IEEE P802.11
Wireless LANs

Enhancing IEEE 802.11 performance with slow CW decrease

Date: November 11, 2002

Authors:

Imad Aad
INRIA Rhône-Alpes
Phone: +33 4 76 61 54 35
e-Mail: Imad.Aad@inrialpes.fr

Qiang Ni
INRIA Sophia Antipolis
Phone: +33 4 92 38 79 59
e-Mail: Qiang.Ni@sophia.inria.fr

Claude Castelluccia
INRIA Rhône-Alpes
Phone: +33 4 76 61 52 15
e-Mail: Claude.Castelluccia@inrialpes.fr

Thierry Turletti
INRIA Sophia Antipolis
Phone: +33 4 92 38 78 79
e-Mail: Thierry.Turletti@sophia.inria.fr

Abstract

In this paper we study slow contention window decrease schemes for IEEE 802.11 and 802.11e. We propose simple functions to decrease the contention window slowly after each successful transmission instead of resetting it to CW_{min}. Simulations show that these mechanisms achieve considerable enhancements in congested environments over the legacy standard and equivalent performance in non-congested ones. Moreover, the complexity of slow contention window decrease scheme remains similar to DCF and EDCF schemes, enabling the design of cheap implementation.

1. Introduction

Wireless access networks are experiencing a huge success similar to that of the deployment of the Internet a decade ago. Wireless devices are used almost everywhere to provide cheap, mobile and easy to deploy networks, with or without access to wired infrastructures like the Internet. Wireless access networks can be grouped into two categories: centralized or distributed. Centralized architectures are mainly controlled by a coordinator (e.g. Access Point (AP)) that grants access to the wireless nodes in its area in a contention free manner. On the other hand, distributed architectures have no central coordinators: all nodes contend to access the channel using a distributed function. The efficiency of this function can be measured by the throughput and latency it achieves.

IEEE 802.11 is the most deployed wireless local area network (LAN) standard nowadays. It supports two access functions described in the next section, one is centralized, the other is distributed. The distributed function is based on carrier sense multiple access (CSMA) with collision avoidance (CA). Using CSMA/CA, each node defers its transmission to a random time in the future and senses the channel before trying to transmit. Upon each collision the node increases the bound of the random defering time, called contention window (CW). Increasing the CW reduces the risks of further collisions, assuming the number of contending nodes is high. Upon each successful transmission, a node resets its CW (to a PHY dependent value, CW_{min}) and contends again with low CW values.

Our work in this paper aims to enhance this last point: Upon a successful transmission a wireless node resets its CW, therefore it takes the risk of experiencing the same collisions and retransmissions until it reaches high CW values again, wasting time and bandwidth. Assuming the number of contending terminals changes slowly, this risk is likely to be high. We propose slow CW decrease functions and compare them to the current standard in different simulation scenarios. Simulations show that these functions outperform the legacy standard in terms of throughput.

The next section shows the motivation of our work. Section 3 introduces the approach of multiplicative CW decrease with several simulation scenarios. It also evaluates CW decrease functions using the throughput gain and the throughput settling time metrics. Section 4 briefly shows linear CW decrease performance. In section 5 an adaptive slow CW decrease scheme is proposed and in section 6 we conclude this paper.

2. Motivations

In a distributed 802.11 wireless LAN environment, a node has no knowledge of the number of contending terminals. However, it adapts its CW to the current congestion level by doubling its CW upon each collision, and resetting it upon each successful transmission. Doubling the CW assumes a higher congestion level therefore the need to increase the CW. When a node increases its CW, it reduces the chances of simultaneous transmissions from other nodes, at the cost of more backoff overhead. This reduces collisions and the corresponding retransmission times. When a node succeeds to transmit a packet, it assumes the congestion level dropped, therefore resets its CW to CW_{min}.

However, when a node succeeds to transmit a packet at a given CW_i, this does not correspond to a congestion level decrease, but to a convenient CW value. Therefore the CW value must be kept the same as long as the congestion level remains the same. Else, namely by resetting the CW, a node takes the risk of experiencing the same collisions and retransmissions "from scratch" until it reaches convenient (high) CW values again, wasting time and bandwidth. To adapt the CW to eventual congestion level drops, we should consider decreasing the CW upon successful transmissions. However, since congestion level is not likely to drop suddenly, we should consider slow CW decrease functions. Intuitively, the advantage of slow CW decrease functions is more collision avoidance during congestion, leading to less collisions and retransmissions, increasing the throughput and decreasing delays. The disadvantage is keeping high CW values after congestion level drops, increasing the overhead and decreasing the throughput. The slow CW decrease scheme is a tradeoff between wasting some backoff time and risking a collision followed by the whole packet transmission. In the following sections we study several slow CW decrease functions and evaluate their performance by comparing them to the existing standard.

The smooth CW decrease was first introduced in [1], among several other extensions to CSMA and MACA (like backoff copying and per-flow backoff counters). The main idea was to increase the CW at each collision by multiplying it by 1.5, and to linearly decrease it (-1) at each successful frame transmission. The approach was called MILD (multiplicative increase, linear decrease), and did not explore the effect of other decrease (or increase) factors on efficiency. Furthermore, only short simulation results were shown for WT-AP communications only. In [2], the smooth CW decrease was considered, but from the fairness point of view. [2] tries to establish local utility functions in order to achieve system-wide fairness, with no explicit global coordination. Then it "translates" a given fairness model into a

corresponding backoff-based collision resolution algorithms that probabilistically achieve the fairness objective. These algorithms include different backoff increase/decrease factors. [2] Tries to enhance the fairness properties of IEEE 802.11, MACAW [1] and CB-Fair proposed in [3]. Always aiming to establish fair contention algorithms, [3] uses smooth CW increase and decrease functions. Each station i contends to access the channel to send a frame to station j with a probability p_{ij}, computed in two ways using time-based and connection-based methods. These methods are pre-established using information broadcasted by each station about the number of logical connections and the contention time.
One can also find some similarity between working for fairness on the MAC sub-layer and working on fairness on the transport layer. TCP Reno [4] uses AIMD (additive increase and multiplicative decrease) based on [5] in order to attain fairness among flows.

In this proposal, our main aim is to investigate the slow CW decrease functions from the data rate and delay efficiency point of view, not the fairness point of view. We consider various network topologies and schemes to validate our analysis. For simplification, first we introduce *Multiplicative CW decrease scheme* and *Linear CW decrease scheme* into basic DCF mechanism, although it can be extended to multi-priority 802.11e EDCF easily. Then we propose an *adaptive CW decrease scheme* with EDCF. This adaptive scheme can also be used in basic DCF function.

3. MULTIPLICATIVE CW DECREASE FUNCTIONS [7]

In this section we propose a multiplicative CW decrease scheme for 802.11. This scheme uses different slow CW decrease factors δ to compare the performance. First we study its performance in single destination scenario in section 3.1 and then in ad-hoc scenario in section 3.2. We use network simulator (ns-2.1b8) [9] to evaluate our schemes. The raw channel capacity is set to 2 Mb/s, resulting in an effective capacity of 1.7 Mb/s, taking the overheads into consideration. The channel is considered clear (no noise), therefore packet losses are only due to collisions.

