Master RSD

University of Nice Sophia-Antipolis

Internship Report

Conception and Analysis of an Adaptive Backoff Algorithm for 802.11 Wireless LANs

Prepared by :

Mouhamad IBRAHIM

Supervised by :

Sara ALOUF

INRIA Sophia-Antipolis, MAESTRO Team

March - June 2005

Acknowledgments

First of all, I wish to express my gratitude to my advisor Dr. Sara Alouf, for her constant support and valuable guidance during my internship, and also for her patience in reviewing the different versions of this thesis.

Special thanks go to Prof. Philipe Nain for giving me the opportunity to work in the group Maestro, for his support and for his efforts during the Master. I would like also to thank the members of our team, in particular Ahmad, Florence, Ephie and my companions in the office Dieter and Nicolas.

Last but not least, thanks to my family for their love, and for being by my side whenever I needed it.

Contents

1 Introduction **1**

2 Preliminaries on IEEE 802.11 **3**
2.1 Overview of IEEE 802.11 . . . 3
2.2 Distributed Coordination Function (DCF) . . . 4
2.3 Analytical Model for the DCF . . . 5
2.3.1 Normalized System Throughput . . . 7
2.3.2 Optimal Transmission Probability . . . 8
2.4 Insights on the Optimality of the Transmission Probability . . . 8
2.5 Conclusion . . . 10

3 Overview of Dynamic Contention Window Algorithms **11**
3.1 Adaptive Backoff algorithm of Bianchi *et al.* . . . 11
3.2 Adaptive Backoff algorithm of Calli *et al.* . . . 12
3.3 Multiplicative Slow Decrease Algorithm . . . 13
3.4 Additive Increase Decrease Algorithm . . . 13
3.5 Conclusion . . . 14

4 Proposed Approach for Adaptive Binary Exponential Backoff **15**
4.1 Adaptive Binary Exponential Backoff Approach . . . 16
4.2 Computing the Optimal Initial Contention Window . . . 17
4.3 The Adjustment Algorithm . . . 19
4.4 Estimating the Effective Number of Contending Stations . . . 19
4.5 Conclusion . . . 20

5 Simulation Results **22**
5.1 Simulation Parameters . . . 22
5.2 Comparing Analytical and Simulative Throughput . . . 23
5.3 Steady and Transient Network State Analysis . . . 23
5.3.1 Fixed Number of Stations . . . 24
5.3.2 Abrupt Change in the Number of Stations . . . 24
5.3.3 Poisson Arrival with Exponential Activity Time . . . 25
5.4 Parameters Analysis of the ARMA Filter . . . 26
5.4.1 Case of Slowly Increasing Number of Stations . . . 27
5.4.2 Case of Abruptly Increasing Number of Stations . . . 29
5.5 Infrastructure Mode Analysis . . . 30
5.6 Robustness to Noisy Environments . . . 31
5.6.1 Bernoulli Model . . . 33
5.6.2 Gilbert Model . . . 34
5.7 Conclusion . . . 39

6 Open Issues **40**
6.1 Fairness Analysis 40
6.2 TCP Traffic 41

7 Conclusion and Future Work **43**

A Infrastructure Mode with Upload and Download Traffic **45**

List of Figures

2.1 An example of the DCF access mechanism. 5
2.2 Markov chain for the DCF mechanism. 6
2.3 An illustration of the virtual transmission time. 9
2.4 Average collision and average idle times between two successive transmissions for different values of n ($T_c = 4335\ \mu s$ and $\sigma = 20\ \mu s$). 10

4.1 Analytical average number of collision between two successive transmissions as a function of the number of stations. 16
4.2 System throughput as a function of the initial contention window size and the number of users (data rate of 2Mbps). 17

5.1 Analytical and simulative throughput of the adaptive BEB algorithm. 24
5.2 Overall throughput in the steady state scenario case. 24
5.3 Mean delay in the steady state scenario case. 25
5.4 Temporal evolution of the number of active stations in the abrupt scenario case. . . 25
5.5 Overall throughput in the abruptly varying scenario case. 26
5.6 Mean delay in the abruptly varying scenario case. 26
5.7 Overall throughput in the Poisson arrival scenario case. 27
5.8 Mean delay in the Poisson arrival scenario case. 27
5.9 Overall throughput comparison in the slow scenario case. 28
5.10 Mean delay comparison in the slow scenario case. 28
5.11 Comparision of different estimations of the number of stations in the slow scenario case. 28
5.12 Contention window evolution comparison in the slow scenario case. 29
5.13 Overall throughput comparison in the abrupt scenario case. 29
5.14 Mean delay comparison in the abrupt scenario case. 29
5.15 Comparison of different estimations of the number of stations in the abrupt scenario case. 30
5.16 Contention window evolution comparison in the abrupt scenario case. 30
5.17 Overall throughput in the slow scenario case (Infrastructure mode). 31
5.18 Mean delay in the slow scenario case (Infrastructure mode). 31
5.19 Overall throughput in the abrupt scenario case (Infrastructure mode). 32
5.20 Mean delay in the abrupt scenario case (Infrastructure mode). 32
5.21 Overall throughput in the Poisson scenario case (Infrastructure mode). 32
5.22 Mean delay in the Poisson scenario case (Infrastructure mode). 33
5.23 Impact of noisy channel on the throughput of 5 sources (Bernoulli model). 33
5.24 Impact of noisy channel on the mean delay of 5 sources (Bernoulli model). 34
5.25 Impact of noisy channel on the throughput of 10 sources (Bernoulli model). 34
5.26 Impact of noisy channel on the mean delay of 10 sources (Bernoulli model). 35
5.27 Impact of noisy channel on the throughput of 15 sources (Bernoulli model). 35
5.28 Impact of noisy channel on the mean delay of 15 sources (Bernoulli model). 36
5.29 2-state Markov channel. 36
5.30 Impact of noisy channel on the throughput of 5 sources (Gilbert model). 36

5.31 Impact of noisy channel on the mean delay of 5 sources (Gilbert model). 37
5.32 Impact of noisy channel on the throughput of 10 sources (Gilbert model). 37
5.33 Impact of noisy channel on the mean delay of 10 sources (Gilbert model). 37
5.34 Impact of noisy channel on the throughput of 15 sources (Gilbert model). 37
5.35 Impact of noisy channel on the mean delay of 15 sources (Gilbert model). 38

6.1 Average throughput of 3 random sources in the steady-state scenario case. 41
6.2 Throughput of five random sources in the abrupt scenario case. 41
6.3 Overall throughput in the abrupt scenario case with TCP traffic (ad-hoc mode). . 42
6.4 Overall throughput in the abrupt scenario case with TCP traffic (infrastructure mode). 42

A.1 Overall throughput in the slow scenario case (Infrastructure mode). 45
A.2 Mean delay in the slow scenario case (Infrastructure mode). 45
A.3 Overall throughput in the abrupt scenario case (Infrastructure mode). 46
A.4 Mean delay in the abrupt scenario case (Infrastructure mode). 46
A.5 Overall throughput in the Poisson scenario case (Infrastructure mode). 46
A.6 Mean delay in the Poisson scenario case (Infrastructure mode). 46

List of Tables

5.1 DSSS system parameters. 23
5.2 Throughput expressed in KBytes/s and averaged over 190 seconds of simulations (Gilbert model). 38
5.3 End-to-end delay expressed in seconds and averaged over 190 seconds of simulations (Gilbert model). 38

6.1 Throughput for each source (KBytes/s) averaged over 90 seconds of simulation. . . 40
6.2 Throughput for each source (KBytes/s). 41

Chapter 1

Introduction

Wireless technologies are becoming day after day an essential networking technology of the twenty-first century. Largely deployed in hot-spots such as universities campus, airports, hospitals, and conference rooms, these technologies provide mobility, easy-to-access and fast-to-deploy networks at very competitive prices. Several wireless standards have appeared to support a standardized architecture for these technologies like the IEEE 802.11, HiperLAN, and Bluetooth. IEEE 802.11 is the most widely deployed one.

Appeared in its initial version in 1997, this standard defines Medium Access Protocol (MAC) and Physical (PHY) frameworks to provide wireless connectivity in Wireless Local Area Network (WLAN) (see [12]). However, the deployment of this standard have engendered several research issues. In fact, WLANs features such as limited bandwidth and energy, time-varying topologies, and sensitiveness to noise on the medium are not fully captured by the standard. The main drawback of IEEE 802.11 is the low performance of its MAC protocol in terms of throughput and access delay in congested and dynamic network topologies. Several research studies have pointed out this problem and proposed solutions to better react to the congestion in the network. Their focus was mainly on adapting the standard to handle congested situations. However, the performance of these algorithms in dynamic and transient situations or in the case of noisy environments was not satisfactory.

In wireless LANs, network topologies can change randomly and abruptly due to the mobility of the nodes, and the traffic load seen in these networks will change accordingly. In addition, wireless mediums are noisy by nature. Hence, an effective MAC algorithm is a one that, in addition to effectively handling congested networks, reacts quickly to dynamic network changes and performs well in noisy channels.

The work presented in this thesis consists in the design and the validation of a new adaptive and robust algorithm that enhances the performance of the DCF mechanism of the IEEE 802.11 standard under various network states and configurations. The conception of the adaptive algorithm proposed in this thesis consists of two parts. The first part consists in deriving the basic analytical formulas for an optimal operation of the standard and a way to implement it in the standard. The second part of the algorithm involves an estimation of the number of active stations. Throughout a series of simulations conducted in *Network Simulator 2* ([19]), we show that the performance obtained with our algorithm in terms of throughput and mean packet delay is very promising and outperforms other proposed algorithms in cases of dynamically changing networks, infrastructure mode of operation, or even noisy channels.

This thesis is organized as follows: in Chapter 2, we give a brief overview of the IEEE 802.11 standard and its famous distributed coordination function (DCF). We also introduce a proposed analytical model for the DCF, and discuss some issues relating to that model. Chapter 3 gives a survey of some recent proposed algorithms related to our work. In Chapter 4, we present our proposed algorithm, motivate the reasons behind its concept, and describe its design and the implementation. In Chapter 5, the effectiveness of our proposal is analyzed and compared to other approaches under various scenario cases. Chapter 6 presents two brief analyses of the fairness and the behavior of adaptive approaches in the case of TCP traffic. Finally, in Chapter 7, we summarize our work and outline some future extensions to our study.

Chapter 2

Preliminaries on IEEE 802.11

2.1 Overview of IEEE 802.11

Released in 1997 by the Institute of Electrical and Electronics Engineers (IEEE), the 802.11 protocol specifies a standardized architecture to support data transmission over the air to up to 2 Mbps. Recognizing the increasing need for higher data transmission rates, IEEE ratified later the 802.11b standard for transmissions of up to 11 Mbps, and afterward, the 802.11a and 802.11g standards to support higher data rate up to 54 Mbps. In our report, we concentrate only on the basic 802.11 standard.

The specifications of 802.11 outlines two possible modes of operation: infrastructure mode and ad hoc mode ([12]). In the infrastructure mode, nodes, either fixed or mobile, communicate with fixed base stations called access points (AP). The access points are connected to wired networks hence bridging wireless nodes to other wired nodes. The coverage area of an access point is referred to as a cell where the radius varies from few tens of meters to some hundred meters. In the ad-hoc mode, there is no base stations, and nodes can communicate directly with each others as in a peer-to-peer network. The coverage area is movable and is generally limited by the transmission range of wireless adapters of the participating nodes.

Like all IEEE 802 standards, the 802.11 standards focus on the bottom two levels of the ISO model [20], the physical layer and the data link layer. In the physical layer, three physical technologies have been defined and interact with a single data link layer:

- Frequency Hopping Spread Spectrun (FHSS) in the 2.4 GHz band.
- Direct Sequence Spread Spectrum (DSSS) in the 2.4 GHz band.[1]
- InfraRed.

The data link layer consists of two sublayers: Logical Link Control (LLC) and Media Access Control (MAC) [12]. 802.11 uses the same LLC sublayer and 48-bit addressing as other 802 LANs, thus allowing for simple bridging from wireless to IEEE wired networks. Similarly to the MAC protocol of 802.3 wired network, 802.11 MAC protocol is designed to regulate the access of multiple users on a shared medium, and to handle collisions that occur when two or more stations try to simultaneously transmit. However, due to the specificities of the wireless medium, 802.11 introduces other mechanisms at the MAC level. Two different medium access coordination functions have been defined in 802.11: *Point Coordination Function* (PCF) and *Distributed Coordination*

[1] DSSS is the used physical layer in the rest of the report.

Function (DCF).

Intended to support real-time transmissions by using a centralized polling mechanism, PCF is a contention-free access method. In fact, the medium access is controlled by a polling coordinator, usually situated at the AP, that polls the participating nodes in the cell, thus eliminating the contention between the nodes. Thereafter we will consider only the DCF mechanism.

2.2 Distributed Coordination Function (DCF)

DCF is the primary access protocol in 802.11 for sharing the wireless medium between active stations, mostly fitted for delay-insensitive applications. It is the only possible way of medium access in the case of ad-hoc networks, while it can be employed with PCF in the case of infrastructured networks.

DCF is based on the *Carrier Sense Multiple Access/Collision Avoidance* (CSMA/CA) protocol, with discrete time scale, slotted into constant intervals called *Time Slot*. The carrier sense activity consists in listening to the channel before transmitting: if the channel is found to be idle for a time interval greater than the *Distributed InterFrame Space* (DIFS) period, the station transmits directly, otherwise the station differs its transmission time until the ongoing transmission terminates. When the channel becomes idle again for a DIFS, the station enters the collision avoidance phase by selecting a random interval of time slots called backoff interval that is used to initialize a backoff counter [12].

Backoff interval is an integer number of time slots where the value is randomly chosen in the interval $[0, CW-1]$. CW denotes the current size of the *Contention Window* and reflects the number of unsuccessful transmissions of the current packet. Starting from an initial value CW_{min} called the minimum contention window size, the value of the contention window is doubled after each non-acknowledged transmitted packet until reaching a maximum value noted by $CW_{max} = 2^m CW_{min}$; m is called the maximum backoff stage. According to the standard, a maximum number of seven retransmissions are allowed before the frame is dropped. The Backoff counter is a timer decreased as long as the channel is sensed idle, stopped when a transmission is in progress, and reactivated when the channel is sensed idle again for more than DIFS. When the backoff counter reaches zero, the station transmits directly, then waits for an *Acknowledgement* (ACK) from the destination station.

If the packet is well received by the destination station, the latter will send an ACK to the source station after a period of time called *Short InterFrame Space* (SIFS); no other station is able to detect the idleness of the channel during the acknowledgement phase since the duration of the SIFS plus the duration of the propagation delay of the ACK packet is shorter than a DIFS. Upon receiving the ACK, the source station will reset its contention window size to the minimum value CW_{min} and proceed to the next packet in the buffer. At this point, two rules are to be envisioned if there are other packets in the buffer ready to be transmitted. Specifically, if the new and the previously transmitted packets are fragments from the same packet delivered to the MAC layer by the higher layers, then the remaining fragments are transmitted in sequence separated by a SIFS period, and therefore no backoff is needed. However, if it is not the case, then in order to avoid channel capture, source station is forced to wait for a random backoff time even if the medium is sensed idle for more than a DIFS. It should be noted here that the values of the mentioned system parameters are physical layer dependent.

On the other hand, if the transmitting station did not receive an ACK within a specified ACK timeout, it will consider that the transmitted packet was lost due to a collision, and it will start the backoff procedure again after doubling its contention window size. Figure 2.1 illustrates the operation of the DCF access mechanism.

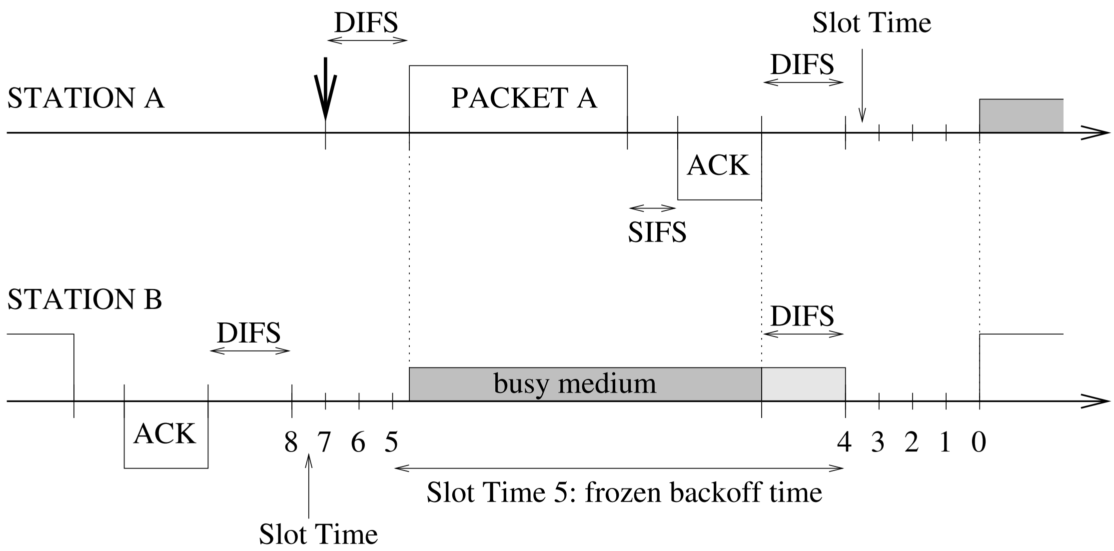

Figure 2.1: An example of the DCF access mechanism.

In addition to the two-way handshaking technique described above and named *Basic Access* mechanism, DCF specifies another optional four-way handshaking technique called *Request To Send* (RTS) / *Clear To Send* (CTS). RTS/CTS constitutes an extension of the basic access technique as it provides a solution for the so-called hidden node problem. The latter problem occurs when the transmissions of two stations collide due to the fact that the two transmitting stations cannot hear each other but their destinations can hear them both. The RTS/CTS technique works as follows: a station willing to transmit will first listen to the channel for a DIFS, follow the backoff procedure before transmitting a short control packet, the RTS packet, which will include both source and destination addresses and the duration of the transmission. If the destination station senses the medium free, it will respond after a SIFS with a Clear To Send (CTS) packet, which will include the same duration information. All stations receiving either the RTS or the CTS packets, will initialize an indicator called *Network Allocation Vector* (NAV) to the duration of the transmission and will use this vector to delay their transmissions, therefore reducing potential collisions.

