

# Performance Analysis of IEEE 802.11 DCF: Throughput, Delay, and Fairness

Zhifei Li

Department of Computer Science  
Johns Hopkins University  
Baltimore, MD 21218 USA  
Email: zli19@jhu.edu

Amitabha Das, Anil K. Gupta

School of Computer Engineering  
Nanyang Technological University  
Singapore, 639798  
Email: {asadas, asgupta}@ntu.edu.sg

Sukumar Nandi

Dept. of Computer Science & Engineering  
Indian Institute of Technology  
Guwahati, India, 781039  
Email: sukumar@iitg.ernet.in

**Abstract**—Throughput, delay, and fairness are three most important performance metrics in IEEE 802.11-based wireless networks. In this paper, we provide a simple but precise analytical model to evaluate the above three metrics. Different from the previous models that have focused either on throughput or on delay, our model is unique as we obtain throughput, delay, and fairness all together. Moreover, by considering the fact that a packet may be dropped after a finite number of retrials, we distinguish five cases of packet delay, and derive the relationship among them. Using the proposed model, we carry out an extensive and insightful performance evaluation of IEEE 802.11 under different system parameters. Our results show that the throughput, delay, and fairness are quite sensitive to the system parameters being chosen, demonstrating the importance of performing dynamic tuning of the system parameters in IEEE 802.11. Moreover, the optimal values of the parameters are inconsistent with each other for different performance metrics, implying that some tradeoff among the metrics must be made in the dynamic tuning. Finally, our results also show that the previous models have overestimated the throughput as well as the packet delay.

## I. INTRODUCTION

Recently, wireless ad-hoc networks have attracted considerable research interest. IEEE 802.11 [4] is the de facto standard for Wireless LANs. The Distributed Coordination Function (DCF) defined in IEEE 802.11 is also popularly used in wireless ad-hoc networks due to the distributed nature.

In this paper, we focus on the performance evaluation of the DCF MAC protocol. Throughput, delay, and fairness are the three performance metrics that are of great interest to us. Most of the published work on the performance analysis of DCF focused either on throughput or on delay. In contrast, in this paper, we develop a simple analytical model that is able to evaluate the above three metrics all together. Our model assumes that a station transmits in a randomly chosen slot time with an independent probability  $\tau$ , and the packet being transmitted experiences a collision with an independent probability  $p$ . In fact,  $\tau$  and  $p$  characterize the main feature of binary exponential back-off (BEB) algorithm in DCF, and they can be obtained using the published models (e.g., [1]). With  $\tau$  and  $p$ , we carry out the analysis for throughput, delay, and fairness. Using the proposed model, we carry out an extensive performance evaluation of DCF by varying the system parameters such as the minimum contention window  $CW_{min}$ , the maximum contention window  $CW_{max}$ , and the

retry limit  $m$ .

### Our Main Contributions:

- In the throughput analysis, we provide a general analysis about the sensitivity of the throughput with respect to the system parameters, which is not available in the published papers. We also improve the preciseness of the throughput calculation by correcting some of the values used in [1].
- In the delay analysis, while the published models ([3], [11]) assume infinite retrials for a given packet, our model considers the fact of finite retrial in IEEE 802.11. Moreover, we have defined five kinds of delay and derived the relationship among these delays. Our analytical results show that these delays have very different values, showing the importance to distinguish them. Such an analysis is original and novel and has not been appeared in the published models on delay analysis.
- We have carried out an original fairness analysis based on the the probability distribution of the delay that is obtained in the delay analysis. The fairness analysis of IEEE 802.11 has not been done by the published models.
- Through an extensive performance evaluation of DCF based on the proposed model, we gain several insights about the behavior of IEEE 802.11. We arrive at several conclusions that are contrary to those obtained in previous models. For example, we find that the previous models assuming infinite retrials have overestimated the delay, and thus led to misleading conclusions. Our results also have great implications on the dynamic tuning of the system parameters in IEEE 802.11.

The remainder of the paper is organized as follows. The analytical model is presented in Section II. A thorough performance evaluation is carried out in Section III. Future work and related work are given in Section IV. The paper is concluded in Section V.

## II. ANALYTICAL MODEL

In the analysis, several popular assumptions have been made: (1) There are a finite number of stations (say  $n$ ) in the network, and all the stations can hear each other (i.e., single-hop); (2) Stations are always backlogged; (3) Ideal channel conditions (i.e., no wireless error); (4) RTS/CTS handshake is used; and (5) The data packets have a fixed length.

### A. Modelling Binary Exponential Back-off

In [1], by modelling the stochastic process representing the back-off counter at a given station as a discrete-time Markov chain, two parameters that characterize the binary exponential back-off (BEB) algorithm are obtained. The first one is the *transmission probability*,  $\tau$ , with which a station transmits in a randomly chosen slot time. The second parameter is the *conditional collision probability*,  $p$ , representing the probability of a collision experienced by a packet given that it is transmitted on the channel. The author in [1] did not consider the frame retry limit used in IEEE 802.11. In light of this, the authors in [13] extended the model by considering the retry limit. For the details about how to calculate  $\tau$  and  $p$ , please refer to [8]. Once we know the values of  $\tau$  and  $p$ , many performance metrics can be obtained. In the following subsections, we discuss how the throughput, delay, and fairness can be obtained. Note that the discussion below can also be made based on any other model (e.g., [2]) as long as  $\tau$  and  $p$  can be obtained from the model.

### B. Throughput Analysis

The normalized system throughput  $S$  is defined as the fraction of time that the channel is used to successfully transmit payload bits. The throughput can be obtained by analyzing the possible events that may happen on the shared medium in a randomly chosen slot time. Let  $P_{idle}$ ,  $P_{col}$ , and  $P_{succ}$  be the probabilities that a randomly chosen slot corresponds to an idle slot, a collision, or a successful transmission, respectively. Moreover, let  $\sigma$ ,  $T_{col}$ , and  $T_{succ}$  be the duration of the slot corresponding to an idle slot, a collision, or a successful transmission, respectively. We can obtain the average duration (represented by  $T_{avg}$ ) that a generic slot lasts as follows,

$$T_{avg} = P_{idle}\sigma + P_{succ}T_{succ} + P_{col}T_{col} \quad (1)$$

Now, the throughput  $S$  can be calculated as [1],

$$\begin{aligned} S &= \frac{E[\text{payload information transmitted in a slot time}]}{E[\text{length of a slot time}]} \\ &= \frac{P_{succ} \times E[P]}{T_{avg}} = \frac{P_{succ} \times E[P]}{P_{idle}\sigma + P_{succ}T_{succ} + P_{col}T_{col}} \end{aligned} \quad (2)$$

where  $E[P]$  is the average payload size (in terms of time units), and thus  $P_{succ} \times E[P]$  is the average amount of payload information successfully transmitted in a generic slot time.

By dividing the numerator and denominator of Equation (2) by  $P_{succ}T_{succ}$ , the throughput can be expressed as follows,

$$S = \frac{E[P]/T_{succ}}{1 + \frac{P_{col}}{P_{succ}} \times \frac{T_{col}}{T_{succ}} + \frac{P_{idle}}{P_{succ}} \times \frac{\sigma}{T_{succ}}} \quad (3)$$

The sensitivity of the throughput to the system parameters (e.g.,  $CW_{min}$ ,  $CW_{max}$ , and  $m$ ) can be easily analyzed from the above formula. Clearly,  $E[P]$ ,  $T_{succ}$ ,  $T_{col}$ , and  $\sigma$  are independent to the system parameters. On the other hand, the values of  $P_{col}$ ,  $P_{succ}$ , and  $P_{idle}$  depend on  $\tau$  and  $p$ , which in turn depend on the system parameters. Therefore, we only need to discuss the sensitivity of the throughput with respect to  $P_{col}$ ,  $P_{succ}$ , and  $P_{idle}$ . From Equation (3), we know that the throughput  $S$  is sensitive to the value of  $P_{col}/P_{succ}$  only when the constant  $T_{col}/T_{succ}$  is large (compared to the first

term of the denominator, i.e., unity). Similarly,  $S$  is sensitive to  $P_{idle}/P_{succ}$  only when the constant  $\sigma/T_{succ}$  is large. The sensitivity analysis gives an idea whether dynamic tuning of the system parameters is necessary or not.

