A dynamic CBWFQ scheme for service differentiation in WLANs

by

George Stamatakis

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Masters of Science

in

Computer Science

in the

GRADUATE DIVISION
of the
UNIVERSITY OF CRETE, HERAKLION

Summer 2005

The dissertation of George Stamatakis is approved:

Prof. Siris Vasilios, Research Advisor

Prof. Traganitis Apostolos, Reader

Prof. Maria Papadopouli, Reader

Prof. Dimitris Plexousakis, Graduate Studies Program Chair

Department of Computer Science
University of Crete, Heraklion

Summer 2005

A dynamic CBWFQ scheme for service differentiation in WLANs

Abstract

The last few years, wireless networks based on the IEEE 802.11 protocol, experience a tremendous growth. Their success can be attributed to utilization of unlicensed spectrum, interoperability granted by adherence to a standard, low cost, ease of installation and above all provision of untethered access to Internet resources. Nowdays, the prominent deployment scenario for 802.11 wireless networks is the so-called "hot-spot" scenario which provides access to nomadic users in both public (hotels, airports, etc.) and private (offices, conferences, etc.) environments.

Within this scenario, mechanisms that support service differentiation, guarantee fairness and improve performance, are a prerequisite for supporting the modern demanding applications. Provision of service differentiation in wireless networks is hindered by several factors such as, location dependent channel errors, uplink-downlink unfairness and performance reduction due to the multirate physical layer. Nevertheless, a plurality of such mechanisms has been proposed at various levels in the protocol hierarchy. Few of the proposed solutions, however, provide a mechanism that can be implemented in a feasible and effective way without requiring alternation of the 802.11 MAC layer.

In this thesis we propose a dynamic Class Based Weighted Fair Queuing (CBWFQ) scheme, properly adapted to the wireless network particularities, that differentiates service on a per node basis, improves fairness and enhances utilization of the network resources. The proposed mechanism is based on a definition of fairness inherently different from the fairness perspective of the Distributed Coordination Function (DCF) of 802.11. Fairness is expressed in terms of channel occupation time shares; a way that suits best the multirate physical layer of 802.11. The mechanism is comprised of two algorithms that utilize cross-layer information to periodically adjust the weights of a CBWFQ scheme in a way that collectively deals with significant issues of WLANs, such as location dependent channel errors, incontrollable uplink traffic and unfairness due to multiple transmission rates. The first algorithm monitors the achieved rate of each node and adjusts weights in order to compensate nodes that suffered losses due to channel errors by redistributing the available bandwidth properly. The second algorithm incorporates transmission rate information in weight calculations so that all nodes occupy the channel approximately the same time.

The dynamic CBWFQ scheme is a network layer mechanism whose implementation does not require changes at the MAC layer of 802.11. Its deployment can be done either in a control station, that interconnects one or more access points with the wired network, or as a service running on an

access router. The proposed scheme was implemented on an experimental testbed, that employs the latter deployment approach, and the conducted experiments confirmed its resultfulness in real life scenarios.

Supervisor of Dissertation: Siris Vasilios

Περίληψη

Τα τελευταία χρόνια, η χρήση ασύρματων δικτύων που βασίζονται στο πρωτόκολλο 802.11 εμφανίζει σημαντική εξάπλωση. Η επιτυχία τους αυτή μπορεί να αποδοθεί στην εκμετάλλευση ελεύθερα διαθέσιμων περιοχών του φάσματος, στην διαλειτουργικότητα που επιτυγχάνεται με την τήρηση των κανόνων του προτύπου, στο χαμηλό τους κόστος και πάνω απ' όλα στη δυνατότητα που παρέχουν για πρόσβαση στους πόρους του διαδικτύου χωρίς τους περιορισμούς των ενσύρματων δικτύων. Το επικρατέστερο σενάριο εφαρμογής τους είναι το αποκαλούμενο hot-spot σενάριο, το οποίο παρέχει πρόσβαση στο ενσύρματο δίκτυο σε κινούμενες ομάδες χρηστών τόσο σε δημόσιους όσο και σε ιδιωτικούς χώρους.

Στα πλαίσια αυτού του σεναρίου, μηχανισμοί που παρέχουν διαφοροποίηση υπηρεσίας, εγκυώνται τη δίκαιη διαμοίραση των πόρων του δικτύου και βελτιώνουν την συνολική του απόδοση, αποτελούν προϋπόθεση για την υποστήριξη των σύγχρονων και δικτυακά απαιτητικών εφαρμογών. Ωστόσο, η επίτευξη των παραπάνω στόχων στα ασύρματα δίκτυα αντιμετωπίζει σημαντικά προβλήματα. Οι εκρήξεις λαθών, τοπικού χαρακτήρα, κατά τη μετάδοση πληροφορίας στο ασύρματο μέσο, η αδικία στη διαμοίραση πόρων μεταξύ της κίνησης που αποστέλεται στο σταθμό βάσης (uplink) και της κίνησης που προέρχεται από αυτόν (downlink), καθώς και η μείωση της συνολικής απόδοσης λόγω των πολλαπλών ρυθμών μετάδοσης που υποστηρίζει το φυσικό επίπεδο του πρωτοκόλλου 802.11 αποτελούν ενδεικτικά προβλήματα που καθιστούν την απευθείας χρήση των γνωστών αλγορίθμων διαφοροποίησης υπηρεσίας που χρησιμοποιούνται στα ενσύρματα δίκτυα, αναποτελεσματική. Λόγοι όπως οι παραπάνω, έχουν οδηγήσει στην ανάπτυξη μιας πληθώρας νέων αλγορίθμων, κατάλληλα προσαρμοσμένων στις ιδιαιτερότητες των ασύρματων δικτύων. Λίγοι όμως από τους αλγόριθμους αυτούς μπορούν να υλοποιηθούν με αποτελεσματικό και αποδοτικό τρόπο χωρίς να απαιτούνται αλλαγές στο πρωτόκολλο 802.11. Σε κάθε περίπτωση, αλγόριθμοι που απαιτούν αλλαγές στο επίπεδο ζεύξης δεδομένων ή το φυσικό επίπεδο του 802.11 δεν μπορούν να εφαρμοστούν στο μεγάλο αριθμό των ήδη εγκατεστημένων δικτύων. Επιπλέον, οι λύσεις που έχουν ήδη προταθεί αντιμετωπίζουν συνήθως τα προβλήματα της διαφοροποίησης υπηρεσίας, της δικαιοσύνης και της απόδοσης διακριτά, καταλήγοντας έτσι σε μη ολοκληρωμένες λύσεις.

Στην εργασία αυτή, προτείνουμε ένα δυναμικό CBWFQ σχήμα, κατάλληλα προσαρμοσμένο στις ανάγκες των ασύρματων δικτύων, το οποίο διαφοροποιεί την παρεχόμενη υπηρεσία ως προς το ρυθμό λήψης και μετάδοσης δεδομένων ανά κόμβο, βελτιώνει την δίκαιη διαμοίραση των πόρων του δικτύου και την συνολική του απόδοση. Η βασική συνεισφορά της προτεινόμενης μεθόδου είναι η επίτευξη των παραπάνω στόχων χρησιμοποιώντας ως μόνο μέσο τη δυναμική μεταβολή των βαρών ενός μηχανισμού CBWFQ. Ο

προτεινόμενος μηχανισμός βασίζεται σε ένα ορισμό της δικαιοσύνης που είναι σημαντικά διαφορετικός από αυτόν που χρησιμοποιείται από την Distributed Coordination Function (DCF) του 802.11. Η δικαιοσύνη, σύμφωνα με το ορισμό που χρησιμοποιήσαμε, εκφράζεται σε όρους χρονικής κατάληψης του διαύλου και είναι προσαρμοσμένή καλύτερα στους πολλαπλούς ρυθμούς μετάδοσης του φυσικού επιπέδου του 802.11. Εξασφαλίζοντας ότι όλοι οι κόμβοι καταλαμβάνουν το ασύρματο κανάλι για ίσα χρονικά διαστήματα, οι χρήστες που έχουν κανάλι υψηλής ποιότητας μπορούν να πετύχουν υψηλούς ρυθμούς μετάδοσης και λήψης, ανεξάρτητα από την ποιότητα του διαύλου των υπολοίπων κόμβων. Ομοίως, οι κόμβοι που έχουν κακή ποιότητα διαύλου λαμβάνουν το μερίδιο χρόνου που δικαιούνται. Με αυτό τον τρόπο, η δικαιοσύνη με βάση το χρόνο παρέχει τη επιθυμητή ιδιότητα της απομόνωσης όσο αφορά στο ρυθμό μετάδοσης και αποτρέπει ανωμαλίες στην απόδοση.

Η εφαρμογή αυτής της προσέγγισης για τη δίκαιη διαμοίραση του χρόνου κατάληψης του διαύλου θα απαιτούσε σημαντικές μεταβολές στο πρωτόκολλο 802.11. Αυτό μας ώθησε στο σχεδιασμό του δυναμικού μηχανισμού CBWFQ, που είναι κατάλληλα διαμορφωμένος ώστε να προσεγγίζει το παραπάνω σχήμα δικαιοσύνης, ενώ μπορεί να υλοποιηθεί στο επίπεδο του δικτύου και συνεπώς δεν απαιτεί μεταβολή στο πρωτόκολλο 802.11. Ο δυναμικός μηχανισμός αποτελείται από δύο αλγόριθμους που χρησιμοποιούν πληροφορία από διαφορετικά επίπεδα του προτύπου OSI προκειμένου να μεταβάλουν τα βάρη του CBWFQ σχήματος με τρόπο που να αντιμετωπίζει συνδυασμένα τα σημαντικά προβλήματα των ασύρματων δικτύων, όπως τα τοπικά σφάλματα μετάδοσης, την ανεξέλεγκτη μετάδοση πληροφορίας προς το σταθμό βάσης και την μείωση της απόδοσης λόγω των πολλαπλών ρυθμών μετάδοσης.

Ο πρώτος αλγόριθμος ελέγχει το ρυθμό μετάδοσης που πέτυχε κάθε κόμβος και ρυθμίζει τα βάρη του CBWFQ μηχανισμού κατάλληλα, παρέχοντας περισσότερους πόρους στους κόμβους που εμφάνισαν απώλειες λόγω λαθών μετάδοσης ή λόγω της απουσίας κίνησης που να προέρχεται από αυτούς. Οι επιπλέον πόροι εξασφαλίζονται με αναδιανομή της διαθέσιμης χωρητικότητας του δικτύου. Ο δεύτερος αλγόριθμος εισάγει την πληροφορία του ρυθμού μετάδοσης που χρησιμοποιεί ο κάθε κόμβος στον υπολογισμό των τελικών βαρών του μηχανισμού CBWFQ έτσι ώστε τελικά όλοι οι κόμβοι να καταλαμβάνουν το δίαυλο για το ίδιο χρονικό διάστημα περίπου. Επιπλέον, παρέχεται πλήρης διαχωρισμός πακέτων δεδομένων και πακέτων επιβεβαιώσεων που προορίζονται σε κάθε ασύρματο κόμβο και στη συνέχεια πραγματοποιείται μετάδοση τους από διαφορετικές ουρές. Η μέθοδος επιτρέπει τον έλεγχο της κίνησης που αποστέλλεται προς το σταθμό βάσης, τουλάχιστον όσο αφορά στο πρωτόκολλο TCP, και την επίλυση του σχετικού προβλήματος αδικίας.

Η εγκατάσταση του δυναμικού μηχανισμού CBWFQ μπορεί να γίνει είτε σε ένα σταθμό ελέγ-

χου, που διασυνδέει έναν ή περισσότερους σταθμούς βάσης με το ενσύρματο δίκτυο, ή ως υπηρεσία που εκτελείται σε ένα δρομολογητή πρόσβασης (access router). Το προτεινόμενο σχήμα υλοποιήθηκε σε ένα πειραματικό δίκτυο, ακολουθώντας την τελευταία προσέγγιση (με δρομολογητή πρόσβασης), και τα πειραματικά αποτελέσματα επιβεβαίωσαν την αποτελεσματικότητα του.

Επόπτης Μεταπτυχιακής Εργασίας: Βασίλειος Σύρης

Acknowledgments

The present study is the result of hard work. However, the endless ours spent on designing and experimenting would have been unfruitful without the help of so many people that a small acknowledgment section would be inadequate to thank them all. I do keep in mind, however, their willingness to spent valuable time to help me out and I hope I will be given the chance to help them back.

Mr. Siris, my supervisor, has contributed to this work in any possible way. His novel ideas and questions helped me avoid dead ends, while his experience and proper guidance ensured the scientific quality of this work. For all these I express my gratitude.

The cornerstone for the successful completion of all serious efforts I have undertaken, including this one, is my family. My parents, my brother and my sister helped me accomplish this difficult task in their own unique way. The least I could do is dedicate this work to them.

John Yannakopoulos has been more than just a friend and colleague. John introduced me to the fascinating world of open source software and guided me through my first steps with his profound advices, he inspired me with his total devotion to whattever he does; his thirst for life and new experiences. Being a friend with John is an invaluable experience. Thank you John!

It is difficult to find words to thank Despoina Triantafyllidou. Her friendship and encourangment as well as our academic discussions were invaluable during my studies.

The pleasant and productive, working environment at ICS-FORTH can be attributed to my colleagues Fotiadis Giorgos, Stefanos Papadakis, Aggelakis Vaggelis, Tzagarakis Giorgos and Xaris Melissaris. Their friendship goes hand in hand with their scientific advises since they are all extremely capable network and telecommunication engineers.

It is true that the completion of this study was an exhaustive process as it is true that I had all the energy in the world, thanks to my little Areti. Her endless love, support and understanding made this difficult task possible. Thank you Areti for this, and all other beautiful things you offer me everyday.

To my family

Στην οικογένεια μου

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Wireless local area networks (WLANs) based on the IEEE 802.11 technology are experiencing a widespread deployment in both indoor and outdoor environments. The prominent reason behind the explosive growth of this technology is the need for tetherless connectivity to Internet resources. Emerging applications for mobile information access, real-time multimedia communications, networked games and cooperative work reflect the necessity to communicate and exchange data while on the move.

However, shaking off the confinements imposed by wired networks is not the only advantage of WLANs. Ease and speed of deployment, flexibility and cost reduction have irrefutably contributed to their success. Cabling industrial facilities and old buildings can be a challenge and the same holds for the quick construction of amorphous wired networks, not to mention moving between cubicles and offices or even bridging buildings. Wireless LANs are used to facilitate matters in all the above cases either as a stand alone solution or in conjunction with a limited wired infrastructure.

What wireless and wired networks have in common is the provision of the traditional best-effort delivery of datagram traffic from senders to receivers. However, modern applications have specific quality of service (QoS) requirements, mainly due to their highly interactive nature. As the Internet evolves, more and more users exploit the functionality provided by these applications to fulfill tasks or entertain themselves. Still, there are others who use non communication intensive applications and are unwilling to pay Internet service providers for more than that. Based on these observations, companies providing Internet services plan to increase their revenue by offering contracts that suit their customers' needs and allow for a larger number of multiplexed traffic sources.