RTS/CTS is well suited for large packet transmissions as it greatly reduces the time spent in packet collision and in collision detection due to the small RTS packet size. However, its performance degrades in the case of small sized packets or in the case of sub-optimal channel usage. In this report, we focus our study on the basic access mechanism, and leave the RTS/CTS access mechanism for future investigation.

2.3 Analytical Model for the DCF

We based our study on the popular markovian model proposed by Bianchi [5]. In fact, by assuming ideal channel conditions of no hidden node problem and no channel capture, and by claiming a fixed number of active stations working in asymptotic conditions (i.e. buffers always have packets ready to be transmitted), the model of Bianchi provides simple closed-form expressions to determine the transmission probability of a station and the system throughput in the stationary regime.

Specifically, the backoff procedure consisting in the decrease of the backoff counter of a station when sensing an idle slot, and in the increment and the reset of the contention window size respectively after a collision and after a successful transmission, is modelled by a two-dimension discrete-time Markov process $\{s(t), b(t)\}$. The process $s(t)$ models the current backoff stage of the station at time t, $CW_i = 2^i CW_{min}$; $s(t)$ takes values in the range $\{0, \ldots, m\}$, where m is the maximum backoff stage. On the other hand, $b(t)$ models the backoff counter of the station at the current stage CW_i and takes value in $\{0, CW_i - 1\}$. Time is discrete, and one step time corresponds either to the fixed length of a slot time, or to the variable duration of a transmission on the channel during which the backoff counter is frozen. Figure 2.2 illustrates the transition diagram of the chain.

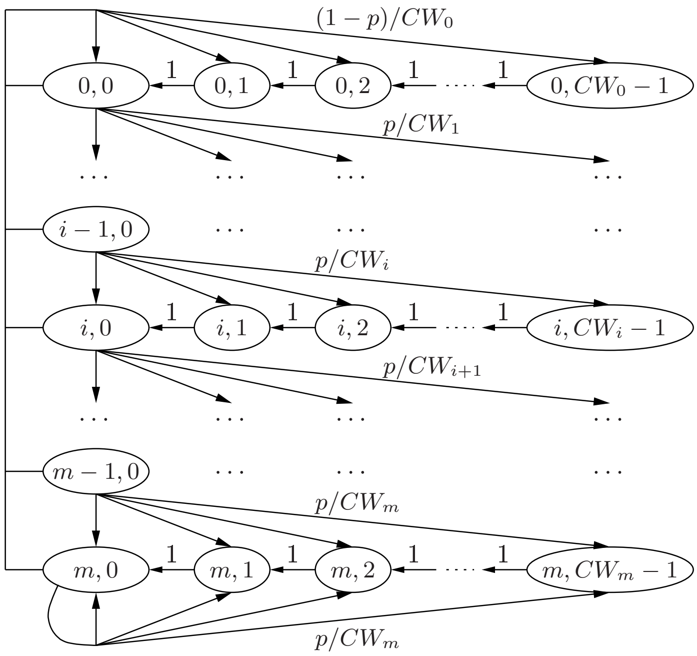

Figure 2.2: Markov chain for the DCF mechanism.

Let p denote the collision probability, i.e. the probability of a collision seen by the transmitting station. p is considered to be constant since the author of [5] is assuming ideal channel conditions with a fixed number of sources having the same collision probability; consequently, the markovian process is homogeneous. The transition probabilities are then given by:

$$\begin{array}{ll} P\{i,k|i,k+1\} = 1 & k \in (0, CW_i - 2), i \in (0, m) \\ P\{0,k|i,0\} = (1-p)/CW_0 & k \in (0, CW_0 - 1), i \in (0, m) \\ P\{i,k|i-1,0\} = p/CW_i & k \in (0, CW_i - 1), i \in (1, m) \\ P\{m,k|m,0\} = p/CW_m & k \in (0, CW_m - 1) \end{array}$$

An alternative expression for the transition probability $P\{i, k|i, k + 1\}$ has been introduced in [21] to take into account the fact that a station decreases its backoff counter only when there is no transmission heard on the channel. The previous transition probability can then be expressed as:

$$\begin{array}{ll} P\{i,k|i,k+1\} = 1 - p_c & k \in (0, CW_i - 2), i \in (0, m) \\ P\{i,k|i,k\} = p_c & k \in (0, CW_i - 2), i \in (0, m) \end{array}$$

where p_c represents the probability of a transmission on the channel; step time will thus be constant and equal to a slot time. However, since the introduction of the closed-form expression of p_c will complicate the formulas of the transmission probability and the system throughput, and thus the analysis, we will not consider in the following these expressions.

The stationary distributions of the markov chain is therefore given by:

$$\pi_{i,k} = \lim_{t \to \infty} P[s(t) = i, b(t) = k] \qquad k \in (0, CW_i - 1), i \in (0, m)$$

By deriving the expressions of the various stationary probabilities, and by normalizing their sum, the transmission probability of a station, which is the probability that the backoff counter of the station is equal to zero in whatever stage, can then be computed, and it is equal to:

$$\tau = \sum_{i=0}^{n} b_{i,0} = \frac{b_{0,0}}{1-p} = \frac{2(1-2p)}{(1-2p)(CW_{min}+1) + p\,CW_{min}(1-(2p)^m)} \tag{2.1}$$

τ depends on the collision probability p, the minimum value of the contention window size CW_{min}, and the maximum number of backoff stage m. On the other hand, the conditional collision probability p can be expressed in terms of the transmission probability τ and the number of active stations n since the probability of a collision seen by a station transmitting in a slot time is equal to the probability that at least one of the remaining stations transmits in that slot time. This gives:

$$p = 1 - (1-\tau)^{n-1} \tag{2.2}$$

2.3.1 Normalized System Throughput

The other metric that this model provides an expression for is the normalized system throughput, defined as the ratio of the expected time to successfully transmit a packet payload in a time interval to the expected length of that interval. The author of [5] expresses the normalized system throughput by considering the step time as the time interval, and hence:

$$Thr = \frac{P_s P_{tr} E[P]}{(1-P_{tr})\sigma + P_{tr} P_s T_s + P_{tr}(1-P_s)T_c} \tag{2.3}$$

where:

- P_{tr} is the probability that there is at least one transmission in a time slot: $P_{tr} = 1-(1-\tau)^n$;
- P_s is the probability of a successful transmission given that there is at least one transmission: $P_s = \frac{n\tau(1-\tau)^{n-1}}{1-(1-\tau)^n}$;
- $E[P]$ is the average transmission time of a packet payload;
- σ denotes the duration of a slot time;
- T_s is the average length of a successful transmission;
- T_c is the average length of a collision.

The previous expression of the throughput holds for both the basic access mechanism and for the RTS/CTS mechanism. However, the values of T_c and T_s depend on the employed access scheme.

Since we are considering only the basic access mechanism, we provide solely the expressions of T_c and T_s for the basic access mechanism. Hence:

$$T_c = T_{PHY} + \frac{\text{PDU}}{\text{data rate}} + DIFS + \delta \tag{2.4}$$

$$T_s = T_{PHY} + \frac{\text{PDU}}{\text{data rate}} + SIFS + \delta + \frac{ACK}{\text{data rate}} + DIFS + \delta \tag{2.5}$$

T_{PHY} represents the transmission time for the physical header, and PDU represents the size of the packet delivered to the physical layer by the MAC layer. PDU includes the payload of the packet in addition to the MAC header. δ is the propagation delay. Note that, if different sources use the same packet size, they will observe the same values of T_c and T_s.

2.3.2 Optimal Transmission Probability

The expression of the throughput given in Equation (2.3) is very interesting since it allows to obtain the optimal value of the probability of transmission that maximizes the normalized system throughput in the saturation condition. In fact, by deriving Equation (2.3) with respect to τ, one can obtain, after approximating $(1-\tau)^n$ to $1 - n\tau + \frac{n(n-1)}{2}\tau^2$ under the assumption of $\tau << 1$, the optimal value of τ that maximizes the saturation throughput of the system:

$$\tau \approx \frac{1}{n\sqrt{T_c/2\sigma}} \tag{2.6}$$

2.4 Insights on the Optimality of the Transmission Probability

In this section, we will provide an interpretation for the optimality of the transmission probability found in [5]. Actually, the optimality of the above value of τ can be explained by looking at the time interval separating two successful transmissions. In fact, the latter interval includes a successful transmission time and a random number of collision periods, separated by idle periods of random length (see Figure 2.3). This time interval is referred to as the virtual transmission time in [8], and its expression is given by:

$$virtual\_time = E[N_c]T_c + (E[N_c]+1)E[idle\_period] + T_s \tag{2.7}$$

where $E[N_c]$ and $E[idle\_period]$ represent respectively the average number of collisions and the average length of an idle period in a virtual transmission time. The expressions of T_c and T_s are those given by Equations (2.4)-(2.5). Considering fixed size packets, which is generally the case, T_c and T_s will become constant. The throughput of the system can thus be expressed as the ratio of the packet payload to the virtual transmission time.

The expressions of $E[N_c]$ and $E[idle\_period]$ are obtained by looking at the following equations. Let:

$$\begin{aligned} P_{collision} &= P\{\text{Transmitting stations} \geq 2 \text{ given} \\ &\quad \text{that there is at least one transmitting station}\} \\ &= \frac{1-(1-\tau)^n - n\tau(1-\tau)^{n-1}}{1-(1-\tau)^n} \end{aligned}$$

and

$$\begin{aligned} P_{success} = P_s = &\ P\{\text{Transmitting stations} = 1 \text{ given} \\ &\quad \text{that there is at least one transmitting station}\} \\ = &\ \frac{n\tau(1-\tau)^{n-1}}{1-(1-\tau)^n} \end{aligned}$$

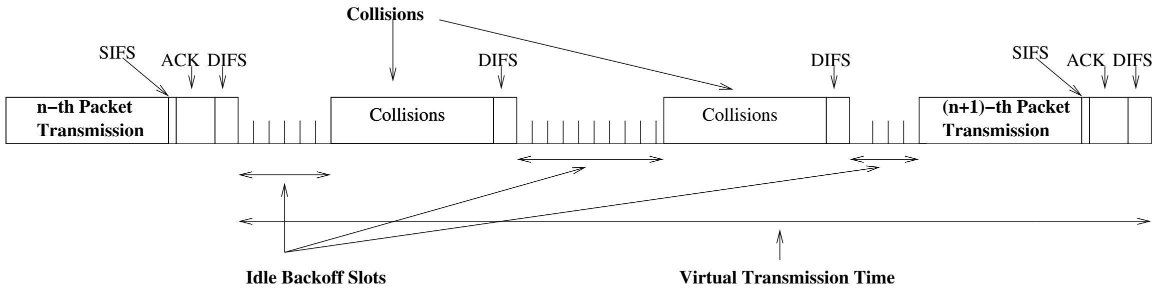

Figure 2.3: An illustration of the virtual transmission time.

Therefore, the distribution of the number of collision in a virtual time is equal to:

$$P[N_c = i] = P^i_{collision} P_{success} \qquad \text{for} \quad i = 0,\, 1\,, 2\,, \ldots$$

and hence, $E[N_c]$ is equal to:

$$E[N_c] = \sum_{i=1}^{\infty} i P^i_{collision} P_{success} = \frac{P_{collision} P_{success}}{(1 - P_{collision})^2}$$

Similarly, let:

$$P(\text{zero station transmitting in a slot time}) = P_{idle} = (1-\tau)^n$$

and

$$P(\text{at least one station transmitting in a slot time}) = P_{tr} = 1 - (1-\tau)^n$$

Hence, $E[idle\_period]$ is equal to:

$$\begin{aligned} E[idle\_period] &= \sigma \sum_{i=1}^{\infty} i P^i_{idle} P_{tr} = \sigma \sum_{i=1}^{\infty} i((1-\tau)^n)^i (1-(1-\tau)^n) \\ &= \sigma \frac{(1-\tau)^n}{1-(1-\tau)^n} \end{aligned}$$

where σ is the slot time duration.

Equation (2.7) includes three terms. The first term corresponds to the total time spent in collisions between two consecutive successful transmissions; the second one corresponds to the total idle time between two consecutive successful transmissions, and the third one is constant and corresponds to the packet transmission time. The first and second terms are respectively monotone increasing and monotone decreasing functions for $0 < \tau < 1$. The intersection of the two functions corresponds on the x-axis to the optimal transmission probability defined in Section 2.3.2. The optimal value of τ represents thus a trade-off between minimizing the time spent in collisions in a virtual transmission time and hence the number of collisions in that virtual time, and keeping a small value of the average wasted idle time. This trade-off is achieved by equating these two quantities. Note also that for different values of the number of active stations n, the intersection points have approximately the same value on the y-axis (see Figure 2.4). This implies that the maximum value of the saturation throughput of the system is almost independent of the number of active stations, and it can be reached in different network topologies by well tuning the transmission probability of the stations to the optimal transmission probability for each network topology.

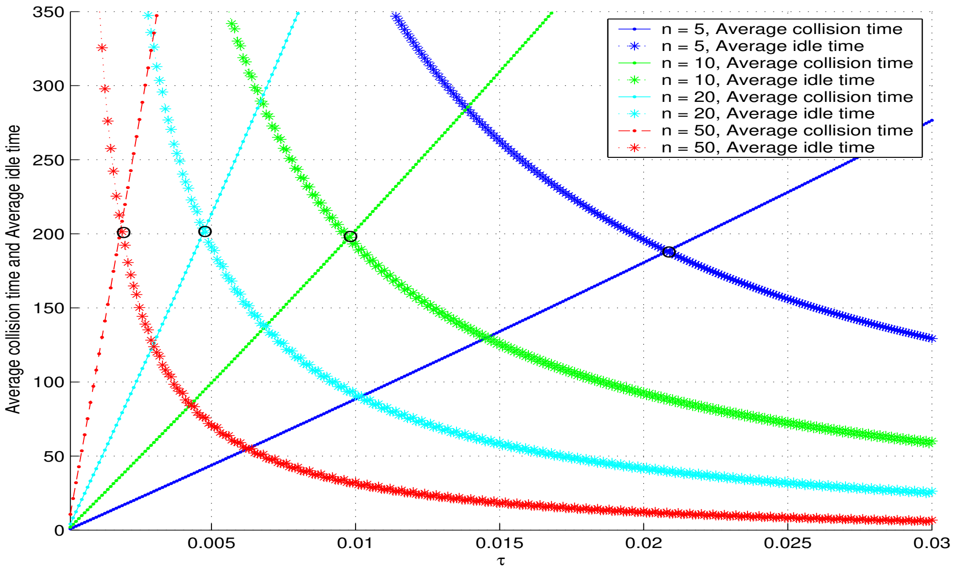

Figure 2.4: Average collision and average idle times between two successive transmissions for different values of n ($T_c = 4335\ \mu s$ and $\sigma = 20\ \mu s$).

2.5 Conclusion

In this chapter, we gave a brief description of the IEEE 802.11 standard and its DCF access mechanism. We also presented an analytical model that allows the derivation of some important system metrics. In the next chapter, we will present some related work on the dynamic tuning of the contention window.

Chapter 3

Overview of Dynamic Contention Window Algorithms

The binary exponential backoff algorithm introduced in the previous chapter has two major drawbacks:

- First, the contention window is increased only if the transmission is not successful (e.g. collision, error on the channel, buffer overflow at the receiver).
- Second, after a successful transmission of a packet, the contention window is reset to CW_{min}, thus forgetting its knowledge of the current congestion level in the network.

These two points are at the base of the inefficiency of the DCF mechanism. Actually, in the case of congested networks, the best contention window size for each packet is obtained at the cost of several collisions. These collisions are necessary to estimate the current congestion level in the network, consequently, wasting channel bandwidth and decreasing system throughput.

Several research works have proposed enhancements to improve the performance of the backoff algorithm of the IEEE 802.11 standard in the case of congested network by trying to provide solutions for the above two points. Other recent studies proposed innovative distributed contention-based MAC protocols like the *Fast Collision Resolution* (FCR) algorithm ([15]) and the *General Contention window Adaptation* (GCA) algorithm ([22]) as a solution for dynamic MAC algorithms in wireless LANs. In this chapter, we will present and discuss briefly four proposed algorithms to adapt the backoff window for the 802.11 standard. More analyses and discussions on two of them will be presented later in Chapter 5.

3.1 Adaptive Backoff algorithm of Bianchi *et al.*

Published in 1996 by Bianchi *et al.*, this algorithm is considered among the earliest proposed algorithms for contention window's control [4]. The idea of the algorithm is simple and based on a single contention window whose size is updated after each transmission attempt based on an estimation of the number of stations. Specifically, each station willing to transmit chooses a random number in the interval $[0, CW-1]$ and starts decreasing it as in the standard until reaching zero. CW represents the optimal contention window size that maximizes the system throughput, and it is found to be equal to

$$CW = \bar{n}\sqrt{2T}$$

under the assumption that all stations use the same contention window size CW. T is the total packet transmission time expressed in slot times, and $\bar{n}$ is an estimation of the current number of stations. The estimation of n is achieved by the use of a the following filter:

$$\bar{n}(t+1) = \alpha \bar{n}(t) + \frac{(1-\alpha)}{q} \sum_{i=1}^{q} \hat{n}_i$$

The parameters α and q are chosen heuristically, and are set respectively to 0.8 and 10. The measured n, noted as $\hat{n}$, is computed by using another formula that includes the current value of the contention window size CW

$$\hat{n} \approx 1 + \frac{E[c(B)](CW+1)}{2B} \tag{3.1}$$

where $E[c(B)]$ represents the average number of busy slots observed during the backoff decrease period B. However, $E[c(B)]$ is set equal to the current number of observed busy slots in the period B. As it is shown, the current **measured** value of n depends on the old value of CW, and hence on the old estimated value of n, which will make the new estimated value erroneous if the old one was inaccurate too, and therefore, the estimation will be unstable. This latter feature imposes to the authors the introduction of another function $s(n) = (1 + h/\sqrt{n})$ with another parameter h set also heuristically to the value of 2.

3.2 Adaptive Backoff algorithm of Calli *et al.*

Appeared in 1998, this study proposed a distributed algorithm to optimize the throughput of the system in a dynamic way [8, 9]. In particular, by establishing an analogy between the p-persistent IEEE 802.11 protocol and the standard protocol under the condition of equal average backoff time, the authors infer the average backoff interval that should be adopted in a given network configuration to optimize the throughput of the system. In fact, a pure p-persistent protocol assumes slotted channel, and works as the following: when the sender has a packet ready to be transmitted and senses the channel idle, it does so with probability p. Otherwise, it waits until the next slot with probability $1-p$. A p-persistent 802.11 protocol inherits the mechanisms of the 802.11 protocol, with the difference that the backoff interval is hence sampled geometrically with parameter p. The authors have validated their analogy assumption through simulation results and showed that the 802.11 standard behavior can be approximated by studying the behavior of the p-persistent 802.11 protocol.

