The Wireless Hierarchical Token Bucket: a Channel Aware Scheduler for 802.11 Networks

R. G. Garroppo, S. Giordano, S. Lucetti, and E. Valori
University of Pisa
Department of Information Engineering
Via Diotisalvi 2, 56126 Pisa, Italy
{*r.garroppo, s.giordano, s.lucetti, e.valori*}*@iet.unipi.it*

Abstract

The paper proposes an architecture for a scheduling algorithm, to be integrated in IEEE 802.11 Access Points (AP), able to take into account, besides the transport service class required by the destination user, the channel quality experimented by the destination mobile STAtion (STA). The relevance of this topic is due to the observation that when one or more STAs experiment poor radio channel conditions, they increase the time spent to transmit a single packet, due to the retransmission of corrupted frames and the adoption of a transmission techniques at lower bit rate, leading to an inefficient use of the shared medium. These phenomena have as a consequence the worsening of the performance of all the STAs sharing the wireless medium independently of their radio channel conditions. The adoption of a scheduling algorithm able to manage information on channel quality permits to reduce these effects, not penalizing the STAs experimenting good channel condition and, as a consequence, their experimented throughput. As a result, only the STAs in bad channel conditions experiment a reduction of throughput. A prototype of an AP equipment implementing the proposed architecture is then presented; it has been obtained modifying the Hierarchical Token Bucket (HTB) and has been indicated as Wireless Hierarchical Token Bucket (WHTB).The performance of the presented prototype are experimentally evaluated and compared with those obtained with standard scheduling algorithm, which do not take into account information on channel quality.

1. Introduction

The IEEE 802.11 standard and its evolutions [1][2][3][4] are the most popular technologies of wireless networking for local area communications. The easiness of the setup of IEEE 802.11 systems has permitted their deployment in public hot-spots such as airports, hotels and conference facilities, as a preferred way to offer connectivity to nomadic users.

The wide success of this technology has driven the creation of new IEEE Task Groups and the design of new standards to satisfy the requirements of increasingly demanding users and applications. In particular, the Task Group 802.11e [5] is defining MAC mechanisms aimed at transport service differentiation, whereas the more recent 802.11n [6] is devoted to define physical layer enhancements to provide higher maximum data rates. At the moment, the Wireless LAN market is almost completely based on 802.11a/b/g devices and their deployments will continue to operate in the network market for the next few years.

The fundamental medium access mechanism used by the IEEE 802.11 family standards is called Distributed Coordination Function (DCF); see [7] for an overview on the IEEE 802.11 network architecture. An optional polling-based protocol, called Point Coordination Function (PCF), is also defined in the standard; however given its complexity it is seldom implemented in commercial devices. DCF is a contention based random access scheme, based on a Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol. After a collision event, the frame retransmission is managed according to binary exponential backoff rules. The basic scheme for frame transmission under the DCF operation mode is a two-way handshaking technique, characterized by the immediate transmission of a positive acknowledgement (ACK) frame by the destination station, upon successful reception of a frame transmitted by the sender station.

The sum of the idle periods between the transmission of the various frames, as well as the overhead time associated to the MAC and PHY layers headers, leads to a maximal achievable throughput for 802.11 networks far lower than the nominal data rate, also if no collisions occur. Further-

more, the 802.11 physical layer provides multiple transmission rates, which are chosen by the transmitter on the basis of the success or failure of previous frame transmissions, failures being caused either by collisions or scarce radio link quality. As an example, in the case of 802.11b devices, the transmission bit rate is reduced from the nominal 11 Mbps rate to 5.5, 2, or 1 Mbps, when a station detects repeated unsuccessful frame transmissions. As a result, a station experiencing poor radio link quality occupies the radio channel a longer time than it would otherwise, due to the retransmissions and the link adaptation.

As a side effect, the throughput experienced by other stations of an IEEE 802.11 cell decreases, because the radio channel availability is reduced. This phenomenon is known as the 802.11 anomaly [8]: the effective throughput of all the stations transmitting at higher rate is degraded below the level of the lowest one. The main reason for this anomaly is the CSMA/CA protocol, which guarantees an equal long term channel access probability to all stations. When one station occupies the channel for a long time because of its low bit rate, it penalizes all the other stations that use the higher rate.

The basic building block of an IEEE 802.11 system is a Basic Service Set (BSS), which is composed by an Access Point (AP) and multiple mobile stations (STAs) associated with the AP. In this framework, the AP bridges the wireless STAs with the Distribution System (DS), offering access to the rest of the wired network infrastructure. In the downlink direction (AP to STAs), the AP manages the traffic towards all the associated STAs, and having a single output interface, has to perform a scheduling on the traffic packets destined to them. The IEEE 802.11 standard does not define any specific scheduling algorithm; hence, currently available commercial products adopt the simple solution provided by First-In-First-Out (FIFO) paradigm, without any reordering and prioritization of the packets received by the DS.

This scheduling implementation, along with the retransmission policy and the link adaptation scheme mentioned above implies that degradation in the downlink quality towards one station affects the performance of all the other traffic managed by the AP in the same BSS. This effect is particular relevant in the downlink direction, because the effects of poor channel quality of one station in the uplink direction are concealed by the contention-based medium access. Furthermore, the use of a simple FIFO scheduling does not permit to offer traffic differentiation, which is a relevant topic considering the different requirements associated to the numerous services available nowadays on IP networks, such as Voice over IP, data transfer, e-mail and so on. In particular, the bottleneck role played by the AP for VoIP services and the relevance of transport service differentiation have been shown in [9] [10].

Given the current unavailability of 802.11 network interfaces implementing service differentiation at MAC layer, i.e. the draft proposed by IEEE 802.11e [11], an alternative solution is to introduce appropriate scheduling mechanisms in the AP implementing them as modifications to the device driver. This scheduling should be able to appropriately choose the packets to be forwarded to the MAC layer (which is contained in binary form in the network interface firmware), considering both the required transport service and the radio link quality experienced by the single STAs.

The main contribution of the paper is the definition of a scheduler able to differentiate the transport service offered by an IEEE 802.11 AP taking into account the radio link quality experimented by the different mobile stations. A similar approach is presented in [12], although the authors do not take into account the downlink transport service differentiation issue.