Provision of service differentiation in wired networks has been hindered by several factors including, but not limited to, the inherent best-effort service model of the IP protocol, the uncertainty

in characterizing the traffic generation process, the unavailability of QoS routing algorithms and protocols, and the poor scalability of several attempts at defining an articulate QoS paradigm. On top of these observations wireless networks in general, and 802.11 based ones in particular, offer a whole set of new challenges. The most significant of them are poor channel quality, that heavily depends on the relative position of wireless stations, interference from hidden terminals, performance reduction due to the multirate physical layer and uplink-downlink unfairness. All these features contribute in a lack of determinism that has several drawbacks. Guarantees, even of statistical nature, cannot be extended to users of wireless networks, because even the most privileged of them can be unpredictably delayed of discarded by temporary channel failures.

1.1 Stating the problem and motivating our solution

Additional mechanisms, properly adapted to the wireless networks particularities, are necessary if service differentiation is to be provided in WLANs. A plurality of such mechanisms has been proposed at various levels in the protocol hierarchy. At the MAC layer, 802.11e, the new IEEE draft, defines the MAC procedures to support LAN applications with QoS requirements. A large number of scheduling algorithms has also been proposed for wireless networks that consider channel state, losses etc. At the network level, protocols such as IntServ and DiffServ, properly modified to fit the wireless channel needs, provide the means to support service differentiation in WLANs.

However, few of the proposed solutions provide a service differentiation mechanism that can be implemented solely at an *Access Router* and address, in a feasible and effective way, fairness and performance problems of WLANs. Mechanisms that require changes at the MAC layer of 802.11 cannot be applied to the large number of networks already installed. Furthermore, fairness and performance issues are often dealt with separately resulting in partial solutions. Such issues though (decreased throughput due to a node's small transmission rate, location dependent bursty channel errors, Uplink-Downlink unfairness, etc.) are major afflictions for WLANs and should be addressed in conjunction.

In this study, we propose a dynamic Class Based Weighted Fair Queuing (CBWFQ) mechanism that improves fairness and aggregate throughput, while supporting weight based service differentiation in wireless networks. Initially, we present a fairness reference model that provides the formal, idealized objective for sharing the wireless network resource. *Time* is considered the resource that should be shared fairly among the wireless nodes, as this approach suits best the multirate physical layer of

802.11 based networks (see [1] and [2]). By assuring that all nodes have an equal time share of channel access, nodes with high quality channels can obtain throughput gains independent of the channel qualities of others. Likewise, nodes with poor channels are guaranteed their fair time share. In this way, time-share fairness provides the desirable "performance isolation" property and avoids the "performance anomaly" of throughput fairness.

This time-based fairness scheme, however, is inherently different from the fairness perspective of 802.11 MAC layer and its implementation would require a complete reformation of the latter. This motivated the design of the dynamic CBWFQ mechanism that is properly configured to approximate the proposed fairness scheme and can be implemented solely at the network layer, requiring thus, no changes at the MAC layer of 802.11. The mechanism is comprised of two algorithms that periodically adjust the weights of a CBWFQ scheme in a way that *collectively* deals with significant issues of WLANs, such as location dependent bursty channel errors, incontrollable uplink TCP traffic and unfairness due to the multirate physical layer.

The first of the two algorithms monitors the achieved rate of each node and adjusts weights in order to compensate nodes that suffered losses due to channel errors by redistributing the available bandwidth properly. The second algorithm incorporates transmission rate information in weight calculations so that all nodes occupy the channel approximately the same time. Moreover, full separation of data and acknowledgment packets destined to the wireless hosts and scheduling them through different queues[1] provides the means to control uplink traffic and resolve the uplink-downlink unfairness problem.

The novel aspect about the dynamic CBWFQ mechanism is the incorporation of weight manipulations as the only mean to address a wide range of problems in wireless networks. This key feature retains the feasibility of our mechanism, whereas, other CBQ approaches, like the one presented in [3], not only deal with a shorter range of problems since they do not consider the multirate physical layer of 802.11 and the necessity to exert control over uplinik traffic, but also they cannot be applied in existing 802.11 networks due to their incompatibility with the 802.11 MAC layer. Furthermore, previous studies concerning fairness and performance in wireless networks [4, 5, 6, 7, 8, 9] focus on the design of MAC layer scheduling algorithms rather than the 802.11 protocol, thus they lack the applicability of our approach, for example most of theses studies assume perfect channel knowledge prior to any packet transmission and that scheduling is performed at the base station (access point) for

[1] Two queues are provided for each node, one for each type of packet.

both uplink and downlink traffic. The base station of an 802.11 wireless network with infrastructure functions as a bridge that interconnects the wireless with the wired part of the network. Furhtermore, wireless nodes communicate through the bridge and not directly with each other using the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) MAC layer of 802.11 to gain access to the wireless medium. Although, the 802.11 protocol defines such a contention free MAC function, called Point Coordination Function (PCF), the latter is not yet implemented in commercial 802.11 products.

A prototype of the proposed scheme has been implemented on an access router that provides the functionality of an access point enhanced with routing and service differentiation capabilities, and a series of experiments has been conducted to test the effectiveness of our mechanism in realistic scenarios. The experimental results validated its resultfulness.

1.2 Thesis Outline

The rest of this thesis is organized as follows. In Chapter 2 we present the problems and theory that form the basis of out dynamic CBWFQ scheme. In Chapter 3, we describe the service differentiation mechanism architecture. Chapter 4 presents the wireless testbed setup. In Chapter 5, we describe the evaluation scenarios and analyze the experimental results. Chapter 6 reviews related work and finally, in Chapter 7, we present our conclusions and identify related ongoing and future work.

The results of this study have been accepted for publication in the international conference PIMRC'05.

Chapter 2

Challenges

In this section, we describe the most significant fairness problems of 802.11 wireless networks and illustrate them with experimental results derived from our testbed. Next, we redefine fairness in terms of temporal shares and present the basic theory for CBWFQ as well as the problems resulting from its application in wireless networks. Finally, we give and outline of the parts that constitute our dynamic CBWFQ scheme and correlate them with the problems they address.

2.1 Fairness perspective of 802.11

In the IEEE 802.11 protocol, the fundamental mechanism to access the medium is called distributed coordination function (DCF). It is a random access scheme, based on the carrier sense multiple access with collision avoidance protocol (CSMA/CA) [1]. DCF guarantees *an equal long term channel access probability* to all hosts. That is, with DCF all mobile nodes (including the access point) will have an equal number of *transmission attempts* in the long run. This makes DCF the root of significant unfairness issues in 802.11 wireless networks with infrastructure. The error prone link, with location and time varying characteristics, makes it impossible for all nodes to utilize their share of transmission attempts successfully resulting thus in an unfair bandwidth distribution. Furthermore, although the access point serves all downlink traffic destined to the wireless hosts, there is no anticipation in DCF to increase its transmission priority causing thus a significant uplink downlink unfairness phainomenon. DCF does not provide per node fairness neither in terms of *throughput* nor of channel occupation *time*. The multirate physical layer of 802.11 and the equal number of transmission attempts guaran-

[1] We will not present DCF in this section. For a complete and detailed presentation please refer to the 802.11 standard [10], [11].

teed by DCF result in some nodes occupying the channel for longer periods than others causing thus the performance anomaly problem of 802.11. These issues will be further discussed in the following sections.

2.2 Unfairness due to link unreliability

Location dependent, bursty channel errors characterize the wireless channel and cause an unfair bandwidth distribution among the nodes of a wireless network. There are various factors that give rise to such errors on the wireless medium:

Noise and interference. Noise refers to unwanted signals that tend to disturb the transmission and processing of message signals. A quantitative way to account for the effect of noise is to introduce the ratio of the average signal power to the average noise power both measured at the same point. For all communication systems, channel noise is intimately tied to bandwidth. All objects which have heat emit RF energy in the form of Gaussian noise. Especially for those communication systems that make use of the unlicensed Industry, Science and Medicine (ISM) band, like the 802.11 physical layer, additional sources of interference include: other 802.11 networks in the area, bluetooth, cordless phones, X10 protocol for home automation and microwave ovens. In Table 2.1 we present the bandwidth and power regulations made by the corresponding authorities in USA, Europe and Japan for the ISM band.

Range and path loss. Another key consideration is the issue of range. As radio waves propagate in free space power falls off as the square of range. For a doubling of range power reaching a receiver antenna is reduced by a factor of four. This effect is due to the spreading of the radio waves as they propagate. In case of indoor transmission propagation losses can be significantly higher. This occurs because of a combination of attenuation by walls and ceilings, and blockage due to equipment, furniture, and people. In Fig. 2.1 we present the estimated indoor propagation losses at 2.4 GHz. The figure was taken from [12].

Multipath fading. Multipath occurs when waves emitted by the transmitter travel along different paths and interfere destructively at the receiver region. Waves travelling along different paths may be completely out of phase when they reach the receiver antenna, thereby cancelling each other.

Intersymbol interference. Multipath fading is a special case of intersymbol interference. Waves that take different paths will travel different distances and will be delayed with respect to each other. The time between the arrival of the first wavefront and the last multipath echo is called the delay

spread. If delay spread is large enough then the first wavefront of an information symbol could suffer interference from delayed multipath echos of previous symbols resulting in a garbled waveform. 802.11 networks can handle delay spreads up to 500 ns but a 65 ns delay spread is required for full-speed 11Mbps performance.

Band	FCC Regs (US)	ETSI (EUROPE)	MPT (JAPAN)
902-928MHz	<1000 mW	N/A	N/A
2400-2483.4MHz	<1000 mW	<100mW	N/A
2471-2497MHz	N/A	N/A	<10mvW/MHz
5725-5875MHz	<1000 mW	<100mW	N/A

Table 2.1: World wide unlicensed frequency allocation RF power limits.

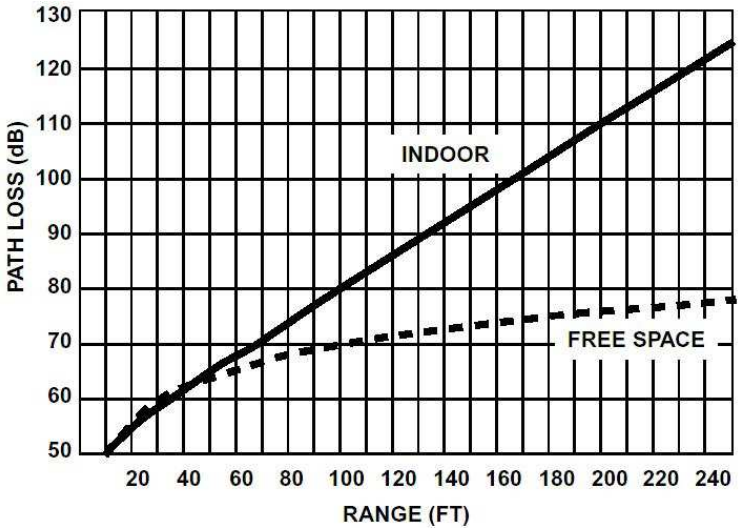

Figure 2.1: Estimated indoor propagation losses at 2.4 GHz.

All factors described above depend on both time and spatial location of the receiver, resulting thus in a significantly error prone link with *location* and *time* varying characteristics. This implies that at any time, it may happen that only a subset of the wireless nodes will be able to receive and transmit successfully; the other nodes will fail in their transmission or reception attempts. Since DCF grants all nodes the same number of transmission attempts in the long run, bandwidth will be shared unfairly among nodes. Many researchers have noted the importance of location dependent bursty channel errors in sharing the wireless bandwidth fairly [6, 7, 9, 4, 8, 13, 3, 14].

In order to illustrate the unfair behavior of DCF we present in Fig. 2.2 the throughput achieved by three wireless hosts in our 802.11b testbed. Our testbed consists of a software access router and three wireless hosts. The access router is based on a Linux box equiped with a wireless card that is driven by the HostAP device driver. Routing of packets is handled by the Linux kernel while the HostAP driver provides the access point functionality. The three wireless hosts are laptops running the Windows XP operating system. The wireless hosts were placed in an office environment and each one of them was receiving a TCP flow. Details concerning the setup of our testbed will be given in Chapter 4. Since the only sender in this experiment is the access router losses are due to reception failures and not collisions. It can be seen from Fig. 2.2 that due to these channel errors node 2 received less service than the rest two nodes.

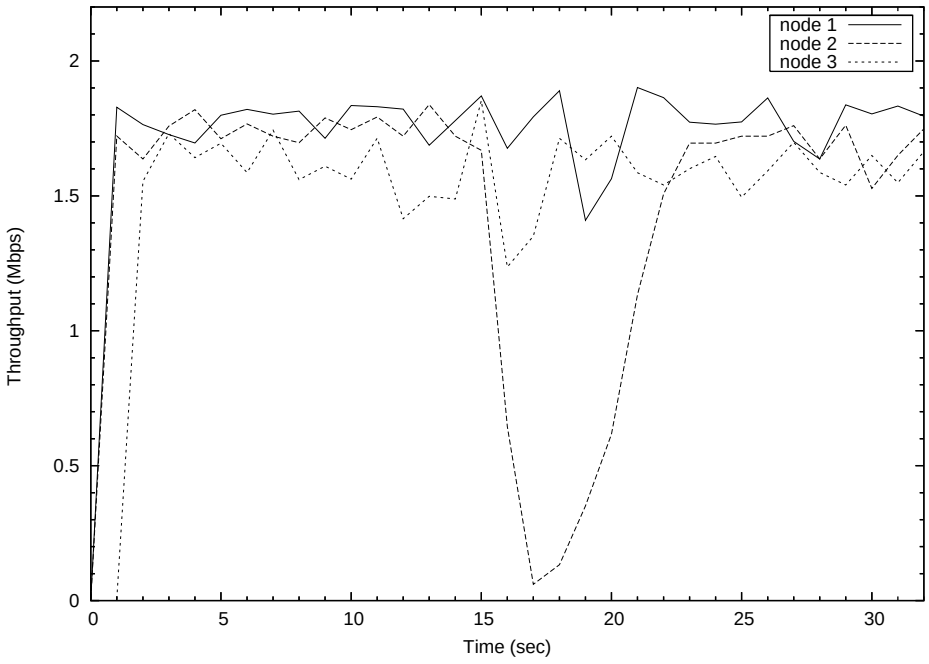

Figure 2.2: Unfairness due to link unreliability.

2.3 Uplink-downlink unfairness

Significant unfairness has been observed between uplink and downlink flows when the DCF is employed in 802.11 WLANs with Access Point (AP). A root cause of this phenomenon is that in a WLAN with N stations, there are N uplink CSMA/CA instances contending with only one downlink

CSMA/CA instance, the one at the AP. There is no anticipation in the DCF to increase the priority of AP transmissions, although the AP serves all downlink flows, and downlink flows commonly constitute the majority of network traffic.

Unfairness between uplink and downlink TCP flows in 802.11 WLANs is also attributed to buffer availability at the AP, as identified by the authors in [15]. The reason is that data and acknowledgment packets clutter at the base station's buffer. If the buffer's size is not large enough to accommodate all packets some of them will be dropped. As a result downstream flows will suffer data packet drops while upstream flows will experience acknowledgment packet drops. The exact buffer size at the access point is not the same value for all commercial access points. Our testbed uses a 100 packet drop tail queue. This is also the default Linux configuration for the wireless interface.