The geometric distribution with the parameter p for the backoff interval of the p-persistent 802.11 protocol allows the authors to get the following basic relationship:

$$p = \frac{1}{E[B]+1} \tag{3.2}$$

which relates the persistence probability p to the average size of the backoff interval $E[B]$. By developing an analytical model for the p-persistent 802.11 protocol, they obtain the expression of the throughput in terms of p, and derive after the optimal value of p which maximizes the throughput of the p-persistent protocol, and hence the optimal average backoff time by using the above equation. Due to the analogy assumption they made, they propose to apply the obtained results to the standard protocol by using a single backoff window with the size set to the previously obtained optimal average backoff time.

Note that the computation of the optimal value of p requires the estimation of the number of current active users $\bar{n}$. By exploiting their analytical model, the authors get the following relationship:

$$\hat{n} = \frac{1-p}{p.Total\_idle\_p} \tag{3.3}$$

which expresses the **measured** number of users $\hat{n}$ in terms of the previous persistence probability p and the average number of idle slots in a virtual transmission time denoted as $Total\_idle\_p$. As in the previous algorithm, the persistence probability is computed based on the estimated $\bar{n}$, and thus the instability problem of the estimation will also be found in this algorithm.

3.3 Multiplicative Slow Decrease Algorithm

Another interesting and simple to implement contention window control approaches are the slow contention window decrease algorithms that we discuss in this section and in the next one. The idea is simple and efficient: the current size of the contention window is decreased slowly upon a successful transmission by some multiplicative or additive factor. This allows to keep some sort of knowledge on the current congestion level in the network by keeping the contention window size close to the previous one, showed to be convenient for the current congestion level. However, there are some issues related to the slow decrease schemes that will be discussed later on.

The earliest algorithm of these approaches is the multiplicative slow decrease algorithm [1, 18] appeared in 2003. An extended version of the work appeared later in 2004. The algorithm uses the same mechanism as the standard backoff, but instead of resetting the contention window to CW_{min} after a successful transmission, the new contention window is set as:

$$CW_{new} = \max(CW_{min}, D \times CW_{old}) \tag{3.4}$$

where D is the decrease factor chosen in the range $[0, 1]$, and set constant for all stations. The choice of D influences greatly the performance of the algorithm, and constitutes a trade-off between avoiding packet collisions by keeping the value of the CW closer to the previous one, and risking high access delay due to the fact that the previous contention window size is not necessary the optimal size. Specified initially as a power of 2 to meet industrial requirements due to hardware ease of implementation, and to simplify mathematical analysis ([18]), the decrease factor is:

$$D = \frac{1}{2^g} \tag{3.5}$$

where g is a positive constant integer > 0. The authors found later in [1] that maximum system throughput corresponds to decrease factors, D, greater than 0.5, which corresponds to the minimum integer value of $g = 1$. The maximum throughput is found to be achieved with a value of $D = 0.9$. However, as we mentioned above, the cost of the response time of the system, referred to as the settling time, will be high at that value of D [1].

It is worth noting that an analytical model of the algorithm has been specified based on the one of Bianchi introduced in Section 2.3.

3.4 Additive Increase Decrease Algorithm

The second interesting and recent algorithm of slow window decrease approach is the additive increase decrease algorithm [13]. Like the previous algorithm, this one also uses the same mechanism as the standard but introduces two new features. Specifically, upon a missed ACK, the current contention window is increased by a constant value ω, set heuristically in the simulation analysis provided in the article to 32. Upon a successful transmission, CW is decreased by ω with probability $1 - \delta$, and with probability δ the contention window is not changed. δ is also set heuristically to 0.4943 in the analysis provided. Note also that the analytical analysis carried out was also based on Bianchi's model but it considers the alternative expression of the transition probability $P\{i, k|i, k + 1\}$ that was introduced in [21].

3.5 Conclusion

In this chapter, we presented four adaptive algorithms to control the contention window of the backoff algorithm. The first two algorithms are based on a single contention window where the size is updated after each transmission, based on a run-time estimation of the number of active stations. The other two algorithms use slow contention window decrease approaches, and window control is done in an automatic way based only on local information concerning successful or unsuccessful transmissions. Further discussions on these algorithms will be provided later in Chapter 4 with special emphasis on the slow decrease approaches in Chapter 5.

Chapter 4

Proposed Approach for Adaptive Binary Exponential Backoff

As we showed in the previous chapter, there are several approaches to control the behavior of the backoff algorithm of the IEEE 802.11 for better performance in terms of throughput and delay – we have not discussed the performance in terms of fairness which is left for future research.

The first two approaches, based on an estimation feedback of the current number of active users, are very attractive in terms of fast adaptation to the change in the network state and have shown good results in terms of system throughput. However, they exhibit some deficiencies in estimating the number of active stations, and the packet access delay may become large. Actually, as shown in Equation (3.1) (Section 3.1) and in Equation (3.3) (Section 3.2), the new **measured** value of n depends on a parameter calculated based on the old estimated value of n; this implies that if the old value of n was erroneous, the new measured value of n will be erroneous too and thus the estimation will be unstable. Another point concerning the estimation of n in the algorithm of Calli *et al.* is the parameter *Total_idle_p* (see Equation (3.3)) which represents the average number of idle slots sensed in a virtual transmission time. Observe that a station cannot know whether a transmission not destined to it is successful or not. Therefore, it is not able to determine the boundary of a virtual transmission time and the parameter *Total_idle_p* will not be measured accurately. Thus, a further error is introduced in the estimation of n.

On the other hand, slow contention window decrease algorithms are efficient and simple to implement. The simulations results obtained showed great performance in terms of throughput and packet access delay when the number of active stations is in steady-state or near steady-state. However, their performance in transient situations with sharp increase in the number of entering or leaving active stations or in noisy environments degrades rapidly. This is due to the fact that the control of the contention window in these approaches is done automatically in a blind way independently of the effective number of users, and it is based only on information such as successful or unsuccessful packet transmissions. In addition, the increase and decrease factors are set heuristically by the algorithms to constant values, and are independent of the network state.

In summary, a good adaptive algorithm is a one that includes the following features:

- High throughput, access delay and fairness performance compared to those of a cell with a single station while keeping the protocol simple.
- Fast response to changes in the network state.
- The previous point necessitates a good and simple estimation of the number of users that

should avoid the shortcomings of the algorithms presented above.

- Robustness to errors on the wireless medium.
- Insensitiveness to the operation mode whether in ad-hoc or infrastructure mode.

Setting these points as mandatory constraints in our design, the algorithm that we propose complies to a large extent to the above-mentioned points.

4.1 Adaptive Binary Exponential Backoff Approach

As explained before, each station adopting the binary exponential backoff and willing to transmit, will transmit its packet after a random amount of time, whose average length depends on the average number of collisions suffered for that packet. The latter number is network state dependent, and it is rapidly increasing with the number of active stations (see Figure 4.1).

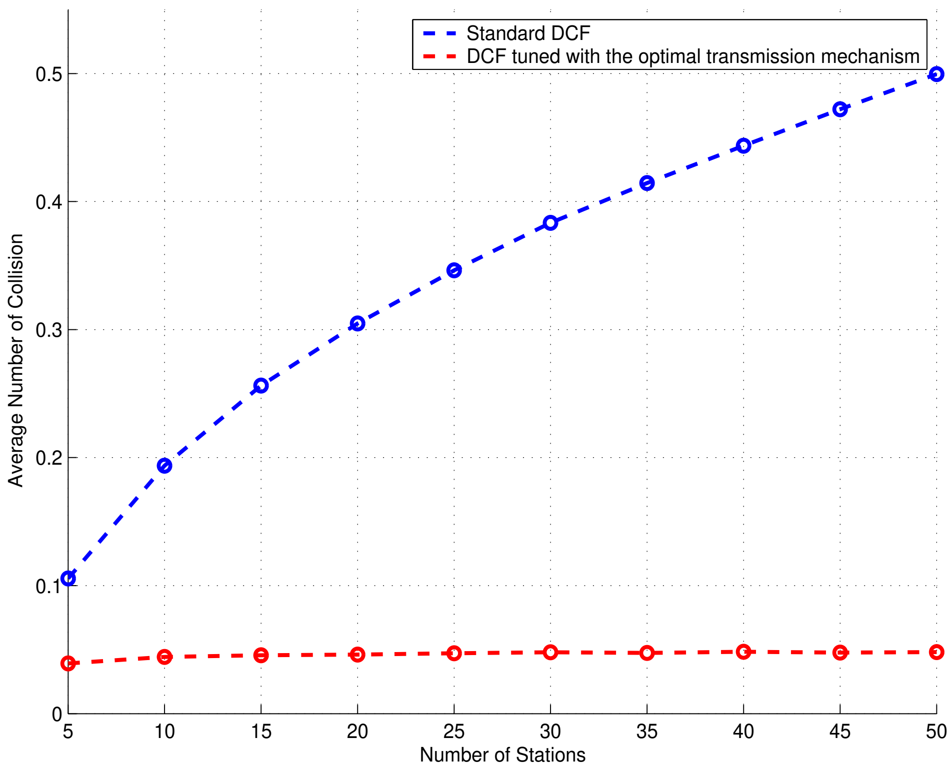

Figure 4.1: Analytical average number of collision between two successive transmissions as a function of the number of stations.

The main reason for this behavior is the reset of the contention window size to a small value CW_{min} after a successful transmission. While starting with a small window size would be convenient in small cells[1], it is severely penalizing the system performance in congested cells where an appropriate time-spread of the users access time over time is not respected nor guaranteed, hence the high number of collisions and the descent in the system performance. This fact explains the efficiency of the simple slow window decrease approaches where the initial window size set for the

[1]When we refer to the size of a cell in our report, we actually refer to the number of active stations in that cell and not to the physical size of that cell.

next packet transmission is closer to the value of the previous one. However, as we mentioned before, these latter techniques suffer from slow reaction to network state changes and from errors on the wireless medium since the feedback for the window control is unaware of the effective current size of the cell. Consequently, the need for an estimation of the number of active stations turns out to be essential.

These last points motivate the idea of our algorithm that can be summarized as follows: After a successful transmission, the initial window size of the binary exponential backoff mechanism, CW_{min}, is set to an optimal value based on an estimation of the actual active stations in the cell. The optimality of this value is computed in such a way to maximize the system throughput and to minimize the packet access delay. Extension of the optimality to include other metrics such as fairness could be envisioned.

4.2 Computing the Optimal Initial Contention Window

Throughout the analysis done in [5], the authors pointed out that the maximum system throughput can be achieved for a given network configuration by well adjusting the size of the initial contention window (see Figure 4.2). Also, as we mentioned in Section 2.3.2, optimal throughput and packet

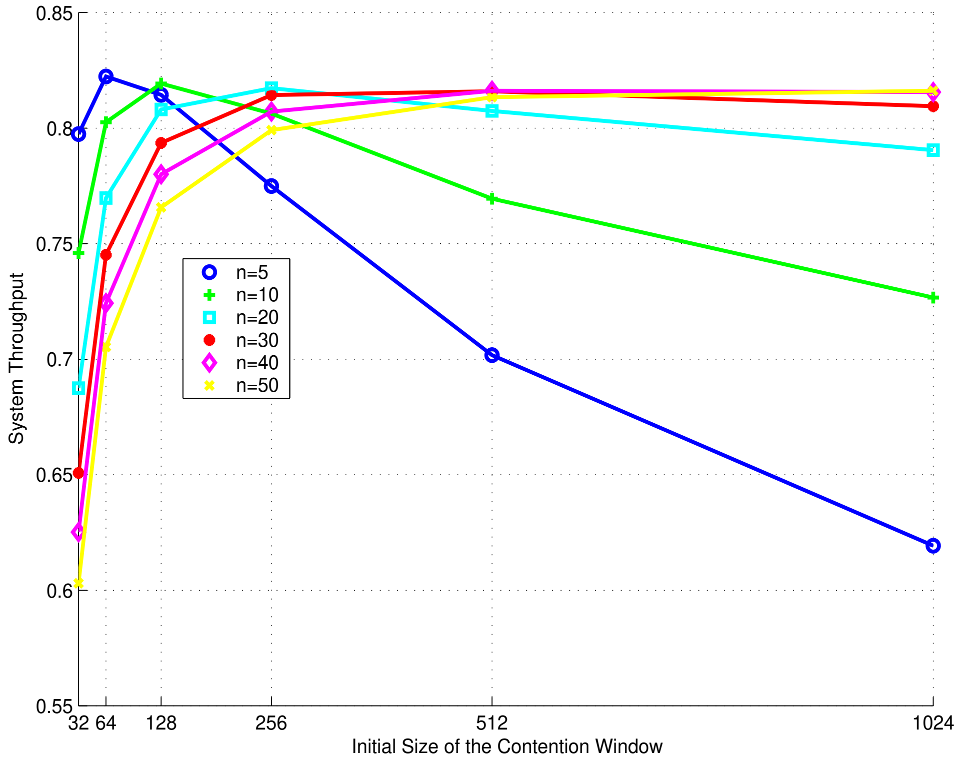

Figure 4.2: System throughput as a function of the initial contention window size and the number of users (data rate of 2Mbps).

access delay of the system are achieved for the following value of the transmission probability:

$$\tau \approx \frac{1}{n\sqrt{T_c/2\sigma}} \tag{4.1}$$

which is a function of the number of active users n. This implies that if we can estimate the number of active users for a given network configuration, we can then know the optimal transmission probability of the stations. But it remains to answer how the stations should behave in order to transmit with that probability. In other words, we need to find the relation between the optimal probability and the initial size of the contention window. To obtain this relationship, we will first consider the computation of the average size of the contention window $E(CW)$ of the binary exponential backoff mechanism for a maximum number of stages m.

We will consider the assumptions made in [5] of ideal channel conditions. We will consider n active stations in a given cell each with an infinite load traffic. Each station uses the 802.11 standard with CW_{min} as the minimum size of the contention window noted hereafter as CW_0, CW_{max} is the maximum window size, and m is the maximum backoff stage, i.e. $CW_{max} = 2^m CW_{min}$. Let p denote the conditional collision probability which represents the probability of a collision seen by a packet being transmitted on the channel.

Consider the following discrete-time Markov chain that describes the evolution of the contention window of a station at temporal steps $j = 0, 1, 2, \ldots$, and let $\{X_j, j = 1, 2, \ldots\}$ denote the collision process seen by a particular node. At temporal steps $j = 1, 2, 3, \ldots$ the source station is informed whether the packet is lost in the temporal interval $[j-1, j]$ i.e. $X_j = 0$, or it is well received i.e. $X_j = 1$. Based on the approximation made in [5] on the independence of the collision probability, $\{X_j, j = 1, 2, \ldots\}$ are i.i.d. random variables having the following probability distribution:

$$Pr(X_j = 0) = p$$
$$Pr(X_j = 1) = 1 - p$$

where p is the conditional collision probability, assumed to be constant as in [5]. The contention window evolution process $\{CW(j), j = 0, 1, 2, \ldots\}$ can then be expressed as follows:

$$CW(j) = \begin{cases} CW_0 & \text{if } X_j = 1 \\ \min(2CW(j-1), CW_{max}) & \text{if } X_j = 0 \end{cases}$$

This process is a discrete-time homogeneous Markov chain taking value in the range $\{CW_0,\ 2CW_0,\ \ldots, 2^m CW_0\}$ – here we assume that after reaching the maximum value CW_{max}, the station continues retransmitting infinitely. Its steady state distribution will be $\pi(i) = \lim_{j\to\infty} Pr(CW(j) = 2^i CW_0)$, and hence we will have:

$$\begin{aligned} \pi(0) &= 1 - p \\ \pi(i) &= (1-p)p^i \\ \pi(m) &= p^m \end{aligned}$$

At steady-state, the average size of the contention window will be given by:

$$E(CW) = \sum_{i=0}^{m} \pi(i)\, 2^i CW_0$$

which will result in

$$E(CW) = \frac{CW_0(1 - p(1 + (2p)^m))}{1 - 2p} \tag{4.2}$$

Recall the expression found for the transmission probability obtained in [5]:

$$\tau = \frac{2(1-2p)}{(1-2p)(CW_{min}+1) + p\,CW_{min}(1-(2p)^m)} \tag{4.3}$$

After some algebraic manipulations in Equations (4.2)-(4.3), we obtain the following relation:

$$E(CW) = \frac{2}{\tau} - 1 \tag{4.4}$$

4.3 The Adjustment Algorithm

The latter formula is quite interesting. It allows to express the average window size in terms of the transmission probability τ. This implies that if a node can estimate the number of stations in the cell, it can know then the optimal value of τ, and hence, the optimal average window size that it should apply. As our objective is to adjust the minimal size of the contention window, dynamic control of the average contention window will thus be implemented by varying the minimum value of the contention window CW_0.

The algorithm will be the following:

- *Estimate the number of active users.*
- *Compute the optimal value of τ:* $\tau \approx \frac{1}{n\sqrt{T_c/2\sigma}}$
- *Compute the probability of collisions p:* $p = 1 - (1-\tau)^{n-1}$
- *Compute the optimal average contention window size according to Equation* (4.4).
- *Infer the value of CW_{min} from Equation* (4.2).
- *Set the new value of CW_{min} from the set of values that a CW can take in the standard by using the following equation:*

$$newCW_{min} = \begin{cases} 2^{i-1}CW & \textit{if } CW_{min} < \frac{2^{i-1}CW+2^{i}CW}{2} \\ 2^{i}CW & \textit{if } CW_{min} > \frac{2^{i-1}CW+2^{i}CW}{2} \end{cases}$$

- *Update the value of m:* $m = \log_2 \frac{CW_{max}}{CW_{min}}$

Note that the reason for making the value of CW_{min} discrete, i.e. belonging to the standard set of values, is to stay in agreement with the analytical model of Bianchi on which we based our analysis. On the other hand, the value of CW_{max} stays constant, while the value of m reflects the size of the starting window. We have assumed that the maximal value of the contention window is constant and equal to 1024. This value fits well with a cell with a maximum number of contending stations of 50. Therefore, for larger number of contending stations, one should consider larger CW_{max} size.