As mentioned in [1], the above analysis applies to both the two-way and four-way handshakes. To specifically compute the throughput for a given handshake, we only need to specify the corresponding values of  $T_{col}$  and  $T_{succ}$ . Note that the idle slot time  $\sigma$  is specific to the physical layer. The specific values of  $T_{succ}$  and  $T_{col}$  for the four-way handshake are as follows,

$$\begin{cases} T_{succ} = RTS + CTS + Data + ACK + 3SIFS + DIFS \\ T_{col} = RTS + DIFS + SIFS + CTS \end{cases} \quad (4)$$

It may be noticed that the above expression is different from that in [1]. First, the propagation delay  $\delta$  is no longer included in the above formula since the Short Inter-Frame Space (SIFS) has already contained  $\delta$  according to the standards [4]. Another difference is that in the expression of  $T_{col}$ , we have added an additional term,  $SIFS + CTS$ . As stated in [1],  $T_{col}$  is the period of time during which the channel is sensed busy by the *non-colliding* stations. After a collision, the colliding stations wait for a time equal to  $CTSTimeout$  (or  $ACKTimeout$  in the two-way handshake case). On the other hand, the non-colliding stations defer by an Extended Inter-Frame Space (EIFS) value since they cannot interpret the contents of the colliding frames. Moreover, the EIFS value is equal to  $DIFS + SIFS + TxTime(ACK)$ , which is large enough for the complete transmission of the CTS or ACK frame (note that CTS has the same length as ACK). Therefore, if  $CTSTimeout$  or  $ACKTimeout$  is set to the time needed for the transmission of the CTS/ACK frame (which is true in the popularly adopted NS-2 simulator), then,  $T_{col}$  for all the stations, whether involved in the collision or not, will be the same. Therefore, the  $SIFS + CTS$  term should be included in  $T_{col}$ . Clearly, the above modifications are important to obtain a more precise value of the throughput. The model in [1] overestimates the throughput as it uses a smaller  $T_{col}$ .

### C. Delay Analysis

In IEEE 802.11, there is no queue at the MAC layer itself, and thus IEEE 802.11 standards have not specified any queuing mechanisms. However, normally there should be a queue on the top of the MAC layer. Therefore, in general, the delay that a packet experiences should include two parts, i.e. the delay experienced in the queue and the delay experienced at the MAC layer. We only focus on the MAC layer delay. Since there are *multiple* stations contending for the shared medium and IEEE 802.11 DCF is a *random* access protocol, the MAC layer delay is a *random* value, which requires a detailed analysis. Five types of delay at the MAC layer are relevant to this analysis, which we will discuss in a little while.

As we know, in IEEE 802.11, if a packet is handed to the MAC layer from the upper layer, the packet may be transmitted successfully in one or several trials, or the packet will be dropped if the retry limit is reached. In both cases,

the MAC layer will notify the upper layer about the final status of the packet (i.e., transmitted successfully or dropped). Therefore, the MAC layer delay should be defined to be the time interval *from the instant* that the packet is handed to the MAC layer *to the instant* that the upper layer gets a notification from the MAC layer regarding the final status of the packet. Accordingly, we can define the following four kinds of delays.

*Delay  $D_{succ}$* : if a packet is successfully transmitted, what is the delay experienced at the MAC layer?

*Delay  $D_{drop}$* : if a packet is dropped, what is the delay experienced at the MAC layer?

*Delay  $D_{notify}$* : combining the above two cases, what is the delay before the upper layer will get a notification from the MAC layer about the final status (transmitted or dropped) of the packet? Certain applications may require a small value of  $D_{notify}$  to take a prompt action.

*Delay  $D_{intersucc}$* : from the viewpoint of an upper layer at a *given* station, what is the delay between two successful packet transmissions? Note that in the definition of  $D_{intersucc}$ , the successful transmissions must belong to a single station only.

At last, to show the error introduced by the infinite retrials assumption [3], [11], we define the following delay:

*Delay  $D_{infinite}$* : if we assume that a packet will always be retransmitted (i.e., infinite retrials) until it is transmitted successfully, what is the delay experienced at the MAC layer?

In the following subsections, we discuss how to compute the above delays.

1) *Derivation of  $D_{succ}$* : Clearly,  $D_{succ}$  depends upon the number of back-offs (called back-off *stages*, or simply stages in this paper) that a packet experiences, as well as upon the back-off counter generated at each stage. Figure 1 gives an example of the back-off process experienced by a packet at a given station. In the figure, the shaded blocks represent the events in which the given node is involved. The packet experiences three stages (from stage 0 to stage 2) before it is transmitted successfully, and the back-off counter (represented by  $B_0$ ,  $B_1$ , and  $B_2$ , respectively) generated at the stages are 3, 4, 2, respectively.

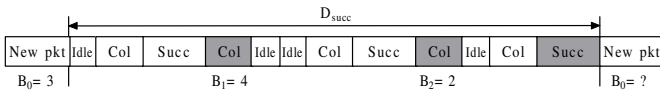


Fig. 1. Illustration of Delay for Successful Packet

From the figure, it is easy to find that the back-off counter at the given station decrements by one after every slot time if the slot corresponds to: (1) an idle slot, (2) a collision between *other* stations, or (3) a successful transmission between *other* stations<sup>1</sup>. Since we know that the duration of the idle, collision, and successful transmission slots are  $\sigma$ ,  $T_{col}$ , and  $T_{succ}$ , respectively, we can easily get the delay that is contributed by the idle slots, as well as by the collisions among other stations, and by the successful transmissions of other stations.

<sup>1</sup>In fact, according to IEEE 802.11, corresponding to a collision or a successful transmission, the counters at the stations do not decrement. However, the imprecision caused by such an approximation is insignificant.

For instance, in the above example, the number of idle slots is 4, while the numbers of collisions and successful transmissions among other stations are 3, and 2, respectively. Therefore, the delay contributed by the back-off slots is  $4\sigma + 3T_{col} + 2T_{succ}$ . In addition to this, the delay  $D_{succ}$  should include the time spent on the collisions that the given station *itself* is involved, e.g.,  $2 \times T_{col}$  in the above example. Finally, the  $D_{succ}$  should also include the time needed for the successful transmission of the given packet, i.e.,  $T_{succ}$ . Therefore, the delay  $D_{succ}$  is  $(4\sigma + 3T_{col} + 2T_{succ}) + 2T_{col} + T_{succ}$ . In the following, we provide a formal method to calculate  $D_{succ}$  for a general case.

Let  $B_i$  be the random back-off counter generated at stage  $i$ , and let  $B(j)$  be the *sum* of the back-off slots generated at stages from 0 to  $j$ . Therefore,

$$B(j) = \sum_{i=0}^j B_i \quad (5)$$

We know that  $B_i$  at stage  $i$  follows the uniform distribution in the range  $[0, CW_i]$ , where  $CW_i$  takes value according to the BEB algorithm. It is easy to get the probability mass function (pmf) of  $B(j)$ , which is simply a *convolution* of the pmfs of all  $B_i$  where  $i \in [0, j]$ . Clearly,  $B(j)$  is in the range of  $[0, B(j)_{max}]$ , where  $B(j)_{max} = \sum_{i=0}^j CW_i$ .