A key observation, with great importance in comprehending the consequences of this phenomenon, is that a loss of an acknowledgment packet has no real influence on TCP's congestion window size. This holds under the assumption that the congestion window is already large enough and is due to the *cumulative acknowledgment* feature of TCP, whereby the next ACK packet will have the appropriate sequence number and make up for losses of previous ACK packets[2]. The downstream TCP window size, on the other hand, changes considerably since TCP reacts to loss of each data packet by halving its congestion window or, in case of a timeout, by setting it to 1.

In Fig. 2.3 we present the uplink-downlink unfairness phenomenon. This experiment was conducted in our experimental testbed and we used three nodes; each one of them sent three TCP data flows and received another three TCP data flows. Fig. 2.3 presents the aggregate uplink and downlink throughput (3 TCP flows each) for one of these nodes.

2.4 Performance anomaly of 802.11

In [16] the authors observed that when some mobile hosts use a lower bit rate than the others, the performance of all hosts is considerably degraded. They named this phenomenon *performance anomaly of 802.11*. Such a situation is a common case in wireless local area networks in which a host far away from the AP is subject to high signal fading and interference. To cope with this problem, the host changes its modulation type, which degrades its bit rate to some lower value. Typically,

[2]The TCP reaction to acknowledgment packets losses can be either getting into a timeout or increasing the congestion window as if they were never lost. Practically for a congestion window of w packets, almost all acknowledgments must be lost in order to have a timeout.

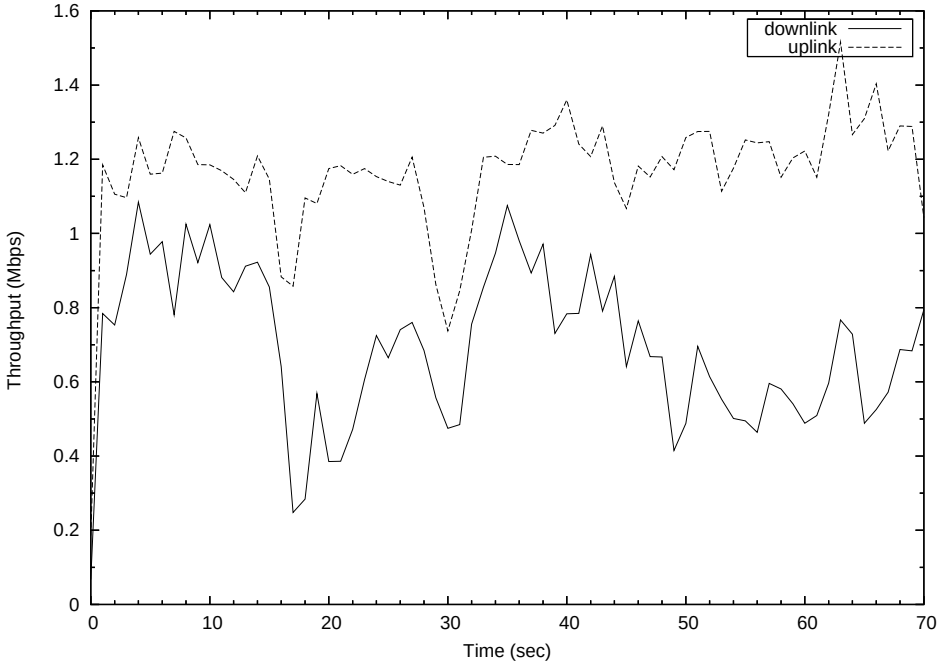

Figure 2.3: Uplink-downlink unfairness.

802.11b products degrade the bit rate from 11 Mbps to 5.5, 2, or 1 Mbps when repeated unsuccessful frame transmissions are detected. A host that uses a degraded bit rate will reduce the throughput of other to a value below it. DCF is the root of this anomaly, since it guarantees an equal number of transmission attempts to all nodes ignoring the prolonged channel occupation time required by nodes using a degraded bit rate. Thus we have an unfair distribution of channel occupation time among the wireless nodes that results in a significant performance degradation. In Table 2.2 we present the time it takes for a node to upload 10 Mbytes in segments of 1024 bytes using the TCP protocol and different physical layer transmission rates.

Transmission Rate (Mbps)	Transmission Duration (sec)
11	17.12
5.5	25.64
2	55.86
1	106.04

Table 2.2: Upload duration of 10 Mbytes in segments of 1024 bytes (TCP).

In order to further illustrate the performance anomaly phenomenon, we present in Fig. 2.4 the

throughputs achieved by three wireless nodes in our experimental testbed. Each node was generating a persistent TCP flow towards a host in the wired network. Node 3 had the capability to change its transmission rate through the wireless card configuration interface. During the experiment we alternated its transmission rate between 11 Mbps and all lower transmission rates, starting from 1 Mbps[3]. It can be seen that, whenever node 3 reduces its transmission rate, the throughput of the remaining nodes decreases to a contiguous value.

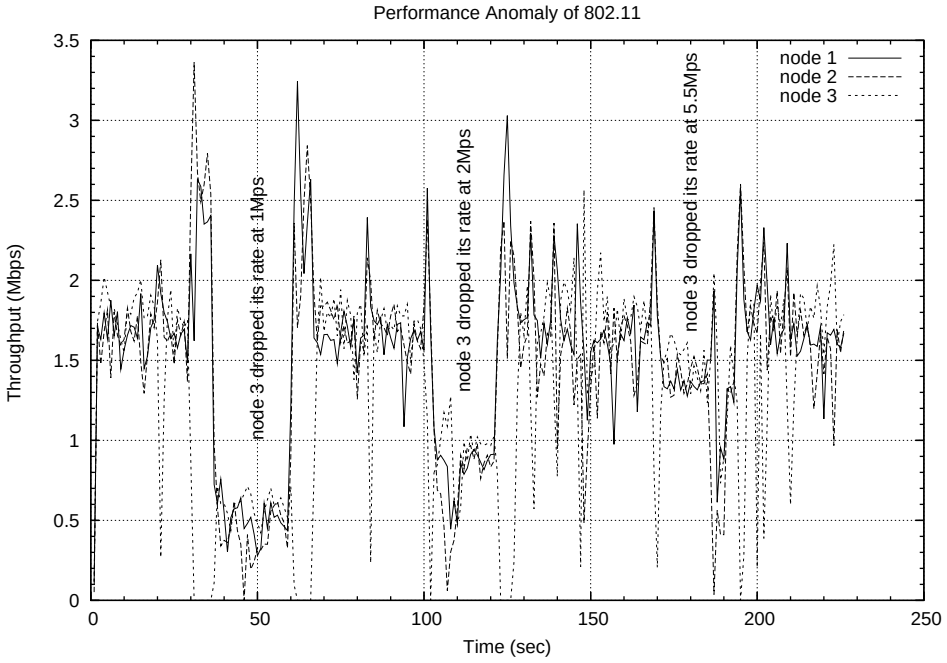

Figure 2.4: Performance anomaly of 802.11.

2.5 Redefining fairness

The fairness definition that suits best wireless networks, in general and 802.11 in particular, can be formulated as a scheme that pursues a *per-node weighted fairness in terms of successful transmission and reception time shares*. The above definition should hold for all nodes that have packets for transmission (backlogged). In the next paragraphs, we will clarify the concepts of this definition.

We propose a per-node fairness scheme instead of a per-flow one, since the implementation of the latter would be an extremely complicated and inevitably inefficient approach. Even for small

[3]Each transmission rate alternation was followed by short loss of contact with the access point which explains the peaks before and after transmission rate changes.

scale wireless networks identifying the ever changing number of active and inactive (idle) flows and providing resources, e.g., classification filters, queues, link state information, for each one of them would be an overwhelming task for an access router to perform.

Defining fairness in terms of time shares suits best the multirate physical layer of 802.11 protocol (see [1] and [2]). The distinction between time and throughput-based fairness is critical in multirate networks since nodes could achieve dramatically different throughout in approximately identical time shares. This is due to their utilization of different transmission rates (modulation schemes) which are determined by the link conditions. Adopting a fairness scheme in terms of time shares results in each node utilizing the amount of time granted to it the best way it can. Instead, a fairness scheme in terms of throughput results in severe performance degradation due to the performance anomaly problem described in the previous section.

Distributing *reception time* fairly is crucial in 802.11 wireless networks with infrastructure since the reception time of all nodes corresponds to the *transmission time* of the access point. To illustrate this correspondence consider the simple case of two competing nodes in a wireless network with infrastructure. According to our fairness definition each node should be granted 25% of the total time for transmission and 25% *for reception* in the long run. This means that the access point should be granted 50% of the total time for transmissions. Half of this interval will be spent for transmissions to the first node and the remaining half to the second one.

2.6 CBWFQ theory and issues

Implementation of the fairness scheme described above at the *network layer* is an extremely difficult task, since, among other things, it requires the knowledge of the transmission rate and the size of *each* packet. Such knowledge is available at the MAC layer and could be easily utilized by a MAC layer scheduler that performs scheduling of both uplink and downlink traffic in a wireless network. Obviously, such an approach would require extensive changes at the MAC layer of 802.11, something we would like to avoid. In order to resolve the problem at hand we adopted a *dynamic Class Based Weighted Fair Queuing (CBWFQ) scheme, that incorporates transmission rate information in order to approximate the behavior of the fairness scheme defined above.*

In wireline networks, CBWFQ has been a popular paradigm for achieving fairness and bounded delays in channel access [17]. In this link sharing scheme, each *class* corresponds to an aggregation of traffic uniquely identified by information included in the packet's header and the packet's size.

Since our intention is to provide *per-node* fairness and service differentiation each class matches traffic destined to or originating from a single node. In CBWFQ each traffic class i is given a weight w_i and for any time interval $[t_1, t_2]$ during which there is no change in the set of backlogged classes $B(t_1, t_2)$ the throughput achieved by each class i, $T_i(t_1, t_2)$ satisfies the following property:

$$\forall\ i, j\ \epsilon\ B(t_1, t_2), \quad |\frac{T_i(t_1, t_2)}{w_i} - \frac{T_j(t_1, t_2)}{w_j}| = 0 \tag{2.1}$$

CBWFQ has four important features:

- Provides fairness among backlogged traffic classes.
- Ensures channel access with a bounded delay.
- Guarantees minimum throughput for backlogged traffic classes.
- Is applicable for channels with constant capacity and channels with time varying capacity.

In summary, CBWFQ provides a *full separation* among traffic classes, i.e. the minimum guarantees provided for a class are unaffected by the behavior of other classes.

However, in CBWFQ the channel is assumed to be error-free or at the very least, errors must not be location dependent. This means that either all backlogged traffic classes have the ability to transmit at a given time, or none of them can. This basic assumption does not hold in wireless networks since one of the key characteristics of the wireless channel is location dependent bursty channel errors. Thus, CBWFQ is neither fair nor able to provide minimum throughput bounds in this environment.

Another unique concern in wireless networks is that within a WLAN all transmissions are either uplink or downlink due to the shared nature of the wireless link. Obviously, in a wireless network with infrastructure, the access point is the only logical scheduling entity for downlink traffic classes. For uplink classes, on the other hand, each mobile host is asked to perform packet scheduling by itself, unrestrained from the CBWFQ mechanism employed at the access point. This distributed packet scheduling mechanism poses considerable challenges in providing any fairness guarantees over all traffic classes. The effectiveness of a CBWFQ mechanism at the access point could be heavily affected by uplink traffic. Unless a *proper control mechanism over uplink traffic* is used, CBWFQ mechanism cannot provide fairness over all classes in a WLAN.

The reason we focus on CBWFQ lies in its capability to provide *service differentiation* between traffic classes. However, the extent of service differentiation depends heavily on the type of traffic

each one of them carries. As stated in [4], under bursty data traffic, when the network utilization is not very high, CBWFQ provides only little service differentiation. This is due to the *absence* of traffic classes. The term *absence* refers to the unavailability of packets for transmission. In such a case, the CBWFQ scheduler distributes the service that belongs to the absent traffic class to all backlogged traffic classes in the system. As a result, backlogged low-weight traffic classes could get extra service and their actual data rate could be very high. Since the CBWFQ model does not compensate a traffic class for granting its resources during an absence period, the high-weight traffic classes cannot reclaim the service they relinquished when they become backlogged again. In other words, CBWFQ guarantees that Equation (2.1) will hold only if there are no changes in the set of backlogged traffic classes $B(t_1, t_2)$ over the time interval $[t_1, t_2]$, which of course is not the case with bursty data traffic and most real life scenarios. Thus CBWFQ can only provide good service differentiation in case all traffic classes are backlogged.

In summary, there are four significant issues concerning the application of the CBWFQ mechanism at the access point and providing fairness in a multirate physical layer:

1. Location dependent bursty channel errors.
2. Absence of classes.
3. Control over uplink traffic.
4. Multiple transmission rates.

The next chapter presents the architecture of our dynamic CBWFQ mechanism that incorporates solutions for all these issues. The dynamic CBWFQ mechanism is constituted of three parts:

1. The throughput compensation algorithm.
2. The service differentiation algorithm.
3. The performance improvement algorithm.

The first algorithm addresses the issues of location dependent bursty channel errors and absence of classes. The second algorithm deals with exerting control over uplink traffic and the third one incorporates transmission rate information in the scheduling decisions.

Chapter 3

Dynamic CBWFQ scheme

In this chapter we present the three modules that comprise our dynamic CBWFQ mechanism. The first module is a properly configured CBWFQ mechanism that provides service differentiation in terms of throughput for downlink traffic and uplink TCP traffic. The second one is a throughput compensation algorithm that modifies the weights of the CBWFQ mechanism so as to compensate traffic classes that suffered losses due to location dependent channel errors or absence. Finally, the last module is the performance improvement algorithm that provides fairness in terms of temporal shares by reducing the weight of classes associated with nodes that use a degraded transmission rate. The combined functionality of the three modules approximates the fairness scheme described in Section 2.5.

3.1 Service Differentiation Mechanism

The service differentiation mechanism constitutes the basis of our dynamic CBWFQ scheme. As its name reveals, it differentiates the service provided to each wireless node. However, it is not limited to this role and its functionality is extended to permit control over uplink traffic, at least as far as TCP is concerned.

The service differentiation mechanism involves two classes of packets for each node associated to the access router, the ACK and the DATA classes. The ACK class matches TCP acknowledgment packets destined to a host, while the DATA class matches data packets (TCP or UDP) destined to it. Every class is assigned a separate queue for its packets. The service provided to each host, on a congested link, is differentiated according to the weights of its ACK and DATA classes. In case there is unutilized bandwidth it will be shared among the nodes that generate traffic in proportion to their

weights; thus a class could receive more resources than the minimum rate its weight guarantees; if demand for resources is low. In Fig. 3.1 we present a diagram of the service differentiation mechanism, where each queue is served by the dynamic CBWFQ scheduler according to its weight $w_i,\ i\ \epsilon\ [1,4]$.