The analytical throughput achieved by our adaptive algorithm can thus be obtained by computing the sub-optimal value of τ that corresponds to the new discrete value of CW_{min}. In fact, by substituting the new values of CW_{min} and m in Equation (4.3) and by using the optimal value of p – as the value of p is very small, the difference between the optimal and the sub-optimal values of p is negligible – we will get the sub-optimal value of τ, and therefore the throughput of our algorithm by using Equation (2.3).

4.4 Estimating the Effective Number of Contending Stations

Contrary to previously proposed methods in [4, 6, 8] which estimate the current number of active stations by measuring the channel activity, e.g. by counting busy and idle slots, we propose to estimate the number of active stations directly by counting as active each station from which the concerned measuring station receives a *sign of life* of it in the period of two successive transmissions.

Actually, previous estimation methods have shown some instability problems, as a consequence of measuring the number of stations indirectly, in particular by using parameters that depend on previous estimations of n, which introduces errors on the future measured values (see Equations (3.1) and (3.3)).

In our algorithm, we propose to implement a cache for counting the number of active stations similar to the sequence number cache already specified and implemented in the current standard[2]. In our simulations, we took advantage of the actual implementation of the cache sequence number to implement our proposed cache. The idea can be summarized as follows: since each active node in a cell is always filtering received packets whether they are destined to it or not[3], each node can thus keep trace of a large number of active nodes in a given time interval by counting the number of received signs of life from these nodes, namely error-free data packets or RTS packets, in that time interval. Estimating the number of stations from the measured value is done by using an Auto Regressive Moving Average (ARMA) filter. A stronger but more complex to implement filter, namely a Kalman filter, is proposed in [3, 6] to estimate the size of active stations. However, since our objective is to implement a simple, adequately accurate and trackable estimation algorithm, we adopt the ARMA filter. In fact, as we will show later, the estimation of the number of stations obtained by using this filter combined with the discretization of the new optimal value of CW_{min}, lessens the need of a highly accurate estimation. Specifically, in our implementation, we use the following simple filter:

$$\bar{n}(j+1) = \alpha\bar{n}(j) + \frac{(1-\alpha)}{q}\sum_{i=1}^{q}\hat{n_i} \tag{4.5}$$

where $\bar{n}(j)$ is the estimated value of the number of stations at the temporal steps j, and $\hat{n_i}$ are the last measured samples of the number of stations. The parameters α and q are set heuristically to 0.3 and 3 respectively.

The choice of these parameters constitutes a trade-off between minimizing the fluctuations in the estimation (value of q), and fast tracking of network state changes (value of α). The time step length in which we obtain $\hat{n_i}$, the measured samples of n, is set to the time period between two successful transmissions of the current node. Note that this time interval is variable, and it changes from estimation to estimation. The effect of using the parameter q is to smooth the fluctuations of the measured value of n over these variable-length intervals. Note also that for a larger measuring time intervals, e.g. intervals taken between two or three contention window resets, better accuracy of the measured n is obtained; nevertheless, no major enhancement in the system performance metrics is observed. More discussions and analysis on the choice of the parameters and the time interval will be provided in Chapter 5.

4.5 Conclusion

In this chapter, we have presented the main contribution of the work: an algorithm for adjusting dynamically the binary exponential backoff mechanism. We started by describing the features and drawbacks of other proposed adaptive algorithms, then we gave a summary of the general features that we believe a good adaptive algorithm should have. After, we presented our algorithm which includes two parts: the first part comprises an estimation of the current active stations, and the second one consists in computing the optimal initial window size based on the obtained estimation.

[2]The objective of this cache is to drop duplicate received packets, i.e. packets that have the same destination and sequence number.

[3]The estimation might not work properly in case of enabling the WEP encryption.

Analysis and validation through simulations of the performance of the algorithm will be addressed in the next chapter.

Chapter 5

Simulation Results

In this chapter, we study the effectiveness of our proposal through a series of simulations. System throughput and end-to-end packet delay are investigated. Results obtained with our algorithm are compared with the analytic results and with those obtained with the standard and with the slow decrease algorithms. We also investigate several scenarios with different characteristics to illustrate the features of the algorithm under various conditions.

All simulations presented in this chapter were conducted in *Network Simulator 2*, patched with a code dated February, 2005 and proposed by M. Federico, University of Padova, to fix a bug in the event scheduler.

5.1 Simulation Parameters

In all simulations we ran, we claimed the following assumptions. We considered a maximum number of wireless active stations at the number of 50, uniformly distributed in a squared zone of $100\,\mathrm{m}\times 100\,\mathrm{m}$ in the range of each other, and operating in ad-hoc mode, unless specified. The choice of that number of maximal active stations is dictated by the maximal value of the contention window CW_{max} set to 1024. In order to stay in conformance with the assumption made in the analytical model of Bianchi, each station will be in saturation conditions whenever it becomes active, i.e. its buffer will not empty unless the station becomes idle. The buffer size of each station is considered to be small, and set to 5. Packet payload is set equal to 1050 bytes, and channel data rate is set to 2 $Mbit/s$. Unless specified, we consider CBR traffic sources generating packets at the rate of 200 $packets/s$. Only the basic access mechanism is considered, and the hidden node problem is omitted.

The physical characteristics are those for the DSSS technique, and the corresponding parameters are reported in Table 5.1. Note that for our algorithm, we set the value of the parameter T_c to a constant value corresponding to the maximum packet data unit (MPDU) delivered by the MAC layer to the PHY layer, and which is equal to 8184 bits in the DSSS system.

$$T_c = T_{PHY} + \frac{MPDU}{data\_rate} + DIFS + \delta = 243 + \frac{8184}{data\_rate}\ \mu s$$

The time step interval in which we measure the number of stations is set equal to the time period between two successful transmissions. Parameters q and α of the ARMA filter of Equation (4.5) are set respectively to 3 and 0.3.

Note also that in the implementations of the two slow decrease approaches, we used the parameters that were found to maximize the performance. Specifically, for the additive increase/decrease method, parameter ω is set to 32 and parameter δ is set to 0.81910. For the multiplicative slow

Parameter	Value
CW_{min}	32
CW_{max}	1024
Slot time σ	20 μs
SIFS	10 μs
DIFS	50 μs
Propagation time δ	1μs
PHY header	192 bits
MAC header	272 bits
ACK	112 bits + PHY header
MPDU	8184 bits

Table 5.1: DSSS system parameters.

decrease method, the decrease factor $D = \frac{1}{2^g}$ is set equal to 0.9.

In the following, we will refer to our proposed algorithm as the Adaptive Binary Exponential Backoff or "Adaptive BEB" for short. The additive increase decrease algorithm and the multiplicative slow decrease algorithm will be referred to respectively as additive and multiplicative.

5.2 Comparing Analytical and Simulative Throughput

In this section, we investigate on the validity of the analytical model. To this end, we will compare the analytical results with the ones obtained from simulations. In fact, Equation (2.3) allows the derivation of the optimal analytical throughput for a given network topology by substituting the value of τ in that equation with its optimal value for that topology. Recall from Section 4.3 that the analytical throughput of our algorithm, noted hereafter as the sub-optimal throughput, can be computed by using the sub-optimal value of τ obtained by using the new discrete value of CW_{min}.

Figure 5.1 plots the optimal analytical throughput, the sub-optimal analytical throughput and the simulative throughput of the algorithm[1] against the number of contending stations. Note that, for each value of the number of stations n, we have computed the mean value of the throughput over 100-second simulation runs for a topology with a fixed number of stations set to that value of n. The figure shows that the difference between the optimal and the sub-optimal analytical throughput is small. On the other hand, the difference between the sub-optimal throughput and the simulative throughput is in the order of 3% and increases to 5% for too small or too large values of n.

5.3 Steady and Transient Network State Analysis

In this section, we investigate the effectiveness of the standard and the different approaches versus dynamic changes in a wireless network. We will consider therefore three scenarios. First, we study the case of a fixed number of stations, then, we study a network with an abruptly varying number of active stations, and finally, we consider the scenario of a Poisson distribution for the stations arrival, with an exponentially distributed activity period for each station. Note that the access delay that we examine is the end-to-end delay seen at the transport level.

[1]We have used packets with size set to 950 Bytes to avoid packets fragmentation which might introduce errors on the throughput values.

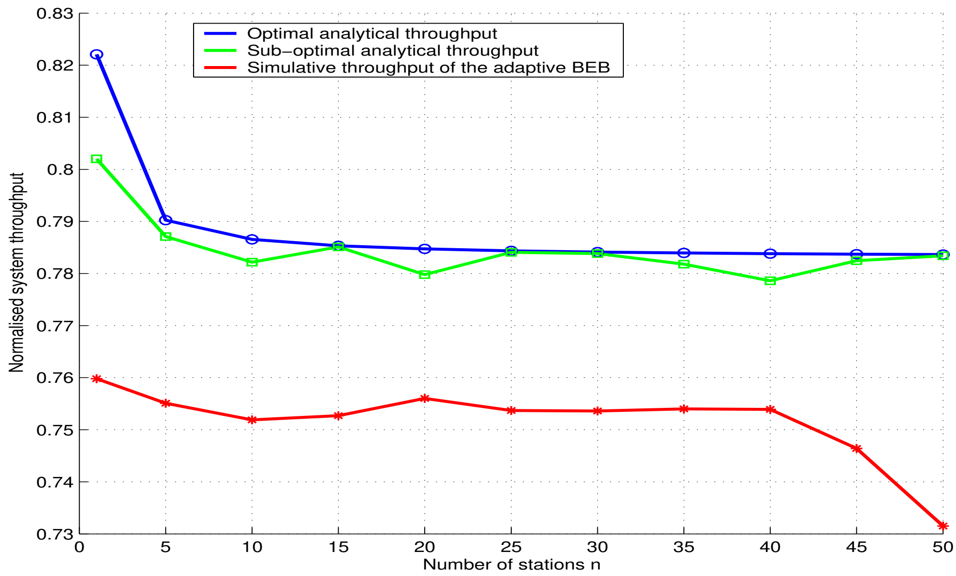

Figure 5.1: Analytical and simulative throughput of the adaptive BEB algorithm.

5.3.1 Fixed Number of Stations

In this scenario, we consider a fixed number of active stations which is 50. At time $10s$, all CBR traffic sources start transmitting, and at time $100s$, they stop. As shown in Figures 5.2 and 5.3, all three adaptive approaches have the same behavior in terms of throughput and packet delay. In addition, all three approaches outperform the standard, and this is due to its resetting scheme: each station starts the contention window increasing process all over again and has to suffer the same average number of collisions before reaching its optimal window size for the current network configuration. Even though the three adaptive approaches perform almost the same, the adaptive BEB is the fastest to react to the activation of the 50 contending stations at the 10^{th} second.

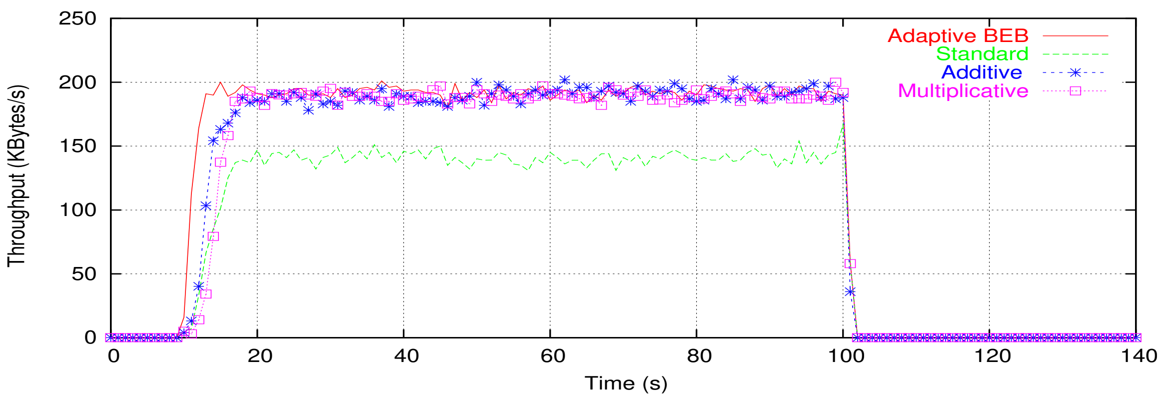

Figure 5.2: Overall throughput in the steady state scenario case.

5.3.2 Abrupt Change in the Number of Stations

In this scenario, we start with five stations at time $10s$, then we activate suddenly a burst of 25 stations at time $25s$, then, we add another burst of 20 stations at time $45s$. At time $65s$ and $80s$, we stop respectively the second and the third bursts, and at time $110s$, we stop the remaining five

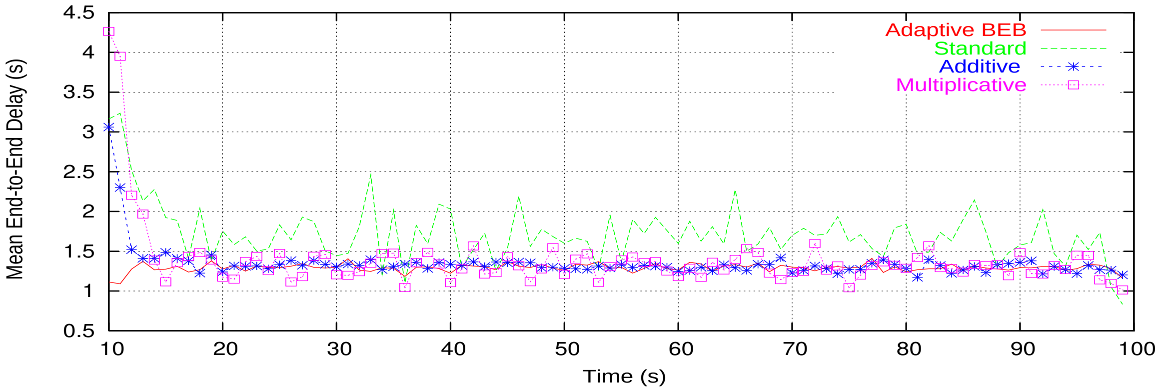

Figure 5.3: Mean delay in the steady state scenario case.

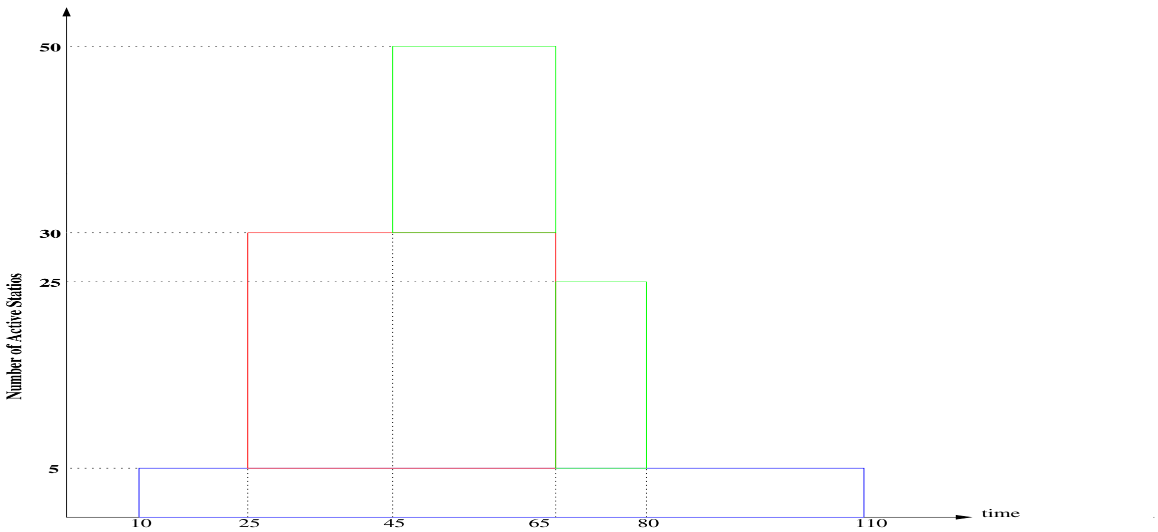

Figure 5.4: Temporal evolution of the number of active stations in the abrupt scenario case.

stations (see Figure 5.4). As we notice from the curves of Figures 5.5 and 5.6, upon the arrivals of the stations, the adaptive BEB is the fastest to react to the network change as it jumps quickly to a larger minimum contention window size, and consequently, avoids a certain number of collisions that affects the performance of slow CW approaches which suffer from a sharp transient period before reaching their steady window sizes again. However, the fact that the buffer of the different stations contain some packets after the end of their activities leads to an effective smooth decrease in the number of active stations. Therefore, the size of the contention window decreases smoothly upon stations departure, and no sharp transient period is observed at these instants.

5.3.3 Poisson Arrival with Exponential Activity Time

In this scenario, we consider a Poisson process for the arrivals of the stations at the rate of 5 $stations/s$, and an exponential activity time with mean $100s$. Figures 5.7 and 5.8 illustrate the throughput and the packet delay performance obtained when using the standard and the three algorithms. As in the previous scenario, we observe a sharp transient period at the beginning of the simulation for the standard, but also for the two slow decrease approaches due to the high rate of

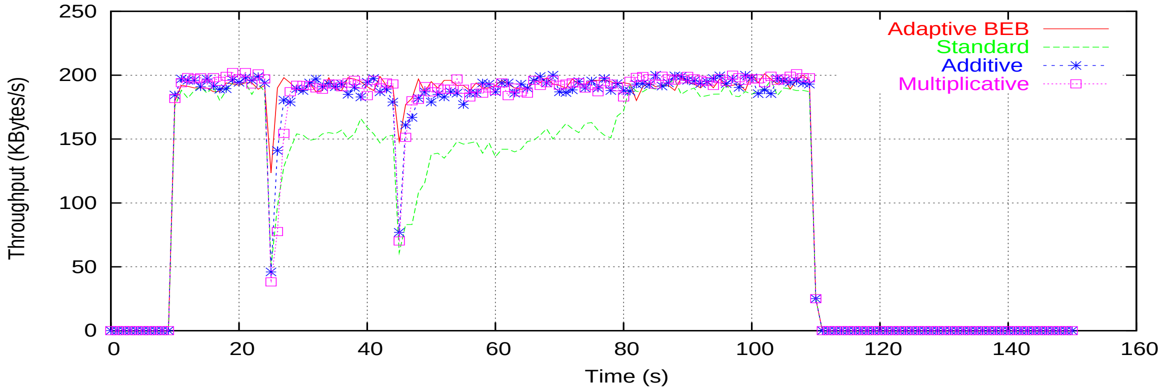

Figure 5.5: Overall throughput in the abruptly varying scenario case.