Introducing service differentiation at layers higher than MAC is clearly suboptimal, because of the limited available control over the subsequent queueing. Anyway, the approach proposed in this paper is relevant since it can be easily extended to integration in the MAC layer as soon as the IEEE 802.11e standard completes its definition and devices able to make this differentiation are made available.

Throughout the paper, the expression channel quality aware, or channel-aware, scheduler indicates the capacity of the scheduler to take into account information on the radio link quality experienced by the STAs. The proposed scheduler is based on the Hierachical Token Bucket (HTB) scheme, and has been called Wireless Hierarchical Token Bucket (WHTB).

The paper is organized as follows. Section II presents the general architecture of a channel-aware scheduler. Section III describes the main features of HTB, while Section IV presents the proposed channel quality aware WHTB scheduler. The performance analysis of WHTB is presented in Section VI using the experimental scenario described in Section V. Lastly, Section VII concludes the paper with final remarks and considerations.

2. Architecture of a Channel-Aware Scheduler

The aim of a channel-aware scheduler is to exploit information on the radio link quality experienced by the different users associated to the same AP in order to optimize the global goodput of the considered WLAN cell. The term goodput refers to the effective throughput perceived by the user at the application level, thus not considering all the retransmissions which may occur at data link layer, nor the actual data rate.

However, the actions taken to optimize the global goodput must consider also the required quality of service at IP level chosen by the users. To take into account both aspects,

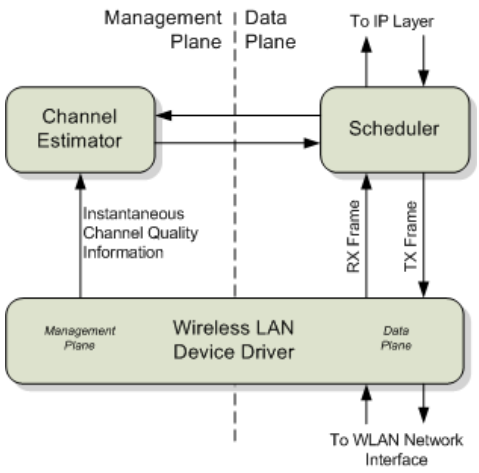

Figure 1. Architecture of a channel-aware scheduler

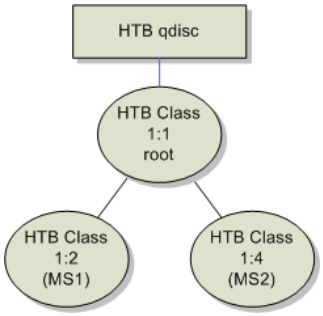

Figure 2. Example of classes defined in the HTB Scheduler

a possible architecture can be obtained modifying a scheduler used in wired networks, such as the HTB, in order to take advantage of the information on the radio link quality experienced by the users.

The reference channel-aware scheduler architecture is represented in Fig. 1. On the right hand side, the Data Plane is devoted to the transmission and reception of the frames; it is composed of the WLAN device driver, which performs the actual transmission and reception, and the scheduler, which is driven by link quality information provided by the Management Plane. The Management Plane is composed of the part of the WLAN device driver deputed to export the information on the channel quality towards each receiver, and of Channel Estimator module, which translate these information into values suitable to feed the scheduler.

As far as the scheduler itself is concerned, it can be either an algorithm designed specifically for wireless network or a scheduler developed for wired networks opportunely modified to take into account the information on wireless channel. In the proposed architecture, the adopted scheduler is the HTB.

3. Main features of HTB

This Section describes the main features of the HTB scheduler; in particular, the implementation available in the Linux Traffic Control (TC) framework has been considered, and will be referred when specifying implementation details. In the HTB scheduler, traffic classes form a tree structure. Fig. 2 shows a sample configuration with two classes: in particular, the two *leaf* classes, 1:2 and 1:4, are associated to two different destinations (indicated as MS1 and MS2), and both descend from *root* class (1:1), which is directly attached to the queuing discipline HTB.

Each class is configured with an average rate to be guaranteed (*rate* parameter) and a maximum rate which cannot be exceeded (*ceil* parameter). To fulfill such requirements, each class is controlled by an internal token bucket, after which the scheduler is named. The bucket is constantly filled with a number of tokens proportional to the *rate* that the class is allowed to transmit. On the other hand, the bucket loses a number of tokens proportionally to the amount of bytes transmitted, i.e. when packets belonging to the class are extracted from the scheduler's queue and transmitted to the lower layers of the protocol stack.

Since each class has two assigned rates (average and maximum) two token buckets are needed: the *rate* bucket contains the so-called *tokens*, while the *ceil* bucket is filled with the *ctokens* (*ceil*-*tokens*). Note that the adopetd HTB implementation interprets the *tockens* and *ctokens* as a measure of the *time* that a specific class can occupy the scheduler output line.

During the actual transmission of packets belonging to the different traffic classes, each of them can be, at a given time, in one out of three possible states, usually associated to three colors:

- *Green*: the class has sent less data than it is allowed to, so it can unconditionally send more data;
- *Yellow*: the class has exceeded *rate*, but not *ceil*, so it can borrow unused bandwidth, if available, by other classes;
- *Red*: the class has exceeded the *ceil*, so it cannot send data.

The principle of operation of an HTB scheduler can be shortly summarized, according to [13], as: when there are packets in the queues, each class should obtain at least its *rate*, but without exceeding its *ceil*.

In the paper, *rate* is assumed equal to *ceil*. Hence, at a given instant, each class can be in the *Green* o in the *Red* state: the state will be based only on the number of available *ctokens*. This assumption means that each class will obtain (at most) the configured *rate*, which in turns coincides with the maximum throughput.