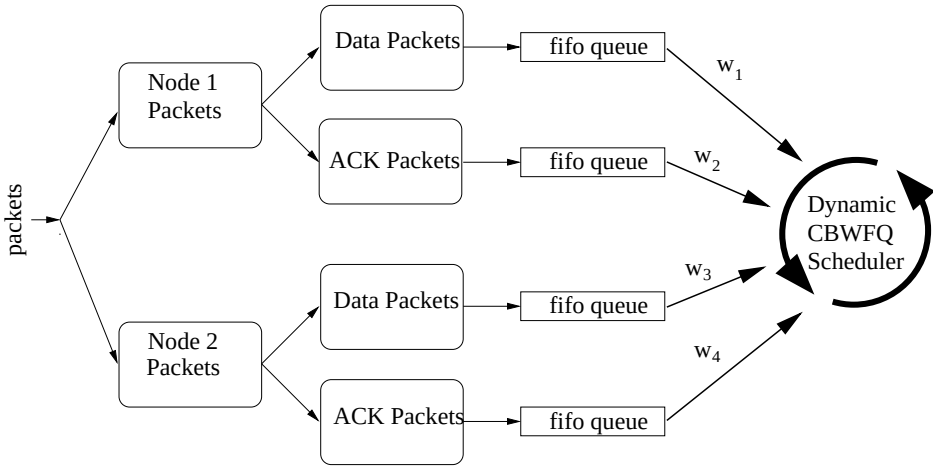

Figure 3.1: Service differentiation mechanism configuration.

Discrimination between acknowledgment and data packets destined to a host provides the means to control its uplink traffic. By controlling the transmission rate of acknowledgment packets we can affect a node's aggregate uplink transmission rate. The function we use for rate controlling ACK classes is based on *experimental* results and is linear:

$$T_{uplink,j} \simeq k \cdot T_{ACK,j} \tag{3.1}$$

where $T_{uplink,j}$ is the uplink rate achievable by node j and $T_{ACK,j}$ is the shaped rate of node's j ACK class (Fig. 3.2) The experiments were conducted on our testbed and a host (europe.csd.uoc.gr) located in a remote LAN. The experiments included scenarios with one or more TCP flows and various packet sizes. Fig. 3.3 presents the aggregate transmission rate for three TCP flows as a function of the acknowledgments aggregate transmission rate. Each point of the plot is a mean value calculated over a period of 60 seconds. The flows originated at a mobile host and were destined to the remote LAN.

Equation (3.1) provides an upper bound for uplink traffic. The rate that a node actually achieves depends on many other parameters as well, e.g., link quality and congestion. The effective deployment of our service differentiation policy depends heavily on the capability of this method to control uplink TCP traffic. Although there is no corresponding method for controlling uplink UDP traffic, our

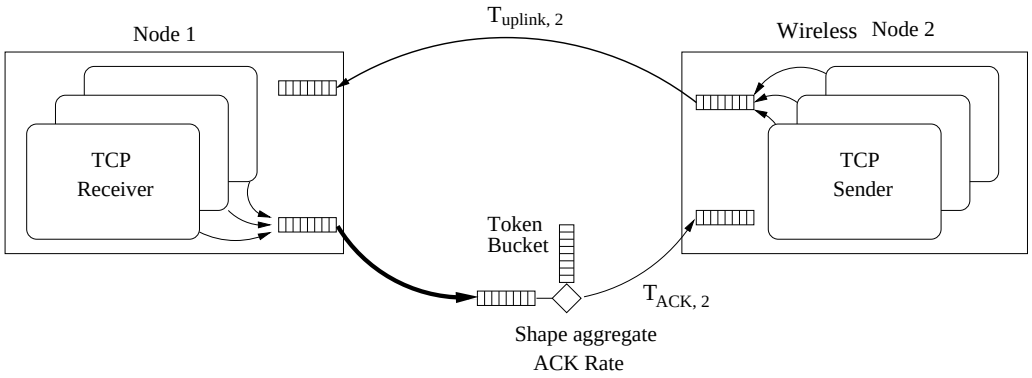

Figure 3.2: Acknowledgment shaping mechanism.

method can be applied when some rate control over UDP is used, as in video streaming and DCCP protocol.

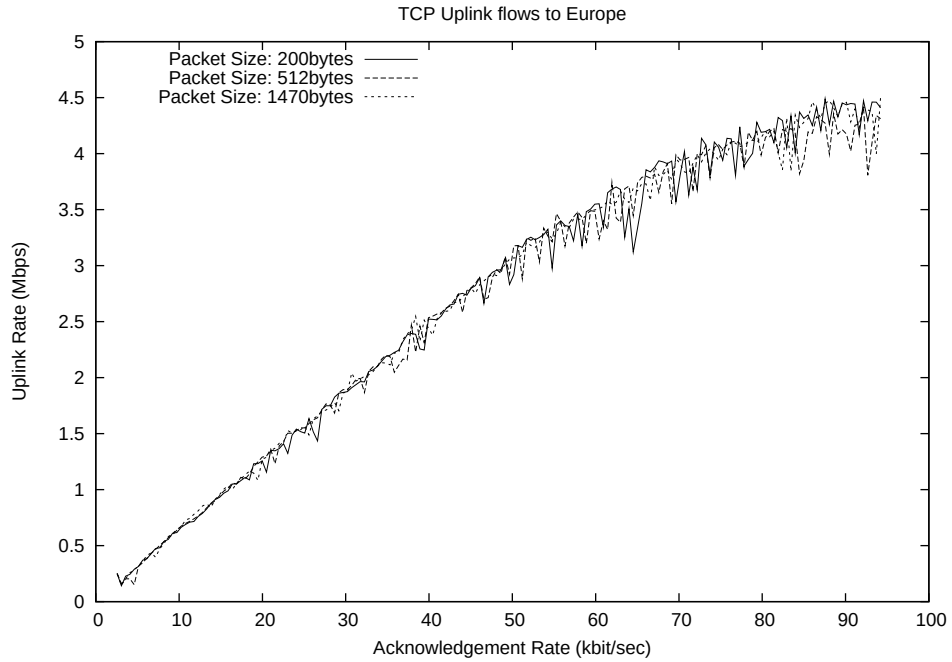

Figure 3.3: Uplink transmission rate controlled by ACK transmission rate.

3.2 Throughput Compensation Algorithm

As mentioned earlier (Section 2.6) the static CBWFQ mechanism presented above has limited effectiveness in providing fairness and service differentiation in the context of wireless networks. This

is due to location dependent channel errors and the possible absence of classes. No guarantees can be given concerning the minimum throughput of backlogged traffic classes, channel access delay and fairness in a short-term basis. However, it is conceivable to achieve long-term fairness by giving extra bandwidth to a class that was absent or that has suffered transmission failures in the past.

There are two major methods for granting extra resources to classes that suffered losses. Before proceeding with their presentation it would be helpful to define the terms of leading and lagging class. A *leading* class is a class that has received more service than its bandwidth share would permit by exploiting unutilized network resources. A *lagging* class is a class that has received less service than its bandwidth share would permit either due to channel errors or absence. The two methods for granting extra resources to a lagging class are:

Reserving bandwidth for compensation. According to this method a fraction of the channel bandwidth is statically reserved for compensation. The reserved resources will be either granted solely to the class with the greatest losses or they will be shared among all lagging classes. The advantage of this method is that compensation of a lagging class does not affect the resources of other classes. However, it can lead to channel underutilization, especially if the reserved resources constitute a large portion of the available bandwidth.

Degrading the service of other nodes. This method is based on borrowing resources from leading classes or lagging classes that have not suffered greater losses than the class to be compensated. There are issues to be addressed concerning the deployment of this method. Special care must be taken to avoid starvation of classes from whom resources are borrowed. Furthermore, degrading the service provided to a leading class must be done in a graceful manner and for a bounded period of time so as to avoid an unfair derangement of the initial bandwidth distribution.

The throughput compensation algorithm proposed in this chapter utilizes the latter method. It performs resource management on top of the CBWFQ mechanism, based entirely on weight manipulation. The algorithm takes feedback periodically by monitoring the actual performance of each node's classes. The duration of the sampling period affects how fast the algorithm responds to changes in the wireless network. The amount of service lost or gained (lag or lead) by class i up to period n is:

$$D_i[n] = D_i[n-1] + achieved\_r_i[n] - target\_r_i[n], \quad n \geq 1 \tag{3.2}$$

where $D_i[0]$ is zero, $achieved\_r_i[n]$ is the rate achieved by class i during period n and $target\_r_i[n]$ is

the rate class i was expected to achieve over the same period :

$$target\_r_i[n] = \frac{w_i[n]}{\sum_j w_j[n]} \cdot C_{cck11} \tag{3.3}$$

where $w_i[n]$ is the weight of class i and C_{cck11} is the 802.11 protocol capacity when complementary code keying at 11Mbps is used. Because of the overheads introduced by the physical and MAC layers, the value of C_{cck11} can not be larger than 6Mbps. The exact value of this parameter will not affect the functionality of the algorithm though it will affect the speed of convergence. Equation (3.2) designates that $D_i[n]$ will be negative, if class i has lost service and positive, if node i has received excess service.

In case class i has not served any packets (absence), its achieved rate will be zero and the whole amount of $target\_r_i$ will be accounted as lost service. In a real world scenario it is expected that all nodes will have prolonged periods of absence. This can result in very small values of D_i for all classes. If D_i is allowed to take limitlessly small values, our aim to compensate each class for its lost service, which is expressed by negative values of D_i, will be impossible to be achieve due to the scarce resources of WLANs. To address this problem, we place a lower bound on D_i values:

$$D_i[n] \geq -(w_{i,initial} \cdot S_l), \qquad \text{for all } i \tag{3.4}$$

where S_l indicates the maximum aggregate loss we will account for in our WLAN and $w_{i,initial}$ is the initially assigned weight to class i. Each class, according to its weight will be compensated for losses up to the value given by equation (3.4). The S_l value is determined by considering the available bandwidth, so that our algorithm can compensate the amount of losses it accounts for.

In a similar fashion, an upper bound is necessary for the values of D_i. As mentioned earlier in this chapter, we intend to compensate lagging classes by reducing the resources assigned to leading classes. The lead of a class is expressed by the positive values of D_i. In case D_i grows too large, it is highly possible that the class will be starved of resources for a prolonged period of time. A scenario that corresponds to this case is when a single class generates traffic, e.g., ftp, while all other classes remain absent. Since the class will be exploiting all the available bandwidth, the value of D_i will grow large. If the other classes begin to serve traffic at some point, the class in the leading state will suffer resource reduction for a prolonged period of time due to compensation of the lagging classes. To address this issue, we use an upper bound for $D_i[n]$.

$$D_i[n] \leq w_{i,initial} \cdot S_g, \qquad \text{for all } i \tag{3.5}$$

where S_g expresses the maximum aggregate gain we will account for in our WLAN. Each class, according to its weight will be charged for the excess service it received up to the value given by

inequality (3.5). A consequence of (3.5) is that small weight classes will not be punished as much as big weight classes, since that would cause their starvation.

The weight of class i will be updated at the end of each period according to:

$$w_i[n+1] = w_{i,initial} + f_i[n] \tag{3.6}$$

where $f_i[n]$ is a function of $D_i[n]$. Function $f_i[n]$ must also be bounded so as to avoid an excess increase of a class's weight that could cause starvation of other traffic classes. It should also result in a smooth decrease or increase of a class's weight. A function that has the above properties and has been used successfully in our experiments is:

$$f_i[n] = \begin{cases} \ln\left(\frac{|D_i[n]|}{C_{cck11}} + 1\right), & \text{if} \quad -w_{i,init} \cdot S_l \leq D_i[n] \leq 0 \\ -\ln\left(\frac{|D_i[n]|}{C_{cck11}} + 1\right), & \text{if} \quad 0 < D_i[n] \leq w_{i,init} \cdot S_g \end{cases} \tag{3.7}$$

The form of this function indicates a non-linear compensation method.

In Fig. 3.4 we plot Eq. 3.7. As Fig. 3.4 illustrates, if $D_i[n]$ grows very large (or very small) compared to the capacity of the wireless network, small changes of its value, due to the compensation algorithm will not be able to change $f_i[n]$ significantly and consequently the weight of the class. This is due to the nature of the logarithmic function. To illustrate this phenomenon suppose we have a traffic class with an initially assigned weight of 25%. For simplification we assume that no weight increase will occur over time (static CBWFQ scheme). If the class has no traffic to send for 10 minutes in an 802.11 network with $C_{cck11} = 6\,\text{Mbps}$ and the value of $D_i[n]$ is updated once in a second, at the end of the 10 minutes interval its will be -900 Mbps. In real-life scenarios it is expected that most classes will have no traffic to send for larger intervals than 10 minutes.

In order to avoid such problems it is important to choose appropriate values for the S_l and S_g parameters of Eq. 3.7 so that we account only for losses that can be compensated. In Fig. 3.5 we present $f_i[n]$ as a function of $D_i[n]$ for $S_l = S_g = 25\,\text{Mbps}$ and $w_{i,initial} = 0.25$.

However, bounding the values of $D_i[n]$ results in an incomplete accounting of lost service and the classes will never be compensated for the total amount of losses they suffered in the past. Although this is necessary considering the limited resources of wireless networks and the need to maintain the stability of the dynamic CBWFQ mechanism, it signals a departure from an absolutely fair approach. Conclusively, the dynamic CBWFQ mechanism has the following properties:

- Provision of short-term fairness among backlogged classes that perceive a *clean channel*.

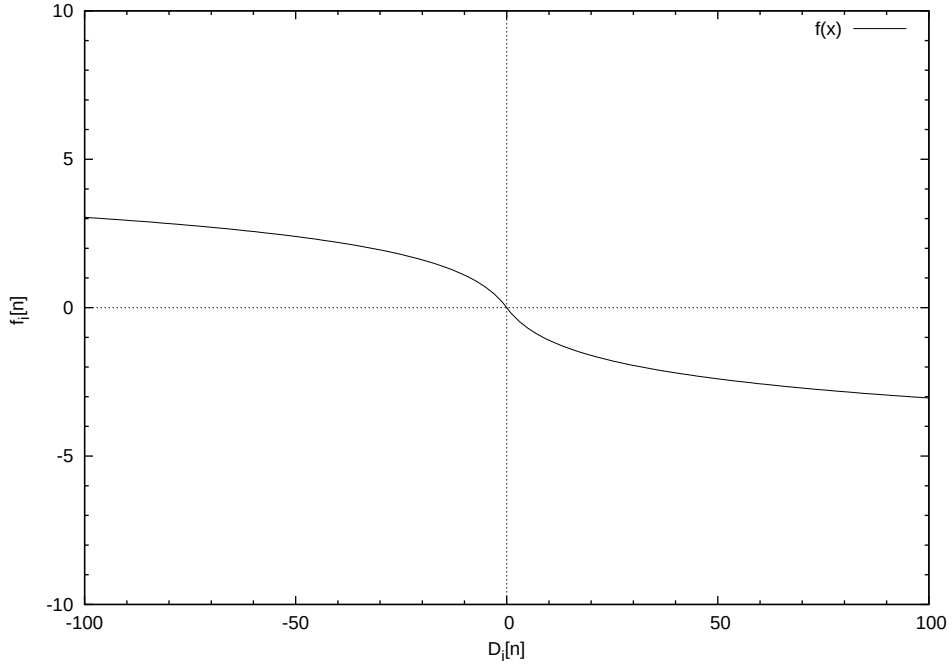

Figure 3.4: $f_i[n]$ for unbounded $D_i[n]$.

- Provision of long-term fairness among all classes with *bounded losses*, due either to channel errors or absence.

Finally, the use of a logarithmic function for $f_i[n]$ has a twofold advantage. If the values of S_l and S_g are chosen small enough, the form of $f_i[n]$ will be almost linear, as in Fig. 3.5, and the system will respond to changes of $D_i[n]$ straightforwardly. On the other hand, if the values of S_l and S_g are chosen to be large, the logarithmic function will prevent the accumulation of weight by the classes that are absent for prolonged periods of time.