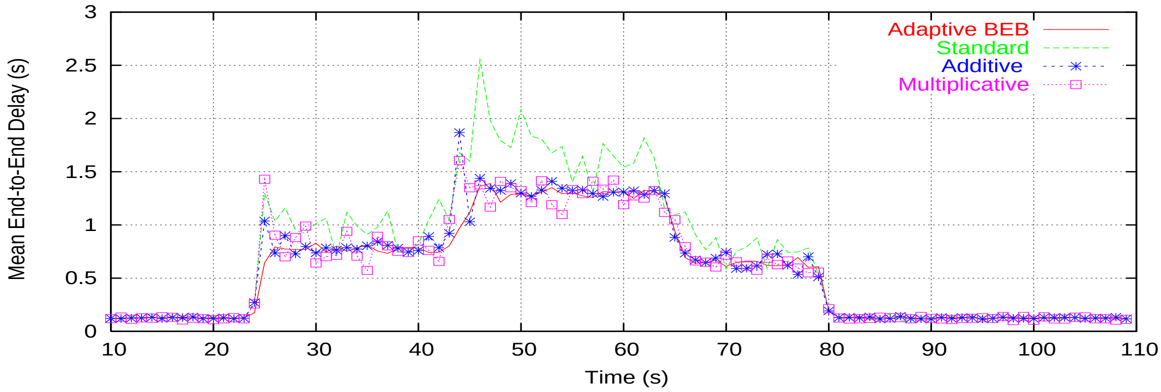

Figure 5.6: Mean delay in the abruptly varying scenario case.

stations arrivals and the slow response time of the standard and the two approaches to adequately control their contention window sizes. On the other hand, adaptive BEB reacts very well, and a small transient period is observed due to the time needed for all stations to estimate the number of active stations and tune their contention window accordingly.

Note also the performance of the standard when arrivals are Poisson and the activity periods are exponentially distributed. When the number of stations becomes greater than five – a value corresponding to an optimal window size of 32 which is the size of the standard's starting window – we observe a decrease in the performance of the standard. However, when the number of stations starts decreasing, the standard achieves an increased performance, and reaches again its maximum performance when the number of stations becomes close to five.

5.4 Parameters Analysis of the ARMA Filter

The ARMA filter introduced in Section 4.4 defines three parameters, namely α, q and the time step length of the estimation, that need to be selected in such a way to obtain a good estimation accuracy and fast filter response time to changes in the network. From the simulations conducted, we derive the following remarks:

- The filter memory α is mainly responsible for fast tracking of the changes in the network. Small values of α will lead to fast tracking capabilities of the filter; however, too small values

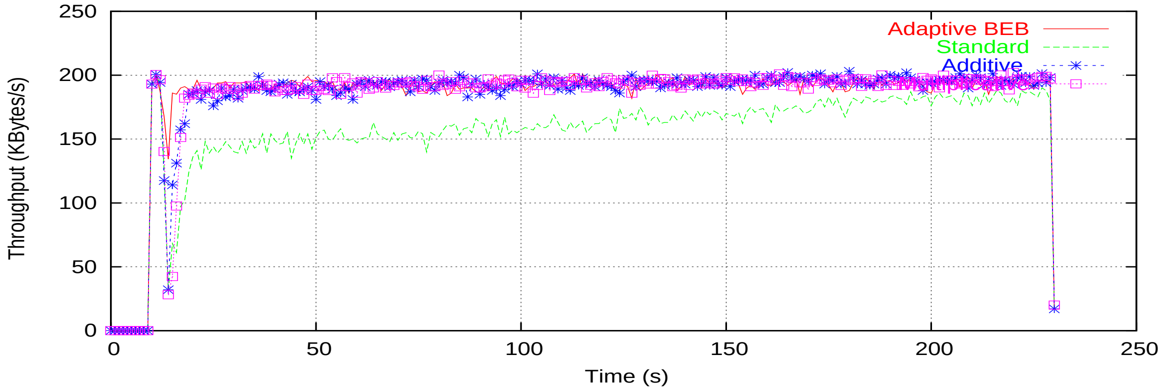

Figure 5.7: Overall throughput in the Poisson arrival scenario case.

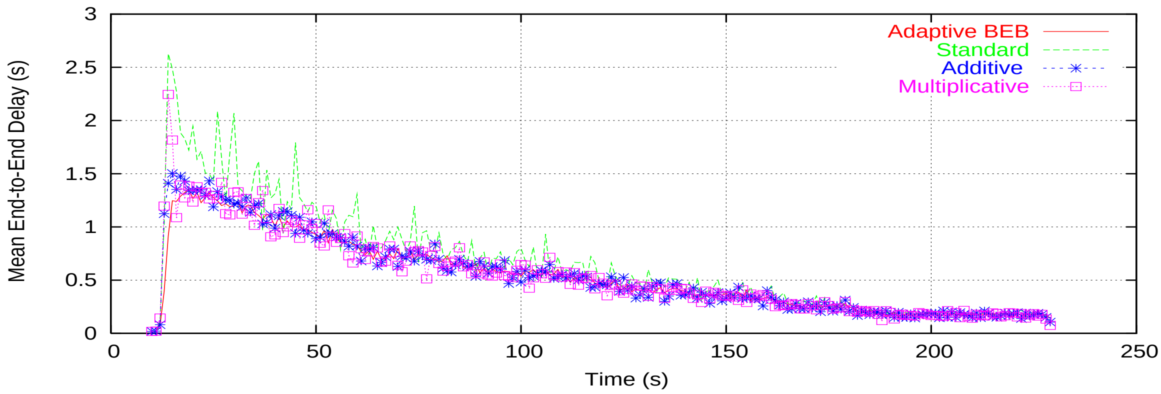

Figure 5.8: Mean delay in the Poisson arrival scenario case.

will lead to the appearance of fluctuations in the estimations. Consequently, we set this parameter heuristically to a trade-off value of 0.3.

- The parameter q allows to better smooth the fast fluctuations in the estimation resulting from sudden changes. It has a minor influence in the filter estimation. We set its value to 3 to take into account the last three measured samples.
- The length of the time step has a major influence on the precision of the estimation. Evidently, the larger the step, the better the estimation. However, a brief decrease in the tracking capability will be observed for large time steps just after sudden changes.

In the following simulations, we will investigate on the time step of the ARMA filter implemented in our algorithm. we want to quantify the impact on its length on the performance of the system under two different scenarios, more precisely a slow scenario and the abrupt scenario of Section 5.3.2. We will compare the performance of the system, the precision of the estimation and the temporal evolution of the contention window achieved with different time steps. We also compare the results with those obtained in the case of applying our algorithm when the number of stations is known.

5.4.1 Case of Slowly Increasing Number of Stations

The objective of this scenario is to explore the behavior of the estimator under near steady-state network configuration. We start the simulation at time $10s$, then we increase the number of active station by one every two seconds until reaching the maximum number of active stations, set to 50. As can be seen in Figures 5.9 and 5.10, the performance of the algorithm is insensitive to the length of the time step of the estimator despite the fact that the estimation of the number of stations obtained with small time steps differs substantially from the actual number of stations, up to 20% when $n = 50$ (see Figure 5.11). Note also that the temporal evolution of the starting window size obtained with small time steps is lower by one stage from the optimal starting window size (see Figure 5.12) during the simulation time.

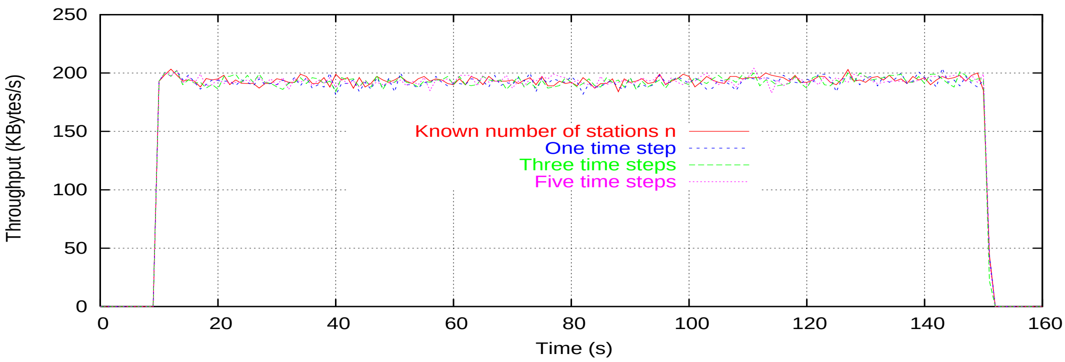

Figure 5.9: Overall throughput comparison in the slow scenario case.

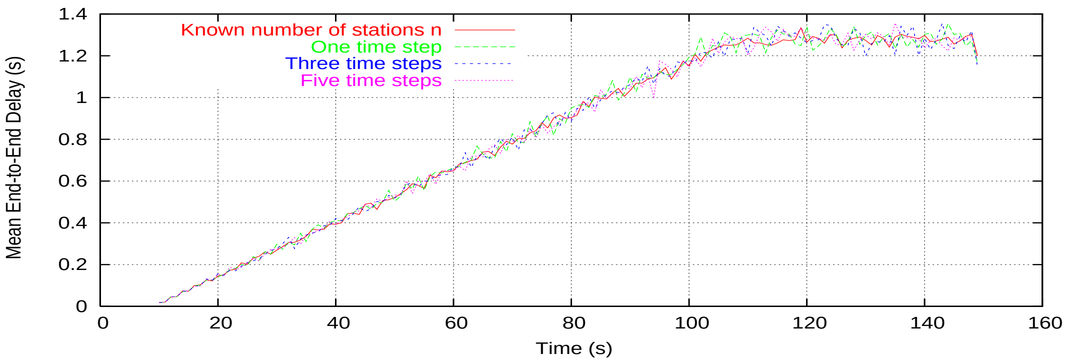

Figure 5.10: Mean delay comparison in the slow scenario case.

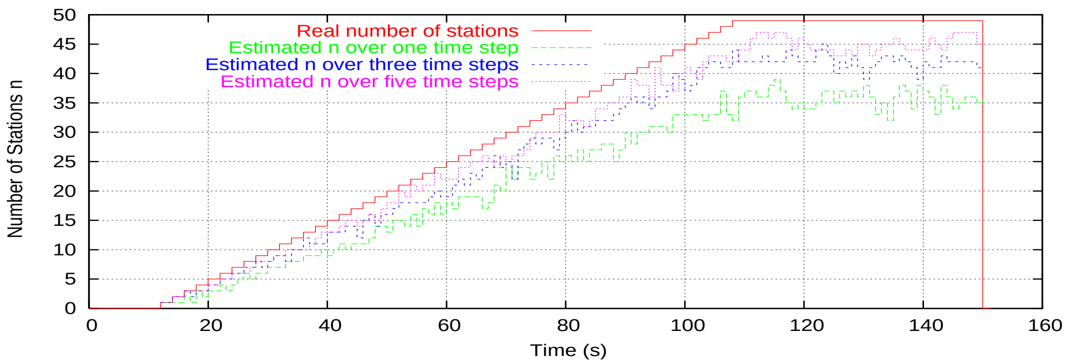

Figure 5.11: Comparision of different estimations of the number of stations in the slow scenario case.

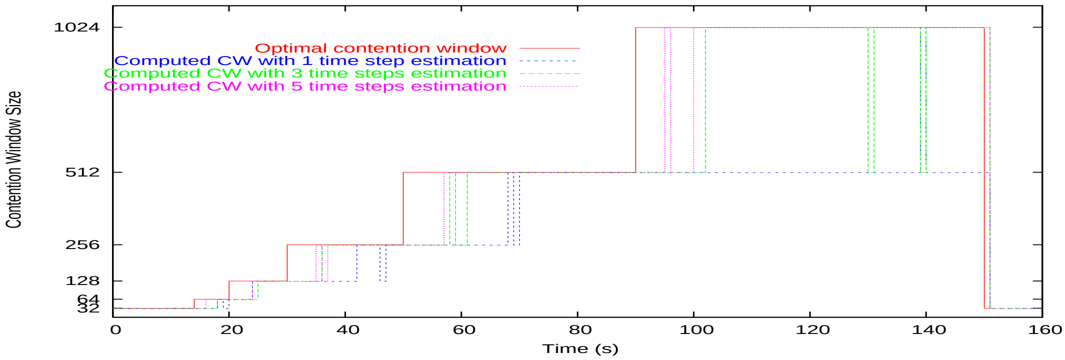

Figure 5.12: Contention window evolution comparison in the slow scenario case.

5.4.2 Case of Abruptly Increasing Number of Stations

We reused the Scenario of section 5.3.2 to analyze the performance of the estimator under an abruptly changing configuration. As in the slow scenario case, the performance obtained in implementations with different time step lengths of the estimator is almost similar to that obtained with an implementation with known number of stations (see Figures 5.13 and 5.14). This is attained despite the fact that an estimator with a small step length gives a lower precision and a lower starting window size than those of the real and optimal values (see Figures 5.15 and 5.16). As a result, and for sake of simplicity, we consider one time step as a sufficient measuring period for a satisfactory operation of the algorithm.

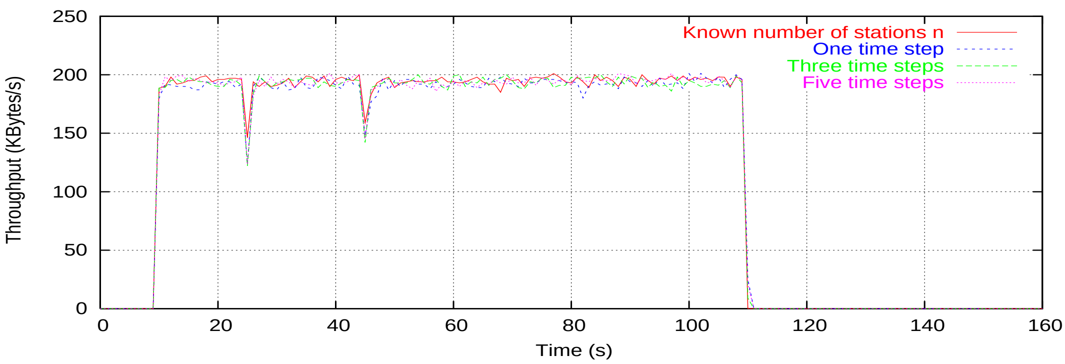

Figure 5.13: Overall throughput comparison in the abrupt scenario case.

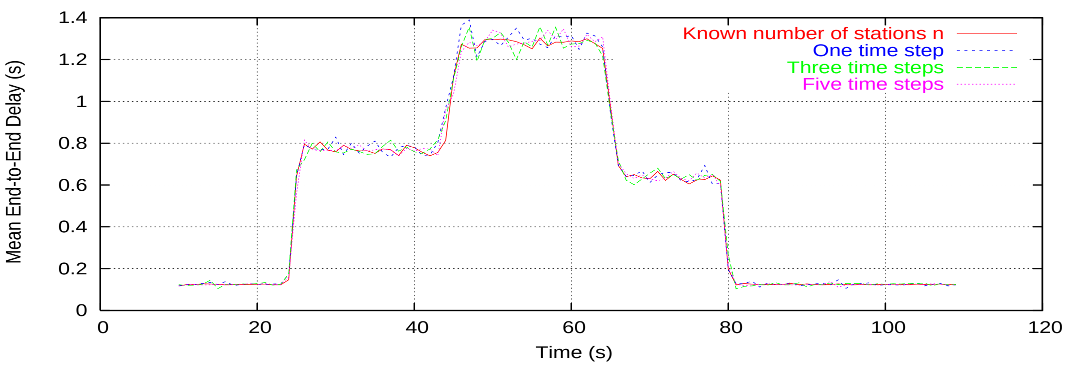

Figure 5.14: Mean delay comparison in the abrupt scenario case.

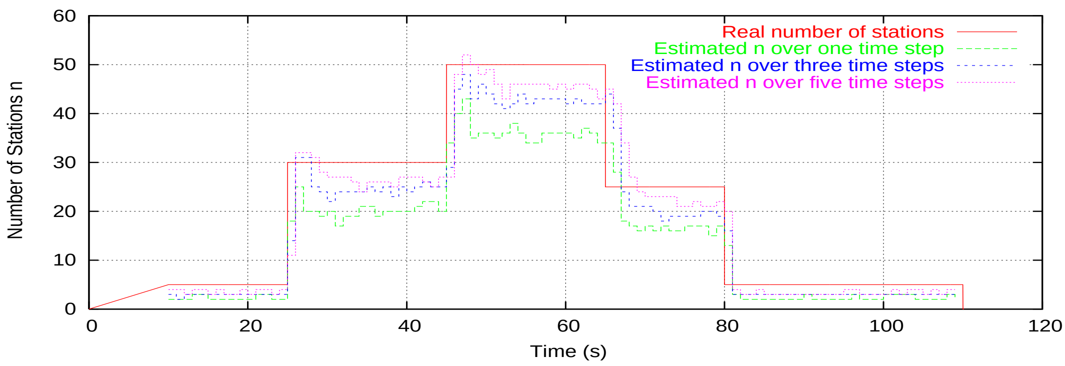

Figure 5.15: Comparison of different estimations of the number of stations in the abrupt scenario case.

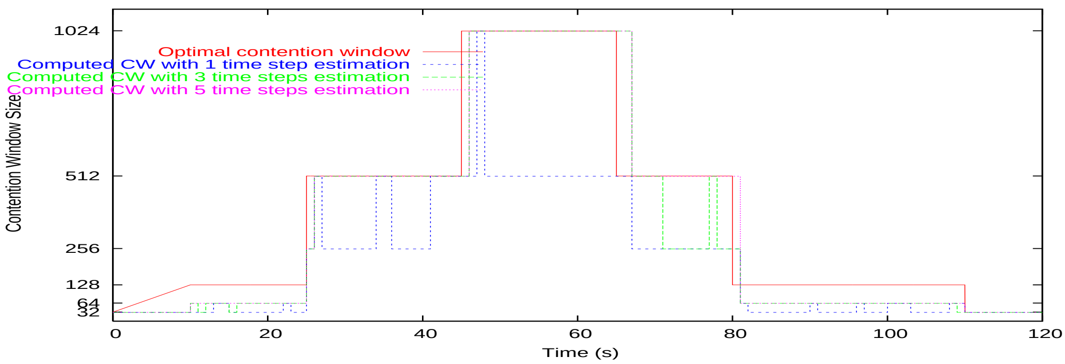

Figure 5.16: Contention window evolution comparison in the abrupt scenario case.