The HTB scheduler grants the right to transmit to *green* classes only, according to a Deficit Round Robin (DRR) algorithm based on a *deficit* variable for each class. In detail, the scheduler extracts from the queue, one by one, packets belonging to a given class. Each time a packet is dequeued, the class *deficit* is decremented by a value equal to the packet length. Moreover, the number of *ctokens* in the class bucket is decremented by the ratio between the packet length and the allowed rate, to take into account the amount of time for which the class occupies the scheduler output line. The value of *ctokens* is then raised by a value of time equal to the time elapsed since the transmission of the previous packet of the same class, to take into account the constant refill of the bucket. In other words, indicating with t_2 the current time and t_1 the time of the last dequeue event:

$$ctokens\,(t_2) = ctokens\,(t_1) + (t_2 - t_1) - \frac{pkt\ length}{rate}$$

Indicating with R the capacity of the output link expressed in bps, any *rate* assigned to a class clearly satisfies the inequality $rate < R$; consequently, $\frac{pkt\ length}{rate} > \frac{pkt\ length}{R}$. This condition implies that, between consecutive transmissions of packets of the same class, the amount of *ctokens* of the class bucket decreases. Indeed, since transmissions are performed at the data rate R, the quantity $t_2 - t_1$ (added to the *ctokens* pool) is exactly equal to $\frac{pkt\ length}{R}$, which is lower than $\frac{pkt\ length}{rate}$ (subtracted by the *ctokens* pool).

Two events may cause the scheduler to switch to the next *green* class (if exists), suspending the service of the current one. They are:

- Expiration of the *deficit*. This event is associated to the DRR nature of the scheduler: after every packet transmission, *deficit* is decremented, and can become negative or zero. In such case, a value *quantum*, proportional to *rate* (by default $\frac{1}{10}$ of *rate*) is added to *deficit*, and the scheduler switches to the next *green* class, if any. If no *green* classes are present, the active one can continue sending its packets until it becomes *red* or one of the others becomes *green*.

- Bucket underflow: when the *ctokens* bucket empties, the scheduler gets aware that the class is exceeding its *ceil*, and consequently starts to serve the next *green* class. The value *ctokens* can assume values in the interval $[-cburst, cburst]$, where *cburst* is a parameter which determines the allowed peak rate. When the value of *ctokens* goes below *-cburst*, underflow happens, and the class becomes *red*; if it goes above *cburst*, it is upper-bounded to *cburst*, and the excess *ctokens* are discarded.

Note that bucket underflow has an higher priority than *deficit* expiration: if the bucket underflows, the class is stopped without taking into account *deficit*; viceversa, if *deficit* expires but all other classes are still *red*, the scheduler continues allowing the active class to transmit (simply adding a *quantum* value to the *deficit* pool).

Disabling the bucket underflow mechanism, the HTB scheduling algorithm reverts to a simple Deficit Round Robin. To obtain such behavior, it is sufficient to configure the HTB scheduler with a large value of *cburst*, so as to avoid underflow. When bucket underflow can not happen, the scheduler allows the transmission of *quantum* bytes for each class before switching to the next one, since all the classes are always *green*.

4. The Wireless HTB (WHTB)

HTB scheduler alone is not able to perform good scheduling on a wireless network, since it is based on a deterministic knowledge of the capacity of the output link. In order to include the information on the wireless channel in the HTB, the architecture proposed in Fig. 1 has been taken into account.

The following principle is at the basis of the scheduler modification: a wireless network interface needs, on average, more time to send packets to a destination experiencing poor channel quality than to a destination experiencing good channel quality. Transmission time increases not because packets are actually longer, but because a lower data rate is adopted when radio propagation conditions are unfavourable; furthermore, it may happen that multiple frame transmission attempts have to performed prior the correct reception at the destination, further increasing the effective transmission time. Such combination of events (rate adaptation and retransmissions) produce a reduction of the transmission capacity towards such destination, which is strongly correlated with the value of the Signal-to-Noise Ratio (SNR).

The macroscopic effect of poor channel quality towards a destination can then be summarized as a reduction of the effective transmission capacity; such reduction will have to be taken into account in the scheduling process. In our approach, this is considered defining an artificial packet length, indicated as *stretched*, as if an actual increase in the number of bits of the packet has occurred (the capacity of the output link being constant and equal to the nominal one).

In this architecture, a key role is played by the module that relates the channel quality towards a particular STA

to an estimated capacity of the link towards such destination. This module is called the Wireless Channel Monitor (WChMon), and feeds the WHTB scheduler with indications about the effective exploitation of the nominal capacity towards each associated STA.

The WHTB then calculates the *stretched* packet length dividing the actual packet length by a corrective parameter, indicated as the *relative throughput*, being defined as the ratio between the estimated effective throughput and the maximum (nominal) one.

4.1. The Wireless Channel Monitor

The Wireless Channel Monitor (WChMon) used in the WHTB has been developed at the Washington University in St. Louis and is described in [14].

In most currently available wireless chipsets an Automatic Gain Control (AGC) unit monitors the signal condition and adapts the RF circuit of the card. This information is also used to compute three values indicating the signal level, the noise level and the signal quality (other designs only provide one value which indicates the signal level). These values are stored in the Parameter Storage Area (PSA) of the wireless card and can be read by a device driver which export them in the kernel memory. The reported values are related to the signal and noise levels in dB (allowing to estimate a SNR indicator), and are computed for each received packet.

SNR and bit error rate are strongly correlated, so there is also a strong correlation between the SNR reported by the card and the probability of receiving a frame with errors (*Frame Error Rate*, *FER*). As a consequence, channel quality degradation is involved in the reduction of the attainable throughput on the channel.

An ideal estimator should take into account that, the event of successful transmission of a packet from one wireless host to another (being both STAs or one AP and one STA), both the data frame and the corresponding acknowledgement frame have to be received correctly, i.e. there must be no errors in both directions of communication. As a result, actual link quality in a transmission from the AP to a STA is a combination of the data frame SNR (at the STA) and the ACK SNR (measured at the AP); an accurate channel estimation, by means of SNR reporting, is possible only considering both of them.

Such complete knowledge is rather difficult to be achieved, because a means for transferring SNR measurements from one side to the other of the communication link should be introduced in the system; instead, a simplified approach has been considered. The adopted WChMon starts from the hypothesis of channel **symmetry**, assuming that the SNR seen by the AP is analogous to the SNR seen by the receiver located on the STA. Therefore the AP, which only knows the SNR associated to the received frames from a given STA, can estimate, with fairly good approximation, the SNR perceived at the STA, and consequently the expected goodput towards that STA.