3.1 Single destination

First, consider 50 wireless terminals (WTs) uniformly distributed in a 100m $\times$ 100m square area. WT_1 is the common receiver for all other 49 WTs. Simulation starts at second 44, we increase the number of transmitting WTs by one each two seconds: WT_i starts transmission at second $40 + 2i$, $i \geq 2$, WT_1 being the common receiver. All nodes are within the range of each other. Each transmitting WT sends 1050-byte CBR packets each 5ms (providing full data rate). At second 150, all traffic sources stop but one ($WT_2 \rightarrow WT_1$). At second 260, all sources stop sending data.

Optimally, when the number of WTs (i.e. n) increases, each WT would get $1/n$ of the available data rate. However, due to the increasing collisions, frames would be corrupted, not acknowledged and retransmitted, which would decrease the actual date rate observed by each WT. The dashed curve in Fig. 1 shows how the total throughput decreases as the number of contending terminal increases (e.g. seconds 44-150).

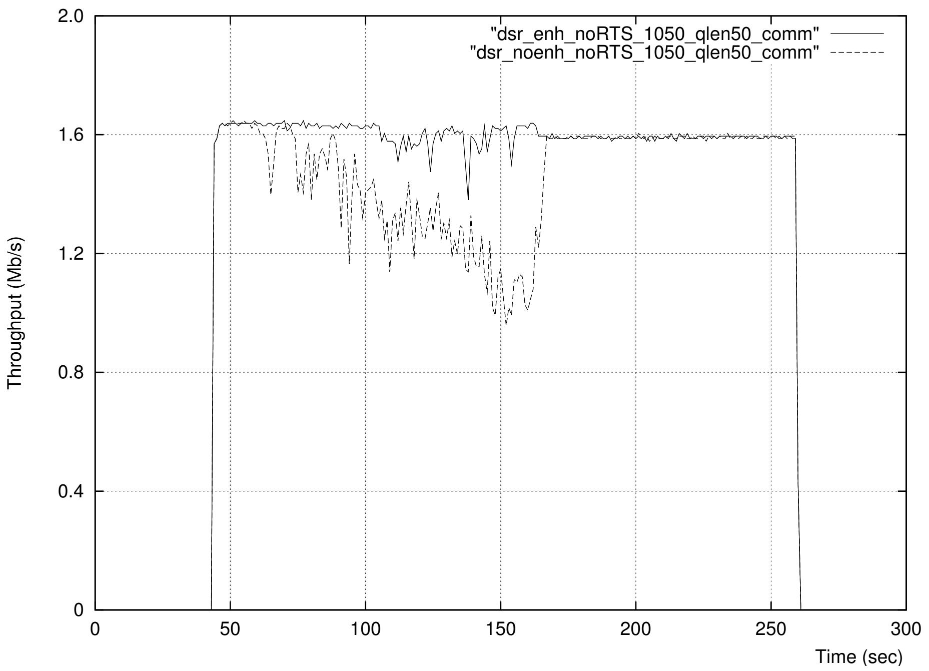


Fig.1 Total throughput comparison, without RTS/CTS.

In fact, after each collision, the source has to wait for a timeout to realize that the frame collided, increases its CW (to reduce further collision risks) then retransmits the frame. After a successful transmission the source resets its CW. As a node resets its CW after a successful transmission, it "forgets" about collision experience it had. If all WTs keep transmitting with the same data rate, most probably the new transmission will observe contention and collisions as before. This can be avoided by keeping some history of the observed collisions: Instead of resetting the CW to CW_{min}, we set the CW to 0.8 times its previous value (low bounded by CW_{min} , i.e. $CW_{new} = \max\{CW_{min}, 0.8\ CW_{prev}\}$). The solid curve in Fig. 1 shows the considerable throughput enhancement we get (up to 53%), especially with high number of transmitting nodes (second 150): When we decrease the CW slowly, we waste more backoff time in favor of collision avoidance. Furthermore, throughput is more stable, due to lower/smoother variations of CW values. The slow CW decrease is a tradeoff between wasting some backoff time and risking a collision followed by the whole frame retransmission. As the time of the latter is much larger than the backoff time, slow CW decrease is much better on the average. The average overhead due to backoff and retransmissions can be written as:

$$E[overhead] = O_{bkof} \times (1 - P_{col}) + O_{retx+bkof} \times P_{col}$$

where P_{col} is the probability of a collision, O_{bkof} is the overhead due to backoff time and

$$O_{retx+bkof} = \sum_{i=1}^{r} (bkof(i) + T_{data})$$

is the overhead due to retransmissions and their corresponding backoffs, r being the number retransmissions until a successful frame reception, *bkof(i)* is the average backoff of the *i*th retransmission and T_{data} is the data retransmission duration.

The worst case for slow CW decrease would be when we consider high CW values, but no congestion is taking place. This is the case at second 150, when we stop all but one transmission ($WT_2 \rightarrow WT_1$) in order to observe the remaining throughput. Fig. 1 shows that the slow CW decrease still behaves better than resetting the CW; after few successful transmissions, the slow CW decrease would reach the CW_{min} value which CW reset would have directly reached. However, the overhead of the slow CW decrease is still negligible compared to a single frame retransmission.

The above analysis is not completely correct. In fact, all traffic sources (but one) stop at second 150, but the effect is "shifted" to around second 168. This is due to the residual packets queued in the interfaces of all 48 WTs (the interface queue length is 50). After sources stop, these remaining packets will continue contending to access the channel, possibly collide and get retransmitted resulting in the following:

- Smooth the sudden sources stop, therefore we cannot observe the real overhead of slow CW decrease when traffic sources suddenly stop.
- As congestion still exists, the slow CW decrease still shows better performance.

To avoid this residual queueing effect, consider now the same scenario as before, but with shorter interface queue lengths (= 2), in order to eliminate the smoothed sources stop and observe the maximum overhead due to slow CW decrease. Fig. 2 shows that the above queueing effects are eliminated, and the overhead due to slow CW decrease can be observed at its worst case (no congestion, high CW values, i.e. second 150).