3.3 Performance Improvement Algorithm

The performance improvement algorithm incorporates transmission rate information in the scheduling decisions to approximate the fairness scheme expressed in terms of transmission and reception temporal shares (Sections 2.5 and 2.6). As described in Chapter 2, the prolonged channel occupation time, required by nodes using a degraded transmission rate, along with the equal long term channel access probability for all nodes, including the AP, cause fairness and performance problems in 802.11 networks.

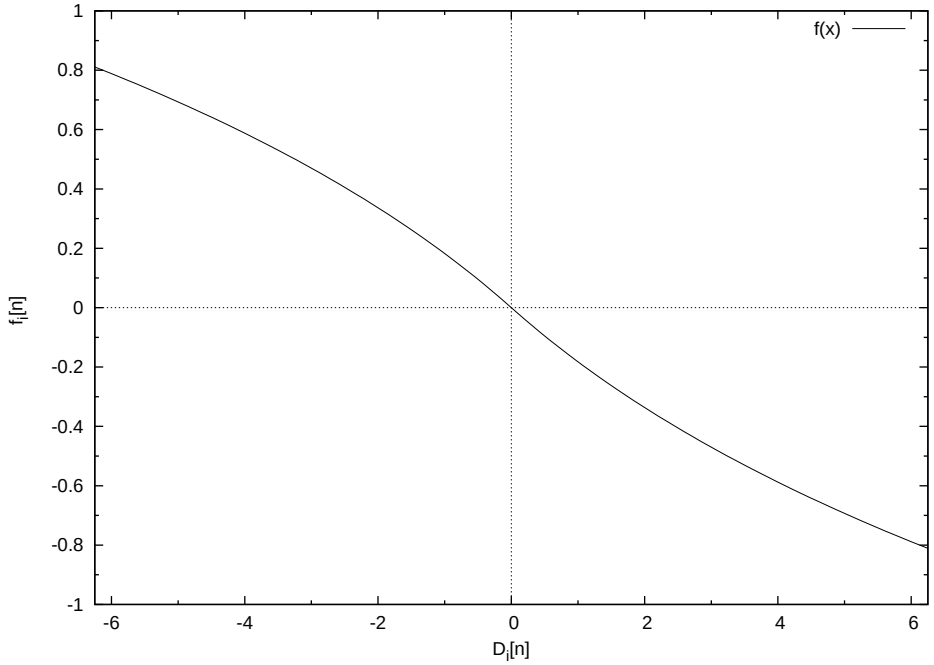

Figure 3.5: $f_i[n]$ for bounded $D_i[n]$.

Obviously nothing can be done to change the channel occupation time needed for the transmission of a single frame by a node using a degraded transmission rate. However, we could reduce the total number of frame transmissions conducted by the node, so that, in long run, the total time it occupies the channel is equal to that of nodes using the 11 Mbps transmission rate. In order to illustrate this approach consider the case of a frame transmission using different transmission rates. The four available modulations in 802.11b networks and the corresponding transmission rates can be seen in Table 3.1. If we define a time unit to be the channel occupation time needed for the transmission of a frame with the CCK-11 Mbps modulation, then transmission of the same frame with the CCK-5.5 Mbps modulation would take 1.5 time units. Similarly, using modulations DQPSK and DBPSK would require 3 and 6 time units, respectively. Consequently, if the node makes, e.g., 1000 frame transmissions, the total channel occupation time for each modulation would be 1000, 1500, 3000 and 6000 time units respectively. Reduce the number of frame transmissions by a factor inversely proportional to the prolonged channel occupation time, the node would have an equal channel occupation time of 1000 time units. We call this factor link quality coefficient (l) and its values appear in Table 3.1.

This can take place by reducing the weight of both ACK and DATA classes that belong to a node transmitting with a degraded rate. Weight reduction forces the scheduler to decrease the number of

Modulation	Transmission Rate (Mbps)	Channel Occupation Time (time units)	Link Quality Coefficient (l)
DBPSK	1	6	1/6
DQPSK	2	6/2	2/6
CCK-5.5 Mbps	4	6/4	4/6
CCK-11 Mbps	6	1	1

Table 3.1: l for each modulation scheme of 802.11b.

packet transmissions from these classes proportionally to the link quality coefficient. Equation (3.6), for updating the weight of class i at the end of each period becomes:

$$w_i[n+1] = (w_{i,initial} + f_i[n]) \cdot l_i[n] \tag{3.8}$$

where l_i is the link quality coefficient of the node. The procedure for periodically updating the value of $l_i[n]$, according to the transmission rate used by the node associated with class i, is implementation specific and will be presented in Chapter 4.

We should note that the correspondence between modulation scheme and the achievable transmission rate is approximate. This approximation was based on experimental results for TCP and UDP traffic in our testbed and holds when the frame payload is larger than the overhead of the frame (52 bytes). Similar results have also been produced via simulations and are presented in Fig. 3.6 [18]. Fig. 3.6 shows the performance, in terms of throughput vs. SNR, for each modulation scheme available in IEEE 802.11b (assuming an optimum link state prediction algorithm).

Weight reduction of nodes with small transmission rate has another significant advantage apart from being a mean to increase fairness in terms of temporal shares. Nodes reduce their transmission rate to cope with continuous transmission failures due to channel errors, since there is a close relationship between the transmission rate of a node and Bit Error Rate (BER). A node with transmission and reception failures not only wastes its resources but also prohibits other nodes from using their own. MAC layer retransmissions, contention window increase, transmission rate reduction and head of line blocking affect the overall network performance. By reducing the weight of such nodes' classes we diminish the appearance of these side-effects.

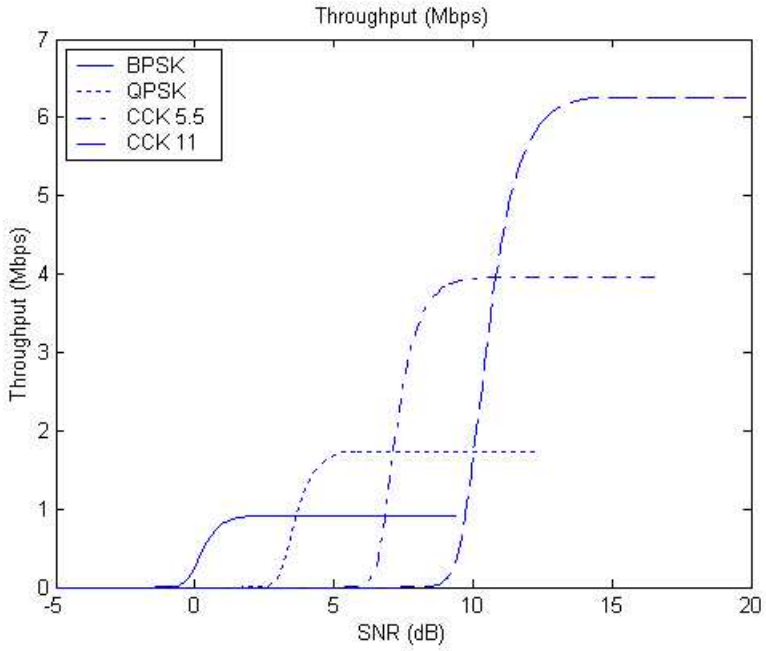

Figure 3.6: IEEE 802.11b throughput vs. SNR.

Chapter 4

Implementation

This chapter contains details about the implementation of our dynamic CBWFQ mechanism and the set up of the experimental testbed. For each module of the mechanism we present the software we used for its implementation and the way it acquires the necessary information to perform its operations, e.g., throughput statistics for each queue and transmission rate information. Furthermore, we analyze the implementation restrictions imposed by the operating system and the drivers of the wireless cards.

4.1 Deployment

There are two possible approaches for the deployment of the dynamic CBWFQ scheme in wireless networks. The first approach incorporates the dynamic CBWFQ mechanism as a service provided by a typical router that interconnects one or more commercial APs with the wired network. This setup has low cost and is capable of providing service differentiation and fairness over a wireless distribution system (WDS). However, an information exchange protocol is necessary for the router to know the transmission rate used by each node associated with an AP. A schematic representation of this generic deployment topology appears in Fig. 4.1.

The second deployment approach implements the dynamic CBWFQ scheme as a service provided by an access router (AR). The AR provides the functionality of an AP enhanced with routing and service differentiation capabilities and facilitates the implementation of our mechanism. Since the AR has the wireless interface attached to it, it is convenient to acquire information from its MAC and physical layers, such as the transmission rate used by each wireless host. The network topology

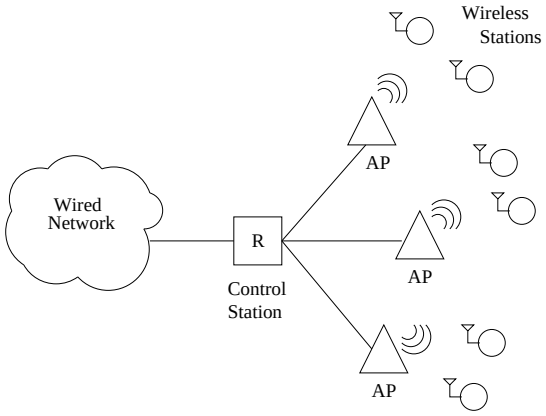

Figure 4.1: Generic network topology.

that corresponds to this approach, which was also the choice for our experimental testbed, appears in Fig. 4.2.

The AR used in our testbed is comprised of a low end, mini-ITX box, with a VIA processor at 533MHz, equipped with an Ethernet card at 100 Mbps and an 802.11b wireless card based on Prism 2.5 chipset. The AR runs the Red Hat Linux operating system with an updated kernel version (2.4.27) which provides the functionality of a software router (ip_forwarding enabled). The wireless card is driven by the HostAP driver (version 0.2.5) in Master mode (functions like an access point).

The wireless stations used for the experiments conducted in our testbed present some variety. Three of them were laptops, whereas the fourth was a desktop computer. Two of the laptops were dual boot systems (Fedora Core 3 and Windows XP) based on Intel Mobile processors with builtin wireless cards (one 802.11b and one 802.11b/g). The third one was a laptop with an external pcmcia wireless card, based on Prism 2.5 chipset, and run the Windows XP operating system. The desktop computer run the Red Hat Linux operating system (Kernel 2.4.27) and was equipped with an external pcmcia card based on Prism 2.5 chipset and a proper adapter. The reason we used such a diversity of wireless stations was the intention to construct a non homogeneous, realistic environment, to experiment with our dynamic CBWFQ mechanism.

4.2 Implementation

The dynamic CBWFQ mechanism was built as a user space program written in the C language. The three algorithms that comprise the mechanism make extensive use of the services provided by the

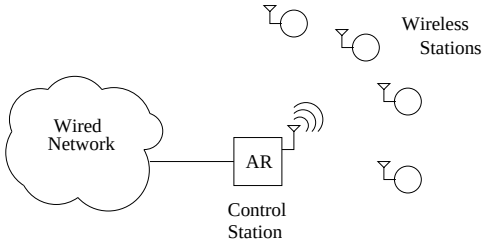

Figure 4.2: Testbed topology.

operating system and the HostAP driver.

4.2.1 Service differentiation mechanism

The service differentiation mechanism is based on Hierarchical Token Bucket (HTB) queuing discipline. HTB is a QoS technique available in Linux based routers that is focused on packet scheduling precision, and ease of use. HTB, as an alternative to Class Based Queuing (CBQ), was developed by Martin Devera at the end of 2001. It has some advantages over other QoS techniques and especially compared with CBQ; it is simpler and more precise in performing traffic sharing and borrowing. It ensures that at least the minimum amount of traffic for a class is provided, and when a class does not use its assigned traffic, the unutilized bandwidth is temporarily distributed to other classes.

Implementation of the HTB queuing discipline is based on Sally Floyd's formal sharing mechanism presented in [17]. It is a prioritized Deficit Round Robin (DRR) scheduler, plus Token Bucket Filter (TBF) in one. The most significant characteristics of HTB are:

- Use of a TBF as an estimator of idle time for each class so that only rate and burst must be specified.
- Precision in rate estimation.
- Precise borrowing control. For each class a guaranteed rate and an upper (ceil) rate are provided so that each class can borrow specific amount of bandwidth from its parent. If several classes are competing for their parent's bandwidth, then they get their share in proportion to their *quantum* (in Mbps). The quantum of a class is defined as its rate divided by the *r2q* parameter, which by default gets the value 10.
- Permits cross-device bandwidth sharing.

- One can check the actual rates of classes and potentially readjust the link-sharing policy implementation.

However, there is an issue concerning the employment of the borrowing mechanism for the ACK classes in the service differentiation mechanism. HTB assumes a Maximum Transfer Unit (MTU) of 1500 bytes and the minimum quantum is taken to be 15 KBps or 120 Kbps when r2q is 10. The calculation of the minimum value of the quantum is as follows:

$$\frac{\frac{1500}{1}\frac{\text{bytes}}{\text{sec}}}{r2q} = 1500 \cdot 10\,\text{bytes/sec} = 15\,\text{KBps}$$

Likewise, if the r2q parameter takes it smallest value, which is 1, then the minimum quantum will be 12 kbps ([19]). If we permit an ACK class to borrow such quantities of bandwidth then we would loose control over uplink traffic as a result of equation (3.1) which we repeat here for convenience:

$$T_{uplink,j} \simeq k \cdot T_{ACK,j}$$

Equation (3.1) describes the relationship between aggregate downlink acknowledgments rate and aggregate uplink data rate for TCP traffic. The value we used for the coefficient k is 62.5 and was derived from experiments similar to the one presented in Fig. 3.3. Even when the MTU takes the smallest acceptable value, which is 256 bytes, the minimum quantum would be adequate to produce *excessive* uplink traffic. Consequently we avoided use of the borrowing mechanism of HTB in the case of ACK classes.

4.2.2 Throughput Compensation Algorithm

The throughput compensation algorithm implementation is based on traffic statistics exported by the Linux `tc` tool. The `tc` tool is used for setting up and *updating* the QoS mechanism described above, as well as monitoring its performance. A typical output of the tc tool is as follows:

```
qdisc pfifo 320:  limit 25p
Sent 311394 bytes 5763 pkts (dropped 0, overlimits 0)
backlog 9p

qdisc pfifo 310:  limit 25p
Sent 313542 bytes 5803 pkts (dropped 0, overlimits 0)
backlog 7p

qdisc pfifo 220:  limit 25p
Sent 5998740 bytes 4265 pkts (dropped 0, overlimits 0)
backlog 17p

qdisc pfifo 210:  limit 25p
Sent 6447258 bytes 4571 pkts (dropped 0, overlimits 0)
backlog 17p

qdisc htb 1:  r2q 10 default 2 direct_packets_stat 62
Sent 13113703 bytes 20463 pkts (dropped 0, overlimits 21324)
```

The above entries export traffic statistics for the priority FIFO queues in a HTB queuing scheme that is presented in Fig. 4.3. Each class is uniquely identified by a handler of the form x:x, the handler 1:0 (or 1:) is typically assigned to the root HTB queuing discipline. Similarly each FIFO queue is identified by a number, e.g. 210, 310 and has a specific size which is given by `: limit 25p` (25 packets). The last entry (qdisc htb 1:), that appears in the `tc` output, provides collective statistics overall queues.