In this kind of approach, the driver, after computing the SNR associated to the received frame transmitted by a STA, transfers this information to the WChMon which, by means of a table, estimates the goodput the AP expects to reach towards that mobile station. Each time the scheduler needs to be informed about the channel available capacity towards a certain destination, it invokes the WChMon specifying the STA address.

A further element of complexity is represented by network cards and corresponding drivers; although all manufacturers have to conform their products to 802.11b standard, the current implementations may differ (and actually does) in several ways, both in the hardware and in the software [15]. As a consequence, receiver sensitivity may change, so different cards may have different transmission capacity for a given SNR. In conclusion, an initial *calibration* of WChMon is necessary; the product of the calibration process is the table which relates the SNR to the available throughput towards a specific STA.

4.2. Calibration of WChMon

As highlighted before, the WChMon module has to be calibrated in order to provide significant estimates of the channel capacity to the scheduler. The *WChMon Signal-to-Goodput Mapper* (*WChMonSigMap*) [14] is used for calibration: it is a Java application for Linux whose task is to build a table on the basis of experimental data. The table associates each SNR value to the corresponding capacity. Data is obtained with two hosts A and B which communicate through a wireless link. An UDP flow is transmitted from A to B to attain a complete saturation of the link capacity: in this way, the MAC layer at A has always a frame to transmit. The *WChMonSigMap*, installed on B, measures the time τ necessary to receive a fixed number n of packets (e.g. 100) and, joining this measure with information about their length L, provides an estimate of the available transmission capacity:

$$\hat{R} = \frac{n \cdot L}{\tau}$$

Then, the *WChMonSigMap* combines this result with the SNR level reported by the driver for the relative position of A and B, obtaining a couple of values (SNR, goodput) every n received packets; several measurement runs are performed for each SNR value, and the results averaged to obtain a single couple of values. Lastly, all these couples are included in a table, which constitute the calibration result.

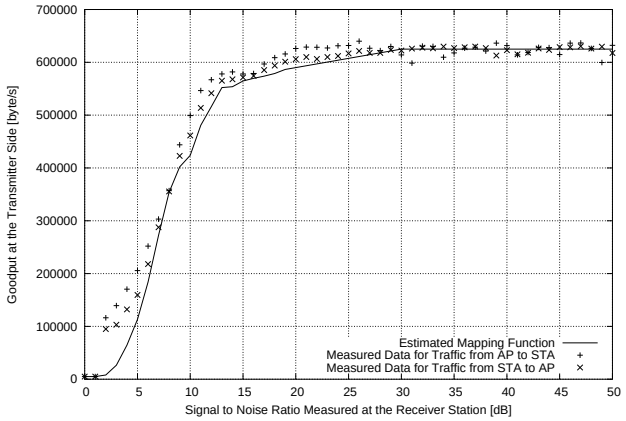

Figure 3. Calibration results obtained using Lucent *Orinoco* 802.11b NIC and *wvlan_cs* driver

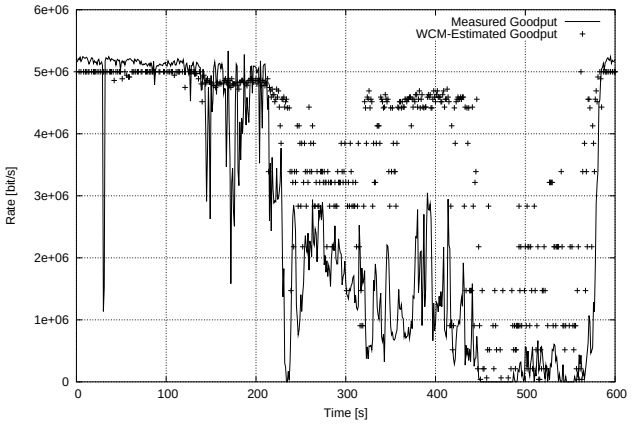

Figure 4. Comparison between measured and WChMon-estimated goodput

4.3. Implementation issues and calibration results

Because of asymmetries in the wireless transmission channel, when packets are sent from the AP to a MS, calibration gives results that are slightly different from those obtained when packets are sent in the opposite direction.

Indeed, it should be taken into account that a successful transmission of a packet is made up of two phases: transmitting a frame and receiving the ACK. When the channel is not symmetric, it may happen that the data frame experiences a fairly good SNR, while the ACK has a higher probability of being currupted due to a higher noise level at the frame transmitter, or viceversa.

To verify channel simmetry, and to mitigate possible differences, calibration in both directions was done and the table to be used for scheduling was obtained by a final extrapolation and smoothing; Fig. 3 shows the calibration results obtained using Lucent Orinoco PCMCIA cards and *wvlan_cs* driver [15].

During normal operations, WChMon applies a first order IIR low pass filter to the SNR values reported by the driver. Indeed, instant SNR reported by the driver manifests a high degree of variability, since it refers to a single received frame and is consequently affected by short-term signal power fluctuations. The coefficients of the filter have been chosen to adequately smooth short-term SNR variations and concurrently allow to follow the average behavior. Actually, the filtering itself is applied to the estimated goodput, according to the following expression:

$$\hat{R}(t) = \frac{3}{4} \cdot \hat{R}(t-1) + \frac{1}{4} \cdot \hat{R}[SNR(t)]$$

where $\hat{R}[SNR(t)]$ is the instantaneously estimated goodput, read from calibration table in the line corresponding to $SNR(t)$.

Figure 4 shows that the WChMon, once calibrated, provides a fairly good approximation of the attainable goodput towards a destination, on the sole basis of the value of signal-to-noise ratio of the received frames from that destination.

4.4. The developed WHTB system

The WHTB has been realized integrating the WChMon in the architecture in Fig. 1, and modifying the HTB accordingly. The HTB code modification can be synthesized into two fundamental operations:

1. Each packet's length is converted into a stretched length:

$$stretched = \frac{length}{\hat{g}(t_0)} = \frac{length}{\frac{\hat{R}(t_0)}{R_{MAX}}} = length \cdot \frac{R_{MAX}}{\hat{R}(t_0)}$$

   where $\hat{R}(t_0)$ is the goodput, estimated by the WChMon at time instant t_0 of the scheduler choice of packet dequeue, towards the proper destination, while R_{MAX} is the attainable goodput towards the same destination when channel quality is maximum.