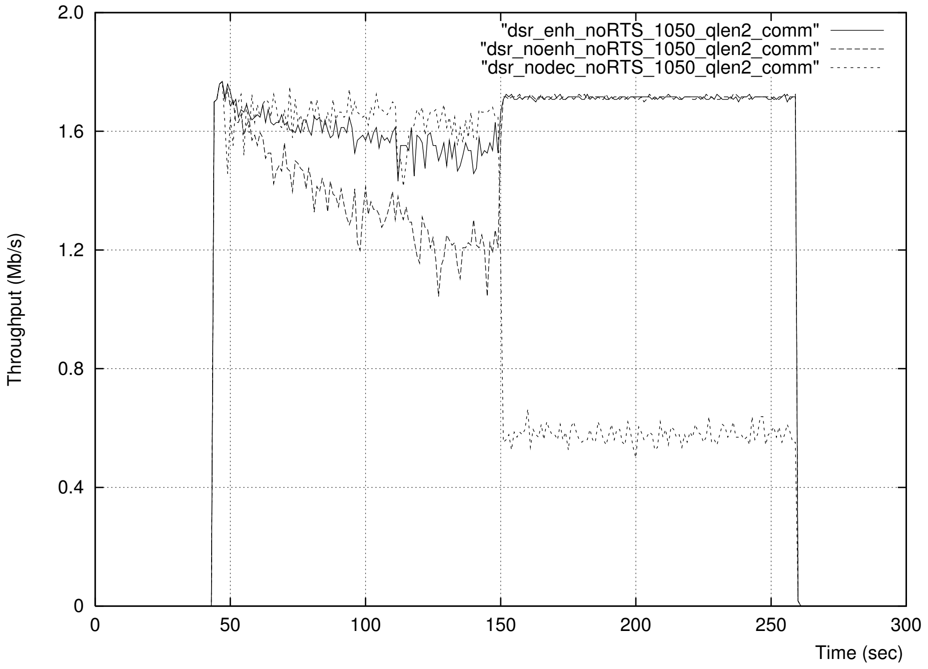


Fig. 2 Total throughput comparison, without RTS/CTS, qlen = 2.

This shows that slow CW decrease (solid-line curve, dsr_enh_noRTS_1050_qlen2_comm) performs as well as CW reset (dashed curve, dsr_noenh_noRTS_1050_qlen2_comm) at low congestion, even right after high congestion. This can be considered as the response of the function to the congestion changing frequency at its maximum. Slow CW decrease performs as well at intermediate congestion variation frequencies, when the number of transmitting sources changes up and down more smoothly.

For comparison convenience, we added a third curve (dsr_nodec_noRTS_1050_qlen2_comm) to Fig. 2, showing the overall throughput when we do not decrease the CW at all, i.e. keeping it at its maximum reached values. This shows that the CW time cannot be absolutely considered as negligible and must be reduced upon successful transmissions. Else, the performance decreases considerably at low congestion (and high CW values, i.e. second 150) as we can see for flow $WT_2 \rightarrow WT_1$ after second 150.

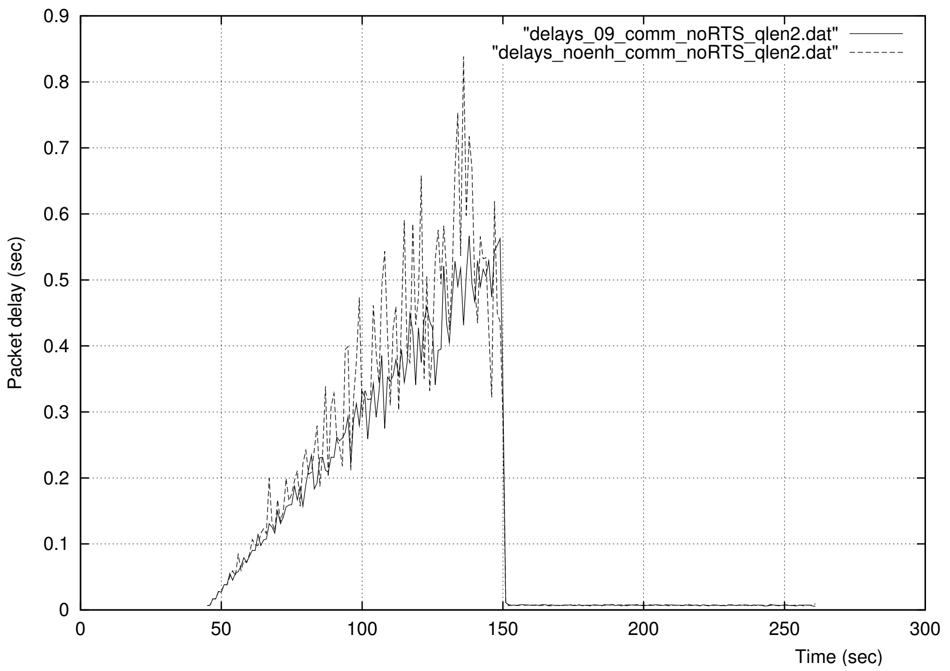


Fig. 3 Packet delays comparison, without RTS/CTS, qlen = 2

Figure 3 shows the delays observed for the same simulation scenarios. We can see how the delays increase with the number of contending nodes for both slow CW decrease (solid curve) and for CW reset (dashed curve). However the slow CW decrease scheme shows relatively lower delays and jitters. Since the CW decreases slowly, we are avoiding more collisions and retransmissions, which is shown by lower average delays. And since the CW varies slowly, staying more adapted to the actual congestion level, the jitter is lower than the one with CW reset by tens of milliseconds. The contention success chances varies with the CW variation, therefore using sudden CW reset after each successful transmission leads to very high jitters. Slow CW decrease has lower jitters, showing the convenience of this approach typically at high congestion levels.

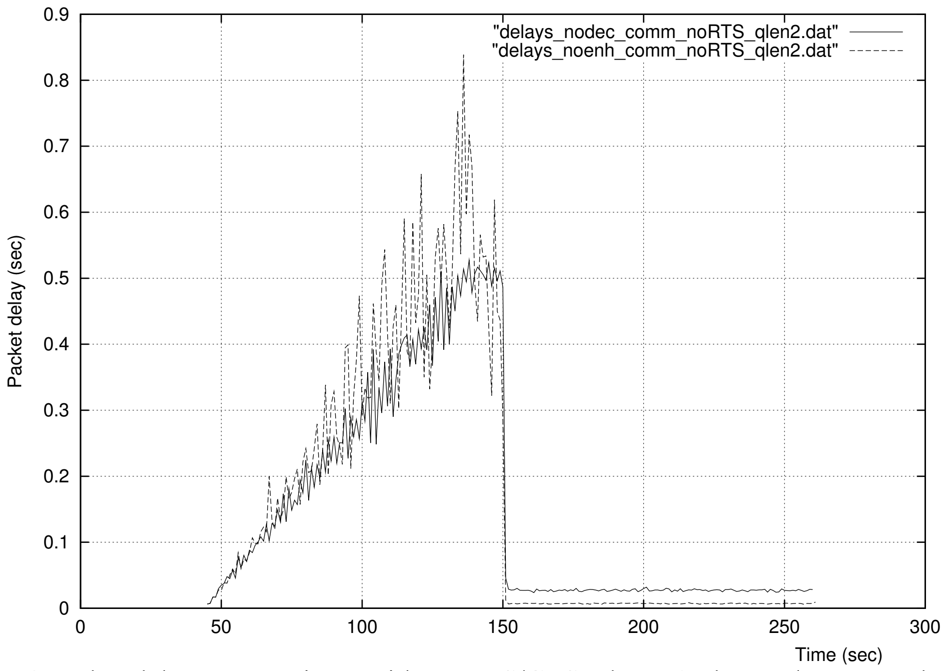


Fig. 4 Packet delays comparison, without RTS/CTS, qlen = 2, the no-decrease scheme

For completeness, we show in Fig. 4 the packet delays when we do not decrease the CW at all (solid curve). It has similar behaviour than slow CW decrease at high congestion levels. However, after the sudden congestion level drop (second 150), this mechanism keeps high CW values leading to high delays (and low throughput, as in Fig. 2). These delays existed before the congestion level drop, but to the avantage of collision avoidance, increasing the throughput and lowering the overall packet delays. We should note that when we consider longer interface queues (e.g. 50), delays become oders of magnitude higher than the delays in Fig. 3 and 4.