It is obvious, from the output given by the `tc` tool, that the information needed for the implementation of the throughput compensation algorithm is given for each queue at the second line of each entry, e.g., `Sent 313542 bytes`.

4.2.3 Performance Improvement Algorithm

Implementation of the performance improvement algorithm requires knowledge of the transmission rate used by each node. This knowledge is provided by the HostAP driver as a separate pseudo file for each node associated with the AP. The files are located at the `/proc` file system. A file of this type has the form:

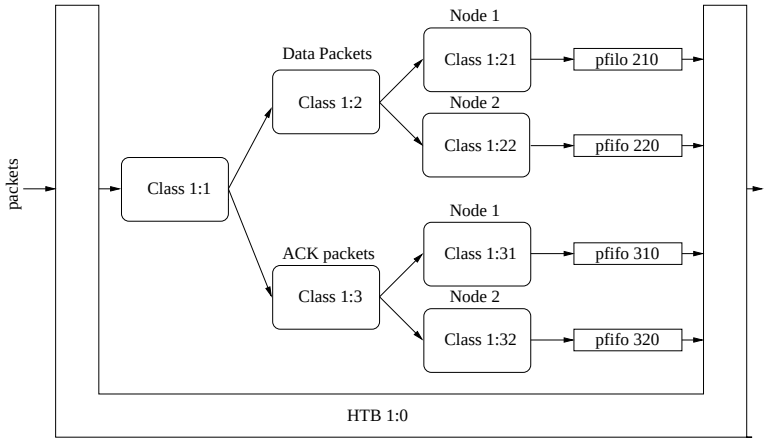

Figure 4.3: HTB class hierarchy.

```
STA=00:04:23:73:15:27
users=0
aid=2
flags=0x0003 AUTH ASSOC
capability=0x01
listen_interval=10
supported_rates=1Mbps 2Mbps 5.5Mbps 11Mbps
jiffies=103394055
last_auth=103388905
last_assoc=103388905
last_rx=103394038
last_tx=103394037
rx_packets=10495
tx_packets=5393
rx_bytes=14267443
tx_bytes=511304
buffer_count=0
last_rx:  silence=-97 dBm signal=-33 dBm rate=11 Mbps
tx_rate=110
tx[1M]=1
```

tx[2M]=0

tx[5.5M]=0

tx[11M]=5392

rx[1M]=1130

rx[2M]=1323

rx[5.5M]=2885

rx[11M]=5188

The information related to the performance improvement algorithm lies within the last 10 lines of the file, presented with text in italic form. The first one of them contains information for the reception of the last packet sent by the station. The lines beginning with `tx` enumerate the packets that have been transmitted to the wireless station by the AP, for all possible transmission rates. The lines beginning with `rx` enumerate the packets that have been transmitted to the AP by the wireless station, for all possible transmission rates.

The algorithm we use to periodically update the value of the link quality coefficients of the two classes associated with the node that is related to the file is Algorithm 1. Reduction of the link quality coefficient, for a node, will occur if at least three packets ($var\_rxR \geq 3$) have been transmitted with the same low transmission rate (R) over a period of one second. One of these three packets must be the last packet ($rate_{lp}$) transmitted by the node at the time the file was opened for reading. This algorithm diminishes the possibility to reduce the link quality coefficient of a node that will not actually use a low transmission rate over the next period. In order to clarify the use of these conditions in our algorithm, it would be helpful to have an insight of the algorithms that control the transmission rate selection for each node.

It is true that neither the IEEE 802.11 standard specification, nor the 802.11b supplement specify the algorithm for performing dynamic rate switching. The companies that are manufacturing 802.11 interfaces have to come up with their own proprietary algorithms. All of the known vendors of 802.11 equipment use statistics based approaches for rate control. Their algorithms are either implemented in software, as part of the device driver, or as part of the chipset hardware.

In the research community, another class of rate control algorithms is being studied. These rate control algorithms use SNR-related information as a feedback to improve the sensitivity to changes in link conditions. Up till now, however, such algorithms have not been implemented in practical systems and only simulation results have been reported. Below we will describe the main representatives of

Algorithm 1 Deriving l_i value based on transmission rate.

1: **if** $(var\_rx1 \geqslant 3$ and $rate_{lp} = 1Mbps)$ **then**
2: $l_i = 1/6$;
3: **else if** $(var\_rx2 \geqslant 3$ and $rate_{lp} = 2Mbps)$ **then**
4: $l_i = 2/6$;
5: **else if** $(var\_rx5.5 \geqslant 3$ and $rate_{lp} = 5.5Mbps)$ **then**
6: $l_i = 4/6$;
7: **else if** $(var\_rx11 \geqslant 3$ and $rate_{lp} = 11Mbps)$ **then**
8: $l_i = 1$;
9: **else**
10: $l_i = l_i$;
11: **end if**

applied rate control algorithms.

Throughput-based rate control. According to this approach a constant small fraction (10%) of the data is sent at the two adjacent rates to the current one. Then, at the end of a specified decision window, the performance of all three rates is determined by dividing the number of bytes transmitted at each rate by their cumulative transmission times. Finally, a switch is made to the rate that provided the highest throughput during the decision window. Atheros uses this algorithm in the NIC driver that they provide for their 802.11a products based on AR5000 chipset.

To collect meaningful statistics, the decision window has to be quite large (i.e., about one second). On the one hand this makes the algorithm resilient to short-lived changes in the link-quality caused by, for example, Rayleigh fading. On the other hand, it prevents swift reactions to long lived changes in link conditions, which noticeably affects the real-time performance of streaming applications.

FER-based control. In this approach, the Frame Error Rate (FER) of the data stream transmitted over the link is used to select an appropriate rate. The FER can easily be determined since under 802.11, all successfully received data frames are explicitly acknowledged by sending an ACK frame to the sender; hence, a missing ACK is a strong indication of a lost data frame. By counting the number of received ACK frames and the number of transmitted data frames during a rather short time window, the FER can be computed as the ratio of the two.

If the FER exceeds some threshold and the current rate is not the minimal rate (1Mbps), then the algorithm will switch to the next lower rate. If, on the other hand, the FER is close to zero or

below some threshold, the link will be probed at the adjacent higher rate with a few (usually even only 1) frames. In case all of them get acknowledged, the algorithm will switch to that rate. In order to prevent the control algorithm from oscillating between two adjacent rates, the upscale action may be prohibited for some time after the downscale decision.

The width of the time window and the thresholds mentioned above are critical for the performance of the FER-based algorithm. The optimal settings of the parameters are dependent on the link and the application, but are generally fixed at design time. Again, this hampers the performance of streaming applications, since a time window tuned for quick responses of typical download applications yields unreliable FER statistics at low traffic rates. Hence many frames are transmitted at a non-optimal rate.

Retry-based rate control. An improvement over the FER-based approach is to downscale immediately when the MAC is struggling to transmit a frame correctly over the link. That is, to select the next lower rate after a small number of unsuccessful retransmissions (usually 5-10 retries). This approach is implemented in hardware, as precise control of the rate setting in between retransmissions (of the same frame) is required.

The advantage of the retry-based approach is that it combines a very short response time (a few frames) for handling deteriorating link conditions (downscaling) with a low sensitivity to traffic rates. The price to be paid is that the control algorithm is rather pessimistic. Relatively short error bursts cause long drops in throughput because upscaling to higher rates takes much longer than downscaling, due to the need to collect a meaningful FER and to prevent oscillation.

The HostAP driver used in our testbed makes use of a slightly different method. If the physical layer fails to transmit two adjacent packets then the transmission rate is reduced to the next lower value. An upscale of the transmission rate is delayed until a series of 50 packets has been transmitted successfully.

Consideration of all the factors described above directed us to design Algorithm 1. We did not know the exact method used by each wireless card in our testbed, although we did know that none of them was using the throughput based rate control, since we never observed frame transmissions at adjacent rates. However, we noticed that one of the nodes was sending a single frame using the 1 Mbps transmission rate from time to time. In order to avoid a needless weight reduction for the node sending single packets with a low transmission rate we required the existence of three packets, transmitted with the same low rate, before conducting a weight reduction.

Furthermore, both FER-based and Retry-based rate control algorithms intentionally delay the upscaling of the transmission rate of a node so as to avoid oscillations between adjacent rates. If the transmission of the last packet sent by a wireless host was made with a low transmission rate then there is a great chance that the next packets will be transmitted with the same rate. This could happen either in case the bad link conditions persist or because of the delay imposed by the rate control algorithms.

Chapter 5

Experimental evaluation

In this chapter we evaluate the performance of the proposed dynamic CBWFQ scheme and demonstrate its effectiveness in resolving the issues presented in Chapter 2. The experimental testbed took place in a typical office environment. All channel distortions, fades and interferences occurred in a random way as people moved inside the office. The channel state for each node was very good when there were no intervening obstacles. The S_l and S_g system parameters were set to 25 Mbit for all five experiments.

Uplink Service Differentiation. The aim of this experiment is to exhibit the capability of our mechanism to differentiate uplink traffic. The scenario included two mobile hosts. Each host generated an uplink TCP flow destined to the wired host. Traffic generation was based on `Iperf` client-server pairs. The two ACK classes were initially assigned equal bandwidth shares. Every 60 seconds the weights of the two classes were changed. The throughput of each uplink flow was measured at the server side (receiver) of iperf and the results can be seen in Fig. 5.1. There we present the ratio of throughputs for the uplink flows as a function of the ratio of weights of the ACK classes. The linearity of the function verifies the ability of our mechanism to differentiate uplink TCP traffic.

Fair bandwidth distribution in case of bursty errors. The target of the second experiment is to demonstrate the effectiveness of the throughput compensation algorithm in redistributing bandwidth fairly. For this experiment three wireless hosts were used. Each one of them was receiving a persistent TCP flow. The available bandwidth was initially allocated equitably among the three DATA classes. In Fig. 5.2 we present the throughput achieved by each flow, over time, without the use of the dynamic CBWFQ mechanism. Nodes 2 and 3 suffered severe losses at the 10th and 30th second respectively, which, combined with other less important losses, resulted in an unfair bandwidth distribution, similar

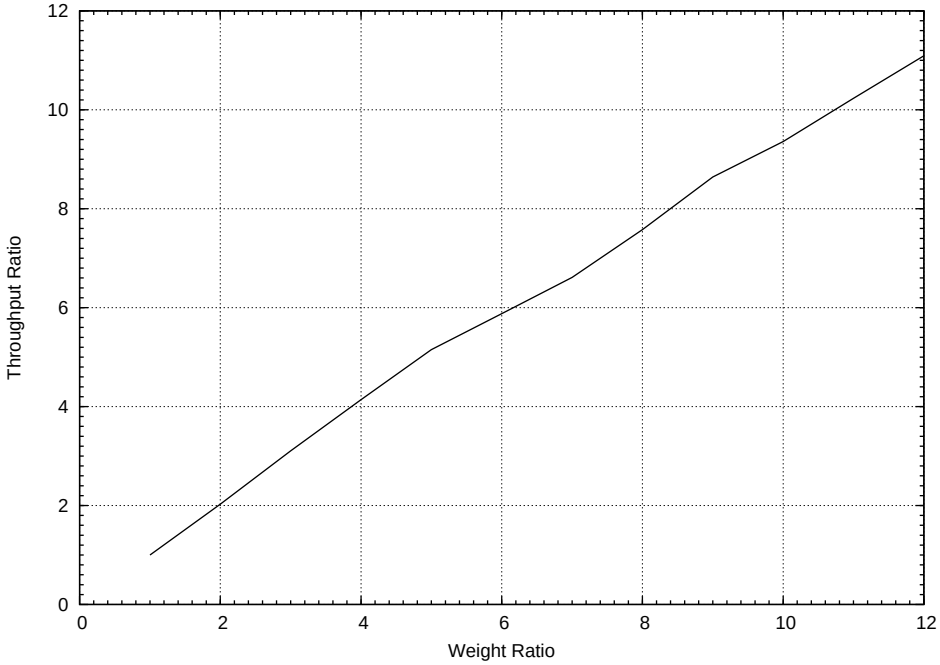

Figure 5.1: Uplink service differentiation.

to the case presented in Fig 2.2. Fig. 5.3 presents the throughput achieved by each flow when the dynamic CBWFQ mechanism was activated. In this case, although nodes 1 and 3 lost service at the 26th and 44th second, they were compensated for their lost service as indicated by the throughput overshoot that follows. Furthermore it can be seen that during the declination of a node's performance other nodes were permitted to utilize the bandwidth assigned to it.

Uplink-Downlink Unfairness. This experiment shows that our mechanism resolves the uplink-downlink unfairness problem. As stated in Chapter 2 the main reasons causing this unfairness issue are buffer availability at the base station and the fact that although all downlink traffic is transmitted by the AP the latter has the same channel access probability as any other node. Our mechanism separates data packets from acknowledgment ones and schedules them through different queues. Furthermore, it performs rate control over ACK classes in order to constrain uplink traffic and thus makes the access router resilient to uplink-downlink unfairness. For this experiment we used three nodes; each one sent 3 TCP data flows and received 3 TCP data flows. Additionally the total buffer of the wireless interface was reduced to 60 packets from the default value of 100 packets. This experimental setup caused the output buffer of the Access Router to overflow and exhibit a severe uplink-downlink unfairness phenomenon. Fig. 5.4 is a snapshot taken from the Windows performance monitor of

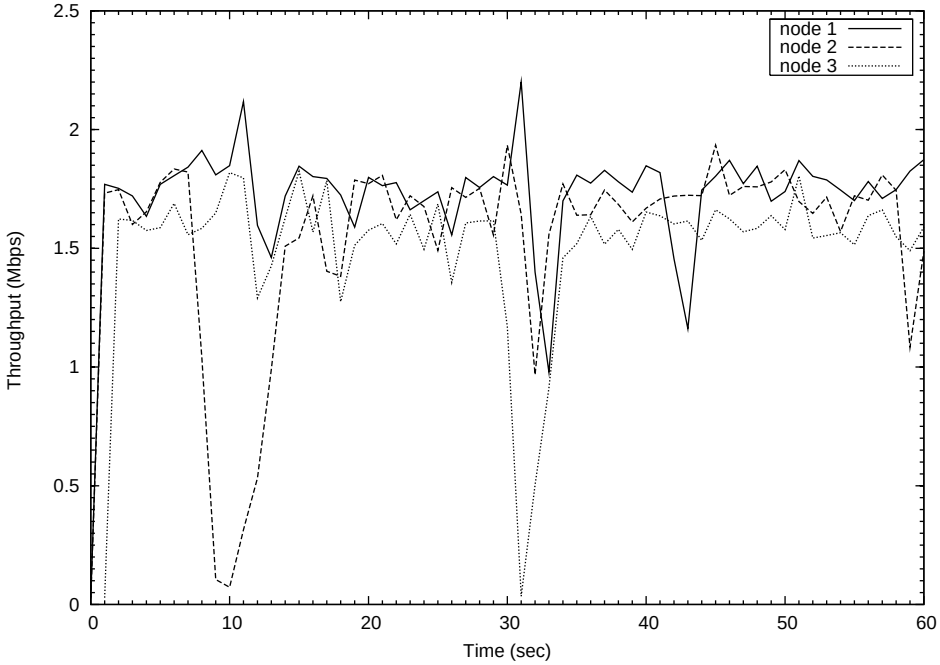

Figure 5.2: Unfairness due to channel errors.

a node and presents this phenomenon as well as its fade out when our mechanism was activated at the 70th second (a vertical continuous line is used to separate the two parts of the figure) of the experiment. DATA and ACK classes were initially assigned equal shares of bandwidth. The roughness of the aggregate downlink throughput seen in the figures occurs because of the large number of flows serviced by the access point in a heavily congested link.