2. The value *stretched* is upper bounded, to prevent that very low values of relative goodput produce an artificial length so large that *deficit* goes under zero (which is a nonsense). The upper bound to *stretched* is set to the *quantum*.

Summarizing, the two operations can be expressed by means of the single following expression:

$$stretched = \min(length \cdot \frac{R_{MAX}}{\hat{R}(t_0)}, quantum)$$

5. Parameters of experimental analysis

The experimental testbed is depicted in Fig. 5; all the devices are PC based linux boxes. In more detail, the functions and characteristics of each device are the following:

- *Granpa*: Pentium III class desktop PC; traffic generator.
- *Edison*: Pentium III class desktop PC; Access Point and IP router functionalities.
- *Escher*: Pentium II class laptop; mobile station MS1, constantly kept in good channel conditions.
- *Russell*: Pentium II class laptop; mobile station MS2, moved around to have variable channel conditions towards the AP.

All the devices run a 2.4.24 Linux kernel; both the MSs and Edison run a modified version of the kernel, which includes the HTB patches for the Linux Traffic Control [16]. The HTB, and its modification WHTB, have then been used to define some bandwidth shares between the two flows. In particular, 1 to 4 and 1 to 1 bandwidth shares have been adopted as test cases in the measurement campaigns. In order to create two data flows towards the two MSs, MGEN [17] traffic generation program has been activated on Granpa. Most of the experiments that have been carried out have been performed slowly moving away one of the MSs (MS2) from tha AP, and jointly measuring the channel quality perceived and the effective goodput received by each MS.

In this context, an interesting issue is related to the expiry of the ARP cache at the stations, and expecially at Edison. Indeed, as long as the MS2 experiments a sufficiently good channel condition towards the AP, ARP Request and Reply messages are periodically exchanged between the two peers to refresh the correspondence between IP addresses and MAC ones. On the other hand, when ARP exchanges do not succeed because of very poor channel conditions, Edison stucks into performing ARP requests to discover MS2 hardware address, and the measurement is compromised. To overcome such behavior, static ARP entries have been configured at Edison; consequently, the only traffic travelling in the network is the one generated by the traffic generator.

MGEN at Granpa has been configured to generate two UDP flows, each one addressed to one of the two MSs. Another instance of MGEN is running, configured as receiver, on each MS; this is necessary in order to have a receiving process listening at the UDP port towards which MGEN at Granpa generates its flows. In the opposite case, each MS would generate a number of ICMP Port Unreachable messages back to Granpa, altering the results of the measure.

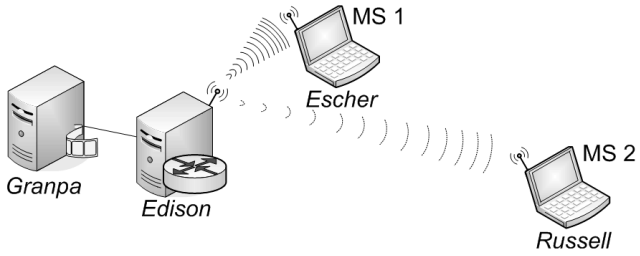

Figure 5. Experimental testbed

The parameters of the traffic generation process at Granpa are as follows; the generated flows are CBR, with costant both inter-departure times and packet size. In detail, inter-departure time has been set to obtain 750 packets per second, whereas the packet size has been set to 1024 byte. The resulting rate is about 6 Mbps for each flow; hence, the overall rate offered to Edison is about 12 Mbps, well above the about 5 Mbps which can be handled in saturation conditions for this packet size, as highlighted by both analytical, simulative and experimental analysis in [18].

The duration of each measurement run has been set to 81.92 seconds. This unusual value (not a round number) has been chosen in order to simplify the tasks, at the end of each measurement session, of the post processing procedure necessary to evaluate the achieved goodput. Indeed, in such period of time, the number of packets received by each MS is numerically equal to ten times the average goodput expressed in Kbps. The task to evaluate the average goodput is then converted in simply counting the received packets and multiplying such value by one hundred. The count of the number of received packets is automatically performed by the MGEN receiver instances at the two MSs.

In addition to the mean goodput received by each MS, another important performance parameter has been considered in our measurement. Such parameter is the percentage of the radio resources occupied in the data transmission attempts towards each STA. The evaluation of such percentage has been performed according to the following considerations. Since MS1 always experiments good channel quality, it is possible to suppose that each transmission attempt towards MS1 is successful. The percentage of the channel occupancy for the transmissions addressed to MS1 is then equal to the ratio between the mean goodput received by MS1 and the saturation goodput of the system in ideal channel conditions (both MS1 and MS2 with good channel quality). On the other hand, the difference between the saturation goodput and the goodput received by MS1 represents the transmission capacity used to deliver packets to MS2. Note that this value is typically different from the achieved gooput because of retransmissions which occur as soon as MS2 moves away from the AP.

The value of the percentage of the effective utilized resources indicates whether the scheduler is efficient in avoiding MS2 to occupy the medium with many unsuccessful retransmission attempts, at the expenses of MS1. Further interpretations of the resource occupancy percentage will be given in the next Section, where some measurement results will be presented and analysed.

6. Performance analysis of WHTB

In order to evaluate the benefit obtainable by the adoption of the WHTB, a number of experimental measurements have been carried out. In this Section, the results related to two different choices of the nominal bandwidth shares assigned to each MS are presented. The first presented case is constituted by an uneven share of the resources between the two stations, which privileges MS1 in a 4 to 1 ratio. The second case is related to an even assignment of the nominal transmission capacity.

Measurements have been repeated using three other standard schedulers, so as to confirm the different behaviors and verify the effectiveness of the proposed scheduling algorithm. In detail, the standard schedulers are the CBQ (which stands for Class Based Queueing [19]), the DRR and the plain HTB.

6.1. Bandwidth share 1:4

In this test case, the IP scheduler at the AP has been configured to assign 80% of the available resources to MS1 and the remaining 20% to MS2. Since the maximum effective goodput which can be achieved using 1024 byte long packets is about 5 Mbps, the schedulers have been configured to reserve 4 Mbps to the class associated to the transmission queue towards MS1 and 1 Mbps to the class associated to MS2.