Since the probability that a transmission experiences a collision is  $p$ , the probability that a packet is successfully transmitted at stage  $j$  (starting from stage 0) is,

$$P_{succ}(j) = p^j \times (1 - p) \quad 0 \leq j \leq m \quad (6)$$

where  $m$  represents the retry limit. Moreover, the probability that a packet is dropped because the retry limit is reached is,

$$P_{drop} = 1 - \sum_{j=0}^m p^j \times (1 - p) = p^{m+1} \quad (7)$$

Let  $P_{succ}(j, b)$  be the probability that the packet is successfully transmitted at stage  $j$  ( $0 \leq j \leq m$ ), and the sum of the back-off slots generated up to stage  $j$  is equal to  $b$ . Therefore,  $P_{succ}(j, b)$  can be expressed as,

$$P_{succ}(j, b) = P_{succ}(j) \times Pr(B(j) = b) \quad 0 \leq j \leq m \quad (8)$$

Similarly, let  $P_{drop}(b)$  be the probability that a packet is dropped, and the sum of the back-off slots generated up to stage  $m$  is equal to  $b$ . Therefore,  $P_{drop}(b)$  can be expressed as,

$$P_{drop}(b) = P_{drop} \times Pr(B(m) = b) \quad (9)$$

Since  $D_{succ}$  represents the delay of a packet that is successfully transmitted, we should only consider the packets that are successfully transmitted while excluding the packets that are dropped. Therefore, we need to define the following conditional probability  $P'_{succ}(j, b)$ ,

$$P'_{succ}(j, b) = \frac{P_{succ}(j, b)}{1 - P_{drop}} \quad (10)$$

As a generalization of the example given in Figure 1, the corresponding delay,  $D_{succ}(j, b)$ , which represents the delay of a packet that is successfully transmitted at stage  $j$  and the sum of the back-off slots generated up to stage  $j$  is  $b$ , is,

$$D_{succ}(j, b) = b \times T_{avg} + j \times T_{col} + T_{succ} \quad (11)$$

Note that there is an approximation in the above equation, that is, we have simply used  $b \times T_{avg}$  to represent the duration of the  $b$  number of back-off slots. In fact, if, among the  $b$  number of back-off slots, there are  $b_{col}$  number of slots corresponding to collisions,  $b_{succ}$  number of slots corresponding to successful transmissions, and  $b_{idle}$  number of idle slots, the delay contributed by the  $b$  number of back-off slots is  $(b_{col}T_{col} + b_{succ}T_{succ} + b_{idle}\sigma)$ . The random variables  $b_{col}$ ,  $b_{succ}$ ,  $b_{idle}$ , and  $b$  can be characterized by a multinomial probability distribution. However, it makes the analysis extremely complex. Moreover, the error introduced by the above approximation is very small if  $b$  is relatively large, which is normally true in IEEE 802.11. Therefore, this approximation is reasonable and is used throughout this paper.

Equations (10) and (11) express the probability mass function of the delay  $D_{succ}$ . Therefore, the average of  $D_{succ}$  is,

$$\overline{D_{succ}} = \sum_{j=0}^m \sum_{b=0}^{B(j)_{max}} D_{succ}(j, b) \times P'_{succ}(j, b) \quad (12)$$

With the probability mass function of the delay expressed by equations (10) and (11), we can compute other important metrics, such as the standard deviation of the delay.

2) *Derivation of  $D_{drop}$* : Similar to the derivation of  $D_{succ}$ , we present an example in Figure 2 for the case that a packet will be dropped. As shown in the figure, in contrast to the case of a successful transmission, if a packet is to be dropped, the back-off process experienced by the packet must reach the retry limit (i.e., stage  $m$ ).

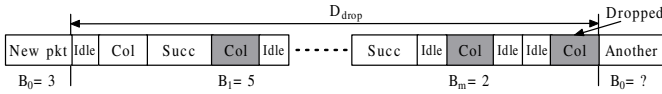


Fig. 2. Illustration of Delay for Dropped Packet

To compute  $D_{drop}$ , we define the following conditional probability as done in the derivation of  $D_{succ}$ ,

$$P'_{drop}(b) = \frac{P_{drop}(b)}{P_{drop}} = Pr(B(m) = b) \quad (13)$$

where  $P_{drop}(b)$ ,  $P_{drop}$ , and  $B(m)$  have been defined in equations (9), (7), and (5), respectively.

The corresponding delay, using the same approximation as in Equation (11), can be expressed as follows,

$$D_{drop}(b) = b \times T_{avg} + m \times T_{col} + T_{col} \quad (14)$$

The above two equations express the probability mass function of the delay  $D_{drop}$ . The average of  $D_{drop}$  is,

$$\overline{D_{drop}} = \sum_{b=0}^{B(m)_{max}} D_{drop}(b) \times P'_{drop}(b) \quad (15)$$

After some simple steps, we get the following expression,

$$\overline{D_{drop}} = T_{col} \times (m + 1) + T_{avg} \times \overline{B(m)} \quad (16)$$

where  $\overline{B(m)}$  is the average value of  $B(m)$ , and is equal to  $B(m)_{max}/2$ .

Based on equations (13) and (14), we can also obtain the standard deviation of  $D_{drop}$ .

3) *Derivation of  $D_{notify}$* : The probability mass function of the delay  $D_{notify}$  is expressed by equations (8), (11), (9) and (14). Note that we should use the probabilities expressed in equations (8) and (9) rather than the conditional probabilities expressed in equations (10) and (13) since  $D_{notify}$  represents the delay for a general packet (either dropped or successfully transmitted). The average of  $D_{notify}$  is,

$$\begin{aligned} \overline{D_{notify}} &= \sum_{j=0}^m \sum_{b=0}^{B(j)_{max}} [D_{succ}(j, b) \times P_{succ}(j, b)] \\ &\quad + \sum_{b=0}^{B(m)_{max}} [D_{drop}(b) \times P_{drop}(b)] \quad (17) \end{aligned}$$

After some simple steps, we get the following expression,

$$\overline{D_{notify}} = (1 - P_{drop})\overline{D_{succ}} + P_{drop}\overline{D_{drop}} \quad (18)$$

The above result is quite intuitive. The first term expresses the fact that the packet is successfully transmitted with a probability  $(1 - P_{drop})$  and the corresponding average delay is  $\overline{D_{succ}}$ , while the second term expresses the fact that the packet is dropped with a probability  $P_{drop}$  and the corresponding average delay is  $\overline{D_{drop}}$ .

We can also compute the standard deviation of  $D_{notify}$ .