Response time measurements for web traffic. This experiment points out the performance improvement that results from compensating service loss due to absence. Since this aspect focuses on bursty data traffic, Web browsing is used as a case study. An Apache web server was installed on the wired host and three objects of sizes 50, 152, and 500 KBytes were available for download. Two wireless stations were used for this experiment; the first one made an ftp transfer of a large file while the second one performed HTTP requests using the `wget()` utility. The model of network traffic produced by the HTTP protocol is empirical and is based on [20]. An exponentially distributed think time was assumed, with a mean of 20 seconds. The wireless host that performed the HTTP requests was a Linux box with a similar setup as the Access Router. It run the Red Hat 9 operating system with 2.4.27 kernel and was a equipped with a Prism 2.5 wireless card driven by HostAP in managed mode (client). Each DATA class was initially assigned half of the available bandwidth. The experiment was

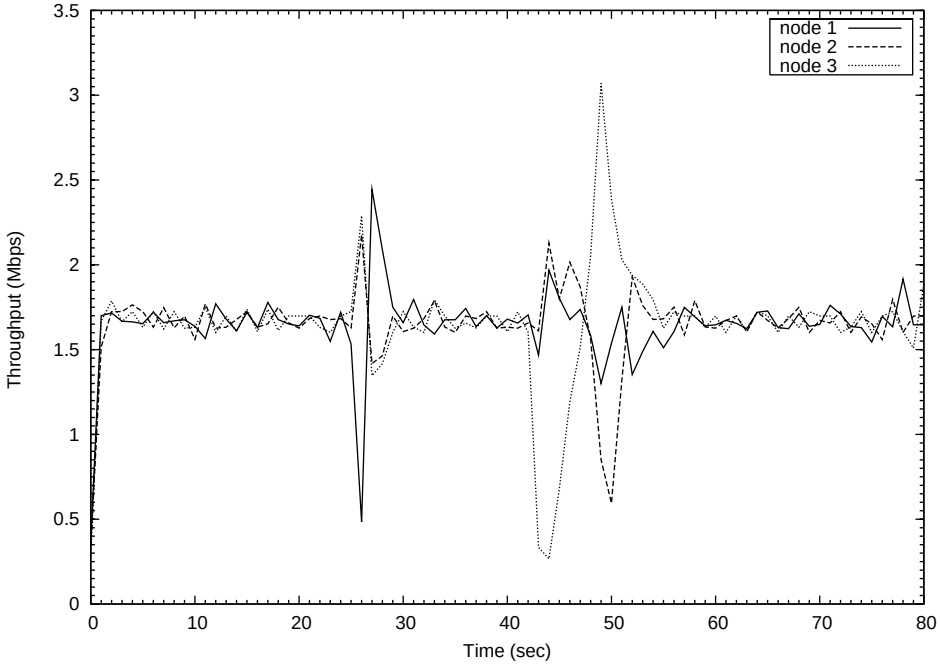

Figure 5.3: Fairness in an error channel.

comprised of 20 HTTP requests for each object and the mean response times can be seen in Fig. 5.5. The performance improvement is significant for small objects while it reduces for larger ones. This is due to the limitation imposed to the amount of service that a class can be compensated for, which is controlled by the S_l parameter.

In order to ascertain the effectiveness of our mechanism in case of increased load we conducted a series of experiments with two laptops performing HTTP requests for the same object (152 KByte). A third laptop was used for reception of a persistent TCP flow using an Iperf client-server pair. In addition to incrementing the number of laptops we gradually reduced the mean think time from 20 sec to 4sec with a step of 2 sec. For each value of the mean think time each station performed 20 HTTP requests. The experiments were conducted once without the dynamic CBWFQ mechanism and then repeated with the mechanism activated. The mean response time over these 20 samples was calculated for each node and the results appear if Figs. 5.6 and 5.7.

Fig. 5.6 and 5.7 illustrate the performance improvement that results from compensating losses due to absence in case of bursty data traffic. This is due to the fact that nodes transmitting or receiving data sporadically will be compensated for their absence with an increase of their classes'

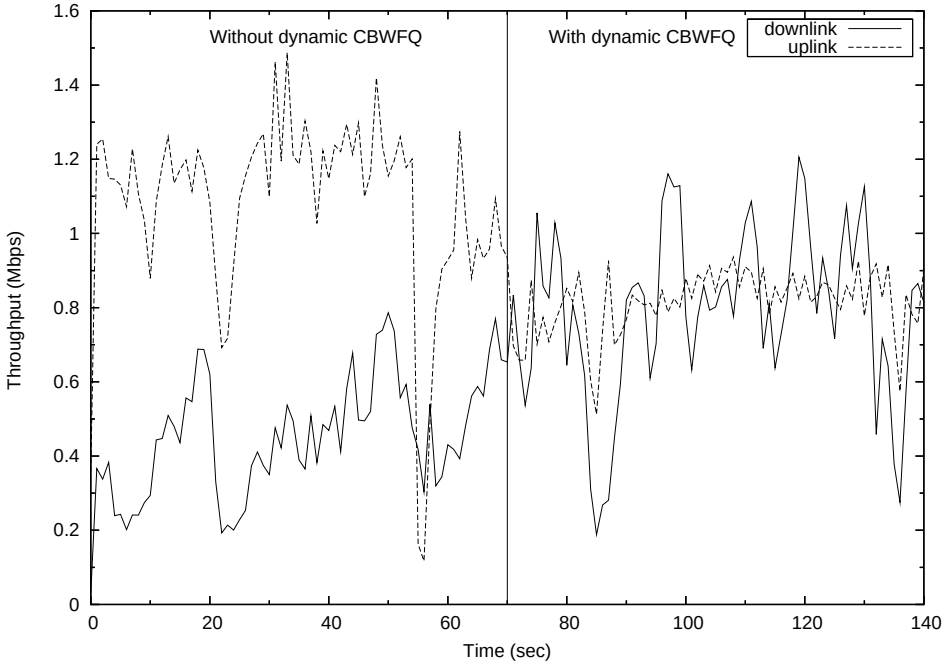

Figure 5.4: Resolving uplink-downlink unfairness.

weight. This increment of the weight is the reason for the performance improvement. Of course, if the HTTP object requests are issued too often, e.g., back to back, there will be no weight increase due to absence and the object will be downloaded with the rate initially assigned to the node and guaranteed by the CBWFQ mechanism. Furthermore, the increment of the weight is not permanent. As the node's class starts servicing traffic, the weight will reduce back to its initial value since the node will be compensated for its losses (as bounded by $-w_{i,init} \cdot S_l$) due to absence . If the size of an HTTP object is big enough, each part of it will be downloaded with a different rate as the weight of the class decreases back to its initial value. As a result, the performance improvement will be less in this case compared to the download of a small HTTP object that can be downloaded completely while the weight sustains a large value.

Performance anomaly of 802.11. This final experiment presents the behavior of our mechanism in a performance anomaly scenario. We used three wireless hosts and each one was generating an uplink TCP flow destined to the wired host. The ACK classes were initially assigned equal shares of the available bandwidth. In Fig. 5.8 we present a typical 802.11 performance anomaly scenario, similar to the one presented in Fig. 2.4. Node's 1 transmission rate oscillated between 11Mbps and the rest three possible rates, and forced an analogous performance reduction for all nodes. Each alternation

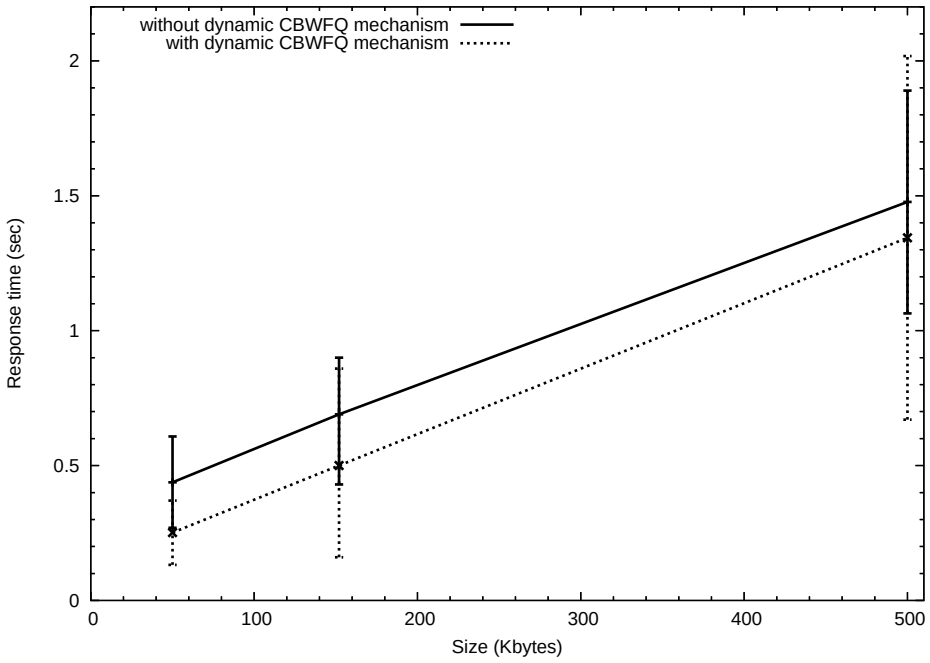

Figure 5.5: Improving response time for Web traffic.

of the transmission rate resulted in a short connection disruption between node 1 and the AP. This explains the short periods of zero throughput that signal the begin and the end of each transmission rate change. Fig. 5.9 shows a similar scenario with the dynamic CBWFQ mechanism activated. It can be seen that, by reducing the weight of the node transmitting with a lower rate than 11Mbps, we prevented the performance degradation of the rest nodes as stated in Section 3.3.

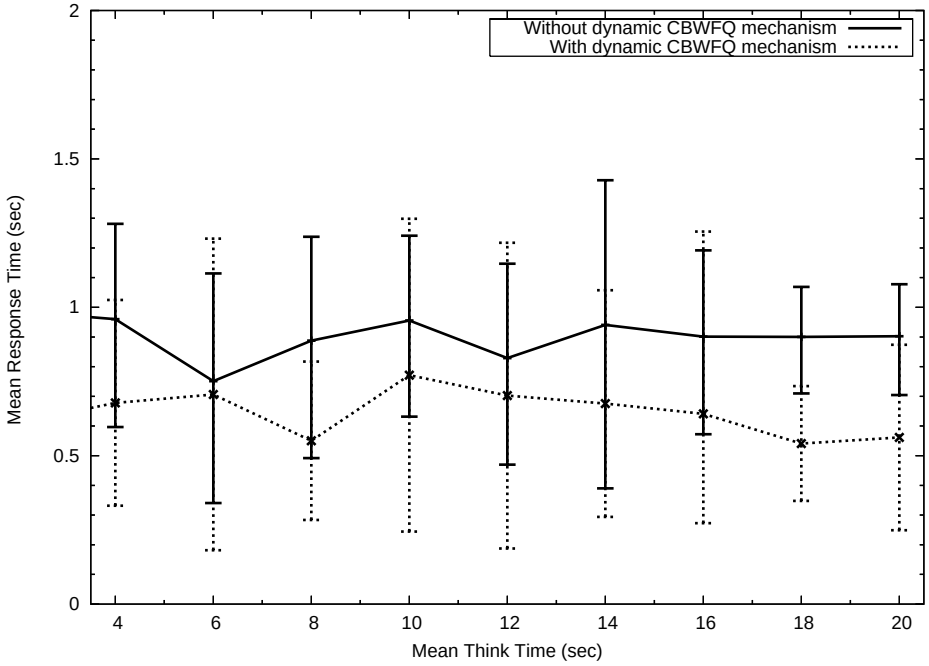

Figure 5.6: Mean response times for the first node.

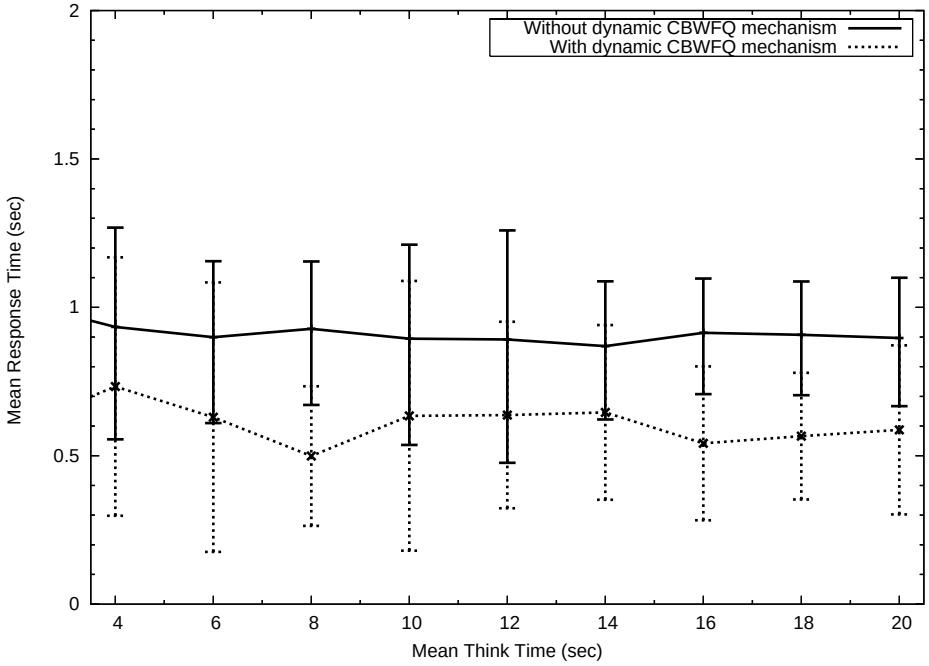

Figure 5.7: Mean response times for the second node.

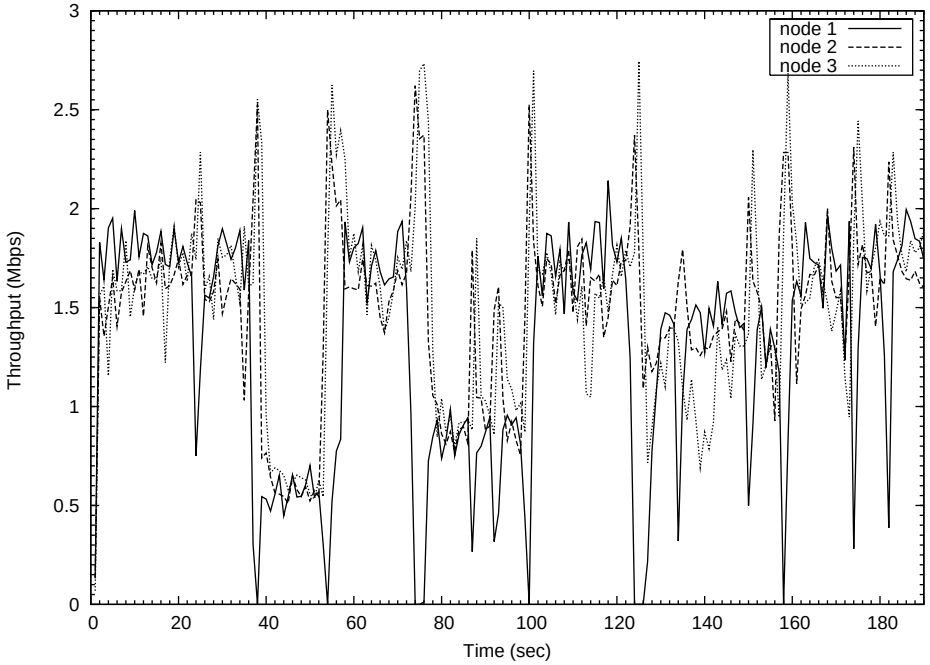

Figure 5.8: Performance anomaly.