Each set of measurements has been performed keeping MS1 close to AP with good channel quality, and placing MS2 in five different positions, characterized by different values of channel quality. The five positions have been chosen on a heuristic basis, ranging from Good (no retransmissions) to Out-of-Range (no successful transmission) through the intermediate states Medium, Bad and Very Bad.

The measurement results obtained according to the procedure described in the previous subsection are reported in Table 1 and Figure 6. Table 1 collects the results related to the adoption of three standard scheduling algorithms (CBQ, HTB and DRR) as well as the proposed WHTB. For each choice of the scheduler, the received goodput for the two MSs has been reported. The results highlight the different behavior of the four scheduling algorithms as far as the isolation of the two flows is concerned. In particular, the

MS2 Position	MS	CBQ	HTB	DRR	WHTB
Good	1	4037	4061	4054	3937
	2	1017	1011	1025	991
Medium	1	3404	3966	3276	3344
	2	859	609	828	788
Bad	1	2621	3544	2161	3960
	2	662	503	546	469
Very Bad	1	1996	2105	1449	3401
	2	497	255	342	160
Out-of-Range	1	692	1205	680	3226
	2	0	0	0	0

Table 1. Performance comparison among the different schedulers: Mean Received Goodput in Kbps with 1:4 bandwidth share

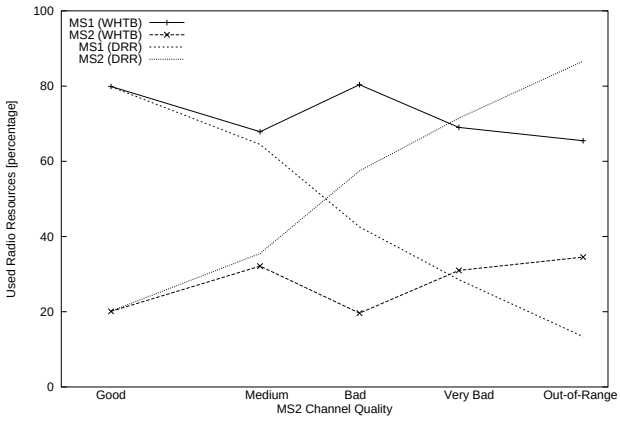

Figure 6. Comparison between the radio resources used to transmit to MS1 and MS2 with DRR and WHTB schedulers - Bandwidth share 4:1

three classical schedulers fail to fulfill the bandwidth reservation associated to MS1 when MS2 starts to experiment bad channel conditions. In detail, the HTB scheduler performs slightly better than the other two, still not managing to deliver the reqested 4 Mbps in any configuration except the ideal one (both MS1 and MS2 in the Good channel quality state).

On the contrary, the WHTB scheduler allows to deliver well above 3 Mbps to MS1 regardless of the position of MS2. Although the received goodput values are below the requested 4 Mbps, they are much higher than those achieved by the other schedulers, highlighting the goodness of the proposed approach. Going into further detail, the analysis of Table 1 allows to derive some interesting observations for the various positions of MS2:

- *Good* – All the schedulers behave correctly; the available radio resources are enough to fulfill the reservation requests, and each scheduler manages to assign 4 Mbps to MS1 and 1 Mbps to MS2
- *Medium* – The plain HTB (with the default *cburst* value) still manages to deliver 4 Mbps to MS1; the others, including the WHTB, decrease the delivered goodput towards both MS1 and MS2, strisctly adhering to the 4 to 1 bandwidth share.
- *Bad, Very Bad, Out-of-Range* – The WHTB scheduler outperforms the other three in all these working conditions. The CBQ and DRR schedulers behave almost the same, the MS2 position severely influencing the gooput towards MS1; also, the HTB performs only slightly better than these two. On the contrary, the WHTB keeps the goodput delivered to MS1 always over 3.2 Mbps, irrespective of the position of MS2. The sensible increase of the network utilization adopting WHTB is particularly relevant when considering the case of MS2 in the Very Bad or Out-of-Range positions. Indeed, the three reference schedulers severely degrade the aggregate goodput of the cell to values well below 1 Mbps, whereas WHTB keeps its value to about 3.5 Mbps, allowing MS1 to experiment transmission conditions only slightly worse than in the ideal case.

A major insight on the effects of the adoption of the WHTB scheduler is given by Fig. 6, where the radio transmission resources utilized by each one of the two classes for the WHTB and DRR schedulers are plotted. Indeed, when both MS1 and MS2 experiment a Good radio channel quality, the used resources share is equal to the received goodput share (80% and 20% in the particular case). As long as MS2 changes its position to Medium, Bad and so on, with the DRR an increasing amount of resources is occupied by the transmission towards MS2 (evaluated at Data Link Layer) in order to deliver the requested amount of data (at IP Layer). This is due to the increasing number of layer 2 retransmissions, as well as to the less efficient modulation adopted, needed to successfully transfer a single IP packet to MS2. Such behavior is unsuccessful in two ways: firstly, it does not manage to handle the requested goodput, and secondly, it heavily reduces the resources left available to MS1, which consequently experiments a reduced available transmission capacity which, in turns, decreases the goodput received by MS1 itself. Analogous results have been measured for HTB and CBQ schedulers, although they have not been reported here for sake of simplicity.

MS2 Position	MS	CBQ	HTB	DRR	WHTB
Good	1	2545	2534	2546	2511
	2	2545	2512	2543	2514
Medium	1	2000	1925	2038	2061
	2	2000	1855	2037	2028
Bad	1	1462	1492	1471	2107
	2	1460	1153	1469	956
Very Bad	1	714	778	573	1825
	2	676	369	562	313
Out-of-Range	1	198	199	201	1878
	2	0	0	0	0

Table 2. Performance comparison among the different schedulers: Mean Received Goodput in Kbps with 1:1 bandwidth share

The WHTB, on the other hand, has the effect of limiting the resources occupied by the transmission towards MS2, which do not exceed 35% in the worst condition. This behavior, although furtherly reducing the goodput received by MS2, leaves the medium free enough for MS1 to receive almost all its assigned bandwidth, with a noticeable increase of the exploitation of the wireless medium.