When we use short data packets, the relative throughput gain decreases and the slow CW decrease becomes less efficient: the time overhead introduced by the slow CW decrease becomes comparable to the packet payload. To the extent, consider the RTS/CTS exchange before a data packet transmission. Slow CW decrease would avoid (short) RTS collisions which are less probable (in case of hidden nodes) and less severe, from the data rate point of view. Therefore we observe low gain of slow CW decrease over CW reset. This can be seen in Fig. 5. When congestion is low, we observe no gain; CW decrease performs as well as CW reset. At high congestion level (second 150), we observe a 6.8% throughput enhancement. This gain will be shown greater in the next section, when we consider ad-hoc scenarios. Obviously, RTS/CTS adds overhead and performs less than the basic scheme, whether using CW decrease or CW reset.

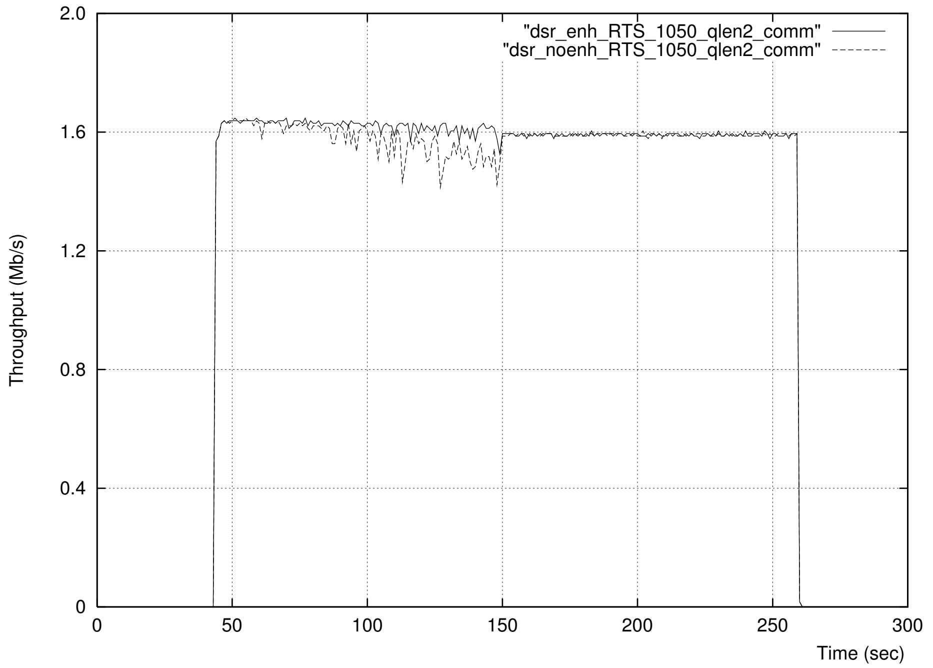


Fig. 5 Total throughput omparison with RTS/CTS.

3.2. Ad-hoc, all-hear scenario

Consider now a different scenario, with 50 WTs sending data to 50 different WTs, all within the range of each other, uniformly spread over a 100m x 100m area. The RTS/CTS scheme is used. Fig. 6 shows a simulation with two similar phases showing different results:

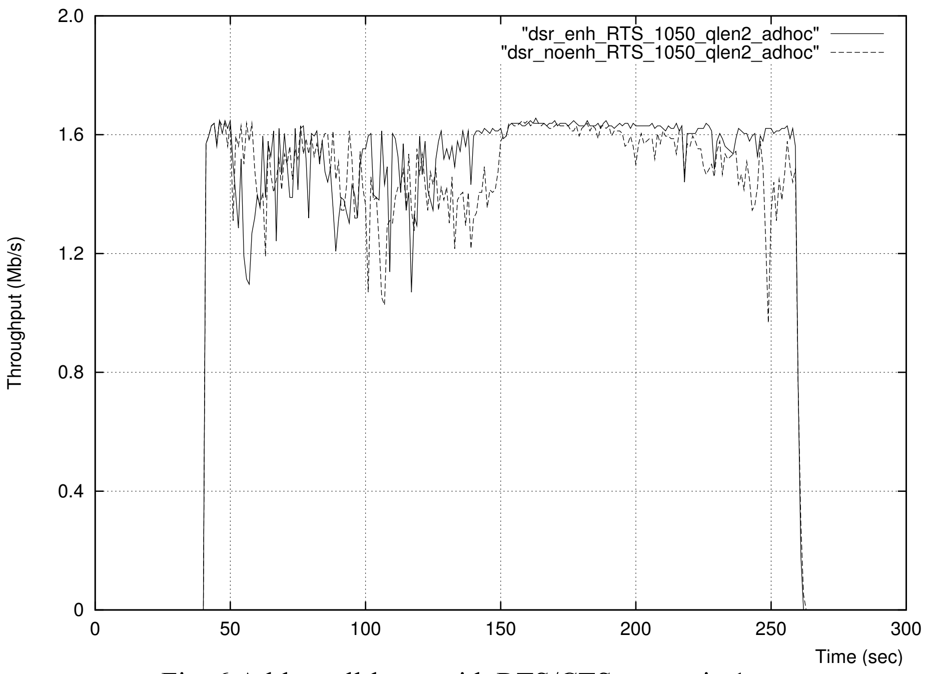


Fig. 6 Ad-hoc all-hear, with RTS/CTS, scenario 1.

In the first phase (seconds 40 to 150) we increase the number of flows by one each 2 seconds. Then we reset the number of flows and start increasing it again (seconds 150 to 260). The throughput in the first phase is lower and varies more than in the second phase, whether using CW decrease or CW reset. This is due to routing information exchange during the first period which, once established, interferes less with throughput results in the second phase. To avoid this transient effect, we added a “warmup” phase to our simulations, seconds 5-40, during which all WTs are active. Then we consider the same scenario as in the previous section: At second 40 all WTs are inactive, and then we activate an additional flow each two seconds, until second 150. Congestion is at its most (second 150) when we turn all WTs off but one flow keeps running in order to observe its CW behavior. Results are shown in Fig. 7.