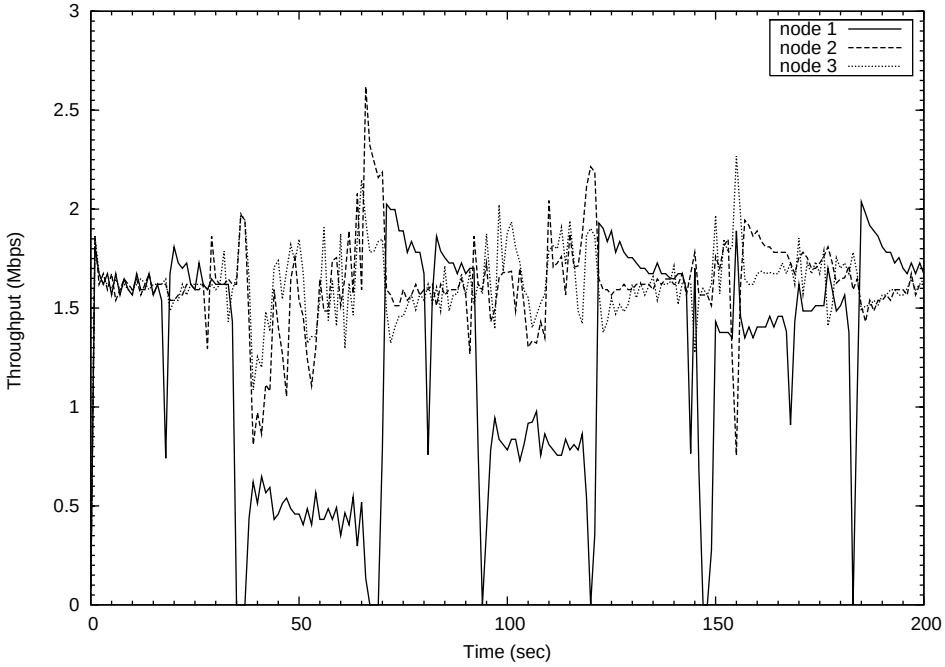

Figure 5.9: Resolving performance anomaly.

Chapter 6

Related Work

Service differentiation in wireless networks has been studied by several researchers. In [3] the authors combined the class-based queueing (CBQ) mechanism with a Channel State Dependent Packet Scheduler (CSDP), that was initially proposed in [14], and a compensation algorithm in a manner that is conceptually similar to ours. However, their scheduler performs packet scheduling based on a characterization of each packet queue rather than on its weight. Each packet queue is characterized as either satisfied or unsatisfied and as either restricted or not. A queue is considered to be unsatisfied (unfairly treated), if it has sent less traffic than its percentage of the throughput and the queue's backlog exceeds a specific threshold; otherwise it is considered satisfied. A queue is restricted if it has sent more than its percentage of throughput and there exists at least another unsatisfied queue[1]. A round-robin scheduler is used which checks the queues sequentially and determines which is the next queue from which a packet should be send. A queue is allowed to transmit if it is not restricted or if it is restricted but the links to the receivers of all the unsatisfied queues are in a bad state, while the link of the receiver for the restricted queue is in a good state. The link goodness estimation is based on RTS-CTS exchange, whereas in our mechanism we utilize the transmission rate as an indication of link quality. The unsuccessful exchange of these messages affects the maximum number of MAC retransmissions for packets destined to the host that initiated the exchange. The latter method, however, is incompatible with the existing implementations of 802.11 standard. The multirate physical layer and its effect in overall network performance are not taken into consideration; the same holds for uplink traffic control and accounting absence as loss of service.

In [21] Mahadevan *et al.* describe the design and implementation of an enhanced Differentiated

[1]The characterization of a queue is updated at every packet arrival.

Services (Diffserv) architectural framework for providing QoS in wireless networks. Their Diffserv framework takes into consideration several factors, including signaling requirements, mobility, losses, lower wireless bandwidth and power constraints. However, their compensation mechanism is based on reserving bandwidth by creating a special purpose compensating class rather than using dynamic weight adjustments. Packets are either intentionally re-marked in order to be scheduled through the compensating class or a borrowing mechanism is utilized so that the compensating class can lend its bandwidth to lagging classes. Finally, there is no anticipation for the impact of the multirate physical layer on the network performance.

With regard to fairness, of particular interest are the works in [4], [5], [6], [7], [8]and [9], presenting, MAC layer, fair scheduling algorithms. All these studies, except [8] and [9] which assume a CSMA/CA MAC protocol[2], consider a packet cellular network, where the base station performs the scheduling of both uplink and downlink packet transmissions in its cell. It should be mentioned here that the 802.11 standard defines the use of a contention free MAC function called Point Coordination Function (PCF), which is not yet implemented in any of the 802.11 commercial products. All these algorithms can be thought of as instances of a unified wireless fair queuing architecture, which consists of the following five components:

- The *error-free service*, which defines an ideal fair service model assuming no channel errors, e.g. Weighted Fair Queuing (WFQ), Weighted Round Robin (WRR) and Start-Time Fair Queuing (STFQ).

- The *lead and lag model* in wireless service, which determines which flows are leading or lagging their error free service, and by how much.

- The *compensation model*, which compensates lagging flows that perceive an error-free channel at the expense of leading flows, and thus addresses the key issues of bursty and location-dependent channel errors in wireless channel access.

- *Slot queues and packet queues*, which allow for the support of both delay sensitive and error sensitive flows in a single framework and also decouples connection-level packet scheduling policies.

- *Channel monitoring and prediction*, which provides a reliable and accurate measurement and estimation of the channel state at any time instant for each backlogged flow.

[2]Incompatible with the 802.11 one.

The dynamic CBWFQ mechanism itself belongs to this big family of algorithms since its architectural design matches this unified wireless fair queuing framework. However, the dynamic CBWFQ mechanism maintains an applicative approach to improving fairness and performance in 802.11 based wireless networks. Unlike these throughput compensation algorithms, it does not require changes at the MAC layer of 802.11 since it is implemented at the network layer and utilizes cross layer information readily available at the access router.

From the studies referred above, of particular interest to our work are [4], where Zhu *et al.* introduced the notion of accounting absence as loss of service, but they didn't correlate it with weight adjustments as a mean to improve fairness and performance, and [1] where the authors redefine fairness in terms of time shares. By granting temporal fair shares of the channel to each traffic source they improve fairness and performance in multi-rate physical layers. However, their algorithm suffers from all weaknesses described above for the fair scheduling algorithms.

An alternative method that achieves service differentiation and improves fairness is proposed by the authors of [22]. In order to control the transmission rate of a wireless node they control its capability to generate bursts of packets. This functionality is provided by the Enhanced DCF function of the 802.11e protocol. Although their method is simple and efficient, it conflicts with current implementations of the 802.11 protocol.

Recently, in [15], Pilosof *et al.*, proposed the dynamic adjustment of TCP window in order to control uplink traffic. This method was also the subject of studies [23] and [24], for wireline networks. This method provides the means to control uplink TCP flows and achieve downlink and uplink service differentiation. However, unlike our ACK shaping dynamic CBWFQ mechanism, it requires access to the packet's header so as to alter the advertised window field. This fact prohibits the coexistence of this method and security protocols like IPSec that encrypt packets' content. Finally, it is difficult to know the exact number of active TCP flows in order to calculate the proper window field value for each one, whereas in our mechanism we differentiate traffic on a per node basis.

The wireless link-state estimation problem has also been addressed by many researchers. In [25], Aida *et al.*, make use of the mean signal to noise ratio in order to characterize the state of the wireless links connecting the base station with the hosts. This method nevertheless requires knowledge of the SNR at the region of the mobile host as well as at the region of the base station. Therefore, a periodical exchange of messages between them, containing such information, is necessary. This form of signaling protocol increases the network overhead and requires software changes at the wireless

stations, whereas, knowledge of the transmission rate used by each node is already included in each packet header and can be used efficiently as an indication of the link quality.

Another line of research, [14], suggests to employ MAC level acknowledgements as a criterion for the characterization of the wireless link. The authors argue that if the transmission error of an acknowledgement occurs because or a bad CRC check then the most probable reason is a collision, otherwise the error is attributed to the bad link state. MAC level ACKs is a fast and reliable way to identify the link quality between a node and the base station, however, utilization of this information can only be made by a MAC layer scheduler that is capable of altering its scheduling decisions in a per packet fashion, e.g., in case a link turns to be a bad state, the scheduler should defer the transmission of the unacknowledged packet and schedule a packet from another queue. Nevertheless, such methods do not comply with the 802.11 standard and we could not incorporate them in our dynamic CBWFQ mechanism.

Chapter 7

Conclusions and future work

In this thesis we presented the design and implementation of a dynamic CBWFQ mechanism that provides service differentiation, improves fairness, and performance in 802.11 based wireless networks. Our work was motivated by the fact that the 802.11 protocol has adopted a fairness perspective in its MAC layer that has become the root of many unfairness and performance issues. The most significant of them are unfairness due to location dependent bursty channel errors, uplink-downlink unfairness and performance anomaly of 802.11. Redefining fairness in terms of time shares, a way that suits best the multirate physical layer of 802.11 protocol, was a decisive step in designing a mechanism to address effectively the issues mentioned above.

The current installed base of 802.11 wireless networks that counts millions, deployed in homes, offices and hot-spots, designated the importance of a solution that is applicable to existing wireless networks as well as future ones. Utilization of existing mechanisms, like the CBWFQ one, properly supported by algorithms that deal with 802.11 fairness and performance problems, proved to be an efficient approximate to the re-defined fairness scheme, implementable at the network layer. The dynamic CBWFQ mechanism offers control over uplink TCP traffic, compensates nodes for losses due to channel errors and absence and deals with the performance reduction due to multiple transmission rates. It also provides a per node service differentiation scheme.

Deployment of our dynamic CBWFQ scheme was done by configuring a Linux Box to function as an access router. The CBWFQ scheme implementation was based on Hierarchical Token Bucket, a Linux technique for implementing real time packet forwarding policies, and HostAP, a driver for wireless cards. The experimental results proved the effectiveness of our mechanism in dealing with significant fairness and performance problems of 802.11 wireless networks.

Part of our future work is to combine the functionality of the dynamic CBWFQ mechanism with the enhanced DCF provided by the 802.11e protocol. The collaboration of these two mechanisms could result in a more efficient and robust service differentiation scheme. Furthermore we are interested in applying the notion of dynamic weights for sharing other network resources, such as the 802.11e contention window parameter. This approach could induce the required intelligence at the MAC layer of 802.11e that is necessary to improve fairness and performance and achieve service differentiation in the dynamic wireless environment.

Bibliography

[1] Y. Yuan, D. Gu, W. Arbaugh, and J. Zhang, "Achieving packet-level quality of service through scheduling in multirate wlans," *VTC IEEE*, 2004.

[2] V. Gambiroza, B.Sadeghi, and E. Knightly, "End-to-end performance and fairness in multihop wireless backhaul networks," *MobiCom'04*, Oct. 2004.

[3] C. Fragouli, M. Srivastava, and V. Sivaraman, "Controlled multimedia wireless link sharing via enhanced class-based queueing with channel state dependent packet scheduling," *IEEE INFOCOM'98*, March 1998.

[4] H.Zhu and G. Cao, "On improving service differentiation under bursty data traffic in wireless networks,"

[5] S. Lu, V. Bharghavan, and R. Srikant, "Fair scheduling in wireless packet networks," *IEEE/ACM Transactions on networking*, vol. 7, August 1999.

[6] T. Ng, I.Stoica, and H. Zhang, "Packet fair queueing algorithms for wireless networks with location dependent errors," *IEEE INFOCOM'98*, March 1998.

[7] S. Lu, T. Nandagopal, and R. Srikant, "Fair scheduling in wireless packet networks," *ACM MOBICOMM'97*, August 1997.

[8] S. Lu, T. Nandagopal, and V. Bharghavan, "A wireless fair service algorithm for packet cellular networks," *MOBICOM'98*, October 1998.

[9] P. Ramanathan and P. Agrawal, "Adapting packet fair queueing algorithms to wireless networks," *ACM MOBICOM*, 1998.

[10] IEEE Std. 802.11-1999, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, Reference number ISO/IEC 8802-11:1999(E), IEEE Std. 802.11, 1999.

[11] IEEE Std. 802.11b, Supplement to Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Higher-speed Physical Layer Extension in the 2.4Ghz Band , IEEE Std. 802.11b-1999, 1999.

[12] J. Zyren and A. Petrick, "Tutorial on basic link budget analysis," *Intersil Application Note*, 1998.

[13] T. Nandagopal, S. Lu, and V. Bharghavan, "A unified architecture for the design and evaluation of wireless fair queueing algorithms," *MOBICOM'99*, 1999.

[14] P. Bhagwat, P. Bhattacharya, A. Krishma, and S. Tripathi, "Enhancing throughput over wireless lans using channel state dependent packet scheduling," *IEEE INFOCOM'97*, April 1996.

[15] S. Pilosof, R. Ramjee, D. Raz, Y. Shavitt, and P. Sinha, "Understanding tcp fairness over wireless lan," *IEEE INFOCOM*, 2003.

[16] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, "Performance anomaly of 802.11b," *IEEE INFOCOM 2003*, 2003.

[17] S. Floyd and V. Jacobson, "Link-sharing and resource management models for packet networks," *IEEE/ACM*, vol. 3, August 1995.

[18] J.Pavon and S.Choi, "Link adaptation strategy for ieee 802.11 wlan via received signal strength measurement,"

[19] M. Devera, "Htb linux queuing discipline manual-user guide,"

[20] B. Mah, "An empirical model of http netwrok traffic," *IEEE*, 1997.

[21] I. Mahadevan and K. Sivalingam, "Quality of service in wireless networks based on differentiated services architecture,"

[22] M. Bottiglienco, K. Casetti, C.-F. Chiasserini, and M. Meo, "Short term fairness for tcp flows in 802.11b wlans,"

[23] L. Kalampoukas, A. Varma, and K. Ramakrishnan, "Explicit window adaptation: a method to enhance tcp performance," *IEEE/ACM Transactions on Networking*, vol. 10, 2002.

[24] S. Karandikar, S. Kalyanaraman, and P. Bagal, "Tcp rate control,"

[25] H. Aida, Y. Tobe, M. Saito, and H. Tokuda, "A software approach to channel-state dependent scheduling for wireless lans," *WOWMOM*, 2001.