4) *Derivation of  $D_{intersucc}$* : In contrast to  $D_{succ}$ ,  $D_{drop}$ , and  $D_{notify}$  discussed above, we are not able to get the probability mass function of the delay  $D_{intersucc}$ . However, we can get the average value of  $D_{intersucc}$ . Specifically, we can view the packet transmissions at a given station as a *bernoulli experiment* with the probability of success  $(1 - P_{drop})$ . Therefore, to observe a successful transmission at the given station, the number of trials required follows a geometric distribution. Clearly, the average number of trials required is simply  $\frac{1}{1 - P_{drop}}$ . Among these trials, a fraction of  $P_{drop}$  trials fail and the packets are dropped, and each of the failing trial lasts on an average  $\overline{D_{drop}}$  duration. Obviously, among the trials the last trial must be successful, and it lasts on an average  $\overline{D_{succ}}$  duration. Therefore, the average delay  $\overline{D_{intersucc}}$  can be obtained as,

$$\begin{aligned} \overline{D_{intersucc}} &= \frac{P_{drop}}{1 - P_{drop}} \overline{D_{drop}} + \frac{1 - P_{drop}}{1 - P_{drop}} \overline{D_{succ}} \\ &= \frac{\overline{D_{notify}}}{1 - P_{drop}} \quad (19) \end{aligned}$$

The last expression is quite intuitive. To observe a successful transmission at a given station, on an average, the station gets  $1/(1 - P_{drop})$  notifications and the average duration between two consecutive notifications is  $\overline{D_{notify}}$ .

5) *Derivation of  $D_{infinite}$* : Now we derive the delay under the infinite retrials assumption that is adopted in [3], [11]. Let  $P_{infinite}(j, b)$  be the probability that a packet is successfully transmitted at stage  $j$  ( $0 \leq j < \infty$ ), and the sum of the back-off slots generated up to stage  $j$  is equal to  $b$ . Therefore,

$$P_{infinite}(j, b) = P_{infinite}(j) \times Pr(B(j) = b) \quad j \in [0, \infty) \quad (20)$$

where  $P_{infinite}(j)$  can be defined as in Equation (6) except that the range of  $j$  is changed to  $[0, \infty)$ , and  $B(j)$  is defined in Equation (5). Clearly, the corresponding delay is,

$$D_{infinite}(j, b) = b \times T_{avg} + j \times T_{col} + T_{succ} \quad j \in [0, \infty) \quad (21)$$

The above two formulas form the probability mass function of the delay  $D_{infinite}$ . Therefore, the average of  $D_{infinite}$  is,

$$\begin{aligned} \overline{D_{infinite}} &= \sum_{j=0}^{\infty} \sum_{b=0}^{B(j)_{max}} [D_{infinite}(j, b) \times P_{infinite}(j, b)] \\ &= \sum_{j=0}^m \sum_{b=0}^{B(j)_{max}} [D_{infinite}(j, b) \times P_{infinite}(j, b)] \\ &+ \sum_{j=m+1}^{\infty} \sum_{b=0}^{B(j)_{max}} [D_{infinite}(j, b) \times P_{infinite}(j, b)] \quad (22) \end{aligned}$$

After several simple steps, we find that the first term is simply equal to  $(1 - P_{drop})\overline{D_{succ}}$ . This is very intuitive as explained now. Under the infinite retrials assumption, the probability that a packet is successfully transmitted at a stage later than stage  $m$  is equal to the probability that the packet is dropped if the retry limit is  $m$ . Therefore, the probability that a packet is transmitted successfully at one of the initial  $m$  stages is  $(1 - P_{drop})$ , and the average delay of such a packet is  $\overline{D_{succ}}$ , explaining the first term. After some simple steps, the second term of Equation (22) (let us represented it by  $T_2$ ), is,

$$\begin{aligned} T_2 &= P_{drop} \left[ T_{succ} + \left( m + \frac{1}{1-p} \right) T_{col} + \right. \\ &\quad \left. \left( \frac{B(m)_{max}}{2} + \frac{CW_{max}}{2} \frac{1}{1-p} \right) T_{avg} \right] \quad (23) \end{aligned}$$

The above term is also quite intuitive. If a packet experiences more than  $m$  stages (with a probability  $P_{drop}$ ), it will first experience  $m$  collisions. Then, the average number of additional collisions experienced is simply  $1/(1-p)$  due to the *memoryless* property of the geometric distribution. The above arguments explain the term  $(m + \frac{1}{1-p})T_{col}$  in Equation (23). Now we explain the term  $(\frac{B(m)_{max}}{2} + \frac{CW_{max}}{2} \frac{1}{1-p})T_{avg}$ . Specifically,  $B(m)_{max}/2$  represents the average number of back-off slots in the first  $m$  stages. Then, at every additional stage, on an average  $CW_{max}/2$  slots are involved, and the average number of additional stages is simply  $1/(1-p)$ .

From equations (22) and (23), we obtain,

$$\overline{D_{infinite}} = (1 - P_{drop})\overline{D_{succ}} + T_2 \quad (24)$$

We can also express the average delay as follows,

$$\begin{aligned} \overline{D_{infinite}} &= (1 - P_{drop})\overline{D_{succ}} + P_{drop}\overline{D_{drop}} + T_3 \\ &= \overline{D_{notify}} + T_3 \quad (25) \end{aligned}$$

where  $T_3 = P_{drop} [T_{succ} + \frac{p}{1-p} T_{col} + \frac{CW_{max}}{2} \frac{1}{1-p} T_{avg}]$ .

We notice that  $\overline{D_{infinite}}$  is always larger than  $\overline{D_{notify}}$  as well as  $\overline{D_{succ}}$  (since  $\overline{D_{notify}}$  is always larger than  $\overline{D_{succ}}$ ).

Therefore, in general  $\overline{D_{infinite}}$  overestimates the delay. In fact, as shown later in Section III, the results of  $\overline{D_{infinite}}$  is very misleading whenever  $p$  (or  $P_{drop}$ ) is large.

From equations (20) and (21), we can also numerically obtain the standard deviation of  $D_{infinite}$ .

#### D. Fairness Analysis

IEEE 802.11 is fair in the *long-term* (e.g., equal average delay or throughput) in a *single-hop* network<sup>2</sup>. However, due to the randomness involved in the binary exponential back-off (BEB) algorithm, substantial *short-term* unfairness remains even in a single-hop scenario as clearly shown in [7].

In general, unfairness can be measured in terms of the throughput or the packet delay. To measure the short-term unfairness through the packet delay, an ideal way is to first get the probability distribution of the delay experienced at all the stations, and then see how variable the delay is. Clearly, higher variability implies a more unfair system. However, to get such a distribution, we need to model the system state in a way that there is at least one separate parameter used to represent the state of each station, making the state space very large if the number of stations is large. On the other hand, if we assume that the probability distribution of the delay experienced at a given station is similar to the distribution of the delay experienced at all the stations, the problem becomes much easier as we only need to consider the states at a typical station. We adopt this approach as it is much simpler but still able to reveal the general trend of the unfairness, though it may not give a very precise measurement of the unfairness.

In the previous subsection we have already obtained the probability distribution of the delay (e.g.,  $D_{succ}$ ) experienced at a typical station. Therefore, we now only need to develop a metric to quantify the variability of the delay and thus reflect the unfairness. The standard deviation is a good candidate, however, it is not independent of scale (i.e., the unit of measurement) [5] as shown by the following example. Considering there are two different schemes to distribute a common resource to three users, the delays experienced by the users under the two schemes are as follows:

Scheme I = {1, 2, 3}

Scheme II = {2, 4, 6}

Clearly, the two schemes deliver the same fairness as far as the delay is concerned. However, the standard deviation of the delay under the two schemes are  $\sqrt{2}$  and  $2\sqrt{2}$ , respectively. Therefore, if we use the standard deviation as a metric to reflect unfairness, we will arrive at a misleading conclusion: the second scheme is two times more unfair than the first one. Clearly, the standard deviation is not a good metric to reflect the fairness as it is not scale independent. Now we look at the coefficient of variation (COV) of the delay, which also reflects the variability of the delay. The COV is defined as,

$$COV = \frac{\text{Standard Deviation}}{\text{Average}} \quad (26)$$

<sup>2</sup>Note that in a *multi-hop* scenario, IEEE 802.11 cannot even maintain the long-term fairness among the contending stations [9].