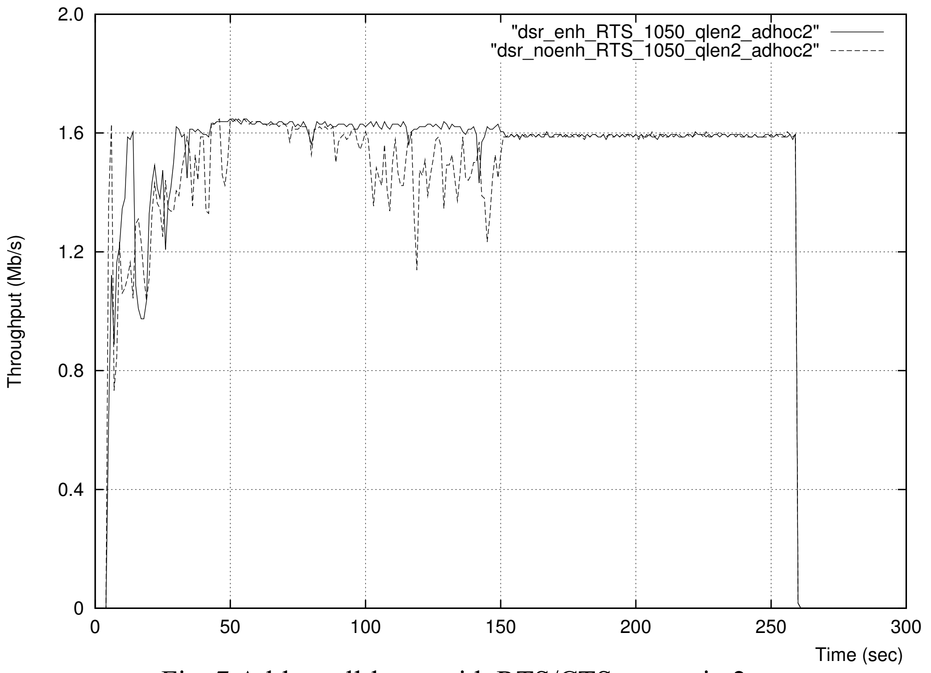


Fig. 7 Ad-hoc all-hear, with RTS/CTS, scenario 2.

We observe that, in ad-hoc topologies and the random scenario we used, the gain of CW decrease over CW reset reaches 15% at high congestion, in contrast to 6.8% obtained with the single destination scenario, even when RTS/CTS is used.

In order to evaluate the performance of the CW decrease approach, we introduce two metrics used in feedback control theory [6]:

- *Throughput gain (G)*: This is the ratio of the throughput obtained by applying CW decrease over the throughput obtained by applying CW reset.
- *Settling time (T_s)*: After a sudden decrease of active WTs number (e.g. second 150), T_s is the time it takes a single flow to reach its throughput steady state, with small CW values. T_s characterizes the system *response time* using CW decrease.

In the following we will use different CW decrease factors δ and different data rates λ (λ = (*data rate*)/(*maximum channel capacity*)) to evaluate *G* and T_s. Fig. 8 shows the throughput gain *G* function of the CW decrease factor δ. We can see that:

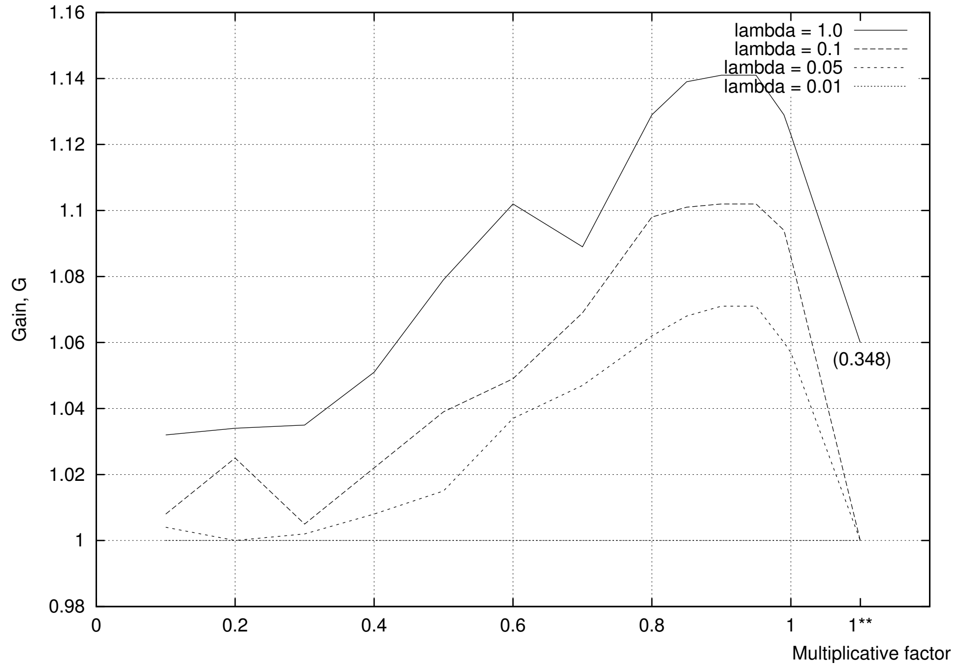


Fig. 8 Throughput gain, *G*, vs. CW decrease factor δ.

- When δ decreases, the CW decrease becomes closer (resembles) to CW reset and shows no enhancement over this last, ($G \rightarrow 1$).
- However, when the multiplicative factor δ is high, CW decreases slowly upon each successful frame transmission, still avoiding future collisions and retransmissions, therefore the throughput is higher than with CW reset (*G*>1). For all δ values, the maximum gain G_{max} is around $\delta_{max} = 0.9$.
- As λ decreases (lower data rates) the gain *G* converges to unity. In fact, when data rates decrease, we observe fewer collisions leading to fewer CW increase and CW decrease. Therefore, the gain of slow CW decrease over CW reset gets lower and converges to one.
- When λ=1, we observe a considerable gain *G*>1 when the channel is highly congested (as seen in Fig. 2). However, when the channel becomes less congested, the CW value keeps constantly high, increasing overhead, and decreasing throughput efficiency. This is what we denoted by (1**) in Fig. 8. For low data rates, this overhead (when δ=1) is negligible relative to the idle channel periods between consecutive packets. Therefore the gain *G*=1. However, when λ=1, this overhead becomes

considerable leaving large idle gaps between packets, reducing efficiency, therefore the gain drops to $G = 0.348$.

- When using $\delta<1$, the CW size (and overhead) will progressively decrease upon each successful transmission. Therefore the overhead cited above (with $\delta=1$) will still exist but for a transient period only, the duration of which is function of δ, the frame data rate λ and the corresponding successful transmissions. This transient period is characterized by T_s, the settling time we defined above.