5.5 Infrastructure Mode Analysis

All scenarios discussed so far were for the ad-hoc mode, i.e. when nodes are communicating with each others with no presence of access point. In this section, we consider the case of an infrastructure mode with a single access point. Specifically, we consider the case of 50 stations uploading traffic to the access point. Results of simulations for the case where 25 stations are uploading while 25 others are downloading are presented in Appendix A. Note that the analysis presented below for the upload traffic case can be adopted also for the case presented in Appendix A. As for the case when there are only downloads, there will be only one source and thus the contention window is kept unchanged.

We consider in this mode three scenarios: the slow, abrupt and Poisson arrivals scenarios, introduced in Sections 5.4.1, 5.3.2 and 5.3.3 respectively. Results of the throughput and mean delay obtained for each one of these scenarios are illustrated respectively in Figures 5.17, 5.18, 5.19, 5.20, 5.21, and 5.22.

The simulation results in the case of infrastructure mode are essentially different from the ones obtained in ad-hoc mode. All considered algorithms are more or less affected by changes in the number of active stations. Also, in contrast with the results obtained for the slow approach algorithms in ad-hoc mode, the results obtained in the case of infrastructure mode show, in some cases, performances below the standard (case of abrupt and Poisson arrival scenarios). On the other hand, the adaptive BEB keeps, to a large degree, a uniform and steady performance in both modes, and under the three scenarios. The sharp drop seen in the performance of the algorithms can be explained by the fact that in the case of an access point, multiple flows share that access point buffer, which is of limited size[2]. When that buffer becomes full, the receiving station starts dropping packets, which leads to lost packets and missed ACKs. Lost packets result in a deteriorated performance of the system which will be seen by all algorithms. However, missed ACK is severely penalizing the slow decrease approaches, since each missed ACK is considered as a result of a packet collision, and hence the size of the contention window is consequently incremented to large values extending considerably the assumed optimal values for the current network state. In contrast, the adaptive BEB will not suffer from missed ACK since the increment of the minimal contention window is based on an estimation of the number of contending stations; only lost packets will decrease the system performance. Note that this decline in the performance of the system with the slow decrease approaches will also occur in the case of small number of stations in the cell due to this behavior.

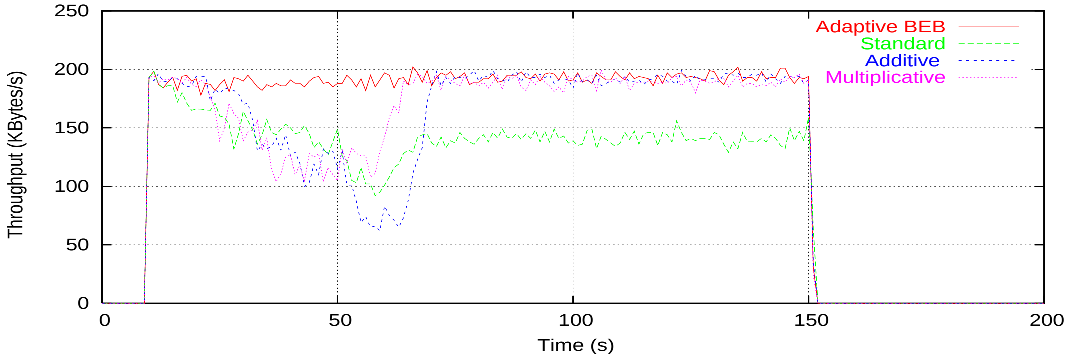

Figure 5.17: Overall throughput in the slow scenario case (Infrastructure mode).

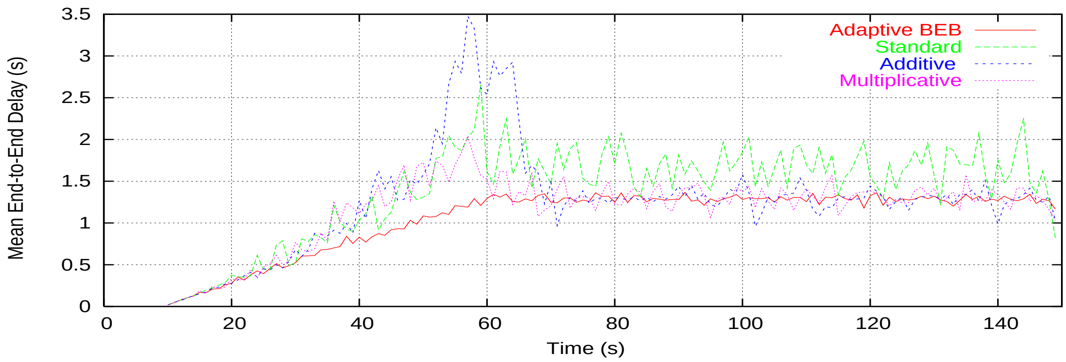

Figure 5.18: Mean delay in the slow scenario case (Infrastructure mode).

5.6 Robustness to Noisy Environments

In this section, we investigate the robustness of our proposed algorithm in noisy environments. We show that this approach performs very well in comparison to slow decrease approaches and under different error types. In fact, since slow decrease approaches are based on automatic and

[2]It is not possible to change the buffer size of the access point in *ns* to a different value than that of the other nodes as all nodes in a wireless scenario have the same buffer size.

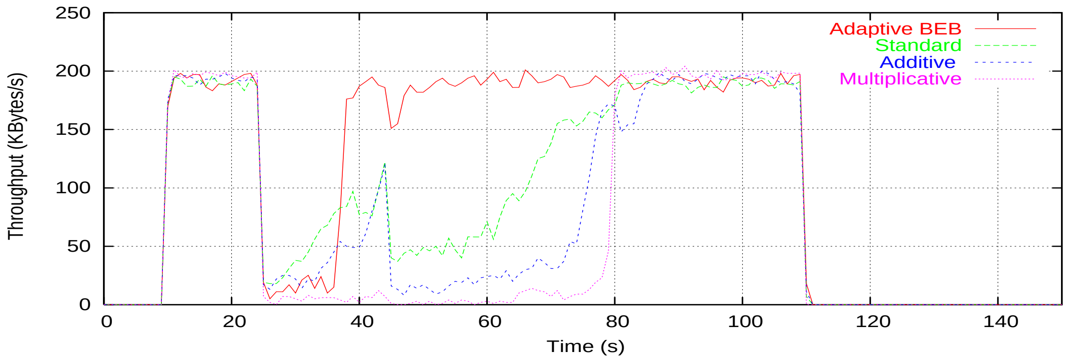

Figure 5.19: Overall throughput in the abrupt scenario case (Infrastructure mode).

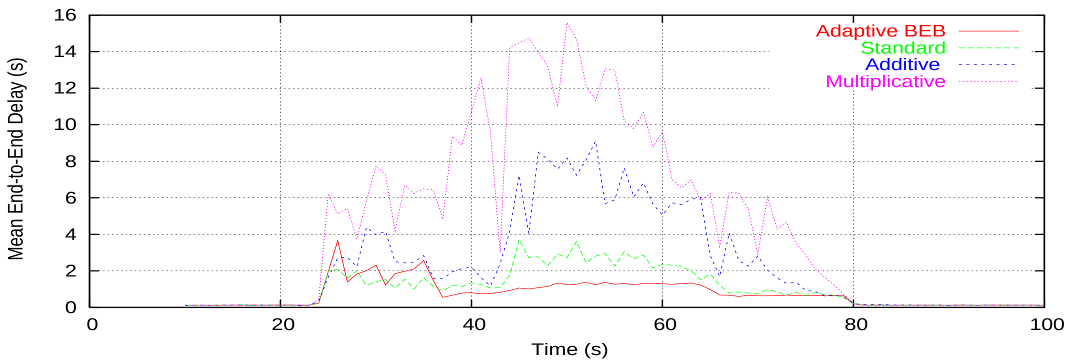

Figure 5.20: Mean delay in the abrupt scenario case (Infrastructure mode).

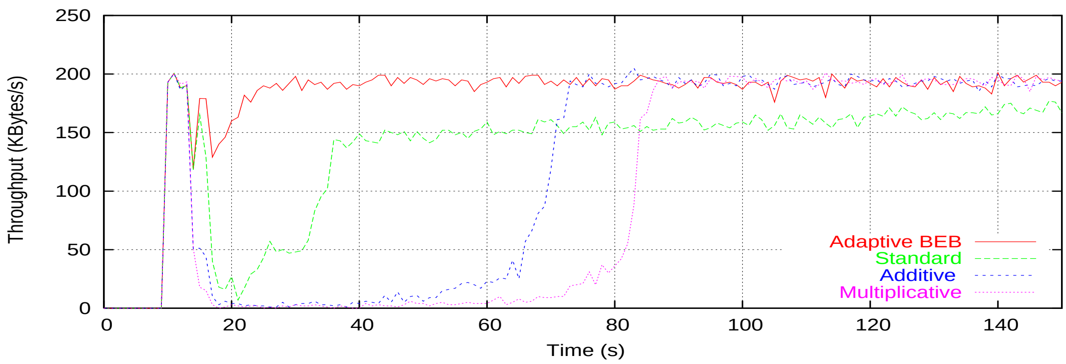

Figure 5.21: Overall throughput in the Poisson scenario case (Infrastructure mode).

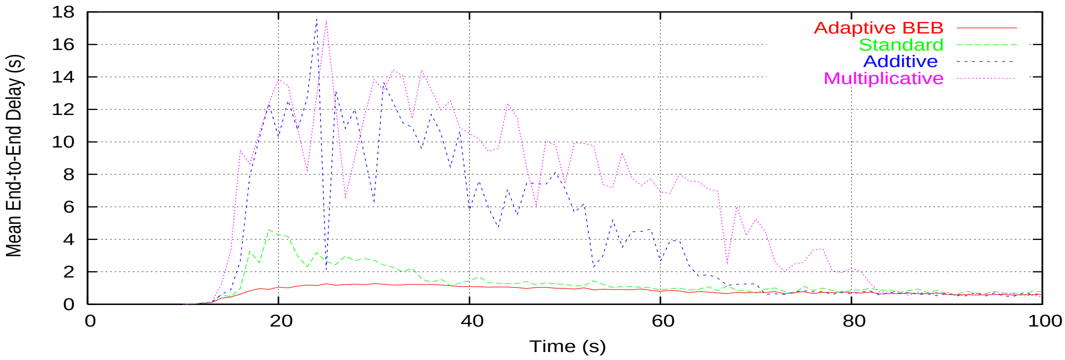

Figure 5.22: Mean delay in the Poisson scenario case (Infrastructure mode).

blind increasing of the contention window size on missed ACKs, they will see a degradation in their performance in noisy environments. This degradation can be best seen in the case of small number of active stations, where consecutive missed ACKs will increase largely the size of contention window to higher values than the optimum size. We will consider two error models, namely Bernoulli model and the Gilbert model. We also consider three network configurations with five, ten and fifteen active stations. In each one of these configurations, we consider a constant number of active sources.

5.6.1 Bernoulli Model

In this case, we consider a loss model with different *Packet Error Rate* (PER) values. More specifically, we considered the following PER values: 0; 0.02; 0.05; 0.07; 0.1; 0.2; 0.4. For each network configuration, and for each value of the PER, we run a simulation, and we compute the mean resulting value of the desired metric. Throughput and mean delay results obtained for various values of the PER under 5, 10 and 15 active stations are shown respectively in Figures 5.23, 5.24, 5.25, 5.26, 5.27, 5.28. As it is shown, for small number of active stations, the impact of noisy channels on the performance of slow decrease approaches is high; however, when the number of sources increases, the effect of noise on slow decrease approaches becomes small. This results from the fact that the resulting window size becomes close to the optimal one in the case of high number of sources under noisy conditions. On the contrary, adaptive BEB is less sensitive to noise, and shows better performance in terms of throughput and packet delay.

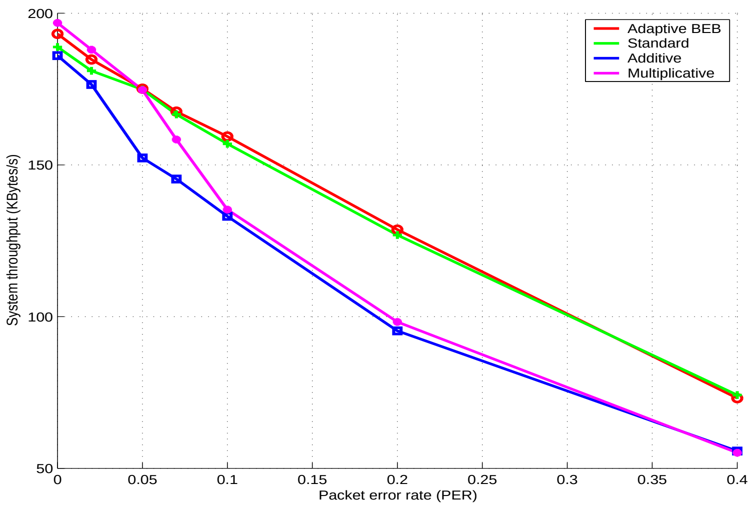

Figure 5.23: Impact of noisy channel on the throughput of 5 sources (Bernoulli model).

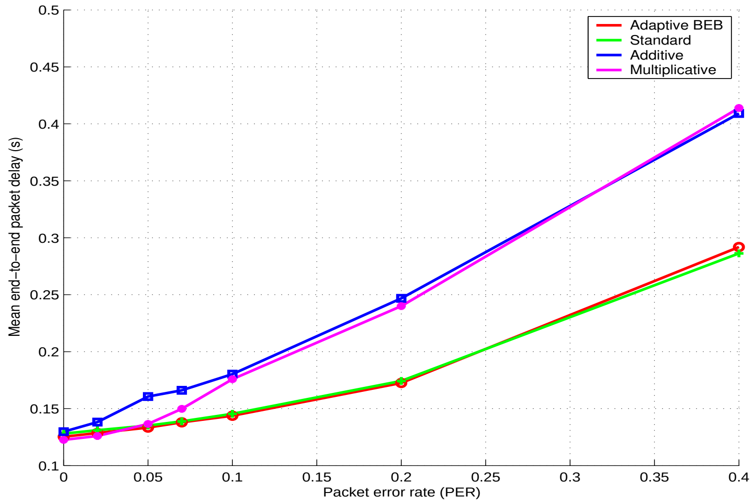

Figure 5.24: Impact of noisy channel on the mean delay of 5 sources (Bernoulli model).

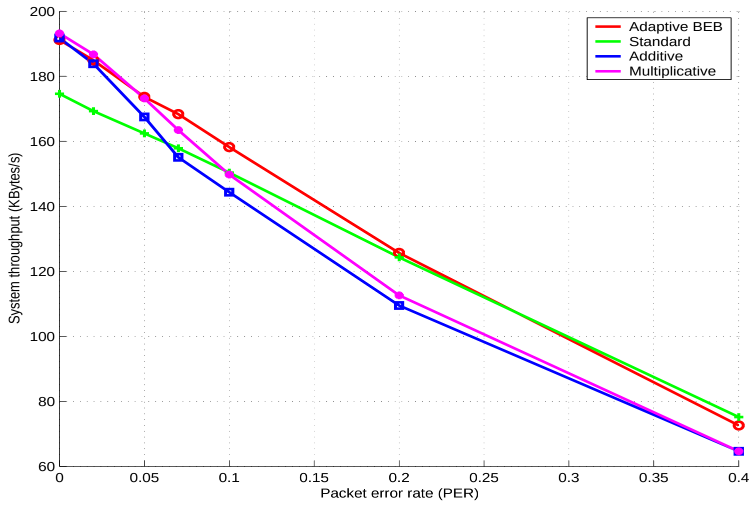

Figure 5.25: Impact of noisy channel on the throughput of 10 sources (Bernoulli model).

5.6.2 Gilbert Model

In the previous section, we have discussed a random error model in which packets are randomly noise altered. In this section, we consider the Gilbert model [10] to simulate the impact of error burstiness on the performance of the different approaches. This simple and widely used error model considers a 2-state Markov chain to characterize the burstiness of the error on the wireless medium. These states, named *Good* and *Bad*, are assigned different PER. When the Markov chain is in the *Good* state, PER is close to 0. When it is in the *Bad* state, PER is high. T_B and T_G denote the time spent by the Markov chain in the *Good* and *Bad* state. $P1$ and $P2$ denote the transition rates between these states (see Figure 5.29). In our simulations, these parameters are set to the following values:

- $\mathrm{PER}_G = 0.1$ and $\mathrm{PER}_B = 0.6$.
- T_G and T_B are set respectively to 0.5 and 0.05 seconds.
- $P1$ and $P2$ are set to 1.

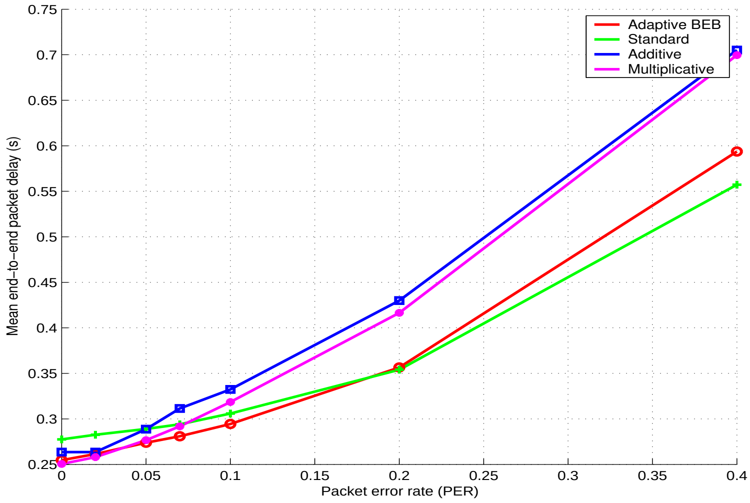

Figure 5.26: Impact of noisy channel on the mean delay of 10 sources (Bernoulli model).

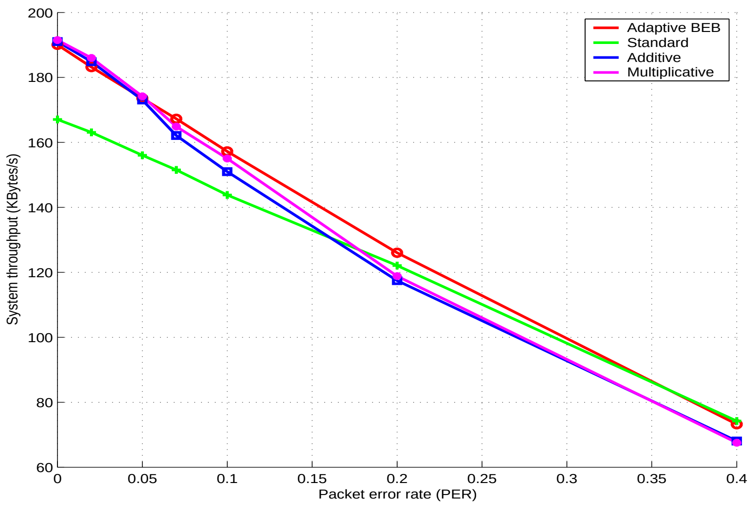

Figure 5.27: Impact of noisy channel on the throughput of 15 sources (Bernoulli model).