6.2. Bandwidth share 1:1

This test case is equivalent to the previous one; the only difference is the bandwidth share between the two MSs, which has been set to 1-to-1. Each MS has a 2.5 Mbps reservation, equal to the 50% of the available transmission capacity in ideal conditions for the selected packet size. Table 2 collects the measurement results obtained in this scenario. Like the previous case, the CBQ, DRR and HTB operate to try and maintain the imposed ratio between the two assigned bandwidth. Consequently, the goodput received by MS1 degrades as long as MS2 suffers an increasing number of losses and retransmissions. On the other hand, the WHTB reduces the influence of MS2 position on the goodput received by MS1, limiting the goodput reduction

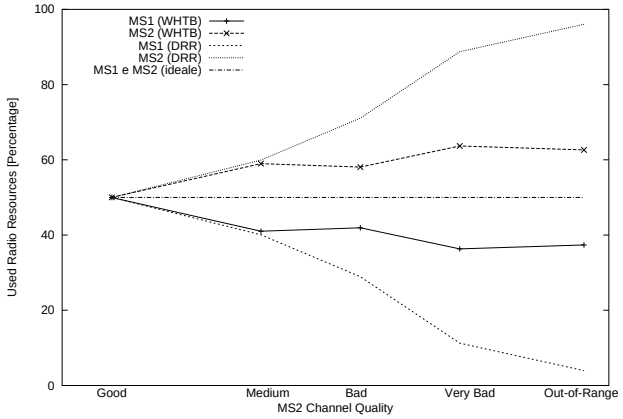

Figure 7. Comparison between the radio resources used to transmit to MS1 and MS2 with DRR and WHTB schedulers

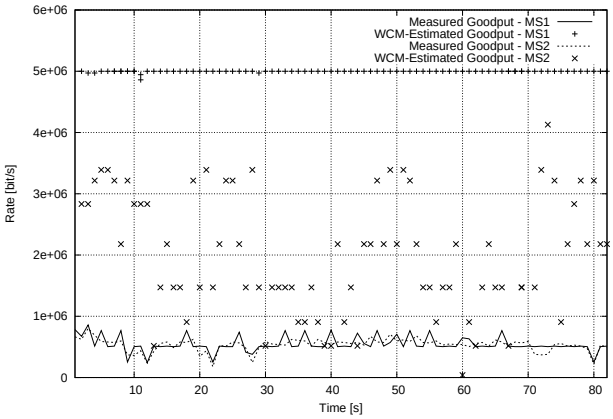

Figure 8. DRR behavior when MS2 has a very bad channel quality

to about 25% versus the over 90% of the other schedulers.

Again, Fig. 7 highlights the different consumption of transmission resources between the WHTB and DRR schedulers. The latter allows the transmission towards MS2 to monopolize the use of the radio resources; indeed, the AP uses only 4 to 30% of the available bandwidth to transmit packets to MS1. The WHTB scheduler allows, on the contrary, to limit the unfairness between the resources used by each class, with a maximum unevenness of 60% to 40%, regardless of the position of MS2 among the Medium, Bad, Very Bad or Out-of-Range conditions. Although it does not guarantee a complete decoupling between the two flows, the WHTB represents a first enhancement with respect to the standard schedulers, since it starts taking into account the effective probability that a transmitted frame is successfully received by the destination host.

To conclude the analysis of the experimental results, Figg. 8 and 9 are reported. The former reports the received goodput of MS1 and MS2, when MS2 is in the Very Bad position, adopting the DRR scheduler. Although the WChMon module is able to detect the different channel quality perceived by each MS, the scheduler does not take into account such information and transmit the same amount of data to both the MSs. On the contrary, as shown in Fig.'9, the WHTB scheduler takes advantage of the available channel information and automatically tunes the shares between the two classes in order to avoid to penalize MS1.

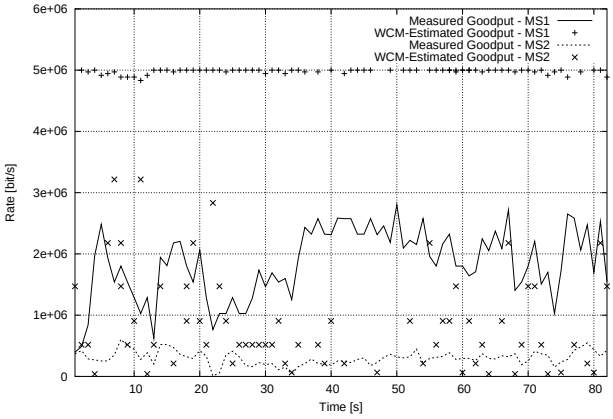

Figure 9. WHTB behavior when MS2 has a very bad channel quality

6.3. Implementation issues of WHTB

The main open issue of WHTB is channel estimation, which is not very accurate. Defects in the behavior of

WHTB are caused mainly by WChMon, as it is explained below with more details.

• *WChMon with medium channel quality* - When one of the two mobile stations is located in a medium-quality area, WChMon overestimates quality and assigns to this station a good value in the throughput mapping; therefore, as for traditional schedulers, the average goodput of the MS in good position decreases as much as the one in medium position. This is mainly related to a WChMon calibration issue, due to the asymmetry of transmission channel. Alternatives methods for actual goodput estimation, based on reception of ACK frames are currently being investigated to reduce this problem.

• *WChMon with very bad channel quality* - When one of the two mobile stations is located near the border of the AP coverage area, two different problems affect the correct behavior of the wireless network:

- Some packets are lost because the number of retries exceeds *Retry_Limit*; WChMon is not designed to handle this case, and hence produces a less accurate goodput estimation in such conditions, worsening the overall performance.
- The WHTB underlying DRR algorithm implies that, each time the *deficit* of a class expires, at least one packet of the other class is transmitted, because the check on *deficit* expiration is performed only for the following packets. Indeed, the scheduler assumes that at least one packet is necessary to make *deficit* expire. In the specific case discussed here, DRR introduces an upper bound to the ratio between data transmitted to MS1 and data sent to MS2, limiting the isolation of the two flows. A possible solution to be implemented is to allow a class to transmit its first packet only if its relative goodput is above a specified threshold: otherwise, the class has to miss one or more transmitting oppurtunities until its estimated channel capacity increases, or a counter expires.