To measure T_s with acceptable precision, we cannot proceed as in Fig. 1 right after all traffic flows (but one) stop and simply measure the time it takes the remaining flow (flow-1) to get to its stable state. In such a scenario, flow-1 is contending to access the channel with other flows. It has a non-zero probability to access the channel right before second 150, and therefore start its transient period with a short CW, unsuitable to measure T_s. We proceed using a different simple scenario, as in Fig. 9. (This scenario corresponds to the system response to an impulse input, from the feedback control point of view):

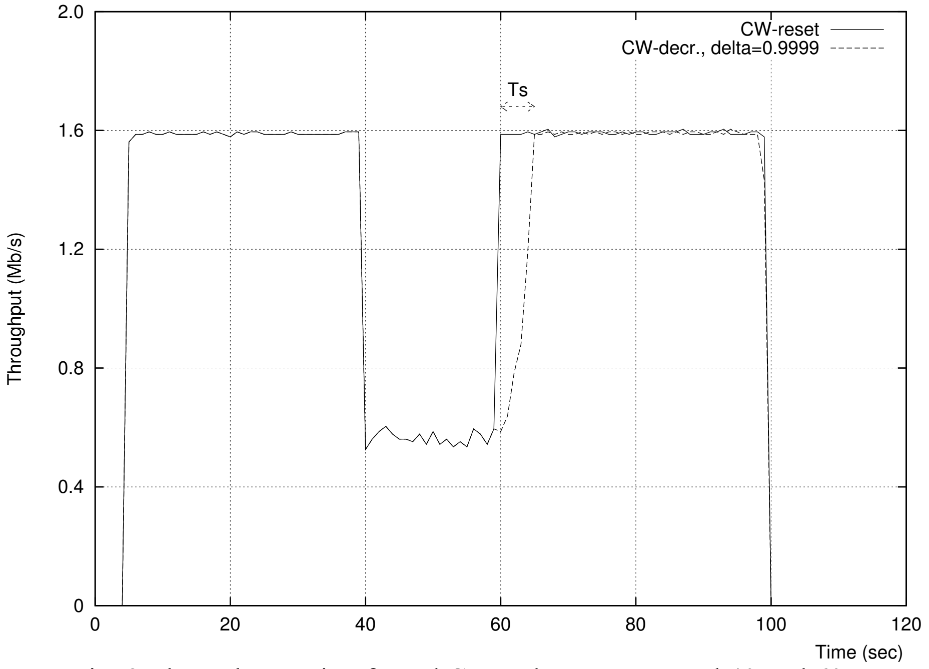


Fig. 9 Throughput using forced CW_{max} between second 40 and 60

A single flow is considered. It starts at second 5, then from second 40 to second 60 we force the CW to its maximum, 1023, as it would be in highly congested environments. This reduces its throughput considerably. At second 60 we let the CW use CW decrease and CW reset respectively, and measure the settling times T_s. We used a large δ value, 0.9999, so T_s would be visible enough on the figure's scale. For lower δ values we will "zoom" into second 60. Figure 10 shows that, as one would intuitively think when δ increases, we need more successful transmissions before throughput reaches its steady state, that is T_s increases.

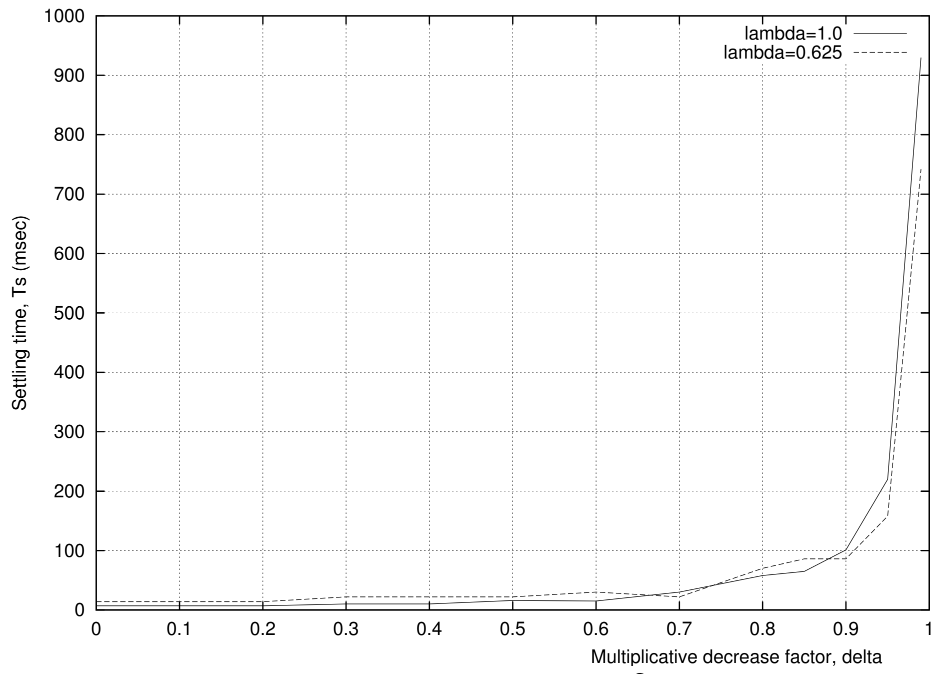


Fig. 10 Settling time *Ts* vs. δ.

This increase is much higher than linear, especially for high δ values. The reader should distinguish the settling time T_s from frame transmission delays. The first concerns throughput stability, while the second concerns transmission delays. In the previous examples, a T_s of one second simply means that 200 frames should be sent successfully before the throughput reaches its high steady state. However, evaluating the user perception of T_s is out of scope of this work. Choosing the right multiplicative decrease factor δ is a compromise between having a high throughput gain *G* and a short settling time T_s, for the case of sudden congestion decrease. Intermediate δ values like 0.6-0.8 would satisfy such a tradeoff. For smoother congestion decrease (practically, this is hard to predict), one would choose higher δ values to get higher throughput gains, without much care about T_s.

4. LINEAR CW DECREASE FUNCTIONS

In this section we repeat the same analysis of the previous section but with linear CW decrease, i.e. upon each successful frame transmission, CW is decreased by a constant value α. From the throughput gain G point of view, Fig. 11 shows that linear CW decrease can reach the same gain values as multiplicative CW decrease.

