forces the first term to be 0. Thus,

$$\tau_i = \tau_j \tag{6.26}$$

and substituting the above into Eqs. (4.1) and (4.2), yields

$$e_{collision,i} = e_{fairness,i} = 0 \ \forall i \tag{6.27}$$

which proves the second part of the theorem.

To proof uniqueness of the solution, we proceed as follows. From the above we have

$$\tau_i = \tau \ \forall i \tag{6.28}$$

Substituting this into Eq. (6.18), we have

$$(1-\tau)^{n-1} = 1 - p_{opt} \tag{6.29}$$

Since the lhs of the above equation decreases from 1 to 0 with τ while the rhs is a constant between 0 and 1, we have that there exists a unique τ value that resolves the above equation. From Eq. (6.28) it further follows that the only solution to the system is $\tau_i = \tau \ \forall i$. The proof follows. □

Theorem 4. *The system of Sec. 4.1 is guaranteed to be stable as long as K_p and K_i meet the following condition:*

$$-(n-1)K_H(K_p - K_i) - 1 < (n-1)K_H(K_p - K_i) + 1 \tag{6.30}$$

Proof. According to (6.22) of [83], we need to check that the following transfer function is stable

$$(I - z^{-1}CH)^{-1}C \tag{6.31}$$

Computing the above matrix, yields

$$(I - z^{-1}CH)^{-1}C = \begin{pmatrix} a & b & b & \dots & b \\ b & a & b & \dots & b \\ b & b & a & \dots & b \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b & b & b & \dots & a \end{pmatrix} \tag{6.32}$$

where

$$a = \frac{C_{PI}(z)}{n}\left(\frac{1}{1-(n-1)z^{-1}K_H C_{PI}(z)} + \frac{n-1}{1-\left(2-\frac{n-3}{n-1}\right)z^{-1}K_H C_{PI}(z)}\right) \tag{6.33}$$

and

$$b = \frac{C_{PI}(z)}{n}\left(\frac{1}{1-(n-1)z^{-1}K_H C_{PI}(z)} - \frac{1}{1-\left(2-\frac{n-3}{n-1}\right)z^{-1}K_H C_{PI}(z)}\right) \tag{6.34}$$

Rearranging terms in a and b, we obtain

$$a = \frac{P_1(z)}{(z^2+a_1z+a_2)(z^2+a_1'z+a_2')} \tag{6.35}$$

and

$$b = \frac{P_2(z)}{(z^2+a_1z+a_2)(z^2+a_1'z+a_2')} \tag{6.36}$$

where $P_1(z)$ and $P_2(z)$ are polynomials and

$$a_1 = -(n-1)K_H K_p - 1 \tag{6.37}$$

$$a_2 = (n-1)K_H(K_p - K_i) \tag{6.38}$$

$$a_1' = -\left(2-\frac{n-3}{n-1}\right)K_H K_p - 1 \tag{6.39}$$

$$a_2' = \left(2-\frac{n-3}{n-1}\right)K_H(K_p - K_i) \tag{6.40}$$

According to Theorem 3.5 of [83], a sufficient condition for the stability of a transfer function is that the zeros of its pole polynomial (which is the least common denominator of all the minors of the transfer function matrix) fall within the unit circle. Applying this theorem to $(I - z^{-1}CH)^{-1}C$ yields that the roots of the polynomials $z^2 + a_1z + a_2$ and $z^2 + a_1'z + a_2'$ have to fall inside the unit circle. This can be ensured by choosing coefficients $\{a_1, a_2\}$ and $\{a_1', a_2'\}$ that belong to the stability triangle [90]:

$$a_2 < 1 \tag{6.41}$$

$$a_1 < a_2 + 1 \tag{6.42}$$

$$a_1 > -1 - a_2 \tag{6.43}$$

and

$$a_2' < 1 \tag{6.44}$$

$$a_1' < a_2' + 1 \tag{6.45}$$

$$a_1' > -1 - a_2' \tag{6.46}$$

Eqs. (6.41), (6.43), (6.44) and (6.46) are satisfied for any $\{K_p, K_i\}$ setting. If Eq. (6.42) is satisfied, then Eq. (6.45) is also satisfied. Therefore, it is enough to guarantee that Eq. (6.42) is met. The proof follows. □

Corollary 1. *The K_p and K_i configuration given by Eqs. (4.40) and (4.41) is stable.*

Proof. It is easy to see that Eqs. (4.40) and (4.41) meet the condition of Theorem 4. □