7. Conclusions

To overcome the performance anomaly of the 802.11 system, evidenced firstly in [8], and to achieve a fixed bandwidth share between the users receiving downlink traffic, an appropriate scheduling algorithm should be implemented in the AP. In particular, the scheduling algorithm should be able to consider both the information on radio channel condition of the different STAs and the transport service class required by the destination user.

The architecture of the channel-aware scheduler described in this paper, allows to cope with these issues. To test the effectiveness of the proposed architecture, an experimental approach has been adopted. This choice has been mainly driven by the difficulty to consider, in a simulation tool, all the system parameters when dealing with wireless network scenario. Hence, a first prototype integrating the proposed architecture has been developed extending the classical HTB algorithm. The result is the definition of the WHTB architecture, whose performance are experimentally analyzed and compared with those obtained by means of classical scheduling algorithms, such as DRR, CBQ and HTB, defined for wired network scenario. The performance analysis has highlighted the efficiency of the WHTB; the experimental results show that classical scheduling are unable to maintain the assigned bandwidth shares to the stations. In particular, the worsening of radio channel quality of one or more STA negatively affects also the throughput towards STAs in good channel condition, as pointed out by the results collected in Tables 1 and 2. On the contrary, the same tables point out that the proposed WHTB permits to not penalize the STAs experimenting good channel condition and, as a consequence, their experimented throughput, independently of the channel conditions of other STAs.

The analysis carried out on the experimental results has pointed out that this result is due to the capacity of the proposed scheduling algorithm to maintain almost unvaried the channel occupancy time of each STA with respect to the assigned bandwidth shares, independently of its channel condition. On the contrary, when scheduling unable to take into account information on STA channel conditions are used, the STAs in bad channel condition increase their channel occupancy time to the detriment of the STAs in good channel condition: indeed, this phenomenon represents the basis of the 802.11 performance anomaly.

However, the performance analysis of the proposed WHTB has highlighted some weaknesses of the developed prototype that should be further improved. In particular, the main weakness is represented by the WChMon module which, in this first analysis, is rather simple and have a few drawbacks, mainly related to the calibration procedure required by the adopted module. Alternative Wireless Channel Monitor algorithm are currently under investigation and will be integrated in the scheduler to improve its performance and robustness.

8. *

Acknowledgment

This work has been sponsored by the Italian FIRB research program VICOM. The authors wishes to thank Dr. Giuseppe Risi for his help and support in setting up the experimental scenario.

References

[1] IEEE Standards for Information Technology – Telecommunications and Information Exchange between Systems – Local and Metropolitan Area Network – Specific Requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification, 1999

[2] IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications – Amendment 1: High-speed Physical Layer in the 5 GHz band, 1999

[3] IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications – Amendment 2: Higher-speed Physical Layer (PHY) extension in the 2.4 GHz band, 1999

[4] IEEE Standard for Information technology – Telecommunications and information exchange between systems - - Local and metropolitan area networks – Specific requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications – Amendment 4: Further Higher-Speed Physical Layer Extension in the 2.4 GHz Band, 2003

[5] `http://grouper.ieee.org/groups/802/11/Reports/tge_update.htm`

[6] `http://grouper.ieee.org/groups/802/11/Reports/tgn_update.htm`

[7] G. Bianchi, *Performance Analysis of the IEEE 802.11 Distributed Coordination Function* IEEE Journal on Selected Areas in Communications, Vol. 18, N. 3, pp. 535-547, March 2000

[8] M. Heusse, F. Rousseau, G. Berger-Sabbatel, A. Duda, *Performance Anomaly of 802.11b*, Proc. of IEEE Infocom 2003, San Francisco, April 2003

[9] D. P. Hole, F. A. Tobagi *Capacity of an IEEE 802.11b Wireless LAN supporting VoIP* Proc. of IEEE ICC 2004, Paris, June 2004

[10] Rosario G. Garroppo, Stefano Giordano, Stefano Lucetti *Admission Region for Multimedia Services in IEEE 802.11e Systems* Proc. Networks 2004, vol. 1, pp. 411-416, Wien 2004

[11] IEEE Std 802.11e/D6.0, Draft Supplement to IEEE standard for Telecommunications and Information exchange between systems - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Medium Access Control (MAC) enhancements for Quality of Service (QoS), November 2003

[12] Marc Portoles, Zhun Zhong, and Sunghyun Choi *IEEE 802.11 Downlink Traffic Shaping Scheme For Multi-User Service Enhancement* Proc. IEEE PIMRC'03, Beijing, China, Sept. 7-10, 2003

[13] Martin Devera, *Hierarchical token bucket theory*, May 2002, `http://luxik.cdi.cz/~devik/qos/htb/manual/theory.htm`

[14] Lars Wischhof, John W. Lockwood, *Packet Scheduling for Link-Sharing and Quality of Service Support in Wireless Local Area Networks*, Technical Report WUCS-01-35, Applied Research Laboratory, Washington University in St. Louis, November 2001

[15] Jean Tourrilhes, *Wireless LAN resources for Linux*, 1996-2003, `http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/`

[16] Martin Devera, *HTB Linux queuing discipline manual - user guide*, May 2002, `http://luxik.cdi.cz/~devik/qos/htb/manual/userg.htm`

[17] Brian Adamson, Hal Greenwald, *MGEN User's and Reference Guide*, 2004, `http://mgen.pf.itd.nrl.navy.mil/mgen.html`

[18] Rosario G. Garroppo, Stefano Giordano, Stefano Lucetti, Franco Russo, *IEEE 802.11b Performance Evaluation: Convergence of Theoretical, Simulation and Experimental Results*, Proc. Networks 2004, vol. 1, pp. 405-410, Wien 2004

[19] S. Floyd, V. Jacobson, *Link-sharing and Resource Management Models for Packet Networks*, IEEE/ACM Transactions on Networking, Vol. 3 No. 4, pp. 365-386, August 1995