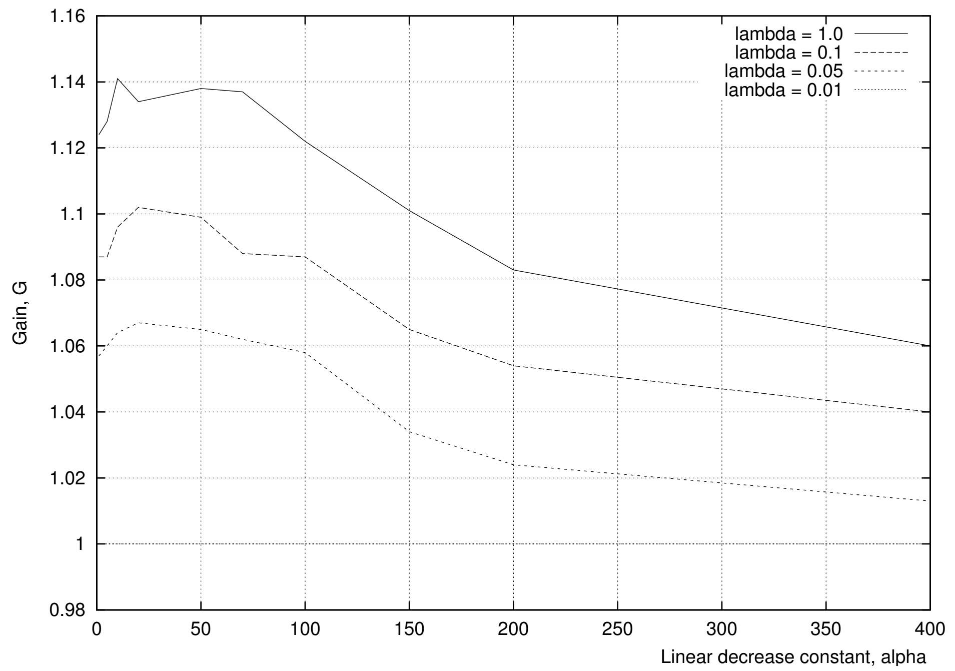


Fig. 11 Throughput gain, *G*, vs. CW decrease constant α.

When α is small, CW decreases slowly, avoiding future collisions and retransmissions, leading to a throughput enhancement, just like high δ values of the previous section. However, the settling time T_s shown in Fig. 12 is higher than with multiplicative CW decrease, especially for small α values (α<100) that would result in good throughput enhancement (*G*>1.12).

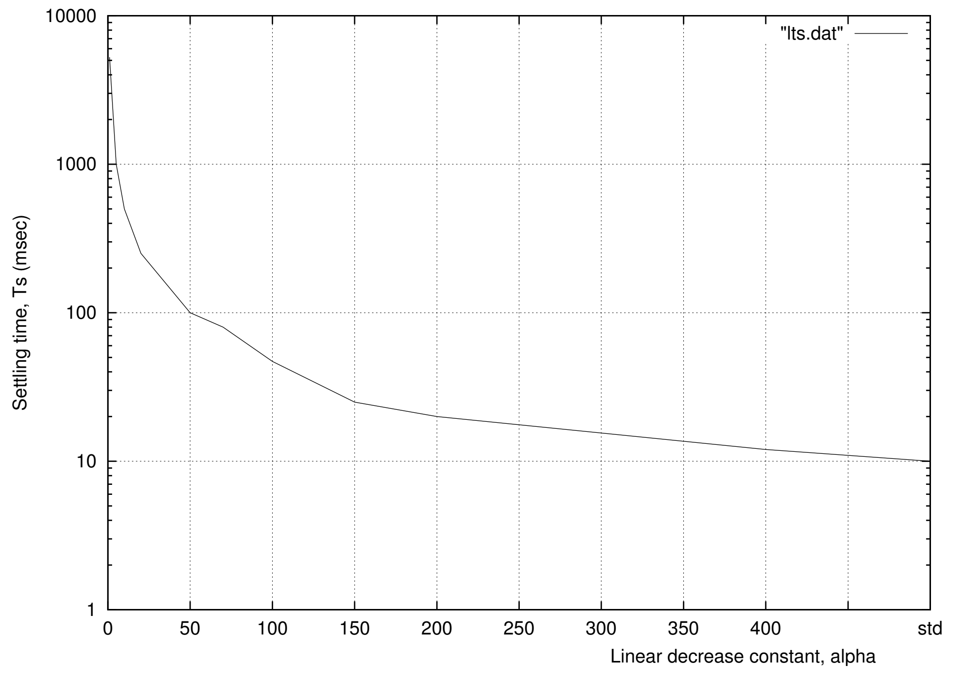


Fig. 12 Settling time Ts vs. α.

Finally we should note that in [1], the authors use linear CW decrease with α=1. This surely enhances throughput, as would very high δ values do with multiplicative CW decrease. However, Fig. 10 and 12 show that very high δ values and very low α values would lead to unacceptable settling times T_s, if one considers sudden congestion level drop. From the user point of view, high settling time values (T_s) mean longer delays before the user gets the maximum throughput after moving from a highly congested area to a low congested one, or when all of his neighbors suddenly end their transmissions.

We aim to push our analysis further to investigate the influence of slow CW decrease on battery consumption.

5. ADAPTIVE CW DECREASE FUNCTIONS

In the previous sections, fixed decrease factors (multiplicative or linear) are used. As mentioned in Section 3.1, if there is no congestion in the wireless channel, fixed slow CW decrease may increase overheads due to high CW values used. That means, a static decrease factor cannot be optimal in time-varying wireless channels. In this section we investigate an adaptive way to decrease slowly the contention window. We propose this scheme for 802.11e EDCF. In this scheme, several classes are used in every station. Each class updates its CW parameter in an adaptive way taking into account the estimated collision rate f_{curr}^{j} in each station. Similar to TCP packet loss rate, the collision rate can give an indication about contentions in a distributed wireless network. The estimated collision rate f_{curr}^{j} is calculated using the number of collisions and the total number of packets sent during a constant period (i.e. a fixed number of slot times) as follows:

$$f_{curr}^{j} = \frac{E(collisions_{j}[p])}{E(data\_sent_{j}[p])}$$

where *E(collisions*$_j$*_[p])* is the number of collisions of station *p* which occurred at step *j*, and *E(data_sent*$_j$ *[p])* is the total number of packets that station *p* has sent during the same period *j*. Note that the above ratio f_{curr}^{j} is always in the range of [0,1].

To minimize the bias against transient collisions, we use an estimator of Exponentially Weighted Moving Average (EWMA) to smoothen the estimated values. Let f_{avg}^{j} be the average collision rate at step *j*. The average collision rate is computed dynamically in each period T_{update} expressed in time-slots. This period is called update period. It should not be too long in order to get good estimation and should not be too short in order to limit the overheads. For each update period, f_{avg}^{j} is computed according to the following iterative relationship:

$$f_{avg}^{j} = (1-\varphi)\times f_{curr}^{j} + \varphi \times f_{avg}^{j-1}$$

where *j* refers to the *jth* update period and f_{curr}^{j} stands for the instantaneous collision rate, φ is the weight (also called the smoothing factor in our scheme) and determines the memory size used in the averaging process.