Clearly, the COV under the above two schemes are the same, i.e.,  $\frac{\sqrt{2}}{2}$ , implying that they deliver the same fairness. In fact, as mentioned in [5], COV in many senses is a good metric to reflect fairness. In general, the larger the COV is, the broader the delay spreads, and thus the more unfair the scheme is. Numerically, we can easily find the value of the COV of the delay from the results in the previous subsection.

Now we discuss how to relate the COV to the well known Jain's index [5]. If there are  $n$  number of users sharing the common resource, and user  $i$  gets a proportion of  $x_i$ , then,

$$\text{Jain's Index} = \frac{[\sum_i x_i]^2}{n \sum_i x_i^2} \quad (27)$$

On the other hand, the square of the COV is as follows,

$$COV^2 = \frac{\text{Variance}}{\text{Average}^2} = \frac{\overline{x^2} - (\overline{x})^2}{(\overline{x})^2} \quad (28)$$

Now we can get the relationship between the COV and the Jain' index as follows,

$$\begin{aligned} \text{Jain's Index} &= \frac{[\sum_i x_i]^2}{n \sum_i x_i^2} = \frac{[\frac{1}{n} \sum_i x_i]^2}{\frac{1}{n} \sum_i x_i^2} = \frac{(\overline{x})^2}{\overline{x^2}} \\ &= \frac{(\overline{x})^2}{(\overline{x})^2 + \overline{x^2} - (\overline{x})^2} = \frac{1}{1 + \frac{\overline{x^2} - (\overline{x})^2}{(\overline{x})^2}} = \frac{1}{1 + COV^2} \end{aligned} \quad (29)$$

Clearly, while the COV is boundless, the Jain's index is in the range of  $[0, 1]$ .

### III. PERFORMANCE EVALUATION OF IEEE 802.11

In this section, we present the performance results for the four-way handshake in IEEE 802.11 DCF under the Direct Sequence Spread Spectrum (DSSS) physical layer [4]. Table I lists the values of the parameters used in the calculation. Rather than using the same transmission rate (i.e., 1 Mbps) for both the Data and control frames as in [1], to closely follow the standards, in our calculation the transmission rate for the Data frame is 2 Mbps while that for the control frames is 1 Mbps. Using the parameters listed in the above table, we can calculate  $E(P)$ ,  $T_{succ}$ , and  $T_{col}$ , which are 4096  $\mu s$ , 5440  $\mu s$ , and 716  $\mu s$ , respectively.

As shown in the previous section, the performance depends on four parameters:  $n$ ,  $CW_{min}$ ,  $CW_{max}$ , and  $m$ . In the following graphs, we will present the performance by varying the network size  $n$  and the three system parameters (i.e.,  $CW_{min}$ ,  $CW_{max}$ , and  $m$ ). Note that in one graph only one of the three system parameters varies while the remaining two parameters are fixed at their standard values. The standard values [4] are:  $CW_{min} = 31$ ,  $CW_{max} = 1023$ , and  $m = 6$ .

#### A. Throughput Results

Figures 3-5 present the throughput results. From the figures, we can make three main observations: (i) Compared to the throughput presented in [1], the throughput obtained here is much smaller; (ii) In general, the larger the  $CW_{min}$ ,  $CW_{max}$ , and  $m$  are, the larger the throughput is. On the other hand, when the network size (i.e.,  $n$ ) increases, the throughput normally decreases; (iii) The throughput is quite sensitive to

TABLE I  
IEEE 802.11 PARAMETERS USED IN THE CALCULATION

Payload of data packet	1024 bytes
Data	1024 bytes + MAC header + PHY header
RTS	20 bytes + PHY header
CTS	14 bytes + PHY header
ACK	14 bytes + PHY header
PHY Header	192 us
MAC Header	28 bytes
Basic rate	1 Mbps
Data rate	2 Mbps
Slot time	20 us
SIFS	10 us
DIFS	50 us
EIFS	364 us

the network size and the system parameters, which is contrary to the observation made in [1].

There are two reasons behind the first observation. Firstly, since the model in [1] has not considered the retry limit (i.e., it assumes infinite retrials), it overestimates the throughput as mentioned in [13]. In fact, this can also be easily verified from Figure 5, which shows that the throughput increases with the increase of the retry limit  $m$ . The second reason is as follows. As mentioned in the discussion of Equation (4),  $T_{col}$  in our calculation is larger than the one in [1]. On the other hand,  $T_{succ}$  is smaller than that in [1] as we use a higher transmission rate for the Data frame. Therefore, in our calculation, the time spent on collisions increases while that spent on the payload transmission decreases. The above arguments explain why the throughput obtained here is smaller.

To explain the second observation (i.e., how the throughput changes with the system parameters), we first need to look at the values of  $T_{col}/T_{succ}$  and  $\sigma/T_{succ}$  as discussed in the context of Equation (3). In our calculation,  $\sigma/T_{succ}$  is very small (i.e., 20/5440) compared to the unity, and thus the throughput  $S$  is insensitive to the value of  $P_{idle}/P_{succ}$ . On the contrary,  $T_{col}/T_{succ}$  is relatively large (i.e., 716/5440), and thus  $S$  is sensitive to the value of  $P_{col}/P_{succ}$ . Therefore, to explain the second observation, we only need to look at the value of  $P_{col}/P_{succ}$  under different system parameters. Clearly, the larger  $P_{col}/P_{succ}$  is, the smaller the throughput is. To exemplify this, Figure 6 presents the value of  $P_{col}/P_{succ}$  when the  $CW_{min}$  is varied. We notice that the ratio increases as the  $CW_{min}$  decreases or as  $n$  increases, explaining why the throughput in Figure 3 decreases in these two cases.

Now let us explain the third observation. As mentioned,  $T_{col}/T_{succ}$  is quite large in our case, and thus the throughput  $S$  is very sensitive to the system parameters. In contrast,  $T_{col}/T_{succ}$  is very small (i.e., 8/191.36) in [1], and thus  $S$  presented there is insensitive to the system parameters.

Here, we would like to point out that in the emerging IEEE 802.11 standards (e.g., IEEE 802.11-b, -a and -g), the  $T_{col}/T_{succ}$  is becoming larger because the transmission rate of the Data frames (which determines  $T_{succ}$ ) is increasing (e.g., can be as high as 54 Mbps) while the transmission rate of the control frames (which determines  $T_{col}$ ) remains small.

Therefore, we expect the throughput in the emerging standards to become even more sensitive to the system parameters.

### B. Delay Results

In figures 7-11, we first present the five kinds of delay by varying  $CW_{min}$ . From the results, we find that: (i) In general, in the first three kinds of delay (i.e.,  $\overline{D}_{succ}$ ,  $\overline{D}_{drop}$ , and  $\overline{D}_{notify}$ ), the larger the  $CW_{min}$  is, the larger is the delay; (ii) In contrast,  $\overline{D}_{intersucc}$  and  $\overline{D}_{infinite}$  decrease when  $CW_{min}$  increases; (iii)  $\overline{D}_{drop}$  is much higher than  $\overline{D}_{succ}$  (note that the vertical axes have different scales), while  $\overline{D}_{notify}$  stands in between the above two; (iv)  $\overline{D}_{infinite}$  is much larger than  $\overline{D}_{succ}$  and  $\overline{D}_{notify}$ , showing the error introduced by the assumption of the infinite retrials.