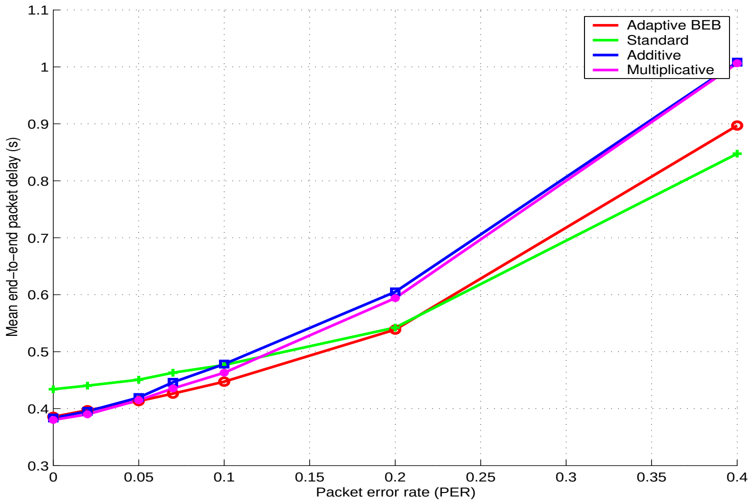

Figure 5.28: Impact of noisy channel on the mean delay of 15 sources (Bernoulli model).

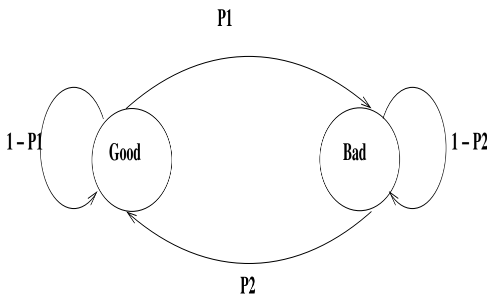

Figure 5.29: 2-state Markov channel.

Results of the system throughput and packet delay for 5, 10, and 15 sources are shown in Figures 5.30, 5.31, 5.32, 5.33, 5.34, 5.35, respectively. As in the Bernoulli model, the effect of noise on slow decrease approaches depends on the number of active stations. When that number increases, the effect of error on the performance of these schemes becomes insignificant, and the drop in the performance results hence from the loss of altered packets. On the other hand, adaptive BEB shows better performance and more stable behavior when the number of sources changes. Tables 5.2 and 5.3 list the average throughput and delay obtained for each case over the simulation runtime.

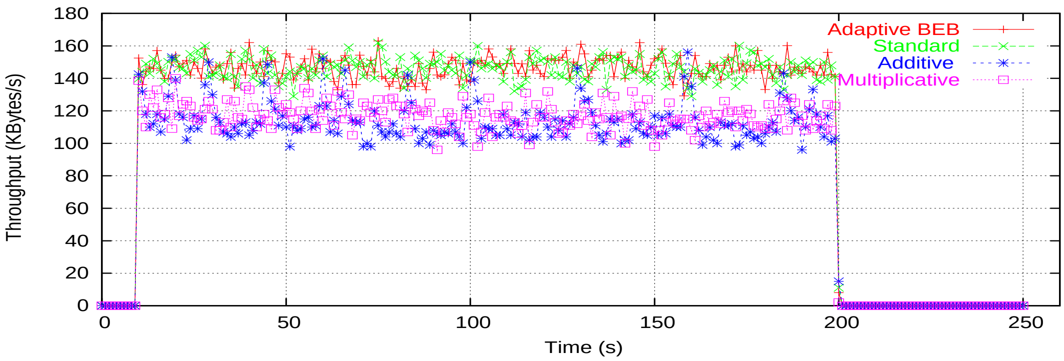

Figure 5.30: Impact of noisy channel on the throughput of 5 sources (Gilbert model).

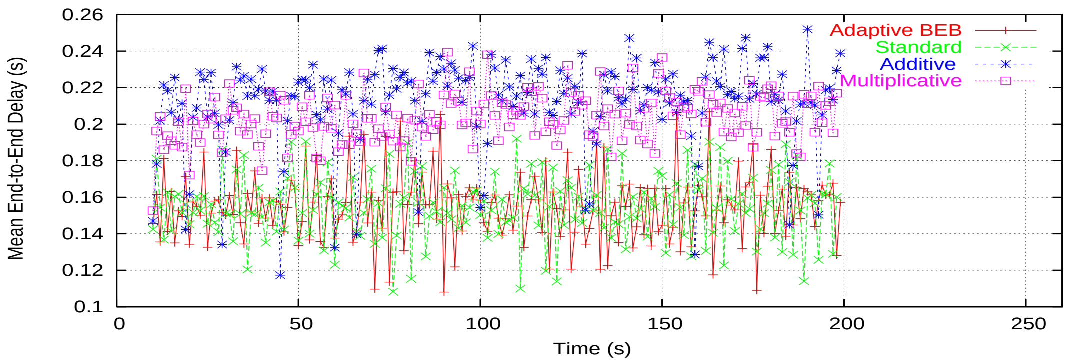

Figure 5.31: Impact of noisy channel on the mean delay of 5 sources (Gilbert model).

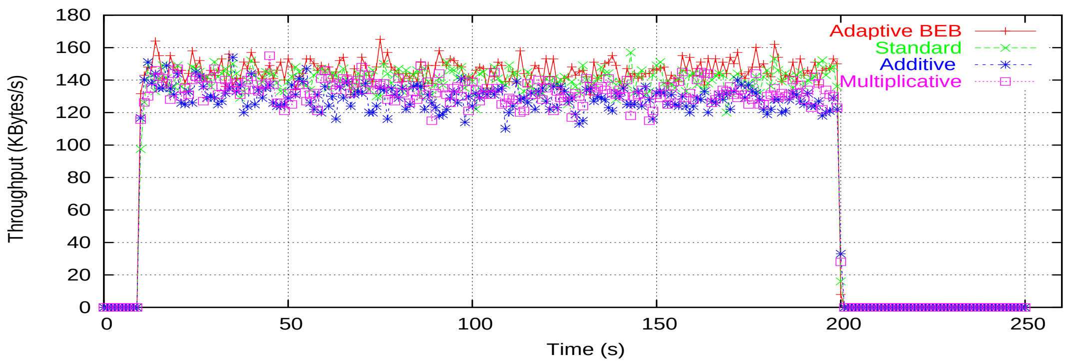

Figure 5.32: Impact of noisy channel on the throughput of 10 sources (Gilbert model).

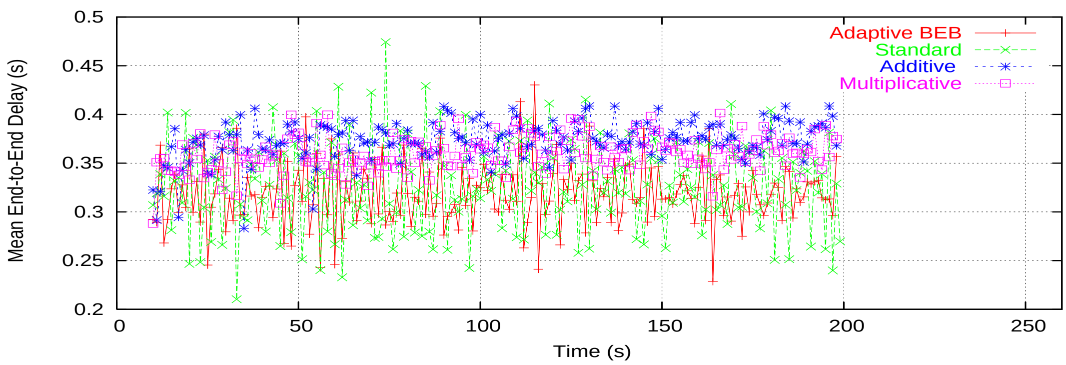

Figure 5.33: Impact of noisy channel on the mean delay of 10 sources (Gilbert model).

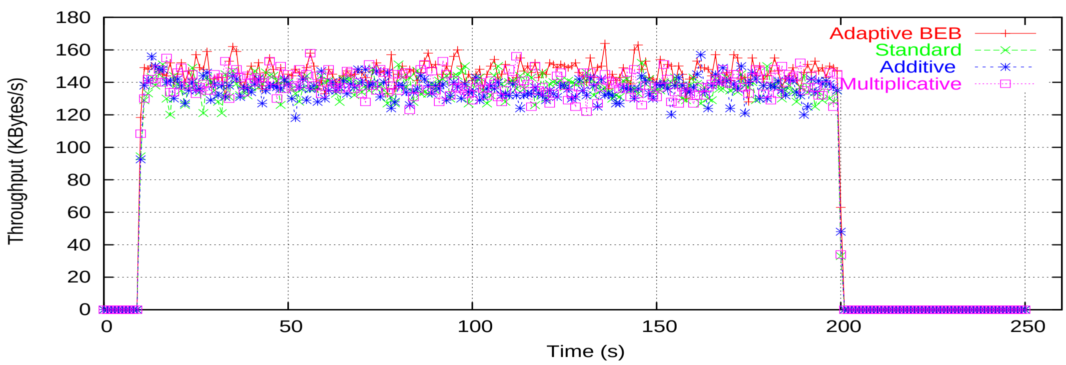

Figure 5.34: Impact of noisy channel on the throughput of 15 sources (Gilbert model).

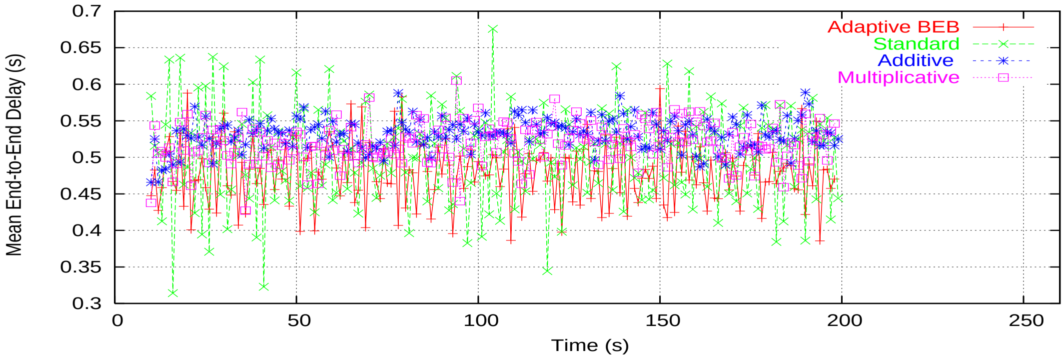

Figure 5.35: Impact of noisy channel on the mean delay of 15 sources (Gilbert model).

	Adaptive BEB	Standard	Additive	Multiplicative
5 sources	146.03	145.503	112.632	116.553
10 sources	145.251	139.377	128.713	132.691
15 sources	145.812	136.505	135.665	138.369

Table 5.2: Throughput expressed in KBytes/s and averaged over 190 seconds of simulations (Gilbert model).

	Adaptive BEB	Standard	Additive	Multiplicative
5 sources	0.15487	0.15484	0.21205	0.2036
10 sources	0.31842	0.32519	0.37303	0.35957
15 sources	0.47966	0.50167	0.53196	0.51831

Table 5.3: End-to-end delay expressed in seconds and averaged over 190 seconds of simulations (Gilbert model).

5.7 Conclusion

In this chapter, we have presented a set of simulations through which we have analyzed and tested the features of our proposed algorithm. We have shown that the difference between the optimal analytical throughput and the simulative throughput of the algorithm is in the order of 3%. We also showed that an adequately accurate but stable estimation of the number of active stations is sufficient to obtain good performance. We have compared the performance of the standard and the slow decrease algorithms to that of our proposed algorithm under various scenario cases. Results obtained illustrate the robustness of our proposal in dynamic and noisy networks, and in infrastructure mode of operation whereas the performance of other proposed approaches in these situations was not satisfactory.

Chapter 6

Open Issues

In this chapter, we outline briefly some issues that we did not detail in the analysis due to the internship limited time. We present simulation results obtained when considering the fairness issue of our proposed algorithm. We also present results obtained when considering FTP sources using the TCP protocol.

6.1 Fairness Analysis

Fairness describes the MAC protocol capability to distribute the available resources equally among communicating terminals [16]. There are a variety of fairness definitions intended to support different QoS and service differentiation. The discussion of these definitions is beyond the scope of our work. We limit our analysis to the basic definition of fairness in wireless networks, i.e. the capability of distributing bandwidth equally between the contending stations. We consider two scenarios: the steady-state scenario of Section 5.3.1 and the abrupt scenario of Section 5.3.2.

In the first scenario, we pick three random sources and we plot their corresponding throughput (see Figure 6.1). As it is shown in the figure, all three sources have approximately the same throughput. Their average values are reported in Table 6.1. The difference between the averages is relatively small and is expected to decrease in longer simulation runs. Thus each source gets an equal bandwidth when it accesses the channel.

Source Id	Average Throughput
source a	3.832
source b	4.011
source c	4.144

Table 6.1: Throughput for each source (KBytes/s) averaged over 90 seconds of simulation.

In the abrupt scenario, we pick five sources: one source from the first entering group (source a), two sources from the second group (sources b and c), and the last two sources are from the third group (sources d and e). The corresponding throughput of each source is drawn in Figure 6.2. As in the previous case, when each source joins the existing ones, it gets an equal share of the available bandwidth. When a source leaves, the other remaining sources manage to share the available bandwidth. Table 6.2 illustrates the average throughput obtained for each source over the simulation run. As we can see, as long as the number of active stations increases, the bandwidth allocated for each source decreases, and the sources tend to get an equal share of the total available

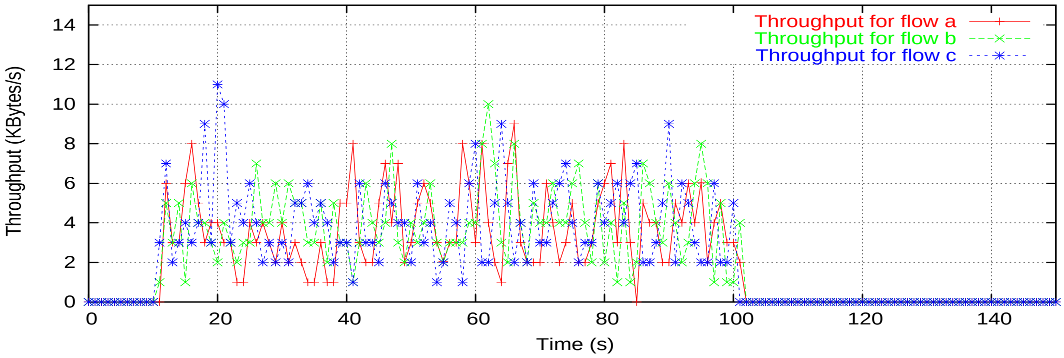

Figure 6.1: Average throughput of 3 random sources in the steady-state scenario case.

bandwidth (e.g. between 25-45s or between 45 and 65s). As in the previous case, the difference between the averages is expected to decrease in longer simulation runs.

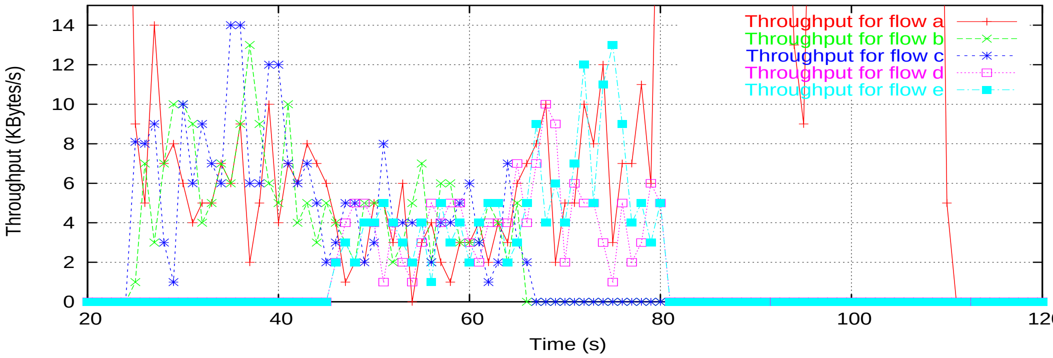

Figure 6.2: Throughput of five random sources in the abrupt scenario case.

Source Id	Average throughput				
	10s-25s	25s-45s	45s-65s	65s-80s	80s-110s
source a	38.584	6.55	3.45	7.214	37.188
source b	0	6.947	3.947	0	0
source c	0	7.789	4	0	0
source d	0	0	3.526	4.857	0
source e	0	0	3.368	6.928	0

Table 6.2: Throughput for each source (KBytes/s).

6.2 TCP Traffic

In this section, we consider only FTP sources that are in saturated conditions (e.g. software or media transfer). We have examined in this analysis the abrupt scenarios of Sections 5.3.2 and 5.5 by replacing the CBR sources by FTP sources and by setting the transport layer to TCP. Recall that in these scenarios, there are 3 different groups of users starting and ending their transmissions at different times. The only difference between these scenarios is that the first is in ad-hoc mode while the other is in infrastructure mode (upload only).

As expected, the maximum throughput achieved by the standard and the adaptive approaches – see Figures 6.3 and 6.4 – is below the corresponding throughput obtained by using the UDP protocol in the corresponding scenarios (results in Figures 5.5 and 5.19). The difference in the performance seen in the case of infrastructure mode between these approaches is almost insignificant, and the performance of the standard in this scenario is similar to that of all adaptive approaches (Figure 6.4). This results from the fact that losses, occurring in the transmitted packets due to the limited size of the access point buffer at the MAC level, impose the TCP layer to decrease the size of its window, and hence the rate of packets sent to the MAC layer is decreased. This highly penalizes the throughput of the adaptive approaches, and makes trivial the control of the contention window done at the MAC level. On the other hand, in the case of ad-hoc mode, the rate at which packets are lost is smaller, and consequently, the influence of the TCP level on the performance of the adaptive algorithms is lessened.