To ensure that the priority relationship between different classes is still fulfilled when a class updates its CW, each class should use different factor according to its priority level (we denote this factor by Multiplicator Factor or *MF*). In our scheme the decrease factor used to reset the CW does not exceed the previous CW, we limit the maximum value of *MF* to 0.8. We have fixed this limit according to a tradeoff between throughput and delay gains from a large number of simulations done with different scenarios. In our scheme, the *MF* of class *i* is defined as follows:

$$MF[i] = \min((1+2\times i)\times f_{avg}^{j}, 0.8)$$

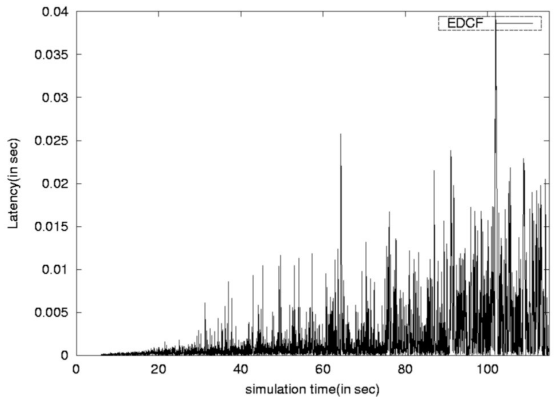
This formula allows the highest priority class to reset the CW parameter with the smallest *MF* value, which maintains the service differentiation effect when optimizes the throughput and delay performance.

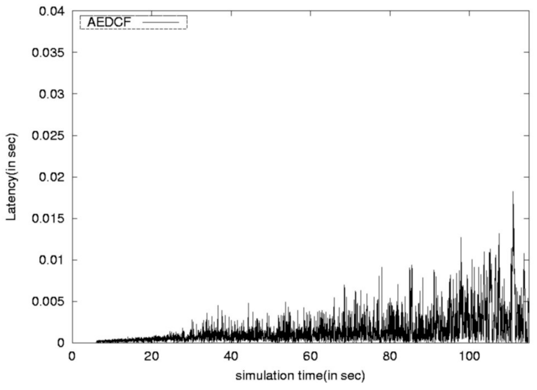
After each successful transmission of packet of class *i*, *CW[i]* is then updated as follows:

$$CW_{new}[i] = \max(CW_{\min}[i], CW_{old}[i]\times MF[i])$$

This equation guarantees that *CW[i]* is always greater than or equal to $CW_{min}[i]$, so the priority access to the wireless medium is always maintained.

After each collision, the persistence factor *PF*[i] is used for CW increase for further differentiation. We denote this adaptive scheme as Adaptive EDCF (AEDCF). Performance comparisons between AEDCF and 802.11e EDCF scheme show that AEDCF outperforms the EDCF, especially at high traffic load conditions: AEDCF increases the medium utilization ratio and reduces more than 50% of the collision rate. While achieving delay differentiation, the overall goodput obtained is up to 25% higher than EDCF. Figure 13 shows the audio flow delay comparison between AEDCF and EDCF. Adaptive EDCF meets target by maintaining the delay lower and smoother than EDCF, which is very important for audio flows. Other simulation results are detailed in [8]. And our ns AEDCF source codes are available from our INRIA research website [10].





(a) EDCF (b) Adaptive EDCF (AEDCF)

Figure 13 Delay variation of audio flow

Complexity of Adaptive EDCF

The complexity of Adaptive EDCF is similar to the complexity of EDCF, only a few more resources are required. Some registers are necessary to buffer the parameters defined above: f_{avg}^{j-1}, T_{update}, *MF[i]* and φ. The f_{avg}^{j} and *MF[i]* parameters are updated only at the beginning of each new update period T_{update}. The calculation of *MF[i]* requires one addition, two multiplications and one comparison for each active class. Then, two multiplications and two additions are required to compute f_{avg}^{j} and one more division to obtain f_{curr}^{j}. One comparison and one multiplication are used to compute *MF[i]* and to decide how to reset the *CW[i]*. Finally, during the update period, two counters are needed to increment collisions and data sent, one comparison and one multiplication are introduced to calculate the $CW_{new}[i]$ and to decide which value will be used to reset the CW.

6. CONCLUSION

In this paper we investigate several slow CW decrease schemes instead of CW_{min} reset scheme after each successful frame transmission. These schemes allow reducing the number of collisions, considering that congestion level is likely to stay constant. They also decrease the number of frame retransmissions (which would also reduce congestion on the channel), increasing the throughput considerably. They perform as well as IEEE 802.11 in non-congested environments, and show considerable gain over the latter in congested ones. The throughput gain is a function of frame lengths and data rates. It reaches 53% when using large data frames. We extend the analysis for the worst gain values, that is for short data frames, e.g. when using RTS/CTS. Both multiplicative and linear CW decrease functions show considerable throughput gains at date rate δ=0.9 and α=50 respectively, with relatively low settling times after sudden congestion level drops. Using more complex CW decrease schemes from feedback control theory, this settling time can be enhanced. However, this adds much more complexity on the MAC sub-layer, slightly enhancing the settling times without any throughput gain enhancement. We also explore adaptive CW decrease algorithms for EDCF, in which decrease parameters change according to the congestion load level. Performance comparisons between AEDCF and EDCF scheme show that AEDCF outperforms the EDCF, especially at high traffic load conditions: AEDCF increases the medium utilization ratio and reduces more than 50% of the collision rate.

Future work will consider modeling these schemes, trying to establish relations between the gain, settling time, data rates, frame lengths, number of active WTs and the CW decrease parameters. Slow CW decrease scheme also enhances delays and battery power consumption, which we also plan to explore in the near future.

REFERENCES

[1] Vaduvur Bharghavan, Alan Demers, Scott Shenker, and Lixia Zhang, “MACAW: A media acess protocol for wireless LANs”, in *Proceedings of ACM Sigcomm*, 1994.
[2] Thyagarajan Nandagopal, Tae-Eun Kim, Xia Gao, and Vaduvur Bharghavan, “Achieving MAC layer fairness in wireless packet networks”, in *Proceedings of Mobicom*, 2000.
[3] Timucin Ozugur, Mahmoud Naghshineh, Kermani Parviz, Michael Olsen, Babak Rezvani, and John Copeland, “Balanced media acess methods for wireless networks”, in *Proceedings of Mobicom*, 1998.
[4] Mo Jeonghoon, J. La Richard, Anantharam Venkat, and C. Walrand Jean, “Analysis and comparison of TCP Reno and Vegas”, in *Proceedings of Infocom*, 1999.
[5] D. Chiu and R. Jain, “Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks”, in *ComputerNetworks and ISDN Systems 17*, 1989.
[6] Charles L. Phillips and Royce D. Harbor, “*Feedback control systems*”, Prentice-Hall, 1988.
[7] Imad Aad’s Ph.D. Thesis, “Quality of service in wireless local area networks”, France, Oct.7, 2002
[8] Lamia Romdhani, Qiang Ni, and Thierry Turletti. “AEDCF: enhanced service differentiation for IEEE 802.11 wireless ad-hoc networks”, in *INRIA Research Report No. 4544*, Sept. 2002
[9] ns website. http://www.isi.edu/nsnam/ns/index.html
[10] AEDCF website. http://www-sop.inria.fr/planete/qni/Research/AEDCF/