The first observation is quite intuitive, and can be explained by the fact that a larger  $CW_{min}$  should result in a larger average number of back-off slots. Also, the trend of  $\overline{D}_{intersucc}$  with respect to  $CW_{min}$  coincides with the trend of throughput with respect to  $CW_{min}$ . The trend of  $\overline{D}_{infinite}$  with respect to  $CW_{min}$  is similar to that obtained in [3], and gives the same misleading conclusion: a larger  $CW_{min}$  results in a smaller delay. Now we explain why the trend corresponding to  $\overline{D}_{infinite}$  is contrary to that of  $\overline{D}_{notify}$  (as well as of  $\overline{D}_{succ}$  and  $\overline{D}_{drop}$ ). As known from Equation (25), the error in  $\overline{D}_{infinite}$  compared to  $\overline{D}_{notify}$  becomes larger when the conditional collision probability  $p$  (or  $P_{drop}$ ) becomes larger. Clearly, when  $CW_{min}$  is smaller,  $p$  is larger, resulting in a larger error in the calculation of  $\overline{D}_{infinite}$ . On the contrary, the error is much smaller when  $CW_{min}$  is larger as  $p$  becomes smaller. Therefore,  $\overline{D}_{infinite}$  becomes larger under smaller  $CW_{min}$ . In conclusion, whenever the conditional collision probability  $p$  is large, the assumption of infinite retrials will result in a large error in the calculation of the delay.

Figures 12-14 present the standard deviation of the first three kinds of delay (i.e.,  $D_{succ}$ ,  $D_{drop}$ , and  $D_{notify}$ ) under variable  $CW_{min}$ . In general, they follow the similar trend as the average delay.  $D_{notify}$  has a larger standard deviation than the other two since it considers the delay of all the packets (i.e., dropped as well as successfully transmitted), and since there is a large difference between the average delay of the packets being dropped and those being transmitted successfully.

In the above, we have presented the delay results when  $CW_{min}$  is variable. Similarly, we have also obtained the results when  $CW_{max}$  or  $m$  varies (see [8] for details).

### C. Fairness Results

Figures 15-17 present the Coefficient of Variation (COV) of  $D_{succ}$  for different values of  $CW_{min}$ ,  $CW_{max}$ , and  $m$ , respectively. To recall, a larger COV implies a more unfair system. From the figures, we can clearly make two observations: (i) In general, the smaller the  $CW_{min}$  is and the larger the  $CW_{max}$  and  $m$  are, the more unfair the system is; (ii) As the network size (i.e.,  $n$ ) increases, first the system becomes more unfair, and then after a certain value of  $n$  it becomes more fair.

We now explain the first observation. In IEEE 802.11, whenever a station transmits a packet successfully, it resets its