Note that, in the case of TCP, the basic assumption of saturated buffers made by the model in [5] is likely to be violated. Therefore, the transmission probability that we derived based on that model may well become non-optimal and the performance of our algorithm in such a situation may lose its optimality.

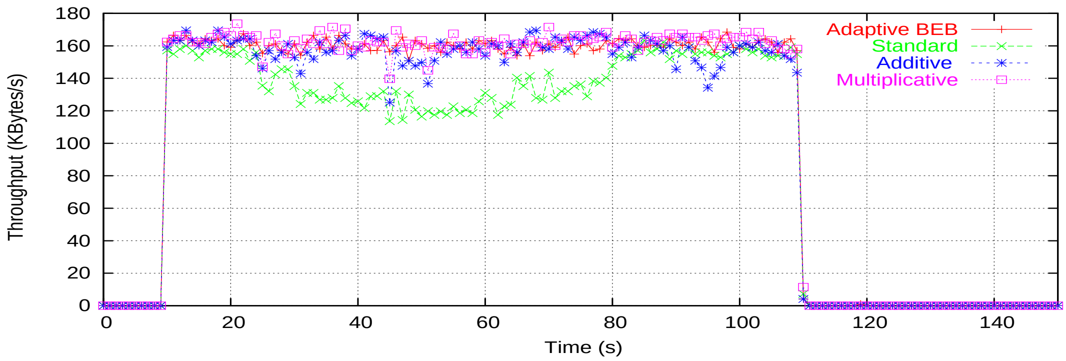

Figure 6.3: Overall throughput in the abrupt scenario case with TCP traffic (ad-hoc mode).

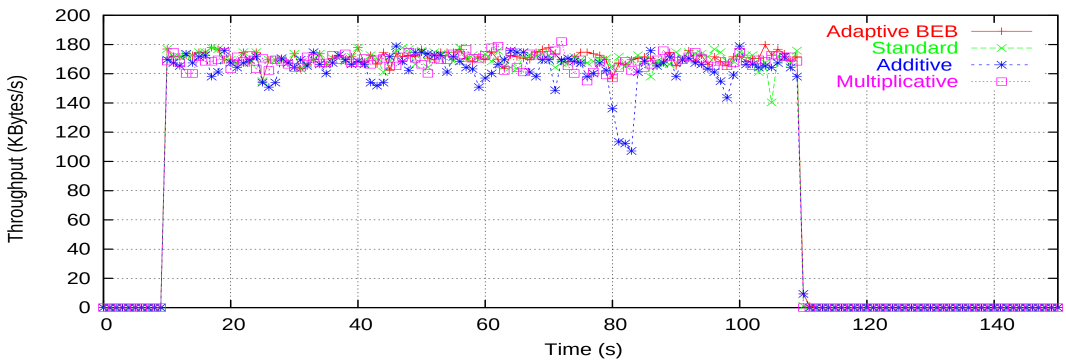

Figure 6.4: Overall throughput in the abrupt scenario case with TCP traffic (infrastructure mode).

Chapter 7

Conclusion and Future Work

In this report, we have proposed and evaluated a simple, efficient and robust mechanism that can be applied to dynamically control the size of the contention window of the IEEE 802.11 DCF mechanism. This mechanism, based on an ARMA filter estimator, has shown good performance in terms of throughput and mean end-to-end delay, under various scenario cases.

After a brief survey on several algorithms proposed in the literature to dynamically adapt the contention window size of the DCF mechanism, we have highlighted what we think are the properties of a good adaptive algorithm.

Based on a Markovian model for the DCF mechanism, proposed in the literature, and which allows to derive the normalized system throughput and the optimal transmission probability, we have discussed a discrete time Markov chain that permits to obtain a relationship between that optimal probability and the optimal value of the minimum contention window. A concrete explanation for the optimality of the transmission probability has been also given.

At this point, it became possible for us to conceive the following adaptive backoff algorithm. Based on an estimation of the number of contending stations, the minimal contention window is set in such a way to achieve near optimal throughput under saturation conditions. We have provided a method to compute the theoretical throughput of our proposal, and validated the model through simulations with ns-2.

We have compared the results obtained in the simulations of our proposal with those obtained with the standard and with two slow decrease approaches proposed in the literature. In all scenarios, our proposed algorithm outperforms the standard. Under dynamic network states, noisy environments or in the case of infrastructure mode of operation, our proposed algorithm shows good performance comparing to that of slow decrease schemes. This results from the fact that the control of the contention window in our proposed algorithm is done based on a estimation of the number of contending stations. This estimation allows to capture the real state of the network, and hence allows to adapt the window size to the actual congestion level in the network.

The estimation we used is simple and based on samples of sign of life measured in the time period separating two resets of the contention window. However, this estimation might not work correctly in the infrastructure mode when frame ciphering is enabled. Even though, frame ciphering is optional and in most of the cases not enabled.

There are several issues that still need to be investigated and are left for future work. The first point that needs to be further evaluated is the fairness issue. Even though we provided brief results on the issue, further analytical analysis needs to be done. Also, the optimization of the algorithm could include, in addition to optimizing the throughput and the delay, an optimization of the fairness.

Another interesting point is the optimization of our algorithm in the case of unsaturated stations, e.g. on-off sources. In fact, all approaches discussed in this report consider the control of the contention window in saturated traffic conditions. However, a more realistic approach is the one that considers the case of finite load traffic (see [2] and [17]).

Another potential area of future work includes the consideration of the stations buffer size in the model, and trying to optimize the algorithm accordingly. Adapting to the congestion in the buffer size is essential in the case of infrastructure mode or in the case of multiple flows destined to a single station in ad-hoc mode (case of UDP traffic). Comparing the performance achieved by our proposal in the case of RTS/CTS access scheme can also be envisaged as a future study.

Appendix A

Infrastructure Mode with Upload and Download Traffic

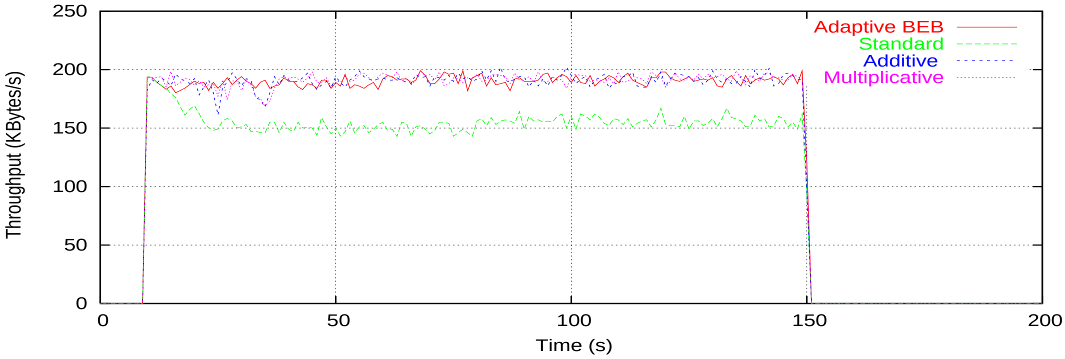

Figure A.1: Overall throughput in the slow scenario case (Infrastructure mode).

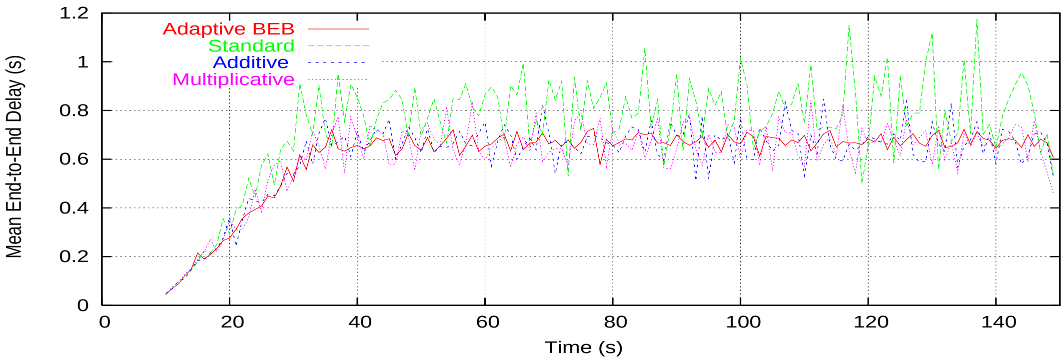

Figure A.2: Mean delay in the slow scenario case (Infrastructure mode).

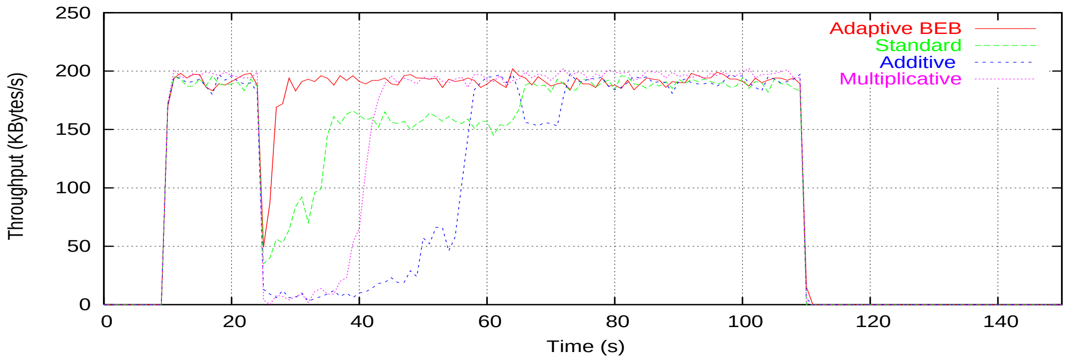

Figure A.3: Overall throughput in the abrupt scenario case (Infrastructure mode).

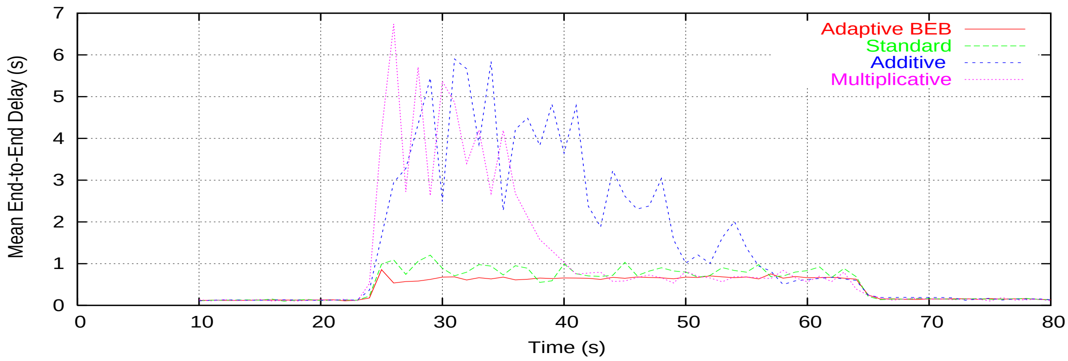

Figure A.4: Mean delay in the abrupt scenario case (Infrastructure mode).

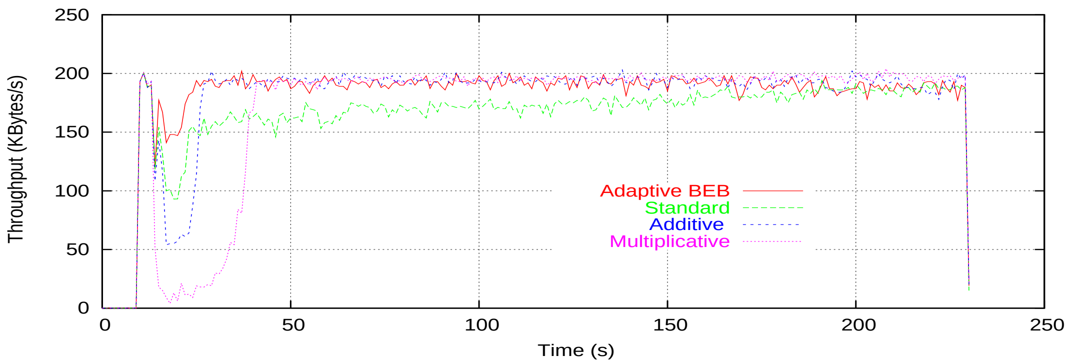

Figure A.5: Overall throughput in the Poisson scenario case (Infrastructure mode).

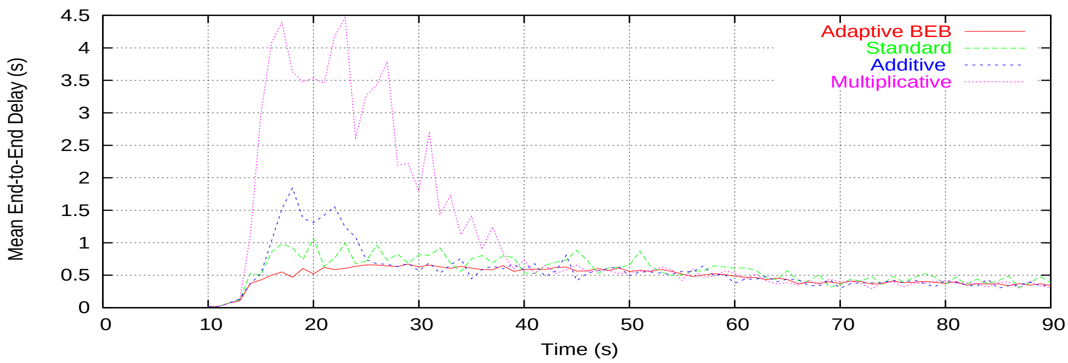

Figure A.6: Mean delay in the Poisson scenario case (Infrastructure mode).

Bibliography

[1] I. Aad, Q. Ni, C. Barakat, and T. Turletti. *Enhancing IEEE 802.11 MAC in congested environments.* IEEE ASWN, August 2004.

[2] F. Alizadeh-Shabdiz and S. Subramaniam. *A finite load analytical model for the IEEE 802.11 distributed coordination function MAC.* Wiopt, March 2003.

[3] S. Alouf, E. Altman and P. Nain. *Optimal on-line estimation of the size of a dynamic multicast group.* In Proceedings of IEEE INFOCOM, june 2002.

[4] G. Bianchi, L. Fratta, and M. Oliveri. *Performance evaluation and enhancement of the CSMA/CA MAC protocol for 802.11 wireless LANs.* In Proc. PIMRC, October 1996.

[5] G. Bianchi. *Performance analysis of the IEEE 802.11 distributed coordination function.* IEEE journal on Selected Areas in Commmunications, March 2000.

[6] G. Bianchi and I. Tinnirello. *Kalman Filter Estimation of the Number of Competing Terminals in an IEEE 802.11 network.* In Proceedings of IEEE INFOCOM, March 2003.

[7] L.Bononi, M.Conti, E.Gregori. *Run-Time Optimization of IEEE 802.11 Wireless LANs performance.* IEEE Transactions on Parallel and Distributed Systems, January 2004.

[8] F. Cali, M. Conti, and E. Gregori. *802.11 Wireless LAN: Capacity Analysis and Protocol Enhancement.* In Proceedings of IEEE INFOCOM, March 1998.

[9] F. Cali, M. Conti, and E. Gregori. *Dynamic Tuning of the IEEE 802.11 Protocol to Achieve a Theoretical Throughput Limit.* IEEE/ACM Trans. on Networking, December 2000.

[10] E.N. Gilbert. *Capacity of a burst-noise channel.* Bell System Technical Journal, 1960.

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

[12] IEEE. *IEEE Std 802.11-1999, Part 11:Wireless LAN Medium Access Control(MAC) and Physical Layer(PHY) Specifications*, 1999.

[13] J. Galtier. *Optimizing the IEEE 802.11b Performance using Slow Congestion Window Decrease.* 16th ITC Specialist Seminar, Septembre 2004.

[14] A. Kumar, E. Altman, D. Miorandi and M. Goyal. *New Insights from a Fixed Point Analysis of Single Cell IEEE 802.11 WLANs.* In Proceedings of IEEE INFOCOM, March 2005.

[15] Y. Kwon, Y. Fang and H. Latchman. *A Novel MAC Protocol with Fast Collision Resolution for Wireless LANs.*In Proceedings of IEEE INFOCOM, March 2003.

[16] S. Lu, V. Bharghavan and R. Srikant. *Fair Scheduling in Wireless Packet Networks.* In Proceedings of ACM SIGCOMM, 1997.

[17] D. Malone, K. Duffy and D.J. Leith. *Modeling the 802.11 distributed coordination function with heterogeneous finite load.* Resource Allocation in Wireless Networks, April 2005.

[18] Q. Ni, I. Aad, C. Barakat and T. Turletti. *Modeling and analysis of slow CW decrease for IEEE 802.11 WLAN.* IEEE PIMRC, September 2003.

[19] Network Simulator. *http://www.isi.edu/vint/nsnam/.*

[20] A. Tanenbaum. *Computer Computer Networks, Fourth Edition.* Prentice Hall PTR, 2003.

[21] Y. Xiao. *Saturation performance metrics of the IEEE 802.11 MAC.* In Proc. of the IEEE Vehicular Technology Conference, October 2003.

[22] Y. Yang, J. Wang and R. Kravets. *Distributed Optimal Contention Window Control for Elastic Traffic in Wireless LANs.* In Proceedings of IEEE INFOCOM, March 2005.

List of abbreviations

ACK	Acknowledgement
AP	Access Point
ARMA	Auto Regressive Moving Average
BEB	Binary Exponential Backoff
CBR	Constant Bit Rate
CTS	Clear To send
CSDMA/CA	Carrier-Sense Multiple Access/Collision Avoidance
CW	Contention Window
CWmin	Minimum Contention Window
CWmax	Maximum Contention Window
DCF	Distributed Coordination Function
DIFS	Distributed Inter Frame Space
DSSS	Direct Sequence Spreading Spectrum
e.g.	example
et al.	et alii
FCR	Fast Collision Resolution
FHSS	Frequency Hopping Spreading Spectrum
FTP	File Transfer Protocol
GCA	General Contention window Adaptation
i.e.	id est
IEEE	Institute of Electrical and Electronics Engineers
i.i.d.	Independent and Identically Distributed
LLC	Logical Link Control
MAC	Medium Access Control
MPDU	Maximum Packet Data Unit
NAV	Network Allocation Vector
ns	network simulator
PCF	Point Coordination Function
PER	Packet Error Rate
PDU	Packet Data Unit
PHY	Physical
QoS	Quality of Service
RTS	Request To Send
SIFS	Short Inter Frame Space
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
WLAN	Wireless Local Area Network