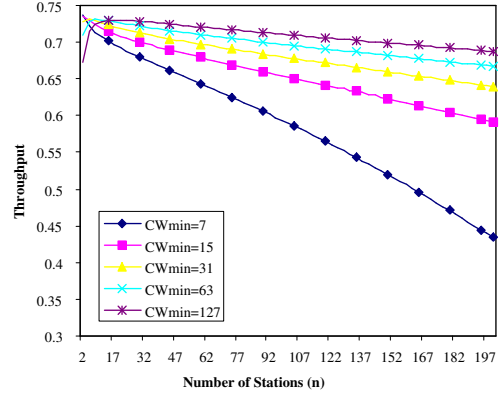


Fig. 3. Throughput under variable  $CW_{min}$

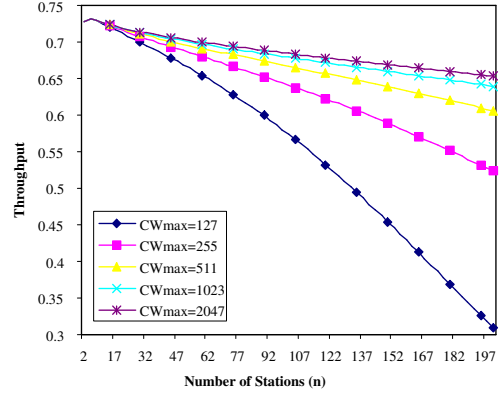


Fig. 4. Throughput under variable  $CW_{max}$

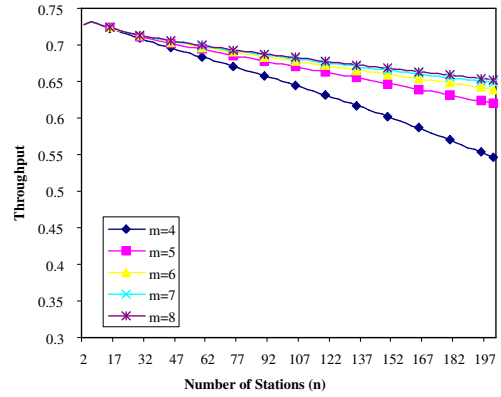


Fig. 5. Throughput under variable retry limit  $m$

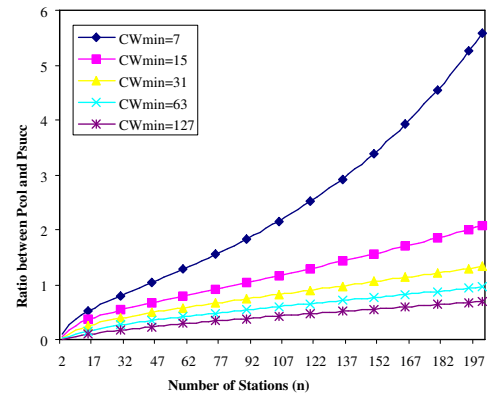


Fig. 6. Ratio  $P_{col}/P_{succ}$  under variable  $CW_{min}$

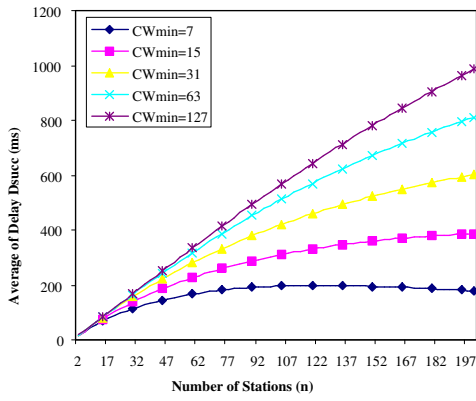


Fig. 7. Average delay  $D_{succ}$  under variable  $CW_{min}$

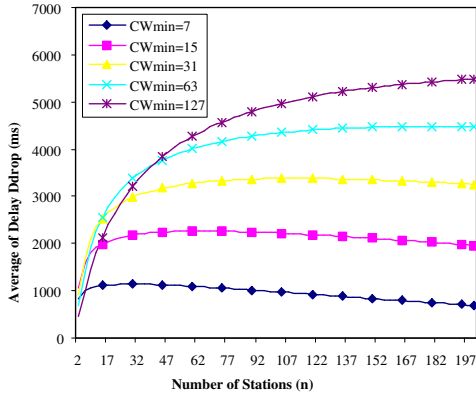


Fig. 8. Average delay  $D_{drop}$  under variable  $CW_{min}$

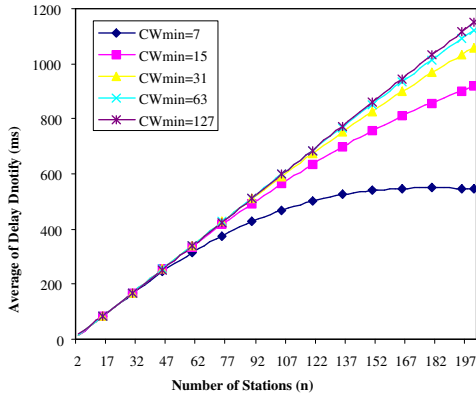


Fig. 9. Average delay  $D_{notify}$  under variable  $CW_{min}$

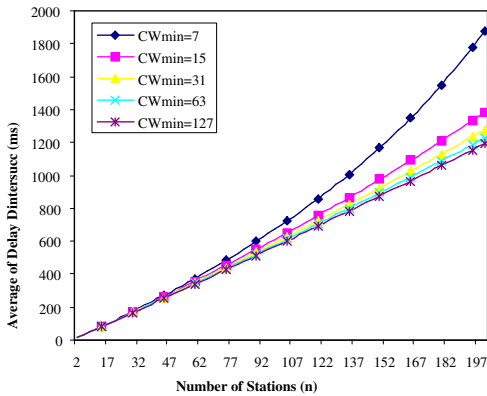


Fig. 10. Average delay  $D_{intersucc}$  under variable  $CW_{min}$

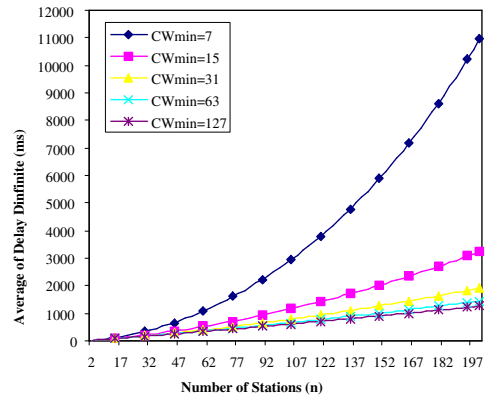


Fig. 11. Average delay  $D_{infinite}$  under variable  $CW_{min}$

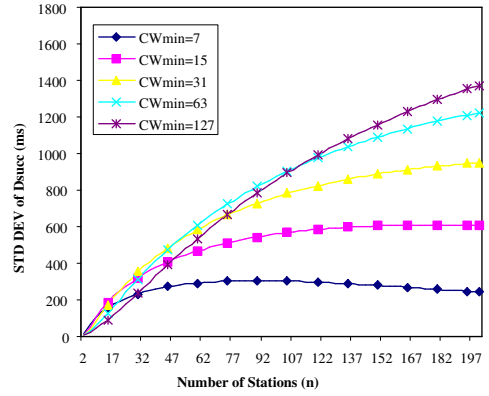


Fig. 12. Standard deviation of  $D_{succ}$  under variable  $CW_{min}$

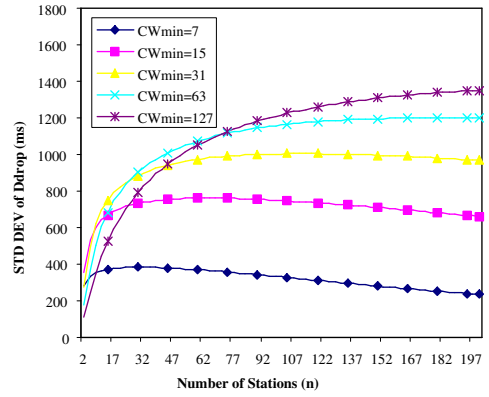


Fig. 13. Standard deviation of  $D_{drop}$  under variable  $CW_{min}$

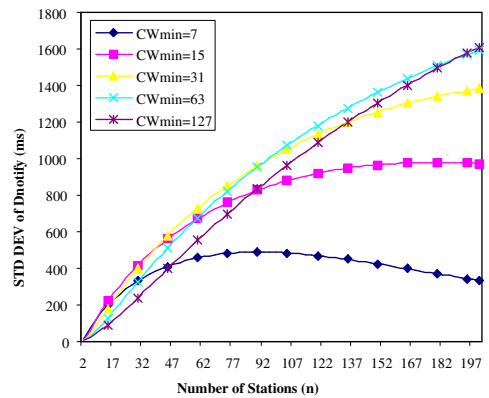


Fig. 14. Standard deviation of  $D_{notify}$  under variable  $CW_{min}$

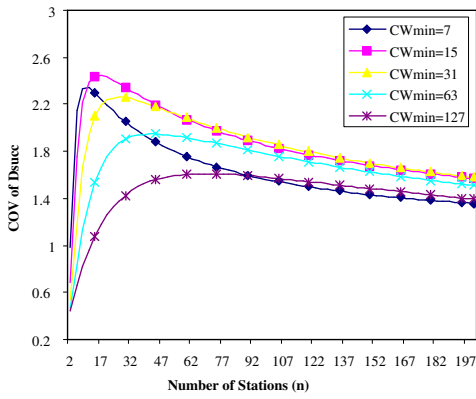


Fig. 15. Fairness under variable  $CW_{min}$

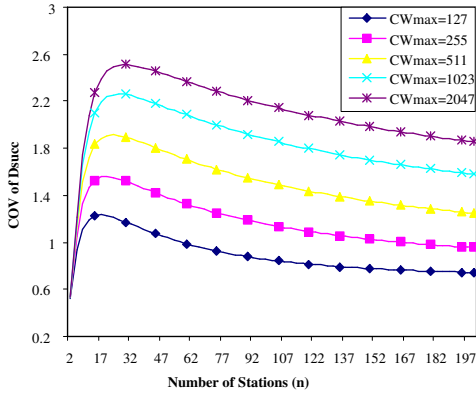


Fig. 16. Fairness under variable  $CW_{max}$

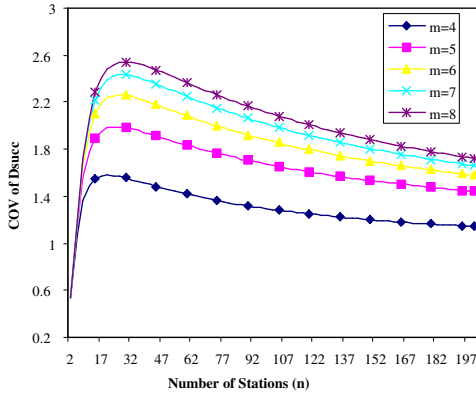


Fig. 17. Fairness under variable retry limit  $m$

contention window to  $CW_{min}$ . Clearly, if  $CW_{min}$  is small, the station that has just transmitted a packet successfully is more likely to get control of the medium *again* compared to the case when  $CW_{min}$  is large, explaining why the system becomes more unfair when  $CW_{min}$  is smaller. Now we consider the fairness with respect to  $CW_{max}$ . Clearly, the contention window at a station will become  $CW_{max}$  only when the station has experienced collisions successively. Therefore, if  $CW_{max}$  is large, the colliding stations, though they have already suffered a lot, will be more unfairly treated, explaining why the system is more unfair when the  $CW_{max}$  is larger. As for the retry limit  $m$ , if it is small, a station that has experienced collision(s) will drop the packet early and reset its CW to  $CW_{min}$ , and then schedule the transmission of a

new packet. On the contrary, if  $m$  is large, the station may have to schedule the retransmission of a packet more number of times, which leads to a larger contention window, and in turn, short-term unfairness. From observation (i), we may reach an interesting conclusion regarding the back-off policy: to achieve better fairness, we should always use a medium-size CW without any retrial for the unsuccessful transmissions.

For the second observation, we first explain the reason why the system becomes more unfair when we begin to increase  $n$  (e.g., from 2 to 17). This is due to the randomness involved in the binary exponential back-off (BEB) algorithm. In particular, some stations may experience collisions consecutively while other stations can transmit packets without experiencing any collision. With the increase of  $n$ , collisions are more likely to happen, explaining the increasing amount of unfairness. In fact, we have also observed this phenomena in the simulation results presented in [7]. However, when the collision probability increases beyond a certain value along with the increase of  $n$ , most of the stations will experience collisions before they can transmit any packet successfully, and therefore, it tends to become more fair for all the stations. In the extreme case, when the number of stations becomes infinite, the collision probability tends to unity, and no station can transmit any packet successfully. Therefore, every station has zero throughput (or infinite delay), implying absolute fairness among the contending stations!

#### D. Summary of the Performance Evaluation

In this subsection, we give a brief summary of the performance of IEEE 802.11 in single-hop networks. With respect to the throughput, we find that it is quite sensitive to the chosen system parameters. This would be particularly true in the emerging IEEE 802.11 standards. Therefore, dynamic tuning of the system parameters is of great interest to achieve optimal performance. We also find that the throughput calculated in the model of [1] is higher than that in our model (which is more precise). As for the delay, we show that the first four kinds of delay under the finite retrials condition have very distinct values, showing the importance of distinguishing among them. We also find that the  $D_{infinite}$  (i.e., delay under the infinite retrials assumption) has a large error, and thus leads to misleading observations. For the fairness performance, we get an interesting conclusion about the back-off policy: to achieve better fairness, a medium-sized contention window should be used without any retrial of the unsuccessful transmissions.

Table II summarizes how the system parameters should be selected (in terms of “large” or “small”) to achieve the desired performance (i.e., higher throughput, smaller delay, and better fairness). By “smaller delay”, we mean a smaller value of the average as well as the standard deviation of  $D_{succ}$ ,  $D_{drop}$ , and  $D_{notify}$ . We do not need to consider  $D_{intersucc}$  and  $D_{infinite}$  as  $D_{intersucc}$  is directly reflected by the throughput while  $D_{infinite}$  is misleading.

From the table, we find that a set of system parameters delivering a good performance for one metric, may deliver a bad performance in terms of another metric. For example, if

we aim to achieve a higher throughput, the table shows that we should use a large value of  $CW_{max}$  and  $m$ . However, the table also tells us that we should use a small value of  $CW_{min}$  and  $m$  if we aim to achieve better fairness. In other words, a trade-off among the metrics must be made in the dynamic tuning of the system parameters.

TABLE II  
INTERDEPENDENCE BETWEEN PERFORMANCE AND PARAMETERS

Parameter \ Metric	Higher Throughput	Smaller Delay	Better Fairness
$CW_{min}$	Large	Small	Large
$CW_{max}$	Large	Small	Small
$m$	Large	Small	Small

#### IV. RELATED AND FUTURE WORK

##### A. Related work

In the literature, many analytical models have focused on the calculation of the throughput (or capacity) of IEEE 802.11. For example, paper [2] derives the theoretical throughput of IEEE 802.11 by relating IEEE 802.11 to a p-persistent CSMA protocol. Paper [10] derives the throughput of IEEE 802.11 by using average analysis technique. However, the above models have oversimplified the calculation of the transmission probability  $\tau$ , which is determined by the BEB algorithm. In contrast, papers [1] and [13] derive a more precise value of  $\tau$  by modelling the stochastic process representing the back-off time counter as a discrete-time Markov chain.

While most of the published models focus on the throughput of IEEE 802.11, two recent work [3], [11] concentrate on the delay analysis. However, neither of them have considered the retry limit, resulting in a large error as shown in this paper. Also, they do not distinguish the five kinds of delay defined in this paper, which have very different values.

Though many papers (e.g., [9], [12]) have focused on achieving fairness in IEEE 802.11, there are very few analytical models to quantify the unfairness in IEEE 802.11. We are only aware of one model [6] that analytically shows the short-term unfairness in IEEE 802.11 due to the hidden-terminal problem. However, the above model is very specific to the topology being given and thus it is not clear whether the model can be extended into a general single-hop network with a large number of stations.

##### B. Future Work

Motivated from the results obtained in this work, one main future work is to develop a dynamic tuning mechanism, which can achieve a reasonable tradeoff among the performance metrics. One possible method is as follows. We should first define a function, which characterizes the weights of the performance metrics according to the user preference. Using such a function, under any given network size (i.e.,  $n$ ), we can calculate the corresponding optimal system parameters (i.e.,  $CW_{min}$ ,  $CW_{max}$ , and  $m$ ) that maximize the preferred function. This can be easily achieved using a table, which

stores the optimum values of the systems parameters for different network sizes and for different weights that could be used in defining the function. The table can be prepared *off-line*. The estimation of the network size can be achieved by using the methods in [2], [7].

Another important future work is to extend the analysis by relaxing the assumptions made at the beginning of Section II. While the analysis should be easily extended when the assumptions (2)-(4) are relaxed as discussed in [11], [1], it may not be trivial to extend the model for a multi-hop network.

#### V. CONCLUSIONS

In this paper, we have proposed a simple analytical model to evaluate the throughput, delay, and fairness performance in single-hop IEEE 802.11 networks. Using the proposed model, we have carried out an extensive performance evaluation of IEEE 802.11 under three different system parameters (i.e.,  $CW_{min}$ ,  $CW_{max}$ , and retry limit  $m$ ). Our results show that the throughput, delay, and fairness are quite sensitive to these system parameters, and thus dynamic tuning of the parameters is of great interest to achieve the optimal performance. Moreover, the optimal values of the parameters may not be consistent with each other for different performance metrics, implying that some tradeoff among the metrics must be made in the dynamic tuning. Finally, our results also show that the published models have overestimated the throughput as well as the packet delay.

#### REFERENCES

- [1] G. Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function," IEEE JSAC, March 2000, pp.535-547.
- [2] F. Cali, M. Conti, E. Gregori, "Dynamic Tuning of the IEEE 802.11 Protocol to Achieve a Theoretical Throughput Limit," IEEE/ACM Transactions on Networking, December 2000, pp.785-799.
- [3] M. M. Carvalho, J. J. Garcia-Luna-Aceves, "Delay Analysis of IEEE 802.11 in Single-hop Networks," in IEEE ICNP, 2003.
- [4] IEEE, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications," IEEE 802.11 standards, June 1999.
- [5] R. Jain, D. Chiu, W. Hawe, "A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer System," DEC Technical Report, 1984.
- [6] Z.F. Li, S. Nandi, A.K. Gupta, "Modeling the Short-term Unfairness of IEEE 802.11 in Presence of Hidden Terminals," in IFIP Networking, 2004.
- [7] Z.F. Li, S. Nandi, A.K. Gupta, "Achieving MAC Fairness in Wireless Ad-hoc Networks using Adaptive Transmission Control," in IEEE ISCC, 2004.
- [8] Full version of this paper, available at <http://www.cs.jhu.edu/~zfli>.
- [9] T. Nandagopal, T. Kim, X. Gao, V. Bharghavan, "Achieving MAC Layer Fairness in Wireless Packet Networks," in ACM MOBICOM, 2000.
- [10] Y. C. Tay, K. C. Chua, "A capacity analysis for the IEEE 802.11 MAC protocol," ACM Wireless Networks, 2, 2001, pp.159-171.
- [11] O. Tickoo, B. Sikdar, "Queueing Analysis and Delay Mitigation in IEEE 802.11 Random Access MAC based Wireless Networks," in IEEE INFOCOM, 2004.
- [12] N.H. Vaidya, P. Bahl, S. Gupta, "Distributed fair scheduling in a wireless LAN," in ACM MOBICOM, 2000.
- [13] H.T. Wu, Y. Peng, K.P. Long, S.D. Cheng, J. Ma, "Performance of Reliable Transport Protocol over IEEE 802.11 Wireless LAN: Analysis and Enhancement," in IEEE INFOCOM, 2002.